Technische Universität München
TUM School of Computation, Information and Technology

Finite sample identification of artificial neural networks

Michael Rauchensteiner

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz: Prof. Dr. Daniel Matthes

Prüfer*innen der Dissertation:

1. Prof. Dr. Massimo Fornasier
2. Prof. Dr. Jan Vybíral
3. Prof. Dr. Radu V. Balan

Die Dissertation wurde am 24.04.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 17.11.2023 angenommen.

Abstract

This dissertation considers the problem of neural network identifiability (i.e., the retrieval of network weights and shifts) by constructive and efficient methods. We establish how the linear span of network derivatives with respect to the network input encodes the weight information of shallow neural networks and suitable deep neural networks. More precisely, for suitable inputs, the corresponding Hessian matrices give rise to a matrix space that admits a basis consisting of rank-one matrices given by outer products of the original first-layer weights and suitable linear combinations of weights associated with deeper layers.

We discuss the characterization of rank-one matrices within symmetric matrix spaces as local maximizers of suitable non-linear programs in the overcomplete regime, where the number of spanning rank-one matrices exceeds the ambient dimension of the matrices. Additionally, it is shown how these local maximizers can be computed efficiently and robustly by simple iterative methods similar to projected gradient ascent algorithms.

We leverage this technique to reconstruct the weights of wide shallow neural networks with sufficiently non-polynomial activations, where the number of neurons scales (up to log-factors) quadratic with the input dimension of the network. We present how the recovery of the shifts of wide shallow networks with weights known up to sign can be tackled by empirical risk minimization via gradient descent paired with an initialization strategy. For the shift recovery, we provide a local convergence analysis that borrows techniques from the neural tangent kernel community. Combining the steps above, we attain an end-to-end recovery pipeline of wide shallow neural networks that provably recovers all underlying network parameters up to numerical accuracy.

Furthermore, we extend this parameter recovery to networks with multiple hidden non-linear layers of pyramidal shape. This approach is based on the so-called entangled weights, which are defined as the gradients of the network's pre-activations. Entangled weights serve as a generalization of ordinary weights, enabling us to decouple weight information accessible via network differentiation. We provide proof and empirical evidence that entangled weights can be recovered from network Hessians and show heuristically that suitable deep neural networks can be identified based on the entangled weights by learning all remaining unknown parameters by empirical risk minimization. As a complement, all theoretical analysis of the network identification is corroborated by extensive numerical experiments.

Zusammenfassung

Diese Dissertation befasst sich mit der effizienten und konstruktiven Parameteridentifizierung von künstlichen neuronalen Netzwerken. Genauer, beschäftigen wir uns mit der Frage, unter welchen Umständen sich die Parameter eines neuronalen Netzwerkes ausgehend von Netzwerkauswertungen rekonstruieren lassen. Wir zeigen - für unterschiedliche Netzwerkarchitekturen - dass die lineare Hülle von Hesse-Matrizen eines neuronalen Netzes dessen Gewichte kodiert. Hierbei lassen sich die Netzwerkgewichte aus einer Basis der linearen Hülle ableiten. Diese Matrixbasis lässt sich dadurch charakterisieren, dass alle ihre Elemente nahe an einer Rang-1 Matrix liegen.

Wir beschreiben, wie eine ausreichend inkohärente Matrixbasis, bestehend aus Rang-1 Matrizen, als Menge der lokalen Maxima eines nichtlinearen Programms über einem symmetrischen Matrixraum identifiziert werden kann. Weiterhin werden effiziente Algorithmen zur Bestimmung dieser lokalen Maxima anhand projizierender Gradientenverfahren behandelt.

Wir zeigen, wie sich mit Hilfe dieser Herangehensweise die Parameter von zweischichtigen neuronalen Netzwerken mit hinreichend nicht-polynomiellen Aktivierungsfunktionen bestimmen lassen. Mittels einer lokalen Konvergenzanalyse wird gezeigt, wie sich alle übrigen unbekannten Netzwerkparameter mittels empirischer Risikominimierung approximieren lassen. Der Fokus der theoretischen Analyse liegt hierbei auf Netzwerken, bei denen die Anzahl der nichtlinearen Neuronen quadratisch mit der Eingabedimension skaliert (abgesehen von logarithmischen Faktoren). Insgesamt wird somit ein Verfahren präsentiert welches alle Parameter eines weiten zweischitigen neuronalen Netzwerkes bis auf numerische Approximationsfehler anhand von Netzwerkauswertungen bestimmt.

Anschließend wird gezeigt wie sich dieser Ansatz auf mehrschichtige neuronale Netzwerke mit einer pyramidischen Architektur anwenden lässt. Hierzu wird eine Generalisierung von herkömmlichen Netzwerkgewichten eingeführt, welche sich aus den Gradienten der Voraktivierungen ableiten lassen. Es wird bewiesen wie sich diese Gewichte mittels eines ähnlichen Verfahrens aus Netzwerk-Hesse-Matrizen bestimmen lassen. Weiterhin zeigen wir anhand heuristischer Methoden, wie sich die übrigen Netzwerkparameter in einem zweiten Schritt isoliert bestimmen lassen. Zusätzlich zu den theoretischen Ergebnissen demonstrieren wir die Auswertungen numerischer Studien, welche die erfolgreiche Parameteridentifikation von zwei- und mehrschichtigen neuronalen Netzwerken zeigen.

Acknowledgements

This dissertation and the positive experience during my doctoral studies were made possible by all the great people and collaborators I met along the way, to whom I would like to express my gratitude: First and foremost, I want to thank my supervisor Massimo Fornasier for being a great teacher and for his guidance throughout this journey. His encouragement and trust not only shaped my academic path but also taught me many valuable lessons about myself. I want to thank all my collaborators, in particular Timo Klock for the great collaboration on many projects and the fruitful discussions over the last few years, which deeply impacted my work, and Marco Mondelli, from whom I gained many insights into the connection of my studies to neural tangent kernel methods.

I am grateful to all my friends and colleagues at M15 for their insight and help on mathematical problems, for the fun we had during coffee breaks, board game nights, barbeques, and beyond.

My deepest gratitude goes to my family and friends for their tremendous support. I want to thank my mother, Hanne, for always having my back throughout all these years and my father, Hans, who is a great inspiration and friend whom I miss dearly. Furthermore, I want to thank my uncle Jakob for his great support. Last but not least, I want to thank my partner Andreea for being there every step of the way, for kindly pushing me when needed, and for always bringing joy into my life.

Contents

Introduction

Over the last two decades, we have witnessed incredible breakthroughs in machine learning and modern deep learning algorithms have crept into many domains of our daily lives. For instance, an average off-the-shelf phone already comes with several pre-installed machine learning algorithms to support face recognition, voice interpretation, and various other applications. At the core of this development stand deep learning architectures such as artificial deep neural networks [81], which can learn representations of complex data by the composition of multiple simple non-linearities and have become state-of-the-art in various data-intensive fields such as computer vision [78], natural language problems [7], playing Go [1], or genomics [47]. The continuous rise of deep learning is fueled by the increasing amount of computational resources and data that have become available over the recent years [81]. Another reason for its success is the large amount of scientific and engineering work invested in optimizing learning algorithms and developing new neural network architectures. The theoretical research on neural networks in their current form dates back to the late 80s and the early 90s, with famous early results such as the *universal approximation theorem* [67, 34, 57, 82] or *back-propagation* [109].

Despite the prevalence and great success of deep learning models in a wide range of data-driven applications, many underlying theoretical phenomena are still not fully understood. In supervised learning, deep neural networks are typically trained by empirical risk minimization via stochastic gradient descent. It is common practice that the number of trainable network parameters far exceeds the number of training data points, which is also referred to as *over-parameterization*. One remarkable observation is that over-parameterization empirically aids both the optimization of the empirical loss and the generalization of the learned network (i.e., how well the network performs on unseen data) [136]. Traditional statistical learning theory fails to explain why these networks manage to generalize well on unseen data. One approach to better understand the generalization capabilities of neural networks is to assume the underlying training data is realizable by a neural network. In this so-called *teacher-student* framework, we can measure how well a trained neural network generalizes by directly comparing the trained student network to the teacher network that gave rise to the training data. Perfect generalization is guaranteed whenever a training algorithm learns the exact parameters of the teacher network. This promotes the fundamental question of whether there exist efficient algorithms that can infer the teacher network parameters from realizations. A series of works in the early 90s show that under fairly mild conditions, the parameters of a network remain fully determined up to natural symmetries by the input-output map of a neural network [121, 5, 48]. However, these works on the *identifiability* of neural networks do not provide constructive methods for parameter retrieval and require that the input-output map of the neural network is known on its full domain. A question remains open: How much information is required to reconstruct the network parameters?

In this dissertation, we present the results from the joint works [52, 50, 51], which provide efficient algorithms that provably recover the parameters of suitable neural networks from a *finite amount* of input-output samples. Moreover, our methods are tractable, i.e., we only require a number of network evaluations that depend polynomially on the complexity of the network

(here, complexity refers to the number of neurons and input dimension of the network), and we demonstrate successful recovery by empirical experiments.

Structure of the dissertation

The main part of this dissertation is divided into four chapters. Ideally, these chapters are read in chronological order since the individual chapters are not entirely self-contained and will regularly use concepts and results from preceding chapters.

- In **Chapter 1**, we introduce artificial feed-forward neural networks and motivate the problem of neural network identification. Artificial neural networks are functions composed of several non-linearities and can be expressed as functions depending on a finite set of parameters, typically encoded as network weights and shifts. The neural network identifiability problem seeks answers to the question under which circumstances these parameters are uniquely defined (up to natural symmetries) by the input-output map of the network. The recovery of network parameters can be linked to other active research questions which study the generalization capabilities of neural networks (e.g., by analyzing the behavior of supervised learning algorithms in a teacher-student model). We will discuss related literature in this domain. We corroborate the discussion with numerical experiments to better understand under which constraints parameter identification of neural networks can be attained by classical approaches, such as empirical risk minimization via stochastic gradient descent, and introduce an alternative approach that identifies the parameters by using network evaluations to approximate higher-order derivatives of the network.

- In **Chapter 2**, we study the recovery of a rank-one basis of a symmetric matrix space known only up to perturbations. This problem arises, for instance, during the recovery of weights from network Hessians or approximations thereof. This chapter provides characterizations of near-rank-one matrices within a symmetric matrix space in the form of non-linear programs. We show that, provided sufficient incoherence of the basis matrices, the near-rank-one basis elements can be identified as the second-order critical points of these non-linear programs. Furthermore, we show how the computation of these second-order critical points can be tackled by suitable algorithms inspired by projected gradient ascent methods.

- In **Chapter 3**, we provide an end-to-end theoretical analysis of the identification of wide shallow neural networks with bounded shifts and suitable smooth activation functions (such as sigmoidal functions) that is based on the work [51]. We show that, under sufficient incoherence of the network weights, we can fully identify all network parameters up to numerical errors originating from the approximation of network derivatives. Our results hold with high probability up to a regime where the number of network neurons m scales quadratic with the input dimension D up to logarithmic factors, i.e., $\mathcal{O}(m \log^2 m) = D^2$. The main novelty of the presented pipeline is that it comes with guarantees for *wide* networks and can recover the network *shifts* while only relying on $\mathcal{O}(Dm^2 \log^2 m)$ evaluations of the teacher network.

- In **Chapter 4**, we demonstrate how the recovery strategy presented in Chapter 3 can be extended from shallow networks to deep neural networks with a pyramidal architecture and cover the results from [52, 50]. To enable this extension, we introduce *entangled weights*, which are defined as the gradients of the pre-activations of individual neurons of a network. As in Chapter 3, we prove that entangled weights are computable as the near-rank-one matrix basis within a subspace of the span of network Hessian approximations.

We show heuristically how access to entangled weight matrices can promote the full identification of all network parameters for pyramidal networks with up to three layers and corroborate our results with extensive numerical experiments.

The theoretical results presented in Chapter 2 - 4 are based on the joint work published in [52, 50, 51]. The related works and collaborators are mentioned explicitly at the beginning of each chapter.

Chapter 1

Recovering Neural Networks

The problem of neural network identifiability can be shortly described as follows: Consider a neural network as a black-box with unknown parameters but assume knowledge of its input-output map. Is it possible to infer the network's internal structure and parameters (weights and biases) based on this information? Existing results dating back to the early 90s [48] prove that this is, in fact, true for generic neural networks with sigmoidal activations, i.e., the network parameters are uniquely determined by the input-output map up of the network to natural symmetries. However, assuming a realizable setting where such an identification is possible, one question that remains unanswered by these early results is how much input-output information is required to constructively recover the network parameters. This question can be answered by studying the sample complexity of constructive reconstruction algorithms. In this chapter, we provide a formal introduction to relevant results and terminology from the theory of neural networks and lay the groundwork for the remaining chapters. We review existing approaches to the network reconstruction problem. More specifically, we will connect this problem to a line of results that studies the generalization capabilities of neural networks learned by classical methods in a so-called *teacher-student* framework. Finally, we corroborate the review of existing literature with numerical experiments exploring the limitations of classical methods in terms of parameter identification when different network architectures are considered.

1.1 Preliminaries

The following section introduces the necessary notation, provides a short introduction to neural networks, and a short discussion that motivates the problem of neural network identification.

1.1.1 Notation and singular subspaces

Functions and derivatives. Given any $n \in \mathbb{N}$, let $[n]$ denote the index set $[n] = \{1, \dots, n\}$. We denote by $\mathcal{C}^n(\mathbb{R})$ the space of functions in $\mathbb{R}$ with n continuous derivatives. Given a scalar function $g \in \mathcal{C}^n(R)$ and $k \in [n]$, we denote by $g^{(k)}$ the k-th derivative of g. To simplify expressions involving neural networks, we follow the convention that vector-valued inputs $x \in \mathbb{R}^D$ applied to scalar functions are evaluated component-wise, i.e., $g(x) = [g(x_1), g(x_2), \dots, g(x_D)]^\top$. For multivariate functions $f : \mathbb{R}^D \to \mathbb{R}$, the n-th derivative w.r.t. to the input $x \in \mathbb{R}^D$ is denoted by $\nabla^n f(x)$ and defines a tensor, i.e., $\nabla^n f(x) \in (\mathbb{R}^D)^{\otimes n}$ (see below), with entries given by

$$\left(\nabla^n f(x) \in (\mathbb{R}^D)^{\otimes n}\right)_{i_1,\dots,i_n} = \frac{\partial f(x)}{\partial x_{i_1} \partial x_{i_2} \dots \partial x_{i_n}}, \quad \text{for } i_1, \dots, i_n \in [D].$$

To compare the limiting behavior of functions and the complexity of algorithms, we write $f(x) = \mathcal{O}(g(x))$ if there exists a constant $C > 0$ and $x_0 \in R$, such that $|f(x)| \leq Cg(x)$ for all $x \geq 0$. Equivalently, we say $f(x) \lesssim g(x)$ whenever f remains bounded by g up to a constant. Similarly, we say $f(x) = o(g(x))$ whenever $\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0$.

Probability distributions. We denote by $\mathrm{Unif}(\mathbb{S}^{D-1})$ the uniform distribution on the unit sphere and by $\mathcal{N}(\mu, \Sigma)$ the multivariate Gaussian distribution with mean $\mu \in \mathbb{R}^D$ and covariance $\Sigma \in \mathbb{R}^{D\times D}$. Consider a random variable X. We say X is sub-Gaussian (cf. [127, Section 2.5.1]) if there exists a constant $C > 0$ s.t.

$$\mathbb{P}(|X| > t) \leq 2\exp(-t^2/C) \quad \text{for all } t \geq 0,$$

and denote by $\|X\|_{\psi_2}$ the so-called sub-Gaussian norm (cf. [127, Definition 2.5.6]) given by

$$\|X\|_{\psi_2} = \inf\left\{t > 0 : \mathbb{E}\exp(X^2/t^2) \geq 2\right\}.$$

A random vector X in $\mathbb{R}^D$ is called sub-Gaussian if all its marginals are sub-Gaussian and

$$\|X\|_{\psi_2} := \sup_{x\in\mathbb{S}^{D-1}} \|\langle x, X\rangle\|_{\psi_2}.$$

A random vector X in $\mathbb{R}^D$ is called *isotropic* if $\mathbb{E}[XX^\top] = \mathrm{Id}_D$, where Id_D denotes the identity matrix in $\mathbb{R}^{D\times D}$.

Vectors, matrices, and tensors. In the following, consider positive integers $D_1, \ldots, D_n \in \mathbb{N}$. Given two vectors $x \in \mathbb{R}^{D_1}$ and $y \in \mathbb{R}^{D_2}$, then $x \otimes y \in \mathbb{R}^{D_1\times D_2}$ denotes their Kronecker product (also referred to as outer product) defined as $xy^\top$ and $\langle u, v\rangle$ denotes the inner product (assuming $D_1 = D_2$). We use the conventional vector norms

$$\|x\|_p := \left(\sum_{i=1}^{D_1} x_i^p\right)^{1/p}, \quad \text{for } p \in [1, \infty),$$

and $\|x\|_\infty := \max_{i\in[D_1]} |x_i|$. We denote the unit sphere in $\mathbb{R}^D$ by $\mathbb{S}^{D-1} := \left\{v \in \mathbb{R}^D \middle| \|v\|_2 = 1\right\}$. Tensors are generalizations of vectors and matrices that can be expressed as multi-arrays with multiple indices. Tensors of order $n \in \mathbb{N}$ (also called n-tensor) are regarded as elements of $\bigotimes_{k=1}^n \mathbb{R}^{D_k} = \mathbb{R}^{D_1\times D_2\times\cdots\times D_n}$. Hence, vectors and matrices can be seen as first- and second-order tensors. Given two tensors $T_1, T_2 \in \mathbb{R}^{D_1\times D_2\times\cdots\times D_n}$ of order $n \geq 2$, let $\langle T_1, T_2\rangle$ denote its Frobenius inner product given by

$$\langle T_1, T_2\rangle = \sum_{i_1=1}^{D_1}\sum_{i_2=1}^{D_2}\cdots\sum_{i_n=1}^{D_n} (T_1)_{i_1,i_2,\ldots,i_n}(T_2)_{i_1,i_2,\ldots,i_n},$$

and denote by $\|T_1\|_F$ the Frobenius norm $\|T_1\|_F := \langle T_1, T_1\rangle^{1/2}$. Additionally, we denote by $T_1 \odot T_2$ the Hadamard product defined as the element-wise product

$$(T_1 \odot T_2)_{i_1,\ldots,i_n} := (T_1)_{i_1,\ldots,i_n} \cdot (T_2)_{i_1,\ldots,i_n} \quad \text{for } k \in [m], i_k \in [D_k].$$

Let $\mathbb{S}^{D_1\times\cdots\times D_n-1}$ denote the Frobenius unit sphere, which is defined as the matrix/tensor equivalent of $\mathbb{S}^D$:

$$\mathbb{S}^{D_1\times\cdots\times D_n-1} := \left\{T \in \mathbb{R}^{D_1\times\cdots\times D_n} \middle| \|T\|_F = 1\right\}.$$

Whenever the underlying dimensions of the sphere are clear from context, we might also abbreviate the notation by simply denoting the respective unit sphere as $\mathbb{S}$. Given a matrix $T \in \mathbb{R}^{D_1 \times D_2}$, let $\|X\|$ denote its spectral norm, which is defined as

$$\|X\| = \sup_{u \in \mathbb{R}^{D_2}} \frac{\|Xu\|_2}{\|u\|_2}.$$

We say a matrix or tensor $T \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_n}$ is rank-one if it can be expressed by outer products of suitable vectors,

$$T = v_1 \otimes v_2 \otimes \cdots \otimes v_n \quad \text{for } v_k \in \mathbb{R}^{D_k}, k \in [n].$$

Let $u \in \mathbb{R}^D$, then the n-th fold outer product w.r.t. u will be denoted as $u^{\otimes n}$, and similarly $(\mathbb{R}^D)^{\otimes n} := \bigotimes^n \mathbb{R}^D$.

Vectorization and linear subspaces. Given a tensor $T \in \mathbb{R}^{D_1 \times D_2 \times \cdots \times D_n}$ we denote by $\mathrm{vec}(T) \in \mathbb{R}^{D_1 D_2 \ldots D_n}$ its vectorization, which equates to the vector that stores all elements of T sorted w.r.t. its indices in increasing numerical order starting from the rightmost axis. For instance, let $X \in \mathbb{R}^{D_1 \times D_2}$ be a matrix with columns $x_1, \ldots, x_{D_2}$, then the vectorization of X is defined as the vector

$$\mathrm{vec}\left(\begin{bmatrix} \vdots & \ldots & \vdots \\ x_1 & \ldots & x_{D_2} \\ \vdots & \ldots & \vdots \end{bmatrix}\right) = \begin{pmatrix} x_1 \\ \vdots \\ x_{D_2} \end{pmatrix} \in \mathbb{R}^{D_1 D_2}.$$

For fixed finite dimensions, the $\mathrm{vec}(\cdot)$-operator acts as an isomorphism between linear tensor spaces and ordinary vector spaces. Given a tensor space $\mathcal{W} \subset \mathbb{R}^{D_1 \times \cdots \times D_n}$, we refer to the vectorization of $\mathcal{W}$ as the space $\{\mathrm{vec}(T) | T \in \mathcal{W}\} \subset \mathbb{R}^{D_1 D_2 \ldots D_n}$. By identifying tensors with vectors in $\mathbb{R}^D$, we can apply a matrix factorization such as the singular value decomposition to ensembles of vectorized tensors, providing us with a simple dimensionality reduction technique that applies to tensor spaces.

Definition 1.1 (Singular subspace)**.** *Let $x_1, \ldots, x_N \in \mathbb{R}^D$ and denote $X = [x_1, \ldots, x_N] \in \mathbb{R}^{D \times N}$ with singular value decomposition $X = U \Sigma V^\top$ such that $U \in \mathbb{R}^{D \times D}, \Sigma \in \mathbb{R}^{D \times N}, V \in \mathbb{R}^{N \times N}$. For any $m \leq D$, denote by $u_1, \ldots, u_m \in \mathbb{R}^D$ the first m columns of U (i.e., the first m left singular vectors of X). We define the m-th singular subspace associated with the vectors $x_1, \ldots, x_N$ as*

$$\mathrm{span}_m(\{x_1, \ldots, x_N\}) := \mathrm{span}(\{u_1, \ldots, u_m\}) \subset \mathbb{R}^D.$$

Furthermore, for any set of tensors $T_1, \ldots, T_N \in \mathbb{R}^{D_1 \times \cdots \times D_n}$ and $m \in \mathbb{N}$ such that the space $\mathrm{span}_m(\{\mathrm{vec}(T_1), \ldots, \mathrm{vec}(T_N)\})$ is well-defined, we denote by

$$\mathrm{span}_m(\{T_1, \ldots, T_N\}) \subset \mathbb{R}^{D_1 \times \cdots \times D_n}$$

the linear tensor space such that its vectorization is given by $\mathrm{span}_m(\{\mathrm{vec}(T_1), \ldots, \mathrm{vec}(T_N)\})$.

Singular subspaces, as defined above, can be seen as a technique of dimensionality reduction. From this point of view, the construction above is identical to what is commonly known as (non-centered) principal component analysis (PCA, see also [103, 68]). The extension of singular subspaces to tensors follows by simply regarding tensors as elements of a linear vector space and using the fact that the vectorization operator is invertible for fixed dimensions. Notably, a matrix's singular vectors are in general not uniquely determined. Take for instance the identity matrix Id_D, then $\mathrm{Id}_D = U \mathrm{Id}_D U^T$ for any unitary matrix $U \in \mathbb{R}^{D \times D}$. However, the m-th singular subspace associated with a set of vectors is uniquely determined whenever the first m singular values of X are separated from the remaining ones by a spectral gap:

Lemma 1.1. *Let $x_1, \dots, x_N \in \mathbb{R}^D$ and denote $X = [x_1, \dots, x_m] \in \mathbb{R}^{D \times N}$ with singular values given by $\sigma_1(X) \geq \sigma_2(X) \cdots \geq \sigma_{\min\{D,N\}}$. Consider $m \leq \operatorname{rank}(X)$ such that either $m = \operatorname{rank}(X)$ or $\sigma_m(X) > \sigma_{m+1}(X) > 0$, then the m-th singular subspace $\operatorname{span}_m(\{x_1, \dots, x_N\})$ in Definition 1.1 is unique.*

Proof. If $m = \operatorname{rank}(X)$, then the singular subspace is well-defined due to $\operatorname{span}_m(\{x_1, \dots, x_N\}) = \operatorname{span}(\{x_1, \dots, x_N\})$. Assume now that $\sigma_m(X) > \sigma_{m+1}(X) > 0$, then, by the Eckart-Young-Mirsky theorem (cf. [44]), the closest rank-m approximation X_m of X is uniquely determined and the image of X_m is given by the span of its first left singular vectors, i.e., $\operatorname{range}(X_m) = \operatorname{span}(\{u_1, \dots, u_m\})$. Since X_m (and its image) is uniquely determined, the same must be true for the space spanned by the left singular vectors. □

It should be noted that m is typically chosen such that the statement in Lemma 1.1 holds whenever singular subspaces are used throughout the upcoming chapters. A common application of singular subspaces in this work is the following: Consider a set of n-tensors $T_1, \dots, T_N \in \mathbb{R}^{D_1 \times \cdots \times D_n}$ for positive integers $n, D_1, \dots, D_n \in \mathbb{N}$. We seek to approximate the linear span

$$\mathcal{W} := \operatorname{span}(\{T_1, \dots, T_N\})$$

under the assumption that we are only given perturbed measurements $\hat{T}_1, \dots, \hat{T}_N \in \mathbb{R}^{D_1 \times \cdots \times D_n}$, where $\hat{T}_i \approx T_i$. Let $m = \dim(\mathcal{W})$, then an m-th singular subspace $\widehat{\mathcal{W}} := \operatorname{span}_m(\{\hat{T}_1, \dots, \hat{T}_N\})$ of the perturbed tensors is a suitable candidate for the approximation of $\mathcal{W}$. Since singular subspaces are defined via singular value decomposition, the resulting distance between $\mathcal{W}, \widehat{\mathcal{W}}$ can then be measured by classical perturbation bounds associated with the singular value decomposition such as Wedin's bound [130]. Throughout the following chapters, the distance between subspaces will typically be measured by

$$\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F \to F} := \sup_{T \in \mathbb{R}^{D_1 \times \cdots \times D_n}} \frac{\|P_{\mathcal{W}}(T) - P_{\widehat{\mathcal{W}}}(T)\|_F}{\|T\|_F}, \tag{1.1}$$

where $P_{\mathcal{W}}, P_{\widehat{\mathcal{W}}}$ denotes the orthogonal projection onto $\mathcal{W}, \widehat{\mathcal{W}}$, respectively. The following proposition provides a bound on $\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F \to F}$ based on Wedin's bound that bound in terms of the individual tensor perturbations and the singular values associated with the data matrix of the vectorized tensors $\operatorname{vec}(\hat{T}_i)$.

Proposition 1.1. *Consider n-tensors $T_1, \dots, T_N, \hat{T}_1, \dots, \hat{T}_N \in \mathbb{R}^{D_1 \times \cdots \times D_n}$ and denote by $\mathcal{W} = \operatorname{span}\{T_1, \dots, T_N\}$ the linear span with dimension $m = \dim(\mathcal{W})$. Let the matrices $M, \widehat{M} \in \mathbb{R}^{D_1 D_2 \dots D_n \times N}$ be given by*

$$M = [\operatorname{vec}(T_1) | \dots | \operatorname{vec}(T_N)], \quad \widehat{M} = [\operatorname{vec}(\hat{T}_1) | \dots | \operatorname{vec}(\hat{T}_N)]. \tag{1.2}$$

If $\sigma_m(\hat{M}) > 0$, then the distance between $\widehat{\mathcal{W}} := \operatorname{span}_m(\{\hat{T}_1, \dots, \hat{T}_N\})$ and $\mathcal{W}$ satisfies

$$\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F \to F} \leq \frac{2\|M - \widehat{M}\|_F}{\sigma_m(\widehat{M})}.$$

Remark: In the special case of 1-tensors (i.e., vectors), we can simply redefine $\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F \to F}$ by replacing the Frobenius norm with the Euclidean norm $\|\cdot\|_2$.

Proof. Let $U\Sigma V^T = M, \hat{U}\hat{\Sigma}\hat{V}^T = \hat{M}$ be the singular value decompositions of $M, \hat{M}$, and denote by $U_m, \hat{U}_m$ the matrix build from the first m columns of $U, \hat{U}$, respectively. Notably, the columns of $U_m, \hat{U}_m$ span the vectorized equivalents of the tensor spaces $\mathcal{W}, \widehat{\mathcal{W}}$, respectively. For $\widehat{\mathcal{W}}$ this

follows by the construction of singular subspaces in Definition 1.1, whereas for $\mathcal{W}$ this follows from $m = \dim(\mathcal{W}) = \dim(\operatorname{range} U_m)$ and the fact that the span of singular vectors must be a subspace of $\operatorname{span}\{\operatorname{vec}(T_1), \dots, \operatorname{vec}(T_N)\}$. Additionally, singular vectors are orthonormal and therefore $U_m U_m^\top, \hat{U}_m \hat{U}_m^\top$ are orthogonal projection matrices onto the vectorizations of $\mathcal{W}, \widehat{\mathcal{W}}$. This now implies

$$\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F} = \|U_m U_m^\top - \hat{U}_m \hat{U}_m^\top\| \leq \|U_m U_m^\top - \hat{U}_m \hat{U}_m^\top\|_F,$$

which relies on the fact that $\|T\|_F = \|\operatorname{vec}(T)\|_2$, which justifies the exchange of the distance measure in (1.1) with the spectral norm of the difference between the orthogonal projection matrices $U_m U_m^\top - \hat{U}_m \hat{U}_m^\top$. Now

$$\|U_m U_m^\top - \hat{U}_m \hat{U}_m^\top\|_F \leq \frac{2\|M - \widehat{M}\|_F}{\sigma_m(\widehat{M})}$$

can be derived from classical results that study the effect of perturbations on the singular vectors such as Wedins bound ([130] and [120, Theorem 4]) or eigenvectors such as the Davis-Kahan theorem ([36] and [20, Theorem 7.3.1]). □

1.1.2 Introduction to neural networks

As neural network models became more popular in solving data-driven problems in many fields, many different network architectures were invented, which were designed with specific problem settings in mind. Consequently, "neural network" refers to various parametric models that share certain design principles. For this work, we focus exclusively on what is known as fully connected feed-forward artificial neural networks (FFNN). Whenever we refer to neural networks without specifying a particular architecture, we refer to the kind of models introduced within this section for simplicity. Neural networks can be broken down into smaller computational units, the so-called *neurons*. Neurons are functions represented by an affine transformation composed of a univariate function that is typically non-linear and referred to as its *activation* (function). More precisely, a neuron with activation $g : \mathbb{R} \to \mathbb{R}$ is a function parameterized by a *weight* vector $w \in \mathbb{R}^D$ and a scalar *shift* $\tau \in \mathbb{R}$ which computes its output according to

$$x \mapsto g(w^\top x + \tau).$$

Classical feed-forward neural networks are then constructed by stacking and composing individual neurons. In fact, one single neuron already constitutes a neural network. Slightly more complex neural networks can be constructed by considering finite linear combinations of different neurons giving rise to the function class

$$\mathcal{SN}(g, \mathbb{R}^D) := \operatorname{span}\{g(w^\top x + \tau) \mid w \in \mathbb{R}^D, \tau \in \mathbb{R}\}. \tag{1.3}$$

We refer to members of $\mathcal{SN}(g, \mathbb{R}^D)$ as *shallow neural networks* with activation g.

Expressivity of neural networks. A natural question one may ask is what type of functions can be represented or sufficiently well approximated by these shallow neural networks. The richness of $\mathcal{SN}(g, \mathbb{R}^D)$ is strongly influenced by the choice of activation function g: If we were to choose an affine function for g, for instance, the activation $g(x) = x$, then any network in $\mathcal{SN}(g, \mathbb{R}^D)$ simply collapses into a linear function. The expressivity of $\mathcal{SN}(g, \mathbb{R}^D)$ was addressed by a series of fundamental publications studying neural networks in the setting of approximation theory in the late 80s and early 90s [67, 34, 57, 82]. The synthesis of these results

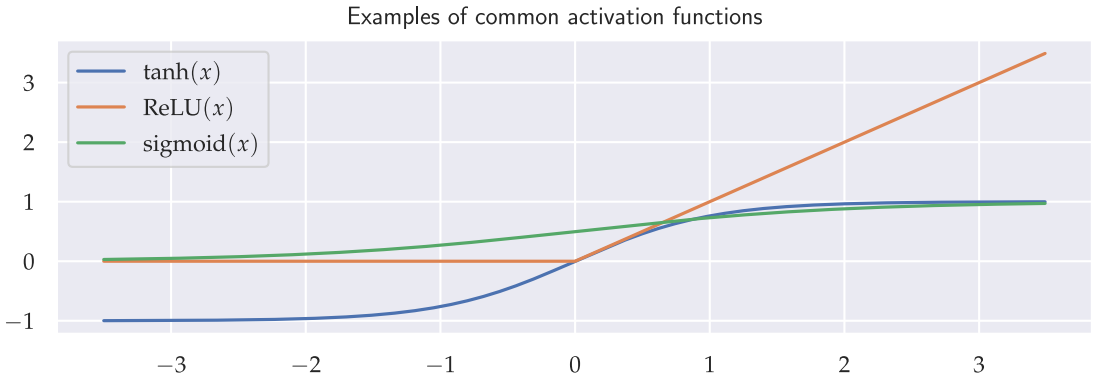

Figure 1.1: Examples of common activation functions.

is nowadays often referred to as the *universal approximation theorem*. These works establish sufficient conditions under which any continuous function on a bounded domain can be approximated by a (sufficiently) wide shallow neural network in the sense that $\mathcal{SN}(g, \mathbb{R}^D)$ is dense in the space of continuous functions (on a bounded domain). Here, the width of a shallow neural network refers to the number of neurons. As shown in [82], if we assume that g is continuous, then shallow neural networks are universal approximators (i.e., they can approximate any continuous function on a bounded domain) if and only if g is not a polynomial. Two aspects make this statement particularly interesting: First, the requirements on the activation g are relatively mild. Second, it is clearly necessary that g is not polynomial since otherwise any element within $\mathcal{SN}(g, \mathbb{R}^D)$ would be a multivariate polynomial with a degree less or equal to the degree of g, which means $\mathcal{SN}(g, \mathbb{R}^D)$ cannot be dense in $\mathcal{C}(\mathbb{R}^D)$. For more details on these results, we refer the interested reader to the survey [106]. To clarify, let us give some examples of common activation functions. One class of activation functions widely used in the early days of neural networks and highly relevant to our work is the class of *sigmoidal functions*. This class includes the hyperbolic tangent $g(t) = \tanh(t)$ or the logistic function $g(t) = (1 + \exp(-t))^{-1}$. Sigmoidal functions are typically smooth monotone functions with bounded horizontal asymptotes characterized by their "S"-shaped curve. Other examples of activation functions are piecewise linear functions, for instance, the rectifier linear unit (ReLU cf. [96, 88]) given by $g(t) = \max\{t, 0\}$. There is evidence (cf. [60]) that the ReLU-activation is better suited for deep neural network architectures (see below), which makes piecewise linear activations the preferred choice amongst practitioners.

Deep neural networks. Deep feed-forward neural networks extend the class of shallow neural networks by composing several layers of neurons into a single network architecture. In part, this extension is motivated by the fact that there exist functions $f : \mathbb{R}^D \to \mathbb{R}$ which require exponentially many neurons (w.r.t. D) to be sufficiently well approximated by shallow neural networks [46, 106, 107]. Examples are given in [92, 107], where it is shown that there exist certain compositional functions (i.e., functions that a recursive function composition can represent) that require exponentially fewer neurons when approximated by deep neural networks. Recent works [18, 32, 35, 38, 45, 91, 104, 112], add further support that suggests that deep network architectures help to avoid the curse of dimensionality when approximating high-dimensional functions. Let us now define deep feed-forward neural networks. First, note that any shallow network $f \in \mathcal{SN}(g, \mathbb{R}^D)$ with m neurons can be written as

$$f(x) = \sum_{k=1}^{m} v_k g(w_k^\top x + \tau_k), \quad \text{where } v_k \in \mathbb{R} \text{ for all } k \in [m]. \tag{1.4}$$

By adopting the convention that scalar functions applied to vectors are computed componentwise, f is expressed more compactly as

$$f(x) = v^\top g(W^\top x + \tau),$$

with $W \in \mathbb{R}^{D\times m}$ and $v, \tau \in \mathbb{R}^m$. This outlines a straightforward way how to extend the function class in (1.4), which only contains shallow networks with a single output, to vector-valued shallow networks: A shallow feed-forward neural network with input-size D, activation function g on m_1 neurons with m_2 outputs can be written in the form

$$f(x) = V^\top g(W^\top x + \tau), \tag{1.5}$$

with $W \in \mathbb{R}^{D\times m_1}$, $\tau \in \mathbb{R}^{m_1}$, and $V \in \mathbb{R}^{m_1\times m_2}$. Hence, networks as in (1.5) are multivariate functions $\mathbb{R}^D \to \mathbb{R}^{m_2}$ uniquely determined by a set of weight matrices W, V, a shift vector τ, and an activation function $g : \mathbb{R} \to \mathbb{R}$. Based on this construction, the class $\mathcal{SN}(g, \mathbb{R}^D)$ can easily be extended to networks with vector-valued outputs. Note that the universal approximation theorem referenced above does also apply to shallow neural networks with multiple outputs by the fact that any output component of such a network lies within $\mathcal{SN}(g, \mathbb{R}^D)$. The evaluation of a shallow network as in (1.5) can be broken down as follows: first, compute $y = g(W^\top x + \tau)$; second, set $f(x) = V^\top y$. These blocks are referred to as the layers of the network, which makes shallow neural networks two-layer neural networks. This can easily be generalized to the following class of parametric models:

Definition 1.2 (Fully connected feed-forward neural network)**.** *Let $L, m_L \in \mathbb{N}$ and consider scalar (activation) functions $(g)_{\ell\in[L]}$. A fully connected feed-forward neural network with L layers on $\mathbb{R}^D$ with m_L outputs is a map $f : \mathbb{R}^D \to \mathbb{R}^{m_L}$ parameterized by a set of weight matrices $W^{[1]}, \dots, W^{[L]}$ and shift vectors $\tau^{[1]}, \dots, \tau^{[L]}$, which given an input $x \in \mathbb{R}^D$ computes an output according to the iterative scheme*

$$\begin{aligned} y^{[0]}(x) &:= x, \\ y^{[\ell]}(x) &:= g_\ell\left((W^{[\ell]})^\top y^{[\ell-1]}(x) + \tau^{[\ell]}\right), \quad \text{for all } \ell \in [L], \\ f(x) &= y^{[L]}(x). \end{aligned}$$

Every network of this form with $L = 2$ is considered to be a shallow network, and networks with $L \geq 3$ are called deep neural networks. A layer can be understood as the parametric function $g_\ell((W^{[\ell]})^\top \cdot + \tau^{[\ell]})$. Notably, the number of neurons per layer, which is denoted by m_ℓ, is determined by the dimensions of the weight matrices and shifts vectors, i.e., we have

$$W^{[\ell]} = \begin{bmatrix} w_1^{[\ell]} & \dots & w_{m_\ell}^{[\ell]} \end{bmatrix} \in \mathbb{R}^{m_{\ell-1}\times m_\ell} \quad \text{and} \quad \tau^{[\ell]} \in \mathbb{R}^{m_\ell} \quad \text{for all } \ell \in [L], \tag{1.6}$$

where $m_0 := D$. Therefore, each individual neuron inside a network is uniquely determined by its corresponding layer $\ell \in [L]$ and its index within that layer $k_\ell \in [m_\ell]$. We distinguish between *hidden* layers/neurons ($\ell = 1, \dots, L-1$) and the *output* layer/neurons ($\ell = L$). Every neuron is equipped with an activation function, and we allow different activation functions $g_1, \dots, g_L$ for each individual layer. It should be noted that this is mainly done for the sake of generality and that typically all hidden layers share the same activation function $g_1 = g_2 = \cdots = g_{L-1}$. It is, however, common to consider different functions at the output neurons. A good example is the linear output function, which already appeared in the definition of shallow neural networks above.

Remark 1.1. *When we refer to the architecture of a network, we refer to the size (number of neurons) of each individual layer. Hence, the architecture is uniquely determined by $m_1, \dots, m_L$, or, equivalently, by the dimensions of the parameters $(W^{[\ell]})_{\ell\in[L]}, (\tau^{[\ell]})_{\ell\in[L]}$.*

1.1.3 Training neural networks

Let us now describe the training of a neural network in the context of *supervised learning*. In supervised learning problems, we are given labeled *training data* $(x_i, y_i)_{i\in[N]}$, $(x_i, y_i) \in \mathbb{R}^D \times \mathbb{R}^{m_L}$, which are regarded as samples of an unknown distribution $\mu_{X,Y}$. Supervised learning algorithms seek to infer a function (from the training data) that realizes the underlying phenomenon inherent to $\mu_{X,Y}$. The space of the learned functions is typically constrained by some structural assumptions. For instance, in deep learning, we are given a class of neural network functions that is characterized by activation functions and a fixed architecture (both of which are typically fixed a priori and determined heuristically). Since neural networks with fixed activation are entirely parameterized by their weight matrices and shift vectors, selecting a candidate network is equivalent to selecting those parameters. Most deep learning algorithms rely on empirical risk minimization to find a candidate network. More precisely, they seek solutions to the minimization problem

$$\operatorname{argmin}_{\theta\in\Theta} J(\theta) := \operatorname{argmin}_{\theta\in\Theta} \frac{1}{N} \sum_{i=1}^{N} \ell(f(x_i, \theta), y_i), \tag{1.7}$$

where Θ denotes the set of all weights and shifts and ℓ is the so-called *loss function*. A common example of loss functions is the quadratic loss $\ell(x, y) = C(x - y)^2$, in which case J in (1.7) is a least-squares objective. Clearly, the objective of any learning algorithm is to find a network that performs well on the entirety of $\mu_{X,Y}$, i.e., we are interested in finding network parameters that *generalize* to unseen data which is not included in the training examples. The generalization capabilities are captured in the *population risk*

$$\mathbb{E}_{(X,Y)\sim\mu_{X,Y}}[J(\theta)] = \mathbb{E}_{(X,Y)\sim\mu_{X,Y}}\left[\ell\left(f(X, \theta), Y\right)\right]. \tag{1.8}$$

Classical statistical learning theory studies how the parameter estimation based on the empirical risk relates to the minimizer of the population risk $\operatorname{argmin}_{\theta\in\Theta}\mathbb{E}_{(X,Y)\sim\mu_{X,Y}}[J(\theta)]$. A fundamental principle in statistical learning is the so-called *bias-variance tradeoff* [113], which describes the inherent conflict between the simultaneous minimization of the population risk and empirical risk. This principle can be described somewhat informally as follows: One way to aid the minimization in (1.7) is to widen the parameter search space Θ since increasingly complex models have more of free parameters to fit the training data. However, this increases the *variance* of the parameter estimation because the learning algorithm might *overfit* onto the training data. On the other hand, decreasing the complexity of the function space induced by Θ too much creates a *bias* such that the learning algorithm is unable to fully capture the underlying phenomena present in $\mu_{X,Y}$ (one also refers to this as *underfitting*).

A very successful approach to the minimization of the objective in (1.7) is the minimization via (stochastic[1]) gradient descent (and its variations). These gradient-based algorithms benefit from an efficient method to compute the gradients $\nabla_\theta J$, called *back-propagation* [109], which morally is a clever application of the chain-rule to the structure of neural networks. The success of gradient-based methods is somewhat surprising since, for neural networks, the objective (1.7) is generally highly non-convex and admits spurious and disconnected local minima [12, 110, 135]. The optimization landscape of gradient descent methods applied to (1.7) is dependent on the richness of the function space induced by Θ, i.e., the complexity of the network class under consideration. With increasingly complex network architectures, it becomes easier to fit the training data and thereby decrease $\operatorname{argmin}_{\theta\in\Theta} J(\theta)$.

[1]Different from ordinary gradient descent, which computes the gradients of $\nabla_\theta J$ on the entire training set, stochastic gradient descent only considers J on a subset of the training data, also called *(mini-)batch* that is randomly selected at each gradient step (cf. [23]).

However, the bias-variance tradeoff suggests that increasing the model complexity beyond a certain point will lead to worse performance on *unseen data*. Nevertheless, practitioners regularly train highly over-parameterized neural networks (i.e., more network parameters than training samples) to zero empirical loss using stochastic gradient descent while still generalizing well on unseen data. Several lines of work have studied the effects of over-parameterization. Many authors consider the evolution of the network parameters learned by gradient descent via the so-called *neural tangent kernel* (NTK). This line of work originates from [73], where it was shown that the network parameters evolve along the kernel gradient of the empirical loss (which is convex). Furthermore, it was shown that the NTK becomes deterministic in the limit case where the widths of the network layers approach infinity (cf. [73]). In this setting, the global convergence of gradient descent can therefore be related to the minimal eigenvalue of the NTK, which lead to several global convergence guarantees (w.r.t. the empirical loss) for shallow neural networks [43, 102, 117, 134, 94, 116] and deep neural networks [6, 40, 140, 141, 99, 98, 22]. The spectrum of the NTK has also been related to the generalization of networks trained by gradient descent [11, 94, 28].

The study [136] shows that state-of-the-art image classification networks (which fall into the over-parameterized regime) can still easily be trained to zero empirical loss when the training labels are randomly permuted or if the training images are replaced by random noise. In the same line of work, further results on memorization capacity show that mildly over-parameterized networks are capable of realizing generic data [22, 26, 70, 94, 106, 128]. As argued in [136], this shows that the deep neural networks considered in modern machine learning problems are complex enough to memorize the entire training data set. This creates a contradiction that classical statistical learning theory fails to explain: if the network class is complex enough to allow for perfect memorization, then the learning algorithm should suffer from overfitting. Notably, modern learning algorithms additionally make use of explicit regularization (e.g., data augmentation, weight decay, dropout cf. [136]). However, the results in [136] suggest that these methods can not fully explain the discrepancy of why highly over-parameterized models are able to generalize well on unseen data. Since the selection of the network architecture and explicit forms of regularization do not seem to explain the generalization capabilities of over-parameterized neural networks, a line of work has explored the implicit regularization mechanisms of training algorithms based on gradient descent methods [136, 9, 10, 13, 95, 97, 118, 133]. These results prove for simple models of neural networks (over-parameterized shallow neural networks and deep networks with linear activations) that gradient descent has an intrinsic bias to converge to networks with low intrinsic complexity. This poses an interesting question that leads us to the main topic of this thesis. If low-complexity neural networks exist that realize arbitrary data, under what circumstances can these networks be *uniquely identified* from a finite amount of samples?

1.2 Identifiability of fully connected Neural Networks

The universal approximation theorem and the preceding discussion suggest that sufficiently rich neural networks can realize complex phenomena. Assume now that a function is exactly realizable by a neural network. We can then ask the question of whether this network is uniquely determined. More precisely, do different network architectures and parameters exist that give rise to the same function? This question motivates the problem of *neural network identifiability*. To understand the problem of neural network identifiability, we first need to address a few inherent symmetries of neural networks: If two neural networks produce the same output for every possible input, we can consider them equal. Note that functional equivalence should not be mistaken for parametric equivalence. Two networks with different parameters can compute the same mapping, and it is not hard to find examples illustrating

this ambiguity between realization and neural network parameters. A trivial cause for this discrepancy is redundancy in the network. Practically speaking, redundancy can be introduced into any network by adding or concatenating the network with other subnetworks that compute the zero function or the identity. Other reasons are the inherent symmetries of the activation function. Take, for instance, the hyperbolic tangent as a representative of sigmoidal activation functions. This function is odd, i.e., $\tanh(x) = -\tanh(-x)$ for all $x \in \mathbb{R}$, and this leads to the identity

$$f(x) = \sum_{k=1}^{m} \tanh(w_k^\top x + \tau_k) = \sum_{k=1}^{m} -\tanh(-w_k^\top x - \tau_k) = \hat{f}(x).$$

The two hypothetical networks $f, \hat{f}$ are functionally equivalent, but their underlying parameters differ. Symmetries are not limited to sigmoidal functions. For instance, the ReLU function is invariant to certain rescaling operations since

$$\max\{a \cdot x, 0\} = a \cdot \max\{x, 0\} \quad \text{for all } a > 0,$$

which leads to other parameter transformations that do not affect the realization of ReLU networks. Lastly, there are many possible permutations of neurons in deep networks. Any notion of identifiability has to account for these symmetries and identification is generally only possible up to equivalence classes formed by the inherent symmetries of the activation and affine transformations.

1.2.1 Identifiability vs. identification of network parameters

We say a neural network is *identifiable* if the network realizations determine its parameters up to natural symmetries. A line of works starting in the early 90s' managed to answer the question of network identifiability positively for shallow networks that fulfill certain genericity conditions [121, 5]. In the same period, Fefferman found conditions under which deep neural networks with sigmoidal activation functions are fully identifiable [48]. Just recently, the results from [48] have been generalized to deep neural networks without clone nodes and piecewise $\mathcal{C}^1$ activations with bounded-variation derivative in [129]. The results presented above show that, aside from technicalities, the identification of the network parameters from its mapping is possible up to natural symmetries, and this problem of network identifiability is, in principle, quite well understood for certain activations like the sigmoidal tanh function. However, the characterization of uniqueness as it is analyzed above relies on knowing the input-output map of a network on its entire domain, and they are not constructive. If we were to introduce a budget on how many evaluations of the network can be used to recover the original architecture and parameters, then the results mentioned before represent the limit case where the budget of network evaluations is large enough. However, these results do not address the amount of information necessary to recover a network of a certain complexity. Additionally, the identifiability results [48, 129] do not explain how network parameters may be recovered algorithmically. Since neural networks depend on a finite number of parameters, we would expect that the parameters of an identifiable neural network are determined by a finite amount of network evaluations as well. The uniqueness results above suggest that there exists a large class of neural networks where the parameters are uniquely defined (up to symmetries) by the output map of the network. The theoretical part of this thesis is concerned with the following question: What neural networks can be reconstructed *efficiently* from a finite amount of samples? The emphasis here lies on efficient reconstruction algorithms. Notably, there exist results dating back to the late 80s' that show that learning one hidden layer neural networks is, in general, NP-complete [75, 21]. Even identifying a single neuron can suffer from the so-called "curse of dimensionality" if we do not include regularity conditions on the activation function

and parameters [53, 90]. Hence, a study of tractable network reconstruction is closely tied to the formulation of regularity conditions that exclude these hard cases (cf. Section 3.1.1, Section 4.1.1). To frame the theory presented within this thesis in the context of neural network identification, let us briefly summarize some of our findings. Chapter 3 covers the results from [51], where we address the recovery of a shallow neural network

$$f : \mathbb{R}^D \to \mathbb{R}, \quad f(x) := \sum_{k=1}^{m} g(\langle w_k, x \rangle + \tau_k), \tag{1.9}$$

with smooth activations, bounded shifts, and incoherent weights $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ (cf. Section 2.6.1) in the overcomplete regime where $m \log^2 m = O(D^2)$ (i.e., considerably more hidden neurons than input neurons). We give guarantees for the recovery of the weights and shifts that can be informally summarized as follows:

Theorem 1.1 (Informal, cf. Theorem 3.1 and [51])**.** *Let f be the shallow network as in* (1.9) *with D inputs and m neurons such that $m \log^2 m = O(D^2)$. Then, for sufficiently large D, a constructive algorithm exists recovering all weights and shifts of the network with high probability from $O(Dm^2 \log^2 m)$ network queries.*

For a precise formulation of this statement, we refer to Theorem 3.2 and the related proofs. Notably, this result allows us to infer an upper bound to the sample complexity for the identification of networks as in (1.9). In Chapter 4, parts of this result will be extended to deep neural networks. We will resume the discussion of the techniques used throughout Chapter 3& 4. Let us first mention the relationship between neural network identification and another relevant line of research that is considering so-called *teacher-student models*.

1.3 Teacher-student models

Identifying a network from a finite amount of samples can be phrased as a supervised learning problem (see Section 1.1.3) by assuming the training data is realizable by a neural network. Notably, this setting is supported by various results that show that generic data can be realized by neural network architectures (cf. Section 1.1.3). This setting is also referred to as the *teacher-student* model and is considered to provide insights into the optimization landscape of empirical risk minimization in deep learning as well as the generalization capabilities of neural networks learned via gradient-based methods [25, 124, 111, 85, 41, 42, 114, 115, 137, 56, 54, 52, 53, 74, 86, 93, 138]. In this model, the *teacher network* is assumed to be an unknown neural network that gives rise to the training data. Then, a *student network* is fit onto the realizations of the teacher by a supervised learning algorithm (e.g., empirical risk minimization via gradient descent methods). Let us also mention that this setup finds various applications outside the theoretical literature. For instance, in a mainly applied context, it can be used for model compression, commonly referred to as *knowledge distillation* [64]. Here, one learns a small (student) network based on the input-output information of a much larger teacher network. The hypothesis behind this setup is that the teacher architecture can be compressed without losing much of its expressive properties due to over-parameterization. Such a reduction in network complexity is, for instance, necessary to deploy large neural network models on devices with limited resources (e.g., mobile devices). For more details, we refer to the survey [61]. From a theoretical viewpoint, the teacher-student model represents a fitting environment to study the generalization capabilities of networks learned via supervised learning algorithms since the underlying phenomenon (i.e., the teacher network) is structurally very similar to the learned network itself. More precisely, the student architecture can be chosen so that it, given the correct configuration of parameters, can fully reproduce the input-output map of the teacher network.

Assume, for instance, that the teacher and student networks have identical architectures. Then, the aforementioned connection can be established by analyzing under which circumstances (empirical) risk minimization via SGD (i.e., training of the student network) identifies the parameters of the teacher network (which then guarantees a generalization to unseen data). Let us mention here that most theoretical results build upon distributional assumptions on the network weights, which are typically assumed to follow a standard Gaussian or drawn uniformly at random from the sphere. Implicitly, these assumptions help to exclude degenerate networks where the training is provable hard (cf. Section 1.2 and [75, 21, 53, 90]).

Remark 1.2. *The discussion below is focused on results related to classical feed-forward network architectures due to their relevance to the topics discussed in this thesis. Notably, there exist works studying a theoretical teacher-student model for convolutional neural networks (e.g., [41, 25, 56, 42]) or residual networks (e.g., [85]).*

The works [115, 102] consider a teacher-student setup with shallow neural networks in the moderately *over-parameterized regime*, where the number of training samples N is dominated by the network width m. The work [115] provides a global[2] landscape analysis for shallow neural networks with quadratic activation functions and a local convergence result for certain smooth activations. Their results require $N \leq 2m$. Similarly, [102] proves global convergence of gradient descent applied to the empirical risk with square loss in the over-parameterized regime where $N \lesssim \sqrt{mD}$ and the activation functions are sufficiently smooth. It should be noted that these results show that gradient descent, under suitable conditions, manages to converge to a network that perfectly interpolates the teacher realization, which does, however, not imply the identification of the teacher parameters (i.e., the generalization of the student network).

Another line of work [138, 137, 56] achieves global guarantees for network identification (hence without over-parameterization) by combining a local convergence analysis of gradient-based methods with an initialization strategy based on tensor decomposition in [138] that applies for $m = \mathcal{O}(D)$. The works [138, 137] consider shallow neural networks without shifts. The work [138] provides global guarantees for the identification of linearly independent weights (i.e., $m \leq D$) under mild regularity conditions w.r.t. the activation (which includes smooth functions such as sigmoidals) with polynomial sample complexity. In [137], a local convergence result for gradient-based methods from noisy teacher samples is proven for the ReLU activation, which achieves global guarantees by applying a similar tensor initialization as in [138]. Hence, their results are limited to $m = \mathcal{O}(D)$. Let us note that the local convergence analysis [137] assumes, to the best of our understanding, that the number of neurons is a small constant and independent of the dimension D.

The publications [124, 139] also consider the identification of the teacher weights. However, their results are based on the minimization of the population risk (cf. Section 1.1.3). In [124], a single ReLU unit is considered, while [139] shows that a shallow teacher network with absolute value activation and incoherent weights can be identified by a *larger* student network locally (i.e., whenever the student network is initialized close to the teacher network).

The results described above mainly consider relatively small shallow neural networks. Notably, most works do not regard the recovery of networks with shifts. In the upcoming section, we corroborate the theoretical discussion of the teacher-student model with an empirical study that focuses on more complex network architectures.

[2]Here, global refers to a setting where the learning algorithm starts from a randomly initialized student network, whereas local results typically assume that the student network is already close to the teacher network in some sense.

1.3.1 Identifying a shallow neural network by empirical risk minimization

This section empirically investigates whether gradient-based methods can achieve global identification of (shallow) neural networks. The following experiments, which have partially been published in [51], are designed to get an intuition as to what degree the network complexity (i.e., the width of the network and the input dimension) influences the recovery of the network parameters and to explore the behavior of gradient-based empirical risk minimization in theoretically unexplained regimes. The experimental setting can be described as follows. In every instance of our experiment, we consider a planted shallow teacher network of the type

$$f(x) = \sum_{k=1}^{m} \tanh(w_k^\top x + \tau_k),$$

where the network weights are drawn uniformly at random from the unit sphere, i.e., $w_1, \dots, w_m \in \mathbb{S}^{D-1}$, and the network shifts are sampled according to $\tau_1, \dots, \tau_m \sim_{\text{i.i.d.}} \mathcal{N}(0, 0.05)$. In every instance, a randomly initialized[3] student network $\hat{f}(\cdot, (\hat{w}_k)_{k\in[m]}, (\hat{\tau}_k)_{k\in[m]})$ of identical architecture with parameters $\hat{w}_1, \dots, \hat{w}_m \in \mathbb{R}^D$, $\hat{\tau}_1, \dots, \hat{\tau}_m \in \mathbb{R}$ is fit by minimizing the least-squares error (cf. Section 1.1.3) over N teacher evaluations for inputs $x_1, \dots, x_N \in \mathbb{R}^D$ that are drawn independently from a standard Gaussian distribution. More precisely, consider the *mean squared error* (MSE) given by

$$J((\hat{w}_k)_{k\in[m]}, (\hat{\tau}_k)_{k\in[m]}) = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{f}\left(x_i, (\hat{w}_k)_{k\in[m]}, (\hat{\tau}_k)_{k\in[m]}\right) - y_i \right)^2, \tag{1.10}$$

where $y_i = f(x_i)$ for $i = 1, \dots, N$. The MSE objective is minimized via stochastic gradient descent (SGD) (w.r.t. $(\hat{w}_k)_{k\in[m]}, (\hat{\tau}_k)_{k\in[m]}$) with batch size 64 and learning rate γ and the training is stopped if (1.10) falls below 10^{-8}. Empirically, we observe that stochastic gradient descent with mini-batches does perform better than ordinary gradient descent in the present teacher-student setting. We also favor SGD over ordinary GD as it helps to prevent stagnation of the optimization in a local minimum (cf. [23]), which makes it the more popular choice in practical applications. We track three additional metrics to assess whether the final student network has identified the teacher network. Firstly, to measure the generalization error, we rely on the *uniform error* $E_\infty := \max_{x\in\mathcal{X}_{\text{test}}} |\hat{f}(x) - f(x)|$, which is computed over a set of $|\mathcal{X}_{\text{test}}| = 10^6$ unseen inputs sampled from a standard Gaussian distribution. Secondly, we track the uniform error of the network weights (while accounting for permutations), i.e., $\max_{k\in[m]} \min_{k'\in[m]} \|w_k - \hat{w}_{k'}\|_2$. Lastly, we track the ℓ_2 distance between the shifts while accounting for potential permutations of the student neurons.

Remark 1.3. *All experiments below were performed using one NVIDIA Tesla® P100 16GB/GPU in an NVIDIA DGX-1 architecture. Our experiments were implemented in TensorFlow [2], which offers efficient implementations of classical deep learning algorithms such as stochastic gradient descent and backpropagation. This setup is suitable for large-scale machine learning problems that significantly exceed the complexity of the problem setting we study below. Hence, we do expect that the results below are not strongly influenced by hardware/implementation bottlenecks.*

In the first experiment (cf. Figure 1.2), we fix $D = 25$ and explore in what regimes of N, m the student successfully learned the teacher parameters. To be more precise, the number of training points N, and the number of hidden neurons m will be set to

$$N = \lceil D^\alpha \rceil, \quad \text{and} \quad m = \lceil \frac{2}{5} D^\beta \rceil, \tag{1.11}$$

[3]The student parameters are initialized randomly using the same distributions that are used to instantiate the teacher networks.

Figure 1.2: Performance of empirical risk minimization for the teacher-student environment where $D = 25$.

for varying exponents $\alpha \in \{1.5, 2, 2.5, \ldots, 5\}$, and $\beta \in \{0.5, 0.75, 1, 1.25, 1.5, 1.75, 1.875, 2\}$. Stochastic gradient descent is run with step-size $\gamma = 0.005$ and is stopped after ten minutes of training. We average all metrics over five independent runs for each pair of (m, N). The results are shown in Figure 1.2 and show a clear phase transition that can be categorized into three different regimes.

Interpretation of the results. Firstly, the area starting in the bottom-left corner in light colors corresponds to cases where $\alpha > 2\beta$. Here, SGD converged to student configurations that generalize well to unseen data (i.e., we observe a low uniform error). We observe that cases with a low generalization error also exhibit an identification of the actual network parameters (error on the shifts and the uniform error show the same behavior as E_∞). This is aligned with theoretical results that show, under suitable conditions, that small generalization errors require identification of the network parameters [93]. Notably, parameter identification and low generalization errors were only observed in the under-parameterized regime, where the number of training data exceeds the total amount of network parameters given by $m + Dm$.

The second area, corresponding to the top-right corner, shows a discrepancy between the training error and the other metrics. Here, we exhibit a well-known phenomenon similar to what is known as overfitting (cf. Section 1.1.3). A network with $m = \mathcal{O}(D^\beta)$ neurons has $\mathcal{O}(D^{\beta+1})$ parameters. In other words, if $\alpha \leq \beta + 1$, then the number of samples is very close to the number of parameters or less. This rule of thumb explains the behavior we observe quite well. In all cases with $\alpha \leq \beta + 1$, SGD managed to fit the student network perfectly to the training data, but the resulting student configurations do not generalize well. In the context of the related discussion in the preceding section, we would like to remark that the student network here is not over-parameterized with respect to the teacher network, in which case we clearly observe overfitting if too few training examples are provided. Finally, let us consider the experiments where the number of neurons scales like $O(D^2)$. Here, the learning algorithm was not able to converge consistently to a low generalization error or training error. The fact that we observe perfect identification for $\beta = 15/8$, while stochastic gradient

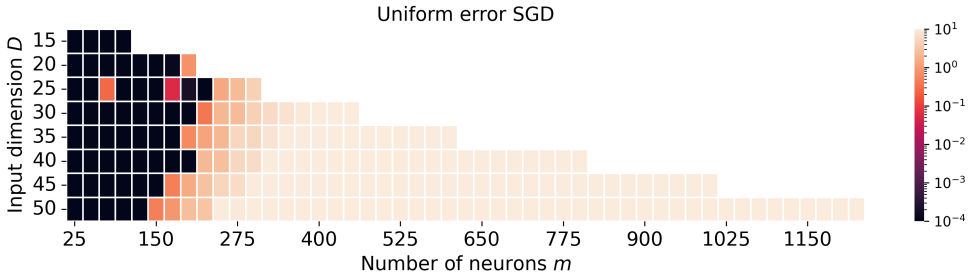

Figure 1.3: Running SGD with fixed learning rate $\gamma = 0.005$ for increasingly bigger networks and for a fixed training time of 10 minutes shows that the observations in Figure 1.2 cannot be easily extrapolated to higher dimensions.

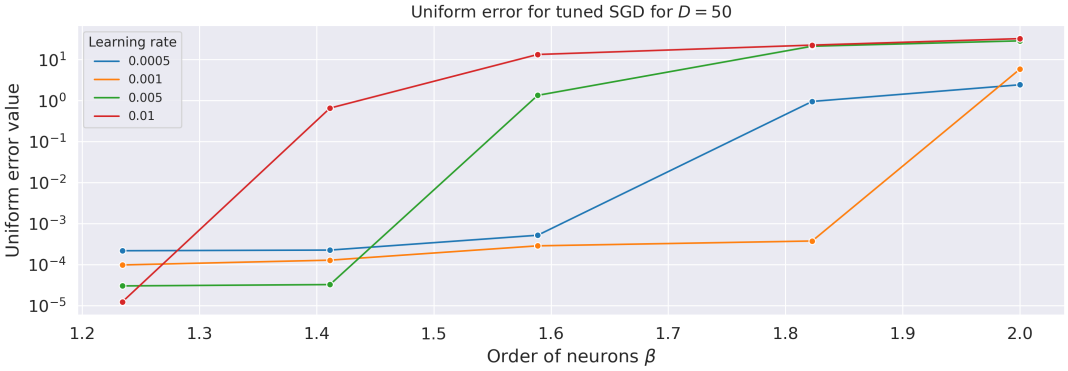

Figure 1.4: Searching for better hyperparameters and running SGD for 50 minutes in the case where $D = 50$ suggests that parameter identification via empirical risk minimization in higher dimensions might be possible. Dots indicate data points averaged over three independent runs, whereas the lines in between are linearly interpolated. Notably, smaller learning rates combined with longer training time have positive effects, but we did not manage to identify the network in the quadratic regime where $\beta = 2$.

descent does not reach a meaningful consensus for $\beta = 2$ (even after a significant increase in training samples), could indicate an inherent hardness of the recovery of very wide networks. Due to the low dimension considered in this experiment, the results in Figure 1.2 are not fully representative since the results might depend additionally on the considered constants. Hence, from Figure 1.2, assessing whether the phenomenon can be extrapolated to the general problem remains challenging. However, we can observe that for increasing D empirical risk minimization only manages to fit relatively small neural networks. This can be seen from Figure 1.3, where the previous experiment is repeated for different dimensions $D = 15, 20, \ldots, 50$ and with $N = 5/2mD^2$ training samples over four repetitions. The reason for this drop-off in performance in higher dimensions might lie in the choice of the hyperparameters (in particular, the learning rate) or simply the training time. To test this hypothesis, we tried different learning rates with a much longer training time of 50 minutes per run for the dimension $D = 50$ and report the results in Figure 1.4. This shows that SGD could potentially be tuned to recover these simple networks in the regime where m outgrows D. However, it seems that this can only be achieved through very long computations. Note that $m = 1000 = 2/5 \cdot 50^2$, so SGD could not recover the regime $\beta = 2$ with a reasonable amount of effort and tuning. We observe that learning shallow neural networks becomes harder as the number of neurons increases. This has also been observed for polynomial activation functions in [93] where a hardness result is provided (see the related discussion in Section 1.4). This observation will be important in the

context of Chapter 3, where we present a reconstruction pipeline that can recover networks in this regime which also comes with theoretical guarantees (see also the numerical experiments in Section 3.6).

1.3.2 Exploring depth in the teacher-student setting

The previous section provides a numerical analysis of empirical risk minimization for shallow neural networks that shows that suitable shallow neural networks up to a certain width can be identified by minimization of the empirical risk via stochastic gradient descent. Another class of networks that is mainly disregarded by the previously discussed theoretical results is the class of deep feed-forward neural networks (cf. Section 1.1.2). The following numerical experiments have partially been published in the joint work [50]. In the following, we consider two different types of teacher architectures: firstly, three-layer neural networks $f : \mathbb{R}^D \to \mathbb{R}^{m_3}$ with $D = 50, m_1 = 50, m_2 = 25, m_3 = 10$ (abbreviated by [50,25,10]); secondly, four-layer neural networks $f : \mathbb{R}^D \to \mathbb{R}^{m_4}$ with $D = 50, m_1 = 50, m_2 = 35, m_3 = 25, m_4 = 10$ ([50,35,25,10]). For each neuron, a weight vector is drawn independently from the respective unit sphere, and a shift is sampled randomly from $\mathcal{N}(0, 0.005)$. Similarly to the previous section, we minimize the empirical risk w.r.t. to the network parameters of the student network $\hat{f}$, i.e., we minimize

$$J((\widehat{W}^{[\ell]})_{\ell\in[L]}, (\hat{\tau})_{\ell\in[L]}) = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{f}\left(x_i, (\widehat{W}^{[\ell]})_{\ell\in[L]}, (\hat{\tau})_{\ell\in[L]}\right) - f(x_i)\right)^2 \tag{1.12}$$

via stochastic gradient descent. Throughout the experiments, the batch size remains fixed at 32, and the learning rate of SGD is varied ($\gamma \in \{2^{-1}, 2^{-2}, 2^{-3}, 2^{-4}, 2^{-6}, 2^{-8}\}$), and the number of samples training ($N = D^2 m$ and $N = D^3 m$), which are sampled from a standard Gaussian distribution. All experiments are repeated five times, and we report the relative uniform error

$$E_\infty := \frac{\max_{x\in\mathcal{X}_{\text{test}}} \|\hat{f}(x) - f(x)\|_2}{\max_{x\in\mathcal{X}_{\text{test}}} \|f(x)\|_2}$$

computed on unseen inputs $\mathcal{X}_{\text{test}}$ sampled from a standard Gaussian distribution, where $|\mathcal{X}_{\text{test}}| = 10^6$. SGD is run for at least two hours for each experiment instance, and E_∞ is recorded after every epoch (i.e., one pass over the training data set). The results are summarized in Figure 1.5 and show the development of the relative uniform error over time. Solid lines indicate the average relative uniform error, whereas the colored regions contain all individual runs similar to confidence intervals. The upcoming results have partially been published in [50].

Interpretation of the results. We can detect several trends. Firstly, higher learning rates generally perform better (cf. brown plots corresponding to $\gamma = 0.5$), and we can see a consistent increase in the averaged relative uniform error for decreasing learning rates. The most plausible explanation for this behavior is that higher learning rates simply converge quicker and manage to reach a better student network configuration within the time limit of two hours. Another reason for this behavior could be that larger learning rates help to avoid spurious local minima in the initial training phase and prevent SGD from getting stuck. There is evidence that large initial learning rates can aid generalization (cf. [84]), but we can not directly assess whether this is the case in our experiments. Notably, however, we do observe a significant improvement for smaller learning rates when learning shallow neural networks in an almost identical setup (cf. Figure 1.4 and Section 1.3.1). Secondly, the results do not exhibit any significant improvement when more training samples are provided to SGD. This follows by comparing the plots in the top row with the bottom row. Hence, $N = D^2 m$ samples seem

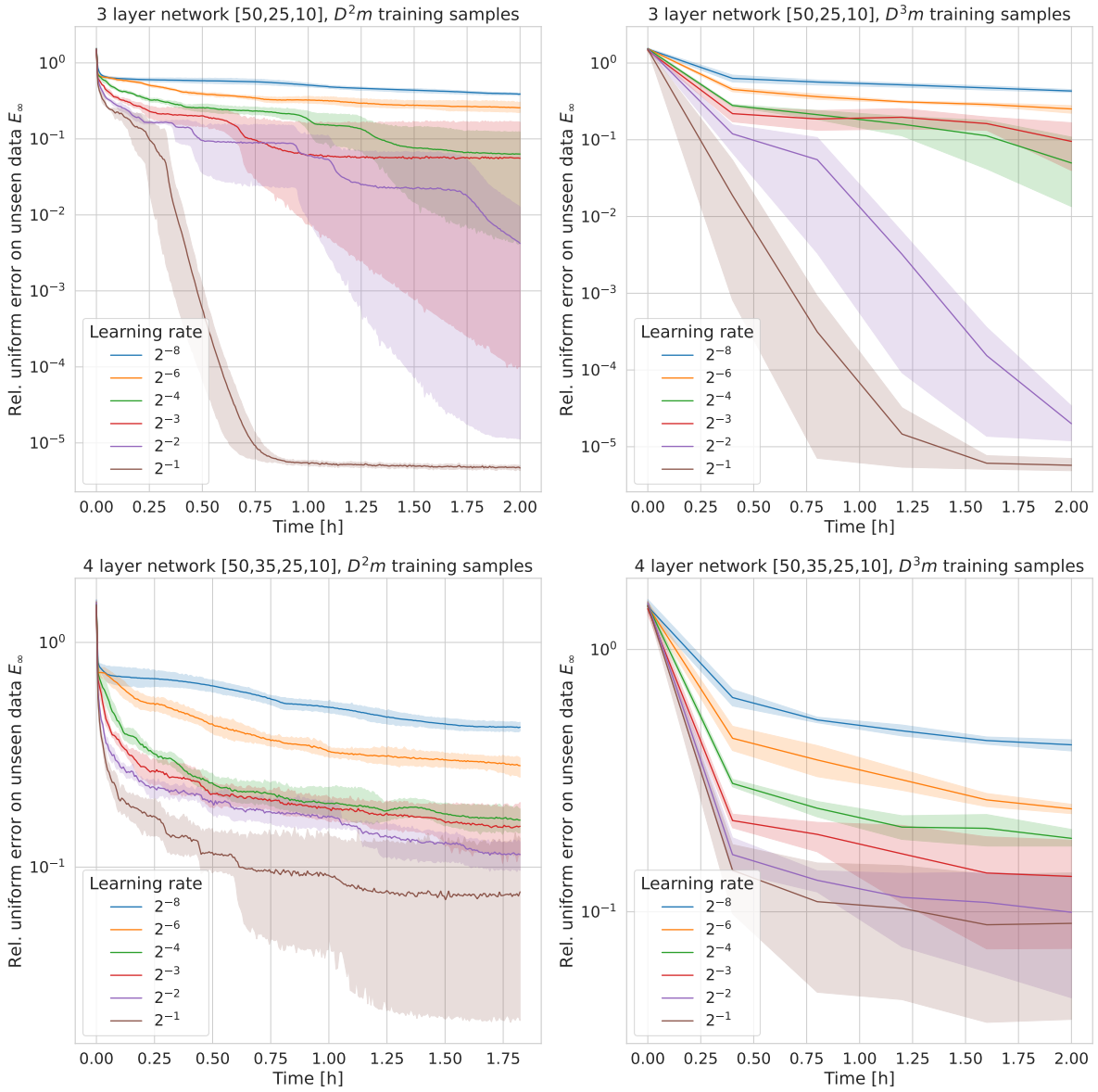

Figure 1.5: Learning the networks [50,25,10], [50,35,25,10] in a teacher-student setting from a random initialization via SGD (cf. [50]). For each learning rate, SGD was run five times with batch size 32 for two hours. The dark lines represent the average trajectories, whereas the light regions contain all individual repetitions. The y-axis depicts the relative uniform error on 10^6 unseen samples.

to provide sufficient information for the identification of the three-layer neural network with architecture $[50, 25, 10]$. This suggests that once sufficiently many samples are provided, the configuration of the supervised training algorithm determines whether a suitable network configuration is attained. Lastly, learning in the four-layer case seems to stagnate (around $E_\infty = 0.1$) after a short initial improvement and generally performs quite poorly compared to the three-layer scenarios (by several orders of magnitude in some cases).

Comparison with the shallow case. If we compare[4] these results with our discussion from the preceding section, then it seems that empirical risk minimization via SGD for deep neural networks is much harder than identifying shallow neural networks of similar size. The deep networks considered here only have slightly more neurons than inputs ($m < 2D$ in the three-layer case and $m < 3D$ in the four-layer case). Shallow networks of similar size were recovered consistently by an identical approach (cf. Section 1.3.1). Furthermore, there seems to be a significant increase in the required number of gradient steps (i.e., computational time) from shallow architectures (two-layer) to three-layer networks. Ultimately, these experiments cannot answer whether neural networks with a high number of layers can be completely identified by empirical risk minimization. The configuration of the training algorithm (in this case, stochastic gradient descent with a fixed learning rate) seems to play a significant role. However, it can be observed that the results for small-sized shallow neural networks from Section 1.3.1 can not be easily extrapolated to deep neural networks.

1.4 Reduction of weight identification to tensor decompositions

Certain small-sized feed-forward neural networks are identifiable via empirical risk minimization (cf. Section 1.1.3 and Section 1.3). However, theoretical guarantees discussed previously either do not extend to more complex network architectures (i.e., networks with wide or multiple hidden layers), or they do not consider a suitable setting for parameter identification (e.g., works that consider over-parameterization). In numerical experiments, stochastic gradient descent does manage to go beyond the aforementioned theoretical guarantees but becomes less consistent with higher network complexity. We did not manage to identify a shallow teacher network where $m = \mathcal{O}(D^2)$ (cf. Section 1.3.1), and we could not consistently train a four-layer student network that generalizes well on unseen data of the teacher network (cf. Section 1.3.2). Therefore, it remains unclear whether empirical risk minimization admits global guarantees for the identification of complex neural networks. This creates a discrepancy since these types of networks should be uniquely determined by their input-output information (cf. Section 1.2 and [48, 129]). This section presents another line of work that approaches the identification of shallow neural networks without empirical risk minimization but, differently from before, leverages higher-order network differentiation and symmetric tensor decompositions [111, 54, 53, 74, 86, 93, 138].

Tensor methods were already briefly mentioned as an initialization strategy for some of the previously discussed works in Section 1.3 that provide a local convergence analysis of gradient-based methods. The work [93] shows that under a suitable setting, the learning of shallow neural networks with polynomial activations can be reduced to the problem of decomposing a n-tensor

$$\mathcal{T}_n = \sum_{k=1}^{m} w_k^{\otimes n} \in (\mathbb{R}^D)^{\otimes n} \tag{1.13}$$

into its rank-one components $w_1^{\otimes n}, \dots, w_m^{\otimes n}$, where $(w_k)_{k\in[m]}$ are the weights of the shallow network and n is the degree of the activation function. Reductions of this type provide valuable insights since the hardness of overcomplete tensors (i.e., tensor with $m \geq D$ components) is an active area of research. Many tensor problems are known to be intractable [37, 62, 63]. In the regime where $m \leq D$, the factorization in (1.13) for third-order tensors is known to be unique under fairly mild conditions [79, 19] and admits tractable computation [24]. The factorization of a third-order symmetric tensor $\mathcal{T}_3$ into its components $w_1^{\otimes 3}, \dots, w_m^{\otimes 3}$ in the overcomplete

[4]The fact that multiple output neurons are considered should not negatively skew this comparison. We observe that multiple output neurons aid the identification of network parameters (cf. Chapter 4).

regime, where $m \geq D$, is considered in a wide range of recent literature (e.g., [15, 87, 66, 8, 93]). For instance, the best currently known polynomial-time algorithms for the decomposition of symmetric 3-tensors with components $w_1, \dots, w_m \sim_{\text{i.i.d.}} \text{Unif}(\mathbb{S}^{D-1})$ only offer guarantees up to the regime where $m \leq D^{3/2}$ (up to poly-logarithmic factors) [87]. The rank-one factorization of symmetric 3-tensors beyond the regime $m > D^{3/2}$ is currently considered to be computationally hard [93]. Hence, under suitable conditions, [93] shows that learning very wide shallow neural networks is provable hard when the activations are given by low-degree polynomials. This also suggests that methods that explicitly consider weight retrieval by decomposition of 3-tensors, such as the initialization of student networks by tensor methods (e.g., [138, 137, 56], which require decomposition of 3-tensors and were mentioned within Section 1.3), as well as other tensor-based approaches (see [74] below) might be challenging to extend to the recovery of very wide shallow networks in the regime $m > D^{3/2}$.

Remark 1.4. *Our review of the literature surrounding tensor decompositions is necessarily incomplete. The aforementioned limitation on the number of components could potentially be circumvented by considering higher-order tensors, such as fourth-order tensors, which do admit efficient decompositions beyond $D^{3/2}$ components (see for example [65]). Since our results discussed in the main theoretical chapters of this thesis (Chapter 2-4) do not directly rely on the theory surrounding higher-order tensor decompositions, a more detailed discussion of tensor methods would go beyond the relevant scope. For more details, we refer to the cited literature and references therein.*

Exposing weights by adaptive sampling strategies. One aspect typically not considered in the study of supervised learning algorithms is the use of controlled sampling strategies. Intuitively, it should be easier to identify the network parameters from input-output pairs when the network inputs can be actively chosen. We refer to this setting as *active sampling* or *querying*. As a matter of fact, adoptive sampling schemes have been considered in a similar context for the uniform approximation of *ridge functions* [33]. A ridge function can be described by the composition of a univariate function with an affine function, i.e., functions $x \mapsto g(w^\top x)$, where $w \in \mathbb{R}^D, g : \mathbb{R} \to \mathbb{R}$. This gives rise to a neuron, and by considering sums of ridge functions

$$f(x) = \sum_{k=1}^{m} g_k(w_k^\top x), \tag{1.14}$$

one can represent any type of shallow neural network. We can view shallow neural networks as a special case of sums of ridge functions since neural networks generally assume that $g_k(\cdot) = b_k g(\cdot + \tau_k)$. Models of the type (1.14) and their approximation have been considered in mathematical statistics under the terminology projection pursuit or projection pursuit regression, cf. [71, 55, 39, 83], as well as single index models [72, 69]. One line of research addressed the uniform approximation of (sums of) ridge functions by leveraging adaptive sampling schemes and queries [33, 53, 90]. Here, the authors consider the approximation of ridge functions $g(w^\top x)$ in a setting where w and $g \in C[0,1]$ are both unknown. The authors also allow for queries and study how to organize the queries to achieve the best uniform approximation.

Assuming sufficient smoothness, then a simple but effective way of active sampling is to design query strategies that allow for the approximation of derivatives with respect to the input. In non-technical terms, this requires local information of the network or ridge function, where several clusters of concentrated inputs contain sufficient information to approximate the desired derivative. Note that differentiation of functions of the type (1.14) exposes the interior parameters since

$$\nabla^n f(x) = \sum_{k=1}^{m} g_k^{(n)}(w_k^\top x) w_k^{\otimes n}. \tag{1.15}$$

The resulting connection between the weights of a shallow neural network (with sufficiently smooth activations) and symmetric tensors based on network differentiation has been explored in [27, 105, 31] and more recently in [111, 54, 53, 74, 86, 93, 138, 54] and our own work [52, 50, 51], which will be discussed in the upcoming chapters. These approaches leverage the structure of (1.15) to decouple the weight recovery of shallow networks from the remaining network parameters. The weights can then be learned in the first phase, which leaves only a few unknown scaling and shift parameters. The remaining unknowns can be estimated by various methods such as Fourier methods (cf. [74]), direct estimation (cf. [54]), or by empirical risk minimization (cf. [51] and Chapter 3). A simple decoupling algorithm for a single neuron is sketched in the following example.

Example 1.1. *Let $f(x) = g(w^T x + \tau)$ be a neural network with weight $w \in \mathbb{R}^D$, shift $\tau \in \mathbb{R}$, and known differentiable activation g. The goal is the recovery of the weight vector w and shifts τ from queries of f. To compute* (1.15) *for $n = 1$ we can now leverage the active sampling. In a certain sense, derivatives represent local information, which is hard to attain from separated samples. However, by probing, we can compute the change of f along any given direction v and, therefore, approximate network derivatives via finite difference schemes such as*

$$\frac{f(x+\epsilon v) - f(x)}{\epsilon} \approx \lim_{\epsilon \to 0} \frac{f(x+\epsilon v) - f(x)}{\epsilon} = \nabla f(x)^T v.$$

If we were able to find an approximation $\Delta f(x) \approx \nabla f(x)$ to a non-zero gradient, we could use it to estimate the weight up to scaling with

$$\hat{w} := \frac{\Delta f(x)}{\|\Delta f(x)\|_2} \approx \alpha w \quad \text{for } \alpha \in \mathbb{R} \setminus \{0\}. \tag{1.16}$$

The remaining part of the initial reconstruction can then be framed as a recovery of the parameters α, τ from a scalar reparametrization $\hat{f}(t) := f(t\hat{w}) \approx g(\|w\|_2 \alpha t + \tau)$ of the original network f. Different approaches can be taken to solve the recovery of the scaling parameters and the shift from the function $\hat{f}$, which already leads to one of the main topics discussed in Chapter 3.

Estimating network derivatives from data. Admittedly, having full control over the input, as assumed in an active sampling setting, is not fulfilled when a neural network is trained on data. Therefore, understanding to what degree the derivatives of networks can be computed from input-output data that is fixed a priori potentially makes derivative-based network reconstruction algorithms applicable to the learning problem. In [54, 74], the authors show that the active sampling assumption can be relaxed and that knowledge of the input distribution can suffice. This approach is primarily based on Stein's lemma [119, 8] or simply differentiation by parts [54]: Assume we are given access to N inputs $x_1, \dots, x_N$ drawn independently from a distribution μ with *known density* $p(x)$ w.r.t. the Lebesgue measure $d\mu(x) = p(x)dx$ and support $\Omega = \operatorname{supp}(\mu)$. If we assume that p is sufficiently smooth, then we estimate $\mathbb{E}_{x\sim\mu}[\nabla^n f(x)]$ without querying the network explicitly. This follows from

$$\begin{aligned}
\frac{1}{N}\sum_{i=1}^{N} f(x_i)(-1)^n \frac{\nabla^n p(x_i)}{p(x_i)} &\approx \int_\Omega f(x)(-1)^n \frac{\nabla^n p(x)}{p(x)} p(x)dx \\
&= \int_\Omega \nabla^n f(x) d\mu_X(x) = \mathbb{E}_{x\sim\mu}[\nabla^n f(x)] \\
&= \sum_{k=1}^{m} \left(\int_\Omega g^{(n)}(w_k^T x) d\mu(x) \right) w_k^{\otimes n}.
\end{aligned} \tag{1.17}$$

One potential benefit associated with this construction is that it might help to avoid potential numerical instabilities that would be caused by derivative approximation [54] in the active

sampling case since the approximation error in (1.17) scales with the number of samples N. It should be noted, however, that any errors in the approximation of derivatives via numerical differentiation will ultimately be caused by inaccuracies of the network evaluations (e.g., machine precision) and should also be present in the left-hand side of (1.17). We will refer to this construction as *passive sampling* (cf. [54]). These ideas have gotten more attention recently (see [74, 93], as well as [53, 54]). What makes this type of sampling appealing is that without explicit probing of the network, it can be applied to the classical learning problem. For instance, if we were given training data $(x_i, y_i)_{i\in[N]}$ where $y_i = f(x_i)$ is known to be exactly realizable by a shallow neural network, then $\mathbb{E}_{x\sim\mu}[\nabla^n f(x)]$ can be approximated as long as we assume the inputs come from an at least approximately known distribution μ. The last part is critical since the estimation of a density becomes increasingly more difficult for higher dimensions.

1.5 Reconstruction of shallow neural networks from Hessian information

The decomposition of network derivatives into sums of outer-weight-products as shown in (1.15) suggests a clear path for the (partial) identification of (smooth) shallow neural networks. In Example 1.1, the identification of a single neuron was connected to its gradient. For general shallow networks $f(x) = \sum_{k=1}^m g(w_k^\top \tau_k)$, however, network gradients will not suffice because the gradient takes the form

$$\nabla f(x) = \sum_{k=1}^m g^{(1)}(x^\top w_k + \tau_k) w_k \in \operatorname{span}\{w_1, \dots, w_m\}.$$

Since identifying individual weights from their linear combination is not possible, higher-order differentiation has been considered (cf. [31, 27, 54]), which gives rise to a symmetrical tensor as in (1.15). Notably, the identification of the individual weights still remains challenging (see also Section 1.4) after its reduction to the problem of decomposing a tensor

$$\mathcal{T}_n = \sum_{k=1}^m \lambda_{k,n} w_k^{\otimes n}, \quad \text{for } n \geq 2, \tag{1.18}$$

with unknown coefficients $\lambda_{1,n}, \dots, \lambda_{m,n} \in \mathbb{R}$. First of all, regardless of whether passive or active sampling is considered (see above), the tensor $\mathcal{T}_n$ can only be estimated up to some accuracy. Hence, one requires a robust method to decompose $\mathcal{T}_n$. Decompositions of symmetric tensors ($n \geq 3$) or matrices ($n = 2$) are known to be unstable without the separation of the individual components. Additionally, the coefficients $\lambda_{k,n}$ can not be allowed to become arbitrarily small (compared to the error caused either by approximations of network derivatives or the approximation of the input distributions whose higher-order moments are used in passive sampling).

Let us now address one particular line of work that is inspired by [54] and considers the identification of shallow neural networks with linear outputs and sufficiently smooth activations [54, 51] (cf. Chapter 3). Notably, these ideas have been extended in parts to deep neural network architectures [52, 50] (cf. Chapter 4). The authors in [54] approach the stability issues mentioned above by using multiple network derivatives, and their reconstruction algorithm relies exclusively on Hessian matrices (i.e., second-order derivatives) of the network. By remaining in the realm of matrices, their approach avoids most computational intractability issues associated with tensors since matrix decompositions are generally well-understood and computable in polynomial time. However, matrix decompositions, such as the spectral- or singular value decomposition, are generally only unique up to unitary transformations and less

suited for the overcomplete regime $m \geq D$. Therefore, [54] considers instead the approximation of the symmetric matrix space spanned by the outer products of the network weights. To be more precise, assume $f : \mathbb{R}^D \to \mathbb{R}$ is a shallow neural network on $m \leq D$ hidden neurons with sufficiently smooth scalar activations $g_1, \dots, g_m$, linearly independent weights $w_1, \dots, w_m \in \mathbb{R}^D$, and one linear output unit, such that

$$f(x) = \sum_{k=1}^{m} g_k(x^\top w_k). \tag{1.19}$$

Note that this definition is equivalent to the sum of ridge functions in (1.14) and that network shifts can easily be incorporated by setting $g_k(t) = g(t + \tau_k)$ for an arbitrary activation g. The goal is the retrieval of $(w_k)_{k\in[m]}, (g_k)_{k\in[m]}$ from network queries. In the first step, the problem is reduced from $m \leq D$ to the setting where $m = D$ by a network transformation that projects network inputs onto the active subspace, i.e., the space $\operatorname{span}\{w_1, \dots, w_m\}$ (see Remark 1.5 below). The recovery algorithm of [54] computes an approximation $\widehat{\mathcal{W}}$ to the symmetric matrix space

$$\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\} \subset \operatorname{Sym}(\mathbb{R}^{D\times D}), \tag{1.20}$$

from the m-th left singular subspace[5] of N Hessian approximations $\Delta^2 f(x_1), \dots, \Delta^2 f(x_N)$, i.e., $\widehat{\mathcal{W}} = \operatorname{span}_m\{\Delta^2 f(x_1), \dots, \Delta^2 f(x_N)\}$. Here, $(x_i)_{i\in[N]}$ are considered to be random vectors sampled uniformly at random from the unit-sphere, and $\Delta^2 f(x_i) \approx \nabla^2 f(x_i)$ are Hessian approximations (e.g., constructed via finite differences as described in Section 1.4). By assuming that combining sufficiently many Hessians matrices at inputs drawn from $\operatorname{Unif}(\mathbb{S}^{D-1})$ provides sufficient information to construct $\mathcal{W}$, [54] then show that, with high probability, the construction of $\widehat{\mathcal{W}}$ serves as a good approximation provided that $N \gtrsim m^2$ (see also Section 3.2 and Algorithm 3.2). More precisely, the key assumption in [54] that enables this statement is that the second moment of Hessians in expectation, i.e., the matrix

$$H[f] := \mathbb{E}_{X\sim\operatorname{Unif}(\mathbb{S}^{D-1})}\left[\operatorname{vec}(\nabla^2 f(X))^{\otimes 2}\right], \tag{1.21}$$

has rank m. Similar assumptions are encountered in the theory presented in Chapter 3 and Chapter 4 (cf. **(SNM3)**, **(DNM3)** defined in Section 3.1.1, Section 4.1.1, respectively). It is worth mentioning that we prove with Theorem 3.5 in Section 3.4.5 that this condition holds for incoherent weights[6] and sufficiently non-polynomial activations. More precisely, we show that the m-th singular value of the second moment matrix satisfies $\sigma_m(\mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id})}\left[\operatorname{vec}(\nabla^2 f(X))^{\otimes 2}\right]) > \alpha$ for a constant $\alpha > 0$ that is *independent* of m, D (which implies the rank condition). Assuming access to $\widehat{\mathcal{W}} \approx \mathcal{W}$, the recovery of individual weights can then be achieved robustly by selecting matrices within $\widehat{\mathcal{W}}$ that are close to one of the rank-one spanning matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ up to sign. These so-called *near-rank-one matrices* in $\widehat{\mathcal{W}}$ are characterized by a significant spectral gap at the leading singular value under fairly mild conditions, which guarantees the stability of the associated eigenvector (this follows from classical perturbation analysis of matrix decompositions [108, 20]). The selection of the near-rank-one matrices from the space $\widehat{\mathcal{W}}$ is formulated as the nonlinear program

$$\operatorname{argmax} \|M\| \quad \text{s.t.} \quad M \in \widehat{\mathcal{W}}, \quad \|M\|_F \leq 1. \tag{1.22}$$

[54] show that for nearly orthonormal weights $w_1, \dots, w_m$ the local maximizers of (1.22) are close to the original spanning elements up to sign (assuming sufficiently accurate derivative

[5] See Section 1.1.1 for a definition of singular subspaces.

[6] Our incoherence assumptions hold with high probability for weights drawn from isotropic random distributions such as the uniform distribution on the unit-sphere for up to $m = o(D^2)$ neurons (cf. Section 2.6.1).

approximations) and that their computation can be achieved by a projected gradient ascent (cf. Section 2.3.4). For more details, we refer to Chapter 2, where the results in [54] are extended to the overcomplete regime $m = \mathcal{O}(D)$ under deterministic frame bounds and additionally discuss further extensions to $m = o(D^2)$ for isotropic random weights based on the recent results [76]. To make this approach applicable to shallow networks with linearly independent weights, which are not necessarily orthogonal, [54] propose an optional *whitening step* that serves as a reduction of the problem to the orthonormal case: Assume access to $\widehat{\mathcal{W}} \approx \mathcal{W}$ and that $w_1, \dots, w_m$ are linearly independent. Select any positive definite matrix $G \in \widehat{\mathcal{W}} \subset \mathrm{Sym}(\mathbb{R}^{D\times D})$ and denote by $G = V\Sigma V^\top$ its spectral decomposition. The transformation $\tilde{f}(x) := f(\Sigma^{-1/2}V^T x)$ then gives rise to a shallow neural network $\tilde{f}$ (of the form (1.19)) with nearly orthonormal weights given by $\tilde{w}_k := \Sigma^{-1/2}V^T w_k / \|\Sigma^{-1/2}V^T w_k\|_2, k \in [m]$ (cf. [54, Theorem 8.3]). After finding weight approximations $\hat{w}_1, \dots, \hat{w}_m$, the remaining identification of the activation functions g_k is then tackled by an adoptive sampling schema that considers the network along the directions $t \mapsto f(tb_k)$, where $b_1, \dots, b_m$ is the dual basis to $\hat{w}_1, \dots, \hat{w}_m$ (cf. [54, Algorithm 7.1]).

Remark 1.5. *Note that any component of an input vector x that does not lie in the span of the first layer weights* $\mathrm{span}\{w_1, \otimes w_m\}$ *will be ignored by the network since*

$$w_k{}^\top x = w_k{}^\top P_{\mathrm{span}\{w_1,\dots,w_m\}}(x),$$

where $P_{\mathrm{span}\{w_1,\dots,w_m\}}$ is the orthogonal projection onto the span of the weights. The linear span of the weigths $w_1, \dots, w_m$ is called the active subspace *(cf. [54]) and has dimension m due to the linear independence of the weight vectors. The active subspace can be identified by gradient approximations similar to the approach that was used to identify the space* (1.20) *under the assumption that $\mathbb{E}_{X\sim\mathrm{Unif}(\mathbb{S}^{D-1})}\left[(\nabla f(X))^{\otimes 2}\right]$ has full rank. This yields a transformation of the network that only considers inputs in the active subspace, and thereby reduces the problem to the setting where $m = D$.*

Chapter 2

Recovery of a rank-one basis from its perturbed span

This chapter is concerned with the following problem: Consider a set of m rank-one symmetric matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m \in \mathrm{Sym}(\mathbb{R}^{D\times D})$. Assume that the linear span of these elements is known approximately, i.e., we are given access to a symmetric matrix space $\widehat{\mathcal{W}}$ that approximates the span $\widehat{\mathcal{W}} \approx \mathrm{span}\,\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$. In this setting, the problem we study is the (approximative) recovery of the original spanning elements $w_1 \otimes w_1, \dots, w_m \otimes w_m \in \mathrm{Sym}(\mathbb{R}^{D\times D})$ from the perturbed matrix space $\widehat{\mathcal{W}}$ (up to natural symmetries). We will refer to this problem as the *rank-one basis recovery (problem)*. Let us mention that the rank-one basis recovery problem occurs, for instance, during the identification of shallow neural networks (cf. Section 1.5 and Chapter 3), as well as during the reconstruction of deep neural networks in Chapter 4. A detailed summary of the upcoming theory is provided as part of the next section.

Section 2.3 of the present chapter covers parts of the joint work with Massimo Fornasier, Timo Klock published in [52], which extends the theory of [54] (cf. Section 1.5) to the overcomplete regime $m < 2D$. Section 2.4 and Section 2.5 extend the analysis of the subspace power method in [77] to the perturbed case for matrix and is based on the joint work [50] with Massimo Fornasier, Timo Klock and Christian Fiedler. Lastly, Section 2.6 is concerned with the overcomplete regime where $D < m < D^2$, where we present a result from [76] and introduce concepts of incoherence that will be highly relevant in Chapter 3. The included discussion addresses similar points that have been part of the joint work [51] with Massimo Fornasier, Timo Klock and Marco Mondelli. Several of the theoretical proofs within this chapter, which originate from these joint works, are stated verbatim as in their underlying references, indicated within the definition of the statement.

2.1 Introduction and preliminaries

We study the approximation of the elements $w_1 \otimes w_1, \dots, w_m \otimes w_m \subset \mathrm{Sym}(\mathbb{R}^{\mathrm{D}\times\mathrm{D}})$ under the assumption that their span is known to us up to small perturbations, i.e., we assume knowledge of a symmetric matrix space $\widehat{\mathcal{W}} \approx \mathcal{W}$ where

$$\mathcal{W} = \mathrm{span}\,\{w_1 \otimes w_1, \dots, w_m \otimes w_m\} \subseteq \mathbb{R}^{D\times D}. \tag{2.1}$$

To understand our approach to this problem, let us first note that this problem is related to the spectral decomposition: Consider, for instance, the unperturbed problem, i.e., we are given direct access to $\mathcal{W}$. Assume further that $m = D$ and that $w_1, \dots, w_m \in \mathbb{R}^D$ form an orthonormal basis. Then, for any non-singular matrix $M \in \mathcal{W}$ with *distinct eigenvalues*, we can compute

all spanning elements using the spectral decomposition, which reads $M = \sum_{k=1}^{D} \lambda_k u_k \otimes u_k$ for eigenvalues $\lambda_1, \dots, \lambda_D \in \mathbb{R} \setminus \{0\}$ and eigenvectors $u_1, \dots, u_D$. Due to the uniqueness of the spectral decomposition, in the case described above, we have

$$M = \sum_{k=1}^{D} \lambda_k u_k \otimes u_k = \sum_{k=1}^{m} \lambda_k w_k \otimes w_k.$$

Unfortunately, this approach comes with several limitations: Firstly, it is well known that even for distinct eigenvalues the spectral decomposition is in general *not stable*, which means this approach can not be directly applied to the perturbed rank-one recovery problem. The stability of the spectral decomposition can only be guaranteed under the presence of sufficient *spectral gaps* (i.e., separation of the eigenvalues). Secondly, the number of components is limited by the dimension of the matrix, i.e., the spectral decomposition of any matrix $M \in \mathcal{W}$ can only be decomposed into D spanning elements. Notably, we can have up to $m = \mathcal{O}(D^2)$ linearly independent rank-one spanning matrices (limited only by the dimension of $\mathrm{Sym}(\mathbb{R}^{\mathrm{D}\times\mathrm{D}})$). Therefore, we are naturally interested in an approach that also applies to the *overcomplete regime*, where the number of spanning elements m is greater than the ambient dimension D. Lastly, assuming that the vectors $w_1, \dots, w_m$ form an orthonormal basis is overly restrictive and does not apply in many relevant settings (cf. Chapter 3-4).

The present chapter addresses the limitations above and is largely based on the theory presented in the publications [52, 50]. More precisely, we provide a perturbation analysis as well as an extension to the overcomplete regime $m > D$. Before we move to the technical results, let us summarize some of the fundamental ideas that address the problems mentioned above and, at the same time, give an overview of the structure of the chapter.

Near rank-one matrices approximate spanning elements (Sec. 2.2). Our first goal is to select matrices within the perturbed space $\widehat{\mathcal{W}}$ such that their eigenvectors are close to a subset of the vectors $\{w_1, \dots, w_m\}$. To achieve such a selection *while maintaining stability*, we select only those matrices in $\widehat{\mathcal{W}}$ that are *near-rank-one*, i.e., matrices with one dominant eigenvalue, which guarantees a large spectral gap. One immediate consequence of this approach is that aside from the first eigenvalue, the remaining eigenvalues of the selected matrix cannot have significant spectral gaps. Therefore, we can only rely on the eigenvector corresponding to the largest eigenvalue, which in turn implies that we cannot expect to recover more than one out of the m rank-one spanning matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ from each selected matrix. Hence, our approach requires at least m different matrices $M_1, \dots, M_m \in \widehat{\mathcal{W}}$ to recover all spanning elements. In Section 2.2, we prove (cf. Theorem 2.1) that near-rank-one matrices are close to one of the spanning elements $w_1 \otimes w_1, \dots, w_m \otimes w_m$ under mild conditions.

Characterization of near-rank-one matrices (Sec. 2.3.1, Sec. 2.5). To select such near-rank-one matrices in $\widehat{\mathcal{W}}$, we provide two different characterizations that select near-rank-one matrices as maximizers of a non-linear program. Depending on the characterization, we end up with one of two different recovery algorithms. In Section 2.3, we present a characterization based on the program that selects the maximizers of the spectral norm constrained on the Frobenius unit ball, i.e., maximizers of the non-linear program

$$\operatorname{argmax} \|M\| \ \text{ s.t. } \ \|M\|_F \leq 1, M \in \widehat{\mathcal{W}}. \tag{2.2}$$

The characterization in (2.2) originates from [54], where it was used as a characterization of the near-rank-one matrices in the case $m = D$. Section 2.3 covers the theory within [52], that extends the characterization in (2.2) to the overcomplete linear regime where the vectors $w_1, \dots, w_m$ form

a frame. More precisely, in Section 2.3.1, we show that for small perturbations and separated spanning elements, the local maximizers of (2.2) are either near-rank-one matrices which by Theorem 2.1 implies that they are close to one of the matrices $\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$, or they have a uniform spectrum. Starting from Section 2.5, we study a different characterization based on the nonlinear program

$$\max_{\|u\|=1} \left\| P_{\widehat{\mathcal{W}}}(u \otimes u) \right\|_F^2, \tag{2.3}$$

where $P_{\widehat{\mathcal{W}}}$ represents the orthogonal projection onto the matrix space $\widehat{\mathcal{W}}$. Note that any maximizer $\hat{u}$ in (2.3) can directly be associated with the rank-one matrix $\hat{u} \otimes \hat{u}$. The problem in (2.3) has been first analyzed in [77] in a more general context of tensor decomposition, and their results apply directly to the unperturbed rank-one recovery problem. It is evident that *global maximizers* of (2.2) and (2.3) must closely resemble rank-one matrices as long as the space $\widehat{\mathcal{W}}$ is sufficiently close to span $\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$. However, to allow a sufficient computation of these maximizers by first-order methods (e.g., projected gradient ascent), we need to provide a perturbation analysis of the *local maximizers* of these programs. The primary technical challenge then becomes the formulation of conditions derived from the first-order and second-order optimality conditions that guarantee that the local maximizers of either program are close to $\pm w_k \otimes w_k$ for some $k \in [m]$. Our main focus in Section 2.5 is the perturbation analysis contained in [50], which extends the results in [77] to our scenario for $m < 2D$. Finally, the recent work [76] provides an analysis of the perturbed objective (2.3) up to $m = o(D^2)$ for vectors $w_1, \dots, w_m \in \mathbb{S}^D$ that can be modeled by isotropic random vectors. Their work presents an improvement over [50] and the main result applied to our case will be discussed in Section 2.6, which will find application in Chapter 3. However, we will not discuss the technical details of [76], as their theory is focused on tensor decompositions.

Remark 2.1. *In both linear programs (*(2.3) *and* (2.2)*), we encounter spurious local maximizers that fulfill the first-order and second-order optimality conditions but have a* uniform spectrum *(i.e., all eigenvalues have similar size). This problem is addressed by showing that the spurious local maximizers can be filtered under fairly mild conditions based on their objective value associated with the non-linear program. For more details, we refer to the respective sections.*

Computing local maximizers (Sec. 2.3.4 and Section 2.5). For both characterizations mentioned above, the computation of the local maximizers of the non-linear programs (2.2) and (2.3) is solved via a variant of projected gradient ascent. For the objective (2.2) we rely on the iteration

$$M_{j+1} = F_\gamma(M_j), \quad \text{where} \quad F_\gamma(X) = \frac{P_{\widehat{\mathcal{W}}}(X + \gamma u_1(X) \otimes u_1(X))}{\left\| P_{\widehat{\mathcal{W}}}(X + \gamma u_1(X) \otimes u_1(X)) \right\|_F}, \tag{2.4}$$

proposed in [54, 52] where γ is a step-size parameter that is up to the algorithms designer and the procedure has been summarized in Algorithm 2.1. This iteration is composed of a gradient ascent step that exalts the first eigenvalue of a matrix by adding $\gamma u_1(M_j) \otimes u_1(M_j)$ and a projection back onto $\widehat{\mathcal{W}}$ intersected with the Frobenius unit-sphere. Here, $u_1(M_j)$ denotes the first eigenvectors of the j-th iterate M_j. We prove that this sequence produces a sequence of matrices with increasing spectral norms. For more details, we refer to Section 2.3.4.

The local maximizers of the objective (2.3) are computed via a projected gradient ascent algorithm that iterates

$$u_{j+1} = P_{\mathbb{S}^{D-1}}(u_j + 2\gamma P_{\widehat{\mathcal{W}}}(u_j \otimes u_j) u_j), \tag{2.5}$$

with fixed step-size $\gamma > 0$. The iteration in (2.5) was introduced in [77] as *subspace power method* (SPM). It has been shown in [77] that the SPM iteration converges to local maximizers of (2.3) under fairly mild conditions for *any symmetric matrix space* $\widehat{\mathcal{W}}$. Hence, the results in [77] do apply to the perturbed objective. We refer to the respective discussion in Section 2.5 for more details. Before we conclude the summary of this chapter and move to the technical problem setting, let us make the following remark regarding the computational complexity of the *recovery of all local maximizers*.

Remark 2.2. *Both iterations introduced above can only return up to one spanning matrix at a time. Due to the nature of gradient-based methods, the initialization of the iteration will strongly influence which near-rank-one matrix is recovered from the perturbed space. Both approaches start their iteration from a random initialization on the respective unit-sphere. Algorithm 2.1, which iterates* (2.4)*, selects randomly a matrix* M_0 *in* $\widehat{\mathcal{W}} \cap \mathbb{S}^{D\times D-1}$*, whereas Algorithm 2.2 picks the starting point* u_0 *uniformly at random from the unit-sphere* $\mathbb{S}^{D-1}$*. We then seek to recover* all local maximizers *by repeatedly sampling independent starting points. As long as the retrieval of every local maximizer is equally likely, the average number of repetitions needed to recover all local maximizers can be derived from the analysis of a classical problem of combinatorial probability. Namely, the* coupon collection problem*, according to which we expect an average number of* $\Theta(m \log m)$ *repetitions until we have recovered all local maximizers (see also [53])*

2.1.1 Problem setting

Let us now introduce some technical aspects and assumptions that play a fundamental role throughout the following section.

Scaling ambiguity of the spanning elements. Consider a symmetric matrix space spanned by rank-one matrices $\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$, where $w_1, \dots w_m \in \mathbb{R}^D$. Without any additional information, we can only recover the basis matrices $w_k \otimes w_k$ up to non-negative scaling factors since for any $\sigma > 0, k \in [m]$ the matrix $\sigma_k w_k \otimes w_k \in \mathcal{W}$ is a positive definite rank-one matrix. Therefore, moving forward, we will constrain the spanning matrices on the Frobenius unit-sphere, which is equivalent to $w_1, \dots, w_m \in \mathbb{S}^{D-1}$.

Linear independence and rank-one ambiguity. According to the recovery problem we stated at the beginning, we consider a symmetric matrix space $\mathcal{W}$, and our goal is the recovery of a *specific* set of rank-one matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ such that

$$\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}.$$

The unique characteristic of the spanning elements is the fact that they are rank-one. Differently put, if there exists a matrix $M \in \mathcal{W}$ such that $\operatorname{rank}(M) = 1$ and $M \notin \operatorname{span}\{w_k \otimes w_k\}$ for any $k \in [m]$, the unique identification of the matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ from $\mathcal{W}$ based on the rank criteria becomes ill-posed. This is related to another aspect of the rank-one basis recovery problem that we have neglected until now, which is the fact that the matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ need to form a basis. Again, without such an assumption, the unique recovery of all spanning elements becomes ill-posed or requires additional information. Assume, for instance, that $w_1 \otimes w_1$ is redundant due to linear dependencies, and we are given the space

$$\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\} = \operatorname{span}\{w_2 \otimes w_2, \dots, w_m \otimes w_m\}.$$

Hence, without linear independence, the set of rank-one spanning elements becomes ambiguous. Note that if the matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ are linearly independent, then we can infer

the number of spanning elements m directly from the matrix space since $\dim(\mathcal{W}) = m$. This implies that there is a *theoretical limit* on the maximal number of spanning elements that can be recovered, due to the fact that $\mathcal{W} \subsetneq \mathrm{Sym}(\mathbb{R}^{D\times D})$. If we were to assume $m = D(D+1)/2$, then by the basis property we get that

$$\dim(\mathcal{W}) = |\mathcal{B}| = D(D+1)/2 = \dim(\mathrm{Sym}(D^{D\times D-1}))$$

implying that $\mathcal{W} = \mathrm{Sym}(\mathbb{R}^{D\times D})$. In this scenario, $w \otimes w \in \mathcal{W}$ for all $w \in \mathbb{R}^D$, and there remains no hope to uniquely identify a rank-one basis of size m. Therefore, the condition $m < D(D+1)/2$ is necessary. Once we approach the regimes of m where $\mathcal{W}$ starts to resemble the space of all symmetric matrices, the rank-one characteristic becomes less specific to the basis elements. In Section 2.1.2, we will introduce assumptions on the incoherence of the vectors $w_1, \ldots, w_m$ from which we can derive linear independence of the spanning matrices. This is followed by Section 2.2, where we show sufficient conditions under which near-rank-one matrices within the perturbed space $\widehat{\mathcal{W}} \approx \mathcal{W}$ are close to the original spanning elements.

Structure of the perturbed space. The last point we want to address is what kind of perturbations we can allow while still being able to develop an analysis of the approximate recovery of all the rank-one spanning elements from the perturbed space $\widehat{\mathcal{W}} \approx \mathcal{W}$. To guarantee that the approximating space $\widehat{\mathcal{W}} \in \mathrm{Sym}(\mathbb{R}^{D\times D})$ exhibits a similar structure as $\mathcal{W}$, we will typically assume that

$$\delta := \left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F\to F} = \sup_{\substack{Z\in\mathbb{R}^{D\times D}\\ \|Z\|_F=1}} \left\|P_{\mathcal{W}}Z - P_{\widehat{\mathcal{W}}}Z\right\|_F < 1, \tag{2.6}$$

where $P_{\mathcal{W}}$, $P_{\widehat{\mathcal{W}}}$ are the orthogonal projections onto $\mathcal{W}, \widehat{\mathcal{W}}$, respectively. Without an assumption like $\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F} < 1$, we would run into degenerate cases like the following: Assume $\widehat{\mathcal{W}} = \mathcal{W} \cap \mathrm{span}\{w_m \otimes w_m\}^\perp$, then $\delta = \left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F\to F} = 1$, but $w_m \otimes w_m$ can not be recovered from $\widehat{\mathcal{W}}$. Additionally,(2.6) has several useful implications that allow us to map elements between $\mathcal{W}, \widehat{\mathcal{W}}$ using their orthogonal projections.

Lemma 2.1. *Let $\mathcal{W}, \widehat{\mathcal{W}} \subset \mathbb{R}^{D\times D}$ be matrix subspaces with corresponding orthogonal projections $P_{\mathcal{W}}$, $P_{\widehat{\mathcal{W}}}$ and assume that $\delta := \|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F} < 1$. Then the following holds:*

(i) Both matrix spaces have the same dimension, i.e., we have $\dim(\mathcal{W}) = \dim(\widehat{\mathcal{W}})$.

(ii) For any matrix $W \in \mathcal{W}$ we have

$$(1-\delta)^{-1}\left\|P_{\widehat{\mathcal{W}}}(W)\right\|_F \le \left\|P_{\widehat{\mathcal{W}}}(W)\right\|_F \le \|W\|_F,$$

and the same statement holds with $\mathcal{W}, \widehat{\mathcal{W}}$ reversed.

(iii) The constrained orthogonal projections $P_{\mathcal{W}} : \widehat{\mathcal{W}} \to \mathcal{W}, P_{\widehat{\mathcal{W}}} : \mathcal{W} \to \widehat{\mathcal{W}}$ are bijective.

Proof. For *(i)*, we start by assuming the contrary. If $\dim(\mathcal{W}) > \dim(\widehat{\mathcal{W}})$, then $\dim(\mathcal{W}) > 0$ and there exists an element $X \in \mathcal{W}$ s.t. $X - P_{\widehat{\mathcal{W}}}X \neq 0$. Denote $Y = X - P_{\widehat{\mathcal{W}}}X$, then $Y \in \mathcal{W} \cap \widehat{\mathcal{W}}^\perp$ and we have

$$\sup_{\substack{Z\in\mathbb{R}^{D\times D}\\ \|Z\|_F=1}} \left\|P_{\mathcal{W}}Z - P_{\widehat{\mathcal{W}}}Z\right\| \ge \left\|P_{\mathcal{W}}\frac{Y}{\|Y\|_F} - P_{\widehat{\mathcal{W}}}\frac{Y}{\|Y\|_F}\right\| = \left\|\frac{Y}{\|Y\|_F}\right\|_F = 1$$

which contradicts our initial assumption. The same argument can be repeated if $\dim(\mathcal{W}) < \dim(\widehat{\mathcal{W}})$, hence $\dim(\mathcal{W}) = \dim(\widehat{\mathcal{W}})$. The left inequality in *(ii)* is trivial, and the right inequality follows by

$$\begin{aligned}\left\|P_{\widehat{\mathcal{W}}}(W)\right\|_F &= \left\|W + P_{\widehat{\mathcal{W}}}(W) - P_{\mathcal{W}}(W)\right\|_F \geq \left|\|W\|_F - \left\|P_{\widehat{\mathcal{W}}}(W) - P_{\mathcal{W}}(W)\right\|_F\right| \\ &\geq \Big(1 - \sup_{\substack{Z\in\mathbb{R}^{D\times D}\\ \|Z\|_F=1}} \left\|P_{\mathcal{W}}Z - P_{\widehat{\mathcal{W}}}Z\right\|\Big) \|W\|_F \geq (1-\delta)\|W\|_F .\end{aligned}$$

The same statement holds with $\mathcal{W}, \widehat{\mathcal{W}}$ reversed by relabeling. For *(iii)*, denote by ψ the constrained projection $P_{\widehat{\mathcal{W}}} : \mathcal{W} \to \widehat{\mathcal{W}}$. Since ψ is an orthogonal projection, it is a linear map. Since $\mathcal{W}, \widehat{\mathcal{W}}$ are both finite-dimensional we have

$$\dim \mathcal{W} = \dim \operatorname{range} \psi + \dim \ker \psi.$$

By *(ii)* we have $\psi(W) \geq \frac{1}{1-\delta}\|W\|_F$ implying $\ker \psi = \{0\}$ and therefore the injectivity of ψ. Plugging this into the relation above, we have $\dim \operatorname{range} \psi = \dim \mathcal{W} = \dim \widehat{\mathcal{W}}$, where we used *(i)* for the last identity. This implies surjectivity. Again, the same proof works for $\psi = P_{\mathcal{W}} : \widehat{\mathcal{W}} \to \mathcal{W}$ by reversing the role of both spaces throughout the arguments. □

Lemma 2.1 establishes the important observation that we can identify each element in $\mathcal{W}$ with a unique element in $\widehat{\mathcal{W}}$ by the bijection $P_{\widehat{\mathcal{W}}} : \mathcal{W} \to \widehat{\mathcal{W}}$, as long as $\delta < 1$. In particular, if we pair this statement with the assumption that $w_1 \otimes w_1, \ldots, w_m \otimes w_m$ form a basis of $\mathcal{W}$, then $P_{\widehat{\mathcal{W}}}$ gives rise to a basis of $\widehat{\mathcal{W}}$ that is derived from the original spanning elements: Denote

$$W_k = w_k \otimes w_k, \quad \text{and} \quad \widehat{W}_k = P_{\widehat{\mathcal{W}}}(W_k) \quad \text{for all} \quad k \in [m], \tag{2.7}$$

then for $\delta < 1$, the matrices $\widehat{W}_k$ form a basis of $\widehat{\mathcal{W}}$ which follows from the fact that $P_{\widehat{\mathcal{W}}}$ constrained onto $\mathcal{W}$ is a bijection by Lemma 2.1. A common construction that will find many applications throughout this chapter is the representation of a matrix $M \in \widehat{\mathcal{W}}$ in terms of the basis $\widehat{W}_1, \ldots, \widehat{W}_m$ derived from the projections of the linearly independent rank-one spanning elements.

2.1.2 Deterministic frame bounds: Measuring incoherence in the linear regime

The last section hinted at the fact that the spanning elements $w_1 \otimes w_1, \ldots, w_m \otimes w_m$ need to form a basis to allow their unique identification from their span. One way this could be guaranteed is to assume the vectors $w_1, \ldots, w_m$ are linearly independent. This follows, for instance, by Lemma 2.22 and the following discussion. However, requiring that vectors $w_1, \ldots, w_m$ form a basis is too restrictive as such a requirement imposes $m \leq D$. In this small interlude, we introduce sufficient conditions that imply linear independence of the outer products $w_1 \otimes w_1, \ldots, w_m \otimes w_m$ that apply to the overcomplete regime $m > D$. The following theory characterizes the vectors $w_1, \ldots, w_m$ based on their *incoherence*. We can define the (mutual) incoherence of a set of vectors $w_1, \ldots, w_m \in \mathbb{S}^{D-1}$ as based on their maximal cross-correlation

$$\epsilon = \max_{k \neq \ell} |\langle w_k, w_\ell \rangle|,$$

and we say the vectors are incoherent if ϵ is sufficiently small. For $m \leq D$, all vectors could be orthogonal, in which case we have $\epsilon = 0$. For the case $m > D$, a lower bound on ϵ is provided

by the so-called Welch bounds [131], which state that

$$\epsilon \geq \sqrt{\frac{m-D}{D(m-1)}}. \tag{2.8}$$

This implies a lower bound $\epsilon = \mathcal{O}(D^{-1/2})$ on the correlation of $m \geq D$ unit vectors. It is interesting to compare this bound to the correlation of generic vectors in high dimensions. Let us, for instance, consider random vectors X, Y drawn uniformly from the unit sphere $X, Y \sim_{\text{i.i.d}} \text{Unif}(\mathbb{S}^{D-1})$. It is well known that high-dimensional isotropic random vectors can be regarded as nearly orthogonal (cf. [127, Remark 3.2.5]), i.e., we have

$$|\langle X, Y\rangle| \sim D^{-1/2},$$

with high probability. In Section 2.6.1, more precisely Proposition 2.2, we will see that this also holds uniformly over a set of $m = o(D^2)$ isotropic random vectors up to logarithmic factors in m. This shows that we can expect a set of generic vectors in high dimensions to exhibit a significant amount of incoherence and that isotropic random distributions are a suitable model for well-separated vectors in high dimensions.

Remark 2.3. *The remaining theory within this section does only apply to the mildly overcomplete regime where $m < 2D$ (cf. Lemma 2.3 below) since this will be our setting in Section 2.2 - 2.4. We will resume the discussion of more general concepts of incoherence in Section 2.6, where we will discuss the scenario $D < m < D^2$ and extend our concepts to higher-order tensors.*

Let us mention that any type of incoherence of vectors $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ trivially extends to their outer products since

$$\langle w_k^{\otimes n}, w_\ell^{\otimes n}\rangle = \langle w_k, w_\ell\rangle^n, \quad \text{for all } n \in \mathbb{N}, k, \ell \in [m].$$

Hence, incoherent vectors will lead to incoherent spanning matrices. Notably, separation between the matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ is crucial. If two spanning elements in $\mathcal{W}$ were allowed to become arbitrarily close, then even small perturbations could make these elements indistinguishable in $\widehat{\mathcal{W}}$.

Remark 2.4. *Notably, for the special case $m = D$, it has recently been shown that a recovery of the spanning elements is in principle possible as long as $\{w_1, \dots, w_m\}$ (and therefore also $\{w_k \otimes w_k | \, k \in [m]\}$) are linearly independent [54],* without relying on strong incoherence. *This is achieved by an orthogonalization procedure, called whitening (cf. Section 1.5 and [54, Section 8]), which allows the authors to artificially put the spanning matrices in a well-separated position. The whitening procedure requires high accuracy of the approximation $\widehat{\mathcal{W}} \approx \mathcal{W}$. To the best of our knowledge, this method has not been extended to the case $m > D$. Even if the whitening procedure could be extended to the overcomplete regime $m > D$, gaining a highly accurate approximation of $\mathcal{W}$ might not be feasible when the approximation error is not controlled as part of the recovery algorithm. We will encounter such a case in Chapter 4, where $\|P_{\widehat{\mathcal{W}}} - P_{\mathcal{W}}\|_{F\to F}$ is a* constant error *that cannot be made arbitrarily small.*

To make the incoherence assumptions more precise, we will borrow an elementary concept from *frame theory* (see [30]) that leads to a generalization of orthonormal bases.

Definition 2.1. *A set of vectors $w_1, \dots, w_m \in \mathbb{R}^D$ is called a frame if there exist constants $A, B > 0$, such that*

$$A\|x\|_2^2 \leq \sum_{k=1}^{m} \langle x, w_k\rangle^2 \leq B\|x\|^2 \quad \textit{for all } x \in \mathbb{R}^D. \tag{2.9}$$

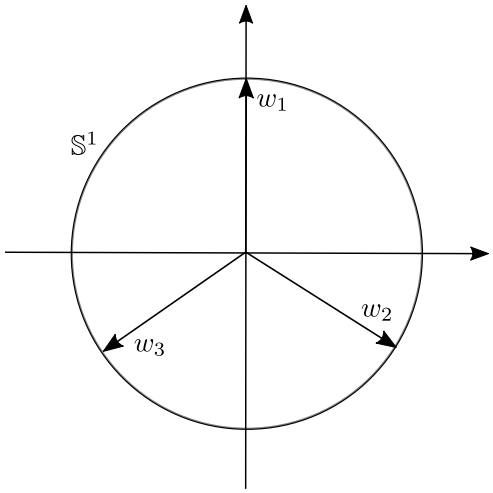

Figure 2.1: The Mercedes-Benz frame. A classical example for a finite normalized tight frame with three elements in $\mathbb{R}^2$

The constants A, B are typically referred to as *frame bounds*. It is clear that the frame condition in (2.9) is fulfilled for $A = B = 1$ when $\{w_1, \dots, w_m\}$ forms an orthonormal basis since then

$$\sum_{k=1}^{m} \langle x, w_k \rangle^2 = \|x\|_2^2 \text{ for all } x \in \mathbb{R}^D.$$

The latter inequality is commonly known as Parseval's identity. In the literature, frames described in Definition 2.1 are also referred to as finite frames [30, 17]. Frames for which the frame bounds coincide, i.e., $A = B$ are also called *tight frames*, and if the vectors $w_1, \dots, w_m$ all have unit norm, then we refer to them *normalized frames*. A finite normalized tight frame is a system of vectors $\{w_1, \dots, w_m\} \subset \mathbb{S}^{D-1}$ which is in the optimal position to achieve the lowest frame constants (i.e., maximal incoherence cf. passages above) that is attainable for m unit vectors in $\mathbb{R}^D$ (see [17, 29]). A popular example of a tight frame is the so-called Mercedes-Benz frame (see Figure 2.1), which consists of three unit vectors in $\mathbb{R}^2$ at maximal separation.

In the following sections, we will primarily rely on the following adaption of the frame conditions, which unites both frame bounds into one constant: Consider a set of unit vectors $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ and assume that there exists a constant $\nu \geq 0$ such that

$$(1 - \nu) \|x\|_2^2 \leq \sum_{k=1}^{m} \langle w_k, x \rangle^2 \leq (1 + \nu) \|x\|_2^2 \quad \text{for all } x \in \mathbb{R}^D. \tag{2.10}$$

If (2.10) holds for some $\nu \in [0, 1)$, then the vectors form a frame according to Definition 2.1. One can interpret the constant ν in 2.10 as the degree to which $\{w_1, \dots, w_m\}$ deviates from an orthonormal basis. We often assume that the vectors fulfill this condition for some $\nu < 1$, which implies the necessary linear independence of the matrices $\{w_k \otimes w_k \,|\, k \in [m]\}$.

Lemma 2.2 (cf. [52, Lemma 29])**.** *Let $\{w_1, \dots, w_m\} \subset \mathbb{R}^D$ have unit norm and satisfy*

$$\sum_{k=1}^{m} \langle w_k, w_\ell \rangle^2 \leq 1 + \nu,$$

for all $\ell = 1, \dots, m$. If $0 \leq \nu < 1$, then the system $\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$ is linearly independent.

Proof. If we assume on the contrary that $w_1 \otimes w_1, \dots, w_m \otimes w_m$ are not linearly independent, then we can find coefficients $\sigma \neq 0 \in \mathbb{R}^m$ such that

$$0 = \sum_{k=1}^{m} \sigma_k w_k \otimes w_k \Leftrightarrow \sum_{k=1}^{m} \sigma_k \langle x, w_k \rangle^2 = 0 \quad \text{for all } x \in \mathbb{R}^D.$$

Let us denote by k^* the first index such that $\|\sigma\|_\infty = |\sigma_{k^*}|$. We can assume $\sigma_{k^*} > 0$ without loss of generality since we could simply flip the signs of the coefficients otherwise. Then we have

$$0 = \sum_{k=1}^{m} \sigma_k \langle w_k, w_{k^*} \rangle^2 = \sigma_{k^*} + \sum_{k \neq k^*} \sigma_k \langle w_k, w_{k^*} \rangle^2 \geq \sigma_{k^*} + \min_k \sigma_k \sum_{k \neq k^*} \langle w_k, w_{k^*} \rangle^2,$$

where the second identity follows from $\|w_{k^*}\| = 1$. We can exclude the case $\min_k \sigma_k \geq 0$, which immediately yields a contradiction. Now if $\min_k \sigma_k < 0$, then

$$0 \geq \sigma_{k^*} + \min_k \sigma_k \sum_{k \neq k^*} \langle w_k, w_{k^*} \rangle^2 \geq \sigma_{k^*} + \min_k \sigma_k \nu \|w_{k^*}\|^2 = \sigma_{k^*} + \min_k \sigma_k \nu.$$

By division through ν, and then subtracting $\min_k \sigma_k$, we obtain $|\min_k \sigma_k| \geq \sigma_{k^*} \nu^{-1}$. According to our outgoing assumption we have $\nu < 1$, which yields a contradiction since the latter would imply that $\|\sigma\|_\infty \geq |\min_k \sigma_k| > \sigma_{k^*} = \|\sigma\|_\infty$. □

Thus, whenever we construct $\mathcal{W}$ from vectors that fulfill (2.10) with $\nu < 1$, it implicitly follows that $\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$ forms a basis. This raises another question: To what degree is the constant ν in condition (2.10) determined by the dimension D and cardinality m of the system $\{w_1, \dots, w_m\}$.

Lemma 2.3 (cf. [17, Theorem 3.1])**.** *Consider a set of unit vectors $\{w_1, \dots, w_m\} \subset \mathbb{S}^{D-1}$ that fulfill condition* (2.10)*. Then*

$$1 - \nu \leq \frac{m}{D} \leq 1 + \nu. \tag{2.11}$$

Proof. Denote by e_i the i-th canonical basis vector in $\mathbb{R}^D$, then we have

$$D = \sum_{i=1}^{D} \|e_i\|_2^2 \leq (1+\nu) \sum_{i=1}^{D} \sum_{k=1}^{m} \langle e_i, w_k \rangle^2 = (1+\nu) \sum_{k=1}^{m} \|w_k\|_2^2 = (1+\nu)m,$$

which confirms the right bound in (2.11). The left bound follows from an identical chain of arguments relying on $\|e_i\|_2^2 \geq (1-\nu) \sum_{k=1}^{m} \langle e_i, w_k \rangle^2$. □

According to the last statement, the frame condition (2.10) requires $m \leq (1+\nu)D$ and, therefore at best, allows for a linear dependency of m on D. Let us note that suitable frames which adhere to the equality ($\nu = \frac{m}{D} - 1$) are so-called finite normalized tight frames.

2.2 Near rank-one matrices approximate spanning elements

As was already mentioned in the introduction of this chapter and Section 2.1.1, the natural selection criteria for the matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m \in \mathrm{Sym}(\mathbb{R}^{D \times D}) \cap \mathbb{S}^{D \times D - 1}$ from their span $\mathcal{W}$ is their rank-one property. We begin this section by discussing conditions that guarantee the spanning elements are the only matrices within $\mathcal{W}$ with rank-one. This is followed by a key result, namely Theorem 2.1, that shows under which conditions near-rank-one matrices in a perturbed space $\widehat{\mathcal{W}} \approx \mathcal{W}$ are close to the rank-one one basis matrices.
The following statement shows that up to a certain number of spanning elements, a characteristic of the linear independence between the matrices $\{w_k \otimes w_k | k \in [m]\}$ will be sufficient to guarantee rank-one matrices as long as the number of spanning elements is not too large.

Lemma 2.4 ([52, Lemma 30]). *Consider* $\mathcal{W} = \text{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$ *with* $m < 2D-1$ *and unit vectors* $w_1, \dots, w_m \in \mathbb{S}^{D-1}$. *Furthermore, assume that any subset of* $\lceil m/2 \rceil + 1$ *vectors* $\{w_{k_j} : k \in [\lceil m/2 \rceil + 1]\}$ *is linearly independent. Then, for any* $X \in \mathcal{W} \cap \mathbb{S}$ *with* $\text{rank}(X) = 1$, *there exists* $k^* \in [m]$ *such that* $X = \pm w_{k^*} \otimes w_{k^*}$.

Proof. Take any $X = \sum_{k=1}^m \alpha_k w_k \otimes w_k \in \mathcal{W} \cap \mathbb{S}$, and denote the set of non-zero indices by $\mathcal{I} = \{k \in [m] : \alpha_k \neq 0\}$. We can exclude the trivial cases $|\mathcal{I}| = 0, |\mathcal{I}| = 1$. Now, if $1 < |\mathcal{I}| \leq \lceil m/2 \rceil + 1$, the vectors $\{w_k | \, k \in \mathcal{I}\}$ are linearly independent, and thus $\text{rank}(X) = |\mathcal{I}| > 1$. In the remaining case, where $|\mathcal{I}| > \lceil m/2 \rceil + 1$, we can split $\mathcal{I} = \mathcal{I}_1 \cup \mathcal{I}_2$ with $|\mathcal{I}_1| = \lceil m/2 \rceil + 1$ and $|\mathcal{I}_2| \leq m - \lceil m/2 \rceil - 1 \leq m/2 - 1$. If we accordingly split $X = X_1 + X_2$ with $X_j := \sum_{k \in \mathcal{I}_j} \alpha_k w_k \otimes w_k$, the assumption implies $\text{rank}(X_1) = \lceil m/2 \rceil + 1$ and $\text{rank}(X_2) \leq m/2 - 1$. Since furthermore $\text{rank}(X) \geq \text{rank}(X_1) - \text{rank}(X_2)$, it follows that $\text{rank}(X) \geq \lceil m/2 \rceil + 1 - (m/2 - 1) \geq 2$. □

In the setting of the lemma above, the elements in $\{\pm w_k \otimes w_k | k \in [m]\}$ are the only rank-one matrices in $\mathcal{W}$. However, the conditions of Lemma 2.4 are not directly related to the incoherence of the vectors $w_1, \dots, w_m$. The next statement addresses this issue and proves an identical statement implied by the incoherence assumption in (2.10).

Corollary 2.1 ([52, Corollary 31]). *Assume that* $m < 2D-1$ *and that* $\{w_k | \, k \in [m]\}$ *satisfies the upper frame bound* (2.10) *with* $\nu < \lceil \frac{m}{2} \rceil^{-1}$. *Let* $X \in \mathcal{W} \cap \mathbb{S}$ *of* $\text{rank}(X) = 1$, *then there exists* k^* *such that* $X = \pm w_{k^*} \otimes w_{k^*}$.

Proof. To apply Lemma 2.4, we establish a lower bound for the size of the smallest linearly dependent subset of $\{w_k : \ell \in [m]\}$, denoted commonly also by $\text{spark}(\{w_\ell : \ell \in [m]\})$, see [125]. Following [125], it is bounded from below by

$$\begin{aligned} \text{spark}(\{w_\ell : \ell \in [m]\}) &\geq \min\{k : \mu_1(k-1) \geq 1\}, \\ \text{where} \quad \mu_1(k-1) &:= \max_{\substack{\mathcal{I} \subset [m] \\ |\mathcal{I}| = k-1}} \max_{j \notin \mathcal{I}} \sum_{i \in \mathcal{I}} \left| \langle w_i, w_j \rangle \right|. \end{aligned}$$

By applying (2.10), we can bound

$$\begin{aligned} \mu_1(k-1) &= \max_{\substack{\mathcal{I} \subset [m] \\ |\mathcal{I}| = k-1}} \max_{j \notin \mathcal{I}} \sum_{i \in \mathcal{I}} \left| \langle w_i, w_j \rangle \right| \\ &\leq \sqrt{k-1} \max_{\substack{\mathcal{I} \subset [m] \\ |\mathcal{I}| = k-1}} \max_{j \notin \mathcal{I}} \sqrt{\sum_{i \in \mathcal{I}} \langle w_i, w_j \rangle^2} \leq \sqrt{(k-1)\nu}. \end{aligned}$$

Taking additionally into account $\nu < \lceil \frac{m}{2} \rceil^{-1}$, it follows that

$$\begin{aligned} \text{spark}(\{w_\ell : \ell \in [m]\}) &\geq \min\{k : \mu_1(k-1) \geq 1\} \\ &\geq \min\{k : \sqrt{(k-1)\nu} \geq 1\} \\ &= \min\left\{k : k \geq 1 + \frac{1}{\nu}\right\} > 1 + \left\lceil \frac{m}{2} \right\rceil. \end{aligned}$$

The result follows by applying Lemma 2.4. □

Unfortunately, requesting Equation (2.10) to hold with $\nu < \lceil \frac{m}{2} \rceil^{-1}$ is very restrictive and does not cover certain scenarios that are studied in the broader context of neural network reconstruction (cf. Chapter 3). Instead, we will rely on the following statement, which shows that if the space $\mathcal{W} = \text{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$ is constructed from incoherent vectors

$w_1, \dots, w_m \in \mathbb{S}^{D-1}$, then any near-rank-one matrix within a close approximation $\widehat{\mathcal{W}}$ of $\mathcal{W}$ will be close to one of the original spanning elements. This statement is crucial since we will later characterize local maximizers of the non-linear programs, which are used as selection criteria, into two categories. One of those categories is formed by matrices with a dominant leading eigenvalue, which falls into the setting of the following theorem.

Theorem 2.1 (cf. [50, Theorem 14])**.** *Let $\{w_1, \dots, w_m\} \subset \mathbb{S}^{D-1}$ satisfy frame condition* (2.10) *with $\nu < 1$. Consider the matrix space $\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$ and let $\widehat{\mathcal{W}} \subseteq Sym(\mathbb{R}^{D\times D})$ satisfy $\left\|P_{\widehat{\mathcal{W}}} - P_{\mathcal{W}}\right\|_{F\to F} \leq \delta < \frac{1}{2}$ and consider a basis $\{\widehat{W}_k := P_{\widehat{\mathcal{W}}}(w_k \otimes w_k) \,|\, k \in [m]\}$. For $M \in \widehat{\mathcal{W}}$, $\|M\|_F \leq 1$ consider the representation $M = \sum_{k=1}^m \sigma_k \widehat{W}_{\pi(k)} = \sum_{j=1}^D \lambda_j u_j \otimes u_j$, where (λ_j, u_j) are eigenpairs, with both σ_k's and λ_j's sorted in descending order and π is a permutation on $[m]$. If $\lambda_1 > \frac{\lambda_2 - 2\delta}{1-\nu}$, then*

$$\min_{s\in\{-1,1\}} \left\|u_1 - s w_{\pi(1)}\right\|_2 \leq \sqrt{2}\frac{\|\sigma_{2\dots m}\|_2 \sqrt{\nu} + 2\delta}{(1-\nu)\lambda_1 - \lambda_2 - 2\delta}, \tag{2.12}$$

where $\sigma_{2\dots m} := (0, \sigma_2, \dots, \sigma_m) \in \mathbb{R}^m$.

Proof of Theorem 2.1. Let $Z := \sum_{k=1}^m \sigma_k w_k \otimes w_k \in \mathcal{W}$ be the unique element in $\mathcal{W}$ such that $M = P_{\widehat{\mathcal{W}}}(Z)$. First notice that $\|Z\| \leq \left\|P_{\mathcal{W}}(Z) - P_{\widehat{\mathcal{W}}}(Z)\right\| + \left\|P_{\widehat{\mathcal{W}}}(Z)\right\| \leq \delta \|Z\| + 1$ implies $\|Z\| \leq (1-\delta)^{-1}$. Therefore, we have

$$\begin{aligned}\lambda_1 &= \langle M, u_1 \otimes u_1\rangle = \langle Z, u_1 \otimes u_1\rangle + \langle M - Z, u_1 \otimes u_1\rangle \\ &\leq \langle Z, u_1 \otimes u_1\rangle + \left\|P_{\widehat{\mathcal{W}}}(Z) - P_{\mathcal{W}}(Z)\right\| \leq \sum_{k=1}^m \sigma_k \langle w_k, u_1\rangle^2 + \frac{\delta}{1-\delta} \leq \sigma_1(1+\nu) + 2\delta,\end{aligned}$$

which implies $\sigma_1 \geq \frac{\lambda_1 - 2\delta}{1+\nu} \geq \lambda_1 - \nu\lambda_1 - 2\delta$. Define now $Q = \mathrm{Id}_D - u_1 \otimes u_1$. Choosing $s \in \{-1, 1\}$ such that $s\langle w_1, u_1\rangle \geq 0$ we can bound the squared left-hand side of (2.12) by

$$\|u_1 - s w_1\|_2^2 = 2\left(1 - s\langle w_1, u\rangle\right) \leq 2(1 - \langle w_1, u\rangle^2) = 2\|Q w_1\|_2^2 = 2\|Q(w_1 \otimes w_1)\|_F^2$$

Denote $W_1 := w_1 \otimes w_1$ and consider the auxiliary matrix $\sigma_1 W_1$. We can view W_1 as the orthogonal projection onto the space spanned by eigenvectors of $\sigma_1 W_1$ associated to eigenvalues in $(\infty, \sigma_1]$. Therefore, using the Davis-Kahan theorem in the form of [20, Theorem 7.3.1], we obtain

$$\|u_1 - s w_1\|_2 \leq \sqrt{2}\,\|Q W_1\|_F \leq \sqrt{2}\frac{\|(\sigma_1 W_1 - M)\, W_1\|_F}{\sigma_1 - \lambda_2}$$

To further bound the numerator, we first use the decomposition

$$\begin{aligned}\|(\sigma_1 W_1 - M) W_1\|_F &\leq \|(\sigma_1 W_1 - Z) W_1\|_F + \left\|(Z - P_{\widehat{\mathcal{W}}}(Z)) W_1\right\|_F \\ &\leq \|(\sigma_1 W_1 - Z) W_1\|_F + \left\|Z - P_{\widehat{\mathcal{W}}}(Z)\right\|_2 \|W_1\|_F \\ &\leq \|(\sigma_1 W_1 - Z) W_1\|_F + \frac{\delta}{1-\delta} \leq \|(\sigma_1 W_1 - Z) W_1\|_F + 2\delta,\end{aligned}$$

and then bound the first term by using the frame property (2.10)

$$\|(\sigma_1 W_1 - Z) W_1\|_F = \left\|\sum_{i=2}^K \sigma_k \langle w_k, w_1\rangle w_i \otimes w_1\right\|_F \leq \sum_{k=2}^m |\sigma_k|\,|\langle w_k, w_1\rangle| \leq \|\sigma_{2\dots m}\|_2 \sqrt{\nu}.$$

Combining the previous three estimates with $\sigma_1 \geq \lambda_1 - \nu\lambda_1 - 2\delta$, we obtain

$$\|s w_1 - u_1\|_2 \leq \sqrt{2}\frac{\|(\sigma_1 W_1 - M)\, W_1\|_F}{\sigma_1 - \lambda_2} \leq \sqrt{2}\frac{\|\sigma_{2\dots m}\|_2 \sqrt{\nu} + 2\delta}{(1-\nu)\lambda_1 - \lambda_2 - 2\delta}.$$

□

2.3 Selection of near-rank-one matrices based on the spectral norm

Consider again a set of rank-one matrices $\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$ constructed from a set of unit vectors $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ that give rise to a symmetric matrix space $\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$. Given an approximation $\widehat{\mathcal{W}}$ of the space $\mathcal{W}$, we now discuss the selection of matrices within $\widehat{\mathcal{W}}$ that approximate the rank-one spanning elements of $\mathcal{W}$. As mentioned at the beginning of this chapter, a retrieval of rank-one matrices from $\mathcal{W}$ can be described by the non-linear program

$$\operatorname{argmin} \operatorname{rank} M \text{ s.t. } M \in \mathcal{W} \cap \mathbb{S}. \tag{2.13}$$

However, this characterization does not extend to the perturbed matrix space $\widehat{\mathcal{W}}$ for obvious reasons. In Section 2.2, more precisely Theorem 2.1, we stated sufficient conditions that motivate a similar approach based on a relaxation of the rank maximization in (2.13) to the spectral norm. Therefore, in this section, we consider a natural relaxation of the rank-one criteria based on the non-convex program

$$\operatorname{argmax} \|M\| \text{ s.t. } M \in \widehat{\mathcal{W}}, \|M\|_F \leq 1, \tag{2.14}$$

which has been considered before in [54, 52]. This program selects those matrices whose spectrum is dominated by one eigenvalue. Clearly, for the unperturbed program associated with $\widehat{\mathcal{W}} = \mathcal{W}$, all elements within $\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$ are still global maximizers due to $\|M\| \leq \|M\|_F$ and $\|u \otimes u\| = 1$ for any $u \in \mathbb{S}^{D-1}$. Additionally, when considering this program in the general case $\widehat{\mathcal{W}} \approx \mathcal{W}$, we have

$$\left\|P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)\right\| \geq 1 - \left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{2\to 2}.$$

Hence, provided that $\mathcal{W}$ and $\widehat{\mathcal{W}}$ are sufficiently close, there exist global maximizers of (2.14) in $\widehat{\mathcal{W}}$ that have spectral norms close to one. Therefore, the perturbation analysis in the remaining part of this section focuses exclusively on local maximizers of (2.14), whose computation can be solved by a projected gradient ascent variant (see Section 2.3.4). From a mathematical standpoint, the prime challenge is the formulation of conditions derived from the first-order and second-order optimality conditions of (2.14) that guarantee that local maximizers have a dominant eigenvalue (and therefore fall into the setting of Theorem 2.1).

2.3.1 Characterization of local maximizers

We begin by categorizing the local maximizers of the perturbed objective (2.14) into two categories. More precisely, we show that for any local maximizer $M \in \widehat{\mathcal{W}}$ there exists absolute constants c, c' such that M satisfies $\|M\|^2 \geq 1 - c\delta - c'\nu$, or $\|M\|^2 \leq c\delta + c'\nu$, where $\delta = \|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F}$ and ν originates from (2.10). Hence, for δ, ν sufficiently small, M will either have one dominant eigenvalue or a uniform spectrum, where all eigenvalues have roughly the same magnitude. This result is proven in Theorem 2.2, which constitutes the main result of this section. Note that local maximizers with one dominant eigenvalue fall in the setting considered in Theorem 2.1, so these will naturally provide good approximations of the spanning matrices $\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$. The same can not be said for the other class of local maximizers. However, for reasonable small δ and ν, there is no overlap between those two classes of local maximizers, and therefore we can distinguish between good and spurious local maximizers by considering only those who lie in a certain superlevel set of $\|\cdot\|$. For the remainder of this section, and whenever the context allows, we denote $u_i := u_i(M)$ and $\lambda_i = \lambda_i(M)$, with

$i \in [D]$, for the eigenvectors and eigenvalues of the matrix M. Furthermore, we assume that eigenvalues are sorted in descending order such that

$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_D.$$

Moreover, the following two assumptions will sometimes be used to shorten technical results further.

(A3.1) We have $\lambda_1 = \|M\|$. Note that this is without loss of generality because $-M$ and M may be both local maximizers,

(A3.2) $\lambda_1 > \lambda_2$ (this is primarily a useful technical condition in order to use the second-order optimality condition introduced below).

The characterization of local maxima of (2.14) into two types depending on their spectra is summarized in the following result.

Theorem 2.2 ([52, Theorem 16])**.** *Consider $\mathcal{W}, \widehat{\mathcal{W}}$ as before with $\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F} \leq \delta$ and that $w_1, \ldots, w_m \in \mathbb{S}^{D-1}$ fulfill condition* (2.10) *with $\nu < 1$. Assume that M is a local maximizer of* (2.14) *satisfying **(A3.1)** and **(A3.2)**, and assume $38\delta + 13\nu < 1/4$. Then we have*

$$\lambda_1(M)^2 \geq 1 - 38\delta - 13\nu \text{ or } \lambda_1(M)^2 \leq 38\delta + 13\nu.$$

The proof has been deferred to the end of Section 2.3.3, which follows directly after a short discussion of the optimality conditions of (2.14).

2.3.2 Optimality Conditions

One fundamental aspect that we still need to explain is the characterization of the local maximizers by first-order and second-order optimality conditions of (2.14). The statement closely follows the ideas in [54, Section 3, Theorem 3.4], which were in turn influenced by [122, 123].

Theorem 2.3 ([52, Theorem 13])**.** *Let $M \in \widehat{\mathcal{W}} \cap \mathbb{S}$ and assume there exists a unique $i^* \in [D]$ satisfying $|\lambda_{i^*}(M)| = \|M\|$. If M is a local maximizer of* (2.14)*, then M fulfills the stationary (first-order) optimality condition*

$$u_{i^*}(M)^T X u_{i^*}(M) = \lambda_{i^*}(M) \langle X, M \rangle \tag{2.15}$$

for all $X \in \widehat{\mathcal{W}}$. A stationary point M (in the sense that M fulfills (2.15)*) is a local maximizer of* (2.14) *if and only if for all $X \in \widehat{\mathcal{W}}$*

$$2 \sum_{k \neq i^*} \frac{(u_{i^*}(M)^T X u_k(M))^2}{|\lambda_{i^*}(M) - \lambda_k(M)|} \leq |\lambda_{i^*}(M)| \, \|X - \langle X, M \rangle M\|_F^2 . \tag{2.16}$$

Proof. The statement requires minor modification of [54, Theorem 3.4] and the proof follows along analogous lines. For the reader's convenience, we give self-contained proof of the statement below, with some key computations borrowed from [54]. For simplicity, we drop the argument M in λ_i, u_i, and without loss of generality we assume $\lambda_{i^*} = \|M\|$, otherwise we consider $-M$. Following the analysis in [54], for $X \in \widehat{\mathcal{W}} \cap \mathbb{S}$ we can consider the function

$$f_X(\alpha) = \frac{\|M + \alpha X\|}{\|M + \alpha X\|_F},$$

because M is a local maximizer if and only if $\alpha = 0$ is a local maximizer of f_X for all $X \in \widehat{\mathcal{W}} \cap \mathbb{S}$. First, let us consider $X \in \widehat{\mathcal{W}} \cap \mathbb{S}$ with $X \perp M$. We note that the simplicity of λ_{i^*} implies that there exist analytic functions $\lambda_{i^*}(\alpha)$ and $u_{i^*}(\alpha)$ with $(M + \alpha X)u_{i^*}(\alpha) = \lambda_{i^*}(\alpha)u_{i^*}(\alpha)$ for all α in a neighborhood around 0 [89, 108]. Therefore we can use a Taylor expansion $\|M + \alpha X\| = \lambda_{i^*} + \lambda'_{i^*}(0)\alpha + \lambda''_{i^*}(0)\alpha^2/2 + \mathcal{O}(\alpha^3)$ and combine it with $\|M + \alpha X\|_F = \sqrt{1 + \alpha^2} = 1 - \alpha^2/2 + \mathcal{O}(\alpha^4)$ to get

$$f_X(\alpha) = \left(1 - \alpha^2/2\right)\left(\lambda_{i^*} + \lambda'_{i^*}(0)\alpha + \lambda''_{i^*}(0)\alpha^2/2\right) + \mathcal{O}(\alpha^3) \quad \text{as } \alpha \to 0.$$

Differentiating once we get $f'_X(0) = \lambda'_{i^*}(0)$, hence $\alpha = 0$ is a stationary point if and only if $\lambda'_{i^*}(0)$ vanishes. Following the computations in [54], we find that $\lambda'_{i^*}(0) = u_{i^*}(0)^T X u_{i^*}(0) = 0$, and thus (2.15) follows for any $X \perp M$. For general X, we split $X = \langle X, M\rangle M + X_\perp$, and get $u_{i^*}(0)^T X u_{i^*}(0) = \langle X, M\rangle\, u_{i^*}(0)^T M u_{i^*}(0) = \lambda_{i^*}(0)\langle X, M\rangle$.
For (2.16), we have to check additionally $f''_X(\alpha) \leq 0$. The second derivative of $f_X(\alpha)$ at zero is given by $f''_X(0) = \lambda''_{i^*}(0) - \lambda_{i^*}(0)$, hence the condition for attaining a local maximum is $\lambda''_{i^*}(0) \leq \lambda_{i^*}(0)$. Again, we can follow the computations in [54] to obtain

$$\lambda''_{i^*}(0) = 2\sum_{k \neq i^*} \frac{(u_{i^*}^T(0) X u_k(0))^2}{|\lambda_{i^*}(0) - \lambda_k(0)|},$$

and (2.16) follows immediately for any $X \perp M$, $\|X\|_F = 1$. For general X we decompose it into $X = \langle X, M\rangle M + X_\perp$. Since $u_{i^*}^T(0) M u_k(0) = 0$ for all $k \neq i^*$, we get

$$\begin{aligned} 2\sum_{k \neq i^*} \frac{(u_{i^*}^T(0)\left(\langle X, M\rangle M + X_\perp\right) u_k(0))^2}{|\lambda_{i^*}(0) - \lambda_k(0)|} &= 2\|X_\perp\|_F^2 \sum_{k \neq i^*} \frac{\left(u_{i^*}^T(0)\left(\frac{X_\perp}{\|X_\perp\|_F}\right) u_k(0)\right)^2}{|\lambda_{i^*}(0) - \lambda_k(0)|} \\ &\leq \lambda_{i^*}(0)\|X_\perp\|_F^2, \end{aligned}$$

and the result follows from $\|X_\perp\|_F = \|X - \langle X, M\rangle M\|_F$. □

Let us provide some context on these optimality conditions. The first-order optimality condition can be understood as follows: Any stationary point M of (2.14), with unique leading eigenvector λ_{i^*}, can be expressed as the image of the outer-product of the respective eigenvector under $P_{\widehat{\mathcal{W}}}$ such that $M = \lambda_{i^*}^{-1} P_{\widehat{\mathcal{W}}}(u_{i^*} \otimes u_{i^*})$. This follows by simple properties of the Frobenius inner product.

Lemma 2.5 ([52, Lemma 18]). *For $M \in \widehat{\mathcal{W}}$ and $c \neq 0$ we have*

$$v^T X v = c\langle X, M\rangle \quad \textit{for all } X \in \widehat{\mathcal{W}} \quad \textit{if and only if} \quad M = c^{-1} P_{\widehat{\mathcal{W}}}(v \otimes v).$$

Proof. Assume that $v^T X v = c\langle X, M\rangle$ for all X. We notice that the assumption is equivalent to $\langle X, v \otimes v - cM\rangle = 0$ for all $X \in \widehat{\mathcal{W}}$. Therefore, we have $P_{\widehat{\mathcal{W}}}(v \otimes v - cM) = 0$, and the result follows from $M \in \widehat{\mathcal{W}}$. In the case where $M = c^{-1} P_{\widehat{\mathcal{W}}}(v \otimes v)$, we compute $c\langle X, M\rangle = \langle X, P_{\widehat{\mathcal{W}}}(v \otimes v)\rangle = v^T X v$ since $X \in \widehat{\mathcal{W}}$. □

From the preceding result, one can directly derive a relationship between the projection of the leading eigenvector component and the associated eigenvalue for local maximizers as in Theorem 2.3. If M is a stationary point of (2.14) with unique leading eigenvalue λ_{i^*}, then, by Lemma 2.5, $\lambda_{i^*} M = P_{\widehat{\mathcal{W}}}(u_{i^*} \otimes u_{i^*})$ and therefore, by taking the Frobenius norm on both sides, we have $|\lambda_{i^*}| = \|P_{\widehat{\mathcal{W}}}(u_{i^*} \otimes u_{i^*})\|_F$. This characterization is unique to stationary points following the result below.

Lemma 2.6 ([52, Lemma 19]). *Let $X \in \widehat{\mathcal{W}} \cap \mathbb{S}$. We have $\left\|P_{\widehat{\mathcal{W}}}(u_j(X) \otimes u_j(X))\right\|_F \geq \left|\lambda_j(X)\right|$ with equality if and only if $X = \lambda_j(X)^{-1} P_{\widehat{\mathcal{W}}}(u_j(X) \otimes u_j(X))$.*

Proof. We drop the argument X for $\lambda_j(X)$ and $u_j(X)$ for simplicity. First, calculate

$$\left\|P_{\widehat{\mathcal{W}}}(u_j \otimes u_j)\right\|_F = \left\|P_{\widehat{\mathcal{W}}}(u_j \otimes u_j)\right\|_F \|X\|_F \geq \left|\left\langle P_{\widehat{\mathcal{W}}}(u_j \otimes u_j), X\right\rangle\right| = \left|\left\langle u_j \otimes u_j, X\right\rangle\right| = \left|\lambda_j\right|. \quad (2.17)$$

Moreover, we have equality if and only if $\left\|P_{\widehat{\mathcal{W}}}(u_j \otimes u_j)\right\|_F = \left|\lambda_j\right|$, hence (2.17) is a chain of equalities. Specifically,

$$\left\|P_{\widehat{\mathcal{W}}}(u_j \otimes u_j)\right\|_F \|X\|_F = \left|\left\langle P_{\widehat{\mathcal{W}}}(u_j \otimes u_j), X\right\rangle\right|,$$

which implies $X = cP_{\widehat{\mathcal{W}}}(u_j \otimes u_j)$ for some scalar c. Since $\|X\|_F = 1$, $c = \lambda_j^{-1}$ follows from

$$1 = \left\langle cP_{\widehat{\mathcal{W}}}(u_j \otimes u_j), X\right\rangle = c\left\langle u_j \otimes u_j, X\right\rangle = c\lambda_j.$$

□

In the form discussed above, the optimality conditions do not leverage the fact that we generally assume that $\widehat{\mathcal{W}}$ exhibits a specific structure, i.e., it is a close approximation of a space spanned by symmetric rank-one matrices. Incorporating this underlying structure (i.e., that $\widehat{\mathcal{W}}$ is close to a space spanned by incoherent rank-one matrices) yields the following additional properties derived from first- and second-order optimality conditions.

Lemma 2.7 ([52, Lemma 14-15]). *Consider $\mathcal{W}, \widehat{\mathcal{W}}$ as before with $\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F} \leq \delta$ and that $w_1, \ldots, w_m \in \mathrm{S}^{D-1}$ fulfill condition* (2.10) *with $\nu > 0$. If $\max\{\delta, \nu\} < 1/4$ and assuming that $M \in \widehat{\mathcal{W}}$ is a stationary point of* (2.14) *satisfying **(A3.1)-(A3.2)**, then*

$$\lambda_D \geq -2\delta\lambda_1^{-1} - 8\delta - 4\nu. \quad (2.18)$$

Additionally, if M is a local maximizer of (2.14)*, then for any $X \in \widehat{\mathcal{W}}$ with $\|X\|_F \leq 1$ we have*

$$\|Xu_1\|_2^2 \leq \frac{\lambda_1^2}{2}\left(1 + \langle X, M\rangle^2\right) + 5\delta + 2\nu. \quad (2.19)$$

The proof of Lemma 2.7 requires one additional auxiliary statement:

Lemma 2.8 ([52, Lemma 33]). *Let $\{w_k | k \in [m]\} \subset \mathbb{R}^D$ be a set of unit-norm vectors satisfying the incoherence condition* (2.10) *with $\nu < 1$ and $\delta = \|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F} < 1$. Then, for any $M = \sum_{k=1}^m \sigma_k P_{\mathcal{W}}(w_k \otimes w_k) = \sum_{k=1}^m \sigma_k \widehat{W}_k \in \widehat{\mathcal{W}} \cap \mathbb{S}$, we have*

$$\|\sigma\|_\infty \leq \frac{1}{(1-\delta)(1-\nu)}, \quad (2.20)$$

$$\left|\sigma_k \|\widehat{W}_k\|_F^2 - \langle \widehat{W}_k, M\rangle\right| \leq \frac{\delta}{1-\delta} + (\delta + \nu)\|\sigma\|_\infty \quad (2.21)$$

for all $k \in [m]$. Moreover, for any unit norm vector v, we have

$$\left|\|\widehat{W}_k v\|_2^2 - v^T \widehat{W}_k v\right| \leq 2\delta. \quad (2.22)$$

Proof of Lemma 2.8. Denote by $Z \in \mathcal{W}$ the unique element such that $M = P_{\widehat{\mathcal{W}}}(Z)$. We first note that $1 = \|M\|_F = \left\|P_{\widehat{\mathcal{W}}}(Z)\right\|_F \geq (1-\delta)\|Z\|_F$ implies $\|Z\|_F \leq (1-\delta)^{-1}$ by Lemma 2.1. For

(2.20), we assume without loss of generality $\max \sigma_k = \|\sigma\|_\infty$ (otherwise we perform the proof for $-M$), and denote $k^* = \operatorname{argmax}_k \sigma_k$. Then we have

$$(1-\delta)^{-1} \geq \|Z\|_F \geq \|Z\| \geq w_{k^*}^\top Z w_{k^*} = \sum_{k=1}^m \sigma_k \langle w_{k^*}, w_k \rangle^2 = \|\sigma\|_\infty + \sum_{k \neq k^*} \sigma_k \langle w_{k^*}, w_k \rangle^2$$
$$\geq \|\sigma\|_\infty \left(1 - \sum_{k \neq k^*} \langle w_{k^*}, w_k \rangle^2\right) \geq \|\sigma\|_\infty (1-\nu).$$

For (2.21) we first notice that

$$\begin{aligned}\left\langle \widehat{W}_{k^*}, M \right\rangle &= \sum_{k=1}^m \left\langle \widehat{W}_{k^*}, \sigma_k \widehat{W}_k \right\rangle \\ &= \sigma_{k^*} \left\| \widehat{W}_{k^*} \right\|_F^2 + \sum_{k \neq k^*} \left\langle \widehat{W}_{k^*}, \sigma_k \widehat{W}_k \right\rangle \\ &= \sigma_{k^*} \left\| \widehat{W}_{k^*} \right\|_F^2 + \sum_{k \neq k^*} \left\langle \widehat{W}_{k^*}, \sigma_k W_k \right\rangle \\ &= \sigma_{k^*} \left\| \widehat{W}_{k^*} \right\|_F^2 + \sum_{k \neq k^*} \left\langle \widehat{W}_{k^*} - W_{k^*}, \sigma_k W_k \right\rangle + \sum_{k \neq k^*} \sigma_k \langle W_{k^*}, W_k \rangle,\end{aligned}$$

and thus, it suffices to bound the last two terms. In the third line, we make use of the fact that $\langle \widehat{W}_{k^*}, \widehat{W}_k \rangle = \langle \widehat{W}_{k^*}, P_{\widehat{\mathcal{W}}}(W_k) + P_{\widehat{\mathcal{W}}^\perp}(W_k) \rangle = \langle \widehat{W}_{k^*}, W_k \rangle$. For the first term, we get

$$\left| \sum_{k \neq k^*} \left\langle \widehat{W}_{k^*} - W_{k^*}, \sigma_k W_k \right\rangle \right| \leq \delta \left\| \sum_{k \neq k^*} \sigma_k W_k \right\|_F = \delta \|Z - \sigma_{k^*} W_{k^*}\|_F \leq \frac{\delta}{1-\delta} + \|\sigma\|_\infty \delta,$$

by Cauchy-Schwarz, and for the second term we get

$$\left| \sum_{k \neq k^*} \sigma_k \langle W_{k^*}, W_k \rangle \right| \leq \|\sigma\|_\infty \sum_{k \neq k^*} \langle w_{k^*}, w_k \rangle^2 \leq \|\sigma\|_\infty \nu.$$

For (2.22), we first rewrite

$$\left| \left\| \widehat{W}_{k^*} u \right\|_2^2 - u^T \widehat{W}_{k^*} u \right| = \left| \left\langle \widehat{W}_{k^*}^2, u \otimes u \right\rangle - \left\langle \widehat{W}_{k^*}, u \otimes u \right\rangle \right| \leq \left\| \widehat{W}_{k^*}^2 - \widehat{W}_{k^*} \right\|.$$

Now denote $\Delta := \widehat{W}_{k^*} - W_{k^*}$. Since $W_{k^*}^2 = W_{k^*}$ we have

$$\begin{aligned}\left\| \widehat{W}_{k^*}^2 - \widehat{W}_{k^*} \right\| &= \left\| (\Delta + W_{k^*})^2 - W_{k^*} - \Delta \right\| = \left\| \Delta^2 + W_{k^*}\Delta + \Delta W_{k^*} - \Delta \right\| \\ &= \left\| \widehat{W}_{k^*}\Delta - \Delta(\mathrm{Id} - W_{k^*}) \right\| \leq \|\Delta\| \left(\left\| \widehat{W}_{k^*} \right\| + \|\mathrm{Id} - W_{k^*}\| \right) \leq 2\delta,\end{aligned}$$

since $\mathrm{Id} - W_{k^*}$ is a projection matrix onto $\operatorname{span}\{w_{k^*}\}^\perp$. □

Proof of Lemma 2.7. As before we have $M = \sum_{k=1}^m \sigma_k \widehat{W}_k$, and further denote by $Z = \sum_{k=1}^m \sigma_k W_k$ the unique element within $\mathcal{W}$ such that $M = P_{\widehat{\mathcal{W}}}(Z)$. We start by showing (2.18). Note that

$$\begin{aligned}\lambda_D &= \langle u_D \otimes u_D, M \rangle = \langle u_D \otimes u_D, Z \rangle + \langle u_D \otimes u_D, M - Z \rangle \\ &\geq \langle u_D \otimes u_D, Z \rangle - \|M - Z\| \geq \sum_{k=1}^m \sigma_k \langle w_k, u_D \rangle^2 - \frac{\delta}{1-\delta}.\end{aligned}$$

Denote $k^* = \operatorname{argmin}_{k\in[m]} \sigma_k$. Clearly, if $\sigma_{k^*} \geq 0$ then this implies $\lambda_D \geq -\frac{\delta}{1-\delta}$ confirming (2.18) for $\delta < 1/4$. Assuming $\sigma_{k^*} < 0$, we continue by applying the frame condition which yields

$$\lambda_D \geq \sigma_{k^*} \sum_{k=1}^{m} \langle w_k, u_D\rangle^2 - \frac{\delta}{1-\delta} \geq \sigma_{k^*}(1+\nu) - \frac{\delta}{1-\delta}.$$

To further bound $\sigma_{k^*}(1+\nu)$ from below, note that the first-order optimality condition (2.15) implies

$$\begin{aligned}\lambda_1 \left\langle \widehat{W}_{k^*}, M \right\rangle = \left\langle \widehat{W}_{k^*}, u_1 \otimes u_1 \right\rangle &= \langle W_{k^*}, u_1 \otimes u_1\rangle + \left\langle \widehat{W}_{k^*} - W_{k^*}, u_1 \otimes u_1 \right\rangle \\ &\geq \langle w_{k^*}, u_1\rangle^2 - \left\| \widehat{W}_{k^*} - W_{k^*} \right\| \geq -\delta.\end{aligned}$$

Applying the auxiliary bound in Lemma 2.8 in combination with $\|\widehat{W}_{k^*}\|_F \geq 1-\delta$, we obtain from the previous inequality

$$\begin{aligned}-\frac{\delta}{\lambda_1} \leq \left\langle \widehat{W}_{k^*}, M \right\rangle &\leq \sigma_{k^*} \left\| \widehat{W}_{k^*} \right\|_F^2 + \left| \sigma_{k^*} \left\| \widehat{W}_{k^*} \right\|_F^2 - \left\langle \widehat{W}_{k^*}, M \right\rangle \right| \\ &\leq \sigma_{k^*}(1-\delta)^2 + \frac{\delta}{1-\delta} + (\delta+\nu)\|\sigma\|_\infty .\end{aligned}$$

In summary, we established the lower bound on σ_{k^*} given by

$$\sigma_{j^*} \geq -\frac{\delta}{\lambda_1(1-\delta)^2} - \frac{\delta}{(1-\delta)^3} - \frac{(\delta+\nu)}{(1-\delta)^2}\|\sigma\|_\infty .$$

Hence, we can continue from the previously derived bound for λ_D with

$$\begin{aligned}\lambda_D &\geq \sigma_{k^*}(1+\nu) - \frac{\delta}{1-\delta} \\ &\geq -(1+\nu)\left(\frac{\delta}{\lambda_1(1-\delta)^2} + \frac{\delta}{(1-\delta)^3} + \frac{(\delta+\nu)}{(1-\delta)^2}\|\sigma\|_\infty\right) - \frac{\delta}{1-\delta}.\end{aligned}$$

Since $\delta, \nu < 1/4$ we obtain from (2.20) that $\|\sigma\|_\infty \leq 2$, and

$$\lambda_D \geq -2\delta\lambda_1^{-1} - 3\delta - 2(\delta+\nu)\|\sigma\|_\infty \geq -2\delta\lambda_1^{-1} - 8\delta - 4\nu. \tag{2.23}$$

Assume now M is a local maximizer. To show (2.19), note that the first-order optimality condition (2.15) implies

$$\begin{aligned}\frac{2}{\lambda_1 - \lambda_D}\left(\|Xu_1\|_2^2 - \lambda_1^2\langle X, M\rangle^2\right) &= \frac{2}{\lambda_1 - \lambda_D}\left(\|Xu_1\|_2^2 - \langle Xu_1, u_1\rangle^2\right) \\ &= \frac{2}{\lambda_1 - \lambda_D}\sum_{i=2}^{D}\langle Xu_1, u_i\rangle^2,\end{aligned}$$

where the second identity follows by the fact that $u_1, \ldots, u_D$ are orthonormal since M is symmetric. Furthermore, we can bound the right-hand side of the last equality by invoking the second-order optimality conditions (2.16) to get

$$\frac{2}{\lambda_1 - \lambda_D}\sum_{i=2}^{D}\langle Xu_1, u_i\rangle^2 \leq 2\sum_{i=2}^{D}\frac{(u_1^T X u_i)^2}{\lambda_1 - \lambda_i} \leq \lambda_1 \|X - \langle X, M\rangle M\|_F^2 .$$

Hence, we have

$$\frac{2}{\lambda_1 - \lambda_D}\left(\|Xu_1\|_2^2 - \lambda_1^2\langle X, M\rangle^2\right) \leq \lambda_1 \|X - \langle X, M\rangle M\|_F^2 ,$$

and by rearranging the inequality, we obtain

$$\begin{aligned}\|Xu_1\|_2^2 &\leq \frac{\lambda_1(\lambda_1-\lambda_D)}{2}\left(\|X\|_F^2 - \langle X, M\rangle^2\right) + \lambda_1^2\langle X, M\rangle^2 \\ &\leq \frac{\lambda_1(\lambda_1-\lambda_D)}{2} + \frac{\lambda_1(\lambda_1+\lambda_D)}{2}\langle X, M\rangle^2 \\ &= \lambda_1^2\frac{1+\langle X, M\rangle^2}{2} - \lambda_1\lambda_D\frac{1-\langle X, M\rangle^2}{2}.\end{aligned}$$

Using the lower bound for λ_D in (2.23), and $\lambda_1 \leq 1$, we get

$$\begin{aligned}\|Xu_1\|_2^2 &\leq \lambda_1^2\frac{1+\langle X, M\rangle^2}{2} + \lambda_1\left(2\delta\lambda_1^{-1} + 8\delta + 4\nu\right)\frac{1-\langle X, M\rangle^2}{2} \\ &\leq \lambda_1^2\frac{1+\langle X, M\rangle^2}{2} + (10\delta + 4\nu)\frac{1-\langle X, M\rangle^2}{2} \\ &\leq \lambda_1^2\frac{1+\langle X, M\rangle^2}{2} + 5\delta + 2\nu.\end{aligned}$$

□

Equipped with a better understanding of local maximizers of (2.14), we can now tackle the proof of Theorem 2.2.

2.3.3 Characterization of local maximizers based on their optimality conditions

This section gives the proof of Theorem 2.2 based on the optimality conditions we derived in Lemma 2.7. Before stating the main proof, we require one more auxiliary result related to the decomposition of matrices within $\widehat{\mathcal{W}}$ in terms of the basis $\{w_1 \otimes w_1, \ldots, w_m \otimes w_m\} \subset \mathcal{W}$.

Lemma 2.9 ([52, Lemma 12])**.** *Consider the setting of Theorem 2.2. For any* $M = \sum_{k=1}^m \sigma_k P_{\widehat{\mathcal{W}}}(w_k \otimes w_k) = \sum_{k=1}^m \sigma_k \widehat{W}_k \in \widehat{\mathcal{W}} \cap \mathbb{S}$ *with* $\lambda_1(M) \geq \delta/(1-\delta)$ *we have* $\max_k \sigma_k \geq 0$.

Proof. Assume on the contrary that $\max_k \sigma_k < 0$, and denote $Z = \sum_{k=1}^m \sigma_k W_k$ with $M = P_{\widehat{\mathcal{W}}}(Z)$. Then Z is negative semidefinite, since $v^T Z v = \sum_{k=1}^m \sigma_k \langle w_k, v\rangle^2$, and $\sigma_k < 0$ for all $k = 1, \ldots, m$. Moreover, we have $\|Z\|_F \leq (1-\delta)^{-1}$ by Lemma 2.1, and thus we get a contradiction since

$$\frac{\delta}{1-\delta} \leq \lambda_1(M) \leq \lambda_1(Z) + \|M - Z\|_F < \|M - Z\|_F \leq \frac{\delta}{1-\delta}.$$

□

We are now able to prove the section's main result by leveraging the optimality conditions of local maximizers.

Proof of Theorem 2.2. As before we have $M = \sum_{k=1}^m \sigma_k \widehat{W}_k$, and let $k^* = \operatorname{argmax}_k \sigma_k$. We first note that we can assume $\sigma_{k^*} \geq 0$ without loss of generality by Lemma 2.9, since there is nothing to show if $\lambda_1 \leq 2\delta$. By (2.19) for $X = \widehat{W}_{k^*}$ we get the inequality

$$2\|\widehat{W}_{k^*}u_1\|_2^2 \leq \lambda_1^2(1 + \langle \widehat{W}_{k^*}, M\rangle^2) + 10\delta + 4\nu.$$

We can further bound the left side using Lemma 2.8 followed by the first-order optimality condition to obtain

$$\|\widehat{W}_{k^*}u_1\|_2^2 \geq u_1^\top \widehat{W}_{k^*}u_1 - 2\delta = \lambda_1\langle \widehat{W}_{k^*}, M\rangle - 2\delta.$$

By rearranging both inequalities, we get

$$0 \leq \lambda_1^2 - 1 + \left(1 - \lambda_1 \langle \widehat{W}_{k^*}, M \rangle\right)^2 + 14\delta + 4\nu,$$

or equivalently

$$0 \leq \lambda_1^2 - 1 + \left(1 - \lambda_1 \sigma_{k^*} \|\widehat{W}_{k^*}\|_F^2 + \lambda_1 \left(\sigma_{k^*} \|\widehat{W}_{k^*}\|_F^2 - \langle \widehat{W}_{k^*}, M \rangle\right)\right)^2 + 14\delta + 4\nu. \tag{2.24}$$

We separate two cases. In the first case we have $\sigma_{k^*} > 1$, which implies $\langle \widehat{W}_{k^*}, M \rangle > 1 - 5\delta - 2\nu$ and thus $\langle W_{k^*}, M \rangle > 1 - 6\delta - 2\nu$ by Lemma 2.8 and $\max\{\delta, \nu\} < 1/4$. Since $\langle W_{k^*}, M \rangle = w_{k^*}^\top M w_{k^*}$, this implies $\lambda_1 > 1 - 6\delta - 2\nu$, i.e., the result is proven. We continue with the case $\sigma_{k^*} \leq 1$, which implies $\lambda_1 \sigma_{k^*} \|\widehat{W}_{k^*}\|_F^2 \leq 1$. Using Lemma 2.8 to bound $\sigma_{k^*} \|\widehat{W}_{k^*}\|_F^2 - \langle \widehat{W}_{k^*}, M \rangle$, $\lambda_1 < 1$ and $\|\widehat{W}_{k^*}\|_F^2 \geq 1 - 2\delta$, the inequality (2.24) implies

$$0 \leq \lambda_1^2 - 1 + (1 - \lambda_1 \sigma_{k^*} + 6\delta + 2\nu)^2 + 14\delta + 4\nu. \tag{2.25}$$

Furthermore, by following the computation we performed for the first part in the proof of Theorem 2.1, we get $\sigma_{k^*} \geq \lambda_1 - \nu - 2\delta$, and inserting it in (2.25) we obtain

$$0 \leq \lambda_1^2 - 1 + \left(1 - \lambda_1^2 + 8\delta + 3\nu\right)^2 + 14\delta + 4\nu, \text{ implying } 0 \leq \lambda_1^2 \left(\lambda_1^2 - 1\right) + 38\delta + 13\nu.$$

Provided that $38\delta + 13\nu < 1/4$, this quadratic inequality (in the unknown λ_1^2) has solutions $\lambda_1^2 \geq 1 - 38\delta - 13\nu$, or $\lambda_1^2 \leq 38\delta + 13\nu$. □

2.3.4 Computation of local maximizers via projected gradient ascent

Theorem 2.2 confirms, that for sufficiently small δ, ν, the local maximizers of (2.14) are either near-rank-one matrices, which, by Theorem 2.1, implies that they are close to one of the matrices $\{w_1 \otimes w_1, \ldots, w_m \otimes w_m\}$, or they satisfy $\lambda_1(M)^2 \leq 38\delta + 13\nu$. Based on these criteria, spurious local maximizers (local maximizers of the second kind) can be discarded based on their spectral norm. Hence, to find candidates within $\widehat{\mathcal{W}}$ which closely approximate the spanning elements of $\mathcal{W}$ it suffices to find all local maximizers of (2.14). What is left open is the actual computation of the local maximizers (or approximations thereof), which will be discussed in the following.

Inspired by [54], we propose tackling this problem by a simple iterative procedure closely related to projected gradient ascent and designed to create a sequence of matrices within $\widehat{\mathcal{W}}$ with increasing spectral norm. More precisely, we will study the following approach: Given a fixed $\gamma > 0$ and $M_0 \in \widehat{\mathcal{W}} \cap \mathbb{S}$, we can generate a sequence of matrices $M_j \in \widehat{\mathcal{W}} \cap \mathbb{S}, j \in \mathbb{N}$ based on the iterative application of the operator

$$M_{j+1} = F_\gamma(M_j), \quad \text{where} \quad F_\gamma(X) = \frac{P_{\widehat{\mathcal{W}}}(X + \gamma u_1(X) \otimes u_1(X))}{\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u_1(X) \otimes u_1(X))\right\|_F}, \tag{2.26}$$

see also Algorithm 2.1. Starting from a point $M_0 \in \widehat{\mathcal{W}}$, the next iterate is computed by *exalting* the first eigenvalue and then projecting back onto the intersection of the space $\widehat{\mathcal{W}}$ with the Frobenius unit-sphere. Exalting the first eigenvalue refers to the operation where the matrix $\gamma u_1(M_j) \otimes u_1(M_j)$ is added to the current iterate M_j, with $\gamma > 0$ being a hyperparameter chosen appropriately.

In the upcoming theoretical part, we prove the following result regarding the matrix iteration produced by Algorithm 2.1.

Algorithm 2.1: Projected gradient ascent

Input: $P_{\widehat{\mathcal{W}}}$ with arbitrary basis $\{b_i\}_{i=1,\dots,m}$, step-size $\gamma > 0$, number of iterations J

begin
  Sample $g \sim \mathcal{N}(0, \mathrm{Id}_m)$, and let $M_0 \leftarrow P_{\mathrm{S}}(\sum_{i=1}^m g_i b_i)$.
  If $\|M_0\|$ is not an eigenvalue, do $M_0 \leftarrow -M_0$.
  for $j = 1,\dots,J$ **do**
    $M_{j+1} \leftarrow P_{\mathrm{S}}(P_{\widehat{\mathcal{W}}}(M_j + \gamma u_1(M_j) \otimes u_1(M_j)))$.
  end
end

Output: $u_1(M_J)$

Theorem 2.4 ([52, Theorem 23]). *Let $\epsilon > 0$, $\gamma > 0$, $M_0 \in \widehat{\mathcal{W}} \cap \mathrm{S}$ with $\lambda(M_0) \geq 1/\sqrt{2} + \varepsilon$ and let $M_{j+1} := F_\gamma(M_j)$ as generated by Algorithm 2.1. Then $(M_{j+1})_{j\in\mathbb{N}}$ has a convergent subsequence, and any such subsequence converges to a fixed point of F_γ, respectively a stationary point of* (2.14).

The preceding statement shows that, in the setting above, there are convergent subsequences of the iteration (2.26) which converge to stationary points. Furthermore, it will be shown that the iteration induced by F_γ produces a sequence of matrices with increasing eigenvalues when started from matrices with a sufficiently large spectral norm. The proof of Theorem 2.4 is broken down into several individual statements and is given at this section's end. More precisely, in Lemma 2.11-2.11 we show that for starting matrices M_0 with positive leading eigenvalue the iteration $(\lambda_1(M_j))_{j\in\mathbb{N}}$ is strictly increasing, converges to a well-defined limit λ_∞, and that the size of the update $\|M_k - M_{j+1}\|_F$ decays. According to Lemma 2.13, convergent subsequences converge to fixpoints of F_γ, which we can prove to be stationary points of (2.14). Let us remark that the operator F_γ is well-defined, i.e., the denominator is non-zero, as can be seen from the following result.

Lemma 2.10 ([52, Lemma 17]). *Let $X \in \widehat{\mathcal{W}} \cap \mathrm{S}$ with $\lambda_1(X) > 0$ and $\gamma > 0$. Then*

$$\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u_1(X) \otimes u_1(X))\right\|_F^2 = 1 + 2\gamma\lambda_1(X) + \gamma^2 \left\|P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X))\right\|_F^2 .$$

In particular, $F_\gamma(X)$ is well-defined.

Proof. The result follows from $\left\langle X, P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X))\right\rangle = \lambda_1(X)$ and computing explicitly the squared norm $\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u_1(X) \otimes u_1(X))\right\|_F^2$. □

Our convergence analysis of the sequence $(M_j)_{j\in\mathbb{N}}$ induced by F_γ first establishes that the sequence $(\lambda_1(M_j))_{j\in\mathbb{N}}$ is a strictly increasing sequence as long as the iteration is started from a matrix with a positive eigenvalue.

Lemma 2.11 ([52, Lemma 20]). *Let $\gamma > 0$ and $X \in \widehat{\mathcal{W}} \cap \mathrm{S}$ with $\lambda_1(X) > 0$. Then we have*

$$0 < \lambda_1(X) < \left\|P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X))\right\|_F \quad \text{if and only if} \quad \lambda_1(F(X)) > \lambda_1(X), \tag{2.27}$$

$$\lambda_1(X) = \left\|P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X))\right\|_F \quad \text{if and only if} \quad F_\gamma(X) = X. \tag{2.28}$$

In particular, if $\lambda_1(M_0) > 0$, then the sequence $(\lambda_1(M_j))_{j\in\mathbb{N}}$ is strictly increasing and converges to a well-defined limit λ_∞.

Lemma 2.11 shows that Algorithm 2.1 will produce an iteration with converging eigenvalues $(\lambda_1(M_j))_{j\in\mathbb{N}}$ by monotonicity.

Proof of Lemma 2.11. For simplicity we denote $u := u_1(X)$ and $\lambda = \lambda_1(X)$ in this proof. We first prove that $0 < \lambda < \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F$ implies $\lambda_1(F(X)) > \lambda$. It suffices to show that there exists any unit vector v such that $v^T F_\gamma(X) v > \lambda$. In particular, we can test $F_\gamma(X)$ with $v = u$, which yields the identity

$$\begin{aligned} u^T F_\gamma(X) u - \lambda &= \left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F^{-1} \left\langle P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u), u \otimes u \right\rangle - \lambda \\ &= \left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F^{-1} \left(\langle X, u \otimes u\rangle + \gamma \left\langle P_{\widehat{\mathcal{W}}}(u \otimes u), u \otimes u \right\rangle\right) - \lambda \\ &= \left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F^{-1} \left(\lambda + \gamma \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2\right) - \lambda \\ &= \frac{\lambda\left(1 - \left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F\right) + \gamma \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2}{\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F}. \end{aligned}$$

By using now $\lambda < \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F$, we can bound

$$\begin{aligned} &1 - \left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F = 1 - \sqrt{\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F^2} = 1 - \sqrt{\left\|X + \gamma P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2} \\ &= 1 - \sqrt{1 + \gamma^2 \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2 + 2\gamma \left\langle X, P_{\widehat{\mathcal{W}}}(u \otimes u)\right\rangle} = 1 - \sqrt{1 + \gamma^2 \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2 + 2\gamma\lambda} \\ &> 1 - \sqrt{1 + \gamma^2 \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2 + 2\gamma \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F} = 1 - \sqrt{\left(1 + \gamma \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F\right)^2} \\ &= -\gamma \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F. \end{aligned}$$

Inserting this inequality in the previous identity, we obtain the wished result by

$$\begin{aligned} u F_\gamma(X) u - \lambda &= \frac{\lambda\left(1 - \left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F\right) + \gamma \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2}{\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F} \\ &> \frac{-\lambda\gamma \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F + \gamma \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2}{\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F} > 0. \end{aligned} \tag{2.29}$$

We show now that $F_\gamma(X) = X$ implies $\lambda = \left\|P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X))\right\|_F$. We notice that $F_\gamma(X) = X$ implies $\lambda(F_\gamma(X)) = \lambda$, and thus $\lambda \geq \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F$ according to (2.27). Since generally $\lambda \leq \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F$ by Lemma 2.6, equality follows. We address now the converse, i.e., $\lambda = \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F$ implies $F_\gamma(X) = X$, and we note that $\lambda = \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F$ implies $X = \lambda^{-1} P_{\widehat{\mathcal{W}}}(u \otimes u)$ by Lemma 2.6. Using this and the definition of $F_\gamma(X)$, we get

$$F_\gamma(X) = \frac{P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)}{\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u \otimes u)\right\|_F} = \frac{(\lambda^{-1} + \gamma) P_{\widehat{\mathcal{W}}}(u \otimes u)}{(\lambda^{-1} + \gamma)\left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F} = \frac{P_{\widehat{\mathcal{W}}}(u \otimes u)}{\left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F} = X.$$

To conclude the proof it remains to show $\lambda_1(F(X)) > \lambda$ implies $0 < \lambda < \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F$. As $\lambda \leq \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F$ and $\lambda_1(F(X)) > \lambda$ implies $F_\gamma(X) \neq X$ and therefore $\lambda \neq \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F$, then necessarily $\lambda < \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F$. We just established that the sequence $(\lambda_j)_{j \in \mathbb{N}}$ is monotone in the bounded domain $[0, 1]$ and therefore converges to a limit λ_∞. □

Next, we will extend the result from Lemma 2.11, which only relates to the eigenvalues of the iteration $(M_j)_{j \in \mathbb{N}}$, to the iterates themselves.

Lemma 2.12 ([52, Lemma 21]). *Let $\gamma > 0$, $M_0 \in \widehat{\mathcal{W}} \cap \mathbb{S}$ with $\lambda_1(M_0) > 0$, and let $M_j := F_\gamma(M_{j-1})$. Then we have*

$$\lim_{j \to \infty} \|M_{j+1} - M_j\|_F = 0.$$

Proof of Lemma 2.12. For simplicity, we denote $U_j := P_{\widehat{\mathcal{W}}}(u_1(M_j) \otimes u_1(M_j))$, $\lambda_j = \lambda_1(M_j)$. To prove $\|M_{j+1} - M_j\|_F \to 0$, we will exploit $(\lambda_{j+1} - \lambda_j) \to 0$. We first have $(\|U_j\|_F - \lambda_j) \to 0$ since (2.29) yields

$$\lambda_{j+1} - \lambda_j \geq \frac{\gamma \|U_j\|_F}{\|M_j + \gamma U_j\|} \left(\|U_j\|_F - \lambda_j \right) \geq \frac{\gamma}{1+\gamma} \|U_j\|_F \left(\|U_j\|_F - \lambda_j \right),$$

and $\|U_j\|_F \geq \lambda_j \geq \lambda_0$ for all j. Define the shorthand $\Delta_j := \|U_j\|_F - \lambda_j$. We will now show that $\|M_{j+1} - M_j\|_F \leq C\Delta_j$ for some constant C. First, notice that

$$\begin{aligned} \left\| M_j - \lambda_j^{-1} U_j \right\|_F &= \sqrt{1 + \frac{\|U_j\|_F^2}{(\lambda_j)^2} - 2\lambda_j^{-1} \langle M_j, U_j \rangle} = \sqrt{\frac{\|U_j\|_F^2}{(\lambda_j)^2} - 1} \\ &= \sqrt{\frac{\|U_j\|_F^2 - (\lambda_j)^2}{(\lambda_j)^2}} \leq \lambda_0^{-1} \sqrt{2\Delta_j}. \end{aligned}$$

Therefore, there exists a matrix E_j with $M_j = \lambda_j^{-1} U_j + E_j$ and $\|E_j\| \leq \lambda_0^{-1} \sqrt{2\Delta_j}$. Furthermore, by the triangle inequality we have

$$\|M_{j+1} - M_j\|_F \leq \left\| M_{j+1} - \lambda_j^{-1} U_j \right\|_F + \lambda_0^{-1} \sqrt{2\Delta_j},$$

hence it remains to bound the first term. Using $M_j = \lambda_j^{-1} U_j + E_j$ and

$$M_{j+1} = \|M_j + \gamma U_j\|_F^{-1} (M_j + \gamma U_j),$$

we have $\|M_j + \gamma U_j\|_F M_{j+1} = (\lambda_j^{-1} + \gamma) U_j + E_j$ and thus

$$\begin{aligned} \left\| \|M_j + \gamma U_j\|_F (M_{j+1} - \lambda_j^{-1} U_j) \right\|_F &= \left\| (\lambda_j^{-1} + \gamma) U_j + E_j - \left\| (\lambda_j^{-1} + \gamma) U_j + E_j \right\|_F \lambda_j^{-1} U_j \right\|_F \\ &\leq \left| \lambda_j^{-1} + \gamma - \left\| (\lambda_j^{-1} + \gamma) U_j + E_j \right\|_F \lambda^{-1} \right| \|U_j\|_F + \|E_j\|_F \\ &\leq \left((\lambda_j^{-1} + \gamma) \|U_j\|_F \lambda_j^{-1} - (\lambda_j^{-1} + \gamma) + 2 \|E_j\|_F \lambda_j^{-1} \right) \|U_j\|_F + \|E_j\|_F \\ &\leq (\lambda_j^{-1} + \gamma) \left(\|U_j\|_F \lambda_j^{-1} - 1 \right) \|U_j\|_F + (1 + 2\lambda_0^{-1}) \|E_j\|_F \\ &\leq (\lambda_0^{-1} + \gamma) \lambda_0^{-1} \Delta_j + (1 + 2\lambda_0^{-1}) \sqrt{\Delta_j}. \end{aligned}$$

Since $\|M_j + \gamma U_j\|_F \geq 1$ according to Lemma 2.10, $\|M_{j+1} - M_j\| \to 0$ follows. □

The statement above shows that the updates of the iteration will decay for increasing j, i.e., $\|M_{j+1} - M_j\|_F \to 0$ for $j \to \infty$. What is left is to make a connection between any potential limit of the sequence $(M_j)_{j \in \mathbb{N}}$ and the local maximizers of (2.14). The following Lemma shows that every convergent subsequence converges to a fixpoint of F_γ. Through (2.28) Lemma 2.5 and Lemma 2.6 it is known that these fixpoints are stationary points of (2.14), thereby Lemma 2.13 establishes the desired connection. However, note that this statement relies on the continuity of F_γ, which we will address soon.

Lemma 2.13 ([52, Lemma 34])**.** *Let (A, d) be a metric space and $F : A \to A$ be a continuous function. Let $(X_j)_{j \in \mathbb{N}}$ be a sequence generated by $X_j = F^j(X_0)$ for some $X_0 \in A$, and assume $d(X_{j+1}, X_j) \to 0$. Then any convergent subsequence of $(X_j)_{j \in \mathbb{N}}$ converges to a fixed point of F.*

Proof. Let $(X_{j_k})_{k\in\mathbb{N}}$ be a convergent subsequence of $(X_j)_{j\in\mathbb{N}}$ with limit $\bar{X} = \lim_{k\to\infty} X_{j_k}$. Then the subsequence X_{j_k+1} satisfies $d(X_{j_k+1}, \bar{X}) \leq d(X_{j_k+1}, X_{j_k}) + d(X_{j_k}, \bar{X}) \to 0$ as $k \to \infty$, and thus also $(X_{j_k+1})_{k\in\mathbb{N}}$ converges to $\bar{X}$. By construction $X_{j_k+1} = F(X_{j_k})$. Taking the limit $k \to \infty$ on both sides and using the continuity of F, we get

$$\bar{X} = \lim_{k\to\infty} X_{j_k+1} = \lim_{k\to\infty} F(X_{j_k}) = F\left(\lim_{k\to\infty} X_{j_k}\right) = F(\bar{X}).$$

□

The following statement proves the continuity of F_γ for arbitrary $\gamma > 0$ if we further constrain the input domain $\widehat{\mathcal{W}} \cap \mathrm{S}$ to those matrices which also have an isolated first eigenvalue and thereby makes Lemma 2.13 applicable to F_γ.

Lemma 2.14 ([52, Lemma 22])**.** *Let $\gamma > 0$, $\epsilon > 0$ arbitrary, and define $M_\epsilon := \{M \in \widehat{\mathcal{W}} \cap \mathrm{S} : \lambda(M) \geq (\frac{1}{2} + \epsilon)^{1/2}\}$. Then $F_\gamma(X) \in \mathcal{M}_\epsilon$ for all $X \in M_\epsilon$, and F_γ is $\|\cdot\|_F$-Lipschitz continuous, with Lipschitz constant $(1 + \gamma/\epsilon)$.*

Proof. $F_\gamma(X) \in M_\epsilon$ follows directly from Lemma 2.11, i.e., from the fact that the largest eigenvalue is only increased by applying F_γ. For the continuity, consider $X, Y \in M_\epsilon$. We first note that by using [20, Theorem 7.3.1] and $\lambda_i(Y) \leq \sqrt{1/2 - \varepsilon}$ for $i = 2, \ldots, m_0$ we get

$$\begin{aligned}
&\left\|X + \gamma P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X)) - Y - \gamma P_{\widehat{\mathcal{W}}}(u_1(Y) \otimes u_1(Y))\right\|_F \\
&\leq \|X - Y\|_F + \gamma \left\|u_1(X) \otimes u_1(X) - u_1(Y) \otimes u_1(Y)\right\|_F \\
&\leq \|X - Y\|_F + \gamma \frac{\|X - Y\|_F}{\sqrt{\frac{1}{2} + \varepsilon} - \sqrt{\frac{1}{2} - \epsilon}} \leq \left(1 + \frac{\gamma}{\epsilon}\right) \|X - Y\|_F .
\end{aligned}$$

Furthermore, we have $\left\|X + \gamma P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X))\right\|_F^2 \geq 1$ according to Lemma 2.10, and therefore P_{S} acts on $X + \gamma P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X))$ and $Y + \gamma P_{\widehat{\mathcal{W}}}(u_1(Y) \otimes u_1(Y))$ as a projection onto the convex set $\{X : \|X\|_F \leq 1\}$. Therefore, it acts as a contraction, and the result follows from

$$\|F_\gamma(X) - F_\gamma(Y)\|_F \leq \left\|X + \gamma P_{\widehat{\mathcal{W}}}(u_1(X) \otimes u_1(X)) - Y - \gamma P_{\widehat{\mathcal{W}}}(u_1(Y) \otimes u_1(Y))\right\|_F .$$

□

Started from an initial matrix with isolated eigenvalue, as described in the setting of Lemma 2.14, we can now apply Lemma 2.13 to prove that any convergent subsequence converges to a stationary point of (2.14).

Proof of Theorem 2.4. By Lemma 2.14 the operator F_γ is continuous on $M_\epsilon := \{M \in \widehat{\mathcal{W}} \cap \mathrm{S} : \lambda_1(M) \geq (\frac{1}{2} + \epsilon)^{-1/2}\}$ for any $\epsilon > 0$. Moreover, by Lemma 2.11 we have $(M_{j+1})_{j\in\mathbb{N}} \subset M_\epsilon$, and by Lemma 2.12 we have $\left\|M_{j+1} - M_j\right\|_F \to 0$. Therefore, we can apply Lemma 2.13 to see that any convergent subsequence converges to a fixed point of F_γ. Moreover, since $(M_{j+1})_{j\in\mathbb{N}}$ is bounded, there exists at least one convergent subsequence by Bolzano-Weierstrass. Finally, any fixed point $\bar{M}$ of F_γ can be written as $\bar{M} = \lambda_1(\bar{M}) P_{\widehat{\mathcal{W}}}(u_1(\bar{M}) \otimes u_1(\bar{M}))$ by Lemma 2.6 and Lemma 2.11. Since $\lambda_1(\bar{M}) > 1/\sqrt{2}$, it is an isolated eigenvalue satisfying $\lambda_1(\bar{M}) = \|\bar{M}\|$, and thus $\bar{M}$ satisfies the first-order optimality condition (2.15) of (2.14) by Theorem 2.3. □

2.3.5 Discussion of open problems

We developed an analysis of the local maximizers of the perturbed non-linear program (2.14) that is designed to select near-rank-one matrices from a symmetric matrix space $\widehat{\mathcal{W}}$ that approximates the span of rank-one matrices. It was shown that local maximizers satisfy $\|M\|^2 \geq 1 - c\delta - c'\nu$, or $\|M\|^2 \leq c\delta + c'\nu$, where $\delta = \|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F}$ and ν originates from (2.10). Therefore, local maximizers of the first kind are close to the spanning elements for ν, δ sufficiently small by Theorem 2.1. We have reason to believe (cf. Section 2.5), that the spurious local maxima of the second kind cannot be avoided for our setting. In the upcoming sections, we provide a counterexample for a similar problem (cf. Lemma 2.19). The convergence analysis of Section 2.3.4 gives guarantees under which subsequences of the iteration (2.26) converge to stationary points and that the iteration produces a sequence of matrices with increasing spectral norm under appropriate starting conditions. As it has been pointed out in [52], the gap between our theoretical guarantees and empirical results suggests that the analysis in Section 2.3.4 is not optimal and that issues could potentially be closed. Three critical open problems are stated below.

Local maximizers near each spanning element. Our characterization of local maximizers establishes conditions under which each local maximizer of (2.14) must be close to one of the spanning elements $w_1 \otimes w_1, \ldots, w_m \otimes w_m$. However, we do not prove that all spanning elements can be associated with one local maximum. A statement of this kind could be proven by showing that for each $k \in [m]$, the objective (2.14) constrained on the compactum $U_k \subset \mathcal{W} \cap \mathbb{S}$ surrounding $P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)/\|P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)\|_F$ has at least one global maximum within its interior $U_k \setminus \partial U_k$. Let us mention that we provide such a statement as part of the next sections, which studies a different characterization of the near-rank-one matrices. We believe that a similar statement could be shown for (2.14), but its proof seems non-trivial due to the more complicated geometry of the compactum U_k.

Computation of near-rank-one matrices. We did not show that Algorithm 2.1 will always converge to near-rank-one matrices in the perturbed space. Since this is what is numerically observed (cf. [52]), we believe such a statement could be achieved by better leveraging the structure of $\mathcal{W}$ in our convergence analysis of Section 2.3.4.

Computational complexity: Practical considerations. One issue with Algorithm 2.1 is that the iteration

$$M_{j+1} = P_{\mathbb{S}}(P_{\widehat{\mathcal{W}}}(M_j + \gamma u_1(M_j) \otimes u_1(M_j)))$$

requires the computation of the eigendecomposition at every iteration. This poses a computational challenge because the computation of the eigendecomposition in higher dimensions is costly and hard to parallelize. This plays a role since we need to restart Algorithm 2.1 on average $\Theta(m \log m)$ times (see Remark 2.2) to be able to compute all near-rank-one matrices.

2.4 Subspace power method

Consider again a set of unit-vectors $\{w_1, \ldots, w_m\} \subset \mathbb{S}^{D-1}$ for $m < 2D$ and a symmetric matrix space $\widehat{\mathcal{W}}$ that approximates the span

$$\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \ldots, w_m \otimes w_m\}.$$

The computation of near-rank-one matrices within $\widehat{\mathcal{W}}$ based on the spectral norm, as presented previously, comes with some issues discussed in the preceding section. We will now focus a different approach to the rank-one basis recovery problem. The upcoming results address many of the issues we mentioned beforehand, and we include an in-depth discussion of these points which is deferred to Section 2.5.4. The approach that we want to discuss within this section is based on the characterization of near-rank-one matrices as the maximizers of the program

$$\max_{\|u\|=1} \Phi_{\widehat{\mathcal{W}}}(u) = \max_{\|u\|=1} \left\| P_{\widehat{\mathcal{W}}}(u \otimes u) \right\|_F^2 . \tag{2.30}$$

To the best of our knowledge, using this non-convex program to recover spanning elements in the context of neural network reconstruction has first been suggested in [49]. Independently, a more general version of (2.30) has been studied in the broader context of symmetric tensor decompositions in [77]. Similar to our previous approach, the authors of [77] propose to compute local maximizers of (2.30) using a projected gradient descent iteration which is called *subspace power method* (SPM) and given by

$$u_j = P_{\mathbb{S}^{D-1}}(u_{j-1} + 2\gamma P_{\widehat{\mathcal{W}}}(u_{j-1} \otimes u_{j-1}) u_{j-1}), \qquad j \in \mathbb{N}_{>0}, \tag{2.31}$$

where $\gamma > 0$ is a fixed step-size parameter. The primary result of [77] adapted to our problem setting comes with strong guarantees and can be summarized as the following statement (see also [50, Theorem 9]).

Theorem 2.5 ([77, Theorem 4.1])**.** *Let $\widehat{\mathcal{W}} \subseteq Sym(\mathbb{R}^{D\times D})$. Let $\gamma > 0$ be a fixed parameter such that $\Phi_{\widehat{\mathcal{W}}}(u) + \frac{\gamma}{4}\|u\|_2^2$ is strictly convex on $\mathbb{R}^D$ (e.g., $\gamma \geq \frac{1}{2}$ works) and consider the iteration*

$$u_j = P_{\mathbb{S}^{D-1}}(u_{j-1} + 2\gamma P_{\widehat{\mathcal{W}}}(u_{j-1} \otimes u_{j-1}) u_{j-1}), \qquad j \in \mathbb{N}_{>0}. \tag{2.32}$$

Then the following points hold:

(SPM1) *For any $u_0 \in \mathbb{S}^{D-1}$ the iteration* (2.32) *is well-defined and converges monotonically to a constrained stationary point of $\Phi_{\widehat{\mathcal{W}}}$ at a power rate.*

(SPM2) *For a full Lebesgue measure subset of initializations $u_0 \in \mathbb{S}^{D-1}$,* (2.32) *converges to a constrained local maximizer of $\Phi_{\widehat{\mathcal{W}}}$.*

(SPM3) *If $\widehat{\mathcal{W}} = \mathcal{W} = \operatorname{span}\{w_k \otimes w_k : k \in [m]\}$ for $m < \frac{1}{2}(D-1)D$ and $w_1, \dots, w_m$ are sampled from any absolutely continuous probability distribution on $\mathbb{S}^{D-1}$, constrained global maximizers of $\Phi_{\widehat{\mathcal{W}}}$ are precisely $\pm w_k$. Moreover, each $\pm w_k$ is exponentially attractive, which means that initializations $u_0 \in \mathbb{S}^{D-1}$ sufficiently close to $\pm w_k$ converge to $\pm w_k$ with an exponential rate.*

Proof. Note that $\Phi_{\widehat{\mathcal{W}}}(u) = \sum_{i=1}^{K} (u^\top W_k u)^2$ for a basis $\{W_k : k \in [\dim(\widehat{\mathcal{W}})]\}$ of $\widehat{\mathcal{W}}$, which implies that $\Phi_{\widehat{\mathcal{W}}}(u)$ is a homogeneous polynomial of degree 4. Hence, [77, Section 5] applies. □

Let us provide some context on this statement. By **(SPM2)**, iteration (2.32) started from u_0 drawn uniformly from the unit sphere will converge to a local maximizer of the perturbed objective $\Phi_{\widehat{\mathcal{W}}}$. Since **(SPM2)** holds for all symmetric matrix spaces, it suits our problem setting and provides us with a statement similar to Theorem 2.4. What remains to show is that we can extend **(SPM3)** to the perturbed setting where $\widehat{\mathcal{W}}$ resembles only approximately a space spanned by rank-one matrices. This means we need to show that the local maximizers of $\Phi_{\widehat{\mathcal{W}}}$ are in some sense close to the local maximizers of $\Phi_{\mathcal{W}}$ which are given by $\pm w_k$. Note that this implies that there is a local maximizer near every spanning element, and thereby directly addresses one of the open problems stated in Section 2.3.5. The extension of **(SPM3)** will be

Algorithm 2.2: Iterating SPM to find all near-rank-one basis approximations.

Input: $P_{\widehat{\mathcal{W}}}$ orthogonal projection onto matrix space $\widehat{\mathcal{W}}$, threshold $\beta > 0$

Set $\mathcal{U} \leftarrow \emptyset$ and $m \leftarrow \dim \widehat{\mathcal{W}}$
while $|\mathcal{U}| < m$ **do**
  Sample $u_0 \sim \mathrm{Unif}(\mathbb{S}^{D-1})$
  Iterate projected gradient ascent

$$u \leftarrow P_{\mathbb{S}^{D-1}}(u + 2\gamma P_{\widehat{\mathcal{W}}}((u)^{\otimes 2})u)$$

  until convergence, and denote the vector of the final iteration by $\hat{u}$.
  if $\left\|P_{\widehat{\mathcal{W}}}(\hat{u}^{\otimes 2})\right\|_F^2 > \beta$ **then**
    if $\hat{u} \notin \mathcal{U}$ *and* $-\hat{u} \notin \mathcal{U}$ **then**
      $\mathcal{U} \leftarrow \mathcal{U} \cup \{\hat{u}\}$
    end
  end
end

Output: $\mathcal{U}$

the central topic of the upcoming section. The combination of all the guarantees then allows us to formulate Algorithm 2.2 that repeatedly draws random initializations from the unit sphere and, under the right conditions, will return all near-rank-one matrices in the perturbed matrix space (cf. Remark 2.2). This algorithm will play a *fundamental role* in the identification of neural network weights discussed in Chapter 3 and Chapter 4. Lastly, let us mention that the results of [77] have recently been extended in [76] to include the perturbed case in the tensor setting. We will separately state relevant parts of their results in Section 2.6.

2.5 Perturbation analysis of the SPM objective under deterministic frame bounds

In this section, we prove an extension of **(SPM3)** to the perturbed case. We will assume the same incoherence assumptions as before. Consider unit-norm vectors $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ that satisfy the deterministic frame condition (2.10) for some $\nu < 1$ (implying $m < 2D$). Under this setting, and assuming the approximation $\widehat{\mathcal{W}} \approx \mathcal{W} = \{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$ is sufficiently accurate, we prove the following main result.

Theorem 2.6 ([50, Theorem 10])**.** *Let* $\mathrm{span}\{w_1, \dots, w_m\} \subset \mathbb{S}^{D-1}$ *satisfy* (2.10) *with* $\nu < 1$ *and denote* $\mathcal{W} := \{w_1 \otimes w_1, \dots, w_m \otimes w_m\}$. *Let* $\widehat{\mathcal{W}} \subseteq \mathrm{Sym}(\mathbb{R}^{D \times D})$ *satisfy* $\left\|P_{\widehat{\mathcal{W}}} - P_{\mathcal{W}}\right\|_{2\to 2} < \delta$ *and assume that* ν *and* δ *are small enough to satisfy*

$$4\nu + \nu^2 + 11\delta < 1 \quad \text{and} \quad \delta < \frac{1}{22}\left(1 - \frac{3\nu}{1+\nu}\right)^2.$$

For each $k \in [m]$ *there exists a local maximizer* u_k^* *of* $\Phi_{\widehat{\mathcal{W}}}$ *with* $\Phi_{\widehat{\mathcal{W}}}(u_k^*) \geq 1 - \delta$ *within the cap*

$$U_k := \left\{u \in \mathbb{S}^{D-1} : \langle u, w_k \rangle \geq \sqrt{(1-3\delta)\frac{1-\nu}{1+\nu}}\right\}.$$

Furthermore, for any constrained local maximizer $u \in \mathbb{S}^{D-1}$ of $\Phi_{\widehat{\mathcal{W}}}$ with $\Phi_{\widehat{\mathcal{W}}}(u) > 7\frac{1+\nu}{1-\nu}\delta$ and basis expansion $P_{\widehat{\mathcal{W}}}(u \otimes u) = \sum_{k=1}^{m} \sigma_k P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)$ ordered according to $\sigma_1 \geq \ldots \geq \sigma_m$, we have

$$\min_{s\in\{-1,1\}} \|u - sw_1\|_2 \leq \frac{\sqrt{2\nu \sum_{k=2}^{m} \sigma_k^2} + 2\delta}{(1-\nu)(1 - 6\frac{\nu}{1+\nu} - 18\delta) - \sqrt{6\frac{\nu}{1+\nu} + 18\delta} - 2\delta}. \tag{2.33}$$

The proof is broken down into several components and given at the end of Section 2.5.3. This result shows that there will be a local maximizer in the vicinity of $\pm w_k$ whenever δ, ν are sufficiently small. More precisely, we provide an error bound between the leading eigenvector of the matrix induced by $P_{\widehat{\mathcal{W}}}(u \otimes u)$ (where u is such a local maximizer) and $\pm w_k$. Like Theorem 2.2, Theorem 2.6 does not exclude the presence of spurious local maximizers.

2.5.1 Optimality conditions

First, constrained stationary points and local maximizers in Theorem 2.5 are derived from the Riemannian gradient and Hessian with respect to $\mathbb{S}^{D-1}$. More precisely, consider any smooth function $f : \mathbb{R}^D \to \mathbb{R}$, then $u \in \mathbb{S}^{D-1}$ is a constrained stationary point of f w.r.t. $\mathbb{S}^{D-1}$ if and only if

$$\nabla_{\mathbb{S}^{D-1}} f(u) := (\mathrm{Id}_D - uu^\top)\nabla f(u) = 0. \tag{2.34}$$

Furthermore, u is a constrained local maximizer of f w.r.t. $\mathbb{S}^{D-1}$ if and only if it is a constrained stationary point and the Riemannian Hessian given by

$$\nabla^2_{\mathbb{S}^{D-1}} f(u) := (\mathrm{Id}_D - uu^\top)\left(\nabla^2 f(u) - \langle u, \nabla f(u)\rangle \,\mathrm{Id}_D\right)(\mathrm{Id}_D - uu^\top) \preccurlyeq 0, \tag{2.35}$$

is negative semidefinite. For our context, it will be enough to compute (2.34) and (2.35) for $\Phi_{\widehat{\mathcal{W}}}$ (see Lemma 2.16 and the corresponding proof) and for further details on Riemannian optimization we refer to [4].

Our first step in proving Theorem 2.6 is to reformulate the general optimality conditions we stated in (2.34) and (2.35) in such a way that we can leverage them in our analysis. We start by computing the first- and second-order derivative of $\Phi_{\widehat{\mathcal{W}}}$.

Lemma 2.15 ([50, Lemma 11])**.** *Let $\widehat{\mathcal{W}} \subseteq Sym(\mathbb{R}^{D\times D})$ and take $u \in \mathbb{S}^{D-1}$. The gradient and Hessian of $\Phi_{\widehat{\mathcal{W}}} : \mathbb{R}^D \to \mathbb{R}$ satisfy*

$$\begin{aligned} \nabla\Phi_{\widehat{\mathcal{W}}}(u) &= 4P_{\widehat{\mathcal{W}}}(u \otimes u)u, \\ v^\top \nabla^2 \Phi_{\widehat{\mathcal{W}}}(u)v &= 8\left\|P_{\widehat{\mathcal{W}}}(u \otimes v)\right\|_F^2 + 4v^\top P_{\widehat{\mathcal{W}}}(u \otimes u)v \quad \textit{for all} \quad v \in \mathbb{S}^{D-1}. \end{aligned} \tag{2.36}$$

Proof of Lemma 2.15. To compute the gradient, we first note that

$$\partial_{u_j}\Phi_{\widehat{\mathcal{W}}}(u) = 2\langle P_{\widehat{\mathcal{W}}}(u \otimes u), \partial_{u_j} P_{\widehat{\mathcal{W}}}(u \otimes u)\rangle = 2\langle P_{\widehat{\mathcal{W}}}(u \otimes u), \partial_{u_j}(u \otimes u)\rangle$$

Furthermore, we have $\partial_{u_j}(u \otimes u) = e_j \otimes u + u \otimes e_j$, where e_j is the j-th standard basis vector. This implies the result by

$$\partial_{u_j}\Phi_{\widehat{\mathcal{W}}}(u) = 2\langle P_{\widehat{\mathcal{W}}}(u \otimes u), e_j \otimes u + u \otimes e_j\rangle = 4e_j^\top P_{\widehat{\mathcal{W}}}(u \otimes u)u.$$

For (2.36) we use an arbitrary orthonormal basis $\{B_1, \ldots, B_m\}$ of $\widehat{\mathcal{W}}$ and write $\nabla\Phi_{\widehat{\mathcal{W}}}(u) = 4\sum_{k=1}^{m}\langle u \otimes u, B_k\rangle B_k u$. Differentiating again with respect to u_j, we obtain the rows of the Hessian

as

$$\partial_{u_j}\left(\sum_{k=1}^m \langle u\otimes u, B_k\rangle B_k\right)u = \left(\sum_{k=1}^m \langle u\otimes u, B_k\rangle B_k\right)e_j + \left(\sum_{k=1}^N \langle (u\otimes e_j + e_j\otimes u), B_k\rangle B_k\right)u$$
$$= P_{\widehat{\mathcal{W}}}(u\otimes u)e_j + 2\sum_{k=1}^m (B_k u)_j B_k u,$$

which implies $\nabla^2\Phi_{\widehat{\mathcal{W}}}(u) = 4P_{\widehat{\mathcal{W}}}(u\otimes u) + 8\sum_{k=1}^m B_k u\otimes(B_k u)$. Multiplying with v from the left and the right, it follows that

$$v^\top\nabla^2\Phi_{\widehat{\mathcal{W}}}(u)v = 4v^\top P_{\widehat{\mathcal{W}}}(u\otimes u)v + 8\sum_{k=1}^m \langle u\otimes v, B_k\rangle^2 = 4v^\top P_{\widehat{\mathcal{W}}}(u\otimes u)v + 8\left\|P_{\widehat{\mathcal{W}}}(u\otimes v)\right\|_F^2.$$

□

The constrained optimality conditions follow now from a straightforward computation by plugging in $\nabla\Phi_{\widehat{\mathcal{W}}}, \nabla^2\Phi_{\widehat{\mathcal{W}}}$ into (2.34) and (2.35):

Lemma 2.16 ([50, Lemma 12])**.** *Let $\widehat{\mathcal{W}} \subseteq Sym(\mathbb{R}^{D\times D})$. A vector $u \in \mathbb{S}^{D-1}$ is a constrained stationary point of $\Phi_{\widehat{\mathcal{W}}}$ if and only if*

$$P_{\widehat{\mathcal{W}}}(u\otimes u)u = \left\|P_{\widehat{\mathcal{W}}}(u\otimes u)\right\|_F^2 u. \tag{2.37}$$

Furthermore, $u \in \mathbb{S}^{D-1}$ is a constrained local maximum of $\Phi_{\widehat{\mathcal{W}}}$ if and only if

$$\left\|P_{\widehat{\mathcal{W}}}(u\otimes u)\right\|_F^2 \geq 2\left\|P_{\widehat{\mathcal{W}}}(u\otimes v)\right\|_F^2 + v^\top P_{\widehat{\mathcal{W}}}(u\otimes u)v \qquad \text{for all } v\in\mathbb{S}^{D-1} \text{ with } v\perp u. \tag{2.38}$$

Proof of Lemma 2.16. According to the last result we have $\nabla\Phi_{\widetilde{\mathcal{W}}}(u) = 4P_{\widetilde{\mathcal{W}}}(u\otimes u)u$. Following the definition of constrained stationary points in (2.34), $u\in\mathbb{S}^{D-1}$ is a constrained stationary point if and only if

$$(\mathrm{Id}_D - u\otimes u)\nabla\Phi_{\widetilde{\mathcal{W}}}(u) = 0 \quad \text{or} \quad 4P_{\widetilde{\mathcal{W}}}(u\otimes u)u - 4\left\|P_{\widetilde{\mathcal{W}}}(u\otimes u)\right\|_F^2 u = 0.$$

Similarly, $u\in\mathbb{S}^{D-1}$ is a constrained local maximum if and only if for any $v\in\mathbb{S}^{D-1}$ we have

$$v^\top(\mathrm{Id}_D - u\otimes u)\left(\nabla^2\Phi_{\widetilde{\mathcal{W}}}(u) - \langle u, \nabla\Phi_{\widetilde{\mathcal{W}}}(u)\rangle\,\mathrm{Id}_D\right)(\mathrm{Id}_D - u\otimes u)v \leq 0. \tag{2.39}$$

Since $\mathrm{Id}_D - u\otimes u$ is the orthogonal projection onto span $u^\perp$ (2.39) is actually equivalent to

$$v^\top\left(\nabla^2\Phi_{\widetilde{\mathcal{W}}}(u) - \langle u, \nabla\Phi_{\widetilde{\mathcal{W}}}(u)\rangle\,\mathrm{Id}_D\right)v \leq 0 \quad \text{for any } v\in\mathbb{S}^{D-1} \text{ with } v\perp u.$$

Using $\langle u, \nabla\Phi_{\widetilde{\mathcal{W}}}(u)\rangle = 4\left\|P_{\widetilde{\mathcal{W}}}(u\otimes u)\right\|_F^2$, this is equivalent to

$$v^\top\nabla^2\Phi_{\widetilde{\mathcal{W}}}(u)v - 4\left\|P_{\widetilde{\mathcal{W}}}(u\otimes u)\right\|_F^2 \leq 0$$

or by the Hessian formula (2.36) in Lemma 2.15,

$$4\left\|P_{\widetilde{\mathcal{W}}}(u\otimes u)\right\|_F^2 \geq 8\left\|P_{\widetilde{\mathcal{W}}}(u\otimes v)\right\|_F^2 + 4v^\top P_{\widetilde{\mathcal{W}}}(u\otimes u)v.$$

□

Lemma 2.16 provides a very useful characterization of the spectral decomposition of $P_{\widehat{\mathcal{W}}}(u \otimes u)$ whenever u is a stationary point or local maximizer. For any stationary point $u^* \in \mathbb{S}^{D-1}$ of $\Phi_{\widehat{\mathcal{W}}}$ we have according to (2.37)

$$P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)u^* = \left\|P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)\right\|_F^2 u^*.$$

Hence, u^* is an eigenvector of $P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)$ with eigenvalue given by $\left\|P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)\right\|_F^2$. Additionally, if u^* is a local maximizer, then it is the eigenvector corresponding to the maximal eigenvalue of $P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)$. This follows from the argument: Assume v is another eigenvector of $P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)$ corresponding to a different eigenvalue, then v will be orthogonal to u^* since $P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)$ is symmetric. Then (2.38) yields to the inequality

$$v^\top P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)v \leq \left\|P_{\widehat{\mathcal{W}}}(u^* \otimes u^*)\right\|_F^2.$$

2.5.2 Describing the optimization landscape

Next, we will prove the presence of one local maximizer near each of the spanning components $\{w_1, \ldots, w_m\}$ provided δ, ν is sufficiently small. This follows by considering $\Phi_{\widehat{\mathcal{W}}}$ constrained onto a compact set surrounding w_k and then showing that the global maximizer of the constrained functional is not attained on the boundary of the compact set.

Proposition 2.1 ([50, Proposition 16]). *Let $\{w_1, \ldots, w_m\} \subset \mathbb{S}^{D-1}$ satisfy* (2.10) *and denote $\mathcal{W} := \{w_k \otimes w_k | k \in [m]\}$. Let $\widehat{\mathcal{W}} \subseteq Sym(\mathbb{R}^{D\times D})$ be a subspace with $\left\|P_{\widehat{\mathcal{W}}} - P_{\mathcal{W}}\right\|_{2\to 2} \leq \delta$, and assume ν and δ satisfy*

$$4\nu + \nu^2 + 6\delta \leq 1.$$

For each $k \in [m]$ there exists a local maximizer u_k^* of $\Phi_{\widehat{\mathcal{W}}}$ with $\Phi_{\widehat{\mathcal{W}}}(u_k^*) \geq 1 - \delta$ within the cap

$$U_k := \left\{u \in \mathbb{S}^{D-1} : \langle u, w_k\rangle \geq \sqrt{(1-3\delta)\frac{1-\nu}{1+\nu}}\right\}.$$

The proof follows below and can be summarized as follows: First, note that due to the compactness U_k, the constrained objective $u \mapsto \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2$ over U_k must have a global maximum within U_k. We show that $\|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2 < \|P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)\|_F^2$ for all $u \in \partial U_k$, where ∂U_k denotes the closure of the compactum U_k. This implies that the constrained global maximizer is not attained on the boundary ∂U_k and, therefore, must be contained in $U_k \setminus \partial U_k$. This implies that the unconstrained objective $u \mapsto \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2$ over $\mathbb{S}^{D-1}$ must have a local maximizer in U_k. This proof for Proposition 2.1 benefits from the following auxiliary result, which expresses the objective value $\Phi_{\mathcal{W}}(u)$ in terms of the Gramian matrix of the elements $w_1^{\otimes 2}, \ldots, w_m^{\otimes 2}$. The proofs use some concepts of Gramian matrices and the Gershgorin circle theorem, which will be discussed in more detail in Section 2.6.1.

Lemma 2.17 ([50, Lemma 13]). *Let $\{w_k : k \in [m]\} \subset \mathbb{S}^{D-1}$, assume $\{w_k \otimes w_k : k \in [m]\}$ are linearly independent, and denote $\mathcal{W} := \operatorname{span}\{w_k \otimes w_k : k \in [m]\}$. Denote $G_{k\ell} := \langle w_k, w_\ell\rangle^2$ as the Gramian matrix of $\{w_k \otimes w_k | k \in [m]\}$. Then we have*

$$\Phi_{\mathcal{W}}(u) = \beta_u G^{-1} \beta_u \qquad \text{for} \qquad (\beta_u)_k := \langle u, w_k\rangle^2. \tag{2.40}$$

Proof. We first write the Frobenius norm as an optimal program

$$\|P_{\mathcal{W}}(u \otimes u)\|_F = \max_{\substack{Z\in\mathcal{W}\\ \|Z\|_F^2=1}} \langle Z, P_{\mathcal{W}}(u \otimes u)\rangle = \max_{\substack{Z\in\mathcal{W}\\ \|Z\|_F^2=1}} u^\top Z u$$

and express $Z = \sum_{k=1}^m \sigma_k w_k \otimes w_k$ in terms of the basis coefficients σ_k. Then using

$$\|Z\|_F^2 = \operatorname{tr}\left(\sum_{k=1}^m \sigma_k w_k \otimes w_k \sum_{\ell=1}^m \tau_\ell w_\ell \otimes w_\ell\right) = \sum_{k=1}^m \sum_{\ell=1}^m \sigma_k \sigma_\ell G_{k\ell} = \sigma^\top G \sigma,$$

we can formulate the initial program as an optimal program over coefficients σ_k by

$$\|P_{\mathcal{W}}(u \otimes u)\|_F = \max_{\sigma^\top G \sigma \le 1} \sum_{k=1}^m \sigma_k \langle u, w_k\rangle^2 = \max_{\sigma^\top G \sigma \le 1} \langle \sigma, \beta\rangle.$$

This is a linear program with quadratic constraints. The Karush–Kuhn–Tucker (KKT) conditions imply (2.40), see also [80]. □

Proof of Proposition 2.1. Let $k \in [m]$ arbitrary. We can prove the statement essentially by showing the inequality $\|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2 < \|P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)\|_F^2$ for all $u \in \partial U_k$ because the compactness of U_k implies the existence of a global maximizer on U_k, and with $\|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2 < \|P_{\mathcal{W}}(w_k \otimes w_k)\|_F^2$ for all $u \in \partial U_k$, the global maximizer is contained in $U_k \setminus \partial U_k$. Hence, it must be a local maximizer of $u \mapsto \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2$ over $\mathbb{S}^{D-1}$. First, note that the perturbed objective at w_k is bounded from below by

$$\begin{aligned}\left\|P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)\right\|_F^2 &= \|P_{\mathcal{W}}(w_k \otimes w_k)\|_F^2 - \|P_{\mathcal{W}}(w_k \otimes w_k)\|_F^2 - \left\|P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)\right\|_F^2 \\ &= 1 - \left|\|P_{\mathcal{W}}(w_k \otimes w_k)\|_F^2 + \left\|P_{\widehat{\mathcal{W}}}(w_k \otimes w_k)\right\|_F^2\right| \\ &= 1 - \left|\langle w_k \otimes w_k, (P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}})(w_k \otimes w_k)\rangle\right| \ge 1 - \delta.\end{aligned}$$

In the last equation, we used the self-adjointness of the ortho-projector and in the inequality, we applied the Cauchy-Schwarz inequality. Next we establish an upper bound for $\|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2$ for $u \in U_k$. We note that with the Gramian matrix $G_{k\ell} := \langle w_k, w_\ell\rangle^2$ of $\{w_k \otimes w_k | k \in [m]\}$ and the vectors $\beta_k := \langle u, w_k\rangle^2$, the unperturbed objective can be written as $\|P_{\mathcal{W}}(u \otimes u)\|_F^2 = \beta^\top G^{-1} \beta$ according to Lemma 2.17. Furthermore, Gershgorin's circle Theorem (cf. Section 2.6.1, Theorem 2.9) implies for any eigenvalue λ of G that there exists $k \in [m]$ such that

$$|1 - \lambda| = |G_{kk} - \lambda| \le \sum_{\ell \ne k}^m \langle w_\ell, w_k\rangle^2 \le \nu.$$

Therefore, $\left\|G^{-1}\right\|_2 \le (1-\nu)^{-1}$ and the noise-free objective can be bounded by

$$\begin{aligned}\|P_{\mathcal{W}}(u \otimes u)\|_F^2 = \beta^\top G^{-1} \beta &\le \left\|G^{-1}\right\|_2 \|\beta\|_2^2 \\ &\le \frac{1}{1-\nu}\|\beta\|_2^2 \le \frac{1}{1-\nu}\|\beta\|_\infty \|\beta\|_1 \le \frac{1+\nu}{1-\nu}\|\beta\|_\infty.\end{aligned}$$

In the last inequality we used frame condition (2.10), which implies $\|\beta\|_1 = \sum_{k=1}^m \langle u, w_k\rangle^2 \le 1 + \nu$. To bound $\|\beta\|_\infty$ note first that

$$\max_{\ell \ne k} \beta_\ell \le \sum_{\ell \ne k} \beta_\ell = \sum_{\ell=1}^m \langle u, w_\ell\rangle^2 - \langle u, w_k\rangle^2 \le 1 + \nu - (1 - 3\delta)\frac{1-\nu}{1+\nu} \le \frac{3\nu + \nu^2 + 3\delta}{1+\nu},$$

which implies with the assumption $4\nu + \nu^2 + 6\delta \le 1$

$$\beta_k = \langle u, w_k\rangle^2 \ge (1 - 3\delta)\frac{1-\nu}{1+\nu} \ge \frac{1 - 3\delta - \nu}{1+\nu} \ge \frac{1 - 6\delta - \nu + 3\delta}{1+\nu} \ge \max_{\ell \ne k} \beta_\ell.$$

Hence, $\|\beta\|_\infty = \langle u, w_k\rangle^2$. It follows that for any $u \in U_k$, we get

$$\begin{aligned}\left\|P_{\widehat{\mathcal{W}}}(u\otimes u)\right\|_F^2 &\leq \|P_{\mathcal{W}}(u\otimes u)\|_F^2 + \left|\left\|P_{\widehat{\mathcal{W}}}(u\otimes u)\right\|_F^2 - \|P_{\mathcal{W}}(u\otimes u)\|_F^2\right| \\ &\leq \frac{1+\nu}{1-\nu}\|\beta\|_\infty + \delta \leq \frac{1+\nu}{1-\nu}\langle u, w_k\rangle^2 + \delta.\end{aligned}$$

Comparing $\left\|P_{\widehat{\mathcal{W}}}(w_k\otimes w_k)\right\|_F^2$ and $\left\|P_{\widehat{\mathcal{W}}}(u\otimes u)\right\|_F^2$ for $u \in \partial U_k$, where $\langle u, w_k\rangle^2 = \frac{1-\nu}{1+\nu}(1-3\delta)$, yields

$$\left\|P_{\widehat{\mathcal{W}}}(u\otimes u)\right\|_F^2 \leq \frac{1+\nu}{1-\nu}\langle u, w_k\rangle^2 + \delta = 1-2\delta < 1-\delta \leq \left\|P_{\widehat{\mathcal{W}}}(w_k\otimes w_k)\right\|_F^2.$$

□

2.5.3 Proof of the Theorem 2.6

To proof Theorem 2.6, we first show a result similar to Theorem 2.2, where it was shown that the local maximizer of $\operatorname{argmax}_{M\in\widehat{\mathcal{W}}\cap\mathbb{S}}\|M\|$ have either a dominant eigenvalue or a uniform spectrum. The theorem below essentially establishes the same type of characterization. However, it further relies on the fact that the leading eigenvalue of any local maximizer of (2.30) can be expressed in terms of the objective value according to Lemma 2.16.

Theorem 2.7 ([50, Theorem 15]). *Let $\{w_1, \dots, w_m\} \subset \mathbb{S}^{D-1}$ satisfy* (2.10) *and denote $\mathcal{W} = \{w_k \otimes w_k | k \in [m]\}$. Let $\widehat{\mathcal{W}} \subseteq Sym(\mathbb{R}^{D\times D})$ satisfy $\|P_{\widehat{\mathcal{W}}} - P_{\mathcal{W}}\|_{F\to F} \leq \delta$ and assume that ν and δ are small enough to satisfy*

$$\frac{3\nu}{1+\nu} + 11\delta < 1 \quad \text{and} \quad \delta < \frac{1}{22}\left(1 - \frac{3\nu}{1+\nu}\right)^2 \tag{2.41}$$

For any constrained local maximizer $u \in \mathbb{S}^{D-1}$ of $\Phi_{\widehat{\mathcal{W}}}$ we have

$$\Phi_{\widehat{\mathcal{W}}}(u) \leq 9\frac{1+\nu}{1-\nu}\delta \qquad \text{or} \qquad \Phi_{\widehat{\mathcal{W}}}(u) \geq 1 - 6\frac{\nu}{1+\nu} - 18\delta.$$

Before stating the proof, we introduce the following auxiliary result.

Lemma 2.18 ([50, Lemma 18]). *Assume the settings of Theorem 2.1 and Theorem 2.7. Let $P_{\widehat{\mathcal{W}}}(u\otimes u) = \sum_{k=1}^m \sigma_k \widehat{W}_k$ for some $u \in \mathbb{S}^{D-1}$ and denote $\lambda_1 := \lambda_1(P_{\widehat{\mathcal{W}}}(u\otimes u))$. Furthermore, assume $1-\delta > \lambda_1 > 3\frac{1+\nu}{1-\nu}\delta$. For any index k^* with $|\sigma_{k^*}| = \|\sigma\|_\infty$ we have $\sigma_{k^*} \geq 0$ and*

$$\sigma_{k^*} \leq \frac{1}{1-\nu}\langle u, w_{k^*}\rangle^2 + \frac{2}{1-\nu}\delta. \tag{2.42}$$

Proof of Lemma 2.18. Denote $M = P_{\widehat{\mathcal{W}}}(u\otimes u) = \sum_{k=1}^m \sigma_k \widehat{W}_k$ and define $Z = \sum_{k=1}^m \sigma_k w_k \otimes w_k$ as the unique matrix in $\mathcal{W}$ with $P_{\widehat{\mathcal{W}}}(Z) = M$. We first note that the statement is trivial if $\|\sigma\|_\infty = 0$ since this implies $\sigma_k = 0$ for all $k \in [m]$ and thus $M = 0$. So without loss of generality, we may assume $\|\sigma\|_\infty \neq 0$ in the following. We will first show $\sigma_{k^*} > 0$ by contradiction, so let us assume $\sigma_\ell < 0$. Using the frame condition (2.10) and $\{w_1, \dots, w_m\} \subset \mathbb{S}^{D-1}$ to show the estimate

$$|\langle Z, w_{k^*}\otimes w_{k^*}\rangle - \sigma_{k^*}| = \left|\sum_{k\neq k^*}\sigma_k\langle w_k, w_{k^*}\rangle^2\right| \leq \|\sigma\|_\infty \sum_{k\neq k^*}\langle w_k, w_{k^*}\rangle^2 \leq \|\sigma\|_\infty \nu, \tag{2.43}$$

and combining this with $\|Z\| \leq (1-\delta)^{-1}\lambda_1(M) < 1$, we can bound $\|\sigma\|_\infty$ by

$$\begin{aligned}\|\sigma\|_\infty = |\sigma_{k^*}| &\leq \left|\sigma_{k^*} - \langle u, w_{k^*}\rangle^2\right| \\ &\leq |\sigma_{k^*} - \langle Z, w_{k^*}\otimes w_{k^*}\rangle| + \left|\langle Z, w_{k^*}\otimes w_{k^*}\rangle - \langle P_{\widehat{\mathcal{W}}}(Z), w_{k^*}\otimes w_{k^*}\rangle\right| \\ &\qquad + \left|\langle P_{\widehat{\mathcal{W}}}(u\otimes u), w_{k^*}\otimes w_{k^*}\rangle - \langle P_{\mathcal{W}}(u\otimes u), w_{k^*}\otimes w_{k^*}\rangle\right| \\ &\leq \nu\|\sigma\|_\infty + \|Z\|\,\delta + \delta \leq \nu\|\sigma\|_\infty + 2\delta.\end{aligned} \tag{2.44}$$

Hence, $\|\sigma\|_\infty \leq \frac{2}{1-\nu}\delta$, which further implies a bound on λ_1 by

$$\begin{aligned}\lambda_1 &= \max_{\|v\|_2=1} v^\top P_{\widehat{\mathcal{W}}}(u\otimes u)v = \max_{\|v\|_2=1}\sum_{k=1}^m \sigma_k\langle \widehat{W}_k, v\otimes v\rangle \\ &\leq \max_{\|v\|_2=1}\sum_{k=1}^m \sigma_k\langle w_k, v\rangle^2 + \max_{\|v\|_2=1}\sum_{k=1}^m \sigma_k\langle \widehat{W}_k - w_k\otimes w_k, v\otimes v\rangle \\ &\leq \|\sigma\|_\infty(1+\nu) + \max_{\|v\|_2=1}\langle P_{\widehat{\mathcal{W}}}(Z) - Z, v\otimes v\rangle \leq 2\frac{1+\nu}{1-\nu}\delta + \delta \leq 3\frac{1+\nu}{1-\nu}\delta.\end{aligned}$$

This contradicts the assumption of the statement and, therefore, we must have $\sigma_{k^*} > 0$. For the estimate (2.42) we can reuse parts of (2.44) to obtain

$$\left|\sigma_{k^*} - \langle u, w_{k^*}\rangle^2\right| \leq \|\sigma\|_\infty\nu + 2\delta = \sigma_{k^*}\nu + 2\delta,$$

which implies

$$(1-\nu)\sigma_{k^*} \leq \langle u, w_{k^*}\rangle^2 + 2\delta.$$

□

Proof of Theorem 2.7. Consider a constrained local maximizer $u \in \mathbb{S}^{D-1}$ of (2.30) and the representation of $P_{\widehat{\mathcal{W}}}(u\otimes u) = \sum_{k=1}^m \sigma_k\widehat{W}_k$ given by the basis expansion induced by $P_{\widehat{\mathcal{W}}}$ and assume w.l.o.g. the coefficients are ordered according to $\sigma_1 \geq \ldots \geq \sigma_m$. Furthermore, denote by $Z = \sum_{k=1}^m \sigma_k w_k\otimes w_k$ the unique element in $\mathcal{W}$ with $P_{\widehat{\mathcal{W}}}(Z) = P_{\widehat{\mathcal{W}}}(u\otimes u)$. The proof of Theorem 2.7 leverages the second-order optimality condition (2.39) for $v = w_{k^*}$, where k^* is any index with $\|\sigma\|_\infty = |\sigma_{k^*}|$, to construct a quadratic inequality for $\langle w_{k^*}, u\rangle^2$, which can only be satisfied by $\langle w_{k^*}, u\rangle^2 \approx 1$ or $\langle w_{k^*}, u\rangle^2 \approx 0$. These inequalities, combined with the fact that $\|\sigma\|_\infty = |\sigma_{k^*}|$, imply $\lambda_1 := \lambda_1(P_{\widehat{\mathcal{W}}}(u\otimes u)) \approx 1$ or $\lambda_1 \approx 0$. By the optimality conditions in Lemma 2.16, we have $\Phi_{\widehat{\mathcal{W}}}(u) = \lambda_1$ and thus the same bounds are transferred to the objective value. Let k^* be any index with $\|\sigma\|_\infty = |\sigma_{k^*}|$. We first notice that the statement is trivially true whenever

$$\lambda_1 \leq 3\frac{1+\nu}{1-\nu}\delta \qquad \text{or} \qquad \lambda_1 \geq 1-\delta,$$

which implies we can concentrate on the cases $1-\delta > \lambda_1 \geq 3\frac{1+\nu}{1-\nu}\delta$ in the following. Under this condition, Lemma 2.18 implies $|\sigma_{k^*}| = \sigma_{k^*} = \max_{k\in[m]}\sigma_k = \sigma_1$ (note the ordering $\sigma_1 \geq \ldots \geq \sigma_K$). Furthermore, using Lemma 2.18 and additionally the frame condition (2.10), we can estimate λ_1 in terms of $\langle u, w_1\rangle^2$ according to

$$\begin{aligned}\lambda_1 = u^\top P_{\widehat{\mathcal{W}}}(u\otimes u)u = \langle Z, u\otimes u\rangle + \langle P_{\widehat{\mathcal{W}}}(Z) - Z, u\otimes u\rangle &\leq \sum_{k=1}^m \sigma_k\langle w_k, u\rangle^2 + \left\|P_{\widehat{\mathcal{W}}}(Z) - Z\right\| \\ \leq \sigma_1(1+\nu) + \|Z\|_2\,\delta \leq \frac{1+\nu}{1-\nu}\langle u, w_1\rangle^2 + 3\frac{1+\nu}{1-\nu}\delta,\end{aligned} \tag{2.45}$$

where we used $\|Z\|_2 \leq (1-\delta)^{-1}\lambda_1 < 1$ and $1 \leq \frac{1+\nu}{1-\nu}$ in the last inequality. Using the perturbation estimates

$$\left| \left\|P_{\widehat{\mathcal{W}}}(u \otimes w_\ell)\right\|_F^2 - \|P_{\mathcal{W}}(u \otimes w_\ell)\|_F^2 \right| = \left|\langle P_{\widehat{\mathcal{W}}}(u \otimes w_\ell) - P_{\mathcal{W}}(u \otimes w_\ell), u \otimes w_\ell \rangle\right| \leq \delta, \tag{2.46}$$

$$\left| w_\ell^\top P_{\widehat{\mathcal{W}}}(u \otimes u) w_\ell - \langle w_\ell, u\rangle^2 \right| = \left|\langle P_{\widehat{\mathcal{W}}}(u \otimes u) - P_{\mathcal{W}}(u \otimes u), w_\ell \otimes w_\ell \rangle\right| \leq \delta, \tag{2.47}$$

which hold for any $\ell \in [m]$, the optimality condition (2.39) with $v = w_1$ implies

$$\begin{aligned}\lambda_1 &\geq 2\left\|P_{\widehat{\mathcal{W}}}(u \otimes w_1)\right\|_F^2 - 2\lambda_1\langle w_1, u\rangle^2 + w_1^\top P_{\widehat{\mathcal{W}}}(u \otimes u) w_1 \\ &\geq 2\|P_{\mathcal{W}}(u \otimes w_1)\|_F^2 - 2\lambda_1\langle w_1, u\rangle^2 + \langle w_1, u\rangle^2 - 2\delta.\end{aligned}$$

Furthermore, by $\|P_{\mathcal{W}}(u \otimes w_1)\|_F^2 \geq \|P_{\mathcal{W}}(u \otimes w_1)\|^2 \geq \langle u, w_1\rangle^2$ and the bound (2.45) for λ_1, this becomes

$$\frac{1+\nu}{1-\nu}\langle u, w_1\rangle^2 \geq 3\langle u, w_1\rangle^2 - 2\frac{1+\nu}{1-\nu}\langle u, w_1\rangle^4 - 6\frac{1+\nu}{1-\nu}\langle u, w_1\rangle^2\delta - 3\frac{1+\nu}{1-\nu}\delta - 2\delta.$$

After dividing by $\frac{1+\nu}{1-\nu}$ and simplifying the terms we obtain

$$\begin{aligned}\langle u, w_1\rangle^2 &\geq 3\frac{1-\nu}{1+\nu}\langle u, w_1\rangle^2 - 2\langle u, w_1\rangle^4 - 11\delta, \\ \text{hence} \qquad 0 &\geq \left(1 - \frac{3\nu}{1+\nu}\right)\langle u, w_1\rangle^2 - \langle u, w_1\rangle^4 - 11/2\delta \\ &= (1-c_1)\langle u, w_1\rangle^2 - \langle u, w_1\rangle^4 - c_2,\end{aligned}$$

with constants $c_1 := \frac{3\nu}{1+\nu}$ and $c_2 := 11/2\delta$. This quadratic inequality for $\langle u, w_1\rangle^2$ has the solutions

$$\langle u, w_1\rangle^2 \leq c_2 = 11/2\delta, \quad \text{and} \quad \langle u, w_1\rangle^2 \geq 1 - c_1 - c_2 - (c_1 + 2c_2)^2 \geq 1 - 2c_1 - 3c_2,$$

provided that $\delta < \frac{1}{22}(1-c_1)^2$ and $c_1 + 2c_2 < 1$ as implied by the condition (2.41) in the statement. In the first case, where $\langle u, w_1\rangle^2 \leq 11/2\delta$, the estimate for λ_1 in (2.45) gives

$$\lambda_1 \leq \frac{1+\nu}{1-\nu}\langle u, w_1\rangle^2 + 3\frac{1+\nu}{1-\nu}\delta \leq 11/2\frac{1+\nu}{1-\nu}\delta + 3\frac{1+\nu}{1-\nu}\delta \leq 9\frac{1+\nu}{1-\nu}\delta.$$

On the other hand, the second case implies $\langle u, w_1\rangle^2 \geq 1 - 2c_1 - 3c_2$ and therefore

$$\begin{aligned}\lambda_1 &\geq w_1^\top P_{\widehat{\mathcal{W}}}(u \otimes u) w_1 = w_1^\top P_{\mathcal{W}}(u \otimes u) w_1 + w_1^\top \left(P_{\widehat{\mathcal{W}}}(u \otimes u) - P_{\mathcal{W}}(u \otimes u)\right) w_1 \\ &\geq \langle u, w_1\rangle^2 - \delta \geq 1 - 2c_1 - 3c_2 - \delta \geq 1 - 6\frac{\nu}{1+\nu} - 35/2\delta.\end{aligned}$$

□

In Theorem 2.1, the distance of any $X \in \widehat{\mathcal{W}}$ to $\pm w_k \otimes w_k$ gets quantified in terms of the spectral gap $\lambda_1 - \lambda_2$. We can directly utilize the characterization of the spectrum in Lemma 2.16 in the form of the following corollary.

Corollary 2.2 ([50, Corollary 1])**.** *Assume the setting of Theorem 2.1 and let $u \in \mathbb{S}^{D-1}$ be a constrained local maximum of $\Phi_{\widehat{\mathcal{W}}}$. We have*

$$\min_{s \in \{-1,1\}} \|u - s w_1\|_2 \leq \sqrt{2}\frac{\|\sigma_{2\ldots m}\|_2\sqrt{\nu} + 2\delta}{(1-\nu)\Phi_{\widehat{\mathcal{W}}}(u) - \sqrt{\Phi_{\widehat{\mathcal{W}}}(u)(1-\Phi_{\widehat{\mathcal{W}}}(u))} - 2\delta}$$

whenever the denominator is positive.

Proof. Recall from Lemma 2.16 that the optimality conditions of $\Phi_{\widehat{\mathcal{W}}}(u)$ imply u is the most dominant eigenvector of $P_{\widehat{\mathcal{W}}}(u \otimes u)$ corresponding to eigenvalue $\lambda_1 = \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2 = \Phi_{\widehat{\mathcal{W}}}(u)$. Furthermore, the second-largest eigenvalue is bounded by

$$\lambda_2 \leq \sqrt{\sum_{i=2}^{D} \lambda_i^2} = \sqrt{\left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2 - \lambda_1^2} = \sqrt{\Phi_{\widehat{\mathcal{W}}}(u)(1 - \Phi_{\widehat{\mathcal{W}}}(u))}.$$

The remaining part of the statement follows from Theorem 2.1 with $X = P_{\widehat{\mathcal{W}}}(u \otimes u)$. □

Proof of Theorem 2.6. The theorem is a straightforward consequence of Corollary 2.2, Theorem 2.7, and Proposition 2.1 below. Namely, Proposition 2.1 implies the existence of a local maximizer u_i^* in U_i and thus the first part of the statement. By Theorem 2.7, we have $\Phi_{\widehat{\mathcal{W}}}(u) \geq 1 - 6\frac{\nu}{1+\nu} - 18\delta$, and inserting this into Corollary 2.2 yields the second part. □

2.5.4 Intermediate Discussion

Let us summarize the findings above and how they relate to Algorithm 2.2. Consider the setting of Theorem 2.6. We know that for δ, ν small enough, there will always be a local maximizer of (2.30) close to one of the original spanning elements with an objective value close to one. This addresses one of the open issues mentioned in Section 2.3.5. Furthermore, by combining **(SPM1)** and **(SPM2)** from Theorem 2.5, we know that by for any starting point $u_0 \in \mathbb{S}^{D-1}$ the SPM iteration (2.32) will converge to a stationary point. Additionally, for a full Lebesgue measure of initializations, the iteration will converge to a local maximizer of (2.32). This guarantees that repeatedly starting the SPM will compute certain local maximizers. The fact that spurious local maximizers cannot be avoided as part of the perturbed setting can be seen in the following example.

Lemma 2.19 (cf. [50, Lemma 17])**.** *Let $w \in \mathbb{S}^{D-1}$ and $\mathcal{W} = \operatorname{span}\{w \otimes w\}$. There exists a subspace $\widehat{\mathcal{W}} \subset Sym(\mathbb{R}^{D \times D})$ with $\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{2\to 2} \leq 2\sqrt{\delta}$ so that $\Phi_{\widehat{\mathcal{W}}}$ has a constrained local maximizer with objective value δ.*

Proof. Choose $u \in \mathbb{S}^{D-1}$ with $u \perp w$ and define $M = \sqrt{1-\delta} w \otimes w - \sqrt{\delta} u \otimes u$ with the corresponding subspace $\widehat{\mathcal{W}} := \operatorname{span} M$. Taking arbitrary $A \in \mathbb{R}^{D \times D}$ with $\|A\|_2 = 1$, the subspace perturbation between $\mathcal{W}$ and $\widehat{\mathcal{W}}$ can be upper bounded by

$$\begin{aligned}\left\|P_{\mathcal{W}}(A) - P_{\widehat{\mathcal{W}}}(A)\right\| &= \left\|(1 - \sqrt{1-\delta})\langle w \otimes w, A\rangle w \otimes w - \sqrt{\delta}\langle u \otimes u, A\rangle u \otimes u\right\| \\ &\leq 1 - \sqrt{1-\delta} + \sqrt{\delta} \leq 2\sqrt{\delta}.\end{aligned}$$

We now check that u is a local maximizer of $\Phi_{\widehat{\mathcal{W}}}$. The corresponding matrix appearing in $\Phi_{\widehat{\mathcal{W}}}(u)$ is

$$P_{\widehat{\mathcal{W}}}(u \otimes u) = \langle M, u \otimes u\rangle M = -\sqrt{\delta} M = -\sqrt{\delta}\sqrt{1-\delta} w \otimes w + \delta u \otimes u,$$

which shows that u is an eigenvector of $P_{\widehat{\mathcal{W}}}(u \otimes u)$ to eigenvalue $\delta = \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2$. In other words, u satisfies (2.37) and is thus a stationary point. Taking now any $q \perp u$, we have

$$\begin{aligned}2\left\|P_{\widehat{\mathcal{W}}}(u \otimes q)\right\|_F^2 + q^\top P_{\widehat{\mathcal{W}}}(u \otimes u) q &= 2\langle M, u \otimes q\rangle^2 - \sqrt{\delta}\sqrt{1-\delta}\langle w, q\rangle^2 \\ &= -\sqrt{\delta}\sqrt{1-\delta}\langle w, q\rangle^2 < \delta,\end{aligned}$$

which shows that u also satisfies constrained second-order optimality as in (2.16). Hence, u is a local maximizer with $\Phi_{\widehat{\mathcal{W}}}(u) = \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2 = \delta$. □

To filter spurious local maxima, Algorithm 2.2 filters out all vectors which do not belong to a certain level set $\{u \in \mathbb{S}^{D-1} | \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2 \geq \beta\}$ of the underlying objective. The recent result in [76], extending upon the result in Theorem 2.6, does provide a threshold on the objective value, which makes this selection precise.

Theorem 2.8 (cf. [76, Theorem 7]). *Let $w_1, \ldots, w_m \in \mathbb{S}^{D-1}$ and $\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \ldots, w_m \otimes w_m\}$. Consider $\widehat{\mathcal{W}} \subset \operatorname{Sym}(\mathbb{R}^{D\times D})$ with $\delta = \|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F}$. If the vectors $w_1, \ldots, w_m$ fulfill the incoherence condition* (2.10)*, such that $44\delta + 60\nu - 120\delta\nu < 1$, then* (2.30) *has exactly $2m$ second-order critical points in the super-level set where*

$$\Phi_{\widehat{\mathcal{W}}}(u) \geq \frac{43 - 60\nu}{1 - 60\nu}\delta. \tag{2.48}$$

Each of these critical points is a local maximizer of (2.30)*. Furthermore, for each such point $u^* \in \mathbb{S}^{D-1}$, there exists one $k \in [m]$ such that*

$$\min_{s\in\{-1,+1\}} \|u^* - s w_k\| \leq \sqrt{\delta}.$$

Proof. The statement follows from [76, Theorem 7] by setting $n = 2$, $\rho_2 = \nu$. □

Hence, by the statement above, there exists a constant $C(\nu) > 0$ depending only on ν, such that all local maximizers with objective values $\Phi_{\widehat{\mathcal{W}}}(u) > C(\nu)\delta$ are $\mathcal{O}(\delta)$ close to the components $\pm w_k, k \in [m]$ of the original spanning elements (cf. [76, Remark 8]). This implies that for sufficiently small ν, δ, the threshold β in Algorithm 2.2 can typically be chosen as constant (for instance, $\beta = 1/2$). Finally, let us note that Chapter 4 will include two numerical sections where Algorithm 2.2 is applied successfully in the regime $m < 2D$.

2.6 Extension of SPM to the overcomplete regime

The analysis of Algorithm 2.2 in the preceding sections only applies to the case $m < 2D$. One reason for that is that the incoherence condition (2.10) in Section 2.1.1 is not applicable once the number of spanning elements $w_1 \otimes w_1, \ldots, w_m \otimes w_m$ exceeds $\mathcal{O}(D)$. To address the regime $m = o(D^2)$, we will now introduce a different set of incoherence conditions, which are inspired by properties of isotropic random vectors [76]. Under these assumptions, we state an additional result form [76], which gives *average-case guarantees* for Algorithm 2.2 that hold for the highly overcomplete regime $D < m < o(D^2)$ (see Theorem 2.10). The extension to the overcomplete regime was fundamental for the work [51], which will be discussed in Chapter 3.

2.6.1 Incoherence in the overcomplete setting

In this section, we present a way to quantify the incoherence of a set of unit vectors $w_1, \ldots, w_m$ and their outer products (this includes the rank-one matrices $w_k \otimes w_k$ but also more general rank-one tensors $w_k^{\otimes n}$) by entries and the spectrum of the associated *Gramian matrix*. The Gramian matrix G associated with a set of vectors $w_1, \ldots, w_m \in \mathbb{R}^D$ is a symmetric matrix with entries given by the inner products $G_{ij} = \langle w_i, w_j \rangle$, i.e., we have

$$G = \begin{pmatrix} \langle w_1, w_1\rangle & \ldots & \langle w_1, w_m\rangle \\ \vdots & \ddots & \vdots \\ \langle w_m, w_1\rangle & \ldots & \langle w_m, w_m\rangle \end{pmatrix} \in \mathbb{R}^{m\times m}.$$

This can easily be extended to matrices or higher-order tensors of the form $w_1^{\otimes n}, \dots, w_m^{\otimes n}$ which are associated with the Gramian matrix G_n given by the entries

$$(G_n)_{ij} = \langle w_i^{\otimes n}, w_j^{\otimes n} \rangle = \langle w_i, w_j \rangle^n, \quad n \in \mathbb{N}, i, j \in [m].$$

The spectrum of the Gramian G is connected to the frame bound (2.10) of Section 2.1.2: Recall that the condition in (2.10) is a bound on the sum of the squared inner products $\langle x, w_k \rangle^2$. Let us denote by W the matrix with columns given by $w_1, \dots, w_m$, i.e., $W = (w_1 | \dots | w_m) \in \mathbb{R}^{D \times m}$. For any $x \in \mathbb{R}^D$, we can then write

$$\sum_{k=1}^m \langle x, w_k \rangle^2 = \sum_{k=1}^m \langle x \otimes x, w_k \otimes w_k \rangle = x^\top \left(\sum_{k=1}^m w_k \otimes w_k \right) x = x^\top W W^\top x.$$

As seen directly from the identity above, the matrix $WW^\top \in \mathbb{R}^{D \times D}$ is symmetric and positive semidefinite since $x^\top WW^\top x \geq 0$ for all $x \in \mathbb{R}^D$. Therefore, $WW^\top$ has a spectral decomposition with non-negative eigenvalues, which can be arranged in descending order

$$\lambda_{\max}(WW^\top) = \lambda_1(WW^\top) \geq \dots \geq \lambda_D(WW^\top) = \lambda_{\min}(WW^\top) \geq 0.$$

Elementary linear algebra yields the bound

$$0 \leq \lambda_D(WW^\top) \|x\|_2^2 \leq \sum_{k=1}^m \langle x, w_k \rangle^2 \leq \lambda_1(WW^\top) \|x\|_2^2. \tag{2.49}$$

Let us now assume an overcomplete setting with $m > D$. Due to $G_1 = W^\top W$, we can rewrite (2.49) using the Gramian matrix such that

$$0 \leq \lambda_D(G_1) \|x\|_2^2 \leq \sum_{k=1}^m \langle x, w_k \rangle^2 \leq \lambda_1(G_1) \|x\|_2^2,$$

which closely resembles the expression in (2.10). If (2.10) holds for $\nu < 1$, then

$$\lambda_D(G_1) \|x\|_2^2 \leq (1 - \nu) \leq (1 + \nu) \leq \lambda_1(G_1).$$

The right-hand side of the last equation is what controls the overall separation of the vectors $w_1, \dots, w_m$ since $\nu \leq \lambda_1(G_1) - 1$, and in that sense, we can interpret $\lambda_1(G_1) = \|G_1\|$ as a measure of the incoherence of the vectors $w_1, \dots, w_m$. Note that we have $\lambda_1(G_1) = \|G_1\|$ because the Gramian matrix is always positive semidefinite. The characterization of incoherence can be generalized to higher-order outer products (tensors) $(w_k^{\otimes n})_{k \in [m]}$.

Lemma 2.20 (cf. [51, Lemma 16])**.** *Let $w_k \in \mathbb{R}^D$ for $k = 1, \dots, m$, and denote by $G_n \in \mathbb{R}^{m \times m}$ the Gramian matrix associated with $(w_k^{\otimes n})_{k \in [m]}$, which is given by $(G_n)_{k\ell} = \langle w_k, w_\ell \rangle^n$. Then, for any n-mode tensor $T \in \bigotimes_{k=1}^n \mathbb{R}^D$, we have*

$$\sum_{k=1}^m \langle T, w_k^{\otimes n} \rangle^2 \leq \lambda_1(G_n) \|T\|_F^2, \tag{2.50}$$

where $\lambda_1(G_n)$ denotes the largest eigenvalue of G_n.

Proof. First, note that we can express the Frobenius inner product as an ordinary inner product over $\mathbb{R}^{D^n}$ with the help of the $\mathrm{vec}(\cdot)$ operator, since $\langle T, w_k^{\otimes n} \rangle = \langle \mathrm{vec}(T), \mathrm{vec}(w_k^{\otimes n}) \rangle$. Let us denote

$$W_n := \left(\mathrm{vec}(w_1^{\otimes n}) \middle| \dots \middle| \mathrm{vec}(w_m^{\otimes n}) \right) \in \mathbb{R}^{D^n \times m}. \tag{2.51}$$

Then, the following chain of inequalities holds

$$\begin{aligned}\sum_{k=1}^{m}\langle T, w_k^{\otimes n}\rangle^2 &= \sum_{k=1}^{m}\langle \operatorname{vec}(T), \operatorname{vec}(w_k^{\otimes n})\rangle^2 \\ &= \sum_{k=1}^{m}\operatorname{vec}(T)^\top \operatorname{vec}(w_k^{\otimes n})\operatorname{vec}(w_k^{\otimes n})^\top \operatorname{vec}(T) \\ &= \operatorname{vec}(T)^\top W_n W_n^\top \operatorname{vec}(T) \\ &\leq \|W_n^\top W_n\| \cdot \|\operatorname{vec}(T)\|_2^2 = \|W_n^\top W_n\| \|T\|_F^2.\end{aligned}$$

Since $\|W_n^\top W_n\| = \|G_n\| = \lambda_1(G_n)$, this finishes the proof. □

Hence, the spectral norm of the Gramian provides a bound on $\sup_{\|T\|_F=1}\sum_{k=1}^{m}\langle T, w_k^{\otimes n}\rangle^2$ that is similar to our frame bound in (2.10). We already mentioned, that the incoherence between $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$ increases with n when the components $(w_k)_{k\in[m]}$ have unit norm. This can be seen from the Gramian matrix since the off-diagonal values G_n will vanish for increasing n assuming $|\langle w_k, w_\ell\rangle| \neq 1$ for all $k \neq \ell$. Furthermore, the characterization of incoherence by Gramian matrices encodes another important property of the ensemble $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$. More precisely, it is easy to prove that the elements $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$ are independent if and only if their Gramian matrix is positive definite. This relates to our prior result Lemma 2.2.

Lemma 2.21. *Let $w_1, \ldots, w_m \in \mathbb{S}^{D-1}$ and $n \in \mathbb{N}$, then the tensors $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$ are linearly independent if and only if their Gramian matrix G_n is positive definite.*

Proof. Denote by W_n the matrix in (2.51). First, note that $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$ are independent if and only if $\operatorname{vec}(w_1^{\otimes n}), \ldots, \operatorname{vec}(w_m^{\otimes n})$ are independent. Consider any vector $\sigma \in \mathbb{R}^m \setminus \{0\}$, then

$$\sum_{k=1}^{m}\sigma_k \operatorname{vec}(w_k^{\otimes n}) = W_n\sigma.$$

Now we can write $(W_n\sigma)^\top(W_n\sigma) = \sigma^\top G_n \sigma$. If G_n is positive definite, then $W_n\sigma \neq 0$ for all $\sigma \in \mathbb{R}^m \setminus \{0\}$. Thus, $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$ must be linearly independent. Likewise, if $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$ are linearly independent, then $W_n\sigma \neq 0$, and therefore $\sigma^\top G_n \sigma = \|W_n\sigma\|_2^2 > 0$ for all $\sigma \in \mathbb{R}^m \setminus \{0\}$. The last part implies $\lambda_{\min}(G_n) > 0$. □

Building upon this result, we can prove that linear independence will always persist when we consider higher-order tensors.

Lemma 2.22 (cf. [51, Lemma 2])**.** *Consider tensors $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$, where $w_k \in \mathbb{S}^{D-1}$ for all $k \in [m]$ and $n \in \mathbb{N}$. If $w_1^{\otimes n}, \ldots, w_m^{\otimes n}$ is linearly independent, then $w_1^{\otimes n'}, \ldots, w_m^{\otimes n'}$ are linearly independent for all $n' > n$. In particular, the minimal eigenvalues of the respective Gramian matrices fulfill the inequality*

$$\lambda_{\min}(G_{n'}) \geq \lambda_{\min}(G_n) > 0.$$

Proof. By Lemma 2.21, the Gramian G_n of the tensors $\{w_1^{\otimes n}, \ldots, w_m^{\otimes n}\}$ is a positive definite matrix. Our proof relies on the fact that higher-order Gramian matrices can be decomposed using the Hadamard product. In particular, we have $G_{n'} = G_n \odot G_{n-n'}$. Since $\lambda_{\min}(A \odot B) \geq \min_i a_{ii}\lambda_{\min}(B)$ for any pair of positive semidefinite matrices A, B, see [14, Theorem 3], we thus have

$$\lambda_{\min}(G_{n'}) = \lambda_{\min}(G_n \odot G_{n'-n}) \geq \lambda_{\min}(G_n)\min_{k\in[m]}\langle w_k, w_k\rangle^{n'-n} = \lambda_{\min}(G_n) > 0.$$

□

For unit-norm vectors $w_1, \dots, w_m$ the Gramian matrices G_n will always have ones on the diagonal. This allows us to bound the spectrum of the Gramian matrices based on the mutual incoherence $\max_{k\neq\ell} \|\langle w_k, w_\ell\rangle\|$ by using *Gershgorins circle theorem* [58]. This classical result states that every eigenvalue of a complex square matrix must lie in a disk centered at one of the diagonal elements with a radius given by the L^1-norm of the off-diagonal elements within the respective row.

Theorem 2.9 (cf. [16, Theorem 0])**.** *Let G be a complex $m \times m$ matrix with entries $G_{k\ell}$. For $k \in [m]$, let R_k be the sum of the moduli of the non-diagonal entries, i.e., let*

$$R_k = \sum_{\ell\neq k} |G_{k\ell}|.$$

Then each eigenvalue z of G lies in union of circles

$$|z - g_{kk}| \leq R_k,$$

for all $k \in [m]$.

2.6.2 Average case guarantees for SPM in the overcomplete regime.

To be able to extend their SPM analysis to the overcomplete regime $m = o(D^2)$, the authors of [76] rely on the strong incoherence present in an ensemble of vectors that can be modeled by isotropic random distributions. In Section 2.1.2, we already mentioned that isotropic random vectors in high dimension can be regarded as nearly orthogonal. For instance, two independent vectors x, y drawn independently uniformly at random from the unit sphere will fulfill $|\langle x, y\rangle| = \mathcal{O}(D^{-1/2})$ with high probability. Definition 2.3 below states three properties regarding the mutual incoherence and the Gramian matrix, which hold for up to $o(D^2)$ independent realizations of a uniform distribution on the unit sphere with high probability (cf. Proposition 2.2). Aside from enabling the main statement in this section (Theorem 2.10), the properties in Definition 2.3 will play a central role in Chapter 3. Let us first introduce a variant of the *restricted isometry property* (RIP) for matrices.

Definition 2.2 (RIP)**.** *Let $W \in \mathbb{R}^{D\times m}$, $1 \leq p \leq m$ be an integer, and $\delta \in (0, 1)$. We say that W is (p, δ)-RIP if every $D \times p$ submatrix W_p of W satisfies $\|W_p^\top W_p - \mathrm{Id}_p\| \leq \delta$.*

Definition 2.3 (Properties of isotropic random weights)**.** *Let $W := [w_1|\dots|w_m]$ and $(G_n)_{k\ell} := \langle w_k, w_\ell\rangle^n$. We define the following incoherence properties:*

(Inc1) *There exists $c_1 > 0$, depending only on δ, such that W is $(\lceil c_1 D/\log(m)\rceil, \delta)$-RIP.*

(Inc2) *There exists $c_2 > 0$, independent of m, D, so that $\max_{k\neq\ell}\langle w_k, w_\ell\rangle^2 \leq c_2 \log(m)/D$.*

(Inc3) *There exists $c_3 > 0$, independent of m, D, so that $\left\|G_n^{-1}\right\| \leq c_3$, for all $n \geq 2$.*

Let us consider unit-norm vectors $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ that fulfill **(Inc1)** - **(Inc3)**. The vectors are almost maximally separated by **(Inc1)** and **(Inc2)**. This can be seen by comparing the bound on the mutual incoherence **(Inc2)**, which states $\max_{k\neq\ell}\langle w_k, w_\ell\rangle^2 \leq c_2 \log(m)/D$, to the Welch bound (2.8) we presented in Section 2.1.2. Plugging in $m = D^2$ into (2.8) yields a *maximal mutual incoherence* bound given by

$$\max_{k\neq\ell} |\langle w_k, w_\ell\rangle| \leq \sqrt{\frac{m-D}{D(m-1)}} = \sqrt{\frac{D-1}{D^2-1}} \approx D^{-1/2}, \tag{2.52}$$

which only differs from **(Inc2)** by a constant and log-factors. The bound on the spectral norm of $\left\|G_n^{-1}\right\|$ in **(Inc3)** does imply that the Gramian matrix G_2 of the matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ is positive definite due to $\|G_2^{-1}\| = \lambda_{\min}(G_2)^{-1}$, which implies $\lambda_{\min}(G_2) > c_3^{-1}$. Thus, by Lemma 2.21 of the last section, the matrices $w_1 \otimes w_1, \dots, w_m \otimes w_m$ are linearly independent. The mutual incoherence can directly be translated to an upper bound on the spectrum of the Gramian matrices, as the following application of Gershgorins circle theorem (Theorem 2.9) shows:

Lemma 2.23 (cf. [51, Lemma 17]). *Let $w_k \in \mathbb{S}^{D-1}$ for $k = 1, \dots, m$ be unit vectors, and denote by $G_n \in \mathbb{R}^{m \times m}$ the Gramian matrix associated with $(w_k^{\otimes n})_{k \in [m]}$, which is given by $(G_n)_{k\ell} = \langle w_k, w_\ell \rangle^n$. Assume that the vectors $w_1, \dots, w_m$ fulfill **(Inc2)** of Definition 2.3, then there exists an absolute constant $C > 0$ only depending on c_2 in **(Inc2)** such that*

$$\|G_n\| \leq C \left(1 + m \left(\frac{\log m}{D}\right)^{n/2}\right).$$

Proof. The result follows directly by Theorem 2.9 since the diagonal elements must be 1 and the off-diagonal elements are bounded in absolute value by $\left(c_2 \frac{\log m}{D}\right)^{n/2}$. □

This shows that Definition 2.3 covers all necessary conditions on which we relied throughout Section 2.5. The following statement due to [76] establishes all three properties **(Inc1)** - **(Inc3)** for vectors drawn independently from the unit-sphere as long as $m = o(D^2)$.

Proposition 2.2 ([76, Proposition 13]). *Let $w_1, \dots, w_m$ be drawn independently from $\mathrm{Unif}(\mathbb{S}^{D-1})$. If $m = o(D^2)$, then, for any arbitrary constant $\delta \in (0, 1)$, there exist constants $C > 0$ and $D_0 \in \mathbb{N}$ depending only on δ such that for all $D \geq D_0$, and with probability at least*

$$1 - m^{-1} - 2\exp(-C\delta^2 D) - C\left(\frac{e \cdot D}{\sqrt{m}}\right)^{-C\sqrt{m}} \tag{2.53}$$

*conditions **(Inc1)** - **(Inc3)** hold with constants $c_2, c_3 < C$.*

Let us conclude this chapter with one of the main results from [76], which extends the guarantees for SPM and Algorithm 2.2 to the overcomplete regime.

Theorem 2.10 ([76, Theorem 16]). *Let $m, D \in \mathbb{N}$ such that $m \log^2(m) \leq D^2$. Assume $w_1, \dots, w_m$ satisfy **(Inc1)** - **(Inc3)** of Definition 2.3 for some $\delta, c_1, c_2, c_3 > 0$. Then there exists δ_0, depending only on c_2, c_3 and D_0, Δ_0, C which depend additionally on c_1, such that if $\delta < \delta_0, D > D_0$ and $\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F \to F} \leq \Delta_0$, the program* (2.30) *has exactly $2m$ second-order critical points in the superlevel set*

$$\left\{u \in \mathbb{S}^{D-1} \,\middle|\, \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2 \geq Cm \log^2(m)/D^2 + 5\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F \to F}\right\}. \tag{2.54}$$

Each of these critical points is a strict local maximizer for $\operatorname{argmax}_{u \in \mathbb{S}^{D-1}} \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2$. Furthermore, for each such point u^*, there exists a unique $k \in [m]$ such that

$$\min_{s \in \{-1,1\}} \|u^* - s w_k\|_2 \leq \sqrt{\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F \to F}}. \tag{2.55}$$

A detailed discussion of the proof is beyond the scope of this work since their proof is stated in a more general setting that has tensor decompositions in mind. For details, we refer to [76]. We defer the discussion concerning this result to the next section, where we summarize the findings of this chapter.

2.7 Conclusion

Let us now summarize our results on the recovery of a rank-one basis from a perturbed matrix space. The present chapter covers two algorithms (Algorithm 2.1 and Algorithm 2.2) to tackle the recovery of the spanning elements $w_1 \otimes w_1, \dots, w_m \otimes w_m$, or equivalently $\pm w_1, \dots, \pm w_m$ from a matrix space

$$\widehat{\mathcal{W}} \approx \mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}.$$

The first approach, is centered around the characterization of near-rank-one matrices within $\widehat{\mathcal{W}}$ given by

$$\operatorname{argmax} \|M\| \text{ s.t. } \|M\|_F \leq 1, M \in \widehat{\mathcal{W}}. \tag{2.56}$$

Our characterization in Theorem 2.2 of local maximizers of (2.56) establishes conditions under which each local maximizer of (2.14) must be close to one of the spanning elements $w_1 \otimes w_1, \dots, w_m \otimes w_m$. However, there are still some open questions (discussed as part of Section 2.3.5) within our theory regarding the computations of the local maximizers. As a reminder, our guarantees for Algorithm 2.1 in Theorem 2.4 only show that the iteration

$$M_{j+1} = F_\gamma(M_j), \quad \text{where} \quad F_\gamma(X) = \frac{P_{\widehat{\mathcal{W}}}(X + \gamma u_1(X) \otimes u_1(X))}{\left\|P_{\widehat{\mathcal{W}}}(X + \gamma u_1(X) \otimes u_1(X))\right\|_F}, \tag{2.57}$$

has increasing spectral norm and converging subsequences that converge to stationary points of (2.56). Strictly speaking, Theorem 2.4 does not guarantee convergence to local maximizers. Additionally, iteration (2.57) is expensive to compute since (i) optimization is done in matrix space and (ii) we need to compute a SVD at every iteration.

The second approach (Algorithm 2.2) we presented is based on the non-linear program

$$\max_{\|u\|=1} \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2, \tag{2.58}$$

and seeks to find the components $\pm w_1, \dots, \pm w_m \in \mathbb{S}^{D-1}$ by iterating

$$u_{j+1} = P_{\mathbb{S}^{D-1}}(u_j + 2\gamma P_{\widehat{\mathcal{W}}}(u_j \otimes u_j)u_j), \tag{2.59}$$

from random initializations $u_0 \in \mathbb{S}^{D-1}$, which is referred to as the subspace power method (SPM) [77, 76, 50]. This approach is favorable in terms of computational complexity since every iteration only relies on linear operations and a projection on the respective unit-sphere. This makes Algorithm 2.2 applicable to relatively high dimensional problems. For extensive numerical experiments with SPM, we refer to Section 3.6, Section 4.2.3, and Section 4.4. Another benefit of SPM over the method discussed in Section 2.3 is stronger theoretical guarantees associated with SPM. In particular, by [77] (see Theorem 2.5), iteration (2.59) will converge to a local maximizer of (2.58) for almost all random initializations $u_0 \in \mathbb{S}^{D-1}$ (cf. **(SPM2)**).

SPM under deterministic frame conditions. Throughout Section 2.5, we provide an extension to the original results of [77] with Theorem 2.6, proving that, under the deterministic frame condition (2.10) (implying $m = \mathcal{O}(D)$, cf. Lemma 2.3) and for sufficiently small perturbations $\delta = \|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F \to F}$, the local maximizers of (2.58) within a superlevel set of the objective (2.58) will be close to the spanning components $\pm w_1, \dots, \pm w_m$. Spurious local maximizers (i.e., local maximizers not belonging to said superlevel set) can be filtered via thresholding (cf. parameters β in Algorithm 2.2). The fact that spurious local maximizers are not a theoretical artifact and

can generally not be avoided was shown in Lemma 2.19. Theorem 2.8 from [76] provide a slightly better lower bound on the superlevel set, which shows that, for δ, ν sufficiently small, setting $\beta = \frac{43-60\nu}{1-60\nu}\delta$ will filter all spurious local maximizers in Algorithm 2.2. Note that SPM will monotonically converge according to **(SPM1)**. Hence, for δ sufficiently small, most random starting points $u_0 \in \mathbb{S}^{D-1}$ will lead to a SPM iteration that stays within the superlevel set that does not include spurious local maxima.

Average case guarantees for SPM for random isotropic components up to $m = o(D^2)$. Within the framework where the components $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ satisfy the deterministic frame condition (2.10), the number of components is limited to the linear regime, more precisely, (2.10) requires $m < 2D$. Recently, in [76], the perturbation analysis of SPM has been extended to the overcomplete regime where the number of components is allowed to scale up to $m \log^2 m = \mathcal{O}(D^2)$. We stated this result in Theorem 2.10. This statement relies on several incoherence conditions (cf. Definition 2.3) that, by Proposition 2.2, provably hold for isotropic random vectors in high dimensions such as $w_1, \dots, w_m \sim_{\text{i.i.d.}} \text{Unif}(\mathbb{S}^{D-1})$ with high probability. The subspace power method, in the overcomplete regime, benefits from the same theoretical guarantees **(SPM1)-(SPM2)** that were given in Theorem 2.5. However, one notable difference is that the cut-off point of the superlevel set, which allows us to distinguish between spurious and non-spurious local maximizers, now depends on m, D. In particular, according to (2.54) in Theorem 2.10, non-spurious maximizers are contained in the set

$$\left\{ u \in \mathbb{S}^{D-1} \,\middle|\, \left\| P_{\widehat{\mathcal{W}}}(u \otimes u) \right\|_F^2 \geq Cm \log^2(m)/D^2 + 5\delta \right\},$$

for some constant $C > 0$ depending on the incoherence of the components $w_1, \dots, w_m$. Clearly, as $m \log^2 m$ approaches D^2, the size of the superlevel set quickly shrinks. Consequently, random starting points of SPM $u_0 \sim \text{Unif}(\mathbb{S}^{D-1})$ do not lie within such a superset with high probability. This affects the choice of the threshold parameter β in Algorithm 2.2. Let us mention, however, that such an effect is partially expected due to the fact that for $m = D(D-1)/2$, the rank-one basis recovery problem becomes ill-posed since $\dim(\text{Sym}(\mathbb{R}^{D\times D})) = D(D-1)/2$ (see the related discussion in Section 2.1.1). Hence, for $m = D(D-1)/2$, every vector $u \in \mathbb{S}^{D-1}$ would be a global maximizer of (2.58) since $u \otimes u$ is rank-one and contained in $\mathcal{W}$. Let us note that we manage empirically (see Figure 3.3 in Section 3.6) to recover all components $w_1, \dots, w_m$ up to the regime where $m = 2/5D^2$ for sufficiently high dimensions (e.g., $D = 50$) with a threshold of $\beta = 0.99$. This shows that, for sufficiently small δ, the filtering of spurious local maximizers is still possible close to the information theoretical limit $m = D(D-1)/2$.

Computational complexity of SPM. Assuming δ is sufficiently small and D is sufficiently large, then all repetitions of SPM where $m \log^2(m) = o(D^2)$ will compute a non-spurious local maximizer for almost every starting point $u_0 \in \mathbb{S}^{D-1}$. Every individual repetition of SPM runs in polynomial time. If $w_1, \dots, w_m$ are well separated, then we can assume that retrieving every local maximizer is almost equally likely. Hence, based on our argumentation in Remark 2.2, we expect to find all components after $\Theta(m \log m)$ repetitions of the SPM iteration. This makes the overall runtime of Algorithm 2.2 polynomial in this regime. For the regime where $m \log^2(m) = \Theta(D^2)$, there are cases where random initializations $u_0 \in \mathbb{S}^{D-1}$ lie outside the superlevel set mentioned above with high probability. In principle, this could imply that one needs an exponential number of restarts of SPM to compute non-spurious local maximizers, which would result in a non-polynomial runtime. Let us mention that we do not encounter such cases in our experiments in Section 3.6. In fact, we observe that SPM does manage to compute all $m = 2/5D^2$ components (drawn uniformly at random from the unit sphere) in dimensions $D = 35, 40, 45, 50$ in the order of seconds from $5m \log m$ repetitions. One additional

benefit of SPM is that the iteration (2.59) can easily be run in parallel and leverage computing on graphic cards: Each step of SPM performs the operation $u_j + 2\gamma P_{\widehat{\mathcal{W}}}(u_j \otimes u_j)u_j$ followed by a projection onto the unit sphere. Note that the projection $P_{\widehat{\mathcal{W}}}(u_j \otimes u_j)$ can be written as a matrix-vector product by considering the vectorized orthogonal projection instead. This requires only minor modifications of Algorithm 2.2 and will be discussed in Section 4.3.1.

Chapter 3

Efficient reconstruction of wide shallow networks

In this chapter, we consider the identification of shallow neural networks of the type

$$f : \mathbb{R}^D \to \mathbb{R}, \quad f(x) := \sum_{k=1}^{m} g(\langle w_k, x \rangle + \tau_k), \tag{3.1}$$

where $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ are unit-norm weights and $\tau_1, \dots, \tau_m \in \mathbb{R}$ are bounded shifts. We study the regime where the number of neurons m scales like $m \log^2 m = \mathcal{O}(D^2)$. As the main result of this chapter, we present a *end-to-end* recovery pipeline that comes with theoretical guarantees for smooth activations of sigmoidal type and sufficiently incoherent weights. More precisely, we assume a setting where the weights are drawn uniformly from the unit sphere and therefore fulfill the incoherence properties that were previously discussed in Section 2.6.1 with high probability. Our recovery pipeline relies on *active sampling* to approximate network derivatives (cf. Section 1.4) and is based on a decoupling of the network parameters that can be broken down into three consecutive phases: In the first phase, we recover the network weights by reducing their identification to the *rank-one basis recovery problem* that was studied in Chapter 2. This reduction was already introduced in Section 1.5. More precisely, we rely on Hessian approximations of the network to construct a matrix space

$$\widehat{\mathcal{W}} \approx \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}. \tag{3.2}$$

The weights can then be recovered up to sign by iterating SPM (cf. Algorithm 2.2 and Chapter 2). In the second phase, the signs of the weights and initial estimates for the shift parameters are computed using algebraic evaluations leveraging certain properties of the activation function (we assume g satisfies properties commonly found in sigmoidal activation functions) and higher-order directional derivatives of the network. In the last phase, the shifts are computed by empirical risk minimization via gradient descent. We present a local convergence analysis that guarantees convergence to the ground-truth biases when combined with the shift initializations of the second phase. The convergence analysis is based on a linearization argument which relies on techniques common in the *neural tangent kernel* (NTK) approach.

This chapter is based on joint work with Massimo Fornasier, Timo Klock, and Marco Mondelli that appeared as a preprint in [51]. Several of the theoretical proofs within this chapter, which originate from this work, are stated verbatim as in their underlying reference, indicated within the definition of the statement. We provide several discussions that add context to the technical statements. Notably, we also address one open problem stated in [51] and related literature [52, 54, 50]. More precisely, we prove that for non-polynomial activation functions, the Hessians

of the network do provide sufficient information for the approximation in (3.2) whenever the weights are sufficiently incoherent.

3.1 Introduction and preliminaries

Chapter 1 includes a discussion that outlines how the identification of neural networks and the analysis of the teacher-student model provide insight into the generalization capabilities of neural networks. However, most known results in the teacher-student setting are limited to shallow neural networks where the number of neurons m does not exceed the number of inputs D or depends at most linearly on D, i.e., we have $m = \mathcal{O}(D)$. Hence, a natural next step is an extension to the overcomplete regime where $D < m < D^2$. Recall that most existing approaches to the teacher-student problem mentioned in Chapter 1 rely on tensor decompositions. Tensor decompositions are either used to compute the network weights directly [8], or to find suitable initialization for SGD. Tensor decomposition in the regime $D^{3/2} < m < D^2$ is known to be computationally hard [93]. Hence, it remains unclear whether existing results based on tensor decompositions can be applied to efficiently recover networks in the regime $D^{3/2} < m < D^2$. Additionally, the numerical experiments in Section 1.3.1 give evidence that the recovery of all network parameters via SGD is not straightforward whenever $D^{3/2} < m < D^2$.
Another unexplored area is the identification of networks with shifts. What makes neural networks theoretically interesting is that they act as universal approximators (cf. Section 1.1.2). However, to the best of our knowledge, the universal approximation theorem does only hold for networks with shifts. A common technique is integrating shifts into the network weights by adding an additional constant input. For example, consider a neuron with activation g, weight w and shift τ, then

$$g(\langle x, w\rangle + \tau) = g\left(\langle \begin{pmatrix} x \\ 1 \end{pmatrix}, \begin{pmatrix} w \\ \tau \end{pmatrix} \rangle\right).$$

At first, this suggests that the recovery of shifts is a by-product of weight recovery. However, note that proving recovery guarantees for the network weights typically build upon distributional assumptions on the network weights. A widely used assumption is that the network weights can be modeled as independent samples of an isotropic random distribution. Absorbing the shifts into the weights *will break* many of the desirable properties connected with those distributional assumptions (e.g., a loss of mutual incoherence).

In this chapter, we will address the recovery of a shallow neural network

$$f : \mathbb{R}^D \to \mathbb{R}, \quad f(x) := \sum_{k=1}^{m} g(\langle w_k, x\rangle + \tau_k), \tag{3.3}$$

with sufficiently smooth non-polynomial activations, bounded shifts, and incoherent weights $w_1, \dots, w_m \in \mathrm{S}^{D-1}$ in the overcomplete regime. Under the setting introduced in the next section, we give guarantees for the recovery of the weights and shifts that can be informally summarized as follows:

Theorem 3.1 (Informal)**.** *Let f be the shallow network as in* (3.3) *with D inputs and m neurons such that $m \log^2 m = O(D^2)$. Then, for sufficiently large D, a constructive algorithm exists recovering all weights and shifts of the network with high probability from $O(Dm^2 \log^2 m)$ network queries.*

Following the steps discussed in Section 1.5, our recovery pipeline finds weight approximations by solving the problem studied throughout Chapter 2 via SPM (cf. Algorithm 2.2). The weight recovery is the determining factor for the computational complexity of the pipeline

that will be presented. For reasons discussed in Section 2.7, that makes the computational complexity of our overall pipeline polynomial in m and D in the regime $m = O(D)$. In the highly overcomplete regime, where $m \log^2 m \lesssim D^2$, then global convergence of Algorithm 2.2 is guaranteed by Theorem 2.10 assuming that spurious local maximizers are filtered (this relates to β in Algorithm 2.2) as in (2.54). Let us mention that we empirically observe an efficient recovery of networks close up to the information-theoretic limit $m = 1/2(D-1)D$ (see Section 3.6). This suggests that the analysis of Algorithm 2.2 might not be optimal. For more details on the computational complexity of the SPM approach, we refer to Chapter 2 (in particular Section 2.7). Before we describe our recovery pipeline and state our main theoretical result (Theorem 3.2), we will first introduce our problem setting throughout the upcoming section.

3.1.1 Problem setting: Shallow neural network model

Let us start by formally introducing our problem setting. The upcoming conditions are separated into two categories. First, we describe the class of networks considered in the technical sections. This includes assumptions about the network parameters as well as conditions concerning the activation function. Second, we state which information is accessible to recover the network parameters.

Network model. We consider planted shallow neural networks $f(x) = \sum_{k=1}^m g(\langle w_k, x\rangle + \tau_k)$ with bounded shifts $\tau_1, \dots \tau_m \in [-\tau_\infty, +\tau_\infty]$ for some known $\tau_\infty > 0$ and assume, that the network weights are drawn uniformly at random from the unit-sphere

$$w_1, \dots, w_m \sim_{\text{i.i.d.}} \text{Unif}(\mathbb{S}^{D-1}).$$

Let us note that the size of the interval $[-\tau_\infty, +\tau_\infty]$ only depends on the activation function g but not on m, D. Assuming that the weights can be modeled by isotropic random vectors of fixed size simulated generic unit-vectors for which we can expect strong mutual incoherence: Recall that by Proposition 2.2 in Section 2.6.1, ensembles of isotropic random vectors fulfill **(Inc1)** - **(Inc3)** of Definition 2.3 with high probability. In particular, **(Inc2)** implies the existence of a constant $C > 0$ such that the mutual incoherence of $w_1, \dots, w_m$ is bound by

$$\max_{k \neq \ell} \langle w_k, w_\ell \rangle^2 \leq C \log(m)/D.$$

This allows us to regard the weights as almost orthogonal for sufficiently high dimensions. Additionally, by Lemma 2.21 the property **(Inc3)** implies the linear independence of the systems $\{w_k^{\otimes n} | k \in [m]\}$ for all $n \geq 2$. We refer to Section 2.6.1 for more details. Aside from these assumptions, we also require that the shallow neural network f satisfies the following points:

Assumption 3.1 (Shallow network model (cf. [51]))**.**

(SNM1) *The activation function satisfies $g \in \mathcal{C}^3(\mathbb{R})$ and its first three derivatives are bounded. We denote*

$$\kappa := \max_{n \in [3]} \left\| g^{(n)} \right\|_\infty < \infty. \tag{3.4}$$

Furthermore, the second derivative $g^{(2)}$ is strictly monotonic on the open interval $(-\tau_\infty, +\tau_\infty)$, $g^{(1)}$ is strictly positive or negative on $(-\tau_\infty, +\tau_\infty)$ and there exists a sign $s \in \{-1, +1\}$ such that for all $\tau \in [-\tau_\infty, +\tau_\infty]$ we have

$$s = \text{sgn}\left(\int_{\mathbb{R}} g^{(1)}(t + \tau) \exp(-t^2/2) dt \right).$$

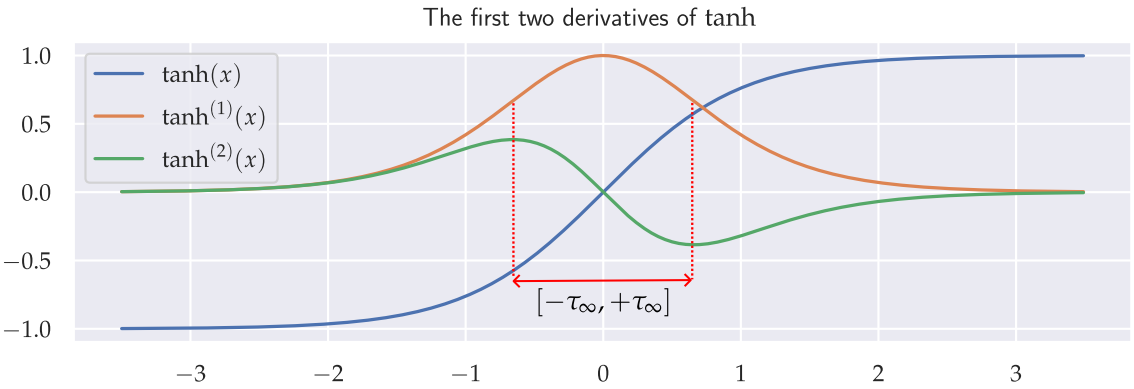

Figure 3.1: Interval of recoverable shifts visualized for the $\tanh(\cdot)$-activation.

(SNM2) *Denote by P_n the set of polynomials of degree n or less. We assume that $g, g^{(1)}, g^{(2)} \notin P_3$. Furthermore, we assume that $g_\tau(\cdot) := g(\cdot + \tau) \in L_2(\mathbb{R}, w_G)$ for all $\tau \in [-\tau_\infty, +\tau_\infty]$, where $w_G(t) := \exp(-t^2/2)$, i.e., we have*

$$\int_{\mathbb{R}} g(t)^2 \exp(-t^2/2) dt < \infty \quad \text{for all } \tau \in [-\tau_\infty, +\tau_\infty].$$

(SNM3) *The Hessians of f have sufficient information for weight recovery, i.e., there exists an $\alpha > 0$ such that*

$$\lambda_m \left(\mathbb{E}_{X \sim \mathcal{N}(0, \mathrm{Id})} [\mathrm{vec}(\nabla^2 f(X))^{\otimes 2}] \right) \geq \alpha > 0. \tag{3.5}$$

*Note: This assumption is directly implied by **(SNM1)-(SNM2)** under sufficiently incoherent weights (cf. Remark 3.1 and Theorem 3.5.)*

Let us first discuss the type of activation functions that satisfy **(SNM1)** and **(SNM2)**. Condition **(SNM1)** unites several properties that are fulfilled by activations of sigmoidal type (see Example 3.1 below): A certain degree of smoothness is necessary for our approach, as we will consider higher-order derivatives of the network. The upper bound on the derivatives in (3.4) allows us to rely on the Lipschitz continuity of g and its derivatives to simplify many of the technical statements. We require the strict monotonicity of $g^{(2)}$ on the interval $(-\tau_\infty, +\tau_\infty)$ to be able to infer the shifts $\tau \in (-\tau_\infty, +\tau_\infty)$ from the value $g^{(2)}(\tau)$. Empirically, we do not observe that this assumption is necessary. However, this assumption allows us to compute an accurate initial estimate of the shifts, which is necessary as our local convergence analysis of GD comes with a small convergence radius. Therefore, it remains necessary as part of our theoretical analysis. The two assumptions on $g^{(1)}$ in **(SNM1)** allow us to simplify several technical statements involving Hermite coefficients (cf. proofs throughout Section 3.4.6 and Section 3.4.4 for a definition of Hermite coefficients) and could potentially be dropped at the price of more complicated expressions in our guarantees. Lastly, condition **(SNM2)** implies several properties which characterize the Hermite expansion (see Definition 3.2) of the activation function and its derivatives. More precisely, we require that g belongs to the L_2 space weighted by the Gaussian kernel $w_G(t) := \exp(-\frac{t^2}{2})$ and therefore admits a Hermite expansion. Furthermore, assuming that $g^{(1)}, g^{(2)}$ cannot be represented by polynomials of degree three or less implies that the Hermite coefficients do not vanish too fast, which we exploit to prove lower bounds on the second moment matrices $\mathbb{E}_{X \sim \mathcal{N}(0, \mathrm{Id})} [\mathrm{vec}(\nabla^n f(X))^{\otimes 2}]$, where $n = 1, 2$ (cf. Section 3.4.5).

Example 3.1. *Classical examples of activation functions that fulfill **(SNM1)-(SNM2)** are sigmoidal functions. These functions are generally smooth and bounded functions with bounded derivatives. In most cases, these functions are also strictly monotonic, implying that there is no sign-change of $g^{(1)}$.*

For typical representatives of this class, such as the hyperbolic tangent $g(x) = \tanh(x)$ or the sigmoid function $g(x) = 1/(1+\exp(-x))$, one can also check that the remaining conditions are satisfied for a suitable choice of τ_∞. More precisely, for the invertibility of $g^{(2)}$ we need to choose $\tau_\infty \approx 0.6$ for $g(x) = \tanh(x)$ and $\tau_\infty \approx 1.5$ for the sigmoidal function $g(x) = 1/(1+\exp(-x))$.

To provide more context on **(SNM3)**, let us consider a network of the form (3.1). Then the Hessian takes the form

$$\nabla^2 f(x) = \sum_{k=1}^{m} g^{(2)}(\langle x, w_k\rangle + \tau_k) w_k \otimes w_k.$$

Condition **(SNM3)** guarantees that sampling sufficiently many Hessians of the network at random inputs provides enough information to approximate the space

$$\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\} \subset \operatorname{Sym}(\mathbb{R}^{D\times D}).$$

This reduces the weight recovery to the rank-one basis recovery problem studied throughout Chapter 2. Condition **(SNM3)** is common in the related literature [52, 54, 50, 8].

Remark 3.1. *Assumption **(SNM3)** is, in fact, a direct consequence of **(SNM1)-(SNM2)** under fairly mild conditions. This implication is proven in Theorem 3.5 and addresses one open problem stated in [54] and related literature [52, 50]. Notably, Theorem 3.5 provides a lower bound α* that does not depend negatively *on D, m.*

Accessible information. The next assumption summarizes what information about the network can be accessed and reflects the points discussed previously in Section 1.4.

Assumption 3.2 (Network queries allow derivative approxiamtion)**.**

(G4.1) *We can query the network f and the activation g at any point without noise, and the number of neurons m is known.*

(G4.2) *We assume access to a numerical differentiation method, denoted by $\Delta^n[\cdot]$, that computes the derivatives for $n = 1, 2, 3$ up to accuracy $\epsilon > 0$. To be more precise, we require that the derivatives of g with respect to a vector input $x \in \mathbb{R}^D$ fulfill*

$$\left\| \nabla^n g(w^\top x) - \Delta^n[g(w^\top x)] \right\|_F \leq C_\Delta \left\| w^{\otimes n} \right\|_F \epsilon, \tag{3.6}$$

where C_Δ is a universal constant only depending on the activation through κ (see (3.4)*). Furthermore, for any $b, t_0 \in \mathbb{R}$ the derivatives of $t \mapsto g(b\,t)$ can be approximated as*

$$\left| \frac{d^n}{dt^n} g(b\cdot t)\Big|_{t=t_0} - \Delta^n[g(b\cdot)](t_0) \right| \leq C_\Delta b^{n+2} \epsilon. \tag{3.7}$$

We also assume that the numerical differentiation method is linear, i.e., it satisfies

$$\Delta^n[a\cdot g + h] = a\cdot \Delta^n[g] + \Delta^n[h], \tag{3.8}$$

for any functions g, h and scalar $a \in \mathbb{R}$. Finally, the numerical differentiation algorithm requires a number of queries proportional to the dimension of the approximated derivative, i.e., $\mathcal{O}(1)$ for partial derivatives and $\mathcal{O}(D^n)$ for n-th order derivative tensors.

In summary, the preceding conditions guarantee that we can approximate network derivatives up to an error that scales like ϵ times a factor depending on the dimension of the derivative. We will not further specify the numerical differentiation method. All the properties in **(G4.2)** are fulfilled by a standard central finite difference scheme.

3.1.2 Summary: Overview and main result

Consider a shallow network f satisfying the conditions of the preceding section. The recovery of the weights $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ and shifts $\tau_1, \dots, \tau_m$ is tackled in a three-step procedure.

Weight recovery (Section 3.2). As already outlined in the introduction, the first step of our network identification is the recovery of the weights. Following ideas from [54, 52, 50] (cf. Section 1.5), we tackle this problem by reducing it to the rank-one basis recovery problem of Chapter 2: Here, the approximation

$$\widehat{\mathcal{W}} \approx \mathcal{W} := \operatorname{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\},$$

is computed by selecting the m-th singular subspace of the space spanned by approximations of network Hessians, i.e., we have

$$\widehat{\mathcal{W}} = \operatorname{span}_m \left\{\Delta^2 f(x_1), \dots, \Delta^2 f(x_{N_h})\right\}.$$

In Section 3.2, we show that, for N_h sufficiently large and $x_1, \dots, x_{N_h} \sim_{\text{i.i.d.}} \mathcal{N}(0, \mathrm{Id}_D)$, this leads to an approximating space $\widehat{\mathcal{W}}$ such that

$$\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F\to F} \lesssim \sqrt{m/\alpha} \cdot \epsilon \quad \text{w.h.p.}$$

Notably, as the parameter ϵ is under our control, the approximation error can be made sufficiently small. We then recover the individual weights from $\mathcal{W}$ by using Algorithm 2.2, which repeatedly samples independent starting points $u_0 \in \mathbb{S}^{D-1}$ and iterates the SPM iteration

$$u_j = P_{\mathbb{S}^{D-1}}(u_{j-1} + 2\gamma P_{\widehat{\mathcal{W}}}(u_{j-1} \otimes u_{j-1})u_{j-1}), \qquad j \in \mathbb{N}_{>0},$$

to compute local maximizers of the objective $u \mapsto \|P_{\widehat{\mathcal{W}}}(u \otimes u)\|_F^2$. Algorithm 2.2 comes with guarantees up to the regime $m \log^2 m = O(D)$, more precisely, Theorem 2.10. We combine these results in Theorem 3.3, which states that, under suitable conditions, this approach eventually leads to weight approximations $\hat{w}_1, \dots, \hat{w}_m \in \mathbb{S}^{D-1}$ such that

$$\min_{s\in\{-1,+1\}} \|w_k - s\hat{w}_k\|_2 \lesssim (m/\alpha)^{1/4}\epsilon^{1/2} \quad \text{for all } k \in [m],$$

where $\alpha > 0$ is taken from the lower bound in **(SNM3)** and can be regarded as a constant independent of m, D (cf. Theorem 3.5). The overall weight recovery is summarized in Algorithm 3.2 and constitutes the first step in Algorithm 3.1.

Recovery of the signs and initial estimation of the shifts (Section 3.3). The second step assumes that we are given weight approximations $\hat{w}_1 \approx s_1 w_1, \dots, \hat{w}_m \approx s_m w_1$ where $s_1, \dots, s_m \in \{-1, +1\}$ are unknown signs. These weight approximations can be interpreted as downstream results of the weight recovery. Furthermore, we assume that the original weights, as well as the approximated network weights, are incoherent according to **(Inc2)-(Inc3)** of Definition 2.3. This assumption is justified because the incoherence of $(w_k)_{k\in[m]}$ is needed for the guarantees concerning the weight recovery (and therefore, it is already incorporated in Theorem 3.3), whereas the incoherence of the ensemble $(\hat{w}_k)_{k\in[m]}$ can be enforced by choosing ϵ sufficiently small (cf. Lemma 3.2). Based on these assumptions, we recover the original signs $s_1, \dots, s_m$ and an initial estimate of the shifts $\tau_1, \dots, \tau_m \in [-\tau_\infty, +\tau_\infty]$ exploiting a linearization argument and the (local) invertibility of $g^{(2)}$ (cf. **(SNM1)**): We start by retrieving an estimate to the vectors $\mathcal{C}_2, \mathcal{C}_3$, which are defined as

$$\mathcal{C}_{n,k} := s_k^n g^{(n)}(\tau_k), \quad \text{for } k \in [m], \quad n \in \{2, 3\}.$$

We show that $\mathcal{C}_2, \mathcal{C}_3$ reveal the signs and shifts assuming condition **(SNM1)** holds. This can be seen quite easily: Assume we are given $s_1^2 g^{(2)}(\tau_1), s_1^3 g^{(3)}(\tau_1)$. Then $s_1^2 g^{(2)}(\tau_1) = g^{(2)}(\tau_1)$ due to $(\pm 1)^2 = 1$ and inferring τ_1 from $g^{(2)}(\tau_1)$ is possible whenever $g^{(2)}$ is locally invertible. Once τ_1 is known, we can solve $s_1 g^{(3)}(\tau_1)$ for s_1. For $n \in \{2, 3\}$, we recover $\mathcal{C}_n$ approximately by solving two linear systems involving the Gramian matrices $\widehat{G}_n$ of the ensembles $(w_k^{\otimes n})_{k\in[m]}$. More precisely, we show that

$$\widehat{G}_n \mathcal{C}_n \approx \tilde{T}_n,$$

where $\tilde{T}_n$ denotes the vector storing the n-th directional derivative approximation of the network at the input $x = 0$ along the directions $\hat{w}_1, \dots, \hat{w}_m$. The initialization step is summarized in Algorithm 3.3. The perturbation analysis provided in Section 3.3 shows that, for suitable conditions, Algorithm 3.3 leads to correct recovery of the original signs $s_1, \dots, s_m$ and provides us with shift initializations $\hat{\tau} = [\hat{\tau}_1, \dots, \hat{\tau}_m]^\top$ of the ground truth shifts with an approximation error that scales like

$$\|\hat{\tau} - \tau\|_2 \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}} \frac{m^{7/4}}{D^{3/4}},$$

up to poly-logarithmic factors.

Refining the shift via empirical risk minimization (Section 3.4). Based on the weight recovery and initialization step, we now assume we are given weight approximations $\hat{w}_1 \approx w_1, \dots, \hat{w}_m \approx w_m$ since the ambiguity of the signs has been resolved in the initialization step. During the last step, the weight approximations will remain fixed, and the objective is to further improve upon the shift estimation. The shifts are then refined by empirical risk minimization in a teacher-student setup: We define the parametrization

$$\hat{f}(x, \hat{\tau}) := \sum_{k=1}^{m} g(s_k \langle \hat{w}_k, x \rangle + \hat{\tau}_k), \tag{3.9}$$

representing the student network. This model is then fit against the teacher network f by minimizing the least-squares objective

$$J(\hat{\tau}) = \frac{1}{2N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \Big(f(X_i) - \hat{f}(X_i, \hat{\tau}) \Big)^2, \quad \text{where } X_1, \dots, X_{N_{\text{train}}} \sim_{\text{i.i.d.}} \mathcal{N}(0, \mathrm{Id}_D), \tag{3.10}$$

via gradient descent. In Section 3.4, we provide a local convergence analysis of the associated gradient descent iteration

$$\hat{\tau}^{(n+1)} = \hat{\tau}^{(n)} - \gamma \nabla_{\hat{\tau}} J(\hat{\tau}^{(n)}), \tag{3.11}$$

where $\gamma > 0$ denotes a step-size parameter. Assuming the setting discussed in Section 3.1.1, we then prove the following result (cf. Theorem 3.4): For the unperturbed case, where teacher and student weights are identical, we prove that (3.11) converges to the original shifts τ at a linear rate when started from an initialization $\hat{\tau}^{(0)}$ given by our initialization step. For the perturbed case, we prove that (3.11) started from our initialization will remain within distance Δ_W of the original shifts τ, where Δ_W is an error term depending on the accuracy of the weight approximation. A discussion of the term Δ_W is included at the end of this section and in Section 3.4.2. The proof of local convergence relies on a linearization argument of the objective J around the true shifts τ, which allows us to relate the behavior of (3.11) to the spectrum of the matrix

$$\mathbb{E}_{X \sim \mathcal{N}(0, \mathrm{Id})} [\nabla_{\hat{\tau}} \hat{f}(X, \tau)^{\otimes 2}]. \tag{3.12}$$

In Section 3.4.5, we rely on tools appearing in the *neural tangent kernel (NTK)* theory [73] to prove a lower bound on the minimal eigenvalue of (3.12).

Algorithm 3.1: Network reconstruction

Input: Shallow neural network f defined in (3.1), numerical differentiation method $\Delta^n[\cdot]$ with accuracy ϵ, number of Hessian locations N_h and gradient descent samples N_{train}, number of steps for refinement via gradient descent N_{GD}.

1 Compute weights $\widehat{W} = [\hat{w}_1|\dots|\hat{w}_m]$ using Algorithm 3.2;

2 Find signs $\hat{s}$ and initial shifts $\hat{\tau} \in \mathbb{R}^m$ by linearization through Algorithm 3.3;

3 Set $\widehat{W} \leftarrow \widehat{W}\,\mathrm{diag}(\hat{s})$ and construct a student network $\hat{f}$ as in (3.9) with parameters $\widehat{W}, \hat{\tau}$;

4 Draw samples $x_1, \dots, x_{N_{\text{train}}} \sim \mathcal{N}(0, \mathrm{Id}_D)$ and refine the shifts of $\hat{f}$ by minimizing $J(\hat{\tau})$ (cf. (3.10)) via gradient descent for N_{GD} steps (cf. Section 3.4). Denote by $\hat{\tau}^{[N_{\text{GD}}]}$ the final iterate.

Output: Weights $\widehat{W}$ and final shifts $\hat{\tau}^{[N_{\text{GD}}]}$ of $\hat{f}$.

Main result Let us now state our main result, which is achieved by combining all three steps discussed above in Algorithm 3.1.

Theorem 3.2 (cf. [51, Theorem 2])**.** *Consider the teacher network f defined in* (3.1)*, where $w_1, \dots, w_m \sim \mathrm{Unif}(\mathbb{S}^{D-1})$ and $\tau_1, \dots, \tau_m \in [-\tau_\infty, \tau_\infty]$. Assume g satisfies **(SNM1)** - **(SNM2)** and f satisfies the learnability condition **(SNM3)** for some $\alpha > 0$. Assume we run Algorithm 3.1 with $N_h > t(m + m^2 \log(m)/D)$ for some $t \geq 1$ and $N_{train} > m\sqrt{D}$. Then, there exists $D_0 \in \mathbb{N}$ and a constant $C > 0$ only depending on g and τ_∞ such that the following holds with probability at least $1 - m^{-1} - 2D^2 \exp\left(-\min\{\alpha, 1\}t/C\right) - Cm^2 \exp(-\sqrt{D}/C)$: If $m \geq D \geq D_0$, $Cm \log^2 m \leq D^2$, and the numerical differentiation accuracy ϵ satisfies*

$$\epsilon \leq \frac{D^{1/2} \min\{1, \alpha^{1/2}\}}{Cm^{9/2} \log(m)^{3/2}}, \tag{3.13}$$

then Algorithm 3.1 returns weights and shifts $(\widehat{W} = [\hat{w}_1|\dots|\hat{w}_m], \hat{\tau}^{(N_{GD})})$ that fulfill

$$\max_{k \in [m]} \|\hat{w}_{\pi(k)} - w_k\|_2 \leq C(m/\alpha)^{1/4} \epsilon^{1/2}, \tag{3.14}$$

$$\|\hat{\tau}_\pi^{(N_{GD})} - \tau\|_2 \leq C \left(\frac{m^{7/4} D^{1/4} \epsilon^{1/2}}{\alpha^{1/4} N_{train}^{1/2}} + \frac{\xi^{N_{GD}}}{m^{1/2}} + \Delta_{W,1} \right), \tag{3.15}$$

for some permutation π and some constant $\xi \in [0, 1)$ where

$$\Delta_{W,1} := \frac{m^{1/2} \log(m)^{3/4}}{D^{1/4}} \cdot \left(\|\widehat{W} - W\|_F + \frac{\Delta_{W,O}^{1/2}}{D^{1/2}} + \left\| \sum_{k=1}^m w_k - \hat{w}_k \right\|_2 \right), \tag{3.16}$$

$$\Delta_{W,O} := \sum_{k \neq k'}^m \left| \langle w_k - \hat{w}_k, w_{k'} - \hat{w}_{k'} \rangle \right|. \tag{3.17}$$

The statement of Theorem 3.2 combines the results from Theorem 3.3, Proposition 3.1 and Theorem 3.4. The proof has been deferred to Section 3.5. Before continuing with the theoretical discussion of each pipeline step, let us provide some context on Theorem 3.2. The output of Algorithm 3.1 is a set of weight approximations $\hat{w}_1 \approx w_1, \dots, \hat{w}_m \approx w_m \in \mathbb{S}^{D-1}$ and shift approximations $\hat{\tau}_1^{(N_{\text{GD}})} \approx \tau_1, \dots, \hat{\tau}_m^{(N_{\text{GD}})} \approx \tau_m$, where $N_{\text{GD}} \in \mathbb{N}$ denotes the number of gradient descent iterations computed in the third step of the pipeline. Since the numerical accuracy $\epsilon > 0$ is up to our choosing, the condition in (3.13) can be satisfied, and the weight approximations

can be made arbitrarily accurate. What remains is the final error bound in the shifts. According to (3.15), we have

$$\|\hat{\tau}_\pi^{(N_{\mathrm{GD}})} - \tau\|_2 \lesssim \left(\frac{m^{7/4} D^{1/4} \epsilon^{1/2}}{\alpha^{1/4} N_{\mathrm{train}}^{1/2}} + \frac{\xi^{N_{\mathrm{GD}}}}{m^{1/2}} + \Delta_{W,1} \right).$$

Due to the dependency on $\epsilon^{1/2}/N_{\mathrm{train}}^{1/2}$, the first term can be made suitably small. Similarly, the second term will vanish at an exponential rate after sufficiently many gradient descent iterations ($\xi < 1$). Hence, we can assume that the dominant factor in this bound is $\Delta_{W,1}$.

Discussion of $\Delta_{W,1}$. The factor $\Delta_{W,1}$ originates from the perturbation analysis of the gradient descent iteration (3.11) in Section 3.4. For simplicity, let us assume the permutation in Theorem 3.2 is such that $\pi(k) = k$ for all $k \in [m]$, which can be achieved by relabeling the parameters accordingly. First, note that the term $\Delta_{W,1}$ can be bounded in terms of the uniform weight approximation error $\delta_{\max} = \max_{k\in[m]} \|\hat{w}_k - w_k\|$. Clearly, if $\delta_{\max} = 0$, then $\Delta_{W,1} = 0$. If we assume that $\delta_{\max}$ scales like (3.14), then one can compute (cf. Section 3.4.2) a crude upper bound that scales like

$$\Delta_{W,1} \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}} \frac{m^{7/4}}{D^{1/4}} \tag{3.18}$$

up to poly-logarithmic factors. This shows that $\Delta_{W,1}$ can be made arbitrarily small for a suitable numerical accuracy $\epsilon > 0$. However, the bound on $\Delta_{W,1}$ in (3.18) is generally *not optimal*, as it assumes the worst-case where the weight approximation errors can align perfectly. In this case, expressions like $\|\sum_k^m w_k - \hat{w}_k\|_2$ suffer from error accumulation.

Remark 3.2. *The scaling of $\Delta_{W,1}$ plays an important role when comparing the error of the shifts after the initialization step to the error bound after running gradient descent. We discuss this dynamic in greater detail in Section 3.4.2, where it is shown that gradient descent does lead to a guaranteed improvement whenever we assume randomness of the weight errors.*

The underlying results on which our weight recovery is based (cf. Theorem 2.10 and [76]) do not specify the distribution of the errors induced by SPM. However, empirically we do not observe error accumulations (see Figure 3.5 and Section 3.6 and the related discussion). An appropriate random model for the residual weight errors based on the scaling in (3.14) would be

$$\hat{w}_{\pi(1)} - w_1, \ldots, \hat{w}_{\pi(m)} - w_m \sim_{\text{i.i.d.}} \mathcal{N}(0, (m/\alpha)^{(1/2)} \epsilon / D \cdot \mathrm{Id}_D).$$

Since these random variables satisfy

$$\mathbb{E}\left[\|\hat{w}_{\pi(k)} - w_k\|_2\right] = \frac{m^{(1/4)} \epsilon^{1/2}}{D^{1/2} \alpha^{(1/4)}} \mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}\left[\|X\|_2\right] = \frac{\epsilon^{1/2} m^{(1/4)}}{\alpha^{(1/4)}},$$

which reflects the upper bound in (3.14). Under these assumptions, the factor $\Delta_{W,1}$ scales like

$$\Delta_{W,1} \lesssim \frac{\epsilon^{1/2} \log^{3/4}(m)}{\alpha^{1/4}} \left(\frac{m^{5/4}}{D^{1/4}} + \frac{m^{7/4}}{D} \right), \tag{3.19}$$

which is an improvement of a factor of $O(D^{-3/4})$ over (3.18). We refer to the discussion in Section 3.4.2 for more details.

3.2 Weight identification

Consider a shallow neural network $f : \mathbb{R}^D \to \mathbb{R}$ of the form

$$f(x) = \sum_{k=1}^{m} g(\langle x, w_k \rangle + \tau_k)$$

that satisfies the conditions stated throughout Section 3.1.1. This section addresses the recovery of the weight vectors $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ up to sign for sufficiently high dimensions, and our results apply to the regime where the number of neurons m scales up to $m \log^2 m = \mathcal{O}(D)$. We show that for our setting, the recovery of the weights $w_1, \dots, w_m$ (up to sign) based on network queries can be reduced to the rank-one basis recovery problem, which was studied in Chapter 2. This reduction has already been discussed in Section 1.5 and relies on the fact that the Hessians expose the weights as

$$\nabla^2 f(x) = \sum_{k=1}^{m} g^{(2)}(\langle w_k, x \rangle + \tau_k) w_k \otimes w_k.$$

In Lemma 3.1 below, we prove that this can be leveraged to construct an approximation $\widehat{\mathcal{W}} \subset \mathrm{Sym}(\mathbb{R}^{D \times D})$ to the space spanned by the rank-one matrices $\{w_k \otimes w_k | k \in [m]\}$. The recovery of $\pm w_k$ from $\widehat{\mathcal{W}}$ is then tackled by Algorithm 2.2. This approach is covered by the recent results from [77] that extend the analysis of SPM to the overcomplete regime $m = o(D^2)$ (cf. Section 2.6 and Theorem 2.10). More precisely, in Theorem 3.3, we prove that our recovery strategy (which is summarized in Algorithm 3.2) computes weight approximations $\hat{w}_1, \dots, \hat{w}_m \in \mathbb{S}^{D-1}$ such that

$$\min_{s \in \{-1,+1\}} \|w_k - s\hat{w}_k\|_2 \lesssim (m/\alpha)^{1/4} \epsilon^{1/2} \quad \text{for all } k \in [m],$$

where α, ϵ are defined in Section 3.1.1. Our result holds with high probability for isotropic random vectors in high dimensions and relies on the mutual incoherence of the system $\{w_k \otimes w_k | k \in [m]\}$ that has previously been covered in Section 2.6.1.

3.2.1 Reduction to the rank-one basis recovery problem

This section is concerned with quantifying the approximation error $\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F \to F}$ (defined in (2.6)), where $\mathcal{W}$ denotes the symmetric matrix space

$$\mathcal{W} := \mathrm{span}\{w_1 \otimes w_1, \dots, w_m \otimes w_m\}, \tag{3.20}$$

and $\widehat{\mathcal{W}} \subset \mathrm{Sym}(\mathbb{R}^{D \times D})$ is its approximating space constructed as the m-th singular subspace of independently sampled Hessian matrices. For a definition of singular subspaces, we refer to Section 1.1.1. Consider Hessian locations $x_1, \dots, x_{N_h} \sim_{\text{i.i.d}} \mathcal{N}(0, \mathrm{Id}_D)$ drawn independently as standard Gaussians. Then, in the setting of Lemma 3.1, we prove that the m-th singular subspace $\widehat{\mathcal{W}} = \mathrm{span}_m\{\nabla^2 f(x_1), \dots, \nabla^2 f(x_{N_h})\}$ satisfies

$$\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F \to F} \lesssim \sqrt{m/\alpha} \cdot \epsilon,$$

w.h.p. for N_h sufficiently large. The important aspect here is that the approximation error scales directly with the numerical accuracy parameter ϵ, which we can, in theory, make arbitrarily small.

Lemma 3.1 ([51, Lemma 1])**.** *Consider the teacher network f defined in* (3.1)*. Assume the activation g satisfies **(SNM1)** - **(SNM2)** and f satisfies the learnability condition **(SNM3)** for some $\alpha > 0$. Furthermore, assume that the network weights $w_1, \dots, w_m \in \mathrm{S}^{D-1}$ fulfill **(Inc2)** of Definition 2.3 with constant c_2. Let $P_{\mathcal{W}}$ be the orthogonal approximation onto $\mathcal{W} = \operatorname{span}\{w_1 \otimes w_1, \dots w_m \otimes w_m\}$ and let $P_{\widehat{\mathcal{W}}}$ be constructed as described in Algorithm 3.2. Then there exists a constant $C > 0$ depending only on g and c_2, such that for numerical differentiation accuracy $\epsilon < \frac{\sqrt{\alpha}}{C\sqrt{m}}$ and $N_h > t(m + m^2 \log(m)/D)$ for some $t \geq 1$ we have*

$$\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F} \leq C\sqrt{m/\alpha} \cdot \epsilon, \tag{3.21}$$

with probability at least $1 - D^2 \exp\left(-\frac{t\alpha}{C}\right)$.

Proof of Lemma 3.1. Consider $X_1, \dots, X_{N_h}$ independent copies of a standard Gaussian, i.e., $X_i \sim \mathcal{N}(0, \mathrm{Id}_D)$. Denote by $\mathcal{P}_{\mathcal{W}} \in \mathbb{R}^{D^2 \times D^2}$ the orthogonal projection matrix onto the vector space $\operatorname{span}\{\operatorname{vec}(w_k \otimes w_k) \mid i = 1, \dots, m\}$ and by M the matrix with columns given by the exact vectorized Hessians at the inputs $X_1, \dots, X_{N_h}$, i.e.,

$$M := \begin{bmatrix} \operatorname{vec}(\nabla^2 f(X_1)) & \dots & \operatorname{vec}(\nabla^2 f(X_{N_h})) \end{bmatrix} \in \mathbb{R}^{D^2 \times N_h}. \tag{3.22}$$

We associate the matrix subspaces $\mathcal{W}$ and $\widehat{\mathcal{W}}$ with their corresponding D^2-dimensional vector subspaces described by the orthogonal projection matrices $\mathcal{P}_{\mathcal{W}}, \mathcal{P}_{\widehat{\mathcal{W}}}$, respectively (cf. Section 1.1.1). Note that

$$\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F} = \sup_{\|M\|_F = 1} \left\| P_{\mathcal{W}}(M) - P_{\widehat{\mathcal{W}}}(M) \right\|_F = \left\| \mathcal{P}_{\mathcal{W}} - \mathcal{P}_{\widehat{\mathcal{W}}} \right\|$$

with $\|\cdot\|$ describing the ordinary spectral normal in $\mathbb{R}^{D^2}$. Hence, to prove the result, we can rely on the well-known Wedin bound, see for instance [54, 52, 50, 76], giving

$$\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F} = \left\| \mathcal{P}_{\mathcal{W}} - \mathcal{P}_{\widehat{\mathcal{W}}} \right\| \leq \frac{\|M - \widehat{M}\|_F}{\sigma_m(\widehat{M})}, \tag{3.23}$$

for as long as $\sigma_m(\widehat{M}) > 0$. We continue to provide separate bounds for the numerator and denominator of (3.23). For the numerator, we obtain

$$\begin{aligned} \|M - \widehat{M}\|_F &\leq \sqrt{N_h} \max_{i \in [N_h]} \left\| \nabla^2 f(X_i) - \Delta^2 f(X_i) \right\|_F \\ &\leq \sqrt{N_h m} \max_{\substack{i \in [N_h] \\ k \in [m]}} \left\| \nabla^2 g(w_k^\top X_i + \tau_k) - \Delta^2 g(w_k^\top X_i + \tau_k) \right\|_F \\ &\leq \max_{k \in [m]} C_\Delta \sqrt{N_h m} \left\| w_k \otimes w_k \right\|_F \epsilon = C_\Delta \sqrt{N_h m} \epsilon, \end{aligned}$$

where we used the linearity of Δ^2, ∇^2 in the second step and our assumptions on the numerical differentiation method **(G4.2)** in the last line, which gives rise to the constant C_Δ that only depends on g. For the denominator in (3.23) we use Weyl's inequality [132] which leads to the lower bound

$$\sigma_m(\widehat{M}) \geq \sigma_m(M) - \|M - \widehat{M}\| \geq \sigma_m(M) - \|M - \widehat{M}\|_F \geq \sigma_m(M) - C_\Delta \sqrt{N_h m} \epsilon. \tag{3.24}$$

Lastly, we need to control $\sigma_m(M)$ by a concentration argument in combination with the learnability assumption **(SNM3)** of Section 3.1.1. We first express $\sigma_m(M)$ as sum of independent matrices:

$$\sigma_m(M)^2 = \sigma_m(MM^\top) = \sigma_m\left(\sum_{i=1}^{N_h} \operatorname{vec}(\nabla^2 f(X_i)) \otimes \operatorname{vec}(\nabla^2 f(X_i)) \right). \tag{3.25}$$

Denote $A_i = \mathrm{vec}(\nabla^2 f(X_i)) \otimes \mathrm{vec}(\nabla^2 f(X_i))$. By **(SNM3)** we know that

$$\sigma_m\left(\sum_{i=1}^{N_h} \mathbb{E}A_i\right) = N_h\alpha > 0.$$

We will make use of the matrix Chernoff (see [126] Corollary 5.2 and the following remark), which states that

$$\mathbb{P}\left(\sigma_m\left(\sum_{i=1}^{N_h} A_i\right) \leq (1-s)\sigma_m\left(\sum_{i=1}^{N_h}\mathbb{E}A_i\right)\right) \leq D^2 \exp\left(-(1-s)^2\sigma_m\left(\sum_{i=1}^{N_h}\mathbb{E}A_i\right)/2K\right) \tag{3.26}$$

for $s \in [0,1]$ and $K = \max_{i\in[N_h]} \|A_i\|_2$. The norm of A_i can be bound uniformly over all $x \in \mathbb{R}^D$ by

$$\begin{aligned}
\left\|\mathrm{vec}(\nabla^2 f(X_i)) \otimes \mathrm{vec}(\nabla^2 f(X_i))\right\|_2 &\leq \sup_{x\in\mathbb{R}^D} \left\|\mathrm{vec}(\nabla^2 f(x))\right\|_2^2 = \sup_{x\in\mathbb{R}^D} \left\|\nabla^2 f(x)\right\|_F^2 \\
&= \sup_{x\in\mathbb{R}^D} \left\|\sum_{k=1}^m g^{(2)}(w_k^\top x + \tau_k) w_k \otimes w_k\right\|_F^2 \\
&= \sup_{x\in\mathbb{R}^D} \sum_{k,\ell=1}^m g^{(2)}(w_k^\top x + \tau_k) g^{(2)}(w_\ell^\top x + \tau_\ell)\langle w_k, w_\ell\rangle^2 \\
&\leq \kappa^2 \sum_{k,\ell=1}^m \langle w_k, w_\ell\rangle^2 \leq \kappa^2(m + c_2 m(m-1)\log m/D).
\end{aligned}$$

The last inequality follows by the incoherence assumption **(Inc2)** from the initial statement. Combining this with (3.26) for $s = 1/2$ together with the bound on the spectrum of the expectation yields

$$\mathbb{P}\left(\sigma_m\left(\sum_{i=1}^{N_h} A_i\right) \geq \frac{1}{2}N_h\alpha\right) \geq 1 - D^2\exp\left(-\frac{N_h D\alpha}{8\kappa^2(Dm + c_2 m^2\log m)}\right). \tag{3.27}$$

Conditioning on this event, and assuming $\epsilon < \sqrt{\alpha/8C_\Delta^2 m}$ the initial subspace bound now holds as

$$\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F\to F} \leq \frac{\left\|M - \widehat{M}\right\|_2}{\sigma_m(\widehat{M})} \leq \frac{C_\Delta\sqrt{N_h m}\epsilon}{\sqrt{\frac{1}{2}N_h\alpha} - C_\Delta\sqrt{N_h m}\epsilon} = \frac{C_\Delta\sqrt{m}\cdot\epsilon}{\sqrt{\frac{\alpha}{2}} - C_\Delta\sqrt{m}\cdot\epsilon} \tag{3.28}$$

$$\leq \frac{\sqrt{2}C_\Delta\sqrt{m}\cdot\epsilon}{\sqrt{\alpha}} \tag{3.29}$$

with said probability. The final result follows by applying the bound on ϵ onto the denominator. More precisely, we need that $C > 2C_\Delta$ to fulfill (3.28) and $C > 8\kappa^2\max\{1, c_2\}$ which implies

$$1 - D^2\exp\left(-\frac{N_h D\alpha}{8\kappa^2(Dm + c_2 m^2\log m)}\right) \leq 1 - D^2\exp\left(-t\alpha/C\right),$$

due to our assumption that $N_h > t(m + m^2\log(m)/D)$. □

Algorithm 3.2: Weight recovery

Input: Shallow neural network f, number of neurons m, number of Hessian locations N_h, β threshold for rejection of spurious local maximizers

1 Draw independent samples $x_1, \dots, x_{N_h} \sim \mathcal{N}(0, \mathrm{Id})$;

2 Compute Hessian approximations $\Delta^2 f(x_1), \dots, \Delta^2 f(x_{N_h})$;

3 Compute the m-th left singular subspace of the Hessian approximations

$$\widehat{\mathcal{W}} \leftarrow \mathrm{span}_m \left\{ \Delta^2 f(x_1), \dots, \Delta^2 f(x_{N_h}) \right\},$$

and denote by $P_{\widehat{\mathcal{W}}}$ the orthogonal projection onto $\widehat{\mathcal{W}}$ (see Section 1.1.1);

4 Compute the set of approximated weights $\mathcal{U}$ using Algorithm 2.2 with input $(P_{\widehat{\mathcal{W}}}, \beta)$;

Output: $\mathcal{U}$

3.2.2 Recovery guarantees

By Lemma 3.1, we can compute an approximation $\widehat{\mathcal{W}}$ to the matrix space $\mathcal{W}$ in (3.20) with accuracy controlled by the numerical accuracy. This result requires that the network weights $w_1, \dots, w_m$ satisfy condition **(Inc2)** of Definition 2.3. The following theorem combines two results. First, we show that Lemma 3.1 can be extended to the case where $w_1, \dots, w_m$ are drawn uniformly at random from the unit-sphere, which is a trivial consequence of Proposition 2.2. Second, applying the results from Theorem 2.8 (based on [76] from Section 2.6 to the resulting matrix space approximation $\widehat{\mathcal{W}}$ guarantees the recovery of all weights leaving only uncertainty of the signs and an approximation error that scales like

$$\min_{s \in \{-1,+1\}} \|w_k - s\hat{w}_k\|_2 \lesssim (m/\alpha)^{1/4} \epsilon^{1/2} \quad \text{for all } k \in [m].$$

For more details on SPM and Algorithm 2.2, we refer to Chapter 2.

Theorem 3.3 (Weight recovery, [51, Theorem 3])**.** *Consider the teacher network f defined in* (3.1)*, where $w_1, \dots, w_m \sim \mathrm{Unif}(\mathbb{S}^{D-1})$ and $\tau_1, \dots, \tau_m \in [-\tau_\infty, \tau_\infty]$. Assume g satisfies **(SNM1)** - **(SNM2)** and f satisfies the learnability condition **(SNM3)** for some $\alpha > 0$. Then, there exists $D_0 \in \mathbb{N}$ and a constant $C > 0$ depending only on g, τ_∞, such that, for all $D \geq D_0$ and $Cm \log^2 m \leq D^2$, the following holds with probability at least $1 - m^{-1} - D^2 \exp\left(-\min\{\alpha, 1\} t/C\right) - C \exp(-\sqrt{m}/C)$:*

*(i) The weights $w_1, \dots, w_m$ fulfill properties **(Inc1)** - **(Inc3)** of Definition 2.3.*

(ii) The output of Algorithm 3.2 with numerical differentiation accuracy $\epsilon \leq \frac{\sqrt{\alpha}}{C\sqrt{m}}$ and using $N_h > t(m + m^2 \log(m)/D)$ Hessian locations for some $t \geq 1$ is a set of approximated weights $\mathcal{U} \subset \mathbb{S}^{D-1}$ such that, for all $\hat{w} \in \mathcal{U}$, there exists a $k \in [m]$ and a sign $s \in \{-1, +1\}$ for which

$$\|w_k - s\hat{w}_k\|_2 \leq C(m/\alpha)^{1/4} \epsilon^{1/2}. \tag{3.30}$$

Proof of Theorem 3.3. The weights $w_1, \dots, w_m$ of f are drawn uniformly from the unit sphere. By Proposition 2.2, and for any $\delta_0 \in (0, 1)$, there exists $D_1 \in \mathbb{N}, C_1 > 0$ depending only on δ_0 such that for all $D \geq D_1$ this set of weights fulfills conditions **(Inc1)** - **(Inc3)** of Definition 2.3 with constants $c_2, c_3 < C_1$ and with probability at least

$$1 - m^{-1} - 2\exp(-C_1 \delta_0^2 D) - C_1 \left(\frac{e \cdot D}{\sqrt{m}} \right)^{-C_1 \sqrt{m}}.$$

We condition on this event and denote it by E_1 for the remaining part of the proof. Now, due to the incoherence of the weights and according to our initial assumption, which includes $N_h > t(m + m^2 \log(m)/D)$, the conditions of Lemma 3.1 are met. This provides an error bound for the subspace, which is constructed in the first part of Algorithm 3.2, such that

$$\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F} \leq C_2 \sqrt{m/\alpha} \cdot \epsilon, \tag{3.31}$$

with probability at least $1 - D^2 \exp\left(-t\alpha/C_2\right)$ for a constant C_2 only depending on g. Denote the event that this subspace bound holds by E_2 and assume it occurs, which only depends on the number of Hessians N_h in relationship to D, m. Note that δ_0 can be freely chosen in $(0, 1)$. By Theorem 2.10 there exist constants D_2, Δ_0, C_3, such that for $D \geq D_2$ and $\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F} \leq \Delta_0$, the local maximizers of the program $\operatorname{argmax}_{u \in \mathbb{S}^{D-1}} \left\| P_{\widehat{\mathcal{W}}}(u \otimes u) \right\|_F^2$ fulfill

$$\min_{s \in \{-1,1\}} \|x^* - s w_k\|_2 \leq \sqrt{\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F}} \leq \sqrt{C_2 \sqrt{m/\alpha} \cdot \epsilon}, \tag{3.32}$$

as long as they belong to the level set

$$\left\{ x \in \mathbb{S}^{D-1} \middle| \left\| P_{\widehat{\mathcal{W}}}(x \otimes x) \right\|_F^2 \geq C_3 m \log^2(m)/D^2 + 5 C_2 \sqrt{m/\alpha} \cdot \epsilon \right\}.$$

By iterating projected gradient ascent until convergence, every vector $\hat{u}$ will be one of these local maximizers. Also note that by construction, all vectors returned by Algorithm 3.2 must have unit norm, hence $\mathcal{U} \subset \mathbb{S}^{D-1}$. We need to make sure that the level set is not empty, which is guaranteed for $C_3 m \log^2(m)/D^2 \leq \frac{1}{4}$ and $\epsilon \leq \frac{\alpha^{1/2}}{20 C_2 \sqrt{m}}$ which leads to the threshold

$$C_3 m \log^2(m)/D^2 + 5 C_2 \sqrt{m/\alpha} \cdot \epsilon \leq \frac{1}{4} + \frac{1}{4} = \frac{1}{2}. \tag{3.33}$$

Therefore, only considering local maximizers that fulfill $\left\| P_{\widehat{\mathcal{W}}}(x \otimes x) \right\|_F^2 \geq 1/2$ will guarantee that all local maximizers are of the kind which satisfies (3.32). Before we conclude, some points still need to be addressed. To achieve the bound (3.32) we had to assume that $\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F} \leq \Delta_0$. This is true due to (3.31) given the accuracy satisfies $\epsilon \leq \frac{\Delta_0 \alpha^{1/2}}{C_3 m^{1/2}}$ which is clearly realizable by our initial assumptions on ϵ, since Δ_0 is independent of m, D. Hence, by further unifying also the constants C_1, C_2, C_3, D_1, D_2, we showed that there exists constants $C > 0, D_0 \in \mathbb{N}$ such that for $D \geq D_0$ and $C m \log^2 m \leq D^2$ all vectors $u \in \mathcal{U}$ returned by Algorithm 3.2 ran with numerical accuracy $\epsilon \leq \frac{\sqrt{\alpha}}{C\sqrt{m}}$ will fulfill the uniform error bound,

$$\min_{s \in \{-1,1\}} \|x^* - s \bar{w}_k\|_2 \leq C (m/\alpha)^{1/4} \epsilon^{1/2}, \tag{3.34}$$

and this result holds with the combined probability

$$1 - D^2 \exp\left(-t\alpha/C\right) - m^{-1} - 2\exp(-D/C) - C \left(\frac{e \cdot D}{\sqrt{m}} \right)^{-\sqrt{m}/C}. \tag{3.35}$$

□

Assuming the weight approximations are sufficiently close to the true weights, we would expect that the system of approximations exhibits similar incoherence properties. The last result of this section combines several auxiliary statements that make this rigorous. In particular, we prove that approximations that match a certain level of accuracy will inherit properties **(Inc2)** - **(Inc3)** from the ground truth weights.

Lemma 3.2 ([51, Lemma 3]). *Assume the ground truth weights $\{w_k \in \mathbb{S}^{D-1} | k \in [m]\}$ fulfill **(Inc1)** - **(Inc3)** of Definition 2.3 with constants c_2, c_3 and that $D \leq m$. Then there exists a constant $C > 0$ only depending on c_2, c_3 such that for approximations $\{\hat{w}_k \in \mathbb{S}^{D-1} | k \in [m]\}$ which satisfy the error bound*

$$\max_{k\in[m]} \min_{s\in\{-1,1\}} \|s\hat{w}_k - w_k\|_2 = \delta_{\max} \leq \frac{1}{C} \frac{D^{1/2}}{m\sqrt{\log m}} \tag{3.36}$$

*condition **(Inc2)** - **(Inc3)** of Definition 2.3 holds with constants $2c_2, 2c_3$. Furthermore, denote $\tilde{G}_n \in \mathbb{R}^{m\times m}$ the matrix with entries $\tilde{G}_{n,\ell k} = \langle \hat{w}_\ell, s_k w_k \rangle^n$, where s_k are the ground truth signs. Then there exists D_0 such that for $m \geq D \geq D_0$, $n = 2, 3$ the following holds true:*

(i) For all $k \neq \ell$ we have $\langle \hat{w}_k, s_\ell w_\ell \rangle^2 \leq 2c_2 \log(m)/D$

(ii) $\tilde{G}_n$ is invertible and $\|\tilde{G}_n^{-1}\| \leq 3c_2$

(iii) Denote by $\tilde{G}_n \in \mathbb{R}^{m\times m}$ the matrix with entries $\tilde{G}_{n,\ell k} = \langle \hat{w}_\ell, s_k w_k \rangle^n$, then

$$\left\| \tilde{G}_n - \widehat{G}_n \right\| \leq Cm \left(\frac{\log m}{D} \right)^{(2n-1)/4} \delta_{\max}. \tag{3.37}$$

Proof of Lemma 3.2. W.l.o.g. we can assume that C is chosen such that

$$\max_{k\in[m]} \min_{s\in\{-1,1\}} \|s\hat{w}_k - w_k\|_2 = \delta_{\max} \leq \min\left\{ \frac{1}{8} \left(\frac{c_2 \log m}{D} \right)^{1/2}, \frac{D^{1/2}}{8c_3 m\sqrt{2c_2 \log m}} \right\} \tag{3.38}$$

holds. We start by showing **(Inc2)** for the approximated weights. Pick any $k, \ell \in [m], k \neq \ell$. A first observation is that we can disregard the sign that appears in (3.36) since $\langle \hat{w}_k, \hat{w}_\ell \rangle^2 = \langle -\hat{w}_k, \hat{w}_\ell \rangle^2$. So w.l.o.g. assume that both signs are correct and therefore $\|\hat{w}_k - w_k\|_2 \leq \delta_{\max}$ and $\|\hat{w}_\ell - w_\ell\|_2 \leq \delta_{\max}$. Then

$$\begin{aligned}
\langle \hat{w}_k, \hat{w}_\ell \rangle^2 &\leq \left(|\langle w_k, w_\ell \rangle| + |\langle \hat{w}_k - w_k, w_\ell \rangle| + |\langle w_k, \hat{w}_\ell - w_\ell \rangle| + |\langle \hat{w}_k - w_k, \hat{w}_\ell - w_\ell \rangle| \right)^2 \\
&\leq \left(|\langle w_k, w_\ell \rangle| + 2\delta_{\max} + \delta_{\max}^2 \right)^2 \leq |\langle w_k, w_\ell \rangle|^2 + 6\delta_{\max} |\langle w_k, w_\ell \rangle| + 9\delta_{\max}^2 \\
&\leq \frac{c_2 \log m}{D} + 6 \left(\frac{c_2 \log m}{D} \right)^{1/2} \delta_{\max} + 9\delta_{\max}^2 \\
&\leq \frac{c_2 \log m}{D} + \frac{48+9}{64} \frac{c_2 \log m}{D} \leq \frac{2c_2 \log m}{D},
\end{aligned} \tag{3.39}$$

which proves that **(Inc2)** is fulfilled by the approximated weights for a constant $2c_2$. Moving on to **(Inc3)**, we need to bound the minimal eigenvalue of $\widehat{G}_n = (\widehat{W}^\top \widehat{W})^{\odot n}$ from below. Assuming $\widehat{G}_2$ is invertible, we know by Lemma 2.22 that

$$\|\widehat{G}_n^{-1}\| \leq \|\widehat{G}_2^{-1}\| \quad \text{for all} \quad n \geq 2.$$

Thus, it is sufficient to show that **(Inc3)** holds for the approximated weights for $n = 2$. Denote $G_2 = (W^\top W)^{\odot 2}$. Clearly $G_2, \widehat{G}_2$ are symmetric, and since **(Inc3)** holds for the ground truth weights, we know that the minimal eigenvalue of G_2 can be bounded by a constant $|\sigma_m(G_2)| \geq c_3^{-1}$. Hence, by Weyl's inequality (cf. [132]), we have

$$\left| \sigma_m(\widehat{G}_2) \right| \geq c_3^{-1} - \left\| \widehat{G}_2 - G_2 \right\|. \tag{3.40}$$

Our goal is to find an upper bound for the spectral norm on the right-hand side. Note that the diagonals of both matrices are identical since all columns of $\widehat{W}$ and W have unit norm, so we focus on the off-diagonal exclusively. Via Gershgorin's circle theorem we attain

$$\begin{aligned}
\left\|\widehat{G}_n - G_n\right\| &\leq \max_{k\in[m]} \sum_{\substack{\ell=1\\ \ell\neq k}}^{m} \left|\langle \hat{w}_k, \hat{w}_\ell\rangle^2 - \langle w_k, w_\ell\rangle^2\right| \\
&= \max_{k\in[m]} \sum_{\substack{\ell=1\\ \ell\neq k}}^{m} \left|\langle s_k\hat{w}_k, s_\ell\hat{w}_\ell\rangle^2 - \langle w_k, w_\ell\rangle^2\right| \\
&\leq \max_{k\in[m]} \sum_{\substack{\ell=1\\ \ell\neq k}}^{m} \left|\langle s_k\hat{w}_k, s_\ell\hat{w}_\ell\rangle + \langle w_k, w_\ell\rangle\right| \left|\langle s_k\hat{w}_k, s_\ell\hat{w}_\ell\rangle - \langle w_k, w_\ell\rangle\right| \\
&\leq 2\left(\frac{2c_2\log m}{D}\right)^{1/2} \max_{k\in[m]} \sum_{\substack{\ell=1\\ \ell\neq k}}^{m} \left|\langle s_k\hat{w}_k - w_k, s_\ell\hat{w}_\ell\rangle - \langle w_k, s_\ell\hat{w}_\ell - w_\ell\rangle\right| \\
&\leq 4\left(\frac{2c_2\log m}{D}\right)^{1/2} m\cdot\delta_{\max} \leq \frac{1}{2c_3},
\end{aligned}$$

where we used the fact that **(Inc2)** holds for the ground truth weights and approximated weights in the penultimate inequality followed by the uniform bound in (3.36) at the end. We conclude with Weyl's inequality (cf. [132]) which yields

$$\left|\sigma_m(\widehat{G}_2^{-1})\right| \leq \left|\sigma_1(\widehat{G}_2)\right|^{-1} \leq 2c_3. \tag{3.41}$$

Hence, the approximated weights fulfill **(Inc3)** with constant $2c_3$ for $n = 2$, which extends to $n \geq 2$ by Lemma 2.22. Let us now proof $(i) - (iii)$. The first statement follows directly from our proof of **(Inc2)** for the approximated weights since for any $k \neq \ell$ we have

$$\langle \hat{w}_k, s_\ell w_\ell\rangle^2 \leq \left(|\langle w_\ell, w_k\rangle| + |\langle \hat{w}_\ell - w_\ell, w_k\rangle|\right)^2 \leq \left(|\langle w_\ell, w_k\rangle| + \delta_{\max}\right)^2 \leq \frac{2c_2\log m}{D},$$

follows by the chain of inequalities started in (3.39). To show (iii), we first split the difference $\tilde{G}_n - \widehat{G}_n = D_n + O_n$ into a diagonal part D_n and an off-diagonal part O_n. We have $\|\tilde{G}_n - \widehat{G}_n\| \leq \|D_n\| + \|O_n\|$ and start by controlling $\|O_n\|$ via Gershgorin's circle theorem:

$$\begin{aligned}
\|O_n\| &\leq \max_{\ell\in[m]} \sum_{\substack{k=1\\ k\neq\ell}}^{m} \left|\langle \hat{w}_k, \hat{w}_\ell\rangle^n - \langle \hat{w}_k, s_\ell w_\ell\rangle^n\right| \\
&\leq \max_{\ell\in[m]} \sum_{\substack{k=1\\ k\neq\ell}}^{m} \left|\langle \hat{w}_k, \hat{w}_\ell\rangle - \langle \hat{w}_k, s_\ell w_\ell\rangle\right| \left|\sum_{i=1}^{n} \langle \hat{w}_k, \hat{w}_\ell\rangle^{n-i}\langle \hat{w}_k, s_\ell w_\ell\rangle^{i-1}\right| \\
&\leq n\left(\frac{2c_2\log m}{D}\right)^{(n-1)/2} \max_{\ell\in[m]} \sum_{\substack{k=1\\ k\neq\ell}}^{m} \left|\langle \hat{w}_k, \hat{w}_\ell - s_\ell w_\ell\rangle\right|.
\end{aligned}$$

From here, we can slightly improve over Cauchy-Schwarz, and, instead, use that

$$\sum_{\substack{k=1\\ k\neq\ell}}^{m} \left|\langle \hat{w}_k, \hat{w}_\ell - s_\ell w_\ell\rangle\right| \leq \sqrt{m-1}\sqrt{\sum_{\substack{k=1\\ k\neq\ell}}^{m} \langle \hat{w}_k, \hat{w}_\ell - s_\ell w_\ell\rangle^2} \leq \sqrt{m}\left\|\widehat{W}\right\|\delta_{\max}.$$

Using $\|\widehat{W}\| = \|\widehat{W}^\top \widehat{W}\|^{1/2} \leq \left(1 + m\left(\frac{2c_2 \log m}{D}\right)^{1/2}\right)^{1/2}$ we arrive at the following bound for the off-diagonal terms:

$$\begin{aligned}\|O_n\| &\leq n\left(\frac{2c_2 \log m}{D}\right)^{(n-1)/2} \sqrt{m}\left(1 + m\left(\frac{2c_2 \log m}{D}\right)^{1/2}\right)^{1/2} \delta_{\max} \\ &\leq Cnm\left(\frac{\log m}{D}\right)^{(n-1)/2}\left(\frac{\log m}{D}\right)^{1/4} \delta_{\max} \leq Cnm\left(\frac{\log m}{D}\right)^{(2n-1)/4} \delta_{\max},\end{aligned}$$

where $C > 0$ is an absolute constant only depending on c_2 and $m \geq D$ was used in the second inequality. For the diagonal part, we receive

$$\|D_n\| = \left|1 - \max_{\ell \in [m]} |\langle \hat{w}_\ell, w_\ell \rangle|^n\right| \leq \left|1 - (1 - \delta_{\max}^2/2)^n\right|.$$

Hence, we attain overall

$$\left\|\tilde{G}_n - \widehat{G}_n\right\| \leq \left|1 - (1 - \delta_{\max}^2/2)^n\right| + Cnm\left(\frac{\log m}{D}\right)^{(2n-1)/4} \delta_{\max}.$$

For $n = 2, 3$ and some constant $C_1 > 0$ depending only on c_2 this can be further simplified using the bound on $\delta_{\max}$ as

$$\begin{aligned}\left\|\tilde{G}_n - \widehat{G}_n\right\| &\leq \delta_{\max}^2 + Cnm\left(\frac{\log m}{D}\right)^{(2n-1)/4} \delta_{\max} \\ &\leq C_1 m\left(\frac{\log m}{D}\right)^{(2n-1)/4} \delta_{\max},\end{aligned}$$

which confirms (iii). To prove (ii), we need to show that $\left\|\tilde{G}_n - \widehat{G}_n\right\| \leq c_4/2$ from which the rest follows as before by Weyl's inequality. We can reuse (iii) in combination with 3.36 obtaining

$$\left\|\tilde{G}_n - \widehat{G}_n\right\| \leq C_1 m\left(\frac{\log m}{D}\right)^{(2n-1)/4} \delta_{\max} \leq C_2\left(\frac{\log m}{D}\right)^{1/4}$$

for some constant C_2. Hence, the statement in (ii) is true for $D \geq D_0$ sufficiently large.

□

3.3 Recovery of the correct signs and initialization of the shifts

Consider a shallow neural network that satisfies the assumptions within Section 3.1.1. By the machinery of Section 3.2 we expect to find weight approximations such that $\hat{w}_k \approx s_k w_k$ for some sign $s_k \in \{-1, +1\}$ for all $k \in [m]$. We can regard this approximation as arbitrarily accurate since the approximation error stated in Theorem 3.3 scales linearly with the numerical accuracy, which we assume to be sufficiently small. This section addresses the recovery of the signs $s = (s_1, \ldots, s_m) \in \{-1, +1\}^{\times m}$. Additionally, our procedure will provide us with initial estimates of the shift parameters $\tau = (\tau_1, \ldots, \tau_m) \in \mathbb{R}^m$, which is required due to the nature of our local convergence analysis in Section 3.4. Throughout this section, we will assume that the network weights fulfill **(Inc2)-(Inc3)** of Definition 2.3 and that the same conditions extend to the weight approximations $\hat{w}_1, \ldots, \hat{w}_m$. It follows by Lemma 3.2 that the second part of our assumption can be fulfilled by choosing $\epsilon > 0$ in Theorem 3.3 to be sufficiently small such that (3.36) holds.

3.3.1 Parameter initialization: Strategy

We begin with a motivation behind the parameter initialization described in Algorithm 3.3 below.

Isolating signs and shifts. First, note that the n-th derivative of the neural network at input $x = 0$ is given by

$$\nabla^n f(0) = \sum_{k=1}^{m} g^{(n)}(\tau_k) w_k^{\otimes n} = \sum_{k=1}^{m} s_k^n g^{(n)}(\tau_k)(s_k w_k)^{\otimes n}, \tag{3.42}$$

which for $n \leq 3$ is well-defined under condition **(SNM1)**. By Lemma 2.21, **(Inc3)** implies linear independence of the system $\{w_k^{\otimes n} | k \in [m]\}$ for $n \geq 2$, which in turn implies linear independence of the system $\{s_k w_k^{\otimes n} | k \in [m]\}$. This means $s_1^n g^{(n)}(\tau_1), \ldots, s_1^n g^{(n)}(\tau_m)$ are the unique coefficients for the decomposition in (3.42) that give rise to $\nabla^n f(0)$. Our initialization strategy is based on the formulation of two linear systems with unique solutions given by the vectors $\mathcal{C}_2 = (\mathcal{C}_{2,1}, \ldots, \mathcal{C}_{2,m})$ and $\mathcal{C}_3 = (\mathcal{C}_{3,1}, \ldots, \mathcal{C}_{3,m})$, which are defined as

$$\mathcal{C}_{n,k} := s_k^n g^{(n)}(\tau_k), \quad \text{for } k \in [m], \quad n \in \{2, 3\}. \tag{3.43}$$

If g satisfies **(SNM1)**, then $g^{(2)}$ is monotonic and therefore admits an inverse $(g^{(2)})^{-1}$ on $(-\tau_\infty, +\tau_\infty)$. Hence, $\mathcal{C}_2$ provides access to τ due to $s_k^2 = 1$, which can be seen from

$$(g^{(2)})^{-1}(\mathcal{C}_{2,k}) = (g^{(2)})^{-1}(s_k^2 g^{(2)}(\tau_k)) = (g^{(2)})^{-1}(g^{(2)}(\tau_k)) = \tau_k \quad \text{for all } k \in [m].$$

The signs can then be inferred from $\mathcal{C}_3$, either by using the shifts via

$$s_k = \operatorname{sign}(\mathcal{C}_{3,k}) \cdot \operatorname{sign}(g^{(3)}(\tau_k)), \tag{3.44}$$

or equivalently by $s_k = \operatorname{sign}(\mathcal{C}_{3,k}) \cdot \operatorname{sign}(g^{(3)}(0))$ since $g^{(3)}$ does not change sign on the interval $(-\tau_\infty, \tau_\infty)$ due to the monotonicity of $g^{(2)}$. This clarifies how $\mathcal{C}_2, \mathcal{C}_3$ are linked to the recovery of the signs and shifts. However, there remain several problems. We need to account for the presence of perturbations caused by the weight approximation as well as approximation errors of the derivative ($\nabla^n f(0)$ is not directly accessible).

Recovering of $\mathcal{C}_2, \mathcal{C}_3$ from (3.42). Assume for now that the approximations are exact up to a sign, i.e., $\hat{w}_k = s_k w_k$ for all $k \in [m]$. Then solving (3.42) for $\mathcal{C}_n$ is equivalent to solving the linear system

$$\widehat{W}_n \mathcal{C}_n = \operatorname{vec}(\nabla^n f(0)), \tag{3.45}$$

where $\widehat{W}_n = (\operatorname{vec}(\hat{w}_1^{\otimes n}) | \ldots | \operatorname{vec}(\hat{w}_m^{\otimes n})) \in \mathbb{R}^{D^n \times m}$. Since our setting implies $n \geq 2$, $m = o(D^2)$ (i.e., $\widehat{W}_n$ is a tall matrix), we might want to consider instead the linear system

$$\widehat{W}_n^\top \widehat{W}_n \mathcal{C}_n = \widehat{W}_n^\top \operatorname{vec}(\nabla^n f(0)).$$

Notably, the matrix $\widehat{W}_n^\top \widehat{W}_n$ is equivalent to the Gramian matrix $\widehat{G}_n$ associated with the system $\{\hat{w}_k^{\otimes n} | k \in [m]\}$ (cf. Section 2.6.1). In the setting of Lemma 3.2, the Gramian $\widehat{G}_n$ is positive definite for all $n \geq 2$. Therefore, we obtain

$$\mathcal{C}_n = \widehat{G}_n^{-1} \widehat{W}_n^\top \operatorname{vec}(\nabla^n f(0)). \tag{3.46}$$

Furthermore, we can apply an additional simplification to (3.45) by leveraging the structure of the vector $\widehat{W}_n^\top \operatorname{vec}(\nabla^n f(0)) \in \mathbb{R}^m$: Note that for each $k \in [m]$

$$(\widehat{W}_n^\top \operatorname{vec}(\nabla^n f(0)))_k = \langle \nabla^n f(0), \hat{w}_k^{\otimes n} \rangle = \sum_{\ell=1}^m g^{(n)}(\tau_\ell) \langle w_\ell, \hat{w}_k \rangle^n,$$

which is equivalent to the n-th directional derivative of f along $\hat{w}_k$ in zero. Consequently, we do not need to compute the full derivative $\nabla^n f(0)$, and can only rely on m directional derivatives, which are much cheaper to compute for large n. We will denote the vector storing all n-th order directional derivatives as

$$T_n := \begin{bmatrix} \langle \nabla^n f(0), \hat{w}_1^{\otimes n} \rangle \\ \vdots \\ \langle \nabla^n f(0), \hat{w}_m^{\otimes n} \rangle \end{bmatrix}. \tag{3.47}$$

In summary, we showed that for the unperturbed case where $\hat{w}_k = s_k w_k$ for all $k \in [m]$, we can recover $\mathcal{C}_2, \mathcal{C}_3$ via

$$\mathcal{C}_n = \widehat{G}_n^{-1} T_n.$$

Let us now discuss the general case where $\hat{w}_k \approx s_k w_k$. Then (3.45) does hold only approximately and instead of the Gramian matrix $\widehat{G}_n$ we need to rephrase (3.46) in terms of the matrix $\tilde{G}_n$ with entries $(\tilde{G}_n)_{\ell,k} = \langle \hat{w}_\ell, s_k w_k \rangle^n$. More precisely $\tilde{G}_n \mathcal{C}_n = T_n$, which can be written as

$$\widehat{G}_n^{-1} \tilde{G}_n \mathcal{C}_n = \widehat{G}_n^{-1} T_n.$$

Additionally, we need to account for the fact that the directional derivatives in T_n are not directly accessible and need to be approximated as $\tilde{T}_n \approx T_n$ via numerical differentiation (cf. **(G4.2)**). Based on the preceding section, we assume that $\tilde{G}_n$ is sufficiently close to $\widehat{G_n}$ (cf. Lemma 3.2 (iii)), and our recovery strategy computes approximations $\tilde{\mathcal{C}}_n \approx \mathcal{C}_n$ for $n \in \{2, 3\}$ where

$$\tilde{\mathcal{C}}_n := \widehat{G}_n^{-1} \tilde{T}_n \approx \widehat{G}_n^{-1} T_n = \widehat{G}_n^{-1} \tilde{G}_n \mathcal{C}_n \approx \mathcal{C}_n.$$

The overall recovery strategy has been summarized in Algorithm 3.3.

3.3.2 Parameter initialization: Guarantees

The last section motivated the individual steps of Algorithm 3.3, and we saw that the recovery of the signs and shifts ultimately depends on the accuracy of the approximations $\tilde{\mathcal{C}}_2 \approx \mathcal{C}_2, \tilde{\mathcal{C}}_3 \approx \mathcal{C}_3$. To make the resulting approximation error on $\mathcal{C}_2, \mathcal{C}_3$ precise, we need to control two error terms. Firstly, the error caused by the approximation of directional derivatives $\|T_n - \tilde{T}_n\|_2$ and secondly, the error $\|\widehat{G}_n^{-1} \tilde{G}_n - \mathrm{Id}_m\|$. This perturbation analysis is carried out in the following statement.

Lemma 3.3 (cf. [51, Lemma 4])**.** *Denote by $\tilde{\mathcal{C}}_n = \widehat{G}_n^{-1} \tilde{T}_n$ the coefficient vectors computed by Algorithm 3.3 for an input network f with ground truth weights $\{w_k \in \mathbb{S}^{D-1} | k \in [m]\}$ which fulfill **(Inc2)** - **(Inc3)** of Definition 2.3 with constants c_2, c_3 and activation g that fulfills **(SNM1)**. Then there exist constants $C > 0$ only depending on g, c_2, c_3 and $D_0 \in \mathbb{N}$, such that for $m \geq D \geq D_0, m \log^2 m \leq D^2$, $n = 2, 3$ and provided approximations $\{\hat{w}_k \in \mathbb{S}^{D-1} | k \in [m]\}$ to the ground truth weights such that*

$$\max_{k \in [m]} \min_{s \in \{-1,1\}} \|s\hat{w}_k - w_k\|_2 = \delta_{\max} \leq \frac{1}{C} \frac{D^{1/2}}{m\sqrt{\log m}}, \tag{3.50}$$

Algorithm 3.3: Parameter Initialization

Input: Approximated weights $\widehat{W}$, numerical differentiation schema $\Delta^n[\cdot]$ with accuracy $\epsilon > 0$, interval on which $g^{(2)}$ is monotonic $[-\tau_\infty, +\tau_\infty]$

Set $\widehat{G}_2 \leftarrow (\widehat{W}^\top \widehat{W})^{\odot 2}$,$\widehat{G}_3 \leftarrow (\widehat{W}^\top \widehat{W})^{\odot 3}$

for $k = 1, \dots, m$ **do**

  Compute directional derivative approximations $\tilde{T}_{2,k} \leftarrow \Delta^2[f(\cdot \hat{w}_k)](0)$, $\tilde{T}_{3,k} \leftarrow \Delta^3[f(\cdot \hat{w}_k)](0)$

end

Set $\tilde{\mathcal{C}}_2 \leftarrow \widehat{G}_2^{-1}\tilde{T}_2$, $\tilde{\mathcal{C}}_3 \leftarrow \widehat{G}_3^{-1}\tilde{T}_3$

for $k = 1, \dots, m$ **do**

$$\tilde{\tau}_k \leftarrow \begin{cases} \left(g^{(2)}\right)^{-1}(\tilde{\mathcal{C}}_{2,k}), & \text{if } \left(g^{(2)}\right)^{-1} \text{ is defined for } \tilde{\mathcal{C}}_{2,k}, \\ \operatorname{argmin}_{t \in [-\tau_\infty, \tau_\infty]} \left| g^{(2)}(t) - \tilde{\mathcal{C}}_{2,k} \right| & \text{else,} \end{cases} \tag{3.48}$$

$$\tilde{s}_k \leftarrow \operatorname{sign}(\tilde{\mathcal{C}}_{3,k}) \cdot \operatorname{sign}(g^{(3)}(0)), \tag{3.49}$$

end

Output: $\tilde{\tau}, \tilde{s}$

we have

$$\left\| \tilde{\mathcal{C}}_n - s^n \odot g^{(n)}(\tau) \right\|_2 \leq C\sqrt{m}\epsilon + Cm^{3/2} \left(\frac{\log m}{D} \right)^{(2n-1)/4} \delta_{\max}, \tag{3.51}$$

where s is the vector storing the true signs that are implied by (3.50)*.*

Proof of Lemma 3.3. Denote as in Algorithm 3.3 $\tilde{T}_{n,k} = \Delta^n[f(\cdot \hat{w}_k)](0)$ and $T_{n,k} = \langle \nabla^n f(0), \hat{w}_k^{\otimes n} \rangle$. By their definition and the linearity of ∇^n, Δ^n we have

$$\begin{aligned} \left\| T_n - \tilde{T}_n \right\|_\infty &= \sup_{k \in [m]} \left| \langle \nabla^n f(0), \hat{w}_k^{\otimes n} \rangle - \Delta_\epsilon^n[f(\hat{w}_k \cdot)](0) \right| && (3.52) \\ &\leq \sup_{k \in [m]} \sum_{\ell=1}^m \left| \frac{\partial^n}{\partial t^n} g(\langle \hat{w}_k, w_\ell \rangle t + \tau_\ell) \Big|_{t=0} - \Delta^n[g(\langle \hat{w}_k, w_\ell \rangle \cdot + \tau_\ell)](0) \right| && (3.53) \\ &\leq C_\Delta \epsilon \sup_{k \in [m]} \sum_{\ell=1}^m |\langle \hat{w}_k, w_\ell \rangle|^{n+2} \leq C_\Delta \epsilon \left(1 + m \left(\frac{2c_2 \log m}{D} \right)^{\frac{n+2}{2}} \right), && (3.54) \end{aligned}$$

where we used the second point of **(G4.2)** in the last line followed by the incoherence of the approximated weights **(Inc2)** established in Lemma 3.2. Making use of $D^2 \geq m \log^2 m$, this simplifies to

$$\left\| T_n - \tilde{T}_n \right\|_\infty \leq C_1 \cdot \epsilon$$

with constant $C_1 = (1 + 4c_2^2)C_\Delta$ for $n = 2, 3$. Coming back to our initial objective, we can express $s^n \odot g^{(n)}(\tau)$ as the product $s^n \odot g^{(n)}(\tau) = T_n \tilde{G}_n$ where $\tilde{G}_n$ describes the matrix with entries given by $(\tilde{G}_n)_{k\ell} = \langle \hat{w}_k, s_\ell w_\ell \rangle^n$. Note that Algorithm 3.3 constructs $\tilde{\mathcal{C}}_n = \widehat{G}_n^{-1}\tilde{T}_n$, where

$(\widehat{G}_n)_{k\ell} = \langle \hat{w}_k, \hat{w}_\ell \rangle^n$. We can reduce our main statement (3.51) into separate bounds

$$\left\| \tilde{\mathcal{C}}_n - s^n \odot g^{(n)}(\tau) \right\|_2 = \left\| \widehat{G}_n^{-1} \tilde{T}_n - \tilde{G}_n^{-1} T_n \right\|_2 \tag{3.55}$$

$$\leq \left\| \widehat{G}_n^{-1} (T_n - \tilde{T}_n) \right\|_2 + \left\| (\widehat{G}_n^{-1} - \tilde{G}_n^{-1}) T_n \right\|_2 \tag{3.56}$$

$$\leq \sqrt{m} \left\| \widehat{G}_n^{-1} \right\| \left\| T_n - \tilde{T}_n \right\|_\infty + \left\| (\widehat{G}_n^{-1} - \tilde{G}_n^{-1}) T_n \right\|_2 \tag{3.57}$$

$$\leq C_1 \sqrt{m} \cdot \epsilon + \left\| (\widehat{G}_n^{-1} - \tilde{G}_n^{-1}) T_n \right\|_2 \tag{3.58}$$

To bound $\left\| (\widehat{G}_n^{-1} - \tilde{G}_n^{-1}) T_n \right\|_2$ we first decompose according to

$$\left\| (\widehat{G}_n^{-1} - \tilde{G}_n^{-1}) T_n \right\|_2 = \left\| \widehat{G}_n^{-1} (\widehat{G}_n - \tilde{G}_n) \tilde{G}_n^{-1} T_n \right\|_2 = \left\| \widehat{G}_n^{-1} (\widehat{G}_n - \tilde{G}_n)(s^n \odot g^{(n)}(\tau)) \right\|_2. \tag{3.59}$$

By invoking Definition **(Inc3)** again, we continue with

$$\left\| \widehat{G}_n^{-1} (\widehat{G}_n - \tilde{G}_n)(s^n \odot g^{(n)}(\tau)) \right\|_2 \leq 2c_3 \left\| \widehat{G}_n - \tilde{G}_n \right\| \left\| s^n \odot g^{(n)}(\tau) \right\|_2 \leq 2c_3 \kappa \sqrt{m} \left\| \widehat{G}_n - \tilde{G}_n \right\|, \tag{3.60}$$

where we used $\left\| g^{(n)} \right\|_\infty \leq \kappa$. The statement then follows by using inequality (iii) of Lemma 3.2 onto $\left\| \widehat{G}_n - \tilde{G}_n \right\|$ and unifying the involved constants. □

The statement above quantifies the recovery error of $\mathcal{C}_2, \mathcal{C}_3$. As explained at the beginning of Section 3.3.1, these coefficients encode the unknown signs and shift parameters of a shallow network. Our main result of this section now estimates the quality of the estimated shift vectors and provides sufficient conditions under which the signs are recovered correctly.

Proposition 3.1 (Parameter initialization, [51, Proposition 1])**.** *Consider the teacher network f defined in* (3.1)*, where the weights $\{w_k \in \mathbb{S}^{D-1}, k \in [m]\}$ satisfy **(Inc2)** - **(Inc3)** with constants c_2, c_3 and the activation g satisfies **(SNM1)**. Then, there exist constants $C > 0$ only depending on g, c_2, c_3, τ_∞ and $D_0 \in \mathbb{N}$, such that, for $m \geq D \geq D_0, m \log^2 m \leq D^2$, the following holds. Given $\hat{w}_1, \ldots, \hat{w}_m \in \mathbb{S}^{D-1}$ such that*

$$\delta_{\max} := \max_{k \in [m]} \min_{s \in \{-1,1\}} \|w_k - s\hat{w}_k\|_2 \leq \frac{D^{1/2}}{Cm\sqrt{\log m}}, \tag{3.61}$$

Algorithm 3.3 returns a set of shifts $\hat{\tau}$ such that

$$\|\hat{\tau} - \tau\|_2 \leq C\sqrt{m}\epsilon + Cm^{3/2} \left(\frac{\log m}{D} \right)^{3/4} \delta_{\max}, \tag{3.62}$$

where $\epsilon > 0$ is the accuracy of the numerical differentiation method and $\tau \in \mathbb{R}^m$ denotes the ground truth shifts of the network. Furthermore, once the RHS of (3.62) *is smaller than* 1 *and $\epsilon \leq (Cm)^{-1}$, the signs returned by Algorithm 3.3 are identical to the ground truth signs.*

The proof of Proposition 3.1 builds upon the results in Lemma 3.3 and quantifies the propagation of the approximation error of $\|\mathcal{C}_n - \tilde{\mathcal{C}}_n\|_2$ through the remaining steps of Algorithm 3.3. Before the proof is stated, we provide some context on the result. According to the preceding Proposition, the shifts estimated by Algorithm 3.3 satisfy

$$\|\hat{\tau} - \tau\|_2 \lesssim \sqrt{m}\epsilon + m^{3/2} \left(\frac{\log m}{D} \right)^{3/4} \delta_{\max},$$

whenever $\delta_{\max}$ adheres to the bound in (3.61). If the approximated weights are computed by Algorithm 3.2, we can fulfill condition (3.61) by choosing ϵ sufficiently small. Then, after replacing $\delta_{\max}$ with the bound $\delta_{\max} \lesssim (m/\alpha)^{1/4}\epsilon^{1/2}$ from Theorem 3.3, we receive

$$\|\hat{\tau} - \tau\|_2 \lesssim \sqrt{m}\epsilon + \frac{\epsilon^{1/2}}{\alpha^{1/4}} \frac{m^{7/4}\log^{3/4} m}{D^{3/4}}. \tag{3.63}$$

Considering a suitably small ϵ and omitting poly-logarithmic factors, the dominant term in (3.63) scales as

$$\|\hat{\tau} - \tau\|_2 \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}} \frac{m^{7/4}}{D^{3/4}}. \tag{3.64}$$

Hence, we can generally expect that the right-hand side of (3.62) is smaller than 1 and therefore, the signs returned by Algorithm 3.3 are identical to the ground truth signs.

Remark 3.3. *Let us mention one final comment about the error in* (3.63)*. If the numerical accuracy ϵ can be made arbitrarily small, then Algorithm 3.3 recovers the shifts exactly under the setting of Proposition 3.1. In the context of our overall recovery pipeline outlined in Algorithm 3.1, this begs the question: Why do we need to further refine the shifts (cf. Section 3.4)? A preliminary answer to this question is that any further refinement of the initial shifts computed by Algorithm 3.3 is justified as long as the refinement can potentially improve over the error bound* (3.63)*. We address this aspect in a more detailed discussion as part of Section 3.4.2.*

Proof of Proposition 3.1. First, note that due to the assumptions made, we can freely apply the results of Lemma 3.2 and Lemma 3.3. As a consequence, the approximated weights considered in the statement of Proposition 3.1 fulfill **(Inc2)-(Inc3)** of Definition 2.3 with constants derived from the ground truth weights as described in Lemma 3.2. We continue with the remark that **(SNM1)** guarantees the existence of the inverse function ${g^{(2)}}^{-1}$ on $[-\tau_\infty, \tau_\infty]$ and here we can disregard the signs such that

$${g^{(2)}}^{-1}(s^2 \odot g^{(2)}(\tau)) = {g^{(2)}}^{-1}\left(1 \odot g^{(2)}(\tau)\right) = \tau \tag{3.65}$$

While $s^2 \odot g^{(2)}(\tau)$ is not directly available, $\tilde{\mathcal{C}}_2$ serves as an approximation $\tilde{\mathcal{C}}_2 \approx s^2 \odot g^{(2)}(\tau)$. Fix any $k \in [m]$, and assume that

$$\tilde{\mathcal{C}}_{2,k} \in \left[\min_{t\in[-\tau_\infty,+\tau_\infty]} g^{(2)}(t), \max_{t\in[-\tau_\infty,+\tau_\infty]} g^{(2)}(t)\right], \tag{3.66}$$

then by the mean value theorem

$$\begin{aligned}
\hat{\tau}_k &= {g^{(2)}}^{-1}(\tilde{\mathcal{C}}_{2,k}) = {g^{(2)}}^{-1}\left(g^{(2)}(\tau_k) + \tilde{\mathcal{C}}_{2,k} - g^{(2)}(\tau_k)\right) \\
&= {g^{(2)}}^{-1}\left(g^{(2)}(\tau_k)\right) + \left(\tilde{\mathcal{C}}_{2,k} - g^{(2)}(\tau_k)\right)({g^{(2)}}^{-1})'(\xi_k) \\
&= \tau_k + \left(\tilde{\mathcal{C}}_{2,k} - g^{(2)}(\tau_k)\right)\frac{1}{g^{(3)}\left({g^{(2)}}^{-1}(\xi_k)\right)}
\end{aligned}$$

for some $\xi_k \in \left[\min_{t\in[-\tau_\infty,+\tau_\infty]} g^{(2)}(t), \max_{t\in[-\tau_\infty,+\tau_\infty]} g^{(2)}(t)\right]$. Since $g^{(2)}$ is strictly monotonic on $[-\tau_\infty, \tau_\infty]$ and differentiable we have

$$\theta := \max_{t\in[-\tau_\infty,\tau_\infty]} \left|g^{(3)}(t)\right| > 0.$$

Hence, we can bound $\left|\frac{1}{g^{(3)}\left(g^{(2)^{-1}}(\xi_k)\right)}\right| \leq \theta^{-1}$ from the outgoing assumption **(SNM1)**. Applying Lemma 3.3 to bound $\left\|g^{(2)}(\tau) - \tilde{\mathcal{C}}_2\right\|_2$ therefore yields

$$\|\hat{\tau} - \tau\|_2 \leq \theta^{-1}\left(C\sqrt{m}\epsilon + Cm^{3/2}\left(\frac{\log m}{D}\right)^{3/4}\delta_{\max}\right) \tag{3.67}$$

Now assume there is a $k \in [m]$ such that $\tilde{\mathcal{C}}_{2,k}$ does not satisfy (3.66). By the monotonicity, we also know that the maximal and minimal value of $g^{(2)}$ are found exactly on $\pm\tau_\infty$. If $\tilde{\mathcal{C}}_{2,k}$ does not lie in the image of $g^{(2)}$ on $[-\tau_\infty, +\tau_\infty]$ it has to exceed one of those. We can assume w.l.o.g. that $\tilde{\mathcal{C}}_{2,k} > \max_{t\in[-\tau_\infty,+\tau_\infty]} g^{(2)}(t) = g^{(2)}(\tau_\infty)$. Then

$$\left|g^{(2)}(\tau_\infty) - g^{(2)}(\tau_k)\right| < \left|\tilde{\mathcal{C}}_{2,k} - g^{(2)}(\tau_k)\right|,$$

which shows that $g^{(2)}(\tau_\infty)$ is simply a better estimate of $g^{(2)}(\tau_k)$ than $\tilde{\mathcal{C}}_{2,k}$, and $g^{(2)^{-1}}$ is also defined for $g^{(2)}(\tau_\infty)$. Hence, the same error bound as above holds for all $k \in [m]$. The expression in (3.49) yields the correct sign if $\operatorname{sign}(\tilde{\mathcal{C}}_{3,k}) = \operatorname{sign}(s_k^{(3)}) \cdot \operatorname{sign}(g^{(3)}(\tau_k)) = \operatorname{sign}(s_k) \cdot \operatorname{sign}(g^{(3)}(\tau_k))$. This is the case if

$$\left|s_k^{(3)} \cdot g^{(3)}(\tau_k)\right| > \left|s_k^{(3)} \cdot g^{(3)}(\tau_k) - \tilde{\mathcal{C}}_{3,k}\right|. \tag{3.68}$$

By our outgoing assumption $\left|s_k^{(3)} \cdot g^{(3)}(\tau_k)\right| \geq \theta$ and together with Lemma 3.3 applied to the right-hand side of the inequality above we get that the signs are correct as long as

$$\theta > \left(C\sqrt{m}\epsilon + Cm^{3/2}\left(\frac{\log m}{D}\right)^{5/4}\delta_{\max}\right). \tag{3.69}$$

Assume now that the RHS of (3.62) is smaller than 1 and $\epsilon \leq (Cm)^{-1}$, this implies in particular

$$Cm^{3/2}\left(\frac{\log m}{D}\right)^{3/4}\delta_{\max} < 1.$$

We can estimate the right-hand side of (3.69) from above by

$$C\sqrt{m}\epsilon + Cm^{3/2}\left(\frac{\log m}{D}\right)^{5/4}\delta_{\max} \leq \frac{1}{m^{1/2}} + \left(\frac{\log m}{D}\right)^{2/4},$$

which clearly is smaller than any constant for D large enough, and therefore the signs will be correct for D_0 chosen accordingly since (3.68) is fulfilled. □

3.4 Refining the approximated shifts using empirical risk minimization

Let us shortly recall the previous results and discuss a suitable setting for the remaining parts of the network recovery. Consider a shallow network $f : \mathbb{R}^D \to \mathbb{R}$ of the form

$$f(x) = \sum_{k=1}^{m} g(\langle x, w_k\rangle + \tau_k)$$

adhering to the conditions made in Section 3.1.1. Furthermore, consider the setting after successfully running the *weight recovery* (Algorithm 3.2) and *parameter initialization* (Algorithm 3.3). Hence, we are given access to approximations $\hat{w}_k \approx w_k, \hat{\tau}_k \approx \tau_k$ for all $k \in [m]$. According to Theorem 3.3 and Proposition 3.1, we can estimate the approximation error of these parameters by

$$\max_{k\in[m]} \|w_k - \hat{w}_k\|_2 \lesssim (m/\alpha)^{1/4}\epsilon^{1/2} \quad \text{and} \quad \|\hat{\tau} - \tau\|_2 \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}} \frac{m^{7/4}}{D^{3/4}}. \tag{3.70}$$

As the weight approximations are a downstream result of Algorithm 3.2, which leverages the incoherence conditions **(Inc1) - (Inc3)** in Definition 2.3, we can furthermore assume that the weights $w_1, \dots, w_m \in \mathbb{S}^{D-1}$ and the approximations $\hat{w}_1, \dots, \hat{w}_m \in \mathbb{S}^{D-1}$ satisfy incoherence conditions **(Inc1) - (Inc3)** (cf. Lemma 3.2). Given this setting, we now study the refinement of the shifts (i.e., improving over the bound above) by empirical risk minimization.

3.4.1 Formulation of a simplified teacher-student problem

Consider the setting above. The parameter approximations $\hat{w}_1, \dots, \hat{w}_m \in \mathbb{S}^{D-1}, \hat{\tau} \in \mathbb{R}^m$ give rise to a neural network $\hat{f}$ defined as

$$\hat{f}(x, \hat{\tau}) = \sum_{k=1}^{m} g(\langle x, \hat{w}_k \rangle + \hat{\tau}_k).$$

For suitable small ϵ, the bounds in (3.70) suggest that $\hat{f}$ closely resembles the original network f, i.e., $\hat{f} \approx f$ for all $x \in \mathbb{R}^D$. By fixing the weights of $\hat{f}$, we can regard it as a parametric function $\hat{f} = \hat{f}(\cdot, \hat{\tau})$ w.r.t. the shifts $\hat{\tau}$. We now consider the further improvement of the estimates $\hat{\tau}$ based on *empirical risk minimization* of the least-squares objective

$$J(\hat{\tau}) = \frac{1}{2N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \left(\hat{f}(x_i, \hat{\tau}) - y_i\right)^2, \tag{3.71}$$

where $y_1 = f(x_1), \dots, y_{N_{\text{train}}} = f(x_{N_{\text{train}}})$ are real network evaluations of the original network. This closely represents the setting of the teacher-student problem. Due to the assumption **(G4.1)**, the network f can be queried at every input, and $\hat{f}(\cdot, \hat{\tau})$ is fully known. Hence, we can in principle sample arbitrary inputs. However, to keep our setting more general, we will assume that we are given generic inputs $X_1, \dots, X_{N_{\text{train}}} \sim_{\text{i.i.d.}} \mathcal{N}(0, \mathrm{Id}_D)$, which is common in the literature related to the teacher-student problem (cf. [25, 124, 111, 85, 41, 42, 114, 115, 137, 56, 54, 52, 53, 74, 86, 93, 138]). In the following, we analyze the minimization of (3.71) via a gradient descent iteration given by

$$\hat{\tau}^{(n+1)} = \hat{\tau}^{(n)} - \gamma \nabla_{\hat{\tau}} J(\hat{\tau}^{(n)}). \tag{3.72}$$

Here, $\gamma > 0$ represents the step-size of the gradient updates and $\nabla_{\hat{\tau}} J$ denotes the derivative of J w.r.t. $\hat{\tau}$. As part of the next section, we provide a local convergence analysis of (3.72).

3.4.2 Local convergence guarantees

Throughout this section we denote by $W, \widehat{W} \in \mathbb{R}^{D\times m}$ the matrices with columns given by the weights $w_1, \dots, w_m$ and approximated weights $\hat{w}_1, \dots, \hat{w}_m$, respectively. As before, the uniform approximation error of the weights will be denoted as

$$\delta_{\max} := \max_{k\in[m]} \|w_k - \hat{w}_k\|_2 .$$

Additionally, let us recall the definition of $\Delta_{W,1}$, which will appear in the statement below. In (3.16), $\Delta_{W,1}$ was defined as

$$\Delta_{W,1} := \frac{m^{1/2}\log(m)^{3/4}}{D^{1/4}} \cdot \left(\|\widehat{W} - W\|_F + \frac{\Delta_{W,O}^{1/2}}{D^{1/2}} + \left\| \sum_{k=1}^{m} w_k - \hat{w}_k \right\|_2 \right),$$

where $\Delta_{W,O}$ takes the form

$$\Delta_{W,O} = \sum_{k \neq k'}^{m} |\langle w_k - \hat{w}_k, w_{k'} - \hat{w}_{k'} \rangle| .$$

We provide the following main result on the shifts produced by iteration (3.72).

Theorem 3.4 (Local convergence, [51, Theorem 4])**.** *Consider the teacher network f defined in* (3.1)*, with shifts $\tau_1, \dots, \tau_m \in [-\tau_\infty, +\tau_\infty]$ and weights $w_1, \dots, w_m \sim \mathrm{Unif}(\mathbb{S}^{D-1})$ that are incoherent according to Definition 2.3. Assume g satisfies **(SNM1)-(SNM2)**, and consider the least-squares objective J in* (3.10) *constructed with $N_{train} \geq m$ network evaluations $y_1, \dots, y_{N_{train}}$ of f, where $y_i = f(X_i)$ and $X_1, \dots, X_{N_{train}} \sim \mathcal{N}(0, \mathrm{Id}_D)$. Let $\hat{f}$ be parameterized by $\widehat{W}$ and $\hat{\tau}$, as in* (3.9)*. Then, there exists a constant $C > 0$ depending only on g, τ_∞ and $D_0 > 0$ such that the following holds with probability at least $1 - m\exp(-N_{train}/Cm) - 2m^2\exp(-t/C)$ for $t > 0$: Assume $Cm\log^2 m \leq D^2, m \geq D \geq D_0$ and*

$$\|\tau - \hat{\tau}\|_2 + \Delta_W \leq \frac{1}{C\sqrt{m}}, \tag{3.73}$$

where $\Delta_W = \Delta_{W,1} + \left(\frac{m^3\delta_{\max}^2 t}{N_{train}}\right)^{1/2}$ and $\Delta_{W,1}$ is given by (3.16)*. Then, the gradient descent iteration* (3.72) *with sufficiently small step-size $\gamma > 0$ started from $\hat{\tau}^{(0)} = \hat{\tau}$ satisfies*

$$\|\hat{\tau}^{(n)} - \tau\|_2 \leq 2\xi^n \|\hat{\tau}^{(0)} - \tau\|_2 + C(1 - \xi^n)\Delta_W. \tag{3.74}$$

The proof (see Section 3.4.8) has been broken down into several individual results, which are compartmentalized into individual sections. Before giving any further technical results, let us provide some insight into the local convergence result stated above.

Interpretation of Theorem 3.4. Under suitable conditions, the main statement of Theorem 3.4 guarantees that the gradient descent iteration in (3.72) after $n \in \mathbb{N}$ steps $\hat{\tau}^{(n)}$ approximates the ground truth shifts τ at least with

$$\|\hat{\tau}^{(n)} - \tau\|_2 \lesssim \xi^n \|\hat{\tau}^{(0)} - \tau\|_2 + (1 - \xi^n)\Delta_W \quad \text{for some } \xi \in [0, 1). \tag{3.75}$$

with high probability provided that $\|\hat{\tau}^{(0)} - \tau\|_2, \Delta_W \lesssim m^{-1/2}$. The term Δ_W, appearing on the right-hand side, scales with the error of the weight approximation. In particular, in the case $\widehat{W} = W$ we have $\Delta_W = 0$, such that (3.75) implies linear convergence $\hat{\tau}^{(n)} \to \tau$ for $n \to \infty$. In the perturbed case $\widehat{W} \approx W$, the teacher-student problem based on the objective (3.71) is generally not able to recover the shifts exactly since the difference between the weights $w_1, \dots, w_m$ and $\hat{w}_1, \dots, \hat{w}_m$ introduce an irreversible difference between the networks $f, \hat{f}$. The bound (3.75) suggests a dynamic, where gradient descent will eventually forget about its initialization, i.e., $\xi^n \|\hat{\tau}^{(0)} - \tau\|_2 \to 0$, and then remain within distance $(1 - \xi^n)\Delta_W \to \Delta_W$ of the ground truth shifts τ. We have

$$\Delta_W = \Delta_{W,1} + \left(\frac{m^3\delta_{\max}^2 t}{N_{\text{train}}}\right)^{1/2}. \tag{3.76}$$

The factor $\Delta_{W,1}$ (stated at the beginning of the section and in (3.16)) depends on the alignment of the weight errors $(w_k - \hat{w}_k)_{k\in[m]}$. As discussed in Section 3.1.2, our weight recovery does not characterize the distribution of individual errors. Therefore, we can not theoretically exclude that the errors align (which leads to accumulation in the theoretical proofs). If the residual weight errors behave randomly, then according to Section 3.1.2, up to poly-logarithmic factors, $\Delta_{W,1}$ scales as

$$\Delta_{W,1} \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}} \left(\frac{m^{5/4}}{D^{1/4}} + \frac{m^{7/4}}{D} \right). \tag{3.77}$$

We can regard $\Delta_{W,1}$ as the dominant factor in Δ_W: Plugging the bound $\delta_{\max} \lesssim (m/\alpha)^{1/4}\epsilon^{1/2}$ from (3.70) into the right-hand term from (3.76) yields

$$\left(\frac{m^3 \delta_{\max}^2 t}{N_{\text{train}}} \right)^{1/2} \lesssim \left(\frac{m^{7/2}\epsilon t}{\alpha^{1/2} N_{\text{train}}} \right)^{1/2} = \frac{\epsilon^{1/2}}{\alpha^{1/4}} \left(\frac{m^{7/4} t^{1/2}}{{N_{\text{train}}}^{1/2}} \right).$$

Hence, $\Delta_{W,1}$ is the dominant factor whenever $t^{1/2}D < N_{\text{train}}^{1/2}$, which can easily be fulfilled by suitable N_{train}, t. The error term $\Delta_{W,1}$, on the other hand, does not decrease for larger amounts of training samples.

Improvement over Algorithm 3.3. When gradient descent is used as part of our pipeline, it will be *after initialization of the shifts* (Algorithm 3.3). This is due to the fact that our analysis only admits local guarantees of GD, and therefore we need to rely on a good prior estimation of the shifts that fall into the convergence radius of our analysis. More precisely, Theorem 3.4 requires $\|\hat{\tau}^{(0)} - \tau\|_2 \lesssim m^{-1/2}$. This introduces a question that has already been brought up in Remark 3.3: Does gradient descent improve over its initialization? At first glance, this question is easy to answer. Theorem 3.4 requires $\|\hat{\tau}^{(0)} - \tau\|_2 \lesssim m^{-1/2}$ and the final error term in the perturbed case (cf. (3.76)) scales with ϵ. Thus, for sufficiently small ϵ, gradient descent will improve upon its initialization. However, whenever the starting point $\hat{\tau}^{(0)}$ is computed by Algorithm (3.3), then $\|\hat{\tau}^{(0)} - \tau\|_2$ will also decrease for smaller ϵ. To fully answer this question, we need to compare the guarantees in Proposition 3.1 with the dominant error term of gradient descent, which is $\Delta_{W,1}$ (see above). Recall that in Section 3.3.2, more precisely in (3.64), we derived the error term of Algorithm 3.3 up to poly-logarithmic factors as

$$\left\| \hat{\tau}^{(0)} - \tau \right\|_2 \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}} \frac{m^{7/4}}{D^{3/4}}. \tag{3.78}$$

If we assume randomness of the weight errors, then (3.78) needs to be compared to the bound in 3.77. Again, we can neglect the factor $\frac{\epsilon^{1/2}}{\alpha^{1/4}}$. Clearly, we have $m^{7/4}/D < m^{7/4}/D^{3/4}$, and therefore gradient descent improves upon the initialization once

$$\frac{m^{5/4}}{D^{1/4}} < \frac{m^{7/4}}{D^{3/4}} \Leftrightarrow D < m, \tag{3.79}$$

which corresponds to the overcomplete regime and is exactly the setting we are addressing. Hence, for $D \ll m = o(D^2)$, we get a provable improvement by gradient descent of order $D^{-1/4}$. However, this relies on the randomness of the errors in Algorithm 3.2. Without such an assumption, we suffer from error accumulation that occurs in the decomposition (3.16) of $\Delta_{W,1}$, which up to poly-logarithmic factors scales as

$$\frac{m^{1/2}}{D^{1/4}} \cdot \left(\|\widehat{W} - W\|_F + \frac{\Delta_{W,O}^{1/2}}{D^{1/2}} + \left\| \sum_{k=1}^{m} w_k - \hat{w}_k \right\|_2 \right).$$

If we assume perfectly aligned errors, i.e., that $\hat{w}_1 - w_1 = \hat{w}_2 - w_2 = \cdots = \hat{w}_m - w_m$, then we receive the individual bounds

$$\frac{m^{1/2}}{D^{1/4}}\|\widehat{W} - W\|_F = \frac{m}{D^{1/4}}\delta_{\max} \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}}\frac{m^{5/4}}{D^{1/4}}, \tag{3.80}$$

$$\frac{m^{1/2}}{D^{1/4}}\frac{\Delta_{W,O}^{1/2}}{D^{1/2}} \leq \frac{m^{3/2}}{D^{3/4}}\delta_{\max} \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}}\frac{m^{7/4}}{D^{3/4}} \tag{3.81}$$

$$\frac{m^{1/2}}{D^{1/4}}\left\|\sum_{k=1}^{m} w_k - \hat{w}_k\right\|_2 = \frac{m^{3/2}}{D^{1/4}}\delta_{\max} \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}}\frac{m^{7/4}}{D^{1/4}}. \tag{3.82}$$

Comparison with 3.78 reveals that the first two bounds are smaller or identical to the initialization error. However, the error accumulation in $\|\sum_{k=1}^{m} w_k - \hat{w}_k\|_2$ only leads to a provable guarantee $\Delta_{W,1} \lesssim \frac{\epsilon^{1/2}}{\alpha^{1/4}}\frac{m^{7/4}}{D^{1/4}}$, which is slightly worse (by a factor $D^{1/2}$) than the error (3.78) at initialization time. Arguably, the term $\|\sum_{k=1}^{m} w_k - \hat{w}_k\|_2$ is also the term that benefits the most from random errors since the sum leads to an error cancelation whenever errors are not positively aligned. Let us mention that empirically we do not observe error alignments (cf. numerical Section 3.6). In particular, we show in Figure 3.5, that, in an empirical setting, the term corresponding to $\|\sum_{k=1}^{m} w_k - \hat{w}_k\|_2$ has the smallest contribution to $\Delta_{W,1}$. Lastly, before we discuss the proof of Theorem 3.4, let us mention that this discussion highlights the need for a tight analysis of the impact of the perturbation errors on the gradient descent iteration. Substantial technical effort was invested to *isolate error terms that suffer from error accumulations* in the following proofs. Based on these results, the issues mentioned in this discussion could then be avoided by simply assuming a random model for the errors introduced by Algorithm 3.2 (which aligns with our experiments).

3.4.3 Proof strategy for Theorem 3.4.

Let us now discuss our proof strategy and simultaneously provide an outline for the upcoming Sections 3.4.4-3.4.8. The proof of convergence of the gradient descent iteration 3.72 given by

$$\hat{\tau}^{(n+1)} = \hat{\tau}^{(n)} - \gamma\nabla_{\hat{\tau}}J(\hat{\tau}^{(n)})$$

is based on a linearization argument. Instead of directly analyzing the optimization landscape of the least-squares objective $J(\hat{\tau}) = \frac{1}{2N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}(\hat{f}(x_i, \hat{\tau}) - y_i)^2$ in (3.71), we analyze an idealized objective function, which we define as the quadratic function

$$J_*(\hat{\tau}) = (\hat{\tau} - \tau)^\top A(\hat{\tau} - \tau), \tag{3.83}$$

where

$$A := \frac{1}{2N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\nabla_{\hat{\tau}}\hat{f}(x_i, \tau)\nabla_{\hat{\tau}}\hat{f}(x_i, \tau)^\top. \tag{3.84}$$

We then prove two results. First, the objective J_* is *strictly convex* with high probability. Note that this implies the linear convergence of gradient descent schemes applied to J_* to the global minimum, which is attained at τ since $J_*(\tau) = (\tau - \tau)^\top A(\tau - \tau) = 0$. Second, in the vicinity of the true shifts τ, the gradient descent iteration $(\hat{\tau}^{(n+1)})_{n\in\mathbb{N}}$ remains close to a gradient descent iteration associated with the idealized objective J_*.

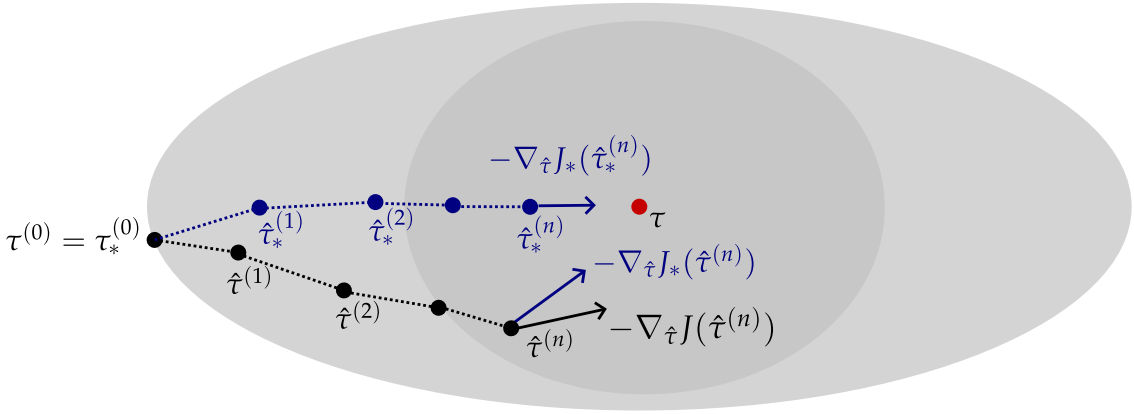

Figure 3.2: The idealized objective J_* is strictly convex around the true shifts τ. Hence, the associated GD iteration converges, i.e., $\hat{\tau}_*^{(n)} \to \tau$. We control the drift of the GD iteration $\hat{\tau}^{(n)}$ from the idealized iteration by carrying out a perturbation analysis estimating the deviation of the descent directions (i.e., $\|\nabla J(\hat{\tau}) - \nabla J_*(\hat{\tau})\|_2$) close to the true solution $\hat{\tau} \approx \tau$.

Strict convexity of J_*. To guarantee the strict convexity of J_*, we prove, by applying techniques from the NTK literature adapted to our setting, that the matrix A in (3.84) is positive definite with high probability. This relates to the results within Section 3.4.5, where we provide a lower bound on the spectrum of the expectation

$$E := \mathbb{E}_{X_1,\dots,X_{N_{\text{train}}} \sim \mathcal{N}(0,\mathrm{Id}_D)}[A].$$

More precisely, in Lemma 3.9, it is shown that

$$\lambda_m(E) \geq \omega - C(m-1)\left(\frac{\log m}{D}\right)^2,$$

where ω and C are constants depending only on g and τ_∞. The proof uses condition **(SNM2)** to characterize the mean E in terms of the Hermite coefficients of the shifted functions $(g(\cdot + \tau_k))_{k\in[m]}$ and the Gramian matrices $\widehat{G}_n$ for $n \geq 4$. This allows us to exploit the incoherence of the weight approximations $\hat{w}_1, \dots, \hat{w}_m$. Hermite decompositions will be introduced in Section 3.4.4. Lemma 3.9 implies the positive definiteness of E, which in turn implies the positive definiteness of A w.h.p., which we prove by applying a classical matrix Chernoff bound (cf. Lemma 3.15). The positive definiteness of A implies that minimizing J_* via the gradient descent iteration given by

$$\hat{\tau}_*^{(n+1)} = \hat{\tau}_*^{(n)} - \gamma \nabla_{\hat{\tau}} J_*(\hat{\tau}_*^{(n)}) = \hat{\tau}_*^{(n)} - \gamma A(\hat{\tau}_*^{(n)} - \tau) \tag{3.85}$$

with step-sizes $\gamma \leq 1/\|A\|$ will converge to the global minimum attained at $\hat{\tau}_* = \tau$.

Controlling the deviation between the two GD iterations. The second part of proving Theorem 3.4 is concerned with controlling the deviation of $\hat{\tau}^{(n)}$ from the idealized objective $\hat{\tau}_*^{(n)}$. Both iterations are started from the same initialization $\hat{\tau}^{(0)} = \hat{\tau}_*^{(0)}$, and will use the same step-size $\gamma > 0$. Any deviation will originate from the gradient updates. In Section 3.4.6, we prove Lemma 3.10, which states that the difference between the gradients of $\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2$ obeys

$$\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2 \lesssim \sqrt{m}\,\|\hat{\tau} - \tau\|_2^2 + \Delta_W \quad \text{w.h.p.},$$

where τ denotes the ground truth shifts of f. Hence, whenever the gradient descent iteration $\hat{\tau}^{(n)}$ is already close to the true shifts, i.e., $\|\hat{\tau}^{(n)} - \tau\|_2^2$ suitably small, then the gradients of J are close to the gradients of the idealized objective (which is converging linearly) up to an error term Δ_W caused by the weight errors. Building upon this result, Lemma 3.14 in Section 3.4.7 then shows that the deviation $\|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2$ between both GD iterations adheres to

$$\|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 \lesssim \xi^n \|\hat{\tau}^{(0)} - \tau\|_2 + (1 - \xi^n)\,\Delta_W \quad \text{for some } \xi \in [0, 1), \tag{3.86}$$

as long as $\|\hat{\tau}^{(n)} - \tau\|_2^2 \leq Cm^{-1/2}$ for an appropriate constant $C > 0$. One notable aspect of Lemma 3.14 is that we manage to *prevent any significant drift* between the iterations (despite the constant error Δ_W appearing in $\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2$). The proof of Theorem 3.4 is then given in Section 3.4.8. It combines the preceding statements and then, by using the triangle inequality. To be more precise, based on (3.86), we can upper bound the distance of the original gradient descent iteration (3.72) to τ via

$$\begin{aligned}\|\hat{\tau}^{(n)} - \tau\|_2 &\leq \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 + \|\hat{\tau}_*^{(n)} - \tau\|_2 \\ &\leq \xi^n \|\hat{\tau}^{(0)} - \tau\|_2 + (1 - \xi^n)\,\Delta_W + (1 - \gamma\lambda_m(A))^n \|\hat{\tau}_*^{(0)} - \tau\|_2 \to \Delta_W,\end{aligned}$$

for $n \to \infty$.

3.4.4 Preliminaries: Hermitian expansions

This section introduces Hermite polynomials [3, 100] which form a set of orthogonal functions over $\mathbb{R}$ when weighted by the Gaussian kernel $w_G(t) := \exp(-t^2/2)$.

Definition 3.1 (Hermite polynomials (cf. [3, p. 778]))**.** *Let $r \in \mathbb{N}$, then the r-th Hermite polynomial is defined as the function*

$$h_r(t) := \frac{1}{\sqrt{r!}}(-1)^r \exp\left(\frac{t^2}{2}\right)\frac{d^r}{dt^r}\exp\left(\frac{-t^2}{2}\right).$$

This definition of Hermite polynomials is also referred to as the "probabilist's Hermite polynomials".

A computation reveals the first few Hermite polynomials: $h_0(t) = 1, h_1(t) = t, h_2(t) = t^2 - 1, h_3(t) = t^3 - 3t$. The fact that Hermite polynomials are an orthogonal basis with respect to the Gaussian kernel w_G (cf. [3]), i.e., they satisfy

$$\int_{\mathbb{R}} h_r(t) h_s(t) w_G(t) dt = \sqrt{2\pi} s! \delta_{rs} \quad \text{for all } r, s \in \mathbb{N},$$

where δ_{rs} denotes the Kronecker delta, combined with the basis property allows us to decompose any function in $f \in L_2(\mathbb{R}, w_G)$ in terms of the Hermite polynomials:

Definition 3.2 (Hermite expansion)**.** *Consider any function $f \in L_2(\mathbb{R}, w_G)$, i.e., f satisfies*

$$\int_{\mathbb{R}} |f(t)|^2 w_G(t) dt < \infty. \tag{3.87}$$

Then the Hermite expansion of the function f is given by $f(t) = \sum_{r=0}^{\infty} \mu_r(f) h_r(t)$, where $\mu_r(f)$ is called the r-th Hermite coefficient of f and defined as

$$\mu_r(f) := \int_{\mathbb{R}} f(t) h_r(t) w_G(t) dt.$$

One particularly useful property of Hermite polynomials is the following relationship with Gaussian random variables:

Lemma 3.4 ([99, Lemma D.2]). *For two unit norm vectors $u, v \in \mathbb{R}^D$ and every $r, s \geq 0$ we have*

$$\mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}\left[h_r(u^\top X)h_s(v^\top X)\right] = \delta_{rs}\langle u, v\rangle^r,$$

where h_r, h_s denotes the r-th, s-th Hermite polynomial, respectively.

Inspired by the NTK literature, we can apply Lemma 3.4 to analyze expressions of the kind $\mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}[\nabla_{\hat{\tau}}\hat{f}(X_i,\tau)\nabla_{\hat{\tau}}\hat{f}(X_i,\tau)^\top]$, which occur for instance in the definition of the idealized objective J_* in (3.83). Each element of this expectation can be written as

$$\mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}\left[g^{(1)}_{\tau_k}(\hat{w}_k^\top X)g^{(1)}_{\tau_\ell}(\hat{w}_\ell^\top X)\right]$$

for a suitable $k, \ell \in [m]$, where we make use of the shorthand $g^{(1)}_{\tau_k}(\cdot) := g^{(1)}(\cdot + \tau_k)$. Assuming that $g^{(1)}_{\tau_k}$ and $g^{(1)}_{\tau_\ell}$ admit a Hermite expansion, we can use the statement in Lemma 3.4 which yields

$$\mathbb{E}\left[\left(\sum_{r=0}^{\infty}\mu_r(g^{(1)}_{\tau_k})h_r(\hat{w}_k^\top X)\right)\left(\sum_{r=0}^{\infty}\mu_r(g^{(1)}_{\tau_\ell})h_r(\hat{w}_\ell^\top X)\right)\right] = \sum_{r=0}^{\infty}\mu_r(g^{(1)}_{\tau_k})\mu_r(g^{(1)}_{\tau_\ell})\langle\hat{w}_k,\hat{w}_\ell\rangle^r.$$

Notably, the series on the right-hand side exposes the term $\langle\hat{w}_k,\hat{w}_\ell\rangle^r$ which enables us to leverage the incoherence conditions **(Inc1)-(Inc3)** associated with the vectors $\hat{w}_1,\ldots,\hat{w}_m$. Constructions of this type will play a fundamental role in the upcoming proofs in Section 3.4.5 - 3.4.6. Let us mention that the Hermite expansion of g and its shifted derivatives is well-defined under the conditions **(SNM1)** and **(SNM2)**. For the activation g, this follows directly from **(SNM2)**. We must check that the derivatives belong to $L_2(\mathbb{R}, w_G)$. Now, as per Assumption **(SNM1)**, the first three derivatives of g are bounded. Hence, also the shifted derivatives are bounded. It is easy to check that this implies that these functions lie within $L_2(\mathbb{R}, w_G)$:

Lemma 3.5 ([51, Lemma 5]). *Assume h is bounded, then $h \in L_2(\mathbb{R}, \omega_G)$ and the sum of squared Hermite coefficients of h is finite, i.e., we have*

$$\sum_{r\geq 0}\mu_r(h)^2 \leq \sqrt{2\pi}\|h\|_\infty^2.$$

Proof.

$$\int_{\mathbb{R}} h(t)^2\exp(-t^2/2)dt \leq \sqrt{2\pi}\|h\|_\infty^2 < \infty.$$

The second statement follows from the fact that $L_2(\mathbb{R}, \omega_G)$ is a Hilbert space and the Hermite polynomials form an orthogonal system within that space. □

Due to their definition, we can relate the Hermite coefficients of the activation function g with the Hermite coefficients of its derivatives. This applies to the case where derivatives have non-exponential tails.

Lemma 3.6. *Let $g \in L_2(\mathbb{R}, w_H)$ be K-times continuously differentiable and assume*

$$\lim_{t\to\infty} g^{(k)}(t)h_r(t)w_H(t) = \lim_{t\to-\infty} g^{(k)}(t)h_r(t)w_H(t) = 0 \tag{3.88}$$

for all $0 \leq k \leq K$. For any $r \in \mathbb{N}\cup\{0\}$ and $k \in [0,\ldots,K]$ we have

$$\mu_r(g^{(n)}) = \sqrt{\binom{n+r}{r}n!}\mu_{r+n}(g).$$

Proof of Lemma 3.6. The Hermite polynomials, weighted by $\exp(-t^2/2)$, satisfy the relation

$$\frac{d}{dt}\left(h_r(t)\exp\left(-\frac{t^2}{2}\right)\right) = \frac{d}{dt}\left(\sqrt{\frac{1}{r!}}(-1)^r\frac{d^r}{dt^r}\exp\left(-\frac{t^2}{2}\right)\right) = \sqrt{\frac{1}{r!}}(-1)^r\frac{d^{r+1}}{dt^{r+1}}\exp\left(-\frac{t^2}{2}\right)$$
$$= -\sqrt{r+1}\sqrt{\frac{1}{(r+1)!}}(-1)^{r+1}\frac{d^{r+1}}{dt^{r+1}}\exp\left(-\frac{t^2}{2}\right) = -\sqrt{r+1}h_{r+1}(t)\exp\left(-\frac{t^2}{2}\right).$$

Therefore, by applying integration by parts, we obtain

$$\mu_r(g^{(n)}) = \int g^{(n)}(t)h_r(t)w_H(t)dt = \left[g^{(n-1)}(t)h_r(t)w_H(t)\right]_{-\infty}^{\infty} - \int g^{(n-1)}(t)\frac{d}{dt}\left(h_r(t)w_H(t)\right)dt$$
$$= 0 + \sqrt{r+1}\int g^{(n-1)}(t)h_{r+1}(t)w_H(t)dt = \sqrt{r+1}\mu_{r+1}(g^{(n-1)}),$$

where the boundary terms vanish due to (3.88). Applying the same computation n-times, we obtain

$$\mu_r(g^{(n)}) = \sqrt{\prod_{\ell=1}^{n}(r+\ell)}\mu_{r+n}(f) = \sqrt{\frac{(r+n)!}{r!}}\mu_{r+n}(g).$$

□

We conclude the introduction to Hermite expansion by proving a direct implication of the two preceding statements, which will find application in the proofs of Lemma 3.12-3.13.

Lemma 3.7 ([51, Lemma 5])**.** *Assume that g fulfills the assumption **(SNM1)-(SNM2)** and that the shifts $(\tau_k)_{k\in[m]}, (\hat{\tau}_k)_{k\in[m]}$ are within $[-\tau_\infty, \tau_\infty]$. Then, for $R \geq 4$, we have*

$$\sum_{r\geq R} r|\max_{k,\ell\in[m]}\mu_r(g_{\tau_k})\mu_r(g^{(1)}_{\hat{\tau}_\ell})| < \infty. \tag{3.89}$$

Proof of Lemma 3.7. By applying Lemma 3.6 (whose condition is met due to **(SNM1)** - **(SNM2)**), we immediately get that, for all $r \geq R$,

$$\mu_r(g_{\tau_k})\mu_r(g^{(1)}_{\hat{\tau}_\ell}) = \left(3!\binom{r}{r-3}\right)^{-1/2}\mu_{r-3}(g^{(3)}_{\tau_k})\cdot\left(2!\binom{r}{r-2}\right)^{-1/2}\mu_{r-2}(g^{(3)}_{\hat{\tau}_\ell})$$
$$= \left((r-2)(r-1)^2r^2\right)^{-1/2}\mu_{r-3}(g^{(3)}_{\tau_k})\mu_{r-2}(g^{(3)}_{\hat{\tau}_\ell}).$$

Plugging this into (3.89) yields

$$\sum_{r\geq R} r|\max_{k,\ell\in[m]}\mu_r(g_{\tau_k})\mu_r(g^{(1)}_{\hat{\tau}_\ell})| \leq \sum_{r\geq R}\frac{1}{\sqrt{r-2}(r-1)}\max_{k,\ell\in[m]}\mu_{r-3}(g^{(3)}_{\tau_k})\mu_{r-2}(g^{(3)}_{\hat{\tau}_\ell})$$
$$\leq \sum_{r\geq R-3}\max_{k\in[m]}\frac{1}{r^{3/2}}\mu_r(g^{(3)}_{\tau_k})^2 + \sum_{r\geq R-2}\max_{\ell\in[m]}\frac{1}{r^{3/2}}\mu_r(g^{(3)}_{\hat{\tau}_\ell})^2,$$

where the second line follows by applying Cauchy-Schwarz. Using assumption **(SNM1)**, according to Lemma 3.5, then gives $\max_{\tau\in[-\tau_\infty,\tau_\infty]}\mu_r(g^{(3)}_\tau)^2 \leq C$ for all $r \geq 0$ and some constant $C > 0$. Therefore, we can conclude with

$$\sum_{r\geq R} r|\max_{k,\ell\in[m]}\mu_r(g_{\tau_k})\mu_r(g^{(1)}_{\hat{\tau}_\ell})| \leq 2C\sum_{r\geq 1}\frac{1}{r^{3/2}} \leq 6C < \infty.$$

□

3.4.5 Strict convexity of the idealized objective in expectation

In this section, we apply Hermite decompositions, in particular, the statement in Lemma 3.4, to prove the strict convexity of J_* (cf. (3.83)) in expectation. As outlined above, this will be done by lower bounding the spectrum of the matrix

$$E := \mathbb{E}_{X_1,\dots,X_{N_{\text{train}}}\sim\mathcal{N}(0,\mathrm{Id}_D)}[A], \tag{3.90}$$

where

$$A := \frac{1}{2N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \nabla_{\hat{\tau}} \hat{f}(X_i,\tau) \nabla_{\hat{\tau}} \hat{f}(X_i,\tau)^\top.$$

We begin with a decomposition of E into a series of positive semidefinite Gramian matrices as shown in Lemma 3.8.

Lemma 3.8 ([51, Lemma 7])**.** *Assume that **(SNM1)** holds, and let E be defined as in* (3.90)*. Then, we have*

$$E = \frac{1}{2}\sum_{r=0}^{\infty} T_r T_r^\top, \quad \text{where } T_r := \begin{bmatrix} \mu_r(g_{\tau_1}^{(1)})\,\mathrm{vec}(\hat{w}_1^{\otimes r}) \\ \vdots \\ \mu_r(g_{\tau_m}^{(1)})\,\mathrm{vec}(\hat{w}_m^{\otimes r}) \end{bmatrix} \in \mathbb{R}^{m\times D^r}.$$

In particular, we have $E \succcurlyeq \frac{1}{2}\sum_{r\in\mathcal{R}} T_r T_r^\top$ for any subset $\mathcal{R} \subseteq \mathbb{N}_{\geq 1}$, where $A \succcurlyeq B$ means $A - B$ is positive semidefinite.

Proof. The matrix A can be written as

$$A_{k\ell} = \frac{1}{2N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} g^{(1)}(\hat{w}_k^\top X_i + \tau_k) g^{(1)}(\hat{w}_\ell^\top X_i + \tau_\ell)$$

and the corresponding expectation reads

$$E_{k\ell} = \frac{1}{2}\mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}\left[g_{\tau_k}^{(1)}(\hat{w}_k^\top X) g_{\tau_\ell}^{(1)}(\hat{w}_\ell^\top X)\right].$$

Now, note that $g_\tau^{(1)} = g^{(1)}(\cdot + \tau) \in L_2(\mathbb{R}, w_H)$ for any $\tau \in \mathbb{R}$ by **(SNM1)** and Lemma 3.5. Hence, $g_\tau^{(1)}$ has a Hermitian expansion, and we can write

$$E_{k\ell} = \frac{1}{2}\mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}\left[\left(\sum_{r=0}^{\infty}\mu_r(g_{\tau_k}^{(1)})h_r(\hat{w}_k^\top X)\right)\left(\sum_{r=0}^{\infty}\mu_r(g_{\tau_\ell}^{(1)})h_r(\hat{w}_\ell^\top X)\right)\right].$$

Using now Lemma 3.4 to express expectations of scalar products of Hermite polynomials, we obtain

$$E_{k\ell} = \frac{1}{2}\sum_{r=0}^{\infty}\mu_r(g_{\tau_k}^{(1)})\mu_r(g_{\tau_\ell}^{(1)})\langle\hat{w}_k,\hat{w}_\ell\rangle^r,$$

which can equivalently be written as $\frac{1}{2}\sum_{r=0}^{\infty} T_r T_r^\top$. The second part of the statement follows from the fact that each individual matrix $T_r T_r^\top$ is a positive semidefinite Gramian matrix. □

Remark 3.4. *Note that an identical statement can be proven for the matrix E_2 with entries given by*

$$(E_2)_{k\ell} = \mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}\left[g^{(2)}_{\tau_k}(w_k^\top X)g^{(2)}_{\tau_\ell}(w_\ell^\top X)\right],$$

yielding a decomposition with

$$E_2 = \frac{1}{2}\sum_{r=0}^{\infty} T_r T_r^\top, \quad \text{where } T_r' := \begin{bmatrix} \mu_r(g^{(2)}_{\tau_1})\operatorname{vec}(w_1^{\otimes r}) \\ \vdots \\ \mu_r(g^{(2)}_{\tau_m})\operatorname{vec}(w_m^{\otimes r}) \end{bmatrix} \in \mathbb{R}^{m\times D^r}.$$

This follows from the fact that $g^{(2)}$ is bounded and, therefore, also admits a Hermite expansion. This will play a role in the proof of Theorem 3.5. Let us remark that the following discussion also applies when we replace E with E_2 and rename the variables accordingly.

Let us provide some interpretation of the preceding result. Denote by H_r^n the matrix whose entries are defined as

$$(H_r^n)_{k\ell} := \mu_r(g^{(n)}_{\tau_k})\mu_r(g^{(n)}_{\tau_\ell}) \quad \text{for } n \in \{1,2\}. \tag{3.91}$$

Lemma 3.8 shows that the expectation E can be written as the matrix series

$$E = \frac{1}{2}\sum_{r=0}^{\infty} H_r^1 \odot \widehat{G}_r,$$

where $\widehat{G}_r$ denotes the Gramian matrix of the system $\{\hat{w}_k^{\otimes r} | k \in [m]\}$ (cf. Section 2.6.1) if $r > 0$ and otherwise $\widehat{G}_0 = \mathbf{1}\otimes\mathbf{1}$. Due to the incoherence of the approximated weights, the off-diagonal entries of higher-order Gramian matrices will vanish at an exponential rate in r, i.e., we have $\widehat{G}_r \approx \mathrm{Id}_m$ for r sufficiently large. Hence, the tail of the matrices series is close to a sum of diagonal matrices. Notably, by the remark in the preceding statement, it is sufficient to prove positive definiteness for a tail series. Assuming the diagonal elements of a tail series remain positive and the off-diagonal elements become sufficiently small, the positive definiteness can be proven by combining Weyl's inequality (cf. [132]) with Theorem 2.9. The diagonal elements of any tail series have the form

$$\left(\frac{1}{2}\sum_{r=\mathcal{R}}^{\infty} H_r \odot \widehat{G}_r\right)_{kk} = \left(\frac{1}{2}\sum_{r=\mathcal{R}}^{\infty} H_r\right)_{kk} = \frac{1}{2}\sum_{r=\mathcal{R}}^{\infty}\mu_r(g^{(1)}_{\tau_k})^2, \quad \text{for all } k \in [m].$$

The assumption **(SNM2)** states that $g^{(1)}$ is not a polynomial of degree three or less. Therefore, $g^{(1)}_{\tau_k}$ can not be represented entirely by the first three Hermite polynomials. The following statement leverages this assumption to show that $\frac{1}{2}\sum_{r=4}^{\infty} H_r \odot \widehat{G}_r$ is positive definite.

Lemma 3.9 ([51, Lemma 8])**.** *Let E be defined as in* (3.90) *and assume that the approximated weights satisfy $\|\hat{w}_k\|_2 = 1$ and **(Inc2)** for some universal constant c_2. Furthermore, assume the activation function adheres to **(SNM1)** and **(SNM2)**. Then, we have*

$$\lambda_m(E) \geq \omega - C(m-1)\left(\frac{\log m}{D}\right)^2, \tag{3.92}$$

where ω and C are constants depending only on g and τ_∞. Specifically, we have

$$\omega = \frac{1}{2}\min_{\tau\in[-\bar{\tau}_\infty,\bar{\tau}_\infty]}\sum_{r\geq 4}\left(\mu_r(g^{(1)}(\cdot+\tau))\right)^2,$$

$$C = \frac{1}{2}c_2^2\max_{\tau,\widetilde{\tau}\in[-\tau_\infty,\tau_\infty]}\sum_{r\geq 4}\left|\mu_r(g^{(1)}_{\tau})\mu_r(g^{(1)}_{\widetilde{\tau}})\right|.$$

Proof of Lemma 3.9. To simplify the notation, we introduce the shorthand $\mu_{r,k} := \mu_r(g^{(1)}_{\tau_k})$. By Lemma 3.8 we have $E \succcurlyeq \frac{1}{2}\sum_{r\geq 4} T_r T_r^\top$, so we concentrate on the expression on the right-hand side. As $\|\hat{w}_k\|_2 = 1$ for all $k \in [m]$, we first note that we can rewrite $\frac{1}{2}\sum_{r\geq 4} T_r T_r^\top$ as $\frac{1}{2}\sum_{r\geq 4} T_r T_r^\top = D_4 + O_4$, where the matrix D_4 is given by

$$D_4 := \frac{1}{2}\operatorname{Diag}\Big(\sum_{r\geq 4}\mu_{r,1}^2, \ldots, \sum_{r\geq 4}\mu_{r,m}^2\Big)$$

and the remainder O_4 equals $\frac{1}{2}\sum_{r\in 4} T_r T_r^\top$ with its diagonal set to 0. To show (3.92), we compute a lower eigenvalue bound for D_4 and an upper eigenvalue bound for O_4 independently, and then complete the argument with Weyl's eigenvalue perturbation bound [132]. The smallest eigenvalue of D_4 can be read from the diagonal and is given by

$$\lambda_{\min}(D_4) = \frac{1}{2}\min_{k\in[m]}\sum_{r\geq 4}\mu_{r,k}^2 \geq \omega > 0,$$

where the last inequality, stating $\omega > 0$, follows from **(SNM2)**. For the spectral norm of O_4 we use L_1/L_∞-Cauchy-Schwarz inequalities and $\|\hat{w}_k\|_2 = 1$ for all $k \in [m]$. Specifically, for any unit norm vector u, we have

$$\begin{aligned} u^\top O_4 u &= \frac{1}{2}\sum_{k=1}^m \sum_{\ell\neq k} u_k u_\ell \sum_{r\geq 4}\mu_{r,k}\mu_{r,\ell}\langle \hat{w}_k, \hat{w}_\ell\rangle^r \\ &\leq \frac{1}{2}\sum_{k=1}^m \sum_{\ell\neq k} |u_k|\,|u_\ell| \sum_{r\geq 4}|\mu_{r,k}\mu_{r,\ell}|\,|\langle \hat{w}_k, \hat{w}_\ell\rangle|^r . \end{aligned}$$

By dragging out the maximum of the sums over Hermitian coefficients, we further bound

$$u^\top O_4 u \leq \Big(\frac{1}{2}\max_{\tau,\tilde{\tau}\in[-\tau_\infty,\tau_\infty]}\sum_{r\geq 4}\Big|\mu_r(g^{(1)}_\tau)\mu_r(g^{(1)}_{\tilde{\tau}})\Big|\Big)\sum_{k=1}^m\sum_{\ell\neq k}|u_k|\,|u_\ell|\,|\langle \hat{w}_k, \hat{w}_\ell\rangle|^4 .$$

The trailing factor is, for all unit norm u, bounded by the spectral norm of the matrix

$$(\widehat{O}_4)_{ij} := \begin{cases} 0, & \text{if } i = j, \\ \big|\langle \hat{w}_i, \hat{w}_j\rangle\big|^4, & \text{else .} \end{cases} \tag{3.93}$$

Therefore, we have $u^\top O_4 u \leq C_{g,\tau_\infty}\|\widehat{O}_4\|$ for all unit norm u, and with the constant C_{g,τ_∞} given as

$$C_{g,\tau_\infty} = \frac{1}{2}\max_{\tau,\tilde{\tau}\in[-\tau_\infty,\tau_\infty]}\sum_{r\geq 4}\Big|\mu_r(g^{(1)}_\tau)\mu_r(g^{(1)}_{\tilde{\tau}})\Big|,$$

and only dependent on g and the shift bound τ_∞. By Gershgorin's circle theorem, we further have

$$\|\widehat{O}_4\| \leq \max_{k\in[m]}\sum_{\ell\neq k}^m |(\widehat{O}_4)_{k\ell}| \leq (m-1)\left(\frac{c_2\log m}{D}\right)^2,$$

where we used the fact that $\hat{w}_1, \ldots, \hat{w}_m$ satisfy **(Inc2)**. □

Remark 3.5. *Let us mention that the lower bound* (3.92) *can be improved by minor adaptions of the proof to*

$$\lambda_m(E) \geq \frac{1}{2}\min_{\tau\in[-\bar{\tau}_\infty,\bar{\tau}_\infty]}\sum_{r\geq R}\Big(\mu_r(g^{(1)}(\cdot+\tau))\Big)^2 - C(m-1)\left(\frac{\log m}{D}\right)^{(R+1)/2}, \tag{3.94}$$

for a suitable constant $C > 0$, assuming $g^{(1)}$ is not a polynomial of degree $R \geq 3$ or less. Differently put, as long as we can guarantee that the Hermite coefficients do not vanish too quickly, we can further benefit from the incoherence of the weights $w_1, \ldots, w_m$.

Discussion of assumption (SNM3). As already prefaced in Remark 3.4, the proof technique used to lower bound the spectrum of E can also be applied to find a lower bound to the spectrum of the second moment matrix E_2. This enables us to provide a lower bound on the m-th singular value of the matrix $\lambda_m\left(\mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id})}[\mathrm{vec}(\nabla^2 f(X))^{\otimes 2}]\right)$ which equates to condition **(SNM3)**. Notably, assumptions similar to **(SNM3)** are integral in other related works such as [54].

Theorem 3.5. *Consider a shallow neural network f as described in Section 3.1.1. Assume the activation g satisfies **(SNM1)-(SNM2)**, and that the network weights $w_1,\dots,w_m \in \mathbb{S}^{D-1}$ fulfill conditions **(Inc1)-(Inc2)** of Definition 2.3 with constants c_2, c_3. Then, there exists a constant $C>0$ depending only on $c_2, g^{(2)}$, such that for $m\log^2 m \leq CD^2$ we have*

$$\lambda_m\left(\mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id})}[\mathrm{vec}(\nabla^2 f(X))^{\otimes 2}]\right) \geq c_3^{-1}\lambda_m(E_2) > 0,$$

*where E_2 is a positive definite matrix with entries $(E_2)_{k\ell} = \mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}\left[g_{\tau_k}^{(2)}(w_k^\top X)g_{\tau_\ell}^{(2)}(w_\ell^\top X)\right]$. In particular, under these assumptions, the condition **(SNM3)** is fulfilled for a constant $\alpha>0$ independent of m, D.*

Proof. Throughout the proof we denote $\mathbb{E}[\cdot] = \mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id})}[\cdot]$ and $\mu_{r,k} = \mu_r(g_{\tau_k}^{(2)})$ for all $k\in[m]$. We begin by expanding the outer product, which yields

$$\begin{aligned}\mathbb{E}\left[\mathrm{vec}(\nabla^2 f(X))^{\otimes 2}\right] &= \mathbb{E}\left[\left(\sum_{k=1}^m g^{(2)}(\langle w_k^\top X+\tau_k\rangle)\,\mathrm{vec}(w_k^{\otimes 2})\right)^{\otimes 2}\right]\\ &= \sum_{k,\ell=1}^m \mathbb{E}\left[g^{(2)}(\langle w_k^\top X+\tau_k\rangle)g^{(2)}(\langle w_\ell^\top X+\tau_\ell\rangle)\right]\mathrm{vec}(w_k^{\otimes 2})\otimes\mathrm{vec}(w_\ell^{\otimes 2}).\end{aligned} \tag{3.95}$$

Due to Lemma 3.5 and (**(SNM1)**), the shifted second derivative of the activation admits a Hermite expansion such that

$$g^{(2)}(\langle w_\ell^\top X+\tau_\ell\rangle) = \sum_{r=0}^\infty \mu_{k,r}h_r(w_k^\top X),$$

where h_r denotes the r-th Hermite polynomial. Plugging this expansion into (3.95) gives

$$\begin{aligned}\mathbb{E}\left[\mathrm{vec}(\nabla^2 f(X))^{\otimes 2}\right] &= \sum_{k,\ell=1}^m \mathbb{E}\left[\sum_{r,s=0}^\infty \mu_{k,r}h_r(w_k^\top X)\mu_{\ell,s}h_s(w_\ell^\top X)\right]\mathrm{vec}(w_k^{\otimes 2})\otimes\mathrm{vec}(w_\ell^{\otimes 2})\\ &= \sum_{r=0}^\infty\sum_{k,\ell=1}^m \mu_{r,k}\mu_{r,\ell}\cdot(G_r)_{k\ell}\,\mathrm{vec}(w_k^{\otimes 2})\otimes\mathrm{vec}(w_\ell^{\otimes 2}),\end{aligned} \tag{3.96}$$

where the last step follows from Lemma 3.4. Here, $G_0 = \vec{\mathbf{1}}\otimes\vec{\mathbf{1}}$ and G_r denotes the Gramian matrix of the system $\{w_k^{\otimes r} | k\in[m]\}$. Denote now $W_2 = \left[\mathrm{vec}(w_1^{\otimes 2})\dots\mathrm{vec}(w_m^{\otimes 2})\right]\in\mathbb{R}^{D^2\times m}$, then (3.96) can be rephrased as the matrix product

$$\sum_{r=0}^\infty\sum_{k,\ell=1}^m \mu_{r,k}\mu_{r,\ell}(G_r)_{k\ell}\,\mathrm{vec}(w_k^{\otimes 2})\otimes\mathrm{vec}(w_\ell^{\otimes 2}) = W_2\left(\sum_{r=0}^\infty H_r^2\odot G_r\right)W_2^\top, \tag{3.97}$$

where H_r^2 is the matrix storing the Hermite coefficients, i.e., $(H_r^n)_{k\ell} := \mu_r(g_{\tau_k}^{(n)})\mu_r(g_{\tau_\ell}^{(n)})$ (cf. Definition (3.91)). Note that the matrix $E_2 := \left(\sum_{r=0}^\infty H_r^2\odot G_r\right)$ is positive definite, with a lower

bound on the smallest singular value that obeys

$$\lambda_m(E_2) \geq \frac{1}{2} \min_{\tau \in [-\bar{\tau}_\infty, \bar{\tau}_\infty]} \sum_{r \geq R} \left(\mu_r(g^{(2)}(\cdot + \tau)) \right)^2 - \tilde{C}(m-1) \left(\frac{\log m}{D} \right)^{(R+1)/2}, \tag{3.98}$$

where $R \in \mathbb{N}$ is the maximal number such that $g^{(2)}$ is not a polynomial of degree R or less and

$$\tilde{C} = \frac{1}{2} c_2^2 \max_{\tau, \widetilde{\tau} \in [-\tau_\infty, \tau_\infty]} \sum_{r \geq 4} \left| \mu_r(g^{(2)}_\tau) \mu_r(g^{(2)}_{\widetilde{\tau}}) \right|. \tag{3.99}$$

This follows directly from the proof of Lemma 3.9, Remark 3.4, and Remark 3.5 since $g^{(2)}$ satisfies the same relevant conditions as $g^{(1)}$ and the weights $w_1, \ldots, w_m$ are satisfying the incoherence condition **(Inc2)**. In particular, since $g^{(2)}$ is not a polynomial of degree three or less, the bound (3.98) is fulfilled for $R = 3$. Hence, similarly to Lemma 3.9, we get

$$\lambda_m(E_2) \geq \frac{1}{2} \min_{\tau \in [-\bar{\tau}_\infty, \bar{\tau}_\infty]} \sum_{r \geq 4} \left(\mu_r(g^{(2)}(\cdot + \tau)) \right)^2 - \tilde{C}(m-1) \left(\frac{\log m}{D} \right)^2.$$

The term $\frac{1}{2} \min_{\tau \in [-\bar{\tau}_\infty, \bar{\tau}_\infty]} \sum_{r \geq 4}$ is constant, only depending on $g^{(2)}$, whereas the right-hand side $\left(\mu_r(g^{(2)}(\cdot + \tau)) \right)^2 - \tilde{C}(m-1) \left(\frac{\log m}{D} \right)^2$ is arbitrarily small whenever $m \leq CD^2 \log^2 m$ holds for a suitable constant C, which reflects our initial setting. Thus, E_2 is positive definite with smallest eigenvalue denoted by $\lambda_m(E_2)$ that only depends on $c_2, g^{(2)}$. Denote now by $V \Sigma V^\top$ the eigendecomposition of E_2, then we can continue from (3.96) with

$$\mathbb{E}\left[\mathrm{vec}(\nabla^2 f(X))^{\otimes 2} \right] = W_2 E_2 W_2^\top = W_2 V \Sigma V^\top W_2^\top.$$

We can now provide a lower bound on the minimal eigenvalue of the expectation in terms of $\lambda_m(E_2)$ and $c_3^{-1} = \|G_2^{-1}\|$. Denote by $\Sigma^{1/2}$ the diagonal matrix such that $\Sigma^{1/2}\Sigma^{1/2} = \Sigma$, then

$$\begin{aligned}
\lambda_m\left(\mathbb{E}\left[\mathrm{vec}(\nabla^2 f(X))^{\otimes 2} \right] \right) &= \lambda_m\left(W_2 V \Sigma V^\top W_2^\top \right) \\
&= \lambda_m\left((W_2 V \Sigma^{1/2})(W_2 V \Sigma^{1/2})^\top \right) \\
&= \lambda_m\left((W_2 V \Sigma^{1/2})^\top (W_2 V \Sigma^{1/2}) \right) \\
&= \lambda_m\left(\Sigma^{1/2} V^\top G_2 V \Sigma^{1/2} \right),
\end{aligned}$$

where the second inequality follows from the identity $\lambda_m(XX^T) = \lambda_m(X^T X)$ and the last step uses the fact that $W_2^\top W_2 = G_2$. Since $w_1, \ldots, w_m$ satisfy **(Inc3)**, the spectrum of the Gramian matrix G_2 is bounded by c_3^{-1}, such that

$$\lambda_m\left(\Sigma^{1/2} V^\top G_2 V \Sigma^{1/2} \right) \geq \lambda_m(\Sigma^{1/2}\Sigma^{1/2}) \cdot \lambda_m(V^\top G_2 V) = \lambda_m(E_2)\lambda_m(G_2) \geq c_3^{-1} \lambda_m(E_2).$$

Here, the first inequality follows from a generalization of Sylvester's law of inertia [101, Theorem 1]. □

3.4.6 Controlling the difference between the gradient upgrades

In this section, we derive an upper bound on the difference $\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2$ between the gradients of the original objective J in (3.71) and the idealized objective J_* in (3.83). This will allow us to characterize the divergence between the two associated gradient descent iterations, which previously defined (3.72) and (3.85). For more details, we refer to the discussion in Section 3.4.2. Our main result in this section is stated below.

Lemma 3.10 (cf. [51, Lemma 13])**.** *Consider a shallow neural network f with unit norm weights described by the matrix $W = [w_1|\dots|w_m]$, shifts $\tau_1, \dots, \tau_m \in [-\tau_\infty, \tau_\infty]$ stored in τ and an activation function g that adheres to **(SNM1)-(SNM2)** with $D \leq m$. Furthermore, consider J, J_* given by* (3.71), (3.83) *constructed with $N_{train} \geq m$ network evaluations $y_1, \dots, y_{N_{train}}$ of f where $y_i = f(X_i)$ and $X_1, \dots, X_N \sim \mathcal{N}(0, \mathrm{Id}_D)$. Denote by $\hat{f}$ an approximation to f constructed from parameters $\widehat{W} = [\hat{w}_1|\dots|\hat{w}_m], \hat{\tau}$ as described above with $\|\hat{w}_k\| = 1$ for all $k \in [m]$. Then, there exists an absolute constant $C > 0$ and D_0 such that, for dimension $D \geq D_0$, the difference between the gradients of J and of the idealized objective J_* obeys*

$$\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2 \leq 2\kappa^2\sqrt{m}\,\|\hat{\tau} - \tau\|_2^2 + C\Delta_{W,1} + \left(\frac{m^3\delta_{\max}^2 t}{N_{train}}\right)^{1/2} \tag{3.100}$$

for $t > 0$ with probability at least $1 - 2m^2\exp\left(-\frac{t}{C\kappa^4}\right)$ and where

$$\Delta_{W,1} \leq \frac{m^{1/2}\log(m)^{3/4}}{D^{1/4}}\left[\|\widehat{W} - W\|_F + \frac{\Delta_{W,O}^{1/2}}{D^{1/2}} + \left\|\sum_{k=1}^m w_k - \hat{w}_k\right\|_2\right]. \tag{3.101}$$

The proof relies on several auxiliary statements and has been deferred to the end of the section. Due to the technical nature of the results within this section, we start with a short sketch of the proof of Lemma 3.10, which helps to motivate the auxiliary statements.

Proof sketch of Lemma 3.10. In the first step, we separate the bound on $\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2$ into two terms Δ_τ, Δ_W, such that

$$\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2 \leq \Delta_\tau + \Delta_W.$$

Here, the error Δ_τ is caused by our linearization argument and only scales with the deviation of $\hat{\tau}$ from the true shifts τ, whereas Δ_W represents the error due to the weight approximation. Using a chain of elementary algebraic arguments that follow from **(SNM1)**, we can estimate $\Delta_\tau \leq 2\kappa^2\sqrt{m}\|\hat{\tau} - \tau\|_2^2$. The more challenging part is the bound Δ_W, which needs to control the term

$$\left\|\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\left(\hat{f}(X_i, \tau) - f(X_i, \tau)\right)\nabla_{\hat{\tau}}\hat{f}(X_i, \hat{\tau})\right\|_2 \leq \Delta_W.$$

Assuming equality, we express Δ_W^2 as sum of random variables

$$\Delta_W^2 = \sum_{\ell=1}^m\left[\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\sum_{k=1}^m Z_{ik\ell}\right]^2,$$

where $Z_{ik\ell} := \varphi_{k,\ell}(X_i)$ and

$$\varphi_{k,\ell}(x) = \Big(g(x^\top\hat{w}_k + \tau_k) - g(x^\top w_k + \tau_k)\Big)g^{(1)}(x^\top\hat{w}_\ell + \hat{\tau}_\ell).$$

To control Δ_W^2, we first make separate the means $\mathbb{E}[Z_{ik\ell}] = \mathbb{E}_{X\sim\mathcal{N}(0,\mathrm{Id}_D)}[\varphi_{k\ell}(X)]$ of the sum according to

$$\Delta_W^2 = \sum_{\ell=1}^m\left[\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\sum_{k=1}^m Z_{ik\ell}\right]^2 \leq 2\Delta_{W,1}^2 + 2\Delta_{W,2}^2,$$

where

$$\Delta_{W,1}^2 := \sum_{\ell=1}^{m} \left[\frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \sum_{k=1}^{m} \mathbb{E}[Z_{ik\ell}] \right]^2, \Delta_{W,2}^2 := \sum_{\ell=1}^{m} \left[\frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \sum_{k=1}^{m} (Z_{ik\ell} - \mathbb{E}[Z_{ik\ell}]) \right]^2.$$

Establishing that $Z_{ik\ell} - \mathbb{E}[Z_{ik\ell}]$ is sub-Gaussian for all $k, \ell \in [m]$ then enables us to use a concentration argument showing that, for suitable $C > 0, t \geq 0$, we have

$$\mathbb{P}\left(\Delta_{W,2}^2 \leq \frac{m^3 \delta_{\max}^2 t}{N_{\text{train}}} \right) \geq 1 - 2m^2 \exp\left(-\frac{t}{C\kappa^4} \right).$$

To bound $\Delta_{W,1}^2$, we make use of the Hermite expansion of the function $\varphi(x)$, which allows us once again to exploit the incoherence of the weights $(w_k)_{k\in[m]}, (\hat{w}_k)_{k\in[m]}$. More precisely, we receive the identity $\Delta_{W,1}^2 = \sum_{\ell=1}^{m} \left(\sum_{r\geq 1} S_{r,\ell} \right)^2$, where $S_{r,\ell}$ is defined as

$$S_{r,\ell} := \sum_{k=1}^{m} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \left(\langle \hat{w}_k, \hat{w}_\ell \rangle^r - \langle w_k, \hat{w}_\ell \rangle^r \right) \quad \text{for all } k, \ell \in [m] \tag{3.102}$$

where $\mu_r(g_{\tau_k}), \mu_r(g_{\hat{\tau}_\ell}^{(1)})$ denotes the k-th and ℓ-th Hermite coefficient of the function g_{τ_k}, $g_{\hat{\tau}_\ell}^{(1)}$, respectively. We split the series $\sum_{\ell=1}^{m} \left(\sum_{r\geq 1} S_{r,\ell} \right)^2$ into three parts using the inequality

$$\sum_{\ell=1}^{m} \left(\sum_{r\geq 1} S_{r,\ell} \right)^2 \leq 2 \sum_{\ell=1}^{m} S_{1,\ell}^2 + \sum_{r=2}^{R-1} 2^r \sum_{\ell=1}^{m} S_{r,\ell}^2 + 2^R \sum_{\ell=1}^{m} \left(\sum_{r\geq R} S_{r,\ell} \right)^2 \quad \text{for } R \geq 2. \tag{3.103}$$

Based on this representation, we prove three individual statements, each of which targets an individual term in the inequality above. In Lemma 3.13, we show that the tail series $2^R \sum_{\ell=1}^{m} \left(\sum_{r\geq R} S_{r,\ell} \right)^2$ is neglectable for $R \geq 9$ when compared to the first two sums. The two dominant terms are then bounded as

$$\sum_{r=2}^{R} 2^r \sum_{\ell=1}^{m} S_{r,\ell}^2 \lesssim \frac{m \log m}{D} \left(\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{1/2} \Delta_{W,O} \right),$$

by Lemma 3.11 (see (3.104) for a definition of $\Delta_{W,O}$), and

$$\sum_{\ell=1}^{m} S_{1,\ell}^2 \lesssim m \left(\frac{\log m}{D} \right)^{1/2} \left\| \sum_{k=1}^{m} w_k - \hat{w}_k \right\|_2^2,$$

by Lemma 3.12. In combination, these bounds give rise to the result in (3.101).

Finding an upper bound for the series in (3.103)**.** In the upcoming results, we will reuse the shorthand $\Delta_{W,O}$ from (3.17) (cf. Theorem 3.2), which was defined as

$$\Delta_{W,O} = \sum_{k\neq k'}^{m} \left| \langle \hat{w}_k - w_k, \hat{w}_{k'} - w_k' \rangle \right|. \tag{3.104}$$

We start with a bound on the inner sum in (3.103):

Lemma 3.11 (cf. [51, Lemma 9])**.** *Consider weights and approximated weights $(w_k)_{k\in[m]}, (\hat{w}_k)_{k\in[m]}$ of unit norm as before that both fulfill* ***(Inc2)*** *and (i) in Lemma 3.2, as well shifts $(\tau_k)_{k\in[m]}, (\hat{\tau}_k)_{k\in[m]}$*

within $[-\tau_\infty, \tau_\infty]$ *for some* $\tau_\infty < \infty$. *Let* $S_{r,\ell}$ *be defined as in* (3.102) *and assume that* g *fulfills the assumption* ***(SNM1)*** *-* ***(SNM2)***. *Then, there exists a constant* $C > 0$ *such that, for* $m \geq D$,

$$\sum_{\ell=1}^{m} S_{r,\ell}^2 \leq C r^2 \max_{k,\ell \in [m]} \mu_r(g_{\hat{\tau}_\ell}^{(1)})^2 \mu_r(g_{\tau_k})^2 \left(1 + m \left(\frac{\log m}{D}\right)^{r/2}\right) \cdot \left[\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D}\right)^{(r-1)/2} \Delta_{W,O}\right].$$

Furthermore, for any fixed $R \geq 2$, *we have*

$$\sum_{r=2}^{R} 2^r \sum_{\ell=1}^{m} S_{r,\ell}^2 \leq \frac{Cm \log m}{D} \left(\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D}\right)^{1/2} \Delta_{W,O}\right),$$

where the constant $C > 0$ *additionally depends on* R.

Proof of Lemma 3.11. Throughout this proof, we use the convention that, for any vector, we have $v^{\otimes 0} = 1$, $1 \otimes 1 = 1$, and $v \otimes 1 = 1 \otimes v = v$, which will be relevant for the case $r = 1$. We start with a chain of equalities that uses elementary properties of the Frobenius inner product:

$$\begin{aligned}
\sum_{\ell=1}^{m} S_{r,\ell}^2 &= \sum_{\ell=1}^{m} \left[\sum_{k=1}^{m} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \left(\langle \hat{w}_k, \hat{w}_\ell \rangle^r - \langle w_k, \hat{w}_\ell \rangle^r\right)\right]^2 \\
&= \sum_{\ell=1}^{m} \left[\sum_{k=1}^{m} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \langle \hat{w}_k - w_k, \hat{w}_\ell \rangle \left(\sum_{i=1}^{r} \langle \hat{w}_k, \hat{w}_\ell \rangle^{r-i} \langle w_k, \hat{w}_\ell \rangle^{i-1}\right)\right]^2 \\
&= \sum_{\ell=1}^{m} \left[\sum_{k=1}^{m} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \langle \hat{w}_k - w_k, \hat{w}_\ell \rangle \left\langle \sum_{i=1}^{r} \hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)}, \hat{w}_\ell^{\otimes r-1} \right\rangle\right]^2 \\
&= \sum_{\ell=1}^{m} \left[\sum_{k=1}^{m} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \left\langle (\hat{w}_k - w_k) \otimes \sum_{i=1}^{r} \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)}\right), \hat{w}_\ell^{\otimes r} \right\rangle\right]^2 \\
&= \sum_{\ell=1}^{m} \left[\mu_r(g_{\hat{\tau}_\ell}^{(1)}) \left\langle \sum_{k=1}^{m} \mu_r(g_{\tau_k}) (\hat{w}_k - w_k) \otimes \sum_{i=1}^{r} \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)}\right), \hat{w}_\ell^{\otimes r} \right\rangle\right]^2 \\
&= \sum_{\ell=1}^{m} \mu_r(g_{\hat{\tau}_\ell}^{(1)})^2 \left\langle \sum_{k=1}^{m} \mu_r(g_{\tau_k}) (\hat{w}_k - w_k) \otimes \sum_{i=1}^{r} \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)}\right), \hat{w}_\ell^{\otimes r} \right\rangle^2.
\end{aligned}$$

At this stage, we separate the coefficients depending on ℓ such that

$$\begin{aligned}
&\sum_{\ell=1}^{m} \left[\sum_{k=1}^{m} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \left(\langle \hat{w}_k, \hat{w}_\ell \rangle^r - \langle w_k, \hat{w}_\ell \rangle^r\right)\right]^2 \\
&\leq \max_{\ell \in [m]} \mu_r(g_{\hat{\tau}_\ell}^{(1)})^2 \sum_{\ell=1}^{m} \left\langle \sum_{k=1}^{m} \mu_r(g_{\tau_k}) (\hat{w}_k - w_k) \otimes \sum_{i=1}^{r} \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)}\right), \hat{w}_\ell^{\otimes r} \right\rangle^2.
\end{aligned}$$

Now, note that the set of tensors $(\hat{w}_\ell^{\otimes r})_{\ell \in [m]}$ forms a frame whose upper frame constant is bounded by the upper spectrum of the Gramian $(\widehat{G}_r)_{ij} = \langle \hat{w}_i, \hat{w}_j \rangle^r$, see also Lemma 2.20. Due to Lemma 2.23, which relies on Gershgorin's circle theorem, we know there exists an absolute constant $C > 0$ such that for D sufficiently large the operator norm of $\widehat{G}_r$ obeys

$$\|\widehat{G}_r\| \leq C \left(1 + m \left(\frac{\log m}{D}\right)^{r/2}\right).$$

As a consequence, we get

$$\begin{aligned}
&\max_{\ell\in[m]} \mu_r(g^{(1)}_{\hat{\tau}_\ell})^2 \sum_{\ell=1}^m \left\langle \sum_{k=1}^m \mu_r(g_{\tau_k})(\hat{w}_k - w_k) \otimes \sum_{i=1}^r \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)} \right), \hat{w}_\ell^{\otimes r} \right\rangle^2 \\
&\leq \max_{\ell\in[m]} \mu_r(g^{(1)}_{\hat{\tau}_\ell})^2 \|\widehat{G}_r\| \left\| \sum_{k=1}^m \mu_r(g_{\tau_k})(\hat{w}_k - w_k) \otimes \sum_{i=1}^r \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)} \right) \right\|_F^2 \\
&\leq C \max_{\ell\in[m]} \mu_r(g^{(1)}_{\hat{\tau}_\ell})^2 \left(1 + m \left(\frac{\log m}{D} \right)^{r/2} \right) \left\| \sum_{k=1}^m \mu_r(g_{\tau_k})(\hat{w}_k - w_k) \otimes \sum_{i=1}^r \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)} \right) \right\|_F^2 .
\end{aligned} \tag{3.105}$$

Denote now $\Delta_{k,r} := \mu_r(g_{\tau_k})(\hat{w}_k - w_k)$ and $T_{k,r} := \sum_{i=1}^r \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)} \right)$, then

$$\begin{aligned}
&\left\| \sum_{k=1}^m \mu_r(g_{\tau_k})(\hat{w}_k - w_k) \otimes \sum_{i=1}^r \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)} \right) \right\|_F^2 \\
&= \sum_{k,k'=1}^m \langle \Delta_{k,r} \otimes T_{k,r}, \Delta_{k',r} \otimes T_{k',r} \rangle = \sum_{k,k'=1}^m \langle \Delta_{k,r}, \Delta_{k',r} \rangle \langle T_{k,r}, T_{k',r} \rangle \\
&= \sum_{k=1}^m \|\Delta_{k,r}\|_2^2 \|T_{k,r}\|_F^2 + \sum_{k\neq k'}^m \langle \Delta_{k,r}, \Delta_{k',r} \rangle \langle T_{k,r}, T_{k',r} \rangle .
\end{aligned} \tag{3.106}$$

Using $\|w_k\|_2 = \|\hat{w}_k\|_2 = 1$ we get

$$\|T_{k,r}\|_F \leq \sum_{i=1}^r \|\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)}\|_F \leq \sum_{i=1}^r \|\hat{w}_k\|_2^{r-i} \|w_k\|_2^{i-1} = r,$$

such that the left part of (3.106) can be estimated by

$$\begin{aligned}
\sum_{k=1}^m \|\Delta_{k,r}\|_2^2 \|T_{k,r}\|_F^2 &\leq r^2 \max_{k\in[m]} \mu_r(g_{\tau_k})^2 \sum_{k=1}^m \|\hat{w}_k - w_k\|_2^2 \\
&= r^2 \max_{k\in[m]} \mu_r(g_{\tau_k})^2 \|\widehat{W} - W\|_F^2 .
\end{aligned} \tag{3.107}$$

To bound the right part of (3.106) first note that, for $k \neq k'$,

$$\begin{aligned}
\langle T_{k,r}, T_{k',r} \rangle &= \sum_{i,i'=1}^r \left\langle \hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)}, \hat{w}_{k'}^{\otimes(r-i')} \otimes w_{k'}^{\otimes(i'-1)} \right\rangle \\
&\leq C \sum_{i,i'=1}^r \left(\frac{\log m}{D} \right)^{(r-1)/2} = C r^2 \left(\frac{\log m}{D} \right)^{(r-1)/2},
\end{aligned}$$

for some absolute constant C, which follows from the pairwise incoherence **(Inc2)** as well as point (i) of Lemma 3.2. Therefore, the right part of (3.106) is bounded by

$$\begin{aligned}
\sum_{k\neq k'}^m \langle \Delta_{k,r}, \Delta_{k',r} \rangle \langle T_{k,r}, T_{k',r} \rangle &\leq C r^2 \left(\frac{\log m}{D} \right)^{(r-1)/2} \sum_{k\neq k'}^m |\langle \Delta_{k,r}, \Delta_{k',r} \rangle| \\
&\leq C r^2 \left(\frac{\log m}{D} \right)^{(r-1)/2} \max_{k\in[m]} \mu_r(g_{\tau_k})^2 \sum_{k\neq k'}^m \left| \langle \hat{w}_k - w_k, \hat{w}_{k'} - w'_k \rangle \right| .
\end{aligned} \tag{3.108}$$

Plugging in (3.107) and (3.108) into (3.106) yields

$$\left\| \sum_{k=1}^{m} \mu_r(g_{\tau_k})(\hat{w}_k - w_k) \otimes \sum_{i=1}^{r} \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)} \right) \right\|_F^2 \tag{3.109}$$

$$\leq Cr^2 \max_{k\in[m]} \mu_r(g_{\tau_k})^2 \left[\|\widehat{W} - W\|_F^2 + \left(\frac{\log m}{D} \right)^{(r-1)/2} \sum_{k\neq k'}^{m} \left| \langle \hat{w}_k - w_k, \hat{w}_{k'} - w'_k \rangle \right| \right]. \tag{3.110}$$

Combining this with (3.105) yields the desired first statement

$$\sum_{\ell=1}^{m} S_{r,\ell}^2 \leq Cr^2 \max_{k,\ell\in[m]} \mu_r(g_{\hat{\tau}_\ell}^{(1)})^2 \mu_r(g_{\tau_k})^2 \left(1 + m \left(\frac{\log m}{D} \right)^{r/2} \right) \cdot \left[\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{(r-1)/2} \Delta_{W,O} \right].$$

For the second statement, note that $\max_{k\in[m]} \mu_r(g_{\tau_k})^2$ is bounded due to **(SNM2)** and that $\max_{\ell\in[m]} \mu_r(g_{\hat{\tau}_\ell}^{(1)})^2$ is bounded according to Lemma 3.5. Hence, it follows that

$$\begin{aligned} \sum_{r=2}^{R} 2^r \sum_{\ell=1}^{m} S_{r,\ell}^2 &\leq \sum_{r=2}^{R} 2^r Cr^2 \left(1 + m \left(\frac{\log m}{D} \right)^{r/2} \right) \left[\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{(r-1)/2} \Delta_{W,O} \right] \\ &\leq \left(1 + m \left(\frac{\log m}{D} \right) \right) \left(\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{1/2} \Delta_{W,O} \right) \sum_{r=1}^{R} 2^r Cr^2. \end{aligned}$$

The second statement follows from the upper bound above by adjusting the constant C due to $\sum_{r=1}^{R} 2^r Cr^2 < \infty$ for fixed R and using $(m \log m)/D > 1$. □

The next statement builds upon Lemma 3.11, in particular (3.105), to control the first sum in (3.103). Notably, we use **(SNM1)** to eliminate all Hermite coefficients in our expressions.

Lemma 3.12 (cf. [51, Lemma 10])**.** *Consider weights and approximated weights $(w_k)_{k\in[m]}, (\hat{w}_k)_{k\in[m]}$ of unit norm as before that both fulfill **(Inc2)** and (i) in Lemma 3.2, as well shifts $(\tau_k)_{k\in[m]}, (\hat{\tau}_k)_{k\in[m]}$ within $[-\tau_\infty, \tau_\infty]$ for some $\tau_\infty < \infty$. Let $S_{r,\ell}$ be defined as in* (3.102) *and assume that g fulfills the assumption **(SNM1)** - **(SNM2)**. Then, there exists a constant $C > 0$ such that for $m \geq D$*

$$\sum_{\ell=1}^{m} S_{1,\ell}^2 \leq Cm \left(\frac{\log m}{D} \right)^{1/2} \left\| \sum_{k=1}^{m} w_k - \hat{w}_k \right\|_2^2.$$

Proof of Lemma 3.12. According to the proof of Lemma 3.11, in particular (3.105), we can bound

$$\sum_{\ell=1}^{m} S_{1,\ell}^2 \leq Cm \left(\frac{\log m}{D} \right)^{1/2} \left\| \sum_{k=1}^{m} \mu_1(g_{\tau_k})(w_k - \hat{w}_k) \right\|_2^2,$$

for some constant $C > 0$. Since $\mu_1(g_{\tau_k})$ is bounded for all $k \in [m]$, what remains to be shown is that the Hermite coefficients do not change signs. Note that the first Hermite polynomial is given by $h_1(u) = u$. According to the definition of the Hermite coefficients, we have

$$\begin{aligned} \mu_1(g_{\tau_k}) &= \int_{\mathbb{R}} u g(u + \tau_k) e^{-u^2/2} du = \left[-g(u + \tau_k) e^{-u^2/2} \right]_{-\infty}^{\infty} + \int_{\mathbb{R}} g^{(1)}(u + \tau_k) e^{-u^2/2} du \\ &= \int_{\mathbb{R}} g^{(1)}(u + \tau_k) e^{-u^2/2} du. \end{aligned}$$

Now note that $g^{(1)}(u + \tau_k)$ will always have the same sign since $g^{(2)}$ is monotonic due to **(SNM1)**. Therefore, $\mu_1(g_{\tau_1}), \ldots, \mu_1(g_{\tau_m})$ must all be either positive or negative, from which the proof follows directly. □

The last auxiliary statement provides an upper bound for the tail series appearing in (3.103) for $R \geq 9$. The proof directly bounds the Cauchy product $\sum_{\ell=1}^{m} \left(\sum_{r\geq R} S_{r,\ell}\right)^2$, and leverages that the Gramian matrix $\widehat{G}_R$ of the system $\{\hat{w}_k^{\otimes R} | k \in [m]\}$ has a spectral norm bounded by a constant (independent of m, D), which follows from Lemma 2.23 in Section 2.6.1. Additionally, mixed terms giving rise to the factor $\Delta_{W,O}$, which appeared in the previous two statements, can be simplified using

$$\left(\frac{\log m}{D}\right)^{(R-1)/2} \Delta_{W,O} \leq \left(\frac{\log m}{D}\right)^4 m^2 \delta_{\max}^2 \leq \delta_{\max}^2 \leq \|W - \widehat{W}\|_F^2,$$

for $m \log^2 m \leq D^2, R \geq 9$. By these arguments, we can reduce the problem to

$$\sum_{\ell=1}^{m} \left(\sum_{r\geq R} S_{r,\ell}\right)^2 \leq C\|W - \widehat{W}\|_F^2 \sqrt{m} \left(\sum_{r\geq R} r |\max_{k,\ell} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)})|\right)^2.$$

The statement then follows by applying Lemma 3.7 to show $\sum_{r\geq R} r |\max_{k,\ell} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)})| < \infty$.

Lemma 3.13 (cf. [51, Lemma 12])**.** *Consider weights and approximated weights $(w_k)_{k\in[m]}, (\hat{w}_k)_{k\in[m]}$ of unit norm as before that both fulfill **(Inc2)** and (i) in Lemma 3.2, as well as shifts $(\tau_k)_{k\in[m]}, (\hat{\tau}_k)_{k\in[m]}$ within $[-\tau_\infty, \tau_\infty]$ for some $\tau_\infty < \infty$. Let $S_{r,\ell}$ be defined as in* (3.102)*, and assume that g fulfills the assumption **(SNM1)**. Then, there exists a constant $C > 0$ such that for $R \geq 9$ we have*

$$\sum_{\ell=1}^{m} \left(\sum_{r\geq R} S_{r,\ell}\right)^2 \leq C\sqrt{m}\|W - \widehat{W}\|_F^2.$$

Proof of Lemma 3.13. We start by applying the Cauchy product to the squared series

$$\begin{aligned}
\sum_{\ell=1}^{m} \left(\sum_{r\geq R} S_{r,\ell}\right)^2 &= \sum_{\ell=1}^{m} \left(\sum_{r\geq 0} \sum_{s=0}^{r} S_{r+R-s,\ell} S_{s+R,\ell}\right) \\
&= \sum_{r\geq 0} \sum_{s=0}^{r} \sum_{\ell=1}^{m} S_{r+R-s,\ell} S_{s+R,\ell} \\
&\leq \sqrt{m} \sum_{r\geq 0} \sum_{s=0}^{r} \left(\sum_{\ell=1}^{m} S_{r+R-s,\ell}^2 S_{s+R,\ell}^2\right)^{1/2}.
\end{aligned}$$

The inner sum is now controlled by a sequence of inequalities similar to Lemma 3.11. Again we denote $\Delta_{k,r} := \mu_r(g_{\tau_k})(\hat{w}_k - w_k)$ and $T_{k,r} := \sum_{i=1}^{r} \left(\hat{w}_k^{\otimes(r-i)} \otimes w_k^{\otimes(i-1)}\right)$, then by applying the same chain of inequality as at the beginning of the proof of Lemma 3.11 we receive

$$\begin{aligned}
\sum_{\ell=1}^{m} S_{r+R-s,\ell}^2 S_{s+R,\ell}^2 &= \sum_{\ell=1}^{m} \mu_{r+R-s}(g_{\hat{\tau}_\ell}^{(1)})^2 \mu_{s+R}(g_{\hat{\tau}_\ell}^{(1)})^2 \\
&\quad \cdot \left\langle \sum_{k=1}^{m} \Delta_{k,r+R-s} \otimes T_{k,r+R-s}, \hat{w}_\ell^{\otimes r+R-s} \right\rangle^2 \left\langle \sum_{k=1}^{m} \Delta_{k,s+R} \otimes T_{k,s+R}, \hat{w}_\ell^{\otimes s+R} \right\rangle^2 \\
&= \sum_{\ell=1}^{m} \mu_{r+R-s}(g_{\hat{\tau}_\ell}^{(1)})^2 \mu_{s+R}(g_{\hat{\tau}_\ell}^{(1)})^2 \\
&\quad \cdot \left\langle \left(\sum_{k=1}^{m} \Delta_{k,r+R-s} \otimes T_{k,r+R-s}\right) \otimes \left(\sum_{k=1}^{m} \Delta_{k,s+R} \otimes T_{k,s+R}\right), \hat{w}_\ell^{\otimes r+2R} \right\rangle^2.
\end{aligned}$$

As before, we now invoke the frame-like condition described in Lemma 2.20 to attain a bound depending on the upper spectrum of the Gramian $(\widehat{G}_{r+2R})_{ij} = \langle \hat{w}_i, \hat{w}_j \rangle^{r+2R}$. More precisely, by using the shorthand

$$\mu'_{r,s} := \max_{\ell \in [m]} \mu_{r+R-s}(g^{(1)}_{\hat{\tau}_\ell})^2 \mu_{s+R}(g^{(1)}_{\hat{\tau}_\ell})^2 \tag{3.111}$$

we then have

$$\sum_{\ell=1}^{m} S^2_{r+R-s,\ell} S^2_{s+R,\ell} \leq \mu'_{r,s} \|\widehat{G}_{r+2R}\| \left\| \left(\sum_{k=1}^{m} \Delta_{k,r+R-s} \otimes T_{k,r+R-s} \right) \otimes \left(\sum_{k=1}^{m} \Delta_{k,s+R} \otimes T_{k,s+R} \right) \right\|_F^2 \tag{3.112}$$

$$\leq \mu'_{r,s} \|\widehat{G}_{r+2R}\| \left\| \sum_{k=1}^{m} \Delta_{k,r+R-s} \otimes T_{k,r+R-s} \right\|_F^2 \left\| \sum_{k=1}^{m} \Delta_{k,s+R} \otimes T_{k,s+R} \right\|_F^2. \tag{3.113}$$

The two Frobenius norms can now be estimated as in Lemma 3.11, more precisely (3.110), where we also use the shorthand $\Delta_{W,O}$ defined in (3.104) as well as

$$\mu_{r,s} := \max_{k \in [m]} \mu_{r+R-s}(g_{\tau_k})^2 \max_{k \in [m]} \mu_{s+R}(g_{\tau_k})^2.$$

This gives for some absolute constant $C > 0$

$$\begin{aligned}
&\mu'_{r,s} \|\widehat{G}_{r+2R}\| \left\| \sum_{k=1}^{m} \Delta_{k,r+R-s} \otimes T_{k,r+R-s} \right\|_F^2 \left\| \sum_{k=1}^{m} \Delta_{k,s+R} \otimes T_{k,s+R} \right\|_F^2 \\
&\quad \leq C \mu'_{r,s} \mu_{r,s} \|\widehat{G}_{r+2R}\| (r+R-s)^2 (s+R)^2 \\
&\quad \cdot \left(\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{(r+R-s-1)/2} \Delta_{W,O} \right) \left(\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{(s+R-1)/2} \Delta_{W,O} \right) \\
&\quad \leq C \mu'_{r,s} \mu_{r,s} \|\widehat{G}_{r+2R}\| (r+R-s)^2 (s+R)^2 \left(\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{(R-1)/2} \Delta_{W,O} \right)^2.
\end{aligned}$$

Next, we identify the dominant factors and simplify the last expression accordingly. Due to Lemma 2.23 we have for some constant $C > 0$ that

$$\|\widehat{G}_{r+2R}\| \leq C \left(1 + m \left(\frac{\log m}{D} \right)^{(r+2R)/2} \right) \leq C \left(1 + m \left(\frac{\log m}{D} \right)^{9} \right),$$

where the last stop follows since $R \geq 9$ and due to $m(\log m)^2 \leq D^2$ this can be further simplified to $\|\widehat{G}_{r+2R}\| \leq C$. Similarly, we have

$$\begin{aligned}
\left(\frac{\log m}{D} \right)^{(R-1)/2} \Delta_{W,O} &= \left(\frac{\log m}{D} \right)^{(R-1)/2} \sum_{k \neq k'}^{m} \left| \langle \hat{w}_k - w_k, \hat{w}_{k'} - w'_k \rangle \right| \\
&\leq \left(\frac{\log m}{D} \right)^{4} m^2 \delta^2_{\max} \leq \delta^2_{\max} \leq \|W - \widehat{W}\|_F^2,
\end{aligned}$$

and therefore, we get

$$\|\widehat{G}_{r+2R}\| \left(\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{(R-1)/2} \Delta_{W,O} \right)^2 \leq C \|W - \widehat{W}\|_F^4$$

for some absolute constant $C > 0$. Plugging these into (3.113) results in

$$\sum_{\ell=1}^{m} S_{r+R-s,\ell}^2 S_{s+R,\ell}^2 \leq C\mu'_{r,s}\mu_{r,s}(r+R-s)^2(s+R)^2\|W-\widehat{W}\|_F^4.$$

Hence, we have

$$\begin{aligned}\sum_{\ell=1}^{m}\left(\sum_{r\geq R} S_{r,\ell}\right)^2 &\leq \sqrt{m}\sum_{r\geq 0}\sum_{s=0}^{r}\left(\sum_{\ell=1}^{m} S_{r+R-s,\ell}^2 S_{s+R,\ell}^2\right)^{1/2} \\ &\leq C\|W-\widehat{W}\|_F^2\sqrt{m}\sum_{r\geq 0}\sum_{s=0}^{r}\sqrt{|\mu'_{r,s}\mu_{r,s}|}(r+R-s)(s+R) \\ &\leq C\|W-\widehat{W}\|_F^2\sqrt{m}\left(\sum_{r\geq R} r|\max_{k,\ell}\mu_r(g_{\tau_k})\mu_r(g^{(1)}_{\hat{\tau}_\ell})|\right)^2.\end{aligned}$$

The result then follows by applying Lemma 3.7 onto the series in the last line followed by a unification of the constants. □

With Lemma 3.11-3.13, we can now prove the main lemma of this section. The proof has already been outlined at the beginning of the section.

Proof of Lemma 3.10. Recall that

$$J(\hat{\tau}) = \frac{1}{2N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\left(\hat{f}(X_i,\hat{\tau}) - f(X_i,\tau)\right)^2.$$

By the chain rule, we compute the gradient of J w.r.t. $\hat{\tau}$ as

$$\nabla_{\hat{\tau}} J(\hat{\tau}) = \frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\left(\hat{f}(X_i,\hat{\tau}) - f(X_i,\tau)\right)\nabla_{\hat{\tau}}\hat{f}(X_i,\hat{\tau}).$$

Adding $0 = (\hat{f}(X_i,\tau) - \hat{f}(X_i,\tau))\nabla_{\hat{\tau}}\hat{f}(X_i,\hat{\tau})$ to $J(\hat{\tau})$ and applying the triangle inequality to $\nabla_{\hat{\tau}}J - \nabla_{\hat{\tau}}J_*$ allows us to separate the error caused by the weight approximation

$$\begin{aligned}&\left\|\nabla_{\hat{\tau}}J(\hat{\tau}) - \nabla_{\hat{\tau}}J_*(\hat{\tau})\right\|_2 \\ &\quad\leq \left\|\frac{1}{N_{\text{train}}}\Big(\sum_{i=1}^{N_{\text{train}}}(\hat{f}(X_i,\hat{\tau}) - \hat{f}(X_i,\tau))\nabla_{\hat{\tau}}\hat{f}(X_i,\hat{\tau}) - \nabla_{\hat{\tau}}\hat{f}(X_i,\tau)\nabla_{\hat{\tau}}\hat{f}(X_i,\tau)^\top(\hat{\tau}-\tau)\Big)\right\|_2 \qquad (3.114) \\ &\quad\quad + \left\|\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\left(\hat{f}(X_i,\tau) - f(X_i,\tau)\right)\nabla_{\hat{\tau}}\hat{f}(X_i,\hat{\tau})\right\|_2. \qquad (3.115)\end{aligned}$$

To bound the first term in (3.114) denote $h(\lambda) = (1-\lambda)\hat{\tau} + \lambda\tau$, then we have

$$\begin{aligned}
\hat{f}(X_i, \hat{\tau}) - \hat{f}(X_i, \tau) &= \sum_{k=1}^{m} g(\hat{w}_k^\top X_i + \tau_k) - g(\hat{w}_k^\top X_i + \hat{\tau}_k) \\
&= \sum_{k=1}^{m} g(\hat{w}_k^\top X_i + h(1)_k) - g(\hat{w}_k^\top X_i + h(0)_k) \\
&= \sum_{k=1}^{m} \int_{h(0)_k}^{h(1)_k} g^{(1)}(\hat{w}_k^\top X_i + u) du \\
&= \sum_{k=1}^{m} \int_0^1 g^{(1)}(\hat{w}_k^\top X_i + h(\lambda)_k) h'(\lambda)_k d\lambda \\
&= \sum_{k=1}^{m} \int_0^1 g^{(1)}(\hat{w}_k^\top X_i + h(\lambda)_k) d\lambda (\tau_k - \hat{\tau}_k).
\end{aligned}$$

Therefore, we can bound (3.114) as follows:

$$\begin{aligned}
&\Big\| \frac{1}{N_{\text{train}}} \Big(\sum_{i=1}^{N_{\text{train}}} (\hat{f}(X_i, \hat{\tau}) - \hat{f}(X_i, \tau)) \nabla_{\hat{\tau}} \hat{f}(X_i, \hat{\tau}) - \nabla_{\hat{\tau}} \hat{f}(X_i, \tau) \nabla_{\hat{\tau}} \hat{f}(X_i, \tau)^\top (\hat{\tau} - \tau) \Big) \Big\|_2 \\
\leq &\Big\| \frac{1}{N_{\text{train}}} \Big(\sum_{i=1}^{N_{\text{train}}} \nabla_{\hat{\tau}} \hat{f}(X_i, \hat{\tau}) \Big(\int_0^1 \nabla_{\hat{\tau}} \hat{f}(X_i, h(\lambda)) d\lambda \Big)^\top - \nabla_{\hat{\tau}} \hat{f}(X_i, \tau) \nabla_{\hat{\tau}} \hat{f}(X_i, \tau)^\top \Big) \Big\| \, \|(\hat{\tau} - \tau)\|_2 .
\end{aligned}$$

Let us fix $\hat{\tau}, \tau$ for now and write the last line in terms of matrices $\hat{F}, F, F^* \in \mathbb{R}^{N_{\text{train}} \times m}$, where the i-th row of these matrices is given by $\nabla_{\hat{\tau}} \hat{f}(X_i, \hat{\tau}), \nabla_{\hat{\tau}} \hat{f}(X_i, \tau)$ and $\int_0^1 \nabla_{\hat{\tau}} \hat{f}(X_i, h(\lambda)) d\lambda$, respectively. We obtain

$$\begin{aligned}
&\Big\| \frac{1}{N_{\text{train}}} \Big(\sum_{i=1}^{N_{\text{train}}} \nabla_{\hat{\tau}} \hat{f}(X_i, \hat{\tau}) \Big(\int_0^1 \nabla_{\hat{\tau}} \hat{f}(X_i, h(\lambda)) d\lambda \Big)^\top - \nabla_{\hat{\tau}} \hat{f}(X_i, \tau) \nabla_{\hat{\tau}} \hat{f}(X_i, \tau)^\top \Big) \Big\| \, \|(\hat{\tau} - \tau)\|_2 \\
\leq &\frac{1}{N_{\text{train}}} \|\hat{F}^\top F^* - F^\top F\| \, \|\hat{\tau} - \tau\|_2 \\
\leq &\frac{1}{N_{\text{train}}} \Big(\|\hat{F} - F\| \|F^*\| + \|F\| \|F - F^*\| \Big) \, \|\hat{\tau} - \tau\|_2 . \qquad (3.116)
\end{aligned}$$

A simultaneous upper bound for $\|\hat{F} - F\|$ and $\|F - F^*\|$ can be established with elementary matrix arithmetic and the Lipschitz continuity of $g^{(1)}$:

$$\begin{aligned}
\|\hat{F} - F\| \leq \|\hat{F} - F\|_F &= \Big[\sum_{i=1}^{N_{\text{train}}} \sum_{k=1}^{m} (g^{(1)}(\langle \hat{w}_k, X_i \rangle + \hat{\tau}_k) - g^{(1)}(\langle \hat{w}_k, X_i \rangle + \tau_k))^2 \Big]^{\frac{1}{2}} \\
&\leq \Big\| g^{(2)} \Big\|_\infty \Big[\sum_{i=1}^{N_{\text{train}}} \sum_{k=1}^{m} \Big(\hat{\tau}_k - \tau_k \Big)^2 \Big]^{\frac{1}{2}} = \kappa \sqrt{N_{\text{train}}} \, \|\hat{\tau} - \tau\|_2 ,
\end{aligned}$$

the same bound follows for $\|F - F^*\|$. A crude bound for $\|F\|$ is given by

$$\|F\| \leq \sqrt{N_{\text{train}} m} \max_{ik} |F_{ik}| \leq \sqrt{N_{\text{train}} m} \Big\| g^{(1)} \Big\|_\infty \leq \kappa \sqrt{N_{\text{train}} m},$$

the same bound follows for $\|F^*\|$. Hence, we can continue from (3.116) with

$$\frac{1}{N_{\text{train}}} \Big(\|\hat{F} - F\| \|F^*\| + \|F\| \|F - F^*\| \Big) \, \|\hat{\tau} - \tau\|_2 \leq 2\kappa^2 \sqrt{m} \|\hat{\tau} - \tau\|_2^2.$$

The error (3.115) caused by the difference between $\widehat{W}$ and the original weights W has the form

$$\begin{aligned}&\Big\|\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\Big(\hat{f}(X_i,\tau)-f(X_i,\tau)\Big)\nabla_{\hat{\tau}}\hat{f}(X_i,\hat{\tau})\Big\|_2\\ =&\Big\|\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\Big(\sum_{k=1}^{m}g(X_i^\top\hat{w}_k+\tau_k)-g(X_i^\top w_k+\tau_k)\Big)\nabla_{\hat{\tau}}\hat{f}(X_i,\hat{\tau})\Big\|_2.\end{aligned}$$

Let us define

$$\begin{aligned}\Delta_W^2&:=\Big\|\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\Big(\sum_{k=1}^{m}g(X_i^\top\hat{w}_k+\tau_k)-g(X_i^\top w_k+\tau_k)\Big)\nabla_{\hat{\tau}}\hat{f}(X_i,\hat{\tau})\Big\|_2^2\\ &=\sum_{\ell=1}^{m}\left[\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\Big(\sum_{k=1}^{m}g(X_i^\top\hat{w}_k+\tau_k)-g(X_i^\top w_k+\tau_k)\Big)g^{(1)}(\langle X_i,\hat{w}_\ell\rangle+\hat{\tau}_\ell)\right]^2\end{aligned}$$

To keep the expressions more compact, we define $Z_{ik\ell}:=\varphi_{k,\ell}(X_i)$ and

$$\varphi_{k,\ell}(x):=\Big(g(x^\top\hat{w}_k+\tau_k)-g(x^\top w_k+\tau_k)\Big)g^{(1)}(x^\top\hat{w}_\ell+\hat{\tau}_\ell).$$

Let us also define

$$\begin{aligned}\Delta_{W,1}^2&:=\sum_{\ell=1}^{m}\left[\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\sum_{k=1}^{m}\mathbb{E}[Z_{ik\ell}]\right]^2,\\ \Delta_{W,2}^2&:=\sum_{\ell=1}^{m}\left[\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\sum_{k=1}^{m}(Z_{ik\ell}-\mathbb{E}[Z_{ik\ell}])\right]^2.\end{aligned}$$

Then,

$$\Delta_W^2=\sum_{\ell=1}^{m}\left[\frac{1}{N_{\text{train}}}\sum_{i=1}^{N_{\text{train}}}\sum_{k=1}^{m}Z_{ik\ell}\right]^2\leq 2\Delta_{W,1}^2+2\Delta_{W,2}^2. \tag{3.117}$$

In what follows, we will control $\Delta_{W,1}^2,\Delta_{W,2}^2$ by using Hermite expansions and a concentration argument, respectively.

We begin with $\Delta_{W,2}^2$: The first step is to establish that $Z_{ik\ell}-\mathbb{E}[Z_{ik\ell}]$ is sub-Gaussian and to compute its sub-Gaussian norm. We remark that all expectations for the remainder of this proof are w.r.t. the inputs $X_1,\dots,X_{N_{\text{train}}}\sim\mathcal{N}(0,\mathrm{Id}_D)$. First note that by the mean value theorem, there exist values $\xi_{i,k}$ such that

$$Z_{ik\ell}=\langle\hat{w}_k-w_k,X_i\rangle g^{(1)}(x^\top\hat{w}_\ell+\hat{\tau}_\ell)g^{(1)}(\xi_{i,k}),$$

where $g^{(1)}$ is a bounded function according to **(SNM1)**. We can combine this with the well-known property of the sub-Gaussian norm, which states that $\|Z_{ik\ell}-\mathbb{E}[Z_{ik\ell}]\|_{\psi_2}\leq C\|Z_{ik\ell}\|_{\psi_2}$ for some absolute constant $C>0$. This leads to

$$\|Z_{ik\ell}-\mathbb{E}[Z_{ik\ell}]\|_{\psi_2}\leq C\|Z_{ik\ell}\|_{\psi_2}\leq C\kappa^2\|\langle\hat{w}_k-w_k,X_i\rangle\|_{\psi_2}\leq C\kappa^2\delta_{\max}$$

for all $i \in [N_{\text{train}}]$, $k, \ell \in [m]$ and some absolute constant $C > 0$. As a consequence, we can apply the general Hoeffding inequality (cf. [127, Theorem 2.6.2]), which yields the estimate

$$\begin{aligned}\Delta_{W,2}^2 &= \frac{1}{N_{\text{train}}^2} \sum_{\ell=1}^{m} \left(\sum_{k=1}^{m} \sum_{i=1}^{N_{\text{train}}} Z_{ik\ell} - \mathbb{E}[Z_{ik\ell}] \right)^2 \\ &\leq \frac{1}{N_{\text{train}}^2} \sum_{\ell=1}^{m} \left(\sum_{k=1}^{m} \left| \sum_{i=1}^{N_{\text{train}}} Z_{ik\ell} - \mathbb{E}[Z_{ik\ell}] \right| \right)^2 \leq \frac{1}{N_{\text{train}}^2} \sum_{\ell=1}^{m} m^2 t^2 = \frac{m^3 t^2}{N_{\text{train}}^2},\end{aligned}$$

which holds using a union bound with probability at least

$$1 - \left(\sum_{k,\ell=1}^{m} 2 \exp\left(- \frac{ct^2}{\sum_{i=1}^{N_{\text{train}}} \| Z_{ik\ell} - \mathbb{E}[Z_{ik\ell}] \|_{\psi_2}^2} \right) \right) \geq 1 - 2m^2 \exp\left(- \frac{t^2}{C N_{\text{train}} \delta_{\max}^2 \kappa^4} \right),$$

for all $t \geq 0$, where $c, C > 0$ are absolute constants. This implies that there exists an absolute constant $C > 0$ such that for all $t \geq 0$

$$\mathbb{P}\left(\Delta_{W,2}^2 \leq \frac{m^3 \delta_{\max}^2 t}{N_{\text{train}}} \right) \geq 1 - 2m^2 \exp\left(- \frac{t}{C\kappa^4} \right). \tag{3.118}$$

What remains is to control the means contained in $\Delta_{W,1}^2$. Using the shorthand $g_\tau(\cdot) = g(\cdot + \tau)$ and the Hermite expansion we get

$$\begin{aligned}\mathbb{E}[Z_{ik\ell}] &= \mathbb{E}\left[(g_{\tau_k}(\hat{w}_k^\top X_i) - g_{\tau_k}(w_k^\top X_i)) g_{\hat{\tau}_\ell}^{(1)}(\hat{w}_\ell^\top X_i) \right] \\ &= \mathbb{E}\left[\left(\sum_{r \geq 0} \mu_r(g_{\tau_k}) (h_r(\hat{w}_k^\top X_i) - h_r(w_k^\top X_i)) \right) \sum_{t \geq 0} \mu_t(g_{\hat{\tau}_\ell}^{(1)}) h_t(\hat{w}_\ell^\top X_i) \right] \\ &= \sum_{r \geq 0} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \left(\langle \hat{w}_k, \hat{w}_\ell \rangle^r - \langle w_k, \hat{w}_\ell \rangle^r \right),\end{aligned}$$

where the last two steps rely on the same properties of the Hermite expansion already used in the previous section. The summand corresponding to $r = 0$ in the last line above vanishes, thus we have

$$\Delta_{W,1}^2 = \sum_{\ell=1}^{m} \left[\sum_{k=1}^{m} \sum_{r \geq 1} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \left(\langle \hat{w}_k, \hat{w}_\ell \rangle^r - \langle w_k, \hat{w}_\ell \rangle^r \right) \right]^2.$$

Denote now

$$S_{r,\ell} := \sum_{k=1}^{m} \mu_r(g_{\tau_k}) \mu_r(g_{\hat{\tau}_\ell}^{(1)}) \left(\langle \hat{w}_k, \hat{w}_\ell \rangle^r - \langle w_k, \hat{w}_\ell \rangle^r \right),$$

then, for any $R \geq 2$, we have

$$\Delta_{W,1}^2 = \sum_{\ell=1}^{m} \left(\sum_{r \geq 1} S_{r,\ell} \right)^2 \leq 2 \sum_{\ell=1}^{m} S_{1,\ell}^2 + \sum_{r=2}^{R-1} 2^r \sum_{\ell=1}^{m} S_{r,\ell}^2 + 2^R \sum_{\ell=1}^{m} \left(\sum_{r \geq R} S_{r,\ell} \right)^2. \tag{3.119}$$

Choose now $R = 9$ and plug in the result from Lemma 3.11, Lemma 3.12 and Lemma 3.13, which yields for an appropriate constant $C > 0$ the bound

$$\Delta_{W,1}^2 \leq Cm \left(\frac{\log m}{D} \right)^{1/2} \left\| \sum_{k=1}^{m} w_k - \hat{w}_k \right\|_2^2 + \frac{Cm \log m}{D} \left(\|W - \widehat{W}\|_F^2 + \left(\frac{\log m}{D} \right)^{1/2} \Delta_{W,O} \right) \tag{3.120}$$

$$+ C\sqrt{m} \|W - \widehat{W}\|_F^2. \tag{3.121}$$

Reordering the terms and taking the square root we receive

$$\begin{aligned}\Delta_{W,1} \leq C\left(m^{1/4} + m^{1/2}\left(\frac{\log m}{D}\right)^{1/2}\right)\|W - \widehat{W}\|_F + Cm^{1/2}\left(\frac{\log m}{D}\right)^{3/4}\Delta_{W,O}^{1/2} \\ + Cm^{1/2}\left(\frac{\log m}{D}\right)^{1/4}\left\|\sum_{k=1}^{m} w_k - \hat{w}_k\right\|_2 \\ \leq C\log(m)^{3/4}\left[\left(m^{1/4} + \frac{m^{1/2}}{D^{1/2}}\right)\|W - \widehat{W}\|_F + \frac{m^{1/2}}{D^{3/4}}\Delta_{W,O}^{1/2} + \frac{m^{1/2}}{D^{1/4}}\left\|\sum_{k=1}^{m} w_k - \hat{w}_k\right\|_2\right].\end{aligned}$$

Lastly, we can use

$$m^{1/4} + \frac{m^{1/2}}{D^{1/2}} \leq m^{1/4} + \frac{m^{1/2}}{D^{1/4}} \leq \frac{2m^{1/2}}{D^{1/4}}$$

since $m \geq D$ followed by $\Delta_W \leq C(\Delta_{W,1} + \Delta_{W,2})$ to conclude the proof. Note that we can simply separate the constant that appears in the definition of $\Delta_{W,1}$ to appear outside of the term $\Delta_{W,1}$, such that we arrive at the formulation appearing in the original statement. □

3.4.7 A lower bound for the drift from the idealized GD iteration

The last statement needed for the proof of Theorem 3.4 is concerned with the drift between the idealized gradient descent iteration $(\hat{\tau}_*^{(n)})_{n\in\mathbb{N}}$ and the gradient decent iteration $(\hat{\tau}^{(n)})_{n\in\mathbb{N}}$. This will be done under the assumption that $(\hat{\tau}_*^{(n)})_{n\in\mathbb{N}}$ converges to the true shifts τ at a linear rate, which is guaranteed whenever the objective $J_*(\hat{\tau}) = (\hat{\tau} - \tau)^\top A(\hat{\tau} - \tau)$ is strictly convex. In Section 3.4.5, more precisely Lemma 3.9, we saw that this assumption holds with high probability as long as the number of training samples N_{train} is sufficiently large. By applying the perturbation result from the last section, which states that the gradient updates of $(\hat{\tau}^{(n)})_{n\in\mathbb{N}}, (\hat{\tau}_*^{(n)})_{n\in\mathbb{N}}$ satisfy

$$\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2 \leq \kappa^2\sqrt{m}\,\|\hat{\tau} - \tau\|_2^2 + \Delta_W,$$

in a vicinity around the true shifts τ, we will now control the drift $\|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2$. The next statement gives sufficient conditions on Δ_W and the starting point $\hat{\tau}^{(0)}$, under which the drift $\|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2$ remains uniformly bounded by $\frac{2\Delta_W}{\lambda_{\min}(A)}$. Consequently, since $(\hat{\tau}_*^{(n)})_{n\in\mathbb{N}}$ does converge to τ, the gradient iteration $(\hat{\tau}^{(n)})_{n\in\mathbb{N}}$ must remain within distance $\frac{2\Delta_W}{\lambda_{\min}(A)}$ of the ground truth shifts.

Lemma 3.14 ([51, Lemma 14]). *Denote by $\hat{\tau}^{(n)}, \hat{\tau}_*^{(n)}$ the gradient descent iterations given by* (3.72) *and* (3.85)*, respectively. Assume that the objective functions J, J_* defined above fulfill*

$$\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2 \leq L\,\|\hat{\tau} - \tau\|_2^2 + \Delta_W, \tag{3.122}$$

for some $L, \Delta_W \geq 0$ and any $\hat{\tau} \in \mathbb{R}^m$. Furthermore, assume that the matrix A in (3.84) *fulfills $\lambda_{\min} := \lambda_{\min}(A) > 0$. If $\Delta_W \leq \frac{\lambda_{\min}^2}{16L}$ and both gradient descent iterations are started with the same step size $\gamma \leq \|A\|^{-1}$ and from the same initialization $\hat{\tau}^{(0)} = \hat{\tau}_*^{(0)}$, adhering to the bound*

$$\|\hat{\tau}^{(0)} - \tau\|_2 \leq \frac{\lambda_{\min}}{4\sqrt{2}L}, \tag{3.123}$$

then the distance between both iterations at gradient step $n \in \mathbb{N}$ satisfies

$$\|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 \leq \xi^n \|\hat{\tau}^{(0)} - \tau\|_2 + \frac{2\Delta_W}{\lambda_{\min}} (1 - \xi^n),$$

for $\xi = 1 - \frac{\gamma \lambda_{\min}}{2} \in [0, 1)$.

Proof of Lemma 3.14. Plugging in the gradient descent iteration with a simple expansion yields

$$\begin{aligned}
&\left\| \hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)} \right\|_2 \\
&= \left\| \hat{\tau}^{(n)} - \hat{\tau}_*^{(n)} - \gamma \left(\nabla_{\hat{\tau}} J(\hat{\tau}^{(n)}) - \nabla_{\hat{\tau}} J_*(\hat{\tau}_*^{(n)}) \right) \right\|_2 \\
&= \left\| \hat{\tau}^{(n)} - \hat{\tau}_*^{(n)} - \gamma \left(\nabla_{\hat{\tau}} J(\hat{\tau}^{(n)}) - \nabla_{\hat{\tau}} J_*(\hat{\tau}^{(n)}) \right) - \gamma \left(\nabla_{\hat{\tau}} J_*(\hat{\tau}^{(n)}) - \nabla_{\hat{\tau}} J_*(\hat{\tau}_*^{(n)}) \right) \right\|_2 \\
&= \left\| \left(\mathrm{Id}_m - \gamma A \right) (\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}) - \gamma \left(\nabla_{\hat{\tau}} J(\hat{\tau}^{(n)}) - \nabla_{\hat{\tau}} J_*(\hat{\tau}^{(n)}) \right) \right\|_2 \\
&\leq \left\| \left(\mathrm{Id}_m - \gamma A \right) (\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}) \right\|_2 + \gamma \left\| \nabla_{\hat{\tau}} J(\hat{\tau}^{(n)}) - \nabla_{\hat{\tau}} J_*(\hat{\tau}^{(n)}) \right\|_2,
\end{aligned}$$

where we used the definition of the iterations in the first line followed by a simple expansion and the triangle inequality in the last line. The left term of the last line can be bounded with the spectral norm of $\mathrm{Id}_m - \gamma A$ and the right term according to our initial assumption (3.122):

$$\begin{aligned}
\|\hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)}\|_2 &\leq \|\mathrm{Id}_m - \gamma A\| \, \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 + \gamma L \|\hat{\tau}^{(n)} - \tau\|_2^2 + \gamma \Delta_W \\
&\leq (1 - \gamma \lambda_{\min}) \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 + \gamma L \|\hat{\tau}^{(n)} - \tau\|_2^2 + \gamma \Delta_W,
\end{aligned}$$

where the second inequality follows from the bound on the minimal eigenvalue of A. Expanding the right term of the last line with $\hat{\tau}_*^{(n)}$ yields

$$\begin{aligned}
&\|\hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)}\|_2 \\
&\quad \leq (1 - \gamma \lambda_{\min}) \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 + \gamma L \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)} + \hat{\tau}_*^{(n)} - \tau\|_2^2 + \gamma \Delta_W \\
&\quad \leq (1 - \gamma \lambda_{\min}) \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 + 2\gamma L \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2^2 + 2\gamma L \|\hat{\tau}_*^{(n)} - \tau\|_2^2 + \gamma \Delta_W. \qquad (3.124)
\end{aligned}$$

We can now use the fact that the gradient descent iteration (3.85) in combination with the convexity of the idealized objective J_* ($\lambda_{\min}(A) > 0$) allows for the recursive bound

$$\begin{aligned}
\|\hat{\tau}_*^{(n)} - \tau\|_2 &= \|\hat{\tau}_*^{(n-1)} - \gamma \nabla_{\hat{\tau}} J_*(\hat{\tau}_*^{(n-1)}) - \tau\|_2 = \|\hat{\tau}_*^{(n-1)} - \gamma A(\hat{\tau}_*^{(n-1)} - \tau) - \tau\|_2 \\
&= \|(\mathrm{Id}_m - \gamma A)(\hat{\tau}_*^{(n-1)} - \tau)\|_2 \leq \|\mathrm{Id}_m - \gamma A\| \|\hat{\tau}_*^{(n-1)} - \tau\|_2 \\
&\leq \|\mathrm{Id}_m - \gamma A\|^n \|\hat{\tau}_*^{(0)} - \tau\|_2 \leq (1 - \gamma \lambda_{\min})^n \delta_0,
\end{aligned}$$

where we have denoted by $\delta_0 = \|\hat{\tau}^{(0)} - \tau\|_2$ the initial error. Plugging this into (3.124) results in

$$\begin{aligned}
&\|\hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)}\|_2 \\
&\quad \leq (1 - \gamma \lambda_{\min}) \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 + 2\gamma L \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2^2 + 2\gamma L (1 - \gamma \lambda_{\min})^{2n} \delta_0^2 + \gamma \Delta_W. \qquad (3.125)
\end{aligned}$$

Define $\Delta_n := \max_{k \leq n} \|\hat{\tau}^{(k)} - \hat{\tau}_*^{(k)}\|_2$. We first show by induction that $\Delta_n \leq \lambda_{\min}/4L$ provided that δ_0 and Δ_W are sufficiently small. For step $n = 0$, we have $\|\hat{\tau}^{(0)} - \hat{\tau}_*^{(0)}\|_2 = 0$, so the statement is clearly true. Assume now it holds for n. We have to show the induction step. In

other words we have to show $\|\hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)}\|_2 \leq \lambda_{\min}/4L$, so the same bound would hold for Δ_{n+1}. We continue from (3.125), and get

$$\|\hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)}\|_2 \leq (1 - \gamma\lambda_{\min} + 2\gamma L\Delta_n)\|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 + 2\gamma L(1 - \gamma\lambda_{\min})^{2n}\delta_0^2 + \gamma\Delta_W.$$

Using the induction hypothesis $\Delta_n \leq \lambda_{\min}/4L$, this simplifies to

$$\|\hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)}\|_2 \leq (1 - \gamma\lambda_{\min}/2)\,\|(\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)})\|_2 + 2\gamma L(1 - \gamma\lambda_{\min})^{2n}\delta_0^2 + \gamma\Delta_W.$$

To keep the computation more compact, we will denote

$$\xi := 1 - \frac{\gamma\lambda_{\min}}{2}.$$

Now we can repeat the same computations for $\|\hat{\tau}^{(k)} - \hat{\tau}_*^{(k)}\|_2$, $k \leq n$ as well. This leads to

$$\|\hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)}\|_2 \leq 2\gamma L\delta_0^2 \sum_{k=0}^{n} \xi^k (1 - \gamma\lambda_{\min})^{2(n-k)} + \gamma\Delta_W \sum_{k=0}^{n} \xi^k,$$

where we used $\left\|\hat{\tau}^{(0)} - \hat{\tau}_*^{(0)}\right\|_2 = 0$. Both sums are uniformly bounded in n, as can be seen by

$$\begin{aligned}
\|\hat{\tau}^{(n+1)} - \hat{\tau}_*^{(n+1)}\|_2 &\leq 2\gamma L\delta_0^2 \frac{\xi^{n+1} - (1 - \gamma\lambda_{\min})^{2(n+1)}}{\xi - (1 - \gamma\lambda_{\min})^2} + \gamma\Delta_W \frac{1 - \xi^{n+1}}{1 - \xi} \\
&\leq 2\gamma L\delta_0^2 \frac{\xi^{n+1} - (1 - \gamma\lambda_{\min})^{2(n+1)}}{\frac{3}{2}\gamma\lambda_{\min} - \gamma^2\lambda_{\min}^2} + \frac{2\Delta_W}{\lambda_{\min}} \\
&\leq 2L\delta_0^2 \frac{\xi^{n+1}}{\frac{3}{2}\lambda_{\min} - \gamma\lambda_{\min}^2} + \frac{2\Delta_W}{\lambda_{\min}} \leq 4L\delta_0^2 \frac{\xi^{n+1}}{\lambda_{\min}} + \frac{2\Delta_W}{\lambda_{\min}}.
\end{aligned} \tag{3.126}$$

Now we have $4L\delta_0^2\xi^{n+1}\lambda_{\min}^{-1} \leq 4L\delta_0^2\lambda_{\min}^{-1}$. Furthermore, $4L\delta_0^2\lambda_{\min}^{-1} \leq \frac{\lambda_{\min}}{8L}$ as long as

$$\delta_0^2 \leq \frac{\lambda_{\min}^2}{32L^2},$$

which holds according to our initial assumption (3.123). Similarly, as $\Delta_W \leq \frac{\lambda_{\min}^2}{16L}$ by assumption, we get $\frac{2\Delta_W}{\lambda_{\min}} \leq \frac{\lambda_{\min}}{8L}$ This means we now have

$$\Delta_{n+1} \leq \frac{\lambda_{\min}}{8L} + \frac{\lambda_{\min}}{8L} \leq \frac{\lambda_{\min}}{4L},$$

which concludes the proof of the induction establishing that the two iterations remain close to each other so that $\max_{k \leq n} \|\hat{\tau}^{(k)} - \hat{\tau}_*^{(k)}\|_2 \leq \lambda_{\min}/4L$ for all $n \in \mathbb{N}$. To arrive at the final statement, we can continue from (3.126)

$$\begin{aligned}
\|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 &\leq 2\gamma L\delta_0^2 \frac{\xi^n - (1 - \gamma\lambda_{\min})^{2n}}{\xi - (1 - \gamma\lambda_{\min})^2} + \gamma\Delta_W \frac{1 - \xi^n}{1 - \xi} \\
&\leq \frac{4L\delta_0^2}{\lambda_{\min}}\xi^n + \frac{2\Delta_W}{\lambda_{\min}}\left(1 - \xi^n\right).
\end{aligned}$$

□

3.4.8 Concluding the proof of Theorem 3.4

Combining all results proven throughout the preceding sections is all that remains to proof Theorem 3.4. For an explanation of how all intermediate results connect, we refer to the proof sketch given in Section 3.4.3. The proof only relies on one additional concentration argument given by the following Chernoff bound.

Lemma 3.15. *Let $Z \in \mathbb{R}^m$ be a random vector and assume $\|Z\|_2^2 \leq R$ almost surely. For N independent copies $Z_1, \dots, Z_N$ of Z, define the random matrix*

$$G := \sum_{i=1}^{N} Z_i Z_i^\top.$$

Then, we have

$$\mathbb{P}\left(\lambda_m(G) \geq \frac{\lambda_m(\mathbb{E}G)}{4}\right) \geq 1 - m0.7^{\frac{\lambda_m(\mathbb{E}G)}{R}}.$$

Proof. The result follows directly from the standard matrix Chernoff bound [126, Theorem 1.1]. □

Proof of Theorem 3.4. Denote $E = \mathbb{E}_{X_1,\dots,X_{N_{\text{train}}} \sim \mathcal{N}(0,\mathrm{Id}_D)}[A]$ with A as in (3.84), and constructed from inputs $X_1, \dots, X_{N_{\text{train}}} \sim \mathcal{N}(0, \mathrm{Id}_D)$. According to Lemma 3.9, there exist constants $\omega, C_1 > 0$, which only depend on g and τ_∞, with

$$\lambda_m(E) \geq \omega - C_1 \frac{(m-1)\log^2 m}{D^2} \geq \frac{\omega}{2},$$

provided $(2C_1/\omega) m \log^2 m \leq D^2$, as assumed in Theorem 3.4. Note now that A is a sum of positive semidefinite rank-one matrices. Thus, we can apply the Matrix Chernoff bound in Lemma 3.15 to get the concentration bound

$$\mathbb{P}\left(\lambda_m(A) \geq \frac{\lambda_m(E)}{4}\right) \geq 1 - m \cdot 0.7^{\frac{N_{\text{train}}\lambda_m(E)}{R}}, \tag{3.127}$$

where $R = \sup_{x \in \mathbb{R}^D} \|\nabla_{\hat{\tau}} \hat{f}(\tau, x)\|_2^2 \leq m \left\|g^{(1)}\right\|_\infty^2 \leq m\kappa^2$. From $0.7 < \exp(-1/3)$ now follows that

$$\mathbb{P}\left(\lambda_m(A) \geq \frac{\omega}{8}\right) \geq 1 - m \cdot \exp\left(-\frac{N_{\text{train}}\omega}{6m\kappa^2}\right). \tag{3.128}$$

For the remainder of the proof, we will condition on the event that the bound in (3.128) holds. By the result of Lemma 3.10, the difference between the gradients $\nabla_{\hat{\tau}} J, \nabla_{\hat{\tau}} J_*$ satisfies

$$\|\nabla_{\hat{\tau}} J(\hat{\tau}) - \nabla_{\hat{\tau}} J_*(\hat{\tau})\|_2 \leq 2\kappa^2 \sqrt{m}\, \|\hat{\tau} - \tau\|_2^2 + \Delta_W \tag{3.129}$$

$$\Delta_W = C\Delta_{W,1} + \left(\frac{m^3 \delta_{\max}^2 t}{N_{\text{train}}}\right)^{1/2}, \tag{3.130}$$

for a constant $C > 0$ and $t > 0$ with probability at least $1 - 2m^2 \exp\left(-\frac{t}{C\kappa^4}\right)$ where

$$\Delta_{W,1} \leq \frac{m^{1/2}\log(m)^{3/4}}{D^{1/4}} \left[\|\widehat{W} - W\|_F + \frac{\Delta_{W,O}^{1/2}}{D^{1/2}} + \left\|\sum_{k=1}^{m} w_k - \hat{w}_k\right\|_2\right].$$

Assuming the event associated with (3.129) occurs, we can invoke Lemma 3.14 with $L = 2\kappa^2\sqrt{m}$ meeting its condition by choosing an appropriate constant C in (3.73). Then, for a step-size $\gamma \leq 1/\|A\|$, $\lambda_{\min} = \lambda_m(A)$ and $\xi = 1 - \gamma\lambda_{\min}/2$, Lemma 3.14 yields

$$\|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 \leq \xi^n \|\hat{\tau}^{(0)} - \tau\|_2 + C\,(1 - \xi^n)\,\Delta_W. \tag{3.131}$$

The bound in (3.131) controls the deviation of the gradient descent iteration (3.72) from the idealized gradient descent iteration (3.85). What remains to be shown is that the idealized iteration converges to the correct parameter τ, which follows directly by the lower bound on the minimal eigenvalue $\lambda_{\min}$. In fact, we have $J_*(\hat{\tau}) = (\hat{\tau} - \tau)^\top A(\hat{\tau} - \tau)$ and

$$\begin{aligned}\|\hat{\tau}_*^{(n)} - \tau\|_2 &= \|\hat{\tau}_*^{(n-1)} - \gamma\nabla_{\hat{\tau}} J_*(\hat{\tau}_*^{(n-1)}) - \tau\|_2 = \|(\mathrm{Id}_D - \gamma A)(\hat{\tau}_*^{(n-1)} - \tau)\|_2 \\ &\leq \|\,\mathrm{Id}_D - \gamma A\|^n \|\hat{\tau}_*^{(0)} - \tau\|_2 \leq (1 - \gamma\lambda_{\min})^n \|\hat{\tau}^{(0)} - \tau\|_2.\end{aligned}$$

Applying the triangle inequality to (3.131) therefore yields

$$\begin{aligned}\|\hat{\tau}^{(n)} - \tau\|_2 &\leq \|\hat{\tau}_*^{(n)} - \tau\|_2 + \|\hat{\tau}^{(n)} - \hat{\tau}_*^{(n)}\|_2 \\ &\leq \left((1 - \gamma\lambda_{\min})^n + \xi^n\right) \|\tau^{(0)} - \tau\|_2 + C\,(1 - \xi^n)\,\Delta_W \\ &\leq 2\xi^n \|\hat{\tau}^{(0)} - \tau\|_2 + C\,(1 - \xi^n)\,\Delta_W.\end{aligned}$$

The main statement follows by a union bound over the events described above and by unifying the involved constants. □

3.5 Proof of the Theorem 3.2

We are now in a position to prove the main theorem of this chapter which combines all individual pipeline steps.

Proof of Theorem 3.2. According to our assumptions, there exist C, D_0 such that the conditions of Theorem 3.3 are fulfilled, and therefore we conclude that the ground truth weights obey **(Inc1)** - **(Inc3)** of Definition 2.3 and that the weight recovery (Algorithm 3.2) returns vectors $\mathcal{U}$ such that for all $\hat{w} \in \mathcal{U}$ we have

$$\max_{k\in[m]} \min_{s\in\{-1,+1\}} \|\hat{w} - s w_k\|_2 \leq C_1 (m/\alpha)^{1/4} \epsilon^{1/2}, \tag{3.132}$$

with probability at least

$$1 - \frac{1}{m} - D^2 \exp\left(-\min\{\alpha, 1\} t / C_1\right) - C_1 \exp(-\sqrt{m}/C_1).$$

Denote the weight approximations obtained in the last step by $\{\hat{w}_1, \ldots, \hat{w}_m\} \subset \mathrm{S}^{D-1}$. There exists a permutation π of these vectors such that $w_k \approx \pm\hat{w}_{\pi(k)}$ for all $k \in [m]$. To invoke Proposition 3.1, we now need to make sure that

$$\max_{k\in[m]} \min_{s\in\{-1,+1\}} \left\|\hat{w}_{\pi(k)} - s w_k\right\|_2 \leq \frac{1}{C_2} \frac{D^{1/2}}{m\sqrt{\log m}}. \tag{3.133}$$

By applying the uniform error bound (3.132) above, we have

$$C_1 (m/\alpha)^{1/4} \epsilon^{1/2} \leq \frac{1}{C_2} \frac{D^{1/2}}{m\sqrt{\log m}} \Leftrightarrow \epsilon \leq \frac{\sqrt{\alpha}}{C_1^2 C_2} \frac{D}{m^{5/2} \log m},$$

which is guaranteed by our upper bound (3.13) on ϵ for an appropriate constant. This, in turn, shows that (3.133) is met. Hence, by Proposition 3.1, Algorithm 3.3 returns initial shifts $\hat{\tau}$ such that there exists a $\alpha' \leq \alpha$ such that

$$\begin{aligned}\|\tau - \hat{\tau}\|_2 &\leq C_2\sqrt{m}\epsilon + C_2 m^{3/2}\left(\frac{\log m}{D}\right)^{3/4} \max_{k\in[m]} \min_{s\in\{-1,+1\}} \left\|\hat{w}_{\pi(k)} - s w_k\right\|_2 \\ &\leq C_2\sqrt{m}\epsilon + C_2 m^{3/2}\left(\frac{\log m}{D}\right)^{3/4} C_1 (m/\alpha)^{1/4}\epsilon^{1/2} \leq \frac{1}{C m^{1/2}},\end{aligned}$$

where the last line follows from (3.13) chosen with an appropriate constant $C > 0$. First, note that this implies that the signs learned by the parameter initialization will be correct. We denote this set of signs as $\bar{s}_1, \ldots, \bar{s}_m$. Additionally, the last inequality implies that, for the given step-size, the condition of Theorem 3.4 (see (3.73)) w.r.t. the error in the initial shift is met. Another criterion that has to be met for Theorem 3.4 is that

$$\frac{C m^{1/2}\log(m)^{3/4}}{D^{1/4}}\left(\|\widehat{W} - W\|_F + \frac{\Delta_{W,O}^{1/2}}{D^{1/2}} + \left\|\sum_{k=1}^m w_k - \hat{w}_k\right\|_2\right) \leq \frac{1}{C\sqrt{m}}, \tag{3.134}$$

$$\left(\frac{m^3\delta_{\max}^2 t}{N_{\text{train}}}\right)^{1/2} \leq \frac{1}{C\sqrt{m}}, \tag{3.135}$$

where $\Delta_{W,O} = \sum_{k\neq k'}^m |\langle w_k - \hat{w}_k, w_{k'} - \hat{w}_{k'}\rangle|$. We begin with the upper term and rely on worst-case bounds which express the different quantities in terms of the uniform error $\delta_{\max} = \max_{k\in[m]} \min_{s\in\{-1,+1\}} \left\|\hat{w}_{\pi(k)} - s w_k\right\|_2$.

$$\|W - \widehat{W}\|_F \leq m^{1/2}\delta_{\max}, \tag{3.136}$$

$$\frac{\Delta_{W,O}^{1/2}}{D^{1/2}} \leq \frac{m\delta_{\max}}{D^{1/2}}, \tag{3.137}$$

$$\left\|\sum_{k=1}^m w_k - \hat{w}_k\right\|_2 \leq m\delta_{\max}. \tag{3.138}$$

Based on these bounds and after adjusting the constants, we can simplify (3.134) to

$$\delta_{\max} \leq \frac{D^{1/4}}{C m^2 \log(m)^{3/4}} \Leftrightarrow \epsilon \leq \frac{D^{1/2}\alpha^{1/2}}{C m^{9/2}\log(m)^{3/2}},$$

which is covered by our initial assumptions on the accuracy. Note that this implies for (3.135) by plugging in the bound for $\delta_{\max}$ that

$$\left(\frac{m^3\delta_{\max}^2 t}{N_{\text{train}}}\right)^{1/2} \leq \left(\frac{t D^{1/2}}{N_{\text{train}} m \log(m)^{3/2}}\right)^{1/2}.$$

Using $N_{\text{train}} \geq m$ and $t = D^{1/2}$ this implies

$$\left(\frac{m^3\delta_{\max}^2 t}{N_{\text{train}}}\right)^{1/2} \leq \frac{1}{N_{\text{train}}^{1/2}\log(m)^{3/4}} \leq \frac{1}{C m^{1/2}},$$

for D, m sufficiently large. Therefore, all conditions of Theorem 3.4 are satisfied. Hence, there exists a constant C_4 such that the gradient descent iteration (3.72) started from initial shifts

$\hat{\tau}^{[0]} = \hat{\tau}$ will produce iterates $\hat{\tau}^{[0]}, \ldots, \hat{\tau}^{[N_{\mathrm{GD}}]}$ such that

$$\left\| \tau - \hat{\tau}_{\pi}^{[n]} \right\|_2 \leq \frac{C_4 m^{1/2} \log(m)^{3/4}}{D^{1/4}} \left(\|\widehat{W} - W\|_F + \frac{\Delta_{W,O}^{1/2}}{D^{1/2}} + \left\| \sum_{k=1}^{m} w_k - \hat{w}_k \right\|_2 \right) \tag{3.139}$$

$$+ \left(\frac{m^3 \delta_{\max}^2 t}{N_{\mathrm{train}}} \right)^{1/2} + C_4 \frac{1}{\sqrt{m}} \xi^n, \tag{3.140}$$

for all $n \in [N_{\mathrm{GD}}]$, some permutation π and some constant $\xi \in [0, 1)$ with probability at least

$$1 - m \exp(-N_{\mathrm{train}}/C_4 m) - 2m^2 \exp\left(-D^{1/2}/C_4\right).$$

After unifying the constants and using the bound on $\delta_{\max}$, the statement of Theorem 3.2 follows. □

3.6 Experiments: Reconstruction of shallow neural networks

We corroborate our theoretical end-to-end results with an empirical study. The following experiments are performed using a very similar setting that was used to test parameter identification by empirical risk minimization with stochastic gradient descent in Section 1.3.1. In every instance of our experiment, the recovery pipeline (cf. Algorithm 3.1) is run against a planted shallow teacher network of the type

$$f(x) = \sum_{k=1}^{m} \tanh(w_k^\top x + \tau_k), \tag{3.141}$$

where the network weights are drawn uniformly at random from the unit sphere, i.e., $w_1, \ldots, w_m \in \mathbb{S}^{D-1}$, and the network shifts are sampled according to

$$\tau_1, \ldots, \tau_m \sim_{\text{i.i.d.}} \mathrm{Unif}([-0.5, 0.5]).$$

As in Section 3.6, we compare different network widths $m = 2/5 D^{\beta}$, where the parameter $\beta \in \{\frac{1}{2}, \frac{3}{4}, 1, \frac{5}{4}, \frac{3}{2}, \frac{7}{4}, \frac{15}{8}, \frac{31}{16}, 2\}$ will be referred to as the *order of neurons*. Aside from the order of neurons, we also vary the input dimension of the network $D \in \{20, 25, \ldots, 50\}$. All remaining parameters either depend on m, D or remain fixed throughout the experiments and are summarized for the reader's convenience in Table 3.1.

Weight recovery (Algorithm 3.2). The weight recovery uses $N_h = \lceil \log(D) m \rceil$ Hessian approximations, which were computed via central finite differences of second order with step-size $\epsilon_{\mathrm{FD}} = 10^{-2}$. As evaluation points of the Hessians ($x_1, \ldots, x_{N_h}$ in Algorithm 3.2) we chose standard Gaussians. SPM (cf. Algorithm 2.2) is run with step-size $\gamma = 2$. Each SPM iteration performs 10^3 steps of projected gradient ascent, and we run $R = 5 \log(D) \cdot m$ independent SPM iterations in parallel to decrease recovery time. Note that this is a slight deviation from the definition of Algorithm 2.2 since we start multiple SPM iterations in parallel (for more details on this setup, we refer to Section 4.3.1). We fix the thresholding parameter as $\beta = 0.9$ within Algorithm 2.2 except for the runs with $m = 2/5 D^2$, where we set $\beta = 0.99$.

Parameter initialization (Algorithm 3.3). To compute the directional derivatives in Algorithm 3.3, we rely on a central finite-difference schema with step-size $\epsilon_{\mathrm{FD}} = 10^{-2}$. The interval from which shifts are recovered is set to $[-\tau_\infty, +\tau_\infty]$ is set to $[-0.6, +0.6]$ throughout all experiments.

Network Model	Weight recovery (see Alg. 3.2)
Architectures: Shallow networks $f : \mathbb{R}^D \to \mathbb{R}$ as in (3.141)	μ_X: distribution for Hessians $\mathcal{N}(0, \mathrm{Id})$
$(w_k)_{k\in[m]}$: random weights $w_1, \dots, w_m \sim_{\text{i.i.d.}} \mathrm{Unif}(\mathbb{S}^{D-1})$	N_h: nr. of sampled Hessians ($N_h = \lceil \log(D)m \rceil$)
τ: random shifts $\tau_1, \dots, \tau_m \sim_{\text{i.i.d.}} \mathrm{Unif}([-0.5, 0.5])$	N_{SPM} : restarts of SPM ($\leq 5m \log m$)
D: input dimension $D \in \{20, 25, \dots, 50\}$	N_{PGA} : number of SPM steps (10^3)
m: number of hidden neurons $m = 2/5D^\beta$	ϵ_{FD} : step-size for finite differences ($\epsilon_{\text{FD}} = 10^{-2}$)
with $\beta = \frac{1}{2}, \frac{3}{4}, 1, \frac{5}{4}, \frac{3}{2}, \frac{7}{4}, \frac{15}{8}, \frac{31}{16}, 2$	β: threshold to discard spurious loc. max (0.9, 0.99)
Parameter initialization (see Alg. 3.3)	**Network completion (see Section 3.4)**
$[-\tau_\infty, \tau_\infty]$: interval on which $\tanh^{(2)}$ is invertible ($\tau_\infty = 0.6$)	γ: learning rate of SGD ($\gamma = 10^{-3}$)
ϵ_{FD} : step-size for finite-differences ($\epsilon_{\text{FD}} = 10^{-2}$)	batch size (64)
	N_{SGD} : number of training points ($D^2 m$)
	training time (180 seconds)

Table 3.1: Summary of all hyperparameters for the numerical experiments. A single value in $(\cdot)$-brackets is the default hyperparameter that is used in all experiments. Otherwise, we list ranges that are tested in experiments.

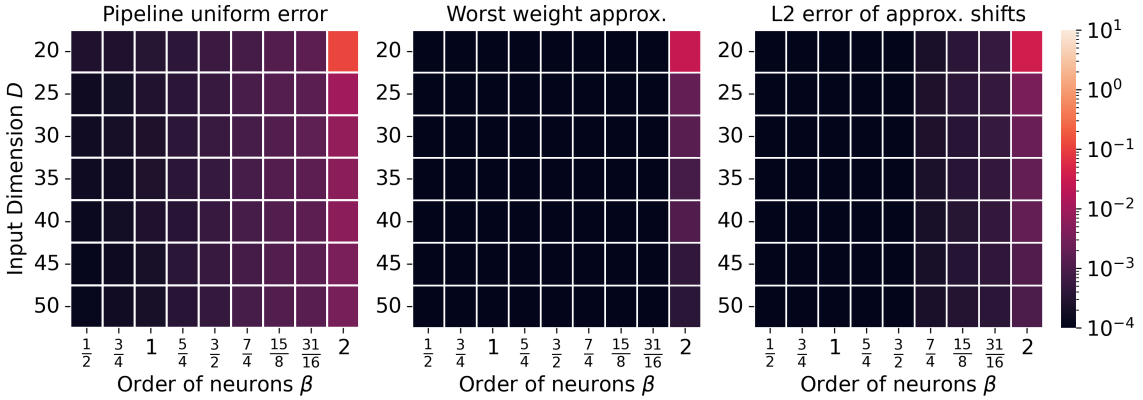

Figure 3.3: Performance of parameter identification via Algorithm 3.1 of shallow networks with tanh activations, $m = 2/5D^\beta$ neurons, and input size D. We observe a consistent identification throughout and increasing performance for higher dimensions (cf. Figure 3.4). Notably, the signs (cf. Algorithm 3.3) were identified correctly in all cases.

Network completion (Section 3.4). For the refinement of the shifts, we run stochastic gradient descent on $N_{GD} = m \cdot D^2$ samples, learning rate $\gamma = 10^{-3}$, and batchsize 64. The training input points are drawn from a standard Gaussian distribution, also used to compute the 10^6 samples for the test data. We stop the refinement by stochastic gradient descent after 180 seconds or once we reach an MSE on the training set below 10^{-8} (which is checked after every epoch).

Evaluation metrics. We track three metrics to assess whether the final network approximation $\hat{f}$ computed by Algorithm 3.1 has identified the teacher network. Firstly, to measure the generalization error, we rely on the uniform error $E_\infty := \max_{x\in\mathcal{X}_{\text{test}}} |\hat{f}(x) - f(x)|$, which is computed over a set of $|\mathcal{X}_{\text{test}}| = 10^6$ unseen inputs sampled from a standard Gaussian distribution. Secondly, we track the uniform error of the network weights (while accounting for permutations), i.e., $\max_{k\in[m]} \min_{k'\in[m]} \|w_k - \hat{w}_{k'}\|_2$. Lastly, we track the ℓ_2 distance between the shifts, i.e., $\|\tau - \hat{\tau}\|_2$, while accounting for potential permutations of the student neurons. All experiments were performed on the hardware described in Section 3.6 (cf. Remark 1.3), and all reported metrics are averaged over ten independent runs.

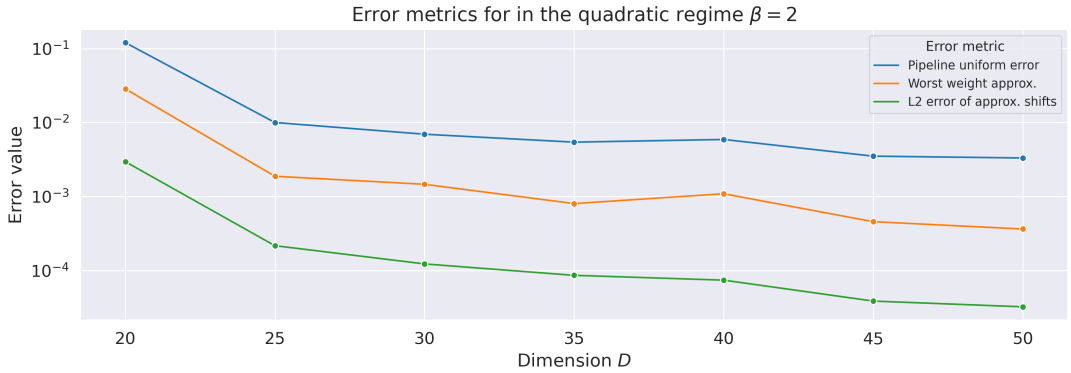

Figure 3.4: Isolated error metrics in the quadratic case $\beta = 2$ as shown in Figure 3.3 for the recovery of shallow networks with $m = 2/5D^{\beta} = 2/5D^2$ neurons. All metrics were averaged over 10 independent runs. We observe an increase in performance for higher dimensions D.

β / D	0.5	0.75	1	1.25	1.5	1.75	1.88	1.94	2
20	4.58	4.53	4.75	4.76	5.25	5.89	6.33	6.6	7.63
25	5.7	5.67	5.75	6.19	7.02	8.38	9.52	10.3	11.4
30	6.77	6.87	6.93	7.44	8.53	10.6	12.6	14.7	17.4
35	8.35	8.41	8.49	9.3	10.6	13.8	18.3	22.1	27.5
40	9.89	9.95	10.3	11.1	12.8	18.2	26.5	33	43.1
45	12.3	12.4	12.9	13.6	16	25.1	38.4	50.7	70.8
50	15	15.2	15.7	16.7	19.9	34.1	57.2	79.8	113

Table 3.2: Elapsed time in seconds after completing weight recovery (Algorithm 3.2) and parameter initialization (Algorithm 3.3).

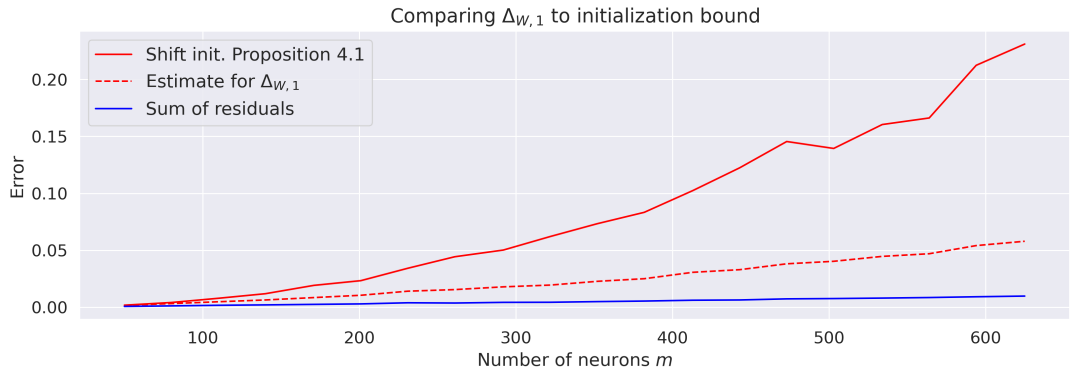

Figure 3.5: Comparison between the guaranteed accuracy of the shift initialization (i.e., before running gradient descent) based on Proposition 3.1 (red) and the residual upper bound $\Delta_{W,1}$ that persists after running the whole pipeline including the refinement via empirical risk minimization (dashed red). The shown results are computed for $D = 50$ and averaged over 10 independent repetitions. Notably, the sum of residual errors $\| \sum_{k=1}^{m} w_k - \hat{w}_k \|_2$ (blue) does not exhibit any error accumulation for our network model. For further details on this discussion we refer to the related comparison in Section 3.4.2.

Interpretation of the results. An overview of the overall performance is given in Figure 3.3. Aside from the quadratic regime $\beta = 2$, these results show a consistent uniform approximation of the original shallow neural network as well as an identification of the true underlying parameters. In the special case where $\beta = 2$ (i.e., the neurons scale like $m = 2/5D^2$), we see slightly worse results for lower dimensions. The fact that the uniform error and the shift error do slightly increase with m is to be expected since the network is a function $f \in [-m, m]$ and $\|\tau\|_2 = \mathcal{O}(m^{1/2})$. By looking at the runs with $\beta = 2$ individually (cf. Figure 3.4) one can clearly see that all relevant error metrics continuously decrease with the ambient dimension D. Hence, the performance for lower dimensions is not representative of the average behavior in higher dimensions. The computation time of the weight recovery and parameter initialization throughout all runs is of the order of seconds (see Table 3.2) reaching a maximum of $112s$ for $D = 50, m = 2/5D^2 = 10^3$ (this includes the time necessary to approximate Hessian matrices and directional derivatives). Therefore, the time necessary for weight identification and parameter initialization (including identification of the signs) remains well below the time budget given to gradient descent, and the overall pipeline finished the network identification in under 5 minutes in all cases. Let us shortly compare these results to the experiments of Section 1.3.1, where we aimed to identify the entire shallow network by empirical risk minimization with stochastic gradient descent. The most notable differences are that our pipeline managed to identify networks in the quadratic case (cf. Figure 3.3) whereas empirical risk minimization struggled for wider shallow networks (cf. Figure 1.3 - 1.4). Additionally, our pipeline managed to reach a good uniform approximation within a much shorter time (< 5 minutes) than what was given to stochastic gradient descent in Section 1.3.1 (50 minutes).

Improvement of the shifts by GD. Corroborating to the discussion in Section 3.4.2, we compare the individual error components that occur in the guarantees for the approximated shifts before gradient descent (i.e., the bound state in Proposition 3.1), which scales with

$$m^{3/2} \left(\frac{\log m}{D} \right)^{3/4} \max_k \|w_k - \hat{w}_k\|_2 \tag{3.142}$$

and after running gradient descent (cf. Theorem 3.2), which scales with

$$\Delta_{W,1} = \frac{m^{1/2}\log(m)^{3/4}}{D^{1/4}} \cdot \left(\|\widehat{W} - W\|_F + \frac{\Delta_{W,O}^{1/2}}{D^{1/2}} + \left\| \sum_{k=1}^{m} w_k - \hat{w}_k \right\|_2 \right). \tag{3.143}$$

In the same setup as before with $D = 50$ we compute the terms (3.142) and (3.143) on real weight approximations over 10 independent runs and for increasing number of neurons $m = 50, 80, 110, \ldots, 620$. The results are shown in Figure 3.5. This shows empirically that within the setting of described in Section 3.1.1 gradient descent is expected to improve upon the shifts computed by Algorithm 3.3. For more details see Section 3.4.2.

Chapter 4

Entangled weights: Moving beyond shallow network architectures

This chapter considers the identification of classical deep artificial neural networks with smooth activations and a pyramidal shape from network queries. More preciously, we provide an extension of the ideas in Chapter 3 to deep neural networks of pyramidal shape with *multiple non-linear hidden layers* and an arbitrary number of outputs. This extension is based on the so-called *entangled weights*, which are defined as the gradients of the network's pre-activations. Entangled weights serve as a generalization of ordinary weights, enabling us to decouple weight information that is accessible via network differentiation. Entangled weights *uniquely identify* suitable deep networks up to a few scaling and shift parameters. We provide proofs, as well as empirical evidence, that entangled weights can be recovered from $\mathcal{O}(D^2 \times m)$ network probes under suitable conditions, where D is the input dimension and m the number of overall neurons of the network. We then show, heuristically, that the remaining parameters of deep neural networks can be identified based on the entangled weights by a reparametrization approach as in Chapter 3 (cf. Section 3.4). These heuristics are complemented by theoretical discussions as well as extensive numerical experiments.

The content of this chapter combines the results within the joint work [52] with Massimo Fornasier, Timo Klock as well as the work [50] with Massimo Fornasier, Timo Klock and Christian Fiedler. We provide a *novel characterization* of entangled weights and extend these underlying publications with further discussions and numerical experiments. Several of the theoretical proofs within this chapter, which originate from these joint works, are stated verbatim as in their underlying references, indicated within the definition of the statement.

4.1 Introduction and preliminaries

In the previous chapter, we demonstrated the reconstruction of smooth shallow neural networks with input dimension D from point queries, where the number of neurons m scales up to $m \log^2 m = \mathcal{O}(D^2)$. The reconstruction pipeline outlined in Chapter 3 relies on decoupling network weights from the remaining network parameters. The weights (or approximations thereof) are learned in the first step by considering the space spanned by second-order derivatives of the network function, which is then decomposed into individual rank-one tensors by the subspace power method (cf. Algorithm 3.2 in Chapter 2). The second step of this approach then learns the remaining parameters (shifts and signs of the weights) via gradient descent, and a local convergence analysis was provided together with an initialization strategy. Nowadays, most artificial neural networks deployed in practice consist of compositions of multiple non-linear layers, so-called deep neural networks (cf. Definition 1.2 in Section 1.1.2). A natural question is

whether the reconstruction method of Chapter 3, in particular the decoupling of the weights, applies to deep neural networks.

In this chapter, we show that the Hessians of certain deep neural networks can be decomposed into rank-one matrices that carry sufficient information to recover all individual weights. Our theory relies on a generalization of ordinary weights, so-called *entangled weights*, defined as the gradients of the pre-activations. Replacing fixed weight vectors with gradients that depend on the network input presents one of the challenges we face with deep architectures. This requires the stabilization of the Hessian span that we realize by a concentration argument for which we provide a theoretical analysis of the induced subspace approximation error. Individual entangled weights can then be learned from this subspace as the unique near-rank-one elements, covered by the theory in Chapter 2. We provide empirical evidence that the recovery of a fixed set of entangled weights is possible for deep pyramidal architectures when the number of neurons m does not exceed a certain threshold that depends on the input dimension and the number of output neurons ($m \lesssim D \cdot m_L$). As a side result, we implicitly extend the reconstruction pipeline from the previous chapter, where shallow neural networks with one output were considered, to several output neurons.
Once the entangled weights have been recovered, we need to infer the remaining unknowns of the network. By constructing the Hessian span and applying SPM to find the rank-one components, we can attain the right set of entangled weights, but we lose any structural information (i.e., which entangled weight corresponds to which neuron/layer). We provide heuristics that consistently detect the weights of the outer layers (first layer and last layer) but currently can not reliably assign the entangled weight vectors of the inner hidden layers. This limits the full reconstruction pipeline to networks with $L \leq 3$ layers. Note, however, that all remaining algorithmic steps do apply to deeper architectures. After a correct assignment of the entangled weights, the remaining part of the reconstruction is to learn the scale and shift parameters of the network. This closely resembles the state in Chapter 3, which is reached after the weights have been learned. The reconstruction of the network from entangled weights requires a pyramidal architecture up to the penultimate layer and that the number of neurons in the first layer does not exceed the input dimension (the last layer can have arbitrarily many neurons). As before, the remaining parameters of the network can be learned by minimizing a suitable least squares objective.

Section 4.1.3 will include a detailed summary of our reconstruction pipeline. Lastly, let us mention that the results contained in this chapter differ from the theory in Chapter 3. We can not offer the same *end-to-end* theoretical analysis that covers all pipeline steps as it has been provided in Chapter 3. The main theoretical guarantees will be centered around the recovery of entangled weights. However, we provide theoretical discussions of all remaining pipeline steps complemented with extensive numerical results.

4.1.1 Problem setting: Deep neural network model

Let us first formally introduce our network model alongside some notation. Then, throughout this chapter, we consider the identification of fully connected feed-forward artificial neural networks from point queries.

Notation for deep networks. According to our Definition 1.2, feed-forward deep neural networks are functions parameterized by a set of weight matrices $W^{[1]}, \dots, W^{[L]}$ and shift vectors $\tau^{[1]}, \dots, \tau^{[L]}$. Here, $L \in \mathbb{N}$ equals the number of network layers. We often rely on the exponent with the layer index in square brackets as in $W^{[\ell]}$ to indicate association to

the ℓ-th layer. Since we are only considering fully connected networks, the architecture of a network is determined by the number of neurons in each layer. We distinguish between hidden layers/neurons ($\ell = 1, \dots, L-1$) and the output layer/neurons ($\ell = L$). Similarly to the previous chapter, we denote the input dimension to a network by either $D \in \mathbb{N}$ or $m_0 \in \mathbb{N}$, the numbers of neurons in layer $\ell \in [L]$ by m_ℓ and the overall number of neurons by $m = \sum_{\ell=1}^{L} m_\ell$. Notably, the dimensions of the weight matrices and shifts are given by the number of neurons in the form

$$W^{[\ell]} = \begin{bmatrix} w_1^{[\ell]} & \dots & w_{m_\ell}^{[\ell]} \end{bmatrix} \in \mathbb{R}^{m_{\ell-1} \times m_\ell} \quad \text{and} \quad \tau^{[\ell]} \in \mathbb{R}^{m_\ell} \quad \text{for all } \ell \in [L]. \tag{4.1}$$

Hence, for a given set of parameters $(W^{[\ell]})_{\ell \in [L]}, (\tau^{[\ell]})_{\ell \in [L]}$ the number of neurons and therefore the overall architecture can be derived from the dimensions of the parameters. Every neuron is equipped with an activation function, which typically refers to a non-linear scalar function $g : \mathbb{R} \to \mathbb{R}$. We allow different activation functions $g_1, \dots, g_L$ for each individual layer. Let us note here that this is mainly done for the sake of generality and that, typically, all hidden layers share the same activation function $g_1 = g_2 = \dots, g_{L-1}$. However, it is common to consider different functions at the output neurons to not limit the network to the image of the activation function. A common example is the linear output function ($g_L = \mathrm{Id}$). For an L-layer neural network f with parameters $(W^{[\ell]})_{\ell \in [L]}, (\tau^{[\ell]})_{\ell \in [L]}$ and activation functions $(g_\ell)_{\ell \in [L]}$, its output for some input $x \in \mathbb{R}^D$ is then computed by the iteration

$$y^{[0]}(x) := x, \tag{4.2}$$

$$y^{[\ell]}(x) := g_\ell \left((W^{[\ell]})^\top y^{[\ell-1]}(x) + \tau^{[\ell]} \right), \quad \text{for all } \ell \in [L], \tag{4.3}$$

$$f(x) := y^{[L]}(x), \tag{4.4}$$

such that f is a mapping from $\mathbb{R}^D$ to $\mathbb{R}^{m_L}$. The output function of layer ℓ w.r.t. some input, defined as $y^{[\ell]}(\cdot) : \mathbb{R}^D \to \mathbb{R}^{m_\ell}$, is typically called the *post-activation* of layer ℓ. Similarly, we define the *pre-activations* of f, as the set of functions w.r.t. the input that is given by

$$z^{[\ell]}(x) := (W^{[\ell]})^\top y^{[\ell-1]}(x) + \tau^{[\ell]}, \quad \text{for all } \ell \in [L], \tag{4.5}$$

which can be seen as the state of the neurons in the ℓ-th layer before any activation function is applied. For the sake of simplicity, throughout this chapter, we will introduce the following notation.

Definition 4.1. *Consider natural numbers $L, D, m_1, \dots, m_L \in \mathbb{N}$ and scalar functions $g_1, \dots, g_L$. Based on the definition of neural networks in Definition 1.2, we define the class of neural networks with L layers, D inputs, m_ℓ neurons in the ℓ-th layer, and activation functions $g_1, \dots, g_L$ as*

$$\mathcal{NN}(D, (m_\ell)_{\ell \in [L]}, (g_\ell)_{\ell \in [L]}) := \{ f : \mathbb{R}^D \to \mathbb{R}^{m_L} | f \text{ is constructed as in } (4.1) - (4.4) \}. \tag{4.6}$$

Assumptions on the network model Notably, we do not make any general distributional assumptions about the network weights or shifts. All statements that rely on specific properties associated with these parameters will state these explicitly. However, let us mention that most of our theory builds upon the assumption that weights within each layer are mutually incoherent. Furthermore, in our numerics we construct the weights and shifts as in Chapter 3, i.e., we typically sample random weights uniformly from the unit-sphere. Similar to Section 3.1.1, we also rely on certain properties of the activation function and the network itself. These conditions will be summarized in the following:

Assumption 4.1 (Deep network model)**.**

(DNM1) *We assume that the activation functions are sufficiently smooth with bounded derivatives such that $g_\ell \in \mathcal{C}^3(\mathbb{R})$ for all $\ell \in [L]$ and that*

$$\kappa := \max_{\ell \in [L], n \in [3]} \left\| g_\ell^{(n)} \right\|_{L^\infty(\mathbb{R})} < \infty, \tag{4.7}$$

for some constant $\kappa > 0$.

(DNM2) *We assume that all hidden layers are equipped with non-linear activation functions with sufficiently rich second-order information ($\|g_\ell^{(2)}\|_{L^\infty(\mathbb{R})} > 0$ for all $\ell < L$). Hence, the overall number of non-linear neurons m^* with sufficiently rich second-order information is at least*

$$m^* = \sum_{\ell=1}^{L} m_\ell \cdot \mathbb{1}_{\|g_\ell^{(2)}\|_{L^\infty(\mathbb{R})} > 0} \geq m_1 + \cdots + m_{L-1}. \tag{4.8}$$

(DNM3) *Consider a neural network $f : \mathbb{R}^D \to \mathbb{R}^{m_L}$ on m^* non-linear neurons (cf. **(DNM2)**) and a sub-Gaussian distribution μ_X with $\operatorname{supp}(\mu_X) \subseteq \mathbb{R}^D$. We say f fulfills the learnability condition on μ_X if*

$$\alpha := \sigma_{m^*} \left(\frac{1}{m_L} \sum_{k_L=1}^{m_L} \mathbb{E}_{X \sim \mu_X} \left[\operatorname{vec}(\nabla^2 f_{k_L}(X)) \otimes \operatorname{vec}(\nabla^2 f_{k_L}(X)) \right] \right) > 0, \tag{4.9}$$

where σ_{m^*} denotes the m^*-th singular value and f_{k_L} denotes the subnetwork of f when only the k_L-th output neuron is considered.

(DNM4) *The hidden layers have a pyramidal structure, i.e., the number of neurons in the hidden layers is non-increasing and bounded by the input size*

$$D \geq m_1 \geq m_2 \geq \cdots \geq m_{L-1}. \tag{4.10}$$

The first condition **(DNM1)** guarantees that we can rely on third-order derivatives of the network. The derivatives' boundedness implies the Lipschitz continuity of the network and its first two derivatives. Clearly, a linear function would satisfy (**(DNM1)**), but its higher-order derivatives do not carry any information. Condition **(DNM2)** guarantees that all hidden neurons have sufficient Hessian information by ensuring that the derivative of their activation is non-affine while still allowing us to consider networks with linear output neurons (e.g., $g_L = \mathrm{Id}$). Condition **(DNM3)** closely resembles the learnability condition made in the shallow network reconstruction of Chapter 3 (cf. Assumption 4.1, **(SNM3)**) and other comparable works [54, 52, 50]. For $\mu_X = \mathcal{N}(0, \mathrm{Id}_D)$ and shallow neural networks with a single output of the type

$$f(x) = \sum_{k=1}^{m} g(w_k^\top x + \tau_k),$$

it was shown in Theorem 3.5 (see Section 3.4.5) that, under suitable incoherence of the involved weights and for sufficiently non-polynomial activations, the condition **(DNM3)** fulfilled for a constant α independent of D, m (see the related discussion in Section 3.1.1). For deep neural networks, the assumption **(DNM3)** is motivated identically to **(SNM3)**: To enable the reconstruction of deep networks via second-order information, we need to ensure that, for inputs taken from a reasonable distribution, the corresponding span of Hessians matrices is sufficiently rich in that it contains one rank-one component for every (non-linear) neuron. For further details, we refer to Section 4.2. The reason why a pyramidal structure of the hidden layers as in **(DNM4)** is required (cf. condition **(DNM4)**) for a full reconstruction of the network will be made clear in the next section.

Assumptions on the general setting. Aside from the conditions **(DNM1) - (DNM4)**, which characterize properties of the network itself, we do assume that the network can be probed, which gives (approximate) access to its first- and second-order derivatives by numerical differentiation.

Assumption 4.2 (Algorithmic assumptions)**.**

(G5.1) *We assume that the network can be queried without noise*

(G5.2) *Furthermore, we assume the setting where the architecture of f (i.e., the number of neurons per layer) is given.*

(G5.3) *We assume access to a numerical differentiation method for the gradients and Hessians of the network outputs. These approximations are denoted by $\Delta f_{k_L} \approx \nabla f_{k_L}, \Delta^2 f_{k_L} \approx \nabla^2 f_{k_L}$ for the gradients and Hessians, respectively. For a given network, we denote by $\hat{C}_\epsilon$ the uniform bound of either two numerical approximation methods, i.e., $\hat{C}_\epsilon$ is such that*

$$\sup_{x\in\mathbb{R}^D} \max_{k_L\in[m_L]} \max \left\{ \|\nabla f_{k_L}(x) - \Delta f_{k_L}(x)\|_2 , \left\|\nabla^2 f_{k_L}(x) - \Delta^2 f_{k_L}(x)\right\|_F \right\} \leq \hat{C}_\epsilon.$$

Since **(G5.1) - (G5.3)** are morally identical to the conditions **(G4.1)-(G4.2)**, we refer to the related discussion in Section 3.1.1.

4.1.2 Preliminaries: Entangled weights

The recovery of the parameters of shallow networks in Chapter 3 is based around a decoupling strategy that relies on the simple principle that differentiation of a function of the type

$$f(x) = \sum_{k=1}^{m} g(w_k^\top x + \tau_k)$$

will expose the weight vectors. More precisely, in Section 3.2, we have relied on the fact that for sufficiently smooth activation functions, the Hessians of shallow networks read as

$$\nabla^2 f(x) = \sum_{k=1}^{m} g^{(2)}(w_k^\top x + \tau_k) w_k \otimes w_k. \tag{4.11}$$

This allows a reduction of the weight recovery to the recovery of a rank-one basis from the span of randomly sampled Hessian matrices using the theory from Chapter 2. Unfortunately, one can easily confirm that such a direct relationship between the weights and higher-order derivatives, as in (4.11), does not exist for networks with multiple hidden layers. In this section, we will introduce a new concept of weight vectors, so-called *entangled weights* (cf. [52, 50]). We will see that these objects serve as a generalization of ordinary network weights that can be accessed through Hessian information. Let us begin by formally introducing entangled weights:

Definition 4.2 (Entangled Weights)**.** *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$ as defined in Definition 4.1 with activations $(g_\ell)_{\ell\in[L]}$ that are differentiable. For each neuron, specified by $\ell \in [L]$ and $k_\ell \in [m_\ell]$, we define the entangled weight (function) $v_{k_\ell}^{[\ell]}(x)$ at input x as the gradient of its pre-activation:*

$$v_{k_\ell}^{[\ell]}(x) := \nabla z_{k_\ell}^{[\ell]}(x), \tag{4.12}$$

where $z_{k_\ell}^{[\ell]}(x)$ is the pre-activation of the k_ℓ-th neuron in layer ℓ. Furthermore, for any $x \in \mathbb{R}^D$ we denote by $\mathcal{V}(f,x)$ the set of all entangled weights fixed at input x, i.e., we have

$$\mathcal{V}(f,x) := \left\{ v_{k_\ell}^{[\ell]}(x) \middle| \ell \in [L], k_\ell \in [m_\ell] \right\} \subset \mathbb{R}^D, \tag{4.13}$$

and for any $\ell \in [L]$ we denote by $\mathcal{V}_\ell(f,x)$ the set of entangled weights

$$\mathcal{V}_\ell(f,x) := \left\{ v_{k_\ell}^{[\ell]}(x) \middle| k_\ell \in [m_\ell] \right\} \tag{4.14}$$

corresponding to layer ℓ and by $V^{[\ell]}(x)$ the entangled weight matrices

$$V^{[\ell]}(x) = \begin{bmatrix} v_1^{[\ell]}(x) & \dots & v_{m_\ell}^{[\ell]}(x) \end{bmatrix} \in \mathbb{R}^{D \times m_\ell}. \tag{4.15}$$

Remark 4.1. *By the preceding definition, entangled weights are vector-valued functions depending on the network input. However, in some cases, we might neglect the dependency of $v_{k_\ell}^{[\ell]}(x)$ on the input x and regard entangled weights as fixed vectors. In this case, we implicitly assume that the input x remains fixed and can be inferred from the context.*

One way to interpret Definition 4.2 is to understand entangled weights as a generalization of normal weights. Clearly, by the definition above, there is no difference between ordinary weights and entangled weights for the neurons of the first hidden layer since the entangled weight functions corresponding to the first layer are constant:

$$v_{k_1}^{[1]}(x) = \nabla \left(\langle w_{k_1}^{[1]}, x \rangle + \tau_{k_1}^{[1]} \right) = w_{k_1}^{[1]}, \quad \text{for all } x \in \mathbb{R}^D, k_1 \in [m_1]. \tag{4.16}$$

Ordinary weights encode the direction of the greatest increase of a neuron w.r.t. its input. However, in the identification problem, we regard deep neural networks as black-box models, which makes it impossible to directly observe how information is passed between hidden layers. Hence, we can not observe the input to individual hidden neurons, but we *can observe* and control the input to the network itself **(G5.1)**. With the interpretation that ordinary weights encode the direction of the greatest increase of individual neurons, Definition 4.2 becomes a natural extension of this concept to a setting where only the input of the network can be observed. This definition of weights has to factor in the state of the preceding network layers since the output of a neuron generally depends on the value of all preceding neurons. This results in the dependency of the entangled weights (4.12) on the input x for all but the first layer.

Let us note that in [52, 50] the authors have introduced entangled weights as the product of weight matrices intertwined with diagonal matrices (cf. statement below). This difference is just a matter of presentation and, in fact, both definitions are equivalent, which can be seen from the following identities.

Lemma 4.1. *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell \in [L]}, (g_\ell)_{\ell \in [L]})$ as defined in Definition 4.1 with activations $(g_\ell)_{\ell \in [L]}$ that are differentiable. Then, the entangled weights in Definition 4.2 admit the following decompositions:*

$$v_{k_\ell}^{[\ell]}(x) = \left(\prod_{j=1}^{\ell-1} W^{[j]} \operatorname{diag}\left(g_j^{(1)}(z^{[j]}(x)) \right) \right) w_{k_\ell}^{[\ell]}, \tag{4.17}$$

$$V^{[\ell]}(x) = \left(\prod_{j=1}^{\ell-1} W^{[j]} \operatorname{diag}\left(g_j^{(1)}(z^{[j]}(x)) \right) \right) W^{[\ell]}, \tag{4.18}$$

where $W^{[1]}, \dots, W^{[L]}$ are the weight matrices of the network f.

Proof. Note that the second statement (4.18) is a trivial consequence of (4.17) since (4.18) follows by applying the first identity column-wise. We follow an induction-type argument. For $\ell = 1$, we have equivalence between weights and the entangled weights (cf. (4.16)), so both statements hold for $\ell = 1$ since $\Pi_{j=1}^{0} = 1$. Assume (4.18) holds up to some $\ell < L$, then, for any $k_{\ell+1} \in [m_{\ell+1}]$, we have

$$v_{k_{\ell+1}}^{[\ell+1]}(x) = \nabla z_{k_{\ell+1}}^{[\ell+1]}(x) = \nabla \left(\langle w_{k_{\ell+1}}^{[\ell+1]}, y^{[\ell]}(x) \rangle + \tau_{k_{\ell+1}}^{[\ell+1]} \right) = \nabla \langle w_{k_{\ell+1}}^{[\ell+1]}, y^{[\ell]}(x) \rangle \quad \text{for all } x \in \mathbb{R}^D.$$

Expanding the inner product yields $\nabla \langle w_{k_{\ell+1}}^{[\ell+1]}, y^{[\ell]}(x) \rangle = \sum_{k_\ell=1}^{m_\ell} w_{k_\ell,k_{\ell+1}}^{[\ell+1]} \nabla y_{k_\ell}^{[\ell]}(x)$. Now, by the chain rule, we have

$$\sum_{k_\ell=1}^{m_\ell} w_{k_\ell,k_{\ell+1}}^{[\ell+1]} \nabla y_{k_\ell}^{[\ell]}(x) = \sum_{k_\ell=1}^{m_\ell} w_{k_\ell,k_{\ell+1}}^{[\ell+1]} g_\ell^{(1)} \left(z_{k_\ell}^{[\ell]}(x) \right) \nabla z_{k_\ell}^{[\ell]}(x) = V^{[\ell]}(x) \operatorname{diag}(g_\ell^{(1)}(z^{[\ell]}(x))) w_{k_{\ell+1}}^{[\ell+1]}.$$

Since (4.18) holds for all $\ell' \leq \ell$, we can conclude the induction step with

$$\begin{aligned} v_{k_{\ell+1}}^{[\ell+1]}(x) &= V^{[\ell]}(x) \operatorname{diag}(g_\ell^{(1)}(z^{[\ell]}(x))) w_{k_{\ell+1}}^{[\ell+1]} \\ &= \left(\prod_{j=1}^{\ell-1} W^{[j]} \operatorname{diag} \left(g_j'(z^{[j]}(x)) \right) \right) W^{[\ell]} \operatorname{diag}(g_\ell^{(1)}(z^{[\ell]}(x))) w_{k_{\ell+1}}^{[\ell+1]}, \end{aligned}$$

which establishes the identity for $\ell + 1$. The proof then follows by induction. □

For a fixed input $x \in \mathbb{R}^D$, every neuron is associated with exactly one entangled weight vector. A trivial consequence of Definition 4.2 is that the entangled weights corresponding to the output neurons can directly be exposed via their gradient: Consider a network $f \in \mathcal{NN}(D, (m_\ell)_{\ell \in [L]}, (g_\ell)_{\ell \in [L]})$. If we take the gradient of the output neurons $f_1, \ldots, f_{m_L}$ at input x, then this yields the entangled weights $v_1^{[L]}(x), \ldots, v_{m_L}^{[L]}(x)$ up to a scaling factor. More precisely, for any $k_L \in [m_L]$, we have

$$\nabla f_{k_L}(x) = \nabla y_{k_L}^{[L]}(x) = \nabla g(z_{k_L}^{[L]}(x)) = g^{(1)}(z_{k_L}^{[L]}(x)) \nabla z_{k_L}^{[L]}(x) = g^{(1)}(z_{k_L}^{[L]}(x)) v_{k_L}^{[L]}(x). \tag{4.19}$$

This still leaves the question of how we can access the entangled weights of the hidden neurons. However, the following statement shows that we can expose the entangled weights of all hidden neurons by the same principle used in Chapter 3, namely considering the Hessians of the output neurons.

Proposition 4.1. *Let $f \in \mathcal{NN}(D, (m_\ell)_{\ell \in [L]}, (g_\ell)_{\ell \in [L]})$ be a neural network as in Definition 4.1 with activations $(g_\ell)_{\ell \in [L]}$ that are twice differentiable. The Hessian of any output neuron f_{k_L}, where $k_L \in [L]$, has the following decomposition into entangled weights:*

$$\nabla^2 f_{k_L}(x) = g_L^{(2)}(z_{k_L}^{[L]}(x)) v_{k_L}^{[L]}(x)^{\otimes 2} + \sum_{\ell=1}^{L-1} \sum_{k_\ell=1}^{m_\ell} \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x)) v_{k_\ell}^{[\ell]}(x)^{\otimes 2}. \tag{4.20}$$

Proof. An output neuron is simply the post-activation of the corresponding neuron $f_{k_L}(x) = y_{k_L}^{[L]}(x)$ in the last layer. We begin by deriving an iterative formula for Hessians of post-activations. Note that the gradient of any post-activation is $\nabla y_{k_\ell}^{[\ell]}(x) = g^{(1)}(z_{k_\ell}^{[\ell]}(x)) \cdot v_{k_\ell}^{[\ell]}(x)$. For the first layer, where $\ell = 1$, and any $k_1 \in [m_1]$ we have

$$\begin{aligned} \nabla^2 y_{k_1}^{[1]}(x) &= g_1^{(2)}(z_{k_1}^{[1]}(x)) v_{k_1}^{[1]}(x)^{\otimes 2} + g_1^{(1)}(z_{k_1}^{[1]}(x)) \nabla^2 z_{k_1}^{[1]}(x) \\ &= g_1^{(2)}(z_{k_1}^{[1]}(x)) v_{k_1}^{[1]}(x)^{\otimes 2} + g_1^{(1)}(z_{k_1}^{[1]}(x)) \nabla^2 \langle w_{k_1}^{[1]}, x \rangle = g_1^{(2)}(z_{k_1}^{[1]}(x)) v_{k_1}^{[1]}(x)^{\otimes 2}. \end{aligned} \tag{4.21}$$

Now let $\ell \geq 2$ and $k_\ell \in [m_\ell]$, then

$$\begin{aligned}\nabla^2 y^{[\ell]}_{k_\ell}(x) &= g^{(2)}_\ell(z^{[\ell]}_{k_\ell}(x))v^{[\ell]}_{k_\ell}(x)^{\otimes 2} + g^{(1)}_\ell(z^{[\ell]}_{k_\ell}(x))\nabla^2 z^{[\ell]}_{k_\ell}(x) \\ &= g^{(2)}_\ell(z^{[\ell]}_{k_\ell}(x))v^{[\ell]}_{k_\ell}(x)^{\otimes 2} + g^{(1)}_\ell(z^{[\ell]}_{k_\ell}(x))\sum_{k_{\ell-1}=1}^{m_{\ell-1}} w^{[\ell]}_{k_{\ell-1},k_\ell}\nabla^2 y^{[\ell-1]}_{k_{\ell-1}}(x) \\ &= g^{(2)}_\ell(z^{[\ell]}_{k_\ell}(x))v^{[\ell]}_{k_\ell}(x)^{\otimes 2} + \sum_{k_{\ell-1}=1}^{m_{\ell-1}} \frac{\partial y^{[\ell]}_{k_\ell}(x)}{\partial y^{[\ell-1]}_{k_{\ell-1}}(x)}\nabla^2 y^{[\ell-1]}_{k_{\ell-1}}(x). \end{aligned} \tag{4.22}$$

Consider now the decomposition (4.22) for $\ell = L$ and any $k_L \in [m_L]$, we can keep expanding the term $\nabla^2 y^{[\ell-1]}_{k_{\ell-1}}(x)$ appearing in the sum on the right-hand side as follows

$$\begin{aligned}\sum_{k_{L-1}=1}^{m_{L-1}} \frac{\partial y^{[L]}_{k_L}(x)}{\partial y^{[L-1]}_{k_{L-1}}(x)}\nabla^2 y^{[L-1]}_{k_{L-1}}(x) &= \sum_{k_{L-1}=1}^{m_{L-1}} \frac{\partial y^{[L]}_{k_L}(x)}{\partial y^{[L-1]}_{k_{L-1}}(x)} g^{(2)}_{L-1}(z^{[L-1]}_{k_{L-1}}(x))v^{[L-1]}_{k_{L-1}}(x)^{\otimes 2} \\ &\quad + \sum_{k_{L-2}=1}^{m_{L-2}}\sum_{k_{L-1}=1}^{m_{L-1}} \frac{\partial y^{[L]}_{k_L}(x)}{\partial y^{[L-1]}_{k_{L-1}}(x)}\frac{\partial y^{[L-1]}_{k_{L-1}}(x)}{\partial y^{[L-2]}_{k_{L-2}}(x)}\nabla^2 y^{L-2}_{k_{L-2}}(x) \\ = \sum_{k_{L-1}=1}^{m_{L-1}} \frac{\partial y^{[L]}_{k_L}(x)}{\partial y^{[L-1]}_{k_{L-1}}(x)} g^{(2)}_{L-1}(z^{[L-1]}_{k_{L-1}}(x))v^{[L-1]}_{k_{L-1}}(x)^{\otimes 2} &+ \sum_{k_{L-2}=1}^{m_{L-2}} \frac{\partial y^{[L]}_{k_L}(x)}{\partial y^{[L-2]}_{k_{L-2}}(x)}\nabla^2 y^{[L-2]}_{k_{L-2}}(x). \end{aligned} \tag{4.23}$$

Plugging the expression in (4.23) into (4.22) for $\ell = L$ yields

$$\begin{aligned}\nabla^2 y^{[L]}_{k_L}(x) = g^{(2)}_L(z^{[L]}_{k_L}(x))v^{[L]}_{k_L}(x)^{\otimes 2} &+ \sum_{k_{L-1}=1}^{m_{L-1}} \frac{\partial y^{[L]}_{k_L}(x)}{\partial y^{[L-1]}_{k_{L-1}}(x)} g^{(2)}_{L-1}(z^{[L-1]}_{k_{L-1}}(x))v^{[L-1]}_{k_{L-1}}(x)^{\otimes 2} \\ &+ \sum_{k_{L-2}=1}^{m_{L-2}} \frac{\partial y^{[L]}_{k_L}(x)}{\partial y^{[L-2]}_{k_{[L-2]}}(x)}\nabla^2 y^{L-2}_{k_{L-2}}(x). \end{aligned} \tag{4.24}$$

We can now reapply the expansion in (4.23) to the sum appearing in the last line of (4.24) iteratively up to the first layer at which the iteration will eventually terminate since for $\ell = 1$ the pre-activation $z^{[1]}_{k_1}(x)$ is linear in x and so the second derivative $\nabla^2 z^{[1]}_{k_1}(x)$ vanishes (cf. (4.21)). By this argument, we will end up exactly with the decomposition in (4.20). □

The preceding proposition allows us to relate the Hessians of deep neural networks to the outer products of the entangled weights. For a fixed input $x \in \mathbb{R}^D$, the decomposition in (4.20) has a strong resemblance with the decomposition (4.11) used in Chapter 3. For every entangled weight, the corresponding matrix $v^{[\ell]}_{k_\ell}(x)^{\otimes 2}$ is weighted by the second derivative of the pre-activation multiplied by the partial derivative $\partial f_{k_L}(x)/\partial y^{[\ell]}_{k_\ell}(x)$ which can be interpreted as the influence of the neuron on the output f_{k_L} at x. Note that different output neurons do not directly depend on each other, i.e., we have

$$\frac{\partial f_{k_L}(x)}{\partial y^{[L]}_{k'_L}(x)} = \frac{\partial f_{k_L}(x)}{\partial f_{k'_L}(x)} = \delta_{k_L,k'_L} \quad \text{for all } k_L, k'_L \in [m_L],$$

where δ_{k_L,k'_L} denotes the Kronecker delta. Hence, the decomposition in (4.20) admits a slightly more compact form: Let $k_L \in [L]$, then

$$\nabla^2 f_{k_L}(x) = \sum_{\ell=1}^{L} \sum_{k_\ell=1}^{m_\ell} \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x)) v_{k_\ell}^{[\ell]}(x)^{\otimes 2} \tag{4.25}$$

$$= \sum_{\ell=1}^{L} V^{[\ell]}(x) S_{k_L}^{[\ell]}(x) V^{[\ell]}(x)^\top, \tag{4.26}$$

for diagonal matrices $S_{k_L}^{[\ell]}(x) \in \mathbb{R}^{m_\ell \times m_\ell}$ with entries given by

$$(S_{k_L}^{[\ell]})_{k_\ell} = \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x)). \tag{4.27}$$

For the special case of networks where g_L is a linear function (cf. network model in [52]), the Hessians at some input will only be decomposable into outer products of entangled weights $\mathcal{V}(f,x) \setminus \mathcal{V}_L(f,x)$. Thus, the decomposition above will not contain the entangled weights of the last layer. This can be seen directly from (4.20), where for $g_L = \mathrm{Id}$ the coefficients $g_L^{(2)}(z_{k_L}^{[L]}(x))$ will always be zero. It was already mentioned at the beginning of the section that the entangled weights corresponding to the last layer can be simply computed separately as the gradients of the output neurons. Therefore, it is unproblematic that the Hessians of networks with linear outputs do not contain the entangled weights of the last layer.

Motivating entangled weights. So far, we have defined entangled weights and established their connection to the first- and second-order derivatives of certain deep neural networks. However, we have not yet fully explained how entangled weights serve in the identification of such networks. To a certain degree, (4.16) already reveals a connection between entangled weights and ordinary network weights of the first layer. Let us now describe how entangled weights *encode the remaining weight information* for suitable network architectures: The proof of Lemma 4.1, in particular, the identity

$$V^{[\ell]} = V^{[\ell-1]}(x)\,\mathrm{diag}(z^{[\ell-1]}(x)) W^{[\ell]},$$

already outlines a path how entangled weights can be used for the recovery of the network weights $(W^{[\ell]})_{\ell \in [L]}$. Assuming that the left-inverse of an entangled weight matrix $V^{[\ell-1]}(x)^\dagger$ exists, then the weight matrices can be computed up to diagonal matrices $D_1, \ldots, D_L$ by

$$V^{[\ell-1]}(x)^\dagger V^{[\ell]}(x) = \mathrm{diag}\left(g^{(1)}(z^{[\ell-1]}(x))\right) W^{[\ell]}.$$

Hence, if we were given the exact entangled weight matrices $V^{[1]}(x^*), \ldots V^{[L]}(x^*)$ for a fixed input x^*, we could infer the original weights up to a transformation with diagonal matrices, i.e., attain the matrices

$$D_1 W^{[1]}, D_2 W^{[2]}, \ldots D_L W^{[L]},$$

as long as the first $L-1$ entangled weight matrices are left invertible. This is equivalent to the condition that $V^{[1]}(x^*), \ldots V^{[L-1]}(x^*)$ all have full column rank and, due to their dimensions, this implies that the hidden layers of the network need to have a pyramidal structure, such that

$$D \geq m_1 \geq m_2 \geq \cdots \geq m_{L-1},$$

which relates to our assumption **(DNM4)**. Therefore, to infer the weights (up to a rescaling of the rows) from the entangled weight matrices, the number of neurons up to the penultimate

layer has to be non-increasing, but the number of output neurons can be arbitrarily large. The connection between entangled weights and ordinary network weights is made precise in Section 4.3. More precisely, we show in Proposition 4.3 that the entangled weights of a neural network, as described in Section 4.1.1, give rise to a loss-free reparametrization of the network. The network's reparametrization only depends on the unknown shifts and diagonal matrices. This approach is closely related to the reparametrization used for the shift recovery in Section 3.4. Similarly to Chapter 3, we then fit the reparametrization via entangled weights by relying on empirical risk minimization. For more details, we refer to the related discussion in Section 4.3.2.

Let us now conclude the introduction to entangled weights with a few auxiliary results derived from **(DNM1)**, which establish that the entangled weight functions are bounded and Lipschitz continuous w.r.t. network complexity and κ from **(DNM1)**.

Lemma 4.2 (cf. [50, Lemma 6])**.** *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$ satisfying **(DNM1)** with $\kappa > 0$. Then, for any $x, x' \in \mathbb{R}^D$, the following bounds are satisfied:*

$$\|V^{[\ell]}(x)\| \leq \kappa^{\ell-1} \prod_{k=1}^{\ell} \|W^{[k]}\|, \tag{4.28}$$

$$\|V^{[\ell]}(x) - V^{[\ell]}(x')\| \leq \left(\prod_{k=1}^{\ell} \|W^{[k]}\|\right) \sum_{k=1}^{\ell-1} \kappa^{\ell+k-2} \left\|W^{[k]}\right\|_{2\to\infty} \left(\prod_{j=1}^{k-1} \|W^{[j]}\|\right) \left\|x - x'\right\|_2, \tag{4.29}$$

$$\|S_{k_L}^{[\ell]}(x)\|_F \leq \left\|g_L^{(1)}\right\|_{L^\infty(\mathbb{R})} \kappa^{L-\ell} \left\|w_{k_L}^{[L]}\right\|_2 \prod_{k=\ell+1}^{L-1} \|W^{[k]}\|. \tag{4.30}$$

Remark: Here, the norm $\|\cdot\|_{2\to\infty}$ appearing in (4.29) *refers to $\|W^{[k]}\|_{2\to\infty} := \sup_{\|x\|_2=1} \|(W^{[k]})^\top x\|_\infty$.*

Proof of Lemma 4.2. For the statement (4.28) note that $\|V^{[1]}(x)\| = \|W^{[1]}\|$ by definition. For $\ell \geq 2$, by matrix norm submultiplicativity applied to (4.18), we have

$$\begin{aligned}\|V^{[\ell]}(x)\| &= \left\|W^{[1]} \prod_{k=2}^{\ell} \operatorname{diag}(g_{k-1}'(z^{[k-1]}(x))W^{[k]}\right\| \\ &\leq \|W^{[1]}\| \left(\prod_{k=1}^{\ell-1} \|\operatorname{diag}(g_k'(z^{[k]}(x))\|\right) \left(\prod_{k=2}^{\ell} \|W^{[k]}\|\right) \leq \kappa^{\ell-1} \prod_{k=1}^{\ell} \|W^{[k]}\|.\end{aligned}$$

A consequence of the Lipschitz continuity (cf. **(DNM1)**) of $g_\ell, g_\ell^{(1)}$ is that

$$\left\|y^{[\ell]}(x) - y^{[\ell]}(x')\right\|_2 \leq \kappa \left\|z^{[\ell]}(x) - z^{[\ell]}(x')\right\|_2.$$

Notably, the Lipschitz continuity of the activation can be transferred to all pre-activations of the network:

$$\|g_\ell(z^{[\ell]}(x)) - g_\ell(z^{[\ell]}(x'))\|_2 \leq \kappa \left\|W^{[\ell]}\right\| \|y^{[\ell-1]}(x) - y^{[\ell-1]}(x')\| \leq \kappa^{\ell-1} \left(\prod_{k=1}^{\ell} \|W^{[k]}\|\right) \left\|x - x'\right\|_2,$$

where the last inequality follows from an iterative application of the Lipschitz continuity. By our assumption, $g^{(1)}$ is also Lipschitz continuous with Lipschitz constant κ. Hence, by the same iterative argument, we have

$$\begin{aligned}&\|\operatorname{diag}(g_\ell^{(1)}(z^{[\ell]}(x))) - \operatorname{diag}(g_\ell^{(1)}(z^{[\ell]}(x')))\| = \|g_\ell'(z^{[\ell]}(x)) - g_\ell'(z^{[\ell]}(x))\|_\infty \\ &\quad\leq \kappa \left\|W^{[\ell]}\right\|_{2\to\infty} \|y^{[\ell-1]}(x) - y^{[\ell-1]}(x')\|_2 \leq \kappa^\ell \left\|W^{[\ell]}\right\|_{2\to\infty} \left(\prod_{k=1}^{\ell-1} \|W^{[k]}\|\right) \left\|x - x'\right\|_2.\end{aligned} \tag{4.31}$$

The Lipschitz continuity of entangled weights $V^{[\ell]}(x)$ as a function of x for $\ell = 1$ is clear, since $\|V^{[1]}(x) - V^{[1]}(x')\| = \|W_1 - W_1\| = 0$. For $\ell \geq 2$, we first use the triangle inequality, norm submultiplicativity, and the relation $V^{[\ell]}(x) = V^{[\ell-1]}(x)\,\mathrm{diag}(g_\ell^{(1)}(z^{[\ell-1]}(x)))W_\ell$ to get

$$\begin{aligned}
\|V^{[\ell]}(x) - V^{[\ell]}(x')\| &= \left\|V^{[\ell-1]}(x)\,\mathrm{diag}(g_\ell^{(1)}(z^{[\ell-1]}(x)))W^{[\ell]} - V^{[\ell-1]}(x')\,\mathrm{diag}(g_\ell^{(1)}(z^{[\ell-1]}(x')))W^{[\ell]}\right\| \\
&\leq \left\|W^{[\ell]}\right\| \left\|V^{[\ell-1]}(x) - V^{[\ell-1]}(x')\right\| \left\|g_\ell^{(1)}(z^{[\ell-1]}(x))\right\|_\infty \\
&\quad + \left\|W^{[\ell]}\right\| \left\|V^{[\ell-1]}(x')\right\| \left\|g_\ell^{(1)}(z^{[\ell-1]}(x)) - g_\ell^{(1)}(z^{[\ell-1]}(x'))\right\|_\infty \\
&\leq \kappa\|W^{[\ell]}\| \left(\|V^{[\ell-1]}(x) - V^{[\ell-1]}(x')\| + \kappa^{\ell-3} \left(\prod_{k=1}^{\ell-1} \|W^{[k]}\|\right) \|g_\ell^{(1)}(z^{[\ell-1]}(x)) - g_\ell^{(1)}(z^{[\ell-1]}(x'))\|_\infty \right),
\end{aligned}$$

where we used (4.28) to bound $\|V_{\ell-1}(x')\|$ in the last step. By repeating the expansion for $\|V^{[\ell-1]}(x) - V^{[\ell-1]}(x')\|$ until we reach the last layer where $\|V^{[1]}(x) - V^{[1]}(x')\| = 0$. Gathering all terms we obtain

$$\|V^{[\ell]}(x) - V^{[\ell]}(x')\| \leq \kappa^{\ell-2} \left(\prod_{k=1}^{\ell} \|W^{[k]}\|\right) \sum_{k=1}^{\ell-1} \|\,\mathrm{diag}(g_k'(z^{[k]}(x))) - \mathrm{diag}(g_k^{(1)}(z^{[k]}(x')))\|,$$

and the result (4.29) follows by using (4.31). What remains is the bound on the Frobenius norm of the diagonal matrix $S_{k_L}^{[\ell]}(x)$ with diagonal elements given by the partial derivatives $(S_{k_L}^{[\ell]})_{k_\ell} = \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g^{(2)}(z_{k_\ell}^{[\ell]}(x))$ (cf. (4.27)). Using norm submultiplicativity we get

$$\left\|S_{k_L}^{[\ell]}(x)\right\|_F \leq \left\|\mathrm{diag}(g_\ell''(z^{[\ell]}(x))\right\| \left\|\mathrm{diag}\left(\frac{df_{k_L}(x)}{dy^{[\ell]}(x)}\right)\right\|_F = \kappa \left\|\frac{df_{k_L}(x)}{dy^{[\ell]}(x)}\right\|_2.$$

Now note that

$$\left\|\frac{df_{k_L}(x)}{dy^{[\ell]}(x)}\right\|_2 = \left\|g_L^{(1)}(z_{k_L}^{[L]}(x))\frac{dz_{k_L}^{[L]}(x)}{dy^{[\ell]}(x)}\right\|_2 \leq \kappa \left\|\frac{dz_{k_L}^{[L]}(x)}{dy^{[\ell]}(x)}\right\|_2.$$

We can interpret $y^{[\ell]}(x)$ as the input to the subnetwork of the remaining $L - \ell$ layers, such that the gradient of $z_{k_L}^{[L]}$ w.r.t. the post-activation of an inner layer is essentially the same as an entangled weight of a slightly smaller network. This allows us to reuse the computations of the proof of Lemma 4.1, in particular (4.17), to find the following expression for $\frac{dz_{k_L}^{[L]}(x)}{dy^{[\ell]}(x)}$

$$\frac{dz_{k_L}^{[L]}(x)}{dy^{[\ell]}(x)} = \left(\prod_{j=\ell+1}^{L} W^{[j]}\,\mathrm{diag}\left(g_j'(z^{[j]}(x))\right)\right) w_{k_L}^{[L]} \leq \kappa^{L-\ell} \prod_{j=\ell+1}^{L-1} \|W^{[j]}\|. \tag{4.32}$$

Combining it with the previous bound yields the desired result. □

4.1.3 Summary: Main results

The present chapter considers the identification of deep neural networks from Hessian information inspired by [54] (see also Section 1.5). Our recovery pipeline covers the results of [52, 50], and we follow the same decoupling strategy that was used throughout Chapter 3 and [54]. This decoupling can be broken down into two high-level stages:

Stage 1: Construct a matrix space from network Hessians approximating a space spanned by rank-one matrices which encode weight information. Assuming the Hessians allow for such a construction (cf. **(DNM3)** and **(SNM3)**), this essentially reduces the weight recovery to the problem studied in Chapter 2, i.e., the recovery of individual rank-one matrices using methods such as Algorithm 2.2 (SPM) or Algorithm 2.1.

Stage 2: Formulate a loss-free reparametrization of the original network depending only on the remaining unknown parameters which could not be recovered during the first stage. Learn the unknown parameters of the reparametrization by fitting the reparametrized model onto the neural network via empirical risk minimization.

The recovery of shallow neural networks in Chapter 3 followed this procedure. Additionally, we employed further heuristics to compute the signs of the weights and to gain a first estimate of the shifts between the two stages (cf. Section 3.3). In the following sections, we will essentially follow the same recipe. However, the first stage is concerned with recovering the entangled weights (for one fixed input cf. Definition 4.2) instead of the exact network weights.

Stage 1: Entangled weight identification. Consider a network $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$. First, in Section 4.2.1, we rely on Hessian matrices sampled from random inputs to construct an approximation

$$\widehat{\mathcal{W}} \approx \mathcal{W} := \operatorname{span}\{v \otimes v | v \in \mathcal{V}(f, x^*)\}, \tag{4.33}$$

where $x^* \in \mathbb{R}^D$ is a fixed input vector that depends on the distribution of the Hessian locations. This construction's primary technical challenge is caused by the fact that entangled weights in (4.20) vary for different Hessian locations. To compensate, we sample Hessian locations from a sub-Gaussian distribution which concentrates around x^*. This leads to a delicate tradeoff, where concentrated Hessians lead to improved stability of the entangled weights but reduce the coverage of the space $\operatorname{span}\{v \otimes v | v \in \mathcal{V}(f, x^*)\}$. This dynamic is discussed in detail within Section 4.2. Our main result regarding the approximation in (4.33) is Theorem 4.1, which states that the approximation error adheres to the upper bound

$$\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F\to F} \leq 2\frac{\hat{C}_\epsilon + \bar{C}\sqrt{D}\left\|X - x^*\right\|_{\psi_2}}{\sqrt{\frac{\alpha}{2} - \hat{C}_\epsilon^2}} \quad \text{w.h.p.,} \tag{4.34}$$

for networks satisfying the conditions summarized in Section 4.1.1 and sufficiently many Hessian locations. Here, $\hat{C}_\epsilon$ is a factor depending on the accuracy of the numerical differentiation, which relates to **(G5.3)**. The factor $\bar{C}$ represents the complexity of the network, $\alpha > 0$ corresponds to **(DNM3)** and $\|X - x^*\|_{\psi_2}$ is the sub-Gaussian norm associated with the Hessian locations. We generally assume that $\hat{C}_\epsilon$ can be made arbitrarily small using a suitable numerical differentiation method. In this case, the bound (4.34) scales like

$$\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F\to F} \lesssim \bar{C}\sqrt{D/\alpha}\left\|X - x^*\right\|_{\psi_2} \quad \text{w.h.p.} \tag{4.35}$$

Note that this bound will not further improve with higher numerical accuracy since it is independent of $\hat{C}_\epsilon$. The bound (4.35) suggests that the subspace approximation error can only be controlled by reducing the variance of the Hessian locations, which leads to a decrease of the sub-Gaussian norm $\|X - x^*\|_{\psi_2}$. This is the main difference to our results in Chapter 3, where the subspace approximation error depended exclusively on the numerical accuracy ϵ. More precisely, in Lemma 3.1 we had

$$\left\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\right\|_{F\to F} \lesssim \sqrt{m/\alpha} \cdot \epsilon \quad \text{w.h.p.,}$$

which can be made arbitrarily small by decreasing ϵ. Let us mention that the dynamics of (4.35) are further complicated by the fact that one has to assume that $\alpha > 0$ also depends on the distribution of the Hessian locations in a non-trivial way. This relates to the abovementioned tradeoff and is further discussed in Section 4.2.1.

Once the space $\mathcal{W}$, spanned by outer products of entangled weights, is identified (approximately), the recovery of individual entangled weights is then tackled by Algorithm 2.2. The overall recovery of entangled weights is summarized in Algorithm 4.1. Assuming suitable incoherence of the entangled weights, the recovery of individual entangled weights is then justified by the theory developed throughout Section 2.4. Furthermore, we provide extensive numerical analysis of Algorithm 4.1 in Section 4.2.3 and Section 4.4.

Stage 2: Network completion This second part of the chapter explains how to leverage the information contained in the entangled weights as part of the overall reconstruction pipeline, which is done in two steps. First, we need to order the output of the weight recovery to reconstruct the entangled weight matrices. This requires an assignment of the vectors in $\mathcal{U}_{\mathrm{SPM}}$ (cf. Algorithm 4.1) to their corresponding layers. In Section 4.3.1, we provide heuristics that are designed to distinguish entangled weight approximations between the first layer, the inner hidden layers, and the last layer (see Algorithm 4.2). Currently, this layer assignment is *limited to networks with three layers* ($L \leq 3$). We provide an extensive numerical analysis of Algorithm 4.2 in Section 4.4.

In Section 4.3.2, we assume that access to all entangled weight matrices up to a rescaling and permutation of the columns (originating from Algorithm 4.1), i.e., we have access to

$$\tilde{V}^{[\ell]} := \tilde{V}^{[\ell]}(x^*) = V^{[\ell]}(x^*)\pi_\ell S_\ell, \quad \text{for all } \ell \in [L],$$

where $\pi_1, \dots, \pi_L$ are permutation matrices. Based on these matrices, a network reparametrization (cf. Definition 4.4) is constructed from the weights $\tilde{W}^{[\ell]} = (\tilde{V}^{[\ell-1]})^\dagger \tilde{V}^{[\ell]}$ that only depends on the unknown shifts and diagonal matrices which model the unknown scales. We prove in Proposition 4.3 that, under suitable conditions, this reparametrization is equivalent to the original network f for the right set of shifts and diagonal matrices. Similar to Section 3.4 in Chapter 3, we formulate an approach applying empirical risk minimization to find these scaling and shift parameters via gradient-based methods. Furthermore, we show in Proposition 4.4 that antisymmetric activation functions (e.g., sigmoidal activations) give rise to multiple global minima. While we do not conduct a rigorous convergence analysis as in Chapter 3, we expect similar performance as in the reparametrization applied to shallow neural networks. We support this claim with numerical evidence in Section 4.4.

4.2 Entangled weight identification

By the theory of Section 4.1.2, the Hessians of twice differentiable networks can be decomposed into symmetric rank-one tensors given by the entangled weight functions. We briefly sketched how for a wide range of pyramidal-shaped networks, entangled weights encode large parts of the network parameters, which only leaves a few unknowns to fully recover this kind of network. This will be made precise in Section 4.3. In this section, we describe how the exposure of the entangled weights via differentiating the network makes them recoverable by following similar steps already successfully applied in our reconstruction pipeline of Chapter 3. More precisely, we aim to reduce the recovery of entangled weights to the problem studied in Chapter 2, which is concerned with identifying rank-one basis matrices within their span under

Algorithm 4.1: Entangled weight recovery

Input: Deep neural network f with m^* non-linear neurons and m_L outputs, input distribution μ_X with mean x^*, number of Hessians N_h

Draw N_h random samples $x_1, \dots, x_{N_h} \sim_{\text{i.i.d.}} \mu_X$;

Compute Hessian approximations $\Delta^2 f_{k_L}(x_i)$ for all $i \in [N_h], k_L \in [m_L]$;

Compute the m^*-th left singular subspace of the Hessian approximations

$$\widehat{\mathcal{W}} = \text{span}_{m^*} \left\{ \Delta^2 f_1(x_1), \dots, \Delta^2 f_1(x_{N_h}), \dots, \Delta^2 f_L(x_1), \dots \Delta^2 f_L(x_{N_h}) \right\},$$

and denote by $P_{\widehat{\mathcal{W}}}$ the orthogonal projection onto $\widehat{\mathcal{W}}$ (see Section 1.1.1);

Compute the set of approximated entangled weights $\mathcal{U}_{\text{SPM}}$ using Algorithm 2.2 (SPM) with input $(P_{\widehat{\mathcal{W}}}, m^*, \beta = 0.5)$;

Set $\mathcal{U}_L = \emptyset$

for $k_L = 1, \dots, m_L$ **do**

  Compute $\Delta[f_{k_L}](x^*)$

  $\mathcal{U}_L \leftarrow \mathcal{U}_L \cup \{\Delta[f_{k_L}](x^*) / \left\|\Delta[f_{k_L}](x^*)\right\|_2\}$

end

Output: $\mathcal{U} = \mathcal{U}_{\text{SPM}} \cup \mathcal{U}_L$

perturbations. The recovery procedure has been summarized in Algorithm 4.1, which we will discuss in the following. The main theoretical result is centered around the approximation of the matrix space defined as follows:

Definition 4.3. *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell \in [L]}, (g_\ell)_{\ell \in [L]})$ adhering to **(DNM2)** with activations $(g_\ell)_{\ell \in [L]}$ that are differentiable and a fixed input $x^* \in \mathbb{R}^D$. We denote by $\mathcal{W}(f, x^*)$ the space of symmetric matrices spanned by all entangled weights of f w.r.t. x^*, which appear in the Hessian decomposition, more precisely*

$$\mathcal{W}(f, x^*) := \text{span}\{v_{k_\ell}^{[\ell]}(x^*) \otimes v_{k_\ell}^{[\ell]}(x^*) \mid \ell \in [L^*], k_\ell \in [m_\ell]\},$$

where $L^* = L$ if the output neurons are sufficiently non-linear, or $L^* = L - 1$ when the entangled weights of the outputs do not occur in the Hessian decomposition (cf. last section).

Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell \in [L]}, (g_\ell)_{\ell \in [L]})$ with entangled weights $\mathcal{V}(f, x^*)$ for some fixed input $x^* \in \mathbb{R}^D$. The first part of Algorithm 4.1 computes an approximation to the space $\mathcal{W}(f, x^*)$ from Hessian approximations. As in Chapter 3, we consider Hessian approximations that are anchored at locations $x_1, \dots, x_{N_h} \in \mathbb{R}^D$ drawn randomly from a distribution μ_X with mean x^*. From these inputs we compute Hessian approximations

$$\Delta^2 f_1(x_1), \dots, \Delta^2 f_1(x_{N_h}), \dots, \Delta^2 f_L(x_1), \dots \Delta^2 f_L(x_{N_h}), \tag{4.36}$$

of every output neuron.

Remark 4.2. *Notably, in the theoretical part, we do not specify the exact numerical scheme used and only rely on the assumption **(G5.3)**. An explicit example of such an approximation scheme is given in the numerical analysis of Section 4.4, where we use a simple central finite difference scheme. For sigmoidal networks considered in our experiments, the error caused by the numerical approximation of Hessians has only a very marginal effect on our overall reconstruction pipeline.*

Given the matrices in (4.36), an approximation to $\mathcal{W}(f, x^*)$ is computed as the m^*-th singular subspace (cf. Definition 1.1) of the Hessian approximations, where m^* is the number of

non-linear neurons (cf. **(DNM2)**). More precisely, we consider

$$\widehat{\mathcal{W}} := \operatorname{span}_{m^*} \left\{ \Delta^2 f_1(x_1), \dots, \Delta^2 f_1(x_{N_h}), \dots, \Delta^2 f_L(x_1), \dots \Delta^2 f_L(x_{N_h}) \right\} \approx \mathcal{W}(f, x^*),$$

and we denote by $P_{\mathcal{W}}, P_{\widehat{\mathcal{W}}}$ the orthogonal projection onto $\mathcal{W}, \widehat{\mathcal{W}}$, respectively. Under condition **(DNM2)**, m^* either coincides with the overall number of neurons $m = \sum_{\ell=1}^{L} m_\ell$ or the number of hidden neurons $m - m_L$. Aside from the presence of several output neurons, this approach is indeed identical to the subspace approximation in Chapter 3 and mirrors the ideas outlined in Section 1.5.

Comparison to Section 3.2. For shallow neural networks with a single output and $\mu_X = \mathcal{N}(0, \mathrm{Id}_D)$, the subspace approximation error was estimated by

$$\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F} = \mathcal{O}(\epsilon \sqrt{m/\alpha}) \quad \text{w.h.p.},$$

where ϵ represents the accuracy of the numerical approximation and $\alpha > 0$ is given by **(SNM3)** (cf. Lemma 3.1). Hence, for shallow neural networks, the accuracy of the subspace approximation could largely be controlled via the accuracy of the numerical differentiation. In the present case of more general deep neural networks, the subspace error is influenced by an additional factor caused by the dependency of the entangled weights on the input. For instance, consider the Hessian approximation $\Delta^2 f_1(x_1)$ of the first output neuron in $x_1 \in \mathbb{R}^D$. If we neglect any errors caused by numerical approximations, i.e., $\Delta^2 f_1(x_1) = \nabla^2 f_1(x_1)$, then, by Proposition 4.1, we know that $\Delta^2 f_1(x_1) \in \mathcal{W}(f, x_1)$. However, that does not imply $\Delta^2 f_1(x_1) \in \mathcal{W}(f, x^*)$ since in general $\mathcal{W}(f, x_1) \neq \mathcal{W}(f, x^*)$ for $x_1 \neq x^*$. We address this problem by sampling the inputs $x_1, \dots, x_{N_h} \sim_{\text{i.i.d.}} \mu_X$ in a concentrated manner around the mean $x^* = \mathbb{E}[\mu_X]$, which is modeled by considering only sub-Gaussian distributions μ_X with small variance. Theorem 4.1 provides a bound on the resulting approximation error for deep neural networks and makes the argument above rigorous by showing that $\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F \to F}$ scales with the sub-Gaussian norm of μ_X.

Remark 4.3. *To prevent all inputs sampled from μ_X from collapsing into one point, we need to limit the degree to which the variance of μ_X can be restricted. Therefore, a residual error will be caused by the mismatch of the space $\mathcal{W}(f, x^*)$ and the spaces $\mathcal{W}(f, x_i)$, which remains independent of the numerical differentiation accuracy. We observe numerically that this error is the most dominant factor in the subspace approximation. The exact error bound on $\|\mathcal{W}(f, x^*) - \widehat{\mathcal{W}}\|_{F \to F}$ will be provided together with a more detailed discussion separately in Section 4.2.1.*

Retrieval of individual entangled weights. Assuming that the approximation $\widehat{\mathcal{W}} \approx \mathcal{W}(f, x^*)$ is sufficiently accurate, let $\mathcal{W} = \mathcal{W}(f, x^*)$. Borrowing from the results in Chapter 2, which closely follows the analysis within [50] and covers parts of the results contained in [77, 76], we know that the rank-one spanning elements of $\mathcal{W}$ are identifiable as global maximizers of the non-convex program

$$\max_{\|u\|_2 = 1} \Phi_{\mathcal{W}}(u) := \|P_{\mathcal{W}}(u \otimes u)\|_F^2 . \tag{4.37}$$

In Chapter 2, we saw that under fairly general incoherence conditions on the vectors $v_{k_\ell}^{[\ell]}(x^*)$, the matrices $v_{k_\ell}^{[\ell]}(x^*) \otimes v_{k_\ell}^{[\ell]}(x^*)$ are the only rank-one matrices within $\mathcal{W}$. Furthermore, we saw that this uniqueness is maintained even under slight perturbations of $\mathcal{W}$, such that an

approximate recovery of the entangled weight outer products remains possible from local maximizers of the perturbed objective

$$\max_{\|u\|_2=1} \Phi_{\widehat{\mathcal{W}}}(u) := \left\|P_{\widehat{\mathcal{W}}}(u \otimes u)\right\|_F^2 \tag{4.38}$$

that belong to a certain level-set of $\Phi_{\widehat{\mathcal{W}}}(\cdot)$. As pointed out in Chapter 2, a simple projected gradient ascent algorithm introduced as subspace power method (SPM) by [77], which is started from a random initialization $u_0 \sim \mathrm{Unif}(\mathbb{S}^{D-1})$ and iterates

$$u_j = P_{\mathbb{S}^{D-1}}(u_{j-1} + 2\gamma P_{\widehat{\mathcal{W}}}((u_{j-1})^{\otimes 2})u_{j-1}) \tag{4.39}$$

until convergence, can be used to find the local maximizers of the program (4.38) (see also Algorithm 2.2 and the related discussion). Assuming the entangled weights $\mathcal{V}(f, x^*)$ are sufficiently incoherent, then based on our prior analysis and the results referenced in Chapter 2, we would expect that Algorithm 2.2 finds all non-spurious local maximizers of (4.38) yielding approximations to all entangled weights in $\mathcal{V}(f, x^*)$ up to sign with an accuracy that scales like $\mathcal{O}(\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F}^{1/2})$. See, for instance, Theorem 2.8 and Theorem 2.6. As this problem has been discussed extensively, we do not provide any further details on the recovery of the near-rank-one spanning elements from $\widehat{\mathcal{W}}$ and refer instead to Chapter 2.

4.2.1 Stabilizing the Hessian subspace

In this section, we discuss the approximation of the matrix space $\mathcal{W}(f, x^*)$ spanned by entangled weights of a network f at one specific point $x^* \in \mathbb{R}^D$ (cf. Definition 4.3). The method applied in [54] and Chapter 3 to recover weights of shallow networks was to combine Hessians $\nabla^2 f(x)$ sampled at different input locations $x_1, \ldots, x_{N_h}$ to reach a stable recovery of the space spanned by symmetric rank-one matrices

$$\mathrm{span}\{\nabla^2 f(x_1), \ldots, \nabla^2 f(x_{N_h})\} \approx \mathrm{span}\{w_1 \otimes w_1, \ldots, w_m \otimes w_m\}. \tag{4.40}$$

We already hinted at the fact that combining Hessians at different inputs seems problematic for deep networks since the underlying decomposition into entangled weights additionally depends on the Hessian location, i.e., for a network input $x \in \mathbb{R}^D$ we have

$$\nabla^2 f_{k_L}(x) = \sum_{\ell=1}^{L} \sum_{k_\ell=1}^{m_\ell} s_{k_\ell}^{[\ell]}(x) \cdot v_{k_\ell}^{[\ell]}(x) \otimes v_{k_\ell}^{[\ell]}(x) \in \mathcal{W}(f, x),$$

with coefficients

$$s_{k_L,k_\ell}^{[\ell]}(x) = \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x)).$$

As mentioned previously, for inputs $x, x' \in \mathbb{R}^D$ we have to assume that $\mathcal{W}(f, x) \neq \mathcal{W}(f, x')$ since we can not exclude the case $v_{k_\ell}^{[\ell]}(x) \otimes v_{k_\ell}^{[\ell]}(x) \neq v_{k_\ell}^{[\ell]}(x') \otimes v_{k_\ell}^{[\ell]}(x')$. The dependency of the entangled weights in one particular Hessian decomposition on the Hessian location poses the primary challenge in our analysis. As the main result of this section, we show that this deviation can be kept under control by sampling the Hessians in a concentrated manner. This stabilizes the entangled weight functions and therefore justifies reapplying the subspace approximation via singular subspaces to approximate a matrix space that is spanned by the symmetric rank-one tensors of a fixed set of entangled weights.

Theorem 4.1 (cf. [50, Theorem 4]). *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$ with activations $(g_\ell)_{\ell\in[L]}$ that fulfill **(DNM1)** with $\kappa > 0$ and **(DNM2)** with m^* non-linear neurons. Let μ_X be a sub-Gaussian distribution with mean $x^* = \mathbb{E}_{X\sim\mu_X}[X]$ such that the pair (f, μ_X) together fulfill **(DNM3)** with $\alpha > 0$. Furthermore, assume access to a numerical differentiation scheme as in **(G5.3)** with accuracy $\hat{C}_\epsilon$. Let $X \sim \mu_X$ and $X_1, \dots, X_{N_h}$ be independent copies of X. Denote $\mathcal{W} = \mathcal{W}(f, x^*)$ and*

$$\widehat{\mathcal{W}} = \operatorname{span}_{m*} \left\{ \Delta^2 f_1(X_1), \dots, \Delta^2 f_1(X_{N_h}), \dots, \Delta^2 f_{m_L}(X_1), \dots, \Delta^2 f_{m_L}(X_{N_h}) \right\}. \tag{4.41}$$

If $\alpha > \max\{2\bar{C}^2 D \|X - x^*\|_{\psi_2}^2, 2\hat{C}_\epsilon^2\}$, then there exists a universal constant $C > 0$ such that

$$\left\| P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}} \right\|_{F\to F} \leq 2 \frac{\hat{C}_\epsilon + \bar{C}\sqrt{D}\, \|X - x^*\|_{\psi_2}}{\sqrt{\frac{\alpha}{2} - \hat{C}_\epsilon^2}} \tag{4.42}$$

with probability at least

$$1 - 2\exp\left(-CN_h\right) - m\exp\left(-C \frac{N_h \alpha}{\kappa^{2L} \left(\prod_{\ell=1}^L \|W^{[\ell]}\| \right)^2 \left(\sum_{\ell=1}^L \kappa^{\ell-1} \prod_{k=1}^\ell \|W^{[k]}\| \right)^2} \right),$$

where $\bar{C}$ encodes the network complexity in terms of $\left(\|W^{[\ell]}\| \right)_{\ell\in[L]}$ and takes the form

$$\bar{C} := C\kappa^L \left(\prod_{k=1}^L \|W^{[k]}\| \right) \sum_{\ell=2}^L \left(\prod_{k=1}^\ell \|W^{[k]}\| \right) \sum_{k=1}^{\ell-1} \kappa^{\ell+k-2} \left\| W^{[k]} \right\|_{2\to\infty} \left(\prod_{j=1}^{k-1} \|W^{[k]}\| \right). \tag{4.43}$$

The proof of this statement is postponed to the end of this section. First, we want to provide some intuition about this result. Aside from α, the bound achieved in (4.42) can be decomposed into three components. First, an error $\hat{C}_\epsilon$ due to the numerical differentiation, which relates to **(G5.3)** and depends on the underlying numerical differentiation method. We assume this error can be made sufficiently small, which has already been discussed previously. Second, the factor $\bar{C}$ represents the complexity of the network in terms of the spectral norm of the weight matrices and Lipschitz constants of the activation. Note that $\|W\|_{2\to\infty}^{[\ell]} = 1$ for unit norm vectors, and $\|W^{[\ell]}\| \approx 1$ for sufficiently incoherent weights. Therefore, for sufficiently incoherent weights of unit-norm, we can consider $\bar{C}$ to be a constant that only depends on L and the activation functions.

Geometric interpretation of the bound (4.42). Let us provide some more context on the term $\|X - x^*\|_{\psi_2}$, which scales with the sub-Gaussian norm of the input distribution μ_X. Hence, assuming $\hat{C}_\epsilon$ is sufficiently small, the subspace error scales with the variance (i.e., sub-Gaussian norm) of the input distribution, and, consequently, the error can not be controlled solely by increasing the number of sampled Hessians. This can be explained as follows (cf. Figure 4.1): For any $x \neq x^*$, there is a mismatch of the Hessians $\nabla^2 f_{k_L}(x) \in \mathcal{W}(f, x), k_L \in [m_L]$ and the space $\mathcal{W} = \mathcal{W}(f, x^*) \neq \mathcal{W}(f, x)$ when x is drawn from μ_X (see also Lemma 4.3). As long as x remains close to x^*, this deviation can be controlled by the Lipschitz continuity of the networks' Hessian. For increasing $\|x - x^*\|_2$, the Hessians $\nabla^2 f_{k_L}(x)$ will fan out away from the ideal space $\mathcal{W}$. One can think of $\widehat{\mathcal{W}}$ as the center of mass of the Hessian approximations $\Delta^2 f_1(X_1), \dots, \Delta^2 f_{m_L}(X_{N_h})$ (cf. (4.41)). Since we have no reason to believe that these Hessian approximations concentrate symmetrically around $\mathcal{W}$, the estimated space $\widehat{\mathcal{W}}$ can only be guaranteed to lie within a cone around the original space $\mathcal{W}$. The only way to control the

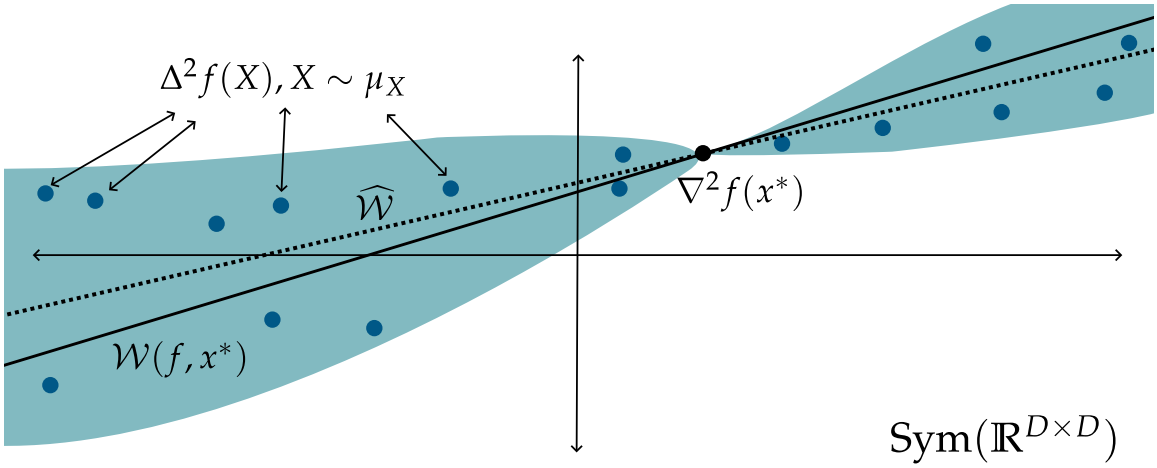

Figure 4.1: Geometric interpretation of the mismatch between $\mathcal{W}$ and $\widehat{\mathcal{W}}$ caused by the differences in the Hessian decompositions. The solid black line depicts the ideal space $\mathcal{W}$, the light blue region indicates the area in which Hessian approximations can lie when they are sampled at random, and the dashed line, symbolizing $\widehat{\mathcal{W}}$, indicates the center of mass of the sampled Hessian approximations.

impact of this phenomenon on the subspace approximation is to decrease the variance of the input distribution.

Let us stress that there is a limitation on how small we can make $\|X - x^*\|_{\psi_2}$ since α might additionally depend on the input distribution in a highly non-trivial way. Leaving other factors aside, the bound on the subspace error (4.42) scales with $\mathcal{O}(\|X - x^*\|_{\psi_2} / \sqrt{\alpha})$ where α comes from the learnability condition **(DNM3)**, more precisely, α is defined via the spectrum of the second moments of Hessians:

$$\alpha := \sigma_{m^*}\left(\frac{1}{m_L}\sum_{k_L=1}^{m_L} \mathbb{E}_{X\sim\mu_X}\left[\mathrm{vec}(\nabla^2 f_{k_L}(X)) \otimes \mathrm{vec}(\nabla^2 f_{k_L}(X))\right]\right) > 0.$$

It is clear that the quantity α is affected by the choice of distribution μ_X. This relationship between α and μ_X was already discussed in Section 4.1.1 and Chapter 3. For shallow neural networks, we proved in Theorem 3.5 that under fairly mild conditions and for $\mu_X = \mathcal{N}(0, \mathrm{Id}_D)$, the factor α can be regarded as a constant independent of m, D (see the related discussion in Section 3.1.1 and Theorem 3.5). For deep neural networks, however, one would expect that α scales with the sub-Gaussian norm of μ_X. This, in turn, implies that the right-hand side of (4.42) cannot be made arbitrarily small within the framework of our analysis, regardless of the numerical accuracy $\hat{C}_\epsilon$ or the number of Hessians used. Our numerical experiments do support this argument, i.e., we observe that we do indeed need a strong concentration of μ_X to reach a stable approximation of the subspace $\mathcal{W}$, but the quantity $\|X - x^*\|_{\psi_2}$ cannot be decreased without limits (cf. Section 4.2.3). However, we also observe (empirically) a certain stabilization of α due to the presence of multiple output neurons. We will describe this dynamic in the following.

Stabilizing effect of multiple output neurons on the Hessian span. The preceding discussion revealed how the dependence between α, μ_X makes the bound (4.42) delicate to interpret since α cannot be regarded as a constant independent of μ_X. Nevertheless, empirically we do observe a consistent approximation of the space $\mathcal{W}$ by Algorithm 4.1 (cf. Section 4.2.3 and Section 4.4). The overall good performance of the recovery begs the question of whether the discussion

above is too pessimistic or if there are other factors at play. Notably, we observe a strong positive correlation between the accuracy of the subspace identification and the number of output neurons (see, for instance, Figure 4.4 or Table 4.2). Intuitively, this can be explained as follows: As long as every output neuron

$$f_{k_L}(\cdot) = y_{k_L}^{[L]}(\cdot) \quad \text{for } k_L \in [m_L],$$

computes a different function (i.e., the columns of $W^{[L]}$ are sufficiently incoherent), we get m_L different measurements of the subnetwork constructed by the $L-1$ first layers. One can directly see this by considering the decomposition of the Hessians into entangled weights. We learned in Proposition 4.1 that the decomposition per output is given by

$$\nabla^2 f_{k_L}(x) = g_L''(z_{k_L}^{[L]}(x)) v_{k_L}^{[L]}(x)^{\otimes 2} + \sum_{\ell=1}^{L-1} \sum_{k_\ell=1}^{m_\ell} \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x)) v_{k_\ell}^{[\ell]}(x)^{\otimes 2},$$

where k_L is the index of the respective output neuron. Now, assume we draw the input x from a distribution that highly concentrates around its mean x^*, then the part of the Hessian decomposition corresponding to the last $L-1$ layers would reflect that concentration and

$$\sum_{\ell=1}^{L-1} \sum_{k_\ell=1}^{m_\ell} \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x)) v_{k_\ell}^{[\ell]}(x)^{\otimes 2} \approx \sum_{\ell=1}^{L-1} \sum_{k_\ell=1}^{m_\ell} \frac{\partial f_{k_L}(x^*)}{\partial y_{k_\ell}^{[\ell]}(x^*)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x^*)) v_{k_\ell}^{[\ell]}(x^*)^{\otimes 2}.$$

We observe the tradeoff described above, where high concentration leads to a stabilization of $v_{k_\ell}^{[\ell]}(x^*)^{\otimes 2}$ but decreases the overall variability represented by the coefficients

$$s_{k_L,k_\ell}^{[\ell]}(x) = \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x)) \approx \frac{\partial f_{k_L}(x^*)}{\partial y_{k_\ell}^{[\ell]}(x^*)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x^*)) = s_{k_L,k_\ell}^{[\ell]}(x^*).$$

The key observation is that the influence of a particular neuron on any particular output neuron, i.e., the factor $\partial f_{k_L}(x^*)/\partial y_{k_\ell}^{[\ell]}(x^*)$, can still cause a significant change in the coefficients $s_{k_L,k_\ell}^{[\ell]}(x^*)$ w.r.t. k_L, k_ℓ. This means one can expect at least m_L different linear combinations of the entangled weight matrices $v_{k_\ell}^{[\ell]}(x^*)^{\otimes 2}$, for $\ell \in [L-1], k_\ell \in [m_\ell]$. This effect seems to compensate for many of the negative effects caused by considering high concentrating input distributions for the subspace approximations for generic networks. In fact, we can see this numerically when measuring $\|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F}$ in Figure 4.4. To clarify this effect, we can consider another hypothetical border case. Assume $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$ with linear output activation g_L. If the number of output neurons exceeds the number of remaining neurons $m_L \geq m_1 + \cdots + m_{L-1}$ and there exists a Hessian location x^* such that $\{\nabla^2 f_{k_L}(x^*) | k_L \in [m_L]\}$ contains at least $m_1 + \cdots + m_{L-1}$ independent matrices, then, using the notation from Definition 4.2, we have

$$\mathcal{W}(f, x^*) = \operatorname{span}\{v \otimes v | v \in \mathcal{V}(f, x^*) \setminus \mathcal{V}_L(f, x^*)\} = \operatorname{span}\left\{\nabla^2 f_{k_L}(x^*) | k_L \in [m_L]\right\}, \tag{4.44}$$

i.e., the span of the Hessians w.r.t. all output neurons for *one single input* contains all entangled weight matrices corresponding to the first $L-1$ layers. The statement follows by the fact that $\nabla^2 f_{k_L}(x^*)$ must lie within $\operatorname{span}\{v \otimes v | v \in \mathcal{V}(f, x^*) \setminus \mathcal{V}_L(f, x^*)\}$ due to Proposition 4.1 which implies (4.44) by a dimensionality argument. Of course, it might not be realistic to have that many output neurons, and further conditions would need to be required to fully develop a rigorous theoretical statement. But it underlines that more outputs potentially make the subspace stabilization more robust. Also, note that this does not go against the type of

network models we consider. As pointed out in Section 4.1.2, we require a pyramidal network architecture only up to the last layer to recover the original weights from the entangled weights, but we allow an arbitrary number of output neurons. Networks with a pyramidal shape are often used in practice for classification tasks. In these models, it is common to have one output neuron per class, and in many classification tasks, the number of classes is large. In summary, we believe that a large number of output neurons benefits us as long as the output weights exhibit some incoherence. This phenomenon is not directly visible from Theorem 4.1 because it's most likely already incorporated in our assumptions (by an increase of α).

Sample complexity Let us conclude the discussion of Theorem 4.1 by stating the expected sample complexity. Algorithm 4.1 only relies on network evaluation for the subspace approximation step since Algorithm 2.2 does not require any further evaluations of the network. If we assume that finite difference schemas approximate Hessians sufficiently well, then we need $\mathcal{O}(D^2)$ network evaluations to compute the Hessian at one output neuron. However, note that the same set of network inputs will simultaneously compute the Hessians of all remaining output neurons. If we assume that the network complexity $\bar{C}$ is bounded by a constant depending only on L, κ, then the overall number of Hessians needed in Theorem 4.1 is $N_h = \mathcal{O}(m/\alpha)$, where $m = m_1, \dots, m_L$. Therefore, we expect an average sample complexity that scales like $\mathcal{O}(mD^2/(m_L\alpha))$.

4.2.2 Proof of Theorem 4.1

Before proving Theorem 4.1, we establish an intermediate result. Note that by constructing $\widehat{\mathcal{W}}$ as the singular subspace associated with approximations of the Hessians

$$\widehat{\mathcal{W}} = \operatorname{span}_{m*} \left\{ \Delta^2 f_1(X_1), \dots, \Delta^2 f_1(X_{N_h}), \dots, \Delta^2 f_{m_L}(X_1), \dots, \Delta^2 f_{m_L}(X_{N_h}) \right\},$$

we can in general only hope to recover a part $\bar{\mathcal{W}} \subset \mathcal{W} = \mathcal{W}(f, x^*)$ of the target space where

$$\bar{\mathcal{W}} = \operatorname{span} \left\{ P_{\mathcal{W}} \nabla^2 f_1(x_1), \dots, P_{\mathcal{W}} \nabla^2 f_1(x_{N_h}), \dots, P_{\mathcal{W}} \nabla^2 f_L(x_1), \dots P_{\mathcal{W}} \nabla^2 f_L(x_{N_h}) \right\}. \tag{4.45}$$

The following proposition establishes to what degree $\bar{\mathcal{W}}$ can be approximated by a singular subspace of approximated Hessians from which the statement in Theorem 4.1 is merely an extension that follows by using the additional lower bound on α provided in Theorem 4.1 since $\alpha > 0$ guarantees $\bar{\mathcal{W}} = \mathcal{W}$ for sufficiently many Hessians.

Proposition 4.2 (cf. [50, Proposition 5]). *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell \in [L]}, (g_\ell)_{\ell \in [L]})$ with activations $(g_\ell)_{\ell \in [L]}$ that fulfill **(DNM1)-(DNM2)** with $\kappa > 0$ and m^* non-linear neurons. Let $X \sim \mu_X$ with mean $\mathbb{E}[X] = x^*$ be sub-Gaussian and let $X_1, \dots, X_{N_h}$ be independent copies of X. Furthermore, assume access to a numerical differentiation scheme as in **(G5.3)** with accuracy $\hat{C}_\epsilon$. Let $\bar{\mathcal{W}} \subseteq \mathcal{W}(f, x^*)$ be the space described above constructed from the inputs $X_1, \dots, X_{N_h}$ with $\bar{m} := \dim(\bar{\mathcal{W}})$ being its dimension. Assume*

$$\bar{\alpha} := \sigma_{\bar{m}} \left(\frac{1}{m_L} \sum_{k_L=1}^{m_L} \int \operatorname{vec}(\nabla^2 f_{k_L}(X)) \otimes \operatorname{vec}(\nabla^2 f_{k_L}(X)) \mathrm{d}\mu_X \right) > 2\hat{C}_\epsilon^2. \tag{4.46}$$

Then the subspace

$$\widehat{\mathcal{W}}_{\bar{m}} = \operatorname{span}_{\bar{m}} \left\{ \Delta^2 f_1(X_1), \dots, \Delta^2 f_1(X_{N_h}), \dots, \Delta^2 f_{m_L}(X_1), \dots, \Delta^2 f_{m_L}(X_{N_h}) \right\}$$

satisfies

$$\| P_{\bar{\mathcal{W}}} - P_{\widehat{\mathcal{W}}_{\bar{m}}} \|_{F \to F} \leq 2 \frac{\hat{C}_\epsilon + \bar{C}\sqrt{D} \, \| X - x^* \|_{\psi_2}}{\sqrt{\frac{\bar{\alpha}}{2} - \hat{C}_\epsilon^2}}$$

with probability at least

$$1 - 2\exp\left(-CN_h\right) - \bar{m}\exp\left(-C\frac{N_h\bar{\alpha}}{m_L\kappa^{2L}\left(\prod_{\ell=1}^{L}\|W^{[\ell]}\|\right)^2\left(\sum_{\ell=1}^{L}\kappa^{\ell-1}\prod_{k=1}^{\ell}\|W^{[k]}\|\right)^2}\right).$$

The constants $C, \hat{C}_\epsilon, \bar{C}$ are as in Theorem 4.1.

The proof of Proposition 4.2 relies on two auxiliary statements and is postponed to the end of the section. Recall that in Section 1.1.1, we generalize the concept of singular subspaces to matrices by considering their vectorizations. The error induced into these singular subspaces by perturbing the underlying elements can be bound by a Wedin bound previously stated in Proposition 1.1. Let us first introduce the auxiliary random matrices

$$M := (M_1|\dots|M_{m_L}) \in \mathbb{R}^{D^2\times N_h m_L}, \tag{4.47}$$

$$\widehat{M} := \left(\widehat{M}_1|\dots|\widehat{M}_{m_L}\right) \in \mathbb{R}^{D^2\times N_h m_L}, \tag{4.48}$$

$$\bar{M} := (\bar{M}_1|\dots|\bar{M}_{m_L}), \tag{4.49}$$

where

$$M_{k_L} := (\operatorname{vec}(\nabla^2 f_{k_L}(X_1))|\dots|\operatorname{vec}(\nabla^2 f_{k_L}(X_{N_h}))) \in \mathbb{R}^{D^2\times N_h}, \tag{4.50}$$

$$\widehat{M}_{k_L} := (\operatorname{vec}(\Delta^2 f_{k_L}(X_1))|\dots|\operatorname{vec}(\Delta^2 f_{k_L}(X_{N_h}))) \in \mathbb{R}^{D^2\times N_h}, \tag{4.51}$$

$$\bar{M}_{k_L} := (\operatorname{vec}(P_{\mathcal{W}}\Delta^2 f_{k_L}(X_1))|\dots|P_{\mathcal{W}}\operatorname{vec}(\Delta^2 f_{k_L}(X_{N_h}))) \in \mathbb{R}^{D^2\times N_h} \tag{4.52}$$

for all $k_L \in [m_L]$ and where $P_{\mathcal{W}}$ is the orthogonal projection onto $\mathcal{W} = \mathcal{W}(f, x^*)$. The proof of Proposition 4.2 is centered around the fact that

$$\|P_{\bar{\mathcal{W}}} - P_{\widehat{\mathcal{W}}_{\bar{m}}}\|_{F\to F} \leq \frac{2\|\bar{M} - \widehat{M}\|_F}{\sigma_{\bar{m}}(\widehat{M})}$$

as a consequence of Proposition 1.1. We begin by establishing an upper bound on the difference between the matrices in $\widehat{M}, \bar{M}$ in (4.48), 4.49.

Lemma 4.3 (cf. [50, Lemma 7])**.** *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$ with activations $(g_\ell)_{\ell\in[L]}$ that fulfill **(DNM1)** with $\kappa > 0$. Let μ_X be a sub-Gaussian distribution with mean $x^* = \mathbb{E}_{X\sim\mu_X}(X)$. Furthermore, assume access to a numerical differentiation scheme as in **(G5.3)** with accuracy $\hat{C}_\epsilon$. Let $X \sim \mu_X$ and $X_1, \dots, X_{N_h}$ be independent copies of $X \sim \mu_X$. There exists a uniform constant C, such that with probability at least $1 - \exp(-CN_h)$ the matrices $\bar{M}, \widehat{M}$ given by* (4.48), (4.49) *satisfy*

$$\|\widehat{M} - \bar{M}\|_F \leq \sqrt{N_h m_L}\left(\hat{C}_\epsilon + \bar{C}\sqrt{D}\,\|X - x^*\|_{\psi_2}\right),$$

with $\bar{C}$ as in Theorem 4.1.

Proof of Lemma 4.3. First, we separate the error caused by numerical differentiation by using the triangle inequality applied to the auxiliary matrix $\bar{M}$ defined in (4.49), which yields

$$\|\widehat{M} - \bar{M}\|_F \leq \|\widehat{M} - M\|_F + \|M - \bar{M}\|_F.$$

We start by estimating $\|\widehat{M} - M\|_F$. From the definition of the Frobenius norm, we immediately get

$$\begin{aligned}\|\widehat{M} - M\|_F &\leq \sqrt{m_L N_h}\max_{i\in[N_h], k_L\in[m_L]}\|\nabla^2 f_{k_L}(X_i) - \Delta^2 f_{k_L}(X_i)\|_F \\ &\leq \sqrt{m_L N_h}\hat{C}_\epsilon,\end{aligned}$$

according to **(G5.3)**. This allows us to focus on the part $\|M - \bar{M}\|_F$ from here on. By Proposition 4.1, in particular (4.26), we obtain

$$\begin{aligned}\|M - \bar{M}\|_F^2 &= \sum_{k_L=1}^{m_L} \sum_{i=1}^{N_h} \|\nabla^2 f_{k_L}(X_i) - P_{\mathcal{W}} \nabla^2 f_{k_L}(X_i)\|_F^2 \\ &\leq \sum_{k_L=1}^{m_L} \sum_{i=1}^{N_h} \left(\sum_{\ell=2}^{L} \left\| V^{[\ell]}(X_i) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(X_i)^\top - P_{\mathcal{W}} \left(V^{[\ell]}(X_i) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(X_i)^\top \right) \right\|_F \right)^2,\end{aligned}$$

where the term inside the sum corresponding to $\ell = 1$ vanishes due to $P_{\mathcal{W}^\perp}(V^{[1]}(X_i)) = P_{\mathcal{W}^\perp}(W^{[1]}) = 0$. Utilizing the properties of orthogonal projections, we can replace the matrix $P_{\mathcal{W}}(V^{[\ell]}(X_i) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(X_i)^\top)$ by an arbitrary matrix in $\mathcal{W}$ as the distance can only increase. In particular, we can replace it with $V^{[\ell]}(x^*) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(x^*)^\top \in \mathcal{W}$, such that

$$\begin{aligned}&\left\| V^{[\ell]}(X_i) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(X_i)^\top - P_{\mathcal{W}} \left(V^{[\ell]}(X_i) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(X_i)^\top \right) \right\|_F \\ &\quad \leq \left\| V^{[\ell]}(X_i) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(X_i)^\top - V^{[\ell]}(x^*) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(x^*)^\top \right\|_F,\end{aligned}$$

and then decompose the error utilizing the submultiplicativity of the Frobenius norm to get for all $i \in [N_h]$

$$\begin{aligned}&\left\| V^{[\ell]}(X_i) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(X_i)^\top - V^{[\ell]}(x^*) S_{k_L}^{[\ell]}(X_i) V^{[\ell]}(x^*)^\top \right\|_F \\ &\quad \leq \left\| V^{[\ell]}(X_i) - V^{[\ell]}(x^*) \right\| \|S_{k_L}^{[\ell]}(X_i)\|_F (\|V^{[\ell]}(X_i)\| + \|V^{[\ell]}(x^*)\|) \\ &\quad \leq 2\kappa^L \left(\prod_{k=1}^{L} \|W^{[k]}\| \right) \left(\prod_{k=1}^{\ell} \|W^{[k]}\| \right) \sum_{k=1}^{\ell-1} \kappa^{\ell+k-2} \left\| W^{[k]} \right\|_{2\to\infty} \left(\prod_{j=1}^{k-1} \|W^{[k]}\| \right) \|X_i - x^*\|_2 .\end{aligned}$$

The last line follows from the bounds derived in Lemma 4.2. Summing up the leading factor from $\ell = 2$ to L gives precisely $\bar{C}$ up to a universal constant. In total, we obtain

$$\|M - \bar{M}\|_F^2 \leq \bar{C}^2 m_L \sum_{i=1}^{N_h} \|X_i - x^*\|_2^2 .$$

By our assumptions the random variable $X_i - x^*$ is sub-Gaussian, hence it is straightforward to deduce that $\|X_i - x^*\|_2$ is sub-Gaussian as well with sub-Gaussian norm given by $\|\|X_i - x^*\|_2\|_{\psi_2} = \sqrt{D}\|X_i - x^*\|_{\psi_2}$. By using elementary properties of the sub-exponential norm (cf. [127, Lemma 2.7.6]), this implies that $Y_i = \|X_i - x^*\|_2^2$ is sub-exponential with sub-exponential norm given by $\|Y_i\|_{\psi_1} = D\|X_i - x^*\|_{\psi_2}^2$. Notably, the sum $\sum_{i=1}^{N_h} Y_i$ can be controlled by a concentration inequality. In particular, by Bernstein's inequality [127, Theorem 2.8.1] we get

$$\mathbb{P}\left(\left| \sum_{i=1}^{N_h} Y_i - \mathbb{E}Y_i \right| > t \right) \leq 2\exp\left(-c \min\left(\frac{t^2}{\sum_{i=1}^{N_h} \|Y_i - \mathbb{E}Y_i\|_{\psi_1}^2}, \frac{t}{\max_{i\in[N_h]} \|Y_i - \mathbb{E}Y_i\|_{\psi_1}} \right) \right),$$

for any $t \geq 0$ and an absolute constant $c > 0$. We can bound the mean by the subexponential norm $\|\cdot\|_{\psi_1}$ up to some constant factor (cf. [127, Proposition 2.7.1 (ii)]), hence

$$\mathbb{E}Y_i = C\|Y_i\|_{\psi_1} = CD\|X_i - x^*\|_{\psi_2}^2$$

for some absolute constant C. Using Bernstein's inequality with $t = N_h C D \|X_i - x^*\|_{\psi_2}^2$ now yields

$$\left| \sum_{i=1}^{N_h} Y_i^2 \right| \leq N_h \mathbb{E} Y_1 + t \leq 2 N_h C D \|X_i - x^*\|_{\psi_2}^2$$

with probability at least $1 - 2\exp(-C_2 N_H)$ for some universal constant $C_2 > 0$. Conditioning on this event, the bound on $\|M - P_{\mathcal{W}} M\|_F^2$ now reads

$$\|M - P_{\mathcal{W}} M\|_F^2 \leq \bar{C}^2 m_L N_h D \|X_i - x^*\|_{\psi_2}^2,$$

where we absorbed all absolute constants in $\bar{C}$. In summary, we have

$$\|\widehat{M} - \mathcal{P}_{\mathcal{W}} M\|_F \leq \|\widehat{M} - M\|_F + \|M - \mathcal{P}_{\mathcal{W}} M\|_F \leq \sqrt{m_L N_H} \left(\hat{C}_\epsilon + \bar{C} \sqrt{D} \|X_i - x^*\|_{\psi_2} \right)$$

with probability at least $1 - 2\exp(-C_2 N_h)$. The result follows after relabeling the constants. □

As a next step, we provide a bound for the spectrum of the matrices $M, \widehat{M}$ given by (4.47), (4.48), respectively.

Lemma 4.4 (cf. [50, Lemma 8])**.** *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell \in [L]}, (g_\ell)_{\ell \in [L]})$ with activations $(g_\ell)_{\ell \in [L]}$ that fulfill **(DNM1)** with $\kappa > 0$. Let $X \sim \mu_X$ with mean $\mathbb{E}[X] = x^*$ be sub-Gaussian and let $X_1, \ldots, X_{N_h}$ be independent copies of X. Furthermore, assume access to a numerical differentiation scheme as in **(G5.3)** with accuracy $\hat{C}_\epsilon$. Let $\bar{\mathcal{W}} \subseteq \mathcal{W}(f, x^*)$ be the space described in* (4.45) *constructed from the inputs $X_1, \ldots, X_{N_h}$ with $\bar{m} := \dim(\bar{\mathcal{W}})$ being its dimension. Assume*

$$\bar{\alpha} := \sigma_{\bar{m}} \left(\frac{1}{m_L} \sum_{k_L=1}^{m_L} \int \operatorname{vec}(\nabla^2 f_{k_L}(X)) \otimes \operatorname{vec}(\nabla^2 f_{k_L}(X)) \mathrm{d}\mu_X \right) > 2\hat{C}_\epsilon^2. \tag{4.53}$$

Then there exists a universal constant $C > 0$ so that the matrices M and $\widehat{M}$ as defined in (4.47), (4.48) *satisfy*

$$\sigma_{\bar{m}}(M) \geq \sqrt{\frac{\bar{\alpha} N_h m_L}{2}} \quad \text{and} \quad \sigma_{\bar{m}}(\widehat{M}) \geq \sqrt{N_h m_L} \left(\sqrt{\frac{\bar{\alpha}}{2} - \hat{C}_\epsilon^2} \right)$$

with probability at least

$$1 - \bar{m} \exp\left(-C \frac{N_h \bar{\alpha}}{\kappa^{2L} \left(\prod_{\ell=1}^{L} \|W^{[\ell]}\| \right)^2 \left(\sum_{\ell=1}^{L} \kappa^{\ell-1} \prod_{k=1}^{\ell} \|W^{[k]}\| \right)^2} \right).$$

Proof. First note $\sigma_{\bar{m}}(\widehat{M}) = \sqrt{\sigma_{\bar{m}}(\widehat{M}\widehat{M}^\top)}$ and that Weyl's inequality (cf. [132]) implies

$$\sigma_{\bar{m}}(\widehat{M}\widehat{M}^\top) \geq \sigma_{\bar{m}}(MM^\top) - \|MM^\top - \widehat{M}\widehat{M}^\top\| \geq \sigma_{\bar{m}}(MM^\top) - m_L N_h \hat{C}_\epsilon^2,$$

where we used $\|MM^\top - \widehat{M}\widehat{M}^\top\| \leq \|M - \widehat{M}\|_F^2$. By the definition of M, we can continue with

$$\sigma_{\bar{m}}(MM^\top) = \sigma_{\bar{m}} \left(\sum_{i=1}^{N_h} \sum_{k_L=1}^{m_L} \operatorname{vec}(\nabla^2 f_{k_L}(X_i)) \otimes \operatorname{vec}(\nabla^2 f_{k_L}(X_i)) \right).$$

This shows that $\sigma_{\bar{m}}(MM^\top)$ is the $\bar{m}$-th eigenvalue of a sum of N_h independent and identically distributed random matrices. Additionally, by our initial assumption, we have $\sigma_{\bar{m}}(\mathbb{E}MM^\top) = N_h m_L \bar{\alpha}$. Applying the matrix Chernoff bound [59, Theorem 4.1] yields

$$\mathbb{P}\left(\sigma_{\bar{m}}(MM^\top) \geq tN_h m_L \bar{\alpha}\right) \geq 1 - \bar{m} \exp\left(\frac{-(1-t)^2 N_h m_L \bar{\alpha}}{2 \max\limits_{x \in \mathrm{supp}(\mu_X)} \| \sum_{k_L=1}^{m_L} \mathrm{vec}(\nabla^2 f_{k_L}(x)) \otimes \mathrm{vec}(\nabla^2 f_{k_L}(x)) \|}\right)$$

for all $t \in [0,1]$. We can leverage Proposition 4.1 and Lemma 4.2 to further bound the Hessian $\nabla^2 f_{k_L}(x)$ by

$$\begin{aligned}\|\nabla^2 f_{k_L}(x)\|_F &\leq \sum_{\ell=1}^{L} \|V^{[\ell]}(x) S_{k_L}^{[\ell]}(x) V^{[\ell]}(x)^\top\|_F \leq \sum_{\ell=1}^{L} \|V^{[\ell]}(x)\|^2 \|S_{k_L}^{[\ell]}(x)\|_F \\ &\leq \sum_{\ell=1}^{L} \left(\kappa^{\ell-1} \prod_{k=1}^{\ell} \|W^{[k]}\|\right)^2 \kappa^{L-\ell+1} \prod_{k=\ell+1}^{L} \|W^{[k]}\| \leq \kappa^L \prod_{k=1}^{L} \|W^{[k]}\| \sum_{\ell=1}^{L} \kappa^{\ell-1} \prod_{k=1}^{\ell} \|W^{[k]}\|,\end{aligned}$$

universally for all $k_L \in [m_L]$ and for inputs $x \in \mathrm{supp}(\mu_X)$. Note that

$$\max_{x \in \mathrm{supp}(\mu_X)} \| \sum_{k_L=1}^{m_L} \mathrm{vec}(\nabla^2 f_{k_L}(x)) \otimes \mathrm{vec}(\nabla^2 f_{k_L}(x)) \| \leq m_L \|\nabla^2 f_{k_L}(x)\|_F^2$$

and $t = \frac{1}{2}$, concluding the proof. □

We can now complete the proof of Proposition 4.2 and Theorem 4.1.

Proof of Proposition 4.2. Combining the bounds in Lemma 4.3 and Lemma 4.4, this statement follows directly from Proposition 1.1 and the union bound of the involved probabilities. □

Proof of Theorem 4.1. The statement of Theorem 4.1 is closely related to Proposition 4.2. The additional assumptions made in the Theorem that

$$\alpha > 2\bar{C}^2 D \|X - x^*\|_{\psi_2}^2$$

implies $\bar{m} = \dim(\mathrm{range}(\bar{M})) = m$, or equivalently $\mathcal{W} = \bar{\mathcal{W}}$ and $\widehat{\mathcal{W}}_{\bar{m}} = \widehat{\mathcal{W}}$. By the proof of Lemma 4.3 we have derived the bound $\|M - \bar{M}\|_F \leq \bar{C}\sqrt{N_h m_L D}\|X - x^*\|_{\psi_2}$ for the distance between M and the projected columns of M. This bound holds with probability $1 - \exp(-CN_h)$. Additionally, repeating the proof of Lemma 4.4, where now

$$\bar{\alpha} = \alpha = \frac{1}{m_L} \sigma_m(\mathbb{E}MM^\top),$$

we get $\sigma_m(M) \geq \sqrt{\alpha N_h m_L / 2}$ with the probability described in Lemma 4.4. Invoking again Weyl's eigenvalue bound (cf. [132]), we can lower bound the extreme eigenvalue as

$$\begin{aligned}\sigma_m(P_{\mathcal{W}} MM^\top P_{\mathcal{W}}^\top) &\geq \sigma_m(MM^\top) - \left\| P_{\mathcal{W}} MM^\top P_{\mathcal{W}}^\top - MM^\top \right\|_2 \geq N_h \frac{\alpha}{2} - \|P_{\mathcal{W}} M - M\|_F^2 \\ &\geq N_H m_L \left(\frac{\alpha}{2} - \bar{C}^2 D \|X - x^*\|_{\psi_2}^2\right).\end{aligned}$$

Since $\alpha > 2\bar{C}^2 D\|X - x^*\|_{\psi_2}^2$, the right-hand side is strictly positive and thus $\sigma_m(P_{\mathcal{W}} M) > 0$, respectively, $m^* = m$. Lastly, note that the events required for the bound above as well as $\|M - P_{\mathcal{W}} M\|_F \leq \bar{C}\sqrt{N_h m_L D}\|X - x^*\|_{\psi_2}^2$ are equivalent to the events in the proof of Proposition 4.2. Therefore, Theorem 4.1 holds with the same probability as Proposition 4.2. □

4.2.3 Empirical analysis of entangled weight recovery

Our primary goal in this section is to support the theory of Section 4.2-4.2.1 by demonstrating the recovery of entangled weights for randomly constructed networks via Algorithm 4.1.

Network Model We consider neural networks with tanh activation functions at all neurons (including the output neurons). Every weight will be drawn as $w_{k_\ell}^{[\ell]} \sim \mathrm{Unif}(\mathbb{S}^{m_{\ell-1}-1})$ for all $k_\ell \in [m_\ell], \ell \in [L]$. Every neuron will be equipped with a small shift $\tau_{k_\ell}^{[\ell]} \sim \mathcal{N}(0, 0.05)$. This reflects the setting of previous experiments for stochastic gradient descent in Section 1.3.2 and wide shallow neural networks in Section 3.6. However, this time, we also explore different network depths. The primary factors that influence the difficulty of our recovery experiment will be the network depth L, for which we chose values $L = 2, \dots, 5$, and the overall number of neurons $m = m_1 + \cdots + m_L$ for which we choose values $m = 200, 300, \dots, 1500$. Throughout most experiments, the input dimension of the network will be fixed as $D = 100$. This value seems large enough to guarantee sufficient incoherence of random weights while still being low enough to run numerous configurations of the recovery. The number of output neurons is usually a small integer $m_L \in [10]$. Once the number of layers exceeds $L = 2$, the number of neurons, together with the input and output dimension of the network, does not uniquely determine the architecture since the neurons can be distributed differently over the inner hidden layers. Therefore, we also introduce a shape parameter called contraction factor $c \in (0, 1]$, which determines the decrease of the number of neurons from one layer to the next. For example, let us assume we have input dimension $D = 100$, $L = 3$ layers, and $m_3 = 10$ output neurons in a network with $m = 200$ overall neurons. To fix the shape, we need to distribute the $m - m_3 = 190$ neurons over layers one and two. For a given contraction factor c, the number of neurons is then split between the first two layers such that $m_1 + m_2 = 190$ and $m_2 = \lfloor c m_1 \rfloor$. So for $c = 1$, we would have 95 neurons in each layer whereas for $c = 0.5$ we would have $m_1 = 127, m_2 = 63$. Practically speaking, due to $c \in (0, 1]$, we only consider networks with a pyramidal shape, and c controls how tapered the pyramid is. As seen from this example, we consider networks where the number of hidden neurons in the first layer exceeds the input dimension, but we do not consider networks where the number of neurons increases from one hidden layer to the next. Additionally, note that the number of output neurons m_L is unaffected by this shape parameter. In summary, the network architectures are uniquely determined by the quintuple (D, L, m_L, m, c), and the weights are drawn uniformly at random from the unit-sphere, whereas the shifts are initialized by small Gaussians.

Entangled weight recovery For networks constructed as described above, the entangled weights are then recovered by Algorithm 4.1. The entangled weight recovery can be broken down into two steps: first, a subspace approximation, followed by the second, performing SPM (Algorithm 2.2) until we recover all entangled weights.
The subspace approximation depends on the neural network function f, N_h Hessians locations $x_1, \dots, x_{N_h}$ sampled independently from a distribution μ_X and a number of components m^*. Since all neurons are assumed to be non-linear, we always choose $m^* = m = m_1 + \cdots + m_L$. For a given network architecture we sample $N_h = 20\lceil m/m_L \rceil$ Hessian locations uniformly from the sphere with radius R around the origin, i.e., $x_1, \dots, x_N \sim R \cdot \mathrm{Unif}(\mathbb{S}^{D-1})$. Different radii $R = 10^{-3}, 10^{-2}, 10^{-1}, 1, 10$ are used to simulate different levels of concentrations.
Notably, we run a slightly adapted version of Algorithm 2.2, where the number of SPM iterations (denoted by N_{PGD}) and repetitions (denoted by N_{SPM}) is fixed. The subspace power method is always run with a step-size $\gamma = 1.5$ and performs $N_{\mathrm{PGD}} = 15000$ steps of projected gradient ascent. SPM is restarted $N_{\mathrm{SPM}} = 10^4$ times for random initializations drawn uniformly at random from the unit sphere. This way, we maintain a high likelihood of finding all entan-

gled weights (cf. Remark 2.2). Let us note here that we commonly do not observe any spurious local maximizers for the network models considered above. To see examples of regimes in which SPM actually benefits from the filtering for spurious local maximizers, we refer to the numerical section of Chapter 3.

Evaluation metrics To measure the accuracy of the subspace approximation, we simply consider the Frobenius distance between the orthogonal projection matrices $\mathcal{P}_{\mathcal{W}}, \mathcal{P}_{\widehat{\mathcal{W}}}$ associated with the vectorizations of the subspaces $\mathcal{W}, \widehat{\mathcal{W}}$ (cf. Definition 1.1 and the preceding remark) normalized by the number of components m, given by

$$\frac{\left\|\mathcal{P}_{\mathcal{W}} - \mathcal{P}_{\widehat{\mathcal{W}}}\right\|_F}{\sqrt{m}} \quad \text{where } \mathcal{P}_{\mathcal{W}}, \mathcal{P}_{\widehat{\mathcal{W}}} \in \mathbb{R}^{D^2 \times D^2},$$

which relates to the error estimated in Theorem 4.1. Every repetition of SPM computes a new potential entangled weight direction which is added to $\mathcal{U}_{\text{SPM}}$ as long as it does not fail the check for spurious local maximizers. By running SPM, we lose scale and sign information about the entangled weights since all returned vectors are normalized, and SPM cannot distinguish $u \in \mathbb{R}^D$ from $-u \in \mathbb{R}^D$ because of

$$\left\|P_{\widehat{\mathcal{W}}}(u^{\otimes 2})\right\|_F^2 = \left\|P_{\widehat{\mathcal{W}}}(-u^{\otimes 2})\right\|_F^2.$$

We exclusively track the accuracy of entangled weight approximations within $\mathcal{U}_{\text{SPM}}$. The entangled weights corresponding to output neurons ($\mathcal{U}_L$ in Algorithm 4.1) will be considered in the experiments of Section 4.4. We check the set $\mathcal{U}_{\text{SPM}}$ for two qualities. First, the number of entangled weights that were recovered. An entangled weight

$$v \in \left\{ v_{k_\ell}^{[\ell]}(0) \,\middle|\, \ell \in [L], k_\ell \in [m_\ell] \right\}$$

counts as recovered as long as there exists an $u \in \mathcal{U}_{\text{SPM}}$ such that

$$\min_{s \in \{-1,+1\}} \left\| u - s \frac{v}{\|v\|_2} \right\|_2 \leq 0.05. \tag{4.54}$$

Additionally, we want to check if there were any vectors returned in $\mathcal{U}_{\text{SPM}}$ that are not close (in the sense of (4.54)) to any entangled weight. We count these vectors as false positives and measure the overall rate of false positives contained in $\mathcal{U}_{\text{SPM}}$.

Recovery of entangled weights for different network configurations and varying levels of concentration of the Hessian locations. Note that all upcoming numerical results are taken from the joint work [50]. The first experiment we want to discuss is designed to test the influence of distribution $\mu_X = R \cdot \text{Unif}(\mathbb{S}^{D-1})$ by comparing different radii $R = 10^{-3}, 10^{-2}, 10^{-1}, 1, 10$, which models the degree of concentration. The input and output dimensions remain fixed $D = 100, m_L = 10$, and we vary the number of neurons, the number of layers $L = 2, 3, 4$ and the network shape modeled with the contraction factor $c = 0.25, 0.5, 0.7, 1$. We report the subspace approximation error in Figure 4.2, the ratio of recovered entangled weights and false positives in Figure 4.3. We can see that the subspace error is strongly influenced by the overall number of neurons. All plots exhibit a sort of phase transition. For a certain number of neurons (the first few hundred), the overall recovery works well and we recover all entangled weights and encounter no false positives. Ultimately, we know from the previous chapter that SPM does successfully recover the target weights in regimes up to $m \log m = \mathcal{O}(D^2)$(as long

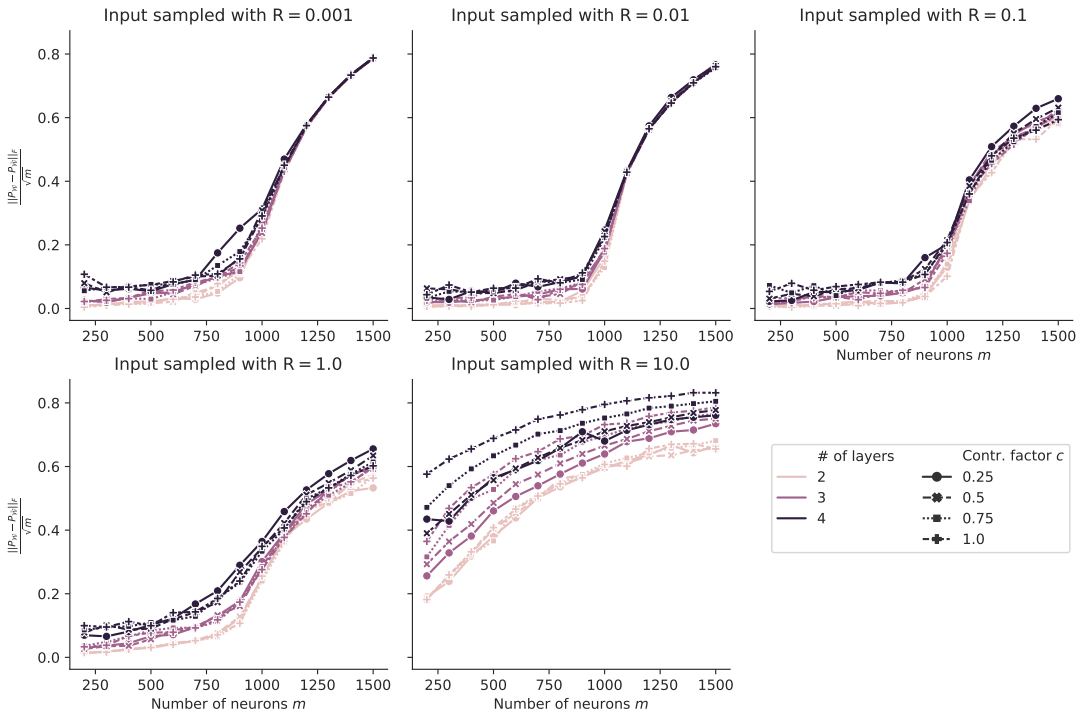

Figure 4.2: Sensitivity of the subspace error w.r.t. the concentration of the input distribution [52]. As long as a sufficient degree of concentration is maintained (top row), the recorded subspace error remains stable for the first few hundred neurons.

as certain incoherence conditions are met). Therefore, the crucial factor is the accuracy of the subspace approximation. In terms of R, the experimental results suggest that our theoretical intuition was correct. Concentration is necessary since for $R = 10$ we see a significant increase in the subspace approximation error. As long as a certain level of concentration is maintained ($R = 0.001, 0.01, 0.1$) the subspace is approximated sufficiently well. We can only observe marginal changes between small radii. Eventually, when a critical number of neurons is reached, the subspace error increases drastically (between 900 and 1000 neurons), even for small R. At this stage, the recovery rate for $L = 2, 3$ remains stable, but we can observe a steady decrease in the recovered entangled weights of the four-layer networks. The fact that the breakdown of the recovery rate lags behind the phase transition of the subspace approximation error suggests that SPM exhibits a certain robustness w.r.t. the difference between the matrix spaces. After $m = 1100$, the error in the approximated subspace seems to be too high, and the ratio of detected entangled weights drops off significantly. At the same time, we also observe a rapid increase in false positives.
We can also observe that the recovery performed worse for a higher number of layers. However, this quantity only influences the overall recovery by a small shift along the y-axis. The contraction factor does not significantly influence two-layer and three-layer networks. Furthermore, we can not see significant differences between the network shapes. For four-layer networks, the results suggest that having almost no reduction from layer to layer (i.e., a rectangular-shaped network) performs slightly worse than the networks with a tapered architecture. However, the overall impact of the contraction factor seems negligible.

Let us put some perspective on these results. The fact that for this type of pyramidal artificial neural network, the number of layers and the shape does not have a significant effect on the recovery of entangled weights suggests that this method scales to deep architectures. We can

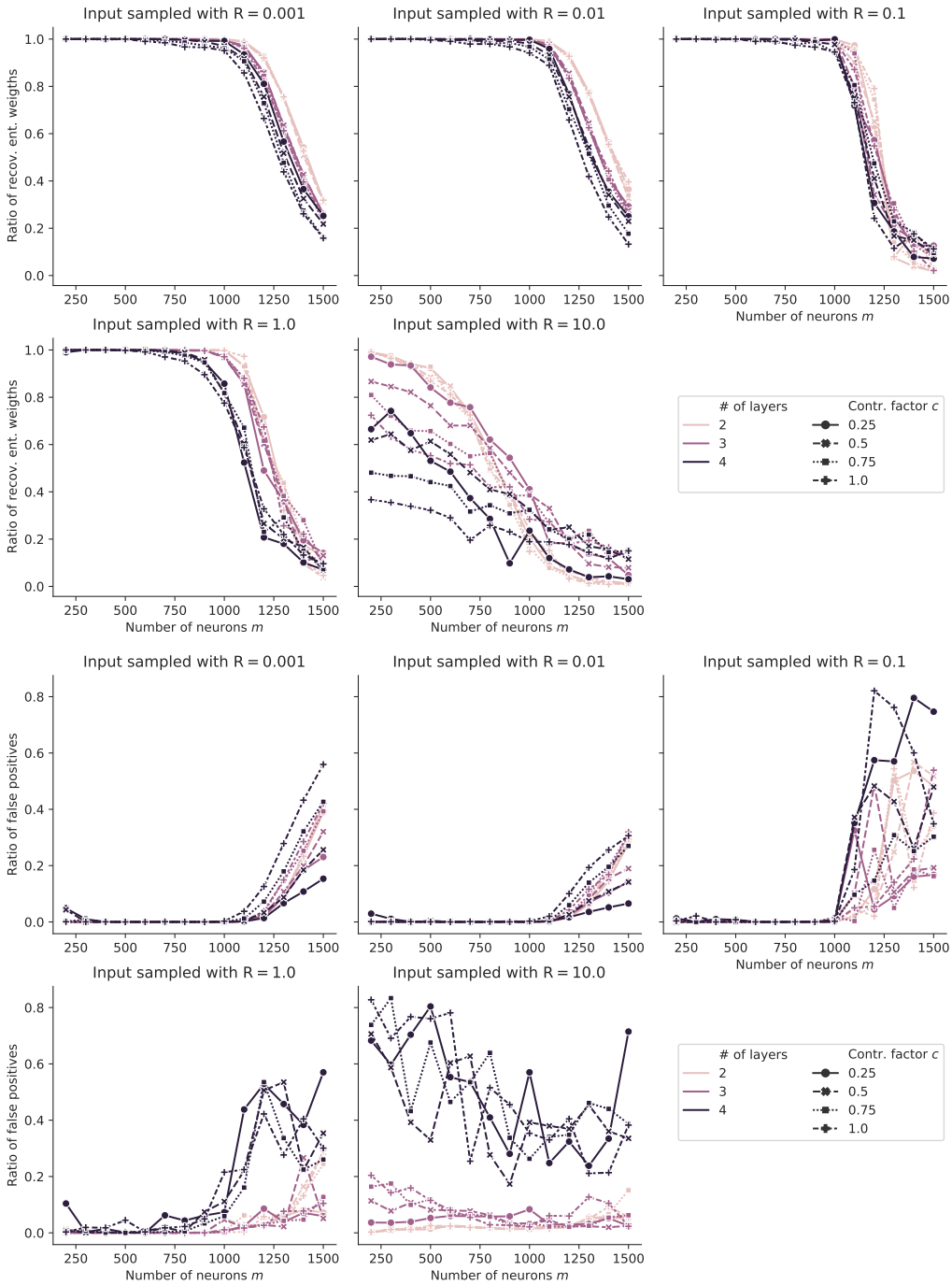

Figure 4.3: Ratio of recovered entangled weights and detection rate of false positives in the set of vectors returned by SPM [52]. In general, we can detect a critical mass of neurons at which the recovery rate decreases significantly. Provided the correct input distribution (top row), we observe a detection of all entangled weights. Only in the runs with 4-layer networks and for contraction $c = 0.75, 1$ there is a slight drop-off in the recovery ratio after roughly 500 neurons.

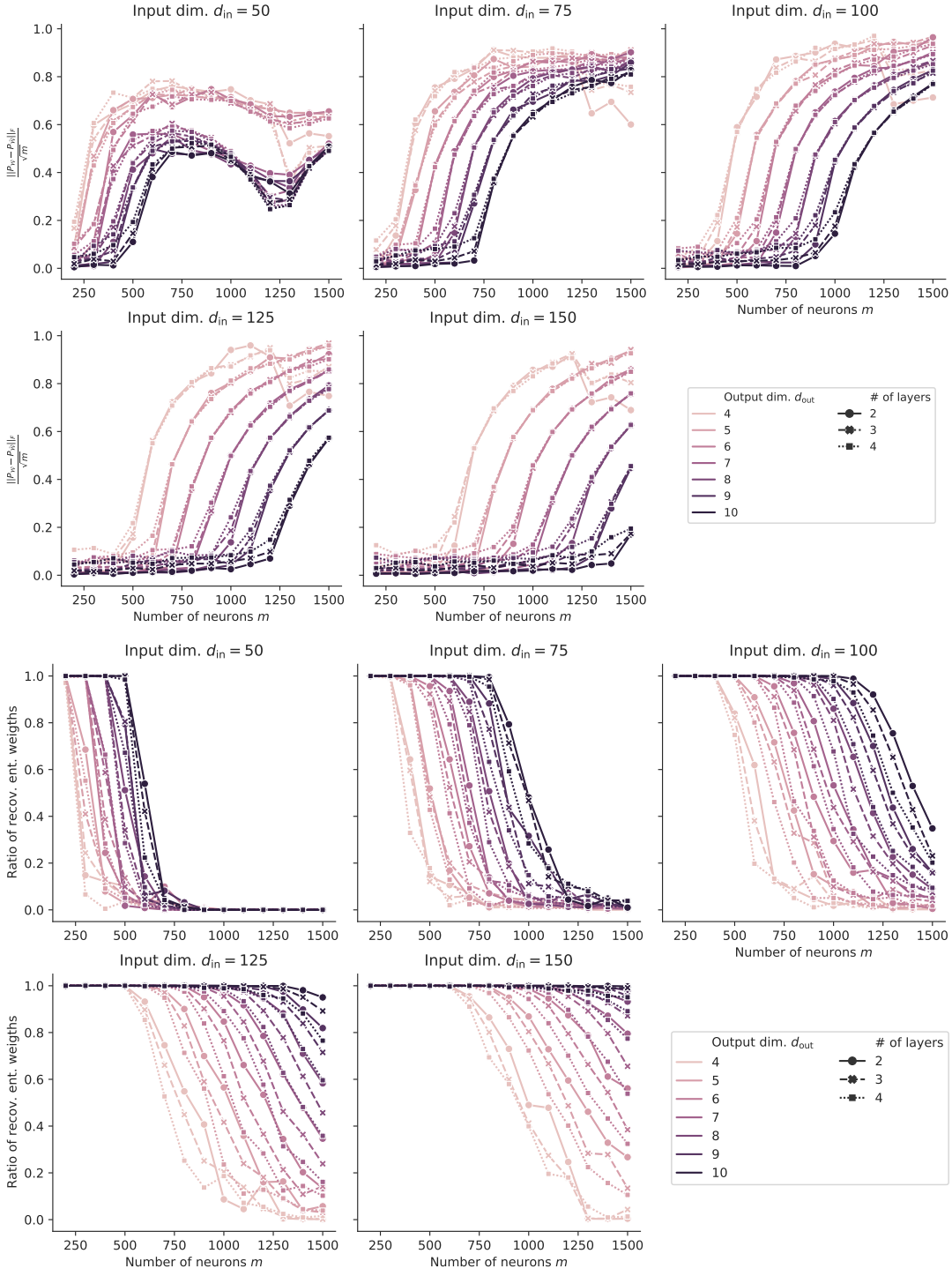

Figure 4.4: Varying the input dimension D and number of output neurons m_L for sigmoidal networks with $c = 0.5$. We observe a strong correlation between those two factors and the number of neurons for which the subspace $\mathcal{W}$ can be sufficiently well approximated.

also derive an interesting question from these results: What is causing the phase-transition at a specific number of neurons which occurs for all different network types considered in Figure 4.3? Based on empirical tests, we believe that this critical transition point is connected to the input dimension D and the number of output neurons m_L. At the end of Section 4.2.1, we already elaborated on why having more output neurons can be beneficial. To provide further evidence for this claim, we repeat the experiments, but now we fix all parameters which had no significant impact on the overall recovery and vary the input dimension D and the number of output neurons m_L.

Investigating the phase-transition We reuse the setup from the last experiments. This time, we let $R = 0.01, c = 0.5$ be fixed and vary the input dimension $D = 50, 75, 100, 125, 150$ and the number of output neurons $m_L = 4, 5, 6, \ldots, 10$. We use the same set of hyperparameters as before, and the evaluation metrics remain unchanged as well. The results are reported in Figure 4.4. Not only do the results align with our hypothesis that input dimension and the number of output neurons are the determining factors that determine the number of recoverable neurons, but they also suggest that the tipping point w.r.t. the number of recoverable neurons is roughly located at $D \cdot m_L$. There are various potential reasons why increasing D and m_L can benefit the overall recovery. Increasing the input dimension D while maintaining a constant number of neurons will increase the incoherence of the weights in layer one and therefore increase the overall incoherence of the entangled weights. It should be clear by now that increased incoherence aids the performance of the subspace approximation and SPM. Additionally, we have already stressed numerous times the strong influence of the number of outputs m_L on the stabilization of the singular subspace. However, these effects should only lead to a gradual improvement. For instance, it is questionable whether the marginal gain of incoherence caused by increasing D from $D = 100$ to $D = 150$ fully explains the strong correlation between the point of phase transition and $D \cdot m_L$.

4.3 Network completion

Entangled weights serve as a generalization of ordinary weights, enabling us to decouple weight information accessible via network differentiation. We established a theoretical framework for the recovery of individual entangled weight vectors and provided empirical evidence that this recovery is possible for deep neural networks with sigmoidal activations. In Section 4.1.2, we briefly motivated the entangled weights by the fact that the weights can be reconstructed up to a diagonal matrix according to

$$V^{[\ell-1]}(x)^\dagger V^{[\ell]}(x) = \operatorname{diag}\left(g^{(1)}(z^{[j]}(x))\right) W^{[\ell]}, \tag{4.55}$$

as long as the left-inverse $V^{[\ell-1]}(x)^\dagger$ exists. With our ultimate objective of network reconstruction in mind, this motivates the concept of entangled weights. However, as it became clear in Section 4.2.1, our approximation of entangled weights suffers from the same ambiguity as the weight recovery in Chapter 3. Namely, by translating their retrieval to the rank-one matrix basis recovery problem of Chapter 2, we lose information about their scale and sign. Furthermore, the entangled weight recovery does not reconstruct entangled weight matrices but only individual entangled weights. To reconstruct weights from entangled weights as in (4.55), we need to know the corresponding layer for each entangled weight computed by Algorithm 4.1. Algorithm 4.1 computes the entangled weights of the last layer individually. Hence, for the last layer, such an assignment is already inherent to Algorithm 4.1. However, we do not know the corresponding neuron in the network for all remaining entangled weight approximations $v \in \mathcal{U}_{\text{SPM}}$. In summary, we end up with the following problem: Consider any $v \in \mathcal{U}_{\text{SPM}}$ after

successfully running Algorithm 4.1 against a network f with inputs concentrating around x^*, then we expect that there is an unknown scale $s > 0$ and a layer $\ell^* \in [L]$ such that

$$v \approx s \cdot v^{[\ell^*]} \text{ for some } v^{[\ell^*]} \in \mathcal{V}_\ell^*(f, x^*),$$

where $\mathcal{V}_\ell(x^*)$ is defined in Definition 4.2. However, based on the output of Algorithm 4.1, we have no information about the value of ℓ^*. Solving this *layer assignment* is essential for reconstructing the entangled weight matrices, which will be tackled heuristically in the following for networks with up to $L = 3$ layers.

4.3.1 Layer Assignment

This section explains how we need to order the output of the weight recovery to reconstruct the entangled weight matrices. More precisely, we study a slightly more general problem. Note that the entangled weight recovery (Algorithm 4.1) invokes Algorithm 2.2 as a subroutine that iterates SPM (cf. Section 2.4). Algorithm 2.2 repeats SPM, where each repetition is run until convergence. Then, the final result is filtered for spurious local maximizers and compared to previous approximations to avoid duplicate computations of local maximizers. The certainty that $\mathcal{U}_{\text{SPM}}$ will only contain each entangled weight approximation once is desirable in our theoretical analysis. However, it enforces a *sequential setup*, where only one SPM iteration can be run at a time (otherwise, duplicates can not be avoided). In practice, it is much more efficient to run a large number of SPM iterations in parallel since most operations in Algorithm 2.2 can be written as matrix-vector products. In fact, in our numerical experiments, we run several thousand ($N_{\text{SPM}} = 5m \log m$, see also Remark 2.2) independent iterations of SPM simultaneously (cf. Section 4.4) for a fixed number of steps. Even on consumer hardware, this leads to an immense speed-up of the entangled weight recovery. However, as a tradeoff, the computed set of entangled weight approximations $\mathcal{U}_{\text{SPM}}$ will contain multiple approximations for each individual entangled weight. Therefore, to be more aligned with our experimental setting, we will consider the case where $\mathcal{U}_{\text{SPM}}$ can contain duplicate approximations. Notably, filtering out duplicate detections can be regarded as redundant from a theoretical point of view.

Problem setting. More precisely, we consider the following problem setting. Let $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$ be a (deep) neural network with entangled weights given by $\mathcal{V}(f, x)$ and assume a setting that is reached after successfully applying the entangled weight recovery (Algorithm 4.1) to f. Based on the discussion above, we assume that there exists $x^* \in \mathbb{R}^D, \epsilon \geq 0$ and that we are given access to a set $\mathcal{U} = \mathcal{U}_{\text{SPM}} \cup \mathcal{U}_L$ that fulfills the following two conditions: Firstly, every vector in $\mathcal{U}$ approximates a member of $\mathcal{V}(f, x^*)$ such that

$$\forall u \in \mathcal{U} \exists v \in \mathcal{V}(f, x^*) \text{ such that } \min_{s\in\{-1,+1\}} \left\| u - s\frac{v}{\|v\|_2} \right\|_2 \leq \epsilon, \tag{4.56}$$

and secondly, every entangled weight is approximated by at least one vector in $\mathcal{U}$, i.e., we have

$$\forall v \in \mathcal{V}(f, x^*) \exists u \in \mathcal{U} \text{ such that } \min_{s\in\{-1,+1\}} \left\| u - s\frac{v}{\|v\|_2} \right\|_2 \leq \epsilon. \tag{4.57}$$

As long as ϵ can be picked sufficiently small, this makes sure that all entangled weights are sufficiently well approximated up to sign and scale. Additionally, the first condition prevents the presence of false positives (spurious local maximizers) within $\mathcal{U}$. Throughout this section, we assume that these conditions are met for an ϵ that is suitably small.

Algorithm 4.2: Constructing entangled weight matrices

Input: Output of entangled weight recovery $\mathcal{U} = \mathcal{U}_{\text{SPM}} \cup \mathcal{U}_L$, input distribution for the detection of first layer weights $\mu_X = \text{Unif}(R \cdot \mathbb{S}^{D-1})$.

$\mathcal{U}^*_{\text{SPM}} \leftarrow \left\{u \in \mathcal{U}_{\text{SPM}} \,\middle|\, \forall u_L \in \mathcal{U}_L : \min_{s\in\{-1,+1\}} \|u - s u_L\|_2 > \delta\right\}$

$\mathcal{U}^*_{\text{SPM}+} \leftarrow \{\text{sign}(u_1)u \mid u \in \mathcal{U}_{\text{SPM}}\}$

Run kMeans on $\mathcal{U}^*_{\text{SPM}+}$ with $m - m_L$ cluster centers

Denote the set of normalized cluster centers by $\widehat{\mathcal{V}}_{1:L-1}$

Draw N_h random samples $x_1, \dots, x_{N_h}$ from μ_X

Compute Hessian approximations $\Delta^2 f_{k_L}(x_i)$ for all $i \in [N_h], k_L \in [m_L]$

Compute the m_1-th left singular subspace of the Hessian approximations

$$\widehat{\mathcal{W}}_1 = \text{span}_{m_1}\left\{\Delta^2 f_1(x_1), \dots, \Delta^2 f_1(x_{N_h}), \dots, \Delta^2 f_L(x_1), \dots \Delta^2 f_L(x_{N_h})\right\},$$

and denote by $P_{\widehat{\mathcal{W}}_1}$ the orth. projection onto $\widehat{\mathcal{W}}_1$

for $v \in \widehat{\mathcal{V}}_{1:L-1}$ **do**

  Compute the score $S_v \leftarrow \left\| P_{\widehat{\mathcal{W}}_1}(v \otimes v) \right\|_F$

end

Sort the vectors in $\widehat{\mathcal{V}}_{1:L-1}$ according to their score S_v in descending order

Build the matrix $\widehat{V}^{[1]} \in \mathbb{R}^{D \times m_1}$ from the m_1 vectors in $\widehat{\mathcal{V}}_{1:L-1}$ with the highest score

Build the matrix $\widehat{V}^{[2:L-1]} \in \mathbb{R}^{D \times m_2 + \dots + m_{L-1}}$ from all remaining vectors in $\widehat{\mathcal{V}}_{1:L-1}$.

Build the matrix $\widehat{V}^{[L]} \in \mathbb{R}^{D \times m_L}$ with columns taken from the set $\mathcal{U}_L$

Output: $\widehat{V}^{[1]}, \widehat{V}^{[2:L-1]}, \widehat{V}^{[L]}$

Step 1: Filtering the potential candidates returned by Algorithm 4.1. Under the assumptions above, we first face two main challenges. The condition in (4.57) only guarantees that for every entangled weight vector $v \in \mathcal{V}(f, x^*)$ there exists a set of vectors $\mathcal{U}_v \subset \mathcal{U}$ that approximates v up to a sign and scale. Therefore, in the first step, we need to find exactly one representative per entangled weight. Assuming the entangled weights are sufficiently incoherent, most clustering methods can be used. It should be noted that there is one special case corresponding to the entangled weights of the last layer. Their approximations were computed in a separate step of Algorithm 4.1 since the gradient of the output neurons in x^* can be directly accessed. In practice, it makes sense to rely solely on the approximations within $\mathcal{U}_L$ as potential candidates for the last layer. As long as the output neurons were equipped with non-linear activations, there can still be vectors in $\mathcal{U}_{\text{SPM}}$ corresponding to these output neurons. To remove all duplicate approximations of the last layer's entangled weights, the set $\mathcal{U}_{\text{SPM}}$ is filtered according to the criteria

$$\mathcal{U}^*_{\text{SPM}} := \left\{ u \in \mathcal{U}_{\text{SPM}} \,\middle|\, \forall u_L \in \mathcal{U}_L : \min_{s\in\{-1,+1\}} \|u - s u_L\|_2 > \delta \right\}, \tag{4.58}$$

for some small $\delta > 0$ which has to be chosen according to the separation between the entangled weights and approximation error. To avoid two clusters per entangled weight due to the ambiguity of the sign (one for $-v/\|v\|_2$ and one for $v/\|v\|_2$), all vectors in the set $\mathcal{U}^*_{\text{SPM}}$ are then projected onto one half of the half sphere. We denote these projected approximations as

$$\mathcal{U}^*_{\text{SPM}+} := \{\text{sign}(u_1)u \mid u \in \mathcal{U}_{-L}\}.$$

At this point, a clustering method like the kmean++ algorithm is applied to $\mathcal{U}^*_{\text{SPM}+}$ to detect $m - m_L$ clusters, each of which corresponds to approximations of one entangled weight

attributed to the hidden layers of f. Each cluster center will be used as a representative approximation of one entangled weight vector, and we denote the ensemble of cluster centers as $\widehat{\mathcal{V}}_{1:L-1}$.

Step 2: Layer assignment. We assume now that the filtering in **Step 1** was successful. Then, the set $\mathcal{U}_L$ contains the approximations of all entangled weights in the output layer with the correct order up to sign and scale. Furthermore, there exists exactly one approximation in $\widehat{\mathcal{V}}_{1:L-1}$ for each entangled weight corresponding to a hidden neuron. Each element in $\widehat{\mathcal{V}}_{1:L-1}$ now needs to be assigned to the right layer, i.e., we need to solve the problem described by

$$\phi : \widehat{\mathcal{V}}_{1:L-1} \to [L-1], \phi(v) = \operatorname{argmin}_{\ell \in [L-1]} \min_{k_\ell \in [m_\ell]} \min_{s \in \{-1,+1\}} \left\| v - s v_{k_\ell}^{[\ell]} / \|v_{k_\ell}^{[\ell]}\|_2 \right\|_2,$$

without having access to any of the true entangled weights $v_{k_\ell}^{[\ell]}$. There have been two approaches to detect the first layer weights within $\widehat{\mathcal{V}}_{1:L-1}$. We only briefly sketch the first approach, which requires that the activation function saturates for large inputs (which is the case for sigmoidals) and is used in [52]. The layer assignment in [52] considers the network along the entangled weight directions, given by the set of functions

$$f_v(t) := f(t \cdot v) \quad \text{for } v \in \widehat{\mathcal{V}}_{1:L-1}, t \in \mathbb{R}.$$

If v is one of the first layer weights, for instance $v = \pm w_1^{[1]} / \|w_1^{[1]}\|_2$, then the output of the first layer evaluated at $t \cdot v$ reads

$$y^{[1]}(t \cdot v) = g\left(\pm \frac{t}{\|w_1^{[1]}\|_2} \| \cdot W^{[1]^\top} w_1^{[1]} + \tau^{[1]} \right).$$

As long as the columns of $W^{[1]}$ are sufficiently incoherent, the first component of the vector $W^{[1]^\top} w_1^{[1]}$ will dominate the rest. For example, if the columns of $W^{[1]}$ were orthogonal then $t/\|w_1^{[1]}\|_2 W^{[1]^\top} w_1^{[1]} = t\|w_1^{[1]}\|_2 e_1$. Clearly, under the orthogonality assumption, we can only manipulate the function value of the first neuron if we evaluate f along $w_1^{[1]}$, such that even for very large t, only one neuron can get saturated (i.e., approach the regime of the activation function which is almost constant). However, if v is not one of the first layer weights, then $tW^{[1]^\top} v$ will not exhibit the same sparsity because all deeper entangled weights are a linear combination of the first layer weights. Hence, by increasing t, we can saturate more (or even all) neurons in the first layer. As argued in [52], once a high number of neurons in the first is saturated, it becomes hard to change the output of the network. This implies that the magnitude of the gradient $\|\nabla f_v(t)\|_2$ decreases faster if v does not belong to the first layer weights. This idea was used successfully in [52] for sigmoidal three-layer networks to detect the first layer weights. Unfortunately, this idea relies heavily on the strong incoherence in the first layer. If the weights do not closely resemble an orthogonal system, it does not provide enough robustness to scale to deeper architectures with several outputs.
Therefore, we decide to rely on another method to detect first-layer weights. This approach has been slightly adopted but is based on our main reference [50]. The unique quality of first-layer entangled weights is that they are identical to the original weights and do not depend on the input (cf. Section 4.1.2). For the recovery of all remaining entangled weights, we had to rely on a stabilization argument which required sampling the Hessian locations close around the point x^*. The most significant part of the subspace error bound in Theorem 4.1 scales with the sub-Gaussian norm of the input distribution, i.e., $\sqrt{D}\,\|X - x^*\|_{\psi_2}$. Additionally, we saw in

the numerical experiment of Section 4.2.3 that without concentration (e.g., $R = 10$ in Figure 4.2), the entangled weight recovery breaks down. While we can not precisely quantify what subspace is spanned by the Hessians if we drastically increase the variance of the Hessian locations, it seems plausible that the subspace spanned by the symmetric tensors of the first layer weights will still be contained in it: According to (4.20), the decomposition of any Hessian at output k_L is given by

$$\nabla^2 f_{k_L}(x) = \sum_{k_1=1}^{m_1} \frac{\partial f_{k_L}(x)}{\partial y_{k_1}^{[1]}(x)} g_1^{(2)}(z_{k_1}^{[1]}(x)) w_{k_1}^{[1]} \otimes w_{k_1}^{[1]} + \sum_{\ell=2}^{L} \sum_{k_\ell=1}^{m_\ell} \frac{\partial f_{k_L}(x)}{\partial y_{k_\ell}^{[\ell]}(x)} g_\ell^{(2)}(z_{k_\ell}^{[\ell]}(x)) v_{k_\ell}^{[\ell]}(x)^{\otimes 2}.$$

The left part of this expression will always be contained in the space

$$\mathcal{W}_1 := \operatorname{span}\left\{ w_1^{[1]} \otimes w_1^{[1]}, \dots, w_{m_1}^{[1]} \otimes w_{m_1}^{[1]} \right\}. \tag{4.59}$$

Now our goal is to construct a set of Hessians such that the subspace approximation Algorithm returns a space $\widehat{\mathcal{W}}_1$ such that contains the first layer tensors $\mathcal{W}_1 \subset \widehat{\mathcal{W}}_1$ and at the same time does not contain the remaining entangled weights, i.e., it satisfies $v_{k_\ell}^{[\ell]}(x^*)^{\otimes 2} \notin \widehat{\mathcal{W}}_1$ for $\ell > 1, k_\ell \in [m_\ell]$. The subspace does not need to perfectly fulfill these conditions since we already obtained all entangled weights. It is sufficient that

$$\left\| P_{\widehat{\mathcal{W}}_1}(w_{k_1}^{[1]} \otimes w_{k_1}^{[1]}) \right\|_F \gg \left\| P_{\widehat{\mathcal{W}}_1}(v_{k_\ell}^{[\ell]}(x^*)^{\otimes 2}) \right\|_F \; \forall k_1 \in [m_1], \ell > 1, k_\ell \in [m_\ell]. \tag{4.60}$$

The construction of the subspace $\widehat{\mathcal{W}}_1$ depends on the distribution μ_X of the Hessian locations and the number of components. For the distribution μ_X we now increase the variance and could also consider a distribution with a mean different than x^*. The magnitude of the projection $\left\| P_{\widehat{\mathcal{W}}}(\cdot) \right\|_F$ is now used as a score to detect the first layer weights. Combining this with the fact that we already know the entangled weights of the last layer, this leaves only entangled weights in $\widehat{\mathcal{V}}_{1:L-1}$ which belong to the inner hidden layers ($\ell = 2, \dots, L-1$). Currently, we do not dispose of a method that distinguishes the inner entangled weights, which ultimately limits us to networks with $L = 3$ layers. We provide numerical evidence in Section 4.4 that this method yields the correct layer assignment for three-layer sigmoidal neural networks. The overall layer assignment has been summarized in Algorithm 4.2.

4.3.2 Loss-free reparametrization

The last section presents heuristics that cluster the entangled weight approximations w.r.t. their corresponding layers (for $L \leq 3$). Notably, this assignment does not recover the original order within one particular layer. Only for the last layer were we able to recover the order based on the order of computation of the elements within $\mathcal{U}_L$. Note, however, the fact that the order of the hidden neurons within one layer is not recoverable is inherent to the network identification problem as it represents one of the natural symmetries (permutations of the neurons) under which the network mapping is invariant (see discussion about natural symmetries within Chapter 1).

The main result of this section provides a reparametrization $\tilde{f}$ of a deep pyramidal neural network f incorporating the information provided by the entangled weights. Our approach is similar to the refinement step for the reconstruction of shallow neural networks (cf. Section 3.4.1). In particular, we construct a parametric function $\tilde{f}$ that only depends on the remaining unknown parameters, which are the shifts of the original network, as well as the signs and scaling of the entangled weights. This reparametrization is loss-free because there is functional

equivalence between f and $\tilde{f}$ for the correct set of signs, scales, and shifts. Based on our previous results, we assume the setting where we have access to $V^{[1]}(x^*), \dots, V^{[L]}(x^*)$ up to signs, scales, and permutations of the columns. To be precise, we assume we are given access to matrices

$$\tilde{V}^{[\ell]} := \tilde{V}^{[\ell]}(x^*) = V^{[\ell]}(x^*)\pi_\ell S_\ell, \quad \text{for all } \ell \in [L],$$

where $S_1, \dots, S_L$ are invertible diagonal matrices and $\pi_1, \dots, \pi_L$ are permutation matrices. One important remark we need to make concerns the special case of the last layer. For this special case, we can infer the original order of the entangled weights by computing the gradient of the output neurons (see, for instance, Algorithm 4.1), such that π_L is simply the identity matrix. Below, for simplicity, the dependency of the entangled weights on the input will often be neglected, i.e., we simply write $V^{[\ell]}$ instead of $V^{[\ell]}(x^*)$, as the entangled weights themselves and therefore x^* will not be manipulated and remain fixed throughout this section. We also extend this simplification to the representation of the entangled weight matrices in terms of the ground truth weights such that

$$\tilde{V}^{[\ell]} = \left(\Pi_{k=1}^{\ell-1} W^{[k]} D_k\right) W^{[\ell]} \pi_\ell S_\ell \in \mathbb{R}^{m_{\ell-1} \times m_\ell}, \quad \text{for all } \ell \in [L], \tag{4.61}$$

where $D_k \in \mathbb{R}^{m_k \times m_k}$ are diagonal matrices and $m_0 = D$. This representation follows by setting $D_k = \operatorname{diag}(g^{(1)}(z^{[k]}(x^*)))$. Before stating the main results of this section, let us give a few auxiliary identities.

Lemma 4.5. *Let $S \in \mathbb{R}^{D \times D}$ be any diagonal matrix and $\pi \in \mathbb{R}^{D \times D}$ a permutation matrix.*

(i) The inverse of π is given by its transpose $\pi^\top$.

(ii) Denote by $\operatorname{diag}(S)$ the vector storing the diagonal elements of S, then

$$\pi^\top S \pi = \operatorname{diag}(\pi^\top \operatorname{diag}(S)), \tag{4.62}$$

in particular, $\pi^\top S \pi$ is also diagonal, and its diagonal elements are given by $\pi^\top \operatorname{diag}(S)$.

Proof. The first point follows from the fact that permutation matrices are, by definition, orthogonal. For statement (ii), note that $\pi^\top S \pi$ must be a diagonal matrix. This follows from the fact that for $S = \mathrm{Id}$ we have $\pi^\top S \pi = \pi^\top \pi = \mathrm{Id}$. Since changing the values of S does not affect how elements are permuted, the matrix $\pi^\top S \pi$ must also be diagonal for any arbitrary diagonal matrix S. Now consider the k-th standard vector e_k, then

$$\pi^\top S \pi e_k = \pi^\top S e_{\pi(k)} = S_{\pi(k),\pi(k)} \cdot \pi^\top e_{\pi(k)} = S_{\pi(k),\pi(k)} e_k.$$

□

Let us formally introduce a class of parametric functions, which represents neural networks that are attained by scaling the columns and rows of weight matrices.

Definition 4.4 (Deep neural network reparametrization using fixed weights)**.** *Assume we are given access to matrices $\tilde{W}^{[1]}, \dots, \tilde{W}^{[L]}$ where $\tilde{W}^{[\ell]} \in \mathbb{R}^{m_{\ell-1} \times m_\ell}$ and activation functions $(g_\ell)_{\ell \in [L]}$. We define the neural network reparametrization w.r.t. these matrices as the parametric function*

$$\tilde{f}(\cdot, (\tilde{L}_\ell, \tilde{R}_\ell, \tilde{N}_\ell, \tilde{\tau}_\ell)_{\ell \in [L]}) : \mathbb{R}^{m_0} \to \mathbb{R}^{m_L},$$

which for given L quadruplets $(\tilde{L}_\ell, \tilde{R}_\ell, \tilde{N}_\ell, \tilde{\tau}_\ell)_{\ell \in [L]}$ consisting of diagonal matrices

$$\tilde{L}_\ell \in \mathbb{R}^{m_{\ell-1} \times m_{\ell-1}}, \tilde{R}_\ell, \tilde{N}_\ell \in \mathbb{R}^{m_\ell \times m_\ell}$$

and shift vectors $\tilde{\tau}_\ell \in \mathbb{R}^{m_\ell}$, produces its output according to the iteration

$$\begin{aligned}\tilde{y}^{[0]}(x) &:= x\\ \tilde{y}^{[\ell]}(x) &:= \tilde{N}_\ell\, g_\ell\left((\tilde{L}_\ell \tilde{W}^{[\ell]} \tilde{R}_\ell)^\top \tilde{y}^{[\ell-1]}(x) + \tilde{\tau}_\ell\right), \quad \text{for } \ell = 1,\dots,L,\\ \tilde{f}(x) &:= \tilde{y}^{[L]}(x).\end{aligned}$$

The first observation is that this reparametrization covers all classical feed-forward neural networks with weights given by $(\tilde{L}_\ell \tilde{W}^{[\ell]} \tilde{R}_\ell)_{\ell\in[L]}$. This follows by simply setting all the outer diagonal matrices $\tilde{N}_\ell$ to the identity matrix in the corresponding dimension. Notably, this reparametrization model covers all classical feed-forward neural networks that can be obtained by rescaling the columns and rows fixed set of weights. The reason why we include the outer diagonal matrices $\tilde{N}_\ell$ in this definition is merely technical since these matrices will be used during optimization. Note, however, even if $(\tilde{N}_\ell)_{\ell\in[L]}$ were not equal to the identity, then, due to

$$\begin{aligned}\tilde{y}^{[\ell]}(x) &= \tilde{N}_\ell\, g_\ell\left((\tilde{L}_\ell \tilde{W}^{[\ell]} \tilde{R}_\ell)^\top \tilde{y}^{[\ell-1]}(x) + \tilde{\tau}_\ell\right)\\ &= \tilde{N}_\ell\, g_\ell\left((\tilde{N}_{\ell-1}\tilde{L}_\ell \tilde{W}^{[\ell]} \tilde{R}_\ell)^\top g_{\ell-1}\left((\tilde{L}_{\ell-1} \tilde{W}^{[\ell-1]} \tilde{R}_{\ell-1})^\top \tilde{y}^{[\ell-2]}(x) + \tilde{\tau}_{\ell-1}\right) + \tilde{\tau}_\ell\right),\end{aligned}$$

we could simply absorb $\tilde{N}_{\ell-1}$ into the parameter $\tilde{L}_\ell$ for all $\ell \in 2,\dots,L$. This excludes only the rescaling of the outputs with $\tilde{N}_L$. Hence, for any parameter combination $(\tilde{L}_\ell, \tilde{R}_\ell, \tilde{N}_\ell, \tilde{\tau}_\ell)_{\ell\in[L]}$ the reparametrization above is identical to feed-forward neural networks and by including the matrices $(\tilde{N}_\ell)_{\ell\in[L]}$ only extends this by a rescaling of the output neurons. The exact role of the matrices $(\tilde{N}_\ell)_{\ell\in[L]}$ will be more clear once we provide further context. The primary behind this reparametrization is stated in the following result.

Proposition 4.3 (Loss-free reparametrization of a network (cf. [50, Proposition 3]))**.** *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$ with weight matrices $W^{[1]},\dots,W^{[L]}$, and shifts $\tau_1,\dots,\tau_L$. Assume access to $\tilde{V}^{[1]},\dots,\tilde{V}^{[L]}$ as in* (4.61)*. Furthermore, assume* $\operatorname{rank}(\tilde{V}^{[\ell]}) = m_\ell$ *for all $\ell \in [L-1]$. Consider the neural network reparametrization $\tilde{f}$ as in Definition 4.4 w.r.t. the weights given by*

$$\tilde{W}^{[1]} = \tilde{V}^{[1]}, \quad \tilde{W}^{[\ell+1]} = (\tilde{V}^{[\ell]})^\dagger \tilde{V}^{[\ell+1]}, \tag{4.63}$$

and activations $(g_\ell)_{\ell\in[L]}$. Then, the following two sets of parameters $(\tilde{L}_\ell, \tilde{R}_\ell, \tilde{N}_\ell, \tilde{\tau}_\ell)_{\ell\in[L]}$ are well-defined and imply functional equivalence $\tilde{f}(\cdot, (\tilde{L}_\ell, \tilde{R}_\ell, \tilde{N}_\ell, \tilde{\tau}_\ell)_{\ell\in[L]}) \equiv \pi_L^\top f$ between the network reparametrization $\tilde{f}$ and the original network with permuted output $\pi_L^\top f$:

(Opt1) $\tilde{L}_1 = \mathrm{Id}_{m_0}$, $\tilde{N}_\ell = \mathrm{Id}_{m_\ell}$, $\tilde{L}_{\ell+1} = \pi_\ell^T D_\ell^{-1} \pi_\ell S_\ell$, $\tilde{R}_\ell = S_\ell^{-1}$, $\tilde{\tau}_\ell = \pi_\ell^\top \tau_\ell$

(Opt2) $\tilde{L}_1 = \mathrm{Id}_{m_0}$, $\tilde{N}_L = \mathrm{Id}_{m_L}$, $\tilde{L}_{\ell+1} = \pi_\ell^T D_\ell^{-1} \pi_\ell$, $\tilde{R}_\ell = S_\ell^{-1}$, $\tilde{N}_\ell = \tilde{R}_\ell^{-1} = S_\ell$, $\tilde{\tau}_\ell = \pi_\ell^\top \tau_\ell$

Since the permutation $\pi_L^\top$ is generally known for the above reasons, one can assume $\pi_L = \mathrm{Id}_{m_L}$.

Proof of Proposition 4.3. We show the functional equivalence of $\tilde{f}(\cdot, (\tilde{L}_\ell, \tilde{R}_\ell, \tilde{N}_\ell, \tilde{\tau}_\ell)_{\ell\in[L]})$ as defined in Definition 4.4 for the given sets of parameters via an induction over the post activations of f and $\tilde{f}$. We start with the first set of parameters in **(Opt1)** and show that $\tilde{y}^{[\ell]}(x) = \pi_\ell^\top y^{[\ell]}(x)$ for all $\ell \in [L]$. Since $\tilde{N}_\ell = \mathrm{Id}_{m_\ell}$ for all $\ell \in [L]$, we can neglect the parameters $(\tilde{N}_\ell)_{\ell\in[L]}$. For $\ell = 1$, we have $\tilde{W}^{[1]} = W^{[1]}\pi_1 S_1$ and therefore

$$\begin{aligned}\tilde{y}^{[1]}(x) &= g_1\left((\tilde{L}_1 \tilde{W}^{[1]} \tilde{R}_1)^\top x + \tilde{\tau}_1\right) = g_1\left((W^{[1]}\pi_1 S_1 S_1^{-1})^\top x + \pi_1^\top \tau_1\right)\\ &= \pi_1^\top g_1\left({W^{[1]}}^\top x + \tau_1\right) = \pi_1^\top y^{[1]}(x).\end{aligned}$$

Assume now that $\tilde{y}^{[\ell]}(x) = \pi_\ell^\top y^{[\ell]}(x)$ is true up to some $\ell < L$. Then

$$\begin{aligned}\tilde{y}^{[\ell+1]}(x) &= g_{\ell+1}\left((\tilde{L}_{\ell+1}\tilde{W}^{[\ell+1]}\tilde{R}_{\ell+1})^\top \tilde{y}^{[\ell]}(x) + \tilde{\tau}_{\ell+1}\right) \\ &= g_{\ell+1}\left((\pi_\ell^T D_\ell^{-1}\pi_\ell S_\ell (\tilde{V}^{[\ell]})^\dagger \tilde{V}^{[\ell+1]} S_{\ell+1}^{-1})^\top \pi_\ell^\top y^{[\ell]}(x) + \pi_{\ell+1}^\top \tau_{\ell+1}\right).\end{aligned}$$

Note that, by (4.61), we have $(\tilde{V}^{[\ell]})^\dagger \tilde{V}^{[\ell+1]} = S_\ell^{-1}\pi_\ell^T D_\ell W^{[\ell+1]}\pi_{\ell+1}S_{\ell+1}$. Plugging this into the expression above yields

$$\begin{aligned}\tilde{y}^{[\ell+1]}(x) &= g_{\ell+1}\left((\pi_\ell^T D_\ell^{-1}\pi_\ell S_\ell S_\ell^{-1}\pi_\ell^T D_\ell W^{[\ell+1]}\pi_{\ell+1}S_{\ell+1}S_{\ell+1}^{-1})^\top \pi_\ell^\top y^{[\ell]}(x) + \pi_{\ell+1}^\top\tau_{\ell+1}\right) \\ &= g_{\ell+1}\left((\pi_\ell^T W^{[\ell+1]}\pi_{\ell+1})^\top \pi_\ell^\top y^{[\ell]}(x) + \pi_{\ell+1}^\top\tau_{\ell+1}\right) \\ &= g_{\ell+1}\left(\pi_{\ell+1}^\top {W^{[\ell+1]}}^\top y^{[\ell]}(x) + \pi_{\ell+1}^\top\tau_{\ell+1}\right) = \pi_{\ell+1}^\top y^{[\ell+1]}(x),\end{aligned}$$

and therefore $\tilde{f}(x) = \tilde{y}^{[L]}(x) = \pi_L^\top y^{[L]}(x) = \pi_L^\top f(x)$. This shows the equivalence for the parameter set in **(Opt1)**. The second case follows by reusing the same argumentation. Assume the set of parameters given in **(Opt2)**. We show by induction that $\tilde{y}^{[\ell]}(x) = \tilde{N}_\ell \pi_\ell^\top y^{[\ell]}(x)$ for all $\ell \in [L]$. Again, for $\ell = 1$ we have $\tilde{W}^{[1]} = W^{[1]}\pi_1 S_1$ and therefore

$$\begin{aligned}\tilde{y}^{[1]}(x) &= \tilde{N}_1 g_1\left((\tilde{L}_1\tilde{W}^{[1]}\tilde{R}_1)^\top x + \tilde{\tau}_1\right) = \tilde{N}_1 g_1\left((W^{[1]}\pi_1 S_1 S^{-1})^\top x + \pi_1^\top\tau_1\right) \\ &= \tilde{N}_1\pi_1^\top g_1\left({W^{[1]}}^\top x + \tau_1\right) = \tilde{N}_1\pi_1^\top y^{[1]}(x).\end{aligned}$$

Assuming the statement is true up to some $\ell < L$, we now receive

$$\begin{aligned}\tilde{y}^{[\ell+1]}(x) &= \tilde{N}_{\ell+1} g_{\ell+1}\left((\tilde{L}_{\ell+1}\tilde{W}^{[\ell+1]}\tilde{R}_{\ell+1})^\top \tilde{N}_\ell\pi_\ell^\top y^{[\ell]}(x) + \tilde{\tau}_{\ell+1}\right) \\ &= \tilde{N}_{\ell+1} g_{\ell+1}\left((\tilde{L}_{\ell+1}\tilde{W}^{[\ell+1]}\tilde{R}_{\ell+1})^\top S_\ell\pi_\ell^\top y^{[\ell]}(x) + \tilde{\tau}_{\ell+1}\right) \\ &= \tilde{N}_{\ell+1}\pi_{\ell+1} y^{[\ell+1]}(x).\end{aligned}$$

The last step follows from the previous induction for the parameters in **(Opt1)** since now

$$(\tilde{L}_{\ell+1}\tilde{W}^{[\ell+1]}\tilde{R}_{\ell+1})^\top S_\ell = \tilde{R}_{\ell+1}^\top(\tilde{W}^{[\ell+1]})^\top \pi_\ell^T D_\ell^{-1}\pi_\ell S_\ell = \tilde{R}_{\ell+1}^\top(\tilde{W}^{[\ell+1]})^\top S_\ell \pi_\ell^T D_\ell^{-1}\pi_\ell,$$

where the last step used that $\pi_\ell^T D_\ell^{-1}\pi_\ell$ is diagonal and therefore commutes with S_ℓ (cf. Lemma 4.5). Since $S_\ell\pi_\ell^T D_\ell^{-1}\pi_\ell$ is identical to $\tilde{L}_{\ell+1}^\top$ from **(Opt1)**, the statement in the induction step is justified. Since we assumed $\tilde{N}_L = \mathrm{Id}_{m_L}$, this equates to $\tilde{f}(x) = \tilde{y}^{[L]}(x) = \tilde{N}_L\pi_L^\top y^{[L]}(x) = \pi_L^\top f(x)$. □

4.3.3 Learning the parameters of the reparametrized network

Let us now return to our original objective and how the reparametrization is used in the overall reconstruction procedure. Provided access to entangled weights matrices of the type (4.61) with full column rank, we consider the network reparametrization $\tilde{f}$ as defined in Definition 4.4 with weights

$$\tilde{W}^{[1]} = \tilde{V}^{[1]}, \quad \tilde{W}^{[\ell+1]} = (\tilde{V}^{[\ell]})^\dagger \tilde{V}^{[\ell+1]}.$$

One approach to finding the right set of parameters for $\tilde{f}$ is to fit the reparametrization to samples of the target network f. Assume we are given N training pairs $(x_i, f(x_i))_{i\in[N]}$, then

Proposition 4.3 shows that there exist parameter combinations such that $\tilde{f}$ is identical to f on these data points. Hence, the least squares problem

$$\operatorname{argmin}_{(\tilde{L}_\ell,\tilde{R}_\ell,\tilde{N}_\ell,\tilde{\tau}_\ell)_{\ell\in[L]}} \sum_{i=1}^{N} \Big(f(x_i) - \tilde{f}(x_i,(\tilde{L}_\ell,\tilde{R}_\ell,\tilde{N}_\ell,\tilde{\tau}_\ell)_{\ell\in[L]})\Big)^2 \tag{4.64}$$

can be realized with global minimum 0 for the parameter combination given in Proposition 4.3. Furthermore, as became clear in Proposition 4.3, optimizing over the parameters $(\tilde{N}_\ell)_{\ell\in[L]}$ is not necessary because we can consider one of the slightly simplified versions given either by

$$\operatorname{argmin}_{(\tilde{L}_\ell,\tilde{R}_\ell,\tilde{\tau}_\ell)_{\ell\in[L]}} \sum_{i=1}^{N} \Big(f(x_i) - \tilde{f}(x_i,(\tilde{L}_\ell,\tilde{R}_\ell,\mathrm{Id}_{m_\ell},\tilde{\tau}_\ell)_{\ell\in[L]})\Big)^2, \tag{4.65}$$

which corresponds to the ideal solution **(Opt1)**, or in line with the parameter combination described in **(Opt2)** by

$$\operatorname{argmin}_{(\tilde{L}_\ell,\tilde{R}_\ell,\tilde{\tau}_\ell)_{\ell\in[L]}} \sum_{i=1}^{N} \Big(f(x_i) - \tilde{f}(x_i,(\tilde{L}_\ell,\tilde{R}_\ell,h_\ell(\tilde{R}_\ell),\tilde{\tau}_\ell)_{\ell\in[L]})\Big)^2, \tag{4.66}$$

with $h_\ell(\tilde{R}_\ell) = \tilde{R}_\ell^{-1}$ for $\ell < L$ and $h_L(\tilde{R}_\ell) = \mathrm{Id}_{m_L}$. Similar to the approach in Chapter 3 (cf. Section 3.4.1), this closely resembles the teacher-student scenario, where f being regarded as the teacher network and $\tilde{f}$ corresponds to the student network. In both minimization programs above, the student has considerably fewer degrees of freedom than we had parameters in the network f. The parameters that have to be tuned in either (4.65) or (4.66) are the diagonal matrices and shifts $(\tilde{L}_\ell,\tilde{R}_\ell,\tilde{\tau}_\ell)_{\ell\in[L]}$, such that the overall degrees of freedom are at most $\mathcal{O}(m)$, where $m = m_1 + \cdots + m_L$. This follows from

$$\sum_{\ell=1}^{L} \dim(\mathrm{diag}(\tilde{L}_\ell) + \dim(\mathrm{diag}(\tilde{R}_\ell)) + \dim(\tilde{\tau}_\ell) = \sum_{\ell=1}^{L} m_{\ell-1} + m_\ell + m_\ell = 3m - m_L + D,$$

where we used $m_0 = D$. This has to be put in contrast to the number of parameters in the original network f, which is given by

$$\sum_{\ell=1}^{L} \dim(W^{[\ell]}) + \dim(\tau_\ell) = \sum_{\ell=1}^{L} m_{\ell-1}\cdot m_\ell + m_\ell.$$

Fitting the reparametrization by empirical risk minimization. Let us now discuss some practical aspects of an optimization strategy based on gradient ascent. Assume again that we are given N_{GD} random network evaluations $(x_i, f(x_i))_{i\in[N_{\mathrm{GD}}]}$ where $x_1,\ldots,x_{N_{\mathrm{GD}}} \sim_{\text{i.i.d.}} \mathcal{N}(0,\mathrm{Id}_D)$. We construct $\tilde{f}$ as the reparametrization of this network w.r.t. the entangled weights up to permutation and scales, i.e., we have

$$\tilde{W}^{[1]} = \tilde{V}^{[1]}, \quad \tilde{W}^{[\ell+1]} = (\tilde{V}^{[\ell]})^\dagger \tilde{V}^{[\ell+1]}.$$

We assume the rank assumption on the entangled weights is met (see the discussion above). Since the right set of parameters (e.g., the parameters in **(Opt2)** of Proposition 4.3) for $\tilde{f}$ is not known, we initialize $\tilde{f}$ with initial parameters $(\tilde{L}_\ell^{[0]},\tilde{R}_\ell^{[0]},h_\ell(\tilde{R}_\ell^{[0]}),\tilde{\tau}_\ell^{[0]})_{\ell\in[L]}$, where $h_\ell(\tilde{R}_\ell^{[0]}) = \left(\tilde{R}_\ell^{[0]}\right)^{-1}$ for $\ell < L$ and $h_L(\tilde{R}_\ell^{[0]}) = \mathrm{Id}_{m_L}$. We consider the objective inherent to (4.66), which is

$$\mathcal{L}((\tilde{L}_\ell,\tilde{R}_\ell,\tilde{\tau}_\ell)_{\ell\in[L]}) := \sum_{i=1}^{N} \Big(f(x_i) - \tilde{f}(x_i,(\tilde{L}_\ell,\tilde{R}_\ell,h_\ell(\tilde{R}_\ell),\tilde{\tau}_\ell)_{\ell\in[L]})\Big)^2. \tag{4.67}$$

A classical gradient descent method ran with a step-size $\gamma > 0$ then produces an iteration $((\tilde{L}_\ell^{[j]}, \tilde{R}_\ell^{[j]}, \tilde{\tau}_\ell^{[j]})_{\ell\in[L]})_{j\in\mathbb{N}}$ described by

$$\begin{aligned}
\tilde{L}_\ell^{[j+1]} &= \tilde{L}_\ell^{[j]} - \gamma \operatorname{diag}\left(\nabla_{\operatorname{diag}\tilde{L}_\ell^{[j]}} \mathcal{L}\left((\tilde{L}_\ell^{[j]}, \tilde{R}_\ell^{[j]}, \tilde{\tau}_\ell^{[j]})_{\ell\in[L]}\right)\right),\\
\tilde{R}_\ell^{[j+1]} &= \tilde{R}_\ell^{[j]} - \gamma \operatorname{diag}\left(\nabla_{\operatorname{diag}\tilde{R}_\ell^{[j]}} \mathcal{L}\left((\tilde{L}_\ell^{[j]}, \tilde{R}_\ell^{[j]}, \tilde{\tau}_\ell^{[j]})_{\ell\in[L]}\right)\right),\\
\tilde{\tau}_\ell^{[j+1]} &= \tilde{\tau}_\ell^{[j]} - \gamma \nabla_{\tilde{\tau}_\ell^{[j]}} \mathcal{L}\left((\tilde{L}_\ell^{[j]}, \tilde{R}_\ell^{[j]}, \tilde{\tau}_\ell^{[j]})_{\ell\in[L]}\right),
\end{aligned}$$

for all $\ell \in [L]$. Notably, we only need to run gradient descent over the diagonal elements of the involved matrices. Therefore, in the iteration above, we consider only the gradients corresponding to the diagonal and then map the gradient updates back to the matrix form. This way, it is guaranteed that all matrices remain diagonal. Proposition 4.3 shows that the objective (4.67) has a global minimum for shifts $\tilde{\tau}_\ell = \pi_\ell^\top \tau_\ell$ and diagonal matrices

$$\tilde{L}_1 = \mathrm{Id}_{m_0}, \tilde{R}_L = S_L^{-1}, \tilde{L}_\ell = \pi_\ell^T D_{\ell-1}^{-1} \pi_\ell,\ \tilde{R}_{\ell-1} = S_{\ell-1}^{-1} \quad \text{for } \ell = 2, \dots, L \tag{4.68}$$

where D_ℓ, S_ℓ are non-zero diagonal matrices, and π_ℓ are unknown permutations, all of which originate from the unknown parameters in the entangled weight matrices described in (4.61) as

$$\tilde{V}^{[\ell]} = \left(\Pi_{k=1}^{\ell-1} W^{[k]} D_k\right) W^{[\ell]} \pi_\ell S_\ell \in \mathbb{R}^{m_0 \times m_\ell}, \quad \text{for all } \ell \in [L].$$

To reach this global minimum, we rely on gradient descent started from $(\tilde{L}_\ell^{[0]}, \tilde{R}_\ell^{[0]}, \tilde{\tau}_\ell^{[0]})_{\ell\in[L]}$. Without any prior information or initialization strategy, we will generally initialize these parameters by identity matrices and zero vectors in case of shifts, i.e., we set

$$(\tilde{L}_\ell^{[0]}, \tilde{R}_\ell^{[0]}, \tilde{\tau}_\ell^{[0]})_{\ell\in[L]} = (\mathrm{Id}_{m_{\ell-1}}, \mathrm{Id}_{m_\ell}, 0_{m_\ell})_{\ell\in[L]}. \tag{4.69}$$

Now, can gradient descent or any of its variants reach the global minima started from these initial values when granted sufficiently many training points? Empirically, we observe that minimizing (4.67) via gradient descent yields good results for sigmoidal neural networks, approximated entangled weights, and a moderate number of training samples (cf. Section 4.4). Admittedly, we do not offer any theoretical convergence analysis of the gradient descent iteration defined above. The significant reduction of the degrees of freedom should, in principle, lead to an improved algorithmic complexity when compared to learning the full network via empirical risk minimization. A theoretical analysis of the optimization landscape suffers from similar problems present when learning ordinary deep neural networks in a teacher-student setting. In Chapter 3, we provide a rigorous local convergence analysis for a similar type of problem for shallow neural networks. Clearly, an extension of the results in Section 3.4 to the present case is non-trivial. One crucial aspect in Section 3.4 was the initialization of the gradient descent iteration. Proposition 4.3 guarantees the existence of global minima, however, the initialization in (4.69) will generally not lie in the vicinity of these global optima. This raises the question of why the naive initialization in (4.69) still yields a consistent recovery of the original network. In the following, we will give one potential explanation for this phenomenon.

Inherent symmetries of sigmoidals lead to multiple global minima. Let us now discuss one crucial point related to the distance of the initialization (4.69) from the global minima **(Opt2)** in Proposition 4.3. The first quantity in **(Opt2)** we need to discuss concerns the non-zero diagonal matrices $(S_\ell)_{\ell\in[L]}$, which model the unknown sign and scale information. We can assume that the diagonal elements of the matrices are equally distributed over the positive

and negative parts of the real numbers due to the equal probability of guessing the sign wrong as a consequence of the SPM (Algorithm 2.2). This implies, that there will be indices $\ell \in [L], k \in [m_\ell]$ such that $S_{k,\ell} < 0$. For the sake of simplicity, assume that $S_{1,1} = -1$, which also implies that $S_{1,1}^{-1} = -1$. On the other hand, we initialized the corresponding diagonal matrix by the identity, such that $\tilde{R}_{1,1}^{[0]} = 1$. Hence, to reach the optimal solution (4.68), the gradient descent scheme described above needs to produce an iteration $\tilde{R}_{1,1}^{[j]} \to S_{1,1}^{-1} = -1$. Regardless of the trajectory, if we want to reach this point and assume a realistically small step size γ, there needs to be a number j' of gradient steps such that $\tilde{R}_{1,1}^{[j']} \approx 0$. Notably, this implies that $h_1(\tilde{R}_{1,1}^{[j']}) = (\tilde{R}_{1,1}^{[j']})^{-1}$ is large. This behavior can have negative consequences on the reparametrization since its pre-activations at this iteration read as

$$\begin{aligned}\tilde{y}^{[\ell]}(x) &= \tilde{N}_\ell^{[j']}\, g_\ell\left((\tilde{L}_\ell^{[j']}\tilde{W}^{[\ell]}\tilde{R}_\ell^{[j']})^\top \tilde{y}^{[\ell-1]}(x) + \tilde{\tau}_\ell^{[j']}\right)\\ &= h_\ell(\tilde{R}_\ell^{[j']})\, g_\ell\left((\tilde{L}_\ell^{[j']}\tilde{W}^{[\ell]}\tilde{R}_\ell^{[j']})^\top \tilde{y}^{[\ell-1]}(x) + \tilde{\tau}_\ell^{[j']}\right)\\ &= (\tilde{R}_\ell^{[j']})^{-1}\, g_\ell\left((\tilde{L}_\ell^{[j']}\tilde{W}^{[\ell]}\tilde{R}_\ell^{[j']})^\top \tilde{y}^{[\ell-1]}(x) + \tilde{\tau}_\ell^{[j']}\right)\end{aligned}$$

for all $\ell = 1, \dots, L-1$. Therefore, the components of the inner pre-activations get amplified by the factor $(\tilde{R}_{1,1}^{[j']})^{-1}$, which would likely increase the mismatch between f and the reparametrization $\tilde{f}$. Fully addressing this dynamic goes beyond the scope of this discussion, but a point can be made that the adjustments of the signs might act as a barrier for gradient descent because it needs to pass through a domain where the least squares error $\mathcal{L}$ in (4.67) is most likely large. Hence, a natural question is why we do not experience this problem during our empirical study on deep sigmoidal neural networks. The answer to this question is that the inherent symmetries of classical sigmoidal functions (e.g., tanh) give rise to multiple global minima, which we will prove in the following. First, note that according to Proposition 4.3, one global minimum of the minimization program (4.66) is given by the set of parameters in **(Opt2)**. More precisely, by the set of shifts $\tilde{\tau}_\ell = \pi_\ell^\top \tau_\ell$ and diagonal matrices

$$\tilde{L}_1 = \mathrm{Id}_{m_0}, \tilde{R}_L = S_L^{-1}, \tilde{L}_\ell = \pi_\ell^T D_{\ell-1}^{-1}\pi_\ell,\ \tilde{R}_{\ell-1} = S_{\ell-1}^{-1} \quad \text{for } \ell = 2, \dots, L$$

where D_ℓ, S_ℓ are non-zero diagonal matrices and π_ℓ are unknown permutations, all of which originate from the unknown parameters in the entangled weight matrices described in (4.61) as

$$\tilde{V}^{[\ell]} = \left(\Pi_{k=1}^{\ell-1} W^{[k]} D_k\right) W^{[\ell]} \pi_\ell S_\ell \in \mathbb{R}^{m_0 \times m_\ell}, \quad \text{for all } \ell \in [L].$$

Let us assume that the underlying network uses tanh activations at all hidden neurons and has either linear output neurons or $g_L = \tanh$. At the beginning of this work, namely Section 1.2, it was already mentioned that tanh is antisymmetric; thereby, the network remains functionally invariant under certain sign-flip operations of the parameters. The example given above for a shallow neural network was

$$\sum_{k=1}^{m} \tanh(w_k x + \tau_k) = \sum_{k=1}^{m} -\tanh(-w_k x - \tau_k).$$

This symmetry gives rise to multiple global minima for the objective in (4.67) and allows us to extend the result of Proposition 4.3 in the sense that it can be shown that we can morally neglect the signs of the diagonal matrices $(\tilde{R}_\ell)_{\ell \in L}$ in the optimal solution **(Opt2)** except for the signs corresponding to the last layer. Before we state the result, let us note that any linear function can be composed of an antisymmetric function and a constant shift. Hence, the theory below does cover networks with linear outputs. Additionally, we introduce the following notation. For any matrix M with entries given by m_{ij}, we denote by $\mathrm{sign}(M)$ the matrix with entries given by $\mathrm{sign}(m_{ij})$, and by $|M|$ the matrix with entries given by $|m_{ij}|$.

Proposition 4.4. *Consider a neural network $f \in \mathcal{NN}(D, (m_\ell)_{\ell\in[L]}, (g_\ell)_{\ell\in[L]})$ with weight matrices $W^{[1]}, \dots, W^{[L]}$, shifts $\tau_1, \dots, \tau_L$ and antisymmetric activations $(g_\ell)_{\ell\in[L]}$. Assume access to $\tilde{V}^{[1]}, \dots, \tilde{V}^{[L]}$ as in* (4.61)*. Furthermore, assume* $\operatorname{rank}(\tilde{V}^{[\ell]}) = m_\ell$ *for all $\ell \in [L-1]$. Consider the neural network reparametrization $\tilde{f}$ as in Definition 4.4 w.r.t. the weights given by*

$$\tilde{W}^{[1]} = \tilde{V}^{[1]}, \quad \tilde{W}^{[\ell+1]} = (\tilde{V}^{[\ell]})^\dagger \tilde{V}^{[\ell+1]}, \tag{4.70}$$

and parameters $(\tilde{L}_\ell, \tilde{R}_\ell, h_\ell(\tilde{R}_\ell), \tilde{\tau}_\ell)_{\ell\in[L]}$ where $h_\ell(\tilde{R}_\ell) = \tilde{R}_\ell^{-1}$ for $\ell < L$ and $h_L(\tilde{R}_\ell) = \mathrm{Id}_{m_L}$. If $\tilde{L}_1 = \mathrm{Id}_{m_0}, \tilde{L}_{\ell+1} = \pi_\ell^T D_\ell^{-1} \pi_\ell, \tilde{\tau}_\ell = \operatorname{sign}(\tilde{R}_\ell S_\ell^{-1}) \pi_\ell^\top \tau_\ell$ and diagonal matrices $(\tilde{R}_\ell)_{\ell\in[L]}$ are such that

$$\left|\tilde{R}_\ell\right| = \left|S_\ell^{-1}\right|, \tag{4.71}$$

then the reparametrization is functionally equivalent to the permuted network up to a sign-flip of the output $\tilde{f} \equiv \operatorname{sign}(\tilde{R}_L S_L) \pi_L^\top f$. Note: Since the permutation $\pi_L^\top$ is generally known for the above reasons, one can generally assume $\pi_L = \mathrm{Id}_{m_L}$ w.o.l.g.

Proof. Since $\tilde{R}_\ell, S_\ell$ are non-zero diagonal matrices for all $\ell \in [L]$, we have

$$\left|\tilde{R}_\ell\right| = \left|S_\ell^{-1}\right| \Leftrightarrow \operatorname{sign}(\tilde{R}_\ell)\tilde{R}_\ell = \operatorname{sign}(S_\ell^{-1}) S_\ell^{-1} \Leftrightarrow \tilde{R}_\ell = \operatorname{sign}(\tilde{R}_\ell S_\ell^{-1}) S_\ell^{-1}, \tag{4.72}$$

and also $\operatorname{sign}(\tilde{R}_\ell S_\ell^{-1})\tilde{R}_\ell^{-1} = S_\ell$. Recall that the pre-activations of the reparametrization given by

$$\tilde{y}^{[\ell]}(x) = h_\ell(\tilde{R}_\ell)\, g_\ell\left((\tilde{L}_\ell \tilde{W}^{[\ell]} \tilde{R}_\ell)^\top \tilde{y}^{[\ell-1]}(x) + \tilde{\tau}_\ell\right)$$

for all $\ell \in [L]$ where $\tilde{y}^{[0]}(x) = x$. As before, we prove by induction that

$$\tilde{y}^{[\ell]}(x) = h_\ell(\tilde{R}_\ell) \operatorname{sign}(\tilde{R}_\ell S_\ell^{-1}) \pi_\ell^\top y^{[\ell]}(x). \tag{4.73}$$

Assume $\ell = 1$: We have $\tilde{W}^{[1]} = W^{[1]} \pi_1 S_1$ and therefore

$$\begin{aligned}
\tilde{y}^{[1]}(x) &= h_1(\tilde{R}_1)\, g_1\left((\tilde{L}_1 \tilde{W}^{[1]} \tilde{R}_1)^\top x + \tilde{\tau}_1\right) \\
&= h_1(\tilde{R}_1)\, g_1\left((W^{[1]} \pi_1 S_1 S_1^{-1} \operatorname{sign}(\tilde{R}_1 S_1^{-1}))^\top x + \tilde{\tau}_1\right) \\
&= h_1(\tilde{R}_1) \operatorname{sign}(\tilde{R}_1 S_1^{-1}) \pi_1^\top g_1\left((W^{[1]})^\top x + \tau_1\right) \\
&= h_1(\tilde{R}_1) \operatorname{sign}(\tilde{R}_1 S_1^{-1}) \pi_1^\top y^{[1]}(x).
\end{aligned}$$

Assume now (4.73) holds up to some $\ell \leq L-1$. Since $h_\ell(\tilde{R}_\ell) = \tilde{R}_\ell^{-1}$ for all $\ell < L$, this is equivalent to

$$\tilde{y}^{[\ell]}(x) = \tilde{R}_\ell^{-1} \operatorname{sign}(\tilde{R}_\ell S_\ell^{-1}) \pi_\ell^\top y^{[\ell]}(x) = S_\ell \pi_\ell^\top y^{[\ell]}(x), \tag{4.74}$$

which follows directly from (4.72) together with the fact that the sign matrices of diagonal matrices are invariant under inversion. Note that, by (4.61), we have

$$\tilde{W}^{[\ell+1]} = (\tilde{V}^{[\ell]})^\dagger \tilde{V}^{[\ell+1]} = S_\ell^{-1} \pi_\ell^T D_\ell W^{[\ell+1]} \pi_{\ell+1} S_{\ell+1}.$$

The induction steps follows from

$$\begin{aligned}
\tilde{y}^{[\ell+1]}(x) &= h_{\ell+1}(\tilde{R}_{\ell+1})\, g_{\ell+1}\left((\tilde{L}_{\ell+1} \tilde{W}^{[\ell+1]} \tilde{R}_{\ell+1})^\top \tilde{y}^{[\ell]}(x) + \tilde{\tau}_{\ell+1}\right) \\
&= h_{\ell+1}(\tilde{R}_{\ell+1})\, g_{\ell+1}\left((\tilde{L}_{\ell+1} \tilde{W}^{[\ell+1]} \tilde{R}_{\ell+1})^\top S_\ell \pi_\ell^\top y^{[\ell]}(x) + \tilde{\tau}_{\ell+1}\right) \\
&= h_{\ell+1}(\tilde{R}_{\ell+1})\, g_{\ell+1}\left((\tilde{L}_{\ell+1} S_\ell^{-1} \pi_\ell^T D_\ell W^{[\ell+1]} \pi_{\ell+1} S_{\ell+1} \tilde{R}_{\ell+1})^\top S_\ell \pi_\ell^\top y^{[\ell]}(x) + \tilde{\tau}_{\ell+1}\right) \\
&= h_{\ell+1}(\tilde{R}_{\ell+1})\, g_{\ell+1}\left((\tilde{L}_{\ell+1} S_\ell^{-1} \pi_\ell^T D_\ell W^{[\ell+1]} \pi_{\ell+1} \operatorname{sign}(S_{\ell+1} \tilde{R}_{\ell+1}))^\top S_\ell \pi_\ell^\top y^{[\ell]}(x) + \tilde{\tau}_{\ell+1}\right) \\
&= \operatorname{sign}(S_{\ell+1} \tilde{R}_{\ell+1}) h_{\ell+1}(\tilde{R}_{\ell+1})\, g_{\ell+1}\left((\tilde{L}_{\ell+1} S_\ell^{-1} \pi_\ell^T D_\ell W^{[\ell+1]} \pi_{\ell+1})^\top S_\ell \pi_\ell^\top y^{[\ell]}(x) + \pi_{\ell+1}^\top \tau_{\ell+1}\right).
\end{aligned}$$

The inner term can be simplified with

$$\begin{aligned}(\tilde{L}_{\ell+1}S_\ell^{-1}\pi_\ell^\top D_\ell W^{[\ell+1]}\pi_{\ell+1})^\top S_\ell &= (\tilde{L}_{\ell+1}S_\ell S_\ell^{-1}\pi_\ell^\top D_\ell W^{[\ell+1]}\pi_{\ell+1})^\top \\ &= (\pi_\ell^\top D_\ell^{-1}\pi_\ell\pi_\ell^\top D_\ell W^{[\ell+1]}\pi_{\ell+1})^\top = (\pi_\ell^\top W^{[\ell+1]}\pi_{\ell+1})^\top.\end{aligned}$$

Therefore, we get

$$\begin{aligned}\tilde{y}^{[\ell+1]}(x) &= \operatorname{sign}(S_{\ell+1}\tilde{R}_{\ell+1})h_{\ell+1}(\tilde{R}_{\ell+1})g_{\ell+1}\left((\pi_\ell^\top W^{[\ell+1]}\pi_{\ell+1})^\top\pi_\ell^\top y^{[\ell]}(x)+\pi_{\ell+1}^\top\tau_{\ell+1}\right)\\ &= \operatorname{sign}(S_{\ell+1}\tilde{R}_{\ell+1})h_{\ell+1}(\tilde{R}_{\ell+1})\pi_{\ell+1}^\top g_{\ell+1}\left((W^{[\ell+1]})^\top y^{[\ell]}(x)+\tau_{\ell+1}\right)\\ &= \operatorname{sign}(\tilde{R}_{\ell+1}S_{\ell+1}^{-1})\pi_{\ell+1}^\top y^{[\ell+1]}(x).\end{aligned}$$

This concludes the induction and therefore implies

$$\tilde{f}(x) = \tilde{y}^{[\ell+1]}(x) = \operatorname{sign}(\tilde{R}_{\ell+1}S_{\ell+1}^{-1})\pi_{\ell+1}^\top f(x)$$

for any input $x \in \mathbb{R}^D$. □

Remark 4.4. *Note that a similar statement can be proven for other common activation functions of sigmoidal type, even if they are not entirely antisymmetric. Take for instance the sigmoid (or logistic) function* $\operatorname{sigmoid}(x) = \frac{1}{1+e^{-x}}$. *Then* $\operatorname{sigmoid}(x)$ *fulfills the identity*

$$\operatorname{sigmoid}(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x} = 1 - \operatorname{sigmoid}(-x), \tag{4.75}$$

and therefore $\operatorname{sigmoid}(\cdot) - 0.5$ *is antisymmetric. The constant offset can always be absorbed in the shift parameters.*

4.4 Experiments: Reconstruction of deep neural networks

In this final numerical section, we apply our complete reconstruction pipeline to sigmoidal neural networks of different architectures. We provide an analysis of each individual algorithmic step. Additionally, we compare the performance to our empirical analysis of the teacher-student problem, where the reconstruction problem is tackled by ordinary SGD (cf. Section 1.3.2).

Experimental setup. The pipeline first recovers approximations to the entangled weights via Algorithm 4.1. Notably, we slightly deviate from Algorithm 4.1 and run several independent repetitions of SPM simultaneously for a fixed number of steps (see the related discussion at the beginning of Section 4.3.1). In the second step, the entangled weight matrices are built by the post-processing and layer assignment in Algorithm 4.2. As the last step, a reparametrization $\tilde{f}$ as in Definition 4.4 is then built from these entangled matrices and initialized as in (4.69). The reparametrization $\tilde{f}$ is then fit onto the original network using stochastic gradient descent with step size $\gamma > 0$ as described in Section 4.3.3. Every metric reported below is averaged over ten independent runs for the given parameter combination. We benchmark our algorithmic pipeline on three different network architectures with ambient dimension $D = 50$: two three-layer networks with neurons given by $[50, 25, 10]$, $[50, 25, 50]$, and one four-layer network with architecture $[50, 35, 25, 10]$. For the four-layer networks, we need to rely on an oracle assignment of the entangled weights to their corresponding layers, as this step has not been extended to $L > 3$ yet. This setup demonstrates that the procedure works for deeper ($L > 3$) architectures provided the correct layer assignment. All networks use tanh-activations at all hidden neurons,

and we consider two types of output activations given by $g_L = \tanh$ and $g_L = \mathrm{Id}$. Sigmoidal output neurons are common for networks used in classification problems, whereas linear output neurons correspond to regression problems, making the distinction between the two output activations interesting. Every weight will be drawn uniformly from the unit sphere, i.e., $w_{k_\ell}^{[\ell]} \sim \mathrm{Unif}(\mathbb{S}^{m_{\ell-1}-1})$ for all $k_\ell \in [m_\ell], \ell \in [L]$, and all neurons will be equipped with a small shift $\tau_{k_\ell}^{[\ell]} \sim \mathcal{N}(0, 0.05)$. Additionally, for every problem instance, the pipeline is run once with exact derivatives (gradients and Hessians) and once with numerical approximations. Whenever numerical gradients or numerical Hessians are used (this affects Algorithm 4.1 and Algorithm 4.2), we rely on a simple schema given by central finite differences (which adheres to the general assumptions that were made in **(G5.3)**). More precisely, for any vector-valued input $x \in \mathbb{R}^D$ and output index $k_L \in [m_L]$, the approximations $\Delta f_{k_L}(x) \approx \nabla f_{k_L}(x), \Delta^2 f_{k_L}(x) \approx \nabla^2 f_{k_L}(x)$ are computed component-wise via

$$\Delta f_{k_L}(x)_i = \frac{f_{k_L}(x + \epsilon e_i) - f_{k_L}(x + \epsilon e_{-i})}{2\epsilon} \tag{4.76}$$

$$\Delta^2 f_{k_L}(x)_{ij} = \frac{f_{k_L}(x + \epsilon e_{i,j}) - f_{k_L}(x + \epsilon e_{-i,j}) - f_{k_L}(x + \epsilon e_{i,-j}) + f_{k_L}(x - \epsilon e_{-i,-j})}{4\epsilon^2} \tag{4.77}$$

for all $i, j \in [D]$, respectively. Here, ϵ is the step-size parameter that is set to 0.005, e_i denotes i-th the canonical basis vector and $e_{i,j} = e_i + e_j, e_{-i,j} = -e_i + e_j, e_{i,-j} = e_i - e_j, e_{-i,-j} = -e_i - e_j$. For the reader's convenience, we summarized all hyperparameters and the network model in Table 4.1. We will only address our choice for the hyperparameters shortly. For a detailed discussion of these parameters, we refer to the respective sections where the Algorithm was introduced.

Discussion of relevant hyperparameters. For the entangled weight recovery, we choose the mean-zero distribution $\mu_X = \frac{1}{10\sqrt{D}} \cdot \mathcal{N}(0, \mathrm{Id})$, such that every input vector is a standard Gaussian scaled by the factor $\frac{1}{10\sqrt{D}}$. Our previous results in Section 4.2.3 motivate this scaling, where the subspace was best recovered for inputs drawn uniformly from the sphere with radius 0.1. Hence, while μ_X is chosen as a normal distribution, its sub-Gaussian norm is equivalent to before (0.1). We sample $4m$ Hessian locations from this distribution, such that the overall number of network queries needed to approximate the space, when numerical Hessians are used, is $4m \cdot 2D(D+1) = \mathcal{O}(mD^2)$. Note that $2D(D+1)$ is the number of samples necessary to compute one Hessian matrix via (4.77). Then, we repeat SPM for $N_{\mathrm{SPM}} = 5m \log m$ random initializations performing $N_{\mathrm{PGA}} = 10^4$ projected gradient ascent steps each time (i.e., steps of SPM). This is a slight deviation from Algorithm 4.1. We refer to the related discussion at the beginning of Section 4.3.1 for our motivation behind this decision. For the second step, the reconstruction of the entangled weight matrices performed by Algorithm 4.2, we choose large inputs to detect the first layer. The inputs are drawn independently from $\mu_X = \mathrm{Unif}(10^3 \cdot \mathbb{S}^{D-1})$. Again, we rely on $4m$ Hessians to compute the space spanned by the outer-product of the first layer entangled weights. Lastly, using empirical risk minimization, we fit the reparametrization described in Section 4.3.2 onto the target network. Here, we construct the reparametrization $\tilde{f}$ of f using the approximating entangled weight matrices. We minimize the objective (4.67) via SGD with a learning rate of $\gamma = 0.005$ and batch size 64. We provide SGD with $N_{\mathrm{SGD}} = mD^2$ training samples in the form of input-output pairs of the network we seek to reconstruct and stop the training after 20 minutes. All experiments are repeated 10 times for each combination of the network architecture, method of differentiation (exact or by a numerical approximation), and output activation g_L. During each execution of our pipeline, we record the outputs of the involved algorithms and will evaluate their performance separately.

Network Model	Entangled weight recovery (see Alg. 4.1)
Architectures: $[50,25,10]$, $[50,25,50]$, $[50,35,25,10]$	μ_X: distribution for Hessians $\mathcal{N}(0, \frac{1}{10^2 D}\,\mathrm{Id})$
$W^{[\ell]}$: random weights $w_{k_\ell}^{[\ell]} \sim \mathrm{Unif}(\mathbb{S}^{m_{\ell-1}-1})$	N_h: nr. of sampled Hessians (max. $4m$)
$\tau^{[\ell]}$: Gaussian shifts $\tau_{k\ell}^{\ell} \sim \mathcal{N}(0, 0.05)$	N_{SPM} : restarts of SPM ($\leq 5m \log m$)
g_ℓ: hidden layer activation $g_\ell = \tanh$ for $\ell < L$	N_{PGA} : number of projected GA steps (10^4)
g_L : output activation $g_L \in \{\mathrm{Id}, \tanh\}$	ϵ : step-size for finite differences (0.005)
Constructing ent. weight matrices (see Alg. 4.2)	**Network completion (see Section 4.3.2)**
N_h: Hessians used to detect 1st layer (max. $4m$)	γ: learning rate of SGD (0.05)
R : radius used to detect 1st layer (10^3)	batch size (64)
ϵ : step-size for finite differences (0.005)	N_{SGD} : number of training points ($D^2 m$)
	training time (20 minutes)

Table 4.1: Summary of all hyperparameters for the numerical experiments. A single value in $(\cdot)$-brackets is the default hyperparameter used in all experiments. Otherwise, we list ranges that are tested in experiments.

Step 1: Analysis of Algorithm 4.1 To evaluate the output of Algorithm 4.1, we will rely on slightly different metrics than before (cf. Section 4.2.3): The accuracy of the subspace approximation is measured by the Frobenius distance between the orthogonal projections of the vectorized subspaces denoted by $\Delta_{\widehat{\mathcal{W}}} := \|\mathcal{P}_{\mathcal{W}} - \mathcal{P}_{\widehat{\mathcal{W}}}\|_F$, which serves as an upper bound to the error estimated in Theorem 4.1. For a given output $\mathcal{U} = \mathcal{U}_{\mathrm{SPM}} \cup \mathcal{U}_L$ of Algorithm 4.1, we track three metrics: First, the worst-case L^2-error (up to sign) of the approximations within $\mathcal{U}$ to any (normalized) entangled weight corresponding to the ℓ-th layer defined by

$$E_\ell(\mathcal{U}) = \max_{k_\ell \in [m_\ell]} \min_{s \in \{-1,+1\}, u \in \mathcal{U}} \|u - s v_{k_\ell}^{[\ell]} / \|v_{k_\ell}^{[\ell]}\|_2\|_2. \tag{4.78}$$

At this stage, we exclusively focus on the error of the vectors that were computed directly via SPM ($E_\ell(\mathcal{U}_{\mathrm{SPM}})$)and exclude the set $\mathcal{U}_L$ that contains the entangled weights of the last layer which were computed via gradients. Hence, $E_\ell(\mathcal{U}_{\mathrm{SPM}})$ provides an upper bound to how well the entirety of the entangled weights in layer ℓ was approximated by SPM. Second, we calculate the ratio of false positives within $\mathcal{U}_{\mathrm{SPM}}$ defined by the number of its elements that are not within L^2-distance 0.1 of any true entangled weight (again disregarding the sign) to make sure no spurious local maximizers were included. Lastly, we track the computational time necessary for the space approximation (line 1 in Algorithm 4.1) and consecutively the run time of SPM (line 2-16 in Algorithm 4.1) in seconds (rounded) denoted by $T_{\widehat{\mathcal{W}}}, T_{\mathrm{SPM}}$, respectively. The average results of ten repetitions with numerical derivatives are reported in Table 4.2, and the results averaged over ten runs with exact Hessians are reported in Table 4.3.
The first noteworthy insight from these results is that the overall difference between running the recovery with numerical approximations or exact Hessians only leads to a very marginal improvement in the latter case. This supports our assumption that the numerical error can be neglected for the most part and that the subspace approximation error is, in fact, mainly caused by the variation of the entangled weights w.r.t. the input (cf. Section 4.2.1). Overall, the recorded subspace error $\Delta_{\widehat{\mathcal{W}}}$ lies in the interval $[0,1]$ and therefore is quite accurate. Note that in Chapter 2 all results were stated in terms of the error $\delta = \|P_{\mathcal{W}} - P_{\widehat{\mathcal{W}}}\|_{F\to F}$, and therefore the Frobenius distance $\Delta_{\widehat{\mathcal{W}}}$ will in general only be a crude upper bound for δ. However, by relying on $\Delta_{\widehat{\mathcal{W}}}$ as a relative measure, we can observe that the subspace approximation becomes less accurate for the 4-layer architecture and that, in general, linear outputs yield slightly better results. On average all detectable entangled weights were found up to an error of $3.1 \cdot 10^{-2}$ in the worst case. Since these metrics were averaged over ten runs, let us mention that we found no run which significantly deviated from these mean values. The fact that the entangled

Architecture	g_L	$\Delta_{\widehat{\mathcal{W}}}$	$E_1(\mathcal{U}_{\text{SPM}})$	$E_2(\mathcal{U}_{\text{SPM}})$	$E_3(\mathcal{U}_{\text{SPM}})$	$E_4(\mathcal{U}_{\text{SPM}})$	fp	$T_{\widehat{\mathcal{W}}}$	T_{SPM}
$[50, 25, 10]$	tanh	0.31	2.1e-02	1.5e-02	8.6e-03	-	0.0	17s	12s
$[50, 25, 10]$	Id	0.08	9.3e-03	3.1e-03	-	-	0.0	17s	12s
$[50, 25, 50]$	tanh	0.11	7.8e-03	3.9e-03	3.3e-03	-	0.0	25s	12s
$[50, 25, 50]$	Id	0.04	3.2e-03	1.4e-03	-	-	0.0	25s	12s
$[50, 35, 25, 10]$	tanh	1.00	2.8e-02	3.1e-02	3.1e-02	2.1e-02	0.0	21s	12s
$[50, 35, 25, 10]$	Id	0.57	1.9e-02	1.9e-02	1.5e-02	-	0.0	20s	12s

Table 4.2: Performance metrics of the subspace approximation and SPM averaged over ten separate runs based on numerical Hessian approximations.

Architecture	g_L	$\Delta_{\widehat{\mathcal{W}}}$	$E_1(\mathcal{U}_{\text{SPM}})$	$E_2(\mathcal{U}_{\text{SPM}})$	$E_3(\mathcal{U}_{\text{SPM}})$	$E_4(\mathcal{U}_{\text{SPM}})$	fp	$T_{\widehat{\mathcal{W}}}$	T_{SPM}
$[50, 25, 10]$	tanh	0.24	1.7e-02	1.4e-02	7.7e-03	-	0.0	13s	11s
$[50, 25, 10]$	Id	0.08	7.4e-03	3.2e-03	-	-	0.0	12s	12s
$[50, 25, 50]$	tanh	0.09	6.6e-03	2.9e-03	2.9e-03	-	0.0	188s	12s
$[50, 25, 50]$	Id	0.04	3.4e-03	1.4e-03	-	-	0.0	153s	12s
$[50, 35, 25, 10]$	tanh	0.93	2.6e-02	3.2e-02	2.8e-02	1.8e-02	0.0	16s	12s
$[50, 35, 25, 10]$	Id	0.31	1.7e-02	1.4e-02	9.1e-03	-	0.0	14s	12s

Table 4.3: Performance metrics of the subspace approximation and SPM averaged over ten separate runs based on exact Hessian matrices.

weights of the last layer can not be detected for linear outputs is expected and was discussed in Section 4.1.2. We can always rely on the approximations via the gradients of the outputs to make up for this fact (see next paragraph). Another noteworthy point is that we did not observe any false positives in any of the runs, which clearly shows the robustness of our approach to small disturbances in the space $\widehat{\mathcal{W}} \approx \mathcal{W}$. The computational time for either step generally was in a range of a few seconds, such that the overall time necessary to find all entangled weights was under one minute in most cases. The time necessary to run SPM, given by T_{SPM}, remains constant over all different architectures because it only depends on the ambient dimension of the matrix space ($\widehat{\mathcal{W}} \in \mathbb{R}^{D \times D}$) and possibly the number of neurons m which did not vary significantly. While the number of random initializations for SPM ($5m \log m$) did depend on m, we do not observe any increase in the computational time due to parallelization within our code. The most computationally expensive part of the subspace approximation is the computation of the derivatives (via finite differences). Again, we see that $T_{\widehat{\mathcal{W}}} < 30s$ except for the runs with exact Hessians corresponding to the architecture $[50, 35, 50]$. This outlier is caused by the high number of output neurons and, ultimately, by our sequential implementation for the computation of the exact Hessians (i.e., compute the Hessian of all output neurons one by one). Since the runs with exact Hessians were only included as a baseline reference, we can ignore these outliers such that the total runtime to recover the entangled weight was less than one minute.

Step 2: Analysis of Algorithm 4.2. Algorithm 4.2 computes the estimated entangled weight matrices denoted by $\tilde{V}^{[1]}, \dots, \tilde{V}^{[L]}$ from the set of vectors $\mathcal{U} = \mathcal{U}_{\text{SPM}} \cup \mathcal{U}_L$ (we use $\mathcal{U}$ from the previous step). We are interested in the accuracy of the layer assignment. In this paragraph, we exclude the network architecture $[50, 35, 25, 10]$ where an oracle was used for the layer assignment. As a first measure, we consider the approximated entangled weights, i.e., the columns of the matrices $\tilde{V}^{[1]}, \dots, \tilde{V}^{[L]}$, and we compute the ratio how many columns were properly assigned. This needs to be checked column-wise since the entangled weight matrices can only be known up to permutations of the columns. A column $\tilde{v}_{k_\ell}^{[\ell]}$ of $\tilde{V}^{[\ell]}$ is counted as

Architecture	g_L	$r_{\text{asgn}}(\tilde{V}^{[1]})$	$r_{\text{asgn}}(\tilde{V}^{[2]})$	$r_{\text{asgn}}(\tilde{V}^{[3]})$	Full detection	T_{asg}
$[50, 25, 10]$	tanh	0.998	0.996	1.000	0.9	17s
$[50, 25, 10]$	Id	0.998	0.996	1.000	0.9	17s
$[50, 25, 50]$	tanh	1.000	1.000	1.000	1.0	24s
$[50, 25, 50]$	Id	1.000	1.000	1.000	1.0	24s

Table 4.4: Summary of the entangled weight assignment step (Algorithm 4.2) based on numerical approximations of the Hessian matrices.

Architecture	g_L	$r_{\text{asgn}}(\tilde{V}^{[1]})$	$r_{\text{asgn}}(\tilde{V}^{[2]})$	$r_{\text{asgn}}(\tilde{V}^{[3]})$	Full detection	T_{asg}
$[50, 25, 10]$	tanh	0.998	0.996	1.000	0.9	13s
$[50, 25, 10]$	Id	0.996	0.992	1.000	0.8	11s
$[50, 25, 50]$	tanh	1.000	1.000	1.000	1.0	191s
$[50, 25, 50]$	Id	1.000	1.000	1.000	1.0	155s

Table 4.5: Summary of the entangled weight assignment step (Algorithm 4.2) based on exact Hessian matrices.

properly assigned when there exists a column $v_{k'_\ell}^{[\ell]}$ in the true entangled weight matrix of the ℓ-th layer such that

$$\min_{s\in\{-1,+1\}} \left\| \tilde{v}_{k_\ell}^{[\ell]} - s v_{k'_\ell}^{[\ell]} / \left\| v_{k'_\ell}^{[\ell]} \right\|_2 \right\|_2 < 0.1. \tag{4.79}$$

We denote the ratio of correct assigned entangled weights per layer by $r_{\text{asgn}}(\tilde{V}^{[\ell]})$. Furthermore, we measure the number of repetitions (out of the ten runs) where all approximate entangled weights have been assigned to the correct layer denoted by "Full detection". Lastly, we also measure the computational time of Algorithm 4.2 in seconds denoted T_{asg}. The results corresponding to the runs with numerical differentiation are reported in Table 4.4, and the results for runs with exact Hessians are reported in Table 4.5.

Due to the strong similarity between both tables, we will only discuss the result where numerical differentiation was used. For the architecture $[50, 25, 50]$, we find a perfect assignment in all cases. For the networks of the type $[50, 25, 10]$, we see that there was one out of ten runs for both types of outputs, where the assignment made a mistake. In those faulty runs, the individual assignment ratios were $r_{\text{asgn}}(\tilde{V}^{[1]}) = 0.98$ and $r_{\text{asgn}}(\tilde{V}^{[1]}) = 0.96$. Hence, the assignment misjudged exactly one second-layer entangled weight for a first-layer entangled weight. Let us mention that this type of error typically occurs because the input distribution μ_X in the Algorithm 4.2 was not optimal. Therefore, the resulting subspace used to distinguish first-layer weights does not include all the first-layer entangled weights. This could be prevented heuristically by first checking how many entangled weight approximations are included in the matrix subspace and if this number does not match the number of neurons in the first layer, then the input distribution can be adjusted appropriately. The experiments show that by only considering $\mu_X = \text{Unif}(10^3 \cdot \mathbb{S}^{D-1})$, we already reach a reasonable precise assignment. It has to be stressed, however, that any mistake in the layer assignment is critical since it introduces irreversible error into the network reparametrization used in the next step of our pipeline This makes the entire reconstruction very sensitive to errors in the assignment. In Table 4.6 (numerical differentiation) and Table 4.7 (exact Hessians) we report the final errors of the constructed entangled weight matrices. This error is computed column-wise by comparing $\tilde{V}^{[\ell]} = [\tilde{v}_1^{[\ell]} \dots \tilde{v}_{m_\ell}^{[\ell]}]$ to the normalized columns of the ground truth entangled weight matrices $V^{[\ell]} = [v_1^{[\ell]} \dots v_{m_\ell}^{[\ell]}]$ while accounting for any possible permutation of the columns.

Architecture	g_L	$E(\tilde{V}^{[1]})$	$E(\tilde{V}^{[2]})$	$E(\tilde{V}^{[3]})$	$E(\tilde{V}^{[4]})$
$[50, 25, 10]$	tanh	2.1e-02	1.5e-02	2.5e-07	-
$[50, 25, 10]$	Id	9.6e-03	3.1e-03	1.9e-07	-
$[50, 25, 50]$	tanh	7.8e-03	3.9e-03	2.7e-07	-
$[50, 25, 50]$	Id	3.2e-03	1.4e-03	2.1e-07	-
$[50, 35, 25, 10]$	tanh	2.8e-02	3.1e-02	3.1e-02	2.9e-07
$[50, 35, 25, 10]$	Id	1.9e-02	1.9e-02	1.5e-02	2.4e-07

Table 4.6: Comparing (column-wise) the entangled weight matrices constructed via Algorithm 4.2 for runs which achieved a full detection of all entangled weights and were using numerical approximations of the Hessian matrices.

Architecture	g_L	$E(\tilde{V}^{[1]})$	$E(\tilde{V}^{[2]})$	$E(\tilde{V}^{[3]})$	$E_1(\tilde{V}^{[4]})$
$[50, 25, 10]$	tanh	1.7e-02	1.4e-02	2.1e-08	-
$[50, 25, 10]$	Id	8.0e-03	3.3e-03	2.1e-08	-
$[50, 25, 50]$	tanh	6.6e-03	2.9e-03	2.8e-08	-
$[50, 25, 50]$	Id	3.4e-03	1.4e-03	2.7e-08	-
$[50, 35, 25, 10]$	tanh	2.6e-02	3.2e-02	2.8e-02	2.0e-08
$[50, 35, 25, 10]$	Id	1.7e-02	1.3e-02	8.7e-03	2.2e-08

Table 4.7: Comparing (column-wise) the entangled weight matrices constructed via Algorithm 4.2 for runs which achieved a full detection of all entangled weights and were using exact Hessian matrices

More precisely,

$$E(\tilde{V}^{[\ell]}) = \max_{k_\ell \in [m_\ell]} \min_{s \in \{-1,+1\}, k'_\ell \in [m_\ell]} \left\| \tilde{v}^{[\ell]}_{k_\ell} - s v^{[\ell]}_{k'_\ell} / \left\| v^{[\ell]}_{k'_\ell} \right\|_2 \right\|_2 . \tag{4.80}$$

For both tables, we exclude the runs where the assignment was incorrect to not negatively skew the results. This affects the two runs mentioned before and three runs in the experiments with exact Hessians. We can see that for all hidden layers $\ell < L$, these errors reflect the errors in Tables 4.2-4.3. This shows that Algorithm 4.2 did generally detect the best vectors within the set $\mathcal{U}$ as representative of each entangled weight. Furthermore, the errors corresponding to the approximations of the last layer have improved significantly, which shows that the approximations via the gradient contained in the set $\mathcal{U}_L$ provide a better estimate than running SPM on the matrix subspace $\widehat{\mathcal{W}}$.

Step 3: Analysis of network completion via empirical risk minimization. For any network f, considered in our setup above, the two previous steps in combination compute a set of approximated entangled weight matrices $\tilde{V}_1, \ldots, \tilde{V}_L$. A network reparametrization $\tilde{f}$ from these matrices is then constructed as described in Section 4.3.2. The parametric function $\tilde{f}$ only depends on the unknown shifts and a set of diagonal matrices. These learn-able parameters are initialized according to (4.69) by zero vectors and identity matrices and then learned by minimizing the objective 4.67 via SGD.

Remark 4.5. *The tuning of the reparametrization is done via SGD. Let us note that empirically we observe the same results when we instead use ordinary GD. As a reference, we can point to the numerical experiments performed [50] where this step was run with ordinary GD.*

After tuning, the function $\tilde{f}$ and its parameters represent our final reconstruction of f. Functional equivalence between $f, \tilde{f}$ was proven in Section 4.3.2 for the right set of parameters. Note that the permutation of the output layer can be ignored as the correct order of these neurons

can be recovered by the approximations in $\mathcal{U}_L$. However, these results (Proposition 4.3 and Proposition 4.4) assumed $\tilde{f}$ was built from the ground truth entangled weight matrices up to permutation and re-scaling of the columns. In our setting, we can only provide disturbed matrices $\tilde{V}_1, \dots, \tilde{V}_L$ computed in the previous steps to build the reparametrization $\tilde{f}$. Therefore, we can only expect $\tilde{f}$ to match f up to some error induced by these irreversible errors. We track the following metrics:

1. Average L^2-error on the training inputs $\mathcal{X}_{\text{train}} = \{x_1, \dots, x_{N_{\text{SGD}}}\}$ (drawn from $\mathcal{N}(0, \text{Id}_D)$):

$$\text{MAE} = \frac{1}{N_{\text{SGD}}} \sum_{x \in \mathcal{X}_{\text{train}}} \|f(x) - \tilde{f}(x)\|_2. \tag{4.81}$$

2. The relative uniform error $E_{\infty,rel}$ on a test set $\mathcal{X}_{\text{test}}$ of $N_{\text{test}} = 10^6$ unseen inputs drawn as standard Gaussians:

$$E_{\infty,rel} = \frac{\max_{x \in \mathcal{X}_{\text{test}}} \left\|f(x) - \tilde{f}(x)\right\|_2}{\max_{x \in \mathcal{X}_{\text{test}}} \|f(x)\|_2} \tag{4.82}$$

3. To demonstrate the convergence in parameter space, we also measure the L^2-error between the ground truth shifts $\tau^{[\ell]}$ and the approximated shifts $\tilde{\tau}^{[\ell]}$ while accounting for any possible permutation and neglecting the sign due to symmetries (cf. Section 4.3.3). More precisely, we compute

$$E(\tilde{\tau}^{[\ell]}) = \frac{\left\|\left|\tau^{[\ell]}\right| - \left|\pi_\ell^\top \tilde{\tau}^{[\ell]}\right|\right\|_2}{\left\|\tau^{[\ell]}\right\|_2} \quad \text{for all } \ell \in [L], \tag{4.83}$$

where the permutation π_ℓ can be inferred from the column order of the entangled weight matrices.

All metrics are tracked during the training whenever an epoch finishes. The results only include those runs that achieved a full detection & assignment of the entangled weights (cf. last section). Additionally, for each architecture and output activation considered, we introduce a new set of 10 independent runs where the exact entangled weight matrices were given instead of the approximations $\tilde{V}^{[1]}, \dots, \tilde{V}^{[L]}$. This allows us to verify whether the reparametrization can exactly recover the original network provided that the entangled weight matrices have been fully reconstructed without any errors. Figures 4.5-4.7 display the average performance metrics.

In all cases, we observe a constant decrease of the MAE, relative uniform error, and distance of the approximated shifts in the first few minutes of running gradient descent. For entangled weights reconstructed via our pipeline, this decrease eventually plateaus at an order of magnitude which is similar to the error present in the approximated entangled weight matrices (see Table 4.6- 4.7). This is expected, as the parameters tuned for the reparametrized network cannot undo any errors made in reconstructing the entangled weight matrices. Hence, irreversible errors will accumulate during the first two steps of our pipeline. At the same time, one can see from the right column of each individual plot that provided exact entangled weight matrices all performance metrics keep steadily decreasing. This further supports the argument that entangled weights in themselves uniquely determine the network up to scales and shifts, and that these remaining parameters are learnable by simple first-order methods like SGD or GD. Networks with linear output neurons perform slightly better than networks with non-linear outputs ($g_L = \tanh$). Overall, the network with the most output neurons $[50, 25, 50]$ achieved the lowest errors in all considered settings (output type, Hessian computation, baseline with

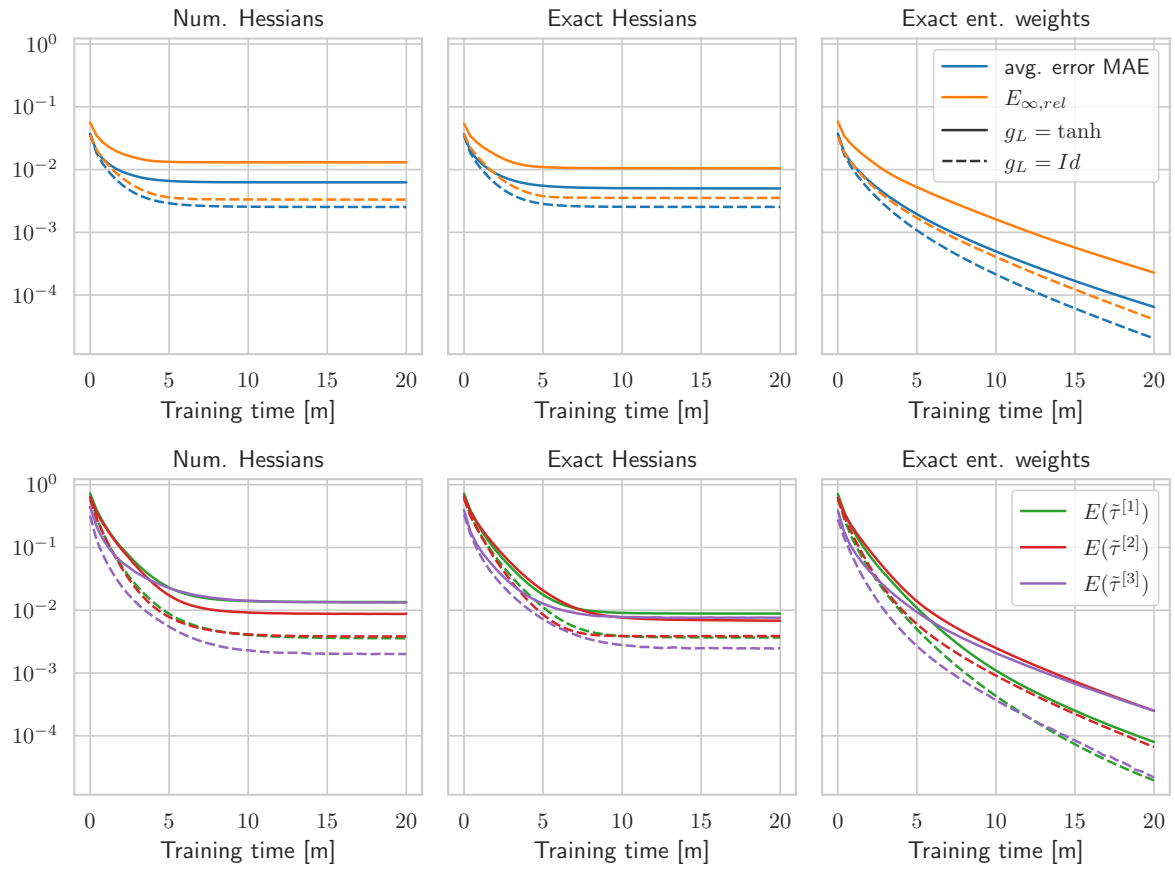

Figure 4.5: Measuring the distance of the network approximation $\tilde{f}$ and its parameters computed via our pipeline to the original network for the architectures with neurons $[50, 25, 10]$. Entangled weights are approximated via numerical differentiation (left column) or exact Hessians (middle column). The right columns show a baseline result where exact entangled weight matrices were provided and therefore only measure the performance of the empirical risk minimization in Section 4.3.2.

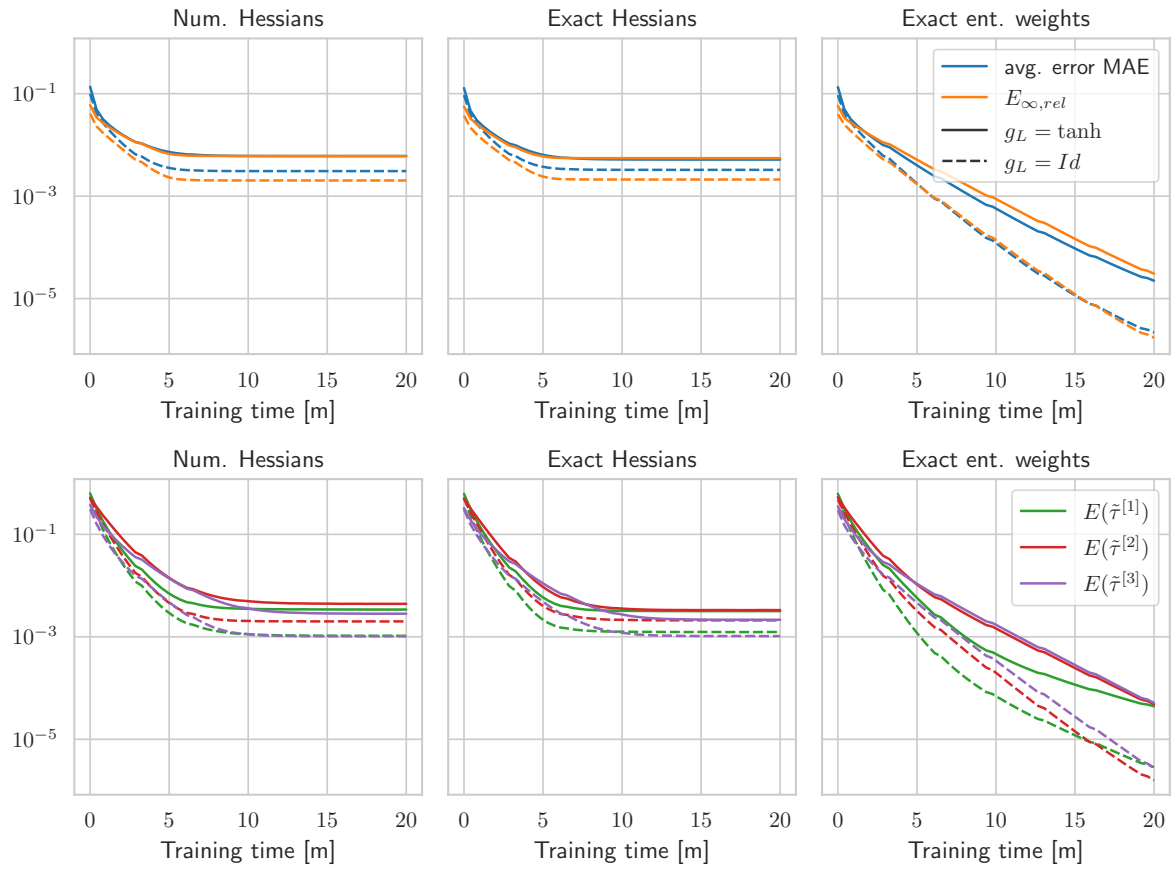

Figure 4.6: Measuring the distance of the network approximation $\tilde{f}$ and its parameters computed via our pipeline to the original network for the architectures with neurons $[50, 25, 50]$ when entangled weights are approximated via numerical differentiation (left column) or exact Hessians (middle column). The right column shows a baseline result where exact entangled weight matrices were provided and therefore only measures the performance of the empirical risk minimization in Section 4.3.2.

exact Hessians). There is a notable decrease in performance caused by increasing the depth if we compare the 3-layer network $[50, 25, 10]$ with the $[50, 35, 25, 10]$.

We want to conclude this section by comparing our results to the performance of SGD when applied in a teacher-student setting, which has been studied previously in Section 1.3.2. First, let us point out that we do not want to assess which method is superior since we operate in a highly artificial setting. We expect that our reconstruction pipeline, and stochastic gradient descent applied to the teacher-student problem in Section 1.3.2 can both be improved by hyperparameter tuning, more training data, and longer computations. Additionally, our pipeline explicitly made use of network queries to approximate Hessian matrices, whereas SGD is not designed to leverage such a setting and relies entirely on random input-output samples. We saw in Section 1.3.2 that with proper parameter tuning, SGD could uniquely identify 3-layer networks with architecture $[50, 25, 10]$ from D^2m samples. More precisely, the best-performing setup in Figure 1.5 in the three-layer case $[50, 25, 10]$ reached an accuracy of $E_\infty \approx 10^{-6}$ after roughly 45 minutes of training. Judging by the results in Figure 4.5, our pipeline can only achieve similar levels of accuracy when the entangled weights were exactly computed. Note, however, that in Figure 4.5 we see a consistent convergence in parameter space (based on the trajectory of $E(\tilde{\tau}^{[1]}), E(\tilde{\tau}^{[2]}), E(\tilde{\tau}^{[3]})$) up to a level where irreparable errors in the entangled weight matrices prevent any further improvement. Ultimately, both methods found the correct network; however, gradient descent attained a slightly better fit. Additionally, SGD in Figure 1.5 was running for two hours, whereas the reparametrized model was only trained for 20 minutes, and the reconstruction of entangled weights generally has a computational time in the order of seconds. Something worth mentioning is that in Section 1.3.2, we could not translate the successful identification of three-layer networks to the four-layer network $[50, 35, 25, 10]$. Our interpretation was that identifying a network in a teacher-student model via SGD becomes significantly harder when the number of layers increases. Whether the identification of general deep neural networks in this setting is consistently possible via SGD is a question we can not fully answer empirically. In particular, due to the lack of guiding principles for hyperparameter selection. Aside from the layer assignment, which is currently still limited to $L \leq 3$, we do not observe such a drastic difference between the recovery of $[50, 25, 10]$ and $[50, 35, 25, 10]$ via our pipeline as can be seen in Figure 4.5 & 4.7.

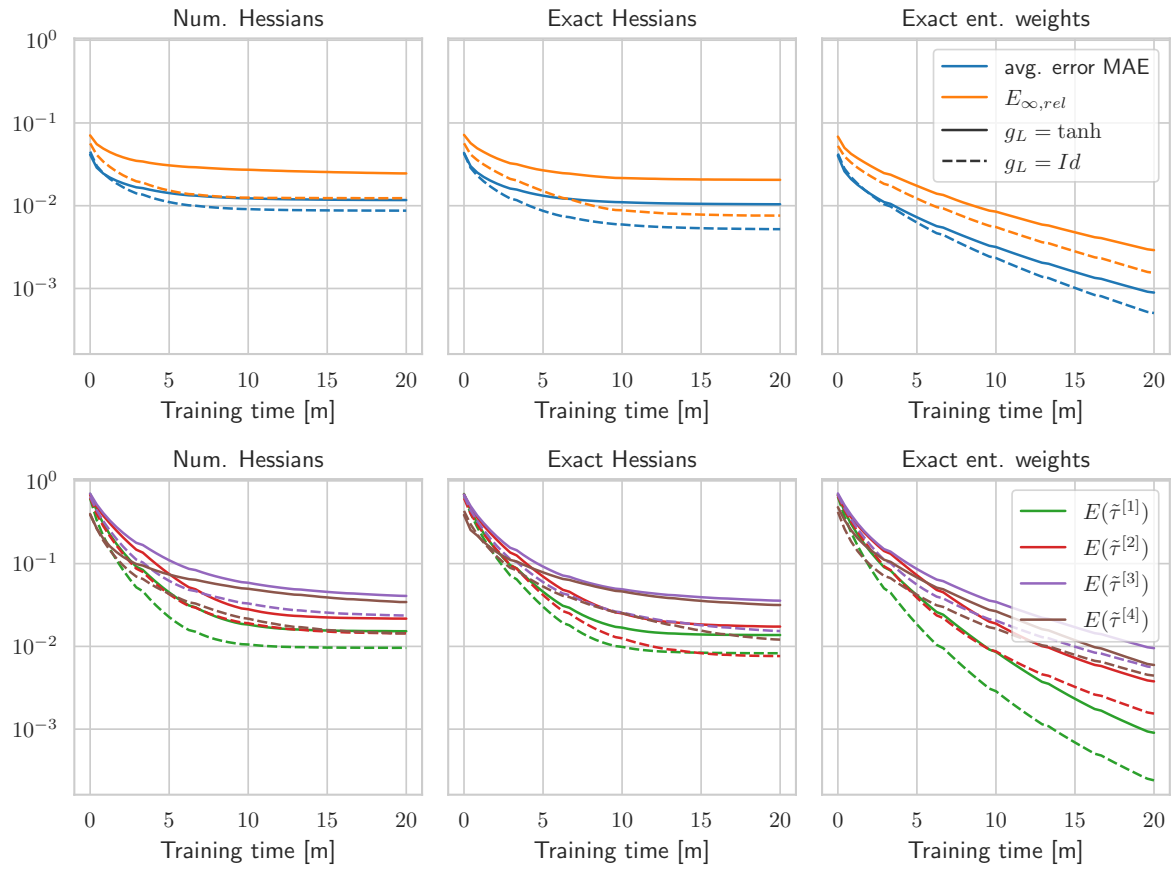

Figure 4.7: Measuring the distance of the network approximation $\tilde{f}$ and its parameters computed via our pipeline to the original network for the architectures with neurons $[50, 35, 25, 10]$ when entangled weights are approximated via numerical differentiation (left column) or exact Hessians (middle column). The right column shows a baseline result where exact entangled weight matrices were provided and therefore only measures the performance of the empirical risk minimization in Section 4.3.2.

Bibliography

[1] Mastering the game of Go with deep neural networks and tree search | Nature. https://www.nature.com/articles/nature16961.

[2] Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.

[3] Milton Abramowitz, Irene A. Stegun, and Robert H. Romer. Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables. *American Journal of Physics*, 56(10):958–958, October 1988.

[4] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, 2008.

[5] Francesca Albertini, Eduardo D Sontag, and Vincent Maillot. Uniqueness of weights for neural networks. *Artificial Neural Networks for Speech and Vision*, pages 115–125, 1993.

[6] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning (ICML)*, 2019.

[7] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, Jie Chen, Jingdong Chen, Zhijie Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Ke Ding, Niandong Du, Erich Elsen, Jesse Engel, Weiwei Fang, Linxi Fan, Christopher Fougner, Liang Gao, Caixia Gong, Awni Hannun, Tony Han, Lappi Johannes, Bing Jiang, Cai Ju, Billy Jun, Patrick LeGresley, Libby Lin, Junjie Liu, Yang Liu, Weigao Li, Xiangang Li, Dongpeng Ma, Sharan Narang, Andrew Ng, Sherjil Ozair, Yiping Peng, Ryan Prenger, Sheng Qian, Zongfeng Quan, Jonathan Raiman, Vinay Rao, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Kavya Srinet, Anuroop Sriram, Haiyuan Tang, Liliang Tang, Chong Wang, Jidong Wang, Kaifu Wang, Yi Wang, Zhijian Wang, Zhiqian Wang, Shuang Wu, Likai Wei, Bo Xiao, Wen Xie, Yan Xie, Dani Yogatama, Bin Yuan, Jun Zhan, and Zhenyao Zhu. Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 173–182. PMLR, June 2016.

[8] Animashree Anandkumar, Rong Ge, and Majid Janzamin. Guaranteed Non-Orthogonal Tensor Decomposition via Alternating Rank-1 Updates. March 2015. `arXiv:1402.5180`.

[9] Sanjeev Arora, Nadav Cohen, Noah Golowich, and Wei Hu. A convergence analysis of gradient descent for deep linear neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[10] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In *Neural Information Processing Systems (NeurIPS)*, 2019.

[11] Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[12] Peter Auer, Mark Herbster, and Manfred Warmuth. Exponentially many local minima for single neurons. In *Neural Information Processing Systems (NIPS)*, 1996.

[13] Bubacarr Bah, Holger Rauhut, Ulrich Terstiege, and Michael Westdickenberg. Learning deep linear neural networks: Riemannian gradient flows and convergence to global minimizers. *Information and Inference: A Journal of the IMA*, February 2021.

[14] Ravindra B. Bapat and Vaikalathur S. Sunder. On majorization and Schur products. *Linear algebra and its applications*, 72:107–117, 1985.

[15] B. Barak and David Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. *Electron. Colloquium Comput. Complex.*, April 2014.

[16] Howard E. Bell. Gershgorin's Theorem and the Zeros of Polynomials. *The American Mathematical Monthly*, 72(3):292–295, 1965.

[17] John J. Benedetto and Matthew Fickus. Finite Normalized Tight Frames. *Advances in Computational Mathematics*, 18(2):357–385, February 2003.

[18] Julius Berner, Philipp Grohs, and Arnulf Jentzen. Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of Black–Scholes partial differential equations. *SIAM Journal on Mathematics of Data Science*, 2(3):631–657, January 2020.

[19] Aditya Bhaskara, Moses Charikar, and Aravindan Vijayaraghavan. Uniqueness of Tensor Decompositions with Applications to Polynomial Identifiability. In *Proceedings of The 27th Conference on Learning Theory*, pages 742–778. PMLR, May 2014.

[20] Rajendra Bhatia. *Matrix Analysis*. Springer Science & Business Media, November 1996.

[21] Avrim Blum and Ronald Rivest. Training a 3-Node neural network is NP-Complete. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1989.

[22] Simone Bombari, Mohammad Hossein Amani, and Marco Mondelli. Memorization and optimization in deep neural networks with minimum over-parameterization. In *Advances in Neural Information Processing Systems*, 2022.

[23] Léon Bottou. Stochastic Gradient Descent Tricks. In Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade: Second Edition*, Lecture Notes in Computer Science, pages 421–436. Springer, Berlin, Heidelberg, 2012.

[24] Jerome Brachat, Pierre Comon, Bernard Mourrain, and Elias Tsigaridas. Symmetric tensor decomposition. *Linear Algebra and its Applications*, 433(11-12):1851–1872, December 2010.

[25] Alon Brutzkus and Amir Globerson. Globally optimal gradient descent for a convnet with gaussian inputs. In *International Conference on Machine Learning*, pages 605–614. PMLR, 2017.

[26] Sébastien Bubeck, Ronen Eldan, Yin Tat Lee, and Dan Mikulincer. Network size and weights size for memorization with two-layers neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[27] Martin D Buhmann and Allan Pinkus. Identifying Linear Combinations of Ridge Functions. *Advances in Applied Mathematics*, 22(1):103–118, January 1999.

[28] Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *Advances in Neural Information Processing Systems*, 32:10836–10846, 2019.

[29] Peter G. Casazza and Nicole Leonhard. Classes of finite equal norm Parseval frames. In David R. Larson, Peter Massopust, Zuhair Nashed, Minh Chuong Nguyen, Manos Papadakis, and Ahmed Zayed, editors, *Contemporary Mathematics*, volume 451, pages 11–31. American Mathematical Society, Providence, Rhode Island, 2008.

[30] Ole Christensen. *An Introduction to Frames and Riesz Bases*. Applied and Numerical Harmonic Analysis. Springer International Publishing, Cham, 2016.

[31] Charles K. Chui and Xin Li. Approximation by ridge functions and neural networks with one hidden layer. *Journal of Approximation Theory*, 70(2):131–141, August 1992.

[32] Alexander Cloninger and Timo Klock. A deep network construction that adapts to intrinsic dimensionality beyond the domain. *Neural Networks*, 141:404–419, 2021.

[33] Albert Cohen, Ingrid Daubechies, Ronald DeVore, Gerard Kerkyacharian, and Dominique Picard. Capturing Ridge Functions in High Dimensions from Point Queries. *Constructive Approximation*, 35(2):225–243, April 2012.

[34] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, December 1989.

[35] I. Daubechies, R. DeVore, S. Foucart, B. Hanin, and G. Petrova. Nonlinear approximation and (deep) ReLU networks. *Constructive Approximation*, April 2021.

[36] Chandler Davis and W. M. Kahan. The Rotation of Eigenvectors by a Perturbation. III. *SIAM Journal on Numerical Analysis*, 7(1):1–46, 1970.

[37] Vin de Silva and Lek-Heng Lim. Tensor Rank and the Ill-Posedness of the Best Low-Rank Approximation Problem. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1084–1127, January 2008.

[38] Ronald DeVore, Boris Hanin, and Guergana Petrova. Neural network approximation. *Acta Numerica*, 30:327–444, 2021.

[39] David L. Donoho and Iain M. Johnstone. Projection-Based Approximation and a Duality with Kernel Methods. *The Annals of Statistics*, 17(1):58–106, 1989.

[40] Simon S. Du, Jason D. Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[41] Simon S. Du, Jason D. Lee, and Yuandong Tian. When is a convolutional filter easy to learn? In *International Conference on Learning Representations (ICLR)*, 2018.

[42] Simon S. Du, Jason D. Lee, Yuandong Tian, Aarti Singh, and Barnabas Poczos. Gradient descent learns one-hidden-layer CNN: Don't be afraid of spurious local minima. In *International Conference on Machine Learning (ICML)*, 2018.

[43] Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

[44] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, September 1936.

[45] Dennis Elbrächter, Dmytro Perekrestenko, Philipp Grohs, and Helmut Bölcskei. Deep Neural Network Approximation Theory. *IEEE Transactions on Information Theory*, 67(5):2581–2623, May 2021.

[46] Ronen Eldan and Ohad Shamir. The Power of Depth for Feedforward Neural Networks. In *Conference on Learning Theory*, pages 907–940. PMLR, June 2016.

[47] Gökcen Eraslan, Žiga Avsec, Julien Gagneur, and Fabian J. Theis. Deep learning: New computational modelling techniques for genomics. *Nature Reviews Genetics*, 20(7):389–403, July 2019.

[48] Charles Fefferman. Reconstructing a neural net from its output. *Revista Matemática Iberoamericana*, 10(3):507–555, 1994.

[49] Christian Fiedler. Learning deep neural networks with very few samples. Master's thesis, Technical University Munich, 2019.

[50] Christian Fiedler, Massimo Fornasier, Timo Klock, and Michael Rauchensteiner. Stable recovery of entangled weights: Towards robust identification of deep neural networks from minimal samples. *Applied and Computational Harmonic Analysis*, 62:123–172, January 2023.

[51] Massimo Fornasier, Timo Klock, Marco Mondelli, and Michael Rauchensteiner. Finite Sample Identification of Wide Shallow Neural Networks with Biases. November 2022. `arXiv:2211.04589`.

[52] Massimo Fornasier, Timo Klock, and Michael Rauchensteiner. Robust and Resource-Efficient Identification of Two Hidden Layer Neural Networks. *Constructive Approximation*, June 2021.

[53] Massimo Fornasier, Karin Schnass, and Jan Vybiral. Learning Functions of Few Arbitrary Linear Parameters in High Dimensions. *Foundations of Computational Mathematics*, 12(2):229–262, April 2012.

[54] Massimo Fornasier, Jan Vybíral, and Ingrid Daubechies. Robust and resource efficient identification of shallow neural networks by fewest samples. *Information and Inference: A Journal of the IMA*, 10(2):625–695, June 2021.

[55] Jerome H. Friedman, Mark Jacobson, and Werner Stuetzle. PROJECTION PURSUIT REGRESSION. *J. Am. Statist. Assoc.*, 76:817, 1981.

[56] Haoyu Fu, Yuejie Chi, and Yingbin Liang. Guaranteed Recovery of One-Hidden-Layer Neural Networks via Cross Entropy. *IEEE Transactions on Signal Processing*, 68:3225–3235, 2020.

[57] Ken-Ichi Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192, January 1989.

[58] Semen Geršgorin. über die Abgrenzung der Eigenwerte einer Matrix. *Dokl. Akad. Nauk (A), Otd. Fiz.-Mat. Nauk (1931)*, pages 749–754.

[59] Alex Gittens and Joel A. Tropp. Tail bounds for all eigenvalues of a sum of random matrices. July 2011. `arXiv:1104.4513`.

[60] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep Sparse Rectifier Neural Networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, June 2011.

[61] Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge Distillation: A Survey. *International Journal of Computer Vision*, 129(6):1789–1819, June 2021.

[62] Johan Håstad. Tensor rank is NP-complete. *Journal of Algorithms*, 11(4):644–654, December 1990.

[63] Christopher J. Hillar and Lek-Heng Lim. Most Tensor Problems Are NP-Hard. *Journal of the ACM*, 60(6):45:1–45:39, November 2013.

[64] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. March 2015. `arXiv:1503.02531`.

[65] Samuel B. Hopkins, Tselil Schramm, and Jonathan Shi. A Robust Spectral Algorithm for Overcomplete Tensor Decomposition. In *Proceedings of the Thirty-Second Conference on Learning Theory*, pages 1683–1722. PMLR, June 2019.

[66] Samuel B. Hopkins, Tselil Schramm, Jonathan Shi, and David Steurer. Fast spectral algorithms from sum-of-squares proofs: Tensor decomposition and planted sparse vectors. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 178–191, New York, NY, USA, June 2016. Association for Computing Machinery.

[67] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, January 1989.

[68] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[69] Marian HRISTACHE, Anatoli JUDITSKY, and Vladimir SPOKOINY. Direct estimation of the index coefficient in a single-index model. *The Annals of statistics*, 29(3):595–623, 2001.

[70] Guang-Bin Huang. Learning capability and storage capacity of two-hidden-layer feedforward networks. *IEEE Transactions on Neural Networks*, 14(2):274–281, 2003.

[71] Peter J. Huber. Projection Pursuit. *The Annals of Statistics*, 13(2):435–475, June 1985.

[72] Hidehiko Ichimura. Semiparametric least squares (SLS) and weighted SLS estimation of single-index models. *Journal of Econometrics*, 58(1):71–120, July 1993.

[73] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2018.

[74] Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. Beating the Perils of Non-Convexity: Guaranteed Training of Neural Networks using Tensor Methods. January 2016. `arXiv:1506.08473`.

[75] Stephen Judd. On the complexity of loading shallow neural networks. *Journal of Complexity*, 4(3):177–192, September 1988.

[76] Joe Kileel, Timo Klock, and João Pereira. Landscape analysis of an improved power method for tensor decomposition. In *Neural Information Processing Systems (NeurIPS)*, 2021.

[77] Joe Kileel and João M. Pereira. Subspace power method for symmetric tensor decomposition and generalized PCA. June 2021. `arXiv:1912.04007`.

[78] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[79] Joseph B. Kruskal. Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18(2):95–138, January 1977.

[80] H. W. Kuhn and A. W. Tucker. Nonlinear Programming. *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, 2:481–493, January 1951.

[81] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[82] Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, January 1993.

[83] Ker-Chau Li. On Principal Hessian Directions for Data Visualization and Dimension Reduction: Another Application of Stein's Lemma. *Journal of the American Statistical Association*, 87(420):1025–1039, 1992.

[84] Yuanzhi Li, Colin Wei, and Tengyu Ma. Towards Explaining the Regularization Effect of Initial Large Learning Rate in Training Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[85] Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with ReLU activation. In *Neural Information Processing Systems (NeurIPS)*, 2017.

[86] Kung-Ching Lin. *Nonlinear Sampling Theory and Efficient Signal Recovery*. PhD thesis, University of Maryland, 2020.

[87] Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-Time Tensor Decompositions with Sum-of-Squares. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 438–446, New Brunswick, NJ, USA, October 2016. IEEE.

[88] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *In ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

[89] Jan R. Magnus. On Differentiating Eigenvalues and Eigenvectors. *Econometric Theory*, 1(2):179–191, 1985.

[90] Sebastian Mayer, Tino Ullrich, and Jan Vybíral. Entropy and Sampling Numbers of Classes of Ridge Functions. *Constructive Approximation*, 42(2):231–264, October 2015.

[91] HN Mhaskar and T Poggio. Function approximation by deep networks. *Communications on Pure & Applied Analysis*, 19(8), 2020.

[92] Hrushikesh N Mhaskar and Tomaso Poggio. Deep vs. shallow networks: An approximation theory perspective. *Analysis and Applications*, 14(06):829–848, 2016.

[93] Marco Mondelli and Andrea Montanari. On the Connection Between Learning Two-Layer Neural Networks and Tensor Decomposition. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, pages 1051–1060. PMLR, April 2019.

[94] Andrea Montanari and Yiqiao Zhong. The interpolation phase transition in neural networks: Memorization and generalization under lazy training, 2020. `arXiv:2007.12826`.

[95] Edward Moroshko, Blake E. Woodworth, Suriya Gunasekar, Jason D. Lee, Nati Srebro, and Daniel Soudry. Implicit bias in deep linear classification: Initialization scale vs training accuracy. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[96] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, Madison, WI, USA, June 2010. Omnipress.

[97] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. In *International Conference on Learning Representations (ICLR)*, 2015.

[98] Quynh Nguyen. On the proof of global convergence of gradient descent for deep relu networks with linear widths. In *International Conference on Machine Learning (ICML)*, 2021.

[99] Quynh Nguyen and Marco Mondelli. Global convergence of deep networks with one wide layer followed by pyramidal topology. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[100] Keit Oldham, Jan Myland, and Jerome Spanier. *An Atlas of Functions*. Springer US, New York, NY, 2009.

[101] A. M. Ostrowski. A Quantitative Formulation of Sylvester's Law of Inertia. *Proceedings of the National Academy of Sciences of the United States of America*, 45(5):740–744, 1959.

[102] Samet Oymak and Mahdi Soltanolkotabi. Toward moderate overparameterization: Global convergence guarantees for training shallow neural networks. *IEEE Journal on Selected Areas in Information Theory*, 1(1):84–105, 2020.

[103] Karl Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, November 1901.

[104] Philipp Petersen and Felix Voigtlaender. Optimal approximation of piecewise smooth functions using deep ReLU neural networks. *Neural Networks*, 108:296–330, 2018.

[105] Pencho P. Petrushev. Approximation by ridge functions and neural networks. *SIAM Journal on Mathematical Analysis*, 30(1):155–189, 1998.

[106] Allan Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8:143–195, January 1999.

[107] Tomaso Poggio, Hrushikesh Mhaskar, Lorenzo Rosasco, Brando Miranda, and Qianli Liao. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519, October 2017.

[108] Franz Rellich and Joan Berkowitz. *Perturbation Theory of Eigenvalue Problems*. CRC Press, 1969.

[109] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986.

[110] Itay Safran and Ohad Shamir. Spurious local minima are common in two-layer ReLU neural networks. In *International Conference on Machine Learning (ICML)*, 2018.

[111] Hanie Sedghi and Anima Anandkumar. Provable methods for training neural networks with sparse connectivity. 2014. `arXiv:1412.2693`.

[112] Uri Shaham, Alexander Cloninger, and Ronald R Coifman. Provable approximation properties for deep neural networks. *Applied and Computational Harmonic Analysis*, 44(3):537–557, 2018.

[113] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - from Theory to Algorithms.* Cambridge University Press, 2014.

[114] Mahdi Soltanolkotabi. Learning ReLUs via gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2004–2014, 2017.

[115] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D. Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, 2018.

[116] Chaehwan Song, Ali Ramezani-Kebrya, Thomas Pethick, Armin Eftekhari, and Volkan Cevher. Subquadratic overparameterization for shallow neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

[117] Zhao Song and Xin Yang. Quadratic Suffices for Over-parametrization via Matrix Chernoff Bound, February 2020. `arXiv:1906.03593`.

[118] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[119] Charles M. Stein. Estimation of the Mean of a Multivariate Normal Distribution. *The Annals of Statistics*, 9(6):1135–1151, 1981.

[120] Gilbert W. Stewart. Perturbation Theory for the Singular Value Decomposition. UMIACS-TR-90-124, 1998.

[121] Héctor J. Sussmann. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Networks*, 5(4):589–593, 1992.

[122] Terence Tao. When are eigenvalues stable? Available: `https://terrytao.wordpress.com/2008/10/28/when-are-eigenvalues-stable/`, October 2008. (Last accessed: April 6th 2023).

[123] Terence Tao. *Topics in Random Matrix Theory*. American Mathematical Soc., March 2012.

[124] Yuandong Tian. An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. In *International Conference on Machine Learning*, pages 3404–3413. PMLR, 2017.

[125] J.A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, October 2004.

[126] Joel A. Tropp. User-Friendly Tail Bounds for Sums of Random Matrices. *Foundations of Computational Mathematics*, 12(4):389–434, August 2012.

[127] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge University Press, September 2018.

[128] Roman Vershynin. Memory capacity of neural networks with threshold and rectified linear unit activations. *SIAM Journal on Mathematics of Data Science*, 2(4):1004–1033, 2020.

[129] Verner Vlačić and Helmut Bölcskei. Affine symmetries and neural network identifiability. *Advances in Mathematics*, 376:107485, January 2021.

[130] Per-Åke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, March 1972.

[131] L. Welch. Lower bounds on the maximum cross correlation of signals (Corresp.). *IEEE Transactions on Information Theory*, 20(3):397–399, May 1974.

[132] Hermann Weyl. Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen (mit einer Anwendung auf die Theorie der Hohlraumstrahlung). *Mathematische Annalen*, 71(4):441–479, December 1912.

[133] Blake Woodworth, Suriya Gunasekar, Jason D. Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in over-parametrized models. In *Conference on Learning Theory (COLT)*, 2020.

[134] Xiaoxia Wu, Simon S. Du, and Rachel Ward. Global convergence of adaptive gradient methods for an over-parameterized neural network, 2019. `arXiv:1902.07111`.

[135] Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. Small nonlinearities in activation functions create bad local minima in neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

[136] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*, 2017.

[137] Xiao Zhang, Yaodong Yu, Lingxiao Wang, and Quanquan Gu. Learning one-hidden-layer ReLU networks via gradient descent. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[138] Kai Zhong, Zhao Song, Prateek Jain, Peter L. Bartlett, and Inderjit S. Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *International Conference on Machine Learning (ICML)*, 2017.

[139] Mo Zhou, Rong Ge, and Chi Jin. A local convergence theory for mildly over-parameterized two-layer neural network. In *Conference on Learning Theory (COLT)*, 2021.

[140] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Gradient descent optimizes over-parameterized deep ReLU networks. *Machine Learning*, 109(3):467–492, 2020.

[141] Difan Zou and Quanquan Gu. An improved analysis of training over-parameterized deep neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2019.