An eigenspace view reveals how predictor networks and stop-grads provide implicit variance regularization

Manu Srinath Halvagal[1,2,*] **Axel Laborieux**[1,*] **Friedemann Zenke**[1,2]

{firstname.lastname}@fmi.ch
[1] Friedrich Miescher Institute for Biomedical Research, Basel, Switzerland
[2] Faculty of Natural Sciences, University of Basel, Basel, Switzerland

Abstract

Self-supervised learning (SSL) learns useful representations from unlabelled data by training networks to be invariant to pairs of augmented versions of the same input. Non-contrastive methods avoid collapse either by directly regularizing the covariance matrix of network outputs or through asymmetric loss architectures, two seemingly unrelated approaches. Here, by building on DirectPred [1], we lay out a theoretical framework that reconciles these two views. We derive analytical expressions for the representational learning dynamics in linear networks. By expressing them in the eigenspace of the embedding covariance matrix, where the solutions decouple, we reveal the mechanism and conditions that provide implicit variance regularization. These insights allow us to formulate a new isotropic loss function that equalizes eigenvalue contribution and renders learning more robust. Finally, we show empirically that our findings translate in nonlinear networks trained on CIFAR-10 and STL-10.

1 Introduction and background

SSL in Siamese architectures allows learning good representations without any labeled data [2–7]. These methods learn representations that are predictive across randomized transformations, i.e, the network's objective is to "pull" together its outputs for two differently augmented versions of the same input. In order to avoid "representational collapse", the trivial solution whereby network output becomes constant, SSL methods use either a contrastive objective to "push" apart representations of unrelated images [4, 5], explicit regularization of the representational covariance matrix [6, 7], or Siamese networks with an asymmetric loss[2, 3]. The latter strategy is used in the BYOL and SimSiam models. It is intriguing because it relies on a simple manipulation of the architecture and the flow of gradients in the network without any additional objective or regularization. Symmetry between the Siamese branches is broken by passing one of the representations through an additional predictor network and blocking gradients from flowing through the other "target" branch. The learning dynamics induced by such asymmetric loss Siamese architectures are surprisingly intricate [1, 8, 9] but not fully understood. A critical insight was provided by Tian et al. [1] for the case of linear feedforward and predictor networks. They showed that throughout training with the SimSiam loss, the eigenspace of the predictor weights aligns with that of the correlation matrix of the representations. Motivated by this insight, they proposed DirectPred, a simple strategy to directly set the predictor as a function of the estimated correlation matrix instead of learning it via gradient

[*] Equal contribution.

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

descent. Crucially, DirectPred and the further developed DirectCopy approach [8] remain effective at scale and outperform equivalent architectures with a linear learned predictor network.

In this work, we consider the linear network setting used previously in the theoretical analysis of non-contrastive SSL in [1, 8], and provide a straightforward analysis of how the learning dynamics in DirectPred and DirectCopy prevent representational collapse. Specifically, we show that these methods optimize an implicit loss function in the eigenspace of the representation correlation matrix that ensures the learning dynamics always converge to non-collapsed solutions. Furthermore, our analysis details out a clear connection to the covariance-regularization-based non-contrastive methods such as VICReg [7]. DirectPred and DirectCopy similarly regularize the variance of the representations, the only difference being that they do this in eigenspace instead of directly in the representation space as in VICReg. Finally, our analysis reveals that DirectPred may suffer from slow learning rates for components with associated small eigenvalues which is also a possible reason for why these approaches require an exponentially moving average (EMA) target network. Based on our analysis we propose an alternative isotropic loss function that mitigates this problem and is robust to removing the EMA.

2 Analysis of representational learning dynamics in Siamese networks

To gain analytical insight into how predictor networks and asymmetric loss configurations jointly prevent collapse, we consider a Siamese neural network $\boldsymbol{z} = f(\boldsymbol{x}; \boldsymbol{\theta})$ with $\boldsymbol{z} \in \mathbb{R}^M$, the input $\boldsymbol{x} \in \mathbb{R}^N$, and the parameters $\boldsymbol{\theta}$. We further assume a linear predictor network $W_{\mathrm{P}} \in \mathbb{R}^{M \times M}$ and use the same parameters for the online and target branches as in SimSiam [3]. We consider pairs of embeddings $\boldsymbol{z}^{(1/2)}$ corresponding to two inputs $\boldsymbol{x}^{(1/2)}$ related through augmentation. Further we define $\boldsymbol{z} = \boldsymbol{z}^{(1)}$ and the correlation matrix of network outputs $\mathbb{E}_{\boldsymbol{x}}\left[\boldsymbol{z}\boldsymbol{z}^\top\right]$ where the expectation is taken over the data distribution. Since the correlation matrix is a real symmetric matrix, we can diagonalize it as $\mathbb{E}_{\boldsymbol{x}}\left[\boldsymbol{z}\boldsymbol{z}^\top\right] = UDU^\top$ where U is the orthogonal matrix whose columns are the eigenvectors of $\mathbb{E}_{\boldsymbol{x}}\left[\boldsymbol{z}\boldsymbol{z}^\top\right]$ and D the diagonal matrix containing the eigenvalues λ_m with $m \in [1, M]$. Finally, instead of gradient-based optimization, we directly set the predictor network as a simple function of this correlation matrix as suggested in DirectPred [1]. Specifically,

$$W_{\mathrm{P}} = f_\alpha\left(\mathbb{E}_{\boldsymbol{x}}\left[\boldsymbol{z}\boldsymbol{z}^\top\right]\right) = UD^\alpha U^\top, \tag{1}$$

where α is a positive constant. We now consider the SimSiam loss $\mathcal{L}_{\mathrm{SimSiam}}$ for a single pair of embeddings $\boldsymbol{z}^{(1)}$ and $\boldsymbol{z}^{(2)}$ and re-write it in the eigenbasis of W_{p}:

$$\begin{aligned}
\mathcal{L}_{\mathrm{SimSiam}} &= \tfrac{1}{2}\|W_{\mathrm{P}}\boldsymbol{z}^{(1)} - \mathrm{SG}(\boldsymbol{z}^{(2)})\|^2 \\
&= \tfrac{1}{2}\|UD^\alpha U^\top \boldsymbol{z}^{(1)} - \mathrm{SG}(UU^\top \boldsymbol{z}^{(2)})\|^2 \\
&= \tfrac{1}{2}\|D^\alpha \hat{\boldsymbol{z}}^{(1)} - \mathrm{SG}(\hat{\boldsymbol{z}}^{(2)})\|^2 \\
&= \tfrac{1}{2}\sum_m^M |\lambda_m^\alpha \hat{z}_m^{(1)} - \mathrm{SG}(\hat{z}_m^{(2)})|^2
\end{aligned} \tag{2}$$

where we used the fact that U is orthogonal and therefore does not change the Euclidean norm. This transformation decouples the SimSiam loss into a sum of per-component losses over the eigenmodes.

We want to gain a theoretical understanding of representational changes during learning in the eigenbasis of W_{P}. To that end, we consider the continuous time gradient descent dynamics of $\mathcal{L}_{\mathrm{SimSiam}}$ in the transformed space:

$$\dot{\hat{\boldsymbol{z}}} = \nabla_{\boldsymbol{\theta}}\hat{\boldsymbol{z}}\dot{\boldsymbol{\theta}} = -\eta\hat{\Theta}_t(x, \mathcal{X})\nabla_{\hat{\mathcal{Z}}}\mathcal{L} = -\eta\hat{\Theta}_t(x, \mathcal{X})\left(\mathcal{D}_t^\alpha \hat{\mathcal{Z}}_t^{(1)} - \hat{\mathcal{Z}}_t^{(2)}\right)\mathcal{D}_t^\alpha, \tag{3}$$

where we introduced the the empirical neural tangent kernel (NTK) [10] $\hat{\Theta}_t(\mathcal{X}, \mathcal{X}) = \nabla_{\boldsymbol{\theta}}\hat{\mathcal{Z}}\nabla_{\boldsymbol{\theta}}\hat{\mathcal{Z}}^\top$ with the transformed representations $\hat{\mathcal{Z}} = \hat{z}_t(\mathcal{X}) = U^\top z_t(\mathcal{X})$ of the whole dataset $\mathcal{X}$.

While $\hat{\Theta}_t$ changes over time and is generally intractable in finite-width networks it is always positive semidefinite. This property guarantees that the cosine angle between the representational training dynamics $\dot{\hat{\mathcal{Z}}} \propto -\hat{\Theta}_t\nabla_{\hat{\mathcal{Z}}}\mathcal{L}$ and the hypothetical dynamics that result from optimizing the representations

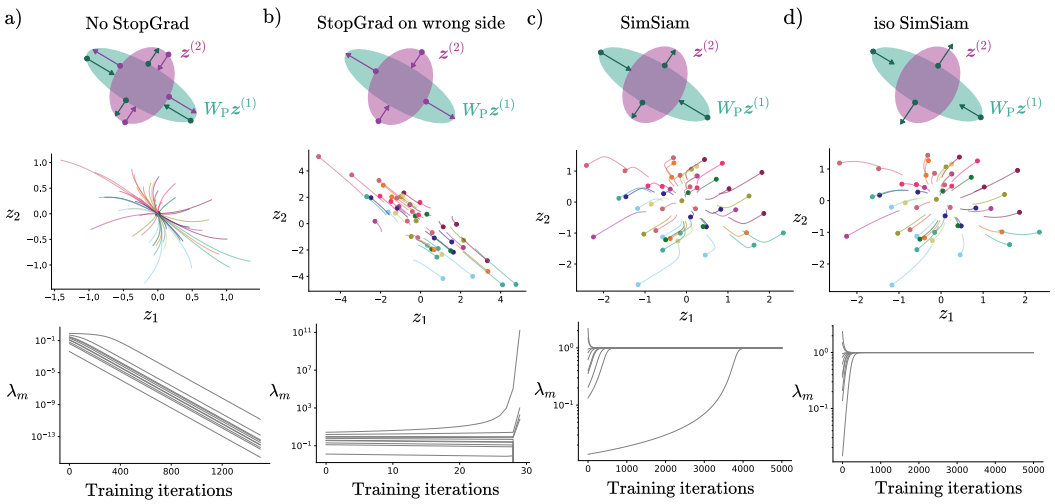

Figure 1: Top and middle rows show the neural updates in different settings, in dimension $M = 2$ for visualization. Bottom row shows the evolution of the eigenvalues of W_P upon training in the settings corresponding to the top row, but in dimensions $N = 15$ and $M = 10$. **a)** Omitting the stop grad leads to representational collapse. **b)** Applying the Stop grad on the wrong side also leads to collapse with potentially diverging eigenmodes. **c)** Optimizing the SimSiam loss leads to isotropic representations. **d)** Optimizing the isotropic loss has the same effect, but uniform for all eigenvalues.

directly $\dot{\hat{\mathcal{Z}}} \propto -\nabla_{\hat{\mathcal{Z}}}\mathcal{L}$ is always positive:

$$\left\langle -\nabla_{\hat{\mathcal{Z}}}\mathcal{L}, \dot{\hat{\mathcal{Z}}} \right\rangle = \eta \left\langle \nabla_{\hat{\mathcal{Z}}}\mathcal{L}, \hat{\Theta}_t \nabla_{\hat{\mathcal{Z}}}\mathcal{L} \right\rangle > 0. \tag{4}$$

Further, in the special case of linear networks with i.i.d Gaussian inputs, the NTK reduces to the identity resulting in perfect alignment (see Appendix A.1 for a proof). Despite the lack of perfect alignment in nonlinear networks, it served as our rationale to study representational dynamics in the idealized setting $\dot{\hat{\mathcal{Z}}} \propto -\nabla_{\hat{\mathcal{Z}}}\mathcal{L}$. In the idealized setting, representational updates decreasing Eq. (2) decouple $\dot{\hat{z}}_m^{(1)} = -\eta \frac{\partial \mathcal{L}}{\partial \hat{z}_m^{(1)}} = \eta \lambda_m^\alpha \left(\hat{z}_m^{(2)} - \lambda_m^\alpha \hat{z}_m^{(1)} \right)$ with learning rate η and in consequence each eigenmode evolves independently in expectation value:

$$\dot{\bar{\hat{z}}}_m^{(1)} = \dot{\bar{\hat{z}}}_m = \eta \lambda_m^\alpha \left(\bar{\hat{z}}_m^{(2)} - \lambda_m^\alpha \bar{\hat{z}}_m^{(1)} \right) = \eta \lambda_m^\alpha \left(1 - \lambda_m^\alpha\right) \hat{z}_m. \tag{5}$$

where we have omitted the time subscript t in $\hat{z}_m$ and λ_m, and $\bar{()}$ is the expectation taken over many random augmentations so that $\bar{\hat{z}}_m^{(1)} = \bar{\hat{z}}_m^{(2)} = \hat{z}_m$. From this expression, we appreciate that $\dot{\hat{z}}_m$ has the same sign as $\hat{z}_m$ whenever $\lambda_m < 1$ and the opposite sign whenever $\lambda_m > 1$, which would subsequently drive the eigenvalue towards one, thereby preventing collapse of the representation.

In contrast, $\dot{\hat{z}}_m$ always has the opposite sign as $\hat{z}_m$ for a loss without the StopGrad function:

$$\mathcal{L}_{\text{noSG}} := \tfrac{1}{2} \| W_P \boldsymbol{z}^{(1)} - \boldsymbol{z}^{(2)} \|^2 \quad \Rightarrow \quad \dot{\hat{z}}_m = -\eta \left(1 - \lambda_m^\alpha\right)^2 \hat{z}_m \quad , \tag{6}$$

which is notorious for causing collapse [3].

Finally, the decoupled loss formulation also predicts the representational behavior when using StopGradient operation on the "wrong" side, i.e, on the online branch instead of the target branch. This manipulation is possible for DirectPred because the predictor needs no optimizing through gradient descent. In this case, the representational dynamics are governed by the following equations:

$$\mathcal{L}_{\text{ws}} := \tfrac{1}{2} \| \text{SG}(W_P \boldsymbol{z}^{(1)}) - \boldsymbol{z}^{(2)} \|^2 \quad \Rightarrow \quad \dot{\hat{z}}_m = \eta \left(\lambda_m^\alpha - 1\right) \hat{z}_m \quad , \tag{7}$$

which leads to collapse for components with $\lambda_m < 1$ and divergence when $\lambda_m > 1$ (Fig. 1b)).

To verify these findings numerically, we simulated a small linear Siamese neural network with input dimension $N = 6$ and representation dimension $M = 2$. We fed the network with isotropic

Gaussian inputs, and pairs of augmentations were generated using isotropic Gaussian perturbations of standard deviation $\sigma = 0.1$. We then trained the linear encoder with the three different loss functions detailed above. Training the network with $\mathcal{L}_{\mathrm{collapse}}$ from Eq. (6) resulted in collapse with exponentially decaying eigenvalues as predicted (Fig. 1a)). Optimizing $\mathcal{L}_{\mathrm{ws}}$ indeed resulted in diverging components with initial $\lambda_m > 1$ and collapse along other dimensions for which $\lambda_m < 1$ at initialization (Fig. 1b)) as predicted by our model. Finally, optimizing $\mathcal{L}_{\mathrm{SimSiam}}$, the representations become increasingly isotropic with all the eigenvalues λ_m converging to one (Fig. 1c)), as predicted by Eq. (5). Importantly, these findings were qualitatively similar in nonlinear networks (see Fig. 2 in the Appendix).

In summary, our eigenspace analysis gives essential insights into the underlying mechanisms of representation learning in Siamese networks with predictor networks and sensible positioning of stop gradients in their loss formulation. First, it illustrates that representational dynamics implicitly drive the eigenvalues of the covariance matrix towards one thereby preventing complete collapse. Second, because collapse is prevented independently for all eigenvalues, the learning dynamics also effectively prevent dimensional collapse [9]. Together, these findings uncover a surprising link to covariance-regularization strategies, which enforce finite output variance and decorrelation [6, 7].

3 Isotropic loss functions speed up and stabilize self-supervised learning

We noted that the current eigenvalues act as effective learning rate modifiers in Eq. (5), meaning that large eigenvalues converge faster than smaller ones, reminiscent of previous theoretical work [11]. We speculated that this effect could lead to slow convergence for small eigenvalue components. However, using our framework it is straightforward to craft alternative "isotropic loss functions" that equalize relaxation dynamics for all eigenmodes. One such loss function is the following:

$$\mathcal{L}_{\mathrm{iso}} = \tfrac{1}{2}\|\boldsymbol{z}^{(1)} - \mathrm{SG}(\boldsymbol{z}^{(2)} + \boldsymbol{z}^{(1)} - W_{\mathrm{P}}\boldsymbol{z}^{(1)})\|^2 \tag{8}$$

with associated learning dynamics $\dot{\hat{z}}_m = \eta\,(1 - \lambda_m^{\alpha})\,\hat{z}_m$. When optimizing $\mathcal{L}_{\mathrm{iso}}$ it resulted in similar convergence properties as the SimSiam loss, but all eigenmodes converged at more similar rates (Fig. 1d)) in agreement with our analysis.

Isotropic losses allow dispensing with moving average parameters in the target network. Previous work by Tian et al. [1] underscored the importance of boosting small eigenvalues and suggested that evolving the weights in the target network using EMA serves as an automatic curriculum for small eigenvalues. We speculated that the EMA may no longer be necessary when training with isotropic losses. To test this idea we trained a deep Siamese network on CIFAR-10 and STL-10 [12] using either the standard SimSiam loss or the isotropic loss introduced in Eq. (8). For these simulations, we used a VGG11 [13] backbone with a one-hidden-layer projection MLP, and SimCLR augmentations [4] using PyTorch Lightning [14, 15].

We first checked that training with IsoLoss $\mathcal{L}_{\mathrm{iso}}$ did not impair training accuracy in settings using an EMA target network. In fact, we found that IsoLoss resulted in slightly better accuracy on STL-10 compared to the regular SimSiam loss. Next we trained the same setup without EMA target network and found that IsoLoss outperformed the SimSiam loss by a large margin.

Table 1: Linear readout validation accuracies $\pm$ stddev over four random seeds.

	CIFAR-10 (800 epochs)		STL-10 (100 epochs)	
	EMA	no EMA	EMA	no EMA
SimSiam loss	74.0 ± 0.6	59.9 ± 2.1	63.8 ± 1.2	33.0 ± 1.9
IsoLoss	72.5 ± 0.4	73.5 ± 0.5	67.1 ± 0.4	65.7 ± 0.8

4 Discussion

Building on DirectPred [1], we presented an eigenspace formulation that illustrates how representational collapse is avoided in Siamese SSL approaches with asymmetric loss configurations. Specifically, we showed that these architectures induce an implicit loss that regularizes the variance

and prevents dimensional collapse. Further, we showed that this formulation allows easily crafting new types of losses, such as IsoLoss that yield faster and more stable learning dynamics, thereby allowing to dispense with EMA target networks. Finally, the eigenspace view laid out in this article builds a conceptual bridge to other SSL approaches that explicitly regularize representational variance through additional loss terms [6, 7] and lays the foundation for better understanding modular architecture choices in terms of their equivalent objective functions.

References

[1] Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning*, pages 10268–10278. PMLR, 2021.

[2] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[3] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

[6] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

[7] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

[8] Xiang Wang, Xinlei Chen, Simon S Du, and Yuandong Tian. Towards demystifying representation learning with non-contrastive self-supervision. *arXiv preprint arXiv:2110.04947*, 2021.

[9] Xiao Wang, Haoqi Fan, Yuandong Tian, Daisuke Kihara, and Xinlei Chen. On the importance of asymmetry for siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16570–16579, 2022.

[10] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[11] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv:1312.6120 [cond-mat, q-bio, stat]*, December 2013. URL `http://arxiv.org/abs/1312.6120`.

[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[14] William Falcon et al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019.

[15] William Falcon and Kyunghyun Cho. A framework for contrastive self-supervised learning and designing a new approach. *arXiv preprint arXiv:2009.00104*, 2020.

A Appendix

A.1 The NTK reduces to the identity for a linear network with high-dimensional i.i.d Gaussian inputs

For a linear network with feedforward weights W, we first note that:

$$\begin{aligned} \hat{\boldsymbol{z}} &= U^\top f(\boldsymbol{x}) = U^\top W \boldsymbol{x} \\ \implies \nabla_\theta \hat{\boldsymbol{z}} &= \nabla_W \hat{\boldsymbol{z}} = \nabla_W \left(U^\top W \boldsymbol{x} \right) = \boldsymbol{x}^\top \otimes U^\top, \end{aligned} \tag{9}$$

where $\otimes$ is the Kronecker product, resulting from the fact that every input component appears in the update once for each output component. Now, we begin our analysis with the gradient flow equation from Eq. (3):

$$\dot{\hat{\boldsymbol{z}}} = \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{z}} \dot{\boldsymbol{\theta}} = -\eta \hat{\Theta}_t(\boldsymbol{x}, \mathcal{X}) \nabla_{\hat{\mathcal{Z}}} \mathcal{L} \tag{10}$$

where $\hat{\Theta}_t(\boldsymbol{x}, \mathcal{X}) = \nabla_\theta \hat{\boldsymbol{z}} \nabla_{\boldsymbol{\theta}} \hat{\mathcal{Z}}^\top$ is the $(M \times M|\mathcal{D}|)$ block row of the full $(M|\mathcal{D}| \times M|\mathcal{D}|)$ empirical NTK $\hat{\Theta}_t(\mathcal{X}, \mathcal{X})$ that corresponds to a single data sample $\boldsymbol{x}$. The diagonal blocks in $\hat{\Theta}_t(\mathcal{X}, \mathcal{X})$ correspond to single samples and the off-diagonal blocks are cross-terms between samples from the full dataset. We can develop a generic expression for each block $\hat{\Theta}_t(\boldsymbol{x}_i, \boldsymbol{x}_j)$ corresponding to the interactions between samples i and j as:

$$\begin{aligned} \hat{\Theta}_t(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \nabla_W \hat{\boldsymbol{z}}_i \nabla_W \hat{\boldsymbol{z}}_j^\top \\ &= \left(\boldsymbol{x}_i^\top \otimes U^\top \right) \left(\boldsymbol{x}_j^\top \otimes U^\top \right)^\top \\ &= \left(\boldsymbol{x}_i^\top \otimes U^\top \right) \left(\boldsymbol{x}_j \otimes U \right) \\ &= \left(\boldsymbol{x}_i^\top \boldsymbol{x}_j \right) \otimes \left(U^\top U \right) \\ &= \left(\boldsymbol{x}_i^\top \boldsymbol{x}_j \right) \otimes I_{\mathrm{M}} \\ &= \left(\boldsymbol{x}_i^\top \boldsymbol{x}_j \right) I_{\mathrm{M}}. \end{aligned} \tag{11}$$

where we have used the fact that $(A \otimes B)^\top = A^\top \otimes B^\top$ and $(A \otimes B)(C \otimes D) = AC \otimes BD$. Here, I_{M} is the identity matrix of size M. We note here that Eq. (11) constitutes a general result that the NTK for a linear network is invariant under orthogonal transformations of the network output.

We now calculate the average interaction between samples drawn from an i.i.d standard Gaussian distribution. In this case, for high-dimensional inputs $\boldsymbol{x}$ we have $\boldsymbol{x}_i^\top \boldsymbol{x}_j \approx \delta_{ij}$. Finally, we see that:

$$\begin{aligned} \dot{\hat{\boldsymbol{z}}}_i &= -\eta \hat{\Theta}_t(\boldsymbol{x}_i, \mathcal{X}) \nabla_{\hat{\mathcal{Z}}} \mathcal{L} \\ &= -\eta \hat{\Theta}_t(\boldsymbol{x}_i, \boldsymbol{x}_i) \nabla_{\hat{\boldsymbol{z}}_i} \mathcal{L} - \eta \sum_{j \neq i} \hat{\Theta}_t(\boldsymbol{x}_i, \boldsymbol{x}_j) \nabla_{\hat{\boldsymbol{z}}_j} \mathcal{L} \\ &= -\eta \left(\boldsymbol{x}_i^\top \boldsymbol{x}_i \right) \nabla_{\hat{\boldsymbol{z}}_i} \mathcal{L} - \eta \sum_{j \neq i} \left(\boldsymbol{x}_i^\top \boldsymbol{x}_j \right) \nabla_{\hat{\boldsymbol{z}}_j} \mathcal{L} \\ &= -\eta \nabla_{\hat{\boldsymbol{z}}_i} \mathcal{L} \end{aligned} \tag{12}$$

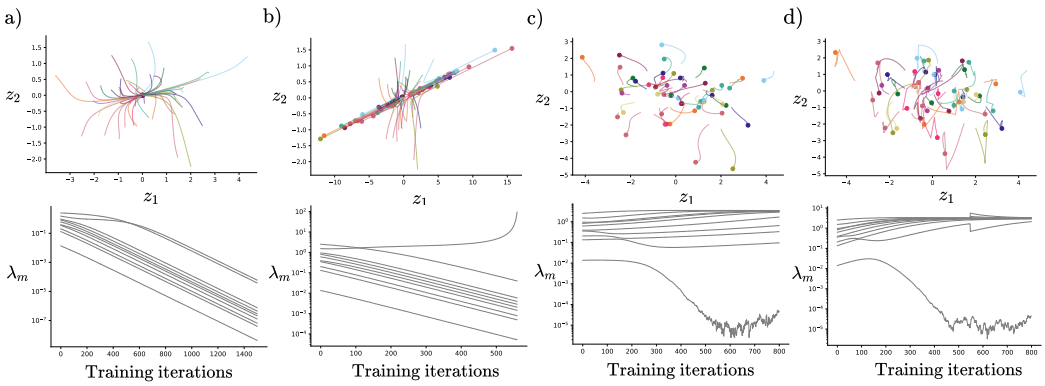

Figure 2: Same as Fig. 1, but for a nonlinear network using rectified linear units.