Singular Learning Theory

Singular Learning Theory, Lecture 1, Michaelmas Term 2026

Adapted from the notes for the Iliad Intensive

1 Introduction **3**

2 Where classical learning theory falls short **4**
2.1 The classical picture 4
2.2 The anomalies: What we observe in deep learning 4
2.3 Resolving the crisis: Complexity of the solution 6
2.4 Where Singular Learning Theory comes in 6

3 Preliminaries **7**
3.1 Neural networks and parameter–function maps 7
3.2 Supervised deep learning and loss functions 9

4 What is degeneracy? **11**
4.1 A definition of degeneracy 11
4.2 Degeneracy and continuous symmetries 13
4.3 Localised degeneracy 15
4.4 Degeneracy and the loss landscape 18

References and further reading **22**

A Solutions **23**
A.1 Solutions to exercises from Section 3 24
A.2 Solutions to exercises from Section 4 25

Contents

These notes accompany the first lecture of the course, which covers the following topics. For each topic, we indicate where it is treated in these notes.

1. *What singular learning theory is, and the questions it sets out to answer:* Section 1. For a longer overview of what SLT says about deep learning, see the position paper by Wei et al. (2023).
2. *Where classical statistical learning theory falls short for neural networks:* Section 2, which reviews observed phenomena that classical theory fails to explain.
3. *Why SLT matters for AI safety:* covered in the lecture rather than in these notes. For further reading, see the position paper by Pepin Lehalleur et al. (2025).
4. *Formalising the learning problem (parameter–function maps, neural networks and loss functions):* Section 3.
5. *Degeneracies, in the parameter–function map and in the loss landscape:* Section 4. Sections 4.1 to 4.3 treat degeneracy of the parameter–function map, and Section 4.4 turns to degeneracy in the loss landscape.

Sections 3 and 4 include definitions and guided exercises, and Section A provides worked solutions for all exercises.

Acknowledgements

These notes are a slightly adapted version of the notes *Degeneracy in Deep Learning: An Invitation to Singular Learning Theory*, developed for the Iliad Intensive by Kai Ogden (University of Oxford), Matthew Farrugia-Roberts (University of Oxford), and Zach Furman (The University of Melbourne). They have been adapted, with the authors' permission, for the first lecture of the Singular Learning Theory course at the University of Oxford (Michaelmas Term 2026). Copyright in the original notes remains with their authors (© Kai Ogden, Matthew Farrugia-Roberts, Zach Furman, 2026; all rights reserved), and the adapted material is reproduced here with their permission. Some exercises in the original notes were adapted from Furman (2024).

1 Introduction

Singular learning theory (SLT) is a theory of learning that accounts for *degeneracy* in neural networks. What is degeneracy? Let us first appeal to the Cambridge dictionary, documenting some features of typical uses of the word within mathematics:

> *Degenerate* (of an equation, curve, line, etc.) unusual or complicated in some way compared to other equations, curves, lines, etc. of a similar type, especially because a variable or parameter is zero.

On the other hand, we should recall that deep learning has as many roots in neuroscience as in mathematics. And as they say, "neural networks are *grown*." Perhaps we should also hear the biologist's definition:

> *Degenerate* (of an organism, chromosome, etc.) simpler than a form that previously existed because of no longer having a particular structure.

Actually, *both* of these definitions are apt. Within a typical neural network architecture, certain weight vectors correspond to neural networks from simpler architectures (often due to certain weights being zero). Structurally, these neural networks are simpler in form than their neighbours. Mathematically, they complicate the relationship between the neural network's parameter space and the resulting space of functions. In turn, learning in neural networks is substantially richer than learning in classical statistical models.

Singular learning theory is a framework for understanding learning that places degeneracy at the centre. The framework leverages powerful tools from algebraic geometry to understand, characterise, and resolve degeneracies, revealing their impact on learning. Unfortunately, the price of this mathematical power is that following the research literature without a deep background in pure mathematics is challenging.

That is where this tutorial comes in. We aim to distil the essence of singular learning theory—the notion of degeneracy and its role in learning—laying bare through simple examples and structured exercises the core intuitions driving research in the field. We believe understanding this essence requires no more than undergraduate-level mathematics. Moreover, to anyone armed with these core intuitions, understanding and contributing to research on the relationship between degeneracy and deep learning is within reach.

2 Where classical learning theory falls short

Why does deep learning work as well as it does? Following Furman (2026), we can split this question into three parts: *approximation* (can the architecture represent a good solution?), *generalisation* (will a solution that fits the training data also perform well on new data?), and *optimisation* (can the training procedure find such a solution?). For neural networks, each part raises puzzles that classical learning theory struggles to explain.

In these notes, we focus on *generalisation*, which in this context means performing well on new data drawn from the same distribution as the training data (a narrower notion than out-of-distribution generalisation). To understand how Singular Learning Theory (SLT) contributes to the generalisation question, we must first understand the classical picture, the empirical phenomena that break it, and the conceptual shift required to resolve the crisis. As we will see, SLT fits into this modern understanding by providing a rigorous measure of the soft inductive biases in Bayesian Learning.

2.1 The classical picture

Suppose we use some architecture with parameter space $\mathcal{W}$ to fit a model to n training examples, with empirical loss $L_n(w)$ (the training error) and population loss $L(w)$ (the expected test error) for each parameter $w \in \mathcal{W}$, as in Section 3.2. Classical learning theory controls the *generalisation gap* between the two through the capacity or size of the hypothesis class $\mathcal{F}$, that is, the set of functions that the model can represent. Schematically (we suppress constants and logarithmic factors), with high probability over the draw of the training set, *every* parameter $w \in \mathcal{W}$ satisfies

$$L(w) \lesssim L_n(w) + \sqrt{\frac{C(\mathcal{F})}{n}}, \tag{1}$$

where $C(\mathcal{F})$ measures the capacity of the whole class, such as its VC dimension or Rademacher complexity. Intuitively, more expressive models are more likely to fit the data by chance, so lower training loss is weaker evidence of small test loss. Such *uniform convergence* bounds do not depend on which hypothesis is learned, and they penalise the size of the entire hypothesis class. They therefore suggest restricting the class in order to generalise (Wilson, 2025).

Two lessons follow. The first is the *bias–variance trade-off*. A small class may not contain a good solution (high bias), while a large class can fit the noise in the training data (high variance). Test error should therefore trace a U-shaped curve as capacity grows, with an optimum at intermediate capacity; in particular, a model that fits noisy training data exactly should generalise poorly. The second lesson is that, for neural networks, capacity grows with the number of parameters d. For example, Zhang et al. (2017) show that a two-layer ReLU network with $2n + m$ weights can represent any labelling of any n inputs in $\mathbb{R}^m$. A network with $d \gg n$ parameters should therefore be expected to overfit.

2.2 The anomalies: What we observe in deep learning

When applied to modern neural networks, the classical predictions fail dramatically. We observe several distinct phenomena that contradict the idea that large, flexible hypothesis

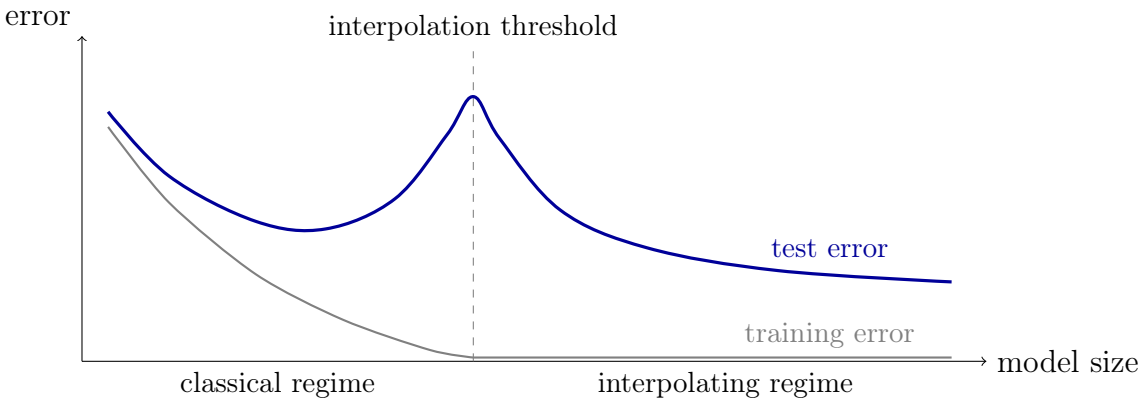

Figure 1: Schematic illustration of double descent (not real data). Below the interpolation threshold, test error follows the U-shaped curve predicted by the bias–variance trade-off. Beyond it, every model fits the training data exactly, yet test error decreases again as the model grows.

classes necessarily lead to poor generalisation.

Fitting noise. Zhang et al. (2017) trained standard image classifiers on CIFAR-10, both with the true labels and with random labels, shuffled pixels, or pure Gaussian noise as inputs. In every case, the networks reached (almost) zero training error. On the true labels, the same networks generalised well. The implication is stark: the architecture, training algorithm, and number of examples are identical in both experiments. A bound of the form (1) must hold in both. Since the training error is zero and the test error is at chance with random labels, the complexity term $\sqrt{C(\mathcal{F})/n}$ must be at least the chance-level error. This makes the bound entirely vacuous for the true labels as well.

Double descent. As the size of a model grows, its test error initially follows the classical U-shaped curve, peaking near the *interpolation threshold* where the model first becomes able to fit the training data exactly. However, as the model grows even larger, the test error *decreases again* (Figure 1). This second descent appears in deep networks, random forests, and even linear models.

Overparameterisation and benign overfitting. Modern networks typically have far more parameters than training examples, yet larger networks often generalise *better*. Furthermore, models can fit noisy or partially corrupted labels perfectly, yet their test accuracy degrades only gradually as the fraction of corrupted labels grows.

Uniform convergence fails. To fix these issues, one might try measuring the complexity of the trained network rather than the whole architecture. However, Nagarajan and Kolter (2019) showed that for an overparameterised linear classifier trained by gradient descent, any two-sided uniform convergence bound is vacuous, even one restricted to the classifiers the algorithm actually outputs. Uniform convergence itself breaks down.

2.3 Resolving the crisis: Complexity of the solution

How do we make sense of this evidence? Wilson (2025) argues that phenomena like benign overfitting and double descent do not require rethinking generalisation entirely. Instead, they require a shift in focus: away from the capacity of the *hypothesis class*, and toward the complexity of the *specific solution*.

Soft inductive biases. An inductive bias is any preference a learning algorithm has for certain solutions over others when multiple solutions perfectly fit the training data.

A *hard* inductive bias strictly restricts the hypothesis class upfront (e.g., forcing a model to be strictly linear or limiting its parameter count). In contrast, a *soft* inductive bias maintains a highly flexible, unrestricted hypothesis class but imposes a preference towards simpler solutions during training. A flexible model equipped with a soft preference for simplicity can interpolate the data (low bias) without fitting noise chaotically (low variance).

At the interpolation threshold of double descent, for example, a model with just enough capacity to fit the data has few ways to do so, making its fit highly sensitive to noise (the variance spike). As capacity increases further, the network gains infinitely many ways to interpolate the data. The training process and the model's inductive bias consistently select the preferred interpolating solutions from this vast set, causing variance to fall again.

2.4 Where Singular Learning Theory comes in

The realisation that generalisation depends on the complexity of the *solution* rather than the size of the hypothesis class is the exact starting point of Singular Learning Theory.

Watanabe showed that in Bayesian learning, models perform something akin to an internal model selection process, with an explicit bias towards "simpler" solutions. In classical models, a solution's complexity is entirely dictated by the parameter count d. However, neural networks are *singular* models. Watanabe proved that in singular models, $d/2$ is replaced by the *learning coefficient* $\lambda \leq d/2$. When a model is highly degenerate, λ can be significantly smaller than $d/2$.

For a given model, SLT provides a mathematically rigorous account of these soft inductive biases. It demonstrates that Bayesian inference naturally favours highly degenerate solutions with a smaller learning coefficient λ, which we will explore mathematically in later lectures, building on the notion of degeneracy introduced in Section 4. While relating these Bayesian results directly to stochastic gradient descent remains an active area of research (which we return to at the end of the course), SLT serves as a possible foundation for a modern theory of deep learning, providing the precise theoretical language such a theory would require.

3 Preliminaries

In this section, we introduce basic terminology and notation for supervised deep learning, along with some example neural network architectures that we will repeatedly study throughout the tutorial.

3.1 Neural networks and parameter–function maps

Let $\mathcal{X}$ denote a space of network inputs (e.g., a space of images or token sequences encoded as vectors), and let $\mathcal{Y}$ denote a space of outputs (e.g., a space of numerical scores, class distributions, or next-token distributions).

Informally, a neural network architecture is a specification for how various **parameters** (e.g., neuron connection strengths / weights or neuron activation thresholds / biases) combine with input signals and each other to describe a function from $\mathcal{X}$ to $\mathcal{Y}$.

Formally, a neural network architecture essentially comprises a **parameter–function map** $\Phi : \mathcal{W} \to \mathcal{F}$, where $\mathcal{W} \subseteq \mathbb{R}^d$ is a d-dimensional **parameter space** and $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ is some **hypothesis class** of functions from $\mathcal{X}$ to $\mathcal{Y}$. It is sometimes convenient to refer to the function $\Phi(w) : \mathcal{X} \to \mathcal{Y}$ as $f_w : \mathcal{X} \to \mathcal{Y}$.

Throughout the rest of this tutorial, we will study many different neural network architectures (parameter–function maps). The rest of this section explores some generally useful examples and some terminological points. To begin with, we have the following remark.

Remark 3.1 (Parameters)**.** Note that the term "parameter" has two senses:

1. A "parameter" is an individual dimension of parameter space (e.g., "initialise this parameter to the value 0.0," or "this neural network has billions of parameters"). A loose synonym in this case is "weight."
2. A "parameter" is also a particular point in the parameter space (e.g., "this parameter is a local minimum of the loss function," or "the zero locus of this loss function contains a continuum of parameters"). A loose synonym in this case is "weight vector."

Both senses are in common usage in the literature as in this tutorial.

The simplest neural network architecture models a single "neuron" with d inputs $x_1, \ldots, x_d$. Each input x_i is multiplied by an incoming weight w_i to produce the neuron's output. We formalise this case in Example 3.2.

Example 3.2 (A linear neuron)**.** Let $\mathcal{X} = \mathbb{R}^d$ for some positive integer d and let $\mathcal{Y} = \mathbb{R}$. Define a parameter space $\mathcal{W} = \mathbb{R}^d$. Define a parameter–function map that maps a column vector $w \in \mathcal{W}$ to the function $f_w : \mathbb{R}^d \to \mathbb{R}$ such that for $x \in \mathbb{R}^d$,

$$f_w(x) = w^\intercal x = \sum_{i=1}^{d} w_i x_i. \tag{2}$$

Let us generalise this basic neural network in three ways: to add multiple outputs, "depth," and non-linearity. First, we can generalise from a scalar output to a vector output as follows.

Example 3.3 (Multi-linear neural network)**.** Let $\mathcal{X} = \mathcal{Y} = \mathbb{R}^m$ for some positive integer m. Define a parameter space $\mathcal{W} = \mathbb{R}^{m^2}$. Let each parameter $w \in \mathcal{W}$ encode an $m \times m$ matrix $W \in \mathbb{R}^{m \times m}$. The parameter–function map transforms each parameter $w \in \mathcal{W}$ to a function $f_w : \mathbb{R}^m \to \mathbb{R}^m$ such that for $x \in \mathbb{R}^m$,

$$f_w(x) = Wx. \tag{3}$$

If we take each row of W to encode the weight vector of a linear neuron (Example 3.2), we see that we have produced m outputs by stacking m independent linear neurons together. Such a group of neurons is called a **layer**.

Remark 3.4 (Encodings)**.** In Example 3.3, the parameter space is formally $\mathcal{W} = \mathbb{R}^{m^2}$, but we find it convenient to identify each parameter vector $w \in \mathbb{R}^{m^2}$ with the $m \times m$ matrix W it encodes. We would thus write f_W in place of f_w. More generally, whenever the parameters of an architecture naturally decompose into named matrices or vectors, we index the parameter–function map by these structured objects directly rather than by the underlying vector w. The remaining examples in this section demonstrate this convention, and we continue to use it throughout this tutorial.

Next, let's add *depth* to our neural network by composing two layers together.

Example 3.5 (Deep linear network; DLN)**.** Let $\mathcal{X} = \mathcal{Y} = \mathbb{R}^m$ for some positive integer m. Let h be a positive integer and define the parameter space $\mathcal{W} = \mathbb{R}^{2mh}$, with each parameter encoding a pair of matrices $A \in \mathbb{R}^{h \times m}$ and $B \in \mathbb{R}^{m \times h}$. The parameter–function map sends (A, B) to $f_{A,B} : \mathbb{R}^m \to \mathbb{R}^m$ such that for $x \in \mathbb{R}^m$,

$$f_{A,B}(x) = BAx. \tag{4}$$

The above architecture is called a two-layer **deep linear network (DLN)**. The architecture can be interpreted as composing two multi-linear neural networks. The vector of outputs of the neurons of the first network becomes the vector of inputs to the second network. The intermediate output vectors are called **activations**.

Note that if $h \geq m$, the two-layer DLN architecture indexes the same hypothesis class as in Example 3.3, that of linear transforms on $\mathbb{R}^m$. If $h < m$, the hypothesis class includes only transforms with rank up to h.

Finally, we'll add *non-linearity* between pairs of layers. To do so, we introduce a non-linear scalar function $\sigma : \mathbb{R} \to \mathbb{R}$, called an *activation function*, to transform the output of each intermediate neuron. Common examples of activation functions, which we will study in this tutorial, include the following:

1. The **hyperbolic tangent** function $\tanh(z) = \dfrac{e^z - e^{-z}}{e^z + e^{-z}}$.
2. The **rectified linear unit (ReLU)** function $\text{relu}(z) = \max(z, 0)$.

This results in the following non-linear neural network architecture.

Example 3.6 (Multi-layer perceptron; MLP)**.** Define input, output, and parameter spaces as in Example 3.5. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be an activation function. The parameter–function map sends (A, B) to the function $f_{A,B} : \mathbb{R}^m \to \mathbb{R}^m$ such that for $x \in \mathbb{R}^m$,

$$f_{A,B}(x) = B \cdot \sigma(Ax), \tag{5}$$

where we lift the activation function to operate element-wise over column vectors $Ax \in \mathbb{R}^h$.

This kind of architecture is called a **multi-layer perceptron (MLP).** Note that the two-layer DLN is recovered if we use the identity function as an activation function. However, if we use a non-linear activation function, we can index a much richer hypothesis class.

The expressivity of these architectures is limited, however, by the omission of a basic detail—the inclusion of a bias parameter for each neuron. We invite the reader to correct this omission as our first exercise, which serves as a chance to familiarise oneself with the concept of a parameter–function map.

Exercise 3.1 (Biased neurons)**.** A biased neuron is a neuron with an additional parameter that is added to the weighted sum of its inputs before it produces its output. Bias parameters allow each layer to represent an *affine* transform, rather than just a linear transform. For each of Examples 3.2, 3.3, 3.5 and 3.6, extend the example to use biased neurons. Precisely define the parameter space and the parameter–function map in each case.

Modern deep learning leverages more involved parameter–function maps. Neural network architectures are typically defined in a modular fashion, including linear (or affine) layers and non-linear layers like the above, along with more specialised layers (well-known examples including convolutional layers, residual layers, and attention layers).

3.2 Supervised deep learning and loss functions

In a typical supervised deep learning setting, we are given a data set of pairs $(x^{(1)}, y^{(1)})$, $\ldots$, $(x^{(n)}, y^{(n)}) \in \mathcal{X} \times \mathcal{Y}$. These pairs exemplify inputs and their corresponding (possibly noisy) outputs from an unknown target function $f : \mathcal{X} \to \mathcal{Y}$ which we would like to approximate using a neural network. That is, we would like to find a parameter $w \in \mathcal{W}$ such that the corresponding function f_w approximates the target function f.

Formally, given an example $(x, y) \in \mathcal{X} \times \mathcal{Y}$, define a **per-example loss** function $L_{x,y} : \mathcal{W} \to \mathbb{R}$ to measure the deviation of $f_w(x)$ from the output y. A typical loss function for vector outputs $\mathcal{Y} \subset \mathbb{R}^m$ is to use the **squared error** loss

$$L_{x,y}(w) = \|f_w(x) - y\|^2. \tag{6}$$

If $\mathcal{Y} = \Delta^{m-1}$ is a space of discrete distributions over m objects, it is typical to use **cross entropy** loss

$$L_{x,y}(w) = -\sum_{i=1}^{m} y(i) \log f_w(i \mid x). \tag{7}$$

Given a per-example loss function and a **data distribution** $q \in \Delta(\mathcal{X} \times \mathcal{Y})$ over the space of examples (representing our possibly-noisy target function), we formalise the objective of supervised learning as finding $w \in \mathcal{W}$ so as to minimise the **population loss** function $L : \mathcal{W} \to \mathbb{R}$ defined such that

$$L(w) = \mathbb{E}_{(x,y)\sim q}[L_{x,y}(w)]. \tag{8}$$

The population loss aggregates differences on outputs for individual inputs into an overall measure of difference between f_w and the target function.

In practice, we often can't evaluate the population loss over the entire input/output space. We instead optimise an estimator based on our data set of n input–output pairs assumed to be sampled independently and identically from q. Define an **empirical loss** function $L_n : \mathcal{W} \to \mathbb{R}$ such that

$$L_n(w) = \frac{1}{n} \sum_{i=1}^{n} L_{x^{(i)}, y^{(i)}}(w). \tag{9}$$

If the per-example loss is squared error, the empirical loss is known as the **mean squared error** objective.

$$L_n(w) = \frac{1}{n} \sum_{i=1}^{n} \left\| f_w(x^{(i)}) - y^{(i)} \right\|^2. \tag{10}$$

Exercise 3.2 (Empirical loss and population loss)**.** Fix $w \in \mathcal{W}$. Prove the following properties of the relationship between the empirical loss $L_n(w)$ and the population loss $L(w)$.

(a) For any n, the empirical loss is an unbiased estimator of the population loss. That is,

$$\mathbb{E}_{(x^{(i)}, y^{(i)}) \sim q}[L_n(w)] - L(w) = 0. \tag{11}$$

(b) As $n \to \infty$, the empirical loss converges almost surely to the population loss (assuming $L(w)$ is finite).

Given a loss function, a **training algorithm** is a search algorithm that aims to find $w \in \mathcal{W}$ such that the loss is approximately minimised. Most modern deep learning algorithms are variants of **stochastic gradient descent (SGD)**, which implements an iterative gradient-based local search of the parameter space using empirical loss on subsamples of the data set.

Through many impressive feats of computer science and hardware/software engineering, we are able to run such training algorithms to find low-loss parameters within parameter spaces with billions of dimensions. The details are vitally important to the success of modern deep learning, but are beyond the scope of this tutorial. We only mention SGD in order to emphasise that *any local search method depends intimately on the properties of the parameter–function map.*

4 What is degeneracy?

In this section, we define degeneracy as a property of parameter–function maps, and study the relationship between this property and other similar notions.

4.1 A definition of degeneracy

Consider a neural network architecture with parameter space $\mathcal{W} \subseteq \mathbb{R}^d$ and parameter–function map $\Phi : \mathcal{W} \to \mathcal{F}$ $(w \mapsto f_w)$. Say the parameter–function map is **degenerate at** $w \in \mathcal{W}$ if there is a non-zero vector $v \in \mathbb{R}^d$ such that the directional derivative of the parameter–function map in the direction v is zero:

$$D_v\Phi(w) = 0. \tag{12}$$

The directional derivative $D_v\Phi(w)$ for non-zero $v \in \mathbb{R}^d$ may be defined variously as

$$D_v\Phi(w) = \frac{v}{\|v\|} \cdot \nabla\Phi(w) = \sum_{i=1}^{d} \frac{v_i}{\|v\|} \frac{\partial\Phi}{\partial w_i}(w) = \lim_{\epsilon \to 0} \frac{\Phi(w + \epsilon v) - \Phi(w)}{\epsilon \|v\|}. \tag{13}$$

We conventionally define $D_0\Phi(w) = 0$. Note that w is fixed, but the directional derivative is still a function (of the same type as $\Phi(w)$; $D_v\Phi(w) : \mathcal{X} \to \mathcal{Y}$), since it represents an infinitesimal change in $\Phi(w)$ in response to a change in w in the direction v. In (12) we mean that this function is identically zero for all inputs $x \in \mathcal{X}$ for the given $w \in \mathcal{W}$.

Intuitively, the parameter–function map is degenerate at $w \in \mathcal{W}$ if there is a direction in which we can infinitesimally perturb w without changing the function f_w.

This definition of degeneracy applies to a single point in parameter space. As we shall soon see, it may be the case that the parameter–function map is degenerate at some points but not at others. We can clarify the situation with new terminology.

- We say that a parameter–function map is **somewhere degenerate** if it is degenerate at *any* point in parameter space.
- We say that a parameter–function map is **everywhere degenerate** if it is degenerate at *all* points in parameter space.

By convention, if we say that the parameter–function map is simply **degenerate** (without specifying somewhere, everywhere, or at a particular point), we mean that it is *somewhere degenerate*.

The following exercises explore this definition in some toy parametrisations of a simple function class (constants), displaying in the simplest possible setting some basic forms of degeneracy that will arise repeatedly throughout the tutorial.

Exercise 4.1 (Parametrising the space of constants)**.** Let $\mathcal{X} = \{*\}$ and $\mathcal{Y} = \mathbb{R}$, so that we have a hypothesis class of constants. Consider the scalar parameter space $\mathcal{W} = \mathbb{R}$ and the parameter–function map that maps w to $f_w = w$ (the output is just the parameter itself).

(a) What is the directional derivative of the parameter–function map in direction $v = 1$? [*Hint:* what kind of derivative does this reduce to?]

(b) At which points in the parameter space is this parameter–function map degenerate, if any?

Exercise 4.2 (Degeneracy from raising a parameter to a power)**.** Again, consider a hypothesis class of constants and the scalar parameter space $\mathcal{W} = \mathbb{R}$. This time, define a parameter–function map that maps w to $f_w = w^3$.

(a) Show that this architecture indexes exactly the same hypothesis class as the architecture described in Exercise 4.1.

(b) What is the directional derivative of the parameter–function map in direction $v = 1$? [*Hint:* what kind of derivative does this reduce to?]

(c) At which points in the parameter space is the parameter–function map degenerate, if any?

Exercise 4.3 (Degeneracy from multiplying two parameters together)**.** Again, consider a hypothesis class of constants. Consider this time the two-dimensional parameter space $\mathcal{W} = \mathbb{R}^2$. Define a parameter–function map that maps $w = (a, b)$ to $f_w = a \cdot b$.

(a) Show that this architecture indexes exactly the same hypothesis class as the architecture described in Exercise 4.1.

(b) What is the directional derivative of the parameter–function map in direction $v = (1, 0)$? [*Hint:* what kind of derivative does this reduce to?]

(c) What is the directional derivative of the parameter–function map in direction $v = (0, 1)$? [*Hint:* what kind of derivative does this reduce to?]

(d) At which points in the parameter space is the parameter–function map degenerate *in these directions,* if any?

(e) At which points in the parameter space is the parameter–function map degenerate, if any?

Exercise 4.4 (Degeneracy from adding two parameters together)**.** Again, consider a hypothesis class of constants. Consider this time the two-dimensional parameter space $\mathcal{W} = \mathbb{R}^2$. Define a parameter–function map that maps $w = (a, b)$ to $f_w = a + b$.

(a) Show that this architecture indexes exactly the same hypothesis class as the architecture described in Exercise 4.1.

(b) What is the directional derivative of the parameter–function map in direction $v = (1, -1)$?

(c) At which points in the parameter space is the parameter–function map degenerate, if any?

4.2 Degeneracy and continuous symmetries

A common source of degeneracy in neural network architectures is the presence of continuous symmetries of the parameter–function map. A continuous symmetry traces out a curve of functionally equivalent parameters. The tangent to this curve is a degenerate direction. In this section, we will explore this kind of symmetry and some examples from deep learning.

A **symmetry** of a parameter–function map Φ is a transformation of parameter space that maps parameters while preserving the implemented function. That is, a transformation $T : \mathcal{W} \to \mathcal{W}$ is a symmetry if for all $w \in \mathcal{W}$, we have $\Phi(w) = \Phi(T(w))$ ($f_w = f_{T(w)}$).

A **continuous symmetry** of a parameter–function map Φ is a family of transformations $\{T_t : \mathcal{W} \to \mathcal{W}\}_{t\in\mathbb{R}}$ indexed by a parameter $t \in \mathbb{R}$, such that

(i) T_t is a symmetry ($\Phi \circ T_t = \Phi$) for all $t \in \mathbb{R}$,

(ii) T_0 is the identity transformation on $\mathcal{W}$, and

(iii) The map $t \mapsto T_t(w)$ is differentiable for each $w \in \mathcal{W}$.

The continuous symmetry is **trivial at** w if $\frac{\mathrm{d}}{\mathrm{d}t}\big|_{t=0} T_t(w) = 0$, or else **non-trivial at** w.

Exercise 4.5 (Continuous symmetry example)**.** Recall the parameter–function map from Exercise 4.4, with $\mathcal{W} = \mathbb{R}^2$ and with $(a, b) \in \mathcal{W}$ mapping to the constant function $f_{a,b} = a + b$. Consider the family of transformations $T_t : \mathcal{W} \to \mathcal{W}$ for $t \in \mathbb{R}$ such that

$$T_t(a, b) = (a + t, b - t).$$

(a) Describe the effect of this family of transformations on the parameter space.

(b) Show that this family of transformations is a continuous symmetry.

(c) Show that this continuous symmetry is non-trivial everywhere.

At each parameter, a non-trivial continuous symmetry traces out a curve of functionally equivalent parameters. This indicates the presence of a direction in parameter space in which the function does not change. The following proposition formalises this connection between continuous symmetries and degeneracy.

Proposition 4.1. *If a parameter–function map Φ admits a continuous symmetry $\{T_t\}$ that is non-trivial at w, then Φ is degenerate at w.*

Proof. Define the curve in parameter space traced by the continuous symmetry, $\gamma : \mathbb{R} \to \mathcal{W}$ with $\gamma(t) = T_t(w)$. Fix $t_0 \in \mathbb{R}$. Since T_t is a continuous symmetry, the derivative along

the curve at t_0 vanishes:

$$\begin{aligned} D_{\gamma'(t_0)}\Phi(\gamma(t_0)) &\propto \gamma'(t_0) \cdot \nabla\Phi(\gamma(t_0)) \\ &= \frac{\mathrm{d}}{\mathrm{d}t}\bigg|_{t=t_0} \Phi(\gamma(t)) && \text{(chain rule)} \\ &= \frac{\mathrm{d}}{\mathrm{d}t}\bigg|_{t=t_0} \Phi(w) && \text{(symmetry, } \Phi \circ T_t = \Phi\text{)} \\ &= 0. && \text{(}\Phi(w) \text{ is constant in } t\text{)} \end{aligned}$$

In particular, at $t = 0$, we have $0 = D_{\gamma'(0)}\Phi(\gamma(0)) = D_{\gamma'(0)}\Phi(w)$ since $\gamma(0) = w$ by identity. Since T_t is non-trivial at w, $\gamma'(0)$ is non-zero, so $\gamma'(0)$ is a degenerate direction at w. □

Corollary 4.2. *If a parameter–function map Φ admits a continuous symmetry $\{T_t\}$ that is non-trivial at every parameter $w \in \mathcal{W}$, then Φ is everywhere degenerate.*

The following exercises exhibit some continuous symmetries in two neural network architectures, a small ReLU MLP and a toy autoencoder. Similar architectures are studied in more detail from an SLT perspective by Carroll (2021) and Chen et al. (2023) respectively.

Exercise 4.6 (ReLU scaling symmetry)**.** Consider a two-layer MLP (Example 3.6) with $\sigma = \mathrm{relu}$, scalar inputs and outputs ($m = 1$), and a single hidden unit ($h = 1$). The parameter space is $\mathcal{W} = \mathbb{R}^2$ with parameters (a, b), and the parameter–function map is $f_{a,b}(x) = b \cdot \mathrm{relu}(ax)$.

(a) Show that relu is *positively homogeneous*: $\mathrm{relu}(\alpha z) = \alpha\,\mathrm{relu}(z)$ for all $\alpha > 0$ and $z \in \mathbb{R}$.

(b) Define $T_t(a, b) = (e^t a,\ e^{-t} b)$ for $t \in \mathbb{R}$. Show that $\{T_t\}$ is a continuous symmetry of this parameter–function map.

(c) For a fixed parameter $(a, b) \neq (0, 0)$, compute the degenerate direction arising from this symmetry at (a, b).

(d) Show that the symmetry is trivial at the origin. Is this parameter–function map degenerate at the origin?

Exercise 4.7 (Rotation symmetry in a linear autoencoder)**.** Consider a *linear autoencoder* with bottleneck dimension h and ambient dimension $m \geq h$. The parameter space encodes a single matrix $W \in \mathbb{R}^{h \times m}$ and the parameter–function map sends W to the function $\Phi(W) = f_W : \mathbb{R}^m \to \mathbb{R}^m$ where

$$f_W(x) = W^\intercal W x$$

for $x \in \mathbb{R}^m$. Here, W serves simultaneously as encoder ($x \mapsto Wx \in \mathbb{R}^h$) and decoder ($z \mapsto W^\intercal z \in \mathbb{R}^m$).

(a) Show that for any orthogonal matrix $R \in \mathbb{R}^{h \times h}$ (i.e., any $h \times h$ matrix such that $R^\intercal R = I$), the map $T_R(W) = RW$ is a symmetry.

(b) Specialise to $h = 2$. Using the rotation matrices

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

show that $T_\theta(W) = R(\theta)W$ defines a continuous symmetry.

(c) For a fixed parameter $W \in \mathbb{R}^{2\times m}$, compute the degenerate direction that arises from this symmetry.

(d) For general bottleneck dimension h, how many independent continuous symmetries does the orthogonal group $O(h)$ contribute? [*Hint:* what is the dimension of $O(h)$?]

4.3 Localised degeneracy

We have seen that a non-trivial continuous symmetry traces out curves of functionally equivalent parameters throughout the entire parameter space, and the tangent directions to these curves are degenerate directions at every point. However, not all degenerate directions arise from globally continuous symmetries. In fact, the more interesting case is when degeneracy affects only some parameters and not others.

Within certain subsets of parameter space, many neural network architectures (including those studied in Exercises 4.6 and 4.7) exhibit degenerate directions that cannot be defined in terms of continuous transformations that are globally symmetries. In this section, we explore localised degeneracies of this kind.

We begin with some examples of transformations that are only continuous symmetries within certain subsets of parameter space.

Exercise 4.8 (Localised symmetry example)**.** Recall the parameter–function map from Exercise 4.3, with $\mathcal{W} = \mathbb{R}^2$ and with $(a, b) \in \mathcal{W}$ mapping to the constant function $f_{a,b} = a \cdot b$. Consider the two families of transformations $T_t^{(a)}, T_t^{(b)} : \mathcal{W} \to \mathcal{W}$ for $t \in \mathbb{R}$ such that

$$T_t^{(a)}(a, b) = (a + t, b), \qquad T_t^{(b)}(a, b) = (a, b + t).$$

(a) Describe the effect of each family of transformations on the parameter space.

(b) In which subset of parameter space does each family of transformations constitute a symmetry for all $t \in \mathbb{R}$?

(c) Compare your results to your answer to Exercise 4.3(d).

The above example is technically a two-layer DLN with $m = h = 1$. Let us now explore localised degeneracy in a general two-layer DLN and then in a non-trivial MLP.

Exercise 4.9 (Degeneracy in the two-layer DLN)**.** Consider the two-layer DLN from Example 3.5, with $h = m$. The parameter space encodes two matrices $A, B \in \mathbb{R}^{m\times m}$, and the parameter–function map sends (A, B) to the function $\Phi(A, B) = f_{A,B} : \mathbb{R}^m \to \mathbb{R}^m$ such that $f_{A,B}(x) = BAx$ for $x \in \mathbb{R}^m$.

(a) Let $(\delta A, \delta B)$ be a unit perturbation in parameter space (a unit vector in the underlying parameter space $\mathbb{R}^{2m^2}$, decoded into a pair of matrices). Show that the directional derivative of the parameter–function map at (A, B) in direction $(\delta A, \delta B)$ is the linear map

$$D_{(\delta A, \delta B)}\Phi(A, B) = B\,\delta A + \delta B\,A.$$

That is, $D_{(\delta A, \delta B)}\Phi(A, B)x = (B\,\delta A + \delta B\,A)x$. [*Hint:* use the limit definition, (13).]

(b) Show that at the zero parameter $(A, B) = (0, 0)$, every direction in parameter space is degenerate. Count the number of dimensions in the subspace of degenerate directions.

(c) Suppose both A and B are invertible. Show that $(\delta A, \delta B)$ is a degenerate direction if and only if $\delta B = -B\,\delta A\,A^{-1}$. Count the number of dimensions in the subspace of degenerate directions.

(d) Now consider the general case. Fix A and B, and let $r_A = \operatorname{rank}(A)$ and $r_B = \operatorname{rank}(B)$. Show that the dimensionality of the space of degenerate directions is

$$m^2 + (m - r_A)(m - r_B).$$

[*Hint:* The following is a guide to one possible approach.]

(i) Describe the image of the linear map T_A sending $\delta B \in \mathbb{R}^{m^2}$ to $\delta B\,A \in \mathbb{R}^{m^2}$ in terms of the row space of A. Show that the dimensionality of this image (the rank of the linear map T_A) is mr_A.

(ii) Similarly, describe the image of the linear map T_B sending $\delta A \in \mathbb{R}^{m^2}$ to $B\,\delta A \in \mathbb{R}^{m^2}$ in terms of the column space of B. Show that the dimensionality of this image (the rank of the linear map T_B) is mr_B.

(iii) Describe the intersection of the images of T_A, T_B in terms of the row/column spaces of A, B. Show that the dimensionality of this intersection is $r_A r_B$.

(iv) Consider a third linear map, T_D, sending $(\delta A, \delta B) \in \mathbb{R}^{2m^2}$ to $B\,\delta A + \delta B\,A \in \mathbb{R}^{m^2}$. Describe the image of this linear map in terms of those of T_A and T_B. Compute the rank of this linear map using Grassmann's identity.

(v) What is the nullity of T_D? Why is this the same as the dimensionality of the space of degenerate directions?

[*Hint:* Alternatively, substitute singular value decompositions $A = U_1\Sigma_1 V_1^\intercal$ and $B = U_2\Sigma_2 V_2^\intercal$ into the degeneracy condition $B\,\delta A + \delta B\,A = 0$, and change variables to absorb the orthogonal factors. The condition becomes an equation between two diagonally-scaled matrices, whose solutions can be counted entry by entry.]

Exercise 4.10 (Redundant units in an MLP)**.** Consider a single-hidden-layer MLP with scalar inputs and outputs, h hidden units with activation function $\sigma : \mathbb{R} \to \mathbb{R}$, and an output bias. The parameter space is $\mathcal{W} = \mathbb{R}^{3h+1}$ with parameters $w =$

$(a_1, b_1, c_1, \ldots, a_h, b_h, c_h, d)$, and the parameter–function map is

$$f_w(x) = d + \sum_{i=1}^{h} b_i \, \sigma(a_i x + c_i).$$

Here, for each hidden unit i, a_i is the incoming weight, b_i is the outgoing weight, and c_i is the bias. The output bias is d.

(a) Suppose $b_i = 0$ (unit i has zero outgoing weight). Show that the directions δa_i and δc_i are degenerate at w.

(b) Suppose $a_i = 0$ (unit i has zero incoming weight). Show that the direction $(\delta b_i, \delta d) = (1, -\sigma(c_i))$ (with all other components zero) is degenerate at w.

(c) Suppose $h \geq 2$ and $(a_i, c_i) = (a_j, c_j)$ for some $i \neq j$ (units i and j have the same incoming weights and biases). Show that the direction $(\delta b_i, \delta b_j) = (1, -1)$ (with all other components zero) is degenerate at w.

Remark 4.3 (Localised symmetries in neural networks)**.** Each degenerate direction identified in the exercises above corresponds to a localised symmetry: a curve of functionally equivalent parameters that continuously extends only within a subset of parameter space.

1. In the DLN, the global change-of-basis symmetry $(A, B) \mapsto (GA, BG^{-1})$ for invertible $G \in \mathbb{R}^{m \times m}$ accounts for all m^2 degenerate directions whenever A or B has full rank. However, as we showed in Exercise 4.9, when both A and B drop rank, the number of degenerate directions grows by $(m - r_A)(m - r_B)$. These extra directions are localised to the rank-deficient region of parameter space.

2. In the MLP, at a parameter with $b_i = 0$, the path $t \mapsto (\ldots, a_i + t, b_i, c_i, \ldots)$ traces equivalent parameters within $\{b_i = 0\}$ (since f_w does not depend on a_i when $b_i = 0$), but this does not extend to a symmetry at nearby parameters with $b_i \neq 0$. Similarly, at a parameter with $(a_i, c_i) = (a_j, c_j)$, the path transferring weight between units i and j is a localised version of the sum symmetry from Exercise 4.5.

Remark 4.4 (Rank of a neural network parameter)**.** In both exercises above, the degree of degeneracy at a parameter is controlled by the amount of redundant capacity in the network. For the two-layer DLN, this is measured by the rank of the product BA: a rank-r linear map can be implemented with a hidden dimension of r, leaving $m - r$ dimensions redundant. For a general single-hidden-layer MLP, this idea generalises in that we can define the rank of a parameter w as the minimum number of hidden units needed to implement f_w (Farrugia-Roberts, 2022, 2024). In the linear case, this "neural network rank" coincides with the matrix rank of BA. In both settings, lower rank corresponds to a higher number of degenerate directions.

Remark 4.5 (Measure zero degenerate sets)**.** When a parameter–function map is degenerate somewhere but not everywhere, it is often the case that it is degenerate only within a measure-zero subset of parameter space. This means that sampling parameters uniformly at random results in a degenerate parameter with probability zero. However, this does not mean that the degeneracies can be dismissed. The learning process applies a non-random selection pressure and may select parameters from a measure-zero set. Moreover,

the existence of degenerate parameters can have practical consequences for nearby non-degenerate parameters, and the collective neighbourhoods of all degenerate parameters comprise a non-measure-zero subset of parameter space.

4.4 Degeneracy and the loss landscape

In the preceding sections, we studied degeneracy as a property of parameter–function maps. We now consider the implications of degeneracy in these maps for the *loss landscape* in which deep learning algorithms operate.

Recall the definitions of loss functions from Section 3. In what follows, we assume that the per-example loss $L_{x,y}(w)$ depends on w only through $\Phi(w)$, and work with the population loss $L(w) = \mathbb{E}_{(x,y)\sim q}[L_{x,y}(w)]$. We assume that L is twice-differentiable with respect to w.

Let us begin with some basic observations about the relationship between degeneracy in parameter–function maps and directional derivatives in the loss landscape.

Exercise 4.11 (Directional derivatives of the loss)**.** Let $L : \mathcal{W} \to \mathbb{R}$ be a population loss function satisfying the assumptions described above.

(a) Show that if Φ is degenerate at w in direction v, then the directional derivative of the loss vanishes in the same direction: $D_v L(w) = 0$.

(b) Suppose $\mathcal{W} = \mathbb{R}^d$. Show that if $d > 1$, then for *any* $w \in \mathcal{W}$, there exists a nonzero v such that $D_v L(w) = 0$. [*Hint:* consider separately the cases $\nabla L(w) = 0$ and $\nabla L(w) \neq 0$.]

We see that parameter–function map degeneracy implies flat loss directions, but a vanishing directional derivative of the loss landscape is commonplace. To find a satisfying definition of degeneracy in a loss landscape, we must look at second-order information.

Define the **Hessian matrix** $H(w) \in \mathbb{R}^{d\times d}$ of the loss at w by

$$H(w)_{jk} = \frac{\partial^2 L}{\partial w_j \partial w_k}(w). \tag{14}$$

The Hessian at w is sometimes denoted $\nabla^2 L(w)$. The Hessian captures the local curvature of the loss landscape via the quadratic approximation

$$L(w+\delta) \approx L(w) + \nabla L(w)^\intercal \delta + \tfrac{1}{2}\, \delta^\intercal H(w)\, \delta. \tag{15}$$

At a critical point ($\nabla L(w) = 0$), the Hessian determines whether the loss curves upward, downward, or remains flat in each direction.

In particular, at a local minimum, $H(w)$ is positive semidefinite. We can therefore classify local minima as follows:

- A local minimum w is **regular** (or **non-degenerate**, or **Morse**) if $H(w)$ is positive definite.
- A local minimum w is **degenerate** (or **non-Morse**) if $H(w)$ is singular (has a zero eigenvalue).

Exercise 4.12 (Some examples of loss landscape degeneracy)**.** Consider the parameter space $\mathcal{W} = \mathbb{R}^2$ with the identity parameter–function map $\Phi = \mathrm{id}$, so that we identify parameters with the functions they implement (cf., Exercise 4.1). Consider the family of loss functions $L_{k,l}(a, b) = a^{2k} + b^{2l}$ for non-negative integers k and l.

(a) Show that the origin is a global minimum of $L_{k,l}$ for all non-negative k and l. For which k and l is it the unique global minimum?

(b) Show that if $k = l = 1$, then the origin is a non-degenerate (regular) minimum. [*Hint:* compute the Hessian.]

(c) Show that if $k > 1$ and $l \geq 1$, then the origin is a degenerate minimum.

(d) Show that if $k = 0$ and $l \geq 1$, then the origin is a degenerate minimum.

Note that the Hessian degeneracy in Exercises 4.12(c) and 4.12(d) arises for qualitatively different reasons. In Exercise 4.12(d), the loss is constant along the a-direction to all orders—a genuinely flat direction. In Exercise 4.12(c), the loss does increase along the a-direction, just more slowly than quadratically (as a^4, a^6, etc., depending on k). Moreover, different values of k give different rates of increase. The Hessian cannot distinguish any of these cases: flat, quartic, and so on all appear the same from the perspective of the Hessian rank. We will later see approaches that can distinguish these degrees of degeneracy.

So much for defining loss landscape degeneracy. What is the relationship between this kind of degeneracy and degeneracy of the parameter–function map?

The relationship is subtle, and generally depends on the choice of loss function. The following exercise explores this relationship through examples.

Exercise 4.13 (Contrasting parametric versus loss degeneracy)**.**

(a) Observe that the identity parameter–function map on $\mathcal{W} = \mathbb{R}^2$ is non-degenerate everywhere. With this in mind, what do your answers to Exercises 4.12(b) to 4.12(d) exemplify about the logical relationship between parameter–function map degeneracy and loss landscape degeneracy?

(b) Consider again the identity parameter–function map on $\mathcal{W} = \mathbb{R}^2$. Consider the loss function $L_\circ(a, b) = (a^2 + b^2 - 1)^2$. Find a global minimum of $L_\circ$ and show that it is a degenerate critical point. What does this example reveal about the logical relationship between parameter–function map degeneracy and loss landscape degeneracy?

(c) Now consider the cubic parameter–function map $\Phi(\alpha, \beta) = (\alpha^3, \beta^3)$.

  (i) Argue that Φ is degenerate at the origin (cf., Exercise 4.2).

  (ii) If we take $L_{k,l}$ from Exercise 4.12 to be defined on the constant function (a, b) (rather than the equivalent underlying parameter), then in this parameterisation, that loss function becomes $L_{k,l}(\alpha, \beta) = (\alpha^3)^{2k} + (\beta^3)^{2l}$. Show that the origin is a degenerate minimum of $L_{k,l}$ for all $k \geq 0$ and $l \geq 0$ under this parametrisation.

(iii) What does this example reveal about the logical relationship between parameter–function map degeneracy and loss landscape degeneracy?

(d) Consider the product parameter–function map $\Phi(a, b) = a \cdot b$ from Exercise 4.3, and define $L(a, b) = \frac{1}{2}(ab - 1)^2$.

(i) Show that $(0, 0)$ is a critical point and that Φ is degenerate in every direction at $(0, 0)$.

(ii) Compute the Hessian at $(0, 0)$ and show it is nonsingular.

(iii) What does this example reveal about the logical relationship between parameter–function map degeneracy and loss landscape degeneracy?

Exercise 4.14 (Hessian degeneracy for a teacher–student loss)**.** Let $\Phi : \mathcal{W} \to \mathcal{F}$ be a parameter–function map, smooth in both $w \in \mathcal{W}$ and $x \in \mathcal{X}$. Let $\mathcal{Y} = \mathbb{R}$ (scalar outputs). Let q be an input distribution, strictly positive on $\mathcal{X}$.
Fix a teacher network $w_0 \in \mathcal{W}$ and set $f_\star = f_{w_0}$. Define a quadratic population loss

$$L(w) = \frac{1}{2}\,\mathbb{E}_{X\sim q}\left[(f_w(X) - f_\star(X))^2\right].$$

Write the residual $r_w = f_w - f_\star$. Throughout assume ∇_w and ∇_w^2 may be exchanged with $\mathbb{E}_q$.

(a) (First derivative.) Show that

$$\nabla_w L(w) = \mathbb{E}_{X\sim q}\left[r_w(X)\,\nabla_w f_w(X)\right].$$

Conclude that $\nabla_w L(w_0) = 0$.

(b) (Second derivative.) Show that

$$\nabla_w^2 L(w) = \mathbb{E}_{X\sim q}\left[(\nabla_w f_w(X))(\nabla_w f_w(X))^\top + r_w(X)\,\nabla_w^2 f_w(X)\right].$$

Hint: use the scalar-product rule

$$\nabla_{\mathbf{w}}\big(s(\mathbf{w})\,\mathbf{v}(\mathbf{w})\big) = \big(\nabla_{\mathbf{w}} s(\mathbf{w})\big)\mathbf{v}(\mathbf{w})^\top + s(\mathbf{w})\,\nabla_{\mathbf{w}}\mathbf{v}(\mathbf{w}).$$

$\nabla_w^2 L(w) \in \mathbb{R}^{d\times d}$ is the Hessian $H(w)$. Writing

$$G(w) = \mathbb{E}_{X\sim q}\left[(\nabla_w f_w(X))(\nabla_w f_w(X))^\top\right] \quad \text{and} \quad R(w) = \mathbb{E}_{X\sim q}\left[r_w(X)\,\nabla_w^2 f_w(X)\right],$$

we have

$$H(w) = G(w) + R(w).$$

Conclude that $H(w_0) = G(w_0)$.

(c) Show that for every $v \in \mathbb{R}^d$,

$$v^\top G(w) v = \|v\|^2\,\mathbb{E}_{X\sim q}\left[\big(D_v\Phi(w)(X)\big)^2\right].$$

Conclude that $G(w)$ and $H(w_0)$ are positive semi-definite.

(d) Show that if Φ is degenerate at w_0 in non-zero direction v, then $H(w_0)v = 0$.

(e) Show that if v is non-zero and $H(w_0)v = 0$, then Φ is degenerate at w_0 in direction v.

(f) Conclude that

$$\ker H(w_0) = \{v \in \mathbb{R}^d : D_v\Phi(w_0) = 0\},$$

and that $H(w_0)$ is positive definite if and only if Φ is non-degenerate at w_0.

References and further reading

Liam Carroll. *Phase Transitions in Neural Networks*. Master's thesis, School of Mathematics and Statistics, the University of Melbourne, 2021. Access via Daniel Murfet. Cited on page 14.

Zhongtian Chen, Edmund Lau, Jake Mendel, Susan Wei, and Daniel Murfet. Dynamical versus Bayesian phase transitions in a toy model of superposition. 2023. Preprint arXiv:2310.06301 [cs.LG]. Cited on page 14.

Matthew Farrugia-Roberts. *Structural Degeneracy in Neural Networks*. Master's thesis, School of Computing and Information Systems, the University of Melbourne, 2022. Access via Matthew Farrugia-Roberts. Cited on page 17.

Matthew Farrugia-Roberts. Losslessly compressible neural network parameters. In *Workshop on Machine Learning and Compression*. 2024. Access via OpenReview. Workshop at NeurIPS 2024. Cited on page 17.

Zach Furman. Singular learning theory: Exercises. 2024. Access via LessWrong. Cited on page 2.

Zach Furman. Mysteries of deep learning. Reading guide, Iliad Intensive, 2026. Access via Iliad Intensive. Cited on page 4.

Vaishnavh Nagarajan and J. Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In *Advances in Neural Information Processing Systems 32*. 2019. Access via arXiv. Cited on page 5.

Simon Pepin Lehalleur, Jesse Hoogland, Matthew Farrugia-Roberts, Susan Wei, Alexander Gietelink Oldenziel, George Wang, Liam Carroll, and Daniel Murfet. You are what you eat–AI alignment requires understanding how data shapes structure and generalisation. 2025. Preprint arXiv:2502.05475 [cs.LG]. Cited on page 2.

Susan Wei, Daniel Murfet, Mingming Gong, Hui Li, Jesse Gell-Redman, and Thomas Quella. Deep learning is singular, and that's good. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):10473–10486, 2023. Cited on page 2.

Andrew Gordon Wilson. Deep learning is not so mysterious or different. In *Proceedings of the 42nd International Conference on Machine Learning*. 2025. Access via arXiv. Cited on pages 4 and 6.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*. 2017. Access via arXiv. Cited on pages 4 and 5.

A Solutions

Solution to Exercise 3.1 (Biased neurons) 24
Solution to Exercise 3.2 (Empirical loss and population loss) 24

Solution to Exercise 4.1 (Parametrising the space of constants) 25
Solution to Exercise 4.2 (Degeneracy from raising a parameter to a power) 25
Solution to Exercise 4.3 (Degeneracy from multiplying two parameters together) . . 25
Solution to Exercise 4.4 (Degeneracy from adding two parameters together) 26
Solution to Exercise 4.5 (Continuous symmetry example) 26
Solution to Exercise 4.6 (ReLU scaling symmetry) 26
Solution to Exercise 4.7 (Rotation symmetry in a linear autoencoder) 27
Solution to Exercise 4.8 (Localised symmetry example) 28
Solution to Exercise 4.9 (Degeneracy in the two-layer DLN) 28
Solution to Exercise 4.10 (Redundant units in an MLP) 29
Solution to Exercise 4.11 (Directional derivatives of the loss) 30
Solution to Exercise 4.12 (Some examples of loss landscape degeneracy) 31
Solution to Exercise 4.13 (Contrasting parametric versus loss degeneracy) 31
Solution to Exercise 4.14 (Hessian degeneracy for a teacher–student loss) 32

A.1 Solutions to exercises from Section 3

Solution to Exercise 3.1 (Biased neurons)**.** For each architecture, we add a bias vector to each layer's output before the next transformation (or activation function) is applied.

(a) **A linear neuron with bias** (Example 3.2). Let $\mathcal{W} = \mathbb{R}^{d+1}$, encoding a weight vector $w \in \mathbb{R}^d$ and a scalar bias $b \in \mathbb{R}$. The parameter–function map sends (w, b) to $f_{w,b} : \mathbb{R}^d \to \mathbb{R}$ defined by

$$f_{w,b}(x) = w^\intercal x + b.$$

(b) **Multi-linear neural network with bias** (Example 3.3). Let $\mathcal{W} = \mathbb{R}^{m^2+m}$, encoding a matrix $W \in \mathbb{R}^{m\times m}$ and a bias vector $b \in \mathbb{R}^m$. The parameter–function map sends (W, b) to $f_{W,b} : \mathbb{R}^m \to \mathbb{R}^m$ defined by

$$f_{W,b}(x) = Wx + b.$$

(c) **Deep linear network with bias** (Example 3.5). Let $\mathcal{W} = \mathbb{R}^{2mh+h+m}$, encoding matrices $A \in \mathbb{R}^{h\times m}$, $B \in \mathbb{R}^{m\times h}$ and bias vectors $b_A \in \mathbb{R}^h$, $b_B \in \mathbb{R}^m$. The parameter–function map sends (A, b_A, B, b_B) to $f_w : \mathbb{R}^m \to \mathbb{R}^m$ defined by

$$f_w(x) = B(Ax + b_A) + b_B.$$

(d) **Multi-layer perceptron with bias** (Example 3.6). The parameter space is $\mathcal{W} = \mathbb{R}^{2mh+h+m}$ as above. The parameter–function map sends (A, b_A, B, b_B) to $f_w : \mathbb{R}^m \to \mathbb{R}^m$ defined by

$$f_w(x) = B \cdot \sigma(Ax + b_A) + b_B.$$ □

Solution to Exercise 3.2 (Empirical loss and population loss)**.**

(a) Since the data pairs $(x^{(i)}, y^{(i)})$ are sampled independently and identically from q, each per-example loss $L_{x^{(i)},y^{(i)}}(w)$ is an identically distributed random variable with expectation

$$\mathbb{E}_{(x,y)\sim q}[L_{x,y}(w)] = L(w).$$

Therefore, by linearity of expectation,

$$\mathbb{E}[L_n(w)] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} L_{x^{(i)},y^{(i)}}(w)\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[L_{x^{(i)},y^{(i)}}(w)] = \frac{1}{n}\cdot n \cdot L(w) = L(w).$$

(b) The random variables $L_{x^{(1)},y^{(1)}}(w), L_{x^{(2)},y^{(2)}}(w), \ldots$ are independent and identically distributed with common mean $L(w)$. By the strong law of large numbers,

$$L_n(w) = \frac{1}{n}\sum_{i=1}^{n} L_{x^{(i)},y^{(i)}}(w) \xrightarrow{n\to\infty} L(w)$$

almost surely. □

A.2 Solutions to exercises from Section 4

Solution to Exercise 4.1 (Parametrising the space of constants).

(a) Since $\mathcal{W} = \mathbb{R}$ is one-dimensional, the directional derivative in direction $v = 1$ reduces to the ordinary derivative:

$$\frac{\partial \Phi}{\partial w} = \frac{\mathrm{d}}{\mathrm{d}w} w = 1.$$

(b) The directional derivative is $1 \neq 0$ for all $w \in \mathcal{W}$. Therefore, the parameter–function map is *not degenerate* at any point. □

Solution to Exercise 4.2 (Degeneracy from raising a parameter to a power).

(a) The hypothesis class of Exercise 4.1 is $\mathcal{F} = \{f_w = w : w \in \mathbb{R}\} = \mathbb{R}$. The hypothesis class here is $\mathcal{F} = \{f_w = w^3 : w \in \mathbb{R}\}$. Since $w \mapsto w^3$ is a bijection on $\mathbb{R}$, as w ranges over $\mathbb{R}$, w^3 takes every real value exactly once. So $\mathcal{F} = \mathbb{R}$ in both cases.

(b) The directional derivative in direction $v = 1$ is the ordinary derivative:

$$\frac{\partial \Phi}{\partial w} = \frac{\mathrm{d}}{\mathrm{d}w} w^3 = 3w^2.$$

(c) The directional derivative $3w^2 = 0$ if and only if $w = 0$. So the parameter–function map is degenerate at $w = 0$ only. It is somewhere degenerate but not everywhere degenerate. □

Solution to Exercise 4.3 (Degeneracy from multiplying two parameters together).

(a) The hypothesis class is $\mathcal{F} = \{f_w = ab : (a, b) \in \mathbb{R}^2\} = \mathbb{R}$, since for any target $c \in \mathbb{R}$ we can choose, e.g., $a = c$ and $b = 1$. This is the same as in Exercise 4.1.

(b) The directional derivative in direction $v = (1, 0)$ is the partial derivative with respect to a:

$$\frac{\partial \Phi}{\partial a} = \frac{\partial}{\partial a}(ab) = b.$$

(c) The directional derivative in direction $v = (0, 1)$ is the partial derivative with respect to b:

$$\frac{\partial \Phi}{\partial b} = \frac{\partial}{\partial b}(ab) = a.$$

(d) From parts (b) and (c), the direction $(1, 0)$ is degenerate at (a, b) if and only if $b = 0$, and the direction $(0, 1)$ is degenerate if and only if $a = 0$. So the parameter–function map is degenerate in at least one of these coordinate directions precisely on the set $\{(a, b) : a = 0 \text{ or } b = 0\}$, i.e. the union of the two coordinate axes.

(e) The directional derivative in a general direction $v = (v_1, v_2)$ is proportional to

$$v_1 \frac{\partial \Phi}{\partial a} + v_2 \frac{\partial \Phi}{\partial b} = v_1 b + v_2 a.$$

This is a single linear equation in two unknowns (v_1, v_2).

- If $a = b = 0$: the equation $0 = 0$ is trivially satisfied, so every direction is degenerate.
- If $(a, b) \neq (0, 0)$: the coefficient vector (b, a) is nonzero, so the solution space is one-dimensional. A nonzero solution is $v = (-a, b)$: indeed, $(-a)b + b \cdot a = 0$.

Therefore, the parameter–function map is degenerate *everywhere*. Note that the previous part found degeneracy only on the coordinate axes because it checked only the coordinate directions. Checking for parameter–function map degeneracy requires checking all possible directions. □

Solution to Exercise 4.4 (Degeneracy from adding two parameters together).

(a) The hypothesis class is $\mathcal{F} = \{f_w = a + b : (a, b) \in \mathbb{R}^2\} = \mathbb{R}$, since for any target $c \in \mathbb{R}$ we can choose, e.g., $a = c$ and $b = 0$. This is the same as in Exercise 4.1.

(b) The directional derivative in direction $v = (1, -1)$ is

$$1 \cdot \frac{\partial}{\partial a}(a + b) + (-1) \cdot \frac{\partial}{\partial b}(a + b) = 1 - 1 = 0.$$

(c) The direction $v = (1, -1)$ gives a zero directional derivative at *every* point $(a, b) \in \mathcal{W}$. Therefore, the parameter–function map is everywhere degenerate. This reflects the fact that the map $(a, b) \mapsto a + b$ has a global continuous symmetry: translating along the direction $(1, -1)$ does not change the output. □

Solution to Exercise 4.5 (Continuous symmetry example).

(a) The transformation T_t translates the parameter (a, b) by t in the direction $(1, -1)$. The orbit of any parameter (a, b) under this family is the line $\{(a+t, b-t) : t \in \mathbb{R}\}$, which has slope -1 in the (a, b)-plane.

(b) We verify the three conditions of a continuous symmetry.

(i) Symmetry: $\Phi(T_t(a, b)) = (a + t) + (b - t) = a + b = \Phi(a, b)$ for all (a, b) and t.

(ii) Identity: $T_0(a, b) = (a + 0, b - 0) = (a, b)$.

(iii) Differentiability: $t \mapsto T_t(a, b) = (a + t, b - t)$ is linear in t, hence differentiable.

(c) We have $\frac{\mathrm{d}}{\mathrm{d}t} T_t(a, b) = (1, -1)$ for all t. In particular, $\frac{\mathrm{d}}{\mathrm{d}t}\big|_{t=0} T_t(a, b) = (1, -1) \neq (0, 0)$ for every $(a, b) \in \mathcal{W}$. So the symmetry is non-trivial at every parameter. □

Solution to Exercise 4.6 (ReLU scaling symmetry).

(a) We consider three cases. If $z > 0$, then $\alpha z > 0$ (since $\alpha > 0$), so $\mathrm{relu}(\alpha z) = \alpha z = \alpha \mathrm{relu}(z)$. If $z = 0$, then $\mathrm{relu}(\alpha \cdot 0) = 0 = \alpha \cdot 0 = \alpha \mathrm{relu}(z)$. If $z < 0$, then $\alpha z < 0$, so $\mathrm{relu}(\alpha z) = 0 = \alpha \cdot 0 = \alpha \mathrm{relu}(z)$.

(b) We verify the three conditions.

(i) Symmetry: using positive homogeneity with $\alpha = e^t > 0$,

$$f_{T_t(a,b)}(x) = e^{-t} b \cdot \mathrm{relu}(e^t a x) = e^{-t} b \cdot e^t \mathrm{relu}(a x) = b \cdot \mathrm{relu}(a x) = f_{a,b}(x).$$

(ii) Identity: $T_0(a,b) = (e^0 a,\, e^0 b) = (a,b)$.

(iii) Differentiability: $t \mapsto (e^t a,\, e^{-t} b)$ is smooth.

(c) The degenerate direction at (a,b) is

$$\left.\frac{\mathrm{d}}{\mathrm{d}t}\right|_{t=0} T_t(a,b) = \left.\frac{\mathrm{d}}{\mathrm{d}t}\right|_{t=0} (e^t a,\, e^{-t} b) = (a, -b).$$

Since $(a,b) \neq (0,0)$, we have $(a,-b) \neq (0,0)$, so this is a non-zero degenerate direction.

(d) At $(a,b) = (0,0)$, we have $\left.\frac{\mathrm{d}}{\mathrm{d}t}\right|_{t=0} T_t(0,0) = (0,0)$, so the symmetry is trivial.

The parameter–function map is nonetheless degenerate at the origin. Both partial derivatives of Φ vanish there:

$$\left.\frac{\partial}{\partial a} f_{a,b}(x)\right|_{(0,0)} = \lim_{\epsilon \to 0} \frac{f_{\epsilon,0}(x) - f_{0,0}(x)}{\epsilon} = \lim_{\epsilon \to 0} \frac{0 \cdot \mathrm{relu}(\epsilon x)}{\epsilon} = 0,$$

$$\left.\frac{\partial}{\partial b} f_{a,b}(x)\right|_{(0,0)} = \mathrm{relu}(0 \cdot x) = 0.$$

Since $\nabla\Phi(0,0) = 0$, every direction is degenerate at the origin. This degeneracy is not explained by the scaling symmetry. □

Solution to Exercise 4.7 (Rotation symmetry in a linear autoencoder)**.**

(a) For orthogonal R (i.e. $R^\intercal R = I$),

$$f_{RW}(x) = (RW)^\intercal (RW)x = W^\intercal R^\intercal R\, W x = W^\intercal W x = f_W(x).$$

(b) We verify the three conditions.

(i) Symmetry: $R(\theta)$ is orthogonal for all θ, since

$$R(\theta)^\intercal R(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

So $f_{R(\theta)W} = f_W$ by part (a).

(ii) Identity: $R(0) = I$, so $T_0(W) = W$.

(iii) Differentiability: the entries of $R(\theta)W$ are smooth functions of θ (they involve $\cos\theta$ and $\sin\theta$ multiplied by entries of W).

(c) The degenerate direction at W is

$$\left.\frac{\mathrm{d}}{\mathrm{d}\theta}\right|_{\theta=0} R(\theta)W = R'(0)\, W.$$

We compute

$$R'(\theta) = \begin{pmatrix} -\sin\theta & -\cos\theta \\ \cos\theta & -\sin\theta \end{pmatrix}, \qquad R'(0) = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$

Writing $W = \begin{pmatrix} w_1^\intercal \\ w_2^\intercal \end{pmatrix}$ where $w_1, w_2 \in \mathbb{R}^m$ are the two rows, the degenerate direction is the matrix

$$R'(0)\, W = \begin{pmatrix} -w_2^\intercal \\ w_1^\intercal \end{pmatrix},$$

which swaps the two rows and negates one. This direction is non-zero whenever $W \neq 0$.

(d) The orthogonal group $O(h)$ consists of all $h \times h$ matrices satisfying $R^\intercal R = I$. This constraint comprises $\frac{h(h+1)}{2}$ independent scalar equations (the entries of the symmetric matrix $R^\intercal R$ on and above the diagonal). Since R has h^2 entries, the dimension of $O(h)$ is

$$h^2 - \frac{h(h+1)}{2} = \frac{h(h-1)}{2}.$$

Each independent direction in $O(h)$ at the identity gives rise to a one-parameter continuous symmetry and hence a degenerate direction at W; when W has full rank h, these degenerate directions are linearly independent (at rank-deficient W, they may become linearly dependent). These directions are the $h \times h$ skew-symmetric matrices S (satisfying $S^\intercal = -S$), since differentiating $R(t)^\intercal R(t) = I$ at $t = 0$ gives $R'(0)^\intercal + R'(0) = 0$. The space of such matrices has dimension $\frac{h(h-1)}{2}$ (the entries strictly above the diagonal are free, and the rest are determined).

So $O(h)$ contributes $\frac{h(h-1)}{2}$ independent continuous symmetries. For $h = 2$ this gives 1, matching the single rotation symmetry from part (b). □

Solution to Exercise 4.8 (Localised symmetry example).

(a) $T_t^{(a)}$ translates along the a-axis: it shifts the first coordinate by t while leaving the second fixed. $T_t^{(b)}$ translates along the b-axis.

(b) $T_t^{(a)}$ is a symmetry at (a, b) for all t if and only if $f_{a+t,b} = f_{a,b}$ for all t, i.e., $(a+t)b = ab$ for all t. This simplifies to $tb = 0$ for all t, which requires $b = 0$. So $\{T_t^{(a)}\}$ is a symmetry when restricted to the a-axis $\{(a, 0) : a \in \mathbb{R}\}$.

Similarly, $\{T_t^{(b)}\}$ is a symmetry when restricted to the b-axis $\{(0, b) : b \in \mathbb{R}\}$.

(c) From Exercise 4.3(d), the coordinate directions $(1, 0)$ and $(0, 1)$ are degenerate precisely on the coordinate axes: $(1, 0)$ is degenerate on $\{b = 0\}$ and $(0, 1)$ is degenerate on $\{a = 0\}$. This matches exactly: the axis $\{b = 0\}$ is where the a-translation symmetry acts, providing the degenerate direction $(1, 0)$, and the axis $\{a = 0\}$ is where the b-translation symmetry acts, providing $(0, 1)$. □

Solution to Exercise 4.9 (Degeneracy in the two-layer DLN).

(a) Encoding (A, B) and $(\delta A, \delta B)$ as vectors in the underlying parameter space $\mathbb{R}^{2m^2}$, the directional derivative is

$$D_{(\delta A, \delta B)}\Phi(A, B)(x) = \lim_{\epsilon \to 0} \frac{f_{A+\epsilon\delta A,\, B+\epsilon\delta B}(x) - f_{A,B}(x)}{\epsilon}.$$

Expanding the matrix product:

$$\begin{aligned} f_{A+\epsilon\delta A,\, B+\epsilon\delta B}(x) &= (B+\epsilon\delta B)(A+\epsilon\delta A)\,x \\ &= BAx + \epsilon(B\,\delta A + \delta B\,A)\,x + \epsilon^2 \delta B\,\delta A\,x. \end{aligned}$$

Subtracting $f_{A,B}(x) = BAx$, dividing by ϵ, and taking $\epsilon \to 0$ gives

$$D_{(\delta A,\delta B)}\Phi(A,B)(x) = (B\,\delta A + \delta B\,A)\,x.$$

(b) At $(A,B) = (0,0)$, part (a) gives $D_{(\delta A,\delta B)}\Phi(0,0)(x) = (0\cdot\delta A + \delta B\cdot 0)\,x = 0$ for every $(\delta A, \delta B)$ and every x. Every non-zero direction is therefore degenerate. The subspace of degenerate directions is the entire parameter space $\mathbb{R}^{2m^2}$, which has dimension $2m^2$.

(c) By part (a), $(\delta A, \delta B)$ is a degenerate direction if and only if $B\,\delta A + \delta B\,A = 0$, that is, $\delta B\,A = -B\,\delta A$. Since A is invertible, right-multiplying by A^{-1} gives $\delta B = -B\,\delta A\,A^{-1}$. Since $\delta A \in \mathbb{R}^{m\times m}$ is free and δB is uniquely determined, the subspace of degenerate directions has dimension m^2.

(d) (i) The i-th row of $\delta B\,A$ is $(\delta B)_i A$, a linear combination of the rows of A. Therefore the image of T_A consists of all matrices whose rows lie in the row space of A. Each of the m rows can be any vector in the r_A-dimensional row space, so $\operatorname{rank}(T_A) = m\,r_A$.

(ii) Similarly, the j-th column of $B\,\delta A$ is $B\,(\delta A)^j$, a linear combination of the columns of B. The image of T_B consists of all matrices whose columns lie in the column space of B, and $\operatorname{rank}(T_B) = m\,r_B$.

(iii) A matrix lies in $\operatorname{im}(T_A)\cap\operatorname{im}(T_B)$ if and only if its rows lie in the row space of A and its columns lie in the column space of B. Such a matrix can be written as $U\Lambda V$ where $U \in \mathbb{R}^{m\times r_B}$ has columns spanning the column space of B and $V \in \mathbb{R}^{r_A\times m}$ has rows spanning the row space of A, with $\Lambda \in \mathbb{R}^{r_B\times r_A}$ free. The dimensionality of this intersection is therefore $r_A\,r_B$.

(iv) The image of T_D is $\operatorname{im}(T_D) = \operatorname{im}(T_B) + \operatorname{im}(T_A)$, since for any $(\delta A, \delta B)$ we have $T_D(\delta A,\delta B) = T_B(\delta A) + T_A(\delta B)$, and conversely any element of $\operatorname{im}(T_B)+\operatorname{im}(T_A)$ can be realised by choosing appropriate δA and δB. By Grassmann's identity,

$$\begin{aligned} \operatorname{rank}(T_D) &= \dim(\operatorname{im}(T_A) + \operatorname{im}(T_B)) \\ &= \operatorname{rank}(T_A) + \operatorname{rank}(T_B) - \dim(\operatorname{im}(T_A)\cap\operatorname{im}(T_B)) \\ &= m\,r_A + m\,r_B - r_A\,r_B. \end{aligned}$$

(v) By the rank–nullity theorem, the nullity of T_D is

$$\begin{aligned} & 2m^2 - (m\,r_A + m\,r_B - r_A\,r_B) \\ &= 2m^2 - m\,r_A - m\,r_B + r_A\,r_B \\ &= m^2 + (m-r_A)(m-r_B). \end{aligned}$$

The kernel of T_D is exactly the set of $(\delta A,\delta B)$ such that $B\,\delta A + \delta B\,A = 0$, which by part (a) is precisely the subspace of degenerate directions. □

Solution to Exercise 4.10 (Redundant units in an MLP)**.**

(a) At $b_i = 0$, the parameter–function map reduces to

$$f_w(x) = d + 0 \cdot \sigma(a_i x + c_i) + \sum_{j \neq i} b_j \, \sigma(a_j x + c_j) = d + \sum_{j \neq i} b_j \, \sigma(a_j x + c_j).$$

This expression does not involve a_i or c_i at all. Therefore Φ is constant in the a_i and c_i directions at this point, so $D_{\delta a_i}\Phi(w) = 0$ and $D_{\delta c_i}\Phi(w) = 0$. Both are degenerate directions.

Note that this argument is purely algebraic, following from the multiplicative structure $b_i \cdot \sigma(\ldots)$, and requires no conditions on σ (not even continuity or differentiability).

(b) At $a_i = 0$, unit i computes the constant $\sigma(0 \cdot x + c_i) = \sigma(c_i)$ for all x, so

$$f_w(x) = d + b_i \, \sigma(c_i) + \sum_{j \neq i} b_j \, \sigma(a_j x + c_j).$$

In the direction $(\delta b_i, \delta d) = (1, -\sigma(c_i))$, we compute

$$\begin{aligned} f_{w+\epsilon v}(x) &= (d - \epsilon \, \sigma(c_i)) + (b_i + \epsilon) \, \sigma(c_i) + \sum_{j \neq i} b_j \, \sigma(a_j x + c_j) \\ &= d + b_i \, \sigma(c_i) + \epsilon\big(\sigma(c_i) - \sigma(c_i)\big) + \sum_{j \neq i} b_j \, \sigma(a_j x + c_j) = f_w(x). \end{aligned}$$

So $D_{(\delta b_i, \delta d)}\Phi(w) = 0$ and this direction is degenerate. Intuitively, increasing b_i scales up the constant contribution $b_i \, \sigma(c_i)$ to the output, and decreasing d by the same amount compensates exactly.

(c) When $(a_i, c_i) = (a_j, c_j)$, units i and j compute the same activation, so their combined contribution to f_w is

$$b_i \, \sigma(a_i x + c_i) + b_j \, \sigma(a_j x + c_j) = (b_i + b_j) \, \sigma(a_i x + c_i).$$

In the direction $(\delta b_i, \delta b_j) = (1, -1)$, this combined contribution changes by $\sigma(a_i x + c_i) - \sigma(a_j x + c_j) = 0$. All other terms in f_w are unchanged, so $D_{(1,-1)}\Phi(w) = 0$. Again, no conditions on σ are needed. □

Solution to Exercise 4.11 (Directional derivatives of the loss)**.**

(a) Since $L_{x,y}(w)$ depends on w only through $\Phi(w)$, we can write $L_{x,y}(w) = \ell_{x,y}(\Phi(w))$ for some functional $\ell_{x,y}$. If $D_v\Phi(w) = 0$, then $\Phi(w+\epsilon v) = \Phi(w) + O(\epsilon^2)$ as $\epsilon \to 0$, and therefore $L_{x,y}(w + \epsilon v) = \ell_{x,y}(\Phi(w) + O(\epsilon^2)) = L_{x,y}(w) + O(\epsilon^2)$. Taking expectations over $(x, y) \sim q$,

$$L(w + \epsilon v) = L(w) + O(\epsilon^2),$$

so $D_v L(w) = \lim_{\epsilon \to 0} \frac{L(w+\epsilon v) - L(w)}{\epsilon \|v\|} = 0$.

(b) If $\nabla L(w) = 0$, then $D_v L(w) = \frac{v}{\|v\|} \cdot \nabla L(w) = 0$ for every nonzero v.

If $\nabla L(w) \neq 0$, the orthogonal complement $\nabla L(w)^\perp$ has dimension $d - 1 \geq 1$ (since $d > 1$). Pick any nonzero $v \in \nabla L(w)^\perp$. Then $D_v L(w) = \frac{v}{\|v\|} \cdot \nabla L(w) = 0$. □

Solution to Exercise 4.12 (Some examples of loss landscape degeneracy).

(a) Since $a^{2k} \geq 0$ and $b^{2l} \geq 0$ for all $a, b \in \mathbb{R}$ (using the convention $0^0 = 1$), we have $L_{k,l}(a,b) \geq 0$ for $k, l \geq 1$, $L_{k,l}(a,b) \geq 1$ if $k = 0$ or $l = 0$, and $L_{k,l}(a,b) = 2$ if $k = l = 0$. At the origin, $L_{k,l}(0,0) = 0^{2k} + 0^{2l}$, which equals 0 if $k \geq 1$ and $l \geq 1$, equals 1 if exactly one of k, l is 0, and equals 2 if $k = l = 0$. In each case this matches the lower bound, so the origin is a global minimum.

The origin is the *unique* global minimum if and only if $k \geq 1$ and $l \geq 1$: then $L_{k,l}(a,b) = 0$ requires both $a^{2k} = 0$ and $b^{2l} = 0$, forcing $a = b = 0$. If $k = 0$, then $L_{0,l}(a,0) = 1$ for all a, so the entire a-axis achieves the minimum. Similarly if $l = 0$. If $k = l = 0$, then all parameters achieve the minimum.

(b) For $k = l = 1$: $L_{1,1}(a,b) = a^2 + b^2$. The Hessian at the origin is

$$H(0,0) = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix},$$

which is positive definite. The origin is a non-degenerate minimum.

(c) For $k > 1$ and $l \geq 1$: $\frac{\partial^2 L_{k,l}}{\partial a^2} = 2k(2k-1)\, a^{2k-2}$. At the origin, $a^{2k-2} = 0$ since $2k - 2 \geq 2$, so this entry vanishes. The Hessian at the origin is therefore

$$H(0,0) = \begin{pmatrix} 0 & 0 \\ 0 & 2l(2l-1) \cdot 0^{2l-2} \end{pmatrix},$$

which has a zero eigenvalue (from the first diagonal entry) regardless of the value of the second. The origin is a degenerate minimum.

(d) For $k = 0$ and $l \geq 1$: $L_{0,l}(a,b) = 1 + b^{2l}$. Since $L_{0,l}$ is independent of a, both $\frac{\partial L}{\partial a}$ and $\frac{\partial^2 L}{\partial a^2}$ vanish identically. The Hessian at the origin has a zero first diagonal entry, so it is singular. The origin is a degenerate minimum. □

Solution to Exercise 4.13 (Contrasting parametric versus loss degeneracy).

(a) The identity parameter–function map $\Phi(a,b) = (a,b)$ has Jacobian equal to the 2×2 identity matrix everywhere, so it is non-degenerate at every point. Exercise 4.12(b) shows that $L_{1,1}$ has a non-degenerate minimum: absence of parameter–function map degeneracy is consistent with absence of loss landscape degeneracy. Exercises 4.12(c) and 4.12(d) show that $L_{2,1}$, $L_{0,1}$, etc., have degenerate minima despite the parameter–function map being non-degenerate. This demonstrates that loss landscape degeneracy does not imply parameter–function map degeneracy.

(b) $L_\circ(a,b) = (a^2 + b^2 - 1)^2 \geq 0$, with equality if and only if $a^2 + b^2 = 1$. So every point on the unit circle is a global minimum. Consider $(1,0)$. The gradient is

$$\nabla L_\circ = \big(4a(a^2 + b^2 - 1),\ 4b(a^2 + b^2 - 1)\big),$$

which vanishes at $(1,0)$. The Hessian entries at $(1,0)$ are:

$$\frac{\partial^2 L_\circ}{\partial a^2} = 4(3a^2 + b^2 - 1) = 8, \qquad \frac{\partial^2 L_\circ}{\partial b^2} = 4(a^2 + 3b^2 - 1) = 0,$$
$$\frac{\partial^2 L_\circ}{\partial a\, \partial b} = 8ab = 0.$$

So $H(1,0) = \mathrm{diag}(8,0)$, which is singular. Therefore, $(1,0)$ is a degenerate minimum.

This provides another example of loss landscape degeneracy without parameter–function map degeneracy: the identity parameter–function map is non-degenerate, but the circle of global minima creates a flat tangential direction.

(c) (i) The map $\Phi(\alpha,\beta) = (\alpha^3, \beta^3)$ is the product of two copies of the cubic parametrisation from Exercise 4.2. In each component, the derivative $\frac{\mathrm{d}}{\mathrm{d}\alpha}(\alpha^3) = 3\alpha^2$ vanishes at $\alpha = 0$. Therefore both coordinate directions are degenerate at the origin, and hence every direction is degenerate at the origin (since the Jacobian $\mathrm{diag}(3\alpha^2, 3\beta^2)$ is the zero matrix there).

(ii) In the new parametrisation, the loss becomes $L_{k,l}(\alpha,\beta) = (\alpha^3)^{2k} + (\beta^3)^{2l} = \alpha^{6k} + \beta^{6l}$.

If $k \geq 1$ and $l \geq 1$: the exponents $6k$ and $6l$ are both at least 6, so $\frac{\partial^2 L}{\partial \alpha^2}\big|_0 = 6k(6k-1)\cdot 0^{6k-2} = 0$ (since $6k-2 \geq 4$), and similarly for β. The Hessian at the origin is the zero matrix, which is singular.

If $k = 0$: $L_{0,l}(\alpha,\beta) = 1 + \beta^{6l}$, which is independent of α, so the Hessian has a zero first diagonal entry. Similarly if $l = 0$. In all cases, the origin is a degenerate minimum.

(iii) Under the identity parameter–function map, $L_{1,1}(a,b) = a^2 + b^2$ had a non-degenerate minimum at the origin (Exercise 4.12(b)). Under the cubic parameter–function map, the same loss became $\alpha^6 + \beta^6$, which has a degenerate minimum at the origin. This shows that introducing degeneracy into the parameter–function map can convert a non-degenerate minimum into a degenerate one.

For other k, l, $L_{k,l}$ was already degenerate at the origin under the non-degenerate parameterisation, the cubic function only makes it *more* degenerate (in terms of the order to which the loss vanishes in the degenerate direction(s)).

(d) (i) The gradient is $\nabla L = (ab-1)(b,\, a)$. At $(0,0)$: $\nabla L = (0-1)(0,0) = (0,0)$. So $(0,0)$ is a critical point.

The Jacobian of $\Phi(a,b) = ab$ is (b,a), which at $(0,0)$ is $(0,0)$. Every direction $v = (v_1, v_2)$ gives $Jv = bv_1 + av_2 = 0$, so Φ is degenerate in every direction at $(0,0)$.

(ii) Computing second partial derivatives:

$$\frac{\partial^2 L}{\partial a^2} = b^2, \qquad \frac{\partial^2 L}{\partial b^2} = a^2, \qquad \frac{\partial^2 L}{\partial a\, \partial b} = 2ab - 1.$$

At $(0,0)$:

$$H(0,0) = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix},$$

which has determinant $-1 \neq 0$ (eigenvalues ± 1). The Hessian is nonsingular.

(iii) Despite full degeneracy of the parameter–function map at $(0,0)$, the Hessian is nonsingular (in fact indefinite, so $(0,0)$ is a saddle point, not a local minimum). This shows that parameter–function map degeneracy does not in general imply loss landscape degeneracy. □

Solution to Exercise 4.14 (Hessian degeneracy for a teacher–student loss)**.**

(a) Compute the derivative:

$$\begin{aligned}\nabla_w L(w) &= \nabla_w \frac{1}{2}\,\mathbb{E}_{X\sim q}\left[r_w(X)^2\right] \\ &= \frac{1}{2}\,\mathbb{E}_{X\sim q}\left[\nabla_w\, r_w(X)^2\right] \\ &= \mathbb{E}_{X\sim q}\left[r_w(X)\,\nabla_w r_w(X)\right] \\ &= \mathbb{E}_{X\sim q}\left[r_w(X)\,\nabla_w f_w(X)\right].\end{aligned}$$

Since $f_{w_0} = f_\star$, we have $r_{w_0} = 0$, and thus $\nabla_w L(w_0) = 0$.

(b) Starting from (a), compute the derivative again:

$$\begin{aligned}\nabla_w^2 L(w) &= \nabla_w \mathbb{E}_{X\sim q}\left[r_w(X)\,\nabla_w f_w(X)\right] \\ &= \mathbb{E}_{X\sim q}\left[(\nabla_w r_w(X))(\nabla_w f_w(X))^\top + r_w(X)\,(\nabla_w^2 f_w(X))\right] \\ &= \mathbb{E}_{X\sim q}\left[(\nabla_w f_w(X))(\nabla_w f_w(X))^\top + r_w(X)\,\nabla_w^2 f_w(X)\right].\end{aligned}$$

Since $r_{w_0} = 0$, we have $R(w_0) = 0$, so $H(w_0) = G(w_0)$.

(c) Let $v \in \mathbb{R}^d$. Then

$$\begin{aligned}v^\top G(w) v &= \mathbb{E}\left[v^\top(\nabla_w f_w(X))(\nabla_w f_w(X))^\top v\right] \\ &= \mathbb{E}\left[\left(v^\top \nabla_w f_w(X)\right)^2\right] \\ &= \mathbb{E}\left[\left(\|v\|\, D_v\Phi(w)(X)\right)^2\right] \\ &= \|v\|^2\,\mathbb{E}\left[\left(D_v\Phi(w)(X)\right)^2\right] \\ &\geq 0.\end{aligned}$$

Since $(D_v\Phi(w)(X))^2$ is non-negative, $G(w)$ and, in turn, $H(w_0) = G(w_0)$ are positive semi-definite.

(d) Let $v \in \mathbb{R}^d$ such that $D_v\Phi(w_0) = 0$. Then

$$\begin{aligned}H(w_0)v &= G(w_0)v \\ &= \mathbb{E}\left[(\nabla_w f_{w_0}(X))(\nabla_w f_{w_0}(X))^\top v\right] \\ &= \mathbb{E}\left[(\nabla_w f_{w_0}(X))\big(\|v\|\, D_v\Phi(w_0)(X)\big)\right] \\ &= \mathbb{E}\left[(\nabla_w f_{w_0}(X))\big(\|v\| \cdot 0\big)\right] \\ &= 0.\end{aligned}$$

(e) Assume $H(w_0)v = 0$. Then

$$0 = v^\top\big(H(w_0)v\big) = v^\top G(w_0) v = \|v\|^2\,\mathbb{E}_{X\sim q}\left[\left(D_v\Phi(w_0)(X)\right)^2\right].$$

Since $q > 0$ and the (directional) derivative is continuous in x by assumption, it follows that $D_v\Phi(w_0) = 0$, that is, Φ is degenerate at w_0 in direction v.

(f) Exercises (d) and (e) combined cover non-zero v. If $v = 0$, clearly $H(w_0)v = 0$, and our convention says $D_v\Phi(w) = 0$. Hence $\ker H(w_0) = \{v \in \mathbb{R}^d : D_v\Phi(w_0) = 0\}$. Moreover, $H(w_0)$ is positive semi-definite by (c), and therefore positive definite if and only if $v = 0$ is the only degenerate direction, that is, if and only if Φ is non-degenerate at w_0. □