OPT2020: 12th Annual Workshop on Optimization for Machine Learning

Learning Rate Annealing Can Provably Help Generalization, Even for Convex Problems

Preetum Nakkiran PREETUM@CS.HARVARD.EDU
Harvard University, USA

Abstract

Learning rate schedule can significantly affect generalization performance in modern neural networks, but the reasons for this are not yet understood. Li et al. [12] recently proved this behavior can exist in a simplified non-convex neural-network setting. In this work, we show that this phenomenon can exist even for convex learning problems – in particular, linear regression in 2 dimensions. We give a toy convex problem where learning rate annealing (large initial learning rate, followed by small learning rate) can lead gradient descent to minima with provably better generalization than using a small learning rate throughout. In our case, this occurs due to a combination of the mismatch between the test and train loss landscapes, and early-stopping.

1. Introduction

The learning rate schedule of stochastic gradient descent is known to strongly affect the generalization of modern neural networks, in ways not explained by optimization considerations alone (e.g. [6, 9, 12, 19]). In particular, training with a large initial learning-rate followed by a smaller annealed learning rate can vastly outperform training with the smaller learning rate throughout – even when allowing both to reach the same train loss. The recent work of Li et al. [12] sheds some light on this, by showing a simplified neural-network setting in which this behavior provably occurs.

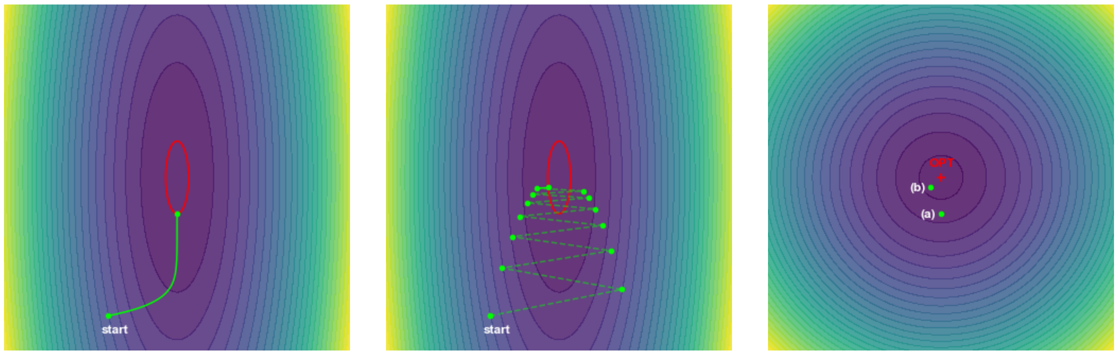
It was proposed (e.g. in Li et al. [12]) that non-convexity is crucial for learning-rate schedule to affect generalization, because strongly-convex objectives have unique global minima. However, we show this behavior can appear even in strongly-convex learning problems. The key insight is that although strongly-convex problems have a unique minima, early-stopping breaks this uniqueness: there is a set of minima with the same train loss ε. And these minima can have different generalization properties when the train loss surface does not closely approximate the test loss surface. In our setting, a large learning-rate prevents gradient descent from optimizing along high-curvature directions, and thus effectively regularizes against directions that are high-curvature in the train set, but low-curvature on the true distribution (see Figure 1).

Technically, we model a small learning rate by considering gradient flow. Then we compare the following optimizers:

(A) Gradient descent with an infinitesimally small step size (i.e. gradient flow).

(B) Gradient descent with a large step size, followed by gradient flow.

We show that for a particular linear regression problem, if we run both (A) and (B) until reaching the same $\varepsilon > 0$ train loss, then with probability $3/4$ over the train samples, the test loss of (A) is twice that of (B). That is, starting with a large learning rate and then annealing to an infinitesimally small one helps generalization significantly.



(*a*) Small learning rate (*b*) Large, then small LR (*c*) Test loss landscape

Figure 1: **How learning rate can affect generalization:** Minima with the same train loss can have different test losses, if the empirical loss landscape is distorted relative to the population loss (e.g. due to undersampling/overparameterization). Figures (a) and (b) show gradient descent on the train loss, with different learning rate schedules, stopping at the same small value of train-loss (red ellipse). Note that the minima found in (a) is worse than (b) in test loss (shown in Figure (c)), through they have identical train loss.

Our Contribution. We show that non-convexity is not required to reproduce an effect of learning rate on generalization that is observed in deep learning models. We give a simple model where the mechanisms behind this can be theoretically understood, which shares some features with deep learning models in practice. We hope such simple examples can yield insight and intuition that may eventually lead to a better understanding of the effect of learning rate in deep learning.

Organization. In Section 2 we describe the example. In Section 3 we discuss features of the example, and its potential implications (and non-implications) in deep learning. In Appendix A, we include formal proofs and provide a mild generalization of the example in Section 2.

1.1. Related Work

This work was inspired by Li et al. [12], which gives a certain simplified neural-network setting in which learning-rate schedule provably affects generalization, despite performing identically with respect to optimization. The mechanisms and intuitions in Li et al. [12] depend crucially on non-convexity of the objective. In contrast, in this work we provide complementary results, by showing that this behavior can occur even in convex models due to the interaction between early-stopping and estimation error. Our work is not incompatible with Li et al. [12]; indeed, the true mechanisms behind generalization in real neural networks may involve both of these factors (among others).

There are many empirical and theoretical works attempting to understand the effect of learning-rate in deep learning, which we briefly describe here. Several recent works study the effect of learning rate by considering properties (e.g. curvature) of the loss landscape, finding that different learning rate schedules can lead SGD to regions in parameter space with different local geometry [8, 11]. This fits more broadly into a study of the implicit bias of SGD. For example, there is debate on whether generalization is connected to "sharpness" of the minima (e.g. Dinh et al.

[1], Hochreiter and Schmidhuber [4], Keskar et al. [9]). The effect of learning-rate is also often also studied alongside the effect of batch-size, since the optimal choice of these two parameters is coupled in practice [3, 10, 16]. Several works suggest that the *stochasticity* of SGD is crucial to the effect of learning-rate, in that different learning rates lead to different stochastic dynamics, with different generalization behavior [5, 12–15, 17]. In particular, it may be the case that the stochasticity in the gradient estimates of SGD acts as an effective "regularizer" at high learning rates. This aspect is not explicitly present in our work, and is an interesting area for future theoretical and empirical study.

2. Convex Example

The main idea is illustrated in the following linear regression problem, as visualized in Figure 1. Consider the data distribution $\mathcal{D}$ over $(x, y) \in \mathbb{R}^2 \times \mathbb{R}$ defined as:

$$x \in \{\boldsymbol{e_1}, \boldsymbol{e_2}\} \text{ uniformly at random ;} \quad y = \langle \beta^*, x \rangle$$

for some ground-truth $\beta^* \in \mathbb{R}^2$. We want to learn a linear model $\hat{y}(x) := \langle \beta, x \rangle$ with small mean-squared-error on the population distribution. Specifically, for model parameters $\beta = (\beta_1, \beta_2) \in \mathbb{R}^2$, the population loss is

$$L_{\mathcal{D}}(\beta) := \mathop{\mathbb{E}}_{x,y \sim D}[(\langle \beta, x \rangle - y)^2]$$

We want to approximate $\beta^* = \text{argmin}_\beta L_{\mathcal{D}}(\beta)$. Suppose we try to find this minima by drawing n samples $\{x_i, y_i\}$ from $\mathcal{D}$, and performing gradient descent on the empirical loss:

$$\hat{L}_n(\beta) := \frac{1}{n} \sum_{i \in [n]} (\langle \beta, x_i \rangle - y_i)^2$$

starting at $\beta = 0$, and stopping when $\hat{L}_n \leq \varepsilon$ for some small ε.

Now for simplicity let $n = 3$, and let the ground-truth model be $\beta^* = (\beta_1^*, \beta_2^*) \in \mathbb{R}^2$. With probability $3/4$, two of the samples will have the same value of x – say this value is $x = \boldsymbol{e_1}$, so the samples (x_i, y_i) are $\{(e_1, \beta_1^*), (e_1, \beta_1^*), (e_2, \beta_2^*)\}$. In this case the empirical loss is

$$\hat{L}_n(\beta) = \underbrace{\frac{2}{3}(\beta_1 - \beta_1^*)^2}_{\text{(A)}} + \underbrace{\frac{1}{3}(\beta_2 - \beta_2^*)^2}_{\text{(B)}} \tag{1}$$

which is *distorted* compared to the population loss, since we have few samples relative to the dimension. The population loss is:

$$L_{\mathcal{D}}(\beta) = \frac{1}{2}(\beta_1 - \beta_1^*)^2 + \frac{1}{2}(\beta_2 - \beta_2^*)^2 \tag{2}$$

The key point is that although the global minima of $L_{\mathcal{D}}$ and $\hat{L}_n$ are identical, their level sets are not: not all points of small train loss $\hat{L}_n(\beta)$ have identical test loss $L_{\mathcal{D}}(\beta)$. To see this, refer to Equation 1, and consider two different ways that train loss $\hat{L}_n = \varepsilon$ could be achieved. In the "good" situation, term (A) in Equation 1 is ε, and term (B) is 0. In the "bad" situation, term (A) is 0 and term (B) is ε. These two cases have test losses which differ by a factor of two, since terms are re-weighted differently in the test loss (Equation 2). This is summarized in the below table.

	Residual $(\beta - \beta^*)$	Train Loss $\hat{L}_n(\beta)$	Test Loss $L_{\mathcal{D}}(\beta)$
Good:	$[\sqrt{\frac{3\varepsilon}{2}}, 0]$	ε	0.75ε
Bad:	$[0, \sqrt{3\varepsilon}]$	ε	1.5ε

Now, if our optimizer stops at train loss ε, it will pick one of the points in $\{\beta : \hat{L}_n(\beta) = \varepsilon\}$. We see from the above that some of these points are twice as bad as others.

Notice that *gradient flow* on $\hat{L}_n$ (Equation 1) will optimize twice as fast along the β_1 coordinate compared to β_2. The gradient flow dynamics of the residual $(\beta - \beta^*)$ are:

$$\frac{d\beta}{dt} = -\nabla_\beta \hat{L}_n \implies \begin{cases} \frac{d}{dt}(\beta_1 - \beta_1^*) = -\frac{4}{3}(\beta_1 - \beta_1^*) \\ \frac{d}{dt}(\beta_2 - \beta_2^*) = -\frac{2}{3}(\beta_2 - \beta_2^*) \end{cases}$$

This will tend to find solutions closer to the "Bad" solution from the above table, where $(\beta_1 - \beta_1^*)^2 \approx 0$ and $(\beta_2 - \beta_2^*)^2 \gg 0$.

However, *gradient descent* with a large step size can oscillate on the β_1 coordinate, and achieve low train loss by optimizing β_2 instead. Then in the second stage, once the learning-rate is annealed, gradient descent will optimize on β_1 while keeping the β_2 coordinate small. This will lead to a minima closer to the "Good" solution, where $(\beta_2 - \beta_2^*) \approx 0$. These dynamics are visualized in Figure 1.

This example is formalized in the following proposition. The proof is straightforward, and included in Appendix A.

Proposition 1 *For all $0 < \alpha < 1$, there exists a distribution $\mathcal{D}$ over $(x, y) \in \mathbb{R}^2 \times \mathbb{R}$ and a learning-rate $\eta > 0$ (the "large" learning-rate) such that for all $0 < \varepsilon < 1$, the following holds. With probability $3/4$ over $n = 3$ samples:*

1. *Gradient flow from 0-initialization, early-stopped at train loss ε, achieves population loss*

$$L_{\mathcal{D}}(\beta_{\text{bad}}) \geq \frac{3}{2}\varepsilon(1 - \alpha)$$

2. *Annealed gradient descent from 0-initialization (i.e. gradient descent with stepsize η for K steps, followed by gradient flow) stopped at train loss ε, achieves population loss*

$$L_{\mathcal{D}}(\beta_{\text{good}}) \leq \frac{3}{4}\varepsilon + \exp(-\Omega(K))$$

In particular, since α can be taken arbitrarily small, and K taken arbitrarily large, this implies that gradient flow achieves a population loss twice as high as gradient descent with a careful step size, followed by gradient flow.

Moreover, with the remaining $1/4$ probability, the samples will be such that gradient flow and annealed gradient descent behave identically:

$$\beta_{\text{good}} = \beta_{\text{bad}}$$

3. Discussion

Similarities. This example, though stylized, shares several features with deep neural networks in practice.

- Neural nets trained with cross-entropy loss cannot be trained to global emperical risk minimas, and are instead early-stopped at some small value of train loss.
- There is a mismatch between train and test loss landscapes (in deep learning, this is due to overparameterization/undersampling).
- Learning rate annealing typically generalizes better than using a small constant learning rate [2].
- The "large" learning rates used in practice are far larger than what optimization theory would prescribe. In particular, consecutive iterates of SGD are often negatively-correlated with each other in the later stage of training [18], suggesting that the iterates are oscillating around a sharp valley. Jastrzebski et al. [7] also finds that the typical SGD step is too large to optimize along the steepest directions of the loss.

Limitations. Our example is nevertheless a toy example, and does not exhibit some features of real networks. For example, our setting has only one basin of attraction. But real networks have many basins of attraction, and there is evidence that a high initial learning rate influences the choice of basin (e.g. Jastrzebski et al. [8], Li et al. [12]). It remains an important open question to understand the effect of learning rate in deep learning.

References

[1] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1019–1028. JMLR. org, 2017.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[3] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

[5] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems*, pages 1731–1741, 2017.

[6] Stanisław Jastrzebski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Three factors influencing minima in sgd. *arXiv preprint arXiv:1711.04623*, 2017.

[7] Stanislaw Jastrzebski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. On the relation between the sharpest directions of dnn loss and the sgd step length. *arXiv preprint arXiv:1807.05031*, 2018.

[8] Stanislaw Jastrzebski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho, and Krzysztof Geras. The break-even point on optimization trajectories of deep neural networks. *arXiv preprint arXiv:2002.09572*, 2020.

[9] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

[10] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

[11] Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

[12] Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards explaining the regularization effect of initial large learning rate in training neural networks, 2019.

[13] Stephan Mandt, Matthew D Hoffman, and David M Blei. Stochastic gradient descent as approximate bayesian inference. *The Journal of Machine Learning Research*, 18(1):4873–4907, 2017.

[14] Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

[15] Samuel L Smith and Quoc V Le. A bayesian perspective on generalization and stochastic gradient descent. *arXiv preprint arXiv:1710.06451*, 2017.

[16] Samuel L Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V Le. Don't decay the learning rate, increase the batch size. *arXiv preprint arXiv:1711.00489*, 2017.

[17] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

[18] Chen Xing, Devansh Arpit, Christos Tsirigotis, and Yoshua Bengio. A walk with sgd. *arXiv preprint arXiv:1802.08770*, 2018.

[19] Zhanxing Zhu, Jingfeng Wu, Bing Yu, Lei Wu, and Jinwen Ma. The anisotropic noise in stochastic gradient descent: Its behavior of escaping from sharp minima and regularization effects. *arXiv preprint arXiv:1803.00195*, 2018.

Appendix A. Formal Statements

In this section, we state and prove Claim 1, along with a mild generalization of the setting via Lemma 2.

A.1. Notation and Preliminaries

For a distribution $\mathcal{D}$ over $(x, y) \in \mathbb{R}^d \times \mathbb{R}$, let $L_{\mathcal{D}}(\beta) := \mathbb{E}_{x,y\sim D}[(\langle x, \beta\rangle - y)^2]$ be the population loss for parameters β. Let $\hat{L}_n(\boldsymbol{\beta}) = \frac{1}{n}\sum_i(\langle x_i, \beta\rangle - y_i)^2$ be the train loss for n samples $\{(x_i, y_i)\}$.

Let $GF_{\beta_0}(L, \varepsilon)$ be the function which optimizes train loss L from initial point β_0, until the train loss reaches early-stopping threshold ε, and then outputs the resulting parameters.

Let $GD_{\beta_0}(L, K, \tau, \varepsilon)$ be the function which first optimizes train loss L with gradient-descent starting at initial point β_0, with step-size η, for K steps, and an early-stopping threshold ε, and then continues with gradient flow with early-stopping threshold ε.

For n samples, let $X \in \mathbb{R}^{n\times d}$ be the design matrix of samples. Let $\Sigma_X = \mathbb{E}_{x,y\sim\mathcal{D}}[xx^T]$ be the population covariance and let $\hat{\Sigma}_X = \frac{1}{n}X^TX$ be the emperical covariance.

We assume throughout that Σ_X and $\hat{\Sigma}_X$ are simultaneously diagonalizable:

$$\Sigma_X = U\Lambda U^T \quad ; \quad \hat{\Sigma}_X = U\Gamma U^T$$

for diagonal $\Lambda = \text{diag}(\lambda_i), S = \text{diag}(\gamma_i)$ and orthonormal U. Further, assume without loss of generality that $\hat{\Sigma}_X \succ 0$ (if $\hat{\Sigma}_X$ is not full rank, we can restrict attention to its image).

We consider optimizing the the train loss:

$$\hat{L}(\beta) = \frac{1}{n}||X(\beta - \beta^*)||_2^2 = ||\beta - \beta^*||^2_{\hat{\Sigma}_X}$$

Observe that the optimal population loss for a point with ε train loss is:

$$\min_{\text{s.t. } L_{\hat{\Sigma}}(\beta)=\varepsilon} L_\Sigma(\beta) = \varepsilon \min_i(\{\frac{\lambda_i}{\gamma_i}\}) \tag{3}$$

Similarly, the worst population loss for a point with ε train loss is:

$$\max_{\text{s.t. } L_{\hat{\Sigma}}(\beta)=\varepsilon} L_\Sigma(\beta) = \varepsilon \max_i(\{\frac{\lambda_i}{\gamma_i}\}) \tag{4}$$

Now, the gradient flow dynamics on the train loss is:

$$\frac{d\beta}{dt} = -\nabla_\beta \hat{L}(\beta) = -\frac{2}{n}X^TX(\beta - \beta^*)$$

Switching to coordinates $\delta := U^T(\beta - \beta^*)$, these dynamics are equivalent to:

$$\frac{d\delta}{dt} = -2\Gamma\delta$$

The empirical and population losses in these coordinates can be written as:

$$L_{\mathcal{D}}(\delta) = \sum_i \lambda_i\delta_i^2 \quad ; \quad \hat{L}(\delta) = \sum_i \gamma_i\delta_i^2$$

And the trajectory of gradient flow, from initialization $\delta(0) \in \mathbb{R}^d$, is

$$\forall i \in [d] : \quad \delta_i(t) = \delta_i(0)e^{-2\gamma_i t}$$

With this setup, we now state and prove the following main lemma, characterizing the behavior of gradient flow and gradient descent for a given sample covariance.

A.2. Main Lemma

Lemma 2 *Let $\Sigma, \hat{\Sigma} \in \mathbb{R}^{d\times d}$ be as above. Let eigenvalues be ordered according to $\hat{\Sigma}$: $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_d$.*

Let $S = \{k, k+1, \dots, d\}$ for some k. That is, let S index a block of "small" eigenvalues. Let $\overline{S} = [n] \setminus S$. Assume there is an eigenvalue gap *between the "large" and "small" eigenvalues, where for some $p > 0$*

$$\forall i \in \overline{S} : \gamma_i/\gamma_k \geq 1 + p.$$

For all $0 < \alpha < 1$, if the initialization $\delta(0) = U^T(\beta_0 - \beta^*) \in \mathbb{R}^d$ satisfies

1. *At initialization, the contribution of the largest eigenspace to the train loss is at least ε:*

$$\gamma_1\delta_1(0)^2 > \varepsilon$$

2. *The eigenvalue gap is large enough to ensure that eventually only the "small" eigenspace is significant, specifically:*

$$\varepsilon^p \sum_{i\in\overline{S}} \frac{\gamma_i\delta_i(0)^2}{(|S|\gamma_{j^*}\delta_{j^*}(0)^2)^{1+p}} \leq \alpha \quad \textit{where } j^* := \operatorname*{argmin}_{j\in S} \gamma_j\delta_j(0)^2$$

Then there exists a learning-rate $\eta := 1/\gamma_1$ (the "large" learning-rate) such that

1. *Gradient flow $\beta_{slow} \leftarrow GF_{\boldsymbol{\beta}_0}(\hat{L}, \varepsilon)$ achieves population loss*

$$L_{\mathcal{D}}(\beta_{slow}) \geq \varepsilon(1-\alpha) \min_{j\in S} \frac{\lambda_j}{\gamma_j}$$

2. *Annealed gradient descent run for K steps $\beta_{fast} \leftarrow GD(\hat{L}, K, \eta, \varepsilon)$ has loss*

$$L_{\mathcal{D}}(\beta_{fast}) \leq \varepsilon \max_{j:\gamma_j=\gamma_1} (\frac{\lambda_j}{\gamma_j}) + \exp(-\Omega(K))$$

In particular, if the top eigenvalue of Γ is unique, then

$$L_{\mathcal{D}}(\beta_{fast}) \leq \varepsilon\frac{\lambda_1}{\gamma_1} + \exp(-\Omega(K))$$

The constant $\Omega(K)$ depends on the spectrum, and taking $\Omega(K) = -K\log(\max_{j:\gamma_j\neq\gamma_1} |1 - 2\gamma_j/\gamma_1|)$ suffices.

Proof

Gradient Flow.

The proof idea is to show that gradient flow must run for some time T in order to reach ε train loss – and since higher-eigenvalues optimize faster, most of the train loss will be due to contributions from the "small" coordinates S.

Let T be the stopping-time of gradient flow, such that $\hat{L}(\delta(T)) = \varepsilon$. We can lower-bound the time T required to reach ε train loss as:

$$|S|\gamma_{j^*}\delta_{j^*}(0)^2 e^{-4\gamma_k T} \leq \sum_{j\in S}\gamma_j\delta_j(0)^2 e^{-4\gamma_j T} \leq \sum_{j\in[n]}\gamma_j\delta_j(0)^2 e^{-4\gamma_j T} = \hat{L}(\delta(T)) = \varepsilon \tag{5}$$

$$\implies T \geq \frac{1}{4\gamma_k}\log(\frac{|S|\gamma_{j^*}\delta_{j^*}(0)^2}{\varepsilon}) \tag{6}$$

Decompose the train loss as:

$$\hat{L}(\delta) = \hat{L}_{\overline{S}}(\delta) + \hat{L}_S(\delta)$$

Where $\hat{L}_S(\delta) := \sum_{i\in S}\gamma_i\delta_i^2$ and $\hat{L}_{\overline{S}}(\delta) := \sum_{i\in\overline{S}}\gamma_i\delta_i^2$. Now, we can bound $\hat{L}_{\overline{S}}(\delta(T))$ as:

$$\begin{aligned}
\hat{L}_{\overline{S}}(\delta(T)) = \sum_{i\in\overline{S}}\gamma_i\delta_i(T)^2 &= \sum_{i\in\overline{S}}\gamma_i\delta_i(0)^2 e^{-4\gamma_i T} \\
&\leq \sum_{i\in\overline{S}}\gamma_i\delta_i(0)^2(\frac{\varepsilon}{|S|\gamma_{j^*}\delta_{j^*}(0)^2})^{\gamma_i/\gamma_k} && \text{(by Equation 6)} \\
&\leq \sum_{i\in\overline{S}}\gamma_i\delta_i(0)^2(\frac{\varepsilon}{|S|\gamma_{j^*}\delta_{j^*}(0)^2})^{1+p} \\
&\leq \alpha\varepsilon
\end{aligned}$$

Where the last inequality is due to Condition 2 of the Lemma. Now, since $\hat{L}(\delta) = \hat{L}_{\overline{S}}(\delta) + \hat{L}_S(\delta)$, and $\hat{L}(\delta(T)) = \varepsilon$, and $\hat{L}_{\overline{S}}(\delta(T)) \leq \alpha\varepsilon$, we must have

$$\hat{L}_S(\delta(T)) \geq (1-\alpha)\varepsilon$$

This lower-bounds the population loss as desired.

$$L_{\mathcal{D}}(\delta(T)) \geq \sum_{i\in S}\lambda_i\delta_i(T)^2 \geq \min_{j\in S}(\frac{\lambda_j}{\gamma_j})\sum_{i\in S}\gamma_i\delta_i(T)^2 = \min_{j\in S}(\frac{\lambda_j}{\gamma_j})\hat{L}_S(\delta(T)) \geq \min_{j\in S}(\frac{\lambda_j}{\gamma_j})(1-\alpha)\varepsilon$$

Annealed Gradient Descent.

For annealed gradient descent, the proof idea is: set the learning-rate such that in the gradient descent stage, the optimizer oscillates on the first (highest-curvature) coordinate δ_1, and optimizes until the remaining coordinates are ≈ 0. Then, in the gradient flow stage, it optimizes on the first coordinate until hitting the target train loss of ε. Thus, at completion most of the train loss will be due to contributions from the "large" first coordinate.

Specifically, the gradient descent dynamics with stepsize η is:

$$\delta_i(t+1) = (1-2\eta\gamma_i)\delta_i(t)$$

for discrete $t \in \mathbb{N}$. We set $\eta := 1/\gamma_1$, so the first coordinate simply oscillates: $\delta_1(t+1) = -\delta_1(t)$. This is the case for the top eigenspace, i.e. all coordinates i where $\gamma_i = \gamma_1$. The remaining coordinates decay exponentially. That is, let $Q := \{i : \gamma_i \neq \gamma_1\}$ be the remaining coordinates (corresponding to smaller eigenspaces). Then we have:

$$\forall i \in Q : \ \delta_i(t)^2 \leq c^{-t}\delta_i(0)^2 \tag{7}$$

for constant $c = \max_{j:\gamma_j \neq \gamma_1} |1 - 2\eta\gamma_j|^2 < 1$.

Recall, the population and empirical losses are:

$$L_{\mathcal{D}}(\delta) = \sum_i \lambda_i \delta_i^2 \quad ; \quad \hat{L}(\delta) = \sum_i \gamma_i \delta_i^2$$

By Equation 7, after K steps of gradient descent, the contribution to the population loss from the coordinates $i \in Q$ is small:

$$\sum_{i \in Q} \lambda_i \delta_i(K)^2 \leq \exp(-\Omega_c(K))$$

By Condition 1 of the Lemma, the train loss is still not below ε after the gradient descent stage, since the first coordinate did not optimize: $\delta_1(0) = \pm\delta_1(0)$. We now run gradient flow, stopping at time $T \in \mathbb{R}$ when the train loss is ε. At this point, we have

$$\hat{L}(\delta(T)) \leq \varepsilon \implies \sum_{i \notin Q} \gamma_i \delta_i(T)^2 \leq \varepsilon$$

And thus,

$$\begin{aligned} L_{\mathcal{D}}(\delta(T)) &= \sum_{i \notin Q} \lambda_i \delta_i(T)^2 + \sum_{i \in Q} \lambda_i \delta_i(T)^2 \\ &\leq (\max_{k \notin Q} \frac{\lambda_k}{\gamma_k}) \sum_{i \notin Q} \gamma_i \delta_i(T)^2 + \exp(-\Omega(K)) \\ &\leq (\max_{k \notin Q} \frac{\lambda_k}{\gamma_k})\varepsilon + \exp(-\Omega(K)) \end{aligned}$$

as desired. ■

As a corollary of Lemma 2, we recover the 2-dimensional of Claim 1.

A.3. Proof of Claim 1

Proof [Proof sketch of Claim 1] Consider the distribution $\mathcal{D}$ defined as: Sample $x \in \{\boldsymbol{e_1}, \boldsymbol{e_2}\}$ uniformly at random, and let $y = \langle \beta^*, x \rangle$ for ground-truth $\beta^* = (100\alpha^{-1/2}, 100\alpha^{-1/2})$. The population covariance is simply

$$\Sigma_X = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix}$$

With probability $3/4$, two of the three samples will have the same value of x. In this case, the sample covariance will be equal (up to reordering of coordinates) to

$$\hat{\Sigma}_X = \begin{bmatrix} 2/3 & 0 \\ 0 & 1/3 \end{bmatrix}$$

This satisfies the conditions of Lemma 2, taking the “small” set of eigenvalue indices to be $S = \{2\}$. Thus, the conclusion follows by Lemma 2.

With probability $1/4$, all the samples will all be identical, and in particular share the same value of x. Here, the optimization trajectory will be 1-dimensional, and it is easy to see that gradient flow and annealed gradient descent will reach identical minima. ■