Supplementary Material: Surprising Instabilities in Training Deep Networks and a Theoretical Analysis

Yuxin Sun[1] **Dong Lao**[2] **Ganesh Sundaramoorthi**[3] **Anthony Yezzi**[1]
[1]Georgia Institute of Technology, [2]UCLA, [3]Raytheon Technologies
{syuxin3,ayezzi}@gatech.edu, lao@cs.ucla.edu, ganesh.sundaramoorthi@rtx.com

A Extended Discussion

What is the relation between Section 3 and Section 4? Section 3 presents theory on how finite precision errors can be attenuated or amplified based on satisfying or not the CFL condition (condition on the learning rate) through PDE tools. Note the continuum limit of SGD as the learning result goes to zero is a PDE. As such and for ease of illustration, a simple PDE is examined to illustrate the key ideas of the PDE framework. Section 4 presents that such amplification of finite precision errors is present in deep network optimization, and shows that this likely results from not satisfying the CFL condition (learning rate is too high) - last experiment of the section.

What is the relation between Section 4 and Section 5? Section 5 presents a detailed theoretical study on how the amplification / attenuation of finite precision errors from Section 4 can arise by applying methodology from Section 3. The study reduces down restrained instabilities to its most basic manifestation in simple one-layer CNN trained on just a single image, and shows how they arise in this CNN. In particular, when the learning rate is chosen in a specific range, restrained instabilites (oscillating between stable and unstable regimes) arise and finite precision errors accumulate. For small enough learning rates, the instabilites disappear and finite errors are attenuated. This matches the behavior of the large networks in Section 4, suggesting the theory applies.

What is the significance of this work? We have taken a step in developing theory for principally choosing and designing learning rate schemes from the perspective of numerical stability, showing in particular the relevance of numerical PDE theory, which has not been explored before. The study also suggests the exploration of adaptive optimization schemes that are constructed to ensure numerical stability. Whether eliminating restrained instabilities in training results in better performance (e.g., generalization, training time, robustness, reduced variance, etc) is an open question for future exploration. It may be well the case that such stabilities are helpful in training, but this requires further investigation and our theory provides a starting point for answering that question. The work also discovered the phenomena of restrained instabilites and error amplification, and significant test accuracy variance resulting from that phenomena. Whether eliminating the instability reduces stochastic test variance from batch selection / initialization is an open question. Moreover, given that there are several papers in literature where performance reported is on the order of the variance reported in this paper, our result may challenge the significance of those results, and further investigation is needed.

B Empirical Validation: All Seeds in Experiments of Section 4 are Fixed

We verify that all random seeds in SGD and the deep learning framework (e.g., initialization, batch selection, CUDA seeds, etc) have been fixed in the experiments in Section 4.2. This would indicate that the only sources of variance across trials is due to the introduced floating point perturbation. To verify that, we run experiments for multiple trials when $k = 1$. Test accuracy over epochs from each

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

trial is shown in the following Table 1. This confirms that the dynamic trajectories of weights strictly coincide, indicating that all seeds have been fixed.

Epoch	0	20	40	60	80	100	120	140	160	180	200
Trial 1	33.84	79.22	90.90	91.19	93.07	93.33	93.38	93.47	93.44	93.39	93.47
Trial 2	33.84	79.22	90.90	91.19	93.07	93.33	93.38	93.47	93.44	93.39	93.47
Trial 3	33.84	79.22	90.90	91.19	93.07	93.33	93.38	93.47	93.44	93.39	93.47

Table 1: Accuracy at different epochs over different trials. This indicates all seeds inducing randomness have been fixed.

To further strengthen this point, we provide additional results on $k' = k \times 2^n$ in Table 2. Since base 2 (binary) is used in floating number, multiplying or dividing by the power of 2 will not introduce floating point arithmetic perturbation, therefore for any integer k, results on k' and k will have the same trajectories. This is another piece of evidence showing that the instability comes ONLY from floating point error. We also provided code in the supplementary (main4.py) for verification.

	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	k=10	k=11	k=12
Trial 1	93.47	93.47	93.29	93.47	93.40	93.29	93.72	94.47	93.62	93.40	93.79	93.29
Trial 2	93.47	93.47	93.29	93.47	93.40	93.29	93.72	93.47	93.62	93.40	93.79	93.29
Trial 3	93.47	93.47	93.29	93.47	93.40	93.29	93.72	93.47	93.62	93.40	93.79	93.29

Table 2: Final Test Accuracy for different k. Notice k that differ by a factor of a power of two have exactly the same accuracy. This provides further evidence that all seeds have been fixed.

C Supplementary Experimental Results for Section 4

We provide supplementary experimental results for Section 4. The bold titles correspond to the section in Section 4.2 for which the supplementary experiment belongs.

Section 4.2, Perturbed SGD with Swish: Table 3 shows detailed results of the variability due to the floating point perturbation in comparison to batch selection with Swish activation for Section 4.2 **Perturbed SGD with Swish**. As noted in Section 4.2, the Swish activation also results in significant variance due to floating point arithmetic.

Seed	1	2	3	4	5	6	STD
$k = 1$	93.81	93.79	93.62	93.83	93.72	93.59	0.09
$k = 3$	93.52	93.94	93.29	93.59	93.73	93.60	0.20
$k = 5$	93.39	93.89	93.89	93.60	93.71	93.62	0.17
$k = 7$	93.35	93.59	93.52	93.48	93.37	93.43	0.08
$k = 9$	93.49	93.50	93.70	93.79	93.55	93.71	0.12
$k = 11$	93.84	93.71	93.72	93.91	93.71	93.56	0.11
STD	0.19	0.15	0.18	0.14	0.13	0.082	

Table 3: Test accuracy variance over different seeds (batch selections) and different floating point perturbations (rows) for Resnet56 using Swish activation that is trained on CIFAR-10. STD is the standard derivation. k indicates different version of perturbed SGD that each introduces a perturbation at the last significant bit of the gradient.

Section 4.2, Other Architectures/Datasets: Table 4 verifies that the variability due to the floating point perturbation generally exists across different network architectures and datasets. Here we repeated the same experiment for a different network (VGG16) and a different dataset (Fashion-MNIST). The Swish activation is used. Six different floating point noises are used and six different seeds are used as in Table 3. Average standard deviations for test accuracy variation for both the floating point perturbations and batch selection. The test accuracy variance for floating point noise is

greater or similar to the stochastic variation due to batch selection. Note the average test accuracy and the average standard deviation over floating point perturbations over k are reported. The variance of SGD due to batch selection is also reported.

	Floating Pt STD	SGD STD
Vgg16BN FMNIST	94.81 ± 0.09	0.10
Vgg16BN CIFAR-10	93.76 ± 0.13	0.09
ResNet56 FMNIST	94.81 ± 0.12	0.10
ResNet56 CIFAR-10	93.72 ± 0.15	0.09

Table 4: A summary of results of variation of test accuracy for different floating point error in comparison to stochastic noise for different architectures and datasets.

Section 4.2, Evidence of Divergence Due to Instability: We verify that the optimization for the stable learning rate (3.125e-6) does not cause fluctuation for various k in perturbed SGD due to being stuck at a local minima. To this end, we show the test accuracy plot for $k = 1, 3$ in Figure 1 corresponding to the experiment in the right side of Figure 2 in the main paper. Note that the accuracy is increasing, indicating the optimization is not stuck at a local minima.

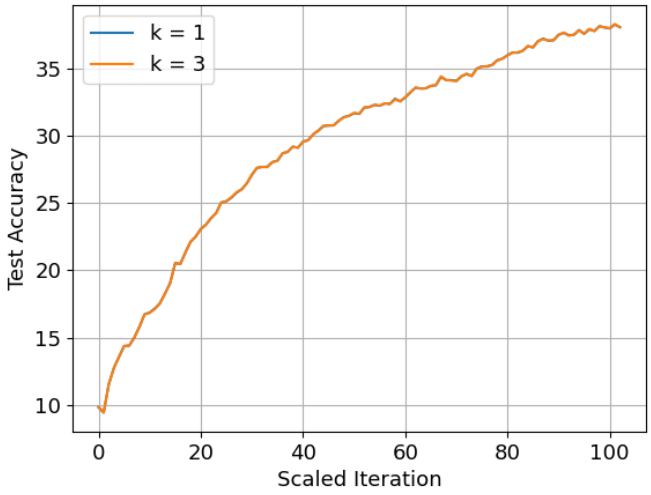

Figure 1: Test accuracy plot for the stable learning rate 3.125e-6. Shows that the optimization is not stuck in a local minima, and the error amplification is eliminated because of the choice of learning rate that satisfies the CFL condition. Note the curves overlap.

D Derivation for Gradient Descent PDE and Details of Stability Analysis

D.1 Derivation of Gradient PDE (Theorem 5.1)

Let $I : \mathbb{R}^2 \to \mathbb{R}$ be an input image to the network, $K : \mathbb{R}^2 \to \mathbb{R}$ be a convolution kernel (we assume I and K to be of finite support), $r : \mathbb{R} \to \mathbb{R}$ be the Swish activation, and $s : \mathbb{R} \to \mathbb{R}$ is the sigmoid function. We consider the following CNN, which is the simplest version of VGG:

$$f(I) = s\left[\int_{\mathbb{R}^2} r(K * I)(x)\,\mathrm{d}x\right]. \tag{1}$$

We consider the following regularized cross-entropy loss:

$$L(K) = \ell(y, \hat{y}) + \frac{1}{2}\alpha\|K\|^2_{\mathbb{L}^2}, \quad \text{where} \quad \hat{y} = f(I) \tag{2}$$

where ℓ denotes the cross-entropy, the second term denotes the regularization ($\mathbb{L}^2$ norm squared of the kernel, K), and $\alpha > 0$ and the second term is weight regularization. Define

$$g(I) = \int_{\mathbb{R}^2} r(K * I)(x) \, \mathrm{d}x. \tag{3}$$

Computing the variation of $g(I)$ with respect to δK yields

$$\begin{aligned}
\delta g(I) \cdot \delta K &= \int r'(K * I)(x) \cdot \delta K * I(x) \, \mathrm{d}x && (4)\\
&= \int_x \int_y r'(K * I)(x) I(x + z) \delta K(z) dz dx && (5)\\
&= \int_z \delta K(z) \int_x r'(K * I)(x) I(x + z) \, \mathrm{d}x \, \mathrm{d}z && (6)\\
&= \int_z \delta K(z) r'(K * I) * I(z) \, \mathrm{d}z. && (7)
\end{aligned}$$

Therefore,

$$\nabla_K g(I) = r'(K * I) * I. \tag{8}$$

Therefore,

$$\nabla_K L(K) = \frac{\partial \ell}{\partial \hat{y}}(y, \hat{y}) s'(g(I)) r'(K * I) * I + \alpha K. \tag{9}$$

Note $s(x) = \frac{1}{1+e^{-(x-b)}}$ is the sigmoid (with bias b), and thus $s'(x) = \frac{e^{-(x-b)}}{[1+e^{-(x-b)}]^2} = s(x)[1 - s(x)]$. Therefore, $s'(g(I)) = f(I)[1 - f(I)] = \hat{y}(1 - \hat{y})$. Therefore,

$$\nabla_K L(K) = \hat{y}(1 - \hat{y}) \frac{\partial \ell}{\partial \hat{y}}(y, \hat{y}) r'(K * I) * I + \alpha K. \tag{10}$$

For cross-entropy, we have $\frac{\partial \ell}{\partial \hat{y}} = -\frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$ and thus $\hat{y}(1 - \hat{y}) \frac{\partial \ell}{\partial \hat{y}} = \hat{y} - y$. Therefore,

$$\nabla_K L(K) = (\hat{y} - y) r'(K * I) * I + \alpha K \tag{11}$$

where $\hat{y} = f(I)$. The gradient descent PDE with respect to the loss (2) is

$$\partial_t K = -\nabla_K L(K) = -(\hat{y} - y) r'(K * I) * I - \alpha K, \tag{12}$$

where r' denotes the derivative of the activation, and t parametrizes the evolution. If we maintain the constraint that K is finite support so that K is zero outside $[-w/2, w/2]^2$ then the constrained gradient descent is

$$\partial_t K = [-(\hat{y} - y) r'(K * I) * I - \alpha K] \cdot W, \tag{13}$$

where W is a windowing function (1 inside $[-w/2, w/2]^2$ and zero outside).

D.2 Derivation of Linearization of the Gradient PDE (Theorem 5.2)

We assume that r is the Swish function and $\beta > 0$ is the parameter of the Swish activation:

$$r(x) = \frac{x}{1 + e^{-\beta x}}. \tag{14}$$

We can show that the first derivative and the second derivative of Siwsh function are:

$$r'(x) = \frac{1 + (1 + \beta x) e^{-\beta x}}{(1 + e^{-\beta x})^2} \tag{15}$$

$$r''(x) = \frac{\beta e^{-\beta x} [\beta x (1 - e^{-\beta x}) + 2(1 + e^{-\beta x})]}{(1 + e^{-\beta x})^3}. \tag{16}$$

We assume $K * I$ is near zero, so that we could express $r'(x)$ using Taylor expansion:

$$r'(x) \approx r'(0) + r''(0) x \tag{17}$$

where

$$r'(0) = \frac{1}{2}, \quad r''(0) = \frac{1}{2}\beta. \tag{18}$$

In this case,

$$r'(K * I) * I \approx \frac{1}{2}(\bar{I} + \beta[(K * I) * I])W \tag{19}$$

where $\bar{I}$ is the input sum. Assuming $\hat{y} - y =: a$ is approximately constant, the PDE becomes

$$\partial_t K = [-\frac{a}{2}(\bar{I} + \beta[(K * I) * I]) - \alpha K]W \tag{20}$$

D.3 Derivation of DFT and CFL Conditions for Lineaized PDE (Theorem 5.3, 5.4)

Consider the forward Euler scheme of (20):

$$K^{n+1} - K^n = \left[-\frac{a}{2}\Delta t(\bar{I} + \beta[(K^n * I) * I]) - \Delta t \alpha K^n\right] W. \tag{21}$$

where n denotes the iteration number, and Δt denotes the step size (learning rate in SGD).

The DFT of the discretization (21) of the linearized PDE is

$$\hat{K}^{n+1} - \hat{K}^n = -\frac{a}{2}\Delta t(\bar{I} + \beta \hat{K}^n |\hat{I}|^2) * \operatorname{sinc}\left(\frac{w}{2}\cdot\right) - \Delta t \alpha \hat{K}^n, \tag{22}$$

where $\hat{K}^n$ denotes the DFT of K^n, and sinc denotes the sinc function. In the case that $w \to \infty$ (the window support becomes large), the DFT of K can be written in terms of the amplifier A as

$$\hat{K}^{n+1}(\omega) = A(\omega)\hat{K}^n(\omega) - \frac{a}{2}\Delta t \bar{I}, \quad \text{where} \quad A(\omega) = 1 - \Delta t\left(\alpha + \frac{1}{2}a\beta|\hat{I}(\omega)|^2\right). \tag{23}$$

To be stable, $|A| < 1$. Provided that α is large enough (when $a < 0$), stability can be achieved with the condition:

$$\Delta t < \frac{2}{\alpha + \frac{1}{2}a\beta|\hat{I}(\omega)|^2}; \tag{24}$$

with $\alpha > -\frac{1}{2}a\beta \max_\omega |\hat{I}(\omega)|^2$ (when $a < 0$), and in this case, we could choose

$$\Delta t < \frac{2}{\alpha + \frac{1}{2}a\beta\|I\|_{L^1}^2}. \tag{25}$$

There are two condition it has to be satisfied to be stable. First, as Δt is positive, the denominator of the right side should be positive. That yields the lower bound for α

$$\alpha > -\frac{1}{2}a\beta \max_\omega |\hat{I}(\omega)|^2. \tag{26}$$

Then Δt has to satisfy the CFL condition (25) so that the system could be stable, which yields the upper bound for α:

$$\alpha < \frac{2}{\Delta t} - \frac{1}{2}a\beta \min_\omega |\hat{I}(\omega)|^2 \tag{27}$$

D.4 Non-constant Linearization of $a = \hat{y} - y$ and Stability Analysis (Section 5.2)

We also consider a non-constant linear model of K for a, so we linearize $a = \hat{y} - y$ around $K = 0$:

$$\begin{aligned} a = \hat{y} - y &= s(g_K(I)) - y \\ &\approx s(g_0(I)) + \langle s'(g_0(I))\nabla_K g_0(I), K\rangle - y \\ &= s(0) + \langle s'(0)r'(0) * I, K\rangle - y \\ &= s(0) - y + \frac{1}{2}s'(0)\bar{I}\bar{K}, \end{aligned}$$

where the subscript on g indicates the dependence on the given kernel. The gradient descent PDE following from the linearization above then becomes:

$$\partial_t K = \left[-\frac{1}{2}[s(0) - y + \frac{1}{2}s'(0)\bar{I}\bar{K}](\bar{I} + \beta[(K * I) * I]) - \alpha K\right] W. \tag{28}$$

Considering only linear terms and ignoring the constant (w.r.t K) terms, this becomes:

$$\partial_t K = \left[-\frac{1}{2}[s(0) - y]\beta[(K * I) * I] - \frac{1}{4}s'(0)\bar{I}^2\bar{K} - \alpha K \right] W. \tag{29}$$

Using forward Euler and computing the Fourier transform yields:

$$\hat{K}^{n+1} = A(\omega)\hat{K}^n(\omega), \tag{30}$$

where

$$A(\omega) = 1 - \Delta t \begin{cases} \frac{1}{2}|\hat{I}(0)|^2 \left(\beta[s(0) - y] + \frac{1}{4}s'(0)\delta(\omega)\right) + \alpha & \omega = 0 \\ \frac{1}{2}[s(0) - y]\beta|\hat{I}(\omega)|^2 + \alpha & \omega \neq 0 \end{cases}, \tag{31}$$

where δ is a Dirac delta function. Note $s(0) - y$ can be either positive or negative, so similar to the previous case where $a = \frac{\partial \ell}{\partial \hat{y}} s'$ was assumed constant, the process can be unstable without regularization.

We approximate δ with a sinc function (Fourier transform of a rectangular pulse, which is the case if the constant is defined on just a finite support), i.e.,

$$\delta(\omega) = \frac{1}{2\pi L^2} \operatorname{sinc}\left(\frac{\omega_1}{2\pi L}\right) \operatorname{sinc}\left(\frac{\omega_2}{2\pi L}\right). \tag{32}$$

We can write A as

$$A(\omega) = 1 - \Delta t \left(\frac{1}{2}[s(0) - y]\beta|\hat{I}(\omega)|^2 + \alpha + \frac{1}{4}s'(0)|\hat{I}(0)|^2\delta(\omega) \right). \tag{33}$$

The sign of $\frac{1}{2}[s(0) - y]\beta|\hat{I}(\omega)|^2 + \frac{1}{4}s'(0)|\hat{I}(0)|^2\delta(\omega)$ can be either positive or negative, therefore, in the case of no regularization, the process can be unstable.

This analysis shows that treating $a = \hat{y} - y$ as a more general non-constant linear function leads to similar CFL conditions and conclusions as the constant case examined in the main paper.

D.5 Linearization of the PDE in the "Activated" Regime (Section 5.4)

Suppose that $K * I$ is positive and away from zero, then $r'(K * I) = 1$. The non-linear PDE (12) reduces to

$$\partial_t K = -a\bar{I} - \alpha K, \tag{34}$$

where $a = \hat{y} - y$. Linearizing this as in Theorem 4.2 gives $a = s(0) - y + 0.5s'(0)\bar{K}\bar{I}$, gives the PDE:

$$\partial_t K = -(s(0) - y)\bar{I} - \frac{1}{8}\bar{I}^2\bar{K} - \alpha K. \tag{35}$$

For stability analysis, we can ignore the constant with respect to K term. One can show that in the DFT domain, the update of K_t is given by

$$\hat{K}^{n+1}(\omega) = A(\omega)\hat{K}^n(\omega),$$

where the amplifier factor is

$$A(\omega) = \begin{cases} 1 - \Delta t(\alpha + \bar{I}^2/8) & \omega = 0 \\ 1 - \alpha\Delta t & \omega \neq 0 \end{cases}.$$

For stability, $|A| < 1$, which implies the following conditions:

$$\alpha > -\frac{\bar{I}^2}{8} \quad \text{and} \quad \alpha < \min\left\{ \frac{2}{\Delta t} - \frac{\bar{I}^2}{8}, \frac{2}{\Delta t} \right\} = \frac{2}{\Delta t} - \frac{\bar{I}^2}{8}.$$

E Stability Analysis of Nesterov Momentum (Section 5.2)

We now extend the stability analysis from the gradient descent of the 1-layer CNN loss function to consider Nesterov momentum, which is widely used in deep neural network training. As we shall see, we arrive at similar conclusions (CFL conditions that result in upper and lower bounds on α) as the gradient PDE.

We start with the continuum PDE, discretize it - resulting in Nesterov's scheme, and then analyze the stability. The continuum equivalent PDE for the optimization of a loss function with momentum is given by (see [1]):

$$\partial_{tt}K + d \cdot \partial_t K = -\nabla L(K), \tag{36}$$

where ∂_{tt} denotes the second derivative in time, $d > 0$ is a constant that denotes the damping coefficient, and u denotes the evolving variable of optimization. We may use a semi-implicit Euler style discretization of the above PDE that results in classic two-part Nesterov recursion (see [1]).

Theorem E.1 (Semi-Implicit Scheme for Momentum PDE)**.** *The semi-implicit scheme for momentum PDE in* (36) *can be expressed as*

$$V^n = K^n + \frac{2 - d\Delta t}{2 + d\Delta t}(K^n - K^{n-1}) \tag{37}$$

$$K^{n+1} = V^n - \frac{2\Delta t^2}{2 + d\Delta t}\nabla_V L(V^n) \tag{38}$$

where V_t is the partial update, and d denotes the damping coefficient.

Proof. We start with discretizing PDE with momentum (36) using a fully explicit scheme. Specifically, we use a central difference approximations for both time derivatives gives a second order discretization in time:

$$\frac{K(t + \Delta t) - 2K(t) + K(t - \Delta t)}{\Delta t^2} + d\frac{K(t + \Delta t) - K(t - \Delta t)}{2\Delta t} = -\nabla_K L[K(t)] \tag{39}$$

which leads to the following update:

$$K^{n+1} = K^n + \frac{2 - d\Delta t}{2 + d\Delta t}\Delta K^n - \frac{2\Delta t^2}{2 + d\Delta t}\nabla_K L(K^n), \tag{40}$$

where $K^n = K(n\Delta t)$, and $\Delta K^n = K^n - K^{n-1}$. To obtain an update which more closely resembles the classic two-part Nesterov recursion, we use a semi-implicit discretization. That is, we replace the explicit discretization $\nabla_K L(K^n)$ with a "predicted estimate" $\widehat{\nabla_K L(K^n)}$. This estimate is obtained by applying the same discretization but evaluating $\nabla_K L$ at the partial update $V^n = K^n + \frac{2-d\Delta t}{2+d\Delta t}\Delta K^n$, which is a "look-ahead" location. Using this strategy yields this two-step update as specified in the theorem. For more details see [1]. □

To perform the stability analysis, we study linearizations of the PDE by using a similar procedure as in the stability analysis for gradient descent PDE presented in Section 5.3. The linearization of $\nabla_K L(V^n)$ is the same as in the linearized PDE (20), given by

$$\nabla_V L(V^n) = \left[-\frac{a}{2}(\bar{I} + \beta[(V * I) * I]) - \alpha V\right] W, \tag{41}$$

We now analyze the stability of the linearized equation using Von-Neumann analysis. To derive the stability conditions, we compute the spatial Discrete Fourier Transform (DFT) of the semi-implicit scheme (37) and solve for the gradient amplifier:

Theorem E.2. *The DFT of the semi-implicit scheme given in* (37)-(38) *of the linearized momentum PDE is given by*

$$\hat{V}^n = \hat{K}^n + \frac{2 - d\Delta t}{2 + d\Delta t}(\hat{K}^n - \hat{K}^{n-1}) \tag{42}$$

$$\hat{K}^{n+1} = \hat{V}^n - \frac{2\Delta t^2}{2 + d\Delta t}z(\omega)\hat{V}^n \tag{43}$$

where the gradient amplifier z is defined as

$$z(\omega) = \alpha + \frac{1}{2}a\beta|\hat{I}(\omega)|^2. \tag{44}$$

In order for the overall combined update updates to be a stable process, the gradient amplifier has to satisfy the following condition (for detailed proof, see[1]):

$$\Delta t < \frac{2}{\sqrt{3\left(\alpha + \frac{1}{2}a\beta|\hat{I}(\omega)|^2\right)}}. \tag{45}$$

This results in the following stability criteria:

Theorem E.3. *The semi-implicit scheme of the linearized momentum PDE* (37) *whose DFT is given by* (42) *is stable (in the case $w \to \infty$) if and only if the following conditions on the weight decay α and the step size Δt are met:*

$$\alpha < \alpha_{min} := \frac{4}{3\Delta t^2} - \frac{1}{2}a\beta \min_{\omega} |\hat{I}(\omega)|^2 \tag{46}$$

$$\alpha > \alpha_{max} := -\frac{1}{2}a\beta \max_{\omega} |\hat{I}(\omega)|^2. \tag{47}$$

We conduct a similar experiment as in section 5.2 to empirically validate this when the kernel is restricted to be finite support, i.e., the gradient is windowed. We empirically find the lower and upper bounds for α using the same method and compare it to the bounds in (46) and (47). We choose the learning rate = 1e-4, $a = -0.5, \beta = 1$, and damping $d = \frac{2-2\times 0.9}{\Delta t(1+0.9)}$ so that the momentum coefficient $\frac{2-d\Delta t}{2+d\Delta t} = 0.9$ is the common choice in deep learning training. Note that the choice of damping d has no impact on the stability bounds. The input data is the same as the experiment in Section 5.2. See Table 5, which verifies there are bounds on α for stability when using momentum as is the case for the gradient descent.

Kernel	α^e_{min}	α_{min}	α^e_{max}	α_{max}
16x16	3.9e6	1.07e9	1.42e8	1.33e8
32x32	1.3e7	1.07e9	1.42e8	1.33e8
64x64	3.9e7	1.07e9	1.42e8	1.33e8

Table 5: Comparison of the bounds on weight decay (α) between the non-windowed linear PDE (in (47) and (46)) and the windowed linear PDE (found empirically).

F Stability Condition for Multi-layer Networks

In the main paper, we analyzed a simplified one-layer CNN through numerical PDE tools, and extending the analysis to deeper networks with precise conditions as we showed for the 1-layer case may be more difficult. However, in this supplement, we show that our theory can makes predictions on multi-layer networks used in practice. Although this result is for highly regularized networks not employed in practice, it does show the validity of our theory.

We obtain a bound on the stable step size when the weight decay α is large. To do this, we generalize our linearized PDE stability analysis in Section 5.2 to general (multi-layer) networks. In this case, the linearized gradient descent PDE is

$$\partial_t K = -\nabla_K \ell(\hat{y}, y) - \alpha K. \tag{48}$$

We can perform Von-Neumann analysis by linearizing $\nabla_K \ell(\hat{y}, y)$ and keeping the non-homogeneous component. If we discretize and compute the DFT, we can get an update scheme of the form $\hat{K}_{t+1}(\omega) = A(\omega)\hat{K}_t(\omega)$, where the amplifier for the multi-layer network is given by

$$A(\omega) = 1 - \Delta t(\alpha + \hat{F}(\omega)) \tag{49}$$

where $\hat{F}(\omega)$ is the DFT of non-homogeneous part representing the linearization of the gradient of the cross-entropy loss, ℓ. Noting that the discretization is stable when $|A| < 1$ results in the following restriction on the time-step (learning rate):

$$\Delta t < \min_{\omega} \frac{2}{\alpha + \hat{F}(\omega)} \tag{50}$$

Note that while $\hat{F}$ may be complicated to compute exactly analytically for complex multi-layer networks. We can nevertheless obtain a prediction on the time-step in the case when α is large, i.e., when α is large (highly regularized), the step size restriction approaches:

$$\Delta t < \frac{2}{\alpha}. \tag{51}$$

We verify the preceding bound empirically on various standard multi-layer networks, by choosing a a regularization level and finding empirically the learning rate when the weight updates move from stable to unstable. We compare this to the bound predicted by (51). Results are shown in Figure 2. They show that as the regularization increases, the empirically determined maximum learning rate approaches the theoretical bound above, validating (50).

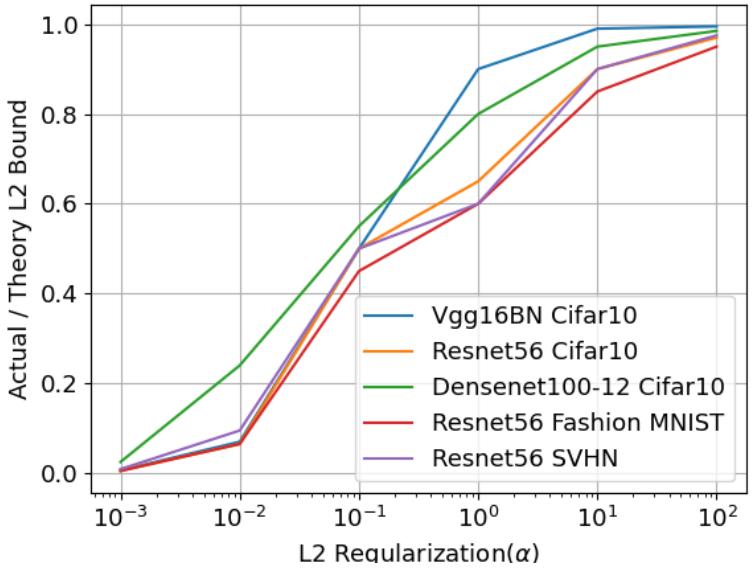

Figure 2: Empirical validation of our learning rate/weight decay bound for standard deep networks. The ratio of the empirically determined maximum stable learning rate to the theoretical maximum stable learning rate $\Delta t = 2/\alpha$ estimated by our analysis. As the L2 regularization (weight decay) increases, the ratio approaches 1, indicating that our bound becomes more accurate.

References

[1] Minas Benyamin et al. "Accelerated variational PDEs for efficient solution of regularized inversion problems". In: *Journal of Mathematical Imaging and Vision* 62.1 (2020), pp. 10–36.