Divergence at the Interpolation Threshold: Identifying, Interpreting & Ablating the Sources of a Deep Learning Puzzle

Rylan Schaeffer
Computer Science
Stanford
rschaef@stanford.edu

Zachary Robertson
Computer Science
Stanford
zroberts@stanford.edu

Akhilan Boopathy
EECS
MIT
akhilan@mit.edu

Mikail Khona
Physics
MIT
mikail@mit.edu

Ila Rani Fiete
BCS
MIT
fiete@mit.edu

Andrey Gromov
Physics & FAIR
UMD & Meta
gromovand@meta.com

Sanmi Koyejo
Computer Science
Stanford
sanmi@cs.stanford.edu

Abstract

Machine learning models misbehave, often in unexpected ways. One prominent misbehavior is when the test loss diverges at the interpolation threshold, perhaps best known from its distinctive appearance in double descent. While considerable theoretical effort has gone into understanding generalization of overparameterized models, less effort has been made at understanding why the test loss misbehaves at the interpolation threshold. Moreover, analytically solvable models in this area employ a range of assumptions and use complex techniques from random matrix theory, statistical mechanics, and kernel methods, making it difficult to assess when and why test error might diverge. In this work, we analytically study the simplest supervised model - ordinary linear regression - and show intuitively and rigorously when and why a divergence occurs at the interpolation threshold using basic linear algebra. We identify three interpretable factors that, when all present, cause the divergence. We demonstrate on real data that linear models' test losses diverge at the interpolation threshold and that the divergence disappears when we ablate any one of the three identified factors. We conclude with insights on recent discoveries in nonlinear models regarding superposition and double descent.

1 Introduction

Machine learning models, while incredibly powerful, can sometimes act unpredictably. One intriguing behavior is when the test loss suddenly diverges at the interpolation threshold: the point where the model perfectly fits the training data, achieving zero training error. This phenomenon is distinctly observed in the double descent curve [5]. Although much theoretical work has been done to comprehend generalization of overparameterized models [27, 18, 8, 26, 5, 4, 6, 17, 21, 2, 15, 1, 24, 22, 23, 16, 11, 3], a comprehensive understanding of why test loss behaves erratically at this threshold remains elusive. Many analytical models rely on a plethora of assumptions (e.g., i.i.d additive Gaussian noise, sub-Gaussian covariates, $(8 + m)$-moments) and use advanced proof techniques from random matrix theory, statistical mechanics, and kernel methods. This complexity muddies the waters, making it challenging to pinpoint the precise conditions leading to test error divergence. For instance, a recent study on toy nonlinear autoencoders by Anthropic unveiled a divergence even in the absence of noise [12], an assumption that many theories relied upon [4, 15, 6, 11, 16, 3]. This unexpected outcome prompts the question: with all this theory, should we have anticipated the result?

1st Workshop on Attributing Model Behavior at Scale. NeurIPS 2023 workshop.

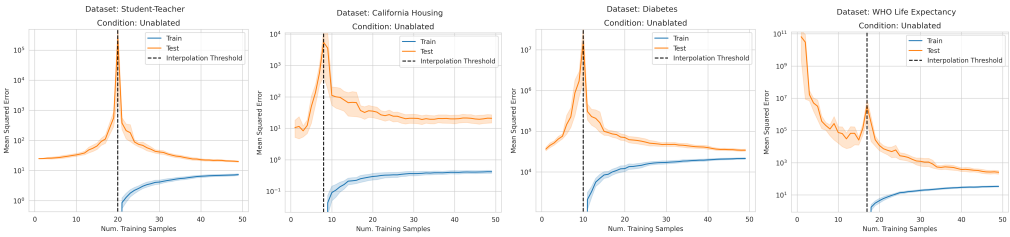

Figure 1: **Ordinary linear regression exhibits a divergence at the interpolation threshold on synthetic and real data.** Left to Right: Synthetic, California Housing [19], Diabetes [9], World Health Organization Life Expectancy [10].

In this work, our goal is *to explain intuitively and quantitatively why the test loss diverges at the interpolation threshold*, without assumptions and without resorting to intricate mathematical tools (e.g., random matrix theory, replica calculations, reproducing kernel Hilbert spaces) but also without sacrificing rigor. To accomplish this, we focus on the basics by examining the simplest supervised model - ordinary linear regression - using the most basic linear algebra primitive: the singular value decomposition[1]. We identify three interpretable factors which, when collectively present, trigger the divergence. Through practical experiments on real data sets, we confirm that both model's test losses diverge at the interpolation threshold, and this divergence vanishes when even one of the three factors is removed. We conclude by shedding light on recent results in nonlinear models concerning superposition [12]. By grounding our understanding in this foundational study, we hope to offer valuable insights into the perplexing behaviors observed in more complex nonlinear models.

2 Divergence in Ordinary Linear Regression

To offer an intuitive yet quantitative understanding of divergence at the interpolation threshold, we turn to ordinary linear regression. Ordinary linear regression is useful for its simplicity, and because closed-form solutions exist for both underparameterized and overparameterized regimes, meaning we can avoid complexity introduced by the learning algorithm and its corresponding learning dynamics.

Notation and Terminology Consider a regression dataset of N training data with features $\vec{x}_n \in \mathbb{R}^D$ and targets $y_n \in \mathbb{R}$. We sometimes use matrix-vector notation to refer to the training data: $X \in \mathbb{R}^{N \times D}$ and $Y \in \mathbb{R}^{N \times 1}$. In ordinary linear regression, we want to learn parameters $\hat{\vec{\beta}} \in \mathbb{R}^D$ such that $\vec{x}_n \cdot \hat{\vec{\beta}} \approx y_n$. We will study three key parameters: number of model parameters P, number of training data N, and dimensionality of the data D. We say that a model is *overparameterized* (a.k.a. underconstrained) if $N < P$ and *underparameterized* (a.k.a. overconstrained) if $N > P$. The *interpolation threshold* refers to $N = P$, because when $N \leq P$, the model can perfectly interpolate the training points. Recall that in ordinary linear regression, the number of fit parameters P must equal the dimension D of the covariates. Consequently, rather than thinking about changing the number of parameters P, we'll instead think about changing the number of data points N. Because the divergence is fundamentally about the ratio of number of parameters P to number of data points N, varying the number of data is equivalent to varying the number of parameters.

Empirical Evidence on Synthetic & Real Data Before studying ordinary linear regression mathematically, does our claim that ordinary linear regression exhibits a divergence at the interpolation threshold hold empirically? We show that it indeed does, using one synthetic dataset and three real datasets (World Health Organization Life Expectancy [10], California Housing [19], Diabetes [9]); these three real datasets were selected on the basis of being easily accessible, e.g., through sklearn [20]. All display a sharp spike in test mean squared error at the interpolation threshold (Fig. 1).

Mathematical Analysis of Ordinary Linear Regression To understand under what conditions and why the test loss diverges at the interpolation threshold in linear regression, we'll study the two

[1]Note: See [25] for a more pedagogical version of this manuscript

parameterization regimes. If the regression is underparameterized, we estimate the linear relationship between covariates $\vec{x}_n$ and target y_n by solving the least-squares minimization problem:

$$\hat{\vec{\beta}}_{under} \stackrel{\text{def}}{=} \arg\min_{\vec{\beta}} \frac{1}{N} \sum_n ||\vec{x}_n \cdot \vec{\beta} - y_n||_2^2 = \arg\min_{\vec{\beta}} ||X\vec{\beta} - Y||_2^2.$$

The solution is the ordinary least squares estimator based on the second moment matrix $X^T X$:

$$\hat{\vec{\beta}}_{under} = (X^T X)^{-1} X^T Y.$$

If the model is overparameterized, the optimization problem is ill-posed since we have fewer constraints than parameters. Consequently, we choose a different (constrained) optimization problem:

$$\hat{\vec{\beta}}_{over} \stackrel{\text{def}}{=} \arg\min_{\vec{\beta}} ||\vec{\beta}||_2^2 \qquad \text{s.t.} \qquad \forall\, n \in \{1, ..., N\} \quad \vec{x}_n \cdot \vec{\beta} = y_n.$$

We choose this optimization problem because it is the one gradient descent implicitly minimizes (App. A). The solution to this optimization problem uses the Gram matrix $XX^T \in \mathbb{R}^{N \times N}$:

$$\hat{\vec{\beta}}_{over} = X^T (XX^T)^{-1} Y.$$

One way to see why the Gram matrix appears is via constrained optimization: define the Lagrangian $\mathcal{L}(\vec{\beta}, \vec{\lambda}) \stackrel{\text{def}}{=} ||\vec{\beta}||_2^2 + \vec{\lambda}^T (Y - X\vec{\beta})$ with Lagrange multipliers $\vec{\lambda} \in \mathbb{R}^N$, then differentiate with respect to the parameters and Lagrange multipliers to obtain the overparameterized solution. After being fit, for test point $\vec{x}_{test}$, the model will make the following predictions:

$$\hat{y}_{test,under} = \vec{x}_{test} \cdot \hat{\vec{\beta}}_{under} = \vec{x}_{test} \cdot (X^T X)^{-1} X^T Y$$
$$\hat{y}_{test,over} = \vec{x}_{test} \cdot \hat{\vec{\beta}}_{over} = \vec{x}_{test} \cdot X^T (XX^T)^{-1} Y.$$

Hidden in the above equations is an interaction between three quantities that can, when all grow extreme, create a divergence in the test loss. To reveal the three quantities, we'll rewrite the regression targets by introducing a slightly more detailed notation. Unknown to us, there are some ideal linear parameters $\vec{\beta}^* \in \mathbb{R}^P = \mathbb{R}^D$ that truly minimize the test mean squared error. We can write any regression target as the inner product of the data $\vec{x}_n$ and the ideal parameters β^*, plus an additional error term e_n that is an "uncapturable" residual from the "perspective" of the model class:

$$y_n = \vec{x}_n \cdot \vec{\beta}^* + e_n.$$

In matrix-vector form, we will equivalently write $Y = X\vec{\beta}^* + E$ with $E \in \mathbb{R}^{N \times 1}$. To be clear, we are *not* imposing assumptions on the model or data. Rather, we are introducing notation to express that there are (unknown) ideal linear parameters, and (possibly) errors E that even the ideal model might be unable to capture; these errors E could be random noise or could be fully deterministic patterns that this particular model class cannot capture. Using this new notation, we rewrite the model's predictions to show how the test datum's features $\vec{x}_{test}$, training data's features X and training data's regression targets Y interact. Let $y^*_{test} \stackrel{\text{def}}{=} \vec{x}_{test} \cdot \beta^*$. In the underparameterized regime:

$$\hat{y}_{test,under} = \hat{y}_{test,under} - y^*_{test} = \vec{x}_{test} \cdot (X^T X)^{-1} X^T E.$$

This equation is important, but opaque. To extract the intuition, replace X with its singular value decomposition $X = U\Sigma V^T$ to reveal how different quantities interact. Let $R \stackrel{\text{def}}{=} \text{rank}(X)$ and let $\sigma_1 > \sigma_2 > ... > \sigma_R > 0$ be X's (non-zero) singular values. We can decompose the underparameterized prediction error $\hat{y}_{test,under} - y^*_{test}$ along the orthogonal singular modes:

$$\hat{y}_{test,under} - y^*_{test} = \vec{x}_{test} \cdot V\Sigma^+ U^T E = \sum_{r=1}^{R} \frac{1}{\sigma_r} (\vec{x}_{test} \cdot \vec{v}_r)(\vec{u}_r \cdot E).$$

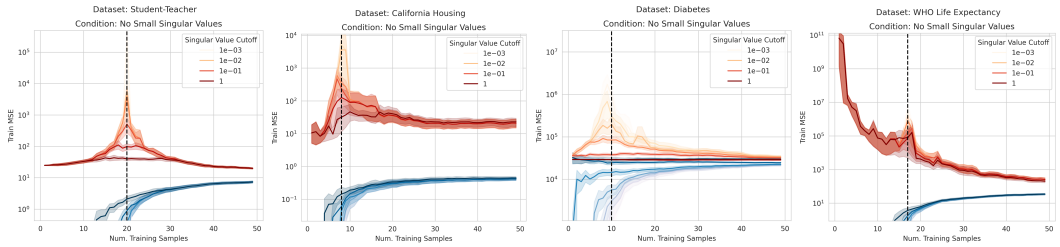

Figure 2: **Required Factor #1: How much the training features vary in each direction.** More formally, the test loss diverges at the interpolation threshold only if the training features X contain small (non-zero) singular values. Ablation: By removing all singular values below a cutoff, the divergence at the interpolation threshold is diminished or disappears entirely.

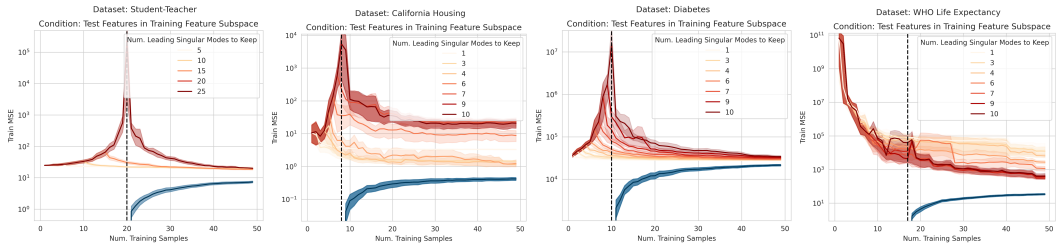

Figure 3: **Required Factor #2: How much, and in which directions, the test features vary relative to the training features.** More formally, the test loss diverges only if the test features $\vec{x}_{test}$ have a large projection onto the training features X's right singular vectors V. Ablation: By projecting the test features into the subspace defined by the leading singular modes, the divergence at the interpolation threshold is diminished or disappears entirely.

In the overparameterized regime, our calculations change slightly:

$$\hat{y}_{test,over} - y^*_{test} = \vec{x}_{test} \cdot (X^T(XX^T)^{-1}X - I_D)\beta^* + \vec{x}_{test} \cdot (X^TX)^{-1}X^TE.$$

The overparameterized prediction error $\hat{y}_{test,over} - y^*_{test}$ has an extra bias term: $\vec{x}_{test} \cdot (X^T(XX^T)^{-1}X - I_D)\beta^*$. The bias exists in the overparameterized regime because for N data points in $D = P$ dimensions, the model can "see" fluctuations in at most N dimensions, but has no "visibility" into the remaining $P - N$ dimensions. This causes information about the optimal linear relationship $\vec{\beta}^*$ to be lost. The other term, the variance, causes the divergence:

$$\sum_{r=1}^{R} \frac{1}{\sigma_r}(\vec{x}_{test} \cdot \vec{v}_r)(\vec{u}_r \cdot E). \tag{1}$$

Eqn. 1 is critical. It reveals that our test prediction error (and thus, our test squared error!) will depend on an interaction between 3 quantities:

1. How much the *training features* X vary in each direction; more formally, the inverse (non-zero) singular values of the *training features* X: $\frac{1}{\sigma_r}$.
2. How much the *test features* $\vec{x}_{test}$ vary relative to the *training features* X (Fig. 2); more formally: how $\vec{x}_{test}$ projects onto X's right singular vectors V: $\vec{x}_{test} \cdot \vec{v}_r$.
3. How well the *best possible model in the model class* can correlate the variance in the *training features* X with the *training regression targets* Y; more formally: how the residuals E of the best possible model in the model class (i.e. insurmountable "errors" from the "perspective" of the model class) project onto X's left singular vectors U: $\vec{u}_r \cdot E$.

When (1) and (3) co-occur, the model's parameters along this mode are likely incorrect. When (2) is added to the mix by a test datum $\vec{x}_{test}$ with a large projection along this mode, the model is forced to extrapolate significantly beyond what it saw in the training data, in a direction where the training data

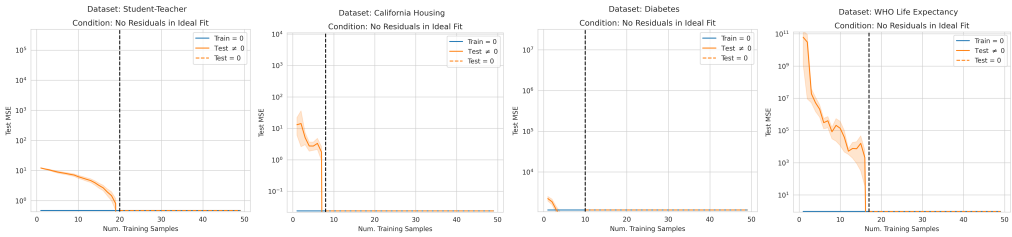

Figure 4: **Required Factor #3: How well the best possible model in the model class can correlate variance in training features with training targets.** More formally, the test loss diverges only if the residuals from the best possible model in the model class have a large projection onto the training features X's left singular vectors. Ablation: By ensuring the true relationship between features and targets is within the function class i.e. linear, the divergence at the interpolation threshold disappears.

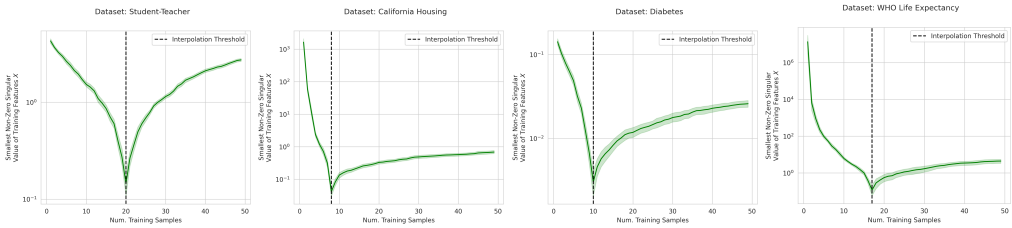

Figure 5: **The training features are most likely to obtain their smallest non-zero singular value when approaching the interpolation threshold.** This means that the first required factor for a divergence (small non-zero singular values; Fig. 2) is likely to occur near the interpolation threshold.

had an error-prone relationship between its training predictions and the training regression targets, using parameters that are likely wrong. As a consequence, the test squared error explodes!

Divergence at the Interpolation Threshold Why does this divergence happen near the interpolation threshold? The answer is that the first factor (small non-zero singular values in the training features X) is likely to occur at the interpolation threshold (Fig. 5), but why? Suppose we're given a single training datum $\vec{x}_1$. So long as this datum isn't exactly zero, that datum varies in a single direction, meaning we gain information about the variance in that direction, but the variance in all orthogonal directions is exactly 0. With the second training datum $\vec{x}_2$, so long as this datum isn't exactly zero, that datum varies, but now, some fraction of $\vec{x}_2$ might have a positive projection along $\vec{x}_1$; if this happens (and it likely will, since the two vectors are unlikely to be exactly orthogonal), the shared direction gives us *more* information about the variance in this shared direction, but *less* information about the second orthogonal direction of variation. Ergo, the training data's smallest non-zero singular value after 2 samples is probabilistically smaller than after 1 sample. As we approach the interpolation threshold, the probability that each additional datum has large variance in a new direction orthogonal to all previous directions grows unlikely (Fig. 6), but as we move beyond the interpolation threshold, the variance in each covariate dimension becomes increasingly clear.

Ablating the Divergence To test our understanding, we conduct three ablation experiments:

1. No Small Singular Values in Training Features: As we run the ordinary linear regression fitting process, and as we sweep the number of training data, we also sweep different singular value cutoffs and remove all singular values of the training features X below the cutoff.
2. Test Features Lie in the Training Features Subspace: As we run the ordinary linear regression fitting process, as we sweep the number of training data, we project the test features $\vec{x}_{test}$ onto the subspace spanned by the training features X singular modes.
3. No Residual Errors in the Optimal Model: We first use the entire dataset to fit a linear model $\vec{\beta}^*$, then replace Y with $X\vec{\beta}^*$ and y^*_{test} with $\vec{x}_{test} \cdot \vec{\beta}^*$ to ensure the true relationship is linear. We then rerun our typical fitting process, sweeping the number of training data.

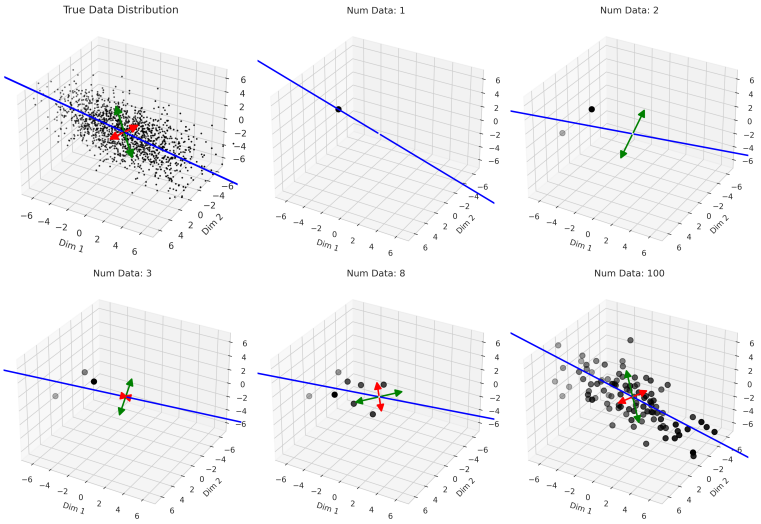

Figure 6: **Geometric intuition for why the smallest non-zero singular value reaches its lowest value when approaching the interpolation threshold.** As one nears the interpolation threshold (here, $D = P = N = 3$), N training data are unlikely to vary substantially in P orthogonal directions, meaning at least one orthogonal direction is likely to have small variance.

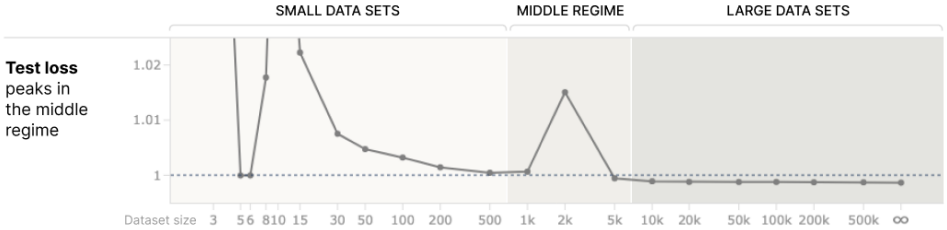

Figure 7: Henighan et al. 2023 [12] recently discovered a divergence in test loss at the interpolation threshold in noiseless nonlinear autoencoders. Despite significant theoretical work in this area, whether such a phenomenon should be expected is unclear because of the numerous assumptions and particular model classes studied by prior work.

We find that each ablation partially or wholly prevents the divergence (Figs. 2 3 4).

3 Intuition Extends to Nonlinear Models

Although we mathematically studied ordinary linear regression, the intuition for why the test loss diverges extends to nonlinear models, such as polynomial regression and including certain classes of deep neural networks [13, 14, 7]. For a concrete example about how our intuition can shed light on the behavior of nonlinear models, Henighan et al. 2023 [12] recently discovered interesting properties of shallow nonlinear autoencoders: depending on the number of training data, (1) autoencoders either store data points or features, and (2) the test loss increases sharply between these two regimes (Fig. 7). Our work sheds light on the results in two ways:

1. Henighan et al. write, “It’s interesting to note that we’re observing double descent in the absence of label noise." Our work clarifies that noise, in the sense of a random quantity, is *not* necessary to produce double descent. Rather, what is necessary is *residual errors from the perspective of the model class* - E, in our notation. Those errors could be entirely deterministic, such as a nonlinear model attempting to fit a noiseless linear relationship, or other model misspecifications.
2. Henighan et al. write, “[Our work] suggests a naive mechanistic theory of overfitting and memorization: memorization and overfitting occur when models operate on ‘data

point features' instead of 'generalizing features'." Our work hopefully clarifies that this terminology can be made more precise: when overparameterized, "data point features" are akin to the Gram matrix XX^T and when underparameterized, "generalizing features" are akin to the second moment matrix X^TX. Our work hopefully clarifies that "data point features" can and very often do generalize, and that there is a deep connection between the two, i.e., their shared spectra.

References

[1] Ben Adlam and Jeffrey Pennington. Understanding double descent requires a fine-grained bias-variance decomposition. *Advances in neural information processing systems*, 33:11022–11032, 2020.

[2] Madhu S Advani, Andrew M Saxe, and Haim Sompolinsky. High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132:428–446, 2020.

[3] Francis Bach. High-dimensional analysis of double descent for linear regression with random projections, 2023.

[4] Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

[5] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[6] Mikhail Belkin, Daniel Hsu, and Ji Xu. Two models of double descent for weak features. *SIAM Journal on Mathematics of Data Science*, 2(4):1167–1180, jan 2020.

[7] Blake Bordelon, Abdulkadir Canatar, and Cengiz Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. In *International Conference on Machine Learning*, pages 1024–1034. PMLR, 2020.

[8] Robert PW Duin. Classifiers in almost empty spaces. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 2, pages 1–7. IEEE, 2000.

[9] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. 2004.

[10] Lasha Gochiashvili. World health organization life expectancy (fixed), 2023.

[11] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *The Annals of Statistics*, 50(2):949–986, 2022.

[12] Tom Henighan, Shan Carter, Tristan Hume, Nelson Elhage, Robert Lasenby, Stanislav Fort, Nicholas Schiefer, and Christopher Olah. Double descent in the condition number. *Transformer Circuits Thread*, 2023.

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

[14] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

[15] Tengyuan Liang and Alexander Rakhlin. Just interpolate: Kernel "ridgeless" regression can generalize. *The Annals of Statistics*, 48(3):1329–1347, 2020.

[16] Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, 75(4):667–766, 2022.

[17] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

[18] Manfred Opper. Statistical mechanics of learning: Generalization. *The handbook of brain theory and neural networks*, pages 922–925, 1995.

[19] R Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[21] Tomaso Poggio, Gil Kur, and Andrzej Banburski. Double descent in the condition number. *arXiv preprint arXiv:1912.06190*, 2019.

[22] Jason W Rocks and Pankaj Mehta. The geometry of over-parameterized regression and adversarial perturbations. *arXiv preprint arXiv:2103.14108*, 2021.

[23] Jason W Rocks and Pankaj Mehta. Bias-variance decomposition of overparameterized regression with random linear features. *Physical Review E*, 106(2):025304, 2022.

[24] Jason W Rocks and Pankaj Mehta. Memorizing without overfitting: Bias, variance, and interpolation in overparameterized models. *Physical Review Research*, 4(1):013201, 2022.

[25] Rylan Schaeffer, Mikail Khona, Zachary Robertson, Akhilan Boopathy, Kateryna Pistunova, Jason W Rocks, Ila Rani Fiete, and Oluwasanmi Koyejo. Double descent demystified: Identifying, interpreting & ablating the sources of a deep learning puzzle. *arXiv preprint arXiv:2303.14151*, 2023.

[26] Stefano Spigler, Mario Geiger, Stéphane d'Ascoli, Levent Sagun, Giulio Biroli, and Matthieu Wyart. A jamming transition from under-to over-parametrization affects loss landscape and generalization. *arXiv preprint arXiv:1810.09665*, 2018.

[27] F Vallet. The hebb rule for learning linearly separable boolean functions: learning and generalization. *Europhysics Letters*, 8(8):747, 1989.

A Why Gradient Descent Implicitly Regularizes

This is a sketch of why gradient descent implicitly regularizes. Suppose we have a model Xw for a vector of data $y \in \mathbb{R}^n$ and want to minimize the norm of the error,

$$L(w) = \|Xw - y\|_2^2 = \|e\|_2^2$$

where we introduce some short-hand notation. We use the gradient learning rule,

$$w(t+1) = w(t) - \eta X^T e(t)$$

$$\Rightarrow e(t+1) = e(t) - \eta X X^T e(t)$$

$$\Rightarrow e(t+1) = (I - \eta X X^T) e(t)$$

Each matrix satisfies $X \in \mathbb{R}^{n \times d_1}$ where n is the number of samples and d_1 is the dimension of each sample. In the overparameterized setting we have $d_1 > n$ and so XX^T will generically have full-rank and the error will go to zero.

This lies in the difference between XX^T which appears here in the error analysis and $X^T X$ which appears in the solution. So we can have $XX^T \in \mathbb{R}^{n \times n}$ generically full-rank only if we have more parameters than there is data. On the other hand, we only have $X^T X$ full-rank if also it's satisfied that there is more data than parameters. This is important because in this case we can compute the pseudo-inverse easily. Generically, we can show that if we use gradient descent we have something like the following,

$$\underbrace{(X^T X)^{-1} X}_{\text{left inverse}} \quad \underbrace{X^{-1}}_{\text{inverse}} \quad \underbrace{X^T (X X^T)^{-1}}_{\text{right inverse}}$$

for the cases where we are under-parameterized, minimally parameterized, or over-parameterized to model the data.

So under gradient flow we'll suppose the parameters update according to,

$$\dot{w} = -\eta X^T e$$

$$w(0) = 0$$

Observe that the gradient $\dot{w}$ is invariantly in the span of X^T so we may conclude that $w(t)$ is always in the span of X^T. Generically, any solution in the over-parameterized setting is a global optimizer such that $Xw = y$. This means that the limit of the flow can be written as $w^* = X^T \alpha$ for some coefficient vector with the constraint that $Xw^* = y$. After some manipulations we find that,

$$y = Xw^* = XX^T \alpha$$

$$\Rightarrow \alpha = (XX^T)^{-1} y$$

$$\Rightarrow w^* = X^T (XX^T)^{-1} y = X^+ y$$

This means that the solution X^+ picked from gradient flow is the Moore-Penrose psuedoinverse. This can be defined as the matrix,

$$X^+ = \lim_{\lambda \to 0^+} X^T (XX^T + \lambda I)^{-1}$$

Also observe that there is a unique minimizer for the regularized problem,

$$\min_w L(w) + \lambda \|w\|_2^2$$

with value $w_\lambda = X^T (XX^T + \lambda I)^{-1} y$. Perhaps, $Xw = y$ has a set of solutions, but it is clear this set is convex so there is a unique minimum norm solution. On the other hand, each w_λ corresponds to a best solution with norm below the minimum. However, we have $w^* = \lim_{\lambda \to 0^+} w_\lambda$ from continuity. Since w^* is an exact solution it can't have less than the minimum-norm and it is clear w^* can't have above the minimum-norm either since this is not the case for any of the w_λ. We conclude that gradient descent does indeed find the minimum norm solution.