Risk bounds for deep learning

Bos, J.M.

Chapter 4

Convergence guarantees for forward gradient descent in the linear regression model

Abstract

Renewed interest in the relationship between artificial and biological neural networks motivates the study of gradient-free methods. Considering the linear regression model with random design, we theoretically analyze in this chapter the biologically motivated (weight-perturbed) forward gradient scheme that is based on random linear combination of the gradient. If d denotes the number of parameters and k the number of samples, we prove that the mean squared error of this method converges for $k \gtrsim d^2 \log(d)$ with rate $d^2 \log(d)/k$. Compared to the dimension dependence d for stochastic gradient descent, an additional factor $d \log(d)$ occurs.

4.1 Introduction

Looking at the past developments, it is apparent that artificial neural networks (ANNs) became more powerful the more they resembled the brain. It is therefore anticipated that the future of AI is even more biologically inspired. As in the past, the bottlenecks towards more biologically inspired learning are computational barriers. For instance, shallow networks only became computationally feasible after the backpropagation algorithm was proposed. Deep neural networks were proposed for a longer time but deep learning became implementable after the development of large scale GPU

This chapter is based on: Thijs Bos and Johannes Schmidt-Hieber. Convergence guarantees for forward gradient descent in the linear regression model. To appear in Journal of Statistical Planning and Inference, Volume ***233****. The research has been supported by the NWO/STAR grant 613.009.034b and the NWO Vidi grant VI.Vidi.192.021.*

computing. Neuromorphic computing aims to imitate the brain on computer chips, but is currently not fully scalable.

The mathematics of AI has focused on explaining the state-of-the-art performance of modern machine learning methods and empirically observed phenomena such as the good generalization properties of extreme overparametrization. To shape the future of AI, statistical theory needs more emphasis on anticipating future developments and proposing biologically motivated methods already at a stage before scalable implementations exist.

This chapter aims to analyze a biologically motivated learning rule building on the renewed interest of the differences and similarities between ANNs and biological neural networks (BNNs) [89, 128, 155] which are rooted in the foundational literature from the 1980s [53, 33]. A key difference between ANNs and BNNs is that ANNs are usually trained based on a version of (stochastic) gradient descent, while this seems prohibitive for BNNs. Indeed, to compute the gradient, knowledge of all parameters in the network is required, but biological networks do not posses the capacity to transport this information to each neuron. This suggests that biological networks cannot directly use the gradient to update their parameters [33, 89, 142].

The brain still performs well without gradient descent and can learn tasks with much fewer examples than ANNs. This sparks interest in biologically plausible learning methods that do not require (full) access of the gradient. Such methods are called derivative-free. A simple example of a derivative-free method is to randomly sample in each step a new parameter. If this decreases the loss one keeps the parameter and otherwise discards it. There is a wide variety of derivative-free strategies [32, 83, 135]. Among those, so-called zero-order methods use evaluations of the loss function to build a noisy estimate of the gradient. This substitute is then used to replace the gradient in the gradient descent routine [92, 41]. [128] establishes a connection between the Hebbian learning underlying the local learning of the brain (see e.g. Chapter 6 of [142]) and a specific zero-order method. A statistical analysis of this zero-order scheme is provided in the companion article [129].

In this chapter, we study (weight-perturbed) forward gradient descent. This method is motivated by biological neural networks [13, 117] and lies between full gradient descent methods and derivative-free methods, as only random linear combination of the gradient are required. The form of the random linear combination is related to zero-order estimators, see Section 4.2. Settings with partial access to the gradient have been studied before. For example, [105] proposes a learning method based on directional derivatives for convex functions. In this chapter we specifically derive theoretical guarantees for forward gradient descent in the linear regression model with random design. Theorem 4.3.1 establishes an expression for the expectation. A bound on the mean squared error is provided in Theorem 4.3.3.

The structure of this chapter is as follows. In Section 4.2 we describe the forward gradient descent update rule in the linear regression model. Results are in Section 4.3 and the corresponding proofs can be found in Section 4.4.

Notation

Vectors are denoted by bold letters and we write $\| \cdot \|_2$ for the Euclidean norm. We denote the largest and smallest eigenvalue of a matrix A by the respective expressions $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$. The spectral norm is $\|A\|_S := \sqrt{\lambda_{\max}(A^\top A)}$. The condition number of a positive semi-definite matrix B is $\kappa(B) := \lambda_{\max}(B)/\lambda_{\min}(B)$.

For a random variable U we denote the expectation with respect to U by $\mathbb{E}_U$. The symbol $\mathbb{E}$ stands for an expectation taken with respect to all random variables that are inside that expectation. The (multivariate) normal distribution with mean vector μ and covariance matrix Σ is denoted by $\mathcal{N}(\mu, \Sigma)$.

4.2 Weight-perturbed forward gradient descent

Suppose we want to learn a parameter vector $\boldsymbol{\theta}$ from training data $(\mathbf{X}_1, Y_1), (\mathbf{X}_2, Y_2), \ldots \in \mathbb{R}^d \times \mathbb{R}$. Stochastic gradient descent (SGD) is based on the iterative update rule

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha_{k+1} \nabla L(\boldsymbol{\theta}_k), \quad k = 0, 1, \ldots \tag{4.2.1}$$

with $\boldsymbol{\theta}_0$ some initial value and $L(\boldsymbol{\theta}_k) := L(\boldsymbol{\theta}_k, \mathbf{X}_k, Y_k)$ a loss that depends on the data only through the k-th sample $(\mathbf{X}_k, Y_k)$.

For a standard normal random vector $\boldsymbol{\xi}_{k+1} \sim \mathcal{N}(0, \mathbf{I}_d)$ that is independent of all the other randomness, the quantity $(\nabla L(\boldsymbol{\theta}_k))^\top \boldsymbol{\xi}_{k+1} \boldsymbol{\xi}_{k+1}$ is called the (weight-perturbed) forward gradient [13, 117]. *(Weight-perturbed) forward gradient descent* is then given by the update rule

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha_{k+1} \big(\nabla L(\boldsymbol{\theta}_k)\big)^\top \boldsymbol{\xi}_{k+1} \boldsymbol{\xi}_{k+1}, \quad k = 0, 1, \ldots \tag{4.2.2}$$

Assuming that the exogenous noise has unit variance is sufficient. Indeed, generalizing to $\boldsymbol{\xi}_{k+1} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d)$ with variance parameter σ^2 has the same effect as rescaling the learning rate $\alpha_{k+1} \to \sigma^{-2} \alpha_{k+1}$.

Since for a deterministic d-dimensional vector $\mathbf{v}$, one has $\mathbb{E}[\mathbf{v}^t \boldsymbol{\xi}_{k+1} \boldsymbol{\xi}_{k+1}] = \mathbf{v}$, taking the expectation of the weight-perturbed forward gradient descent scheme with respect to the exogenous randomness induced by $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots$ gives

$$\mathbb{E}_{(\boldsymbol{\xi}_i)_{i \geq 1}}[\boldsymbol{\theta}_{k+1}] = \mathbb{E}_{(\boldsymbol{\xi}_i)_{i \geq 1}}[\boldsymbol{\theta}_k] - \alpha_{k+1} \mathbb{E}_{(\boldsymbol{\xi}_i)_{i \geq 1}}[\nabla L(\boldsymbol{\theta}_k)], \tag{4.2.3}$$

resembling the SGD dynamic (4.2.1). If $\nabla L(\boldsymbol{\theta}_k)$ depends on $\boldsymbol{\theta}_k$ linearly then also $\mathbb{E}_{(\boldsymbol{\xi}_i)_{i \geq 1}}[\nabla L(\boldsymbol{\theta}_k)] = \nabla L(\mathbb{E}_{(\boldsymbol{\xi}_i)_{i \geq 1}}[\boldsymbol{\theta}_k])$.

While in expectation, forward gradient descent is related to SGD, the induced randomness of the d-dimensional random vectors $\mathbf{x}_{k+1}$ induces a large amount of noise. To control the high noise level in the dynamic is the main obstacle in the mathematical analysis. One of the implications is that one has to make small steps by choosing a small learning rate to avoid completely erratic behavior. This particularly effects the first phase of the learning.

First order multivariate Taylor expansion shows that $L(\boldsymbol{\theta}_k + \boldsymbol{\xi}_k) - L(\boldsymbol{\theta}_k)$ and $(\nabla L(\boldsymbol{\theta}_k))^\top \boldsymbol{\xi}_{k+1}$ are close. Therefore, forward gradient descent is related to the zero-order method

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha_{k+1}\big(L(\boldsymbol{\theta}_k + \boldsymbol{\xi}_k) - L(\boldsymbol{\theta}_k)\big)\boldsymbol{\xi}_k, \tag{4.2.4}$$

[92]. Consequently, forward gradient descent can be viewed as an intermediate step between gradient descent, with full access to the gradient, and zero-order methods that are solely based on (randomly) perturbed function evaluations.

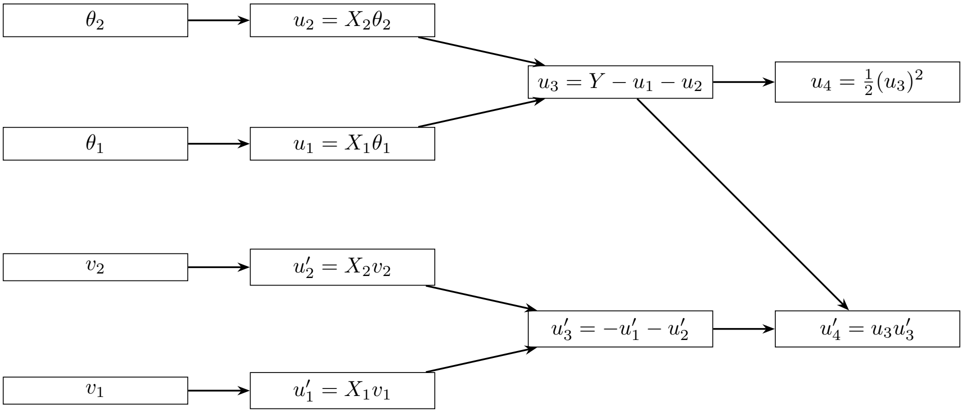


Figure 4.2.1: Computional graphs for computing in a forward pass $L(\boldsymbol{\theta}) = \frac{1}{2}(Y - X_1\theta_1 - X_2\theta_2)^2$ (upper half) and $(\nabla L(\boldsymbol{\theta}))^\top \mathbf{v}$ (lower half).

We now comment on the biological plausibility of forward gradient descent. As mentioned in the introduction, it is widely accepted that the brain cannot perform (full) gradient descent. The backpropagation algorithm decomposes the computation of the gradient in a forward pass and a backward pass. The forward pass evaluates the loss for a training sample by sending signal through the network. This is biologically plausible. For a given vector $\mathbf{v}$, it is even possible to compute both $L(\boldsymbol{\theta}_k)$ and $\big(\nabla L(\boldsymbol{\theta}_k)\big)^\top \mathbf{v}$ in one forward pass, [13, 117, 12]. The construction can be conveniently explained for two variables $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top$, see Figure 4.2.1. The loss function $L(\boldsymbol{\theta}) = \frac{1}{2}(Y - X_1\theta_1 - X_2\theta_2)^2$

is implemented by first computing $u_1 = X_1\theta_1$ and $u_2 = X_2\theta_2$ in parallel. Subsequently, one can infer $u_3 = Y - u_1 - u_2 = Y - X_1\theta_1 - X_2\theta_2$ and $u_4 = \frac{1}{2}(u_3)^2 = L(\boldsymbol{\theta})$. For a given vector $\mathbf{v} = (v_1, v_2)^\top$, the update value $(\nabla L(\boldsymbol{\theta}))^\top \mathbf{v}$ in the forward gradient descent routine can be computed from v_1, v_2, and $u_3 = Y - X_1\theta_1 - X_2\theta_2$. Indeed, after computing X_1v_1 and X_2v_2 in a first step, one can compute $u_3' = -X_1v_1 - X_2v_2$ and finally $u_4' = u_3u_3' = (Y - X_1\theta_1 - X_2\theta_2)(-X_1v_1 - X_2v_2) = -(Y - \mathbf{X}^\top\boldsymbol{\theta})\mathbf{X}^\top\mathbf{v} = (\nabla L(\boldsymbol{\theta}))^\top\mathbf{v}$. For more background on the implementation, see for instance [12].

In [128], it has been shown that under appropriate conditions, Hebbian learning of excitatory neurons in biological neural networks leads to a zeroth-order learning rule that has the same structure as (4.2.4).

To complete this section, we briefly compare forward gradient descent with feedback alignment as both methods are motivated by biological learning and are based on additional randomness. Inspired by biological learning, feedback alignment proposes to replace the learned weights in the backward pass by random weights chosen at the start of the training procedure [88, 89]. The so-called direct feedback alignment method goes even further: instead of back-propagating the gradient through all the layers of the network by the chain-rule, layers are updated with the gradient of the output layer multiplied with a fixed random weight matrix [106, 84]. (Direct) feedback alignment causes the forward weights to change in such a way that the true gradient of the network weights and the substitutes used in the update rule become more aligned [88, 106, 89]. The linear model can be viewed as neural network without hidden layers. The absence of layers means that in the backward step, no weight information is transported between different layers. As a consequence, both feedback alignment and direct feedback alignment collapse in the linear model into standard gradient descent. The conclusion is that feedback alignment and forward gradient descent are not comparable. The argument also shows that to unveil nontrivial statistical properties of feedback alignment, one has to go beyond the linear model. We leave the statistical analysis as an open problem.

4.3 Convergence rates in the linear regression model

We analyze weight-perturbed forward gradient descent for data generated from the d-dimensional linear regression with Gaussian random design. In this framework, we observe i.i.d. pairs $(\mathbf{X}_i, Y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, 2, \ldots$ satisfying

$$\mathbf{X}_i \sim \mathcal{N}(0, \Sigma), \quad Y_i = \mathbf{X}_i^\top \boldsymbol{\theta}_\star + \epsilon_i, \quad i = 1, 2, \ldots \tag{4.3.1}$$

with $\boldsymbol{\theta}_\star$ the unknown d-dimensional regression vector, Σ an unknown covariance matrix, and independent noise variables ϵ_i with mean zero and variance one.

For the analysis, we consider the squared loss $L(\boldsymbol{\theta}_k, \mathbf{X}_k, Y_k) = \frac{1}{2}(Y_k - \mathbf{X}_k^\top \boldsymbol{\theta}_k)^2$. The gradient is given by

$$\nabla L(\boldsymbol{\theta}_k) = -\big(Y_k - \mathbf{X}_k^\top \boldsymbol{\theta}_k\big)\mathbf{X}_k. \tag{4.3.2}$$

We now analyze the forward gradient estimator assuming that the initial value $\boldsymbol{\theta}_0$ can be random or deterministic but should be independent of the data. We employ a similar proving strategy as in the recent analysis of dropout in the linear model in [31]. In particular, we will derive a recursive formula for $\mathbb{E}\left[(\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star)(\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star)^\top\right]$. In contrast to this work, we consider a different form of noise and non-constant learning rates.

The first result shows that forward gradient descent does gradient descent in expectation.

Theorem 4.3.1. *We have* $\mathbb{E}[\boldsymbol{\theta}_k] - \boldsymbol{\theta}_\star = \big(\mathbf{I}_d - \alpha_k\Sigma\big)\big(\mathbb{E}[\boldsymbol{\theta}_{k-1}] - \boldsymbol{\theta}_\star\big)$ *and thus*

$$\mathbb{E}[\boldsymbol{\theta}_k] = \boldsymbol{\theta}_\star + \left(\prod_{\ell=1}^{k}(\mathbf{I}_d - \alpha_\ell\Sigma)\right)\big(\mathbb{E}[\boldsymbol{\theta}_0] - \boldsymbol{\theta}_\star\big). \tag{4.3.3}$$

The proof does not exploit the Gaussian design and only requires that $\mathbf{X}_i$ is centered and has covariance matrix Σ. The exogenous randomness induced by $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots$ disappears in the expected values but heavily influences the recursive expressions for the squared expectations.

Theorem 4.3.2. *Consider forward gradient descent* (4.2.2). *If* $A_k := \mathbb{E}\left[(\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star)(\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star)^\top\right]$, *then*

$$\begin{aligned}A_k =&(\mathbf{I}_d - \alpha_k\Sigma)A_{k-1}(\mathbf{I}_d - \alpha_k\Sigma)\\ &+ 3\alpha_k^2\Sigma A_{k-1}\Sigma + 2\alpha_k^2\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)\big]\Sigma + 2\alpha_k^2\Sigma\\ &+ 2\alpha_k^2 \operatorname{tr}\big(\Sigma A_{k-1}\Sigma\big)\mathbf{I}_d + \alpha_k^2\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)\big]\operatorname{tr}\big(\Sigma\big)\mathbf{I}_d\\ &+ \alpha_k^2\operatorname{tr}(\Sigma)\mathbf{I}_d.\end{aligned}$$

Since A_k depends on $\boldsymbol{\theta}_k^2$, the fourth moments of the design vectors $\mathbf{X}_i$ and the exogenous random vectors $\boldsymbol{\xi}_k$ play a role in this equation.

The risk $\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big]$ is the trace of the matrix A_k. Setting

$$\kappa(\Sigma) := \frac{\|\Sigma\|_S}{\lambda_{\min}(\Sigma)}$$

for the condition number and building on Theorem 4.3.2, we can establish the following risk bound for forward gradient descent.

Theorem 4.3.3 (Mean squared error)**.** *Consider forward gradient descent* (4.2.2) *and assume that* Σ *is positive definite. For constant* $a > 2$, *choosing the learning rate*

$$\alpha_k = \frac{a\lambda_{\min}(\Sigma)}{k\lambda_{\min}^2(\Sigma) + a\|\Sigma\|_S^2(d+2)^2}, \quad k = 1, 2, \ldots, \tag{4.3.4}$$

yields

$$\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] \leq \left(\frac{1 + a\kappa^2(\Sigma)(d+2)^2}{k + a\kappa^2(\Sigma)(d+2)^2}\right)^a \mathbb{E}\big[\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_2^2\big] + \frac{2ea\kappa(\Sigma)(d+2)^2}{\lambda_{\min}(\Sigma)(k + a\kappa^2(\Sigma)(d+2)^2)}.$$

Alternatively, the upper bound of Theorem 4.3.3 can be written as

$$\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] \leq \Big(1 - a^{-1}\lambda_{\min}(\Sigma)(k-1)\alpha_k\Big)^a \mathbb{E}\big[\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_2^2\big] + 2e\kappa(\Sigma)(d+2)^2\alpha_k.$$

In the upper bound, the risk $\mathbb{E}\big[\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_2^2\big]$ of the initial estimate $\boldsymbol{\theta}_0$ appears. A realistic scenario is that the entries of $\boldsymbol{\theta}_\star$ and $\boldsymbol{\theta}_0$ are all of order one. In this case, the inequality $\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_2^2 \leq d\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_\infty^2$ shows that the risk of the initial estimate will scale with the number of parameters d. Taking $a = \log(d)$ (for $d \geq 8 > e^2$ such that $a > \log(e^2) = 2$), Theorem 4.3.3 implies that

$$\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] \lesssim d\left(\frac{d^2\log(d)}{k}\right)^{\log(d)} \mathbb{E}\big[\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_\infty^2\big] + \frac{d^2\log(d)}{k}.$$

For $k_\star = e^2d^2\log(d)$, $d^2\log(d)/k_\star = e^{-2}$ and $d(d^2\log(d)/k_\star)^{\log(d)} = 1/d$. Since $d > e^2$, this means that $d\big(d^2\log(d)/k_\star\big)^{\log(d)} < d^2\log(d)/k_\star$. Moreover, $k^{-\log(d)}$ tends faster to zero than k^{-1} as $k \to \infty$. So, for $k \geq k_\star = e^2d^2\log(d)$,

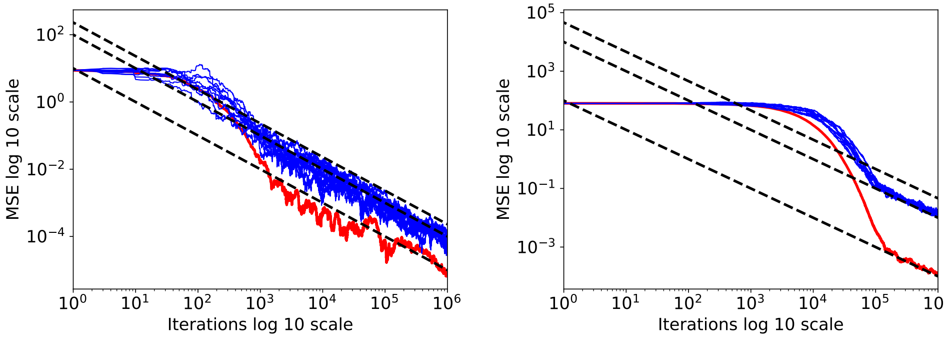
$$d\left(\frac{d^2\log(d)}{k}\right)^{\log(d)} \mathbb{E}\big[\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_\infty^2\big] + \frac{d^2\log(d)}{k} \leq \frac{d^2\log(d)}{k}\Big(1 + \mathbb{E}\big[\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_\infty^2\big]\Big). \tag{4.3.5}$$

The rate for $k \geq e^2d^2\log(d)$ is thus $d^2\log(d)/k$. This means that forward gradient descent has dimension dependence $d^2\log(d)$. This is by a factor $d\log(d)$ worse than the minimax rate for the linear regression problem, [144, 63, 98]. In contrast, methods that have access to the gradient can achieve optimal dimension dependence in the rate, [114, 82]. The obtained convergence rate is in line with results for zero-order methods, which show that for convex optimization problems these methods have a higher dimension dependence, [41, 92, 105].

We believe that faster convergence rates are obtainable if the same datapoint is assessed several times. This means that each data point is used for several updates of the forward gradient $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha_{k+1}\big(\nabla L(\boldsymbol{\theta}_k)\big)^\top \boldsymbol{\xi}_{k+1}\boldsymbol{\xi}_{k+1}$, for instance by running multiple epochs. However, in every iteration a new random direction $\boldsymbol{\xi}_{k+1}$ is sampled. We expect that if every data point is used $m \leq d$ times, one should be able to achieve the convergence rate $d^2/(km)$, up to some logarithmic terms. If this is true and if m is of the order of d, one could even recover the minimax rate d/k. Using the same datapoints multiple times induces additional dependence among the parameter updates. To deal with this dependence is the key challenge to establish the convergence rate $d^2/(km)$.

Assuming that the covariance matrix Σ is positive definite is standard for linear regression with random design [63, 98, 132].

For $k \gtrsim d^2$, the decrease of the learning rate α_k is of the order $1/k$, which is the standard choice [81, 55, 17]. A constant learning rate is used for Ruppert-Polyak averaging in [114, 55]. For least squares linear regression, it is possible to achieve (near) optimal convergence with a constant (universal) stepsize [6]. Conditions under which a constant (universal) stepsize in more general settings than linear least squares works or fails are investigated in [82].



(a) $d = 10$

(b) $d = 100$

Figure 4.3.1: Comparison of the MSE of forward gradient descent (blue) and SGD (red) for dimensions $d = 10$ and $d = 100$. The upper dashed line is $k \mapsto d^2 \log(d)/k$, the middle dashed line is $k \mapsto d^2/k$, and the lower dashed line is $k \mapsto d/k$.

In a small simulation study, we investigated whether there is a discrepancy between the derived convergence rates and the empirical decay of the risk. For dimensions $d = 10$ and $d = 100$, data according to (4.3.1) with $\Sigma = \mathbf{I}_d$ are generated. On these data, we run ten times weight perturbed forward gradient descent (4.2.2), and compare the mean squared errors (MSEs) to one realization of SGD (4.2.1). For all simulations of forward gradient descent and SGD, we use the same initialization $\boldsymbol{\theta}_0$, drawn from a $\mathcal{N}(0, \mathbf{I}_d)$ distribution, and the learning rate α_k specified in (4.3.4) with $a = \log(d)$. Thus, only the random perturbation vectors $\boldsymbol{\xi}_k$ in the forward gradient descent schemes differ across different runs. The outcomes are reported in Figure 4.3.1. For each of the 10+1 simulations, we report on a log-log scale the MSE for the first one million iterations. The upper dashed line gives the derived convergence rate $k \mapsto d^2 \log(d)/k$, the middle dashed line is d^2/k, and the lower dashed line is d/k. The ten paths from the ten forward gradient descent runs are shown in blue. The path from the SGD is displayed in red. We see three regimes. In the first regime, the risk remains nearly constant. For dimension $d = 100$, this is true up to the first ten thousand of iterations. Afterwards there is a sudden decrease of the risk. Eventually, for large number of iterations k, the MSE of forward gradient descent concentrates near the line $k \mapsto d^2/k$, while the MSE of SGD concentrates around $k \mapsto d/k$. This suggest that up to the $\log(d)$-factor, the derived theory does in fact describe the rate of the MSE. Equation (4.3.5) predicts that the rate $d^2 \log(d)/k$ will occur for $k \geq k_\star = e^2 d^2 \log(d)$. For $d = 10$, $k_\star \approx 1.7 \times 10^3$ and for $d = 100$, $k_\star \approx 3.4 \times 10^5$. Thus, in terms of orders of magnitude, there is a close agreement between theory and simulations.

Starting with a good initializer that lies already in the neighborhood of the true parameter, one can avoid the long burn-in time in the beginning. Otherwise, it remains an open problem, whether one can modify the procedure such that also for smaller values of k, the risk behaves more like $d^2 \log(d)/k$.

Python code is available on Github [24].

4.4 Proofs

Proof of Theorem 4.3.1. By (4.3.2) and the linear regression model $Y_{k-1} = \mathbf{X}_{k-1}^\top \boldsymbol{\theta}_\star + \epsilon_{k-1}$, we have

$$\begin{aligned} \nabla L(\boldsymbol{\theta}_{k-1}) &= -(Y_{k-1} - \mathbf{X}_{k-1}^\top \boldsymbol{\theta}_{k-1})\mathbf{X}_{k-1} \\ &= -(\mathbf{X}_{k-1}^\top (\boldsymbol{\theta}_\star - \boldsymbol{\theta}_{k-1}) + \epsilon_{k-1})\mathbf{X}_{k-1} \\ &= -\epsilon_{k-1}\mathbf{X}_{k-1} - \mathbf{X}_{k-1}\mathbf{X}_{k-1}^\top (\boldsymbol{\theta}_\star - \boldsymbol{\theta}_{k-1}). \end{aligned} \tag{4.4.1}$$

Since $\mathbb{E}[\mathbf{X}_{k-1}\mathbf{X}_{k-1}^\top] = \Sigma$, $\mathbb{E}[\epsilon_{k-1}] = 0$, and $\mathbf{X}_{k-1}, \epsilon_{k-1}, \boldsymbol{\theta}_{k-1}$ are jointly independent, we obtain

$$\begin{aligned} \mathbb{E}\big[\nabla L(\boldsymbol{\theta}_{k-1}) \,\big|\, \boldsymbol{\theta}_{k-1}\big] &= \mathbb{E}\big[-\epsilon_{k-1}\mathbf{X}_{k-1} - \mathbf{X}_{k-1}\mathbf{X}_{k-1}^\top (\boldsymbol{\theta}_\star - \boldsymbol{\theta}_{k-1}) \,\big|\, \boldsymbol{\theta}_{k-1}\big] \\ &= -\Sigma(\boldsymbol{\theta}_\star - \boldsymbol{\theta}_{k-1}). \end{aligned} \tag{4.4.2}$$

Combined with (4.2.3), we find

$$\mathbb{E}\big[\boldsymbol{\theta}_k\big] = \mathbb{E}\big[\boldsymbol{\theta}_{k-1}\big] - \alpha_k \mathbb{E}\big[\nabla L(\boldsymbol{\theta}_{k-1})\big] = \mathbb{E}\big[\boldsymbol{\theta}_{k-1}\big] + \alpha_k \Sigma \mathbb{E}\big[\boldsymbol{\theta}_\star - \boldsymbol{\theta}_{k-1}\big].$$

The true parameter $\boldsymbol{\theta}_\star$ is deterministic. Subtracting $\boldsymbol{\theta}_\star$ on both sides, yields the claimed identity $\mathbb{E}[\boldsymbol{\theta}_k] - \boldsymbol{\theta}_\star = \big(\mathbf{I}_d - \alpha_k \Sigma\big)\big(\mathbb{E}[\boldsymbol{\theta}_{k-1}] - \boldsymbol{\theta}_\star\big)$.

□

4.4.1 Proof of Theorem 4.3.2

Lemma 4.4.1. *If* $\mathbf{Z} \sim \mathcal{N}(0, \Gamma)$ *is a d-dimensional random vector and* $\mathbf{U}$ *is a d-dimensional random vector that is independent of* $\mathbf{Z}$, *then*

$$\mathbb{E}\big[(\mathbf{U}^\top \mathbf{Z})^2 \mathbf{Z}\mathbf{Z}^\top\big] = 2\Gamma \mathbb{E}\big[\mathbf{U}\mathbf{U}^\top\big]\Gamma + \mathbb{E}\big[\mathbf{U}^\top \Gamma \mathbf{U}\big]\Gamma.$$

Proof. Because $\mathbf{U}$ and $\mathbf{Z}$ are independent, the (i,j)-th entry of the $d \times d$ matrix $\mathbb{E}\big[(\mathbf{U}^\top \mathbf{Z})^2 \mathbf{Z}\mathbf{Z}^\top\big]$ is

$$\sum_{\ell,m=1}^{d} \mathbb{E}\big[U_\ell U_m\big]\mathbb{E}\big[Z_\ell Z_m Z_i Z_j\big].$$

Since $\mathbf{Z} \sim \mathcal{N}(0, \Gamma)$,

$$\mathbb{E}\big[Z_\ell Z_m Z_i Z_j\big] = \Gamma_{\ell,m}\Gamma_{i,j} + \Gamma_{\ell,i}\Gamma_{m,j} + \Gamma_{\ell,j}\Gamma_{m,i},$$

see for instance the example at the end of Section 2 in [143]. Thus

$$\sum_{\ell,m=1}^{d} \mathbb{E}\big[U_\ell U_m\big]\mathbb{E}\big[Z_\ell Z_m Z_i Z_j\big] = \sum_{\ell,m=1}^{d} \mathbb{E}\big[U_\ell U_m\big]\big(\Gamma_{\ell,m}\Gamma_{i,j} + \Gamma_{\ell,i}\Gamma_{m,j} + \Gamma_{\ell,j}\Gamma_{m,i}\big).$$

Because of

$$\sum_{\ell,m=1}^{d} \mathbb{E}\big[U_\ell U_m\big]\Gamma_{\ell,m}\Gamma_{i,j} = \sum_{\ell,m=1}^{d} \mathbb{E}\big[U_\ell \Gamma_{\ell,m} U_m\big]\Gamma_{i,j} = \mathbb{E}\left[\mathbf{U}^\top \Gamma \mathbf{U} \Gamma_{i,j}\right],$$

$$\sum_{\ell,m=1}^{d} \mathbb{E}\big[U_\ell U_m\big]\Gamma_{\ell,i}\Gamma_{m,j} = \sum_{\ell,m=1}^{d} \mathbb{E}\big[U_\ell \Gamma_{\ell,i} U_m \Gamma_{m,j}\big] = \mathbb{E}\left[\big(\mathbf{U}^\top \Gamma\big)_i \big(\mathbf{U}^\top \Gamma\big)_j\right],$$

and

$$\sum_{\ell,m=1}^{d} \mathbb{E}\big[U_\ell U_m\big]\Gamma_{\ell,j}\Gamma_{m,i} = \sum_{\ell,m=1}^{d} \mathbb{E}\big[U_m \Gamma_{m,i} U_\ell \Gamma_{\ell,j}\big] = \mathbb{E}\left[\big(\mathbf{U}^\top \Gamma\big)_i \big(\mathbf{U}^\top \Gamma\big)_j\right],$$

the (i,j)-th entry of the matrix $\mathbb{E}\left[(\mathbf{U}^\top\mathbf{Z})^2\mathbf{Z}\mathbf{Z}^\top\right]$ is

$$2\mathbb{E}\left[\big(\mathbf{U}^\top\Gamma\big)_i\big(\mathbf{U}^\top\Gamma\big)_j\right]+\mathbb{E}\left[\mathbf{U}^\top\Gamma\mathbf{U}\Gamma_{i,j}\right].$$

For a vector $\mathbf{a}=(a_1,\dots,a_d)^\top$, the scalar a_ia_j is the (i,j)-th entry of the matrix $\mathbf{a}\mathbf{a}^\top$. Combined with the previous display, the result follows. □

Proof of Theorem 4.3.2. As Theorem 4.3.2 only involves one update step, we can simplify the notation by dropping the index k and analyzing $\boldsymbol{\theta}''=\boldsymbol{\theta}'-\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}$ for one data point $(\mathbf{X},Y)$ and independent $\boldsymbol{\xi}\sim\mathcal{N}(0,I_d)$. With $A':=\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\big]$ and $A'':=\mathbb{E}\big[(\boldsymbol{\theta}''-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}''-\boldsymbol{\theta}_\star)^\top\big]$, we then have to prove that

$$\begin{aligned}A'' =&(\mathbf{I}_d-\alpha\Sigma)A'(\mathbf{I}_d-\alpha\Sigma)+3\alpha^2\Sigma A'\Sigma+2\alpha^2\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)\big]\Sigma+2\alpha^2\Sigma\\ &+2\alpha^2\operatorname{tr}\big(\Sigma A'\Sigma\big)\mathbf{I}_d+\alpha^2\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)\big]\operatorname{tr}\big(\Sigma\big)\mathbf{I}_d+\alpha^2\operatorname{tr}(\Sigma)\mathbf{I}_d.\end{aligned}$$

Substituting the update rule (4.2.2) in A_k gives by the linearity of the transpose that

$$\begin{aligned}A''&=\mathbb{E}\big[(\boldsymbol{\theta}''-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}''-\boldsymbol{\theta}_\star)^\top\big]\\ &=\mathbb{E}\left[\Big(\boldsymbol{\theta}'-\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}-\boldsymbol{\theta}_\star\Big)\Big(\boldsymbol{\theta}'-\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}-\boldsymbol{\theta}_\star\Big)^\top\right]\\ &=A'-\alpha\mathbb{E}\left[\Big(\boldsymbol{\theta}-\boldsymbol{\theta}_\star\Big)\Big(\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)^\top\right]-\alpha\mathbb{E}\left[\Big(\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)\Big(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star\Big)^\top\right]\\ &\quad+\mathbb{E}\left[\Big(\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)\Big(\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)^\top\right].\end{aligned}\tag{4.4.3}$$

First, consider the terms with the minus sign in the above expression. The random vector $\boldsymbol{\xi}$ is independent of all other randomness and hence $\mathbb{E}_{\boldsymbol{\xi}}\Big[\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big]=\nabla L(\boldsymbol{\theta}')$. Moreover, together with (4.4.2),

$$\begin{aligned}\mathbb{E}\left[\Big(\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)\Big(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star\Big)^\top\,\middle|\,\boldsymbol{\theta}'\right]&=\mathbb{E}\big[\nabla L(\boldsymbol{\theta}')\,|\,\boldsymbol{\theta}'\big](\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\\ &=\Sigma(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top.\end{aligned}$$

Taking the transpose and tower rule, we find

$$\begin{aligned}&-\alpha\mathbb{E}\left[\Big(\boldsymbol{\theta}-\boldsymbol{\theta}_\star\Big)\Big(\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)^\top\right]-\alpha\mathbb{E}\left[\Big(\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)\Big(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star\Big)^\top\right]\\ &=-\alpha\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\big]\Sigma-\alpha\Sigma\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\big].\end{aligned}\tag{4.4.4}$$

In a next step, we derive an expression for $\mathbb{E}\Big[\Big(\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)\Big(\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)^\top\Big]$. Since $\boldsymbol{\xi}\sim\mathcal{N}(0,\mathbf{I}_d)$ is independent of $\nabla L(\boldsymbol{\theta}')$ we can apply Lemma 4.4.1 to derive

$$\begin{aligned}&\mathbb{E}\Big[\Big(\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)\Big(\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)^\top\Big]\\&=\alpha^2\mathbb{E}\Big[\Big(\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\Big)^2\boldsymbol{\xi}\boldsymbol{\xi}^\top\Big]\\&=2\alpha^2\mathbb{E}\Big[\big(\nabla L(\boldsymbol{\theta}')\big)\big(\nabla L(\boldsymbol{\theta}')\big)^\top\Big]+\alpha^2\mathbb{E}\Big[\big(\nabla L(\boldsymbol{\theta}')\big)^\top\big(\nabla L(\boldsymbol{\theta}')\big)\Big]\mathbf{I}_d\\&=2\alpha^2\mathbb{E}\Big[\big(\nabla L(\boldsymbol{\theta}')\big)\big(\nabla L(\boldsymbol{\theta}')\big)^\top\Big]+\alpha^2\operatorname{tr}\Big(\mathbb{E}\Big[\big(\nabla L(\boldsymbol{\theta}')\big)\big(\nabla L(\boldsymbol{\theta}')\big)^\top\Big]\Big)\mathbf{I}_d.\end{aligned}\tag{4.4.5}$$

Arguing as for (4.4.1) gives $\nabla L(\boldsymbol{\theta}')=-\epsilon\mathbf{X}-\mathbf{X}\mathbf{X}^\top(\boldsymbol{\theta}_\star-\boldsymbol{\theta}')$ and this yields

$$\mathbb{E}\Big[\big(\nabla L(\boldsymbol{\theta}')\big)\big(\nabla L(\boldsymbol{\theta}')\big)^\top\Big]=\mathbb{E}\Big[\mathbb{E}_\epsilon\Big[\big(\epsilon\mathbf{X}+\mathbf{X}\mathbf{X}^\top(\boldsymbol{\theta}_\star-\boldsymbol{\theta}')\big)\big(\epsilon\mathbf{X}+\mathbf{X}\mathbf{X}^\top(\boldsymbol{\theta}_\star-\boldsymbol{\theta}')\big)^\top\Big]\Big].$$

Because ϵ has mean zero and variance one and is independent of $(\mathbf{X},\boldsymbol{\theta}')$, we conclude that

$$\begin{aligned}\mathbb{E}\Big[\big(\nabla L(\boldsymbol{\theta}')\big)\big(\nabla L(\boldsymbol{\theta}')\big)^\top\Big]&=\mathbb{E}\Big[\big(\mathbf{X}\mathbf{X}^\top(\boldsymbol{\theta}_\star-\boldsymbol{\theta}')\big)\big(\mathbf{X}\mathbf{X}^\top(\boldsymbol{\theta}_\star-\boldsymbol{\theta}')\big)^\top+\mathbf{X}\mathbf{X}^\top\Big]\\&=\mathbb{E}\Big[\big(\mathbf{X}^\top(\boldsymbol{\theta}_\star-\boldsymbol{\theta}')\big)^2\mathbf{X}\mathbf{X}^\top\Big]+\Sigma,\end{aligned}\tag{4.4.6}$$

where for the last equality we used that $\mathbf{X}^\top(\boldsymbol{\theta}_\star-\boldsymbol{\theta}')$ is a scalar and that $\mathbf{X}\sim\mathcal{N}(0,\Sigma)$. Since $\mathbf{X}\sim\mathcal{N}(0,\Sigma)$ is independent of $\boldsymbol{\theta}'$ we get by Lemma 4.4.1 that

$$\mathbb{E}\Big[\big(\mathbf{X}^\top(\boldsymbol{\theta}_\star-\boldsymbol{\theta}')\big)^2\mathbf{X}\mathbf{X}^\top\Big]=2\Sigma\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\big]\Sigma+\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)\big]\Sigma.$$

Substituting this in (4.4.6) and (4.4.5) yields

$$\begin{aligned}&\mathbb{E}\Big[\Big(\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)\Big(\alpha\big(\nabla L(\boldsymbol{\theta}')\big)^\top\boldsymbol{\xi}\boldsymbol{\xi}\Big)^\top\Big]\\&=4\alpha^2\Sigma\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\big]\Sigma+2\alpha^2\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)\big]\Sigma+2\alpha^2\Sigma\\&+2\alpha^2\operatorname{tr}\Big(\Sigma\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\big]\Sigma\Big)\mathbf{I}_d+\alpha^2\operatorname{tr}\Big(\mathbb{E}\big[(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}'-\boldsymbol{\theta}_\star)\big]\Sigma\Big)\mathbf{I}_d\\&+\alpha^2\operatorname{tr}(\Sigma)\mathbf{I}_d.\end{aligned}\tag{4.4.7}$$

Combining (4.4.3) with (4.4.4) and (4.4.7) yields the statement of the theorem. □

4.4.2 Proof of Theorem 4.3.3

For two vectors $\mathbf{u}, \mathbf{v}$ of the same length, $\operatorname{tr}(\mathbf{u}\mathbf{v}^\top) = \mathbf{u}^\top\mathbf{v}$. Thus, $\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] = \operatorname{tr}\big(\mathbb{E}\big[(\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star)(\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star)^\top\big]\big)$. Together with Theorem 4.3.2, $\operatorname{tr}(\mathbf{I}_d) = d$ and $\operatorname{tr}(AB) = \operatorname{tr}(BA)$ for square matrices A and B of the same size, this yields

$$\begin{aligned}
\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] &= \operatorname{tr}\Big((\mathbf{I}_d - \alpha_k\Sigma)\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\big](\mathbf{I}_d - \alpha_k\Sigma)\Big) \\
&\quad + 3\alpha_k^2 \operatorname{tr}\Big(\Sigma\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\big]\Sigma\Big) \\
&\quad + 2\alpha_k^2 \operatorname{tr}\Big(\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)\big]\Sigma\Big) + 2\alpha_k^2 \operatorname{tr}\big(\Sigma\big) \\
&\quad + 2\alpha_k^2 \operatorname{tr}\Big(\Sigma\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\big]\Sigma\Big) \operatorname{tr}\big(\mathbf{I}_d\big) \\
&\quad + \alpha_k^2\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)\big] \operatorname{tr}\big(\Sigma\big) \operatorname{tr}\big(\mathbf{I}_d\big) \\
&\quad + \alpha_k^2 \operatorname{tr}(\Sigma) \operatorname{tr}\big(\mathbf{I}_d\big) \\
&= \mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top(\mathbf{I}_d - 2\alpha_k\Sigma)^\top(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)\big] \\
&\quad + 2(d+2)\alpha_k^2 \operatorname{tr}\Big(\Sigma\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\big]\Sigma\Big) \\
&\quad + (d+2)\alpha_k^2\Big(\mathbb{E}\big[(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)^\top\Sigma(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)\big] \operatorname{tr}\big(\Sigma\big) + \operatorname{tr}\big(\Sigma\big)\Big).
\end{aligned} \tag{4.4.8}$$

If λ is an eigenvalue of Σ then $(1 - 2\alpha_k\lambda)$ is an eigenvalue of $\mathbf{I}_d - 2\alpha_k\Sigma$. By assumption, $0 < \alpha_k \leq \lambda_{\min}(\Sigma)/\big(2\|\Sigma\|_S^2\big) \leq 1/\big(2\lambda_{\max}(\Sigma)\big)$ and therefore the matrix $\mathbf{I}_d - 2\alpha_k\Sigma$ is positive semi-definite and $(1 - 2\alpha_k\lambda_{\min}(\Sigma))$ is the largest eigenvalue.

For a positive semi-definite matrix A and a vector $\mathbf{v}$, the min-max theorem states that $\mathbf{v}^\top A\mathbf{v} \leq \lambda_{\max}(A)\|\mathbf{v}\|_2^2 = \|A\|_S\|\mathbf{v}\|_2^2$. Using that for a vector $\mathbf{x}$ it holds that $\operatorname{tr}(\mathbf{x}\mathbf{x}^\top) = \mathbf{x}^\top\mathbf{x}$, with $\mathbf{x} = \Sigma(\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star)$ in (4.4.8) and applying $\mathbf{v}^\top A\mathbf{v} \leq \|A\|_S\|\mathbf{v}\|_2^2$ with $\mathbf{v} = \boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star$ and $A \in \{\Sigma, \mathbf{I}_d - 2\alpha_k\Sigma, \Sigma^2\}$, yields

$$\begin{aligned}
&\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] \leq \big(1 - 2\alpha_k\lambda_{\min}(\Sigma)\big)\mathbb{E}\big[\|\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star\|_2^2\big] \\
&\quad + (d+2)\alpha_k^2\bigg(\operatorname{tr}(\Sigma)\|\Sigma\|_S\mathbb{E}\big[\|\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star\|_2^2\big] + 2\|\Sigma\|_S^2\mathbb{E}\big[\|\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star\|_2^2\big] + \operatorname{tr}(\Sigma)\bigg).
\end{aligned}$$

The spectral norm of a positive semi-definite matrix is equal to the largest eigenvalue and so $\operatorname{tr}(\Sigma) = \sum_{i=1}^d \lambda_i \leq d\lambda_{\max} = d\|\Sigma\|_S$. Therefore,

$$\begin{aligned}
\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] &\leq \Big(1 - 2\alpha_k\lambda_{\min}(\Sigma) + \|\Sigma\|_S^2(d+2)^2\alpha_k^2\Big)\mathbb{E}\big[\|\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star\|_2^2\big] \\
&\quad + \|\Sigma\|_S(d+2)^2\alpha_k^2.
\end{aligned}$$

Using that $\alpha_k \leq \lambda_{\min}(\Sigma)/\big(\|\Sigma\|_S^2(d+2)^2\big)$ yields

$$\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] \leq \big(1-\alpha_k\lambda_{\min}(\Sigma)\big)\mathbb{E}\big[\|\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_\star\|_2^2\big] + \|\Sigma\|_S(d+2)^2\alpha_k^2.$$

Rewritten in non-recursive, we obtain

$$\begin{aligned}\mathbb{E}\big[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2\big] \leq & \mathbb{E}\big[\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_2^2\big]\prod_{\ell=1}^{k}\big(1-\alpha_\ell\lambda_{\min}(\Sigma)\big) \\ & + \|\Sigma\|_S(d+2)^2\sum_{m=0}^{k-1}\alpha_{k-m}^2\prod_{\ell=k-m+1}^{k}\big(1-\alpha_\ell\lambda_{\min}(\Sigma)\big),\end{aligned} \tag{4.4.9}$$

where we use the convention that the (empty) product over zero terms is assigned the value 1. For ease of notation define $c_d := a\kappa^2(\Sigma)(d+2)^2$, with condition number $\kappa(\Sigma) = \|\Sigma\|_S/\lambda_{\min}(\Sigma)$. From the definition of α_k, (4.3.4), it follows that $\alpha_k = \frac{a}{\lambda_{\min}(\Sigma)} \cdot \frac{1}{k+c_d}$. Using that for all real numbers x it holds that $1+x \leq e^x$, we get that for all integers $k^* < k$,

$$\prod_{\ell=k^*}^{k}\big(1-\alpha_\ell\lambda_{\min}(\Sigma)\big) \leq \exp\Bigg(-\lambda_{\min}(\Sigma)\sum_{\ell=k^*}^{k}\alpha_\ell\Bigg) = \exp\Bigg(-a\sum_{\ell=k^*}^{k}\frac{1}{\ell+c_d}\Bigg). \tag{4.4.10}$$

The function $x \mapsto 1/(x+c)$ is monotone decreasing for $x > 0$ and $c \geq 0$ and thus,

$$\begin{aligned}\sum_{\ell=k^*}^{k}\frac{1}{\ell+c_d} &\geq \sum_{\ell=k^*}^{k}\int_{\ell}^{\ell+1}\frac{1}{x+c_d}dx \\ &= \int_{k^*}^{k+1}\frac{1}{x+c_d}dx \\ &= \log(k+1+c_d) - \log(k^*+c_d) \\ &= \log\Big(\frac{k+1+c_d}{k^*+c_d}\Big).\end{aligned} \tag{4.4.11}$$

Using (4.4.10) and (4.4.11) with $k^* = 1$ gives

$$\prod_{\ell=1}^{k}\big(1-\alpha_\ell\lambda_{\min}(\Sigma)\big) \leq \exp\Bigg(-a\log\Big(\frac{k+1+c_d}{1+c_d}\Big)\Bigg) = \bigg(\frac{1+c_d}{k+1+c_d}\bigg)^a. \tag{4.4.12}$$

Using (4.4.10) and (4.4.11) with $k^* = k - m + 1$ gives

$$\begin{aligned}
&\sum_{m=0}^{k-1} \alpha_{k-m}^2 \prod_{\ell=k-m+1}^{k} \big(1 - \alpha_\ell \lambda_{\min}(\Sigma)\big) \\
&\leq \frac{a^2}{\lambda_{\min}^2(\Sigma)} \sum_{m=0}^{k-1} \frac{1}{\big((k-m)+c_d\big)^2} \left(\frac{k-m+1+c_d}{k+1+c_d}\right)^a \\
&= \frac{a^2}{\lambda_{\min}^2(\Sigma)(k+1+c_d)^a} \sum_{m=0}^{k-1} \frac{\big(k-m+1+c_d\big)^a}{\big((k-m)+c_d\big)^2} \\
&= \frac{a^2}{\lambda_{\min}^2(\Sigma)(k+1+c_d)^a} \sum_{m=1}^{k} \frac{\big(m+1+c_d\big)^a}{\big(m+c_d\big)^2}.
\end{aligned} \tag{4.4.13}$$

Observe that $c_d = a\kappa^2(\Sigma)(d+2)^2 \geq a$. This gives us that $c_d + 1 \leq (1+1/a)c_d$ and thus $m + 1 + c_d \leq (1 + 1/a)(m + c_d)$. For all real numbers x, $(1 + x) \leq e^x$ and thus $(1 + 1/a)^a \leq e$. Therefore,

$$\sum_{m=1}^{k} \frac{\big(m+1+c_d\big)^a}{\big(m+c_d\big)^2} \leq e \sum_{m=1}^{k} \big(m+c_d\big)^{a-2}. \tag{4.4.14}$$

For $p > 0$, the function $x \mapsto (x + c)^p$ is monotone increasing for $x, c > 0$, Hence,

$$\begin{aligned}
\sum_{\ell=1}^{k} (\ell+c)^p &\leq \sum_{\ell=1}^{k} \int_{\ell}^{\ell+1} (x+c)^p dx \\
&= \int_{1}^{k+1} (x+c)^p dx \\
&= \frac{(k+1+c)^{p+1}}{p+1} - \frac{(1+c)^{p+1}}{p+1} \\
&\leq \frac{(k+1+c)^{p+1}}{p+1}.
\end{aligned}$$

Since $a > 2$, we can apply this with $p = a - 2 > 0$ to find

$$e \sum_{m=1}^{k} \big(m+c_d\big)^{a-2} \leq e \frac{(k+1+c_d)^{a-1}}{a-1}.$$

Combining (4.4.9), (4.4.12), (4.4.13) and (4.4.14) finally gives

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_\star\|_2^2] \leq & \left(\frac{1 + a\kappa^2(\Sigma)(d+2)^2}{k + 1 + a\kappa^2(\Sigma)(d+2)^2} \right)^a \mathbb{E}[\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_\star\|_2^2] \\ & + \frac{ea^2\kappa(\Sigma)(d+2)^2}{\lambda_{\min}(\Sigma)\big(a-1\big)\big(k+1+a\kappa^2(\Sigma)(d+2)^2\big)}.\end{aligned}$$

Using that $0 < a/(a-1) < 2$ for $a > 2$, now yields the result. □