Università Commerciale "Luigi Bocconi"
PhD School

PhD program in: Statistics and Computer Science

Cycle: XXXVI

Disciplinary Field (code): FIS/02

Methods of statistical physics for explaining neural networks performance

Advisor: Carlo Lucibello

Co-Advisor: Enrico Malatesta

PhD Thesis by

Elizaveta Demyanenko

ID number: 3078688

Year: 2025

Contents

Chapter 1

Thesis Roadmap

Neural networks are one of the most remarkable success stories of computer science of the current age. Despite their inarguably outstanding performance on tasks as different as image recognition, natural language processing, and generative modeling, our theoretical understanding of these methods remains considerably limited. In this thesis, we undertake a theoretical study of neural networks through the lens of statistical physics, and corroborate our findings with phenomena observed on real-life, state-of-the-art models. We focus specifically on understanding the impressive ability of neural networks to *generalize* on unseen data. When and why does this generalization occur, and how can it be predicted? We offer partial answers to these questions by analyzing the loss landscape of neural networks, drawing insights from the study of disordered systems.

This introductory chapter contains a general roadmap for the thesis. Each chapter is dedicated to one of the research projects I worked on during my PhD, and focuses on different aspects of our main driving question.

Mean dimension of neural networks. Chapter 2 contains a study (see [1] for a published version) that addresses multiple phenomena in machine learning using a complexity measure of the function realized by a neural network.

Specifically, we study how the generalization of a neural network evolves as we vary its size. It is known that the test error of these models typically exhibits a *double descent* as a

function of the number of parameters. The first descent in test error occurs as long as the network is unable to fully interpolate the training data. Once it achieves interpolation, the test error increases due to overfitting. However, with a further increase in model size towards the heavily overparametrized region, the test error undergoes a second descent.

We demonstrate that the locations of the transition between these different regimes can be accurately predicted by studying the phase transitions of a new sensitivity metric for the neural networks that we call *Boolean Mean Dimension* (BMD). Focusing on a teacher-student setting for the random feature model, we derive a theoretical analysis based on the replica method that yields an interpretable expression for the BMD, in the high dimensional regime where the number of data points, the number of features, and the input size grow to infinity. We also observe that the BMD curve carries information on other crucial properties of neural networks, such as their robustness to adversarial attacks and the strength of regularization used during training. The BMD is independent of the training data, and as such, it provides insight on the architectural inductive bias of the networks.

Flatness of the loss landscape. **Chapter** 3 delves into the relationship between the generalization capabilities of neural networks and the concept of *flatness*, which is evaluated through local entropy measures for different loss functions (see [2] for a published version). Our goal is to enhance the understanding of how properties of the loss landscape connect with generalization.

We show that the generalization properties of a loss minimizer can be inferred by examining the empirical risk landscape around it. Specifically, we show that generalization correlates with the existence of clusters of near-minimal solutions nearby, which we refer to as *wide flat minima*. Conversely, isolated minimizers tend to overfit the model. Following this idea, we employ variants of gradient descent methods to train neural networks that specifically target these wide flat minima. We demonstrate that these algorithms improve generalization across different architectures, datasets, and loss functions.

Our findings offer a concrete pathway for making design choices (architecture, loss function, training method) that improve generalization. More broadly, the new concepts and techniques introduced here promise to bridge the gap between theoretical understanding and practical application of neural networks.

Diffusion-based sampling. Having reduced the problem of understanding generalization to that of describing the local loss landscape, we conduct in **Chapter** 4 a more precise study of the latter in a simple model of shallow neural networks known as the *perceptron* problem [3]. We analyze the *Stochastic Localization* (SL) algorithm, which employs a denoising diffusion process to sample from a known but intractable distribution. In our setting, the score function is provided by an oracle using *Belief Propagation* or *Approximate Message Passing*.

We consider different variants of non-convex perceptron problems: the negative spherical perceptron, the binary perceptron, and the binary perceptron with modified loss functions. We show that stochastic localization successfully samples solutions for any satisfiable parameter regime of the negative spherical perceptron and fails throughout an entire regime of parameters for the binary perceptron, but also that the latter issue can be overcome by adapting the loss function to target flat minima. Additionally, we provide numerical evidence for the *uniformity* of sampled solutions in case of the negative spherical perceptron.

References

[1] Elizaveta Demyanenko, Christoph Feinauer, Enrico Maria Malatesta, and Luca Saglietti. The twin peaks of learning neural networks. *Machine Learning: Science and Technology*, 2024.

[2] Fabrizio Pittorino, Carlo Lucibello, Christoph Feinauer, Gabriele Perugini, Carlo Baldassi, Elizaveta Demyanenko, and Riccardo Zecchina. Entropic gradient descent al-

gorithms and wide flat minima. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124015, 2021.

[3] Elizaveta Demyanenko, Carlo Lucibello, Davide Straziota, and Carlo Baldassi. Investigation of Perceptron models through Stochastic Localization. *Manuscript*, 2024.

Chapter 2

Mean Dimension of Neural Networks

This chapter presents a study of the sensitivity of neural networks based on the *mean dimension* [1]. The mean dimension measures the average order of interactions between input variables in a function, making it a useful marker of the complexity of a function. It was initially studied within the ANOVA framework [2, 3]. This approach allows in theory computation of the mean dimension for an arbitrary queryable function; however, its main drawback is its potentially high computational complexity.

The problem becomes particularly challenging when estimating this measure for complex functions like deep neural networks, especially when the input follows a strongly structured or possibly unknown distribution. To address the issue, we introduce in this chapter the notion of *Boolean mean dimension* (BMD). As the name suggests, our approach is rooted in Boolean function analysis, which enables simple and efficient calculations using Fourier analysis. We focus on efficiently computing the BMD of neural networks, and demonstrate, both analytically and numerically, a correlation between this metric and several phenomena in neural networks, such as the double descent of the generalization error and robustness to adversarial initialization.

A detailed definition and explanation of the Boolean mean dimension can be found in

Section 2.2. Here, we provide basic intuition: a function of a vector input can be decomposed into a weighted sum of terms. These terms range from lower-order components, involving one or a few input variables, to higher-order interactions that may include many or all input variables. The mean dimension quantifies the average order of the terms that dominate in this expansion.

2.1 The double descent phenomenon

The bias-variance tradeoff has long been a cornerstone of classical machine learning theory, providing a structured framework for understanding model performance. In this traditional view, increasing model complexity lowers its bias, while simultaneously increasing its variance, thus leading to overfitting. The tradeoff highlights the balance between simplicity and flexibility, with the goal of finding an optimal middle ground where both bias and variance are minimized. For decades, this framework guided model selection and informed strategies to avoid overfitting. However, the recent discovery of the double descent phenomenon challenged this classical understanding, revealing that highly overparameterized models can generalize well.

Specifically, the double descent is demonstrated by the phenomenon when increasing the capacity of a model (e.g. measured by the number of parameters) the generalization error demonstrates a sudden peak around the interpolation point (where approximately zero training error is achieved), followed by a second decrease towards a low asymptotic value.

Several studies [4, 5, 6] have demonstrated the robustness of the double descent phenomenon across a wide range of architectures, datasets, and learning paradigms. In particular, an analytical investigation of double descent was rigorously carried out in the context of the random feature model, specifically for the square loss in [7] and for more

general loss functions using the replica method in [8][9].

Other variations of this phenomenon have been identified, such as epoch-wise double descent and sample non-monotonicity [10], as well as triple descent, which can arise with noisy labels and can be regularized by the non-linearity of the activation function [11].

Further in this chapter, we establish a connection between the traditional double descent of generalization error and the behavior of the mean dimension, a complexity metric that can be computed independently of task-specific data.

2.2 Computation of the mean dimension

In this section we provide with the precise definition of the MD (and BMD) and demonstrate an efficient way to estimate it using a Monte Carlo method.

2.2.1 Mathematical definition

To give a proper mathematical definition of the mean dimension, for a real-valued function $f(\boldsymbol{x})$ of n variables $f : \mathbb{R}^n \to \mathbb{R}$, it is convenient to introduce some notation that will be used in the rest of the chapter. We will denote the set of indexes $\{1, \dots n\}$ by $[n]$. We define $\boldsymbol{x}_u$ the set of input variables x_i, with $i \in u \subseteq [n]$ and by $\boldsymbol{x}_{\setminus u}$ the set of variables for which $i \notin u$. We will also assume that $\boldsymbol{x}$ is drawn from a distribution $p(\boldsymbol{x})$. The basic idea of the mean dimension is to derive a complexity measure for f from an expansion of the type

$$f(\boldsymbol{x}) = \sum_{u \subseteq [n]} f_u(\boldsymbol{x}_u) \tag{2.1}$$

where the "components" $f_u(\boldsymbol{x}_u)$ can be computed from the following recursion relation

$$f_u(\boldsymbol{x}_u) \equiv \int f(\boldsymbol{x})\, p(\boldsymbol{x}_{\setminus u}|\boldsymbol{x}_u)\, d\boldsymbol{x}_{\setminus u} - \sum_{v \subset u} f_v(\boldsymbol{x}_v) \tag{2.2}$$

with the initial condition $f_\emptyset = \int f(\boldsymbol{x})\, p(\boldsymbol{x})\, d\boldsymbol{x} \equiv \mathbb{E}[f]$. It can be shown that coefficients of the expansion have zero average if u is non empty

$$\int f_u(\boldsymbol{x}_u) p_u(\boldsymbol{x}_u)\, d\boldsymbol{x}_u = 0 \qquad u \neq \emptyset \tag{2.3}$$

where we have denoted by $p_u(\boldsymbol{x}_u)$ the marginal probability distribution over the set u. Moreover, they satisfy orthogonality relations, namely

$$\int f_u(\boldsymbol{x}_u) f_v(\boldsymbol{x}_v) p_{u\cup v}(\boldsymbol{x}_{u\cup v})\, d\boldsymbol{x}_{u\cup v} = 0\,, \qquad \text{if } u \neq v\,. \tag{2.4}$$

Using those relations we can write the variance of the function as a decomposition of $2^n - 1$ terms

$$\sigma^2 = \mathbb{E}[f^2] - \mathbb{E}[f]^2 = \sum_{u\subseteq[n]\setminus\emptyset} \sigma_u^2 \tag{2.5}$$

where

$$\sigma_u^2 \equiv \int f_u^2(\boldsymbol{x}_u)\, p_u(\boldsymbol{x}_u)\, d\boldsymbol{x}_u\,. \tag{2.6}$$

The mean dimension M_f is then defined as

$$M_f = \sum_{u\subseteq[n]} |u| \frac{\sigma_u^2}{\sigma^2}, \tag{2.7}$$

i.e. a weighted sum over possible interactions, with each subset of inputs contributing based on how much they influence the variance.

2.2.2 Pseudo-boolean functions and Fourier coefficients

We now derive an explicit expression for the mean dimension of n-dimensional pseudo-Boolean functions taking values on the real domain, $f : \{-1, 1\}^n \to \mathbb{R}$ under the assumption of input features that are i.i.d. from $\{-1, 1\}$.

Denoting by $\boldsymbol{s} \in \{-1, 1\}^n$ the n-dimensional binary input of f, such a function can be uniquely written as a *Fourier expansion* [12] in terms of a finite set of *Fourier coefficients*

$\hat{f}_u$, $u \subseteq [n]$ as

$$f(\boldsymbol{s}) = C + \sum_i h_i s_i + \sum_{i<j} J_{ij} s_i s_j + \sum_{i<j<k} K_{ijk} s_i s_j s_k + \ldots = \sum_{u \subseteq [n]} \hat{f}_u \, \chi_u(\boldsymbol{s}_u) \tag{2.8}$$

where

$$\chi_u(\boldsymbol{s}_u) = \prod_{i \in u} s_i \tag{2.9}$$

represent the Fourier basis of the decomposition that are orthonormal $\langle \chi_u(\boldsymbol{s}) \chi_v(\boldsymbol{s}) \rangle = \delta_{u,v}$ with respect to the uniform distribution over $\{-1, 1\}^n$, where we use the notation

$$\langle \bullet \rangle \equiv \frac{1}{2^n} \sum_{\boldsymbol{s} \in \{-1,1\}^n} \bullet \, . \tag{2.10}$$

The Fourier coefficients $\hat{f}_u$ can give information about the moments of the function f with respect to the uniform distribution (2.10) over $\boldsymbol{s}$; for example the first moment is

$$\langle f(\boldsymbol{s}) \rangle = \hat{f}_\emptyset \tag{2.11}$$

whereas the variance can be obtained as

$$\sigma^2 = \langle f^2(\boldsymbol{s}) \rangle - \langle f(\boldsymbol{s}) \rangle^2 = \sum_{u \subseteq [n] \setminus \emptyset} \hat{f}_u^2 \, . \tag{2.12}$$

We can quantify the contribution c_k of interaction of order k to the variance of $f(\boldsymbol{s})$ as the ratio

$$c_k = \frac{\sum_{u \subseteq [n] \setminus \emptyset : |u| = k} \hat{f}_u^2}{\sigma^2} \, . \tag{2.13}$$

Notice that $\sum_k c_k = 1$, so that c_k can be interpreted as a (discrete) probability measure over interactions. The mean dimension of f can then be written as the mean interaction degree when weighted according to it contribution to the variance, i.e. as a weighted sum

of feature influences divided by the total variance of the function, so

$$M_f \equiv \sum_{k=1}^{n} k c_k = \frac{\sum_{u \subseteq [n]} |u| \, \hat{f}_u^2}{\sigma^2} \tag{2.14}$$

This expression is equivalent to Eq. (2.7) for pseudo-Boolean functions under the assumptions that all features are i.i.d from $\{-1, 1\}$. The expression connects the notion of simplicity in terms of variance contributions to the same notion in terms of explicit expansion coefficients. Intuitively, a large mean dimension is indicating that the function fluctuates due to a large contribution of high-degree interactions.

2.2.3 Estimating the mean dimension through Monte Carlo

The expression of the mean dimension in (2.7) involves a sum over all the set of subsets of n variables, and its numerical evaluation through a brute-force approach would be intractable in high dimension. However, it can be shown that a more efficient evaluation scheme of equation (2.7), can be achieved through a Monte Carlo approach [13]. First, the MD can be rewritten as a sum over the n input components:

$$M_f = \frac{\sum_{i=1}^{n} \tau_i^2}{\sigma^2} \tag{2.15}$$

where the *influence* of the i-th input component τ_i is defined as:

$$\tau_i^2 = \frac{1}{2} \int d\boldsymbol{x} \, dx_i' \, p(\boldsymbol{x}) \, p(x_i' | \boldsymbol{x}_{\setminus i}) (f(\boldsymbol{x}) - f(\boldsymbol{x}^{\oplus i}))^2 \, . \tag{2.16}$$

and where we have denoted by $\boldsymbol{x}^{\oplus i}$ a vector $\boldsymbol{x}$ with a resampled i_{th} coordinate. We show an original proof of this identity in the next section of the chapter.

2.2.4 Proof of equation (2.15)

In order to prove the relation of the mean dimension of equation (2.15), we anticipate two Lemmas that are useful for the proof.

Lemma 2.2.4.1. *For any $u \subseteq [n]$, the ANOVA coefficients satisfy the following relation:*

$$f_u(x) = \sum_{v \subseteq u} (-1)^{|u|-|v|} \int f(\boldsymbol{x}) p(\boldsymbol{x}_{\setminus v} | \boldsymbol{x}_v) d\boldsymbol{x}_{\setminus v} = \sum_{v \subseteq u} (-1)^{|u|-|v|} \mathbb{E}[f(\boldsymbol{x}) | \boldsymbol{x}_v] \tag{2.17}$$

Proof. This can be seen by induction. The base of induction for the empty set and a singleton $u = \{i\}, \forall i \in n$ are respectively

$$f_\emptyset = \mathbb{E}[f(\boldsymbol{x})] \tag{2.18a}$$

$$f_{\{i\}} = \mathbb{E}[f(\boldsymbol{x})|x_i] - \mathbb{E}[f(\boldsymbol{x})] = \mathbb{E}[f(\boldsymbol{x})|x_i] - f_\emptyset \tag{2.18b}$$

having used the ANOVA recursion in equation (2.2). Assuming equation (2.17) holds for the sets up to the size k we can show the induction step up to the set size $k+1$. Given a set $S \subseteq [n]$, $i \notin S$ and $|S| = k$ denoting $u = S \cup \{i\}$ we have:

$$\begin{aligned} f_u(\boldsymbol{x}_u) &= \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{x}_u] - \sum_{v \subset u} f_v(\boldsymbol{x}_v) = \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{x}_u] - \sum_{v \subset u} \sum_{t \subseteq v} (-1)^{|v|-|t|} \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{x}_t] \\ &= \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{x}_u] - \sum_{t \subseteq S} (-1)^{-|t|} \sum_{t \subseteq v \subset u} (-1)^{|v|} \, \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{x}_t] \, . \end{aligned} \tag{2.19}$$

The summation over sets $t \subseteq v \subset u$ can be performed

$$\sum_{t \subseteq v \subset u} (-1)^{|v|} = \sum_{k=|t|}^{|u|-1} (-1)^k \binom{|u|-|t|}{k-|t|} \tag{2.20}$$

$$= (-1)^{-|t|} \sum_{k=0}^{|u|-|t|-1} (-1)^k \binom{|u|-|t|}{k} = (-1)^{|u|+1} \, . \tag{2.21}$$

Inserting this back into (2.19) we have finally

$$f_u(\boldsymbol{x}_u) = \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{x}_u] + \sum_{t \subseteq S} (-1)^{|u|-|t|} \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{x}_t] = \sum_{v \subseteq u} (-1)^{|u|-|v|} \mathbb{E}[f(\boldsymbol{x})|\boldsymbol{x}_v] \tag{2.22}$$

□

Now given the previous Lemma 2.2.4.1 we can show:

Lemma 2.2.4.2. *We can write*

$$f(\boldsymbol{x}) = \sum_{u \ni i} f_u(\boldsymbol{x}_u) + \mathbb{E}\left[f(\boldsymbol{x})|\boldsymbol{x}_{\setminus i}\right] \tag{2.23}$$

or equivalently

$$\sum_{u \not\ni i} f_u(\boldsymbol{x}_u) = \mathbb{E}\left[f(\boldsymbol{x})|\boldsymbol{x}_{\setminus i}\right] \tag{2.24}$$

Proof. To show that we consider:

$$\sum_{u \ni i} f_u(\boldsymbol{x}_u) = \sum_{u \ni i} \sum_{v \subseteq u} (-1)^{|u|-|v|} \mathbb{E}\left[f(\boldsymbol{x})|\boldsymbol{x}_v\right] = \sum_{v} \left[\sum_{u \supseteq v, u \ni i} (-1)^{|u|}\right] (-1)^{-|v|} \mathbb{E}\left[f(\boldsymbol{x})|\boldsymbol{x}_v\right] \tag{2.25}$$

The term in the squared brackets can be computed. We need to distinguish two cases, i.e. $i \in v$, and $i \notin v$. In the first case we get

$$\sum_{u \supseteq v, u \ni i} (-1)^{|u|} = \sum_{k=|v|}^{n} (-1)^k \binom{n-|v|}{k-|v|} = \sum_{k=0}^{n-|v|} (-1)^{k+|v|} \binom{n-|v|}{k} = (-1)^{|v|} \delta_{n,|v|} \tag{2.26}$$

where $\delta_{i,j}$ is the Kronecker delta function. If $i \notin v$, instead

$$\sum_{u \supseteq v, u \ni i} (-1)^{|u|} = \sum_{k=|v|+1}^{n} (-1)^k \binom{n-|v|-1}{k-|v|-1} = (-1)^{|v|+1} \delta_{n,|v|+1} \tag{2.27}$$

i.e. we get a non-zero result only if $v = [n]$ or $v = [n] \setminus \{i\}$. Therefore we get

$$\sum_{u \ni i} f_u(\boldsymbol{x}_u) = f(\boldsymbol{x}) - \mathbb{E}\left[f(\boldsymbol{x})|\boldsymbol{x}_{\setminus i}\right]$$

□

In the following we will denote by $\boldsymbol{x}^{\oplus i}$ a vector $\boldsymbol{x}$ with a resampled i_{th} coordinate. We are now ready to prove the following theorem:

Theorem 2.2.4.3. *The mean dimension can be written as*

$$M_f = \sum_{u \subseteq [n]} |u| \frac{\sigma_u^2}{\sigma^2} = \frac{\sum_{i=1}^n \tau_i^2}{\sigma^2} \tag{2.28}$$

where

$$\tau_i^2 = \frac{1}{2} \int d\boldsymbol{x}\, dx_i'\, p(\boldsymbol{x})\, p(x_i'|\boldsymbol{x}_{\setminus i})(f(\boldsymbol{x}) - f(\boldsymbol{x}^{\oplus i}))^2 \equiv \frac{1}{2}\mathbb{E}\left(f(\boldsymbol{x}) - f(\boldsymbol{x}^{\oplus i})\right)^2 \tag{2.29}$$

Proof. We can write the numerator of the mean dimension as

$$\sum_{u \subseteq [n]} |u| \sigma_u^2 = \sum_{k=1}^n k \sum_{\substack{u \subseteq [n] \\ |u|=k}} \sigma_u^2 = \sum_{i=1}^n \sum_{u \subseteq [n] : i \in u} \sigma_u^2 \,. \tag{2.30}$$

In the first equality we divided the summation over the sets into a double summation over the size k of the set and a summation over the sets of fixed size k. In the second equality we have used the fact that the summation over k can be interpreted as a summation over the indices i of the variable $\boldsymbol{x}$; the inner sum can be therefore written as a summation over the sets (of any possible size) that contain the variable i itself. Reminding that

$$\sigma_u^2 \equiv \int f_u^2(\boldsymbol{x}_u)\, p_u(\boldsymbol{x}_u)\, d\boldsymbol{x}_u = \mathbb{E}[f_u^2(\boldsymbol{x}_u)] \,, \tag{2.31}$$

and using Lemma 2.2.4.2 and orthogonality of the coefficients of the ANOVA expansion, we have

$$\begin{aligned}
\tau_i^2 \equiv \frac{1}{2}\mathbb{E}\left(f(\boldsymbol{x}) - f(\boldsymbol{x}^{\oplus i})\right)^2 &= \frac{1}{2}\mathbb{E}\left(f(\boldsymbol{x}) - f(\boldsymbol{x}^{\oplus i})\right)\left(\sum_{u \ni i} f_u(\boldsymbol{x}_u) - \sum_{u \ni i} f_u(\boldsymbol{x}_u^{\oplus i})\right) \\
&= \mathbb{E}\left[f(\boldsymbol{x}) \sum_{u \ni i} f_u(\boldsymbol{x}_u)\right] - \mathbb{E}\left[f(\boldsymbol{x}) \sum_{i \ni i} f_u(\boldsymbol{x}_u^{\oplus i})\right] \\
&= \mathbb{E}\left[\sum_{u \ni i} f_u^2(\boldsymbol{x}_u)\right] - \mathbb{E}\left[f(\boldsymbol{x})\left(f(\boldsymbol{x}^{\oplus i}) - \mathbb{E}\left[f(\boldsymbol{x}^{\oplus i}|\boldsymbol{x}_{\setminus i})\right]\right)\right] \\
&= \mathbb{E}\sum_{u \ni i} f_u^2(\boldsymbol{x}_u) - 0 = \sum_{u \ni i} \sigma_u^2 \,.
\end{aligned}$$

□

We emphasize that in the general case, the underlying input distribution of the training dataset is not known and estimating the MD on this distribution becomes unfeasible. In the present chapter, we propose employing the estimation procedure presented in the previous section, based on binary sequences, as an easily computable proxy of the sensitivity of the neural network function. In order to distinguish this proxy from the mean dimension over the dataset distribution, we call the resulting quantity the Boolean Mean Dimension (BMD). We show in the further sections that the BMD can in some cases be computed analytically, and that it is qualitatively related to the generalization phenomenology in neural networks.

2.2.5 Mean dimension in the boolean case

Expression (2.16) can be specialized to the case of binary i.i.d. inputs, where one can identify the influence functions τ_i^2 with the discrete derivatives:

$$\tau_i^2 = \left\langle \left(\mathcal{D}_i f(\boldsymbol{s})\right)^2 \right\rangle \tag{2.32}$$

where $\mathcal{D}_i f(\boldsymbol{s})$ denotes the i_{th} (discrete) derivative of $f(\boldsymbol{s})$, i.e.

$$\mathcal{D}_i f(\boldsymbol{s}) \equiv \frac{f(s_1, \ldots, s_i = 1, \ldots, s_n) - f(s_1, \ldots, s_i = -1, \ldots, s_n)}{2} \tag{2.33}$$

and measures the average sensitivity of the function to a flip of the i_{th} variable. The sum of the influences $\sum_i \left\langle \left(\mathcal{D}_i f(\boldsymbol{s})\right)^2 \right\rangle$ is known in the field of the analysis of pseudo-Boolean functions as *total influence* of f [12]. In terms of the Fourier expansion, we have

$$\mathcal{D}_i f(\boldsymbol{s}) = \sum_{u \subseteq [n]: i \in u} \hat{f}_u \, \chi_{u \setminus i}(\boldsymbol{s}_{u \setminus i}) \tag{2.34}$$

Therefore computing the mean dimension for pseudo-Boolean functions boils down to querying the function f on uniformly sampled binary sequences of length $n - 1$.

Algorithm 1 BMD estimation

Input: N (data dimension), **nb** (number of batches), **bs** (batch size), $f : \mathbb{R}^N \to \mathbb{R}^C$, C (number of classes)

Initialize $\mathbf{M} \coloneqq \bar{0}_C$, $\mathbf{Msq} \coloneqq \bar{0}_C$, $\mathbf{C} \coloneqq \bar{0}_{N\times C}$, $\mathbf{MD} \coloneqq \bar{0}_C$
for i in $1, \dots, N$ **do**
  for b in $1, \dots,$ **nb do**
    $\mathbf{X}_b = \text{sample}(\mathbf{bs}, N)$ *// samples uniformly binary sequences of size* $(\mathbf{bs} \times N)$
    for c in $1, \dots, C$ **do**
      $\mathbf{M}[c]$ += $\sum_j f(\mathbf{X}_b[j, c])/(\mathbf{nb} \times \mathbf{bs})$ *// computes the 1st moment*
      $\mathbf{Msq}[c]$ += $\sum_j f(\mathbf{X}_b[j, c])^2/(\mathbf{nb} \times \mathbf{bs})$ *// computes the 2nd moment*
      $\mathbf{C}[i][c]$ += $\sum_j \left\langle (\mathcal{D}_i f(\mathbf{X}_b[j, c]))^2 \right\rangle /(\mathbf{nb} \times \mathbf{bs})$
    end for
  end for
end for
for c in $1, \dots, C$ **do**
  $\mathbf{M}[c] \coloneqq \mathbf{M}[c]/N$
  $\mathbf{Msq}[c] \coloneqq \mathbf{Msq}[c]/N$
  $\mathbf{MD}[c] \coloneqq \sum_j(\mathbf{C}[j][c])/(\mathbf{Msq}[c] - \mathbf{M}[c]^2)$
end for
return M

A simple way of estimating the numerator of the mean dimension implies therefore estimating the quantities $\left\langle (\mathcal{D}_i f(\boldsymbol{s}))^2 \right\rangle$ separately for every i and then summing them. In Algorithm 1 we present the algorithm for estimating the mean dimension.

2.3 Analytical results

We now derive an analytic expression for the mean dimension in the special case of the random feature model [14, 15, 16, 17], focusing on the same high dimensional regime where the double descent phenomenon can be detected. In the next sections, we will define the model, the learning task and the high dimensional limit precisely, and we will provide the analytical derivation of the expression for the Boolean Mean Dimension.

2.3.1 Model definition and learning task

The random feature model (RFM) is a two-layer neural network with random and fixed first-layer weights (also called features) and trainable second-layer weights. Given a D-dimensional input, $\boldsymbol{x} \in \mathbb{R}^D$, and denoting by $F \in \mathbb{R}^{D\times N}$ the $D \times N$ frozen feature matrix, the pre-activation of the RFM is given by:

$$\hat{y}(\boldsymbol{w}; \boldsymbol{x}) = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} w_i \, \sigma \left(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} F_{ki} \, x_k \right) \tag{2.35}$$

where $\boldsymbol{w}$ is an N-dimensional weight vector and σ is a (usually non-linear) function. The parameter N indicates the number of features in the RFM and can be varied to change the degree of over-parametrization of the model. As in [17], we will hereafter focus on the case of i.i.d. standard normal distributed feature components $F_{ki} \sim \mathcal{N}(0, 1)$, although the formalism allows for a simple extension to a generic fixed feature map, under a simple weak correlation requirement (see [8, 16] for additional details).

We consider a classification task defined by a training dataset of size P, denoted as $\mathcal{D} = \{\boldsymbol{x}^\mu, y^\mu\}_{\mu=1}^{P}$. The inputs are assumed to be i.i.d. with first and second moments fixed respectively to $\mathbb{E}x_i = 0$ and $\mathbb{E}x_i^2 = 1$. Note that, for example, both binary input components $x_i \in \{-1, 1\}$ and Gaussian components $x_i \sim \mathcal{N}(0, 1)$ satisfy the above assumption. The binary labels $y^\mu \in \{-1, 1\}$ are assumed to be produced by a "teacher" linear model $\boldsymbol{w}^T \in \mathbb{R}^D$, with normalized weights on the D-sphere $\|\boldsymbol{w}^T\|_2^2 = D$, according to:

$$y^\mu = \text{sign} \left(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} w_k^T x_k^\mu \right) , \qquad \mu \in [P] \, . \tag{2.36}$$

The learning task is then framed as an optimization problem with generic loss function ℓ and ridge regularization

$$\boldsymbol{w}_\star = \underset{\boldsymbol{w}\in\mathbb{R}^N}{\arg\min} \left[\sum_{\mu=1}^{P} \ell(y^\mu, \, \hat{y}^\mu(\boldsymbol{w}; \boldsymbol{x}^\mu)) + \frac{\lambda}{2} \|\boldsymbol{w}\|_2^2 \right] , \tag{2.37}$$

where λ is a positive external parameter controlling the regularization strength. In the following we will consider the two most common convex loss functions, namely the mean squared error (MSE) and the cross-entropy (CE) losses, defined as

$$\ell_{mse}(y, \hat{y}) = \frac{1}{2}\left(y^\mu - \hat{y}^\mu\right)^2 \tag{2.38a}$$

$$\ell_{ce}(y, \hat{y}) = \log\left(1 + e^{-y\hat{y}}\right) . \tag{2.38b}$$

We analyze the learning problem in the high-dimensional limit where the number of features, input components and training-set size diverge N , D , $P \to \infty$ at constant rates $\alpha \equiv P/N = \mathcal{O}(1)$ and $\alpha_D \equiv D/N = \mathcal{O}(1)$. In this limit, strong concentration properties allow for a deterministic characterization of the above-defined learning problem in terms of a finite set of scalar quantities called order parameters. In the next sections, and in detail in the appendices, we will provide the derivation of this reduced description.

2.3.2 Rephrasing the problem in terms of the Boltzmann measure

The learning task in (2.37) can be characterized within a statistical physics framework. One can introduce a probability measure over the weights $\boldsymbol{w}$ in terms of the Boltzmann distribution

$$\boldsymbol{w} \sim p_\beta(\boldsymbol{w}; \mathcal{D}) = \frac{e^{-\beta \sum_{\mu=1}^{P} \ell(y^\mu, \hat{y}^\mu(\boldsymbol{w}; \boldsymbol{x}^\mu)) - \frac{\beta\lambda}{2}\sum_{i=1}^{N} w_i^2}}{Z_\beta} \tag{2.39}$$

where β is the inverse temperature, the loss function in (2.37) plays the role of an energy, and the partition function Z_β is a normalization factor that reads

$$Z_\beta = \int d\boldsymbol{w}\, e^{-\beta \sum_{\mu=1}^{P} \ell(y^\mu, \hat{y}^\mu(\boldsymbol{w}; \boldsymbol{x}^\mu)) - \frac{\beta\lambda}{2}\sum_{i=1}^{N} w_i^2} . \tag{2.40}$$

The distribution $p_\beta(\boldsymbol{w}; \mathcal{D})$ can be interpreted in a Bayesian setting as the posterior distribution over the weights $\boldsymbol{w}$ given a dataset $\mathcal{D}$, and (2.39) corresponds to Bayes theorem, where the term $e^{-\beta \sum_\mu \ell(y^\mu, \hat{y}^\mu(\boldsymbol{w}; \boldsymbol{x}^\mu))}$, corresponds to the likelihood and $e^{-\frac{\beta\lambda}{2}\|\boldsymbol{w}\|_2^2}$ is the prior

distribution over the weights.

In the zero-temperature limit, when $\beta \to \infty$, the probability measure $p_\beta(\boldsymbol{w}; \mathcal{D})$ concentrates on the solutions to the optimization problem in (2.37). To characterize the typical (i.e. the most probable) properties of these solutions, one needs to perform an average over the possible realizations of the training set $\mathcal{D}$ and of the features F, computing the free-energy of the system

$$f = -\lim_{\beta\to\infty}\lim_{N\to\infty}\frac{1}{\beta N}\mathbb{E}_{\mathcal{D},F}\ln Z_\beta \ . \tag{2.41}$$

The computation of this "quenched" average can be achieved via the replica method [9] from spin-glass theory, which reduces the characterization of the solutions of (2.37) to the determination of a finite set of scalar quantities called order parameters [18, 19].

In the next section we sketch the replica calculation for the free energy, first presented in [8], in the simplifying case of an odd non-linear activation σ.

2.4 Replica computation of the BMD in the random feature model

2.4.1 Free entropy

We review here the replica calculations of the free entropy of the model, defined as

$$\phi_\beta = \lim_{N\to\infty}\frac{1}{N}\mathbb{E}_{\mathcal{D},F}\ln Z_\beta \, . \tag{2.42}$$

This in turn will give the information necessary to compute the BMD. The term Z_β, that is also called a partition function, can be estimated using the replica method, described in the next subsection.

Replica method. The replica method is a non-rigorous but powerful technique used in statistical physics to study disordered systems, like random neural networks. It helps compute averages over systems with randomness, particularly when directly calculating the logarithm of a partition function is intractable. The method works by replicating the system n times, calculating Z^n, and then analytically deriving the result for $n \to 0$ using the identity:

$$\ln Z = \frac{Z^n - 1}{n}$$

The replica method has been employed for over 40 years in theoretical physics and has emerged as an essential tool in probability theory, particularly following M. Talagrand's rigorous contributions nearly 20 years ago. In computer science, it has been utilized to investigate phase transitions in random constraint satisfaction problems. Since the 1990s, it has also been applied to analyze the asymptotic properties of neural networks (including those with one hidden layer) that store random patterns.

While the replica method often leads to correct results and has been widely validated by comparison with numerical simulations, it is technically non-rigorous because the analytic continuation step is not always mathematically justified. However, its effectiveness has made it a central tool in many fields of theoretical physics and complex systems analysis.

Thus the average over the dataset in (2.42) can be performed as follows

$$\mathbb{E}_{\mathcal{D},F} \ln Z_\beta = \lim_{n\to 0} \frac{1}{n} \ln \left(\mathbb{E}_{\mathcal{D},F} Z_\beta^n \right) . \tag{2.43}$$

In the following we will consider n as an integer and we will denote by a, b as replica indices running from 1 to n.

Gaussian equivalence theorem. In order to compute the average over the input patterns of Z_β^n, we will apply a central limit theorem [20], valid in the thermodynamic

limit where N, D, P go to infinity with fixed $\alpha \equiv \frac{P}{N}$ and $\alpha_D \equiv \frac{D}{N}$. In the statistical physics literature this central limit is often called Gaussian equivalence theorem (GET) [15, 16]. It can indeed be shown that the model is *equivalent* to a Gaussian covariate model [7] and the following identification can be made

$$\sigma\left(\frac{1}{\sqrt{D}}\sum_{k=1}^{D} F_{ki}\, x_k\right) = \kappa_0 + \frac{\kappa_1}{\sqrt{D}}\sum_{k=1}^{D} F_{ki}\, x_k + \kappa_\star \eta_i \tag{2.44}$$

where η_i is Gaussian noise with zero mean and unit variance and we have defined the following coefficients

$$\kappa_0 = \int Dz\, \sigma(z) \tag{2.45a}$$

$$\kappa_1 = \int Dz\, z\, \sigma(z) = \int Dz\, \sigma'(z) \tag{2.45b}$$

$$\kappa_2 = \int Dz\, \sigma^2(z) \tag{2.45c}$$

$$\kappa_\star^2 = \kappa_2 - \kappa_1^2 - \kappa_0^2 \tag{2.45d}$$

with $Dz \equiv \frac{e^{-z^2/2}}{\sqrt{2\pi}} dz$. In the following we will considered for simplicity $\sigma(\cdot)$ to be an odd activation function, so that in this case $\kappa_0 = 0$.

Average over the dataset Using GET, we therefore arrive to

$$\begin{aligned}
\mathbb{E}_{\mathcal{D}} Z_\beta^n &= \mathbb{E}_{\boldsymbol{w}^T} \int \prod_a d\boldsymbol{w}^a \int \prod_\mu \frac{du^\mu d\hat{u}^\mu}{2\pi} \prod_{\mu a} \frac{d\lambda_a^\mu d\hat{\lambda}_a^\mu}{2\pi}\, e^{-\beta \sum_{\mu a} \ell\left(\mathrm{sign}(u^\mu)\lambda_a^\mu\right) - \frac{\beta\lambda}{2}\sum_{ia}(w_i^a)^2} \\
&\quad \times \prod_\mu e^{iu^\mu \hat{u}^\mu + i\sum_a \lambda_a^\mu \hat{\lambda}_a^\mu - \frac{(\hat{u}^\mu)^2}{2} - \frac{1}{2}\sum_{ab} Q_{ab}\hat{\lambda}_a^\mu \hat{\lambda}_b^\mu - \sum_a M_a \hat{u}^\mu \hat{\lambda}_a^\mu}\,. \\
&= \int \prod_a d\boldsymbol{w}^a \int \prod_\mu du^\mu \prod_{\mu a} d\lambda_a^\mu\, e^{-\beta \sum_{\mu a} \ell\left(\mathrm{sign}(u^\mu)\lambda_a^\mu\right) - \frac{\beta\lambda}{2}\sum_{ia}(w_i^a)^2} \prod_\mu \mathcal{N}(u^\mu, \lambda_a^\mu; \mathbf{0}, \Sigma_{ab})
\end{aligned} \tag{2.46}$$

where the correlation matrix of the $n+1$-dimensional multivariate Gaussian $\mathcal{N}$ is

$$\Sigma_{ab} \equiv \begin{pmatrix} \rho & M_a \\ M_a & Q_{ab}\,. \end{pmatrix} \tag{2.47}$$

Here $\rho = \frac{1}{D}\sum_{k=1}^{D}(w_k^T)^2 = 1$, since the teacher has fixed norm. In the previous equation we have also denoted by Q_{ab} and M_a the following quantities

$$M_a = \kappa_1 \frac{1}{D}\sum_{k=1}^{D} s_k^a w_k^T \equiv \kappa_1 r_a \tag{2.48a}$$

$$Q_{ab} = \kappa_\star^2 \frac{1}{N}\sum_{i=1}^{N} w_i^a w_i^b + \kappa_1^2 \frac{1}{D}\sum_{k=1}^{D} s_k^a s_k^b \equiv \kappa_\star^2 q_{ab} + \kappa_1^2 p_{ab} \tag{2.48b}$$

where we have defined the projected student weights in the space of the teacher s_k^a as

$$s_k^a \equiv \frac{1}{\sqrt{N}}\sum_{i=1}^{N} F_{ki} w_i^a\,. \tag{2.49}$$

Average over Gaussian Features We can enforce the definition of the projected weights in (2.49) using delta functions and their integral representation. It then becomes easy to perform the average over random Gaussian features. We get a term of the following form

$$\int \prod_{ka} \frac{ds_k^a d\hat{s}_k^a}{2\pi}\, e^{i\sum_{ka} s_k^a \hat{s}_k^a} \prod_{ki} \mathbb{E}_{F_{ki}} \left[e^{-i\frac{F_{ki}}{\sqrt{N}}\sum_a \hat{s}_k^a w_i^a} \right] = \int \prod_{ka} \frac{ds_k^a d\hat{s}_k^a}{2\pi}\, e^{i\sum_{ka} s_k^a \hat{s}_k^a - \frac{1}{2}\sum_{ab,k} \hat{s}_k^a \hat{s}_k^b \left(\frac{1}{N}\sum_i w_i^a w_i^b\right)}\,, \tag{2.50}$$

which only depends on the q_{ab} defined in (2.48b).

Saddle point method We can now impose the definitions of the order parameters

$$q_{ab} \equiv \frac{1}{N}\sum_i w_i^a w_i^b\,, \qquad p_{ab} \equiv \frac{1}{D}\sum_k s_k^a s_k^b\,, \qquad r_a \equiv \frac{1}{D}\sum_k s_k^a w_k^T\,. \tag{2.51}$$

Denoting by $\langle\cdot\rangle_{\mathcal{D},F}$ the average over both patterns and random features, the final result reads

$$\mathbb{E}_{\mathcal{D},F} Z_\beta^n = \int \prod_{a\le b} \frac{dq_{ab}d\hat{q}_{ab}}{2\pi} \prod_{a\le b} \frac{dp_{ab}d\hat{p}_{ab}}{2\pi} \prod_a \frac{dr_a d\hat{r}_a}{2\pi}\, e^{N\phi_\beta^{(n)}} \tag{2.52}$$

where

$$\phi_\beta^{(n)} = -\frac{1}{2}\sum_{ab} q_{ab}\hat{q}_{ab} - \frac{\alpha_D}{2}\sum_{ab} p_{ab}\hat{p}_{ab} - \alpha_D \sum_a r_a\hat{r}_a + G_{SS} + \alpha_D G_{SE} + \alpha G_E \tag{2.53a}$$

$$G_{SS} = \ln \int \prod_a dw_a\, e^{\frac{1}{2}\sum_{ab}\hat{q}_{ab}w_a w_b - \frac{\beta\lambda}{2}\sum_a w_a^2} \tag{2.53b}$$

$$G_{SE} = \ln \int \prod_a \frac{ds_a d\hat{s}_a}{2\pi}\, e^{i\sum_a s_a\hat{s}_a + \sum_a \hat{r}_a s_a + \frac{1}{2}\sum_{ab}\hat{p}_{ab}s_a s_b - \frac{1}{2}\sum_{ab} q_{ab}\hat{s}_a\hat{s}_b} \tag{2.53c}$$

$$G_E = \ln \int \prod_a \frac{d\lambda_a d\hat{\lambda}_a}{2\pi}\frac{dud\hat{u}}{2\pi}\, e^{iu\hat{u} + i\sum_a \lambda_a\hat{\lambda}_a - \beta\sum_a \ell\left(\mathrm{sign}(u)\lambda_a\right) - \frac{\hat{u}^2}{2} - \frac{1}{2}\sum_{ab} Q_{ab}\hat{\lambda}_a\hat{\lambda}_b - \hat{u}\sum_a M_a\hat{\lambda}_a} \tag{2.53d}$$

and M_a, Q_{ab} are defined in terms of q_{ab}, p_{ab}, r_a in (2.48). Notice that the integrals inside the "entropic" G_S and the G_{SE} tersm can be solved analytically by using multivariate Gaussian integrals identities.

Replica Symmetric Ansatz We focus on the saddle points that preserve the symmetry between the exchange of replicas (RS)

$$r_a = r\,, \qquad \hat{r}_a = \hat{r}\,, \qquad\qquad \forall a \in [n] \tag{2.54a}$$

$$q_{aa} = q_d\,, \qquad \hat{q}_{aa} = -\hat{q}_d\,, \qquad p_{aa} = p_d\,, \qquad \hat{p}_{aa} = -\hat{p}_d\,, \qquad \forall a \in [n] \tag{2.54b}$$

$$q_{ab} = q\,, \qquad \hat{q}_{ab} = \hat{q}\,, \qquad p_{ab} = p\,, \qquad \hat{p}_{ab} = \hat{p}\,, \qquad 1 \le a \le b \le n \tag{2.54c}$$

Denoting with

$$M \equiv \kappa_1 r\,, \tag{2.55a}$$

$$Q \equiv \kappa_\star^2 q + \kappa_1^2 p\,, \tag{2.55b}$$

$$Q_d \equiv \kappa_\star^2 q_d + \kappa_1^2 p_d \tag{2.55c}$$

we finally get in the small n limit the free entropy reads

$$\phi_\beta \equiv \lim_{n\to 0} \frac{\phi_\beta^{(n)}}{n} = \frac{1}{2}(q_d\hat{q}_d + q\hat{q}) + \frac{\alpha_D}{2}(p_d\hat{p}_d + p\hat{p}) - \alpha_D r\hat{r} + \mathcal{G}_{SS} + \alpha_D\mathcal{G}_{SE} + \alpha\mathcal{G}_E\,, \quad (2.56)$$

where

$$\mathcal{G}_{SS} \equiv \lim_{n\to 0} \frac{G_{SS}}{n} = \frac{1}{2}\ln\left(\frac{2\pi}{\hat{q}_d + \hat{q} + \beta\lambda}\right) + \frac{\hat{q}}{2(\hat{q}_d + \hat{q} + \beta\lambda)} \quad (2.57a)$$

$$\mathcal{G}_{SE} \equiv \lim_{n\to 0} \frac{G_{SE}}{n} = -\frac{q}{2(q_d - q)} - \frac{1}{2}\ln\left[1 + (\hat{p} + \hat{p}_d)(q_d - q)\right] + \frac{1}{2}\frac{(\hat{p} + \hat{r}^2)(q_d - q) + \frac{q}{q_d - q}}{1 + (\hat{p} + \hat{p}_d)(q_d - q)} \quad (2.57b)$$

$$\mathcal{G}_E \equiv \lim_{n\to 0} \frac{G_E}{n} = 2\int Dz_0\, H\left(-\frac{Mz_0}{\sqrt{Q - M^2}}\right)\ln\int Dz_1\, e^{-\beta\ell\left(\sqrt{Q}z_0 + \sqrt{Q_d - Q}z_1\right)} \quad (2.57c)$$

and $H(x) = \int_x^\infty Dz = \frac{1}{2}\mathrm{Erfc}\left(\frac{x}{\sqrt{2}}\right)$. The values of the 10 order parameters, namely q, $\hat{q}$, q_d, $\hat{q}_d$, p, $\hat{p}$, p_d, $\hat{p}_d$, r, $\hat{r}$ must be found self-consistently by solving the saddle point equations obtained by differentiating ϕ.

Large β limit We now restrict the discussion to the case of the convex loss functions as the MSE and the CE loss cited in the main text (2.38). In this case, we have to impose the following scalings

$$\hat{r} \to \beta\hat{r}\,, \quad (2.58a)$$

$$q = q_d - \frac{\delta q}{\beta}\,, \qquad p = p_d - \frac{\delta p}{\beta}\,, \quad (2.58b)$$

$$\hat{q} = \beta^2\delta\hat{q} + \frac{\beta}{2}\delta\hat{Q}\,, \qquad \hat{q}_d = -\beta^2\delta\hat{q} + \frac{\beta}{2}\delta\hat{Q}\,, \quad (2.58c)$$

$$\hat{p} = \beta^2\delta\hat{p} + \frac{\beta}{2}\delta\hat{P}\,, \qquad \hat{p}_d = -\beta^2\delta\hat{p} + \frac{\beta}{2}\delta\hat{P}\,. \quad (2.58d)$$

We have that the free energy which was defined in the main text in equation (2.41) is

$$-f = \lim_{\beta\to\infty}\frac{\phi_\beta}{\beta} = \frac{1}{2}\left(q_d\delta\hat{Q} - \delta q\delta\hat{q}\right) + \frac{\alpha_D}{2}\left(p_d\delta\hat{P} - \delta p\delta\hat{p}\right) - \alpha_D r\hat{r} + \mathcal{G}_{SS} + \alpha_D\mathcal{G}_{SE} + \alpha\mathcal{G}_E\,, \quad (2.59)$$

where

$$\mathcal{G}_{SS} = \frac{\delta\hat{q}}{2(\delta\hat{Q}+\lambda)} \tag{2.60a}$$

$$\mathcal{G}_{SE} = -\frac{q_d}{2\delta q} + \frac{1}{2}\frac{(\delta\hat{p}+\hat{r}^2)\,\delta q + \frac{q_d}{\delta q}}{1+\delta\hat{P}\,\delta q} \tag{2.60b}$$

$$\mathcal{G}_E = 2\int Dz_0\, H\left(-\frac{Mz_0}{\sqrt{Q_d-M^2}}\right)\max_{z_1}\left[-\frac{z_1^2}{2} - \ell(\sqrt{Q_d}z_0+\sqrt{\delta Q}z_1)\right]. \tag{2.60c}$$

We have also denoted by $\delta Q = \kappa_\star^2\delta q + \kappa_1^2\delta p$. Again the order parameters δq, $\delta\hat{q}$, q_d, $\delta\hat{Q}$, δp, $\delta\hat{p}$, p_d, $\delta\hat{P}$, r, $\hat{r}$ must be found self-consistently by solving the saddle point equations obtained by differentiating f.

Physical observables of interest As shown in multiple papers, see e.g. [8, 17] the generalization error can be obtained by computing

$$\epsilon_g = \frac{1}{\pi}\arccos\left(\frac{M}{\sqrt{Q_d}}\right). \tag{2.61}$$

The training loss can be computed by a derivative with respect to β

$$\ell_t = \lim_{\beta\to\infty}\frac{\partial(\beta f)}{\partial\beta} = 2\int Dz_0\, H\left(-\frac{Mz_0}{\sqrt{Q_d-M^2}}\right)\ell\left(\sqrt{Q_d}z_0+\sqrt{\delta Q}z_1^\star\right) \tag{2.62}$$

with

$$z_1^\star = \arg\max_{z_1}\left[-\frac{z_1^2}{2} - \ell(\sqrt{Q_d}z_0+\sqrt{\delta Q}z_1)\right] \tag{2.63}$$

The test loss can be computed as follows [17]

$$\begin{aligned}\ell_g &\equiv \langle \mathbb{E}_{\boldsymbol{\xi}^\star}\ell\left(y^\star\hat{y}^\star\right)\rangle \\ &= \int\frac{dud\hat{u}}{2\pi}\frac{d\lambda d\hat{\lambda}}{2\pi}\,\ell\left(\mathrm{sign}(u)\lambda\right)e^{iu\hat{u}+i\lambda\hat{\lambda}-\frac{\hat{u}^2}{2}-\frac{1}{2}Q_d\hat{\lambda}^2-M\hat{u}\hat{\lambda}} \\ &= 2\int Dv\,\ell\left(-\sqrt{Q_d}v\right)H\left(-\frac{Mv}{\sqrt{Q_d-M^2}}\right).\end{aligned} \tag{2.64}$$

where $\boldsymbol{\xi}^\star$ represents a new extracted pattern, $y^\star$ its corresponding label and $\hat{y}^\star$ the prediction of the model.

2.4.2 Analytical determination of the BMD

In the following subsections we will show the derivation of the annealed averages on the inputs of the numerator and denominator of the BMD.

The definition (2.15) reads

$$M_f(\boldsymbol{w}) \equiv \frac{\frac{1}{2}\sum_{k=1}^{D}\left\langle(\hat{y}(\boldsymbol{w};\boldsymbol{x})-\hat{y}(\boldsymbol{w};\boldsymbol{x}^{\oplus k}))^2\right\rangle}{\langle\hat{y}^2(\boldsymbol{w};\boldsymbol{x})\rangle-\langle\hat{y}(\boldsymbol{w};\boldsymbol{x})\rangle^2}\,. \tag{2.65}$$

where $\langle\bullet\rangle$ and $x^{\oplus k}$, defined in (2.10) and (2.16), entail an expectation over i.i.d. uniform binary inputs.

Annealed average of the denominator of the BMD

The denominator in the definition of the BMD is easy to analyze. Applying GET, we obtain

$$\begin{aligned}\left\langle\hat{y}^2(\boldsymbol{w};\boldsymbol{x})\right\rangle &= \left\langle\frac{1}{N}\sum_{i,j}w_iw_j\,\sigma\left(\frac{1}{\sqrt{D}}\sum_{k=1}^{D}F_{ki}\,x_k\right)\sigma\left(\frac{1}{\sqrt{D}}\sum_{k=1}^{D}F_{kj}\,x_k\right)\right\rangle\\ &= \left\langle\frac{1}{N}\sum_{i,j}w_iw_j\left(\kappa_0+\frac{\kappa_1}{\sqrt{D}}\sum_{k=1}^{D}F_{ki}\,x_k+\kappa_\star\eta_i\right)\left(\kappa_0+\frac{\kappa_1}{\sqrt{D}}\sum_{k=1}^{D}F_{kj}\,x_k+\kappa_\star\eta_j\right)\right\rangle\\ &= \left(\frac{\kappa_0}{\sqrt{N}}\sum_i w_i\right)^2+\frac{\kappa_1^2}{D}\sum_k\left(\frac{1}{\sqrt{N}}\sum_i F_{ki}w_i\right)\left(\frac{1}{\sqrt{N}}\sum_j F_{kj}w_j\right)+\frac{\kappa_\star^2}{N}\sum_i w_i^2 = Q_d\end{aligned} \tag{2.66}$$

The average squared is instead simply given by

$$\langle\hat{y}(\boldsymbol{w};\boldsymbol{x})\rangle^2 = \left(\frac{\kappa_0}{\sqrt{N}}\sum_i w_i\right)^2 \tag{2.67}$$

Therefore the denominator in the BMD reads

$$\left\langle \hat{y}^2(\boldsymbol{w};\boldsymbol{x}) \right\rangle - \left\langle \hat{y}(\boldsymbol{w};\boldsymbol{x}) \right\rangle^2 = \frac{\kappa_1^2}{D} \sum_k \left(\frac{1}{\sqrt{N}} \sum_i F_{ki} w_i \right) \left(\frac{1}{\sqrt{N}} \sum_j F_{kj} w_j \right) + \frac{\kappa_\star^2}{N} \sum_i w_i^2 = Q_d \tag{2.68}$$

where Q_d is the overlap obtained by the RS saddle point equations provided in the previous section.

Annealed average of the numerator of the BMD

More work has to be done to compute the numerator of the BMD. We can write it as

$$\begin{aligned}
&\sum_{k'=1}^{D} \left\langle (\hat{y}(\boldsymbol{w};\boldsymbol{x}) - \hat{y}(\boldsymbol{w};\boldsymbol{x}^{\oplus k'}))^2 \right\rangle = \\
&= \sum_{k'=1}^{D} \left\langle \left(\frac{1}{\sqrt{N}} \sum_{i=1}^{N} w_i \, \sigma \left(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} F_{ki} \, x_k \right) - \frac{1}{\sqrt{N}} \sum_{i=1}^{N} w_i \, \sigma \left(\frac{1}{\sqrt{D}} \sum_{k \neq k'} F_{ki} \, x_k + \frac{F_{k'i} \tilde{x}_{k'}}{\sqrt{D}} \right) \right)^2 \right\rangle \\
&= \sum_{k'=1}^{D} \left\langle \left(\frac{1}{\sqrt{N}} \sum_{i=1}^{N} w_i \, \sigma \left(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} F_{ki} \, x_k \right) - \frac{1}{\sqrt{N}} \sum_{i=1}^{N} w_i \, \sigma \left(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} F_{ki} \, x_k - \frac{F_{k'i}(x_{k'} - \tilde{x}_{k'})}{\sqrt{D}} \right) \right)^2 \right\rangle
\end{aligned} \tag{2.69}$$

Taylor expanding the second term we have

$$\begin{aligned}
&\frac{1}{2} \sum_{k'=1}^{D} \left\langle (\hat{y}(\boldsymbol{w};\boldsymbol{x}) - \hat{y}(\boldsymbol{w};\boldsymbol{x}^{\oplus k'}))^2 \right\rangle \\
&= \frac{1}{2} \sum_{k'=1}^{D} \left\langle \left(\frac{1}{\sqrt{N}} \sum_{i=1}^{N} w_i \, \sigma' \left(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} F_{ki} \, x_k \right) \frac{F_{k'i}(x_{k'} - \tilde{x}_{k'})}{\sqrt{D}} \right)^2 \right\rangle \\
&= \frac{1}{2} \sum_{k'=1}^{D} \left\langle \frac{1}{N} \sum_{i,j} w_i w_j \, \sigma' \left(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} F_{ki} \, x_k \right) \sigma' \left(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} F_{kj} \, x_k \right) \frac{F_{k'i} F_{k'j} (x_{k'} - \tilde{x}_{k'})^2}{D} \right\rangle .
\end{aligned} \tag{2.70}$$

We then apply the GET and we take the average over the inputs getting

$$\begin{aligned}
&\frac{1}{2}\left\langle \sigma'\left(\frac{1}{\sqrt{D}}\sum_{k=1}^{D}F_{ki}\,x_k\right)\sigma'\left(\frac{1}{\sqrt{D}}\sum_{k=1}^{D}F_{kj}\,x_k\right)(x_{k'}-\tilde{x}_{k'})^2\right\rangle \\
&=\frac{1}{2}\left\langle \left[\bar{\kappa}_0+\frac{\bar{\kappa}_1}{\sqrt{D}}\sum_{k=1}^{D}F_{ki}\,x_k+\bar{\kappa}_\star\eta_i\right]\left[\bar{\kappa}_0+\frac{\bar{\kappa}_1}{\sqrt{D}}\sum_{k=1}^{D}F_{kj}\,x_k+\bar{\kappa}_\star\eta_j\right](x_{k'}-\tilde{x}_{k'})^2\right\rangle \\
&=\bar{\kappa}_0^2+\bar{\kappa}_\star^2\delta_{ij}+\frac{\bar{\kappa}_1^2}{2D}\sum_{k=1}^{D}\sum_{k''=1}^{D}F_{ki}\,F_{k''j}\left\langle x_k x_{k''}(x_{k'}-\tilde{x}_{k'})^2\right\rangle \\
&=\bar{\kappa}_0^2+\bar{\kappa}_\star^2\delta_{ij}+\frac{\bar{\kappa}_1^2}{D}\left(2F_{k'i}\,F_{k'j}+\sum_k F_{ki}\,F_{kj}\right)=\bar{\kappa}_0^2+\bar{\kappa}_\star^2\delta_{ij}+\frac{\bar{\kappa}_1^2}{D}\sum_k F_{ki}\,F_{kj}\,.
\end{aligned} \tag{2.71}$$

where we have denoted for simplicity by $\bar{\kappa}_0$, $\bar{\kappa}_1$, $\bar{\kappa}_2$, $\bar{\kappa}_\star$ the coefficients in equation (2.45) for σ'. Notice that in all the steps that we have done to arrive performing the average over the inputs we have not used anywhere the binary nature of the inputs. It is therefore easy to see that we would have obtained the same result if we had chosen a probability distribution on the inputs with the same first two moments (e.g., a standard normal distribution). Inserting this back into the previous equation, we get

$$\begin{aligned}
&\frac{1}{2}\sum_{k'=1}^{D}\left\langle(\hat{y}(\boldsymbol{w};\boldsymbol{x})-\hat{y}(\boldsymbol{w};\boldsymbol{x}^{\oplus k'}))^2\right\rangle \\
&=\sum_{k'=1}^{D}\frac{1}{N}\sum_{ij}w_i w_j\left[\bar{\kappa}_0^2+\bar{\kappa}_\star^2\delta_{ij}+\frac{\bar{\kappa}_1^2}{D}\sum_k F_{ki}\,F_{kj}\right]\frac{F_{k'i}F_{k'j}}{D} \\
&=\frac{\bar{\kappa}_0^2}{N}\sum_{ij}\Omega_{ij}w_i w_j+\frac{\bar{\kappa}_\star^2}{N}\sum_i\Omega_{ii}w_i^2+\frac{\bar{\kappa}_1^2}{N}\sum_{ij}\Omega_{ij}^2 w_i w_j=\frac{1}{N}\sum_{ij}\bar{\Psi}_{ij}w_i w_j
\end{aligned} \tag{2.72}$$

where we have introduced

$$\bar{\Psi}_{ij}\equiv\bar{\kappa}_\star^2\,\Omega_{ii}\delta_{ij}+\bar{\kappa}_0^2\,\Omega_{ij}+\bar{\kappa}_1^2\,\Omega_{ij}^2\,, \tag{2.73a}$$

$$\Omega_{ij}\equiv\frac{1}{D}\sum_{k=1}^{D}F_{ki}F_{kj}\,. \tag{2.73b}$$

Therefore the mean dimension for a generic non-linearity σ is

$$M_f(\boldsymbol{w})=\frac{\frac{1}{N}\sum_{ij}\bar{\Psi}_{ij}w_i w_j}{Q_d} \tag{2.74}$$

BMD for an odd non-linearity σ

If we assume the activation function to be odd we have $\bar{\kappa}_1 = 0$, and $\bar{\kappa}_0 = \kappa_1$. Therefore we can write the BMD in terms of the order parameters only

$$M_f \equiv \left\langle M_f(\boldsymbol{w}) \right\rangle_{\boldsymbol{w}} = \frac{Q_d + \left(\bar{\kappa}_\star^2 - \kappa_\star^2\right) q_d}{Q_d} \tag{2.75}$$

where

$$\bar{\kappa}_\star^2 - \kappa_\star^2 = \int Dz \left(\sigma'(z)\right)^2 - \int Dz\, \sigma^2(z) = \bar{\kappa}_2 - \kappa_2 \tag{2.76}$$

Notice that in the case of a linear activation function the BMD is always 1 since a flip in the inputs will induce always the same response. Notice that equation (2.75) can be

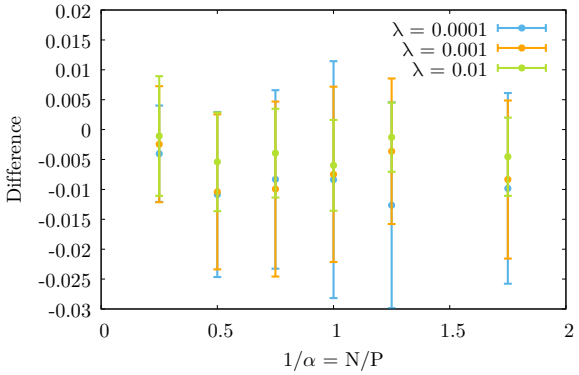

Figure 1: Difference between the BMD estimated using Monte Carlo method and using equation (2.75). The compact formula is in agreement with the computationally more expensive Monte Carlo estimation of the BMD.

used as an alternative (very efficient) way of computing the BMD, without using Monte Carlo method. We show in Fig. 1 how the difference between the BMD estimated using Monte Carlo and the one using equation (2.75) showing good agreement.

In the limit of $1/\alpha \to 0$, at fixed $\alpha_T = P/D$ (i.e. $N \to 0$), the solution to the saddle point equations show that $p_d \to q_d$; in this limit the BMD goes to

$$M_f = 1 + \frac{\bar{\kappa}_\star^2 - \kappa_\star^2}{\kappa_\star^2 + \kappa_1^2} = \frac{\bar{\kappa}_2}{\kappa_2}\,. \tag{2.77}$$

For example for $\sigma(x) = \tanh(x)$ activation we get $M_f = 1.1778$ as it is displayed in Fig. 2.

2.4.3 The resulting analytical BMD

Combining the results for the numerator and the denominator from the previous section we obtain:

$$M_f(\boldsymbol{w}) = \frac{\frac{1}{N}\sum_{ij}\bar{\Psi}_{ij}w_i w_j}{\frac{1}{N}\sum_{ij}\Psi_{ij}w_i w_j} \tag{2.78}$$

where we defined

$$\Omega_{ij} \equiv \frac{1}{D}\sum_{k=1}^{D} F_{ki}F_{kj}\,. \tag{2.79a}$$

$$\bar{\Psi}_{ij} \equiv \bar{\kappa}_\star^2\,\Omega_{ii}\mathbb{I}_{ij} + \bar{\kappa}_0^2\,\Omega_{ij} + \bar{\kappa}_1^2\,\Omega_{ij}^2\,, \tag{2.79b}$$

$$\Psi_{ij} \equiv \kappa_\star^2\,\mathbb{I}_{ij} + \kappa_1^2\,\Omega_{ij}\,. \tag{2.79c}$$

and the coefficients κ are defined as expectations of derivatives of the activation function over a standard Gaussian measure $Dz = \frac{e^{-z^2/2}}{\sqrt{2\pi}}dz$:

$$\kappa_0 = \int Dz\,\sigma(z)\,, \qquad \kappa_1 = \int Dz\,\sigma'(z)\,, \qquad \kappa_2 = \int Dz\,\sigma^2(z)\,, \tag{2.80a}$$

$$\bar{\kappa}_0 = \kappa_1\,, \qquad \bar{\kappa}_1 = \int Dz\,\sigma''(z)\,, \qquad \bar{\kappa}_2 = \int Dz\,\left(\sigma'(z)\right)^2\,, \tag{2.80b}$$

$$\kappa_\star^2 = \kappa_2 - \kappa_1^2 - \kappa_0^2\,, \qquad \bar{\kappa}_\star^2 = \bar{\kappa}_2 - \bar{\kappa}_1^2 - \bar{\kappa}_0^2\,. \tag{2.80c}$$

The mean dimension therefore explicitly depends on the model parameters $\boldsymbol{w}$. We stress again that the above expression (2.78) is universal: evaluating the MD with respect to a different i.i.d. input distributions with matching first and second moments would give exactly the same result.

In Fig. 2 we show the plot of the generalization error and the corresponding BMD of the RFM at a fixed α_T, as a function of $1/\alpha$ for the MSE (left panels) and CE loss (right panels). As shown in [8], for small regularization λ the generalization error develops

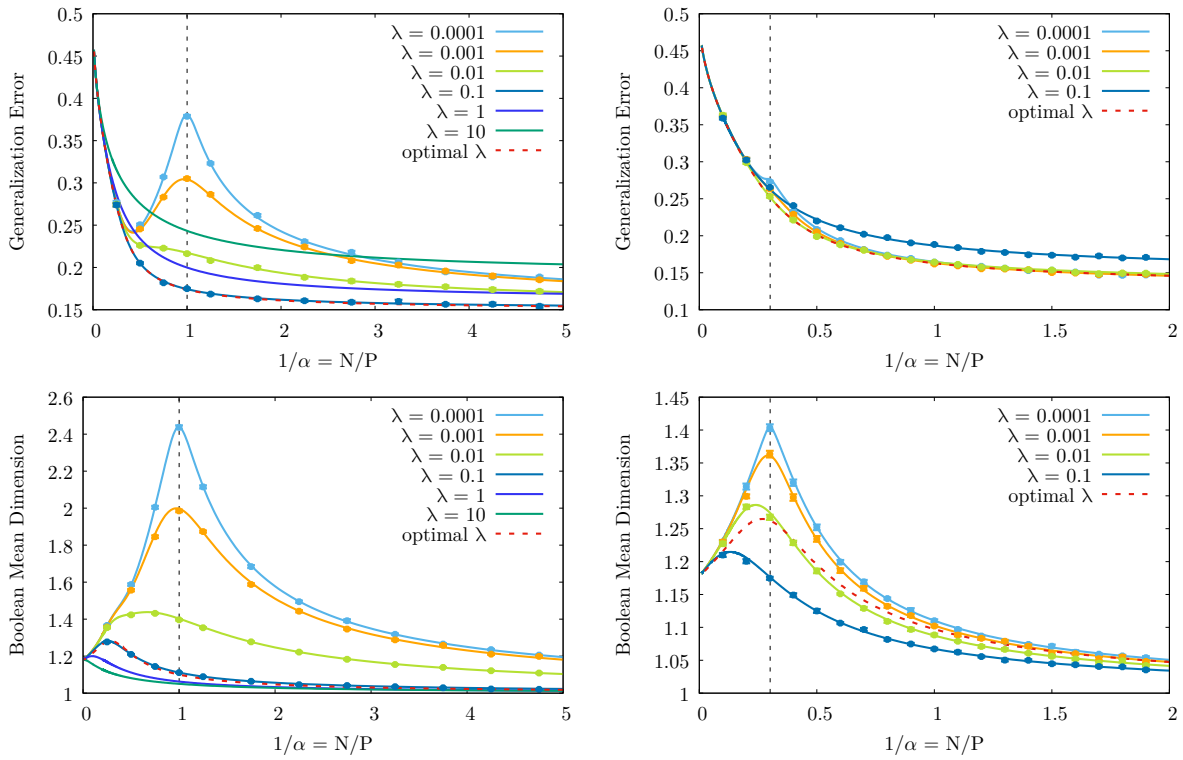

Figure 2: Generalization error (top panels) and BMD (bottom panels) as a function of the overparameterization degree $1/\alpha = N/P$, for fixed $\alpha_T = P/D = 3$ and with $\sigma = \tanh$. The left and right panels represents respectively the case of the MSE and of the CE loss. Several values of the regularization λ are displayed, together with the optimal one (which was found by minimizing the generalization error for each value of α, see red dashed line). As it can be seen in both plots, for small regularization λ, the location of the peak in the generalization error exactly coincides with the one in the BMD (vertical dashed lines). As one increases the regularization the peak in the both the generalization and the BMD is milded.

a peak approximately where the model starts to fit all training data. In the case of the MSE loss, this threshold is often called interpolation threshold and it is located at $N = P$. When using the CE loss, this happens when the projected data becomes linearly separable and the exact location of the threshold strongly depends on the input statistics and features. Exactly in the correspondence of the generalization error peak the BMD displays its own peak, meaning that the function implemented by the network is more sensitive to perturbation of the inputs.

An interesting insight can be deduced from the behavior of the BMD at the optimal value of regularization for the RFM (dashed red curves in Fig. 2). While the generalization

error becomes monotonic as the over-parametrization is increased, the BMD still reaches a peak at first and then descends to 1 only in the kernel limit $N/P \to \infty$. This might be surprising since the ground-truth linear model, the teacher, has BMD equal to 1 and one would expect the best generalizing RFM to achieve the best possible approximation of this function and therefore to match its BMD. However, blind minimization of the BMD is not compatible with good generalization, as seen from the performance of the RFM with very large regularization λ. The explanation of this comes from the architectural mismatch between the linear teacher and the RFM: according to the GET the RFM learning problem is equivalent to a linear problem with an additional noise with an intensity regulated by the degree of non-linearity of the activation function [11]. This noise initially forces the under-parameterized RFM to overstretch its parameters to fit the data, causing an increased sensitivity to input perturbations. As the over-parameterization is increased, the RFM becomes equivalent to an optimally regularized linear model [8] and the BMD slowly drops to 1 in this limit.

Note that in the large dataset limit, when $\alpha, \alpha_T \to \infty$ with $\alpha_D = \mathcal{O}(1)$, a secondary peak for the BMD of the RFM emerges around $\alpha_D = 1$, i.e. when the number of parameters of the RFM is the same as the number of input features. This peak is caused by the insurgence of singular values in the spectrum of the covariance matrix Ω and is more accentuated at lower values of the regularization. Since modern deep networks operate in a completely different regime from the large dataset limit specified above, we expect this secondary peak not to be visible in realistic settings. For example, in the above plots in the low regularization regime, this peak is overshadowed by the main BMD peak. We analyze this phenomenology in detail in the next section.

2.5 Double peak behavior of the BMD

As can be seen in Fig. 3, if α_T is sufficiently large the BMD can display a peak in

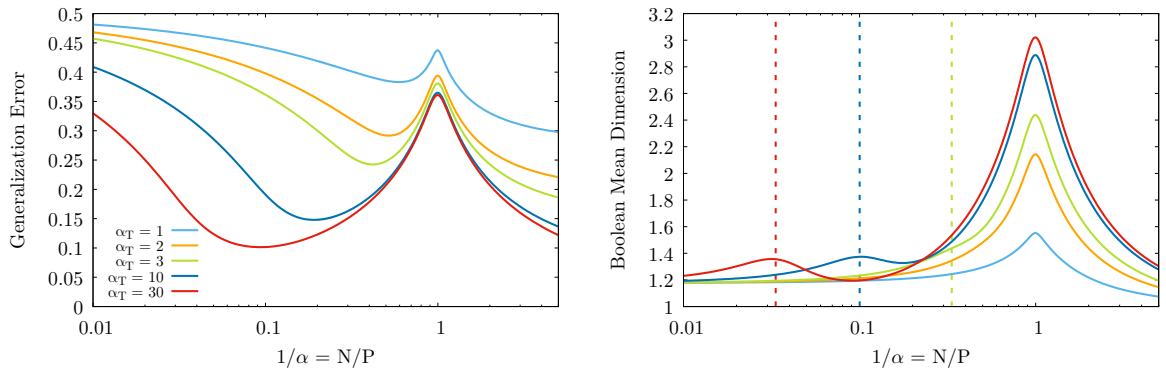

Figure 3: Generalization error (left) and BMD (right) as a function of $1/\alpha$ for $\lambda = 10^{-4}$ and $\sigma = \tanh$, for several values of $\alpha_T = P/D$. The loss used is the MSE. For low values of α_T the BMD displays a double descent behavior as showed in the main text. Increasing α_T, contrary to generalization, the BMD shows a triple descent behavior: a secondary peak in the BMD appears at $\alpha = \alpha_T$, i.e. $N = D$ (dashed vertical lines).

addition to the one located at the interpolation threshold ($N = P$ for the MSE loss). This secondary peak is located at $\alpha = \alpha_T$, i.e. when the number of parameters is equal to the input dimension $N = D$. This peak is not present in the generalization. We remark here that this behavior observed in the BMD is reminescent of the triple descent behaviour observed in [11], but is nonetheless different in nature. Indeed, in [11] the triple descent was observed in the test loss, when fixing N and D (i.e. $\alpha_D = D/N$) and changing P[1]; the authors observe a peak in the test loss when $P = D$ in addition to the "classical" double descent peak when $P = N$. This "secondary" peak can be observed only if the activation function is linear $\sigma(x) = x$ or if the labels are corrupted by Gaussian noise $\zeta^\mu \sim \mathcal{N}(0, 1)$

$$y^\mu = \text{sign}\left(\frac{1}{\sqrt{D}}\sum_k w_k^T \xi_k^\mu + \sqrt{\Delta}\zeta^\mu\right) . \tag{2.81}$$

where Δ is a non-negative parameter modulating the noise intensity. It is easy to show that the only term to change in the free energy (2.59) because of the noise is the energetic term which is modified as

$$\mathcal{G}_E = 2\int Dz_0\, H\left(-\frac{Mz_0}{\sqrt{Q_d - M^2 + \Delta}}\right) \max_{z_1}\left[-\frac{z_1^2}{2} - \ell(\sqrt{Q_d}z_0 + \sqrt{\delta Q}z_1)\right] . \tag{2.82}$$

[1]this is different from our setting where we fix P and D i.e. $\alpha_T = P/D$ and change N.

We show in Fig. 4 that even if the test loss has a secondary peak when $\Delta > 0$ at $P = D$,

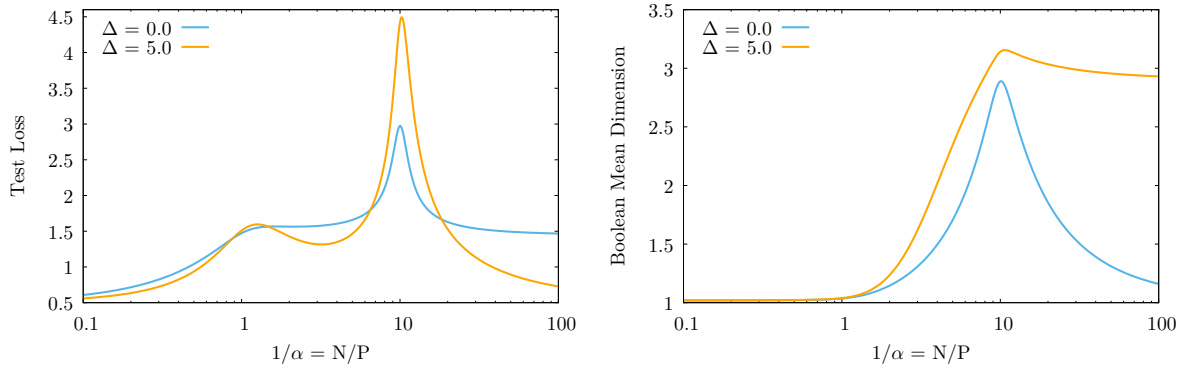

Figure 4: Test loss (left) and BMD (right) as a function of $1/\alpha$ for fixed $\alpha_D = 0.1$, $\lambda = 10^{-4}$, $\sigma = \tanh$ and for 2 values of the noise in the labels $\Delta = 0, 5$. The loss used is the MSE. Even if the test loss displays a secondary peak at $P = D$, the BMD does not.

this peak is not present in the BMD.

In Fig. 5, we show how the regularization λ can not only attenuate the "primary" peak at $P = N$, but also make the secondary peak disappear at $N = D$.

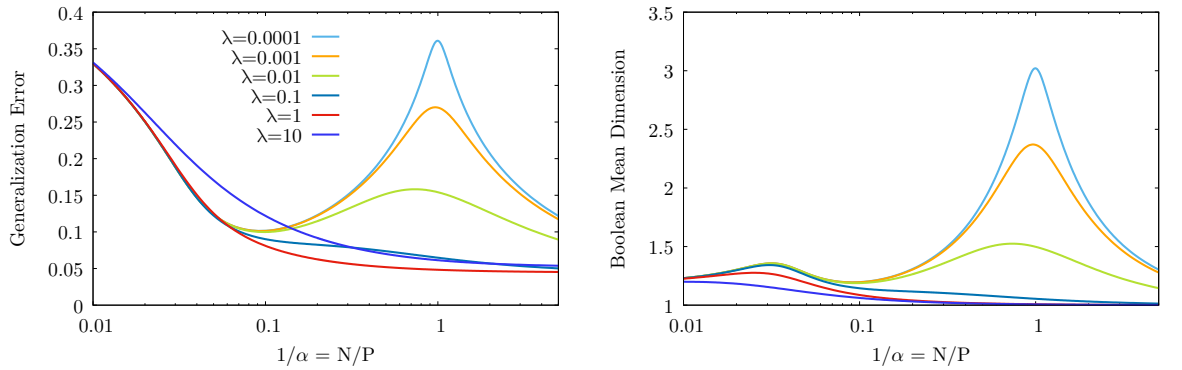

Figure 5: Generalization error (left) and BMD (right) as a function of $1/\alpha$ for $\alpha_T = 30$ and $\sigma = \tanh$ for several values of the regularization λ. The loss used is the MSE. Increasing the regularization the peak corresponding to $P = N$ disappears before the one located at $N = D$.

2.5.1 Explanation of the secondary peak of the BMD around $N = D$

The root of this phenomenon can be found in the behavior of the spectrum of the covariance matrix $\Omega = F^T F/D$. To see this, we first consider a simplified setting where the

phenomenon can be easily analytically traced. Consider a linear regression in the RFM with linear teacher. Calling $X_p \in \mathbb{R}^{P\times N} = \sigma(XF/\sqrt{D})$ the projected inputs of the RFM, with $X \in \mathbb{R}^{P\times D}$, and $F \in \mathbb{R}^{D\times N}$, the Gaussian Equivalence implies that the learning problem is equivalent to a linear regression with data:

$$X_p^{GET} = \kappa_1 XF/\sqrt{D} + \kappa_\star Z \tag{2.83}$$

with $Z \in \mathbb{R}^{P\times N}$ and $Z_{ij} \sim \mathcal{N}(0,1)$. The labels $Y \in \mathbb{R}^P$ are given by a linear teacher $w_T \in \mathbb{R}^D$:

$$Y = Xw_T \tag{2.84}$$

The ordinary least square (OLS) estimator gives a closed form solution for the trained weights:

$$\begin{aligned}
w &= (X_p^T X_p/N + \lambda\mathbb{I})^{-1} X_p^T Y/\sqrt{N} \\
&= \frac{\alpha}{P}\left(\frac{\alpha}{P}(\kappa_1\frac{XF}{\sqrt{D}} + \kappa_\star Z)^T(\kappa_1\frac{XF}{\sqrt{D}} + \kappa_\star Z) + \lambda\mathbb{I}\right)^{-1}\left(\kappa_1\frac{XF}{\sqrt{D}} + \kappa_\star Z\right)^T Xw_T \\
&= \frac{\alpha}{P}\left(\frac{\alpha}{P}(\kappa_1\frac{XF}{\sqrt{D}} + \kappa_\star Z)^T(\kappa_1\frac{XF}{\sqrt{D}} + \kappa_\star Z) + \lambda\mathbb{I}\right)^{-1}\left(\kappa_1\frac{XF}{\sqrt{D}} + \kappa_\star Z\right)^T Xw_T \\
&= \frac{\alpha}{P}\left(\frac{\alpha}{P}\left(\kappa_1^2\frac{F^TX^TXF}{D} + \kappa_\star^2 Z^TZ + \kappa_1\kappa_\star\left(\frac{Z^TXF + F^TXZ}{\sqrt{D}}\right)\right) + \lambda\mathbb{I}\right)^{-1} \\
&\times\left(\kappa_1\frac{F^TX^TXw_T}{\sqrt{D}} + \kappa_\star Z^TXw_T\right)
\end{aligned} \tag{2.85}$$

By squaring this expression we can get the norm $q_d\, N = \|w\|^2$ of the OLS estimator. We would like to understand the behavior of this quantity when $P/N \to \infty$ and when $D/N = \alpha_D = \mathcal{O}(1)$ is varied. We can thus perform an annealed average over the dataset, averaging out X, Z, and w_T. Since we are going to square the expression, we can take advantage of:

$$\mathbb{E}\frac{X^TX}{P} = \mathbb{E}\frac{Z^TZ}{P} = \mathbb{I} \tag{2.86}$$

and defining $\Omega = F^T F/D$ we can simplify the expression as:

$$\mathbb{E}\|w\|^2 = \left(\kappa_1 \frac{w_T^T F}{\sqrt{D}} + \alpha\kappa_\star w_T^T \frac{X^T Z}{P}\right)\left(\left(\alpha\left(\kappa_1^2\Omega + \kappa_\star^2\mathbb{I}\right) + \lambda\mathbb{I}\right)^{-1}\right)^T \times$$

$$\times \left(\alpha\left(\kappa_1^2\Omega + \kappa_\star^2\mathbb{I}\right) + \lambda\mathbb{I}\right)^{-1}\left(\kappa_1 \frac{F^T w_T}{\sqrt{D}} + \alpha\kappa_\star \frac{Z^T X}{P} w_T\right) \tag{2.87}$$

If we now move to the eigenbasis of $\Omega = V\Lambda X^T$, where we also have that $F/\sqrt{D} = U\sqrt{\Lambda}V^T$, we can write this expression as a trace over the eigenvalues in Λ:

$$q_d = \mathbb{E}\frac{\|w\|^2}{N} = \frac{\sigma_{w_T}^2}{N}\sum_{i=1}^{N}\frac{(\kappa_1^2\rho_i + \alpha\kappa_\star^2)}{\left((\kappa_1^2\rho_i + \kappa_\star^2) + \lambda\right)^2} \tag{2.88}$$

Similarly, one can get an expression for the average overlap:

$$Q_d = \mathbb{E}\frac{w^T(\kappa_1^2\Omega + \alpha\kappa_\star^2)w}{N} = \frac{\sigma_{w_T}^2}{N}\sum_{i=1}^{N}\frac{(\kappa_1^2\rho_i + \kappa_\star^2)^2}{\left((\kappa_1^2\rho_i + \kappa_\star^2) + \lambda\right)^2} \tag{2.89}$$

So if we now focus on the ratio q_d/Q_d, which determines the fluctuations of the MD above $MD = 1$, we can see the impact of the spectrum of Ω, which follows a Marchenko-Pastur law with parameter $1/\alpha_D$. When $\alpha_D > 1$ the spectrum is continuous and strictly positive, with a minimum eigenvalue $\rho_- = (1 - \sqrt{1/\alpha_D})^2$. At $\alpha_D = 1$ the spectrum touches the origin, and then at smaller values of α_D (in the overparameterized regime of the RFM) the distribution splits into a delta in 0 with weight $1 - \alpha_D$ and a continuous component with increasing left extremum $\rho_- = (1 - \sqrt{1/\alpha_D})^2$ and weight α_D. Because of the additional ρ_i in the numerator of expression (2.89), when the eigenvalues of Ω approach zero they have a larger effect in q_d, therefore the MD reaches a peak.

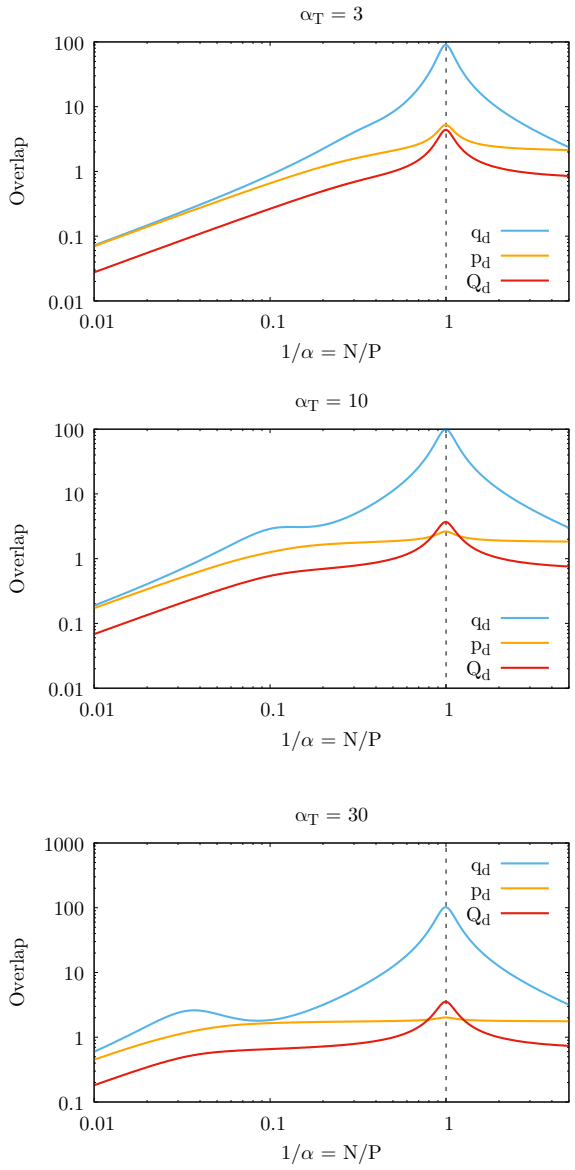

Figure 6: Plots of the overlaps q_d, p_d and $Q_d = \kappa_1^2 p_d + \kappa_\star^2 q_d$ as a function of $1/\alpha$ for $\alpha_T = 3, 10, 30$. In all plots the loss used is MSE, the regularization is $\lambda = 10^{-4}$ and $\sigma = \tanh$.

The same relationship between the parameters holds also at finite α and for a generic loss. The corresponding saddle-point equations read:

$$q_d = \frac{1}{N}\sum_{i=1}^{N} \frac{((\hat{m}^2 + \hat{q})\kappa_1^2\rho_i + \hat{q}\kappa_\star^2)}{\left(\delta\hat{q}(\kappa_1^2\rho_i + \kappa_\star^2) + \lambda\right)^2} \tag{2.90}$$

$$Q_d = \frac{1}{N}\sum_{i=1}^{N} \frac{((\hat{m}^2+\hat{q})\kappa_1^2\rho_i + \hat{q}\kappa_\star^2)(\kappa_1^2\rho_i+\kappa_\star^2)}{\Big(\delta\hat{q}(\kappa_1^2\rho_i+\kappa_\star^2)+\lambda\Big)^2} \tag{2.91}$$

where the loss function and the number of constraints determine the value of the conjugate parameters $\hat{m}, \hat{q}, \delta\hat{q}$, but the presence of an additional $(\kappa_1^2\rho_i + \kappa_\star^2)$ in the numerator of Q_d induces the same behavior of the MD around $\alpha_D = D/N = 1$, independent of the specific setting.

In Fig. 6 we show the plot of the overlaps q_d, p_d and Q_d that confirms the intuitions showed above.

2.6 Numerical results

In the following subsections of the chapter, we explore numerically the robustness of the BMD phenomenology analyzed in the RFM, considering different types of data distribution, model architecture and learning task.

Furthermore, we show that adversarially initialized models also display higher BMD, and that the increased sensitivity associated with a large BMD can hinder the robustness of the model against random perturbations of the training inputs.

Finally, we show that the location of the BMD peak is robust to the choice of input statistics used for its measurement, even in non-i.i.d. settings.

2.6.1 Experimental setup

In the following subsections, each panel displays the performance of a large number of different model architectures with varying degree of over-parameterization, trained on different datasets. Except where specified otherwise, all model are initialized with the common *Xavier* method [21] and use the Adam optimizer [22], with batch size 128 and learning rate 10^{-4}. No specific early stopping criterion is implemented. As in other works analysing the double descent, we experiment with different levels of uniformly random label noise during training (which is introduced by corrupting a random fraction of labels),

which tends to make the double descent peak more pronounced [10]. We discuss the effect of label noise below.

Model architectures

We consider different types of model architectures:

- Random feature model (RFM), described above, where the number of hidden neurons in the first (fixed) layer controls the degree of over-parameterization.
- Two-layer fully-connected network (MLP) with tanh activation, where the number of hidden neurons in the first layer controls the degree of over-parameterization.
- ResNet-18: a family of minimal ResNet [23] architectures based on the implementation of [10]. The structure is finalized with fully connected and softmax layers. As in [10], we control the over-parameterization of the model by changing the number of channels in the convolutional layers. Namely, the 4 ResNet blocks contain convolutional layers of widths $[k, 2k, 4k, 8k]$, with k varying from 1 to 20.

Both RFM and two-layer fully connected networks in our experiments use hyperbolic tangent activation functions and have weights initialized from a Gaussian distribution and bias terms initialized with zeros. The loss function optimized during training is the cross-entropy loss with L_2 regularization (the intensity of the regularization is set to zero if not specified otherwise).

Data preprocessing

In the following experiments, we use continuous inputs during the training of the models, normalizing the input features to lie within the $[-1, 1]$ interval. While such normalizations are common in preprocessing pipelines, here this procedure has also the benefit of matching the range of variability of the training inputs with that of the randomly i.i.d. sampled binary sequences used to estimate the BMD. We explore the effect of different normalization ranges in Appendix Sec. 2.B.

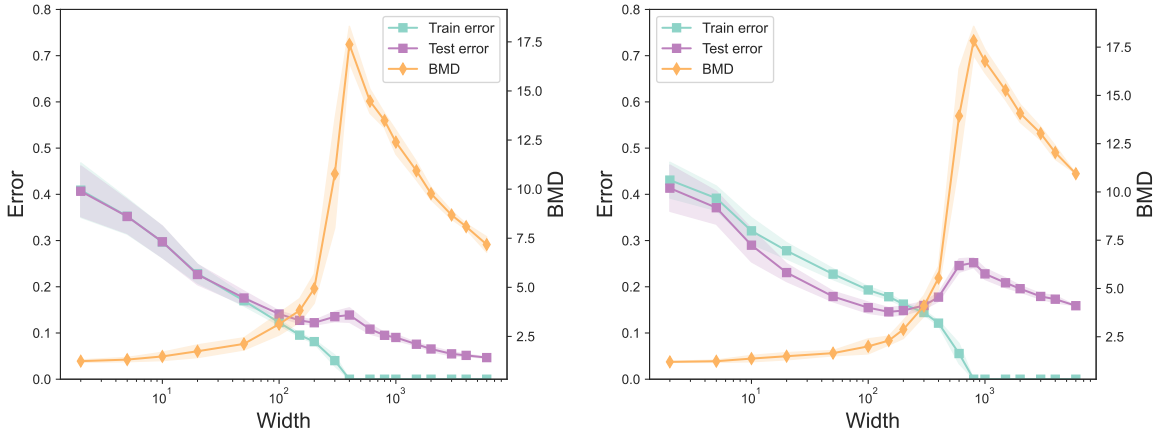

Figure 7: Train error (turquoise), test error (violet) and BMD (orange) curves of the random feature model trained on the MNIST dataset with binary labels on 5K train samples with 0% label noise (left) and 10% label noise (right), tested on 5K samples. The resulting plots represent an average (and standard deviations) obtained repeating 20 different times the experiment.

2.6.2 MD and generalization peaks as a function of overparametrization

In Fig. 7 we show train and test error, and the BMD for an RFM trained with and without label noise on binary MNIST (even vs odd digits) as a function of the hidden layer width. In Fig. 8, we instead consider a two-layer MLP trained on 10-digits MNIST (varying width) and a ResNet-18 trained on CIFAR10 (varying number of channels), both with label noise. In the multi-label case, we are defining the BMD of the network as the average of the BMDs over the classes, where the output of the network is a vector of predicted log-probabilities for each class (i.e. there is a log-softmax activation in the last layer).

Position of the BMD peak The BMD displays a peak around the point where the number of parameters of the model allows it to reach zero training error, in close correspondence with the generalization error peak. We find this phenomenology to be robust with respect to the model class, the dataset, and the over-parameterization procedure. Notice however, that standard optimizers based on SGD are able to implicitly regularize

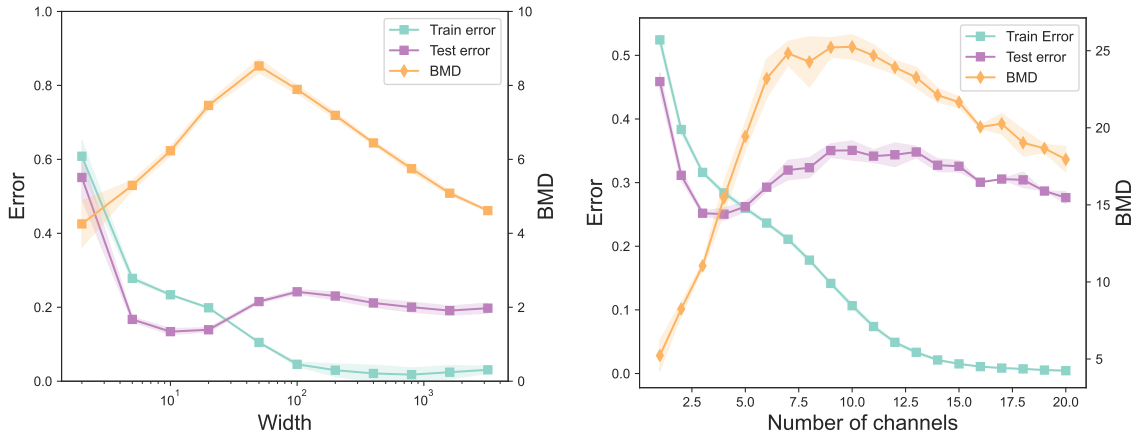

Figure 8: (Left) Train error (turquoise), test error (violet) and BMD (orange) of the two-layer fully-connected network trained on the MNIST dataset with 10 labels on 20K train samples with 20% label noise, tested on the 5K samples. The resulting plot represents an average (and standard deviations) obtained repeating 20 different times the experiment. (Right) Train error, test error and BMD of ResNet-18 trained on the CIFAR-10 with 15% label noise in the train set. The resulting plot represents an average (and standard deviations) obtained repeating 5 different times the experiment.

the trained models and can strongly reduce the peaking behavior, as already observed in the context of double descent. In the presented figures we introduced label-noise, which ensures the presence of over-fitting and is thus able to restore both peaks.

An important observation is that, in order to see this phenomenology, it is not necessary to account for the training input distribution for the evaluation of the MD, which would not be possible in the case of real data. In fact, in the over-fitting regime, it is possible to detect an increased sensitivity of the neural network function for multiple input distributions, including the i.i.d. binary inputs entailed in the BMD evaluation. This is explored further in the sub-section 2.6.6.

Asymptotic behavior of the BMD When the degree of parametrization of the model is further increased, the BMD decreases and settles on an asymptotic value. The decrease of the BMD in the number of parameters is faster with lower label noise, see Fig. 7 (left panel vs. right panel). The asymptotic value, reached in the limit of an infinite number of parameters, is task- and model-dependent. For example, in Fig. 7, the functions learned by the RFMs no longer approximate a linear model (BMD equal to 1), and are instead

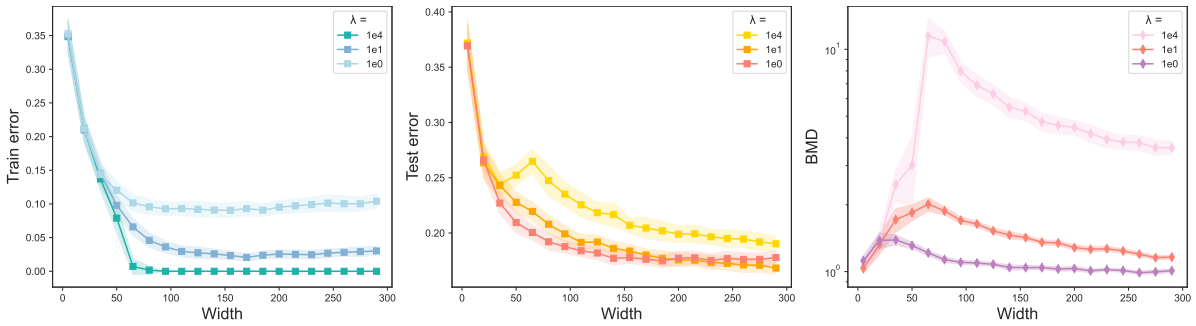

Figure 9: Impact of training with L2 regularization of the random feature model using the MNIST dataset with 10 labels, 200 train samples with no label noise and evaluating on 5K test samples. The loss used is cross-entropy. In all plots, the curves are colored by the strength of the regularization weight λ. (Left) Regularization effect on the train error. (Center) Regularization effect on the test error. (Right) Regularization effect on the BMD. The generalization error smoothly decreases with the degree of over-parameterization. Similarly, the BMD peak can be dampened by adding stronger regularization.

bound to higher values of the BMD.

Visibility of the BMD peak and Label Noise The double-descent generalization peak can be a very subtle phenomenon when the learning task is highly coherent and the data has minimal noise. In such cases, this peak becomes more evident by introducing label noise to the training data, as shown in [10]. This approach lowers the signal-to-noise ratio, increasing the potential for overfitting during training. The BMD peak, however, seems to be easily identifiable even with zero label noise, (see left panel of Fig. 7) where the generalization peak is less pronounced. Note that the BMD does not require any data (neither training nor test) in order to be estimated, so it can be used as a black-box test for assessing the proximity to the separability threshold and therefore as a signal of over-fitting.

Impact of regularization It has been shown that regularizing the model weakens the double-descent peak and that, at the optimal value of the regularization intensity, the generalization error smoothly decreases with the degree of over-parameterization. Similarly, the BMD peak can be dampened by adding stronger regularization, as shown in Fig. 9.

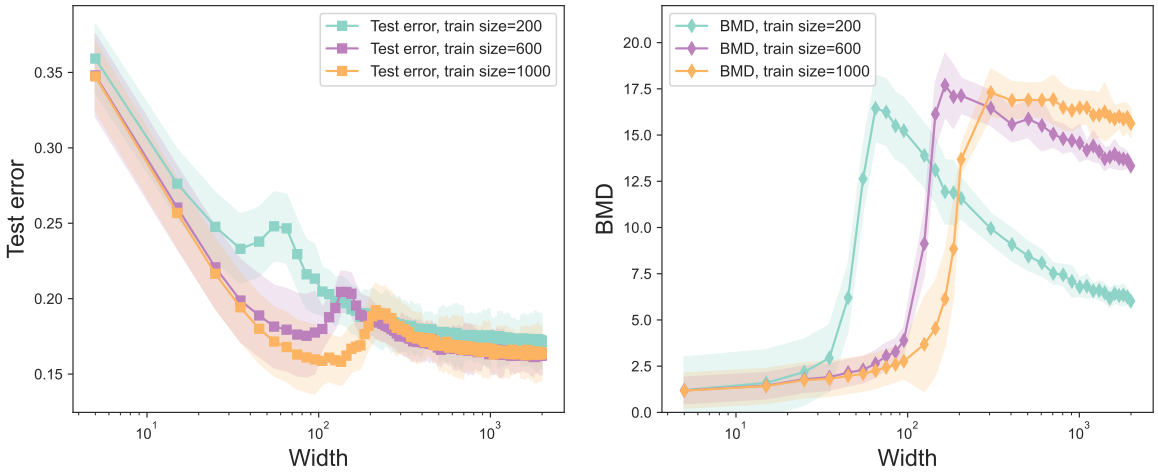

Figure 10: Effect of changing the training set size on the test error (left panel) and the BMD (right panel) of the random feature model using the MNIST dataset with 10 labels, with no label noise and evaluating on 200 test samples. The resulting plot represents an average (and standard deviations) obtained repeating 15 different times the experiment.

BMD and Training Set Size

In this section, we investigate the effect of varying the number of training samples for a fixed model capacity and training procedure. By increasing the number of training samples, starting from a low number, the same model can switch from being over- to under-parameterized. Therefore increasing the number of training samples has two effects on the test error curve: on the one hand, increasing the number of training samples decreases the test error, shifting the test error curve mostly downwards. On the other hand, increasing the number of training samples increases the capacity at which the double descent peak occurs since a higher capacity is needed until the training set is effectively memorized. This shifts the test error curve (and the BMD curve) to the right. This effect can be seen in Figure 10.

2.6.3 BMD and Adversarial Initialization

In this section, we analyze the BMD of two-layer fully connected networks under adversarial initialization [24] on the MNIST dataset. This initialization scheme can be used

to artificially hinder the generalization performance of the model, forcing it to converge on a bad minimum of the loss. We here aim to show that the initialization has also an effect on the BMD of the model, increasing the sensitivity of the network.

The adversarial initialization protocol works as follows. We train a two-layer fully connected network in two different phases: in the first phase, we push the network towards an adversarial initialization by pretraining the model with 100% label noise for a fixed amount of epochs; in the second phase, we train the model on the original dataset, with no label noise, for 200 epochs. The resulting plot, in Fig. 11 (left panel), represents an average over 15 different realizations of the experiment and shows the effect of the length of the pretraining phase on both generalization performance and BMD of the network. In agreement with our analysis, we observe a simultaneous increase of the two metrics when the adversarial initialization phase is longer and the network is driven towards worse generalization.

2.6.4 BMD and Robustness Against Adversarial Attacks

In this section, we analyze the connection between BMD of a model and its robustness to adversarial attacks. We consider a two-layer fully-connected network trained on MNIST with 10 classes. We define as our robustness measure the average count of sign flips of randomly chosen pixels, needed to change the model prediction on a test sample that was previously classified correctly. The lower the counts, the lower the robustness of the model. Varying the capacity of the model by varying the width of the hidden layer, we plot this robustness measure against the BMD of the model in Fig. 11 (right panel). We observe that BMD and robustness strongly anti-correlate, with the peak in BMD coinciding with a minimum of robustness.

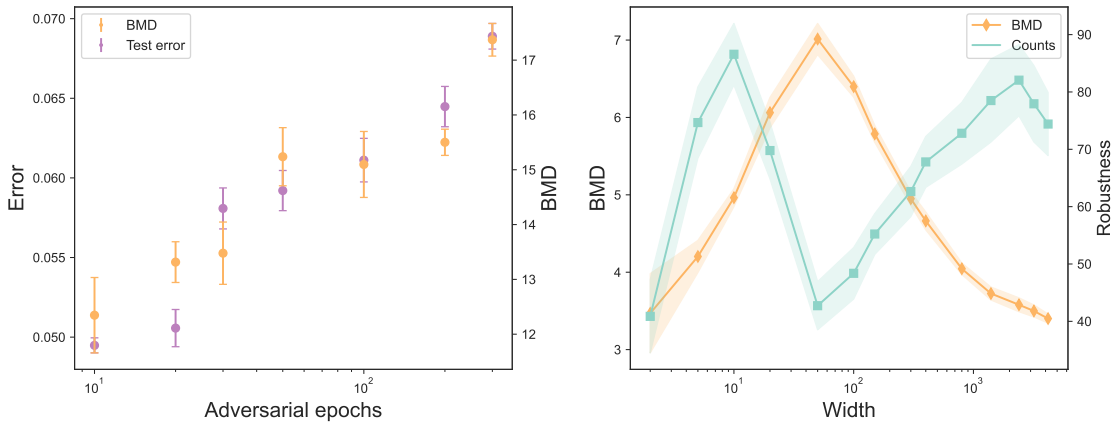

Figure 11: (Left): BMD (orange points), and test error (violet points) estimated for a two-layer fully connected network of width 10^3, trained according to the adversarial initialization protocol described in section 2.6.3 on 20K samples of the MNIST dataset. On the horizontal axis we vary the number of pretraining epochs, and plot the corresponding increase in the generalization error and the BMD of the model after the second learning stage. The points represent an average of 15 different realizations of the experiment. (Right): BMD (orange line) and counts (turquoise line) estimated for a two-layer fully connected network trained on the MNIST dataset using 20K train samples with 20% label noise and tested on 5K samples. Counts represent the average amount of sign flips of random pixels of a correctly predicted test image that are necessary to fool the model to a wrong class label. The amount is averaged over all the correctly predicted test data samples. The resulting plot represents an average of 40 different realizations of the experiment. We observe that higher values of the BMD correspond to lower robustness of the model and vice versa.

2.6.5 Pixel-Wise Contributions to BMD

The MD as expressed in eq (2.15) is proportional to a sum of contributions τ_i^2 of single features indexed by i. Similar to [25], we plot these contributions in Fig. 12 as a heatmap, where the bright spots indicate features that contribute strongly to the MD. We show four heatmaps, corresponding to different capacities and at different distances from the BMD peak, for a two-layer fully connected network trained on MNIST.

Note that the colors are normalized to the $[0, 1]$ range, so that very bright spots correspond to pixels that contribute to the BMD the most. It can be seen that for under-parametrized networks few pixels give the largest contribution to the BMD. Near the BMD peak, a large fraction of the pixels in the center of the image dominate the BMD, and for even larger capacities we again have fewer pixels with maximal values. This can be interpreted as the classifier losing "focus" at the interpolation point and paying attention to fewer patterns in the over-parametrized regime.

2.6.6 Different Distributions for Estimating BMD

In BMD estimates for the previous experiments, Eq. (2.16), we focused on the case of i.i.d. binary input features. In the RFM, however, we have shown in the section 2.4.2 analytically that there exists a universality for the MD when one considers separable input distributions with the same first and second moments. In the numerical experiments, we have also shown evidence that the BMD peak can still provide insights into the behavior of the neural network function on the training and test data, which follow very different input statistics. To explore in detail the role of the input statistics, and of the presence of correlations in the input features, we measure the MD by resampling the inputs from different distributions: in Fig. 13 we plot the normalized MD curves for features sampled from:

- a uniform binary distribution (BMD)

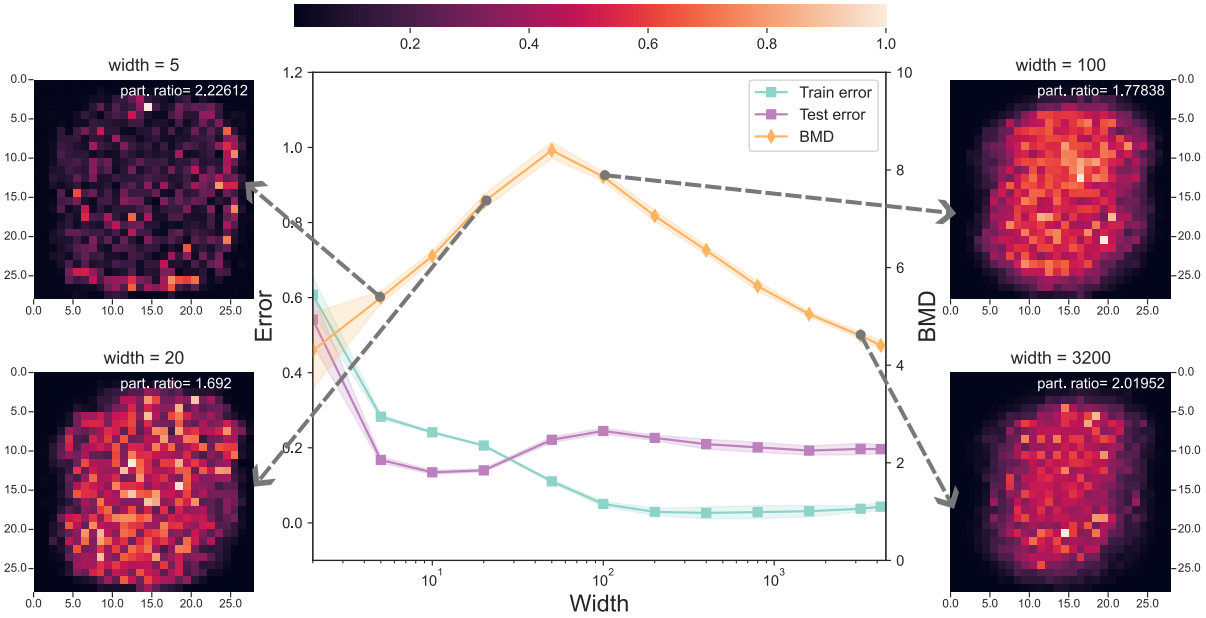

Figure 12: Heatmaps of the pixel contributions (τ_i^2 for $1 \leq i \leq 784$) estimated on the two-layer fully connected network trained on 20K samples of the MNIST dataset with 10 classes, 20% label noise and normalized to lie within $[0, 1]$ interval. The (rescaled) participation ratio is defined as $n \times \frac{\sum_{i=1}^{n} \tau_i^2}{(\sum_{i=1}^{n} \tau_i)^2}$. After the rescaling, a participation ratio of 1 indicates a uniform distribution of pixel contributions, while a value of n indicates a distribution concentrated over a single pixel. The heatmaps correspond to the contributions estimated with respect to label 0 for the models of different capacities (hidden layer dimensions) and represent only one seed, while the resulting curves on the plot represent an average over 20 different runs of the experiment.

- a standard normal (Gaussian) distribution $\mathcal{N}(0, 1)$.
- a uniform distribution in the range $[-1, 1]$.
- empirical distribution of the training data with random uniform resampling in the range $[-1, 1]$.

As one can see in Fig. 13, the MD curves estimated with binary and Gaussian i.i.d. inputs, with matching moments, are identical. With the uniform distribution, the second moment is 1/3 and this results in a slightly rescaled MD curve. Introducing correlations in the inputs, in the MD estimated over the training data distribution, the curve still shows a similar behavior, and importantly the peak is found at the same value.

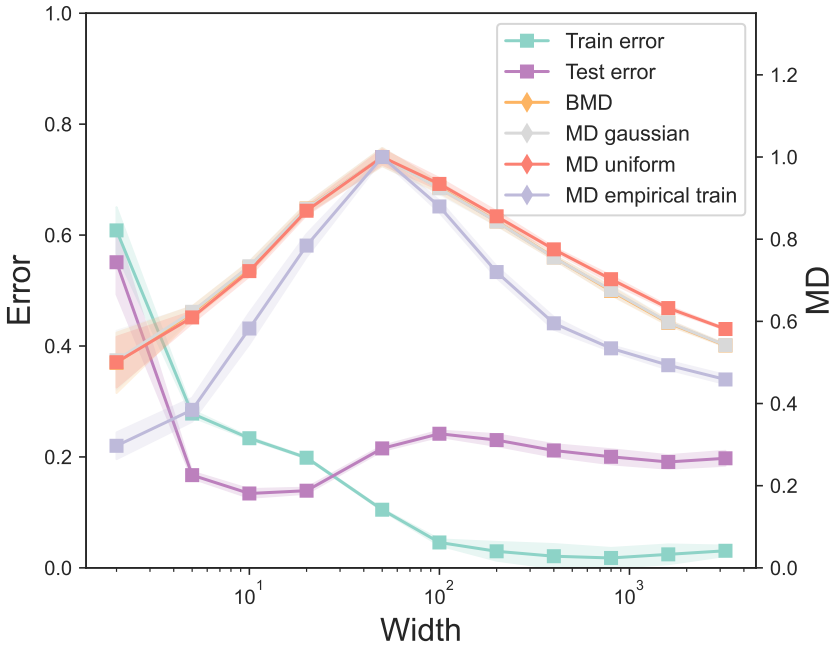

Figure 13: Mean dimensions estimated using Monte Carlo (eq. 2.16) w.r.t. different distributions of the two-layer fully connected network trained on the 20K of MNIST samples with 20% label noise and tested on 5K samples. The MD values are normalized to lie within [0,1] interval. The choice of the distribution does not affect the location of the peak. Moreover, distributions, which first two moments coincide (e.g. binary uniform Unif$\{-1, 1\}$ and Gaussian $\mathcal{N}(0, 1)$) yield the same MD pattern. The resulting plot represents an average over 20 different runs of the experiment.

Appendix

2.A Self-averaging property of the MD

The MD of the RFM at initialization demonstrates the self-averaging properties, e.g. for the case of i.i.d. gaussian weights that were projected on the space orthogonal to the $\mathbb{I}_D$ vector.

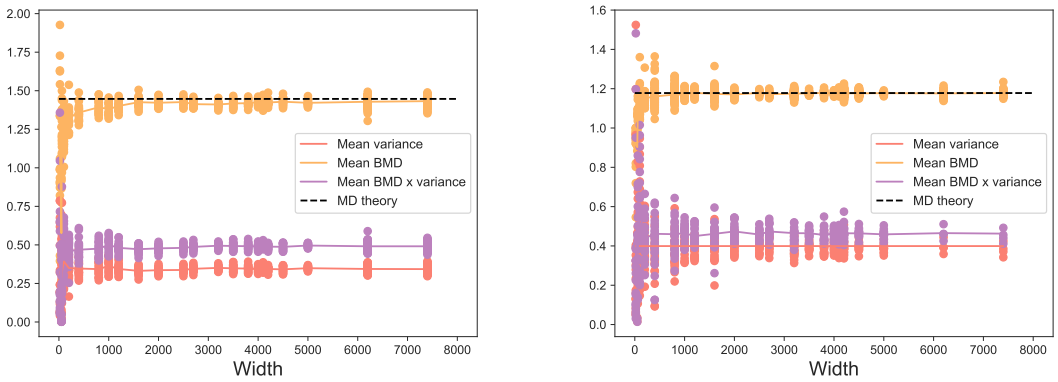

Figure 14: Correspondence of the empirical evidence of the self-averaging of the BMD computed for the RFM initialized with the gaussian weights (projected orthogonally to the $\mathbb{I}_D$ vector) and theoretical prediction of the asymptotic BMD value in the limit of the input size D and the hidden layer size N. (Left) RFM with leaky ReLU activation, (Right) RFM with tanh activation. The resulting plot represents an average over 70 different runs of the experiment.

2.A.1 Self-averaging of the MD for the trained models

In the Table 2.1 below we can observe the phenomenon of the concentration of the MD in the trained deep learning models as compared to the MD of the models at initialization.

2.B BMD and Data Normalization

In Fig. 15, we repeat the BMD calculations for RFMs trained with different ranges used for normalization. As can be seen in the figure the choice of the input data normalization does not affect the BMD pattern, but only its absolute values in this setting.

Model	Var(MD) trained model	Var(MD) untrained model
ResNet20	0.02503	2.78465
ResNet32	0.0393	21.49987
ResNet44	0.05377	138.04827
ResNet56	0.0449	792.22086
mobilenetv2 x0 5	0.07485	318.17157
mobilenetv2 x0 75	0.05270	271.84772
mobilenetv2 x1 0	0.04614	81.66546
mobilenetv2 x1 4	0.03148	128.32706
shufflenetv2 x0 5	0.02638	146.33676
shufflenetv2 x1 0	0.0266	34.79431
shufflenetv2 x1 5	0.02843	27.3121
shufflenetv2 x2 0	0.03713	24.21249
repvgg a0	0.03894	104.79538
repvgg a1	0.04875	109.59382
repvgg a2	0.05133	136.44133
repvgg a0	3.44696	74.58045
repvgg a1	4.27395	83.92942
repvgg a2	2.66022	114.86224

Table 2.1: MD empirical variances for randomly initialized and trained on cifar-10 dataset deep models estimated over 30 seeds

2.C Effect of the label corruption on the train error

In Fig. 16 we demonstrate the effect of the label corruption of the train data on the train error, which is allowing to explain the difference between the test and train errors for smaller model widths.

References

[1] Christopher Hoyt and Art B Owen. Efficient estimation of the anova mean dimension, with an application to neural net classification. *SIAM/ASA Journal on Uncertainty Quantification*, 9(2):708–730, 2021.

[2] Bradley Efron and Charles Stein. The jackknife estimate of variance. *The Annals of Statistics*, pages 586–596, 1981.

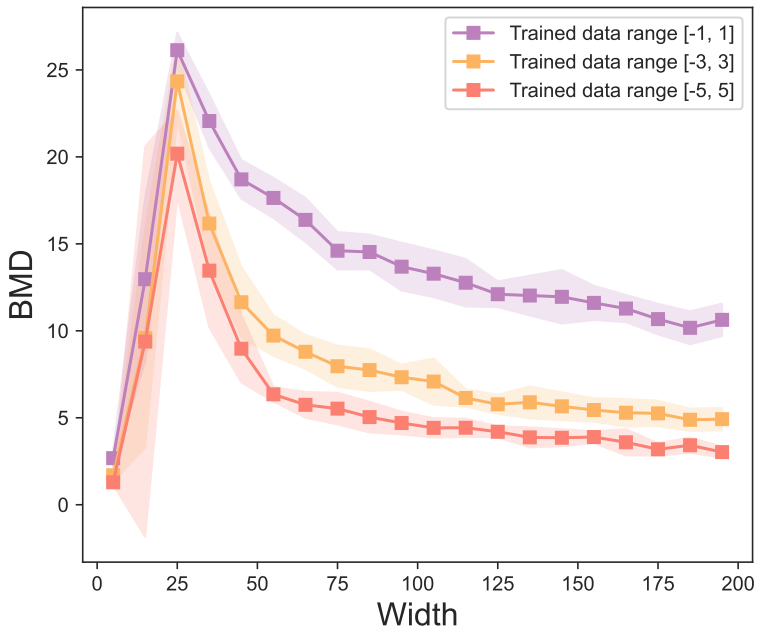

Figure 15: BMDs of the three RFMs trained on the MNIST dataset with 10 classes on 200 train samples with 0% label noise. The RFMs were trained on the datapoints normalized to lie within the intervals [-5, 5] (red line), [-3, 3] (orange line) and [-1, 1] (violet line). The resulting plot represents an average (and standard deviations) obtained repeating 12 different times the experiment.

[3] Art B Owen. The dimension distribution and quadrature test functions. *Statistica Sinica*, pages 1–17, 2003.

[4] Marco Baity-Jesi, Levent Sagun, Mario Geiger, Stefano Spigler, Gérard Ben Arous, Chiara Cammarota, Yann LeCun, Matthieu Wyart, and Giulio Biroli. Comparing dynamics: Deep neural networks versus glassy systems. In *International Conference on Machine Learning*, pages 314–323. PMLR, 2018.

[5] Mario Geiger, Stefano Spigler, Stéphane d'Ascoli, Levent Sagun, Marco Baity-Jesi, Giulio Biroli, and Matthieu Wyart. Jamming transition as a paradigm to understand the loss landscape of deep neural networks. *Physical Review E*, 100(1):012115, 2019.

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

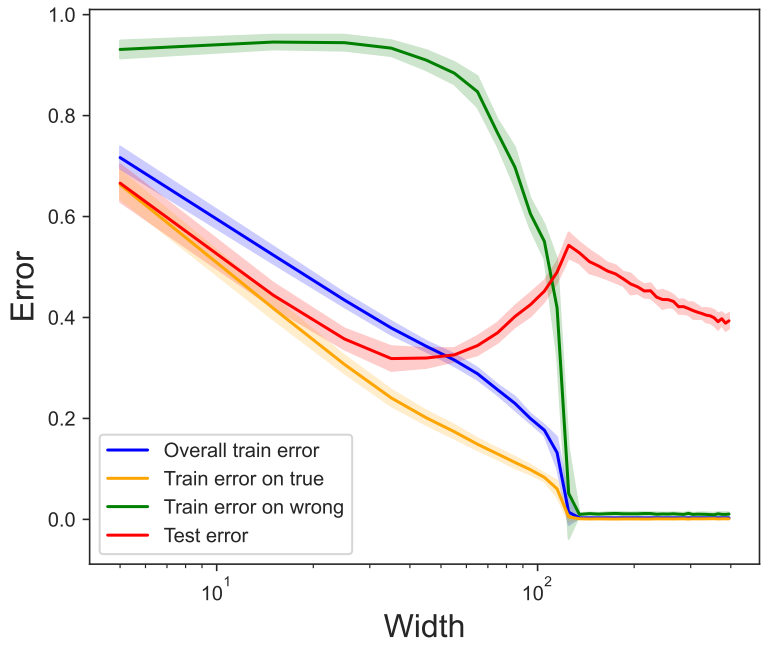

Figure 16: Train error curves of the RFM trained on the MNIST dataset with 10 classes on 1000 train samples with 20 % label noise and estimated on 1000 test samples. The plot demonstrates that better performance of the model on the test data rather than on the train data for smaller widths can be explained by a disproportionally high error on the train examples with the corrupted (wrong) labels (green line), which therefore leads to the higher overall train error (blue line), while tested only on the data with non-corrupted labels the train error (yellow line) is comparable to the test error (red line)

[7] Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and double descent curve. *arXiv preprint arXiv:1908.05355*, 2019.

[8] Federica Gerace, Bruno Loureiro, Florent Krzakala, Marc Mezard, and Lenka Zdeborova. Generalisation error in learning with random features and the hidden manifold model. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3452–3462. PMLR, 13–18 Jul 2020.

[9] Marc Mézard, Giorgio Parisi, and Miguel Angel Virasoro. *Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications*, volume 9. World Scientific Publishing Company, 1987.

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

[11] Stéphane d'Ascoli, Levent Sagun, and Giulio Biroli. Triple descent and the two kinds of overfitting: Where & why do they appear? *Advances in Neural Information Processing Systems*, 33:3058–3069, 2020.

[12] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[13] Ruixue Liu and Art B Owen. Estimating mean dimensionality of analysis of variance decompositions. *Journal of the American Statistical Association*, 101(474):712–721, 2006.

[14] Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *NIPS*, volume 3, page 5. Citeseer, 2007.

[15] Sebastian Goldt, Marc Mézard, Florent Krzakala, and Lenka Zdeborová. Modelling the influence of data structure on learning in neural networks: the hidden manifold model. *Physical Review X*, 10:041044, 2019.

[16] Bruno Loureiro, Cédric Gerbelot, Hugo Cui, Sebastian Goldt, Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Capturing the learning curves of generic features maps for realistic data sets with a teacher-student model. *arXiv preprint arXiv:2102.08127*, 2021.

[17] Carlo Baldassi, Clarissa Lauditi, Enrico M Malatesta, Rosalba Pacelli, Gabriele Perugini, and Riccardo Zecchina. Learning through atypical phase transitions in overparameterized neural networks. *Physical Review E*, 106(1):014116, 2022.

[18] Andreas Engel and Christian Van den Broeck. *Statistical mechanics of learning*. Cambridge University Press, 2001.

[19] Enrico M Malatesta. High-dimensional manifold of solutions in neural networks: insights from statistical physics. *arXiv preprint arXiv:2309.09240*, 2023.

[20] Jeffrey Pennington and Pratik Worah. Nonlinear random matrix theory for deep learning. *Advances in neural information processing systems*, 30, 2017.

[21] Xavier Glorot and Yoshua Bengio. Xavier initialization. *J. Mach. Learn. Res*, 2010.

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[24] Shengchao Liu, Dimitris Papailiopoulos, and Dimitris Achlioptas. Bad global minima exist and sgd can reach them. *Advances in Neural Information Processing Systems*, 33:8543–8552, 2020.

[25] Roman Hahn, Christoph Feinauer, and Emanuele Borgonovo. The mean dimension of neural networks–what causes the interaction effects? *arXiv preprint arXiv:2207.04890*, 2022.

Chapter 3

Entropic Gradient Descent Algorithms and Wide Flat Minima

This section continues the study of the factors that affect the generalization of neural networks and is dedicated to investigating how the geometry of the loss landscape relates to their generalization capabilities. Specifically, we examine how generalization correlates with the presence of clusters of near-minimal solutions in close proximity, which we refer to as *wide flat minima*.

The properties of flat minima in the empirical risk landscape of neural networks have been debated for some time. Increasing evidence suggests they possess better generalization capabilities with respect to sharp ones. In this chapter we firstly discuss Gaussian mixture classification models and show analytically that there exist Bayes optimal pointwise estimators which correspond to minimizers belonging to wide flat regions. These estimators can be found by applying maximum flatness algorithms either directly on the classifier (which is norm independent) or on the differentiable loss function used in learning. Next, we extend the analysis to the deep learning scenario by extensive numerical validations. Using two algorithms, Entropy-SGD and Replicated-SGD, that explicitly include in the optimization objective a flatness measure known as local entropy, we consistently improve the generalization error for common architectures (e.g. ResNet,

EfficientNet). In our experiments an easy to compute flatness measure demonstrates a clear correlation with test accuracy.

3.1 Introduction

The geometrical structure of the loss landscape of neural networks has been a key topic of study for several decades [1, 2]. One key area of investigation looks at the connection between the flatness of minima found by optimization techniques like stochastic gradient descent (SGD) and the generalization performance of the network [3, 2]. However, there are several conceptual challenges in this field: while there is growing evidence that flatness is a reliable predictor of generalization [4], modern deep networks with ReLU activations show output invariance to weight rescaling across different layers [5], which complicates the mathematical understanding.

The aim of this chapter is to explore the relationship between flatness and generalization by applying methods and algorithms from the statistical physics of disordered systems, and to validate these findings through a performance analysis of state-of-the-art deep architectures.

Recent advancements in statistical physics have yielded several insights. Notably, it has been demonstrated that wide flat minimizers are a structural feature of shallow networks, existing even when trained on random data and accessible via relatively simple algorithms, despite the presence of exponentially more numerous minima [6, 7, 3]. This, often-overlooked characteristic of neural networks, may potentially enhance their learning capabilities. In analytically manageable scenarios, previous research has shown that flatness is related to the choice of loss functions, activation functions, and generalization performance [3, 8].

We employ a notion of flatness referred to as Local Entropy [6, 7]. It measures the low-loss volume in the weight space around a minimizer. This framework has been used to introduce a variety of learning algorithms (which we will call *entropic algorithms* in

this chapter) that focus their search on flat regions [7, 9, 10].

In this chapter we aim to connect flatness and generalization properties by providing both analytical and state-of-the-art numerical results. The key contributions of our study are:

- In our analysis of Gaussian mixture classification using shallow networks, we demonstrate that the minimum norm condition required for achieving Bayes optimal performance corresponds to solutions that maximize local entropy for the classifier, which is invariant to norm. These solutions can be obtained through entropic algorithms applied to the learning loss function.
- We systematically apply two entropic algorithms, Entropy-SGD (eSGD) and Replicated-SGD (rSGD), on advanced deep architectures. With minimal or no hyperparameter tuning, we achieve enhanced generalization performance, which we find to be correlated with a measurable degree of flatness.

3.2 Related work

The idea of using the flatness of a minimum of the loss function, also called the *flatness of the posterior* and the *local area estimate of quality*, for evaluating different minimizers is several decades old [1, 11, 12]. These works connect the flatness of a minimum to information theoretical concepts like the *minimum description length* of its minimizer: flatter minima correspond to minimizers that can be encoded using fewer bits. For neural networks, a recent empirical study [2] shows that large-batch methods find sharp minima while small-batch ones find flatter ones, with a positive effect on generalization performance.

PAC-Bayes bounds can be used for deriving generalization bounds for neural networks [13]. In [14], a method for optimizing the PAC-Bayes bound directly is introduced and the authors note similarities between the resulting objective function and an objective function that searches for flat minima. This connection is further analyzed in [15]. In

[4], the authors present a large-scale empirical study of the correlation between different complexity measures of neural networks and their generalization performance. The authors conclude that PAC-Bayes bounds and flatness measures are the most predictive measures of generalization.

The concept of local entropy has been introduced in the context of a statistical mechanics approach to machine learning for discrete neural networks in [6], and subsequently extended to models with continuous weights. The general definition of the local entropy loss $\mathcal{L}_{\mathrm{LE}}$ for a system in a given configuration w (a vector of size N) can be given in terms of any common (usually, data-dependent) loss $\mathcal{L}$ as:

$$\mathcal{L}_{\mathrm{LE}}(w) = -\frac{1}{\beta} \log \int dw' \; e^{-\beta \mathcal{L}(w') - \beta\gamma \mathrm{d}(w', w)}. \tag{3.1}$$

The function d measures a distance and is commonly taken to be the squared norm of the difference of the configurations w and w':

$$\mathrm{d}(w', w) = \frac{1}{2} \sum_{i=1}^{N} \left(w'_i - w_i\right)^2 \tag{3.2}$$

The integral is performed over all possible configurations w'; for discrete systems, it can be substituted by a sum. The two parameters β and $\tilde{\gamma} = \beta\gamma$ are Legendre conjugates of the loss and the distance. For large systems, $N \gg 1$, their effect is to jointly restrict the integral to configurations w' below a certain loss $\mathcal{L}^*(w, \beta, \gamma)$ and within a certain distance $\mathrm{d}^*(w, \beta, \gamma)$ from the reference configuration w. In general, increasing β reduces $\mathcal{L}^*$ and increasing $\tilde{\gamma}$ reduces d^*.

The interpretation of this quantity is that it computes the log-volume of the configurations w' in a region of size d^* around w that have loss less or equal than $\mathcal{L}^*$. Compared to the original loss $\mathcal{L}$, it can be interpreted as a Gaussian smoothing, or a site-dependent regularization. It also has the interpretation of a non-local measure of flatness, since, for large β, configurations w with small $\mathcal{L}_{\mathrm{LE}}$ must lie in the middle of extensive regions in which a large fraction of the configurations have small loss $\mathcal{L}$.

In a neural network that performs a classification task, the most natural choice for $\mathcal{L}$ in eq. (3.1) is the training error. This is the definition that has been used in detailed analytical studies on relatively tractable shallow networks accompanied by numerical experiments, where indeed $\mathcal{L}_{\mathrm{LE}}$ has been shown to correlate with generalization error and eigenvalues of the Hessian [6, 3]. Another interesting finding is that the cross-entropy loss [3] and ReLU transfer functions [8], which have become the de-facto standard for neural networks, tend to bias the models towards high local entropy regions (computed based on the error loss).

Using the local entropy as an objective is in general computationally intractable. However, it can be approximated to derive general algorithmic schemes. Replicated stochastic gradient descent (rSGD) replaces the local entropy objective by an objective involving several replicas of the model, each one moving in the potential induced by the loss while also attracting each other. The method has been introduced in [7], but only demonstrated on shallow networks. The rSGD algorithm is very closely related to Elastic Averaging SGD (EASGD), presented in [16], even though the latter was motivated purely by the idea of enabling massively parallel training and had no theoretical basis. The main distinguishing feature of rSGD compared to EASGD when applied to deep networks is the focusing procedure by which the interaction parameter γ is gradually increased, discussed in more detail below. Another difference is that in rSGD there is no explicit main replica.

Entropy-SGD (eSGD), introduced in [9], is a method that directly optimizes the local entropy using stochastic gradient Langevin dynamics (SGLD) [17]. While the goal of this method is the same as rSGD, the optimization technique involves a double loop instead of replicas. Parle [10], combines eSGD and EASGD (with added focusing) to obtain a distributed algorithm that shows also excellent generalization performance, consistently with the results obtained in this work.

3.3 Analytical Results on shallow networks

The relation between local entropy, flatness and generalization properties has been investigated theoretically and numerically for several models in [6, 3, 8]. So far, the theoretical results were limited to the so-called teacher-student scenario in the context of classification: a training set with i.i.d. randomly-generated inputs and labels provided by a (shallow) teacher network is presented to a student network with the same architecture as the teacher. In the under-parameterized regime, in which the training set does not contain sufficient information, several local minima exist, and the ones with high local entropy were shown to have generalization errors close to the Bayes-optimal ones.

The under-parametrized teacher-student scenario considered in the above-mentioned studies is highly non-convex, and using random i.i.d inputs is not particularly realistic. Although it was shown in [3] that the phenomenology is similar with real datasets, the problem of obtaining theoretical insight into other classification tasks with different distributions remains open.

Here, we confirm the general scenario in a very simple model often used in high-dimensional statistical machine learning [18, 19, 20, 21]: Gaussian mixtures. The generative model for this task is as follows: for a given problem size N, an N-dimensional vector $\mathbf{v}^\star$ is randomly generated from a standard multivariate normal $\mathcal{N}(\mathbf{0}, \mathrm{I}_N)$; then, two classes of patterns are generated, with labels $\sigma = \pm 1$, each class being distributed as $\mathcal{N}\left(\sigma \mathbf{v}^\star/\sqrt{N}, \mathrm{I}_N\right)$. We call αN the size of the training set. Here, for simplicity, we will restrict ourselves to the case in which the two classes are balanced.

The performance of a linear classifier (a single-unit neural network, a.k.a. a perceptron) on this model has been studied recently in [22], in particular in the case in which the network is trained using the mean square error (MSE) loss. This system is prone to overfitting, especially around $\alpha \approx 1$. In [22] it was shown that generalization performances are improved by introducing an ℓ_2 regularization controlled via a positive parameter λ, and that, in the balanced case, optimal performances are achieved in the limit $\lambda \to \infty$.

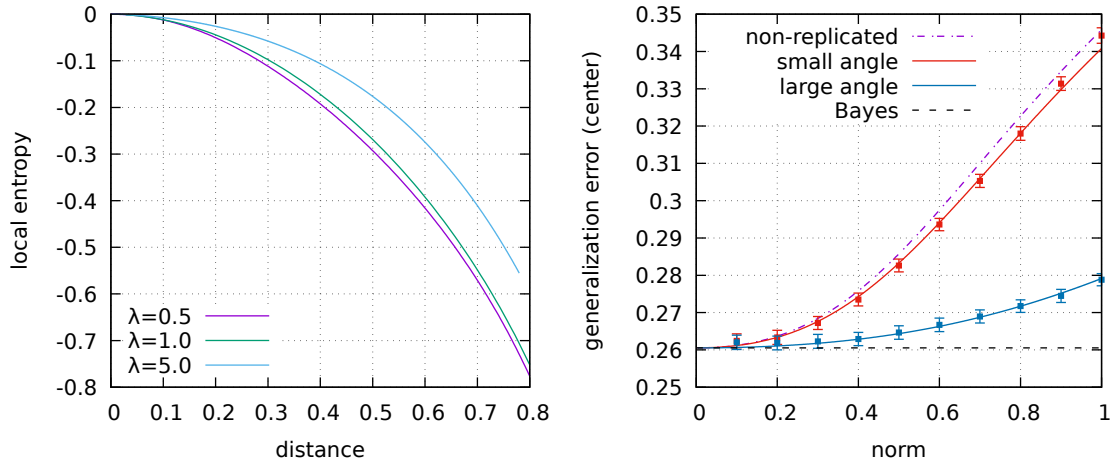

Figure 1: Left: Normalized local entropy as a function of the squared-distance d computed from reference configurations found by optimizing the regularized MSE loss, with varying regularization strength λ. Larger values of λ correspond to minimizers with better generalization properties. Right: Generalization error of the center $\bar{w}$ of a system of $y = 10$ replicas, each optimizing the MSE loss with $\alpha = 0.7$ and with a constraint on the angle between the replicas, as a function of the norm n of the replicas. The small-angle case corresponds to $\cos(\theta) = 0.9$, the large-angle case to $\cos(\theta) = 0.1$. Solid curves are theoretical results, points are numerical results obtained with $N = 1000$, averaged over 30 samples. In the limit $\theta = 0$ the results reproduce those of a single device; increasing θ the dependence on the norm reduces (the curve flattens onto the Bayes-optimal dashed line in the limit $\theta = \pi/2$ and $y \to \infty$).

In the limit of large N, adding the regularization term is equivalent to fixing the norm of the weights.

This problem is rather peculiar when compared to typical classification tasks performed with neural networks, since the training loss is convex. Indeed, Bayes-optimal performance can be achieved with a single configuration (instead of requiring a distribution) and can be easily found analytically. Additionally, the task is impossible, in the sense that no classifier can achieve zero test error (in the teacher-student context this would be similar to the case of having a "noisy", unreliable teacher).

It is interesting to consider that the output of the network (and thus the generalization error) is independent of the norm. On one hand, this is also true for most deep neural network models that use ReLU activations in the intermediate layers and an argmax operation to produce the output label, and are therefore invariant to uniform scaling of all their weights and biases. On the other hand, this shows that the norm is only relevant

due to the choice of the loss, which is often only used as a continuous relaxation of the classification error. In light of this, the norm cannot affect the generalization capabilities of the network, and it thus seems unlikely that a norm-based regularization could be a valid general strategy.[1]

We have performed a replica-theory calculation for this model (for the details on the replica method see the previous chapter, Section 2.4), in which we have studied analytically the solutions found by optimizing the regularized MSE loss. In particular, we have explored the normalized local entropy landscape of these configurations (defined below) in the space of the training *error*. We stress that by using the error instead of the MSE we explore the properties of the model in the regime in which it is applied. Furthermore, we can freely renormalize all the configurations and simplify the analysis.

In this case the normalized local entropy around a given (normalized) configuration w measures the logarithm of the fraction of configurations whose training error is smaller or equal than that of the reference w in a volume within a given squared-distance d around w. More precisely, we compute:

$$\Phi_{\mathrm{LE}}(\lambda, d) = \lim_{N\to\infty} \mathbb{E}_{\mathbf{v}^\star}\mathbb{E}_{\sigma}\mathbb{E}_{\xi} \log \frac{\int_{S_N} dw'\ \Theta\left(\varepsilon\left(w^*\right) - \varepsilon\left(w'\right)\right)\Theta\left(d - \mathrm{d}\left(w', w^*\right)\right)}{\int_{S_N} dw'\ \Theta\left(d - \mathrm{d}\left(w', w^*\right)\right)} \tag{3.3}$$

where $w^\star$ is the normalized minimizer of the λ-regularized MSE loss, ε is the training error, and $\Theta(x)$ is the Heaviside step function $\Theta(x) = 1$ if $x \geq 0$ and 0 otherwise. The distance parameter d ranges in $[0, 2]$ and is essentially the Legendre transform of γ of eq. (3.41). In this definition we also used ε instead of a generic loss $\mathcal{L}$ and a hard cutoff $\varepsilon(w^*)$ instead of using β, to make the formula more explicit. The denominator is the volume of configurations within squared distance d, and the domain of integration S_N is the unit sphere in N dimensions. The expectations provide the average behavior on the whole distribution of the generative model, which with high probability is the same as

[1] There is a caveat to this statement: for particular choices of the loss, e.g. cross-entropy, it is possible to reparametrize the problem in an invariant way and interpret the norm in terms of a time-evolving parameter of the loss with a similar role to the focusing procedure discussed below, see e.g. [3].

the behavior of any one random instance for large N.

Due to the log and the normalization term, $\Phi_{\mathrm{LE}}(\lambda, d)$ is upper-bounded by zero and always zero at $d = 0$. For sharp minima, it is expected to drop rapidly with d, whereas for flat regions it's expected to stay very close to zero at least within some range. Some representative results are shown in fig. 1 (left panel), and they confirm that the configurations that generalize better (which for this model are those that have been obtained with the largest regularization parameter λ) have generally higher local entropy curves, i.e. they lie in the middle of fairly dense regions of good configurations, a.k.a. wide flat minima. Further in the chapter we provide the full derivation, and additional results that show that the same general scenario holds for different values of the parameters. We also show that a reasonable alternative choice for the cutoff could be used in the definition and lead to analogous conclusions.

These results thus confirm that the local entropy landscape constructed using the training error is a good predictor of generalization performance. However, when dealing with much more complex architectures, using the training error as the loss function in eq. (3.41) is not (yet) algorithmically feasible. In particular, the entropic algorithms rSGD and eSGD must still operate on a differentiable loss. This leaves the question whether targeting high-local-entropy regions in a differentiable loss landscape can still lead to good generalization results open. We have investigated this question analytically on the Gaussian mixture model with a linear classifier and the MSE loss, using the same technique explained in [3, 8]. This amounts to studying the generalization error of the barycenter of a replicated system of y classifiers, each with its own parameters w^a with $a = 1, \ldots, y$, each optimizing the MSE under constraints on their norms n and on their mutual angles θ, that is: $\forall a, a' : \ \|w^a\| = n, \ w^a \cdot w^{a'} = n^2 \cos(\theta)$. The barycenter is defined as $\bar{w} = \frac{1}{y} \sum_a w^a$. Due to the peculiarities of this model, we are interested in whether it is aligned with the solution of the norm-regularized model with large λ. In this analysis we used the angle θ rather than the distance in order to compare situations with different norms (if $n = 1$ then $\cos(\theta)$ is the same as $(1 - d)$ used previously). Our results indicate

that, with sufficiently many replicas (even just $y = 3$) and with sufficiently large angles the generalization performance is nearly optimal and the dependence on the norm is mild, and much less pronounced than at small angles (the limit of zero angles reproduces the results of the norm-regularized analysis without replicas). Some representative results are shown in fig. 1 (right panel) and are confirmed by numerical experiments. The fact that for this model the best results are obtained with widely separated replicas is due to the convex nature of the problem, and we do not expect this phenomenon to carry over to the non-convex landscapes of deep neural networks.

3.4 Replica theory analysis for Local Entropy and Replicated Systems

The analytical framework of Local Entropy was introduced in [6], while the connection between Local Entropy and systems of real replicas (as opposed to the "fake" replicas of spin glass theory [23]) was made in [7]. For completeness, we briefly recap here the derivation.

We start the computation from the definition of the local entropy loss given in Section 3.2:

$$\mathcal{L}_{\mathrm{LE}}(w) = -\frac{1}{\beta}\log\int dw'\; e^{-\beta\mathcal{L}(w')-\frac{1}{2}\beta\gamma\|w'-w\|^2}. \tag{3.4}$$

We then consider the Boltzmann distribution of a system with energy function $\beta\mathcal{L}_{\mathrm{LE}}(w)$ and with an inverse temperature y, that is

$$p(w) \propto e^{-\beta y\mathcal{L}_{\mathrm{LE}}(w)}, \tag{3.5}$$

where equivalence is up to a normalization factor. If we restrict y to integer values, we can then use the definition of $\mathcal{L}_{\mathrm{LE}}$ to construct an equivalent but enlarged system, containing $y + 1$ replicas. Their joint distribution $p(w, \{w^a\}_a)$ is readily obtained by plugging Eq. (3.4) into Eq. (3.5). We can then integrate out the original configuration w

and obtain the marginal distributional for the y remaining replicas

$$p(\{w^a\}_a) \propto e^{-\beta \mathcal{L}_{\mathrm{R}}\left(\{w^a\}_a\right)}, \tag{3.6}$$

where the energy function is now given by

$$\mathcal{L}_R(\{w^a\}_a) = \sum_{a=1}^{y} \mathcal{L}(w^a) + \frac{1}{2}\gamma \sum_{a=1}^{y} \|w^a - \bar{w}\|^2, \tag{3.7}$$

with $\bar{w} = \frac{1}{y}\sum_a w^a$. Thus, in this way we can recover the loss function for the replicated SGD (rSGD) algorithm.

3.5 Detailed analysis of the Gaussian Mixtures model

In this section we will provide details of the analytical computations performed on the Gaussian mixture model that we have presented in the previous section.

As mentioned before, at the first step an N-dimensional vector $\mathbf{v}^\star$ is randomly generated from a Gaussian centered at the origin and with covariance matrix equal to the identity matrix. Samples from two classes are then generated in the following way: First we generate a label $\sigma = 1$ or $\sigma = -1$ with probability ρ and $1-\rho$ respectively. Then, we generate a pattern $\boldsymbol{\xi}$ by using a Gaussian distribution centered in $\frac{\mathbf{v}^\star \sigma}{\sqrt{N}}$ and with covariance matrix proportional to the identity matrix; the proportionality constant (which we will call by Δ) controls the width of the two clusters. We generate P such points; the coordinate $i \in \{1, ..., N\}$ of point μ is therefore given by

$$\xi_i^\mu = \frac{v_i^\star}{\sqrt{N}}\sigma^\mu + \sqrt{\Delta}\, z_i^\mu \tag{3.8}$$

where z_i^μ are i.i.d Gaussian random variables with mean zero and unit variance. This results in two clusters, with the label indicating the cluster a pattern belongs to. In the following we will always limit ourselves to the symmetric case $\rho = \frac{1}{2}$ and unit noise $\Delta = 1$.

We consider the case of a linear classifier (i.e. a perceptron). Training this classifier corresponds to the minimization of the overall loss

$$\mathcal{L}(\boldsymbol{w}, b) = \sum_{\mu=1}^{P} \ell \left[\sigma_i^\mu \left(\frac{1}{\sqrt{N}} \sum_{i=1}^{N} w_i \xi_i^\mu + b \right) \right] \tag{3.9}$$

where $\boldsymbol{w}$ and b are respectively the weights and the bias of the network that need to be learned. $\ell(\cdot)$ is a generic loss function for a single pattern. We will consider in particular the case of the MSE loss $\ell(x) = \frac{1}{2}(x-1)^2$. As usual in statistical physics, we will consider the high-dimensional limit, where both $N \to \infty$ and $P \to \infty$ with the ratio $\alpha \equiv \frac{P}{N}$ fixed.

Recently, this model has been studied in [22] by using Gordon's inequality. They showed that the MSE loss is severely prone to overfitting, especially when $\alpha \simeq 1$. However, if a parameter λ for controlling the regularization of the weight norms is introduced, the generalization performance is improved. In the limit $\lambda \to \infty$ (corresponding to vanishing values for the norm of the weights), the generalization error of the network is equal to the Bayes-optimal one.

3.5.1 Typical case analysis

In this section we briefly review how the geometry of the space of typical Gibbs configurations of the model can be studied using statistical physics techniques [24, 25].

Denoting by β the inverse temperature, we define the partition function as

$$Z = \int \prod_i dw_i \, e^{-\beta \mathcal{L}(\boldsymbol{w}, b)} + \frac{\lambda}{2} \sum_i w_i^2 \tag{3.10}$$

We will denote the average over the distribution of patterns, labels and the centroids $\boldsymbol{v}^\star$ with $\langle \cdot \rangle$. The average of the log-volume $\langle \ln Z \rangle / N$ is the free entropy of the model $-\beta f$, where f is the free energy. We can evaluate it in the large-N limit by using the "replica

trick" presented in Section 2.4, i.e. the formula

$$\ln Z = \lim_{n\to 0} \partial_n Z^n \,.$$

We first compute the average for integer values of n and then we analytically continue n to 0. As usual, in replica computations one needs to introduce several order parameters in order to use the saddle point method when N is large. Indicating by a, $b \in \{1, \dots, n\}$ the replica indexes, those order parameters are:

- the overlap matrix between two weights $q^{ab} = \frac{1}{N}\sum_i w_i^a w_i^b$ for $a \neq b$
- the squared norm $Q^a = \frac{1}{N}\sum_i (w_i^a)^2$
- the overlap between a weight and the centroid $M^a = \frac{1}{N}\sum_i v_i^\star w_i^a$.

In order to enforce the previous three definitions via Dirac delta functions we need also the corresponding conjugated parameters, that we denote by $\hat{q}^{ab}$, $\hat{Q}^a$ and $\hat{M}^a$ respectively. The order parameters and the bias b satisfy saddle point equations that, once solved, permit to evaluate the free entropy of the model. Note that when $\rho = \frac{1}{2}$ the bias is always zero.

In the replica-symmetric ansatz we seek solutions to the saddle point equations of the form $q^{ab} = q$ for $a \neq b$, $Q^a = Q$, $M^a = M$ and similarly for conjugated order parameters. The final expression of the free entropy is given by

$$-\beta f = \mathcal{G}_S + \alpha \mathcal{G}_E \tag{3.11}$$

where we have defined the entropic and energetic terms as

$$\mathcal{G}_S = \frac{q\hat{q}}{2} - Q\hat{Q} - M\hat{M} + \frac{1}{2}\ln\left(\frac{2\pi}{\hat{q} - 2\hat{Q} + \lambda}\right) + \frac{\hat{q} + \hat{M}^2}{2\left(\hat{q} - 2\hat{Q} + \lambda\right)} \tag{3.12a}$$

$$\mathcal{G}_E = \mathbb{E}_\sigma \int Dy \ln \int Dh\, e^{-\beta\ell\left(\sqrt{\Delta(Q-q)}h + \sqrt{\Delta q}y + M + \sigma b\right)} \tag{3.12b}$$

and $Dy \equiv \frac{e^{-z^2/2}}{\sqrt{2\pi}}$ is the standard Gaussian measure. The train loss is found simply by taking the derivative $\epsilon_\ell = \frac{\partial(\beta f)}{\partial \beta}$.

When $\beta \to \infty$ the interesting regime is found when the regularization parameter λ is itself scaled with β as $\lambda \to \beta\lambda$. Moreover, if we consider a loss ℓ with a unique minimum (such as the MSE), the overlap q between two replicas must go to the squared norm Q. Therefore we must impose a scaling for q of the type

$$q = Q - \frac{\delta q}{\beta}\,. \tag{3.13}$$

Correspondingly, one can verify from the saddle point equations that the conjugated order parameters must be scaled as

$$\hat{q} = \beta^2 \delta\hat{Q} - \beta\delta\hat{q} \tag{3.14a}$$

$$\hat{Q} = \frac{\beta^2}{2}\delta\hat{Q} - \beta\delta\hat{q} \tag{3.14b}$$

$$\hat{M} = \beta\delta\hat{M} \tag{3.14c}$$

All the new order parameters introduced in those scalings must satisfy new saddle point equations obtained by taking derivatives of the free energy $-f = \mathfrak{G}_S + \alpha\mathfrak{G}_E$; the entropic and energetic terms (rescaled with β) are now given by

$$\mathfrak{G}_S \equiv \lim_{\beta\to\infty} \frac{\mathcal{G}_S}{\beta} = -\frac{\delta q \delta\hat{Q}}{2} + \frac{Q\delta\hat{q}}{2} - M\delta\hat{M} + \frac{\delta\hat{Q} + \delta\hat{M}^2}{2(\lambda + \delta\hat{q})} \tag{3.15a}$$

$$\mathfrak{G}_E \equiv \lim_{\beta\to\infty} \frac{\mathcal{G}_E}{\beta} = -\alpha\,\mathbb{E}_\sigma \int Dy\, A_\sigma(y) \tag{3.15b}$$

where $A_\sigma(y) \equiv \min_h \left[\frac{h^2}{2} + \ell\left(\sqrt{\Delta\delta q}h + \sqrt{\Delta Q}y + M + b\sigma\right)\right]$. Calling by $h^*_\sigma(y)$ the corresponding argmin, the training loss is

$$\epsilon_\ell = \alpha\,\mathbb{E}_\sigma \int Dy\, \ell\left(\sqrt{\Delta\delta q}h^*_\sigma(y) + \sqrt{\Delta Q}y + M + \sigma b\right) \tag{3.16}$$

The training error can be found by plugging $\ell(x) = \Theta(-x)$ inside equation (3.16), where $\Theta(x) = 1$ if $x \geq 0$ and 0 otherwise (i.e. the Heaviside step function). For the MSE loss, $h^*_\sigma(y)$ is easily found, so that the training error is

$$\epsilon_t = \alpha \, \mathbb{E}_\sigma H\left(\frac{\Delta\delta q + M + b\sigma}{\sqrt{\Delta Q}}\right), \tag{3.17}$$

where $H(x) = \int_x^\infty Dy = \frac{1}{2}\text{erfc}\left(\frac{x}{\sqrt{2}}\right)$. Also the MSE training loss can be easily found by explicitly performing the y integral in (3.16). One can verify that when λ is increased not only the corresponding squared norm Q lowers, but also, and more importantly, the training error/loss increases (even below the critical capacity $\alpha_c = 1$ of the model, where a zero training error solution can be found). This means that insisting in searching zero error solutions with the MSE loss is counterproductive and leads to overfitting. This is to be expected since the Gaussian mixture model is a particular case of general noisy teacher problems, in which the training set is no longer generated by a rule that can be inferred [26].

3.5.2 Local entropy around a given typical configuration: Franz-Parisi approach

In order to quantify the local geometrical landscape around a typical configuration $\tilde{\boldsymbol{w}}$ of the Gibbs measure with loss function $\mathcal{L}_r = \sum_\mu \ell_r$, regularization parameter λ_r and inverse temperature β_r, we have studied the so-called Franz-Parisi free entropy [27, 28]. It is defined as

$$-\beta f_{\text{FP}}(S) \equiv \frac{1}{N}\left\langle \frac{\int \prod_i d\tilde{w}_i \, e^{-\beta_r \mathcal{L}_r(\tilde{\boldsymbol{w}},\tilde{b}) + \lambda_r \sum_i \tilde{w}_i^2} \ln \mathcal{N}(\tilde{\boldsymbol{w}}, S)}{\int \prod_i d\tilde{w}_i \, e^{-\beta_r \mathcal{L}_r(\tilde{\boldsymbol{w}},\tilde{b}) + \lambda_r \sum_i \tilde{w}_i^2}} \right\rangle \tag{3.18}$$

where the quantity

$$\mathcal{N}(\tilde{\boldsymbol{w}}, S) \equiv \int d\mu_P(\boldsymbol{w}) \, e^{-\beta \mathcal{L}(\tilde{\boldsymbol{w}},\tilde{b})} \delta\left(\sum_i w_i \tilde{w}_i - NS\right) \tag{3.19}$$

is the volume of configurations $\boldsymbol{w}$ at inverse temperature β that have overlap S with the reference configuration $\tilde{\boldsymbol{w}}$. $d\mu_P(\boldsymbol{w})$ is the flat measure over the admissible values of $\boldsymbol{w}$ with a fixed squared norm P; in other terms the weights $\boldsymbol{w}$ live on the hyper-sphere $\frac{1}{N}\sum_i w_i^2 = P$. P is chosen to match the squared norm of the reference $\tilde{\boldsymbol{w}}$, that is $P = Q$. Note that Q is fixed via the soft constraint with the regularization parameter λ_r.

In order to compute the average over the disorder induced by the patterns, we use two replica tricks, one for the denominator of (3.18), which is just the partition function $\frac{1}{Z} = \lim_{r\to 0} Z^{r-1}$ and one for the log in the numerator of the same equation $\ln Z = \lim_{n\to 0} \partial_n Z^n$. From now on we will use indexes a or b for replicas in $\{1,\dots,r\}$ and c, $d \in \{1,\dots,n\}$. Therefore we get

$$-\beta f_{\mathrm{FP}}(S) = \frac{1}{N}\lim_{n\to 0}\lim_{r\to 0}\partial_n \left\langle \int \prod_{a,i} d\tilde{w}_i^a \prod_a e^{-\beta_r \mathcal{L}_r(\tilde{\boldsymbol{w}}^a,\tilde{b})+\lambda_r\sum_i (\tilde{w}_i^a)^2} \mathcal{N}^n(\tilde{\boldsymbol{w}}^{a=1}, S)\right\rangle \tag{3.20}$$

The computation proceeds as usual by averaging over the disorder and introducing some other order parameters (in addition to those involving only the reference $\tilde{\boldsymbol{w}}$), namely $p^{cd} = \frac{1}{N}\sum_i w_i^c w_i^d$, $t^{ac} = \frac{1}{N}\sum_i \tilde{w}_i^a w_i^c$, $O^c = \frac{1}{N}\sum_i v_i^\star w_i^c$, $P^c = \frac{1}{N}\sum_i (w_i^c)^2$ and the corresponding conjugated ones. Note that P^c is just the squared norm P because of the spherical constraint inside the measure $d\mu(\boldsymbol{w})$. We obtain that the Franz-Parisi free entropy can be split into the sum of an entropic and an energetic term as $\mathcal{F}_{\mathrm{FP}}(S) = \mathcal{G}_S + \alpha\mathcal{G}_E$. Using a RS ansatz on all the order parameters involved, the entropic term can be written as

$$\begin{aligned}\mathcal{G}_S = {} & \frac{p\hat{p}}{2} + t\hat{t} - P\hat{P} - O\hat{O} - S\hat{S} + \frac{1}{2}\ln\left(\frac{2\pi}{\hat{p}-2\hat{P}}\right) \\ & + \frac{1}{\hat{p}-2\hat{P}}\left[\frac{\hat{p}+\hat{O}^2}{2} + \frac{(\hat{S}-\hat{t})^2\left(2(\hat{q}-\hat{Q})+\hat{M}^2+\lambda_r\right)}{2(\hat{q}-2\hat{Q}+\lambda_r)^2} + \frac{(\hat{S}-\hat{t})(\hat{t}+\hat{M}\hat{O})}{\hat{q}-2\hat{Q}+\lambda_r}\right]\end{aligned} \tag{3.21}$$

whereas the energetic one is

$$\begin{aligned}\mathcal{G}_E = \mathbb{E}_\sigma \int Dx \, \frac{1}{\mathcal{Z}(x)} \int Dh \, e^{-\beta_r \ell_r \left(\sigma \tilde{b} + M + \sqrt{\Delta q} x + \sqrt{\Delta(Q-q)} h\right)} \\ \times \int Dy \ln \int Du \, e^{-\beta \ell \left[\sigma b + O + \sqrt{\Delta \gamma - \frac{\Delta (S-t)^2}{Q-q}} y + \frac{\Delta t}{\sqrt{\Delta q}} x + \frac{\Delta (S-t)}{\sqrt{\Delta (Q-q)}} h + \sqrt{\Delta (P-p)} u\right]}\end{aligned} \tag{3.22}$$

In the previous equation have defined $\gamma = p - \frac{t^2}{q}$ and

$$\mathcal{Z}(x) \equiv \int Dh \, e^{-\beta_r \ell_r \left(\sigma \tilde{b} + M + \sqrt{\Delta q} x + \sqrt{\Delta(Q-q)} h\right)} . \tag{3.23}$$

Note that the parameters involving only the reference $\tilde{\boldsymbol{w}}$ i.e. q, $\hat{q}$, $\hat{Q}$, M and $\hat{M}$ satisfy the same saddle point equations of the previous subsection. We are now interested in sending β_r to infinity. In order to do that, we need to add to the scalings of the order parameters involving only the reference (3.14), together with the ones for the overlaps between reference $\tilde{\boldsymbol{w}}$ and $\boldsymbol{w}$ and their conjugated ones. The new scalings are

$$t = S - \frac{\delta t}{\beta_r} \tag{3.24a}$$

$$\hat{t} = \beta_r \delta \hat{t} \tag{3.24b}$$

$$\hat{S} - \hat{t} = \delta \hat{S} . \tag{3.24c}$$

Using these scalings, the entropic term becomes

$$\begin{aligned}\mathcal{G}_S = \frac{p\hat{p}}{2} - \delta t \delta \hat{t} - P\hat{P} - O\hat{O} - S\delta\hat{S} + \frac{1}{2} \ln \left(\frac{2\pi}{\hat{p} - 2\hat{P}}\right) \\ + \frac{1}{\hat{p} - 2\hat{P}} \left[\frac{\hat{p} + \hat{O}^2}{2} + \frac{\delta \hat{S}^2 \left(\delta \hat{Q} + \delta \hat{M}^2\right)}{2(\delta \hat{q} + \lambda_r)^2} + \frac{\delta \hat{S} (\delta \hat{t} + \delta \hat{M} \hat{O})}{\delta \hat{q} + \lambda_r}\right]\end{aligned} \tag{3.25}$$

and the energetic term becomes

$$\mathcal{G}_E = \mathbb{E}_\sigma \int Dx Dy \ln \int Du \, e^{-\beta \ell \left[\sigma b + O + \sqrt{\Delta \gamma} y + \frac{\Delta S}{\sqrt{\Delta Q}} x + \frac{\Delta \delta t}{\sqrt{\Delta \delta q}} h_\sigma^*(x) + \sqrt{\Delta (P-p)} u\right]} \tag{3.26}$$

where we have redefined γ as $\gamma = p - \frac{S^2}{Q}$.

Once $f_{\rm FP}$ is known, we can compute the energy ϵ_ℓ of the configuration $\boldsymbol{w}$ with overlap S with the reference $\tilde{\boldsymbol{w}}$ as $\epsilon_\ell = \frac{\partial(\beta f_{\rm FP})}{\partial\beta}$ and the local entropy $\mathcal{S}$ as $\mathcal{S} = \beta(\epsilon_\ell - f_{\rm FP})$. The same formulas are valid if we look at the local entropy landscape in the space of the training error, where $\ell(x) = \Theta(-x)$. As in the previous subsection we indicate by ϵ_t the training error of the configuration $\boldsymbol{w}$ to distinguish it with respect to the training loss. The training error can be written as

$$\epsilon_t = \frac{\partial(\beta f_{\rm FP})}{\partial\beta} = \alpha e^{-\beta}\mathbb{E}_\sigma \int DxDy \frac{H\left(\frac{\sigma b + O + \sqrt{\Delta\gamma}y + \frac{\Delta S}{\sqrt{\Delta Q}}x + \frac{\Delta\delta t}{\sqrt{\Delta\delta q}}h^*_\sigma(x)}{\sqrt{\Delta(P-p)}}\right)}{H_\beta\left(-\frac{\sigma b + O + \sqrt{\Delta\gamma}y + \frac{\Delta S}{\sqrt{\Delta Q}}x + \frac{\Delta\delta t}{\sqrt{\Delta\delta q}}h^*_\sigma(x)}{\sqrt{\Delta(P-p)}}\right)} \tag{3.27}$$

where $H_\beta(x) \equiv e^{-\beta} + (1 - e^{-\beta})H(x)$.

At $\alpha = 0$ the Franz-Parisi free entropy is

$$-\beta f_{\rm FP}(S, \alpha = 0) = \frac{1}{2}\left[1 + \ln(2\pi) + \ln\left(\frac{1}{\lambda_r} - \lambda_r S^2\right)\right], \tag{3.28}$$

and gives the total volume of configurations at overlap S with the reference.

As stated in the previous sections, we are interested in studying the local entropy landscape of configurations $\tilde{\boldsymbol{w}}$ found by optimizing the regularized MSE loss in the space of the training error. Therefore we choose $\ell_r(x) = \frac{1}{2}(x-1)^2$ and $\ell(x) = \Theta(-x)$. On the other hand, the parameter β has been chosen in such a way that the training error of $\boldsymbol{w}$ given in (3.27) is equal to a certain cutoff $\bar{\epsilon}$.

Notice that (3.28) gives an upper bound to the local entropy. Therefore, if we normalize the local entropy with respect to (3.28) it will be either negative, or equal to zero for distances $d = 1 - \frac{S}{P}$ equal to zero. For sharp minima $\tilde{\boldsymbol{w}}$ we expect that the normalized local entropy will have a sharp drop near $d \simeq 0$, whereas for flat minima it will be close to zero for some range of distances.

We have studied two different values for the energy $\bar{\epsilon}$:

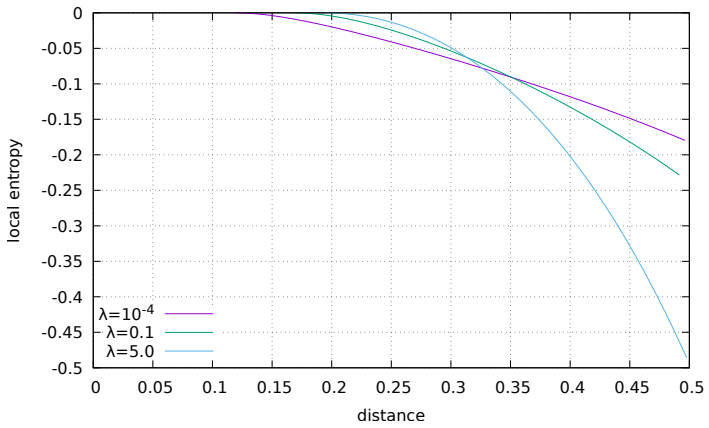

Figure 2: Normalized local entropy as a function of the squared-distance d computed from reference configurations found by optimizing the regularized MSE loss at $\alpha = 0.7$, with varying regularization strength λ. Here the cutoff $\bar{\epsilon}$ is given by the training error of the teacher $v^\star$. Larger values of λ correspond to minimizers with better generalization properties.

- in the first case $\bar{\epsilon}$ is chosen to be equal to the training error of the reference given in equation (3.17). This case corresponds the left panel of Fig. 1, where we plot the normalized local entropy as a function of the distance d.

- in the second case $\bar{\epsilon}$ is equal to the training error of the teacher $v^\star$, which is given by $\epsilon_t^T = \alpha H\left(\frac{1}{\sqrt{\Delta}}\right)$. This case is depicted in Fig.2.

In both cases we clearly see that references with better generalization properties (corresponding to larger values of the regularization parameter λ) have higher local entropy curves.

3.5.3 Replicated system in the loss landscape

We now study a system of y real replicas where each one optimizes a loss ℓ under constraints on their squared norm Q and on their mutual angles, namely: $\forall a, b: \ \frac{1}{N}\sum_i (w_i^a)^2 = Q, \ \frac{1}{N}\sum_i w_i^a w_i^b = Q^2\cos(\theta)$. This problem is equivalent to imposing a 1RSB ansatz on the standard equilibrium measure (3.10) with the Parisi parameter m and the intra-block overlap parameter q_1 fixed as external parameters; their physical meaning is identified respectively with the number of replicas y and the overlap between replicas $Q^2\cos(\theta)$ (see

also [29, 3]). Therefore the partition function of this system of y replicas is

$$Z_y = \int \prod_a d\mu_Q(\boldsymbol{w}^a)\, e^{-\beta \sum_a \mathcal{L}(\boldsymbol{w}^a, b)} \delta\left(q_1 - \frac{1}{N}\sum_i w_i^a w_i^b\right) \tag{3.29}$$

The free entropy of a single replica is $-\beta f = \frac{1}{Ny}\overline{\ln Z_y}$ and can be evaluated by the usual replica trick

$$-\beta f = \lim_{s\to 0} \frac{1}{Ny} \partial_s \overline{Z^s} \tag{3.30}$$

If we choose $s = n/y$, this formalism of the replicated partition function (3.29) reduces to the 1RSB formalism on the standard equilibrium measure given in (3.10), with the only difference being that y and q_1 are fixed as external parameters. The final result is again $-\beta f = \mathcal{G}_S + \alpha \mathcal{G}_E$, where

$$\begin{aligned}\mathcal{G}_S = {} & \frac{q_1\hat{q}_1}{2} - \frac{m}{2}(q_1\hat{q}_1 - q_0\hat{q}_0) - Q\hat{Q} - M\hat{M} + \frac{1}{2}\ln\left(\frac{2\pi}{\hat{q}_1 - 2\hat{Q} + \lambda}\right) \\ & + \frac{1}{2}\frac{\hat{q}_0 + \hat{M}^2}{\hat{q}_1 - 2\hat{Q} + \lambda - m(\hat{q}_1 - \hat{q}_0)} + \frac{1}{2m}\ln\left(\frac{\hat{q}_1 - 2\hat{Q} + \lambda}{\hat{q}_1 - 2\hat{Q} + \lambda - m(\hat{q}_1 - \hat{q}_0)}\right)\end{aligned} \tag{3.31}$$

and

$$\mathcal{G}_E = \frac{1}{m}\mathbb{E}_\sigma \int Dy \ln \int Dz \left[\int Dh\, e^{-\beta\ell\left(\sqrt{\Delta(Q-q_1)}h + \sqrt{\Delta q_0}y + \sqrt{\Delta(q_1-q_0)}z + M + \sigma b\right)}\right]^m \tag{3.32}$$

Computing the barycenter of the replicas

In this subsection we want to evaluate the relevant quantities in the barycenter of the replicas, which is defined as

$$\bar{w}_i \equiv \frac{1}{m}\sum_a w_i\,. \tag{3.33}$$

The relevant order parameters that we need to find in order to compute physical quantities are the overlap with the teacher $\bar{M} = \frac{1}{N}\sum_i \bar{w}_i v_i^\star$ and the norm of the center $\bar{Q} = \frac{1}{N}\sum_i \bar{w}_i^2$. We can see that $\tilde{M}$ and $\tilde{Q}$ can be expressed in terms of the known replica-overlap quan-

tities: $\tilde{M}$ is simply

$$\bar{M} = \frac{1}{Ny} \sum_i \sum_a w_i^a v_i^\star = \frac{1}{y} \sum_a M^a \tag{3.34}$$

whereas $\bar{Q}$ is

$$\bar{Q} = \frac{1}{N} \sum_i \bar{w}_i^2 = \frac{1}{y^2 N} \sum_{ab} \sum_i w_i^a w_i^b = \frac{1}{y^2} \sum_a Q^a + \frac{1}{y^2} \sum_{a \neq b} q^{ab} \tag{3.35}$$

Since all real replicas are have the same mutual overlap and the same squared norm, we get

$$\bar{M} = M \tag{3.36a}$$

$$\bar{Q} = \frac{Q - q_1}{m} + q_1 \tag{3.36b}$$

Large-β limit for the MSE loss

For the MSE loss all the integrals in the energetic term can be solved, giving

$$\begin{aligned} \mathcal{G}_E = &-\frac{1}{2} \ln\left(1 + \beta\Delta(Q - q_1)\right) + \frac{1}{2m} \ln\left(\frac{1 + \beta\Delta(Q - q_1)}{1 + \beta\Delta(Q - q_1) + m\beta\Delta(q_1 - q_0)}\right) \\ &- \frac{\beta}{2} \frac{\Delta q_0 + \mathbb{E}_\sigma (M + b\sigma - 1)^2}{1 + \beta\Delta(Q - q_1) + \beta m \Delta(q_1 - q_0)} \end{aligned} \tag{3.37}$$

When β is large we get the following scaling for q_0

$$q_0 = \frac{Q - q_1}{m} + q_1 + \frac{\delta q_0}{\beta}\,. \tag{3.38}$$

The other scalings are

$$\hat{q}_0 = \beta^2 \delta\hat{q}_0 + \frac{\beta}{2}\delta\hat{q}_1 \tag{3.39a}$$

$$\hat{q}_1 = \beta^2 \delta\hat{q}_0 - \frac{\beta}{2}\delta\hat{q}_1 \tag{3.39b}$$

$$\hat{Q} = \frac{\hat{q}_1}{2} - \frac{1}{2(Q-q_1)} \tag{3.39c}$$

$$\hat{M} = \beta\delta\hat{M} \tag{3.39d}$$

The new entropic and energetic terms (rescaled with β) are therefore

$$\mathfrak{G}_S \equiv \lim_{\beta\to\infty} \frac{\mathcal{G}_S}{\beta} = \frac{1}{2}(Q-q_1)\delta\hat{q}_1 + \frac{m}{2}\left[q_1\delta\hat{q}_1 + \delta q_0\delta\hat{q}_0\right] - M\delta\hat{M} + \frac{1}{2}\frac{\delta\hat{q}_0 + \delta\hat{M}^2}{m\delta\hat{q}_1} \tag{3.40a}$$

$$\mathfrak{G}_E \equiv \lim_{\beta\to\infty} \frac{\mathcal{G}_E}{\beta} = -\frac{1}{2}\frac{\Delta\left(\frac{Q-q_1}{m} + q_1\right) + \mathbb{E}_\sigma(M + b\sigma - 1)^2}{1 - m\Delta\delta q_0} \tag{3.40b}$$

Notice that in the large β limit the training error/loss of one of the replicas is not zero, because of distance constraint and the convexity of the loss landscape.

3.6 Flatness and local entropy estimates

Throughout this chapter, we argue that the local entropy function

$$\mathcal{L}_{\mathrm{LE}}(w) = -\frac{1}{\beta}\log\int dw'\; e^{-\beta\mathcal{L}(w') - \beta\gamma \mathrm{d}(w',w)}, \tag{3.41}$$

with $\mathrm{d}(w',w) = \frac{1}{2}\sum_{i=1}^{N}(w'_i - w_i)^2$, provides a good measure of generalization and can be effectively targeted by heuristic algorithms such as eSGD end rSGD.

While it is convenient to use Eq. (3.41) as an objective function and for the replica analysis of shallow networks, we consider a normalized version for the sake of providing an interpretable metric of the flatness of a minimizer.

In Section 3.6.1, we compute the normalized local entropy Φ_{LE} (introduced in Section 3.3) on a small neural network with one hidden layer using the Belief Propagation algo-

rithm. For more complex architectures, we have to resort to a cheap proxy of the local entropy also considered in [4]. Let $E_{\text{train}}(w)$ be the training error for a weight configuration $w \in \mathbb{R}^N$. Then we define $\delta E_{\text{train}}(w, \sigma)$ as the average training error difference with respect to $E_{\text{train}}(w)$ obtained perturbing w by a noise proportional to a parameter σ and w. Therefore we define the training error difference as

$$\delta E_{\text{train}}(w, \sigma) = \mathbb{E}_z \, E_{\text{train}}(w + \sigma z \odot w) - E_{\text{train}}(w), \tag{3.42}$$

where $\odot$ denotes element-wise product and the expectation is over normally distributed $z \sim \mathcal{N}(0, I_N)$.

In the cases where comparing the curves given by $\delta E_{\text{train}}(w, \sigma)$ and $\Phi_{\text{LE}}(w, d)$ was feasible, as in the case of the committee machine, we found a good qualitative agreement, as well as a good agreement with generalization.

For a given architecture and dataset, it remains to be assessed which value of σ gives the best indicator of good generalization, how to compare different architectures, and whether some meaningful scalar measure can be constructed from the whole flatness profile at all. Hence, we report the whole profiles instead of single estimates. A partial solution to some of these problems was given in [4], where the σ value corresponding to an a-priori arbitrarily chosen value of δE_{train} was used to construct a generalization measure. This generalization criterium proved to be one of the best performing among the many analyzed.

3.6.1 Local entropy on the committee machine

In this section we provide a study of the fully-connected committee machine. This is one of the simplest non-convex neural network models with continuous weights, and it is possible to compute the local entropy defined in Eq. (3.3) for it.

For this model, it has been shown in [3] that rare flat minima of the error loss function coexist with narrower ones. We follow the numerical setting of [3] and show that minima

found by entropy driven algorithms have higher local entropy. We also show that the local entropy measure is correlated with the generalization error and the cheaper flatness measure defined in Eq. 3.42. We use this cheaper flatness measure for deeper networks, where a direct local entropy estimate is unfeasible.

The fully-connected committee machine is a 2-layer fully connected neural network where only the first layer is trained, while the weights of the second and last layer are all fixed to +1. For a network with K hidden units, the output predicted for a given input pattern x reads:

$$\hat{\sigma}(w, x) = \text{sign}\left[\frac{1}{\sqrt{K}}\sum_{k=1}^{K}\text{sign}\left(\frac{1}{\sqrt{N}}\sum_{i=1}^{N} w_{ki}x_i\right)\right] \tag{3.43}$$

We train this network to perform binary classification on two classes of the Fashion-MNIST dataset with binarized patterns. In order to have a differentiable objective, we approximate sign activation functions on the hidden layer with $\tanh(\beta * x)$ functions, where the β parameter increases during the training. We normalize the weights during training by rescaling after each iteration, factoring out an overall scale parameter γ^{CE} that we insert explicitly in the binary cross-entropy loss:

$$\mathcal{L}(w) = \mathbb{E}_{x,\sigma\sim D}\ f(\sigma \cdot \hat{\sigma}(w, x), \gamma^{\text{CE}}) \tag{3.44}$$

Here, we have defined $f(x, \gamma^{\text{CE}}) = -\frac{x}{2} + \frac{1}{2\gamma^{\text{CE}}}\log(2\cosh(\gamma^{\text{CE}}x))$ as in [3]. The γ^{CE} parameter is increased gradually in the SGD training process in order to control the growth rate of the weight norms. Notice that the weight norms are also controlled by the parameter β, as it multiplies the normalized output of each hidden unit.

As shown in [3], slowing down the norm growth rate results in better generalization performance and increased flatness of the minima found at the end of the training. To appreciate this effect we used two different parameters settings for optimizing the loss in Eq.(3.44) with SGD, that we name "SGD slow" and "SGD fast". In the fast setting both β and γ^{CE} start with a large value and grow quickly, while in the slow setting they start from small values and grow more slowly, requiring more epochs to converge.

We compare test error, local entropy and training error differences for solutions found by rSGD and eSGD (see Alg.2 and 3), with those found by SGD. As the flatness of the solution found by such algorithms depends on the hyperparameters, we also report the results for rSGD in two different settings, that we call again "fast" and "slow", where the difference is in a faster or slower increase of the γ parameter, which controls the distance between replicas.

The normalized local entropy (3.3) around a given solution can be computed using the Belief Propagation (BP) algorithm (we refer to [3] for analytic and algorithmic details). For each solution we also computed δE_{train} as explained in the previous section to see if it provides the same information as the local entropy.

The results are shown in Fig. 3. In the left panel, we report Φ_{LE} computed with BP around the solutions found by the different algorithms, as a function of the distance from the solution. Even if controlling the norm with the parameters β and γ^{CE} improves the flatness of the solution found, entropy driven algorithms are biased towards flatter minima. This is confirmed in the central panel where we plot δE_{train} for the same solutions. From this experiment it can be seen that the training error difference curve δE_{train} preserves the same ordering of the different solutions as the one resulting from Φ_{LE}, thus apparently providing a valid alternative way to quantitatively estimate the sharpness of a given minimum. Finally, in the right panel the distribution of the generalization error for different solutions clearly shows a high correlation between the generalization performance of a given algorithm and the local entropy of the minima it finds.

In what follows, we describe the details of the numerical experiments. We define a reduced version of the Fashion-MNIST dataset following [3]: we choose the classes Dress and Coat as they are non-trivial to discriminate but also different enough so that a small network as the one we used can generalize. The network is trained on a small subset of the available examples (500 patterns) binarized to ± 1 by using the median of each image as a threshold on the inputs; we also filter both the training and test sets to use only images in which the median is between 0.25 and 0.75. For the test set we used all the

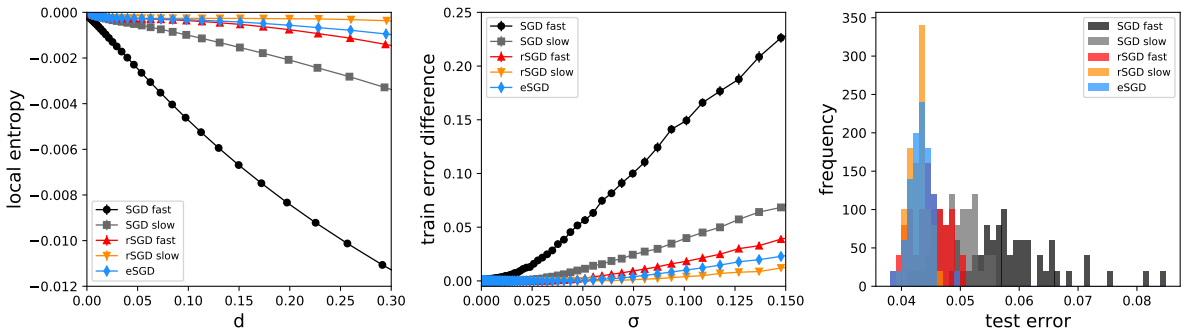

Figure 3: Normalized local entropy Φ_{LE} as a function of the squared distance d (left), training error difference $\delta E_{\mathrm{train}}$ as a function of perturbation intensity σ (center) and test error distribution (right) for a committee machine trained with various algorithms on the reduced version of the Fashion-MNIST dataset. Results are obtained using 50 random restarts for each algorithm.

patterns of the chosen classes in the original test set that passed our median filtering.

The network has $N = 784$ and $K = 9$ hidden units, and it is trained using minibatches of 100 patterns. All the results are averaged over 50 independent restarts. For each algorithm we initialize the weights with a uniform distribution and then normalize the weights of the hidden units norm before the training starts and after each weight update. The β and γ^{CE} parameters are updated using exponential schedules, $\beta(t) = \beta_0 \left(1 + \beta_1\right)^t$ and $\gamma^{\mathrm{CE}}(t) = \gamma_0^{\mathrm{CE}} \left(1 + \gamma_1^{\mathrm{CE}}\right)^t$, where t is the current epoch. An analogous exponential schedule is used for the elastic interaction γ for rSGD and eSGD, as described in the main text. In the SGD fast case, we stop as soon as a solution with zero errors is found, while for SGD slow we stop when the cross entropy loss reaches a value lower than 10^{-7}. For eSGD and rSGD, we stop the training when the distance between the reference weights and the one used to estimate the local entropy is smaller than 10^{-8}.

We used the following hyper-parameters for the various algorithms:

SGD fast : $\eta = 2 \cdot 10^{-4}$, $\beta_0 = 2.0$, $\beta_1 = 10^{-4}$, $\gamma_0^{\mathrm{CE}} = 5.0$, $\gamma_1^{\mathrm{CE}} = 0.0$;

SGD slow : $\eta = 3 \cdot 10^{-5}$, $\beta_0 = 0.5$, $\beta_1 = 10^{-3}$, $\gamma_0^{\mathrm{CE}} = 0.5$, $\gamma_1^{\mathrm{CE}} = 10^{-3}$;

rSGD fast : $\eta = 10^{-4}$, $y = 10$, $\gamma_0 = 2 \cdot 10^{-3}$, $\gamma_1 = 2 \cdot 10^{-3}$, $\beta_0 = 1.0$, $\beta_1 = 2 \cdot 10^{-4}$, $\gamma_0^{\mathrm{CE}} = 0.5$, $\gamma_1^{\mathrm{CE}} = 10^{-3}$;

rSGD slow : $\eta = 10^{-3}$, $y = 10$, $\gamma_0 = 10^{-4}$, $\gamma_1 = 10^{-4}$, $\beta_0 = 1.0$, $\beta_1 = 2 \cdot 10^{-4}$, $\gamma_0^{\mathrm{CE}} = 0.5$, $\gamma_1^{\mathrm{CE}} = 10^{-3}$;

eSGD : $\eta = 10^{-3}$, $\eta' = 5 \cdot 10^{-3}$, $\epsilon = 10^{-6}$, $L = 20$, $\gamma_0 = 10.0$, $\gamma_1 = 5 \cdot 10^{-5}$, $\beta_0 = 1.0$, $\beta_1 = 10^{-4}$, $\gamma_0^{\mathrm{CE}} = 0.5$, $\gamma_1^{\mathrm{CE}} = 5 \cdot 10^{-4}$;

3.7 Numerical experiments on deep networks

3.7.1 Entropic algorithms

For our numerical experiments on deep network we have used two entropic algorithms, rSGD and eSGD, mentioned in the introduction. They both approximately optimize the local entropy $\mathcal{L}_{\mathrm{LE}}$ as defined in eq. (3.41), for which an exact evaluation of the integral is intractable. The two algorithms employ different but related approximation strategies, as detailed below. Our aim is to explore the characteristics of these algorithms on difficult datasets and state-of-the-art networks, comparing their performance with each other and with standard SGD. We also investigate the relationship between their generalization properties and the flatness of the minima that they produce.

Entropy-SGD. Entropy-SGD (eSGD), introduced in [9], minimizes the local entropy (3.41) by approximate evaluations of its gradient. The gradient can be expressed as

$$\nabla \mathcal{L}_{\mathrm{LE}}(w) = \gamma \left(w - \langle w' \rangle \right) \tag{3.45}$$

where $\langle \cdot \rangle$ denotes the expectation over the measure $Z^{-1} e^{-\beta \mathcal{L}(w') - \beta \gamma \mathrm{d}(w', w)}$, where Z is a normalization factor. The eSGD strategy is to approximate $\langle w' \rangle$ (which implicitly depends on w) using L steps of stochastic gradient Langevin dynamics (SGLD). The

resulting double-loop algorithm is presented as Algorithm 2. The noise parameter ϵ in the algorithm is linked to the inverse temperature by the usual Langevin relation $\epsilon = \sqrt{2/\beta}$. In practice we always set it to the small value $\epsilon = 10^{-4}$ as in [9]. For $\epsilon = 0$, eSGD approximately computes a proximal operator [30].

Replicated-SGD. Replicated-SGD (rSGD) consists in a replicated version of the usual stochastic gradient (SGD) method. In rSGD, a number y of replicas of the same system, each with its own parameters w_a where $a = 1, ..., y$, are trained in parallel for K of iterations, interacting through an attractive term with their center of mass. As detailed in [7, 3], the replicated system trained with a stochastic algorithm such as SGD collectively explores an approximation of the local entropy landscape, and the replication bypasses the need to explicitly estimate the integral in eq. (3.41). In principle, the larger y the better the approximation, but already with $y = 3$ the effect of the replication is significant. To summarize, rSGD replaces the local entropy (3.41) with the replicated loss $\mathcal{L}_R$:

$$\mathcal{L}_R(\{w^a\}_a) = \sum_{a=1}^{y} \mathcal{L}(w^a) + \gamma \sum_{a=1}^{y} \mathrm{d}\left(w^a, \bar{w}\right) \tag{3.46}$$

Here, $\bar{w}$ is a center replica defined as $\bar{w} = \frac{1}{y}\sum_{a=1}^{y} w^a$. The algorithm is presented as Algorithm 3. Any of the replicas or the center $\bar{w}$ can be used after training as the resulting model for inference. This procedure is parallelizable over the replicas, so that wall-clock time for training is comparable to SGD, excluding the communication which happens every K parallel optimization steps. In order to decouple the communication period and the coupling hyperparameter γ, we let the coupling strength take the value $K\gamma$. In our experiments, we did not observe any degradation in generalization performance with of K up to 10.

Focusing. A common feature of both algorithms is that the parameter γ in the objective $\mathcal{L}_{\mathrm{LE}}$ changes during the optimization process. We start with a small γ (targeting large regions and allowing a wider exploration of the landscape) and gradually increase it. We

Algorithm 2 Entropy-SGD (eSGD)

Input: w
HyperParams: $L, \eta, \gamma, \eta', \epsilon, \alpha$
for $t = 1, 2, \ldots$ **do**
  $w', \mu \leftarrow w$
  for $l = 1, \ldots, L$ **do**
    $\Xi \leftarrow$ sample minibatch
    $dw' \leftarrow \nabla \mathcal{L}\left(w'; \Xi\right) + \gamma\left(w' - w\right)$
    $w' \leftarrow w' - \eta' dw' + \sqrt{\eta'}\, \epsilon\, \mathcal{N}(0, I)$
    $\mu \leftarrow \alpha\mu + (1 - \alpha)\, w'$
  end for
  $w \leftarrow w - \eta\left(w - \mu\right)$
end for

Algorithm 3 Replicated-SGD (rSGD)

Input: $\{w^a\}$
HyperParams: y, η, γ, K
for $t = 1, 2, \ldots$ **do**
  $\bar{w} \leftarrow \frac{1}{y} \sum_{a=1}^{y} w^a$
  for $a = 1, \ldots, y$ **do**
    $\Xi \leftarrow$ sample minibatch
    $dw^a \leftarrow \nabla \mathcal{L}\left(w^a; \Xi\right)$
    if $t = 0 \bmod K$ **then**
      $dw^a \leftarrow dw^a + K\gamma\left(w^a - \bar{w}\right)$
    end if
    $w^a \leftarrow w^a - \eta\, dw^a$
  end for
end for

call this process *focusing*. Focusing improves the dynamics by driving the system quickly to wide regions and then, once there, gradually trading off the width in order to get to the minima of the loss within those regions, see [31, 7]. We adopt an exponential schedule for γ, where its value at epoch τ is given by $\gamma_\tau = \gamma_0(1 + \gamma_1)^\tau$. For rSGD, we fix γ_0 by balancing the distance and the data term in the objective before training starts, i.e. we set $\gamma_0 = \sum_a \mathcal{L}(w^a) / \sum_a \mathrm{d}(w^a, \bar{w})$ for rSGD. The parameter γ_1 is chosen such that γ increases by a factor 10^4. For eSGD, we were unable to find a criterion that worked for all experiments and manually tuned it.

Optimizers. Vanilla SGD updates in Algorithms 2 and 3 can be replaced by optimization steps of any commonly used gradient-based optimizers.

3.7.2 Comparisons across several architectures and datasets

In this section we show that, by optimizing the local entropy with eSGD and rSGD, we are able to systematically improve the generalization performance compared to standard SGD. We perform experiments on image classification tasks, using common benchmark datasets, state-of-the-art deep architectures and the usual cross-entropy loss. The detailed settings of the experiments are reported in 3.A. For the experiments with eSGD and

rSGD, we use the same settings and hyper-parameters (architecture, dropout, learning rate schedule,...) as for the baseline, unless otherwise stated in the Appendix and apart from the hyper-parameters specific to these algorithms.

While we do some little hyper-parameter exploration to obtain a reasonable baseline, we do not aim to reproduce the best achievable results with these networks, since we are only interested in comparing different algorithms in similar contexts. For instance, we train PyramidNet+ShakeDrop for 300 epochs, instead of the 1800 used in [32], and we start from random initial conditions for EfficientNet instead of doing transfer learning as done in [33]. In the case of the ResNet110 architecture instead, we use the training specification of the original paper [34].

All combinations of datasets and architectures we tested are reported in Table 3.1. Blanks correspond to untested combinations. The first 3 columns correspond to experiments with the same number of effective epochs, that is considering that in each iteration of the outer loop in Algorithms 2 and 3 we sample L and y mini-batches respectively. In the last column instead, each replica consumes individually the same amount of data as the baseline. Being a distributable algorithm, rSGD enjoys the same scalability of the related EASGD and Parle [16, 10].

For rSGD, we use $y = 3$ replicas and the scoping schedules described in Section 3.7.1. In our explorations, rSGD proved to be quite robust with respect to specific choices of the hyper-parameters. The error reported is that of the center $\bar{w}$. For eSGD, we set $L = 5$, $\epsilon = 1e - 4$ and $\alpha = 0.75$ in all experiments, and we perform little tuning for the the other hyper-parameters. The algorithm is a little more sensitive to hyper-parameters than rSGD, while still being quite robust. Moreover, it misses an automatic γ scoping schedule.

Results in Table 3.1 and Fig. 4 show that entropic algorithm generally outperform the corresponding baseline with roughly the same amount of parameter tuning and computational resources. In the next section we also show that they land in flatter minima.

Dataset	Model	Baseline	rSGD	eSGD	rSGD$\times y$
CIFAR-10	SmallConvNet	16.5 ± 0.2	15.6 ± 0.3	14.7 ± 0.3	14.9 ± 0.2
	ResNet-18 [34]	13.1 ± 0.3	12.4 ± 0.3	12.1 ± 0.3	11.8 ± 0.1
	ResNet-110 [34]	6.4 ± 0.1	6.2 ± 0.2	6.2 ± 0.1	5.3 ± 0.1
	PyramidNet+ShakeDrop [35, 36]	2.0			1.8
CIFAR-100	PyramidNet+ShakeDrop [35, 36]	13.9	13.5		12.7
	EfficientNet-B0 [33]	20.5	20.6		19.5
Tiny ImageNet	ResNet-50 [34]	45.2 ± 1.2	41.5 ± 0.3	41.7 ± 1	39.2 ± 0.3
	DenseNet-121 [37]	41.4 ± 0.3	39.8 ± 0.2	38.6 ± 0.4	38.9 ± 0.3

Table 3.1: Test set error (%) for vanilla SGD (baseline), eSGD and rSGD. The first three columns show results obtained with the same number of passes over the training data. In the last column instead, each replica in the parallelizable rSGD algorithm consumes the same amount of data as the baseline.

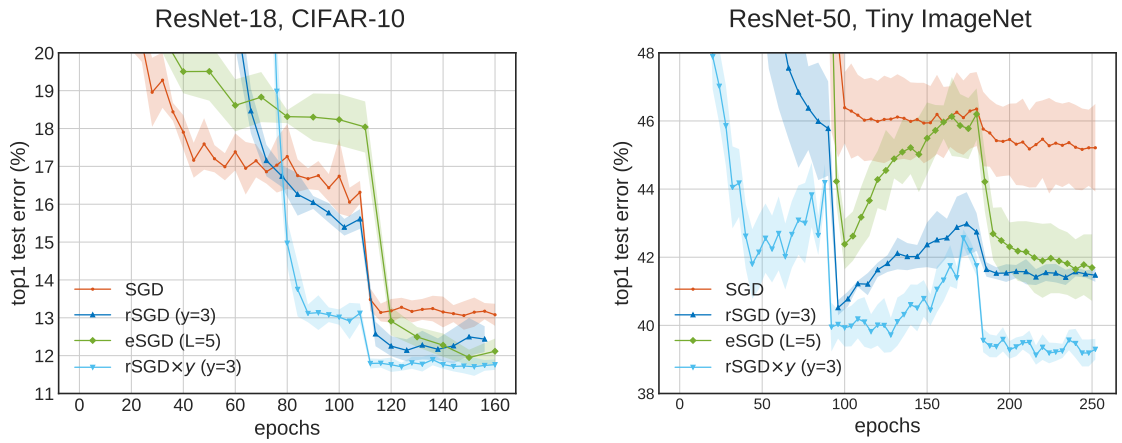

Figure 4: Left: Test error of ResNet-18 on CIFAR-10. Right: Test error of ResNet-50 on Tiny ImageNet. The curves are averaged over 5 runs. Training data consumed is the same for SGD, rSGD and eSGD. Epochs are rescaled by y for rSGD and by L for eSGD (they are not rescaled for rSGD$\times y$).

3.7.3 Flatness curves for deep networks

In this section we present flatness curves, $\delta E_{\text{train}}(w, \sigma)$ from Eq. (3.42), for some of the deep networks architecture examined in this chapter.

Results are reported in Figures 5 and 6 for different architectures and datasets. The expectation in Eq. (3.42) is computed over the complete training set using 400 and 100 realizations of the Gaussian noise for each data point in Figures 5 and 6 respectively. In experiments where data augmentation was used during training, it is also used when computing the flatness curve.

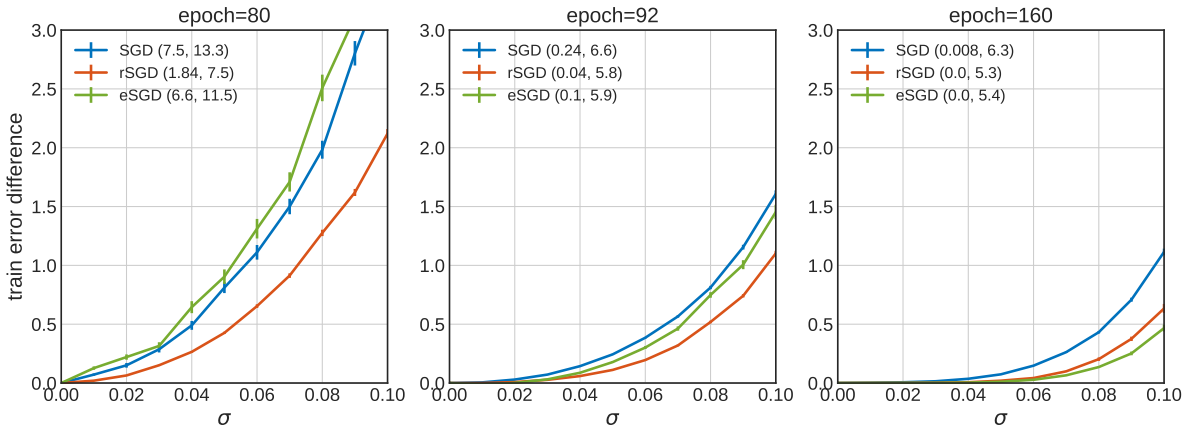

Figure 5: Train error difference δE_{train} from eq. (3.42) for ResNet-110 on Cifar-10. Values are computed along the training dynamics of different algorithms and as a function of the perturbation intensity σ. Unperturbed train and test errors (%) are reported in the legends.

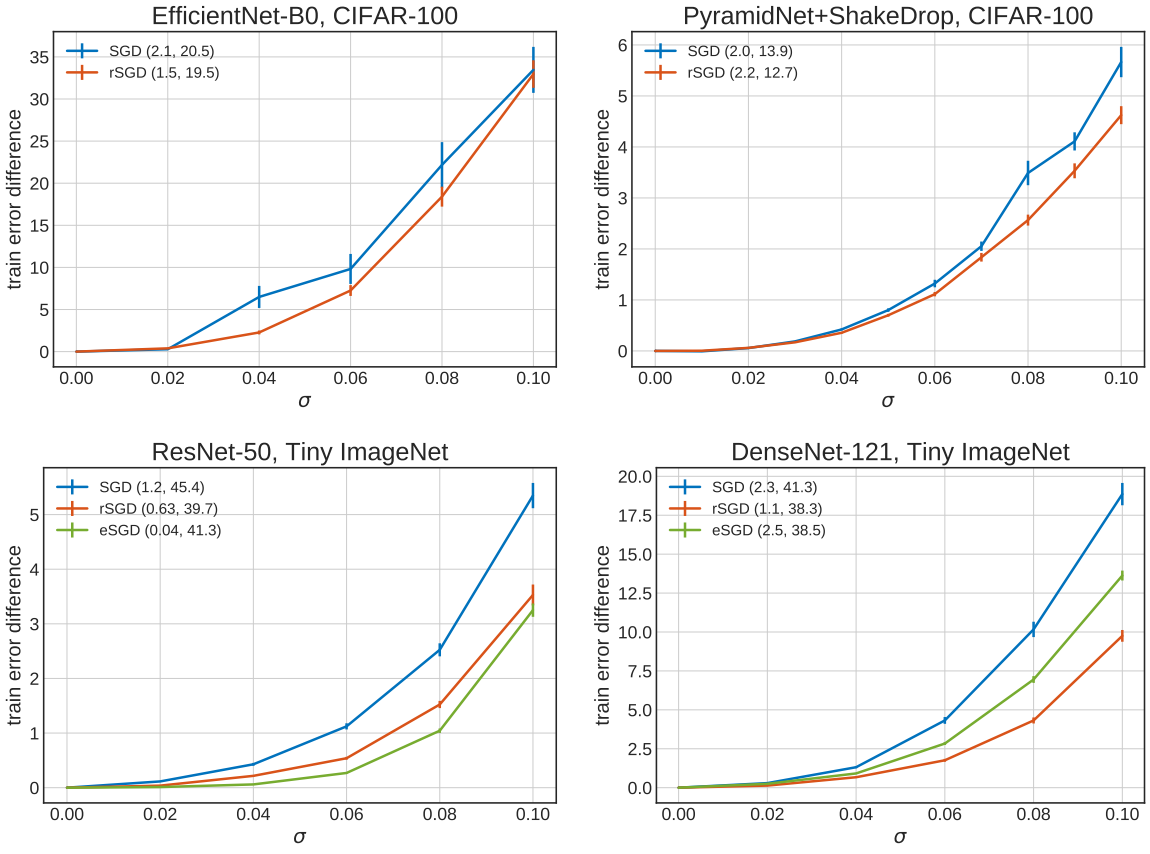

Figure 6: Train error difference δE_{train} from eq. (3.42), for minina obtained on various architectures, datasets and with different algorithms, as a function of the perturbation intensity σ. Unperturbed train and test errors (%) are reported in the legends.

The comparison is performed between minima found by different algorithms, at a point where the training error is near zero and the loss has reached a stationary value. If the comparison is made for minima that show different train errors, the correlation between test error and flatness is not clearly observed.

When there is a large residual training error instead, correlation of test and training error overshadows the correlation of test error with flatness. We report a generally good agreement between the flatness of the δE_{train} curve and the generalization performance, for a large range of σ values.

3.8 Discussion and conclusions

We studied analytically and numerically the connection between generalization and flatness, as defined by the local entropy measure for the classification error loss function and for its differentiable relaxations. Starting with analytically tractable models, we have discussed new results for Gaussian mixtures classification, which show that optimal Bayesian predictors correspond to high local entropy regions of the classifier and of the differentiable loss. These optimal solutions can be found algorithmically both by adding a strong ℓ_2 regularization to the learning loss function or by an entropic algorithm (rSGD). Observing that the classifier itself is independent of the norm of the weights and that flatness can be properly defined on any loss, our results give further support to the idea that the flatness of minima plays an important role for generalization. A similar scenario is known to exist in DNNs with ReLU activations and argmax operations for the output labels, which are invariant to weights rescaling. We have performed an extensive numerical study on state of the art deep architectures to verify that the improvement in performance is correlated with estimates of flatness. Our future efforts will be devoted to study the connection between generalization bounds and the existence of wide flat regions in the landscape of the classifier.

Appendix

3.A Deep networks experimental details

In this section we describe in more detail the experiments reported in the Table 3.1 of the main text. In all experiments, the loss $\mathcal{L}$ is the usual cross-entropy and the parameter initialization is Kaiming normal. We normalize images in the train and test sets by the mean and variance over the train set. We also apply random crops (of width w if image size is $w \times w$, with zero-padding of size 4 for CIFAR and 8 for Tiny ImageNet) and random horizontal flips. In the following we refer to the latter procedure by the name "standard preprocessing". All experiments are implemented using PyTorch [38].

For the experiments with eSGD and rSGD, we use the same settings and hyper-parameters used for SGD (unless otherwise stated and apart from the hyperparameters specific to these two algorithms). For rSGD and unless otherwise stated, we set $y = 3$, $K = 10$ and use the automatic exponential focusing schedule for γ reported in the main text.

For eSGD, we use again an exponential focusing protocol. In some experiments, we use a value of γ_0 automatically chosen by computing the distance between the configurations w' and w after a loop of the SGLD dynamics (i.e. in the first L steps with $\gamma = 0$) and setting $\gamma_0 = \mathcal{L}(w)/d(w', w)$. Unfortunately, this criterion is not robust. Therefore, in some experiments the value of γ_0 was manually tuned. However, we found that eSGD is not sensitive to the precise value but to the order of magnitude.

We choose γ_1 such that γ is increased by a factor of 10 by the end of the training. Unless otherwise stated, we set the number of SGLD iterations to $L = 5$, SGLD noise to $\epsilon = 10^{-4}$ and $\alpha = 0.75$. Moreover, we use 0.9 Nesterov momentum and weight decay in both the internal and external loop. As for the learning rate schedule, when we rescale the total number of epochs for eSGD and rSGD, we use a rescaled schedule giving a comparable final learning rate and with consequently rescaled learning rate drop times

as well.

3.A.1 CIFAR-10 and CIFAR-100

SmallConvNet The smallest architecture we use in our experiments is a LeNet-like network [39]:

$$Conv(5\times5, 20) - MaxPool(2) - Conv(5\times5, 50) - MaxPool(2) - Dense(500) - Softmax$$

Each convolutional layer and the dense layer before the final output layer are followed by ReLU non-linearities.

We train the SmallConvNet model on CIFAR-10 for 300 epochs with the following settings: SGD optimizer with Nesterov momentum 0.9; learning rate 0.01 that decays by a factor of 10 at epochs 150 and 225; batch-size 128; weight decay 1e-4; standard preprocessing is applied; default parameter initialization (PyTorch 1.3). For rSGD we set $lr = 0.05$ and $\gamma_0 = 0.001$. For eSGD, we train for 60 epochs with: $\eta = 0.5$ that drops by a factor of 10 at epochs 30 and 45; $\eta' = 0.02$; $\gamma_0 = 0.5$; $\gamma_1 = 2 \cdot 10^{-5}$.

ResNet-18 In order to have a fast baseline network, we adopt a simple training procedure for ResNet-18 on CIFAR-10, without further optimizations. We train the model for 160 epochs with: SGD optimizer with Nesterov momentum 0.9; initial learning rate 0.01 that decays by a factor of 10 at epoch 110; batch-size 128; weight decay 5e-4; standard preprocessing.

For rSGD we set $K = 1$ and learning rate 0.02. For eSGD, we train for 32 epochs with initial learning rate $\eta = 0.25$ that drops by a factor of 10 at epochs 16 and 25; $\eta' = 0.01$. In the case in which we drop the learning rate at certain epochs, we notice that it is important not to schedule it before that the training error has reached a plateau also for eSGD and rSGD.

ResNet-110 We train the ResNet-110 model on CIFAR-10 for 164 epochs following the original settings of [34]: SGD optimizer with momentum 0.9; batch-size 128; weight decay 1e-4. We perform a learning rate warm-up starting with 0.01 and increasing it at 0.1 after 1 epoch; then it is dropped by a factor of 10 at epochs 82 and 124; standard preprocessing is applied.

For both eSGD and rSGD, we find that the learning rate warm-up is not necessary. For rSGD we set $\gamma_0 = 5\text{e-}4$. For eSGD, we train for 32 epochs with initial learning rate $\eta = 0.9$ that drops at epochs 17 and 25, SGLD learning rate $\eta' = 0.02$ and we set $\gamma_0 = 0.1$ and $\gamma_1 = 5 \cdot 10^{-4}$.

PyramidNet+ShakeDrop PyramidNet+ShakeDrop [35, 36], together with AutoAugment or Fast-AutoAugment, is currently the state-of-the-art on CIFAR-10 and CIFAR-100 without extra training data. We train this model on CIFAR-10 and CIFAR-100 following the settings of [32, 40]: PyramidNet272-α200; SGD optimizer with Nesterov momentum 0.9; batch-size 64; weight decay 5e-5. At variance with [32, 40] we train for 300 epochs and not 1800. We perform a cosine annealing of the learning rate (with a single annealing cycle) with initial learning rate 0.05. ShakeDrop is applied with the same parameters as in the original paper [36]. For data augmentation we add to standard preprocessing AutoAugment with the policies found on CIFAR-10 [32] (for both CIFAR-10 and CIFAR-100) and CutOut [41] with size 16.

For rSGD, we use a cosine focusing protocol for γ, defined at epoch τ by $\gamma_\tau = 0.5\gamma_{\max} \cos(\pi\tau/\tau_{\text{tot}})$, with $\gamma_{\max} = 0.1$. On CIFAR-10, we decrease the interaction step K from 10 to 3 towards the end of the training (at epoch 220) in order to reduce noise and allow the replicas to collapse.

EfficientNet-B0 EfficientNet-B0 is the base model for the EfficientNet family. In this section we train EfficientNet-B0 on CIFAR-10, starting from random initial conditions. We follow the same settings as [33], with some differences: we train for 350 epochs with RMSprop optimizer with momentum 0.9; batch-size 64; weight decay 1e-5; initial learning

rate 0.01 that decays by 0.97 every 2 epochs. We rescale image size to 224×224 and as data augmentation we apply standard preprocessing (with zero-padding of size 32) adding AutoAugment with the policies found on CIFAR-10 [32] (for both CIFAR-10 and CIFAR-100). For rSGD we set $\gamma_0 = 5e - 6$.

3.A.2 Tiny ImageNet

ResNet-50 Entropic algorithms are effective also on more complex datasets. We train ResNet-50 on Tiny ImageNet (data downloaded from: "Tiny ImageNet Visual Recognition Challenge") for 270 epochs with: SGD optimizer with Nesterov momentum 0.9; initial learning rate 0.05 that decays by a factor of 10 at epochs 90, 180 and 240; batch-size 128; weight decay 1e-4. Standard preprocessing is applied together with Fast-AutoAugment with the policies found on ImageNet [40].

For eSGD we train the model for 50 epochs with $\eta = 0.8$ that drops by a factor of 10 at epochs 18, 36, 48 and $\eta' = 0.02$.

DenseNet-121 For DenseNet-121 on Tiny ImageNet, the setting is the same as ResNet-50, except that we train the model for 200 epochs with learning rate drops at epochs 100 and 150.

For eSGD we train the model for 40 epochs with $\eta = 0.5$ that drops by a factor of 10 at epochs 25 and 30, $\eta' = 0.02$ and we set $\gamma_0 = 1.0$ and $\gamma_1 = 2 \cdot 10^{-5}$

References

[1] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

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

[3] Carlo Baldassi, Fabrizio Pittorino, and Riccardo Zecchina. Shaping the learning landscape in neural networks around wide flat minima. *Proceedings of the National Academy of Sciences*, 117(1):161–170, 2020.

[4] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them, 2019.

[5] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. *34th International Conference on Machine Learning, ICML 2017*, 3:1705–1714, 2017.

[6] Carlo Baldassi, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Subdominant dense clusters allow for simple learning and high computational performance in neural networks with discrete synapses. *Phys. Rev. Lett.*, 115:128101, Sep 2015.

[7] Carlo Baldassi, Christian Borgs, Jennifer T. Chayes, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Unreasonable effectiveness of learning neural networks: From accessible states and robust ensembles to basic algorithmic schemes. *Proceedings of the National Academy of Sciences*, 113(48):E7655–E7662, 2016.

[8] Carlo Baldassi, Enrico M. Malatesta, and Riccardo Zecchina. Properties of the geometry of solutions and capacity of multilayer neural networks with rectified linear unit activations. *Phys. Rev. Lett.*, 123:170602, Oct 2019.

[9] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

[10] Pratik Chaudhari, Carlo Baldassi, Riccardo Zecchina, Stefano Soatto, and Ameet Talwalkar. Parle: parallelizing stochastic gradient descent. *CoRR*, abs/1707.00424, 2017.

[11] Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, page 5–13, New York, NY, USA, 1993. Association for Computing Machinery.

[12] Wray L Buntine and Andreas S Weigend. Bayesian back-propagation. *Complex systems*, 5(6):603–643, 1991.

[13] Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P. Adams, and Peter Orbanz. Non-vacuous generalization bounds at the imagenet scale: A pac-bayesian compression approach, 2018.

[14] Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data, 2017.

[15] Gintare Karolina Dziugaite and Daniel M. Roy. Entropy-SGD optimizes the prior of a PAC-bayes bound: Data-dependent PAC-bayes priors via differential privacy, 2018.

[16] Sixin Zhang, Anna Choromanska, and Yann LeCun. Deep learning with elastic averaging sgd, 2014.

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

[18] Xiaoyi Mai and Zhenyu Liao. High dimensional classification via empirical risk minimization: Improvements and optimality. *arXiv preprint arXiv:1905.13742*, 2019.

[19] Marc Lelarge and Leo Miolane. Asymptotic bayes risk for gaussian mixture in a semi-supervised setting. *arXiv preprint arXiv:1907.03792*, 2019.

[20] Zeyu Deng, Abla Kammoun, and Christos Thrampoulidis. A model of double descent for high-dimensional binary linear classification. *arXiv preprint arXiv:1911.05822*, 2019.

[21] Thibault Lesieur, Caterina De Bacco, Jess Banks, Florent Krzakala, Cris Moore, and Lenka Zdeborová. Phase transitions and optimal algorithms in high-dimensional gaussian mixture clustering. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 601–608. IEEE, 2016.

[22] Francesca Mignacco, Florent Krzakala, Yue M Lu, and Lenka Zdeborová. The role of regularization in classification of high-dimensional noisy gaussian mixture. *arXiv preprint arXiv:2002.11544*, 2020.

[23] Marc Mézard, Giorgio Parisi, and Miguel Virasoro. *Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications*, volume 9. World Scientific Publishing Company, 1987.

[24] E Gardner. The space of interactions in neural network models. *Journal of Physics A: Mathematical and General*, 21(1):257–270, jan 1988.

[25] E Gardner and B Derrida. Optimal storage properties of neural network models. *Journal of Physics A: Mathematical and General*, 21(1):271–284, jan 1988.

[26] Andreas Engel and Christian Van den Broeck. *Statistical mechanics of learning*. Cambridge University Press, 2001.

[27] Silvio Franz and Giorgio Parisi. Recipes for metastable states in spin glasses. *Journal de Physique I*, 5(11):1401–1415, 1995.

[28] Haiping Huang and Yoshiyuki Kabashima. Origin of the computational hardness for learning with binary synapses. *Physical Review E*, 90(5):052813, 2014.

[29] Rémi Monasson. Structural glass transition and the entropy of the metastable states. *Physical review letters*, 75(15):2847, 1995.

[30] Pratik Chaudhari, Adam Oberman, Stanley Osher, Stefano Soatto, and Guillaume Carlier. Deep relaxation: partial differential equations for optimizing deep neural networks. *Research in the Mathematical Sciences*, 5(3):30, 2018.

[31] Carlo Baldassi, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Local entropy as a measure for sampling solutions in constraint satisfaction problems. *Journal of Statistical Mechanics: Theory and Experiment*, 2016(2):P023301, February 2016.

[32] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data. *CoRR*, abs/1805.09501, 2018.

[33] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2019.

[34] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[35] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. *CoRR*, abs/1610.02915, 2016.

[36] Yoshihiro Yamada, Masakazu Iwamura, and Koichi Kise. Shakedrop regularization. *CoRR*, abs/1802.02375, 2018.

[37] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[39] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[40] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *CoRR*, abs/1905.00397, 2019.

[41] Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017.

Chapter 4

Sampling through Algorithmic Diffusion in Non-Convex Perceptron Problems

In the previous chapter, we have reduced the problem of understanding generalization to that of describing the local loss landscape. In the current chapter, we continue with a more precise study of the loss landscape in a simple model of shallow neural networks known as the *perceptron* problem. We investigate under which conditions it is possible to *sample* random configurations from the perceptron loss landscape as we vary a smoothing temperature parameter.

This is an instance of the more general problem of sampling from a known but intractable distribution, which we will solve using a denoising diffusion process (see e.g. [1, 2]), whose score function will be provided by an Approximate Message Passing algorithm [3]. Moreover, we introduce a theoretical formalism based on the replica method that can be applied to a generic ensemble of problems and allows us to characterize the sampling process and its limitations in the infinite-dimensional limit. For the *spherical* perceptron problem with negative stability, we show that approximate uniform sampling is achievable across nearly the entire replica symmetric region of the phase

diagram. In contrast, for the perceptron problem with *binary* weights, sampling from the uniform distribution is intractable due to the overlap gap property exhibited by the typical set of solutions. A theoretical analysis of this obstruction leads us to identify a different potential under which our diffusion process successfully samples. Finally, we show numerically that an annealing procedure over the shape of this potential also yields a fast and robust Markov Chain Monte Carlo algorithm for sampling from the solution space of the binary perceptron.

4.1 Introduction

Diffusion models are a class of generative models that has emerged as a powerful framework for modeling complex data distributions. Rooted in statistical mechanics, these techniques conceptualize the generation of data as a gradual process of reverse diffusion, where noise is incrementally transformed into structured data. Initially introduced within the context of denoising and inpainting (e.g. [4]), diffusion models have found applications in image and video generation, audio synthesis and other various high-dimensional data sampling tasks. This introductory chapter explores the evolution of diffusion models, highlights key contributions, and provides a focused examination of their role in sampling.

The concept of diffusion models is primarily inspired by the study of stochastic processes. The general idea is to transform a data distribution through a forward diffusion process, which progressively corrupts data (e.g., an image) by adding Gaussian noise at each timestep, while in parallel, a reverse diffusion process is learned, which aims to denoise the noisy data step-by-step, effectively recovering the original data distribution.

One of the earliest works in this line of research was by [4], who introduced the idea of denoising score matching, laying the groundwork for the development of score-based generative models. This work demonstrated the potential of models that iteratively reverse the noise process. A significant breakthrough in the broader context of generative models was achieved in [1], who proposed the Diffusion Probabilistic Model (DPM). While

the forward process of the diffusion is standard and represented by an iterative addition of the Gaussian noise, the reverse process at every time step of the update is modeled by a Gaussian or a binomial Markov transition which is learned so that it maximizes the likelihood lower bound.

Following this, the approach of training neural networks to learn the reverse diffusion process was popularized in [2] with the introduction of the Denoising Diffusion Probabilistic Model (DDPM). DDPM utilized a U-Net architecture ([5]) to learn the reverse process and demonstrated state-of-the-art results in image generation tasks (measured by such quality metrics such as Fréchet Inception Distance).

In this chapter we study a problem of sampling, considering settings in which the target probability density is known up to an intractable normalization factor:

$$p(\boldsymbol{w}) = \frac{\psi(\boldsymbol{w})}{Z}, \tag{4.1}$$

as is often the case in Bayesian inference or in the large-sample regime. Typical sampling algorithms in this setting belong to the Markov Chain Monte Carlo (MCMC) family [6]. These can suffer from slow mixing times, and their theoretical analysis is generally challenging. In contrast, for generative diffusion, the continuous stochastic process can be well approximated by a small number of discrete steps, and the Bayesian structure induced by the noising/denoising process facilitates its theoretical analysis. Recent efforts have explored hybrid approaches combining generative diffusion with MCMC methods (see [7, 8, 9, 10]).

A fundamental question is whether the score function can be approximated at any time t of the process by a polynomial-time algorithm with access to the unnormalized density $\psi(\boldsymbol{w})$, enabling efficient sampling from $p(\boldsymbol{w})$ via generative diffusion.

In this chapter, we develop a theoretical framework to address this question precisely in the high-dimensional limit, considering random instances of the distribution p itself (i.e., assuming quenched disorder, in statistical physics terminology). Our framework

relies on the non-rigorous but extensively validated replica method from spin glass theory [11, 12]. The sampling algorithm associated with the replica analysis (Stochastic Localization or SL) employs Approximate Message Passing (AMP) [3, 13] as the score approximator within the diffusion process. AMP is conjectured to be optimal among polynomial-time algorithms for the denoising task involved in computing the score function.

We first analyze SL-based sampling for spherical weights, then extend our investigation to the more challenging problem of binary perceptron solutions. We find that efficient sampling is feasible in a large region of the hyperparameters' space of the spherical model. In the binary case instead, uniform sampling from the solution space is always unfeasible. This result was expected since the typical solutions are isolated and therefore hard to find (see [14, 15]). Nevertheless, this limitation motivates the exploration of other statistical measures that allow for efficient sampling.

In the following sections we introduce the formalism for analyzing the stochastic localization process as well as discuss its properties in detail.

4.2 Stochastic Localization for Sampling

In this section, we introduce the key components of the sampling algorithm based on the SL process. Given a probability density $p(\boldsymbol{w})$, with $\boldsymbol{w} \in \mathbb{R}^N$, referred to as the *target distribution*, our goal is to generate samples from p. We assume p to be known, possibly up to a hard-to-compute normalization factor, and write it as $p(\boldsymbol{w}) = \psi(\boldsymbol{w})/Z$, where we call *partition function* the normalization $Z = \int \mathrm{d}\boldsymbol{w}\, \psi(\boldsymbol{w})$ and we call $\psi(\boldsymbol{w})$ *unnormalized density*.

Given the target density p, SL can be defined as a stochastic differential equation (SDE) that goes from time $t = 0$ to $t = +\infty$ for a vector $\mathbf{h}_t \in \mathbb{R}^N$, which we call the time-dependent *field*. The SL's SDE is the analogous of the reverse process SDE in

denoising diffusion [16]. The initial condition is $\mathbf{h}_0 = \mathbf{0}$, and for $t \geq 0$ the SDE reads:

$$\mathrm{d}\mathbf{h}_t = \mathbf{m}_t(\mathbf{h}_t)\,\mathrm{d}t + \mathrm{d}\mathbf{b}_t\,, \tag{4.2}$$

where $(\mathbf{b}_t)_{t\geq 0}$ is the standard Wiener process in N dimensions. The drift term $\mathbf{m}_t(\mathbf{h}_t)$ is computed as the expectation

$$\mathbf{m}_t(\mathbf{h}) = \mathbb{E}_{\boldsymbol{w}\sim p_{\mathbf{h},t}}\left[\boldsymbol{w}\right]\,, \tag{4.3}$$

over what we call the *tilted distribution* $p_{\mathbf{h},t}(\boldsymbol{w})$, obtained by convolving the target distribution with Gaussian noise:

$$p_{\mathbf{h},t}(\boldsymbol{w}) = \frac{\psi(\boldsymbol{w})\, e^{\langle \mathbf{h}, \boldsymbol{w}\rangle - \frac{t}{2}\|\boldsymbol{w}\|^2}}{Z_{\mathbf{h},t}}\,. \tag{4.4}$$

The key feature of the SL process is that as $t \to +\infty$ the field diverges and $p_t \coloneqq p_{\mathbf{h}_t,t}$ peaks around a single configuration $\boldsymbol{w}^*$ that is statistically distributed (over the realizations of the process) as a sample from the target distribution p, that is we have

$$p_t \xrightarrow{t\to+\infty} \delta_{\boldsymbol{w}^*}, \quad \boldsymbol{w}^* \sim p. \tag{4.5}$$

Therefore, a sample from the target distribution p can be obtained as the value of $\mathbf{m}_t(\mathbf{h}_t)$ at large times.

4.2.1 Bayesian interpretation

It can be shown [16] that at any time $t \geq 0$, the solution $\mathbf{h}_t$ of (4.2) has the same distribution as

$$\mathbf{h}_t = t\boldsymbol{w}^* + \sqrt{t}\mathbf{g}. \tag{4.6}$$

where $\boldsymbol{w}^* \sim p$ and $\mathbf{g} \sim \mathcal{N}(\mathbf{0}, I_N)$ are sampled independently. This is similar to the forward process of denoising diffusion. The distribution $p_{\mathbf{h}_t,t}$ can then be interpreted as the posterior over $\boldsymbol{w}^*$ given the noisy observation $\mathbf{h}_t$. The function $\mathbf{m}_t(\mathbf{h}_t)$ corresponds to the Bayesian denoiser.

4.3 Asymptotic Analysis of Algorithmic Stochastic Localization with the Replica Formalism

We devise a formalism to investigate the asymptotic behavior of the algorithmic SL sampling process in the large system size limit $N \to +\infty$. The formalism relies on the non-rigorous but well-established [11] replica method. The main outcome will be a criterion for the feasibility of fair sampling involving the evaluation of a time-dependent free entropy. What follows is a generalization of the scheme in [17] that allows for handling unnormalized densities, as the ones we will deal with in the next sections.

Given the target distribution p, which we assume to be stochastic and drawn from an ensemble of distributions (quenched disorder), the solution $\mathbf{h}_t$ of the SDE (4.2) (assuming the drift term is correctly estimated) is distributed as $\mathbf{h}_t = t\boldsymbol{w}^* + \sqrt{t}\mathbf{g}$, where $\boldsymbol{w}^*$ is a sample from p—referred to as the *reference sample*—and $\mathbf{g}$ is a standard Gaussian noise. A relevant parameter for tracking the dynamics is the overlap

$$q(t) = \frac{1}{N}\langle \boldsymbol{w}^*, \boldsymbol{w}_t \rangle. \tag{4.7}$$

We argue that this quantity, while fluctuating over the realization of p and of the SDE path, concentrates for large N to a deterministic quantity. This quantity emerges naturally as an order parameter in the replica computation that we now describe.

As usual in statistical physics, the free entropy of the system is a central object of investigation, since it provides access to several quantities of interest. In our case, for a given time t, and considering the presence of disorder, we define the asymptotic average

free entropy (or just free entropy) as

$$\phi_t = \lim_{N\to+\infty} \frac{1}{N} \mathbb{E} \log Z_{\mathbf{h}_t,t}, \tag{4.8}$$

where the expectation is computed over the realization of the target unnormalized density ψ over the reference sample $\boldsymbol{w}^*$ and over the Gaussian noise $\mathbf{g}$ of (4.6). In order to handle the usually intractable expectation of a logarithm and the sampling from an unnormalized distribution, we employ the replica method twice. We introduce s replicas of the target distribution and n replicas for the tilted distribution and obtain[1]

$$\phi_t = \lim_{N\to+\infty} \frac{1}{N} \lim_{s\to 0} \lim_{n\to 0} \partial_n \, \mathbb{E}_{\psi,\boldsymbol{g}} \int \prod_{\alpha=1}^{s} \psi\left(\mathrm{d}\boldsymbol{w}_\alpha^*\right) \prod_{a=1}^{n} \psi\left(\mathrm{d}\boldsymbol{w}_a\right) e^{\langle \mathbf{h}_t, \boldsymbol{w}_a\rangle - \frac{t}{2}\|\boldsymbol{w}_a\|^2}, \tag{4.9}$$

where $\mathbf{h}_t$ is computed from $\boldsymbol{w}_1^*$, the first of the s replicas associated to the target distribution: $\mathbf{h}_t = t\boldsymbol{w}_1^* + \sqrt{t}\mathbf{g}$. The structure of this computation is closely related to the one presented in the seminal work of [18]. Notice that the n replicas $\boldsymbol{w}_a$ have symmetric roles, while in the set of s replicas $\boldsymbol{w}_\alpha^*$, the $\alpha = 1$ one has a special role. Carrying out the computation, in dense systems as the ones we will consider in this chapter, all the possible pairwise overlaps among the $n+s$ replicas emerge naturally as order parameters subject to saddle point optimization. We will then assume the Replica Symmetric (RS) ansatz [11, 19], that is, we will restrict the saddle point evaluation to the most symmetric overlap structure under replica exchanges compatible with the symmetries of (4.9). Moreover, thanks to the Bayesian structure of the problem, one can apply the Nishimori conditions [20, 21, 17] to further simplify the overlap structure, thanks to additional symmetries, as demonstrated in Section 4.5.1. Under these hypotheses, the overlap $q(t)$ of (4.7) can be computed from the overlaps of replicas involved in (4.9) as

$$q(t) = \frac{1}{N}\langle \boldsymbol{w}_1^*, \boldsymbol{w}_a\rangle = \frac{1}{N}\langle \boldsymbol{w}_a, \boldsymbol{w}_b\rangle, \qquad \forall a, b \in [n] \text{ and } b \neq a. \tag{4.10}$$

[1] We use $\log Z = \lim_{n\to 0} \partial_n Z^n$ and $Z^{-1} = \lim_{s\to 0} Z^{s-1}$.

At time $t = 0$, all the $n + s$ replicas become equivalent to each other, therefore $q(t = 0) = r$, where r is the overlap between two distinct samples of the target distribution: $r = \langle \boldsymbol{w}^*_\alpha, \boldsymbol{w}^*_\beta \rangle / N$ for $\alpha, \beta \in [s]$ and $\alpha \neq \beta$. We further mention that while the Nishimori conditions found for the planted distributions analyzed in [17] guarantee the correctness of the RS ansatz, in our more general setting they guarantee only that the analysis of the tilted system doesn't involve further replica symmetry breaking (RSB) compared to the standard replica analysis of the reference systems. Therefore, our RS analysis is exact only in the RS region of p.

In general, the replica computation will arrive at an expression where the free entropy ϕ_t is given by the saddle point of a function of several order parameters, one of which is q:

$$\phi_t = \max_q \ \phi_t(q), \qquad \phi_t(q) := \operatorname*{extr}_\theta \ \phi_t(q, \theta), \tag{4.11}$$

where we denoted with θ all the remaining order parameters except q. The value $q_{\max}$ that maximizes $\phi_t(q)$ represents the typical value of $q(t)$.

4.3.1 Success and Failure of Algorithmic SL

As shown in [17], the study of $\phi_t(q)$ can reveal whether the algorithmic SL (ASL) sampling scheme can recover samples from the target distribution, in the large N limit. More specifically, the success of ASL hinges on the unimodality of $\phi_t(q)$ as a function of q. If it has a single maximum at all times t, moving smoothly from low q to $q = 1$ (assuming $\|\boldsymbol{w}\|^2 = N$), then the AMP messages correctly recover (with high probability in the limit of large N) the value $q(t) = \arg\max_q \phi_t(q)$ and the algorithm successfully samples from the target distribution. Conversely, if $\phi_t(q)$ becomes multimodal, it could be the case that AMP doesn't return the correct estimate of $\mathbf{m}_t(\mathbf{h}_t)$ and the SDE integration fails. The mechanism is as follows: at $t = 0$, there is always a maximum located at low q, which is the one found by AMP. As t increases, this maximum will in general move toward higher

q, and AMP will follow it; however, if at any t there is a second, higher maximum and at higher q, it should be the correct one that solves the saddle point equations, but AMP will miss it. As we show below, this can happen both if the global maximum exists from the outset or if it develops over time. See Fig. 1 (Left) for a setting in which a high q maximum first appears and then becomes the global one.

4.4 Applications on Perceptron Models

4.4.1 Definitions

The perceptron [22] is the simplest neural network model, used for binary classification tasks. Instead of the usual optimization perspective, we adopt a constraint satisfaction one [23], and define a family of probability distributions over the solution space. The problem is defined by a dataset X, containing M examples $\boldsymbol{x}^\mu \in \mathbb{R}^N$, and a scalar κ that we call margin. Without loss of generality we can assume that all labels are equal to 1. A given *weight configuration* $\boldsymbol{w} \in \mathbb{R}^N$ is called a *solution* if the corresponding stabilities, defined as $s^\mu = \frac{\langle \mathbf{x}^\mu, \boldsymbol{w} \rangle}{\sqrt{N}}$, satisfy $s^\mu \geq \kappa \ \forall \mu$, meaning that all examples are correctly classified. We define a family of distributions over the solution space by the unnormalized density (from (4.1)):

$$\psi(\boldsymbol{w}) = P(\boldsymbol{w}) \prod_{\mu=1}^{M} \Theta\left(s^\mu - \kappa\right) e^{-\frac{1}{T} U(s_\mu - \kappa)}, \quad \text{with} \quad s^\mu = \frac{\langle \boldsymbol{x}^\mu, \boldsymbol{w} \rangle}{\sqrt{N}} \tag{4.12}$$

Here $\Theta(s)$ is the Heaviside step function, $\Theta(s) = 1$ if $s > 0$ and 0 otherwise. The parameter $T \geq 0$ is called temperature, and we call *potential* the function $U(s)$. Notice that for $U(s) = 0$ (or any constant) or for $T \to +\infty$, the distribution $p(\boldsymbol{w}) = \psi(\boldsymbol{w})/Z$ becomes the uniform distribution over the solution space. The distribution $P(\boldsymbol{w})$ is a prior on the weights, possibly unnormalized. In this chapter, we consider two different priors, corresponding to the spherical weights perceptron, $P(\boldsymbol{w}) = \delta(\|\boldsymbol{w}\|^2 - N)$, and to the binary weights perceptron, $P(\boldsymbol{w}) = \prod_i P(w_i)$ and $P(w_i) = \delta(w_i - 1) + \delta(w_i + 1)$.

The perceptron problems are generated by considering i.i.d. examples $\boldsymbol{x}^\mu \sim \mathcal{N}(0, I_N)$ or $\boldsymbol{x}^\mu \sim \text{Unif}(\{-1, +1\}^N)$. The two settings are equivalent for our asymptotic analysis. The high-dimensional limit is obtained for $N \to +\infty$ and $M \to +\infty$ with fixed finite load $\alpha = M/N$. This statistical setting has been largely investigated in the statistical physics literature, see [23] and [24] for broad reviews.

4.4.2 Implementation of ASL

In order to adapt the ASL sampling scheme, as discussed in Section 4.2, to the target distribution (4.12), we have to implement the corresponding AMP algorithm. Since (4.12) can be seen as a specific type of generalized linear model, we can adapt the GAMP algorithm from [25] for our purposes. The message passing scheme is detailed in Algorithm 4. The AMP framework relies on the definition of two key functions (ϕ_{in} and ϕ_{out}), commonly referred to as the input- and output-channel free entropies, given by:

$$\phi_{\text{in}}(A, B) = \log \int P(\mathrm{d}w)\ e^{wB - \frac{1}{2}w^2 A}, \tag{4.13}$$

$$\phi_{\text{out}}(\omega, V) = \log \int \mathrm{d}s\ \Theta(s - \kappa)\, \frac{e^{-\frac{1}{T}U(s-\kappa) - \frac{(s-\omega)^2}{2V}}}{\sqrt{2\pi V}} = \log \tilde{H}_V(-\omega), \tag{4.14}$$

where $\tilde{H}_a(b) = \int Dz\ \Theta(-b - \kappa + \sqrt{a}z) e^{-\frac{1}{T}U(-b-\kappa+\sqrt{a}z)}$ and $Dz = \frac{dz}{\sqrt{2\pi}} e^{-z^2/2}$. For the case $U(s) = 0$, $\tilde{H}$ simplifies to $\tilde{H}_a(b) = H(-\frac{b+\kappa}{\sqrt{a}})$, with $H(x) = \frac{1}{2}\text{erfc}\left(\frac{x}{\sqrt{2}}\right)$.

In the binary weights case $\int P(\mathrm{d}w) = \sum_{w=\pm 1}$. For the spherical case, the $\|\boldsymbol{w}\|^2 = N$ norm constraint has to be relaxed to a factorized Gaussian prior, $P(w) = e^{-\frac{1}{2}\gamma w^2}$, with γ tuned adaptively during the AMP iterations. The relaxed prior is equivalent to the hard one in the limit of large N, but AMP can handle only factorized priors. As a consequence, the integral in ϕ_{in} can always be computed in closed form. Furthermore, for some choices of the potential $U(s)$, notably $U(s) = 0$ and $U(s) = -\log(s)$, the integration in ϕ_{out} can also be carried out analytically. Therefore, the AMP used in the settings discussed in the main text is quite fast since it does not contain any integrals.

In the case of spherical perceptron, the algorithm enforces (in expectation) the norm

constraint $\|\boldsymbol{w}\|^2 = N$ by assuming a unnormalized prior $P(w) = e^{-\frac{1}{2}\gamma w^2}$, and then solving for γ the equation

$$-2\sum_{i=1}^{N} \partial_\gamma \, \phi_{\text{in}}(A_i^k + t, B_i^k + h_{t,i}) = N. \tag{4.15}$$

in each AMP iteration using a root-finding algorithm. Notice that γ appears implicitely in ϕ_{in} defined in (4.13). We stress that the AMP equations for the binary perceptron problem, that is studied in the following sections, do not essentially differ from the algorithm presented above, except in the way the ϕ_{in} is defined and consequently they do not require the dynamic reinforcement of the weights norm.

Algorithm 4 AMP for ASL on the Perceptron

Input: data $\boldsymbol{X} \in \mathbb{R}^{M\times N}$, ASL time $t \geq 0$, ASL field $\mathbf{h}_t \in \mathbb{R}^N$, stopping criterion $\epsilon > 0$, max iterations $K > 0$.

Initialize: $\mathbf{m}^0 \coloneqq \bar{0}_N$, $\boldsymbol{\Delta}^0 \coloneqq 0_N$, $\boldsymbol{g}^0_{\text{out}} \coloneqq 0_M$, $\boldsymbol{A} \coloneqq 0_N$, $\boldsymbol{B} \coloneqq 0_N$, $k = 0$

repeat

$k \leftarrow k + 1$

Updating mean and variance estimates ω_μ, V_μ

$$V_\mu^k \leftarrow \sum_i X_{\mu i}^2 \Delta_i^{k-1} \tag{4.16}$$

$$\omega_\mu^k \leftarrow \sum_i X_{\mu i} m_i^{k-1} - V_\mu^k g_\mu^{k-1} \tag{4.17}$$

Updating estimates A_i, B_i, g_μ

$$g_\mu^k \leftarrow \partial_\omega \phi_{\text{out}}(\omega_\mu^k, V_\mu^k) \tag{4.18}$$

$$A_i^k \leftarrow -\sum_\mu X_{\mu i}^2 \partial_\omega^2 \phi_{\text{out}}(\omega_\mu^k, V_\mu^k) \tag{4.19}$$

$$B_i^k \leftarrow \sum_\mu X_{\mu i}\, g_\mu^k + m_i^{k-1} A_i^k \tag{4.20}$$

Only for spherical case: enforce norm constraint solving (4.15) for γ.

Updating marginals m_i and Δ_i

$$m_i^k \leftarrow \partial_{B_i} \phi_{\text{in}}\left(A_i^k + t, B_i^k + h_{t,i}\right) \tag{4.21}$$

$$\Delta_i^k \leftarrow \partial^2_{B_i} \phi_{\text{in}}\left(A_i^k + t, B_i^k + h_{t,i}\right) \tag{4.22}$$

$\Delta_{\text{iter}} \leftarrow \|\mathbf{m}^k - \mathbf{m}^{k-1}\|^2/N$

until $k = K$ or $\Delta_{\text{iter}} < \epsilon$

return $\mathbf{m}^k$

Algorithm 5
Sampling a solution to the perceptron problem

Input: Data: $\boldsymbol{X} \in \mathbb{R}^{M\times N}$,
parameters: ϵ, L, δ, T

Initialize the problem:
$\boldsymbol{A}^0_{ext} = \bar{0}_N,\ \boldsymbol{B}^0_{ext} = \bar{0}_N,\ t = 0,\ l = 1$

while $l \leq L$ **do**

AMP updates of the marginals:

$$\mathbf{m}^l = AMP(\boldsymbol{X}, \boldsymbol{A}^{l-1}_{ext}, \boldsymbol{B}^{l-1}_{ext},\ \epsilon,\ T)$$

Update the fields using the Euler-Maruyama scheme:

$$t^l \leftarrow t^{l-1} + \delta \tag{4.23}$$

$$(A^l_{ext})_i \leftarrow t \tag{4.24}$$

$$(B^l_{ext})_i \leftarrow (B^{l-1}_{ext})_i + w_i^{l-1}\delta + \sqrt{\delta}\ Z_i \tag{4.25}$$

$$\text{with } \boldsymbol{Z} \sim \mathcal{N}(0, \mathbb{I}_N), \forall i \in \{1, \ldots, N\} \tag{4.26}$$

$$l \leftarrow l + 1 \tag{4.27}$$

end while

$$\mathbf{m_i}^L = \operatorname{sgn} \mathbf{m_i}^L, \forall i \in \{1 \ldots N\}$$

return a solution sample $\mathbf{m_i}^L$

4.5 Analytical details on Spherical Perceptron Investigation

4.5.1 Replica computation for Spherical Perceptron

In this Section, we outline the key steps of the replica calculation used to derive the free-entropy dynamics of the spherical perceptron under stochastic localization, as outlined in Section 4.3. We assume $M = \alpha N$ Gaussian-distributed examples, $X = \{\boldsymbol{x}^\mu\}_{\mu=1}^M$, $\boldsymbol{x}^\mu \sim \mathcal{N}\Big(\mathbf{0}, I_N\Big)$. The weights have a spherical (improper) prior $P(\boldsymbol{w}) = \delta\Big(\|\boldsymbol{w}\|^2 - N\Big)$. We want to compute the free entropy at time t of the model in (4.12) averaged over the realization of the sample and the noise in the SL process ((4.6)):

$$\phi_t = \lim_{N\to+\infty} \frac{1}{N} \mathbb{E} \log Z_{\mathbf{h}_t, t}. \tag{4.28}$$

This computation is performed using the replica trick twice (the second one is used to express the expectation over the reference configurations):

$$\log Z = \lim_{n\to 0} \partial_n Z^n \quad \text{and} \quad Z^{-1} = \lim_{s\to 0} Z^{s-1}.$$

We can thus write

$$\phi_t = \lim_{N\to+\infty} \frac{1}{N} \lim_{s\to 0} \lim_{n\to 0} \partial_n \, \mathbb{E} \int \prod_{\alpha=1}^{s} \psi\left(\mathrm{d}\boldsymbol{w}_\alpha^*\right) \prod_{a=1}^{n} \psi\left(\mathrm{d}\boldsymbol{w}_a\right) e^{\langle \mathbf{h}_t, \boldsymbol{w}_a\rangle - \frac{t}{2}\|\boldsymbol{w}_a\|^2}, \tag{4.29}$$

where $\mathbf{h}_t = t\boldsymbol{w}_1^* + \sqrt{t}\mathbf{g}$, with $\mathbf{g} \sim \mathcal{N}(0, I_N)$. For convenience, we define the replicated partition function

$$Z_t^{n,s} = \int \prod_{\alpha=1}^{s} \psi\left(\mathrm{d}\boldsymbol{w}_\alpha^*\right) \prod_{a=1}^{n} \psi\left(\mathrm{d}\boldsymbol{w}_a\right) e^{\langle \mathbf{h}_t, \boldsymbol{w}_a\rangle - \frac{t}{2}\|\boldsymbol{w}_a\|^2}. \tag{4.30}$$

and we also define the function

$$\tilde{\Theta}(s) = \Theta(s-\kappa)e^{-\frac{1}{T}U(s-\kappa)}, \tag{4.31}$$

where $\Theta(s)$ is the Heaviside function. The expectation over disorder reads

$$\mathbb{E}Z_t^{n,s} = \mathbb{E}_{X,\mathbf{g}} \int \prod_{\alpha=1}^{s} P(\mathrm{d}\boldsymbol{w}_\alpha^*) \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) \prod_{\mu\alpha} \tilde{\Theta}\left(\sum_i \frac{w_{\alpha i}^* x_i^\mu}{\sqrt{N}}\right) \prod_{\mu a} \tilde{\Theta}\left(\sum_i \frac{w_{ai} x_i^\mu}{\sqrt{N}}\right) \times \tag{4.32}$$

$$\times\, e^{t\sum_{ai} w_{1i}^* w_{ai} + \sqrt{t}\sum_{ai} g_i w_{ai} - \frac{1}{2}t\sum_{ai} w_{ai}^2}. \tag{4.33}$$

Introducing $\lambda_\alpha^\mu = \sum_i \frac{w_{\alpha i}^* x_i^\mu}{\sqrt{N}}$ and $u_a^\mu = \sum_i \frac{w_a x_i^\mu}{\sqrt{N}}$, and their conjugate Lagrange multiplier, we obtain

$$\mathbb{E}Z_t^{n,s} = \mathbb{E}_{X,\mathbf{g}} \int \prod_{\alpha=1}^{s} \mathrm{d}\boldsymbol{w}_\alpha^* \prod_{a=1}^{n} \mathrm{d}\boldsymbol{w}_a \prod_{\mu\alpha} \frac{\mathrm{d}\lambda_\alpha^\mu \mathrm{d}\hat{\lambda}_\alpha^\mu}{2\pi} \prod_{\mu a} \frac{\mathrm{d}u_a^\mu \mathrm{d}\hat{u}_a^\mu}{2\pi} \prod_{\mu\alpha} \tilde{\Theta}\left(\lambda_\alpha^\mu\right) \prod_{\mu a} \tilde{\Theta}\left(u_a^\mu\right) \tag{4.34}$$

$$\times\, e^{-i\sum_{\mu\alpha} \hat{\lambda}_\alpha^\mu \lambda_\alpha^\mu - i\sum_{\mu a} \hat{u}_a^\mu u_a^\mu + i\sum_{\mu\alpha i} \hat{\lambda}_\alpha^\mu \frac{w_{\alpha i}^* x_i^\mu}{\sqrt{N}} + i\sum_{\mu a i} \hat{u}_a^\mu \frac{w_{ai} x_i^\mu}{\sqrt{N}} + t\sum_{ai} w_{1i}^* w_{ai} + \sqrt{t}\sum_{ai} g_i w_{ai} - \frac{1}{2}t\sum_{ai} w_{ai}^2}. \tag{4.35}$$

It is now possible to compute the average over $\boldsymbol{x}^\mu$ and $\mathbf{g}$, obtaining

$$\mathbb{E}Z_t^{n,s} = \int \prod_{\alpha=1}^{s} \mathrm{d}\boldsymbol{w}_\alpha^* \prod_{a=1}^{n} \mathrm{d}\boldsymbol{w}_a \prod_{\mu\alpha} \frac{\mathrm{d}\lambda_\alpha^\mu \mathrm{d}\hat{\lambda}_\alpha^\mu}{2\pi} \prod_{\mu a} \frac{\mathrm{d}u_a^\mu \mathrm{d}\hat{u}_a^\mu}{2\pi} \prod_{\mu\alpha} \tilde{\Theta}\left(\lambda_\alpha^\mu\right) \prod_{\mu a} \tilde{\Theta}\left(u_a^\mu\right) \tag{4.36}$$

$$\times\, e^{-i\sum_{\mu\alpha} \hat{\lambda}_\alpha^\mu \lambda_\alpha^\mu - i\sum_{\mu a} \hat{u}_a^\mu u_a^\mu + t\sum_{ai} w_{1i}^* w_{ai} + \frac{1}{2}t\sum_i \sum_{ab} w_{ai} w_{bi} - \frac{1}{2}t\sum_{ai} w_{ai}^2} \tag{4.37}$$

$$\times\, e^{-\frac{1}{2N}\sum_{\mu i}(\sum_{\alpha\beta} \hat{\lambda}_\alpha^\mu \hat{\lambda}_\beta^\mu w_{\alpha i}^* w_{\beta i}^* + \sum_{ab} \hat{u}_a^\mu \hat{u}_b^\mu w_{ai} w_{bi} + 2\sum_{\alpha a} \hat{\lambda}_\alpha^\mu \hat{u}_a^\mu w_{\alpha i}^* w_{ai})}. \tag{4.38}$$

We can now define the overlap parameters as

$$q_{ab} = \frac{1}{N}\sum_i w_{ai} w_{bi} \qquad r_{\alpha\beta} = \frac{1}{N}\sum_i w_{\alpha i}^* w_{\beta i}^* \qquad p_{\alpha a} = \frac{1}{N}\sum_i w_{\alpha i}^* w_{ai}. \tag{4.39}$$

These overlap parameters measure, respectively, (i) the similarity between two independent replicas of the *tilted* measure, (ii) the similarity between two replicas of the *reference*

system, and (iii) the cross-overlap between one reference configuration and one tilted replica. Introducing conjugate parameters $\hat{q}_{ab}, \hat{r}_{\alpha\beta}, \hat{p}_{\alpha a}$ to enforce their definitions allows us to rewrite the disorder-averaged replicated partition function as an integral over both the overlaps and their Lagrange multipliers. The overlaps q_{aa} and $r_{\alpha\alpha}$ are fixed to 1 in order to enforce the spherical constraints. Carrying on the computation, we have

$$\mathbb{E}Z_t^{n,s} = \int \prod_{\alpha=1}^{s} \mathrm{d}\boldsymbol{w}_\alpha^* \prod_{a=1}^{n} \mathrm{d}\boldsymbol{w}_a \prod_{\alpha\leq\beta} \mathrm{d}r_{\alpha\beta}\mathrm{d}\hat{r}_{\alpha\beta} \prod_{a\leq b} \mathrm{d}q_{ab}\mathrm{d}\hat{q}_{ab} \prod_{\alpha a} \mathrm{d}p_{\alpha a}\mathrm{d}\hat{p}_{\alpha a} \tag{4.40}$$

$$\times\, e^{-N\frac{1}{2}\sum_{\alpha\beta}\hat{r}_{\alpha\beta}r_{\alpha\beta} - \frac{1}{2}N\sum_{ab}\hat{q}_{ab}q_{ab} - N\sum_{\alpha a}\hat{p}_{\alpha a}p_{\alpha a}} \tag{4.41}$$

$$\times\, e^{+\frac{1}{2}\sum_{\alpha\beta}\hat{r}_{\alpha\beta}\sum_i w_{\alpha i}^* w_{\beta i}^* + \frac{1}{2}\sum_{ab}\hat{q}_{ab}\sum_i w_{ai}w_{bi} + \sum_{\alpha a}\hat{p}_{\alpha a}\sum_i w_{\alpha i}^* w_{ai}} \tag{4.42}$$

$$\times \int \prod_{\mu\alpha} \frac{\mathrm{d}\lambda_\alpha^\mu \mathrm{d}\hat{\lambda}_\alpha^\mu}{2\pi} \prod_{\mu a} \frac{\mathrm{d}u_a^\mu \mathrm{d}\hat{u}_a^\mu}{2\pi} \prod_{\mu\alpha} \tilde{\Theta}\left(\lambda_\alpha^\mu\right) \prod_{\mu a} \tilde{\Theta}\left(u_a^\mu\right) \tag{4.43}$$

$$\times\, e^{-i\sum_{\mu\alpha}\hat{\lambda}_\alpha^\mu\lambda_\alpha^\mu - i\sum_{\mu a}\hat{u}_a^\mu u_a^\mu + tN\sum_a p_{1a} + \frac{1}{2}tN\sum_{ab}q_{ab} - \frac{t}{2}N\sum_a q_{aa}} \tag{4.44}$$

$$\times\, e^{-\frac{1}{2}\sum_\mu\left(\sum_{\alpha\beta}\hat{\lambda}_\alpha^\mu\hat{\lambda}_\beta^\mu r_{\alpha\beta} + \sum_{ab}\hat{u}_a^\mu\hat{u}_b^\mu q_{ab} + 2\sum_{\alpha a}\hat{\lambda}_\alpha^\mu\hat{u}_a^\mu p_{\alpha a}\right)}. \tag{4.45}$$

We obtain the following expression amenable to saddle point evaluation

$$\mathbb{E}Z_t^{n,s} = \int \prod_{\alpha\leq\beta} \mathrm{d}r_{\alpha\beta}\mathrm{d}\hat{r}_{\alpha\beta} \prod_{a\leq b} \mathrm{d}q_{ab}\mathrm{d}\hat{q}_{ab} \prod_{\alpha a} \mathrm{d}p_{\alpha a}\mathrm{d}\hat{p}_{\alpha a}\; e^{N\phi_t}, \tag{4.46}$$

where

$$\phi_t = G_I + G_S + \alpha G_E \tag{4.47}$$

$$G_I = -\frac{1}{2}\sum_{\alpha\beta}\hat{r}_{\alpha\beta}r_{\alpha\beta} - \frac{1}{2}\sum_{ab}\hat{q}_{ab}q_{ab} - \sum_{\alpha a}\hat{p}_{\alpha a}p_{\alpha a} + t\sum_a p_{1a} + \frac{1}{2}t\sum_{ab}q_{ab} - \frac{t}{2}\sum_a q_{aa}, \tag{4.48}$$

$$G_S = \log\int \prod_{\alpha=1}^{s} P(\mathrm{d}w_\alpha^*) \prod_{a=1}^{n} P(\mathrm{d}w_a)\; e^{+\frac{1}{2}\sum_{\alpha\beta}\hat{r}_{\alpha\beta}w_\alpha^* w_\beta^* + \frac{1}{2}\sum_{ab}\hat{q}_{ab}w_a w_b + \sum_{\alpha a}\hat{p}_{\alpha a}w_\alpha^* w_a}, \tag{4.49}$$

$$G_E = \log\int \prod_\alpha \frac{\mathrm{d}\lambda_\alpha \mathrm{d}\hat{\lambda}_\alpha}{2\pi} \prod_a \frac{\mathrm{d}u_a \mathrm{d}\hat{u}_a}{2\pi} \prod_\alpha \tilde{\Theta}\left(\lambda_\alpha\right) \prod_a \tilde{\Theta}\left(u_a\right) \tag{4.50}$$

$$\times\, e^{-i\sum_\alpha\hat{\lambda}_\alpha\lambda_\alpha - i\sum_a\hat{u}_a u_a - \frac{1}{2}\sum_{\alpha\beta}\hat{\lambda}_\alpha\hat{\lambda}_\beta r_{\alpha\beta} - \frac{1}{2}\sum_{ab}\hat{u}_a\hat{u}_b q_{ab} - \sum_{\alpha a}\hat{\lambda}_\alpha\hat{u}_a p_{\alpha a}}. \tag{4.51}$$

Before performing the optimization task through the saddle point equations, for a fixed value of t and α, one needs to simplify the expression of the free entropy. In the following, we introduce the Replica Symmetric assumption.

Replica Symmetric Ansatz

To compute the expression of the free entropy, we assume complete symmetry between the replicas (RS):

$$q_{ab} = \begin{cases} q_d = 1 & \text{if } a = b \\ q & \text{if } a \neq b \end{cases} \qquad r_{\alpha\beta} = \begin{cases} r_d = 1 & \alpha = \beta \\ r & \alpha \neq \beta \end{cases} \qquad p_{\alpha a} = \begin{cases} p^* & \alpha = 1 \\ p & \alpha \neq 1. \end{cases} \tag{4.52}$$

Plugging it in the ϕ_t expression and carrying over the $n \to 0$ and $s \to 0$ limit lets us compute the analytic expression for G_I, G_S, and G_E.

Bayesian Optimality conditions

It is possible to notice that Nishimori conditions, and Bayesian Optimality of the order parameters [21, 20] apply. They are a set of conditions stating that:

$$\mathbb{E}_{w_*,w_1}[f(w_*, w_1)] = \mathbb{E}_{w_1,w_2}[f(w_1, w_2)], \tag{4.53}$$

where

$$\mathbb{E}_{1,w_2}[f(w_1, w_2)] = \int \mathrm{d}h \, \mathrm{d}w_1 \, \mathrm{d}w_2 \; f(w_1, w_2) P(w_1 \mid h) P(w_2 \mid h) P(h) \tag{4.54}$$

$$\mathbb{E}_{w_*,w_1}[f(w_*, w_1)] = \int \mathrm{d}w_* \, \mathrm{d}w_1 \; f(w_*, w_1) P(w_1, w_*) \tag{4.55}$$

$$= \int \mathrm{d}h \, \mathrm{d}w_* \, \mathrm{d}w_1 \; f(w_*, w_1) P(w_1 \mid h) P(h \mid w_*) P(w_*) \tag{4.56}$$

$$= \int \mathrm{d}h \, \mathrm{d}w_* \, \mathrm{d}w_1 \; f(w_*, w_1) P(w_1 \mid h) P(w_* \mid h) P(h). \tag{4.57}$$

We can use this result to understand order parameter relations. In particular, using previous reasoning, we can notice that

$$\mathbb{E}_{w^*_\alpha, w_a}[w^*_{\alpha i} w_{ai}] = \mathbb{E}_{w_a, w_b}[w_{ai} w_{bi}]. \tag{4.58}$$

leading to

$$p^* = q. \tag{4.59}$$

At first glance, this identity may appear counterintuitive, but it reflects a fundamental property of the Bayes-optimal setting: any two samples drawn from the same posterior (the tilted distribution) are statistically indistinguishable from a sample drawn jointly with the true (reference) weights. Since all replicas of the tilted distribution are independent and identically distributed, their mutual overlap must coincide with their cross-overlap with the reference configuration. Using the same idea, it is possible to show

$$\mathbb{E}_{w^*_\alpha, w^*_\beta}[w^*_{\alpha i} w^*_{\beta i}] = \mathbb{E}_{w_1, w^*_\beta}[w_{1i} w^*_{\beta i}]. \tag{4.60}$$

leading to

$$p = r. \tag{4.61}$$

A similar intuition concerning the overlap of tilted distribution samples can be applied to the overlap among reference weights. Similar reasoning can be applied to the conjugate order parameters:

$$\hat{r}_d = \hat{q}_d \tag{4.62}$$

$$\hat{p}^* = \hat{q} \tag{4.63}$$

$$\hat{p} = \hat{r}. \tag{4.64}$$

These conditions simplify considerably the expressions in the computation of the free entropy.

Entropic term

Let's first look at the entropic term G_S within the RS ansatz:

$$G_S = \log \int \prod_{\alpha=1}^{s} P(\mathrm{d}w_\alpha^*) \prod_{a=1}^{n} P(\mathrm{d}w_a)\ e^{+\frac{1}{2}(\hat{r}_d-\hat{r})\sum_\alpha w_\alpha^{*2}+\frac{1}{2}(\hat{r}-\hat{p})(\sum_\alpha w_\alpha^*)^2} \tag{4.65}$$

$$\times\ e^{+\frac{1}{2}(\hat{q}_d-\hat{q})\sum_a w_a^2+\frac{1}{2}(\hat{q}-\hat{p})(\sum_a w_a)^2+(\hat{p}^*-\hat{p})w_1^*\sum_a w_a+\frac{\hat{p}}{2}(\sum_\alpha w_\alpha^*+\sum_a w_a)^2}. \tag{4.66}$$

Using the Nishimori conditions this simplifies to:

$$G_S = \log \int \prod_{\alpha=1}^{s} P(\mathrm{d}w_\alpha^*) \prod_{a=1}^{n} P(\mathrm{d}w_a)\ e^{\frac{1}{2}(\hat{r}_d-\hat{r})\sum_\alpha w_\alpha^{*2}} \tag{4.67}$$

$$\times\ e^{+\frac{1}{2}(\hat{r}_d-\hat{q})\sum_a w_a^2+\frac{1}{2}(\hat{q}-\hat{r})(\sum_a w_a)^2+(\hat{q}-\hat{r})w_1^*\sum_a w_a+\frac{\hat{r}}{2}(\sum_\alpha w_\alpha^*+\sum_a w_a)^2}. \tag{4.68}$$

Using the Hubbard-Stratonovich transformation twice we can get rid of the quadratic summations like $(\sum_a w_a)^2$ in the exponentials, at the cost of introducing two extra integration variables:

$$G_S = \log \int \prod_{\alpha=1}^{s} P(\mathrm{d}w_\alpha^*) \prod_{a=1}^{n} P(\mathrm{d}w_a) \int Dz D\gamma\ e^{\frac{1}{2}(\hat{r}_d-\hat{r})\sum_\alpha w_\alpha^{*2}} \tag{4.69}$$

$$\times\ e^{\frac{1}{2}(\hat{r}_d-\hat{q})\sum_a w_a^2+z\sqrt{\hat{q}-\hat{r}}\sum_a w_a+(\hat{q}-\hat{r})w_1^*\sum_a w_a+\gamma\sqrt{\hat{r}}(\sum_\alpha w_\alpha^*+\sum_a w_a)}, \tag{4.70}$$

where we used $Dz = \mathrm{d}z\frac{e^{-z^2/2}}{\sqrt{2\pi}}$ as a shorthand to denote a gaussian integral. This can now be factorized:

$$G_S = \log \int P(\mathrm{d}w_1^*) \int Dz D\gamma\ e^{\frac{1}{2}(\hat{r}_d-\hat{r})w_1^{*2}+\gamma\sqrt{\hat{p}}w_1^*}[\mathcal{Z}(w_1^*,z,\gamma)]^n[\mathcal{Z}^*(\eta,\gamma)]^{s-1}, \tag{4.71}$$

where

$$\mathcal{Z}(w_1^*, z, \gamma) = \int P(\mathrm{d}w) e^{\frac{1}{2}(\hat{r}_d - \hat{q})w^2 + \left(z\sqrt{\hat{q}-\hat{r}} + \gamma\sqrt{\hat{r}} + (\hat{q}-\hat{r})w_1^*\right)w} \tag{4.72}$$

$$= \frac{e^{-\frac{1}{2}\frac{\left(z\sqrt{\hat{q}-\hat{r}} + \gamma\sqrt{\hat{r}} + (\hat{q}-\hat{r})w_1^*\right)^2}{(\hat{r}_d - \hat{q})}}}{\sqrt{\hat{q} - \hat{r}_d}} \tag{4.73}$$

$$\mathcal{Z}^*(z, \gamma) = \int P(\mathrm{d}w^*) e^{\frac{1}{2}(\hat{r}_d - \hat{r})w^{*2} + \gamma\sqrt{\hat{r}}w^*} \tag{4.74}$$

$$= \frac{e^{-\frac{1}{2}\frac{\gamma^2 \hat{r}}{\hat{r}_d - \hat{r}}}}{\sqrt{\hat{r} - \hat{r}_d}} \tag{4.75}$$

Finally, by taking the limits for $n \to 0$ and $s \to 0$ in the same order we obtain

$$\mathcal{G}_S(\hat{q}) = \lim_{s\to 0}\lim_{n\to 0} \partial_n G_S \tag{4.76}$$

$$= -\frac{1}{2}\log(\hat{q} - \hat{r}_d) + \frac{1}{2(\hat{q} - \hat{r}_d)}\left(\hat{q} + 2\hat{r}\frac{(\hat{q} - \hat{r})}{(\hat{r} - \hat{r}_d)} + \frac{(\hat{q} - \hat{r})^2}{(\hat{r} - \hat{r}_d)}\left(\frac{\hat{r}}{\hat{r} - \hat{r}_d} + 1\right)\right). \tag{4.77}$$

Energetic term

Let's now focus our attention on the energetic term

$$G_E = \log \int \prod_\alpha \frac{\mathrm{d}\lambda_\alpha \mathrm{d}\hat{\lambda}_\alpha}{2\pi} \prod_a \frac{\mathrm{d}u_a \mathrm{d}\hat{u}_a}{2\pi} \prod_\alpha \tilde{\Theta}(\lambda_\alpha) \prod_a \tilde{\Theta}(u_a) \tag{4.78}$$

$$\times\, e^{-i\sum_\alpha \hat{\lambda}_\alpha \lambda_\alpha - i\sum_a \hat{u}_a u_a - \frac{1}{2}\sum_{\alpha\beta}\hat{\lambda}_\alpha \hat{\lambda}_\beta r_{\alpha\beta} - \frac{1}{2}\sum_{ab}\hat{u}_a \hat{u}_b q_{ab} - \sum_{\alpha a}\hat{\lambda}_\alpha \hat{u}_a p_{\alpha a}}. \tag{4.79}$$

After using the Nishimori conditions and some manipulations we get

$$G_E = \log \int \prod_\alpha \frac{\mathrm{d}\lambda_\alpha \mathrm{d}\hat{\lambda}_\alpha}{2\pi} \prod_a \frac{\mathrm{d}u_a \mathrm{d}\hat{u}_a}{2\pi} \prod_\alpha \tilde{\Theta}(\lambda_\alpha) \prod_a \tilde{\Theta}(u_a) \tag{4.80}$$

$$\times\, e^{-i\sum_\alpha \hat{\lambda}_\alpha \lambda_\alpha - i\sum_a \hat{u}_a u_a - \frac{1}{2}(1-r)\sum_\alpha \hat{\lambda}_\alpha^2 + \frac{q-r}{2}\hat{\lambda}_1^2} \tag{4.81}$$

$$\times\, e^{-\frac{1}{2}(1-q)\sum_\alpha \hat{u}_\alpha^2 - \frac{r}{2}(\sum_\alpha \hat{\lambda}_\alpha + \sum_a \hat{u}_a)^2 - \frac{q-r}{2}\left(\hat{\lambda}_1 + \sum_a \hat{u}_a\right)^2}. \tag{4.82}$$

We use again two Hubbard-Stratonovich substitutions to make the integrand factorized in the a and α indices, at the cost of introducing two more gaussian integrals:

$$G_E = \log \int Dz\, D\gamma \prod_\alpha \frac{\mathrm{d}\lambda_\alpha \mathrm{d}\hat{\lambda}_\alpha}{2\pi} \prod_a \frac{\mathrm{d}u_a \mathrm{d}\hat{u}_a}{2\pi} \prod_\alpha \tilde{\Theta}\left(\lambda_\alpha\right) \prod_a \tilde{\Theta}\left(u_a\right) \tag{4.83}$$

$$\times\, e^{-i\sum_\alpha \hat{\lambda}_\alpha(\lambda_\alpha+\gamma\sqrt{r})-i\sum_a \hat{u}_a(u_a+\gamma\sqrt{r}+z\sqrt{q-r})-\frac{1}{2}(1-r)\sum_\alpha \hat{\lambda}_\alpha^2+\frac{q-r}{2}\hat{\lambda}_1^2} \tag{4.84}$$

$$\times\, e^{-\frac{1}{2}(1-q)\sum_\alpha \hat{u}_\alpha^2 - iz\sqrt{q-r}\hat{\lambda}_1}. \tag{4.85}$$

Finally, collecting the factors and taking the limits for $n \to 0$ and $s \to 0$, we arrive at

$$\mathcal{G}_E(q) = \lim_{s\to 0}\lim_{n\to 0} \partial_n G_E \tag{4.86}$$

$$= \int Dz D\gamma \frac{\tilde{H}_{1-q}\left(\gamma\sqrt{r}+z\sqrt{q-r}\right)}{\tilde{H}_{1-r}\left(\gamma\sqrt{r}\right)} \log \tilde{H}_{1-q}\left(\gamma\sqrt{r}+z\sqrt{q-r}\right). \tag{4.87}$$

where

$$\tilde{H}_a(b) = \int Dz\ \Theta(-b-\kappa+\sqrt{a}z) e^{-\frac{1}{T}U(-b-\kappa+\sqrt{a}z)} \tag{4.88}$$

In the uniform measure case $U(s) = 0$, $\tilde{H}$ simplifies to $\tilde{H}_a(b) = H(-\frac{b+\kappa}{\sqrt{a}})$, with $H(x) = \int_x^{+\infty} Dz = \frac{1}{2}\mathrm{erfc}\left(\frac{x}{\sqrt{2}}\right)$.

Interaction term

Finally we compute the interaction term

$$G_I = -\frac{1}{2}\sum_{\alpha\beta}\hat{r}_{\alpha\beta}r_{\alpha\beta} - \frac{1}{2}\sum_{ab}\hat{q}_{ab}q_{ab} - \sum_{\alpha a}\hat{p}_{\alpha a}p_{\alpha a} + t\sum_a p_{1a} + \frac{1}{2}t\sum_{ab}q_{ab} - \frac{t}{2}\sum_a q_{aa}, \tag{4.89}$$

which, after using the RS ansatz, the Nishimori conditions, and taking the limits $n \to 0$ and $s \to 0$, reduces to:

$$\mathcal{G}_I = \lim_{s\to 0}\lim_{n\to 0}\partial_n G_I = -\frac{1}{2}\hat{q}_r q_d - \frac{1}{2}\hat{q}q + \hat{r}r + \frac{1}{2}tq = -\frac{1}{2}\hat{r}_d - \frac{1}{2}\hat{q}q + \hat{r}r + \frac{1}{2}tq \tag{4.90}$$

Saddle Point Equations

The optimization of ϕ_t can now be performed using the so called saddle point method [11]. The equations that one needs to optimize are the following

$$\hat{q} = t + 2\alpha \frac{\partial}{\partial q} G_E(q), \tag{4.91}$$

$$q = 2\frac{\partial}{\partial \hat{q}} G_S(\hat{q}). \tag{4.92}$$

It is important to notice that the overlap parameters for the reference network are not optimized here. The values of the reference overlaps need to be optimized ahead of the learning procedure for the tilted ones. Their optimization procedure is schematically shown in the next section, and the full computation is reported in [23].

Reference system

In the optimization procedure, there are still parameters that are not optimized: r, $\hat{r}$ and $\hat{r}_d$. They are the parameters of the reference perceptron, and need to be optimized ahead of the optimization of tilted ones. The reason behind this consists in the fact that, the sampling procedure of Stochastic Localization is realized on the tilted measure, since the target distribution is fixed from the origin. The optimization of them is a known problem in Statistical Physics of Machine learning [23], and is performed using optimizing the following set of saddle point equations:

$$r = 1 - 2\frac{\partial}{\partial \hat{r}} \mathcal{G}_S^{(r)}\left(\hat{r}, \hat{r}_d\right), \tag{4.93}$$

$$1 = 1 - 2\frac{\partial}{\partial \hat{r}_d} \mathcal{G}_S^{(r)}\left(\hat{r}, \hat{r}_d\right), \tag{4.94}$$

$$\hat{r} = -2\alpha \frac{\partial}{\partial r} \mathcal{G}_E^{(r)}\left(r\right), \tag{4.95}$$

where the free entropy associated to the reference system is

$$\phi^{(r)}(r,\hat{r},\hat{r}_d) = \mathcal{G}_I^{(r)}(r,\hat{r},\hat{r}_d) + \mathcal{G}_S^{(r)}(\hat{r},\hat{r}_d) + \alpha\mathcal{G}_E^{(r)}(r), \tag{4.96}$$

$$\mathcal{G}_I^{(r)}(r,\hat{r},\hat{r}_d) = \frac{1}{2}(r\hat{r} - \hat{r}_d), \tag{4.97}$$

$$\mathcal{G}_S^{(r)}(\hat{r},\hat{r}_d) = \frac{1}{2}\left(\log(2\pi) - \log(\hat{r}-\hat{r}_d) + \frac{\hat{r}}{(\hat{r}-\hat{r}_d)}\right), \tag{4.98}$$

$$\mathcal{G}_E^{(r)}(r) = \int Dz \log \tilde{H}_{1-r}\left(\sqrt{r}z\right). \tag{4.99}$$

The above expressions allows us to derive $\phi_t(q) = \mathrm{extr}_{\hat{q}}\,\phi_t(q,\hat{q})$ as

$$\phi_t(q,\hat{q}) = -\frac{1}{2}\left(\hat{r}_d + \hat{q}q\right) + \hat{r}r + \frac{1}{2}tq + \mathcal{G}_S(\hat{q}) + \alpha\,\mathcal{G}_E(q), \tag{4.100}$$

$$\tag{4.101}$$

and gain direct insight into whether the SL sampling scheme can recover samples from the target distribution asymptotically. The left panel of Fig. 1 shows $\phi_t(q)$ as a function of q for different values of t, with parameters $\alpha = 278$ and $\kappa = -2.5$. Initially, $\phi_t(q)$ exhibits a single global maximum. However, a second local maximum emerges as t increases, eventually becoming the global optimizer of ϕ_t. As discussed in Section 4.3, this transition marks the onset of multimodality in the free entropy landscape. The central panel of Fig. 1 presents the phase diagram for the ASL scheme for a fixed margin κ, delineating the different sampling regimes. For each point in the t-α plane, we test for the presence or absence of distinct local maxima by initializing the optimization of $\phi_t(q,\hat{q})$ in (4.100) at two different values of q, low and high, and checking whether the results coincide. The right panel of Fig. 1 shows the region in the α-κ plane where ASL succeeds. The figure also shows replica symmetry-breaking transition lines and the 1RSB prediction for the sat/unsat transition line reported in [26]. Notably, the samplability frontier of ASL coincides with the De Almeida-Thouless (DAT) line and comes slightly short of the dynamical 1-step Replica Symmetry Breaking (1RSB) transition. This result highlights a fundamental limitation: the SL algorithm fails to sample from the target distribution

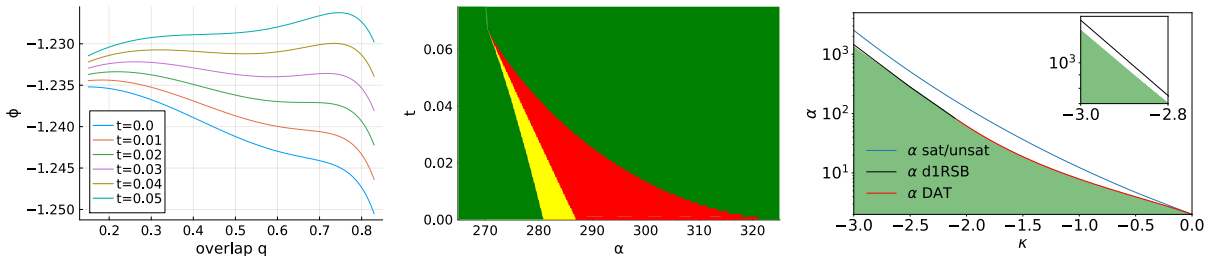

Figure 1: Asymptotic analysis of ASL sampling for the Spherical Perceptron with uniform distribution. **Left:** Free entropy function $\phi_t(q)$ for different values of t and for $\alpha = 278, \kappa = -2.5$. Initially, $\phi_t(q)$ has a single maximum, but as t increases, a second maximum appears, eventually becoming the global one. **Center:** Phase diagram of ASL in the t-vs-α plane for $\kappa = -2.5$. *Green region:* $\phi_t(q)$ has a single optimizer, meaning the AMP succeeds at denoising. *Yellow region:* $\phi_t(q)$ has two optimizers, but the global maximum corresponds to the smaller overlap q. AMP still succeeds. *Red region:* $\phi_t(q)$ has two optimizers, but the global maximum corresponds to a larger overlap q. In this case, AMP fails the denoising task. In order for ASL to succeed at sampling, a vertical line at the corresponding α should lie entirely in the green region. **Right:** Phase diagram delineating the samplable and non-samplable regions for ASL in α-vs-κ plane. Transition lines predicted from replica theory are taken from [26]. The green region can be sampled by ASL. The zoom in the inset shows the failure of ASL at reaching the d1RSB line.

when Replica Symmetry Breaking (RSB) occurs, which signals the fragmentation of the solution space into disconnected clusters. In other words, ASL ceases to function as soon as ergodicity is broken. In the case of the DAT transition, ergodicity is continuously broken and ASL reaches the transition line. In presence of a discontinuous (d1RSB) transition, failure of ASL happens earlier. See [17] for more details of phase transitions in similar contexts and [11, 19] for the generic RSB picture.

To validate numerically the uniformity of the ASL sampling, we compare the empirical distribution of stabilities s^μ obtained via sampling to the asymptotic theoretical prediction, which can also be obtained by the replica method. The derivation and the results are reported in Appendix 4.A. The empirical results match perfectly with the predictions, in the whole region where ASL successfully samples a solution to the constraint satisfaction problem. This strongly suggests that the obtained samples are distributed according to the target distribution, i.e., uniformly over the solution space in this case, as expected.

4.6 Binary Perceptron

4.6.1 Sampling From Binary perceptron uniform distribution

In this section, we investigate the performance of the ASL sampling scheme through our replica analysis, in the case the target distribution is the one in (4.12), specialized on the binary weights case $P(w_i) = \delta(w_i - 1) + \delta(w_i + 1)$. We focus on the zero margin case, $\kappa = 0$.

Replica computation for Binary Perceptron uniform distribution

The replica computation for the binary perceptron involves steps identical to the ones in Section 4.5.1, adjusted to the fact that in this case $P(w_i) = \delta(w_i - 1) + \delta(w_i + 1)$. One can notice that this change does not affect the specific form of the interaction term $\mathcal{G}_I$, but only the entropic term $\mathcal{G}_S$, which depends on the pattern distribution. By following the same steps as before, one obtains that the entropic term can be computed as:

$$\mathcal{G}_S(\hat{q}) = \int P(\mathrm{d}w_1^*) Dz D\gamma \; \frac{\mathcal{A}(w_1^*, \gamma)}{\mathcal{Z}^*(\gamma)} \log \mathcal{Z}(w_1^*, z, \gamma) \tag{4.102}$$

where

$$\mathcal{Z}(w_1^*, z, \gamma) = \int P(\mathrm{d}w) e^{+\frac{1}{2}(\hat{q}_d - \hat{q})w^2 + \left(z\sqrt{\hat{q} - \hat{r}} + \gamma\sqrt{\hat{r}} + (\hat{q} - \hat{r})w_1^*\right)w} \times \tag{4.103}$$

$$= e^{+\frac{1}{2}(\hat{q}_d - \hat{q})} \cosh\left(z\sqrt{\hat{q} - \hat{r}} + \gamma\sqrt{\hat{r}} + (\hat{q} - \hat{r})w_1^*\right) \tag{4.104}$$

$$\mathcal{Z}^*(z, \gamma) = \int P(\mathrm{d}w^*) e^{+\frac{1}{2}(\hat{r}_d - \hat{r})w^{*2} + \gamma\sqrt{\hat{r}}w^*} \tag{4.105}$$

$$= e^{+\frac{1}{2}(\hat{r}_d - \hat{r})} \cosh(\gamma\sqrt{\hat{r}}) \tag{4.106}$$

$$\mathcal{A}(w_1^*, \gamma) = e^{+\frac{1}{2}(\hat{r}_d - \hat{r})w_1^{*2} + \gamma\sqrt{\hat{p}}w_1^*} \tag{4.107}$$

The final expression of the Entropic term G_S is

$$\mathcal{G}_S(\hat{q}) = \sum_{w^*=\pm 1} \int Dz\, D\gamma\, \frac{e^{\gamma\sqrt{\hat{r}}w^*}}{2\cosh(\gamma\sqrt{\hat{r}})} \log\left(2e^{\frac{1}{2}(\hat{r}_d-\hat{q})}\cosh\left(z\sqrt{\hat{q}-\hat{r}}+\gamma\sqrt{\hat{r}}+(\hat{q}-\hat{r})w^*\right)\right). \tag{4.108}$$

Analogously, the expression of the entropic term of the reference systems is

$$\mathcal{G}_S^{(r)}(\hat{r},\hat{r}_d) = -\frac{1}{2}(\hat{r}-\hat{r}_d) + \int Dz \log\left(2\cosh\left(z\sqrt{r}\right)\right). \tag{4.109}$$

Replica Results for binary perceptron uniform distribution

We investigate the output of the replica formalism on the uniform distribution for the binary perceptron. The right panel of Fig. 2 shows the free entropy $\phi_t(q)$ as a function of the overlap q for different values of t[2], with parameters $\alpha = 0.5$ and $\kappa = 0.0$. The phenomenology observed in the case of the uniform distribution for the binary perceptron is completely different from that observed for the uniform distribution for the spherical one. The second maximum is always present. The persistent multimodality of $\phi_t(q)$ for $q = 1$ implies that during the sampling procedure, the information over the target is not correctly reconstructed, leading to an impossibility to sample from the target distribution. As we will show in Section 4.D, the persistent peak in $q = 1$ originates by the presence of $\Theta(\cdot)$ in the definition of the target distribution.

[2]In this scenario, the overlaps of the reference weights are fixed in advance and depends only on t and α. Instead, $\hat{q}$ can be derived implicitly as a function of q. This idea implies that ϕ_t can be seen as a function of the only q.

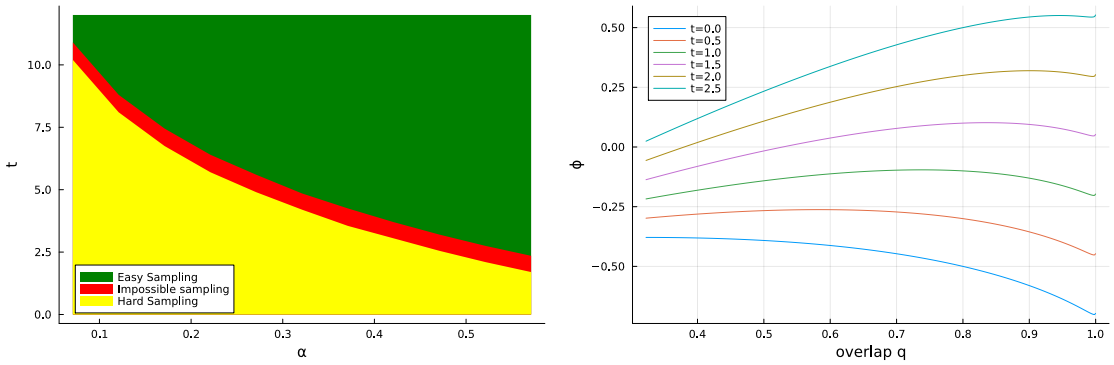

Figure 2: **Left:** Phase transition of the SL algorithm for the binary perceptron problem. The unfeasible phase is represented by all the parameter domains. **Right:** The free entropy of the SL for the binary perceptron problem as a function of the overlap q. Double peak behavior is exhibited for all the values of t with the global optimizer corresponding to the overlap $q = 1$.

Following the methodology outlined in Section 4.5.1, we conclude that sampling solutions from the flat binary perceptron measure using the ASL algorithm, in the limit $N \to +\infty$, is intrinsically infeasible. This result is not surprising, as the asymptotic analysis performed via the replica method is known to accurately predict the behavior of the AMP algorithm for large system sizes [25]. As demonstrated in [27], due to the inherent stability of AMP, the exact evaluation of the expected values in (4.3) becomes impossible whenever the solution space is fragmented, as is the case in the frozen-1RSB phase and overlap gap property is observed [28]. Thus, the failure mode of the ASL sampling scheme can be directly attributed to the limitations of AMP. The global optimizer corresponding to an overlap value of $q = 1$, suggests the presence of a spinodal transition.

The fundamental difference between these two models lies in the statistical properties of their solution spaces. For sufficiently small values of α the negative spherical perceptron can be fully described within the RS ansatz, implying a connected solution space. In contrast, the binary perceptron is characterized by a frozen 1RSB phase, where the solution space consists of disconnected clusters. This structural fragmentation directly impacts the effectiveness of the SL sampling scheme, ultimately rendering it unsuitable for this problem.

4.7 The cross entropy model for perceptron

Cross-entropy loss Due to the negative result for the uniform measure presented in the last section, we consider a different measure where interpolation is not strictly enforced but it is obtained in the large β and γ limit of the following factor which contains the binary cross-entropy:

$$\psi(w) = \prod_{\mu} e^{-\frac{\beta}{\gamma}\log\left(1+e^{-2\gamma\frac{\langle \boldsymbol{x}^{\boldsymbol{\mu}},w\rangle}{\sqrt{N}}}\right)}. \tag{4.110}$$

As studied in [29] this loss allows for targeting dense regions of solutions (the high local entropy regions) which is controlled by the $\gamma > 0$ parameter. One may note that in the limit of $\beta, \gamma \to \infty$ the model is forcing the constraints $\frac{\langle \boldsymbol{x}^{\boldsymbol{\mu}},\boldsymbol{w}\rangle}{\sqrt{N}} > 0, \forall \mu \in [M]$ to be strictly satisfied, thus the solution to such a problem corresponds to a binary perceptron one. Our results indicate the possibility of sampling from this target measure with the ASL scheme for a sufficiently low density constraint α. The derivation of the free entropy for this model can be found in the supplementary material. This measure while sampleable is not entirely satisfying, since the necessity to set β large enough to comply with the constraints greatly reduces the support of the measure.

Performing the experiments in Fig. 3 we demonstrate that the SL is sampling the solutions from the target distribution for β values that allow for non-zero free entropy.

Analysing the free entropy with an approach similar to the previous models we are able to obtain the phase diagram for the SL on the CE model (left panel of Fig. 4). The corresponding computation of the free entropy can be found in the 4.B.

Selecting a samplable potential

Sampling fails for non-diverging potentials As stated in the Section 4.3, ASL fails in the presence of a second peak in the free entropy $\phi_t(q)$ that becomes the global maximum at some time. For the binary perceptron under the uniform distribution, i.e.

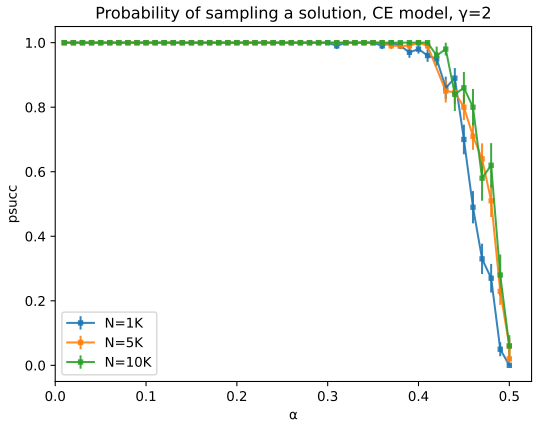

Figure 3: The probability of sampling a solution satisfying the binary perceptron constraints with the SL algorithm as a function of the constraint density α. The points represent the mean (with the corresponding standard deviation) obtained using the parallel 100 runs of the experiment.

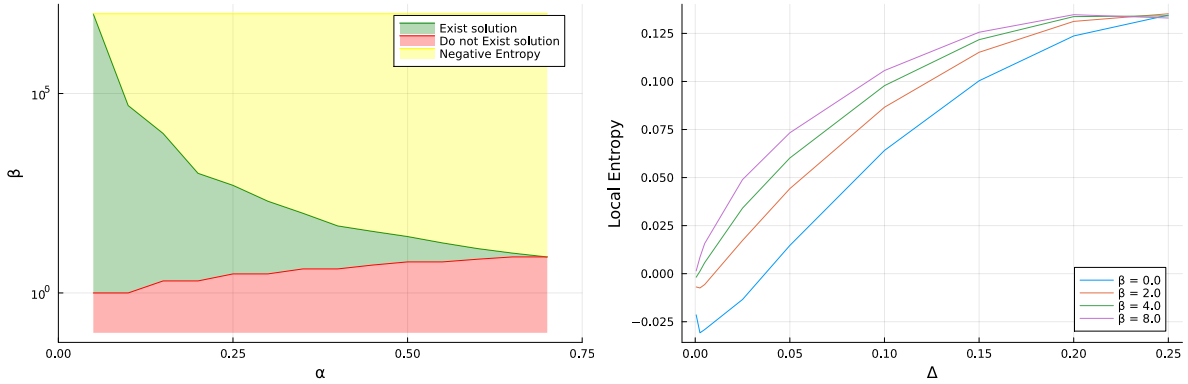

Figure 4: **Left:** Phase Diagram β vs α, plotted for $t = 0.1$ and $\gamma = 2$. The red region represents the not samplable parameter regime, the green one - samplable regime, and the yellow one - where the entropy gets negative, as a function of β. **Right:** Representation of the local entropy of the CE model as a function of the Hamming distance Δ, for different values of β.

with $U(s) = 0$, $\phi_t(q)$ exhibits a permanent peak at $q = 1$ with an infinite derivative, for all $\alpha > 0$ and all $t > 0$. This observation is related to the frozen-1RSB nature of the binary perceptron [30], implying that most solutions are isolated, and it is consistent with the known hardness of sampling from the uniform distribution [14, 31]. More surprisingly, this difficulty cannot be removed even by using a non-zero potential, unless it diverges at the origin, as we will now discuss.

To understand the origin of the peak at $q = 1$, we compute $\frac{d\phi_t(q)}{dq}$, for $q = 1 - \epsilon$, with $\epsilon \ll 1$ [14]. The computations are reported in Appendix 4.D. For the binary case, independently of the choice of the potential, the expression of the free entropy derivative takes the form:

$$\left.\frac{d\phi_t(q)}{dq}\right|_{q=1-\epsilon} = \frac{1}{2}\log(\epsilon) + \alpha\, C(\epsilon)\, \epsilon^{-\frac{1}{2}} + O(1) \tag{4.111}$$

where $C(\epsilon)$ is a function whose scaling with ϵ depends on the explicit form of the potential $U(s)$. As it turns out, unless $U(s)$ diverges at 0, $C(\epsilon) = O(1)$ and is always positive, and as a consequence the free entropy unavoidably exhibits a peak at $q = 1$ at all $t > 0$ and all $\alpha > 0$. The second peak becomes dominant at large enough time, before the first peak disappears, leading to the failure of ASL sampling.

Diffusion with Log-potential The only way to avoid the peak at $q = 1$ is to let the potential $U(s)$ diverge at the origin. The choice $U(s) = -\log(s)$ is particularly simple to analyze: in that case, $C(\epsilon) = O(\epsilon^{\frac{1}{2T}})$ and therefore the corresponding term $C(\epsilon)\,\epsilon^{-\frac{1}{2}}$ in (4.111) becomes negligible for small ϵ when $T < 1$. The derivative of the free entropy at $q = 1$ becomes dominated by the logarithmic term and thus $\phi_t(q)$ always has a local minimum at $q = 1$.

In this case, the phenomenology becomes similar to the one observed for the spherical perceptron case, as shown in Figure 5 (Left): with the choice $T = 0.5$, there is a whole range of α up to about $\alpha_{\text{alg}} \approx 0.65$ that is samplable. The transition is close to, although slightly lower than, the best known algorithmic thresholds from heuristic solvers [32, 33,

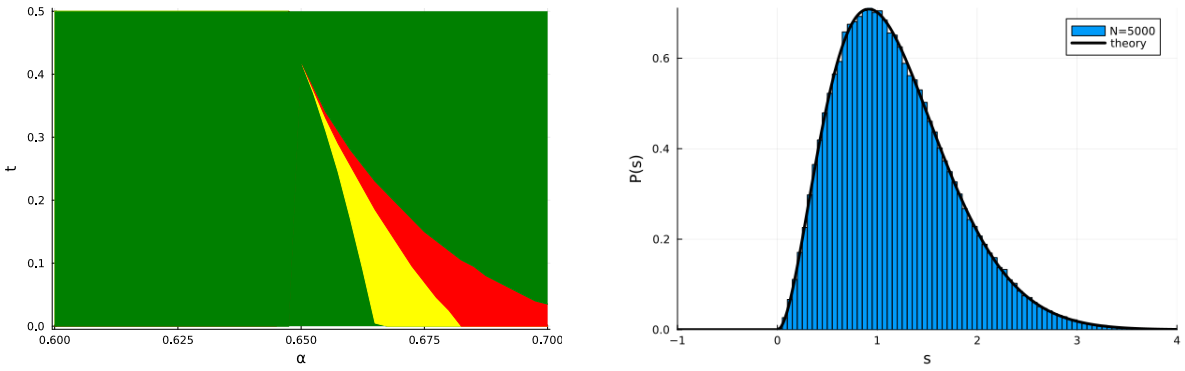

Figure 5: ASL sampling for the Binary Perceptron with $U(s) = -\log(s)$ potential. **Left:** Phase diagram of ASL in the t-vs-α plane for $\kappa = 0$ and $T = 0.5$. The color scheme is as explained for the central panel of Figure 1. **Right:** Empirical distribution of the stabilities s^μ for a configuration obtained by ASL in the case of binary perceptron with the log-potential, $N = 5000$, $\kappa = 0, T = 0.5$, and $\alpha = 0.3$. The black line is the asymptotic theoretical prediction. The excellent agreement shows that ASL produces fair samples from the target distribution.

34]. In our case though, and contrary to all other solver algorithms we are aware of, the solutions found are fair samples from a target distribution which is fully under control analytically. This is corroborated by the comparison between the empirical stabilities from the ASL sampler and the theoretical predictions, shown in Fig. 5 (Right).

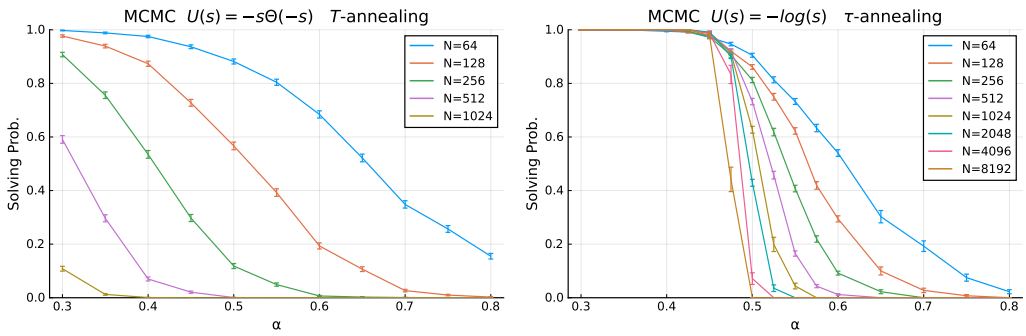

Figure 6: Results for the Binary Perceptron problem, showing the probability of finding a solution as a function of constraint density α and for different system sizes N, after 100 sweeps of MCMC. Simulated Annealing on temperature T (left) is compared to our proposed and much more effective τ-annealing scheme.

Experimental results for sampling binary perceptron solutions with Log-potential

In Figure 7, we report the probability of sampling a correct configuration from the binary perceptron loss measure with the Log-potential, for $\beta = 2$, using the ASL sampling scheme. We are able to correctly sample configurations for $\alpha \sim 0.55$.

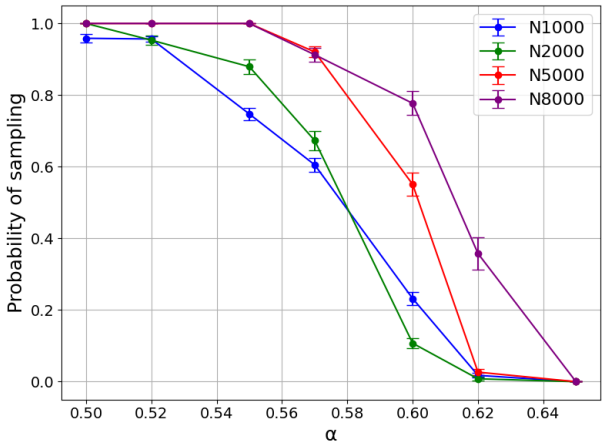

Figure 7: The probability of sampling a solution satisfying the binary perceptron constraints with the SL algorithm as a function of the constraint density α from a measure with the Log-potential with $T = 0.5$. Each point is averaged over 250 runs.

τ-annealed MCMC

Using the ASL algorithm in combination with the potential $U(s) = -\log(s)$, we are able to sample solutions of random instances of the binary perceptron problem. ASL, however, inherits the well-known limitations of AMP, generally failing to converge in the presence of structured data, and therefore ASL should not be considered a practical sampling algorithm for generic perceptron problems. On the other hand, direct use of the log-potential in an MCMC algorithm is infeasible, since the potential is not defined on negative stabilities s, which means that one should initialize the MC chain from a solution. As a workaround, we propose a reshaping of the potential inspired by the

replica trick, which, given a parameter $\tau > 0$, is defined by

$$U_\tau(s) = \begin{cases} \frac{1}{\tau}(1 - s^\tau) & s > 0, \\ \frac{1}{\tau}(1 - s) & s \leq 0. \end{cases} \tag{4.112}$$

Notice that $\lim_{\tau \to 0} U_\tau(s) = -\log(s)$ for $s > 0$ and $+\infty$ for $s \leq 0$. We set $p_\tau(\boldsymbol{w}) \propto e^{-\frac{1}{T}\sum_\mu U_\tau(s^\mu)}$ as the moving target density of a Metropolis-Hasting algorithm, where at each MC sweep we decrease linearly τ, starting with initial value 1 and down to 0. Keeping T fixed and with a sufficiently slow annealing, we should be able to sample from the solution space weighted by the log-potential. We call this procedure τ-annealing. In Figure 6, we compare it with a standard Simulated Annealing (SA) on the potential $U(s) = -s\Theta(-s)$, where the temperature T is decreased linearly at each MC sweep from 1 down to 0, so that final samples at $T = 0$ should be distributed according to the uniform measure for a slow enough annealing. While for SA fails quickly when increasing N, the τ-annealing scheme remains very effective at finding solutions. In the Appendix 4.C.1, we present further experiments showing that, by scaling the number of sweeps as $n_{\text{sweeps}} = N$, we solve up to $\alpha_{\text{alg}} \approx 0.55$ using τ-annealing, while temperature annealing keeps struggling at large system sizes.

4.8 Conclusion

In this chapter, we investigated the feasibility of sampling solutions to perceptron problems via the diffusion scheme based on Approximate Message Passing, known as Algorithmic Stochastic Localization (ASL). For the spherical perceptron, we showed that ASL can sample from the target distribution as long as the free entropy landscape remains unimodal along the trajectory, which holds for constraint density $\alpha = M/N$ below a threshold depending on the margin κ. In contrast, in the binary case, the uniform distribution is always unsamplable. An investigation of the origin of the issue that prevents ASL from working led us to introducing a potential $U(s) = -\log(s)$ to bias the distri-

bution. This enables efficient sampling over a broad range of α values. This potential also leads to a robust MCMC scheme, τ-annealing, that overcomes ASL's limitations (inherited from AMP) on structured instances of the problem. Looking forward, similar analyses and tailored potentials could enhance sampling and solving in other hard constraint satisfaction problems, especially with isolated solutions. A promising direction is to rigorously establish our results, particularly for the mathematically simpler binary symmetric perceptron [35].

References

[1] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[3] David L. Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.

[4] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[6] S. Brooks, A. Gelman, G. Jones, and X.L. Meng. *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. Taylor & Francis, 2011.

[7] Louis Grenioux, Maxence Noble, Marylou Gabrié, and Alain Oliviero Durmus. Stochastic localization via iterative posterior sampling, 2024.

[8] Maxence Noble, Louis Grenioux, Marylou Gabrié, and Alain Oliviero Durmus. Learned reference-based diffusion sampling for multi-modal distributions. *arXiv preprint arXiv:2410.19449*, 2024.

[9] Francisco Vargas, Will Grathwohl, and Arnaud Doucet. Denoising diffusion samplers. *arXiv preprint arXiv:2302.13834*, 2023.

[10] Francisco Vargas, Shreyas Padhy, Denis Blessing, and Nikolas Nüsken. Transport meets variational inference: Controlled monte carlo diffusions. *arXiv preprint arXiv:2307.01050*, 2023.

[11] Marc Mézard, Giorgio Parisi, and M. A. Virasoro. Spin glass theory and beyond: An introduction to the replica method and its applications. 1986.

[12] Patrick Charbonneau, Enzo Marinari, Giorgio Parisi, Federico Ricci-tersenghi, Gabriele Sicuro, Francesco Zamponi, and Marc Mezard. *Spin glass theory and far beyond: replica symmetry breaking after 40 years*. World Scientific, 2023.

[13] Jean Barbier, Florent Krzakala, Nicolas Macris, Léo Miolane, and Lenka Zdeborová. Optimal errors and phase transitions in high-dimensional generalized linear models. *Proceedings of the National Academy of Sciences*, 116(12):5451–5460, 2019.

[14] Haiping Huang and Yoshiyuki Kabashima. Origin of the computational hardness for learning with binary synapses. *Physical Review E*, 90(5):052813, 2014.

[15] Carlo Baldassi, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Subdominant dense clusters allow for simple learning and high computational performance in neural networks with discrete synapses. *Phys. Rev. Lett.*, 115:128101, Sep 2015.

[16] Andrea Montanari. Sampling, diffusions, and stochastic localization, 2023.

[17] Davide Ghio, Yatin Dandi, Florent Krzakala, and Lenka Zdeborová. Sampling with flows, diffusion, and autoregressive neural networks from a spin-glass perspective. *Proceedings of the National Academy of Sciences*, 121(27):e2311810121, 2024.

[18] Silvio Franz and Giorgio Parisi. Recipes for metastable states in spin glasses. *J. Phys. I France*, 5(11):1401–1415, 1995.

[19] Marc Mezard and Andrea Montanari. *Information, Physics, and Computation*. Oxford University Press, Inc., USA, 2009.

[20] H Nishimori. Exact results and critical properties of the ising model with competing interactions. *Journal of Physics C: Solid State Physics*, 13(21):4071, jul 1980.

[21] Yukito Iba. The nishimori line and bayesian statistics. *Journal of Physics A: Mathematical and General*, 32(21):3875–3888, January 1999.

[22] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[23] A. Engel and C. Van den Broeck. *Statistical Mechanics of Learning*. Cambridge University Press, 2001.

[24] Marylou Gabrié, Surya Ganguli, Carlo Lucibello, and Riccardo Zecchina. Neural networks: From the perceptron to deep nets. In *Spin Glass Theory and Far Beyond: Replica Symmetry Breaking After 40 Years*, pages 477–497. World Scientific, 2023.

[25] Sundeep Rangan. Generalized approximate message passing for estimation with random linear mixing. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 2168–2172. IEEE, 2011.

[26] Carlo Baldassi, Enrico M. Malatesta, Gabriele Perugini, and Riccardo Zecchina. Typical and atypical solutions in non-convex neural networks with discrete and continuous weights, 2023.

[27] Jean Barbier, Nicolas Macris, Mohamad Dia, and Florent Krzakala. Mutual information and optimality of approximate message-passing in random linear estimation. *IEEE Transactions on Information Theory*, 66(7):4270–4303, 2020.

[28] David Gamarnik. The overlap gap property: A topological barrier to optimizing over random structures. *Proceedings of the National Academy of Sciences*, 118(41):e2108492118, 2021.

[29] Carlo Baldassi, Fabrizio Pittorino, and Riccardo Zecchina. Shaping the learning landscape in neural networks around wide flat minima. *Proceedings of the National Academy of Sciences*, 117(1):161–170, 2020.

[30] Werner Krauth and Marc Mézard. Storage capacity of memory networks with binary couplings. *Journal de Physique*, 50(20):3057–3066, 1989.

[31] Ahmed El Alaoui and David Gamarnik. Hardness of sampling solutions from the symmetric binary perceptron. *arXiv preprint arXiv:2407.16627*, 2024.

[32] Alfredo Braunstein and Riccardo Zecchina. Learning by message passing in networks of discrete synapses. *Physical review letters*, 96(3):030201, 2006.

[33] Carlo Baldassi and Alfredo Braunstein. A max-sum algorithm for training discrete neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2015(8):P08008, aug 2015.

[34] Carlo Baldassi, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Local entropy as a measure for sampling solutions in constraint satisfaction problems. *Journal of Statistical Mechanics: Theory and Experiment*, 2016(2):023301, feb 2016.

[35] Benjamin Aubin, Will Perkins, and Lenka Zdeborová. Storage capacity in symmetric binary perceptrons. *Journal of Physics A: Mathematical and Theoretical*, 52(29):294003, 2019.

Appendix

4.A Computation of the stability distribution

In this section, we perform the computation of the stabilities distribution in the perceptron problem with an arbitrary potential $U(\cdot)$. Given an N-dimensional weight vector $\boldsymbol{w}$, representing a solution of the perceptron problem with a margin κ, we consider the stability distribution $P(s) = \frac{1}{M}\sum_{\mu}\delta(s - s^{\mu})$, where $s^{\mu} = \frac{\langle \boldsymbol{w}, \boldsymbol{x^{\mu}}\rangle}{\sqrt{N}},\ 1 \leq \mu \leq M$.

Since averaging over $\boldsymbol{x^{\mu}}$ renders all the coordinates of $P(s)$ equivalent, we will consider, without loss of generality, a stability distribution w.r.t the first input, i.e. $P(s^1)$.

We define the probability of the stability to be equal s, averaging over all the weight vectors representing the solutions to the problem and the inputs $\{\boldsymbol{x}^{\mu}\}_{\mu}$

$$P(s) = \left\langle \frac{\int P(d\boldsymbol{w}) \prod_{\mu=1}^{M} \Theta\left(\frac{1}{\sqrt{N}}\langle \boldsymbol{w}, \boldsymbol{x^{\mu}}\rangle - \kappa\right) \exp\left(-\frac{1}{T}U(\frac{1}{\sqrt{N}}\langle \boldsymbol{w}, \boldsymbol{x^{\mu}}\rangle - \kappa)\right) \delta\left(\frac{1}{\sqrt{N}}\langle \boldsymbol{w}, \boldsymbol{x^{1}}\rangle - s\right)}{\int P(d\boldsymbol{w}) \prod_{\mu=1}^{M} \Theta\left(\frac{1}{\sqrt{N}}\langle \boldsymbol{w}, \boldsymbol{x^{\mu}}\rangle - \kappa\right) \exp\left(-\frac{1}{T}U(\frac{1}{\sqrt{N}}\langle \boldsymbol{w}, \boldsymbol{x^{\mu}}\rangle - \kappa)\right)} \right\rangle . \tag{4.113}$$

Now we introduce replicas rewriting the denominator as $Z^{-1} = \lim_{n\to 0} Z^{n-1}$, subsequently taking the $n \to 0$ limit

$$P(s) = \lim_{n\to 0} \left\langle \int \prod_{a} P(d\boldsymbol{w}_a) \prod_{\mu,a} \Theta\left(\frac{1}{\sqrt{N}}\langle \boldsymbol{w}_a, \boldsymbol{x}^{\mu}\rangle - \kappa\right) e^{\left(-\frac{1}{T}U\left(\frac{1}{\sqrt{N}}\langle \boldsymbol{w}_a, \boldsymbol{x}^{\mu}\rangle - \kappa\right)\right)} \delta\left(\frac{1}{\sqrt{N}}\langle \boldsymbol{w}, \boldsymbol{x}^{1}\rangle - s\right) \right\rangle_{x} . \tag{4.114}$$

Introducing the $\delta-$ and $\Theta-$ integrals we obtain

$$
\begin{aligned}
P(s) = \Theta(s-\kappa) e^{-\frac{1}{T}U(s-\kappa)} \lim_{n\to 0} \int \prod_{a=1}^{n} P(d\boldsymbol{w}_a) \int \frac{d\hat{\lambda}_{11}}{2\pi} \int_{\kappa}^{\infty} \prod_{a>1} \frac{d\lambda_{1a}}{2\pi} e^{-\frac{1}{T}U(\lambda_{1a}-\kappa)} \int d\hat{\lambda}_{1a} \\
\times \int_{\kappa}^{\infty} \prod_{a,\mu>1} \frac{d\lambda_{\mu a}}{2\pi} e^{-\frac{1}{T}U(\lambda_{\mu a}-\kappa)} \int \prod_{a,\mu>1} d\hat{\lambda}_{\mu a} \exp\left(is\hat{\lambda}_{11} + i\sum_{a>1} \lambda_{1a}\hat{\lambda}_{1a} + i \sum_{\mu>1,a} \lambda_{\mu a}\hat{\lambda}_{\mu a} \right) \\
\times \langle \exp\left(-\frac{i}{\sqrt{N}} \sum_{a,\mu} \hat{\lambda}_{\mu a} \langle \boldsymbol{w}_a, \boldsymbol{x}^{\boldsymbol{\mu}} \rangle \right) \rangle_x .
\end{aligned}
\tag{4.115}
$$

Averaging over $\boldsymbol{x}^{\mu}$ and introducing the following order parameters and their conjugates

$$
q_{ab} \equiv \frac{1}{N} \sum_i w_{ia} w_{ib} \,, \qquad k_a \equiv \frac{1}{N} \sum_i (w_{ia})^2, \tag{4.116}
$$

$$
\begin{aligned}
P(s) = \Theta(s-\kappa) e^{-\frac{1}{T}U(s-\kappa)} \lim_{n\to 0} \int \prod_a \frac{d\hat{k}_a}{4\pi} \int \prod_{a<b} \frac{dq_{ab} d\hat{q}_{ab}}{2\pi/N} \int \frac{d\hat{\lambda}_{11}}{2\pi} \int_{\kappa}^{\infty} \prod_{a>1} \frac{d\lambda_{1a}}{2\pi} \int d\hat{\lambda}_{1a} \\
\times \exp\left(is\hat{\lambda}_{11} + i\sum_{a>1} \lambda_{1a}\hat{\lambda}_{1a} - \frac{1}{2}\sum_a (\hat{\lambda}_{1a})^2 - \frac{1}{2}\sum_{a,b} \hat{\lambda}_{1a}\hat{\lambda}_{1b} q_{ab} \right) \\
\times \exp\left(\frac{iN}{2} \sum_a \hat{k}_a + iN \sum_{a<b} q_{ab}\hat{q}_{ab} + N G_S(\hat{k}^2, \hat{q}_{ab}) + (\alpha N - 1) G_E(q_{ab}) \right).
\end{aligned}
\tag{4.117}
$$

where the entropic and energetic parts are respectively

$$
G_S(\hat{k}_a, \hat{q}_{ab}) = \ln \int \prod_a \frac{d\boldsymbol{w}_a}{\sqrt{2\pi e}} \exp\left(-\frac{i}{2} \sum_a \hat{k}_a (\boldsymbol{w}_a)^2 - i \sum_{a<b} \hat{q}_{ab} \boldsymbol{w}_a \boldsymbol{w}_b \right), \tag{4.118a}
$$

$$
G_E(q_{ab}) = \ln \int_k^{\infty} \prod_a d\lambda_a e^{-\frac{1}{T}U(\lambda_a - \kappa)} \int \prod_a \frac{d\hat{\lambda}_a}{2\pi} \prod_a \exp(i \sum_a \lambda_a \hat{\lambda}_a - \frac{1}{2}\sum_a (\hat{\lambda}_a)^2 - \frac{1}{2}\sum_{a,b} \hat{\lambda}_a \hat{\lambda}_b q_{ab}). \tag{4.118b}
$$

Imposing replica symmetry assumption

$$
q_{ab} = q \qquad \hat{q}_{ab} = \hat{q} \qquad \hat{k}^a = \hat{k}, \tag{4.119}
$$

Taking the remaining integrals and the $n \to 0$ limit we recover the distribution of the stabilities, e.g. in case of the spherical perceptron and $U(\cdot) = 0$ and $\kappa = 0$ we get

$$P(s) = \Theta(s-\kappa)\frac{1}{\sqrt{2\pi(1-q)}}\int Dt \exp\left(-\frac{(s-\sqrt{q}t)^2}{2(1-q)}\right)\left[H\left(\frac{\kappa-\sqrt{q}t}{\sqrt{1-q}}\right)\right]^{-1}. \tag{4.120}$$

while for the binary perceptron with the potential $U(\cdot)$, temperature T and $\kappa = 0$

$$P(s) = \Theta(s-\kappa)e^{-\frac{1}{T}U(s-\kappa)}\frac{1}{\sqrt{2\pi(1-q)}}\int Dt \exp\left(-\frac{(s-\sqrt{q}t)^2}{2(1-q)}\right) \times \left[\tilde{H}_{1-q}(\sqrt{q}t)\right]^{-1}. \tag{4.121}$$

where $\tilde{H}$ has been defined in (4.88).

The distribution of the stabilities is reported in Fig.10 for $\kappa = -2.1$ and several values of $\alpha \in [5, 20, 80]$.

4.B Replica computation for the CE model

For the CE model the replica computation can be done in the following way:

$$\mathbb{E}\tilde{Z}_t = \mathbb{E}_X \int \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) \prod_{\mu a} e^{-\beta \frac{\log\left(1+e^{-2\gamma \sum_i \frac{\boldsymbol{w}_{ai}\boldsymbol{x}_i^\mu}{\sqrt{N}}}\right)}{\gamma}} \tag{4.122}$$

$$= \mathbb{E}_X \int \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) \prod_{\mu a} \frac{du_a^\mu d\hat{u}_a^\mu}{2\pi} \prod_{\mu a} e^{-\beta \frac{\log\left(1+e^{-2\gamma u_a^\mu}\right)}{\gamma}} e^{-i\sum_{a,\mu} u_a^\mu \hat{u}_a^\mu + i\sum_i \sum_{a,\mu} \hat{u}_a^\mu \frac{\boldsymbol{w}_{ai}\boldsymbol{x}_i^\mu}{\sqrt{N}}} \tag{4.123}$$

$$= \int \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) \prod_{\mu a} \frac{du_a^\mu d\hat{u}_a^\mu}{2\pi} \prod_{\mu a} e^{-\beta \frac{\log\left(1+e^{-2\gamma u_a^\mu}\right)}{\gamma}} e^{-i\sum_{a,\mu} u_a^\mu \hat{u}_a^\mu - \frac{1}{2N}\sum_{i\mu}\left(\sum_a \hat{u}_a^\mu \boldsymbol{w}_{ai}\right)^2} \tag{4.124}$$

$$= \int \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) \prod_{\mu a} \frac{du_a^\mu d\hat{u}_a^\mu}{2\pi} \prod_{\mu a} e^{-\beta \frac{\log\left(1+e^{-2\gamma u_a^\mu}\right)}{\gamma}} e^{-i\sum_{a,\mu} u_a^\mu \hat{u}_a^\mu - \frac{1}{2N}\sum_{i\mu}\left(\sum_{ab} \hat{u}_a^\mu \boldsymbol{w}_{ai} \hat{u}_b^\mu \boldsymbol{w}_{bi}\right)} \tag{4.125}$$

Let's now define the overlap term

$$q_{ab} = \frac{1}{N}\sum_i \boldsymbol{w}_{ai}\boldsymbol{w}_{bi}$$

Now making the change of variables

$$\phi = \int \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) \prod_{a\leq b} \frac{dq_{ab}d\hat{q}_{ab}}{2\pi} \prod_{\mu a} \frac{du_a^\mu d\hat{u}_a^\mu}{2\pi} \prod_{\mu a} e^{-\beta \frac{\log\left(1+e^{-2\gamma u_a^\mu}\right)}{\gamma}} \times \tag{4.126}$$

$$\times e^{-i\sum_{a,\mu} u_a^\mu \hat{u}_a^\mu - \frac{1}{2}\sum_\mu \left(\sum_{ab} \hat{u}_a^\mu \hat{u}_b^\mu q_{ab}\right) + \frac{1}{2N}\sum_{ab}\sum_i \boldsymbol{w}_{ai}\boldsymbol{w}_{bi}\hat{q}_{ab} - \frac{1}{2}N\sum_{ab}\hat{q}_{ab}q_{ab}} \tag{4.127}$$

$$= \int \prod_{a\leq b} \frac{dq_{ab}d\hat{q}_{ab}}{2\pi} e^{N(G_S+G_I+\alpha G_E)} \tag{4.128}$$

where

$$G_E = \log \int \prod_a \frac{du_a d\hat{u}_a}{2\pi} \prod_a e^{-\beta \frac{\log\left(1+e^{-2\gamma u_a}\right)}{\gamma}} e^{-i\sum_a u_a\hat{u}_a - \frac{1}{2}\sum_{ab}\hat{u}_a\hat{u}_b q_{ab}} \tag{4.129}$$

$$G_S = \log \int \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) e^{\frac{1}{2}\sum_{ab}\boldsymbol{w}_a\boldsymbol{w}_b\hat{q}_{ab}} \tag{4.130}$$

$$G_I = \frac{1}{2}\sum_{ab}\hat{q}_{ab}q_{ab} \tag{4.131}$$

The RS ansatz is

$$q_{ab} = \begin{cases} q_d = 1 & a = b \\ q_0 & a \neq b \end{cases}$$

The Energetic term is the following

$$G_E = \log \int \prod_a \frac{du_a d\hat{u}_a}{2\pi} \prod_a e^{-\beta \frac{\log\left(1+e^{-2\gamma u_a}\right)}{\gamma}} e^{-i\sum_a u_a\hat{u}_a - \frac{1}{2}\sum_{ab}\hat{u}_a\hat{u}_b q_{ab}} \tag{4.132}$$

$$= \log \int \prod_a \frac{du_a d\hat{u}_a}{2\pi} \prod_a \left(1+e^{-2\gamma u_a}\right)^{-\frac{\beta}{\gamma}} e^{-i\sum_a u_a\hat{u}_a - \frac{1}{2}\sum_{ab}\hat{u}_a\hat{u}_b q_{ab}} \tag{4.133}$$

$$= \log \int \prod_a \frac{du_a d\hat{u}_a}{2\pi} \prod_a \left(1+e^{-2\gamma u_a}\right)^{-\frac{\beta}{\gamma}} e^{-i\sum_a u_a\hat{u}_a - \frac{q_d-q_0}{2}\sum_a(\hat{u}_a)^2 - \frac{q_0}{2}\left(\sum_a \hat{u}_a\right)^2} \tag{4.134}$$

$$= \log \int Dz \prod_a \frac{du_a d\hat{u}_a}{2\pi} \prod_a \left(1+e^{-2\gamma u_a}\right)^{-\frac{\beta}{\gamma}} e^{-i\sum_a u_a\hat{u}_a - \frac{q_d-q_0}{2}\sum_a(\hat{u}_a)^2 - iz\sqrt{q_0}\left(\sum_a \hat{u}_a\right)} \tag{4.135}$$

Taking the limit for $n \to 0$,

$$\mathcal{G}_E = \int Dz \log \left[\int \frac{du d\hat{u}}{2\pi} \left(1 + e^{-2\gamma u}\right)^{-\frac{\beta}{\gamma}} e^{-i\hat{u}\left(u + z\sqrt{q_0}\right) - \frac{q_d - q_0}{2}(\hat{u})^2} \right] \tag{4.136}$$

$$= \int Dz \log \left[\int \frac{du}{\sqrt{2\pi}\sqrt{q_d - q_0}} \left(1 + e^{-2\gamma u}\right)^{-\frac{\beta}{\gamma}} e^{-\frac{\left(u + z\sqrt{q_0}\right)^2}{2(q_d - q_0)}} \right] \tag{4.137}$$

$$= \int Dz \log \left[\int Du \left(1 + e^{-2\gamma\left(u\sqrt{q_d - q_0} - z\sqrt{q_0}\right)}\right)^{-\frac{\beta}{\gamma}} \right] \tag{4.138}$$

The entropic term can be expressed as

$$G_S = \log \int \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) e^{\frac{1}{2}\sum_{ab} w_a w_b \hat{q}_{ab}} \tag{4.139}$$

$$= \log \int \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) e^{\frac{\hat{q}_d - \hat{q}_0}{2}\sum_a w_a^2 + \frac{\hat{q}_0}{2}\left(\sum_a w_a\right)^2} \tag{4.140}$$

$$= \log \int Dx \prod_{a=1}^{n} P(\mathrm{d}\boldsymbol{w}_a) e^{\frac{\hat{q}_d - \hat{q}_0}{2}\sum_a w_a^2 + x\sqrt{\hat{q}_0}\left(\sum_a w_a\right)} \tag{4.141}$$

Taking the limit for $n \to 0$

$$\mathcal{G}_S = \int Dx \log \int P(\mathrm{d}\boldsymbol{w}) e^{\frac{\hat{q}_d - \hat{q}_0}{2} w^2 + x\sqrt{\hat{q}_0} w} = \frac{\hat{q}_d - \hat{q}_0}{2} + \int Dx \log\cosh\left(x\sqrt{\hat{q}_0}\right) \tag{4.142}$$

Finally, the interaction term

$$G_I = \frac{1}{2}\sum_{ab} \hat{q}_{ab} q_{ab} = \frac{n}{2}\hat{q}_d q_d + \frac{n\,(n-1)}{2}\hat{q}_0 q_0 \tag{4.143}$$

Taking the limit for $n \to 0$

$$\mathcal{G}_I = \frac{1}{2}\hat{q}_d q_d - \frac{1}{2}\hat{q}_0 q_0$$

4.C Replica computation for Binary perception tunable measure.

In this appendix, we present the replica computation for the general form of the tunable measure. The computation follows the steps performed for the cross-entropy loss measure, adding a Θ-function:

$$G_E = \log \int \prod_\alpha \frac{d\lambda_\alpha d\hat{\lambda}_\alpha}{2\pi} \prod_a \frac{du_a d\hat{u}_a}{2\pi} \prod_\alpha \Theta\left(\lambda_\alpha\right) e^{-\beta U(\lambda_\alpha)} \prod_a \Theta\left(u_a\right) e^{-\beta U(u_a)} \\ \times e^{-i\sum_\alpha \hat{\lambda}_\alpha \lambda_\alpha - i\sum_a \hat{u}_a u_a - \frac{1}{2}\sum_{\alpha\beta} \hat{\lambda}_\alpha \hat{\lambda}_\beta r_{\alpha\beta} - \frac{1}{2}\sum_{ab} \hat{u}_a \hat{u}_b q_{ab} - \sum_{\alpha a} \hat{\lambda}_\alpha \hat{u}_a p_{\alpha a}},$$

where $U(x)$ is a chosen potential. Following the same steps as in the computation for the spherical perceptron, we obtain:

$$G_E = \int Dz D\gamma \frac{\tilde{H}\left(-\frac{\gamma\sqrt{r_0}+z\sqrt{q_0-r_0}}{\sqrt{1-q_0}}\right)}{\tilde{H}\left(-\frac{\gamma\sqrt{r_0}}{\sqrt{1-r_0}}\right)} \log \tilde{H}\left(-\frac{z\sqrt{q_0-r_0}+\gamma\sqrt{r_0}}{\sqrt{1-q_0}}\right),$$

where

$$\tilde{H}(x) = \int_x^\infty Dy\, e^{-\beta U(y-x)}$$

is a generalized version of the standard complementary error function.

4.C.1 Other plots for τ-annealing

For the τ-annealing scheme described in Section 4.7, we show in Figure 8 additional experiments with different values of the number of MC sweeps, eventually also scaling as $O(N)$. Similar experiments but with temperature T annealing from 1 to 0, linearly in the number of sweeps and on the potential $U(s) = -s\Theta(-s)$ are presented in Figure

9. It is important to notice that T-annealing on $U(s) = -s\Theta(-s)$ fails at large N also when scaling the number of sweeps as N, while the τ-annealing reaches an algorithmics threshold of $\alpha_{\text{alg}} \approx 0.55$.

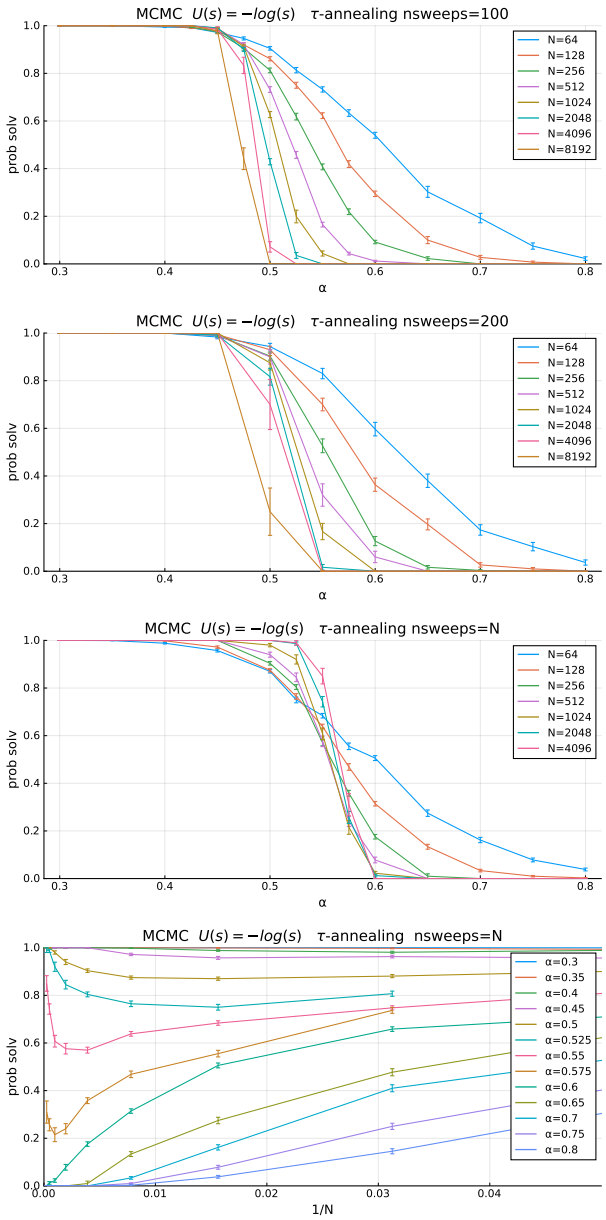

Figure 8: Probability of finding a solution for the τ-annealed MCMC scheme in the Binary Perceptron, with $T = 0.5$.

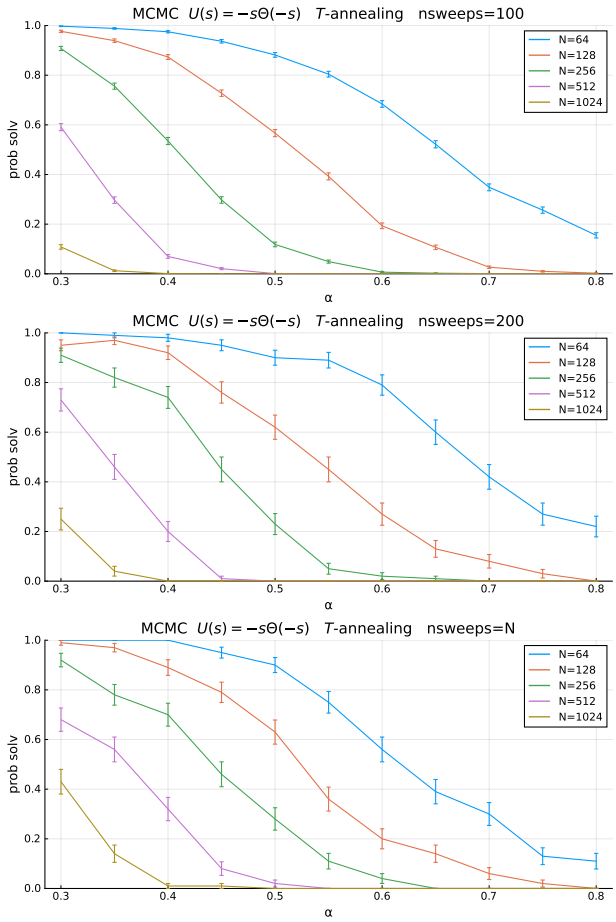

Figure 9: Probability of finding a solution for the T-annealed MCMC in the Binary Perceptron with potential $U(s) = -s\Theta(-s)$.

4.D Limiting behavior of the free entropy derivative

In this section, we report the analysis of $\frac{\phi_t(q)}{q}$ as $q \to 1$. We set $q = 1 - \epsilon$ and expand for small ϵ, leading to the results reported in 4.6.

The starting expression is:

$$\phi_t(q, \hat{q}) = -\frac{1}{2}(\hat{r}_d + \hat{q}q) + \hat{r}r + \frac{1}{2}tq + \mathcal{G}_S(\hat{q}) + \alpha\,\mathcal{G}_E(q) \tag{4.144}$$

and its derivative is:

$$\frac{d\phi_t}{dq} = \frac{t - \hat{q}}{2} + \alpha\frac{d\mathcal{G}_E(q)}{dq} \tag{4.145}$$

We consider the two terms separately.

4.D.1 Interaction term

We start from the first (interaction) term. The dependency of $\hat{q}$ on ϵ can be determined from the saddle point equations. The spherical and binary perceptron models need to be considered separately, since the relevant equation involves $\mathcal{G}_S$, which differs between the two.

Spherical Perceptron

In the spherical case, the saddle point equation for q reads:

$$q = 2\frac{\partial \mathcal{G}_S}{\partial \hat{q}} \tag{4.146}$$

$$= -\frac{\partial}{\partial \hat{q}}\left(\frac{1}{2}\log(\hat{q}-\hat{r}_d) - \frac{1}{2(\hat{q}-\hat{r}_d)}\left(\hat{q} + 2\hat{r}\frac{(\hat{q}-\hat{r})}{(\hat{r}-\hat{r}_d)} + \frac{(\hat{q}-\hat{r})^2}{(\hat{r}-\hat{r}_d)}\left(\frac{\hat{r}}{\hat{r}-\hat{r}_d}+1\right)\right)\right), \tag{4.147}$$

$$= -\frac{1}{\hat{q}-\hat{r}_d} + \frac{2\hat{r}-\hat{r}_d}{(\hat{r}-\hat{r}_d)^2}. \tag{4.148}$$

We can now use the saddle point for the reference system, see Section 4.5.1, to rewrite the expression of q, concluding that $\hat{q}$ diverges as

$$\hat{q} = \frac{1}{\epsilon} + \hat{r}_d \tag{4.149}$$

with c constant. This shows that the part of the derivative due to the interaction term diverges as $O(\epsilon^{-1})$. The sign of the derivative is negative, meaning that the free entropy has a local minimum at $q = 1$ unless this is overcome by the energetic contribution (which will not be the case with $U(s) = 0$, as we will show below).

Binary perceptron

An analogous computation for the binary perceptron case, already presented in [14], results in:

$$\frac{1+q}{2} = \frac{\partial}{\partial q} \sum_{w^*=\pm 1} \int Dz\, D\gamma\, \frac{e^{\gamma\sqrt{\hat{r}}w^*}}{2\cosh(\gamma\sqrt{\hat{r}})} \log\left(2\cosh\left(z\sqrt{\hat{q}-\hat{r}} + \gamma\sqrt{\hat{r}} + (\hat{q}-\hat{r})\, w^*\right)\right) \tag{4.150}$$

$$= \sum_{w^*=\pm 1} \int Dz\, D\gamma\, \frac{e^{\gamma\sqrt{\hat{r}}w^*}}{2\cosh(\gamma\sqrt{\hat{r}})} \tanh\left(z\sqrt{\hat{q}-\hat{r}} + \gamma\sqrt{\hat{r}} + (\hat{q}-\hat{r})\, w^*\right)\left(\frac{z}{\sqrt{\hat{q}-\hat{r}}} + w^*\right) \tag{4.151}$$

As for the spherical case, $\hat{q} \to \infty$ as $q \to 1$. We can then rewrite the previous expression as

$$1 - \frac{1}{2}\epsilon = 1 - e^{-2(\hat{q}-\hat{r})} \int Dz\, D\gamma\, \frac{e^{\gamma\sqrt{\hat{r}}} e^{-2\left(z\sqrt{\hat{q}-\hat{r}}+\gamma\sqrt{\hat{r}}\right)} - e^{-\gamma\sqrt{\hat{r}}} e^{2\left(z\sqrt{\hat{q}-\hat{r}}+\gamma\sqrt{\hat{r}}\right)}}{\cosh(\gamma\sqrt{\hat{r}})} \tag{4.152}$$

and thus

$$\hat{q} = -\frac{1}{2}\log\left(\frac{\epsilon}{2}\right) + C \tag{4.153}$$

$$C = \hat{r} + \frac{1}{2}\log\left(\int Dz\, D\gamma\, \frac{e^{\gamma\sqrt{\hat{r}}} e^{-2\left(z\sqrt{\hat{q}-\hat{r}}+\gamma\sqrt{\hat{r}}\right)} + e^{-\gamma\sqrt{\hat{r}}} e^{2\left(z\sqrt{\hat{q}-\hat{r}}+\gamma\sqrt{\hat{r}}\right)}}{\cosh(\gamma\sqrt{\hat{r}})}\right). \tag{4.154}$$

This shows that, as $q \to 1$, the part of the derivative that comes from the interaction term also diverges in the binary case, but this time logarithmically, as $O(\log\epsilon)$. The sign of the derivative is again negative, meaning that the free entropy has a local minimum at $q = 1$ unless the effect is overcome by the energetic contribution. As we shall show, the energetic contribution does indeed overcome this effect and produce a local maximum at $q = 1$ unless the potential $U(s)$ diverges at the origin.

4.D.2 Energetic term

Next, we compute the derivative of the energetic term. This does not depend on the model. We expand $\mathcal{G}_E(1-\epsilon)$ for small ϵ and then study $\mathcal{G}_E(1-\epsilon)/\epsilon$, keeping in mind that:

$$\frac{d\mathcal{G}_E}{dq} = -\frac{d\mathcal{G}_E}{d\epsilon} \tag{4.155}$$

We perform the expansion trying to keep the setting general with respect to the potential U. Indeed, our only starting assumption is that U is twice differentiable over $(0,\infty)$.

The energetic term, after some manipulations, and using $q = 1-\epsilon$, can be written as:

$$\mathcal{G}_E = \int D\gamma Dz\, \frac{\mathcal{N}(z,\gamma)}{\mathcal{D}(\gamma)} \tag{4.156}$$

$$\mathcal{D}(\gamma) = \int_{\frac{\gamma\sqrt{r}}{\sqrt{1-r}}}^{\infty} D\lambda e^{-\beta U\left(\lambda\sqrt{1-r}-\gamma\sqrt{r}\right)} \tag{4.157}$$

$$\mathcal{N}(z,\gamma) = \int_{\frac{\gamma\sqrt{r}}{\sqrt{1-r}}}^{\infty} D\lambda e^{-\beta U\left(\lambda\sqrt{1-r}-\gamma\sqrt{r}\right)} \mathcal{A}(z,\gamma,\lambda) \tag{4.158}$$

$$\mathcal{A}(z,\gamma,\lambda) = \log \int_{-\frac{a(z,\gamma,\lambda,u)}{\sqrt{\epsilon}}}^{\infty} Du\, e^{-\beta U\left(u\sqrt{\epsilon}+a(z,\gamma,\lambda,u)\right)} \tag{4.159}$$

$$a(z,\gamma,\lambda,u) = -\sqrt{r}\gamma - z\sqrt{1-r-\epsilon}\sqrt{\frac{\epsilon}{1-r}} + \sqrt{1-r}\lambda - \frac{\epsilon\lambda}{\sqrt{1-r}} \tag{4.160}$$

We first perform two change of variables, in both $\mathcal{N}$ and $\mathcal{D}$, from λ to $\rho = -\gamma\sqrt{r} + \lambda\sqrt{1-r}$:

$$\mathcal{D}(\gamma) = \frac{1}{\sqrt{1-r}} \int_0^{\infty} \mathrm{d}\rho\, G\left(\frac{\rho+\gamma\sqrt{r}}{\sqrt{1-r}}\right) e^{-\beta U(\rho)} \tag{4.161}$$

$$\mathcal{N}(z,\gamma) = \frac{1}{\sqrt{1-r}} \int_0^{\infty} \mathrm{d}\rho\, G\left(\frac{\rho+\gamma\sqrt{r}}{\sqrt{1-r}}\right) e^{-\beta U(\rho)} \tilde{\mathcal{A}}(z,\gamma,\rho) \tag{4.162}$$

$$\tilde{\mathcal{A}}(z,\gamma,\rho) = \log \int_{-\frac{a'(z,\gamma,\rho,u)}{\sqrt{\epsilon}}}^{\infty} Du\, e^{-\beta U\left(u\sqrt{\epsilon}+\tilde{a}(z,\gamma,\lambda,u)\right)} \tag{4.163}$$

$$\tilde{a}(z,\gamma,\rho,u) = \rho - z\sqrt{\epsilon}\sqrt{1-\frac{\epsilon}{1-r}} - \epsilon\frac{\rho+\gamma\sqrt{r}}{\sqrt{1-r}} \tag{4.164}$$

where $G(x) = \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}$. Next, we expand the potential U inside the expression of $\tilde{\mathcal{A}}$ for small ϵ up to the first order:

$$U\left(u\sqrt{\epsilon} + \tilde{a}(z,\gamma,\lambda,u)\right) \approx U(\rho) + \sqrt{\epsilon}U'(\rho)(u-z) - \epsilon\frac{U'(\rho)}{1-r}\left(\sqrt{r}\gamma + \rho\right) + \frac{1}{2}\epsilon(u-z)^2 U''(\rho) \tag{4.165}$$

This allows us to compute the integral $\tilde{\mathcal{A}}$, which converges as long as $1 + U''(\rho)\beta\epsilon \geq 0$ (which is obviously always true if U is convex). After some explicit integration and further expansions, we arrive at:

$$\tilde{\mathcal{A}}(z,\gamma,\rho) \approx \mathcal{A}_0 + \sqrt{\epsilon}\mathcal{A}_{1/2} + \epsilon\mathcal{A}_1 + \log H\left(z - \frac{\rho}{\sqrt{\epsilon}}\right) \tag{4.166}$$

$$\mathcal{A}_0 = -\beta U(\rho) \tag{4.167}$$

$$\mathcal{A}_{1/2} = \beta z U'(\rho) \tag{4.168}$$

$$\mathcal{A}_1 = \beta\left(-\frac{U''(\rho)}{2}\left(1+z^2\right) + \frac{\beta\left(U'(\rho)\right)^2}{2} + \frac{U'(\rho)\left(\rho+\gamma\sqrt{r}\right)}{1-r}\right) \tag{4.169}$$

At this point, we can observe that the term $\mathcal{A}_{1/2}$ does not contribute because it gets canceled by the integral over z. Indeed, with further manipulations and changes of variables we arrive at:

$$\mathcal{G}_E = \int D\gamma \frac{\mathcal{N}'(\gamma)}{\mathcal{D}'(\gamma)} \tag{4.170}$$

$$\tilde{\mathcal{D}}(\gamma) = \int_0^\infty \mathrm{d}\rho\, G\left(\frac{\rho+\gamma\sqrt{r}}{\sqrt{1-r}}\right) e^{-\beta U(\rho)} \tag{4.171}$$

$$\tilde{\mathcal{N}}(\gamma) = \int_0^\infty \mathrm{d}\rho\, G\left(\frac{\rho+\gamma\sqrt{r}}{\sqrt{1-r}}\right) e^{-\beta U(\rho)}\left[\mathcal{A}_0 + \epsilon\tilde{\mathcal{A}}_1\right] + \tag{4.172}$$
$$+ \sqrt{\epsilon}\int_0^\infty \mathrm{d}\tau\, G\left(\frac{\tau\epsilon+\gamma\sqrt{r}}{\sqrt{1-r}}\right) e^{-\beta U(\tau\epsilon)} \int Dz \log H(z-\tau)$$

$$\mathcal{A}_0 = -\beta U(\rho) \tag{4.173}$$

$$\tilde{\mathcal{A}}_1 = \beta\left(-U''(\rho) + \frac{\beta\left(U'(\rho)\right)^2}{2} + \frac{U'(\rho)\left(\rho+\gamma\sqrt{r}\right)}{1-r}\right) \tag{4.174}$$

We can observe that $\mathcal{G}_E$ receives three contributions from $\tilde{\mathcal{N}}$. The ones from $\mathcal{A}_0$ and $\tilde{\mathcal{A}}_1$ are straightforward: they represent a zero-order and a first-order contributions. In particular the first one gives us the limiting value $\mathcal{G}_E^0 = \lim_{\epsilon\to 0} \mathcal{G}_E$. The first-order term then gives a finite contribution to the derivative.

The term containing $\int Dz \log H(z-\tau)$, on the other hand, has a less obvious scaling with ϵ, depending on the form of the potential U. If U does not diverge at 0, then for small ϵ this term is $O\left(\sqrt{\epsilon}\right)$. Therefore, the derivative of $\mathcal{G}_E$ is always divergent in this case, because $\frac{d\mathcal{G}_E}{d\epsilon} = \lim_{\epsilon\to 0} \frac{\mathcal{G}_E - \mathcal{G}_E^0}{\epsilon} = O\left(\epsilon^{-1/2}\right)$. Further analysis shows that the coefficient is always negative. For the spherical case, this term is sub-dominant with respect to the one that comes from the interaction term, which is $O(\epsilon^{-1})$. For the binary case, however, the interaction term only provides a logarithmic contribution, therefore, a peak at $q = 1$ is unavoidable in this scenario.

Thus, the only way to avoid the local maximum at $q = 1$ in the binary perceptron case is to choose a potential $U(s)$ that diverges for $s \to 0$. Then the scaling of the last term with ϵ can be manipulated. For the specific choice $U(s) = -\log(s)$ we can easily see that we obtain an additional factor $\epsilon^{\beta/2}$, which makes the term $O\left(\epsilon^{\frac{\beta+1}{2}}\right)$ and the derivative $O\left(\epsilon^{\frac{\beta-1}{2}}\right)$. For $\beta > 1$, this term therefore becomes sub-dominant and it does not contribute to the derivative: we revert to the situation where the dominant contribution no longer comes from the energetic term, but from the interaction term, and it has the right sign to ensure that $q = 1$ will correspond to a local minimum rather than a maximum.

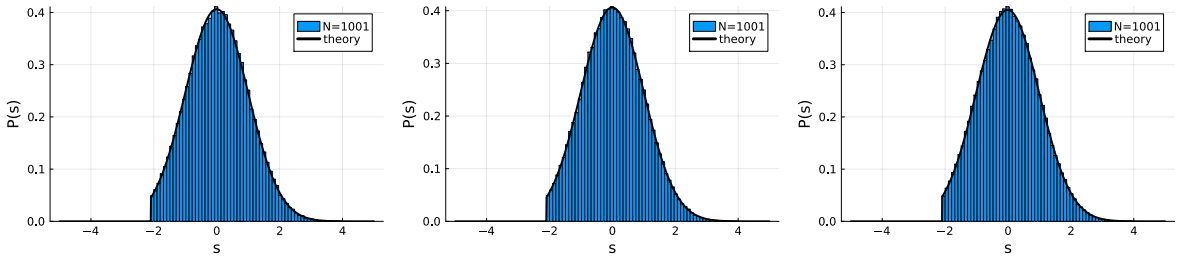

Figure 10: Distribution of the stabilities s^μ of a spherical perceptron with negative margin, where $s^\mu = \frac{\langle \boldsymbol{w}, \boldsymbol{x}^\mu \rangle}{\sqrt{N}}$. For number of variables $N = 1001$, $\kappa = -2.1$ and $\alpha = 5$ (Left), $\alpha = 20$ (Center) and $\alpha = 80$ (Right). The empirical distribution of the stabilities of the ASL samples (blue) coincides for different parameter regimes with the predicted distribution (black solid line).

Chapter 5

Stochastic Localization for Sparse Constraint Satisfaction Problems

In this chapter, we continue our analysis of the stochastic localization sampling algorithm, now applied to constraint satisfaction problems (CSPs). CSPs are a fundamental class of problems in computer science due to their simple but rich algorithmic structure.

Sparse CSPs. A CSP instance consists of a list of constraints (or clauses), each involving a subset of variables. A solution is an assignment of the variables that satisfies all constraints simultaneously. The perceptron model studied previously can be seen as a "dense" CSP over real variables where every constraint is a half-space, so that every constraint depends on the value of all variables. In contrast, we now focus on *sparse* CSPs, where each constraint involves only a *constant* number of variables (i.e., independent of the total number of variables N). Typical examples include:

1. satisfying k–sparse Boolean formulas (k–SAT);
2. satisfying systems of k–sparse linear equations over $\mathbb{F}_2$ (k–XORSAT);
3. coloring sparse graphs with q colors (q–coloring).

Just as moving from fully connected spin glasses to *diluted* ones complicates the anal-

ysis in statistical mechanics, sparse CSPs pose additional challenges. In particular, their locality invalidates the mean-field approximations that are crucial in the dense regime. For instance, the explicit replica-symmetric computation of the free energy used in the previous chapter for the perceptron model becomes more intricate: the natural order parameters are distributions over messages, i.e., functional objects [1]. Therefore, the theoretical understanding of stochastic localization for sparse CSPs requires new tools.

The planting trick. As discussed in the previous chapter, algorithmic stochastic localization is a diffusion-based method that can be employed for sampling uniformly from a known but intractable distribution. The original paper [2] focuses on the Gibbs distribution of the Sherrington–Kirkpatrick model, which is a dense model akin to the perceptron. We would now like to understand how stochastic localization performs in structurally different settings. In the related recent work [3], the authors manage to theoretically analyze stochastic localization for Ising and spherical p–spin models, rank-one matrix estimation, and the NAE–SAT problem—the latter being another example of a sparse CSP. The analysis in [3] crucially relies on the following "planting trick":

1. First, the authors argue that it is possible to plant a fixed solution into a random instance in such a way that it remains indistinguishable from an instance without a planted solution (a technique known as *quiet planting*).

2. Then, they use the fact that the analysis of the free entropy of the tilted measure simplifies considerably in the planted model.

However, one limitation of this approach is that for many sparse CSPs such as k–SAT, no method currently exists to quietly plant a solution all the way up to the key *dynamical threshold* [4]. This suggests the need for alternative techniques to analyze stochastic localization for CSPs when quiet planting is infeasible.

Our contributions. We directly address these new challenges in this chapter. We make two main contributions:

1. We empirically investigate the performance of stochastic localization as a method for approximately sampling from the solution space of sparse CSPs.

2. We develop a new framework based on the *population dynamics* method [5] to predict the success or failure of the sampling algorithm without needing to run it. Our approach succeeds even in settings where quiet planting is infeasible.

We observe that for sparse CSPs, the region where stochastic localization successfully samples solutions roughly coincides with the region where even recovering *single* solution is computationally tractable. Finally, we provide numerical evidence supporting the claim that our sampler approaches the *uniform* distribution over solutions.

Our approach is inspired by the analysis of belief propagation-guided decimation in [6]. In each iteration of the algorithm from [6], belief propagation (BP) is run on the instance to find the marginal of each variable, and a fraction of the variables is then fixed to their most probable value using the output of BP. These variables are then eliminated from the instance—a step that gives the name "decimation" to the algorithm. The authors of [6] extend the population dynamics framework [5] to determine the limitations of this decimation algorithm for sampling solutions of sparse CSPs such as k–SAT and k–XORSAT. We note that our sampling algorithm can be interpreted as a "smoothed" variant of sampling via decimation.

5.1 Background and definitions

The CSPs studied in this chapter are defined over N variables $\mathbf{x} = (x_1, \ldots, x_N)$. Each x_i takes values in a finite alphabet $\mathcal{X}$ (typically, $\mathcal{X} = \{-1, 1\}$). An instance consists of M constraints $\psi_1, \ldots, \psi_M$, where for every $a \in [M]$,

$$\psi_a : \mathcal{X}^{\partial a} \to \{0, 1\} \tag{5.1}$$

encodes whether assignments to the variables $(x_i)_{i\in\partial a}$ satisfy a particular predicate. Here, $\partial a \subseteq [N]$ denotes the set of variables appearing in the constraint a. We will deal with *sparse* constraint satisfaction problems—namely, each constraint a will involve a fixed number $|\partial a| = k$ of variables.

Random instances. In a random CSP instance, each constraint ψ_a is generated independently by selecting a random k-ary predicate (from a prescribed family) and assigning it to a randomly chosen set ∂a of k variables.

Energy function. An assignment $\mathbf{x} \in \mathcal{X}^N$ is a solution to a CSP instance if $\psi_a(\mathbf{x}_{\partial a}) = 1$ for every $a \in [M]$. Therefore, the energy (or loss) function of a sparse CSP can be written as

$$L(\mathbf{x}) = \prod_{a=1}^{M} \psi_a(\mathbf{x}_{\partial a})\,, \tag{5.2}$$

where $\mathbf{x}_{\partial a}$ is a shortcut for $(x_i)_{i\in\partial a}$.

Factor graph. We will repeatedly use the *factor graph* representation of a CSP instance. The factor graph is a bipartite graph with vertices $[N]\cup[M]$, where each variable vertex $i \in [N]$ has an edge to each constraint it belongs to, and each constraint vertex $a \in [M]$ has an edge to each variable it contains. In this representation, ∂a can be equivalently seen as the set of neighbors of a constraint vertex a, and we similarly use the notation ∂i to denote the set of neighbors of a node vertex $i \in [N]$ in the factor graph.

5.2 The sampling algorithm

Let p be (the density of) the uniform distribution over solutions to a CSP instance. Formally, p is the the uniform distribution over $\mathbf{x}^* \in \mathcal{X}^N$ achieving value $L(\mathbf{x}^*) = 1$ in Eq. (5.2). Similarly to the previous chapter, our goal will be to understand the ability of stochastic localization to sample from p.

As before, our sampling algorithm discretizes and simulates the stochastic differential equation

$$\mathrm{d}\mathbf{h}_t = \mathbf{m}_t(\mathbf{h}_t)\,\mathrm{d}t + \mathrm{d}\mathbf{b}_t\,, \tag{5.3}$$

where $(\mathbf{b}_t)_{t\geq 0}$ is a standard N-dimensional Brownian motion, and

$$\mathbf{m}_t(\mathbf{h}) = \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{h},t}}\left[\mathbf{x}\right], \tag{5.4}$$

$$p_{\mathbf{h},t}(\mathbf{x}) = \frac{1}{Z_{\mathbf{h},t}} e^{\langle \mathbf{h},\mathbf{x}\rangle - \frac{t}{2}\|\mathbf{x}\|^2} p(\mathbf{x})\,, \tag{5.5}$$

$$Z_{\mathbf{h},t} = \sum_{\mathbf{x}\in\mathcal{X}^N} e^{\langle \mathbf{h},\mathbf{x}\rangle - \frac{t}{2}\|\mathbf{x}\|^2} p(\mathbf{x})\,. \tag{5.6}$$

Recall that by the Bayesian interpretation of stochastic localization, the distribution of $\mathbf{h}_t$ admits an equivalent representation as

$$\mathbf{h}_t = t\mathbf{x}^* + \sqrt{t}\mathbf{g}\,, \tag{5.7}$$

where $(\mathbf{x}^*, \mathbf{g}) \sim p \otimes \mathcal{N}(0, \mathrm{I}_N)$.

As in the previous chapter, our approach relies on analyzing the free entropy lanscape of the tilted measure p_t at time t, which is

$$\phi_t = \lim_{N\to\infty} \frac{1}{N} \mathop{\mathbb{E}}_{\mathbf{h}} \log\left(\sum_{\mathbf{x}\in\mathcal{X}^N} e^{\langle \mathbf{h}_t,\mathbf{x}\rangle} p(\mathbf{x})\right) \tag{5.8}$$

$$= \lim_{N\to\infty} \frac{1}{N} \mathop{\mathbb{E}}_{\substack{\mathbf{x}^*\sim p \\ \mathbf{g}\sim\mathcal{N}(0,\mathrm{I}_N)}} \log\left(\sum_{\mathbf{x}\in\mathcal{X}^N} \prod_{a=1}^{M} \psi_a(\mathbf{x}_{\partial a}) \exp\left(\langle t\mathbf{x}^* + \sqrt{t}\mathbf{g}, \mathbf{x}\rangle\right)\right). \tag{5.9}$$

While we will not be able to compute ϕ_t in closed form, we will introduce a simple numerical algorithm to estimate it.

5.2.1 Computing the marginals with belief propagation

One technical difference with the previous chapter is that since the tilted measure is defined over sparse factor graphs, we will use belief propagation (instead of approximate

message passing) to compute the mean vector $\mathbf{m}_t(\mathbf{h}_t)$ at every step.

Original BP equations.

We first describe the message-passing equations used to compute the marginals in the original system at time $t = 0$. The derivation of these equations can be found for example in the classical textbook [7].

For every variable $i \in [N]$ and clause $a \in [M]$, we define a clause-to-variable message $\{\hat{\nu}_{a\to i}\}$ and a variable-to-clause message $\{\nu_{i\to a}\}$ as follows.

$$\hat{\nu}_{a\to i}(x_i) \propto \sum_{\mathbf{x}_{\partial a\setminus i}} \psi_a(\mathbf{x}_{\partial a}) \prod_{j\in\partial a\setminus i} \nu_{j\to a}(x_j)\,, \tag{5.10}$$

$$\nu_{i\to a}(x_i) \propto \prod_{b\in\partial i\setminus a} \hat{\nu}_{b\to i}(x_i)\,, \tag{5.11}$$

where we recall that $\partial a \subseteq \{1,\dots,N\}$ denotes the set of variables appearing in the a-th constraint, and $\partial i \subseteq \{1,\dots,M\}$ denotes the set of constraints to which the i-th variable belongs.

We write these equations more compactly as

$$\hat{\nu}_{a\to i} = \hat{F}\left(\{\nu_{j\to a}\}_{j\in\partial a\setminus i}; \psi_a\right)\,, \tag{5.12}$$

$$\nu_{i\to a} = F\left(\{\hat{\nu}_{b\to i}\}_{b\in\partial i\setminus a}\right)\,, \tag{5.13}$$

where ψ_a is the disorder of the constraint a.

Once the message passing iteration has converged, the marginals of the i-th variable can be computed as

$$\nu_i(x_i) \propto \prod_{a\in\partial i} \hat{\nu}_{a\to i}(x_i)\,. \tag{5.14}$$

Tilted BP equations.

To compute the marginals at time $t > 0$, we need to slightly modify the BP equations to take into account a field $\mathbf{h}$ on the variables. The above equations become

$$\hat{\nu}^{\text{tilt}}_{a\to i}(x_i) \propto \sum_{\mathbf{x}_{\partial a\setminus i}} \psi_a(\mathbf{x}_{\partial a}) \prod_{j\in\partial a\setminus i} \nu^{\text{tilt}}_{j\to a}(x_j)\,, \tag{5.15}$$

$$\nu^{\text{tilt}}_{i\to a}(x_i) \propto e^{h_i x_i} \prod_{b\in\partial i\setminus a} \hat{\nu}^{\text{tilt}}_{b\to i}(x_i)\,. \tag{5.16}$$

Again, we write these equations more compactly,

$$\hat{\nu}^{\text{tilt}}_{a\to i} = \hat{F}\left(\{\nu^{\text{tilt}}_{j\to a}\}_{j\in\partial a\setminus i}; \psi_a\right)\,, \tag{5.17}$$

$$\nu^{\text{tilt}}_{i\to a} = F\left(\{\hat{\nu}^{\text{tilt}}_{b\to i}\}_{b\in\partial i\setminus a}; h_i\right)\,. \tag{5.18}$$

5.3 Population dynamics

We now describe our framework to analyze the success/failure of stochastic localization based on the free entropy (5.9). In general, we expect the following two sources of algorithmic failure:

1. Belief propagation fails to converge at time t due to replica symmetry breaking effects. However, due to Nishimori conditions, this can happen only if the system is already RSB at time $t = 0$. That is, the tilted measure cannot be "more RSB" than the target measure.

2. The Bethe free entropy at some time t has two maxima: 1) a global "informed" one more aligned with $\mathbf{x}^*$ and 2) an "uninformed" one less aligned with $\mathbf{x}^*$. In this case, belief propagation may not provide the correct marginals because it gets trapped in the uninformed maximum.

Since failures of type (1) can be easily understood from the target measure, we focus on failures of type (2). In the asymptotic limit, we can investigate them with a *population*

dynamic algorithm.

In general, population dynamics refers to the study of how populations of interacting entities evolve over time, governed by certain probabilistic rules. In statistical physics models, the "population" consists of states that evolve over time. These states may represent different possible configurations of a system, such as the assignment of values in a spin system or the configuration of a CSP. Our main inspiration here is the population dynamic framework introduced in [6] to analyze an idealized version of belief propagation-guided decimation.

Broadcasting. Imagine that we were to generate a CSP instance and the messages corresponding to the tilted measure $p_{\mathbf{h},t}$ on a tree, starting from a single vertex and exploring the tree. We write a recursion for the statistics of the messages at depth ℓ by conditioning $\nu_{i\to a}^{\mathrm{tilt}}$ and $\hat{\nu}_{a\to i}^{\mathrm{tilt}}$ on the event $x_i^* = \tilde{x}$. Hence, at generation ℓ away from the root, we keep track of the joint distribution of the quantities $(\nu, \{\nu^{\tilde{x}}\}_{\tilde{x}\in\mathcal{X}})_\ell$ and $(\hat{\nu}, \{\hat{\nu}^{\tilde{x}}\}_{\tilde{x}\in\mathcal{X}})_\ell$.

The messages on the ℓ-th layer would follow the distributional identity

$$(\nu, \{\nu^x\}_{x\in\mathcal{X}})_\ell \sim (F(\hat{\nu}), (F(\{\hat{\nu}_c^{\tilde{x}}\}_{c\in[C]}; h_x))_{\tilde{x}\in\mathcal{X}})_{\ell-1}\,, \tag{5.19}$$

where $h_{\tilde{x}} = t\tilde{x} + \sqrt{t}g$, and C is the random residual degree of a variable. We recall that the BP update functions F have been defined in Eq. (5.12) and (5.17).

The other distributional recursion rule is

$$(\hat{\nu}, \{\hat{\nu}^{\tilde{x}}\}_{\tilde{x}\in\mathcal{X}})_\ell \sim (\hat{F}(\{\nu_i\}_{i\in[k]}; \hat{\psi}), (\hat{F}(\{\nu_i^{x_i(\tilde{x})}\}_{i\in[k]}; \hat{\psi}))_{\tilde{x}\in\mathcal{X}})_{\ell-1}\,, \tag{5.20}$$

where $\hat{\psi}$ is a freshly generated CSP constraint, and for each $\tilde{x} \in \mathcal{X}$ we sample $\{x_i(\tilde{x})\}_{i\in[k]}$ from

$$\Pr\left(x_1, \ldots, x_k \mid x_k = \tilde{x}, \psi = \hat{\psi}\right) \propto \psi(x_1, \ldots, x_k) \prod_{i=1}^{k} \nu_i(x_i)\,. \tag{5.21}$$

Population dynamics is a simple fixed point iteration that numerically computes solutions to Eq. (5.19) and Eq. (5.20). More precisely, we define populations of $\mathcal{N}$ messages: $\nu_1, \ldots, \nu_{\mathcal{N}}$ and $\hat{\nu}_1^{\tilde{x}}, \ldots, \hat{\nu}_{\mathcal{N}}^{\tilde{x}}$ for all $\tilde{x} \in \mathcal{X}$ that we make evolve over time. At every time step, we iterate the equations (5.19) and (5.20) on our population by resampling independently a new factor graph. At the end for the procedure, we output the empirical distribution over our population as a proxy for the empirical distribution of the BP messages.

In the next section, we instantiate this population dynamics framework in detail for the k–SAT problem.

5.4 The k–SAT problem

The k–SAT problem is a satisfiability problem on N Boolean variables $x_1, \ldots, x_N \in \{-1, 1\}$ in which we are given M clauses of the form:

$$(x_{i_1} \vee \ldots \vee x_{i_\ell} \vee \bar{x}_{i_\ell+1} \vee \ldots \vee \bar{x}_{i_k}), \quad \text{for } i_1, \ldots, i_k \in [N], \ell \in \{0, \ldots, k\}.$$

Intuitively, a clause is satisfied when at least one variable is aligned with its literal. The goal is to find an assignment to $x_1, \ldots, x_N$ that simultaneously satisfies all M clauses.

In general, N and M are independent quantities, however we will restrict ourselves to the case in which $M = \alpha N$.

The k–SAT problem can be represented by its factor graph. A factor node corresponding to clause a involving variables $\mathbf{x}_{\partial a}$ reads

$$\psi_a(\mathbf{x}_{\partial a}) = 1 - \prod_{i \in \partial a} \frac{1 - J_{a,i} x_i}{2} = \begin{cases} 1 & \text{if } a \text{ is satisfied by } x \\ 0 & \text{otherwise} \end{cases} \tag{5.22}$$

where $J_{a,i} = +1$ if x_i is not negated in clause a, and -1 otherwise.

5.4.1 Belief propagation equations for k–SAT

We first write down the belief propagation equations from Section 5.2.1 in the special case of k–SAT and introduce a well-chosen parametrization.

Absence of fields.

Instanciating Eq. (5.10) and (5.11) for the k–SAT problem, we get

$$\hat{\nu}_{a\to i}(x_i) \propto \sum_{\mathbf{x}_{\partial a\setminus i}} \left(1 - \prod_{j\in\partial a} \frac{1 - J_{a,j}x_j}{2}\right) \prod_{j\in\partial a\setminus i} \nu_{j\to a}(x_j) \tag{5.23}$$

Following [7], we parametrize the factor to variable messages with $\hat{\zeta}_{a\to i} = \hat{\nu}_{a\to i}(-J_{a,i})$. Therefore, we have

$$\begin{aligned}
\hat{\zeta}_{a\to i} &\propto \sum_{\mathbf{x}_{\partial a\setminus i}} \left(1 - \prod_{j\in\partial a\setminus i} \frac{1 - J_{a,j}x_j}{2}\right) \prod_{j\in\partial a\setminus i} \nu_{j\to a}(x_j) && (5.24)\\
&= 1 - \prod_{j\in\partial a\setminus i} \left(\sum_{x_j} \frac{1 - J_{a,j}x_j}{2} \nu_{j\to a}(x_j)\right) && (5.25)\\
&= 1 - \prod_{j\in\partial a\setminus i} \zeta_{j\to a}\,, && (5.26)
\end{aligned}$$

where we use the parametrization for the variable to factor message $\zeta_{i\to a} = \nu_{i\to a}(-J_{a,i})$. Taking into account the normalization, we have

$$\hat{\zeta}_{a\to i} = \frac{1 - \prod_{j\in\partial a\setminus i} \zeta_{j\to a}}{2 - \prod_{j\in\partial a\setminus i} \zeta_{j\to a}}\,. \tag{5.27}$$

We can compute similarly that for the other type of messages,

$$\nu_{i\to a}(x_i) \propto \prod_{b\in\partial i\setminus a} \hat{\nu}_{b\to i}(x_i)\,. \tag{5.28}$$

Let us define two sets of the factor nodes: those that align with the clause a on the literal for the variable i and those that do not,

$$N_{i,a}^{\pm} = \{b \in \partial i \setminus a : J_{i,a} = \pm J_{i,b}\}\,. \tag{5.29}$$

Closing the equations, we obtain

$$\zeta_{i\to a} = \frac{\prod_{b\in\partial i/a}\hat{\nu}_{b\to i}(-J_{a,i})}{\prod_{b\in\partial i\setminus a}\hat{\nu}_{b\to i}(-J_{a,i}) + \prod_{b\in\partial i\setminus a}\hat{\nu}_{b\to i}(J_{a,i})} \tag{5.30}$$

$$= \frac{\prod_{b\in N_{ia}^{+}}\hat{\zeta}_{b\to i}\prod_{b\in N_{ia}^{-}}(1-\hat{\zeta}_{b\to i})}{\prod_{b\in N_{ia}^{+}}\hat{\zeta}_{b\to i}\prod_{b\in N_{ia}^{-}}(1-\hat{\zeta}_{b\to i}) + \prod_{b\in N_{ia}^{-}}\hat{\zeta}_{b\to i}\prod_{b\in N_{ia}^{+}}(1-\hat{\zeta}_{b\to i})}\,. \tag{5.31}$$

Presence of fields.

Taking into consideration the effect of an external field $\mathbf{h}$, the variable-to-factor message becomes

$$\zeta_{i\to a} = \frac{e^{-h_i J_{a,i}}\prod_{b\in\partial i\setminus a}\hat{\nu}_{b\to i}(-J_{a,i})}{e^{-h_i J_{a,i}}\prod_{b\in\partial i\setminus a}\hat{\nu}_{b\to i}(-J_{a,i}) + e^{h_i J_{a,i}}\prod_{b\in\partial i\setminus a}\hat{\nu}_{b\to i}(J_{a,i})} \tag{5.32}$$

$$= \frac{e^{-h_i J_{a,i}}\prod_{b\in N_{ia}^{+}}\hat{\zeta}_{b\to i}\prod_{b\in N_{ia}^{-}}(1-\hat{\zeta}_{b\to i})}{e^{-h_i J_{a,i}}\prod_{b\in N_{ia}^{+}}\hat{\zeta}_{b\to i}\prod_{b\in N_{ia}^{-}}(1-\hat{\zeta}_{b\to i}) + e^{h_i J_{a,i}}\prod_{b\in N_{ia}^{-}}\hat{\zeta}_{b\to i}\prod_{b\in N_{ia}^{+}}(1-\hat{\zeta}_{b\to i})}\,. \tag{5.33}$$

5.4.2 Experimental results for k–SAT

We first describe the experimental results of running the stochastic localization sampling algorithm on k–SAT for $k = 3$ and $k = 4$.

We run stochastic localization across various system sizes. We depict on Fig. 1 the number of BP iterations before convergence, and on Fig. 2 the probability that our algorithm succeeds in sampling a solution to a random instance, as a function of the clause-to-variable ratio $\alpha = M/N$. We observe a sharp transition of the success probability of sampling around a threshold $\alpha_{\mathrm{SL}}(k)$ in the asymptotic limit $N \to \infty$.

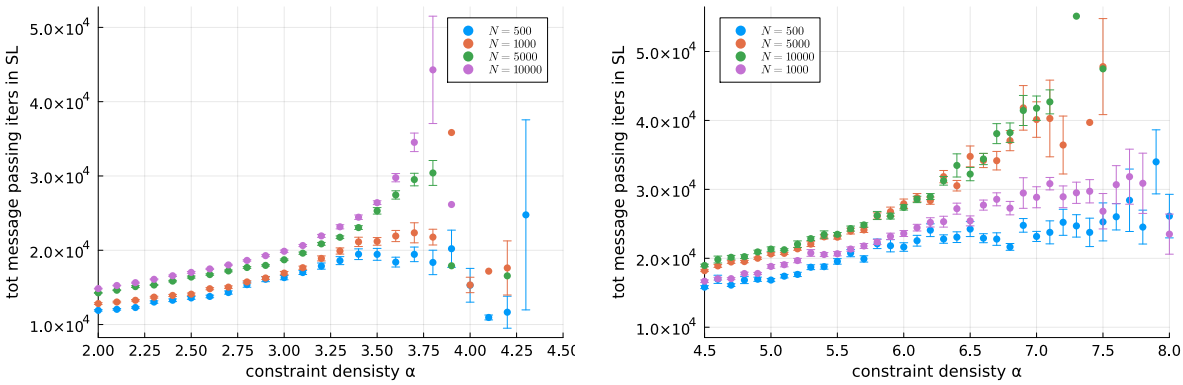

Figure 1: Number of BP iterations before convergence as a function of α.
Left: $k = 3$. **Right:** $k = 4$.

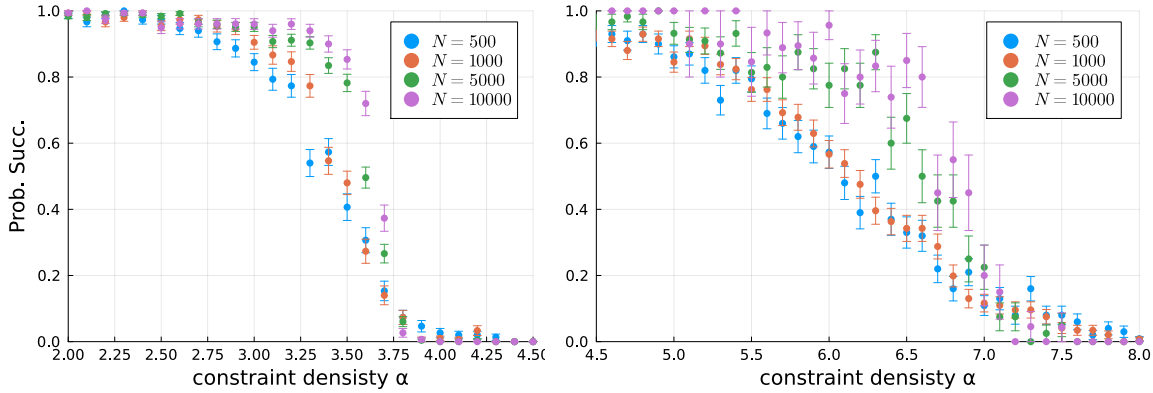

Figure 2: Probability of sampling a solution as a function of α.
Left: $k = 3$. **Right:** $k = 4$.

5.4.3 Population dynamics for k–SAT

To instantiate population dynamics in this setting, we find it more convenient to work with the messages $u_{a\to i}$ and $v_{i\to a}$ in the following alternative parametrization (also used in [6]):

$$\hat{\nu}_{a\to i}(x_i) = \frac{1 - J_{a,i}x_i \tanh u_{a\to i}}{2}\,, \qquad \nu_{i\to a}(x_i) = \frac{1 - J_{a,i}x_i \tanh v_{i\to a}}{2}\,.$$

In presence of the fields h, this yields the following BP equations:

$$u_{a\to i} = \frac{1}{2}\ln\left(1 - \prod_{j\in\partial a\setminus i}\frac{1+\tanh(v_{j\to a})}{2}\right) \tag{5.34}$$

$$v_{i\to a} = -J_i^a h_i + \sum_{b\in\partial_+ i(a)} u_{b\to i} - \sum_{b\in\partial_- i(a)} u_{b\to i} \tag{5.35}$$

where $\partial_+ i(a) = \{b \in \partial i\setminus a \mid J_i^b = J_i^a\}, \partial_- i(a) = \{b \in \partial i\setminus a \mid J_i^b = -J_i^a\}$ and $h_i = t\tilde{x}_i + \sqrt{t}g$, where $g \sim \mathcal{N}(0,1)$, and $\tilde{x}_i \in \{-1,1\}$ is the latent solution to the given k–SAT instance.

Then, the fixed point solutions to Eq. (5.19) and Eq. (5.20) become in our setting

$$(v, v^+, v^-) \overset{d}{=} \left(\sum_{i=1}^{\ell^+} u_i - \sum_{i=1}^{\ell^-} r_i,\ \sum_{i=1}^{\ell^+} u_i^+ - \sum_{i=1}^{\ell^-} r_i^- - (t-\sqrt{t}g),\ \sum_{i=1}^{\ell^+} u_i^- - \sum_{i=1}^{\ell^-} r_i^+ + (t+\sqrt{t}g)\right), \tag{5.36}$$

and

$$(u, u^+, u^-) \overset{d}{=} \left(f(v_1,\dots,v_{k-1}), f(v_1^{\tilde{x}_1^+},\ \dots, v_{k-1}^{\tilde{x}_{k-1}^+}),\ f(v_1^{\tilde{x}_1^-},\ \dots, v_{k-1}^{\tilde{x}_{k-1}^-})\right), \tag{5.37}$$

where:

- $\ell^+, \ell^- \sim \text{Poisson}(\alpha k/2)$ are random realizations of the degree of a variable (respectively the number of clauses where the variable appears with the same or a different literal than in the reference solution).
- (v_i, v_i^+, v_i^-) for $i \in [k-1]$ are i.i.d copies of (v, v^+, v^-).
- (u_i, u_i^+, u_i^-) for $i \in [\ell^+]$ and (r_i, r_i^+, r_i^-) for $i \in [\ell^-]$ are i.i.d copies of (u, u^+, u^-).
- $g \sim \mathcal{N}(0,1)$.
- The values $(\tilde{x}_1^+, \dots \tilde{x}_{k-1}^+)$ and $(\tilde{x}_1^-, \dots \tilde{x}_{k-1}^-)$ for u^+ and u^- are sampled respectively as

$$\Pr\left(\tilde{x}_1^+, \dots \tilde{x}_{k-1}^+ \mid v_1, \dots v_{k-1}\right) = \prod_{i=1}^{k-1}\frac{1-\tilde{x}_i^+\tanh(v_i)}{2}, \tag{5.38}$$

and

$$\Pr\left(\tilde{x}_1^-, \dots \tilde{x}_{k-1}^- \mid v_1, \dots v_{k-1}\right) = \frac{1 - \mathbb{I}(\tilde{x}_1^- = \cdots = \tilde{x}_{k-1}^- = -1)}{1 - \prod_{i=1}^{k-1} \frac{1+\tanh v_i}{2}} \prod_{i=1}^{k-1} \frac{1 - \tilde{x}_i^- \tanh(v_i)}{2} \,. \tag{5.39}$$

- $f(v_1, \dots, v_{k-1}) = \frac{1}{2} \ln\left(1 - \prod_{i=1}^{k-1} \frac{1+\tanh(v_i)}{2}\right)$ is the BP update function from Eq. (5.34).

Our goal would be to find a fixed point solution to the equations (5.36) and (5.37).

We proceed by introducing two populations of $\mathcal{N}$ triplets for the u and v messages, respectively $\{(v_i, v_i^+, v_i^-)\}_{i=1}^{\mathcal{N}}$ and $\{(u_i, u_i^+, u_i^-)\}_{i=1}^{\mathcal{N}}$. To initialize $\{v_i\}_{i=1}^{\mathcal{N}}$, we run the fixed point iteration for k–SAT at time $t = 0$, i.e., without any field. We depict on Algorithm 6 our final population dynamics algorithm for k–SAT.

We argue that the empirical distribution of the messages in stochastic localization are well-captured by population dynamics. We compare in Fig. 3 the empirical distributions of (v^-, v^+, u^-, u^+) for some choice of parameters t, k, α. We observe more generally that the empirical distributions of population dynamics and BP messages match in the region where sampling is possible.

We note that a principled way to predict whether sampling is possible would be to check whether the following two initializations of the messages in population dynamics lead to two different distributional fixed points of Eqs. (5.36) and (5.37):

1. I_0 with $v_i^+ = v_i^- = v_i$ for all $i \in [\mathcal{N}]$.

2. I_1 with $v_i^+ = -\infty$ and $v_i^- = +\infty$ for all $i \in [\mathcal{N}]$.

5.5 The planted q–coloring problem

We now complement our findings for k–SAT with experimental results on the q–coloring problem. Here the problem instance is a graph on N vertices, and the goal is to find a *proper* coloring of the vertices of the graph with q colors, i.e., a coloring such that all

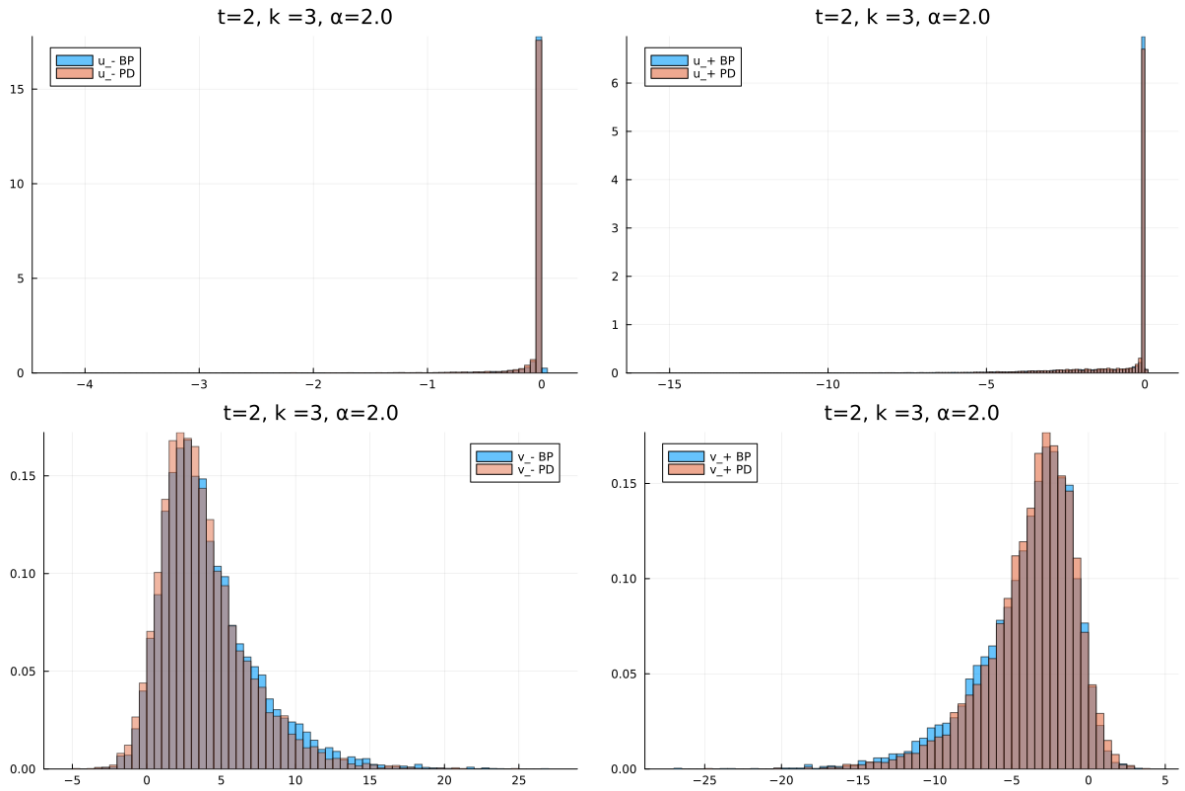

Figure 3: An example of empirical (normalized) distributions of the messages (v^+, v^-, u^+, u^-) for a k–SAT instance. The blue histogram is the empirical distribution observed by running stochastic localization and belief propagation. The orange histogram is simulated with population dynamics. We observe empirically that the distributions are identical across the regime where the sampling of solutions is possible.

adjacent vertices are colored differently. This corresponds to a sparse CSP with $k = 2$, $\mathcal{X} = \{1, \dots, q\}$, and all constraints are of the form

$$\psi(x, y) = \mathbb{I}\{x \neq y\} \ ,$$

for $x, y \in \{1, \dots, q\}$. We study random instances of this problem, where every edge of the graph is sampled independently with probability c/N. Note that the constraint density coincides up to a factor 2 with the average degree c of the graph.

In the *planted* q–coloring model, a coloring of the vertices is first sampled, and then edges of the graphs are drawn at random between vertices of different colors (i.e., conditionally on the planted coloring being proper). As studied in [8], the planted q–coloring model is known for its hard-easy-hard phase transition for retrieving the solutions, and until the second hard phase, the model is known to be contiguous to a fully random instance. This implies in particular that their solution space share the same geometry. In Fig. 4, we depict an example of this phenomenon for $q = 5$.

5.5.1 Belief propagation equations for q–coloring

In this section, we define formally the q–coloring problem, and derive the BP equations for it.

The q–coloring model.

Let $G = (V, E)$ be a graph with vertex set $V = [N]$. We denote by $s_i \in [q]$ the color assigned to vertex $i \in [N]$. The number of proper colorings of the graph G is

$$Z_N = \sum_{s_1, \dots, s_N \in [q]} \prod_{\{i,j\} \in E} (1 - \delta_{s_i, s_j}) \, . \tag{5.42}$$

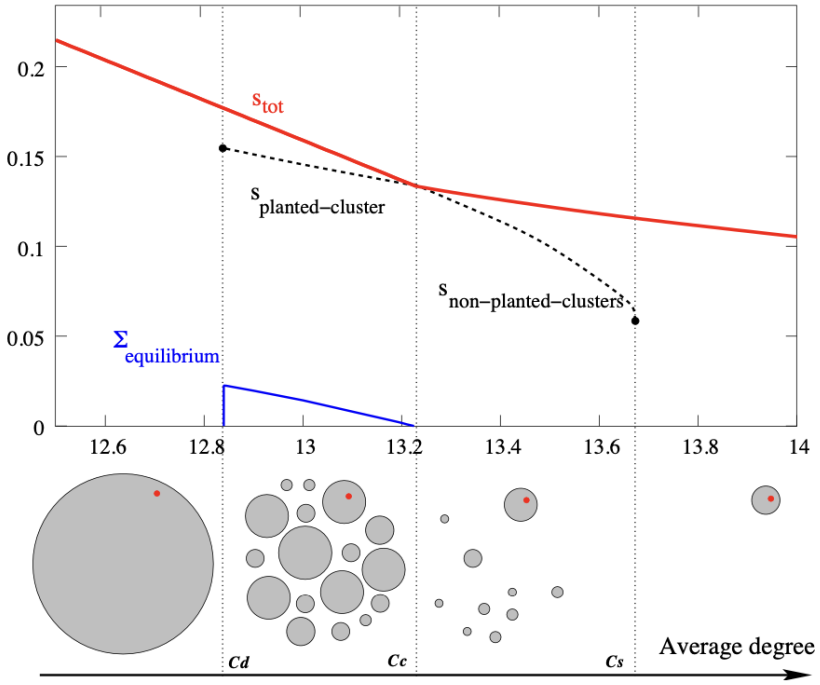

Figure 4: Phase diagram for $q = 5$ of the planted q–coloring problem.
Bottom: Clustering phase transitions. At the dynamical threshold c_d, the space of solutions shatters into exponentially many clusters, one of which corresponding to the planted solution. Beyond the condensation threshold c_c, the planted cluster contains more solutions than all other clusters together. At the satisfiability threshold c_s, the last non-planted cluster disappears.
Top: Total entropy s_{tot} with the sub-dominant part (dashed). The equilibrium complexity $\Sigma_{\text{equilibrium}}$ is the logarithm of the number of dominant clusters.
Source of the diagram: [8]

Algorithm 6 Population Dynamics for k–SAT

Input: k (number of variables in a clause), $\alpha = M/N$ (constraint density), L (number of iterations of population dynamics), $\mathcal{N}$ (population size)

Initialize $(v_1, \dots, v_{\mathcal{N}})$ by running population dynamics to find a fixed point of

$$v \stackrel{d}{=} \sum_{i=1}^{\ell^+} u_i - \sum_{i=1}^{\ell^-} r_i \tag{5.40}$$

$$u \stackrel{d}{=} f(v_1, \dots, v_{k-1}) \tag{5.41}$$

where $(v_1, \dots, v_{k-1})$ are i.i.d copies of v, $(u_1, \dots u_{\ell^+}, r_1, \dots, r_{\ell^-})$ are i.i.d copies of u and $\ell^+, \ell^- \stackrel{\text{i.i.d}}{\sim} \text{Poisson}(\alpha k/2)$.

for iter from 1 to L **do**

for j from 1 to $\mathcal{N}$ **do**

$u_j \leftarrow f(v_{i_1}, \dots v_{i_{k-1}})$, where $(i_1, \dots i_{k-1}) \stackrel{\text{i.i.d}}{\sim} \text{Unif}([\mathcal{N}])$
$u_j^+ \leftarrow f(v_{i_1}^{\tilde{x}_1}, \dots v_{i_{k-1}}^{\tilde{x}_1})$, where $(\tilde{x}_1, \dots \tilde{x}_{k-1})$ is sampled from (5.38) with $v_{i_1}, \dots, v_{i_k}$
$u_j^- \leftarrow f(v_{i_1}^{\tilde{x}_1}, \dots v_{i_{k-1}}^{\tilde{x}_1})$, where $(\tilde{x}_1, \dots \tilde{x}_{k-1})$ is sampled from (5.39) with $v_{i_1}, \dots, v_{i_k}$

end for

for j from 1 to $\mathcal{N}$ **do**

$$\begin{aligned} v_j &\leftarrow \sum_{n=1}^{\ell^+} u_{i_n^+} - \sum_{n=1}^{\ell^-} u_{i_n^-} \\ v_j^+ &\leftarrow \sum_{n=1}^{\ell^+} u_{i_n^+}^+ - \sum_{n=1}^{\ell^-} u_{i_n^-}^- - (t - \sqrt{t}g) \\ v_j^- &\leftarrow \sum_{n=1}^{\ell^+} u_{i_n^+}^- - \sum_{n=1}^{\ell^-} u_{i_n^-}^+ + (t + \sqrt{t}g) \end{aligned}$$

where $\ell^+, \ell^- \stackrel{i.i.d}{\sim} \text{Poisson}(\alpha k/2)$, $(i_1^+, \dots i_{\ell^+}^+, i_1^-, \dots i_{\ell^-}^-) \stackrel{i.i.d}{\sim} \text{Unif}([\mathcal{N}])$, and $g \sim \mathcal{N}(0,1)$

end for
end for

Output: $\{u_i, u_i^+, u_i^-, v_i, v_i^+, v_i^-\}_{i=1}^{\mathcal{N}}$

In particular, G is q–colorable if and only if $Z_N \geq 1$. In that case, the uniform distribution over all proper colorings is:

$$\Pr(s_1, \ldots, s_N) = \frac{1}{Z_N} \prod_{\{i,j\} \in E} (1 - \delta_{s_i, s_j}) \,. \tag{5.43}$$

More generally, we consider the relaxed Gibbs distribution parametrized by an inverse temperature β,

$$\Pr(s_1, \ldots, s_N) = \frac{1}{Z_N(\beta)} \prod_{\{i,j\} \in E} e^{-\beta \delta_{s_i, s_j}} \,. \tag{5.44}$$

In the limit $\beta \to \infty$, we recover the case of proper colorings (with strict constraints).

BP equations in the absence of fields.

In the factor graph representation, each factor node corresponds to an edge i, j of the original graph and carries the "smoothed" predicate $\psi_{i,j}(s_i, s_j) = e^{-\beta \delta_{s_i, s_j}}$. From the generic BP formulation, we obtain the following message-passing update equations:

$$\nu_{j \to \{i,j\}}(s_j) = \frac{1}{Z_{j \to \{i,j\}}} \prod_{\{k,j\} \in \partial j \setminus \{i,j\}} \hat{\nu}_{\{k,j\} \to j}(s_j) \,, \tag{5.45}$$

and

$$\begin{aligned}
\hat{\nu}_{\{i,j\} \to i}(s_j) &= \frac{1}{Z_{\{i,j\} \to i}} \sum_{s_j \in [q]} e^{-\beta \delta_{s_i, s_j}} \nu_{j \to \{i,j\}}(s_j) \\
&= \frac{1}{Z_{\{i,j\} \to i}} \left(\sum_{s_j \neq s_i} \nu_{j \to \{i,j\}}(s_j) + e^{-\beta} \nu_{j \to \{i,j\}}(s_i) \right) \\
&= \frac{1}{Z_{\{i,j\} \to i}} \left[1 - (1 - e^{-\beta}) \nu_{j \to \{i,j\}}(s_i) \right] \,.
\end{aligned}$$

Substituting the expression for factor-to-variable messages into the variable-to-factor update equation, we obtain:

$$\nu_{j\to\{i,j\}}(s_j) = \frac{1}{Z_{j\to\{i,j\}}\prod_{\{k,j\}\in\partial j\setminus\{i,j\}} Z_{\{k,j\}\to j}} \prod_{\{k,j\}\in\partial j\setminus\{i,j\}} \left[1-(1-e^{-\beta})\nu_{k\to\{k,j\}}(s_j)\right] . \tag{5.46}$$

Finally, we rewrite the final equation with the neighborhood notations *in the original graph*,

$$\nu_{j\to i}(s_j) = \frac{1}{Z_{j\to i}} \prod_{k\in\partial j\setminus i} \left[1-(1-e^{-\beta})\nu_{k\to j}(s_j)\right] , \tag{5.47}$$

where we introduced

$$\frac{1}{Z_{j\to i}} = Z_{j\to\{i,j\}} \prod_{\{k,j\}\in\partial j\setminus\{i,j\}} Z_{\{k,j\}\to j} . \tag{5.48}$$

BP equations in the presence of fields.

In the presence of an external field $\mathbf{h} = (h_i(s))_{s\in[q]}$ acting on each variable i, the BP update for variable-to-factor messages is modified as follows:

$$\nu_{j\to\{i,j\}}(s_j) = \frac{e^{h_j(s_j)}}{Z_{j\to\{i,j\}}} \prod_{\{k,j\}\in\partial j\setminus\{i,j\}} \hat{\nu}_{\{k,j\}\to j}(s_j) . \tag{5.49}$$

Using a similar derivation as before, we obtain

$$\nu_{j\to i}(s_j) = \frac{e^{h_j(s_j)}}{Z_{j\to i}} \prod_{k\in\partial j\setminus i} \left[1-(1-e^{-\beta})\nu_{k\to j}(s_j)\right] . \tag{5.50}$$

5.5.2 Experimental results for q–coloring

Our experiments consist of running stochastic localization on the planted q–coloring model across different system sizes. We report results for $q = 5$ on Fig. 5. Overall, we observe a very similar easy-hard-easy pattern for successfully sampling a solution, similarly to the known phase diagram from Fig. 4. This suggests that the region where stochastic localization is effective coincides with the tractable region for recovering even

a single proper coloring.

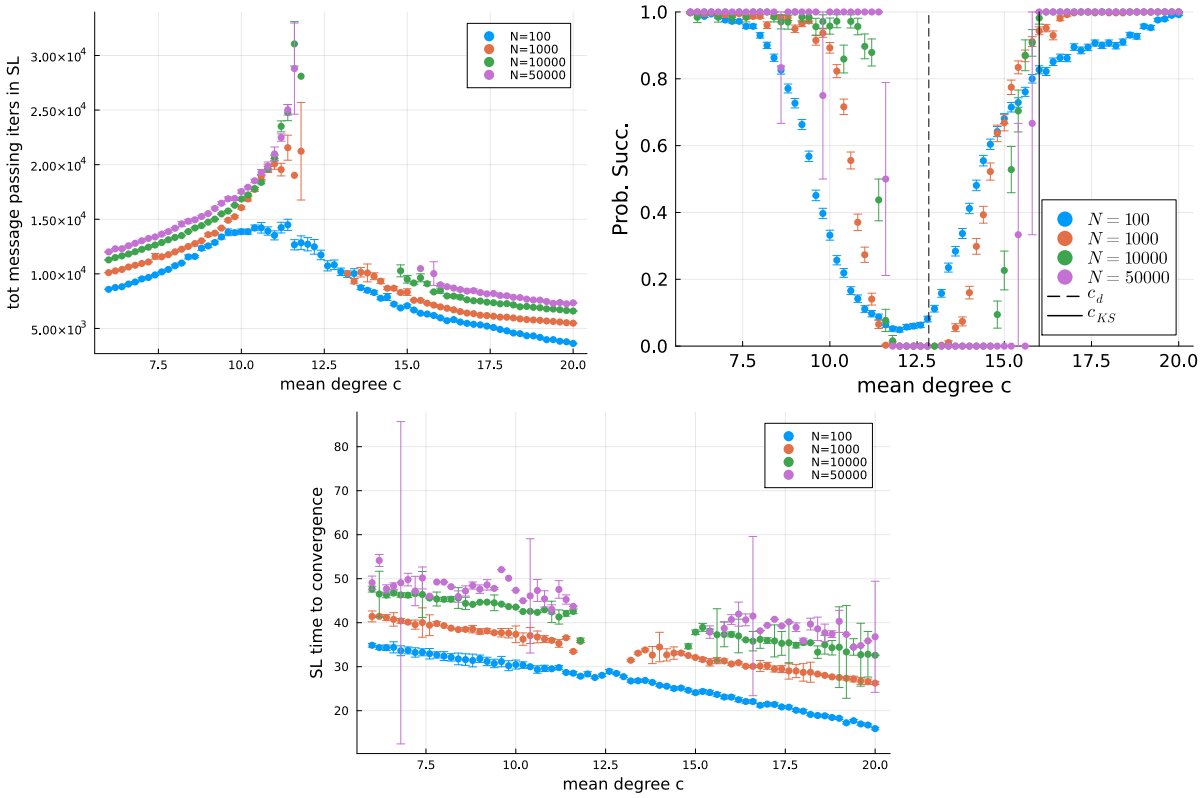

Figure 5: Experimental results for q–coloring for $q = 5$. The average degree c parametrizes the constraint density for this problem. The dashed c_d and c_{KS} lines are respectively the dynamical and the Kesten-Stigum thresholds.
Top left: number of BP iterations before convergence.
Top right: probability that the algorithm finds a proper coloring.
Bottom: number of iterations of stochastic localization before finding a solution (in case of convergence).

5.6 The k–XORSAT problem

In this section, we continue our experimental investigation of stochastic localization of sparse CSPs, now focusing on the k–XORSAT problem. This problem has a unique feature: it admits a simple algebraic algorithm—Gaussian elimination—that efficiently finds a solution to any instance. This makes k–XORSAT a tractable problem, unlike most CSPs such as k–SAT, which are NP-complete. Moreover, this algorithm can also be modified to efficiently sample a uniformly random solution to an arbitrary instance. Therefore, k–XORSAT is an excellent testbed for our sampling problem. We will use it

to verify experimentally that our sampler indeed samples from the *uniform* distribution over all satisfying assignments.

5.6.1 Belief propagation equations for k–XORSAT

In this section, we formally define the k–XORSAT problem and derive its associated belief propagation (BP) equations.

The k–XORSAT problem is a CSP on Boolean variables $\mathbf{x} = (x_1, \ldots, x_N) \in \{-1, 1\}^N$, in which we are given M clauses of the form

$$x_{i_1} \cdot \ldots \cdot x_{i_k} = J_a \,, \quad \text{for } i_1, \ldots, i_k \in [N], J_a \in \{-1, 1\} \,. \tag{5.51}$$

Again, we will restrict ourselves to the case where $M = \alpha N$.

The k–XORSAT problem also has a factor graph representation, where the nodes represent the clauses, and predicates are of the form

$$\psi_a(\mathbf{x}_{\partial a}) = \mathbb{I}\left\{ \prod_{i \in \partial a} x_i = J_a \right\} \,, \tag{5.52}$$

where $J_a \in \{-1, +1\}$. As before, an assignment $\mathbf{x}$ to the variables is a solution if it satisfies all clauses simultaneously.

BP equations.

We reparametrize for convenience the generic BP messages in the following way,

$$\nu_{i \to a}(x_i) = \frac{1 - x_i \tanh(v_{i \to a})}{2} \,, \tag{5.53}$$

$$\hat{\nu}_{a \to i}(x_i) = \frac{1 - x_i \tanh(u_{a \to i})}{2} \,. \tag{5.54}$$

In this parametrization, we derived the BP equations and obtained

$$v_{i\to a} = h_i + \frac{1}{2}\sum_{b\in\partial i\setminus a}\log(1+e^{2u_{b\to i}}) - \frac{1}{2}\sum_{b\in\partial i\setminus a}\log(1+e^{-2u_{b\to i}})\,, \tag{5.55}$$

$$u_{a\to i} = \operatorname{arctanh}\left(J_a\prod_{j\in\partial a\setminus i}\tanh\left(v_{j\to a}\right)\right)\,. \tag{5.56}$$

5.6.2 Experimental results for k–XORSAT

We ran similar experiments for k–XORSAT that we did for k–SAT in Sec. 5.4.2. We observe once again a phase transition around a threshold $\alpha_{\rm SL}(k)$ of the probability that stochastic localization successfully samples solution to random linear systems.

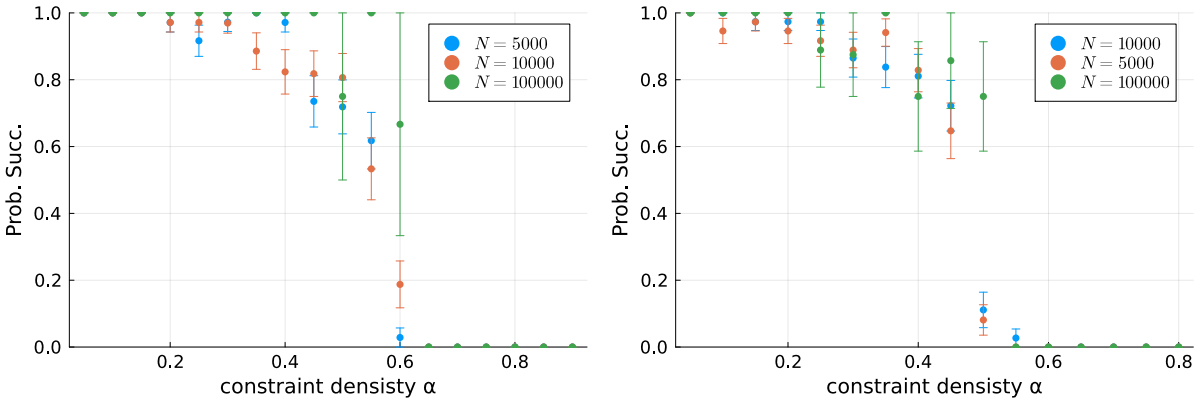

Figure 6: Probability that stochastic localization successfully samples a solution to a k–XORSAT instance, as a function of α.
Left: $k = 3$. **Right:** $k = 4$.

5.6.3 Uniformity of sampling

Among all constraint satisfaction problems considered in this chapter, k–XORSAT is the only one that admits a polynomial-time algorithm for finding solutions in the worst case. This can be achieved using Gaussian elimination over $\mathbb{F}_2$. In contrast, all other problems discussed before (coloring, k–SAT, ...) are NP-hard.

In fact, the special algebraic structure of XORSAT also gives a simple description of how a random solution looks like. The set of solutions to an XORSAT instance forms an

affine subspace, so given an arbitrary solution x^*, a random solution can be decomposed as $x^* + z$, where z is uniformly distributed in the kernel of the coefficient matrix of the instance. This suggests a simple and principled test for uniformity: projecting each sampled solution onto the kernel basis and checking whether the coefficients are uniformly distributed.

We reproduce the experimental results in Fig. 7. We note that the distribution of the projection coefficients indeed appears to be close to uniform.

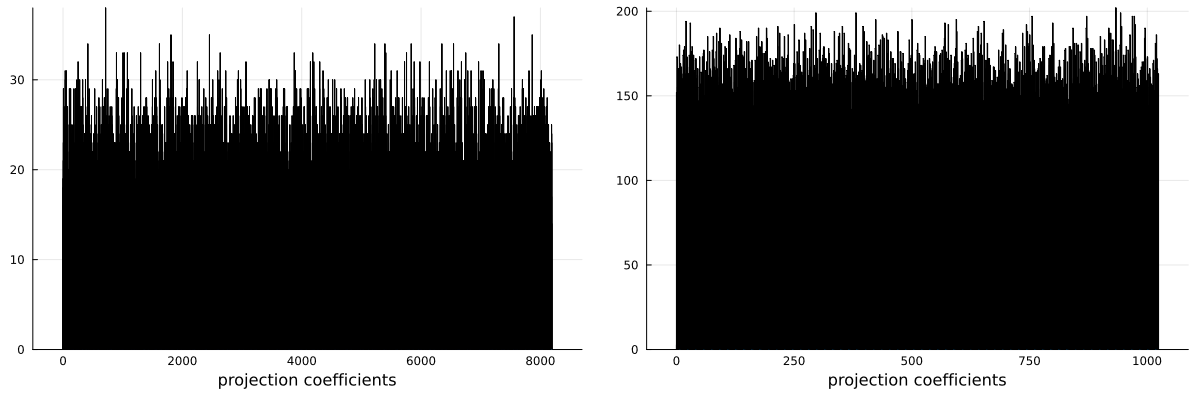

Figure 7: Projection coefficients onto the kernel basis of sampled solutions to 3-XORSAT instances. The plots represent the distribution of the base-10 digits of the vector of projection coefficients .
Left: $\alpha = 0.1, N = 80K$, subset of 13 bits. **Right:** $\alpha = 0.5, N = 8K$, subset of 10 bits.

5.7 Conclusion

In this chapter, we extended the analysis of the stochastic localization algorithm to sparse constraint satisfaction problems (CSPs). We addressed two main directions: first, we generalized the population dynamics framework to probe the limitations of the algorithm; second, we conducted extensive experiments on random instances of k–SAT, k–XORSAT, and q–coloring. Our results suggest that the region in which stochastic localization can efficiently sample solutions coincides closely with the region where finding a single solution remains computationally tractable.

In particular, the k–XORSAT problem served as a valuable testbed: it allows for exact uniform sampling via Gaussian elimination, and our numerical experiments confirm that

stochastic localization reproduces the uniform distribution of solutions in this case.

These findings offer further insight into the capabilities of diffusion-based sampling methods for CSPs, and lay the groundwork for a more systematic exploration of their limitations. As a direction for future work, it would be interesting to express the free entropy of the tilted measures in terms of the fixed-point messages obtained from population dynamics. This perspective could provide a complementary analytical approach to assessing the algorithm's performance and phase transitions.

References

[1] Rémi Monasson and Riccardo Zecchina. Statistical mechanics of the random K-satisfiability model. *Physical Review E*, 56:1357–1370, 1997.

[2] Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Sampling from the Sherrington-Kirkpatrick Gibbs measure via algorithmic stochastic localization. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 323–334. IEEE, 2022.

[3] Davide Ghio, Yatin Dandi, Florent Krzakala, and Lenka Zdeborová. Sampling with flows, diffusion, and autoregressive neural networks from a spin-glass perspective. *Proceedings of the National Academy of Sciences*, 121(27):e2311810121, 2024.

[4] Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Reweighted Belief Propagation and Quiet Planting for Random k-SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, 8(3-4):149–171, 2014.

[5] Marc Mézard and Giorgio Parisi. The Bethe lattice spin glass revisited. *The European Physical Journal B: Condensed Matter and Complex Systems*, 20(2):217–233, 2001.

[6] Federico Ricci-Tersenghi and Guilhem Semerjian. On the cavity method for decimated random constraint satisfaction problems and the analysis of belief propagation guided decimation algorithms. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(09):P09001, 2009.

[7] Marc Mezard and Andrea Montanari. *Information, physics, and computation*. Oxford University Press, 2009.

[8] Florent Krzakala and Lenka Zdeborová. Hiding quiet solutions in random constraint satisfaction problems. *Physical review letters*, 102(23):238701, 2009.