Implicitly Learned Invariance and Equivariance in Linear Regression

Yonatan Gideoni[1]

Abstract

Can deep learning models generalize if their problem's underlying structure is unknown *a priori*? We analyze this theoretically and empirically in an idealistic setting for linear regression with invariant/equivariant data. We prove that linear regression models learn to become invariant/equivariant, with their weights being decomposed into a component that respects the symmetry and one that does not. These two components evolve independently over time, with the asymmetric component decaying exponentially given sufficient data. Extending these results to more complex systems will be pursued in future work.[1]

1. Introduction

It is often believed that a deep learning model cannot adequately solve a problem without utilizing its underlying structure (Bronstein et al., 2021). Concretely, if a problem has a certain symmetry then an architecture that does not respect it will do worse than one that does. This line of thought has led to much work on discovering symmetries in data (Krippendorf & Syvaeri, 2020; Dehmamy et al., 2021; Desai et al., 2021; Yang et al., 2023) and to the design of architectures that respect them, usually through invariance or equivariance (Cohen & Welling, 2016; Finzi et al., 2020; van der Ouderaa & van der Wilk, 2022).

However, many such beliefs in deep learning were later found not to hold. For example, neural networks were thought to have no chance of generalizing because they are always overparameterized, yet this was disproven by phenomena like double descent (Belkin et al., 2018; Nakkiran et al., 2019). This begs the question—can a task be adequately learned without having its symmetries encoded into the architecture?

[1]Department of Computer Science and Technology, University of Cambridge. Correspondence to: Yonatan Gideoni <yg403@cl.cam.ac.uk>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

[1]Code is given at https://github.com/YonatanGideoni/ImplLearntSymmLinReg.

In this work we take a modest step in answering this question, with **our contributions being as follows**. To make the analysis tractable we analytically examine linear regression's training dynamics for data that is invariant/equivariant to a symmetry. Its weights are found to decompose into a symmetric and asymmetric component, with the former being invariant/equivariant to the symmetry. The asymmetric component is found to exponentially decay given sufficient data, making the final predictor respect the symmetry. Proofs and additional findings are in the appendix.

2. Preliminaries

Representation theory: A familiarity with basic group theory is assumed. In certain cases, abstract notions regarding groups can be conveniently expressed using vector spaces. A *group representation* is a homomorphism $\rho : G \to GL_n(\mathbb{R})$ such that $\rho(gh) = \rho(g)\rho(h)$, with $GL_n(\mathbb{R})$ being the group of invertible matrices over $\mathbb{R}^n$. A representation makes the group's operation become standard matrix multiplication, with the elements of X being vectors. Throughout this paper only such groups will be considered, having ρ be implicit.

Invariance and equivariance: Often it is desirable that functions applied to a symmetric object preserve its symmetry. This is formalized through invariance and equivariance. A function is invariant to a group if acting on the input does not affect the function's output—$f(gx) = f(x)$. Equivariance accordingly means that the output changes with the input, such that $gf(x) = f(gx)$.

Gradient flow and linear regression: When training models with gradient descent the weights' w discrete update rule is $w_{i+1} = w_i - \eta\nabla_w L$, where η is the learning rate and L is the loss. By taking η to be infinitesimal this becomes continuous, such that instead of gradient descent the weights evolve as $\dot{w} = -\nabla_w L$. This is known as *gradient flow*.

This flow can sometimes yield analytical results for the weights' trajectory over time. For example, this occurs for linear regression with a squared error loss, $L = \frac{1}{2}||y - Xw||^2$, with y being the labels, X the design matrix, and w the weights. The total instead of mean loss is taken for convenience, as scaling the loss does not change the weights' trajectory. Under gradient flow the weights over time are known to be

$$w(t) = e^{-Kt}w_0 + (I - e^{-Kt})K^\dagger X^T y, \qquad (1)$$

where $K := X^T X$ is the uncentered empirical covariance matrix, $K^\dagger$ is its pseudo-inverse, and w_0 is the initial weights vector. The exponentials are matrix exponents.

3. Learned Invariance

To understand the effect symmetries in the data have on training dynamics we start by analyzing an idealistic noiseless setting. For simplicity, we start with the invariant case.

Let $f_w(x) = xw$ be a linear predictor, where x is a row vector for notational convenience, and let $L = \frac{1}{2}||y - Xw||^2$ be the loss. The total squared error instead of the mean squared error is used for convenience, as a constant factor only affects the speed with which the weights evolve and not their trajectory. We assume the following:

Assumption 1. (Data Invariance) There exists a linear group G such the dataset only contains entire orbits having the same labels. Thus, for all $g \in G$ both (x, y) and (gx, y) are in the dataset.

Example 3.1. When trying to fit a straight 1D line the features are $\begin{pmatrix}1 & x\end{pmatrix}^T$. An implicit reflection symmetry implies that both (x, y) and $(-x, y)$ exist in the dataset. In this case the group is $G := \left\{ I_2, \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \right\}$, such that the second group element represents the reflection $x \to -x$.

Corollary 3.2. *Assumption 1 allows rewriting the loss as*

$$L = \frac{1}{2}\sum_{g \in G} ||y - Xgw||^2 \qquad (2)$$

where we now take a single point over each orbit and explicitly sum over it. g acts from the right because it acts on the design matrix' features, which are its columns.

Assumption 2. (Gradient Flow) We assume that training is done using gradient flow, such that the learning rate η is sufficiently small.

The following technical assumption is required for parts of our proofs. It holds for many common groups, including rotations, permutations, and reflections. A case where it does not hold is given in Appendix A.

Assumption 3. For all $g \in G$, g^T is also in the group.

These assumptions together yield the following result:

Theorem 3.3. *(Induced Invariance in Linear Regression) Given these three assumptions, the following holds:*

1. *The weights over time are:*

$$w(t) = e^{-K_G t}w_0 + (I - e^{-K_G t})K_G^\dagger R^T X^T y, \qquad (3)$$

where w_0 is the initial weights vector, $K_G := \sum_{g \in G} g^T X^T X g$, and $R := \sum_{g \in G} g$. R is an unnormalized Reynolds operator (Billik & Rota, 1960). $K_G^\dagger$ is the pseudoinverse of K_G.

2. *The weights can be decomposed as $w = w_r + w_k$, where $w_k \in \ker(R)$ and $w_r \in \ker(R)^\perp$. These two components are decoupled from one another—the value of one does not affect the dynamics of the other.*

3. *w_k does not depend on the labels y and thus appears only in the first term of Equation 3. By definition, K_G is positive semi-definite (PSD), being a sum of PSD matrices. If it is positive definite (PD) then w_k will exponentially decay, regardless of the initial weights.*

Intuitively, the weights decompose into a part that respects the symmetry, w_r, and a part that does not, w_k. As the weights in linear regression converge to a point spanned by the data (Bishop & Nasrabadi, 2006, p.163), if it spans the entire space then K_G will be PD. This is because there will be no directions along which the data has zero variance. As all the group elements g are invertible, K_G is PD iff $X^T X$ is PD. In this case, we arrive at the following result:

Corollary 3.4. *If K_G is PD, $w^* := w(t \to \infty)$ will be invariant to the symmetry, so $\forall g \in G : f_{w^*}(x) = f_{w^*}(gx)$.*

Thus, although the initial model had no information about the symmetry, it becomes invariant after training. A different, restricted version of this was proven by Lyle et al. (2020).

4. Learned Equivariance

A similar albeit weaker version of these results holds when the data is equivariant to a given symmetry. To meaningfully discuss equivariance we train a function f that maps the domain onto itself, giving our linear predictor the form $f_W(x) = xW$, with W being a matrix and x a row vector. Accordingly, the labels are now vectors instead of scalars. Denoting the label matrix as Y, the loss now becomes $L = \frac{1}{2}||Y - XW||^2_{Fr}$, where $|| \cdot ||_{Fr}$ is the Frobenius norm.

In this case Assumptions 1, 3 and Corollary 3.2 are replaced by the following:

Assumption 4. (Data Equivariance) There exists a linear group G such the dataset only contains entire orbits having equivariant labels. Thus, for all $g \in G$ both (x, y) and (gx, gy) are in the dataset.

Assumption 5. The group G is orthogonal, such that $gg^T = g^T g = I$ and $\forall g \in G : g^T \in G$.

Corollary 4.1. *Assumption 5 allows rewriting the loss as*

$$L = \frac{1}{2}\sum_{g \in G} ||Yg - XgW||^2_{Fr}. \qquad (4)$$

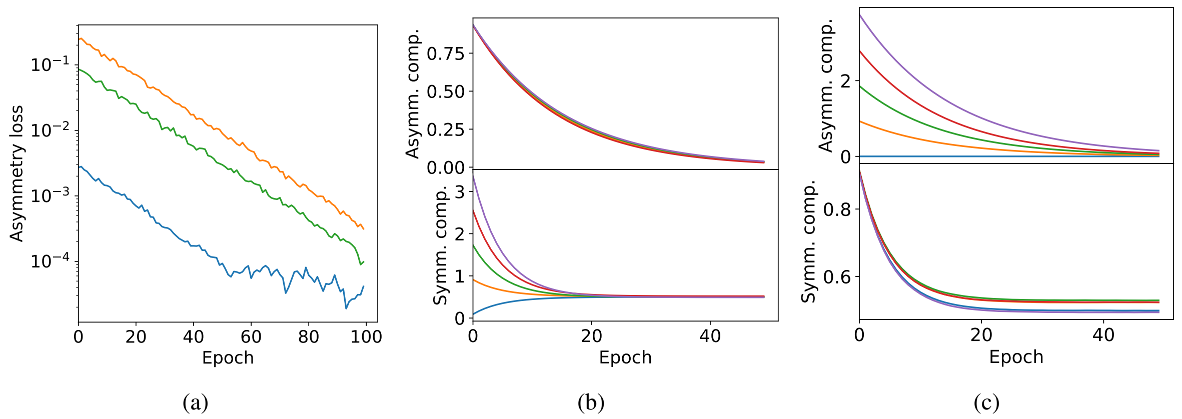

Figure 1. Empirical verifications of Theorem 3.3 on the task given in Example 3.1. Each line/color corresponds to a differently initialized model. (a) Shows the predictions' antisymmetric part decaying exponentially until the gradient flow assumption breaks down, as per Theorem 3.3.3. (b) and (c) show that the components that respect/disrespect the symmetry are decoupled from one another, as per Theorem 3.3.2. In (b) the asymmetric components have the same initialization while the symmetric are different, with the opposite case given in (c).

These yield the following results:

Theorem 4.2. *(Induced Equivariance in Linear Regression) Given these assumptions and defining $C_G := \sum_{g\in G} g^T X^T Y g$, the following hold:*

1. *Given initial weights W_0, the weights over time are:*

$$W(t) = e^{-K_G t}W_0 + (I - e^{-K_G t})K_G^{\dagger} C_G. \qquad (5)$$

2. *The weights can be decomposed as $W = W_r + W_k$, where W_r is in the subspace of matrices that commute with the group elements, defined as $V_G = \{A | \forall g \in G : [g, A] = 0\}$, and W_k is in $V_G^{\perp}$. These two components are decoupled from one another.*

3. *W_k does not depend on the labels Y and thus appears only in the first term of Equation 5. Thus, if K_G is PD then W_k will always decay exponentially over time, regardless of the initial weights.*

Corollary 4.3. *If K_G is PD then $W^* := W(t \to \infty)$ commutes with the group elements. This results in f_{W^*} being equivariant, such that $\forall g \in G : f_{W^*}(gx) = g f_{W^*}(x)$.*

Like before, although no knowledge of the symmetry was known in advance, the final model becomes equivariant.

5. Experiments

We verify our findings for the learned invariance case using several artificial datasets. The simple case given in Example 3.1 of learning a reflection invariant function such as $|x|$ with the features being $\begin{pmatrix}1 & x\end{pmatrix}^T$ is shown in Figure 1, with other experiments and findings detailed in Appendix D. The asymmetry loss is a measure of how non-invariant the model is and is defined as $\mathbb{E}_x[|f(x) - \mathbb{E}_g[f(gx)]|]$, where the expectation over the group elements is with respect to the Haar measure. Empirical verifications for learned equivariance are given in Appendix E.

6. Wide Neural Networks

As wide neural networks are approximated by their linearised versions, one would naïvely expect these results to extend to them. However, this turns out not to be the case—a thorough discussion is given in Appendix F, with a thorough analysis being left for future work.

7. Discussion

We analyzed whether invariance and equivariance can be implicitly learned in an idealized setting. We prove that this generally occurs in linear regression, where given sufficient data the final predictor becomes perfectly invariant/equivariant.

It is interesting to consider whether this occurs in more complex models, such as neural networks. Olah et al. (2020) qualitatively and empirically observed approximately equivariant features being learned in images. Gruver et al. (2022) empirically showed that some pretrained models with fine-tuning can reach approximate equivariance on par with architectures that have it baked in. These results, although mostly empirical and qualitative, are interesting—they imply this could occur for neural networks as well. If so, this begs the question—how much should we invest in encoding such priors into architectures? This question is relevant only given sufficient data, as for low-data regimes it is unlikely this would occur. Extending this analysis to more complex systems is an interesting avenue for future work.

Acknowledgements

In no particular order, I would like to thank Challenger Mishra, Francisco Vargas, Aditya Ravuri, Kamil Bujel, Theodore Long, Jonas Jürß, and Dulhan Jayalath for helpful comments and discussions.

References

Belkin, M., Hsu, D. J., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116:15849 – 15854, 2018.

Billik, M. and Rota, G.-C. On reynolds operators in finite-dimensional algebras. *Indiana University Mathematics Journal*, 9:927–932, 1960.

Bishop, C. M. and Nasrabadi, N. M. Pattern recognition and machine learning. *J. Electronic Imaging*, 16:049901, 2006.

Bronstein, M. M., Bruna, J., Cohen, T., and Velivckovi'c, P. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *ArXiv*, abs/2104.13478, 2021.

Chizat, L., Oyallon, E., and Bach, F. R. On lazy training in differentiable programming. In *Neural Information Processing Systems*, 2018.

Cho, Y. and Saul, L. K. Kernel methods for deep learning. In *NIPS*, 2009.

Cohen, T. and Welling, M. Group equivariant convolutional networks. In *International conference on machine learning*, pp. 2990–2999. PMLR, 2016.

Daniely, A., Frostig, R., and Singer, Y. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *NIPS*, 2016.

Dehmamy, N., Walters, R., Liu, Y., Wang, D., and Yu, R. Automatic symmetry discovery with lie algebra convolutional network. In *Neural Information Processing Systems*, 2021.

Desai, K., Nachman, B. P., and Thaler, J. Symmetry discovery with deep learning. *Physical Review D*, 2021.

Fan, Z. and Wang, Z. Spectra of the conjugate kernel and neural tangent kernel for linear-width neural networks. *ArXiv*, abs/2005.11879, 2020.

Finzi, M., Stanton, S., Izmailov, P., and Wilson, A. G. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *International Conference on Machine Learning*, pp. 3165–3176. PMLR, 2020.

Gruver, N., Finzi, M., Goldblum, M., and Wilson, A. G. The lie derivative for measuring learned equivariance. *ArXiv*, abs/2210.02984, 2022.

Hu, Z. and Huang, H. On the random conjugate kernel and neural tangent kernel. In *International Conference on Machine Learning*, 2021.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: convergence and generalization in neural networks (invited paper). *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, 2018.

Krippendorf, S. and Syvaeri, M. Detecting symmetries with neural networks. *Machine Learning: Science and Technology*, 2, 2020.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. N. Deep neural networks as gaussian processes. *ArXiv*, abs/1711.00165, 2017.

Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Novak, R., Sohl-Dickstein, J. N., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. *Journal of Statistical Mechanics: Theory and Experiment*, 2020, 2019.

Lyle, C., van der Wilk, M., Kwiatkowska, M. Z., Gal, Y., and Bloem-Reddy, B. On the benefits of invariance in neural networks. *ArXiv*, abs/2005.00178, 2020.

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. Deep double descent: where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021, 2019.

Olah, C., Cammarata, N., Voss, C., Schubert, L., and Goh, G. Naturally occurring equivariance in neural networks. *Distill*, 2020. doi: 10.23915/distill.00024.004. https://distill.pub/2020/circuits/equivariance.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *NIPS*, 2007.

van der Ouderaa, T. F. A. and van der Wilk, M. Learning invariant weights in neural networks. *ArXiv*, abs/2202.12439, 2022.

Yang, J., Walters, R., Dehmamy, N., and Yu, R. Generative adversarial symmetry discovery. *ArXiv*, abs/2302.00236, 2023.

A. Assumption 3 counterexample

Note that $\left\{ \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix} \middle| a \in \mathbb{R} \right\}$ is a group which describes symmetry to 1D translations of a point x given as $\begin{pmatrix} 1 \\ x \end{pmatrix}$. Clearly, for $a \neq 0$ the transposed matrices $\begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}$ are not in it.

B. Proof of Induced Invariance

Proof of 3.3.1: Assuming gradient flow, by taking the gradient of the loss we find the weights' dynamics to be:

$$\dot{w} = -\nabla_w L = \sum_g g X^T \left(y - X g^T w \right). \tag{6}$$

This is a linear matrix differential equation with constant coefficients. The particular solution is $K_G^\dagger R X^T y$ and the general solution is $e^{-K_G t}\tilde{w}$ for some constant $\tilde{w}$. Requiring that $w(t = 0) = w_0$ yields the solution given in the theorem. □

Proof of 3.3.2: The weights' decomposition can always be carried out, as the direct sum of a subspace (here $\ker(R)$) and its perpendicular subspace span the entire space. Using this decomposition for Equation 6 yields the single equation for both components:

$$\dot{w}_r + \dot{w}_k = -\nabla_w L = R^T X^T y - K_G(w_r + w_k). \tag{7}$$

As $\ker(R)^\perp = \mathrm{Im}(R^T)$, the first term on the RHS is wholly contained in $\mathrm{Im}(R^T)$ and thus only affects w_r. As for the second term, we will show that it induces no mixing between the subspaces, such that $K_G w_r \in \ker(R)^T$ and $K_G w_k \in \ker(R)$. For the latter case, note that:

$$\forall v \in \ker(R) : R K_G v = \sum_g R g^T X^T X g v. \tag{8}$$

Here Assumption 3 becomes necessary. Note that $Rg = gR = R$ as acting on R with a group element simply permutes the elements' order in the sum. As $g^T \in G$, we have that $\sum_g R g^T X^T X g v = \sum_g R X^T X g v = R X^T X R v$, and as $v \in \ker(R)$ this is zero, proving this case.

Similarly, one can show that $\forall v \in \mathrm{Im}(R^T) : K_G v \in \mathrm{Im}(R^T)$. As $v \in \mathrm{Im}(R^T)$, $\exists u : v = R^T u$. Thus, $K_G v = K_G R^T u = \sum_g g^T X^T X g R^T u$. Again, $R^T g = g R^T = R^T$. Therefore, $\sum_g g^T X^T X g R^T u = \sum_g g^T X^T X R^T u = R^T X^T X g R^T u \in \mathrm{Im}(R^T)$.

As the first term in Equation 3 is wholly in one of the subspaces and the second term does not mix the subspaces, they each evolve independently over time. □

Proof of 3.3.3: This follows directly from our results in the previous section. They imply that w_k's evolution over time is per $\dot{w}_k = -K_G w_k$, resulting in $w_k(t) = e^{-K_G t} w_{0k}$. The theorem's statement directly follows. □

As for Corollary 3.4, this immediately follows from the theorem—if K_G is PD at $t \to \infty$ the first term in Equation 3 will be zero. As this term is the one that is in the symmetry-disrespecting subspace, the final model will be invariant to the symmetry.

C. Proof of Induced Equivariance

The proof of 4.2.1 is identical to that of 3.3.1, just with the weights being a matrix instead of a vector and the labels defined differently.

Proof of 4.2.2: We will prove this in similar fashion to 3.3.2. It is easy to see that V_G forms a vector space. The decomposition is valid because of identical reasons, and results in the weights' components' dynamics being

$$\dot{W}_r + \dot{W}_k = C_G - K_G(W_r + W_k). \tag{9}$$

Note that C_G commutes with the group elements as $\forall g' \in G : C_G g' = \sum_g g^T X^T Y g g'$, so by defining $g = g'\tilde{g}$ one can readily see that $C_G g' = g' C_G$. Thus, the first term is wholly contained in V_G. As for the second, we shall show that it does not mix the subspaces. As a product of commuting matrices commutes, $K_G W_r \in V_G$.

To show that $K_G W_k \in V_G^\perp$ we note that $\forall C \in V_G : V^T \in V_G$ because the group is orthogonal, so $[C, g] = 0 \Leftrightarrow [C^T, g^T] = 0$ and the first equation holds also for $g = g'^T$. Moreover, $K_G \in V_G$ as it commutes with the group elements, exactly like C_G. Thus, note that $\forall C \in V_G : \mathrm{Tr}(C^T K_G W_k) = \mathrm{Tr}((K_G C)^T W_k)$, and as $K_G C \in V_G$ as a product of commuting matrices, this is zero. Thus, $K_G W_k \in V_G^\perp$.

Therefore, the dynamics do not mix the subspaces, so W_r, W_k evolve independently of one another. □

Proof of 4.2.3: This follows directly from the previous results. W_k over time will be $W_k(t) = e^{-K_G t} W_{k0}$, so the statement directly follows. □

D. Learned Invariance Experiments

Given sufficient data, these results hold when the symmetry is encoded in the data only on average and not as detailed in Assumptions 1 and 4, where each orbit is precisely encoded in the input. Figure 2 shows this, where for a dataset with only 20 points with Assumption 1 not being enforced the asymmetry loss still exponentially decreases.

Moreover, the results are found to hold even when the data has some level of noise. Figure 3 shows that when the data

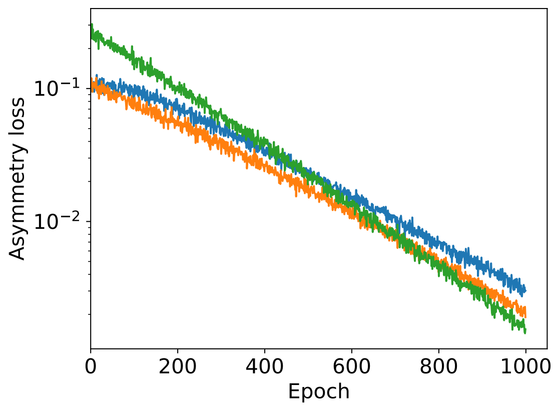

Figure 2. Learning $x + y$ given x, y as inputs, with the symmetry being $x \leftrightarrow y$. Here there are only 20 points in the input, with Assumption 1 not being enforced. The asymmetry loss exponentially decreases also in this case. The number of epochs was increased to compensate for the smaller dataset.

has Gaussian noise with a standard deviation as high as 0.5, the asymmetry loss still exhibits its typical exponential decay. As the asymmetric weights do not depend on the labels, some level of robustness to noise is unsurprising.

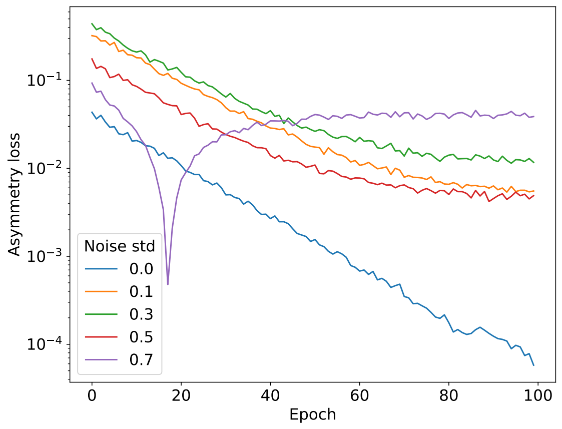

Figure 3. Learning $|x|$ with various levels of Gaussian noise. Standard deviations as high as 0.5 all exhibit relatively typical asymmetry loss curves.

E. Learned Equivariance Experiments

To verify Theorem 4.2 we use a setting similar to that in the previous Appendix. We train a linear predictor with the same hyperparameters used for the learned invariance experiments. This is done over a dataset with an $x \leftrightarrow y$ permutation symmetry, with the labels being $\begin{pmatrix} y^2 & x^2 \end{pmatrix}^T$. The asymmetry loss for this setup is shown in Figure 4.

Note that here the symmetric components of a matrix $\begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix}$ for a $x \leftrightarrow y$ symmetry are $\frac{w_{11}+w_{22}}{2}$ and $\frac{w_{12}+w_{21}}{2}$. The asymmetric components are naturally thus $\frac{w_{11}-w_{22}}{2}$ and $\frac{w_{12}-w_{21}}{2}$.

F. Wide Neural Networks

As sufficiently wide neural networks (NNs) are well approximated by linear models (Lee et al., 2019), one would expect these results to be seen there as well. However, in practice NNs are known to struggle when learning data with symmetries if not given suitable geometric priors (Bronstein et al., 2021, p.9). This raises the question, why don't NNs exhibit implicitly learned invariance and equivariance?

Formally, when the hidden layers' width approaches infinity the network $f_w(x)$ is well approximated by its linearized version (Jacot et al., 2018; Chizat et al., 2018; Lee et al., 2019),

$$f_t^{\text{lin}}(x) = f_0(x) + \nabla_w f_0(x)|_{(w=w_0)} \Delta w, \qquad (10)$$

where f_0 is the network at initialization and $\Delta w := w(t) - w_0$. This limit holds regardless of depth. As the weights are symmetrically distributed, in the infinite width case f_0 approaches 0 for all inputs.

Another approach to the infinite width limit is given by the Conjugate Kernel (Cho & Saul, 2009; Daniely et al., 2016; Lee et al., 2017; Hu & Huang, 2021), a kernel describing the covariance matrix after a certain layer. Explicitly it is $X_l^T X_l$, where $X_l \equiv f_l(X)$ are the outputs after the l-th layer. At infinite width this kernel is known to describe the linear regression model $X_l w_{l+1}$, where w_{l+1} are the $l+1$-th layer's weights (Rahimi & Recht, 2007).

In both cases the linearized neural network acts as a feature kernel, transforming the input data to some high-dimensional space.

F.1. Kernelized Group Representations

Because the network acts as a kernel when linearized, one could argue that this work's main theorems do not hold as it seems that the group element acting on the input x would not translate to it acting on the weights.

However, this does not rule out the option of the kernelized features having a different group representation than the original ones. If $\phi(x)$ is the kernel and $\tilde{g}$ is a group element's representation with respect to the kernelized features, this implies that $\tilde{g}\phi(x) = \phi(gx)$. This can occur for standard kernels, as is illustrated in the following example.

Example F.1. Assume that a given linear regression problem's features are $\begin{pmatrix} x & y \end{pmatrix}^T$ and that there is a $x \leftrightarrow y$ permutation symmetry. Using cycle notation, the group acting on the features is $G := \{I_2, (1\ 2)\}$. If a sec-

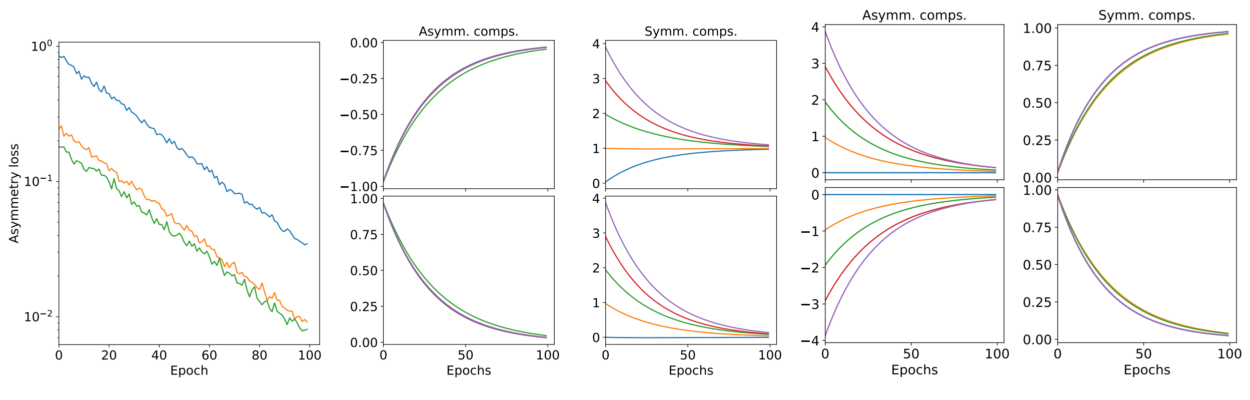

Figure 4. Empirical verifications of Theorem 4.2 on the given task. As each matrix has $2 \times 2 = 4$ components there are 2 symmetric/asymmetric components respectively. (a) Shows that the predictions' antisymmetric part decays exponentially, as per Theorem 4.2(c). (b) and (c) show that the components that respect/disrespect the symmetry are decoupled from one another, as per Theorem 4.2(b). In (b) the asymmetric components are initialized to be identical while the symmetric are different, with the opposite case given in (c).

ond order polynomial kernel is used to get the features $\begin{pmatrix} x & y & x^2 & xy & y^2 \end{pmatrix}^T$ the group's action on these new features is given by $\tilde{G} \coloneqq \{I_5, (1\,2)(3\,5)\}$.

We empirically verify this holds when the network acts as a kernel. For every group element a matrix $\tilde{g}$ is found, such that $\nabla_w f_0(gx) \approx \tilde{g}\nabla_w f_0(x)$ if linearizing with respect to the weights and where $\tilde{g}x_l = f_l(gx)$ if using the Conjugate Kernel (CK). The results and how $\tilde{g}$ is found are detailed in Appendix G. As this shows that the group's action approximately translates to the kernelized features, the theorems from the previous section hold for the linearized network.

F.2. Difficulties in Learned Invariance/Equivariance

A crucial requirement for the invariance/equivariance to be learned is that there is sufficient data. This is defined based on if K_G is PD, which requires there to be at least as many samples as there are features. In the linearized NN case the number of features is the network's number of parameters. As the linearized NNs predictor is as per Equation 10, wherever X^TX appeared in the previous results it should be replaced with $\nabla_w f_0(X)^T \nabla_w f_0(X)$, which we shall call the empirical weights kernel (EWK). Note that this is not the empirical tangent kernel given in Jacot et al. (2018) and Lee et al. (2019) as it acts on the weights instead of the inputs. Here f_0 is interpreted as acting row-wise on the design matrix X's inputs. The EWK's rank is at most the number of samples, such that if the network is overparameterized it will necessarily be singular. This results in K_G also being singular, preventing the symmetry from being fully learned. Intuitively, because the data does not span the feature space the system is underdetermined, therefore permitting suboptimal solutions with respect to the symmetry.

These conclusions stay the same when approximating the linearized network using the CK, with the number of relevant parameters being reduced to the last layer's width. Here X^TX should be replaced with $X_L^T X_L$, with L being the network's number of layers.

However, this still does not rule out two cases that could allow circumventing these limitations. First, if the network is underparameterized, and second, if the weights are initialized to be in a direction that K_G spans. Both of these regimes are difficult to realize because the EWK and CK's spectrums are dominated by relatively small eigenvalues. This has been shown for the CK by Fan & Wang (2020), while we demonstrate it empirically for the EWK in Appendix I. Even while these kernels are not singular, the abundance of small eigenvalues makes the majority of directions in weight space almost degenerate.

While this poses a difficulty, initializing the weights to be in a non-degenerate direction should still be possible. However, if the kernels change during training such that the weights are partially contained within a degenerate direction then this will not hold. As Fan & Wang (2020) have shown, this is indeed the case—even for networks with widths on the order of $10^2 - 10^3$ the CK's eigenvalues noticeably shift during training. Failed empirical attempts to induce learned invariance are detailed in Appendix H.

G. Finding Kernelized Representations

We want to find a matrix $\tilde{g}$ such that $\tilde{g}f(x) \approx f(gx)$. We do this by minimizing the squared error $|\tilde{g}f(x) - f(gx)|_2^2$, where as $f(x), f(gx)$ are known this can be solved in closed form. When given a design matrix X as input, the minimizer is $\tilde{g} = (f(X)^T f(X))^\dagger f(X)^T f(Xg)$. The resulting errors are given in Table 1 for the network acting on data with an S_3 symmetry.

H. Attempts at Inducing Invariance

While the results in Appendix F pose a difficulty, initializing the weights to be in a non-degenerate direction should still be possible. One would expect this to allow the network to learn the symmetry as then the asymmetric weights would fully decay. However, if the kernels change during training such that the weights are partially contained within a degenerate direction then this will not hold. As Fan & Wang (2020) have shown, this is indeed the case—even for networks with widths on the order of $10^2 - 10^3$ the CK's eigenvalues noticeably shift during training.

However, there are more ways to induce learned invariance in networks. Chizat et al. (2018) show that lazy training, where a network barely deviates from its linearised version's dynamics, can occur in narrow networks. This results from the loss and network's outputs being scaled in a certain manner. While this could theoretically make even finite width, underparameterised networks have dynamics similar to those of their infinite width versions, following their procedure did not induce learned invariance in practice. This may be because it is harder to find the group's action on the induced kernel when the networks have a low width, as there are fewer parameters to optimise. Further understanding this subject is an interesting avenue for future work.

I. EWK Spectrum

The empirical spectrum of the EWK is given in Figure 5. We see that a significant portion of the eigenvalues are small, showing that many directions in weight-space would thus be degenerate and difficult to learn. While a scaling of the features would increase the eigenvalues, it would not change the many orders of magnitude difference between them.

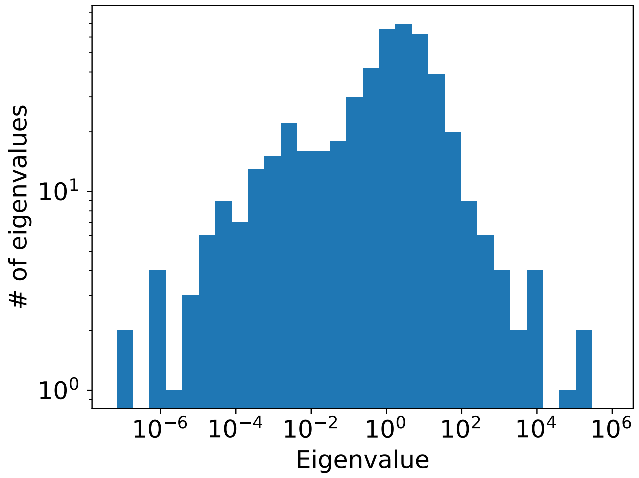

Figure 5. Empirical spectrum of EWK eigenvalues. The eigenvalues that are less than 0.2, which is about a third of them, would require more than 1000 epochs with a learning rate of 10^{-2} with full-batch gradient descent to have their corresponding weights decay by 90%. Zero eigenvalues are not displayed, with them accounting for 2% of all the eigenvalues. This is for an MLP with a single hidden layer with 100 neurons and 10^5 points from $[0, 1]^3$, such that the network has 501 parameters and is underparameterized. The EWK is in this case accordingly a 501×501 matrix.

S_3 group elements / Group kernelisation error	(1 2)	(1 2 3)
$\mathbb{E}\left[\lvert\nabla_w f(x) - \nabla_w f(gx)\rvert\right]$	$1.1 \cdot 10^{-2}$	$1.3 \cdot 10^{-2}$
$\mathbb{E}\left[\lvert\tilde{g}\nabla_w f(x) - \nabla_w f(gx)\rvert\right]$	$2.4 \cdot 10^{-3}$	$1.0 \cdot 10^{-3}$
$\mathbb{E}\left[\lvert x_l - f_l(gx)\rvert\right]$	$1.4 \cdot 10^{-2}$	$1.5 \cdot 10^{-2}$
$\mathbb{E}\left[\lvert\tilde{g}x_l - f_l(gx)\rvert\right]$	$8.1 \cdot 10^{-4}$	$8.1 \cdot 10^{-4}$

Table 1. Error induced when using a kernelized representation of the group elements relative to baselines. Absolute values represent L_2 norms. The first two columns are when linearizing the network relative to the weights while the last two are when using the CK. Finding approximate kernelized representations of the group action reduces the error by factors of 5-20 relative to the baselines. This is done for an MLP with one hidden layer with 100 neurons. For the CK x_l is a vector of the last post-activations.