
Supplementary materials

SING: A Plug-and-Play DNN Training Technique

Anonymous Author(s)

Affiliation

Address

email

35 A Theorems & Proofs

36 **Theorem 3.2.** The iterates defined by (2) are invariant w.r.t. transformation (5), and preserve the
37 mean (6).

38 *Proof.* The first property is satisfied thanks to the normalization. Indeed, consider the iterates

$$y_{t+1} = y_t - \eta \frac{\phi(\nabla \tilde{F}(y_t))}{\Gamma(\phi(\nabla \tilde{F}(y_t)))},$$

39 where \tilde{F} is defined in (5). Hence,

$$y_{t+1} = y_t - \eta \frac{\phi(\nabla \alpha F(y_t))}{\Gamma(\phi(\alpha \nabla F(y_t)))},$$

40 Since ϕ and Γ are both homogeneous operators,

$$y_{t+1} = y_t - \eta \frac{\alpha \phi(\nabla F(y_t))}{\alpha \Gamma(\phi(\nabla F(y_t)))} = y_t - \eta \frac{\phi(\nabla F(y_t))}{\Gamma(\phi(\nabla F(y_t)))}.$$

41 Therefore, we have the property $y_t = x_t$ (2). Moreover, define the mean operator $m(x) = \frac{1}{p} \sum_{i=1}^p x_i$.

42 For $z \in \mathbb{R}^p$, we have

$$m\left(\frac{\phi(z)}{\Gamma(\phi(z))}\right) = \frac{1}{p} \sum_{k=1}^D \frac{1}{\|\phi(z_{I_k})\|_2} \sum_{l \in I_k} [\phi(z)]_l = 0,$$

43 where the last inequality comes from the definition of the gradient centralization operation ϕ . Hence,
44 since $m(\cdot)$ is a linear function,

$$\begin{aligned} m(x_{t+1}) &= m\left(x_t - \eta \frac{\phi(\nabla F(x_t))}{\Gamma(\phi(\nabla F(x_t)))}\right), \\ &= m(x_t) - \eta m\left(\frac{\phi(\nabla F(x_t))}{\Gamma(\phi(\nabla F(x_t)))}\right), \\ &= m(x_t). \end{aligned}$$

45

□

46 **Theorem 3.3** (Convergence without gradient centralization). Let assumptions (7) and (8) hold.
47 Assume the gradient is computed across a mini-batch of size $B = \frac{\sigma^2}{\epsilon^2}$. Let x_t be the sequence of
48 iterates (2) with $\phi = I$. Then, we have

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}[\|\nabla F(x_t)\|_2] \leq \frac{F(x_0)}{\eta T} + (1 + \sqrt{D})\epsilon + \frac{\eta LD}{2}. \quad (9)$$

49 If we set $\tau \sim \mathcal{U}([0, T-1])$, $\eta = \frac{2\epsilon}{L}$ and $T = \frac{LF(x_0)}{2\epsilon^2}$, we obtain $\mathbb{E}[\|\nabla F(x_\tau)\|_2] \leq (2 + \sqrt{D} + D)\epsilon$.
 50 Therefore, the iteration complexity and computation complexity to achieve an ϵ -stationary point are
 51 $\mathcal{O}(1/\epsilon^2)$ and $\mathcal{O}(1/\epsilon^4)$, respectively.

52 Before proving this theorem, we will introduce and prove two technical lemmas.

53 **Lemma A.1.** For every $x \in \mathbb{R}^p$, the following equality holds

$$\left\| \frac{x}{\Gamma(x)} \right\|_2 = \sqrt{D}.$$

54 *Proof.* The function Γ is block-wise constant such that

$$\forall k \in [1, D], \forall i \in I_k, [\Gamma(x)]_i = \|x_{I_k}\|_2 = \sqrt{\sum_{j \in I_k} [x]_j^2}.$$

55 Hence

$$\begin{aligned} \left\| \frac{x}{\Gamma(x)} \right\|_2^2 &= \sum_{j=1}^d \left[\frac{x}{\Gamma(x)} \right]_j^2 = \sum_{k=1}^D \sum_{i \in I_k} \frac{[x]_i^2}{[\Gamma(x)]_i^2} \\ &= \sum_{k=1}^D \sum_{i \in I_k} \frac{[x]_i^2}{\|x_{I_k}\|_2^2} = \sum_{k=1}^D \frac{\|x_{I_k}\|_2^2}{\|x_{I_k}\|_2^2} \\ &= D. \end{aligned}$$

56 □

57 **Lemma A.2.** For every $x \in \mathbb{R}^p$, the following equality holds

$$\left\langle x, \frac{x}{\Gamma(x)} \right\rangle = \sum_{k=1}^D \|x_{I_k}\|_2 \stackrel{\text{def}}{=} N(x).$$

58 In particular, we have

$$\|x\|_2 \leq N(x).$$

59 *Proof.* The first part of the lemma can be derived directly using the same notation as for Lemma A.1:

$$\begin{aligned} \left\langle x, \frac{x}{\Gamma(x)} \right\rangle &= \sum_{j=1}^d \frac{[x]_j^2}{[\Gamma(x)]_j} = \sum_{k=1}^D \sum_{i \in I_k} \frac{[x]_i^2}{\|x_{I_k}\|_2} \\ &= \sum_{k=1}^D \frac{\|x_{I_k}\|_2^2}{\|x_{I_k}\|_2} = \sum_{k=1}^D \|x_{I_k}\|_2. \end{aligned}$$

60 The second part can be shown using the fact that $\|z\|_2 \leq \|z\|_1$ for every $z \in \mathbb{R}^D$. We define $z \in \mathbb{R}^D$
 61 such that

$$\forall k \in [1, D], z_k = \|x_{I_k}\|_2,$$

62 then $\|z\|_1 = N(x)$ and

$$\|z\|_2^2 = \sum_{k=1}^D \|x_{I_k}\|_2^2 = \sum_{k=1}^D \sum_{i \in I_k} [x]_i^2 = \sum_{j=1}^d [x]_j^2 = \|x\|_2^2.$$

63 □

64 Now, onto the proof of Theorem 3.3.

65 *Proof.* We note ∇f the stochastic approximation of the real gradient ∇F and we assume that the
 66 stochastic gradient has a σ -bounded variance ($\sigma > 0$) i.e.

$$\forall x \in \mathbb{R}^p, \mathbb{E} [\|\nabla F(x) - \nabla f(x)\|_2^2] \leq \sigma^2, \quad (11)$$

67 and that the gradient of F is L -Lipschitz such that

$$F(x_+) \leq F(x) + \langle \nabla F(x), x_+ - x \rangle + \frac{L}{2} \|x_+ - x\|_2^2. \quad (12)$$

68 When $\phi = I$, the gradient updates are given by

$$x_+ = x - \eta \frac{\nabla f(x)}{\Gamma(\nabla f(x))}, \quad (13)$$

69 where the division is element-wise. Then, using (12) with the updates defined in (13):

$$\begin{aligned} F(x_+) &\leq F(x) - \eta \left\langle \nabla F(x), \frac{\nabla f(x)}{\Gamma(\nabla f(x))} \right\rangle + \eta^2 \frac{L}{2} \left\| \frac{\nabla f(x)}{\Gamma(\nabla f(x))} \right\|_2^2 \\ &\leq F(x) - \eta \left\langle \nabla F(x) - \nabla f(x), \frac{\nabla f(x)}{\Gamma(\nabla f(x))} \right\rangle - \eta \left\langle \nabla f(x), \frac{\nabla f(x)}{\Gamma(\nabla f(x))} \right\rangle + \frac{\eta^2 LD}{2} \\ &\leq F(x) + \eta \sqrt{D} \|\nabla F(x) - \nabla f(x)\|_2 - \eta N(\nabla f(x)) + \frac{\eta^2 LD}{2}, \end{aligned} \quad (14)$$

70 where the last inequality comes from the Cauchy-Schwartz inequality used with Lemma A.1 and the
 71 first part of Lemma A.2. Using the second part of Lemma A.2, we get

$$\|\nabla F(x)\|_2 \leq \|\nabla F(x) - \nabla f(x)\|_2 + \|\nabla f(x)\|_2 \leq \|\nabla F(x) - \nabla f(x)\|_2 + N(\nabla f(x)). \quad (15)$$

72 Upper-bounding $N(\nabla f(x))$ using (14) gives us

$$\|\nabla F(x)\|_2 \leq \frac{F(x) - F(x_+)}{\eta} + (1 + \sqrt{D}) \|\nabla F(x) - \nabla f(x)\|_2 + \frac{\eta LD}{2}. \quad (16)$$

73 Then, if $\nabla f(x)$ is the average of $B = \frac{\sigma^2}{\epsilon^2}$ gradient approximations over a mini-batch, we get

$$\mathbb{E}[\|\nabla F(x) - \nabla f(x)\|_2] \leq \sqrt{\mathbb{E}[\|\nabla F(x) - \nabla f(x)\|_2^2]} \leq \frac{\sigma}{B} = \epsilon. \quad (17)$$

74 Taking the expectation in (16) for $x_+ = x_{t+1}$ and $x = x_t$ for a given $t \in \mathbb{N}$ and $(x_t)_{t \in \mathbb{N}}$ being the
 75 updates defined by (13), we get

$$\begin{aligned} \frac{1}{T} \sum_{t=0}^{T-1} \|\nabla F(x_t)\|_2 &\leq \frac{F(x_0) - F(x_T)}{\eta T} + \frac{1 + \sqrt{D}}{T} \sum_{t=0}^{T-1} \mathbb{E}[\|\nabla F(x_t) - \nabla f(x_t)\|_2] + \frac{\eta LD}{2} \\ &\leq \frac{F(x_0)}{\eta T} + (1 + \sqrt{D})\epsilon + \frac{\eta LD}{2}. \end{aligned}$$

76 The last inequality comes from (17) and the assumption that $F \geq 0$. This shows the first part of the
 77 theorem. Now, if we let $\eta = \frac{2\epsilon}{L}$ and $T = \frac{LF(x_0)}{2\epsilon^2}$ we get

$$\frac{1}{T} \sum_{t=0}^{T-1} \|\nabla F(x_t)\|_2 \leq (2 + \sqrt{D} + D)\epsilon. \quad (18)$$

78 □

79 **Theorem 3.4** (Convergence with gradient centralization). Let assumptions (7) and (8) hold. Assume
 80 the gradient is computed across a mini-batch of size $B = \frac{\sigma^2}{\epsilon^2}$. Let x_t be the sequence of iterates (2).
 81 Then we have

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}[\|\nabla F(x_t)\|_\phi] \leq \frac{F(x_0)}{\eta T} + (1 + \sqrt{D})\epsilon + \frac{\eta LD}{2}, \quad (10)$$

82 where $\|\cdot\|_\phi^2 = \langle \cdot, \phi(\cdot) \rangle_2$ is a pseudo-norm. If we set $\tau \sim \mathcal{U}([0, T-1])$, $\eta = \frac{2\epsilon}{L}$ and $T = \frac{LF(x_0)}{2\epsilon^2}$,
 83 we obtain $\mathbb{E}[\|\nabla F(x_\tau)\|_\phi] \leq (2 + \sqrt{D} + D)\epsilon$. Therefore, the iteration complexity and computation
 84 complexity to achieve an (ϵ, ϕ) -stationary point are $\mathcal{O}(1/\epsilon^2)$ and $\mathcal{O}(1/\epsilon^4)$, respectively.

85 As for the proof of Theorem 3.3, we start by introducing and proving two technical lemmas.

86 **Lemma A.3.** For every $x \in \mathbb{R}^p$, the following equality holds

$$\left\| \frac{\phi(x)}{\Gamma(\phi(x))} \right\|_2 = \sqrt{D}.$$

87 *Proof.* The proof is direct using Lemma A.1 with $x = \phi(z)$. □

88 **Lemma A.4.** For every $x \in \mathbb{R}^p$, the following inequality holds

$$\left\langle x, \frac{\phi(x)}{\Gamma(\phi(x))} \right\rangle = \sum_{k=1}^D \|x_{I_k}\|_\phi \stackrel{\text{def}}{=} N_\phi(x),$$

89 where $\|\cdot\|_\phi \stackrel{\text{def}}{=} \langle \cdot, \phi(\cdot) \rangle$ is pseudo-norm. In particular, we have

$$\|x\|_\phi \leq \|x\|_2, \quad \text{and} \quad \|x\|_\phi \leq N_\phi(x).$$

90 *Proof.* As for the other lemmas, the first part is direct

$$\begin{aligned} \left\langle x, \frac{\phi(x)}{\Gamma(\phi(x))} \right\rangle &= \sum_{j=1}^d [x]_j \frac{[\phi(x)]_j}{[\Gamma(\phi(x))]_j} = \sum_{k=1}^D \sum_{i \in I_k} [x]_i \frac{[\phi(x)]_i}{[\Gamma(\phi(x))]_i} \\ &= \sum_{k=1}^D \frac{1}{\|\phi(x_{I_k})\|_2} \sum_{i \in I_k} [x]_i [\phi(x)]_i = \sum_{k=1}^D \frac{\langle x_{I_k}, \phi(x_{I_k}) \rangle}{\|\phi(x_{I_k})\|_2} \\ &= \sum_{k=1}^D \sqrt{\langle x_{I_k}, \phi(x_{I_k}) \rangle} = \sum_{k=1}^D \|x_{I_k}\|_\phi. \end{aligned}$$

91 The penultimate inequality comes from the fact that ϕ is self-adjoint and such that $\phi \circ \phi = \text{id}$:

$$\|\phi(x)\|_2^2 = \langle \phi(x), \phi(x) \rangle = \langle x, \phi(\phi(x)) \rangle = \langle x, \phi(x) \rangle.$$

92 The Cauchy-Schwartz inequality applied on the above equality gives us the second part of the lemma:

$$\|\phi(x)\|_2^2 \leq \|x\|_2 \|\phi(x)\|_2,$$

hence $\|\phi(x)\|_2 = \|x\|_\phi \leq \|x\|_2$. Finally, we use the fact that $\|z\|_2 \leq \|z\|_1$ for every $z \in \mathbb{R}^D$ to show the last inequality. We define $z \in \mathbb{R}^D$ such that

$$\forall k \in \llbracket 1, D \rrbracket, z_k = \|x_{I_k}\|_\phi,$$

93 and we have $\|z\|_1 = \sum_{k=1}^D \|x_{I_k}\|_\phi = N_\phi(x)$. Furthermore,

$$\begin{aligned} \|z\|_2^2 &= \sum_{k=1}^D \|x_{I_k}\|_\phi^2 = \sum_{k=1}^D \langle x_{I_k}, \phi(x_{I_k}) \rangle \\ &= \sum_{k=1}^D \sum_{i \in I_k} [x]_i [\phi(x)]_i = \sum_{j=1}^d [x]_j [\phi(x)]_j \\ &= \langle x, \phi(x) \rangle = \|x\|_\phi^2. \end{aligned}$$

94 □

95 Now, we can prove Theorem 3.4.

Proof. Under the same assumptions as for Theorem 3.3, and with the gradient updates defined in (2), we have

$$x_+ = x - \eta \frac{\phi(\nabla f(x))}{\Gamma(\phi(\nabla f(x)))}.$$

96 The gradient centralization operator is linear, self-adjoint and such that $\phi^2 = \phi$. We note $\|\cdot\|_\phi^2 =$
 97 $\langle \cdot, \phi(\cdot) \rangle_2$ the pseudo-norm induced by ϕ . Using (12) with the iterates of (2) gives us

$$\begin{aligned}
 & F(x_+) \\
 & \leq F(x) - \eta \left\langle \nabla F(x), \frac{(\phi \circ \nabla f)(x)}{(\Gamma \circ \phi \circ \nabla f)(x)} \right\rangle + \eta^2 \frac{L}{2} \left\| \frac{(\phi \circ \nabla f)(x)}{(\Gamma \circ \phi \circ \nabla f)(x)} \right\| \\
 & \leq F(x) - \eta \left\langle \nabla F(x) - \nabla f(x), \frac{(\phi \circ \nabla f)(x)}{(\Gamma \circ \phi \circ \nabla f)(x)} \right\rangle - \eta \left\langle \nabla f(x), \frac{(\phi \circ \nabla f)(x)}{(\Gamma \circ \phi \circ \nabla f)(x)} \right\rangle + \frac{\eta^2 LD}{2} \\
 & \leq F(x) - \eta \sqrt{D} \|\nabla F(x) - \nabla f(x)\|_2 - \eta N_\phi(\nabla f(x)) + \frac{\eta^2 LD}{2}, \tag{19}
 \end{aligned}$$

98 where the last inequality comes from Lemma A.3 and the first part of Lemma A.4. We can derive the
 99 following upper-bound using the second and third parts of Lemma A.4:

$$\begin{aligned}
 \|\nabla F(x)\|_\phi & \leq \|\nabla F(x) - \nabla f(x)\|_\phi + \|\nabla f(x)\|_\phi \\
 & \leq \|\nabla F(x) - \nabla f(x)\|_2 + N_\phi(\nabla f(x)). \tag{20}
 \end{aligned}$$

100 Finally, we use (19) to upper-bound $N_\phi(\nabla f(x))$ and inject it in (20) to get

$$\|\nabla F(x)\|_\phi \leq \frac{F(x) - F(x_+)}{\eta} + (1 + \sqrt{D}) \|\nabla F(x) - \nabla f(x)\|_2 + \frac{\eta LD}{2}. \tag{21}$$

101 We can conclude the proof using the same argument as for Theorem 3.3. \square

102 **Theorem 3.1** (Escaping from narrow local minima). Let x_t be the sequence of iterates defined by (2)
 103 and y_t the sequence of iterates of gradient descent,

$$y_{t+1} = y_t - \eta_{\text{GD}} \nabla F(y_t). \tag{3}$$

104 Assume that $x_t \in \mathcal{B}(x^*)$ (resp. $y_t \in \mathcal{B}(x^*)$) i.e. the ball contained in the basin of attraction of x^* ,
 105 defined in Definition 3.1. Also, assume that x_t (resp. y_t) is not a critical point i.e. $\nabla F(x_t) \neq 0$ (resp.
 106 $\nabla F(y_t) \neq 0$). If the stepsize is sufficiently large,

$$\eta_{\text{SING}} \geq \frac{2r}{\sqrt{D}}, \quad \eta_{\text{GD}} \geq \frac{2r}{\|\nabla F(y_t)\|_2}, \tag{4}$$

107 then the iterates x_{t+1} (resp. y_{t+1}) is outside the set $\mathcal{B}(x^*)$.

108 *Proof.* Let us consider the more general setting

$$x_{t+1} = x_t - \eta g_t. \tag{22}$$

109 Provided $g_t \neq 0$, we can note $\|x_{t+1} - x^*\|_2^2$ is a degree two polynomial in η :

$$\|x_{t+1} - x^*\|_2^2 = \|g_t\|_2^2 \left(\eta - \left\langle \frac{g_t}{\|g_t\|_2^2}, x_t - x^* \right\rangle \right)^2 + \|x_t - x^*\|_2^2 - \left\langle \frac{g_t}{\|g_t\|_2^2}, x_t - x^* \right\rangle^2. \tag{23}$$

110 Cauchy-Schwartz inequality ensure the term outside the square is always positive. Hence

$$\|x_{t+1} - x^*\|_2 \geq \|g_t\|_2 \left| \eta - \left\langle \frac{g_t}{\|g_t\|_2^2}, x_t - x^* \right\rangle \right|. \tag{24}$$

111 Therefore, for $\eta \geq \left\langle \frac{g_t}{\|g_t\|_2^2}, x_t - x^* \right\rangle + \frac{r}{\|g_t\|_2}$ we have that if $x_t \in A(x^*) \setminus \{x^*\}$, $x_{t+1} \notin A(x^*)$. In
 112 the worst case, the RHS is equal to $2r/\|g_t\|_2$. We can further simplify this bound by considering the
 113 expression of g_t :

$$g_t^{\text{GD}} = \nabla F(x_t), \quad \|g_t^{\text{GD}}\|_2 = \|\nabla F(x_t)\|_2, \tag{25a}$$

$$g_t^{\text{NGD}} = \frac{\nabla F(x_t)}{\|\nabla F(x_t)\|_2}, \quad \|g_t^{\text{NGD}}\|_2 = 1, \tag{25b}$$

$$g_t^{\text{SING}} = \frac{\phi(\nabla F(x_t))}{\Gamma(\phi(\nabla F(x_t)))}, \quad \|g_t^{\text{SING}}\|_2 = \sqrt{D}, \tag{25c}$$

	Learning rate	Weight decay	Test loss ($\times 10^{-3}$)	Accuracy
AdamW + SING	5×10^{-2}	5×10^{-2}	0.34	96.56%
Lamb [15]	1×10^{-2}	5×10^{-4}	0.50	93.50%
NAdam [4]	1×10^{-3}	5×10^{-5}	28.3	0.02%
Yogi [16]	1×10^{-2}	5×10^{-5}	24.3	0.24%
AdamW [5, 7]	1×10^{-3}	5×10^{-2}	1.70	78.13%
AdaBelief [17]	1×10^{-3}	5×10^{-4}	3.73	60.26%
AdaFactor [10]	1×10^{-2}	5×10^{-4}	1.87	74.98%
RAdam [6]	1×10^{-3}	5×10^{-5}	20.1	0.75%
AdaBound [8]	1×10^{-3}	5×10^{-5}	38.2	0.03%

Table 1: List of the performance and best-performing hyper-parameters for the training of a ViT-S on RDE using different optimizers. As suggested by the ablation study, normalization is the main reason why SING works but doesn’t fully explain its success.

where NGD stands for normalized gradient descent. Therefore, to ensure $x_{t+1} \notin A(x^*)$ it is sufficient to have

$$\eta_{\text{GD}} \geq \frac{2r}{\|\nabla F(x_t)\|_2}, \quad (26a)$$

$$\eta_{\text{NGD}} \geq 2r, \quad (26b)$$

$$\eta_{\text{SING}} \geq \frac{2r}{\sqrt{D}}. \quad (26c)$$

□

B Additional experiments

B.1 Comparison against other optimizers

We compared other optimizers on the RDE dataset for depth estimation (Section 5.2). We chose this dataset to make the comparison because training a ViT-S on this task using Adam is unstable, and most competing methods try to fix Adam’s instabilities. For all the optimizers, we carefully tuned the learning rate and weight decay using the same methodology as for the classification task in ImageNet (see Section 5.1). We set other hyper-parameters to their default value. The results are available in Table 1. Notably, we found that most optimizers simply do not converge: the combination of AdamW and SING sometimes outperform competitors by a factor of **100** in terms of test loss. Notably, most of the competitors do not use decoupled weight decay [7]. We claim this is part of the reason why AdamW, AdaBelief and AdaFactor (which include it by default) outperform their counterparts by a factor of ten.

C Training details

C.1 Depth Estimation

The metric we used to measure accuracy consists in averaging the prediction given for each visible pixel of each rectangle. Then a prediction counts as 1 if *every* rectangle within the image was attributed a correct depth. Examples of images of the dataset can be seen in Figure 1, as well as predictions when trained with AdamW and AdamW+SING. Each training lasts for one hour and a half on one Tesla V100 GPU with 32GB of VRAM.

For the evaluation of SING when combined with other optimizers, we carefully tuned each method using the same method as for ImageNet. We then reported the result corresponding to the best test loss. All the hyper-parameters can be found in Table 2. In particular, while training AdaFactor [10] we disabled the option to train with Adaptive Step Size (see [10]) as we found it to lower the performance with and without SING.

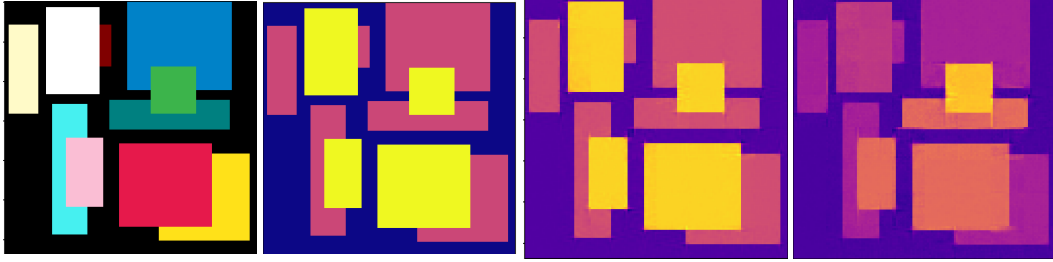


Figure 1: Example of predictions on the RDE [3] dataset. From left to right: input to the network, associated ground truth, prediction using AdamW + SING, prediction using AdamW. The mean squared error for AdamW + SING is 3×10^{-4} and 5×10^{-2} for AdamW.

	w/ softplus [12]	Learning rate	Weight decay	Accuracy
SGD	N/A	5×10^{-1}	5×10^{-5}	0.02%
SGD + SING	N/A	5×10^{-1}	5×10^{-3}	94.25%
AdamW	×	1×10^{-3}	5×10^{-2}	78.13%
AdamW + SING	✓	5×10^{-2}	5×10^{-2}	96.56%
AdamW + SING	×	5×10^{-3}	5×10^{-2}	89.38%
AdaBelief	×	1×10^{-3}	5×10^{-4}	60.26%
AdaBelief + SING	✓	5×10^{-2}	5×10^{-2}	96.70%
AdaBelief + SING	×	1×10^{-3}	5×10^{-1}	94.31%
AdaFactor	×	1×10^{-2}	5×10^{-4}	74.98%
AdaFactor + SING	✓	1×10^{-2}	5×10^{-2}	73.06%
AdaFactor + SING	×	1×10^{-2}	5×10^{-4}	76.26%

Table 2: List of the best hyper-parameters for the training of a ViT-S on RDE using different combinations of optimizers and SING. N/A stands for Not Applicable.

145 To use a ViT for an image-to-image task, we simply got rid of the classification token and instead
 146 reverted the patchification of the input: each output token is used to output a patch of the output
 147 image. This can create discontinuities on the image at the patch borders, but for the piece-wise
 148 constant images of the RDE dataset we didn't find this to be an issue.

149 C.2 Classification

150 As the FFCV library we used had already tuned SGD, we didn't modify the hyper-parameters. Indeed,
 151 when we tried other configurations the performance dropped. For AdamW + GC and AdaBelief, we
 152 kept the same hyper-parameters than those found for AdamW. We tested other configurations but
 153 found them to be sub-optimal. The best hyper-parameters found for each network are reported in
 154 Table 3. Notably, we found as a rule of thumb that the best learning rate of SING was ten times the
 155 best for AdamW, and the weight decay to be ten times lower. For AdamW and SING, no weight
 156 decay was applied to the biases of the network and to the normalization layers.

157 For CIFAR100, the training was very cumbersome due to the large tendency to overfitting. We
 158 used label smoothing [11] with a value of 0.1 and small batch sizes of 128 to increase the variance.
 159 The best learning rate found was also 10^{-1} but the best weight decay was 5×10^{-2} , probably
 160 due to the large chance of overfitting. For AdamW, the best learning rate was 10^{-3} and the best
 161 weight decay was 5×10^{-1} . We also froze the learning of the batch normalization's parameters.
 162 We carefully verified that these choices benefited all optimizers. The data were augmented using
 163 random crops, random horizontal flips and random rotations of maximum 15 degrees. We tuned the
 164 hyper-parameters the same way we did for ImageNet. Notably, the optimal weight decay we found
 165 for AdamW was 5×10^{-1} and 5×10^{-2} for AdamW+SING.

		Learning rate	Weight decay
ResNet18	SGD	5×10^{-1}	5×10^{-5}
ResNet18	AdamW	10^{-2}	5×10^{-2}
ResNet18	AdamW + SING	10^{-1}	5×10^{-3}
ResNet34	SGD	5×10^{-1}	5×10^{-5}
ResNet34	AdamW	10^{-2}	5×10^{-2}
ResNet34	AdamW + SING	10^{-1}	5×10^{-3}

Table 3: For each configuration, the list of the best hyper-parameters when training on ImageNet.

	SGD	W + SING	AdamW
ResNet18	75.63%	78.24% ($\pm 0.21\%$)	77.95% ($\pm 0.15\%$)

Table 4: Top-1 accuracy on CIFAR100. The results are averaged across five runs and the values are reported in the format mean \pm std.

166 C.3 Natural language processing

167 For the training on IWSLT14, we used the code of [14] which is a fork of the FAIRSEQ [9] library.
168 As stated, we took the code as is and launched it but found the BLUE score to be one point lower
169 than the reported one (for AdamW and AdaHessian [14]). We therefore re-tuned AdamW but found
170 the hyper-parameters reported in [13] to be the best. During training, we found the LayerNorm [1] to
171 be the cause of an increasing gradient norm throughout training. We therefore froze the layers and
172 recovered a normal dynamic. Notably, we found our optimizer to generalize better than AdamW,
173 see Figure 2 for more details. For SING, the best learning rate found was 10^{-2} and the best weight
174 decay was 5×10^{-2} . The learning rate is decreased using a cosine decay. For AdamW and for the
175 rest of the hyper-parameters, we copied the setting of [13]: a dropout of 0.3, a cross-entropy loss with
176 Label Smoothing [11] of 0.1, the gradient is accumulated for four iterations, the maximum number of
177 tokens is 4096, the beam size is five and the length penalization is one. The networks are trained for
178 200 epochs. Notably, the learning rate of AdamW is decayed using an invert square-root scheduler.
179 We tested using a cosine decay but it lowered the final performance.

180 For the fine-tuning tasks, we used the code provided by the HuggingFace library as is and launched
181 the trainings, only to find the performance to be lower than suggested. We suspect the numerous
182 changes in the library since the scripts were made to be the cause of it. We trained SING by freezing
183 the LayerNorm layers and tuned the hyper-parameters. For SQuAD, we found the performance to be

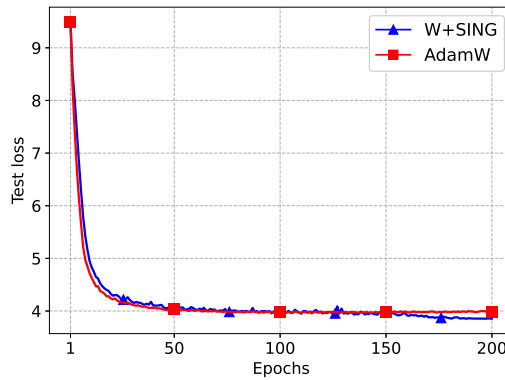


Figure 2: Test losses on the IWSLT14 dataset. We see that the combination of AdamW + SING outperforms AdamW at the end of the training. For AdamW, an inverse square-root scheduling is used as we found it performed best. For AdamW + SING, we used a cosine decay. Overall, it seems SING generalizes better than its counterpart.

reached in three epochs instead of just two for AdamW. This suggests SING overfits less rapidly than AdamW. Since the trainings were very fast, we could afford to tune more our hyper-parameters: the best learning rate found was 8×10^{-4} and the weight decay 3×10^{-3} . For AdamW, it was 3×10^{-5} and 0. For both optimizers, the best batch size was 12. For SWAG, the best batch size is 32, the number of epochs is four, the best learning rate is 2×10^{-4} and the best weight decay is 3×10^{-3} .

C.4 On the learning rate scheduler

Theorem 3.1 suggests that the learning rate must be as high as possible to escape narrow local minima. Therefore, it would make sense to consider a learning rate scheduling that keeps the learning rate constant to a high value for as long as possible. Then, once all the narrow local minima have been escaped and a large one has been found, the learning rate can be decreased to converge. While this strategy works, we found in practice that using a cosine decay to be working slightly better. We argue this strategy is to be preferred as it doesn't involve any tuning.

D Other properties

D.1 Invariant to gradient clipping

The update given by (2) cancels most clipping strategies. Indeed, most clipping strategies amount to multiplying the gradient by a certain factor (for instance $\alpha/\|\nabla f(x)\|_2$ or $\alpha\|x\|_2/\|\nabla f(x)\|_2$). Since ϕ and Γ are both homogeneous operators, the normalization cancels any layer-wise multiplication of the gradient by a scalar.

E Broader impact

The goal of our technique is to improve the stability of the training of neural networks. This may open the door to further improvements in neural networks as the architecture of tomorrow might be difficult to train with the today's optimizers. Our technique also allows for faster trainings, which reduces the amount of energy needed to train one network. Furthermore, our method allows the usage of light hyper-parameter-search strategies which further reduces the need in resources.

There are risks associated with easier and better training of neural networks, as these aspects will also benefit applications that can be detrimental to society. In addition, better learning on a biased dataset might result in networks with stronger biases (e.g. [2]). While such biases are considered to be caused by the dataset, they could be exacerbated by the model capacity and the effectiveness of the training algorithm.

References

- [1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- [2] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR, 2018.
- [3] Adrien Courtois, Jean-Michel Morel, and Pablo Arias. Investigating neural architectures by synthetic dataset design. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4890–4899, 2022.
- [4] Timothy Dozat. Incorporating nesterov momentum into adam. *ICLR 2016 Workshop*, 2016.
- [5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *Int. Conf. on Learning Representations*, 2015.
- [6] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*, 2019.
- [7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [8] Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. *arXiv preprint arXiv:1902.09843*, 2019.
- [9] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*, 2019.
- [10] Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR, 2018.
- [11] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.
- [12] Qianqian Tong, Guannan Liang, and Jinbo Bi. Calibrating the adaptive learning rate to improve convergence of adam. *Neurocomputing*, 481:333–356, 2022.
- [13] Haoran Xu, Benjamin Van Durme, and Kenton Murray. Bert, mbert, or bibert? a study on contextualized embeddings for neural machine translation. *arXiv preprint arXiv:2109.04588*, 2021.
- [14] Zhewei Yao, Amir Gholami, Sheng Shen, Mustafa Mustafa, Kurt Keutzer, and Michael Mahoney. Adahessian: An adaptive second order optimizer for machine learning. In *proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 10665–10673, 2021.
- [15] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.
- [16] Manzil Zaheer, Sashank Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex optimization. *Advances in neural information processing systems*, 31, 2018.
- [17] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Advances in neural information processing systems*, 33:18795–18806, 2020.