Neural Mechanics: symmetry and broken conservation laws in deep learning dynamics

Daniel Kunin*, Javier Sagastuy-Brena, Surya Ganguli, Daniel L.K. Yamins, Hidenori Tanaka*†

Stanford University
† Physics & Informatics Laboratories, NTT Research, Inc.

Abstract

Predicting the dynamics of neural network parameters during training is one of the key challenges in building a theoretical foundation for deep learning. A central obstacle is that the motion of a network in high-dimensional parameter space undergoes discrete finite steps along complex stochastic gradients derived from real-world datasets. We circumvent this obstacle through a unifying theoretical framework based on intrinsic symmetries embedded in a network's architecture that are present for *any* dataset. We show that any such symmetry imposes stringent geometric constraints on gradients and Hessians, leading to an associated conservation law in the continuous-time limit of stochastic gradient descent (SGD), akin to Noether's theorem in physics. We further show that finite learning rates used in practice can actually break these symmetry induced conservation laws. We apply tools from finite difference methods to derive *modified gradient flow*, a differential equation that better approximates the numerical trajectory taken by SGD at finite learning rates. We combine modified gradient flow with our framework of symmetries to derive exact integral expressions for the dynamics of certain parameter combinations. We empirically validate our analytic predictions for learning dynamics on VGG-16 trained on Tiny ImageNet. Overall, by exploiting symmetry, our work demonstrates that we can analytically describe the learning dynamics of various parameter combinations at finite learning rates and batch sizes for state of the art architectures trained on *any* dataset.

Just like the fundamental laws of classical and quantum mechanics taught us how to control and optimize the physical world for engineering purposes, a better understanding of the laws governing neural network learning dynamics can have a profound impact on the optimization of artificial neural networks. This raises a foundational question: *what, if anything, can we quantitatively predict about the learning dynamics of large-scale, non-linear neural network models driven by real-world datasets and optimized via stochastic gradient descent with a finite batch size, learning rate, and with or without momentum?* In order to make headway on this extremely difficult question, existing works have made major simplifying assumptions on the network, such as restricting to identity activation functions [1], infinite width layers [2], or single hidden layers [3]. Many of these works have also ignored the complexity introduced by stochasticity and discretization by only focusing on the learning dynamics under gradient flow. In the present work, we make the first step in an orthogonal direction. Rather than introducing unrealistic assumptions on the model or learning dynamics, we uncover restricted, but meaningful, combinations of parameters with simplified dynamics that can be solved exactly without introducing a single assumption. We make this fundamental contribution by constructing a framework harnessing the geometry of the loss shaped by symmetry and realistic continuous equations of learning.

* Equal contribution. Correspondence to kunin@stanford.edu & hidenori.tanaka@ntt-research.com

Third Workshop on Machine Learning and the Physical Sciences (NeurIPS 2020), Vancouver, Canada.

1 Symmetries in the Loss shape Gradient and Hessian Geometries

While we initialize neural networks randomly, their gradients and Hessians at all points in training, no matter the loss or dataset, obey certain geometric constraints. Some of these constraints have been noticed previously as a from of implicit regularization [4, 5], while others have been leveraged algorithmically in applications from network pruning [6] to interpretability [7]. Remarkably, all these geometric constraints can be understood as consequences of numerous differentiable symmetries in the loss introduced by neural network architectures. A set of parameters observes a differentiable symmetry in the loss if the loss doesn't change under a certain differentiable transformation of these parameters. This invariance introduces associated geometric constraints on the gradient and Hessian.

Consider a function $f(\theta)$ where $\theta \in \mathbb{R}^m$. This function possesses a differentiable symmetry if it is invariant under the differentiable action ψ of a group G on the parameter vector θ, i.e., if $\theta \mapsto \psi(\theta, \alpha)$ where $\alpha \in G$, then $F(\theta, \alpha) = f(\psi(\theta, \alpha)) = f(\theta)$ for all (θ, α). The existence of a symmetry enforces a geometric structure on the gradient, ∇F. Evaluating the gradient at the identity element of G, so that $\psi(\theta, \alpha) = \theta$, yields the result, $\langle \nabla f, \partial_\alpha \psi \rangle = 0$, which implies that the gradient ∇f is perpendicular to the vector field $\partial_\alpha \psi$ that generates the symmetry, for all θ. The symmetry also enforces a geometric structure on the Hessian, $\mathbf{H}F$. Evaluating the Hessian at the identity element of G yields the result, $\mathbf{H}f \partial_\theta \psi \partial_\alpha \psi + \partial_\theta \partial_\alpha \psi \nabla f = 0$, which constrains the Hessian $\mathbf{H}f$. See appendix A for the derivation of these properties and other geometric consequences of symmetry.

We will now consider the specific setting of a neural network parameterized by $\theta \in \mathbb{R}^m$, the training loss $\mathcal{L}(\theta)$, and three families of symmetries (translation, scale, and rescale) that commonly appear in modern network architectures.

Translation symmetry. Translation symmetry is defined by the group $\mathbb{R}$ and action $\psi_\alpha(\theta) = \theta + \alpha \mathbb{1}_\mathcal{A}$ where $\mathbb{1}_\mathcal{A}$ is the indicator vector for some subset $\mathcal{A}$ of the parameters $\{\theta_1, \ldots, \theta_m\}$. The loss function $\mathcal{L}$ thus possesses translation symmetry if $\mathcal{L}(\theta) = \mathcal{L}(\theta + \alpha \mathbb{1}_\mathcal{A})$ for all $\alpha \in \mathbb{R}$. Such a symmetry in turn implies the loss gradient $\partial_\theta \mathcal{L} = g$ is orthogonal to the indicator vector $\partial_\alpha \psi_\alpha = \mathbb{1}_\mathcal{A}$, $\langle g, \mathbb{1}_\mathcal{A} \rangle = 0$, and that the Hessian matrix $H = \partial_\theta^2 \mathcal{L}$ has the indicator vector in its kernel, $H \mathbb{1}_\mathcal{A} = 0$. Any network using the softmax function gives rise to translation symmetry for the parameters immediately preceding the function.

Scale symmetry. Scale symmetry is defined by the group $\mathrm{GL}_1^+(\mathbb{R})$ and action $\theta \mapsto \alpha_\mathcal{A} \odot \theta$ where $\alpha_\mathcal{A} = \alpha \mathbb{1}_\mathcal{A} + \mathbb{1}_{\mathcal{A}^c}$. The loss function possesses scale symmetry if $\mathcal{L}(\theta) = \mathcal{L}(\alpha_\mathcal{A} \odot \theta)$ for all $\alpha \in \mathrm{GL}_1^+(\mathbb{R})$. This symmetry immediately implies the loss gradient is everywhere perpendicular to the parameter vector itself $\partial_\alpha \psi_\alpha = \theta \odot \mathbb{1}_\mathcal{A} = \theta_\mathcal{A}$, $\langle g, \theta_\mathcal{A} \rangle = 0$, and relates to the Hessian matrix, where $\partial_\theta \mathcal{L} \partial_\alpha \partial_\theta \psi_\alpha = g \ \mathrm{diag}(\mathbb{1}_\mathcal{A}) = g_\mathcal{A}$, as $H \theta_\mathcal{A} + g_\mathcal{A} = 0$. Batch normalization leads to scale symmetry for the parameters immediately preceding the batch normalization layer.

Rescale symmetry. Rescale symmetry is defined by the group $\mathrm{GL}_1^+(\mathbb{R})$ and action $\theta \mapsto \alpha_{\mathcal{A}_1} \odot \alpha_{\mathcal{A}_2}^{-1} \odot \theta$ where $\mathcal{A}_1$ and $\mathcal{A}_2$ are two disjoint sets of parameters. The loss function possesses rescale symmetry if $\mathcal{L}(\theta) = \mathcal{L}(\alpha_{\mathcal{A}_1} \odot \alpha_{\mathcal{A}_2}^{-1} \odot \theta)$ for all $\alpha \in \mathrm{GL}_1^+(\mathbb{R})$. This symmetry immediately implies the loss gradient is everywhere perpendicular to the sign inverted parameter vector $\partial_\alpha \psi = \theta_{\mathcal{A}_1} - \theta_{\mathcal{A}_2} = \theta \odot (\mathbb{1}_{\mathcal{A}_1} - \mathbb{1}_{\mathcal{A}_2})$, $\langle g, \theta_{\mathcal{A}_1} - \theta_{\mathcal{A}_2} \rangle = 0$ and relates to the Hessian matrix, where $\partial_\theta \mathcal{L} \partial_\alpha \partial_\theta \psi = g \mathrm{diag}(\mathbb{1}_{\mathcal{A}_1} - \mathbb{1}_{\mathcal{A}_2}) = g_{\mathcal{A}_1} - g_{\mathcal{A}_2}$, as $H(\theta_{\mathcal{A}_1} - \theta_{\mathcal{A}_2}) + g_{\mathcal{A}_1} - g_{\mathcal{A}_2} = 0$. For networks with continuous, homogeneous activation functions $\phi(z) = \phi'(z) z$ (e.g. ReLU, Leaky ReLU, linear), this symmetry emerges at every hidden neuron by considering all incoming and outgoing parameters to the neuron.

2 Symmetry leads to Conservation Laws Under Gradient Flow

We now explore how geometric constraints on gradients and Hessians arising as a consequence of symmetry, impact network learning dynamics. In this work we focus on stochastic gradient descent (SGD), the workhorse of modern deep learning optimization. We will consider a model parameterized by θ, a training dataset $\{x_1, \ldots, x_N\}$ of size N, and a training loss $\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^N \ell(\theta, x_i)$ with corresponding gradient $g(\theta) = \frac{\partial \mathcal{L}}{\partial \theta}$.

The gradient descent update with learning rate η is $\theta^{(n+1)} = \theta^{(n)} - \eta g(\theta^{(n)})$, which is a forward Euler discretization with step size η of the ordinary differential equation (ODE) $\frac{d\theta}{dt} = -g(\theta)$. In the

limit as $\eta \to 0$, gradient descent exactly matches the dynamics of this ODE, which is commonly referred to as gradient flow [8]. Equipped with a continuous model for the learning dynamics, we now ask how do the dynamics interact with the geometric properties introduced by symmetries?

Symmetry leads to conservation. Strikingly similar to Noether's theorem, which describes a fundamental relationship between symmetry and conservation for physical systems governed by Lagrangian dynamics, *every symmetry of a network architecture has a corresponding conserved quantity through training under gradient flow*. Consider some subset of the parameters $\mathcal{A}$ that respects either a translation, scale, or rescale symmetry. As shown in section 1, the gradient of the loss $g(\theta)$ is always perpendicular to the vector field that generates the symmetry $\partial_\alpha \psi$. Projecting the gradient flow learning dynamics onto the generator vector field yields a differential equation $\langle \frac{d\theta}{dt}, \partial_\alpha \psi \rangle = 0$. Integrating this equation through time results in the conservation laws,

$$\textbf{Translation:} \quad \langle \theta_{\mathcal{A}}(t), \mathbb{1} \rangle = \langle \theta_{\mathcal{A}}(0), \mathbb{1} \rangle \tag{1}$$

$$\textbf{Scale:} \quad |\theta_{\mathcal{A}}(t)|^2 = |\theta_{\mathcal{A}}(0)|^2 \tag{2}$$

$$\textbf{Rescale:} \quad |\theta_{\mathcal{A}_1}(t)|^2 - |\theta_{\mathcal{A}_2}(t)|^2 = |\theta_{\mathcal{A}_1}(0)|^2 - |\theta_{\mathcal{A}_2}(0)|^2 \tag{3}$$

Each of these equations define a conserved constant of learning through training. For parameters with translation symmetry, their sum is conserved, effectively constraining their dynamics to a hyperplane. For parameters with scale symmetry, their euclidean norm is conserved, effectively constraining their dynamics to a sphere. For parameters with rescale symmetry, their difference in squared euclidean norm is conserved, effectively constraining their dynamics to a hyperbola.

3 A Realistic Continuous Model for Stochastic Gradient Descent

In section 2 we combined the geometric constraints introduced by symmetries with gradient flow to derive conservation laws for simple combinations of parameters during training. Gradient flow is too simple of a continuous model for realistic SGD training. Here, we construct a more realistic continuous model for stochastic gradient descent.

Modeling weight decay. Explicit regularization through the addition of an L_2 penalty on the parameters, with regularization constant λ, is very common practice when training modern deep learning models. For stochastic gradient descent, the result leads to the updated continuous model $\frac{d\theta}{dt} = -g(\theta) - \lambda\theta$.

Modeling stochasticity. Stochastic gradients $\hat{g}_{\mathcal{B}}(\theta)$ arise when we consider a batch $\mathcal{B}$ of size S drawn uniformly from the indices $\{1, \ldots, N\}$ forming the unbiased gradient estimate $\hat{g}_{\mathcal{B}}(\theta) = \frac{1}{S}\sum_{i \in \mathcal{B}} \nabla \ell(\theta, x_i)$. When the batch size is much smaller than the size of the dataset, $S \ll N$, then we can model the batch gradient as an average of S i.i.d. samples from a noisy version of the true gradient $g(\theta)$. Using the central limit theorem, we assume $\hat{g}_{\mathcal{B}}(\theta) - g(\theta)$ is a Gaussian random variable with mean $\mu = 0$ and covariance matrix $\Sigma = \frac{1}{S} G(\theta) G(\theta)^\intercal$. Under this assumption, the stochastic gradient update can be written as $\theta^{(n+1)} = \theta^{(n)} - \eta g(\theta^{(n)}) + \frac{\eta}{\sqrt{S}} G(\theta)\xi$, where ξ is a standard normal random variable. This update is an Euler-Maruyama discretization with step size η of the stochastic differential equation $d\theta = -g(\theta)dt + \sqrt{\frac{\eta}{S}} G(\theta) dW_t$, where W_t is a standard Wiener process. Without *any* additional assumptions, the differential symmetries intrinsic to neural network architectures add fundamental constraints on $G(\theta)$. As shown in section 1, the gradient of the loss, regardless of the batch, is orthogonal to the generator vector field $\partial_\alpha \psi$ associated with a symmetry. In particular, the stochastic noise must also observe the same property, implying $G(\theta)^\intercal \partial_\alpha \psi = 0$. In other words, the differential symmetry inherent in neural network architectures projects the noise introduced by stochastic gradients to low rank subspaces.

Modeling discretization. The effect of discretization when modeling continuous dynamics is a well studied problem in the numerical analysis of partial differential equations. One tool commonly used in this setting, is modified equation analysis [9], which determines how to better model discrete steps with a continuous differential equation by introducing higher order spatial or temporal derivatives.

Gradient descent always moves in the direction of steepest descent on a loss function $\mathcal{L}$ at each step, however, due to the finite nature of the learning rate, it fails to remain on the continuous steepest descent path given by gradient flow. [10, 11] and most recently [12], demonstrate that the gradient descent trajectory closely follows the steepest descent path of a *modified loss* function $\widetilde{\mathcal{L}}$.

The divergence between these trajectories fundamentally depends on the learning rate η and the curvature H. As derived in [12], this divergence is given by the gradient correction $-\frac{1}{2}Hg$, which is the gradient of the norm $-\frac{\eta}{4}|\nabla\mathcal{L}|$. Thus, the modified loss is $\widetilde{\mathcal{L}} = \mathcal{L} + \frac{\eta}{4}|\nabla\mathcal{L}|^2$ and the modified gradient flow ODE is $\frac{d\theta}{dt} = -g(\theta) - \frac{\eta}{2}H(\theta)g(\theta)$.

4 Combining Symmetry and Modified Gradient Flow to Derive Exact Learning Dynamics

As shown in section 2, each symmetry results in a conserved quantity under gradient flow. We now study how weight decay, momentum, stochastic gradients, and finite learning rates all interact to break these conservation laws. Remarkably, even when using a more realistic continuous model for stochastic gradient descent, as discussed in section 3, we can derive exact learning dynamics for the previously conserved quantities. To do this we (i) consider a realistic continuous model for SGD, (ii) project these learning dynamics onto the generator vector fields $\partial_\alpha\psi$ associated with each symmetry, (iii) harness the geometric constraints from section 1 to derive simplified ODEs, and (iv) solve these ODEs to obtain exact dynamics for the previously conserved quantities. We first consider the continuous model of SGD without momentum incorporating weight decay, stochasticity, and modified loss. In this setting, the exact dynamics, for the parameter combinations tied to the symmetries are,

$$\textbf{Translation:} \quad \langle\theta_{\mathcal{A}}(t), \mathbb{1}\rangle = e^{-\lambda t}\langle\theta_{\mathcal{A}}(0), \mathbb{1}\rangle \tag{4}$$

$$\textbf{Scale:} \quad |\theta_{\mathcal{A}}(t)|^2 = e^{-2\lambda t}|\theta_{\mathcal{A}}(0)|^2 + \eta\int_0^t e^{-2\lambda(t-\tau)}\,|g_{\mathcal{A}}|^2\,d\tau \tag{5}$$

$$\textbf{Rescale:} \quad |\theta_{\mathcal{A}_1}(t)|^2 - |\theta_{\mathcal{A}_2}(t)|^2 = \tag{6}$$

$$e^{-2\lambda t}(|\theta_{\mathcal{A}_1}(0)|^2 - |\theta_{\mathcal{A}_2}(0)|^2) + \eta\int_0^t e^{-2\lambda(t-\tau)}\left(\left|g_{\theta_{\mathcal{A}_1}}\right|^2 - \left|g_{\theta_{\mathcal{A}_2}}\right|^2\right)d\tau$$

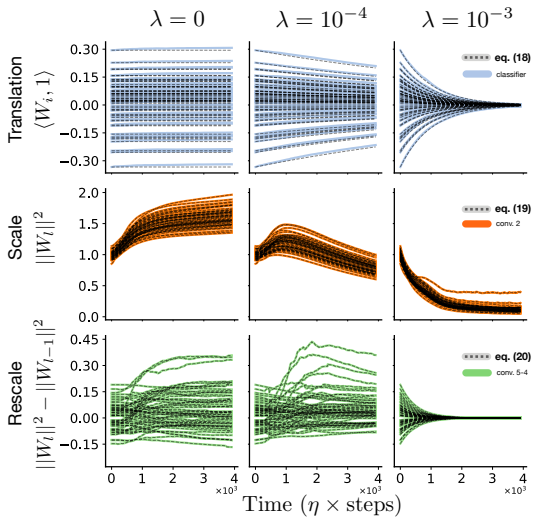
Notice how these equations are equivalent to the conservation laws derived in section 2 when $\eta = \lambda = 0$. Remarkably, even in typical hyperparameter settings (weight decay, stochastic batches, finite learning rates), these solutions match nearly perfectly with empirical results from modern neural networks (VGG-16) trained on real-world datasets (Tiny ImageNet), as shown in Fig. 1a. We will now discuss each equation individually.

Translation dynamics. For parameters with translation symmetry, equation 4 implies that the sum of these parameters ($\langle\theta_{\mathcal{A}}(t), \mathbb{1}\rangle$) decays exponentially to zero at a rate proportional to the weight decay. Equation 4 does not directly depend on the learning rate η nor any information of the dataset or task. This is due to the lack of curvature in the gradient field for these parameters. This implies that at initialization we can deterministically predict the trajectory for the parameter sum as simple exponential functions with a rate defined by the weight decay. The first row in Fig. 1a demonstrates this qualitatively, as all trajectories are smooth exponential functions that converge faster for increasing levels of weight decay.

Scale dynamics. For parameters with scale symmetry, equation 5 implies that the norm for these parameters ($|\theta_{\mathcal{A}}|^2$) is the sum of an exponentially decaying memory of the norm at initialization and an exponentially weighted integral of gradient norms accumulated through training. Compared to the translation dynamics, the scale dynamics do depend on the data through the gradient norms accumulated throughout training. Without weight decay $\lambda = 0$, the first term stays constant and the second term grows monotonically. With weight decay $\lambda > 0$, the first term decays monotonically to zero, while the second term can decay or grow, but always stays positive. The second row in Fig. 1a demonstrates these qualitative relationships. Without weight decay the norms increases monotonically as predicted and with weight decay the dynamics are non-monotonic and present more complex behavior. To better understand the forces driving these complex dynamics, we can examine the time derivative of equation 5, $\frac{d}{dt}|\theta_{\mathcal{A}}(t)|^2 = -2\lambda|\theta_{\mathcal{A}}(t)|^2 + \eta\left|g_{\mathcal{A}}\right|^2$. From this equation we see that there is a competition between a centripetal effect due to weight decay ($-2\lambda|\theta_{\mathcal{A}}(t)|^2$) and a centrifugal effect due to discretization ($\eta\left|g_{\mathcal{A}}\right|^2$). The centripetal effect due to weight decay is a direct consequence of its regularizing influence, pulling the parameters towards the origin. The centrifugal effect due to discretization originates from the spherical geometry of the gradient field in parameter space – because scale symmetry implies the gradient is always orthogonal to the parameter

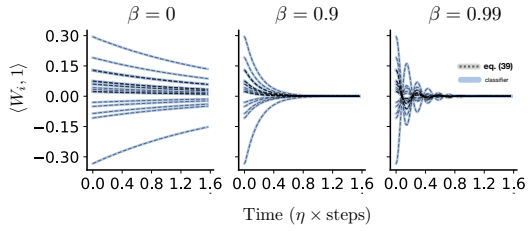
itself, each discrete update with a finite learning rate effectively pushes the parameters away from the origin. At the stationary state of the dynamics, these forces will balance leading the dynamics of these parameters to be constrained to the surface of a high-dimensional sphere. In particular, at stationarity, then $\frac{d}{dt}|\theta(t)|^2 = 0$, which gives the condition $\omega(t) \equiv \left|\frac{d\theta}{dt}\right| / |\theta| = \sqrt{\frac{2\lambda}{\eta}}$. Consistent with the results of [13], this implies that at stationarity the angular speed $\omega(t)$ of the weights is constant and governed only by the learning rate η and weight decay constant λ.

Rescale dynamics. For parameters with rescale symmetry, equation 6 is the sum of an exponentially decaying memory of the difference in norms at initialization and an exponentially weighted integral of difference in gradient norms accumulated through training. Similar to the scale dynamics, the rescale dynamics do depend on the data through the gradient norms, however unlike the scale dynamics we have no guarantee that the integral term is always positive. This leads to quite sophisticated, complex dynamics, consistent with the third row in Fig. 1a. Despite the complexity, our theory, nevertheless, quantitatively matches the empirics. The only apparent pattern from the empirics is that for large enough weight decay, the regularization dominates any complexity introduced by the gradient norms and the difference in parameter norms decays exponentially to zero.



(a) **Exact dynamics of VGG-16 on Tiny ImageNet.** We plot the column sum of the final linear layer (top row) and the difference between squared channel norms of the fifth and fourth convolutional layer (bottom row) of a VGG-16 model without batch normalization. We plot the squared channel norm of the second convolution layer (middle row) of a VGG-16 model with batch normalization. Both models are trained on Tiny ImageNet with SGD with learning rate $\eta = 0.1$, weight decay λ, batch size $S = 256$, for 100 epochs . Colored lines are empirical and black dashed lines are the theoretical predictions from equations (4), (5), and (6).

Harmonic oscillation with momentum As shown in Fig. 1b and explained in appendix B, if we consider a continuous model of SGD *with* momentum then the solution we obtain take the form of driven harmonic osccilators where the driving force is given by the gradient norms, the friction is defined by the momentum constant, the spring coefficient is defined by the regularization rate, and the mass is defined by the the learning rate and momentum constant.



(b) **Momentum leads to harmonic oscillation.** Here we use the same training setup as above with weight decay $\lambda = 5 \times 10^{-3}$ and momentum $\beta \in \{0, 0.9, 0.99\}$. See appendix B for details of how momentum leads to harmonic motion.

5 Conclusion

Despite being the central guiding principle in the exploration of the physical world [14, 15], symmetry has been underutilized in understanding the mechanics of neural networks. In this paper, we constructed a unifying theoretical framework harnessing the geometric properties of symmetry and realistic continuous equations for learning dynamics modeling weight decay, momentum, stochasticity, and discretization. We use this framework to derive *exact* dynamics for meaningful combinations of parameters, which we experimentally verified on large scale neural networks and datasets. For example, in the case of a VGG-16 model with batch normalization trained on Tiny-ImageNet (one of the model/dataset combinations we considered in section 4) there are $12,751$ distinct parameter combinations whose dynamics we can analytically describe. Overall, this work provides the first solid step towards the mechanics of learning in neural networks.

Broader Impact

As we deploy deep neural networks in society, building solid theoretical foundation of the training dynamics is crucial to make them efficient, reliable, and bias-free. Our work provides quantitative predictions that are applicable even in large scale models and dataset helping us achieve these goals.

References

[1] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[2] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[3] David Saad and Sara Solla. Dynamics of on-line gradient descent learning for multilayer neural networks. *Advances in neural information processing systems*, 8:302–308, 1995.

[4] Simon S Du, Wei Hu, and Jason D Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. In *Advances in Neural Information Processing Systems*, pages 384–395, 2018.

[5] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. *arXiv preprint arXiv:1802.06509*, 2018.

[6] Hidenori Tanaka, Aran Nayebi, Niru Maheswaranathan, Lane McIntosh, Stephen Baccus, and Surya Ganguli. From deep learning to mechanistic understanding in neuroscience: the structure of retinal prediction. In *Advances in Neural Information Processing Systems*, pages 8535–8545, 2019.

[7] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7), 2015.

[8] Harold Kushner and G George Yin. *Stochastic approximation and recursive algorithms and applications*, volume 35. Springer Science & Business Media, 2003.

[9] RF Warming and BJ Hyett. The modified equation approach to the stability and accuracy analysis of finite-difference methods. *Journal of computational physics*, 14(2):159–179, 1974.

[10] Qianxiao Li, Cheng Tai, and E Weinan. Stochastic modified equations and adaptive stochastic gradient algorithms. In *International Conference on Machine Learning*, pages 2101–2110, 2017.

[11] Yuanyuan Feng, Tingran Gao, Lei Li, Jian-Guo Liu, and Yulong Lu. Uniform-in-time weak error analysis for stochastic gradient descent algorithms via diffusion approximation. *arXiv preprint arXiv:1902.00635*, 2019.

[12] David GT Barrett and Benoit Dherin. Implicit gradient regularization. *arXiv preprint arXiv:2009.11162*, 2020.

[13] Ruosi Wan, Zhanxing Zhu, Xiangyu Zhang, and Jian Sun. Spherical motion dynamics of deep neural networks with batch normalization and weight decay. *arXiv preprint arXiv:2006.08419*, 2020.

[14] Philip W Anderson. More is different. *Science*, 177(4047):393–396, 1972.

[15] David J Gross. The role of symmetry in fundamental physics. *Proceedings of the National Academy of Sciences*, 93(25):14256–14259, 1996.

[16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[17] Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017.

[18] Behnam Neyshabur, Russ R Salakhutdinov, and Nati Srebro. Path-sgd: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2422–2430, 2015.

[19] Guodong Zhang, Chaoqi Wang, Bowen Xu, and Roger Grosse. Three mechanisms of weight decay regularization. *arXiv preprint arXiv:1810.12281*, 2018.

[20] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. *arXiv preprint arXiv:1812.03981*, 2018.

[21] Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 888–896, 2019.

[22] Hidenori Tanaka, Daniel Kunin, Daniel LK Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *arXiv preprint arXiv:2006.05467*, 2020.

[23] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[24] Nikola B Kovachki and Andrew M Stuart. Analysis of momentum methods. *arXiv preprint arXiv:1906.04285*, 2019.

A Symmetry and Conservation Laws

Here we derive in detail the geometric properties of the loss landscape introduced by symmetry, as discussed in section 1. Consider a function $f(\theta)$ where $\theta \in \mathbb{R}^m$. This function possesses a symmetry if it is invariant under the action ψ of a group G on the parameter vector θ, i.e., if $\theta \mapsto \psi(\theta, \alpha)$ where $\alpha \in G$, then $F(\theta, \alpha) = f(\psi(\theta, \alpha)) = f(\theta)$ for all (θ, α). Symmetry enforces a geometric relationship between the gradient ∇F and Hessian $\mathbf{H}F$ of the composition $F(\theta, \alpha)$ with the gradient ∇f and Hessian $\mathbf{H}f$ of the original function $f(\theta)$. This relationship can be described by five constraints on ∇f and $\mathbf{H}f$. Considering these general formulae when $f(\theta) = \mathcal{L}(\theta)$, the training loss of a neural network, yields fifteen distinct equations describing the geometrical relationships between architectural symmetries and the loss landscapes, some of which have been identified individually in existing literature (Table 1).

		Translation	Scale	Rescale
∇F	$\partial_\theta F$	—	[16], [17], [18]	[5]
	$\partial_\alpha F$	—	[19], [20]	[4],[21],[22]
$\mathbf{H}F$	$\partial_\theta^2 F$	—	[19], [18]	—
	$\partial_\theta \partial_\alpha F$	—	—	—
	$\partial_\alpha^2 F$	—	—	[22]

Table 1: **Unifying existing literature through symmetry.** Here we provide references to existing literature describing geometric properties of either the gradient or Hessian introduced by a network's architecture. All of these properties can be unified as consequences of either a translation, scale, or rescale symmetry in the training loss.

A.1 Gradient Geometry

If a function f posses a symmetry, then there exists a geometric constraint on the relationship between the gradients ∇F and ∇f at all (θ, α),

$$\nabla F = \begin{pmatrix} \partial_\theta F \\ \partial_\alpha F \end{pmatrix} = \begin{pmatrix} \partial_\psi F \partial_\theta \psi \\ \partial_\psi F \partial_\alpha \psi \end{pmatrix} = \begin{pmatrix} \nabla f \\ 0 \end{pmatrix}.$$

The top element of the gradient relationship, $\partial_\theta F$, evaluated at any (θ, α), yields the property

$$\partial_\theta \psi \; \nabla f|_{\psi(\theta,\alpha)} = \nabla f|_\theta \,, \tag{7}$$

which describes how the symmetry transformation affects the function's gradients despite leaving the output unchanged. The bottom element of the gradient relationship, $\partial_\alpha F$, evaluated at the identity element of G so that $\psi(\theta, \alpha) = \theta$, yields the property

$$\langle \nabla f, \partial_\alpha \psi \rangle = 0, \tag{8}$$

which implies the gradient ∇f is perpendicular to the vector field $\partial_\alpha \psi$ that generates the symmetry, for all θ. In the specific setting when $f(\theta) = \mathcal{L}(\theta)$, the training loss of a neural network, these gradient properties are summarized in Table 2 for the translation, scale, and rescale symmetries described in section 1.

	Translation	Scale	Rescale
$g(\theta) =$	$g(\psi(\theta, \alpha))$	$\text{diag}(\alpha_\mathcal{A}) g(\psi(\theta, \alpha))$	$\text{diag}(\alpha_{\mathcal{A}_1} \odot \alpha_{\mathcal{A}_2}^{-1}) g(\psi(\theta, \alpha))$
$g(\theta) \perp$	$\mathbb{1}_\mathcal{A}$	$\theta_\mathcal{A}$	$\theta_{\mathcal{A}_1} - \theta_{\mathcal{A}_2}$

Table 2: **Geometric properties of the gradient.** The gradients of a neural network with either translation, scale or rescale symmetry observe certain geometric properties no matter the dataset or step in training.

Notice that the first row of Table 2 implies that symmetry transformations affect learning dynamics governed by gradient descent for scale and rescale symmetries, while it does not for translation

symmetry. These observations are in agreement with [17] who has shown that effective learning rate is inversely proportional to the norm of parameters immediately preceding the batch normalization layers and [18] who have noticed that SGD is not invariant to the rescale symmetry that the network output respects and proposed Path-SGD to fix the discrepancy.

A.2 Hessian Geometry

If a function f posses a symmetry, then there also exists a geometric constraint on the relationship between the Hessian matrices $\mathbf{H}F$ and $\mathbf{H}f$ at all (θ, α),

$$\mathbf{H}F = \begin{pmatrix} \partial_\theta^2 F & \partial_\theta \partial_\alpha F \\ \partial_\alpha \partial_\theta F & \partial_\alpha^2 F \end{pmatrix} = \begin{pmatrix} \partial_\psi F \partial_\theta^2 \psi + \partial_\psi^2 F (\partial_\theta \psi)^2 & \partial_\psi^2 F \partial_\theta \psi \partial_\alpha \psi + \partial_\psi F \partial_\theta \partial_\alpha \psi \\ \partial_\psi^2 F \partial_\theta \psi \partial_\alpha \psi + \partial_\psi F \partial_\theta \partial_\alpha \psi & (\partial_\alpha \psi)^\intercal \partial_\psi^2 F \partial_\alpha \psi + (\partial_\psi F)^\intercal \partial_\alpha^2 \psi \end{pmatrix} = \begin{pmatrix} \mathbf{H}f & 0 \\ 0 & 0 \end{pmatrix}.$$

The first diagonal element, $\partial_\theta^2 F$, evaluated at any (θ, α), yields the property

$$\partial_\theta^2 \psi \; \nabla f|_{\psi(\theta,\alpha)} + (\partial_\theta \psi)^2 \; \mathbf{H}f|_{\psi(\theta,\alpha)} = \mathbf{H}f|_\theta \,, \tag{9}$$

which describes how the symmetry transformation affects the function's Hessian despite leaving the output unchanged. The off-diagonal elements, $\partial_\theta \partial_\alpha F = \partial_\alpha \partial_\theta F$, evaluated at the identity element of G so that $\psi(\theta, \alpha) = \theta$, yields the property

$$\mathbf{H}f \partial_\theta \psi \partial_\alpha \psi + \partial_\theta \partial_\alpha \psi \nabla f = 0, \tag{10}$$

which implies the geometry of gradient and Hessian are connected through the action of the symmetry. Lastly, the second diagonal element, $\partial_\alpha^2 F$, represents an equality, evaluated at the identity element of G, yields the property

$$(\partial_\alpha \psi)^\intercal \mathbf{H}f (\partial_\alpha \psi) + \langle \nabla f, \partial_\alpha^2 \psi \rangle = 0, \tag{11}$$

which combines the geometric relationships in equation 8 and equation 10. In the specific setting when $f(\theta) = \mathcal{L}(\theta)$, the training loss of a neural network, these Hessian properties are summarized in Table 2 for the translation, scale, and rescale symmetries described in section 1.

	Translation	Scale	Rescale
$H(\theta) =$	$H(\psi(\theta, \alpha))$	$\text{diag}(\alpha_{\mathcal{A}}^2) H(\psi(\theta, \alpha))$	$\text{diag}(\alpha_{\mathcal{A}_1}^2 \odot \alpha_{\mathcal{A}_2}^{-2}) H(\psi(\theta, \alpha))$
$0 =$	$H \mathbb{1}_{\mathcal{A}}$	$H\theta_{\mathcal{A}} + g_{\mathcal{A}}$	$H(\theta_{\mathcal{A}_1} - \theta_{\mathcal{A}_2}) + g_{\mathcal{A}_1} - g_{\mathcal{A}_2}$
$0 =$	$\mathbb{1}_{\mathcal{A}}^\intercal H \mathbb{1}_{\mathcal{A}}$	$\theta_{\mathcal{A}}^\intercal H \theta_{\mathcal{A}}$	$(\theta_{\mathcal{A}_1} - \theta_{\mathcal{A}_2})^\intercal H (\theta_{\mathcal{A}_1} - \theta_{\mathcal{A}_2}) + g_{\mathcal{A}_1} \theta_{\mathcal{A}_1} + g_{\mathcal{A}_2} \theta_{\mathcal{A}_2}$

Table 3: **Geometric properties of the Hessian.** The Hessian matrix of a neural network with either translation, scale or rescale symmetry observe certain geometric properties no matter the dataset or step in training.

B Learning Dynamics with Momentum

Modeling momentum. Momentum is a common extension to SGD that uses an exponentially moving average of gradients to update parameters rather than a single gradient evaluation [23]. The method introduces two additional hyperparameters, a damping coefficient α and a momentum coefficient β, and applies the two step update equation, $\theta^{(n+1)} = \theta^{(n)} - \eta v^{(n+1)}$ where $v^{(n+1)} = \beta v^{(n)} + (1 - \alpha) g(\theta^{(n)})$. When $\alpha = \beta = 0$, we regain classic gradient descent. In general, α effectively reduces the learning rate and β controls how past gradients are used in future updates resulting in a form of "inertia" accelerating and smoothing the descent trajectory. Rearranging the two-step update equation, we find that gradient descent with momentum is a first-order discretization with step size $\eta(1 - \alpha)$ of the ODE, $(1 - \beta) \frac{d\theta}{dt} = -g(\theta)$.

Modified flow. Rather than modifying gradient flow with higher order "spatial" derivatives of the loss function, here we introduce higher order temporal derivatives. We start by assuming the existence of a continuous trajectory $\theta(t)$ that weaves through the discrete steps taken by gradient descent and then identify the differential equation that generates the trajectory. Rearranging the update equation for gradient descent, $\theta_{t+1} = \theta_t - \eta g(\theta_t)$, and assuming $\theta(t) = \theta_t$ and $\theta(t + \eta) = \theta_{t+1}$, gives the equality $-g(\theta_t) = \frac{\theta(t+\eta) - \theta(t)}{\eta}$, which Taylor expanding the right side results in the differential

equation $-g(\theta_t) = \frac{d\theta}{dt} + \frac{\eta}{2}\frac{d^2\theta}{dt^2} + O(\eta^2)$. Notice that in the limit as $\eta \to 0$ we regain gradient flow. For small η, we obtain a modified version of gradient flow with an additional second-order term, $\frac{d\theta}{dt} = -g(\theta) - \frac{\eta}{2}\frac{d^2\theta}{dt^2}$. This approach to modifying first-order differential equation with higher order temporal derivatives was applied by [24] to construct a more realistic continuous model for momentum.

Harmonic oscillation with momentum. We will now consider a continuous model of SGD *with* momentum. We consider the continuous model incorporating weight decay, momentum, stochasticity, and modified flow. Under this model, the solutions we obtain take the form of driven harmonic oscillators where the driving force is given by the gradient norms, the friction is defined by the momentum constant, the spring coefficient is defined by the regularization rate, and the mass is defined by the the learning rate and momentum constant. For most standard hyperparameter choices, these solutions are in the overdamped setting and align well with the first-order solutions for SGD without momentum up to a time rescaling, as shown in the left and middle panel of Fig. 1b. However, for large values of beta we can push the solution into the underdamped regime where we would expect harmonic oscillation and indeed, we can empirically verify our predictions, even at scale for VGG-16 trained on Tiny ImageNet, as in right panel of Fig. 1b.