High dimensional theory of two-phase optimizers

Atish Agarwala
Google DeepMind

Abstract

The trend towards larger training setups has brought a renewed interest in partially asynchronous two-phase optimizers which optimize locally and then synchronize across workers. Additionally, recent work suggests that the one-worker version of one of these algorithms, DiLoCo, shows promising results as a (synchronous) optimizer. Motivated by these studies we present an analysis of LA-DiLoCo, a simple member of the DiLoCo family, on a high-dimensional linear regression problem. We show that the one-worker variant, LA, provides a different tradeoff between signal and noise than SGD, which is beneficial in many scenarios. We also show that the multi-worker version generates more noise than the single worker version, but that this additional noise generation can be ameliorated by appropriate choice of hyperparameters. We conclude with an analysis of SLA–LA with momentum – and show that stacking two momentum operators gives an opportunity for acceleration via a non-linear transformation of the "effective" Hessian spectrum, which is maximized for Nesterov momentum. Altogether our results show that two-phase optimizers represent a fruitful new paradigm for understanding and improving training algorithms.

1 Introduction

The success of deep learning has brought a pressing need for training algorithms that operate at scale, due to the desire to train larger models with larger and more complex datasets for longer time. Most progress in this area has been made by synchronous systems, which compute gradients across multiple machines but sync them before changing parameters. However, the cost of operating synchronous systems can grow superlinearly with scale due to issues like memory, communications bandwidth, and hardware failures.

This has lead to numerous attempts to develop asynchronous training systems. While fully asynchronous training remains challenging, an intermediate approach is to design distributed optimizers where groups of workers maintain in-group synchrony, but between group synchronization takes a longer time [1, 2]. This allows for more flexible hardware configurations, and the possibility for more compute to be brought to bear on a single training run.

Recent work suggests that a two-phase optimization framework is a promising approach [2–6]. Workers first perform an *inner optimization* using algorithms (e.g. SGD), and generate a *pseudo-gradient* — the difference between the parameters at the start and end of the inner optimization. These pseudo-gradients are then averaged and sent to an *outer optimizer* (e.g. SGD momentum) and used to perform a single optimization step — the results of which are sent to the workers as the starting point for the new inner optimization.

One particular configuration of this two-phase framework, DiLoCo, exhibits promising results in distributed LLM training [7], a fact which has since been repeatedly observed empirically [8–10], but the theoretical underpinnings of this performance remain mysterious. Even more surprising is the fact that DiLoCo shows benefits in the one-worker (synchronous) setting [10]. There is even less theoretical understanding of this phenomenon, with some conjectures that the benefits are due to implicit regularization of the algorithms [11].

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

In this work we provide theoretical evidence that two-phase algorithms like DiLoCo provide *direct benefits to optimization itself*. We use tools from high-dimensional learning dynamics theory to exactly analyze the most basic algorithm in this class, Lookahead (LA) [12], as well as the multi-worker version LA-DiLoCo (with outer optimizer SGD), and write an evolution equation for the expected value of the loss, which concentrates in high dimensions. We show the following:

- Two-phase optimizers induces a noise structure that is quantitatively different from SGD.
- LA allows a different tradeoff between signal and noise than SGD, which is provably better than SGD at optimal learning rates.
- LA-DiLoCo generates more noise in the distributed setting than the synchronous setting, but there are hyperparameter choices which minimize the discrepancy between the methods for small number of workers.

We then provide analysis of the extension Super Lookahead (SLA) — LA with momentum in the inner and outer optimizer. In the deterministic setting, we prove some key features of stacking momentum operators:

- The outer optimizer operates on an "effective spectrum" generated by the inner optimizer, non-linearly transforming the relationship between Hessian spectrum and convergence rate.
- Nesterov-style momentum has large benefits over heavy ball momentum in this setting.

Our analysis suggests that there is a rich family of behaviors induced by LA and SLA which can't be captured by more traditional algorithms and is worth further experimental and theoretical study.

2 Basics of high-dimensional linear regression

We introduce the high-dimensional linear regression setup studied previously [13, 14]. We will present known results for the dynamics of SGD in this setting, so we can build on them to analyze the LA optimizer. Consider a scenario with D targets $\mathbf{y}_{tr}$ that we are trying to fit with a linear model with a P dimensional parameter vector $\boldsymbol{\theta}$ by minimizing MSE loss:

$$\mathbf{f}(\boldsymbol{\theta}) = \mathbf{y}_0 + \mathbf{J}\boldsymbol{\theta},\ \mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2D}||\mathbf{z}||^2,\ \mathbf{z} \equiv \mathbf{f}(\boldsymbol{\theta}) - \mathbf{y}_{tr}. \tag{1}$$

Here $\mathbf{J}$ is the $D \times P$ dimensional Jacobian, and $\mathbf{z}$ are the residuals. Previous works studied mini-batch SGD, where B examples are sampled from the D total datapoints in order to compute the gradient. The update rule for SGD in this setting is given by

$$\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t = -\frac{\eta}{B}\mathbf{J}^\top \mathbf{P}_t \mathbf{z}_t. \tag{2}$$

where $\mathbf{P}_t$ is a $D \times D$ projection matrix, diagonal with B entries with value 1, and the rest 0, chosen at random. The $\mathbf{P}_t$ are i.i.d. for different t. One feature of this model is that the updates close in function space. Defining the empirical NTK $\hat{\Theta} \equiv \frac{1}{D}\mathbf{J}\mathbf{J}^\top$, the residuals evolve as

$$\mathbf{z}_{t+1} - \mathbf{z}_t = -\eta \frac{D}{B}\hat{\Theta}\mathbf{P}_t \mathbf{z}_t. \tag{3}$$

The dynamics of the first and second moments can also be computed exactly (see Equation 6 and 7 and Appendix A.2 of [14]).

For MSE loss, computing the diagonal of the covariance $\mathrm{E}_{\mathbf{P}}[\mathbf{z}_{t+1}\mathbf{z}_{t+1}^\top]$ would give us the expected loss trajectory. In the high dimensional limit ($B \gg 1$, $D \gg 1$, $B/D = O(1)$), Lee et al. [13] showed that under appropriate conditions on $\hat{\Theta}$, the dynamics of the second moment simplify considerably and the loss trajectories concentrate. We recap the concentration results in the notation of Agarwala and Pennington [14], to ease later exposition. Given the eigendecomposition $\hat{\Theta} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top$, where $\mathbf{V}$ is orthogonal and $\boldsymbol{\Lambda}$ is diagonal and PSD, we define the normalized diagonal $\mathbf{p}_t \equiv \boldsymbol{\Lambda}^+ \mathrm{diag}(\mathbf{V}^\top \mathrm{E}_{\mathbf{P}}[\mathbf{z}_t\mathbf{z}_t^\top]\mathbf{V})$. The loss is $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{D}\mathbf{1}^\top\boldsymbol{\Lambda}\mathbf{p}$. $\mathbf{p}$ evolves as

$$\mathbf{p}_{t+1} = (\mathbf{A} + \mathbf{B})\mathbf{p}_t,\ \mathbf{A} \equiv (\mathbf{I} - \eta\boldsymbol{\Lambda})^2,\ \mathbf{B} \equiv \left(B^{-1} - D^{-1}\right)\eta^2\boldsymbol{\Lambda}\mathbf{1}\mathbf{1}^\top\boldsymbol{\Lambda} \tag{4}$$

in the high-dimensional limit. The $\mathbf{A}$ term corresponds to the deterministic dynamics of full batch training, and the $\mathbf{B}$ term controls the noise, which is larger for small batch size B. This system of equations can be used to derive quantities like the stochastic edge of stability — the critical learning rate above which dynamics diverges due to noise accumulation.

3 Stochastic dynamics of LA-DiLoCo

We use the model from the previous section and extend the analysis to understand LA-DiLoCo. We first study LA, the one-worker case of LA-DiLoCo, and prove that in this setting the two-tier dynamics of LA can reduce the loss faster than SGD with optimal learning rate. We then study the multi-worker case, and analyze how the number of workers reshapes the dynamics of the noise.

We first analyze the lookahead (LA) algorithm, first introduced in Zhang et al. [12]. This is one of the most basic training algorithms compatible with the DiLoCo framework, but has only one worker. Given a sync time S, the update rule is given by:

$$\boldsymbol{\psi}_t := \boldsymbol{\theta}_t \text{ (when } t//S = 0) \tag{5}$$

$$\boldsymbol{\psi}_{t+1} - \boldsymbol{\psi}_t = -\eta \mathbf{g}_t(\boldsymbol{\psi}_t) \text{ for } S \text{ steps} \tag{6}$$

$$\boldsymbol{\theta}_{t+S} - \boldsymbol{\theta}_t = \nu(\boldsymbol{\psi}_{t+S} - \boldsymbol{\theta}_t) \tag{7}$$

where $\mathbf{g}_t$ is the minibatch gradient. In other words, there is an inner optimizer that applies SGD to a copy $\boldsymbol{\psi}$ of $\boldsymbol{\theta}_t$ for S steps with *inner learning rate* η, after which an outer optimizer applies SGD based on the difference $\boldsymbol{\psi}_{t+S-1} - \boldsymbol{\theta}_t$ with *outer learning rate* ν. For $\nu = 1$ this corresponds to regular SGD, and for $\nu = 0$ this corresponds to no change to the parameters at all. In the high dimensional linear model, the gradient is $\mathbf{J}^\top \mathbf{P}_t \mathbf{z}_t(\boldsymbol{\psi}_t)/B$ and the algorithm can again be written entirely in function space:

$$\tilde{\mathbf{z}}_t := \mathbf{z}_t \text{ (when } t//S = 0) \tag{8}$$

$$\tilde{\mathbf{z}}_{t+1} - \tilde{\mathbf{z}}_t = -\eta \frac{D}{B} \hat{\Theta} \mathbf{P}_t \tilde{\mathbf{z}}_t \text{ for } S \text{ steps} \tag{9}$$

$$\mathbf{z}_{t+S} - \mathbf{z}_t = \nu(\tilde{\mathbf{z}}_{t+S} - \mathbf{z}_t) \tag{10}$$

$$\mathbf{z}_t \equiv f(\boldsymbol{\theta}_t) - \mathbf{y}_{tr},\ \tilde{\mathbf{z}}_t \equiv f(\boldsymbol{\psi}_t) - \mathbf{y}_{tr} \tag{11}$$

Using these equations, we can compute the dynamics of the second moment eigenmodes in the high-dimensional limit of Lee et al. [13] (Appendix A.1):

$$\mathbf{p}_{t+S} = \left([(1-\nu)\mathbf{I} + \nu(\mathbf{I} - \eta\boldsymbol{\Lambda})^S]^2 + \ \nu^2 [\mathbf{T}_S - (\mathbf{I} - \eta\boldsymbol{\Lambda})^{2S}] \right) \mathbf{p}_t,\ \mathbf{T}_k \equiv (\mathbf{A} + \mathbf{B})^k \tag{12}$$

The first term corresponds to the the deterministic dynamics of LA. The second term can then be interpreted as the dynamical contribution of noise after S steps of SGD, and generally slows convergence rate relative to the deterministic setting.

We immediately see that LA allows a different path for trading off progress on the objective from the deterministic terms and the generation of noise. In SGD, increasing η speeds up the convergence encoded by the $\mathbf{A}$ matrix but also increases the noise encoded by the $\mathbf{B}$ matrix, and an optimal choice of η trades these off efficiently. In LA we have the additional option to tune the outer learning rate ν with similar consequences; increasing it improves deterministic progress but also increases noise. This gives us a two-dimensional space of tradeoffs between deterministic progress and noise generation. Increase in noise from a larger inner learning rate η can be offset by a smaller outer learning rate ν and vice versa.

For small learning rates these effects are redundant; if $||\mathbf{B}||_{\text{op}} \ll ||\mathbf{A}||_{\text{op}}$ and $\eta\lambda S \ll 1$, then configurations with a fixed product $\nu\eta$ will give near-identical dynamics. For larger learning rates, this is not the case. For example $\nu > 1$ decreases stability non-linearly (Appendix A.2). However, as we will show in the next section, nontrivial ν can bring net benefits to optimization.

3.1 Lookahead can optimize faster than SGD

An immediate question is: are there scenarios where nontrivial LA ($\nu \neq 1$) optimizes faster than SGD? Perhaps surprisingly, the answer is yes. We can prove the following theorem about the inifinite-time convergence rate of LA induced by the spectral radius of the per-cycle transition matrix:

Theorem 3.1. *Consider the asymptotic infinite-time convergence rate of the optimization dynamics, governed by the spectral radius ρ of the transition matrix $\hat{\mathbf{T}}$ which controls the single cycle dynamics via Equation 16. Let $S \geq 2$ be even, and let $\boldsymbol{\Lambda}$ be non-degenerate. If the operator $\mathbf{T}_1$ controlling the one-step dynamics of SGD has non-degenerate largest eigenvalue at the optimal learning rate η^* for SGD ($\nu = 1$), then there exist settings of LA with $\nu > 1$ which have faster infinite-time convergence rate than SGD with the optimal learning rate for that spectrum and batch size.*

The proof can be found in Appendix A.3. The key is that using a larger outer learning rate combined with a smaller inner learning rate while preserving their product gives, on net, a favorable signal-to-noise ratio tradeoff. This effect also seems to hold for finite-time horizons, see Appendix C.1 for details. Our theoretical results suggest that the benefits of LA might come from a pure optimization advantage, rather than from implicit regularization, and that training algorithms which can trade off singal-to-noise ratio without adjusting the base learning rate are generally useful.

3.2 Dynamics of LA-DiLoCo

We can extend the analysis of LA to LA-DiLoCo— that is, LA-DiLoCo with SGD as both the inner and outer optimizer. Given R workers indexed by k, the dynamics is given by:

$$\boldsymbol{\psi}_{t;k} := \boldsymbol{\theta}_t \text{ (when } t//S = 0) \tag{13}$$

$$\boldsymbol{\psi}_{t+1;k} - \boldsymbol{\psi}_{t;k} = -\frac{\eta}{B}\mathbf{J}^\top \mathbf{P}_{t;k}\mathbf{z}(\boldsymbol{\psi}_{t;k}) \text{ for } S \text{ steps} \tag{14}$$

$$\boldsymbol{\theta}_{t+T} - \boldsymbol{\theta}_t = \nu \left(\frac{1}{R}\sum_{k=1}^{R} \boldsymbol{\psi}_{t+S;k} - \boldsymbol{\theta}_t \right) \tag{15}$$

where the $\mathbf{P}_{t;k}$ are all i.i.d. distributed as the $\mathbf{P}$ before. This is similar to the dynamics of LA, except there are R independent copies of the parameters that are propagated for S steps. The average of these parameters is then propagated to the outer optimizer. Here B is the batch size per-worker, so the total batch size across workers is RB.

In the linear regression setting, the dynamics can once again be written over function space and we can derive the high dimensional dynamics of $\mathbf{p}$ (Appendix A.1):

$$\mathbf{p}_{t+S} = \left([(1-\nu)\mathbf{I} + \nu(\mathbf{I} - \eta\boldsymbol{\Lambda})^S]^2 + \frac{\nu^2}{R}[\mathbf{T}_S - (\mathbf{I} - \eta\boldsymbol{\Lambda})^{2S}] \right) \mathbf{p}_t \tag{16}$$

This is identical to the dynamics of LA, but with the noise reduced by R. This suggests that adding more workers with the same B is always beneficial to training in this setting.

A more interesting and practically relevant scenario is one where the *total batch size* remains fixed, while the number of workers is varied. This allows, for example, a comparison between data-parallel training and LA-DiLoCo style training. If $R > 1$ is optimal here it would suggest that even when synchronous training is no more costly than asynchronous training, it is beneficial to decouple the parameters across workers.

To analyze this setting, it is useful to examine the structure of the noise term — the only place where LA-DiLoCo departs from LA. It is informative to write the full noise term as a power series. We note that $\mathbf{T}_S$ can be written as the sum of all possible S-words made of products of $\mathbf{A}$ and $\mathbf{B}$. We define the power series $\mathbf{M}_s$ by $\mathbf{M}_s \equiv \sum_{\mathbf{W}\in\pi_s^S} \mathbf{W}$, where π_s^S is the set of products of length S of $\mathbf{A}$ and $\mathbf{B}$ with exactly s copies of $\mathbf{A}$. We note that $\mathbf{M}_0 = (\mathbf{I} - \eta\boldsymbol{\Lambda})^{2S}$. A simple but tedious calculation shows us that for $s > 0$, $\mathbf{M}_s$ can be written as $\frac{\eta^{2s}}{B^s}\tilde{\mathbf{M}}_s(\eta)$, where $\tilde{\mathbf{M}}_s$ is independent of B and R:

$$\frac{\nu^2}{R}[\mathbf{T}_S - (\mathbf{I} - \eta\boldsymbol{\Lambda})^{2S}] = \sum_{s=1}^{S} a_s \tilde{\mathbf{M}}_s(\eta),\ a_s \equiv \frac{\nu^2\eta^{2s}}{RB^s} \tag{17}$$

In terms of the total batch size $B_{tot} = BR$, the coefficients are given by

$$a_s = \frac{R^{s-1}}{B_{tot}^s}\nu^2\eta^{2s} \tag{18}$$

This shows that at the same *total batch size*, using multiple workers $R > 1$ (LA-DiLoCo) generates *more noise* than single worker LA at matching η and ν. This suggests that in this setting, using LA-DiLoCo does not improve optimization — unlike the benefit gained from switching to LA from SGD. This can be formally proven for near-isotropic data (Appendix A.4):

Theorem 3.2. *Consider the dynamics of Equation 16 for fixed ν and η. Then if $\boldsymbol{\Lambda} = \lambda_0\mathbf{I}$, $\lambda_0 > 0$, for fixed total batch size $B_{\text{tot}} \equiv BR$, the eigenvalues of the linear system strictly increase with R. Additionally, given a diagonal matrix $\boldsymbol{\Delta}$, given a pair $R_1 < R_2$, there exists an $\epsilon > 0$ such that the eiegenvalues for the system with R_2 workers has larger eigenvalues than the system for R_1 workers for $\boldsymbol{\Lambda} = \lambda_0\mathbf{I} + \epsilon\boldsymbol{\Delta}$.*

Equation 16 and the power series decomposition can also be used to derive the relationships between hyperparameters in the two methods. We first consider the simple case where $\eta\lambda S \ll 1$ and $\nu\eta\lambda S \ll 1$ for all eigenvalues λ in $\mathbf{\Lambda}$. In this setting the noise is small, and the dynamics is dominated by the deterministic term. The gradient varies slowly, and the update equation is approximately $\mathbf{p}_{t+S} \approx (1 - \nu\eta\mathbf{\Lambda})\mathbf{p}_t$. In this setting, keeping the product $\nu\eta$ constant preserves the dynamics, across values of η and R.

The more interesting case is where there is non-trivial noise. Consider two training scenarios: one with LA with learning rate pair $(\nu_{\text{loc}}, \eta_{\text{loc}})$ and the other with LA-DiLoCo with R workers and learning rate pair $(\nu_{\text{dil}}, \eta_{\text{dil}})$. If the total batch size B_{tot} is fixed across the methods, we can compute the ratio of the power series coefficients a_s. The B_{tot} dependence vanishes and we are left with:

$$\frac{a_{s,\text{dil}}}{a_{s,\text{loc}}} = \left(\frac{\nu_{\text{dil}}}{\nu_{\text{loc}}}\right)^2 R^{s-1} \left(\frac{\eta_{\text{dil}}}{\eta_{\text{loc}}}\right)^{2s} \tag{19}$$

This ratio can be set to 1 with the choice

$$\eta_{\text{dil}} := R^{-1/2}\eta_{\text{loc}},\ \nu_{\text{dil}}\eta_{\text{dil}} := \nu_{\text{loc}}\eta_{\text{loc}} \tag{20}$$

This suggests that one way to combat the increase in noise due to additional workers is to decrease the inner learning rate, but compensate by increasing the outer learning rate. In Appendix A.5 we provide numerical evidence for the linear case demonstrating that this scaling rule makes learning dynamics similar over a non-trivial range of R and η. Because of the higher order terms the correspondence is not exact and breaks down for large R and large η. We get similar results for a two-layer MLP on MNIST (Appendix C.2), suggesting that these effects may survive in the non-linear setting.

In summary, our analysis suggests that at similar total batch sizes, synchronous training is favored over asynchronous training. However, the outer learning rate in LA-DiLoCo can partially compensate for the increased noise generation by using the theoretically motivated scaling rule.

4 The benefits of dual momentum

In practical scenarios, DiLoCo is used with more complex optimizers in both the inner and outer loop. In particular, using momentum in both optimizers is crucial to successful methods. Successful formulations will often use Nesterov-style momentum in the outer loop [10, 11], which we also found to be better than EMA style momentum (Appendix C.3).

In order to improve understanding of what momentum does in these methods, and to generate hypotheses for why Nesterov-style momentum is preferred, we analyze "Super"-Lookahead (SLA) — LA with SGD momentum in both the inner and outer optimizers. We focus on the deterministic setting to obtain interpretable results, but note that existing methods can be used in-principle to extend the analysis to the stochastic setting (see Collins-Woodfin et al. [15] for the relevant methodology).

We show that the outer optimizer acts on a normalized, "effective" problem, where the outer momentum can accelerate both small eigenmodes, but also those at the edge of stability, and this effect is amplified with Nesterov-style momentum. Our results are compatible with prior work which used worst-case analysis to analyze the effective condition number of related optimizers [2, 5], which also found an "effective geometry" that determines the dynamics of a two-phase optimizer, with improved bounds for Nesterov momentum.

4.1 Review of single momentum

We review the behavior of heavy ball and Nesterov momentum in our linear regression setting, establishing notation and building intuition for later analyses. EMA style momentum can be written as:

$$\boldsymbol{\zeta}_{t+1} = (1 - \beta_1)\mathbf{g}_t + \beta_1\boldsymbol{\zeta}_t \tag{21}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\boldsymbol{\zeta}_{t+1} \tag{22}$$

where $\mathbf{g}$ is the gradient at $\boldsymbol{\theta}$. Similar to the SGD, we can write the update equations in function space, using the residuals $\mathbf{z}$ and the function space momentum $\mathbf{k} \equiv \mathbf{J}\boldsymbol{\zeta}$. The resulting dynamics diagonalize over the eigenbasis of $\hat{\Theta}$; for each eigenvalue λ the projections z and k onto the eigenmodes evolve as

$$\begin{pmatrix} z_{t+1} \\ k_{t+1} \end{pmatrix} = \begin{pmatrix} 1 - \eta(1-\beta_1)\lambda & -\eta\beta_1 \\ (1-\beta_1)\lambda & \beta_1 \end{pmatrix} \begin{pmatrix} z_t \\ k_t \end{pmatrix} \tag{23}$$

The convergence rate of the eigenmode is determined by the magnitude of the largest eigenvalue $\omega_{\max}$ of the transition matrix, since applying the transition matrix t times results in a total max eigenvalue of $\omega_{\max}^t$. We define the convergence rate $r \equiv -\log(|\omega_{\max}|)$. The eigenvalues $\omega_\pm$ are given by

$$\omega_\pm = \frac{1}{2}\left[(1-\eta(1-\beta_1)\lambda+\beta_1) \pm \sqrt{(1-\eta(1-\beta_1)\lambda+\beta_1)^2 - 4\beta_1}\right] \tag{24}$$

For small learning rates $\eta\lambda \ll 1-\beta_1$ the largest eigenvalue is $\omega_+ \approx 1-\eta\lambda$, with associated convergence rate $r \equiv -\log(1-\eta\lambda) \approx \eta\lambda$ —the same as SGD. The dynamics diverges for eigenvalues where $\eta\lambda > 2\frac{1+\beta_1}{1-\beta_1}$, allowing a maximal learning rate that is $\frac{1+\beta_1}{1-\beta_1}$ larger than GD. The ability to use this larger learning rate without sacrificing convergence of larger eigenmodes allows for faster convergence of smaller eigenmodes.

One interesting phenomenon induced by momentum is the existence of complex eigenmodes. These correspond to oscillatory modes whose convergence rate given by the norm of their eigenvalues $|\omega_\pm| = \sqrt{\beta_1}$ — which is independent of η and λ. If $\beta_1 = 1-\delta$, $\delta \ll 1$ the convergence rate of the modes is $r \approx \delta/2$ — a potentially slow convergence. These modes occur when $(1-\eta(1-\beta_1)\lambda + \beta_1)^2 - 4\beta_1 < 0$, which for $1-\beta_1 \ll 1$ corresponds to

$$\frac{1-\beta_1}{4} < \eta\lambda < \frac{2(1+\beta_1)}{1-\beta_1} \tag{25}$$

That is, for β_1 close to 1 there is a range of $\eta\lambda$ spanning a factor of $O((1-\beta_1)^{-2})$ which converge slowly. This is one of the issues with using β_1 too close to 1; even in this simple problem there is a tradeoff between stability and convergence rate that can't be mitigated through choice of learning rate. Similarly, for Nesterov-style momentum we have:

$$\boldsymbol{\zeta}_t = (1-\beta_1)\mathbf{g}_t + \beta_1\boldsymbol{\zeta}_{t-1} \tag{26}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta[(1-\beta_1)\mathbf{g}_t + \beta_1\boldsymbol{\zeta}_t] \tag{27}$$

In the eigenspace this nets the linear system

$$\begin{pmatrix} z_{t+1} \\ k_{t+1} \end{pmatrix} = \begin{pmatrix} 1-(1-\beta_1^2)\eta\lambda & -\eta\beta_1^2 \\ (1-\beta_1)\lambda & \beta_1 \end{pmatrix} \begin{pmatrix} z_t \\ k_t \end{pmatrix} \tag{28}$$

This form is in units which matches the dynamics of EMA momentum for small eigenvalues $\eta\lambda \ll 1$. The traditional form can be recovered with the substitution $\tilde{\eta} := (1-\beta_1)\eta$. $\omega_\pm$ are complex when

$$\frac{(1-\beta_1)}{(1+\beta_1)^2} < \eta\lambda < \frac{1}{1-\beta_1} \tag{29}$$

once again giving a range of $O((1-\beta_1)^{-2})$ for complex eigenvalues. However, for Nesterov-style momentum the eigenvalue magnitude and convergence rate depend on $\eta\lambda$:

$$|\omega_\pm| = \sqrt{\beta_1(1-(1-\beta_1)\eta\lambda)},\ r = -\frac{1}{2}\log(\beta_1(1-(1-\beta_1)\eta\lambda)) \tag{30}$$

For $\eta\lambda \ll (1-\beta_1)^{-1}$ with complex eigenvalues we still have the same convergence issue as before; however, for $\eta\lambda \sim (1-\beta_1)^{-1}$ (near the end of the complex region) the convergence is much improved. We will see in the next section that this makes Nesterov momentum uniquely suited to use in the outer optimizer of SLA.

4.2 SLA and dual momenta

We define the general Super Lookahead (SLA) optimizer as:

$$\boldsymbol{\psi}_t := \boldsymbol{\theta}_t \text{ (when } t//S = 0) \tag{31}$$

$$\boldsymbol{\psi}_{t+1} - \boldsymbol{\psi}_t = InnerOpt(\mathbf{g}(\boldsymbol{\psi}_t)) \text{ for } S \text{ steps} \tag{32}$$

$$\boldsymbol{\theta}_{t+T} = \boldsymbol{\theta}_t - OuterOpt(\boldsymbol{\theta}_t - \boldsymbol{\psi}_{t+T}) \tag{33}$$

where $InnerOpt$ and $OuterOpt$ are gradient based optimizers. The outer optimizer uses the difference $\boldsymbol{\theta}_t - \boldsymbol{\psi}_{t+T}$ in place of a gradient. We consider the case where $InnerOpt$ and $OuterOpt$ are GD with EMA style momentum (full definition in Appendix B.1). We have inner and outer

learning rates η and ν as before, and now have inner and outer momentum parameters β_{in} and β_{out}. In this setting the function space updates can be written as:

$$\tilde{\mathbf{z}}_t := \mathbf{z}_t \text{ (when } t//S = 0) \tag{34}$$

$$\left.\begin{aligned} \tilde{\mathbf{z}}_{t+1} &= (\mathbf{I} - \eta(1-\beta_{\text{in}})\hat{\Theta})\tilde{\mathbf{z}}_t - \eta\beta_{\text{in}}\tilde{\mathbf{k}}_t \\ \tilde{\mathbf{k}}_{t+1} &= (1-\beta_{\text{in}})\hat{\Theta}\tilde{\mathbf{z}}_t + \beta_{\text{in}}\tilde{\mathbf{k}}_t \end{aligned}\right\} \quad \text{for } S \text{ steps} \tag{35}$$

$$\mathbf{k}_{t+T} = -(1-\beta_{\text{out}})[\tilde{\mathbf{z}}_{t+T-1} - \mathbf{z}_t] + \beta_{\text{out}}\mathbf{k}_t,\ \mathbf{z}_{t+S} = \mathbf{z}_t - \nu\mathbf{k}_{t+T} \tag{36}$$

We note that only $\tilde{\mathbf{z}}_t$ is synced between cycles; $\tilde{\mathbf{k}}$ is only modified via the inner optimizer. In this setting, momentum rules can be generally written as

$$\begin{pmatrix} \tilde{\mathbf{z}}_{t+S} \\ \tilde{\mathbf{k}}_{t+S} \\ \mathbf{z}_{t+S} \\ \mathbf{k}_{t+S} \end{pmatrix} = \mathbf{T}_{\text{sync}}\mathbf{T}_{\text{out}}\mathbf{T}_{\text{in}} \begin{pmatrix} \tilde{\mathbf{z}}_t \\ \tilde{\mathbf{k}}_t \\ \mathbf{z}_t \\ \mathbf{k}_t \end{pmatrix} \tag{37}$$

where $\mathbf{T}_{\text{in}}$ represents the S steps of the inner optimizer, $\mathbf{T}_{\text{out}}$ represents the outer optimizer step, and $\mathbf{T}_{\text{sync}}$ represents the syncing of inner and outer optimizers (in this case, the syncing of $\tilde{\mathbf{z}}_t$ to $\mathbf{z}_t$ at the end of the cycle. By analyzing the eigenvalues of the product $\mathbf{T}_{\text{tot}} \equiv \mathbf{T}_{\text{sync}}\mathbf{T}_{\text{out}}\mathbf{T}_{\text{in}}$, we can understand the convergence rates of SLA as a function of the optimizer parameters. Detailed expressions for the matrices can be found in Appendix B.3.

4.3 Analysis of a 2-dimensional reduction

In order to build intuition for the convergence behavior, it is useful to define a variant of the optimizer where the eigenvalues of $\mathbf{T}_{\text{tot}}$ are analytically tractable. Consider the additional rule $\tilde{\mathbf{k}}_t := 0$ at the end of the S step cycle. This modifies $\mathbf{T}_{\text{sync}}$ only. Working once again in the $\hat{\Theta}$ eigenbasis, the dynamics per-cycle for each eigendirection can be described in terms of the projections z and k alone, since no information from $\tilde{\mathbf{z}}$ or $\tilde{\mathbf{k}}$ is carried over cycle-to-cycle. Then $\mathbf{T}_{\text{in}}$ is given by

$$\mathbf{T}_{\text{in}} = \begin{pmatrix} \mathbf{F} & 0 \\ 0 & \mathbf{I} \end{pmatrix},\ \mathbf{F} = \begin{pmatrix} 1-(1-\beta_{\text{in}})\eta\lambda & -\eta\beta_{\text{in}} \\ (1-\beta_{\text{in}})\lambda & \beta_{\text{in}} \end{pmatrix}^S \tag{38}$$

then the update rule in an eigendirection reduces to

$$\begin{pmatrix} z_{t+S} \\ k_{t+S} \end{pmatrix} = \begin{pmatrix} 1-\nu(1-\beta_{\text{out}})[1-F_{11}] & -\nu\beta_{\text{out}} \\ (1-\beta_{\text{out}})[1-F_{11}] & \beta_{\text{out}} \end{pmatrix} \begin{pmatrix} z_t \\ k_t \end{pmatrix} \tag{39}$$

Therefore SLA is equivalent to Momentum-GD on an effective spectrum $1 - F_{11}$. This structure holds for any inner optimizer whose steps can be written as a linear function of z_t and k_t.

If $1 - F_{11} \ll 1$, then the corresponding convergence rate per-cycle is $r \approx \nu(1 - F_{11})$. This occurs for small eigenvalues $\eta\lambda S \ll 1$, where the *per step* convergence rate $r \equiv -\frac{1}{S}\log(|\omega_{\max}|)$, where ω are the eigenvalues of $\mathbf{T}_{\text{tot}}$ is given by $r \approx \nu\eta\lambda$. We get acceleration over GD if $\nu\eta > \eta_{\text{GD}}$.

We also have $1 - F_{11} \ll 1$ for slowly converging eigenmodes near the edge of stability. For example if $\beta_{\text{in}} = 0$ (no inner momentum) and $\eta\lambda = 2 - \delta$ for $\delta \ll 1$, then $F_{11} \approx 1 - S\delta$ and the per-cycle convergence rate becomes $r \approx \nu\delta$. A similar effect applies to the inner eigenmodes with imaginary eigenvalues if $S(1 - \beta_{\text{in}}) \ll 1$. In these cases, we have acceleration if $\nu > 1$. Therefore outer momentum can improve convergence of any slowly converging mode with sufficiently large ν.

The downside of outer momentum is that any modes which make significant optimization progress in the inner loop will be slowed down. For example, consider damped eigenmodes; for $\beta_{\text{out}} = 1 - \delta$, $\delta \ll 1$ the per-step convergence rate will be $r = \frac{1}{S}(1 - \delta/2)$ – *decreasing* for longer cycles, since the *per-cycle* convergence rate only depends on β_{out}. The best-optimizing eigenmodes in the inner optimizer can become the slowest in the outer optimizer. These results suggest that β_{out} will likely need to be even further from 1 than β_{in}.

These convergence issues can be greatly mitigated by using Nesterov-style momentum. From the full dynamics in Appendix B.2, combined with inner momentum zeroing out the 2-d dynamics becomes:

$$\begin{pmatrix} z_{t+S} \\ k_{t+S} \end{pmatrix} = \begin{pmatrix} 1-\nu(1-\beta_{\text{out}}^2)(1-F_{11}) & -\nu\beta_{\text{out}}^2 \\ (1-\beta_{\text{out}})(1-F_{11}) & \beta_{\text{out}} \end{pmatrix} \begin{pmatrix} z_t \\ k_t \end{pmatrix} \tag{40}$$

Once again, the effective eigenvalues are given by $1 - F_{11}$. This means that in the damped/complex eigenvalue regime, the per-cycle eigenvalue magnitude is $\sqrt{\beta_{\text{out}}(1 - (1 - \beta_{\text{out}})\nu(1 - F_{11}))}$. Any eigenvalues where $\nu(1 - F_{11}) \ll (1 - \beta_{\text{out}})^{-1}$ will still experience slowdown. Since $1 - F_{11} \in [0, 1]$ and $(1 - \beta_{\text{out}})^{-1} > 1$, the only hope to combat this is to pick $\nu \geq (1 - \beta_{\text{out}})^{-1}$ (corresponding to $\nu \geq 1$ with traditionally parameterized Nesterov momentum).

Unlike the inner optimizer, where the learning rate can only be tuned to accelerate a small range of eigenmodes (those with $\eta\lambda \sim (1 - \beta_{\text{in}})^{-1}$, there is a special value of ν that can mitigate the damping for many eigenmodes. The choice $\nu = (1 - \beta_{\text{out}})^{-1}$ accelerates all eigenmodes with effective eigenvalues $1 - F_{11} \approx 1$, or alternatively $F_{11} \ll 1$. There is a large class of eigendirections with this property: namely eigenmodes which make significant progress on the loss in a single cycle. These modes are normally damped by the outer optimizer, but with Nesterov momentum and a well-tuned outer learning rate the damping is mitigated. For example, if $F_{11} = ae^{-\tilde{\lambda}S}$ as S is varied, and $\nu = (1 - \beta_{\text{out}})^{-1}$, the convergence rate per cycle is $r > S\tilde{\lambda}$, and per step is $r > \tilde{\lambda}$.

Overall, this analysis shows that applying two momenta at different scales has non-trivial effects that can't be replicated by a single momentum. With the correct choice of momentum rule the outer momentum can speed up convergence of eigendirections slowed down by the inner momentum, while preserving progress on well-converging modes. This effect can't be captured by simple interventions like adding weight averaging on top of momentum (Appendix B.4) or proposed algorithms like GPA [16] which try to capture some of the effects of SLA (Appendix B.5).

4.4 Numerical results

We analyzed convergence rates for EMA and Nesterov momenta by performing eigendecomposition of $\mathbf{T}_{\text{tot}}$ for both the reduced system analyzed above, as well as the original system described by Equations 34 to 36 where $\tilde{\mathbf{k}}$ is not reset to 0 every cycle. We fixed $\eta = 1$, $\lambda = 0.2$, $\beta_{\text{in}} = 0.9$, $\beta_{\text{out}} = 0.8$, and kept the inner optimizer fixed to EMA momentum. This put the inner optimizer in the imaginary eigenvalue regime. These values were chosen to illustrate our general points, which hold across a wide range of parameter values.

The convergence rates of the reduced system and the original system are similar for larger S (Figure 1). Resetting $\tilde{\mathbf{k}}$ to 0 improves the convergence rate for small S. If we set ν to a generic stable value, we see that the convergence rate for Nesterov is larger than the convergence rate for EMA momentum, by a fixed factor for large S (Figure 1, top). However we see that the per-step convergence rate decreases as $1/S$ because the per-cycle convergence rate is fixed, as predicted by the theory.

However, if we set ν to the critical value $(1 - \beta_{\text{out}})^{-1}$, then we see that the per-step convergence rate of Nesterov momentum goes to a constant as predicted (Figure 1, bottom). This highlights the point that the critical outer learning rate is useful for preserving convergence rates for large S.

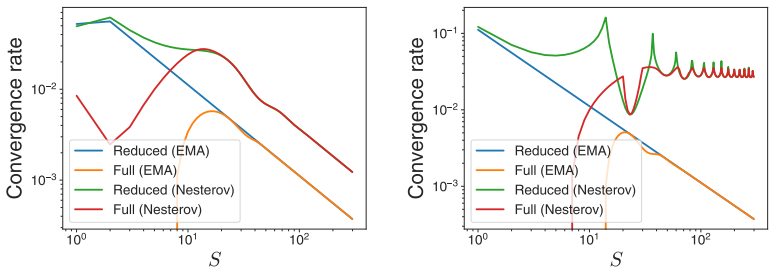

Figure 1: Eigenmodes damped by inner optimizer show decreasing per-step convergence rate with momentum-GD in the outer optimizer (left, $\nu = 2$). Convergence rate is improved for Nesterov momentum. At the critical $\nu = (1 - \beta_{\text{out}})^{-1}$, Nesterov convergence rate becomes S-invariant at large S (right). Reduced system with $\tilde{\mathbf{k}}_t := 0$ every cycle has similar behavior to full dynamics with $\tilde{\mathbf{k}}_t$ preserved for larger S, but shows faster convergence/better stability at smaller S. All experiments have $\eta = 1$, $\lambda = 0.2$, $\beta_{\text{in}} = 0.9$, $\beta_{\text{out}} = 0.8$.

5 Discussion

This work represents the first average case analysis of LA-DiLoCo in the high dimensional setting, and reveals the complex tradeoff between signal and noise that controls the convergence rates. Even with a single worker LA-DiLoCo is quantitatively different from SGD, with provably improved convergence rates in certain settings. Our analysis suggests that all else equal, synchronous optimizers are better than asynchronous ones, but it is possible to design good hyperparameter transfer schemes for a modest number of workers. Our analysis of the momentum-equipped SLA variants in the deterministic setting give new insights into the opportunities for acceleration, and suggest that Nesterov momentum minimizes the downsides of the approach — mirroring and perhaps explaining observations in practical settings [10, 11].

Overall the benefits of two-phase optimizers in this setting come from the fact that the outer optimizer operates on an "effective" problem defined by the action of the inner optimizer. These benefits are due to new tradeoffs between speed and stability surfaced by the structure of the optimizer, that are easily accessible via hyperparameter tuning. These effects can't be recreated with traditional optimizers equipped with weight averaging (Appendix B.4), nor by algorithms like GPA [16] which seek to mimic the benefits of SLA (Appendix B.5). In general it seems that there is a benefit to surfacing control over the variance properties of the optimizer without affecting the "overall" learning rate.

The main weakness of our analysis is that it assumes the geometry of the loss landscape (e.g. the Hessian/NTK) is fixed. One next step would be to analyze cases where the eigenvalues and eigenvectors of the Hessian shift over time, or where the loss landscape is not well characterized by first and second derivatives on the scales relevant for the learning dynamics. In particular, it is known that algorithmic choices induce feedback on the large eigenvalues of the loss landscape [17–19]; understanding how this phenomenon affects two-phase optimizers is likely crucial to improve these methods further, and scale them to larger workloads.

Our analysis suggests that the optimization dynamics of these two-phase optimizers has unique properties not found in most commonly used training algorithms. Though the periodic structure is an aesthetic departure from typical algorithms, the ability to explicitly access different time and curvature scales is robustly beneficial. We speculate that these effects can eventually be bridged with other methods that exploit multiple timescales [20, 21]. The insights presented here are just the starting point to new exciting paradigms in training algorithms.

6 Acknowledgments

The author would like to thank Keith Rush and Zachary Charles for inspiring the project, lending their expertise about DiLoCo and other optimizers, and their guidance in preparing the manuscript.

References

[1] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[2] Zachary Charles and Jakub Konečnỳ. Convergence and accuracy trade-offs in federated learning and meta-learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2575–2583. PMLR, 2021.

[3] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=LkFG3lB13U5`.

[4] Grigory Malinovskiy, Dmitry Kovalev, Elnur Gasanov, Laurent Condat, and Peter Richtarik. From local sgd to local fixed-point methods for federated learning. In *International Conference on Machine Learning*, pages 6692–6701. PMLR, 2020.

[5] Zachary Charles and Keith Rush. Iterated vector fields and conservatism, with applications to federated learning. In Sanjoy Dasgupta and Nika Haghtalab, editors, *Proceedings of The 33rd International Conference on Algorithmic Learning Theory*, volume 167 of *Proceedings*

of Machine Learning Research, pages 130–147. PMLR, 29 Mar–01 Apr 2022. URL https://proceedings.mlr.press/v167/charles22a.html.

[6] Ahmed Khaled, Satyen Kale, Arthur Douillard, Chi Jin, Rob Fergus, and Manzil Zaheer. Understanding outer optimizers in local sgd: Learning rates, momentum, and acceleration. *arXiv preprint arXiv:2509.10439*, 2025.

[7] Arthur Douillard, Qixuan Feng, Andrei A Rusu, Rachita Chhaparia, Yani Donchev, Adhiguna Kuncoro, Marc'Aurelio Ranzato, Arthur Szlam, and Jiajun Shen. Diloco: Distributed low-communication training of language models. *arXiv preprint arXiv:2311.08105*, 2023.

[8] Sami Jaghouar, Jack Min Ong, and Johannes Hagemann. Opendiloco: An open-source framework for globally distributed low-communication training, 2024. URL https://arxiv.org/abs/2407.07852.

[9] Arthur Douillard, Yani Donchev, J Keith Rush, Satyen Kale, Zachary Charles, Gabriel Teston, Zachary Garrett, Jiajun Shen, Ross McIlroy, David Lacey, Alexandre Rame, Arthur Szlam, MarcAurelio Ranzato, and Paul R Barham. Streaming diloco with overlapping communication. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=yYk3zK0X6Q.

[10] Zachary Charles, Gabriel Teston, Lucio Dery, Keith Rush, Nova Fallen, Zachary Garrett, Arthur Szlam, and Arthur Douillard. Communication-efficient language model training scales reliably and robustly: Scaling laws for diloco. *arXiv preprint arXiv:2503.09799*, 2025.

[11] Dominik Kallusky, Vinay Rao, Vishal Nandavanam, and Hao-Jun Michael Shi. Snoo: Step-k nesterov outer optimizer-the surprising effectiveness of nesterov momentum applied to pseudo-gradients. *arXiv preprint arXiv:2510.15830*, 2025.

[12] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. *Advances in neural information processing systems*, 32, 2019.

[13] Kiwon Lee, Andrew Cheng, Elliot Paquette, and Courtney Paquette. Trajectory of mini-batch momentum: batch size saturation and convergence in high dimensions. *Advances in Neural Information Processing Systems*, 35:36944–36957, 2022.

[14] Atish Agarwala and Jeffrey Pennington. High dimensional analysis reveals conservative sharpening and a stochastic edge of stability. *arXiv preprint arXiv:2404.19261*, 2024.

[15] Elizabeth Collins-Woodfin, Courtney Paquette, Elliot Paquette, and Inbar Seroussi. Hitting the high-dimensional notes: An ode for sgd learning dynamics on glms and multi-index models. *Information and Inference: A Journal of the IMA*, 13(4):iaae028, 2024.

[16] Aaron Defazio, Konstantin Mishchenko, Parameswaran Raman, Hao-Jun Michael Shi, and Lin Xiao. Smoothing diloco with primal averaging for faster training of llms. *arXiv preprint arXiv:2512.17131*, 2025.

[17] Jeremy M Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. *arXiv preprint arXiv:2103.00065*, 2021.

[18] Jeremy M Cohen, Behrooz Ghorbani, Shankar Krishnan, Naman Agarwal, Sourabh Medapati, Michal Badura, Daniel Suo, David Cardoze, Zachary Nado, George E Dahl, and Justin Gilmer. Adaptive gradient methods at the edge of stability. *arXiv preprint arXiv:2207.14484*, 2022.

[19] Vincent Roulet, Atish Agarwala, Jean-Bastien Grill, Grzegorz Swirszcz, Mathieu Blondel, and Fabian Pedregosa. Stepping on the edge: Curvature aware learning rate tuners. *Advances in Neural Information Processing Systems*, 37:47708–47740, 2024.

[20] Aaron Defazio, Xingyu Yang, Harsh Mehta, Konstantin Mishchenko, Ahmed Khaled, and Ashok Cutkosky. The road less scheduled. *Advances in Neural Information Processing Systems*, 37:9974–10007, 2024.

[21] Damien Ferbach, Katie Everett, Gauthier Gidel, Elliot Paquette, and Courtney Paquette. Dimension-adapted momentum outscales sgd. *arXiv preprint arXiv:2505.16098*, 2025.

[22] George E. Dahl, Frank Schneider, Zachary Nado, Naman Agarwal, Chandramouli Shama Sastry, Philipp Hennig, Sourabh Medapati, Runa Eschenhagen, Priya Kasimbeg, Daniel Suo, Juhan Bae, Justin Gilmer, Abel L. Peirson, Bilal Khan, Rohan Anil, Mike Rabbat, Shankar Krishnan, Daniel Snider, Ehsan Amid, Kongtao Chen, Chris J. Maddison, Rakshith Vasudev, Michal

Badura, Ankush Garg, and Peter Mattson. Benchmarking Neural Network Training Algorithms, 2023.

[23] Priya Kasimbeg, Frank Schneider, Runa Eschenhagen, Juhan Bae, Chandramouli Shama Sastry, Mark Saroufim, Feng Boyuan, Less Wright, Edward Z. Yang, Zachary Nado, Sourabh Medapati, Philipp Hennig, Michael Rabbat, and George E. Dahl. Accelerating neural network training: An analysis of the AlgoPerf competition. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=CtM5xjRSfm.

A High dimensional dynamics of LA and LA-DiLoCo

A.1 Derivation of the dynamics of p

We derive the dynamics in the case of LA-DiLoCo here; LA is then the special case where $R = 1$. We have the dynamics:

$$\tilde{\mathbf{z}}_{t;k} := \mathbf{z}_t \text{ (when } t//S = 0) \tag{41}$$

$$\tilde{\mathbf{z}}_{t+1;k} - \tilde{\mathbf{z}}_{t;k} = -\eta \frac{D}{B} \hat{\Theta} \mathbf{P}_{t;k} \tilde{\mathbf{z}}_{t;k} \text{ for } S \text{ steps} \tag{42}$$

$$\mathbf{z}_{t+S} - \mathbf{z}_t = \frac{\nu}{R} \sum_{k=1}^{R} (\tilde{\mathbf{z}}_{t+S;k} - \mathbf{z}_t) \tag{43}$$

where the $\mathbf{P}_{t;k}$ are i.i.d. projection matrices with exactly B 1s on the diagonal and 0 everywhere else, for randomly chosen indices. The dynamics of the first moment of $\mathbf{z}$ is given by:

$$\mathrm{E}[\mathbf{z}_{t+S}|\mathbf{z}_t] = \left[(1-\nu) + \nu(1-\eta\hat{\Theta})\right] \mathbf{z}_t \tag{44}$$

The second moment evolves as

$$\mathrm{E}[\mathbf{z}_{t+S}\mathbf{z}_{t+S}^\top|\mathbf{z}_t] = \mathrm{E}\left[\left((1-\nu)\mathbf{z}_t + \frac{\nu}{R}\sum_{k=1}^{R}\tilde{\mathbf{z}}_{t+S;k}\right)\left((1-\nu)\mathbf{z}_t + \frac{\nu}{R}\sum_{k=1}^{R}\tilde{\mathbf{z}}_{t+S;k}\right)^\top\right] \tag{45}$$

Simplifying, we have:

$$\begin{aligned}\mathrm{E}[\mathbf{z}_{t+S}\mathbf{z}_{t+S}^\top|\mathbf{z}_t] = (1-\nu)^2\mathbf{z}_t\mathbf{z}_t^\top + \frac{\nu(1-\nu)}{R}\left(\sum_{k=1}^{R}\mathrm{E}[\tilde{\mathbf{z}}_{t+S;k}]\mathbf{z}_t^\top + \mathbf{z}_t\mathrm{E}[\tilde{\mathbf{z}}_{t+S;k}^\top]\right) \\ + \frac{\nu^2}{R^2}\sum_{j=1}^{R}\sum_{k=1}^{R}\mathrm{E}[\tilde{\mathbf{z}}_{t+S;j}\tilde{\mathbf{z}}_{t+S;k}^\top]\end{aligned} \tag{46}$$

Note that $\tilde{\mathbf{z}}_{t+S;j}$ and $\tilde{\mathbf{z}}_{t+S;k}$ are independent for $j \neq k$. We also note that the first and second moments of $\tilde{\mathbf{z}}_{t+S;k}$ given $\mathbf{z}_t$ are identical to those of $\mathbf{z}_{t+S}$ from SGD. We define:

$$\boldsymbol{\mu}_{t,s} \equiv \mathrm{E}_{\mathrm{SGD}}[\mathbf{z}_{t+s}|\mathbf{z}_t],\ \boldsymbol{\Sigma}_{t,s} \equiv \mathrm{E}_{\mathrm{SGD}}[\mathbf{z}_{t+s}\mathbf{z}_{t+s}^\top|\mathbf{z}_t] \tag{47}$$

Then we have:

$$\mathrm{E}[\mathbf{z}_{t+S}\mathbf{z}_{t+S}^\top|\mathbf{z}_t] = (1-\nu)^2\mathbf{z}_t\mathbf{z}_t^\top + \nu(1-\nu)\left(\boldsymbol{\mu}_{t,S}\mathbf{z}_t^\top + \mathbf{z}_t\boldsymbol{\mu}_{t,S}^\top\right) + \nu^2\left(1-\frac{1}{R}\right)\boldsymbol{\mu}_{t,S}\boldsymbol{\mu}_{t,S}^\top + \frac{1}{R}\boldsymbol{\Sigma}_{t,S} \tag{48}$$

Grouping terms by R we have:

$$\mathrm{E}[\mathbf{z}_{t+S}\mathbf{z}_{t+S}^\top|\mathbf{z}_t] = \left[(1-\nu)\mathbf{z}_t + \nu\boldsymbol{\mu}_{t,S}\right]\left[(1-\nu)\mathbf{z}_t + \nu\boldsymbol{\mu}_{t,S}\right]^\top + \frac{\nu^2}{R}\left(\boldsymbol{\Sigma}_{t,S} - \boldsymbol{\mu}_{t,S}\boldsymbol{\mu}_{t,S}^\top\right) \tag{49}$$

To obtain the dynamics of $\mathbf{p}_t$, we use the definition $\mathbf{p}_t \equiv \boldsymbol{\Lambda}^+\mathrm{diag}(\mathbf{V}^\top\mathrm{E}_{\mathbf{P}}[\mathbf{z}_t\mathbf{z}_t^\top]\mathbf{V})$ combined with the previous results from SGD. We have:

$$\boldsymbol{\mu}_{t,S} = (\mathbf{I}-\eta\hat{\Theta})^S\mathbf{z}_t, \mathbf{V}^\top\boldsymbol{\mu}_{t,S}\boldsymbol{\mu}_{t,S}^\top\mathbf{V} = (\mathbf{I}-\eta\boldsymbol{\Lambda})^S\mathbf{V}^\top\mathbf{z}_t\mathbf{z}_t^\top\mathbf{V}(\mathbf{I}-\eta\boldsymbol{\Lambda})^S \tag{50}$$

which allows us to simplify the first two terms as

$$\boldsymbol{\Lambda}^+\mathrm{diag}(\mathbf{V}^\top\left[(1-\nu)\mathbf{z}_t + \nu\boldsymbol{\mu}_{t,S}\right]\left[(1-\nu)\mathbf{z}_t + \nu\boldsymbol{\mu}_{t,S}\right]^\top\mathbf{V}) = [(1-\nu)\mathbf{I} + \nu(\mathbf{I}-\eta\boldsymbol{\Lambda})^S]^2\mathbf{p}_t \tag{51}$$

In high dimensions, the results on SGD show that $\boldsymbol{\Sigma}_{t,S}$ transforms to $(\mathbf{A}+\mathbf{B})^S\mathbf{p}_t$, where

$$\mathbf{A} \equiv (\mathbf{I}-\eta\boldsymbol{\Lambda})^2,\ \ \mathbf{B} \equiv \left(\frac{1}{B}-\frac{1}{D}\right)\eta^2\boldsymbol{\Lambda}\mathbf{1}\mathbf{1}^\top\boldsymbol{\Lambda}. \tag{52}$$

This gives us the final equations for $\mathbf{p}_t$ for LA-DiLoCo:

$$\mathbf{p}_{t+S} = \left([(1-\nu)\mathbf{I} + \nu(\mathbf{I}-\eta\boldsymbol{\Lambda})^S]^2 + \frac{\nu^2}{R}[\mathbf{T}_S - (\mathbf{I}-\eta\boldsymbol{\Lambda})^{2S}]\right)\mathbf{p}_t \tag{53}$$

where

$$\mathbf{T}_s \equiv (\mathbf{A}+\mathbf{B})^s = \left[(\mathbf{I}-\eta\boldsymbol{\Lambda})^2 + \left(\frac{1}{B}-\frac{1}{D}\right)\eta^2\boldsymbol{\Lambda}\mathbf{1}\mathbf{1}^\top\boldsymbol{\Lambda}\right]^s \tag{54}$$

A.2 Deterministic dynamics

In the deterministic setting (full batch training), the update equation is:

$$\mathbf{p}_{t+1} = [(1-\nu)\mathbf{I} + \nu(\mathbf{I} - \eta\mathbf{\Lambda})^S]^2\mathbf{p}_t \tag{55}$$

This diagonalizes in the eigenbasis. To understand convergence it is sufficient to analyze the 1-dimensional system:

$$p_{t+1} = [(1-\nu) + \nu(1 - \eta\lambda)^S]^2 p_t \tag{56}$$

The dynamics is stable/convergent if and only if:

$$[(1-\nu) + \nu(1 - \eta\lambda)^S]^2 < 1 \tag{57}$$

Let $a = ((1-\eta\lambda)^S) - 1$. Then we have:

$$(1 + a\nu)^2 < 1 \rightarrow -\frac{2}{\nu} < a < 0 \tag{58}$$

This gives us

$$1 - \frac{2}{\nu} < (1 - \eta\lambda)^S < 1 \tag{59}$$

For $\nu = 1$, we recover the familiar $\eta\lambda < 2$ to prevent runaway growth. For even S, for $\nu < 2$ the condition is the same; after that the critical learning rate decreases with ν. For odd S, the condition is the same for all $\nu < 1$. For $\nu > 1$ gain the stable regime decreases. There is always some stable regime for all ν.

This analysis suggests that for $\nu < 1$, the stability of the inner SGD sets the stability of the overall dynamics. For larger ν, the training becomes harder to stabilize (requiring a smaller inner learning rate η), but there is always a convergent regime.

A.3 Convergence proof in the spectral norm

Theorem 3.1. *Consider the asymptotic infinite-time convergence rate of the optimization dynamics, governed by the spectral radius ρ of the transition matrix $\hat{\mathbf{T}}$ which controls the single cycle dynamics via Equation 16. Let $S \geq 2$ be even, and let $\mathbf{\Lambda}$ be non-degenerate. If the operator $\mathbf{T}_1$ controlling the one-step dynamics of SGD has non-degenerate largest eigenvalue at the optimal learning rate η^* for SGD ($\nu = 1$), then there exist settings of LA with $\nu > 1$ which have faster infinite-time convergence rate than SGD with the optimal learning rate for that spectrum and batch size.*

Proof. Recall that the transition matrix $\hat{\mathbf{T}}$ can be written as

$$\hat{\mathbf{T}}(\nu, \eta) = \left[(1-\nu)\mathbf{I} + \nu(\mathbf{I} - \eta\mathbf{\Lambda})^S\right]^2 + \nu^2\left[\mathbf{T}_S(\eta) - (\mathbf{I} - \eta\mathbf{\Lambda})^{2S}\right] \tag{60}$$

where $\mathbf{T}_S(\eta) = (\mathbf{A}(\eta) + \mathbf{B}(\eta))^S$, $\mathbf{A}(\eta) = (\mathbf{I} - \eta\mathbf{\Lambda})^2$, and $\mathbf{B}(\eta) = \gamma\eta^2\mathbf{\Lambda}\mathbf{1}\mathbf{1}^\top\mathbf{\Lambda}$ for $\gamma = 1/B - 1/D$. Let $\rho(\nu, \eta)$ be the corresponding spectral radius.

Consider varying ν and η while keeping their product constant. If we fix $\nu\eta = c$ and vary ν, we have $\rho(\nu, \eta(\nu)) = \rho(\nu, c/\nu)$. This gives us the derivative:

$$\frac{d\rho}{d\nu} = \frac{\partial\rho}{\partial\nu} - \frac{\partial\rho}{\partial\eta}\frac{c}{\nu^2} \tag{61}$$

At the SGD optimal learning rate $c = \eta^*$, with $\nu = 1$, $\frac{\partial\rho}{\partial\eta} = 0$. This gives us:

$$\frac{d\rho}{d\nu} = \frac{\partial\rho}{\partial\nu} \tag{62}$$

Therefore, if we can show that $\frac{d\rho}{d\nu} \neq 0$ for a particular (B, D) combination, there exists a setting of LA which has a better infinite-time convergence rate than SGD.

Since by assumption $\hat{\mathbf{T}}$ has a unique largest eigenvalue, we can write:

$$\rho(1, \eta^*) = \mathbf{u}^\top\hat{\mathbf{T}}\mathbf{u} \tag{63}$$

where $\mathbf{u}$ is the eigenvector corresponding to the largest eigenvalue. This means that the derivative at $\nu = 1$ is given by:

$$\frac{d\rho}{d\nu} = \mathbf{u}^\top \left.\frac{d\hat{\mathbf{T}}}{d\nu}\right|_{\nu=1} \mathbf{u} = 2\mathbf{u}^\top[\mathbf{T}_S - (1-\eta\mathbf{\Lambda})^S]\mathbf{u} \tag{64}$$

which gives us

$$\frac{1}{2}\frac{d\rho}{d\nu} = \lambda_1^S - \mathbf{u}^\top(1-\eta\mathbf{\Lambda})^S\mathbf{u} \tag{65}$$

where λ_1 is the largest eigenvalue of the one step operator $\mathbf{T}_1$, and is given by

$$\lambda_1 = \mathbf{u}^\top\mathbf{T}_1\mathbf{u} = \mathbf{u}^\top[(1-\eta\mathbf{\Lambda})^2 + \gamma\eta^2\mathbf{\Lambda}\mathbf{1}\mathbf{1}^\top\mathbf{\Lambda}]\mathbf{u} \tag{66}$$

for $\gamma = B^{-1} - D^{-1}$. If we define

$$a \equiv \mathbf{u}^\top\mathbf{\Lambda}\mathbf{u},\ b \equiv \mathbf{u}^\top\mathbf{\Lambda}^2\mathbf{u},\ c \equiv \mathbf{u}^\top\mathbf{\Lambda}\mathbf{1}\mathbf{1}^\top\mathbf{\Lambda}\mathbf{u} \tag{67}$$

then we have:

$$\lambda_1 = 1 - 2\eta a + \eta^2 b + \gamma\eta^2 c \tag{68}$$

Since $\rho = \lambda_1^S$ when $\nu = 1$ (SGD), at the optimal learning rate η^* we have

$$\left.\frac{d\lambda_1}{d\eta}\right|_{\eta=\eta^*} = 0 = -2a + 2\eta^* b + \gamma\eta^* c \tag{69}$$

Substituting, at the optimal learning rate we have:

$$\lambda_1 = 1 - \eta^* a \tag{70}$$

Substituting back into the formula for $d\rho/d\nu$, we have:

$$\frac{1}{2}\frac{d\rho}{d\nu} = (1-\eta^* a)^S - \mathbf{u}^\top(1-\eta^*\mathbf{\Lambda})^S\mathbf{u} = (1-\eta^*\mathbf{u}^\top\mathbf{\Lambda}\mathbf{u})^S - \mathbf{u}^\top(1-\eta^*\mathbf{\Lambda})^S\mathbf{u} \tag{71}$$

This expression is a function of moments of $\mathbf{\Lambda}$. Since S is even, $f(\lambda) = (1-\eta^*\lambda)^S$ is convex. Therefore we can apply Jensen's inequality

$$(1-\eta^*\mathbf{u}^\top\mathbf{\Lambda}\mathbf{u})^S < u^\top(1-\eta^*\mathbf{\Lambda})^S\mathbf{u} \tag{72}$$

The inequality is strict since $\mathbf{\Lambda}$ is non-degenerate by assumption. This gives us

$$\frac{d\rho}{d\nu} < 0 \tag{73}$$

Therefore, ρ can be reduced by increasing ν while leaving the product of inner and outer η fixed; there exist settings with which converge faster at infinite time than SGD with the optimal learning rate. □

A.4 LA-DiLoCo slows convergence rates with isotropic data and fixed total batch size

Here we prove the following theorem:

Theorem 3.2. *Consider the dynamics of Equation 16 for fixed ν and η. Then if $\mathbf{\Lambda} = \lambda_0\mathbf{I}$, $\lambda_0 > 0$, for fixed total batch size $B_{\text{tot}} \equiv BR$, the eigenvalues of the linear system strictly increase with R. Additionally, given a diagonal matrix $\mathbf{\Delta}$, given a pair $R_1 < R_2$, there exists an $\epsilon > 0$ such that the eiegenvalues for the system with R_2 workers has larger eigenvalues than the system for R_1 workers for $\mathbf{\Lambda} = \lambda_0\mathbf{I} + \epsilon\mathbf{\Delta}$.*

Proof. From Equation 16, the dynamics of $\mathbf{p}$ can be written as:

$$\mathbf{p}_{t+1} = \mathbf{F}(\eta, \nu, R, B_{\text{tot}}, \mathbf{\Lambda})\mathbf{p}_t \tag{74}$$

Our goal will be to show that $\mathbf{F}$ has eigenvalues that are strictly increasing in R when η, ν, and B_{tot} are fixed. We can explicitly parametrize $\mathbf{F}$ by $\mathbf{\Lambda}$ and write:

$$\mathbf{F}(\eta, \nu, R, B_{\text{tot}}, \mathbf{\Lambda}) = \mathbf{Q}(\eta, \nu, \mathbf{\Lambda}) + \sum_{s=1}^{S} a_s(\eta, \nu, R, B_{\text{tot}})\tilde{\mathbf{M}}_s(\eta, \mathbf{\Lambda}) \tag{75}$$

where $\mathbf{Q}$ and $\tilde{\mathbf{M}}_s$ are independent of R and B_{tot}.

First, consider the isotropic base case where $\epsilon = 0$ and $\mathbf{\Lambda} = \lambda_0 \mathbf{I}$. By definition, each $\tilde{\mathbf{M}}_s(\eta, \lambda_0 \mathbf{I})$ can be written as a sum of alternating products of matrices of the forms $\mathbf{\Lambda}$ and $\mathbf{\Lambda 1 1}^\top \mathbf{\Lambda}$. Since the base data is isotropic, $\mathbf{\Lambda}$ is proportional to the identity and commutes with $\mathbf{\Lambda 1 1}^\top \mathbf{\Lambda}$. Therefore the $\tilde{\mathbf{M}}_s(\eta, \lambda_0 \mathbf{I})$ are sums of products of commuting PSD matrices, which are PSD, so $\tilde{\mathbf{M}}_s(\eta, \lambda_0 \mathbf{I})$ are also PSD.

Recall that in this setting, the coefficients are:

$$a_s(\eta, \nu, R, B_{\text{tot}}) = \frac{R^{s-1}}{B_{\text{tot}}^s} \nu^2 \eta^{2s} \tag{76}$$

Since the power series starts at $s = 1$, for any $0 < R_1 < R_2$, we have $a_s(R_2) > a_s(R_1)$. This means that in the isotropic case, the difference between the operators is:

$$\mathbf{D}_0 \equiv \mathbf{F}(R_2, \lambda_0 \mathbf{I}) - \mathbf{F}(R_1, \lambda_0 \mathbf{I}) = \sum_{s=1}^{S} c_s \tilde{\mathbf{M}}_s(\eta, \lambda_0 \mathbf{I}) \tag{77}$$

for coefficients $c_s = a_s(R_2) - a_s(R_1) > 0$. Therefore, $\mathbf{D}_0$ is a strictly positive definite matrix on the relevant subspace, with some minimum eigenvalue be $\delta > 0$. By Weyl's Inequality, the rank-ordered eigenvalues of $\mathbf{F}(R_2, \lambda_0 \mathbf{I})$ are strictly greater than those of $\mathbf{F}(R_1, \lambda_0 \mathbf{I})$.

Now, consider the perturbed spectrum $\mathbf{\Lambda} = \lambda_0 \mathbf{I} + \epsilon \mathbf{\Delta}$. We can Taylor expand the matrix-valued function $\mathbf{F}$ around $\epsilon = 0$:

$$\mathbf{F}(R, \lambda_0 \mathbf{I} + \epsilon \mathbf{\Delta}) = \mathbf{F}(R, \lambda_0 \mathbf{I}) + \epsilon \mathbf{F}_1(R, \mathbf{\Delta}) + \mathcal{O}(\epsilon^2) \tag{78}$$

where $\mathbf{F}_1$ is the first-order derivative operator with respect to the spectrum.

Taking the difference between R_2 and R_1 for the perturbed system yields:

$$\mathbf{F}(R_2, \mathbf{\Lambda}) - \mathbf{F}(R_1, \mathbf{\Lambda}) = \mathbf{D}_0 + \mathbf{B} \tag{79}$$

where $\mathbf{B} = \epsilon[\mathbf{F}_1(R_2, \mathbf{\Delta}) - \mathbf{F}_1(R_1, \mathbf{\Delta})] + \mathcal{O}(\epsilon^2)$.

By Weyl's Inequality, the minimum eigenvalue of the perturbed difference is bounded by:

$$\lambda_{\min}(\mathbf{D}_0 + \mathbf{B}) \geq \lambda_{\min}(\mathbf{D}_0) + \lambda_{\min}(\mathbf{B}) \tag{80}$$

Since $\lambda_{\min}(\mathbf{D}_0) = \delta > 0$ and the magnitude of $\lambda_{\min}(\mathbf{B})$ scales linearly with ϵ, we can bound the spectrum as:

$$\lambda_{\min}(\mathbf{D}_0 + \mathbf{B}) \geq \delta - \epsilon \|\mathbf{F}_1(R_2, \mathbf{\Delta}) - \mathbf{F}_1(R_1, \mathbf{\Delta})\|_2 - \mathcal{O}(\epsilon^2) \tag{81}$$

As long as the anisotropy ϵ is sufficiently small—specifically $\epsilon \ll \frac{\delta}{\|\mathbf{F}_1(R_2, \mathbf{\Delta}) - \mathbf{F}_1(R_1, \mathbf{\Delta})\|_2}$—the lower bound remains strictly positive.

Thus, $\mathbf{F}(R_2, \mathbf{\Lambda}) - \mathbf{F}(R_1, \mathbf{\Lambda})$ remains positive definite, and the eigenvalues of$\mathbf{F}(R_2, \mathbf{\Lambda})$ are greater than those of $\mathbf{F}(R_1, \mathbf{\Lambda})$ for appropriately bounded ϵ □

A.5 $R^{-1/2}$ scaling heuristic

To further understand the $R^{-1/2}$ scaling heuristic, we trained the high-dimensional linear regression model using LA-DiLoCo with various R and η. We compared the scaling suggestion in Equation 20 to the baseline of not scaling η at all with R. The resulting curves show that using the same (ν, η) pairs across R causes discrepancies between the trajectories and larger R values diverge at relatively low values of η (Figure 2, top). In contrast, the heuristic $R^{-1/2}$ scaling (with counterscaling of ν) keeps learning trajectories approximately collapsed at small learning rates and keeps convergence at larger learning rates (Figure 2, bottom). In all cases, we found that the theoretical high-dimensional predictions (solid lines) well-matched the theoretical predictions (dashed lines) at the chosen D and B_{tot}

B Deterministic dynamics of SLA

In this section we derive the deterministic dynamics of SLA in the linear setting, and show how the dynamics are different from weight averaging and GPA.

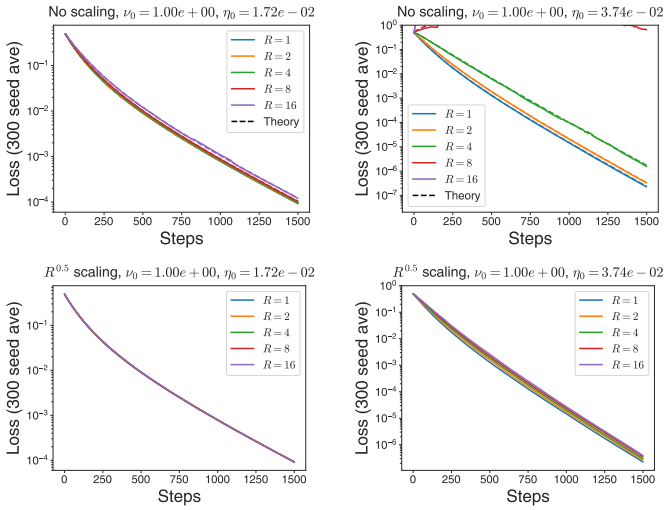

Figure 2: Averaged loss curves for LA diloco (solid lines) are well captured by theoretical model (dashed lines) across many settings with fixed total batch size (all panels, $D = 3200$, $B_{tot} = 64$, various learning rates). Keeping η and ν independent of number of workers R leads to similar dynamics at smaller learning rates (top left) but causes learning curves for larger R to diverge early at larger learning rates (top right). $R^{-1/2}$ scaling rule for η, with $\nu\eta$ fixed gives better correspondence across R for all learning rates (bottom), but dynamics still becomes non-universal at larger η (bottom right)

B.1 SLA with EMA momentum

The update rules for EMA style momentum in the inner and outer loop both are given by:

$$\boldsymbol{\psi}_t := \boldsymbol{\theta}_t \text{ (when } t//S = 0) \tag{82}$$

$$\left.\begin{aligned} \tilde{\mathbf{m}}_{t+1} &= \beta_{\text{in}}\tilde{\mathbf{m}}_t + (1-\beta_{\text{in}})\mathbf{g}(\boldsymbol{\psi}_t) \\ \boldsymbol{\psi}_{t+1} &= \boldsymbol{\psi}_t - \eta\tilde{\mathbf{m}}_{t+1} \end{aligned}\right\} \quad \text{for } S \text{ steps} \tag{83}$$

$$\mathbf{m}_{t+S} = \beta_{\text{out}}\mathbf{m}_t + (1-\beta_{\text{out}})(\boldsymbol{\psi}_{t+S} - \boldsymbol{\psi}_t) \tag{84}$$

$$\boldsymbol{\theta}_{t+T} = \boldsymbol{\theta}_t - \nu\mathbf{m}_{t+S} \tag{85}$$

Note that the inner momentum $\tilde{\mathbf{m}}$ is not typically synced across cycles.

B.2 SLA with Nesterov momentum

The update rules for Nesterov style momentum in the inner and outer loop both are given by:

$$\boldsymbol{\psi}_t := \boldsymbol{\theta}_t \text{ (when } t//S = 0) \tag{86}$$

$$\left.\begin{aligned} \tilde{\mathbf{m}}_{t+1} &= \beta_{\text{in}}\tilde{\mathbf{m}}_t + (1-\beta_{\text{in}})\mathbf{g}(\boldsymbol{\psi}_t) \\ \boldsymbol{\psi}_{t+1} &= \boldsymbol{\psi}_t - \eta[(1-\beta_{\text{in}})\mathbf{g}(\boldsymbol{\psi}_t) + \beta_{\text{in}}\tilde{\mathbf{m}}_{t+1}] \end{aligned}\right\} \quad \text{for } S \text{ steps} \tag{87}$$

$$\mathbf{m}_{t+S} = \beta_{\text{out}}\mathbf{m}_t + (1-\beta_{\text{out}})(\boldsymbol{\psi}_{t+S} - \boldsymbol{\psi}_t) \tag{88}$$

$$\boldsymbol{\theta}_{t+T} = \boldsymbol{\theta}_t - \nu[(1-\beta_{\text{out}})[\boldsymbol{\psi}_{t+S} - \boldsymbol{\psi}_t] + \beta_{\text{out}}\mathbf{m}_{t+S}] \tag{89}$$

B.3 Full linear dynamics

In the deterministic setting of the linear regression problem (full batch size), we can write the SLA dynamics in function space as:

$$\begin{pmatrix} \tilde{\mathbf{z}}_{t+S} \\ \tilde{\mathbf{k}}_{t+S} \\ \mathbf{z}_{t+S} \\ \mathbf{k}_{t+S} \end{pmatrix} = \mathbf{T}_{\text{sync}}\mathbf{T}_{\text{out}}\mathbf{T}_{\text{in}} \begin{pmatrix} \tilde{\mathbf{z}}_t \\ \tilde{\mathbf{k}}_t \\ \mathbf{z}_t \\ \mathbf{k}_t \end{pmatrix} \tag{90}$$

Here $\mathbf{T}_{\text{in}}$ and $\mathbf{T}_{\text{out}}$ represent the inner and outer optimizers respectively, and $\mathbf{T}_{\text{sync}}$ represents the "syncing" operation that prepares $\mathbf{z}$ and $\mathbf{k}$ for the next cycle.

The standard definition of SLA which only syncs $\mathbf{z}$ has the syncing operator:

$$\mathbf{T}_{\text{sync}} = \begin{pmatrix} 0 & 0 & \mathbf{I} & 0 \\ 0 & \mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{pmatrix} \tag{91}$$

while the 2-D simplification where $\mathbf{k}_t$ is initialized to 0 every cycle has the operator:

$$\mathbf{T}_{\text{sync}} = \begin{pmatrix} 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{pmatrix} \tag{92}$$

For EMA style momentum, the inner and outer optimizer are given by the operators

$$\mathbf{T}_{\text{in}} = \begin{pmatrix} \mathbf{I} - \eta(1-\beta_{\text{in}})\frac{1}{D}\hat{\Theta} & -\eta\beta_{\text{in}}\mathbf{I} & 0 & 0 \\ (1-\beta_{\text{in}})\frac{1}{D}\hat{\Theta} & \beta_{\text{in}}\mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{pmatrix}^S, \mathbf{T}_{\text{out}} = \begin{pmatrix} \mathbf{I} & 0 & 0 & 0 \\ 0 & \mathbf{I} & 0 & 0 \\ \nu(1-\beta_{\text{out}})\mathbf{I} & 0 & \mathbf{I} - \nu(1-\beta_{\text{out}})\mathbf{I} & -\nu\beta_{\text{out}}\mathbf{I} \\ -(1-\beta_{\text{out}})\mathbf{I} & 0 & (1-\beta_{\text{out}})\mathbf{I} & \beta_{\text{out}}\mathbf{I} \end{pmatrix} \tag{93}$$

while for Nesterov (in EMA units) we have

$$\mathbf{T}_{\text{in}} = \begin{pmatrix} \mathbf{I} - \eta(1-\beta_{\text{in}}^2)\frac{1}{D}\hat{\Theta} & -\eta\beta_{\text{in}}^2\mathbf{I} & 0 & 0 \\ (1-\beta_{\text{in}})\frac{1}{D}\hat{\Theta} & \beta_{\text{in}}\mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{pmatrix}^S, \mathbf{T}_{\text{out}} = \begin{pmatrix} \mathbf{I} & 0 & 0 & 0 \\ 0 & \mathbf{I} & 0 & 0 \\ \nu(1-\beta_{\text{out}}^2)\mathbf{I} & 0 & \mathbf{I} - \nu(1-\beta_{\text{out}}^2)\mathbf{I} & -\nu\beta_{\text{out}}^2\mathbf{I} \\ -(1-\beta_{\text{out}})\mathbf{I} & 0 & (1-\beta_{\text{out}})\mathbf{I} & \beta_{\text{out}}\mathbf{I} \end{pmatrix} \tag{94}$$

We can once again shift to the eigenbasis of $\frac{1}{D}\hat{\Theta}$. This reduces the dynamics to a 4-dimensional system for each eigenmode. For EMA momentum, for example, we have:

$$\mathbf{T}_{\text{in}} = \begin{pmatrix} 1-(1-\beta_{\text{in}})\eta\lambda & -\eta\beta_{\text{in}} & 0 & 0 \\ (1-\beta_{\text{in}})\lambda & \beta_{\text{in}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}^S, \mathbf{T}_{\text{out}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \nu(1-\beta_{\text{out}}) & 0 & 1-\nu(1-\beta_{\text{out}}) & -\nu\beta_{\text{out}} \\ -(1-\beta_{\text{out}}) & 0 & (1-\beta_{\text{out}}) & \beta_{\text{out}} \end{pmatrix} \tag{95}$$

The inner and outer optimizers have a block diagonal structure:

$$\mathbf{T}_{\text{in}} = \begin{pmatrix} \mathbf{F} & 0 \\ 0 & \mathbf{I} \end{pmatrix}, \mathbf{T}_{\text{out}} = \begin{pmatrix} \mathbf{I} & 0 \\ \mathbf{G} & \mathbf{H} \end{pmatrix}, \mathbf{T}_{\text{out}}\mathbf{T}_{\text{in}} = \begin{pmatrix} \mathbf{F} & 0 \\ \mathbf{GF} & \mathbf{H} \end{pmatrix} \tag{96}$$

which for EMA momentum gives

$$\mathbf{F} = \begin{pmatrix} 1-(1-\beta_{\text{in}})\eta\lambda & -\eta\beta_{\text{in}} \\ (1-\beta_{\text{in}})\lambda & \beta_{\text{in}} \end{pmatrix}^S, \mathbf{G} = \begin{pmatrix} \nu(1-\beta_{\text{out}}) & 0 \\ -(1-\beta_{\text{out}}) & 0 \end{pmatrix}, \mathbf{H} = \begin{pmatrix} 1-\nu(1-\beta_{\text{out}}) & -\nu\beta_{\text{out}} \\ (1-\beta_{\text{out}}) & \beta_{\text{out}} \end{pmatrix} \tag{97}$$

The $\mathbf{F}$ matrix is of particular interest, as it defines the "effective spectrum" seen by the outer optimizer in the linear regression setting.

B.4 Momentum with weight averaging

Our analysis of SLA in the deterministic linear regression setting begs the immediate question: are there algorithms which don't have the two-phase structure, but have similar behavior after S steps? While this question is still generally open, we can answer in the negative for the most basic idea: pairing the dynamics of SGD momentum with an exponential moving average of the parameters.

Consider the deterministic setting of the linear regression problem. In the diagonalized basis, given an eval point $\hat{z}$ which is an EMA of z with coefficient β_{ema} (where $\beta_{\text{ema}} = 0$ corresponds to normal

evaluation at the latest parameter value), the update rule for the 3-dimensional system becomes:

$$\begin{pmatrix} z_{t+1} \\ k_{t+1} \\ \hat{z}_{t+1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1-\beta_{\text{ema}} & 0 & \beta_{\text{ema}} \end{pmatrix} \begin{pmatrix} 1-\eta(1-\beta_1)\lambda & -\eta\beta_1 & 0 \\ (1-\beta_1)\lambda & \beta_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} z_t \\ k_t \\ \hat{z}_t \end{pmatrix} \tag{98}$$

Evaluating we have:

$$\begin{pmatrix} z_{t+1} \\ k_{t+1} \\ \hat{z}_{t+1} \end{pmatrix} = \begin{pmatrix} 1-\eta(1-\beta_1)\lambda & -\eta\beta_1 & 0 \\ (1-\beta_1)\lambda & \beta_1 & 0 \\ (1-\beta_{\text{ema}})[1-\eta(1-\beta_1)] & -(1-\beta_{\text{ema}})\eta\beta_1 & \beta_{\text{ema}} \end{pmatrix} \begin{pmatrix} z_t \\ k_t \\ \hat{z}_t \end{pmatrix} \tag{99}$$

Since $\hat{z}$ depends on z and k, but does not causally affect them, two of the eigenvalues of this system are the eigenvalues of the original system. This can be seen directly by the fact that there are two left eigenvectors of the transition matrix which lie in the z/k subspace alone. These eigenvectors have the same eigenvalues as the original problem. The remaining eigenvalue has eigenvalue β_{ema}, since the trace of the new system is the trace of the old system plus β_{ema}.

This means that EMA of the evaluation point does not appreciably change the eigenstructure of the problem in the same way that SLA does via the outer momentum applied to the "effective" problem.

B.5 GPA in linear regression

The GPA algorithm in Defazio et al. [16] (and indeed, the paper title) is partially inspired by coming up with and alternative of single-worker DiLoCo which maintains its benefits, but doesn't require the very discrete structure of the two-phase optimization process. The paper suggests that GPA can be thought of as a continuous relaxation of DiLoCo. Here we show that in the deterministic setting, SLA and GPA have quite different dynamics.

The original GPA formulation can be defined by the following equations:

$$y_t = \mu_y x_t + (1-\mu_y) z_t \tag{100}$$

$$z_{t+1} = z_t + \eta_t d(y_t) \tag{101}$$

$$x_{t+1} = \mu_x x_t + (1-\mu_x) z_{t+1} \tag{102}$$

Here $d(y_t)$ represents the "update" of a base optimizer - for example, the negative gradient in the case of SGD. The variable x_t is used for evaluation.

To understand the dynamics, we first re-write everything so that we have equations for y_{t+1}, z_{t+1}, and x_{t+1} as functions of y_t, z_t, and x_t only. This gives us:

$$y_{t+1} = \mu_x x_{t+1} + (1-\mu_y) z_{t+1} = \mu_y\mu_x x_t + \mu_y(1-\mu_x) z_{t+1} + (1-\mu_y) z_{t+1} \tag{103}$$

$$y_{t+1} = \mu_x\mu_y x_t + (1-\mu_x\mu_y)[z_t + \eta_t d(y_t)] \tag{104}$$

The corresponding x and z equations are simple to compute, and we end up with:

$$\begin{pmatrix} y_{t+1} \\ z_{t+1} \\ x_{t+1} \end{pmatrix} = \begin{pmatrix} 0 & 1-\mu_x\mu_y & \mu_x\mu_y \\ 0 & 1 & 0 \\ 0 & 1-\mu_x & \mu_x \end{pmatrix} \begin{pmatrix} y_t \\ z_t \\ x_t \end{pmatrix} + \eta_t d(y_t) \begin{pmatrix} 1-\mu_x\mu_y \\ 1 \\ 1-\mu_x \end{pmatrix} \tag{105}$$

In the deterministic setting of the linear regression model, the optimization problem decomposes over the eigenmodes and we have $d(y_t) = -\lambda y_t$. This gives us:

$$\begin{pmatrix} y_{t+1} \\ z_{t+1} \\ x_{t+1} \end{pmatrix} = \begin{pmatrix} -\eta_t\lambda(1-\mu_x\mu_y) & 1-\mu_x\mu_y & \mu_x\mu_y \\ -\eta_t\lambda & 1 & 0 \\ -(1-\mu_x)\eta_t\lambda & 1-\mu_x & \mu_x \end{pmatrix} \begin{pmatrix} y_t \\ z_t \\ x_t \end{pmatrix} \tag{106}$$

This has very different properties from any of the SLA matrices we analyzed in Section 4, particularly in how λ, η, and the other hyperparameters fold in to the overall spectrum. There is no clear "effective problem" dominating the eigenstructure like in SLA.

Another way we can see this is by changing variables. We define:

$$\hat{k}_t \equiv \frac{1}{\mu_x\mu_y}\left(-y_t + (1-\mu_x\mu_y) z_t + \mu_x\mu_y x_t\right) \tag{107}$$

Then the dynamics of y_t and $\hat{k}_t$ close and we have:

$$\hat{k}_{t+1} = \mu_x \hat{k}_t + (1 - \mu_x)\eta_t d(y_t) \tag{108}$$

$$y_{t+1} = y_t + \mu_y \hat{k}_{t+1} + (1 - \mu_y)\eta_t d(y_t) \tag{109}$$

We can see that this is very similar to Nesterov momentum, with an additional hyperparameter that decouples the contribution of the momentum and gradient to the parameter update. This also will have very different eigenstructure from SLA with Nesterov outer; in particular, it misses the interaction between the outer momentum and highly convergent eigenmodes seen in SLA. This suggests that the two algorithms are indeed qualitatively different.

C Additional Experiments

C.1 Single cycle advantage of LA vs. SGD

In order to study the finite-time advantage of LA, we numerically iterated Equation 12 for a variety of spectra $\mathbf{\Lambda}$, batch fraction B/D, and sync cycle period S. For each such setting, we ran S steps of dynamics over a grid of η and ν values to calculate the change in loss after one cycle. As expected, we generally found that the optimal value of η for a particular ν decreased in ν (Figure 3, left).

We found that there were many settings for which the optimal (ν, η) pair involved $\nu > 1$, as it does at infinite time. In general, variance in the spectrum causes the optimal ν to shift away from 1. For example, consider a "spiked spectrum" with two eigenvalues λ_0 (99% of eigenmodes) and $20\lambda_0$ (1% of eigenmodes). Plotting the loss after one cycle as a function of ν (optimizing η for each ν), we see that the optimal ν is greater than 1 (Figure 3, middle).

For a single cycle, we also found some settings where the optimal $\nu < 1$, such as certain power law spectra (Figure 3, right, $\alpha = -1.5$). We note that the total benefit of LA amounts to a $10 - 20\%$ improvement in the loss decrease per cycle in the settings we present here. We conjecture that LA can provide modest but bounded gains in more general settings. The largest benefits tend to occur for smaller batch fraction B/D and longer cycles S.

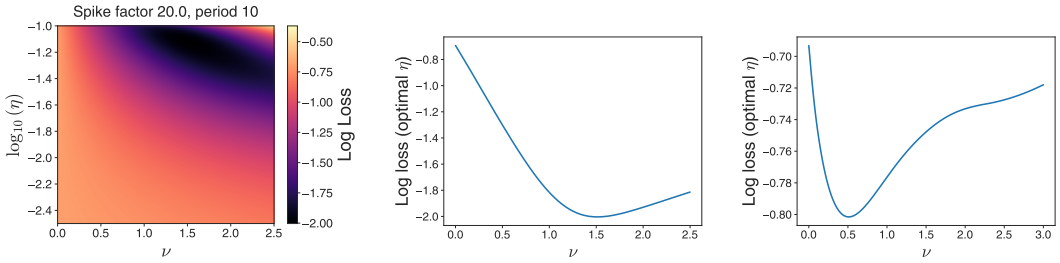

Figure 3: Loss after one cycle of $S = 10$ steps as a function of η and ν for spiked spectrum (left). Batch fraction $B/D = 0.2$, and spectrum has two eigenvalues λ_0 (99% of eigenmodes) and $20\lambda_0$ (1% of eigenmodes). Optimal $\eta^*(\nu)$ decreases in ν. Loss at $\eta^*(\nu)$ is minimized for $\nu > 1$ (middle). Power law spectrum with exponent $\alpha = -1.5$ has optimal $\nu < 1$ (right, $B/D = 0.005$).

C.2 $R^{-1/2}$ scaling in a non-linear setting

Our theoretical results, including the predicted $R^{-1/2}$ scaling rule in the fixed total batch size setting, were derived for linear models. Some version of the results holds in the NTK limit where learning is linearized. There is an open question as to what extent the results hold in the non-linear setting.

As a first step to answer this question, we conducted experiments training a two-layer MLP on MNIST using LA-Diloco (batch size 256, hidden dimension 128, cycle length $S = 10$). We first trained with $R = 1$ and optimized the inner learning rate. We then varied R and set the learning rates in two ways: first, by leaving both learning rates fixed, and second by using the proposed $R^{-1/2}$ scaling rule for the inner optimizer (and counter-scaling the outer learning rate ν).

The results can be seen in figure 4. We see that there is no collapse in learning curves when the inner and outer learning rates are kept fixed across R (left plot). When we do scale the inner learning rate

(right), we have very good agreement with two exceptions: the early time dynamics at $R = 2$ does not collapse (though the later time dynamics do), and for large R the training becomes unstable (since $\nu > 1$ using the scaling rule). We suspect that the early-time differences are due to non-linear effects.

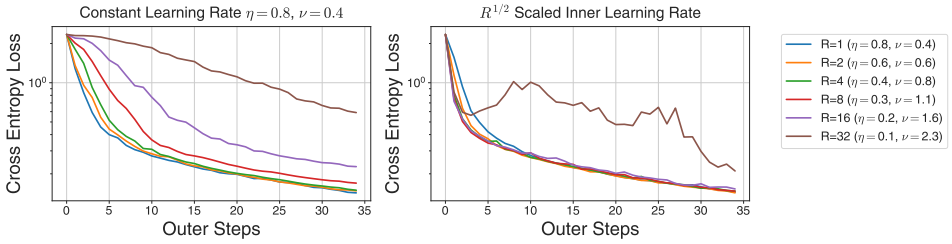

Figure 4: Loss curves for a two-layer MLP on MNIST, trained using LA-Diloco with various R. When the inner and outer learning rates are kept fixed as a function of R (left), we see no collapse. When we use the proposed scaling rule (right), we see good agreement except at early times or for higher values of R.

The original motivation for the scaling rule was based on an analysis of the optimization noise. In the non-linear setting, we can analyze the noise via the per-worker dispersion $\mathrm{E}[||\psi_{t,k} - \bar{\psi}_t||^2]$, which measures the deviation between the workers at the end of the cycle. Here we take an empirical average over the workers, but also average over the randomness in the optimization trajectory.

We measured this quantity in the same MNIST setting, by replaying a single cycle 100 times at various training checkpoints (re-randomizing the data sampling each time). When the learning rate was not scaled with $R^{-1/2}$, this deviation grows roughly linearly with R (Figure 5, left). With the scaling rule, the deviation still grows with R but at a much slower rate (Figure 5, right). This suggests that the scaling rule does indeed control the between-worker deviation, but that the scaling does not fully normalize the noise — which is true even in the linear theory due to the higher order terms.

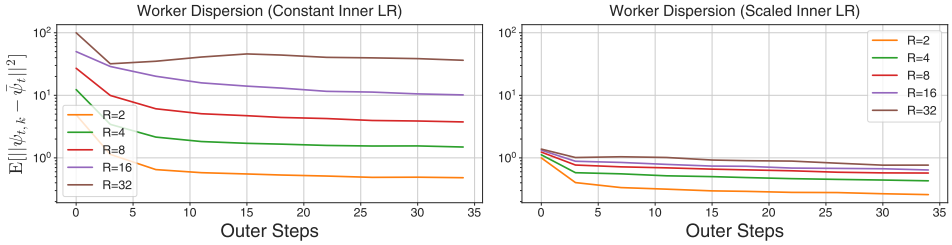

Figure 5: Per-worker dispersion $\mathrm{E}[||\psi_{t,k} - \theta_t||^2]$ for a two-layer MLP on MNIST, trained using LA-Diloco with various R. When the inner and outer learning rates are kept fixed as a function of R, we see a large increase in dispersion. When we use the proposed scaling rule, the dispersions are similar across R, reflecting a more similar noise process.

These results suggest that the scaling rule may be beneficial in the non-linear setting. More work is needed to establish this as a good strategy in practice; however both the scaling itself and the underlying noise analysis are worth further study.

C.3 SLA momentum variations on deep learning workloads

In order to provide additional evidence that outer loop Nesterov is often favored over outer loop heavy ball momentum, we tuned SLA with different momentum variations on the FineWebEDU and Imagenet-ViT workloads from AlgoPerf [22]. We followed the tuning methodology in Kasimbeg et al. [23], using a 50-point Halton search for each workload-momentum variant pair. The inner optimizer was AdamW, and the Halton points were chosen to cover the same hyperparameters for each workload. The best runs for each workload-algorithm pair can be found in Table 1.

We found that the two outer optimizers behaved similarly, but there was a slight advantage to Nesterov in the outer optimizer; this advantage was small but consistent across the best performing

Workload	Algorithm	Validation Metric	$\eta\nu$	ν	β_{out}	β_{in}
fineweb_edu_lm	Nesterov	Perplexity: 28.5	0.000785	0.866	0.832	0.804
fineweb_edu_lm	Heavy Ball	Perplexity: 29.3	0.000785	0.866	0.832	0.804
imagenet_vit_s16	Nesterov	Error Rate: 0.265	0.00189	1.29	0.817	0.930
imagenet_vit_s16	Heavy Ball	Error Rate: 0.269	0.00189	1.29	0.817	0.930

Table 1: Validation metrics and key hyperparameters for best training runs with SLA on the AlgoPerf benchmark (50 point Halton search). Nesterov momentum appears to perform better than heavy ball momentum in this setting.

configurations. The best hyperparameters in our search are equal for both momentum methods because the most important parameter is the total learning rate $\nu\eta$, followed by the outer learning rate ν.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We checked the abstract and introduction to make sure they accurately reflect the rest of the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed in detail the theoretical settings to which our work applies.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes],

Justification: We have one theorem, formally proved; the rest of the paper uses simple linear algebraic calculations and numerical simulations.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All numerical studies are of the simple models presented in the paper, and do not require any special hardware configurations to run.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We have not included code for our linear regression training experiments; however given their simplicity they should be reproducible in any framework including pure numpy.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [N/A]

Justification: Study is of theoretical models only.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [N/A]

Justification: Any numerical results are related to pure theoretical models.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [N/A]

   Justification: Numerical results are simple, running on CPU in a few seconds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Authors read the code of ethics and found no violations.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: Work has no impact beyond whatever impact can be gained from improving understanding of training algorithms for ML systems.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No data or models were released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

    Justification: No existing assets used.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new datasets or models released.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: No LLM useage in the paper.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.