UNIVERSITÀ
DI PARMA

UNIVERSITÀ DEGLI STUDI DI PARMA

DOTTORATO DI RICERCA IN

FISICA

CICLO XXXVI

Unraveling the role of topology in Complex Long Range Systems and Deep Neural Networks

Coordinatore:

Chiar.mo Prof. Stefano Carretta

Supervisor:

Prof. Alessandro Vezzani

Co-supervisor:

Chiar.ma Prof. Raffaella Burioni

Dott. Pietro Rotondo

Candidato:

Dott. Riccardo Aiudi

Anni Accademici 2020/2021 – 2022/2023

A Gloria.

"La cosa più incomprensibile dell'universo è che esso sia comprensibile."

Acknowledgements

Prima di tutto, desidero esprimere la mia più profonda gratitudine ai miei due professori, Raffaella ed Alessandro, che mi hanno affiancato durante questi tre anni. Grazie alla vostra esperienza, siete riusciti ad accompagnarmi nel regno della Meccanica Statistica, facendomene comprendere le potenzialità e, soprattutto, la bellezza. Devo inoltre ringraziarvi per avermi dato l'opportunità di seguire le mie personali aspirazioni e desideri, sempre pronti a fornire un consiglio esperto. Desidero poi ringraziare tutti i colleghi che ho incontrato, in particolar modo Rosalba e Pietro, per avermi introdotto allo studio teorico delle reti neurali e per aver allietato gli ultimi mesi del dottorato con la vostra energia. Una menzione speciale va indubbiamente a Paolo, che fin dai primi giorni si è rivelato un compagno insostituibile nei momenti di pausa, nelle avventure e nelle disavventure. Hai sempre occupato un posto speciale nelle mie giornate lavorative e i tuoi preziosi consigli mi sono stati d'aiuto più di una volta.

Al di fuori dell'ambito accademico, desidero ovviamente ringraziare tutti i miei amici sparsi per l'Italia e l'Europa: senza di voi, ogni percorso sarebbe più buio e noioso. Un ringraziamento particolare va alla mia famiglia e, in modo speciale, ai miei genitori, Hector e Michela, per avermi sempre supportato durante tutta la mia carriera accademica. Senza il vostro aiuto, non avrei mai potuto scrivere questi ringraziamenti e questo lavoro è tanto mio quanto vostro. Vorrei ringraziare anche tutta la mia famiglia acquisita di Pisa, per avermi sempre fatto sentire a casa e per essermi stata d'aiuto nei momenti di difficoltà.

Infine, il ringraziamento più grande va alla mia dolce metà, Gloria. Sei sempre stata la mia roccia salda e grazie al tuo amore ho sempre trovato l'energia per superare tutti i momenti bui. Per questo e altri mille motivi, dedico a te in particolare questo lavoro.

Parma, 2023 Riccardo Aiudi

List of Publications

Chapter 3 is based on the following publication:

- **R. Aiudi**, R. Burioni and A. Vezzani, *Critical dynamics of long-range models on dynamical Lévy lattices*, Physical Review B, 107(22), June 2023

Chapter 6 is based on the following publication:

- **R. Aiudi**, R. Pacelli, A. Vezzani, R. Burioni and P. Rotondo, *Local kernel renormalization as a mechanism for feature learning in overparametrized convolutional neural networks*, arXiv pre-print, arXiv:2307.11807, July 2023

Other publications during this Ph.D. course:

- **R. Aiudi**, C. Bonanno, C. Bonati, G. Clemente, M. D'Elia, L. Maio, D. Rossini, S. Tirone, K. Zambello *Quantum Algorithms for the computation of quantum thermal averages at work*, Phys. Rev. D 108, 114509 , December 2023

Abstract

This research delves into two primary domains: the intricate critical properties of long-range systems and the feature learning mechanisms in deep neural networks.

In the study of long-range systems, we examined the ferromagnetic Ising model in both one and two dimensions, characterized by interactions of the form $J_{ij} \propto r^{-(d+\sigma)}$. Utilizing a novel local dynamics on a dynamical Lévy lattice (DLL), we were able to reproduce the static critical exponents consistent with established literature, based on the interaction parameter σ. This localized approach offers a versatile methodology to probe the dynamical properties of various long-range models. Notably, our analysis of the relaxation time at the critical temperature revealed nuances in the relationship between the dynamical exponent z and the decay parameter σ, suggesting a potential disparity between dynamical and equilibrium critical properties. Moreover, due to the versatility of our strategy (DLL), we were able to conduct preliminary work in the study of the critical properties of the Long Range XY model.

Turning to deep neural networks, we explored the disparities in feature learning between fully-connected networks (FCN) and convolutional architectures (CNNs). Empirical studies on fully-connected networks in the infinite-width regime revealed a plateau in performance enhancement, attributed to the static nature of their kernel during training. This suggests that any inherent feature learning in such FCN structures has limited impact on generalization. Conversely, convolutional architectures (CNNs), particularly in the finite-width setting, have shown superior performance. Our theoretical framework for single hidden layer networks elucidates this disparity. While an infinite-width FCN's performance can be replicated by its finite-width counterpart using specific Gaussian priors, CNNs with a single convolutional hidden layer undergo a different kernel renormalization process. Unlike the global adjustments seen in FC networks, CNNs experience a localized renormalization, enabling adaptive selection of data-dependent components for predictions. This distinction emphasizes the advanced feature learning potential present in overparametrized shallow CNNs, which is not observed in equivalent FC architectures.

Collectively, these studies shed light on the profound influence of topology in diverse systems, ranging from the behavior of long-range physical models to the intricate feature extraction processes in neural architectures.

Keywords – Statistical Mechanics, Long-Range models, Monte Carlo methods, Deep Learning, Kernel methods, Bayesian Learning

Contents

1 Introduction **1**

I Long Range Complex Systems **3**

2 Long Range models **4**
2.1 Brief review on critical exponents and spectral dimension 5
2.1.1 Scaling laws 7
2.1.2 Statistical physics on graph 8
2.2 Long Range classical spin models 9
2.2.1 The Long Range Ising model 10
2.2.1.1 Fisher Renormalization Group approach 10
2.2.1.2 Sak picture for $\sigma \in (2 - \eta_{SR}, 2)$ 12
2.2.2 The Long Range XY model 13
2.2.2.1 BKT transition 13
2.2.2.2 Presence of Long Range interaction 14
2.3 Existing numerical solutions and known results 16
2.3.0.1 Cluster Algorithm 16
2.3.0.2 Kinetic Monte Carlo 18
2.3.0.3 Static Lévy Lattice 19

3 Dynamical Lévy Lattice (DLL) **20**
3.1 Algorithm 22
3.2 Theoretical expectations for the Long Range Ising model 24
3.2.1 Equilibrium properties 25
3.2.2 Dynamical behavior 27
3.3 Methods 29
3.3.1 The susceptibility critical exponent y 29
3.3.2 The dynamical critical exponent z 31
3.4 Results and numerical subtleties 34
3.4.1 1D Ising spin chain 35
3.4.1.1 Statics (y exponent) 35
3.4.1.2 Dynamics (z exponent) 37
3.4.2 2D Ising spin lattice 38
3.4.2.1 Statics (y exponent) 39
3.4.2.2 Dynamics (z exponent) 41
3.4.3 2D Long Range XY 44

II Deep Neural Networks **51**

4 Deep Learning theory in the Infinite-Width limit **52**
4.1 Supervised learning 53
4.1.1 Fully-Connected neural Network (FCN) 53
4.1.2 Convolutional Neural Network (CNN) 55
4.1.3 Loss Function 57

4.2 The problem of generalization in overparameterized neural networks . . . 59
4.3 Kernel methods and Gaussian Processes . . . 62
4.4 Bayesian Learning and Neural Network Gaussian Process (NNGP) . . . 67
4.5 Gradient Descent and Neural Tangent Kernel . . . 72

5 Beyond the Infinite-Width limit **75**
5.1 Finite-width networks: empirical evidence and theoretical approaches . . 76
5.2 Bayesian effective action for FCNs in the proportional limit . . . 81

6 Feature Learning in finite-width networks **90**
6.1 Bayesian effective action for CNNs in the proportional limit . . . 90
6.1.1 Local Kernel Renormalization as a mechanism of feature learning 94
6.2 Computing observables: the *similarity matrix* . . . 96
6.3 FCNs vs CNNs . . . 99
6.3.1 Generalization error through the predictor statistics . . . 100
6.3.2 Differences in the similarity matrix . . . 103
6.3.2.1 FCN at $\alpha_1 = 1$ . . . 106
6.3.2.2 CNN at $\alpha_1 = 1$ . . . 107
6.4 The predictive power of the theory at non-zero temperature . . . 109

References **113**

Appendix **120**
A1 LCN effective action derivation . . . 120
A2 Similarity matrix . . . 123
A2.1 FCN averaged similarity matrix . . . 124
A2.2 CNN averaged similarity matrix . . . 128
A3 Beyond the IW limit: numerical subtleties . . . 132
A3.1 Experiments with 1d convolutions . . . 133
A3.2 Experiments with 2d convolutions . . . 135

List of Figures

2.1 **Long Range XY model phase diagram** . . . 15
3.1 **Sak picture** . . . 27
3.2 **Finite-size extrapolation example for a 1D spin chain at $\sigma = 0.35$** 31
3.3 **Example of normalized error $\mathcal{E}_N$ trend against the block size time t_{BS}, before and after the rescaling with z, for a 2D spin lattice at $\sigma = 1.75$** . . . 34
3.4 **Short Range regime for a 1D spin chain** . . . 36
3.5 **$2 - y$ exponent for a 1D spin chain** . . . 37
3.6 **z exponent for a 1D spin chain** . . . 38
3.7 **$2 - y$ exponent for a 2D spin lattice** . . . 40
3.8 **Linear dependency of $\chi L^{-(2-\eta)}$ respect to $L^{-\delta}$, with $\delta = 0.42$** . . . 41
3.9 **a/b ratio at various σ** . . . 41
3.10 **z exponent for a 2D spin lattice** . . . 43
3.11 **Behavior of the Binder cumulant at $\sigma = 1.2$** . . . 46
3.12 **Behavior of the Binder cumulant at $\sigma = 4$** . . . 46
3.13 **Behavior of the Binder cumulant at $\sigma = 1.8$** . . . 47
3.14 **Collapsing the $G_c(r)$ curves with the η found with our strategy** . 48
3.15 **η extrapolation for the ferromagnetic-paramagnetic region $\sigma \in (0, 7/4)$** . . . 49
3.16 **$\eta(T)$ at $\sigma = 2.5$** . . . 49
3.17 **Behavior of η at different T at $\sigma = 1.8$** . . . 50
4.1 **Example of a FCN architecture** . . . 54
4.2 **Example of convolution operation** . . . 56
4.3 **Example of the simplest CNN architecture** . . . 57
4.4 **Bias-variance tradeoff** . . . 61
4.5 **Double-descent** . . . 62
4.6 **Example of kernel method** . . . 65
5.1 **Performance on the test set for different architectures, compared with the corresponding NNGP kernel method** . . . 76
5.2 **Performance behavior with increasing hidden layer size for different architectures** . . . 77
5.3 **Testing the predictions of the effective theory of a FCN** . . . 89
6.1 **Global and local kernels in fully-connected and convolutional one hidden layer networks** . . . 97
6.2 **Test loss vs number of $2d$ convolutional filters for one hidden layer CNNs with different filter sizes M** . . . 104
6.3 **Predicting the effect of global kernel renormalization on the internal representations of FCNs. A finite-size scaling analysis** . 108
6.4 **Predicting the effect of global and local kernel renormalization on the internal representations of CNNs. A finite-size scaling analysis** . . . 109
6.5 **Predicting the generalization error dependency on N_1 at $\alpha = 0.25$ fixed** . . . 111
6.6 **Predicting the generalization error dependency on λ_1** . . . 111
6.7 **Predicting the generalization error dependency on N_1 at fixed $P = 1000$** . . . 112

A3.1 **Additional finite-size scaling experiments on the similarity matrix,** $\alpha = 0.1$ 133
A3.2 **Additional finite-size scaling experiments on the similarity matrix,** $\alpha = 10$ 134

List of Tables

2.1 **Long-Range model regimes** 13
3.1 **Numerical details of our $2-y$ extrapolation for a 1-dimensional spin chain** 37
3.2 **Numerical details of our z extrapolation for a 1-dimensional spin chain** 38
3.3 **Numerical details about our estimate of $2-y$ in the 2D case** . . 40
3.4 **Numerical details of our z extrapolation for a 2D spin lattice** . . 43
6.1 **Summary of the analytical results for 1HL networks** 96
A3.1 **Additional finite-size scaling experiments on the similarity matrix** 134

1 Introduction

Complex systems, characterized by a multitude of elementary units, inherently exhibit behaviors that challenge conventional analytical approaches. The discipline of Statistical Mechanics provides a rigorous mathematical framework, facilitating a structured examination of these systems.

A critical observation in this field is the deterministic role of spatial configurations and the connections that guide inter-unit interactions. These factors fundamentally alter system behavior. This highlights the essential role of topology in determining the physics of complex systems.

For systems with long-range interactions, the conventional understanding based on Euclidean lattices is insufficient. These systems necessitate a conceptual shift, viewing them as existing on a more intricate graph. The specific nature of long-range interactions introduces modifications to the underlying geometry, leading to emergent behaviors that diverge from standard models.

Similarly, in the context of deep neural networks, topological considerations are fundamental. The architecture, defined by the connections between neurons, directly influences information transmission, signal processing, and overall network efficacy. Variations in this topological structure can result in significant differences in network performance and adaptability.

The focus of my Ph.D. research has been to investigate the behaviors of long-range systems and deep neural networks through the lens of Statistical Mechanics, employing both theoretical and numerical methodologies. For this reason, this thesis is divided into two main parts, one for each topic. In the following and for the entire elaborate, I will use the first plural person, to indicate the fact this work is the fruit of a collaboration between me, my supervisors, and some colleagues.

Part I explores the study of Long Range Complex Systems. Chapter 2 introduces the basic concepts required in our work with long-range models. This will be followed by a Renormalization Group overview concerning the Long Range Ising and Long Range XY, concluding with a summary of current algorithms for simulating these models. Chapter

3 elucidates our approach, the *Dynamical Lévy Lattice* algorithm. Once introduced, we show our numerical experiments aimed at identifying a static and a dynamical critical exponent for the Long Range Ising model. Finally, we mention our preliminary work on the Long Range XY model.

Part II is devoted to the study of Deep Neural Networks, emphasizing the differences between two prevalent architectures, the fully-connected and the convolutional neural network. In Chapter 4, we provide a review of Neural Networks working on a particular setting called Infinite-Width limit, where they become equivalent to Gaussian Processes, a type of kernel method. Standardly employed Neural Networks surpass kernel methods in capability, and in Chapter 5 we discuss contemporary approaches to understanding non-infinite-width networks, introducing our mathematical framework. This strategy employs Statistical Mechanics tools and allows the formulation of an effective action for fully-connected networks, working in the finite-width setting. Conclusively, in Chapter 6 we show how we managed to compute an effective action for convolutional neural networks. Observing the order parameters that guide this action, we discern a mechanism for feature learning in these networks. We analytically extrapolate some observables of interest and employ them to highlight the differences between the two architectures examined. In the end, we conclude by sharing our numerical tests to validate our theoretical assumptions and some ongoing works.

Part I

Long Range Complex Systems

2 Long Range models

Nature is rich in examples of Long Range interacting systems, both in the classical [1] and quantum realm [2], in the microscopic world of charged particles and the macroscopic one of astrophysical objects. The degrees of freedom of these systems interact through a potential which is a power-law function of the distance. The long-range nature of the interaction raises many difficulties in building a theoretical framework for describing these systems. Indeed, these systems exhibit non-additivity, in the sense that the energy and other thermodynamic functions of a composed system are not necessarily equivalent to the naive sum of the corresponding quantities of its parts. This fact implies that many of the properties of short-range systems are no longer present in long-range ones, as the convexity of the free energy and the equivalence of statistical mechanics ensembles. Let us make an example by taking the potential of the form

$$V(r) \propto r^{-\rho}\,, \tag{2.1}$$

with r the absolute value of the distance between two degrees of freedom, or particles. One can compute the interaction energy $\mathcal{E}$ of a particle. Assuming all the other particles are distributed homogeneously within a d dimensional sphere of radius R, the interaction energy goes as

$$\mathcal{E} \sim \frac{R^{d-\rho}}{d-\rho}\,. \tag{2.2}$$

Thus, the energy remains finite increasing R if and only if $\rho > d$ and so the total energy $E = V\mathcal{E}$ is extensive. For our purposes, we will restrict our next considerations only to this type of interaction, and so it is useful to define the *decay parameter* σ as

$$\rho = d + \sigma\,, \tag{2.3}$$

with $\sigma \geq 0$.

In the following, we will briefly present how one can exploit the theoretical framework given by Statistical Mechanics to investigate Long Range Complex Systems, in particular spin models.

2.1 Brief review on critical exponents and spectral dimension

Statistical Mechanics is a branch of physics that deals with systems composed of many parts or degrees of freedom, also referred to as particles. Statistical Mechanics provides the tools for investigating the behavior of these particles and for making predictions about their future. The starting point for studying a system made of many particles is to quantify how they interact with each other. This interaction could be of any kind: short-range, long-range, pairwise, absent (free model)... The behavior of these particles is fully characterized by the Hamiltonian H. At this point, to find statistical quantities of interest, the goal of any Statistical Mechanics theory is to compute the partition function, defined as

$$Z = \int \mathcal{D}\sigma \, e^{-\beta H(\sigma)} \,, \tag{2.4}$$

where β is the inverse temperature and the differential $\mathcal{D}\sigma$ means that the integral is performed over all the degrees of freedom, including every possible value of each one of them. From Z, one can compute all the most important statistical functions and calculate the average value of observables as $\langle O \rangle = Z^{-1} \int \mathcal{D}\sigma \, O(\sigma) e^{-\beta H(\sigma)}$. Moreover, given a particular configuration $\{\sigma_i\}$, the probability of finding the system in such a state is given by the Boltzmann distribution

$$P(\{\sigma_i\}) = \frac{e^{-\beta H(\{\sigma_i\})}}{Z} \,. \tag{2.5}$$

One of the key phenomena that occurs when dealing with a large number of particles is the emergence of a collective behavior, which serves as a bridge between the microscopic realm where these particles live and the macroscopic one. This collective behavior, arising from the intricate interplay of individual particle interactions, often manifests in ways that are not immediately intuitive based on the properties of individual particles. Such behaviors, when they lead to distinct macroscopic states of a system, are at the heart of what we term as **phase transitions**. A phase transition can be thought of as a transformation of a system from one state to another, driven by changes in external conditions like temperature or pressure. The system, in the passage from one phase to another, often

undergoes dramatic changes in its properties. For instance, the rigid structure of a solid melting into a liquid form, or a magnet suddenly losing its magnetization, are classic examples of phase transitions.

A simple, yet still studied, example of a system that undergoes a phase transition is the **Ising model**. Conceived initially to understand ferromagnetism, the Ising model consists of a lattice of spins σ, where each spin can either point up $(+1)$ or down (-1). These spins interact with their neighbors, and the nature of their interactions determines the overall state or phase of the system. In the absence of an external magnetic field, the Ising model has two distinct phases:

- Ordered Phase: At low temperatures, the spins tend to align with each other, resulting in a net magnetization. This phase is analogous to a **ferromagnetic** state, where the material exhibits a net magnetic moment.
- Disordered Phase: At high temperatures, thermal fluctuations dominate, causing the spins to orient randomly. This results in a net magnetization of zero, akin to a **paramagnetic** state.

The transition between these two phases, driven by temperature, is the phase transition of the Ising model. As the system is heated from the ordered phase, there comes a critical temperature, T_c, at which the system loses its net magnetization and transitions into the disordered phase. This temperature, T_c, is the critical point of the phase transition.

The beauty of phase transitions lies in their **universality**. Despite the vast diversity of systems and materials in nature, phase transitions often exhibit common features and behaviors. This universality stems from the fact that, near the transition point, the details of the individual particles become less important than the collective behavior of the system as a whole. One way of characterizing such universality is through the use of **critical exponents** and **scaling laws**. Indeed, systems having the same critical exponents, behave in the same way near the phase transition and they are said to belong to the same universality class.

2.1.1 Scaling laws

As the system approaches the critical temperature T_c, the point at which it undergoes a phase transition, several physical quantities exhibit power-law behavior. This means they can be described by functions of the form $f(x) \sim x^n$, where n is the critical exponent. Taking as an example the Ising model, let us explore some of these quantities and their associated exponents:

- **Magnetization** (M): This is a measure of the net magnetic moment of the system. Near T_c, it behaves as $M \sim |T - T_c|^\beta$.
- **Susceptibility** (χ): It measures how much the magnetization changes in response to an external magnetic field and near the critical points we have $\chi \sim |T - T_c|^{-\gamma}$.
- **Correlation length** (ξ): This is a measure of how far apart two spins can be and still be correlated with each other. As the system approaches T_c, the correlation length behaves as $\xi \sim |T - T_c|^{-\nu}$.
- **Spatial correlation function** ($G(r)$): It describes how two spins separated by a distance r are correlated:

$$G(r) \equiv \langle \mathbf{S}(0) \cdot \mathbf{S}(r) \rangle \ , \tag{2.6}$$

  where $\langle \cdot \rangle$ denotes the average over all sites and the Gibbs ensemble, and it is defined for spins of any dimension. Near T_c, it behaves as: $G(r) \sim r^{-(2-d+\eta)}$. This exponent η modifies the naive dimensional scaling of the spatial correlation function and for this reason is also called **anomalous dimension**.

As mentioned before, these exponents are not specific to a particular material but are shared among various systems that belong to the same universality class. This means that systems with entirely different microscopic details can exhibit the same macroscopic behavior near phase transitions, as characterized by their critical exponents. The concept of **scaling laws** further deepens our understanding of universality. Scaling laws describe how physical quantities change when the system length scales are transformed. They capture the essence of how quantities like magnetization or susceptibility vary with the system size, especially near the critical point. A fundamental scaling law near a critical point can be represented as $f(x, L) = L^b g(xL^a)$, where f is a physical quantity, x a control

parameter (like the temperature), L is the system size and g a scaling function.

The interplay between critical exponents and scaling laws offers a comprehensive framework for understanding the universality of phase transitions. Indeed, the critical exponents can be derived from the symmetries and the dimensionality of the system, further emphasizing the universality of critical phenomena.

One interesting thing about critical exponents is that they are not all independent. Indeed, it can be shown that a universality class is defined with just 2 critical exponents, while all the others can be derived through scaling relations. In our work, we will make use of one of them:

$$\gamma = \nu(2 - \eta) \tag{2.7}$$

2.1.2 Statistical physics on graph

Our work focused on the study of statistical systems having long-range interactions. The non-local nature of these interactions allows us to perceive the system as being established on a (strongly or fully-connected) graph instead of a traditional lattice. Essentially, long-range interactions can be seen as generating an effective graph topology that progressively breaks the 2D nearest-neighbor lattice topology as the system becomes increasingly interconnected. Within this context, there is an effective mapping between the physical system on the lattice and a comparable model on a suitable graph. This perspective benefits from various findings in network and graph theory. Additionally, this approach necessitates the introduction or reinterpretation of specific concepts, primarily the concept of dimension.

The properties of a spin system defined on a graph depend on the large-scale topology of the graph itself, which can be determined by means of a **random walker** (RW) [3]. Let us assume that the RW is isotropic, is forced to move every time step and there are no traps. Let $P_{ij}(t)$ be the probability for the RW of going from node i to node j in t steps.

It is known that the asymptotic power-laws of RW on lattices depend on the Euclidean dimension d, i.e. for all site i, $P_{ii}(t) \sim t^{-d/2}$. This relation still holds for infinite graphs when one considers the average over the sites and can be used to define a generalized

version of the Euclidean dimension, the so-called **(averaged) spectral dimension** d_s:

$$\bar{P} \sim t^{-\frac{d_s}{2}} \iff d_s = -2 \lim_{t\to\infty} \frac{\ln(\bar{P}(t))}{\ln(t)}\,, \tag{2.8}$$

assuming such a limit exists, and where $\bar{P} = \langle P_{ii} \rangle$ over the sites. This spectral dimension characterizes the presence or the absence of a phase transition of a model defined on a graph, analogously to the Euclidean dimension for systems on a lattice. Indeed, a generalized Mermin-Wagner theorem can be formulated [4, 5]:

Theorem 1. *Spin models with continuous symmetries on a general graph with nearest-neighbors bounded ferromagnetic coupling can not have a finite-temperature phase transition with spontaneous symmetry breaking if the spectral dimension of that graph is $d_s < 2$, while it must present such a transition if $d_s > 2$.*

Moreover, it can be shown that the last statement (presence of a phase transition if $d_s > 2$) still holds in the presence of a discrete symmetry, like in the Ising model.

As a result, the critical behavior of a long-range model on a d-dimensional lattice can be mapped with the spectral dimension of the corresponding graph where the model is represented. Thus, a continuous phase transition must occur whenever a model is defined on a graph whose spectral dimension is $d_s > 2$. Since in the next sections we will deal with long-range systems, it is useful to see what is d_s in such cases. In general, for a model defined on a d-dimensional lattice, whose long-range interactions decay as $r^{-(d+\sigma)}$, the spectral dimension of the equivalent graph is found to be [6]:

$$d_s = \begin{cases} \frac{2d}{\sigma} & \text{for } \sigma \in (0,2) \\ d & \text{for } \sigma \geq 2 \end{cases}\,. \tag{2.9}$$

2.2 Long Range classical spin models

Our work focuses on the study of how long-range interactions affect classical spin models, and for this purpose, we concentrate on two particular models, the simplest one but yet still under study, the Long Range Ising model, and a more general one, the Long Range XY model.

2.2.1 The Long Range Ising model

The general Hamiltonian which describes a model made of Ising 1/2-spins reads:

$$H = -\frac{1}{2}\sum_{i,j=1}^{N} J_{ij}\sigma_i\sigma_j, \tag{2.10}$$

where $\sigma_i = \pm 1$. The Hamiltonian written in Eq. (2.10) is general and the particular nature of the model depends on the choice of the parameter J_{ij}. We will focus on the ferromagnetic Long Range Ising model and so the interaction parameter is defined as

$$J_{ij} \equiv J r_{ij}^{-(d+\sigma)}\,, \tag{2.11}$$

where r_{ij} is the euclidean distance between i and j, σ is the decaying parameter as mentioned in Eq. (2.3), and for simplicity we will take from now on $J = 1$. The sum in this Hamiltonian covers all the degrees of freedom and so the interactions have infinite range and connect all the spin, making the system living on the nodes of a fully connected graph. As we will see, this could be a problem of resources in numerical simulations, because the number of interactions scales as $O(N^2)$, with N the total number of spins. The peculiarity of this model is that the long-range interaction allows the presence of a second-order phase transition at dimensions smaller than the lower critical one for the short-range counterpart. Indeed, the one-dimensional Long Range Ising model exhibits a transition at non-zero temperature. Furthermore, Renormalization Group calculations show that this kind of system changes universality class, continuously with the decay parameter σ. Indeed, there is an entire region in the parameter landscape where the critical exponent that governs spatial correlation, η is found to be

$$\eta = 2 - \sigma\,. \tag{2.12}$$

In the following, a review of known analytical results and conjecture is presented.

2.2.1.1 Fisher Renormalization Group approach

The first calculation of the d-dimensional Long Range Ising model, employing Renormalization Group techniques is found in Ref. [7]. A few months before the

publishing of this work, Wilson and Fisher itself [8] showed that for Short Range models, the fixed point values are of order $\epsilon = 4 - d$. Then, Fisher and some of his collaborators extended this idea to Long Range forces, arguing that the role of dimension 4 is taken over by 2σ. Thus, they managed to calculate the critical exponents for Long Range spin models as power series in $\epsilon = 2\sigma - d$. Through this definition, we can distinguish two regimes, the "classical" one, valid for $\epsilon < 0$, and the non-trivial, for $\epsilon \geq 0$.
In order to compute these exponents, the authors wrote the Hamiltonian in momentum space as

$$\begin{aligned}\frac{H}{k_B T} =&(2\pi)^{-d} \int d^d k\, u_2(\mathbf{k}) \mathbf{s}_{\mathbf{k}} \cdot \mathbf{s}_{-\mathbf{k}} + \\ &+ (2\pi)^{-3d} \int d^d k d^d k' d^d k''\, u_4(\mathbf{k}, \mathbf{k}', \mathbf{k}'') (\mathbf{s}_{\mathbf{k}} \cdot \mathbf{s}_{\mathbf{k}'}) (\mathbf{s}_{\mathbf{k}''} \cdot \mathbf{s}_{-\mathbf{k}-\mathbf{k}'-\mathbf{k}''}),\end{aligned} \tag{2.13}$$

where $\mathbf{k}$ denotes a d-dimensional momentum variable and $\mathbf{s}_{\mathbf{k}}$ is the Fourier transform of the spin variable. The interaction term u_4 corresponds to a four-spin local term, and so it is constant for this model, $u_4 \equiv u$. From the Fourier transform of the long-range interaction (2.11), the first term reads

$$u_2(k) = r + j_\sigma k^\sigma + j_2 k^2 + o(k^2), \tag{2.14}$$

with r varying linearly with the reduce temperature temperature $(T - T_c)/T_c$.
Now, if $2\sigma - d < 0$, the fixed point $u^* = r^* = 0$ is "Gaussian" and stable and one finds the following critical exponents

$$\eta = 2 - \sigma, \;\; \nu = 1/\sigma, \;\; \gamma = 1. \tag{2.15}$$

At the value $\epsilon = 0$, the fixed point becomes marginally stable, and a logarithmic correction to (2.15) appears. Finally, for $\epsilon > 0$ the fixed point is unstable with respect to u and one finds a new fixed point $u^* = O(\epsilon)$. In this regime, one has to distinguish two cases, $\sigma > 2$ and $\sigma < 2$, and Fisher with Wilson managed these calculations following a simple scaling argument. If $\sigma > 2$, the leading order of $u_2(k)$ is governed by k^2, so we recover the short-range regime, and the Renormalization Group analysis of the Short-Range Ising model can be applied. Instead, if $\sigma < 2$, the leading term is j^σ, and the following scaling arguments lead to Eq. (2.12). The length scale is renormalized by a factor b, such that

the correlation length changes as $\xi \to \xi/b$. Then, if the spin variables are rescaled by a factor c, the η exponent is determined, by definition, by the rescaling of the spin-spin interaction term, for which we get $c^2 = b^{2-d-\eta}$. Then, if the leading order is given by j^σ and u_4 is small, the rescaling factor of the Hamiltonian (2.13) is $b^{d-\sigma}c^2$. This implies that at the fixed point $\eta = 2-\sigma$. More involved Renormalization Group calculations show that the η exponent does not renormalize up to terms of $O(\epsilon^3)$ and it is believed to be valid also for the successive orders. Note that this is not longer true for the exponent ν, which renormalizes, and then its value could in principle depend on the model under study.
The previous argument about η would lead to two singular consequences. First of all, it implies that the lower critical exponent, which for Long Range systems is defined as the value of the decay parameter σ such that the Short Range behavior is recovered, is equal to $\sigma_L = 2$, for every dimension. This is in contrast with the one-dimensional case, for which at $\sigma = 1$ there is a Kosterlitz-Thouless transition, as supported by analytical and numerical evidence [9], and then for $\sigma > 1$ the model does not exhibit a non-zero temperature phase transition. Moreover, for the two-dimensional system, setting the Long-Short range crossover at $\sigma = 2$ would imply a jump discontinuity in η, which is a quite peculiar yet not prohibited behavior. In order to overcome these facts, Sak [10] provides a new way of thinking about the SR-LR crossover.

2.2.1.2 Sak picture for $\sigma \in (2-\eta_{SR}, 2)$

Sak argued that there is an entire interval of σ values lower than 2, such that term j_2 is not negligible. He showed that this term arises from Renormalization Group consecutive transformations even if j_2 is set to zero from the beginning. At the fixed point Fisher et al. showed that $r^* = u^* = O(\epsilon)$, $j^*_\sigma = constant$ and $j^*_2 = 0$ at order ϵ, but this last statement is not true for every $\sigma < 2$. Indeed, Sak managed to compute the fixed point recursion equations at order ϵ^2, found that $j^*_2 = O(\epsilon^2) \neq 0$. This implies that, if ϵ^2 and $2-\sigma$ are of the same order, we must consider and not neglect the Short Range interaction term. Sak, with a Renormalization Group *tour de force*, showed that this happens in the interval $\sigma \in (2-\eta_{SR}, 2)$, where η_{SR} denotes the η critical exponent of the short-range model. Thus, for $\sigma > 2-\eta_{SR}$, all the Short Range exponents are retrieved and the discontinuity mentioned earlier disappears. Sak picture solves also the one-dimensional problem, since in this case η_{SR} is equal to 1, and so the Short Range regime starts exactly at $\sigma = 1$.

Let us summarize and look at the whole picture. The $\epsilon = 2\sigma - d$ expansion allows one to compute the critical exponents of the Long Range Ising Model. If $\epsilon < 0$, i.e. $\sigma < d/2$, the system behaves like a "classical" one, and the exponent can be found with a Mean Field theory. If $\epsilon > 0$ and the interaction decays relatively fast, i.e. $\sigma > 2 - \eta_{SR}$, as argued by Sak, the system is equivalent to its Short Range counterpart, and so are the critical exponents. Finally, if $\epsilon > 0$ but $\sigma < 2 - \eta_{SR}$, the system is in a "non-classical" regime, and one has to compute the exponents as power series in ϵ. Interestingly, one finds that the η critical exponent does not renormalize, up to $O(\epsilon^3)$, and assumes its classical value $\eta = 2 - \sigma$. In Table 2.1 we summarize schematically this picture.

σ interval	Regime	η
$[0, \frac{d}{2})$	Mean-Field	$2 - \sigma$
$[\frac{d}{2}, 2 - \eta_{SR})$	Non-Classical (Long-Range)	$2 - \sigma$
$[2 - \eta_{SR}, \infty)$	Short-Range	η_{SR}

Table 2.1: Long-Range model regimes. Scheme of the landscape in Long Range models, varying the decay parameter σ, as argued by Sak.

2.2.2 The Long Range XY model

2.2.2.1 BKT transition

The XY is a model that describes spins that are 2-dimensional vectors free to rotate on a plane (the XY plane). This model respects the continuous $O(2)$ symmetry, instead of the discrete $\mathbb{Z}_2$ of Ising. The Hamiltonian reads

$$H = -J \sum_{<i,j>} \mathbf{S}_i \cdot \mathbf{S}_j = -J \sum_{<i,j>} \cos(\theta_i - \theta_j)\,, \tag{2.16}$$

where the sum runs over nearest-neighbor pairs and the spins are defined as $\mathbf{S}_i = (\cos(\theta_i), \sin(\theta_i))$. As a consequence of the Mermin-Wagner theorem [11], there can not be an ordered phase due to the presence of the $O(2)$ continuous symmetry. However, Berenziskii, Kosterlitz and Thouless (BKT) [12, 13] showed that below a critical temperature, the model exhibits quasi-long-range order and a transition must take place. It can be shown that this one can be described from a *topological* point of view, using the concept of vortex. A vortex (antivortex) is defined as a topological defect that satisfies the Laplace equation ($\nabla^2\theta(\mathbf{r}) = 0$), and whose contour integral is quantized and characterized by a positive

(negative) integer, $\oint \nabla\theta(\mathbf{r}) \cdot d\mathbf{l} = 2\pi k$. They showed that the low-temperature phase is characterized by the presence of vortex-antivortex pairs, which can not be decoupled. Instead, in the high-temperature phase isolated vortices are favorable and they proliferate.

Furthermore, there exists a whole phase (and not just a single point) in which the spatial correlation function $G(r)$, Eq. (2.6), decays as a power-law and not exponentially. This is in contrast with the usual exponentially decaying at high temperatures, implying the presence of a phase transition. Surprisingly, this power-law is governed by a temperature-dependent exponent $\eta(T) \propto T$. Renormalization Group calculations show that at the transition temperature, T_{BKT} the value of the correlation critical exponent is $\eta(T_{BKT}) = 1/4$.

2.2.2.2 Presence of Long Range interaction

Analogously to the Ising model, we can build a Long Range XY model, by considering the Hamiltonian

$$H = \frac{1}{2} \sum_{i,j=1}^{N} J_{ij} \left[1 - \cos(\theta_i - \theta_j)\right] , \tag{2.17}$$

where J_{ij} is defined as in Eq. (2.11). The presence of a long-range interaction alters the behavior of the XY model, depending on the decay parameter σ. Indeed, we have seen that in long-range systems the presence of a spontaneous symmetry breaking depends on the spectral dimension d_s. Since the XY model is defined in two dimensions, the generalized Mermin-Wagner theorem implies the existence of a magnetized phase for $\sigma < 2$ and its absence for $\sigma > 2$. Thus, for $\sigma > 2$ the system will behave as in the presence of only short-range interaction, and a paramagnetic-BKT phase transition will occur. At this point, one can wonder if the Sak criterion can be applied to this model and then if the system is found in a short-range regime for $\sigma \in (2 - \eta_{SR}, 2)$. The problem is that for the XY model we can not formally define a value of η_{SR}, since there is a whole line of fixed point and the η exponent is temperature-dependent. However, one can conjecture that the value of η_{SR} for applying the Sak criterion is the one at the BKT temperature, i.e $\eta_{SR} = \eta(T_{BKT}) = 1/4$. This would imply the presence of a BKT phase transition for $\sigma \in (7/4, 2)$. Thus, from one point of view, this region should be characterized by the presence of a BKT transition, and from the other, the generalized Mermin-Wagner theorem states that the system should magnetized since $\sigma < 2$. In order to avoid the fact that the theorem will be broken, the system should exhibit three different phases

in this region: ferromagnetic (magnetized), BKT, and paramagnetic (disordered phase). Recently, the authors of [14, 15] provided a Renormalization Group calculation based on a perturbative treatment of the long-range interaction. Interestingly, they found that the previous highlighted picture is valid and the region $\sigma \in (7/4, 2)$ exhibits the presence of three phases: ferromagnetic up to a temperature T_c, BKT in the range (T_c, T_{BKT}), paramagnetic above T_{BKT}.

Moreover, they found an interesting behavior of the scaling form of the connected spatial correlation $\langle \mathbf{S_i} \cdot \mathbf{S_j} \rangle_c \equiv \langle \mathbf{S_i} \cdot \mathbf{S_j} \rangle - \langle \mathbf{S_i} \rangle^2$. They found that in the whole ferromagnetic phase, the connected spatial correlation goes as $\langle \mathbf{S_i} \cdot \mathbf{S_j} \rangle_c \sim r_{ij}^{\sigma-2}$, while in the paramagnetic one it reads $\langle \mathbf{S_i} \cdot \mathbf{S_j} \rangle_c \sim r_{ij}^{-\sigma-2}$. Their calculations correctly provide that in the BKT region, the temperature dependence of the correlation exponent is recovered.

We report in Fig. 2.1 a sketch of the expected behavior of the Long Range XY model in the $T - \sigma$ plane, taken directly from [15].

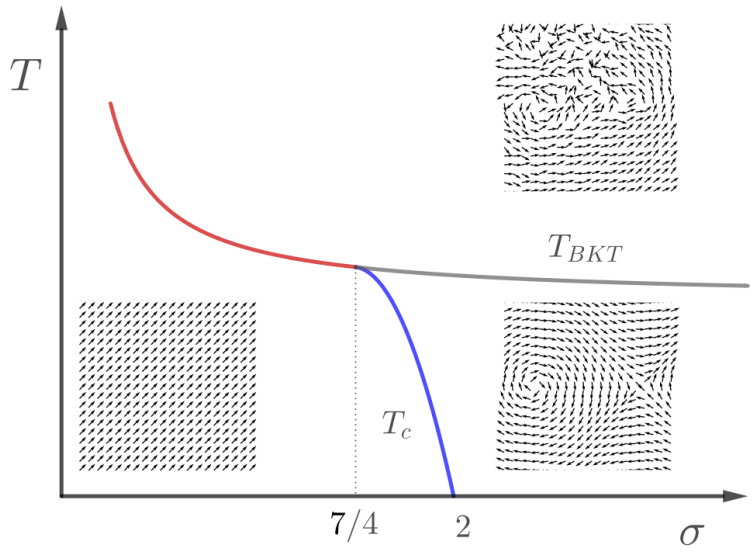

Figure 2.1: Long Range XY model phase diagram. This sketch is directly taken from [15]. It provides a visual representation of the behavior of the Long Range XY model. For $\sigma \in (0, 7/4)$ the system behaves accordingly to the generalized Mermin-Wagner theorem and a ferromagnetic-paramagnetic phase transition (red line) will occur. For $\sigma > 2$, the system is equivalent to the short-range model and it presents only a BKT-paramagnetic phase transition (grey line). Finally, for $\sigma \in (7/4, 2)$ all three phases are present (ferromagnetic-BKT phase transition is represented with the blue line).

2.3 Existing numerical solutions and known results

The model described by the Hamiltonian (2.10) lives in a hypercubic lattice in d dimension and the long-range nature of the interaction connects every spin with all other spin. Then, if N is the total number of degrees of freedom, the number of interactions scales as $O(N^2)$, requiring large resources for simulations. Indeed, the fast-growing number of connections with the size makes it unfeasible to simulate this kind of system with a standard local Monte Carlo algorithm. To overcome this numerical challenge, several solutions were proposed before ours, and in this section, we will briefly present them. One solution is to drop locality and build a cluster algorithm, similar to the one conceived by Wolff [16]. Indeed, the efficiency of this algorithm is independent of the number of interactions, i.e. it has a computational cost of $O(N)$, and due to the non-locality, it is not affected by the critical slowing down effect near the transition. Another recent solution is the so-called *kinetic Monte Carlo* algorithm. This is a local algorithm that implements a clever way of reducing the rejection rate of spin flip. It is highly effective in conditions of lower temperatures where the occurrence of spin flips is quite rare, while in scenarios with high temperatures and at critical points, its efficiency matches that of the standard Metropolis. Due to these characteristics, its application has been primarily in examining non-equilibrium properties within the coarsening dynamics. Another interesting solution, which is the playground of ours, is static Lévy lattices. These are randomly diluted graphs, where interactions between pairs are constant and set with a probability proportional to $r^{-(d+\sigma)}$, with the overall quantity of interactions being on the order of $O(N)$. Lévy lattices significantly cut down the computational expenses while maintaining local dynamics, but an average computed over multiple iterations is necessary. Let us illustrate these three solutions, and why we need a new algorithm to simulate Long Range models.

2.3.0.1 Cluster Algorithm

The main feature of Monte Carlo cluster algorithms resides in the fact that they can flip an entire set of spins at each time. The total amount of computational cost is due to the formation of a cluster, which has to be built following the correct probability distribution. By changing many degrees of freedom at one time, they are not affected by the critical slowing down, because we can move entire domains, without any limit of size, at the same

cost as a single spin-flip in local algorithms. Thus, the crucial step is the identification of a cluster of spins with identical orientations. The basic steps of any cluster algorithm are the following:

(i) Randomly choose a spin σ_i from the lattice;

(ii) Consider each spin that interacts with this spin, having the same orientation, and add it to the cluster with a probability $p_j = 1 - \exp(-2\beta J_{ij})$, where J_{ij} is the interaction strength between the two spins σ_i and σ_j, and β is the inverse temperature;

(iii) Repeat step 2 for each spin that is newly added to the cluster, until all neighbors of all spins in the cluster have been considered.

While it is possible to construct a cluster algorithm for systems with long-range interactions, its efficiency rapidly declines as the number of interactions increases. The usual value for the coupling J_{ij} in such a system tends to be small with the distance, meaning that each spin considered for cluster inclusion has a lower chance of being added. As a result, a substantial number of operations are needed to incorporate just a single spin into the cluster. Then, Luijten and Blöte [17] designed a more efficient way for selecting a cluster of spin, in the presence of long-range forces. Their strategy starts by computing the probability $P(n)$ that in the first step, $n-1$ spins are skipped and the nth is added and from that defining the cumulative bond probability $C(j) = \sum_{n=1}^{j} P(n)$. Thus, if we extract a random number r uniformly in $[0, 1]$ and the condition $C(j-1) < r < C(j)$ is satisfied, then the spin added will be the jth, having checked that it has the same sign of the starting spin. Having inserted the first spin, the next one to be added has to be at a distance $k > j$. To do that, it is sufficient to define a generalized form of the probability and its cumulative. This generalized cumulative probability reads $C_j(k) = 1 - \exp\left(-2\beta \sum_{n=j+1}^{k} J_n\right)$, where J_n denotes the strength of the interaction between two spins at distances n. Spins continue to be added in this manner until the maximum distance of $L\sqrt{d}/2$ is reached, where L is the system size. Following that, attempts are made to add neighbors, starting from all the other spins that have already been added to the cluster, using the same process, until all the spins added to the cluster are checked. Once the cluster is built, all the spins within it are flipped and the whole procedure is started again.

This strategy is very efficient, compared to standard local Metropolis algorithms. Indeed,

if the probability for adding a spin were equal for each spin pair, let us say p, then it would take $O(p^{-1}) \sim O(L^0)$ operations per spin to update a configuration, compared to the $O(L^d)$ of local algorithms. Moreover, by changing many spins each time, the effect of critical slowing down near criticality decreases drastically, and so one can obtain a gain of $O(L^{d+z})$, z being the dynamical critical exponent of time correlation.

This algorithm has been utilized to examine Sak scenario and to affirm that it provides an accurate depiction of the critical behavior in the Ising case [18, 19, 20, 21, 22]. However, due to the absence of local dynamics, the algorithm cannot be employed to investigate temporal evolution, that is, dynamical properties. Furthermore, while the identification of clusters is simple for the Ising model, it cannot be executed in models with continuous variables, such as the Long Range XY model. Moreover, the efficiency of the cluster algorithm drops rapidly at low temperatures, since a $O(N)$ of spins are added to the cluster at each update, making the overall cost of $O(N^2)$. Recently, a cluster algorithm has also been developed for the study of long-range percolation [23].

2.3.0.2 Kinetic Monte Carlo

The idea behind the Kinetic Monte Carlo algorithm is to avoid repetition of the expensive calculation of the energy difference ΔE_i in the Metropolis step. The strategy relies on saving the effective field associated with each spin σ_i, i.e. $\sum_{j\neq i} J_{ij}\sigma_j$. In this way, the energy difference due to a spin flip is easily computed as $\Delta E_i = 2\sigma_i \sum_{j\neq i} J_{ij}\sigma_j$. Then, if the spin is flipped, the effective field of all other spins σ_j changes of a quantity $-2\sigma_i J_{ij}$. It is easy to see that this operation has the same computational cost of calculating a single energy difference in standard approaches. However, this is done if and only if the spin σ_i is flipped, i.e. only when the configuration update is accepted. Thus, the computational cost gain of this algorithm stands only for temperature below the critical one, where the acceptance rate is low, while it behaves similarly to standard Monte Carlo methods at higher T. For this reason, this algorithm is mostly used in the study of aging phenomena [24, 25, 26, 27], where one studies how the system behaves when cooled rapidly from a high-temperature state to a temperature below the critical one, also called quenching, while it is not so efficient for numerical extrapolation of static properties.

2.3.0.3 Static Lévy Lattice

The Lévy lattice is a diluted graph with an adjacency matrix $A_{i,j} = 0, 1$, constructed to exhibit the same characteristics as the fully connected long-range model [28, 29, 30, 31]. Specifically, the long-range ferromagnetic model on a d-dimensional lattice is approximated by a graph where two points are linked with a probability proportional to the long-range interaction, i.e., $A_{ij} = 1$ with a probability of $P_{ij} \propto r_{ij}^{-(d+\sigma)}$. On the diluted Lévy lattice, the standard local Monte Carlo simulations are implemented with the same cost of cluster algorithm, i.e. $O(N)$, as each node is connected to a finite and non-extensive number of edges.

There is no analytical proof that models on Lévy lattices are equivalent to their fully connected long-range counterparts. Specifically, for 1-dimensional lattices, numerical evidence suggests that static long-range correlations can alter the critical properties in the free model, modifying the spectral dimension [32]. Indeed, once the graph is built, a random walker on it can not feel the presence of a long-range connection, i.e. starting from a site, it could perform a small and a long jump with equal probability, which does not depend on the distance, changing inevitably its dynamics. However, in the 2-dimensional case, the spectral dimension of the Lévy lattice appears to match that of the long-range, suggesting that the difference is confined to lower dimensions [31, 33].

Unlike the cluster algorithm, the Lévy diluted lattice can be used to simulate continuous symmetry models. Valid simulations have been carried out for the 2-dimensional XY model with long-range interaction [31, 34], amid a Kosterlitz Thouless phase transition [14]. We have already seen that the Long Range XY model presents an interesting region where the presence of three phases should be found. However, numerical simulations in [34] do not find any intermediate BKT region for $\sigma = 1.875$, and the question about the equivalence between the diluted model and the original Long Range XY model still holds.

On finite systems, simulations on Lévy lattices heavily rely on graph realization, and averages over a significant number of samples are necessary to achieve stable results. In this context, the issue of self-averaging remains unresolved; additionally, the averaging procedure can be highly computationally demanding [34].

3 Dynamical Lévy Lattice (DLL)

The core idea of our approach is a combination of two algorithms: the last one described in the previous chapter, the static Lévy lattice, and the so-called q-Ising model [35, 36]. The scope is to exploit the non-extensive number of connections of Lévy lattices, while simultaneously getting rid of averaging over different graph realizations. Moreover, we want a model that has with certainty the same spectral dimension as the fully connected system.

The q-Ising model is realized by an algorithm for which, at each interval of time, a spin σ_i is randomly selected and interacts with the field created by q neighbors, which are also uniformly and randomly selected from the remaining $N-1$ spins. Then, according to the Metropolis [37] or Glauber [38] rule, the spin σ_i undergoes a flip. As pictured in [39], this model has two fluctuating elements: the spins and the connections. The spins interact with the heat bath, whereas the links are randomly rearranged during the dynamics, without any need for an acceptance-rejection process. The links can be thought of as being in thermal contact with a heat bath at an infinite temperature $T = \infty$, implying a rewiring probability of 1. With two distinct heat baths controlling the dynamics, detailed balance is not achieved and thermal equilibrium is not immediately clear.

Our approach is to extend the reasoning of the q-Ising model to long-range interactions, simply changing the selection of the q neighbors. We start with a regular lattice, with a spin on each node. At each time step, we select one spin σ_i at random uniformly and its q neighbors following the power-law probability distribution $P_{ij} \propto r_{ij}^{-(d+\sigma)}$. Then σ_i flips, according to a constant interaction J with the q randomly selected spins. Note that in this way the decay of the interaction as a power-law is recovered in a statistical sense, like in static Lévy lattices. However, our algorithm does not explore all possible realizations of the disorder and it does not sample different quenched realizations of the Lévy lattices: it samples directly and dynamically the fully-connected topology. The physical quantities are computed by averaging over the dynamic evolution of the fully connected model and thus feature the same symmetries of the statistical model on the fully-connected graph. Indeed, the model can be thought of as living on a *dynamical* Lévy lattice, i.e. a random

diluted graph that evolves in time, with an interaction between spins of the form

$$J_{ij}(t) = J \times A_{ij}(t)\,, \tag{3.1}$$

where $A_{ij}(t)$ represents the adjacency matrix of the underlying graph, and it is 1 if at time t the spin σ_j is drawn as a neighbor of σ_i and 0 otherwise.

By definition, random walks on a Dynamically Lévy Lattice (DLL) exhibit the same behavior as random walks on a fully connected long-range graph. Indeed, in this scenario, the walker jumps at each step from the initial node i to any other nodes of the network j with a probability of $P_{ij} \propto r_{ij}^{-(d+\sigma)}$. This probability is identical to that used on a DLL at each step to select the q-neighbors of the walker, with the jumps occurring uniformly among these q nodes. Therefore, the two dynamics are equivalent and, consequently, the spectral dimension - as measured from the return probability of the random walker [40, 3] - is the same on the fully connected graph and the DLL. For this reason, we expect that the DLL belongs to the same universality class of the Long Range Ising model since it shares with it the same symmetries and the same dimensionality.

On the contrary, on a static Lévy lattice, the walker evolves on a random static network, thus in the presence of spatial correlations. For instance, when the walker crosses a link connecting two distant sites, there is a significant probability of it retracing its steps along the same link that remains active on the static graph. Conversely, it fails to reach the lattice sites unconnected by long-distance links in that specific quenched realization within a few steps. This introduces static long-range correlations that are absent in both the fully connected lattice and the DLL. Specifically, in one dimension, such correlations [32] can alter the spectral dimension compared to the fully connected long-range model. Moreover, in the DLL, the correlation functions $\langle \sigma_i \sigma_j \rangle$ are computed by averaging over the dynamic evolution of the model, hence naturally maintaining the symmetries of the statistical model on the fully connected graph. Conversely, on a Lévy lattice, $\langle \sigma_i \sigma_j \rangle$ relies on the quenched realization of the random structure, and the resulting disorder disrupts the original translation invariance. As a result, the original spatial symmetries are only reestablished after averaging over multiple realizations of the random lattice.

In the following, we present in detail the *algorithm* which implements our model, and the *methods* that will allow us to verify *a posteriori* its validity since it is not guaranteed due

to the fact the model does not preserve detailed balance.

3.1 Algorithm

The algorithm which implements the DLL is quite similar to a standard Monte Carlo local algorithm. In the beginning, a spin σ_i is chosen randomly among all the N spins and then a flip of its sign is proposed. For doing that, we choose a set S_q of q spins, from which the effective temporal field interacting with σ_i is computed. Those spins are chosen following the distribution $P(r) \propto r^{-(d+\sigma)}$, where r is the Euclidean distance from the initial spin. Dealing with finite-size systems, there are several choices of this probability distribution, depending on the choice of the boundary conditions, and later we will discuss them in detail. The effective field is easily computed following the Ising Hamiltonian, considering a constant interaction for every couple. From this, the energy difference in the system due to flipping the initial spin is $\Delta E_i = 2 \cdot J \cdot \sigma_i \cdot \sum_{\sigma_j \in S_q} \sigma_j$. Finally, the spin-flip is proposed following a standard Metropolis or Glauber prescription. A single Monte Carlo step is defined as repeating this procedure N times. We schematize this procedure in Algo. 1.

Algorithm 1 DLL (single Monte Carlo step)

Require: $q \in \mathbb{N}$ and $\sigma \in \mathbb{R}^+$

1: Choose a random spin σ_i
2: Draw a node $j \neq i$ from the probability distribution $P(r_{ij}) \sim r_{ij}^{-(d+\sigma)}$ where r_{ij} is the distance between nodes i and j. Repeat the extraction q times so that you get q random nodes (the extraction of the same node more than one time in the list S_q is allowed)
3: Calculate the interaction energy $\Delta E = 2 \cdot J \cdot \sigma_i \cdot \sum_{j \in S_q} \sigma_j$
4: Flip σ_i following a dynamics, Metropolis or Glauber, using ΔE as flipping energy
5: Repeat steps 1-4 $O(N)$ times

We want to stress that this algorithm can be easily generalized to more complex systems, and also to model with continuous symmetries, for example, the XY model. Indeed, the only part that changes with the model is step number 3 in Algo. 1, for which one has to adapt it to the correct energy calculation, following the corresponding Hamiltonian.

In the fully connected standard q-Ising model i.e. $P(r_{ij}) = constant$, analytical calculations show that for $q < 2$ the system does not present a phase transition at finite temperature i.e. $T_c = 0$; with Metropolis dynamics, for $q = 3$ there is a continuous phase transition of the mean field universality class while for $q > 2$ a first order transition is observed

[35, 36]. For Glauber dynamics instead, a continuous transition is always observed for $q > 2$. For this reason we choose the Glauber dynamics and set $q = 3$ so that we expect a second order phase transition, as we indeed observe in our simulations. In particular, we verify that both lowering and increasing temperature the system never presents hysteresis. A small value of q allows for an efficient implementation of the algorithm while large values are typically more demanding since the algorithm requires the extraction of q random numbers in a microscopic step. In general the study of the dependence on q of the dynamical evolution is an interesting open issue.

Let us discuss the possible choices of the boundary condition and $P(r_{ij})$. Note that in the thermodynamic limit, where $N \to \infty$, all the choices lead to the same physical model, and so they are all equivalent from a statistical point of view. Nevertheless, some choices can better compensate for finite-size effects or they could be more suited depending on the particular statistical analysis needed.

One possibility is to draw r_{ij} from a truncated Pareto distribution, i.e. from a distribution proportional to the correct power-law, but such that the possible distance lies in the interval $(\sqrt{d}/2, L/2)$, where L is the linear size of the lattice. The choice of the upper bound $L/2$, allows us to impose periodic boundary conditions, i.e. the system lives in a hypertorus in d dimensions. While this strategy is efficient from a computational cost point of view, since it does not require any control, it could lead to high finite-size effects. Indeed, when dealing with systems of small sizes, such as $O(10^2)$, as it is typical when simulating high dimensions systems, the bounded Pareto differs significantly from the infinite, no-bounded one. This implies that the spins tend to interact more with their nearest neighbors, making the long-range interaction weaker. Another possible choice, which turns out to be very efficient, is to use the infinite images method. In practice, one has to attach infinite copies of the original lattice across the borders, in every direction. This is realized by drawing the distance r_{ij} from the Pareto distribution, without an upper limit, and taking the neighbor winding around the torus the correct number of times, allowing also a spin to interact with itself. This method has the same computational cost as the previous one, because there is no need for controls, at least when using our algorithm. Instead, when dealing with cluster algorithms, one has to compute with attention the interaction to correctly build the cluster and this was done [20, 22] by taking

the continuum limit of the discrete sum in (2.10). This has the double effect to raise the computational cost and to deal with an approximated version of the real interaction. However, it was shown that this strategy deals better with finite-size effects, especially near the long-short range crossover. In our analysis, we have tested both of these two conditions. The former led to strong finite-size effects, due to the upper bound on the probability distribution, making the interaction too similar to a short-range one, even with small values of σ. The latter mitigated this effect, but in this way, the system tended to stay magnetized for a longer time, also at temperatures slightly above the critical one. Indeed, the presence of infinite images has the effect of making the effective field of neighbors spin stronger when dealing with systems of small size. This can lead to wrong estimates of magnetization statistics, such as the mean and the variance, i.e. the magnetic susceptibility χ, which is needed for our scope of numerically computing the η exponent. To overcome this issue, we adopted the following strategy. We drew again the distance r_{ij} from the unbounded distribution, but setting the value of spins σ_j situated farther than $L/2$ from the starting spin σ_i to zero, $\sigma_j = 0$, while imposing standard periodic boundary conditions if the distance is within half of the linear size. In this manner, when distant coordinates are considered, it is supposed that interactions occur with nodes that have an average magnetization of zero, i.e. the torus is surrounded by an infinite temperature system. Meanwhile, spin-spin correlations are significant only at distances smaller than L. This choice has a computational cost slightly greater than the previous ones, but we found it compensates better both finite-size effects and spurious magnetization near the critical temperature.

3.2 Theoretical expectations for the Long Range Ising model

We have just seen that our algorithm, the *Dynamical Lévy Lattice*, is efficient compared to existing solutions. Indeed, by choosing a not extensive value of q, the number of neighbors extracted, the computational cost of a Monte Carlo step is of order $O(N)$, the same as cluster methods or static Lévy lattice (we are not considering here the problem of critical slowing down, typical of local algorithms). Moreover, due to its locality, our algorithm is easily generalized to many types of spin models, like the XY model or spin glasses (it

is not clear if the DLL will lead a spin glass system to have only a paramagnetic phase, due to the dynamical averaging process). However, it has an insidious drawback. As was mentioned before, due to the presence of two different heath baths the detailed principle is no longer valid, and then the existence of a equilibrium distribution in the Markovian sense, is not guaranteed. For this reason, we have to test its validity *a posteriori*, via a numerical comparison with existing results and the widely accepted theory, i.e. the picture highlighted by Fisher and then by Sak. Once we have evidence of the correctness of our model, we can make use of the algorithm to numerically estimate dynamical exponents, exploiting its local nature.

3.2.1 Equilibrium properties

One way to check the validity of the DLL algorithm is to find numerically a critical exponent, which depends on the decay parameter σ, for example, the one governing the behavior of the magnetic susceptibility at criticality. Indeed, this exponent changes its value depending on the regime. In many cases, this exponent is identical to the one of spatial correlations, but this is not true in the Mean Field regime. Then for clarity, we will refer to it as y. We recall here the definition of the magnetic susceptibility χ,

$$\chi = \frac{L^d}{k_b T}(\langle m^2 \rangle - \langle |m| \rangle^2), \tag{3.2}$$

where m is the magnetization of the system, $m = L^{-d}\sum_{i=1}^{N}\sigma_i$, the average operator $\langle\cdot\rangle$ refers to the average over the Gibbs ensemble, d is the dimension of the underlying lattice and for simplicity, we set all the quantities in units such that $k_b = 1$. χ diverges at criticality in the presence of a continuous phase transition, following a power-law

$$\chi \sim L^y \tilde{\chi}(tL^{\frac{1}{\nu}}), \tag{3.3}$$

where $t = \frac{T-T_c}{T_c}$ is the reduced temperature (T_c being the critical temperature) and $\tilde{\chi}$ is a finite scaling function. Then, since $y > 0$, for infinite systems, the susceptibility diverges.

Let us clarify now what is the expected value of y in the three different regimes. For $\sigma \in (0, \frac{d}{2})$, the Long Range Ising model attains a classical behavior and in this region, the system can be described by the Mean Field (MF) theory. For magnetic spin models, the MF

free energy can be written in terms of the total magnetization, $F_{MF} = L^d(Atm^2 + Bm^4)$, with t the reduced temperature and A, B constants. From that, we can derive the power-law Eq. (3.3) from a scaling argument. The average over the Gibbs ensemble of the magnetization squared reads for definition

$$\langle m^2 \rangle \propto \int dm\, e^{-L^d(Atm^2+Bm^4)} m^2 \,. \tag{3.4}$$

In order to compute how this average scales with the size, we have to normalize the exponential, so that the system size dependency is made explicit. Since we are interested in the behavior of χ at criticality, and here $t = 0$, it is sufficient to rescale the quartic term in the free energy, changing the variable of integration such that $L^d m^4 = m'^4$. By doing so, Eq. (3.4) becomes

$$\langle m^2 \rangle \propto L^{-\frac{d}{2}} \int dm'\, e^{-(AL^{-\frac{d}{2}}tm'^2+Bm'^4)} m'^2 \,. \tag{3.5}$$

At the critical point, the integral is no longer size-dependent, then, from the definition of the susceptibility, we get

$$\chi \sim L^d \langle m^2 \rangle \sim L^d L^{-\frac{d}{2}} \sim L^{\frac{d}{2}}, \tag{3.6}$$

which implies that, in the Mean Field region, the y exponent has to be $y = \frac{d}{2}$. This has already been tested numerically via cluster algorithm in [19].

For $\sigma > d/2$, the MF approximation is no longer valid, and one has to rely on Renormalization Group calculations, briefly presented in Section 2.2. We have to distinguish two cases depending on the value of the decay parameter, and from now on we will consider the boundary found by Sak. For both these cases, the magnetic susceptibility scales with the size as $L^{\frac{\gamma}{\nu}}$, and so from the scaling relations we can derive $y = 2 - \eta$. For $\sigma \in (\frac{d}{2}, 2 - \eta_{SR})$, η_{SR} being the η critical exponent of the corresponding short-range model, the system behaves in a non-classical way, and the strength of the long-range interaction modifies the value of the critical exponents. We will call this regime the Long Range (LR) region. Here, Renormalization Group calculations show that $y = 2 - \eta = \sigma$. Finally, for values of the decay parameter such that $\sigma > 2 - \eta_{SR}$, the interaction decays so fast that the system behaves like the corresponding short-range model, sharing the same critical exponents. In this region, which will be referred to as

Short Range (SR), $y = 2 - \eta_{SR}$. Note that, the value of η_{SR} for a one-dimensional Ising model is $\eta_{SR}^{(1D)} = 1$, and so for $\sigma > 1$, the Long Range 1D Ising model will not exhibit a phase transition for any $T > 0$. We summarize the expected value of y in Fig. 3.1. This picture, theorized by Sak, has been numerically corroborated by many experiments, exploiting algorithms previously mentioned [18, 19, 20, 22, 21].

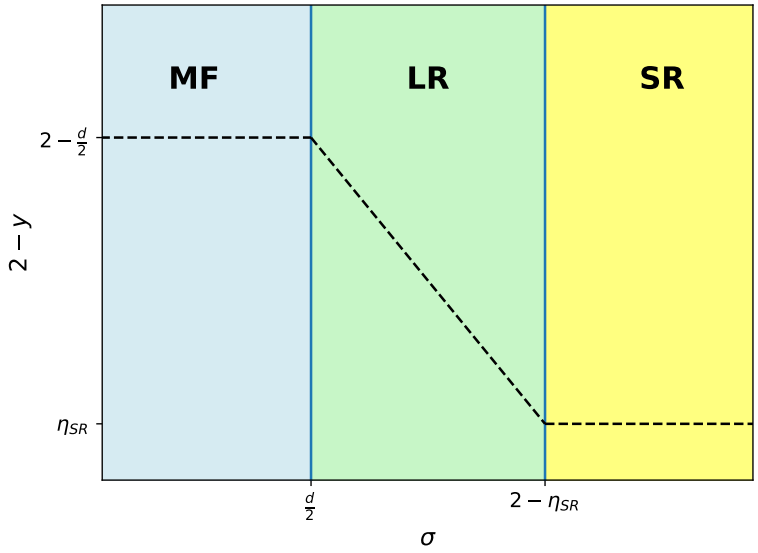

Figure 3.1: Sak picture. This sketch represents the expected value of the critical exponent $2 - y$ when varying the decay parameter σ, according to the picture outlined by Sak. Note that, in the case of a 1D spin chain, the right region should be omitted, since there is not a phase transition.

3.2.2 Dynamical behavior

One of the relevant features of our algorithm is its locality. On one hand, this makes simulations near criticality numerically demanding, due to the critical slowing down effect, on the other, we can exploit it for studying out-of-equilibrium and dynamical properties. One of these is the dynamical critical exponent z, governing the behavior of the relaxation time. Indeed, the autocorrelation time τ diverges at criticality following the power-law $\tau \sim \xi^z$, where ξ is the correlation length. We recall τ being the Monte Carlo time required to sample two statistically independent configurations and its divergence is a quantitative measure of the critical slowing down [41]. Since we are dealing with finite-size systems, the only characteristic length scale is the linear size L, then the τ power-law can be written as

$$\tau \sim L^z \,. \tag{3.7}$$

We choose to compute τ from the autocorrelation time of the absolute value of the magnetization $|m|$. Until now, there are no analytical studies of this dynamical critical exponent, for Long Range spin models. Thus, in the following, we describe heuristically what we can expect in the three regimes outlined in the Sak scenario.

In the Mean Field region, the long-range interactions are so strong that the system can be considered as a Curie-Weiss model. This implies that the underlying geometry disappears and the only remaining information is the total number of spins, which can be regarded as the volume $V = L^d = N$. We can derive the dependence of τ on V via the following argument. Let us write the Fokker-Planck equation for the magnetization

$$\frac{\partial P(m,\tau)}{\partial \tau} = \frac{\partial}{\partial m}\left[\left(A(T-T_c)m + Bm^3\right)P(m,\tau)\right] + \frac{\tilde{D}}{V}\frac{\partial^2}{\partial m^2}P(m,\tau)\,, \tag{3.8}$$

with A, B arbitrary constant. Note that we have written the explicit dependency of the diffusion coefficient on the system size. Thus, at the critical point, the equilibrium distribution found in solving the Fokker-Planck equation is $P^{(\mathrm{eq})}(m) = Be^{-\frac{C}{V}m^4}$, which implies $\langle m^2\rangle \sim V^{-\frac{1}{2}}$. Near T_c, the magnetization remains close to 0 and the expected dynamics is purely diffusive. Then, initializing the system with zero magnetization we get $\langle m^2\rangle \approx \frac{D}{V}t$. This is sufficient to conclude that the time needed to reach the equilibrium scales as $t \sim V^{\frac{1}{2}}$, and from that we derive our expectation for the z exponent in the Mean Field region $z = \frac{d}{2}$, i.e. $\tau \sim L^{\frac{d}{2}}$.

As was briefly described in Section 2.2, Renormalization Group analysis shows that at $\sigma = \frac{d}{2}$ a long-range spin model is equivalent to a Gaussian one and this implies that the system is in a free super-diffusive regime, for which the dynamical critical exponent z is

$$z = \frac{2d}{d_s}\,, \tag{3.9}$$

with d_s the spectral dimension of the underlying graph, defined in Section 2.1.2. The authors of [6] have found that in long-range systems, the spectral dimension is $d_s^{(\mathrm{LR})} = \frac{2d}{\sigma}$, and from that and Eq. (3.9) we derive $z = \sigma$. When σ becomes larger than half the dimension, the free random walk behavior will be perturbed by the interaction, which implies z to be slightly larger than σ. This will be true up to a value of σ near d.

As soon as σ approaches the dimension of the underlying lattice, we have to make the

distinction between $d = 1$ and higher dimensions. For $d = 1$, we know that for values of $\sigma > 1$ the short-range model is recovered, and so for the case of Ising spins, critical dynamics will disappear. Instead, for $d > 1$, we know from the picture of Sak that the short-range behavior is recovered when $\sigma \geq 2 - \eta_{SR}$. However, the anomalous diffusion $z = \sigma$ of a free random walk surely becomes Gaussian when $\sigma = 2$. Thus, it is not clear if the dynamical exponent z follows the static picture of long-range spin models, and if the short-range dynamics is recovered at $\sigma = 2 - \eta_{SR}$ or $\sigma = 2$. We will see that our simulations support the latter hypothesis.

3.3 Methods

Previously, we have outlined what are the expectations about the static critical exponent y and the dynamical one z. In this section, we are going to explain our strategy for numerically extrapolating such quantities to compare them with the theory.

3.3.1 The susceptibility critical exponent y

We know that the magnetic susceptibility χ becomes infinity at the critical point. However, in numerical simulations, we deal with finite-size systems so the magnetic susceptibility can not diverge. Instead, it attains its maximum at a size-dependent temperature $T_c(L)$ close to the critical temperature of the infinite-size system. The value of this maximum follows the same power-law of Eq. (3.3), at least at the leading order. Thus, by taking the logarithm of this equation, we can efficiently extrapolate the wanted exponent y,

$$\log\left(\chi_{max}(L)\right) = y \cdot \log L + c, \tag{3.10}$$

where $(\chi_{max}(L))$ is the maximum value that χ can attain for a system of size L and c is a constant.

In order to estimate the maximum of the susceptibility, we adopt the following strategy. During the Monte Carlo simulations the value of the magnetization, $m = L^{-d} \sum_{i=1}^{N} \sigma_i$, is saved every step, i.e. every N spin-flip trial. For every simulation, at every temperature, we checked *a posteriori* that the system had thermalized, discarding the very first steps. Obviously, due to the locality nature of the algorithm, near the critical temperature the

integrated autocorrelation time, a measure of how different Monte Carlo configurations are correlated, diverges, and so the number of steps required for thermalization becomes high. This issue affects also the analysis of average quantities because to have a significant statistical sample, one has to simulate longer Monte Carlo stories. To mitigate this effect and to have a better estimate of the statistical error on the average magnetization, we simulate $O(10^2)$ independent stories in parallel. Thus, we take as the average value of the wanted quantities, the mean value over the simulations of the Monte Carlo temporal averages, and similarly, we consider as its error the standard deviation over these parallel stories. Once we have a statistical sample of the magnetization, we can compute the susceptibility following its definition (3.2), where the averages and errors are estimated as previously mentioned. The next step is to extrapolate the maximum value of χ. A standard way to do so is to apply the so-called *histogram reweighting method* [42]. In very few words, it is a technique that allows to compute the expected value of some observable at a given temperature from that simulated at a nearby temperature, simply by storing the energy of the system at each Monte Carlo step. While this method turns out to be very efficient, it is not directly applicable to our algorithm, since it requires computing the energy. Indeed, we recall that the underlying graph of the DLL is built dynamically and there is no way to define and compute the global energy of a particular configuration at each Monte Carlo step. Therefore, we estimate the maximum concentrating the simulations near the peak and extrapolating this value with a quadratic fit, supposing a polynomial trend. In this way, for every setting of parameters (dimension and σ), we collect a set of pairs $(\chi_{max}(L), L)$ and start our analysis of the wanted critical exponent.

Long Range models are known to be affected by strong finite-size effects, especially near the LR-SR crossover [43, 22]. Typically, Eq. (3.10) is valid only asymptotically and one has to manage higher order correction due to the distance of the system from the true thermodynamic limit. Our strategy is to measure how much the angular coefficient y varies with the linear size. We take three (two for the 1D case) consecutive sizes in the power of 2, and we compute $y(L)$ with a linear fit against them. Finally, we extrapolate the thermodynamic limit $L = \infty$ with a new linear fit against the reciprocal of the size, supposing a dependency of the form $y(L) = y + const. \cdot L^{-1}$. Fig. 3.2 shows an example of how the slope varies when bigger and bigger sizes are considered.

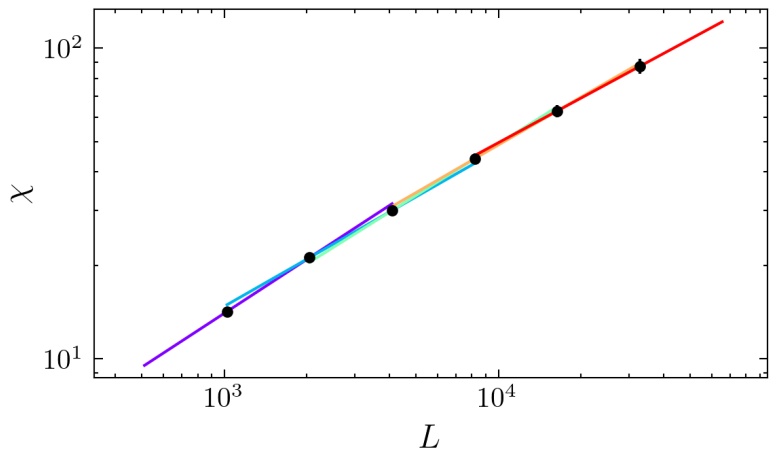

Figure 3.2: Finite-size extrapolation example for a 1D spin chain at $\sigma = 0.35$. Each line is the result of a linear fit of two consecutive sizes and it is clear how the slope changes with increasing sizes.

As we will show in the next section, this strategy seems to work well for a wide range of σ, but it fails to manage stronger finite-size effects near the LR-SR crossover. Near this region, one has to take into account explicitly higher order terms in L in Eq. (3.10). Indeed, the authors of [22] showed evidence that the correction to scaling of this equation at $\sigma = 1.75$ vanishes more slowly than linearly. In particular, they considered a scaling law of the form

$$\chi \sim L^{2-\eta}(a + bL^{-\delta}), \tag{3.11}$$

where a, b and δ are parameters in principle σ dependent. They empirically found that a value of $\delta = 0.42$ can be taken into account for the correct scaling corrections. In our methods, we can incorporate this correction, by simply modifying the angular coefficient dependency, i.e. by considering a fit with the law $y(L) = y + const. \cdot L^{-\delta}$. In the next section, we will see that, by using the value $\delta = 0.42$ for every σ simulated, we can get rid of finite-size effects and get results in great agreement with Sak picture. Moreover, we anticipate that we found an interesting relation between σ and the ratio a/b, which seems to have a sharp minimum exactly at the LR-SR crossover, validating again the hypothesis that here the finite-size effects are stronger than in any other value of σ.

3.3.2 The dynamical critical exponent z

In the previous section, we argued about what is the expected behavior of the dynamical critical exponent z in the three regimes. In the following, we define the numerical procedure

we adopted to test it. As was mentioned before, we proceed by estimating the scaling of the autocorrelation time of the magnetization with the size. In standard practice [41], this quantity is related to the error $\mathcal{E}$ on the mean value $\bar{m}$ by the following relation

$$\tau = \frac{\mathcal{E}^2}{\mathcal{E}_1^2}\,, \tag{3.12}$$

where we have defined by $\mathcal{E}_1$ the estimate of the error on $\bar{m}$ without considering Monte Carlo time correlations, i.e. $\mathcal{E}_1^2 = \sum_{i=1}^{T_{MC}} \frac{(m_i-\bar{m})^2}{N(N-1)}$, with m_i the value of the magnetization at the i-th time step and T_{MC} the total simulation time. One way to numerically estimate τ is the Jackknife method, which we recall briefly. The strategy is to construct new samples aggregating temporally consecutive blocks of magnetization, substituting an entire block with its mean m_i^{BS}. Thus, if the block size (BS) is t_{BS}, we build a new sample made of T_{MC}/t_{BS} elements. Then, we compute the statistical error of this new sample

$$\mathcal{E}^2(t_{BS}) = \sum_{i=1}^{T_{MC}/t_{BS}} \frac{(m_i^{BS}-\bar{m})^2}{\frac{N}{t_{BS}}\left(\frac{N}{t_{BS}}-1\right)}\,. \tag{3.13}$$

For large values of t_{BS}, the quantity $\mathcal{E}^2(t_{BS})$ is independent of the block size. So the best estimate of the error on the mean $\bar{m}$ of the time series is the limit value of $\mathcal{E}(t_{BS})$, i.e. from Eq. (3.12) $\tau = \lim_{t_{BS}\to\infty} \mathcal{E}^2(t_{BS})/\mathcal{E}^2(1)$. Let us define the normalized error ratio for each linear size of the system to be $\mathcal{E}_N(t_{BS}, L) \equiv \mathcal{E}(t_{BS}, L)/\mathcal{E}(1, L)$. At criticality and for large values of t_{BS}, this quantity is expected to scale with the only intrinsic time in the dynamical evolution, L^z, and since t_{BS} is a time length, we can write

$$\mathcal{E}_N(t_{BS}, L) = L^{z/2}\tilde{\mathcal{E}}_c(\sqrt{t_{BS}L^{-z}})\,, \tag{3.14}$$

with $\tilde{\mathcal{E}}_c$ a scaling function. Let us discuss the limits of this function. Since for $t_{BS} \gg \tau$, $\mathcal{E}_N(t_{BS}, L) = \tau^{1/2} \sim L^{z/2}$, then $\tilde{\mathcal{E}}_c(x)$ is constant for large x. Instead, when $t_{BS} \ll \tau$, the variance of the new sample should be quite similar to the one of the original time series. Then, from the definitions, their ratio is near to the value t_{BS}. Thus, for $x \sim 0$, $\tilde{\mathcal{E}}_c(x) \sim x$. The previous arguments are valid exactly at criticality, $T = T_c$. Close to this point, we expect a more general scaling relation, considering the reduced temperature,

$$\mathcal{E}_N(t_{BS}, L, t) = L^{z/2}\tilde{\mathcal{E}}(\sqrt{t_{BS}L^{-z}}, tL^{1/\nu})\,. \tag{3.15}$$

In order to deal with the dependency on t, we exploit the scaling form of the susceptibility (3.3). Indeed, at the peak of χ, the quantity $tL^{1/\nu} \equiv x_M$ is constant, and independent of the system size. Then, for every size we have $\mathcal{E}_N(t_{BS}, L, t) = L^{z/2}\tilde{\mathcal{E}}(\sqrt{t_{BS}L^{-z}}, x_M) \equiv \mathcal{E}_N(t_{BS}, L)$, and the previous analysis continues to be valid. In the following, we explain how we managed to estimate z for all the σ simulated since there is no standard procedure for doing it. The strategy relies on defining a function of z, whose minimum is an indication of the best estimate of the exponent. For every possible values of z, we rescale the normalized error $\mathcal{E}_N$ with the corresponding power of the linear size, $L^{-z/2}$, and we consider the curves $L^{-z/2}\mathcal{E}_N$ as a function of $\tilde{t} \equiv \sqrt{t_{BS}L^{-z}}$. Since the best estimate of z is the one for which the rescaled curves collapse onto each other, we compute, for every time step $\tilde{t}$, the difference between consecutive curves in size,

$$\Delta\mathcal{E}_N(\tilde{t}, L) \equiv L^{-z/2}\mathcal{E}_N(\tilde{t}, L) - (L/2)^{-z/2}\mathcal{E}_N(\tilde{t}, L/2)\,. \tag{3.16}$$

Then, To consider the impact of finite-size effects, we estimate the thermodynamic limit of $\Delta\mathcal{E}_N$, relying on an assumption of linear correlation with the inverse of the size:

$$\Delta\mathcal{E}_N(\tilde{t}, L) = const. \cdot L^{-1} + \Delta\mathcal{E}_N(\tilde{t}, \infty)\,. \tag{3.17}$$

During our experiments, we have tested also other trends L^{-x}, with $x \neq 1$, but we found small deviations from the final z estimate. Finally, to penalize discrepancies from 0, we take the square sum of $\Delta\mathcal{E}_N(\tilde{t}, \infty)$ for all values of $\tilde{t}$. Thus, this entire procedure leads to the definition of the following function

$$\Delta\mathcal{E}_N(z) \equiv \sum_{\{\tilde{t}\}} \Delta\mathcal{E}_N^2(\tilde{t}, \infty)\,, \tag{3.18}$$

where the sum is performed over all the value of $\tilde{t}$ considered. We take as the best estimate of z, the one for which this function presents a minimum. To estimate its error, we take an interval over this z, such that the function $\mathcal{E}_N(z)$ reaches five times its minimum value.

We want to stress that our newly proposed approach proves to be highly effective. Indeed, when dealing with strong finite-size effects, one needs to include in the scaling analysis large systems, for which often the simulation time is not significantly greater than the

decorrelation time. Because of this, the asymptotic value of τ is not reached, as it is evident for example for the size $L = 512$ in Fig. 3.3, where we show $\mathcal{E}_N$ for a 2D spin lattice at $\sigma = 1.75$. While a distinct plateau has not been reached for this size, our technique enables us to extrapolate z, even when working with short time series. An alternative way to measure the z exponents using even shorter simulations is through a critical quench [44], in which equilibration of the system is not even required. This method, however, requires knowledge of the static equilibrium critical exponents, while our approach is completely agnostic to them.

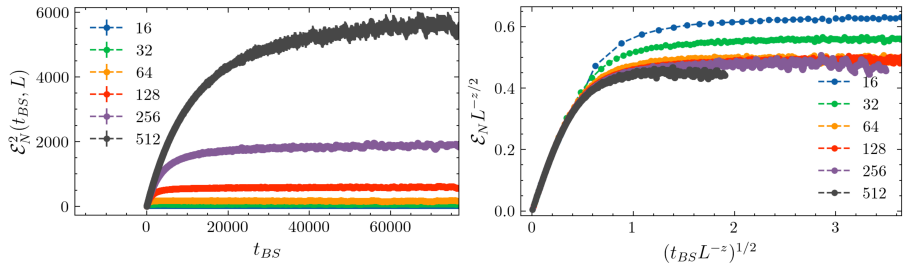

Figure 3.3: Example of normalized error $\mathcal{E}_N$ trend against the block size time t_{BS}, before and after the rescaling with z, for a 2D spin lattice at $\sigma = 1.75$. Left: $\mathcal{E}_N^2$ for $\sigma = 1.75$ for a 2D spin lattice. The value of the autocorrelation time τ is the plateau, which is not yet reached for the biggest size $L = 512$. Right: Rescaling of $\mathcal{E}_N$ using our estimation $z = 1.75$. The collapse improves with increasing size, suggesting the differences at small sizes are due to finite-size effects. Our methods enable us to estimate z even if not all the curves have reached their asymptotic value.

3.4 Results and numerical subtleties

In this section, we will show all the numerical results obtained for the Long Range Ising model. All the simulations were carried out on the HPC (High Performance Computing) facility of the University of Parma. The codes that run the simulations are written in C++, while the data analysis was performed using Python as the programming language.

To test the validity of our algorithm, we have to compare our numerical results with theoretical expectations. In order to do so, we simulate a 1-dimensional and 2-dimensional Ising spin lattice, and then check if the vastly accepted Sak picture is recovered. As mentioned before, we manage this comparison by numerically extrapolating the critical exponent governing the magnetic susceptibility behavior at criticality, i.e. we estimate the value of the y exponent, defined in Eq. (3.3). For every parameter setting, dimension and σ, and for every temperature, we collect the total magnetization $m = L^{-d} \sum_{i=1}^{N} \sigma_i$,

for a total of $O(10^6)$ realizations. In each case, we check *a posteriori* the thermalization of the system, by waiting for a sufficient number of Monte Carlo steps, before saving the data. The standard way is to wait for a number of steps proportional to the integrated autocorrelation time τ, which depends on the size and temperature. Since τ is maximum at criticality, we wait for every temperature simulated at least $5 \times \tau_{T_c(L)}$, where $T_c(L)$ is the temperature for which χ presents a peak. To improve statistics and overcome the critical slowing down effect, we perform a $O(10^2)$ of parallel simulations for every parameter setting, and thus every $\langle m \rangle$ is the result of an average over different time series of mean values over Monte Carlo time. In this way, we can estimate the error on $\langle m \rangle$ and χ, by taking the standard deviation of the average over the different stories. In the previous section, we explained how we extrapolated the maximum of the susceptibility and how we managed finite-size effects, through the application of various consecutive fits, linear and quadratic. The errors on each final estimate are computed by propagating the statistical error from the direct measurement, ensuring that the corresponding reduced χ^2 remains close to unity.

3.4.1 1D Ising spin chain

First of all, we test our algorithm on a 1-dimensional spin chain. We consider length sizes from $L = 2^8$ to $L = 2^{15}$, covering three orders of magnitude. We choose five values of σ: 0.35 for the Mean Field region, 0.6, 0.8, and 0.9 for the Long Range region, and 1.2 for the Short Range one. We will first present our estimate of y exponent, for which we find good agreement with Renormalization Group expectations and then we will show the results about the dynamical critical exponent z.

3.4.1.1 Statics (y exponent)

We begin by looking at the Short Range region, for which the one-dimensional Ising model behavior should be recovered. It is known that this model does not exhibit a phase transition for any temperature $T > 0$. This implies that the spins can not be aligned to a global direction, and the system is always found in a paramagnetic phase, i.e. the total magnetization per spin m is zero for every temperature. Obviously, this is true only in the thermodynamic limit, when the number of spins N is led to infinity. Thus, one way to find if our simulated system is compatible with a 1-dimensional Ising model is

to check the behavior of $\langle|m|\rangle$ with the linear size. Indeed, we simulate a system with decay parameter $\sigma = 1.2 > d$ and we find that $\langle|m|\rangle$ goes to zero for each temperature as the linear size is increased, as it is shown in Fig. 3.4. This is true for a wide range of temperatures, up to the lowest simulated, $T = 0.01$.

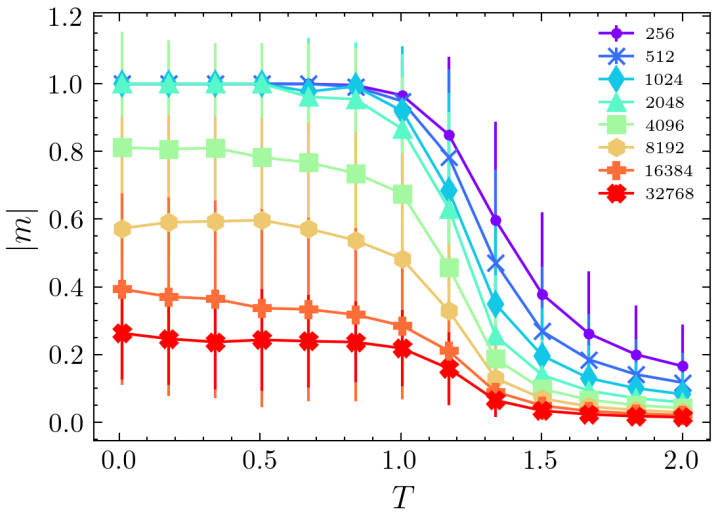

Figure 3.4: Short Range regime for a 1D spin chain. $|m|$ vs T, for increasing values of the linear size, at $\sigma = 1.2$. It is clear that increasing the size, the magnetization goes to 0, even at low temperatures (the lowest temperature is $T = 10^{-2}$). Errors are computed considering the integrated autocorrelation time, indeed they grow bigger near $T = 0$, which is the transition temperature for a 1D Short Range Ising model.

Having checked the short-range behavior is recovered, we move to the extrapolation of the y exponent in the remaining regions. For the Mean Field one, we simulate $\sigma = 0.35$, while for the Long Range region, we took $\sigma = 0.6, 0.8, 0.9$. We find good agreements with the picture outlined by Renormalization Group calculations, as it is clear from the plot of Fig. 3.5, whose details are collected in Table 3.1. Near the LR-SR crossover, the simulations are more demanding, as can be seen by the presence of larger errors in the y estimate. This could be a consequence of the fact that at $\sigma = 1$, the one-dimensional Long Range Ising model exhibits a Kosterlitz-Thouless phase transition [9], and a detailed study requires further investigations.

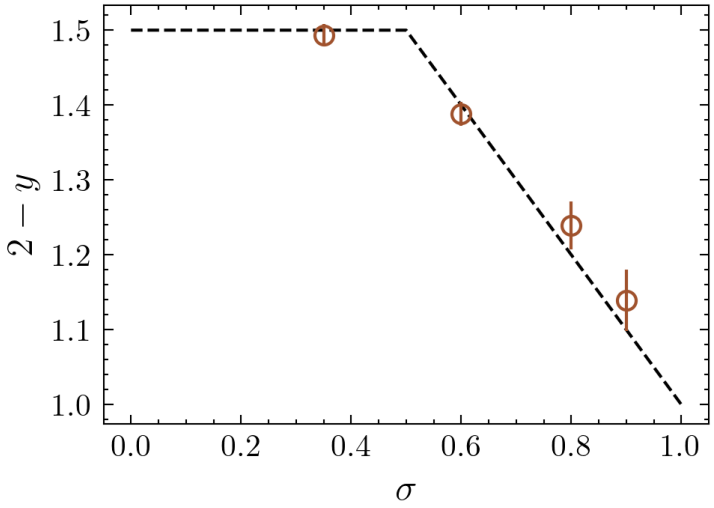

Figure 3.5: $2-y$ exponent for a 1D spin chain. Our estimate of the y critical exponent for the values of σ simulated. The black dashed line represents the theoretical expectation, as described in Section 3.2, for which we have $y = d/2 = 0.5$ in the Mean Field region and $y = 2 - \eta = \sigma$ in the Long Range one. As mentioned in the text, the errors are computed propagating the statistical error on the measures. We found good agreements with Sak scenario. Getting closer to $\sigma = 1$, the simulations become demanding and the errors grow. This could be due to the vicinity of a Kosterlitz-Thouless transition, which takes place at exactly $\sigma = 1$ [9].

σ	$2-y$ extrapolated	$\Delta(2-y)$
0.35	0.507 ± 0.015	0.007 ± 0.015
0.6	0.612 ± 0.016	0.012 ± 0.016
0.8	0.761 ± 0.032	0.039 ± 0.032
0.9	0.861 ± 0.041	0.039 ± 0.041

Table 3.1: Numerical details of our $2-y$ extrapolation for a 1-dimensional spin chain. The last column indicates the discrepancy concerning the expected theoretical exponent.

3.4.1.2 Dynamics (z exponent)

There is clear evidence that our algorithm works well for a one-dimensional Long Range Ising model, and then we exploit its local nature to extrapolate the dynamical critical exponent z, through the strategy previously mentioned. We collect our results in Fig. 3.6 and Table 3.2. The errors are computed as explained in Section 3.3.

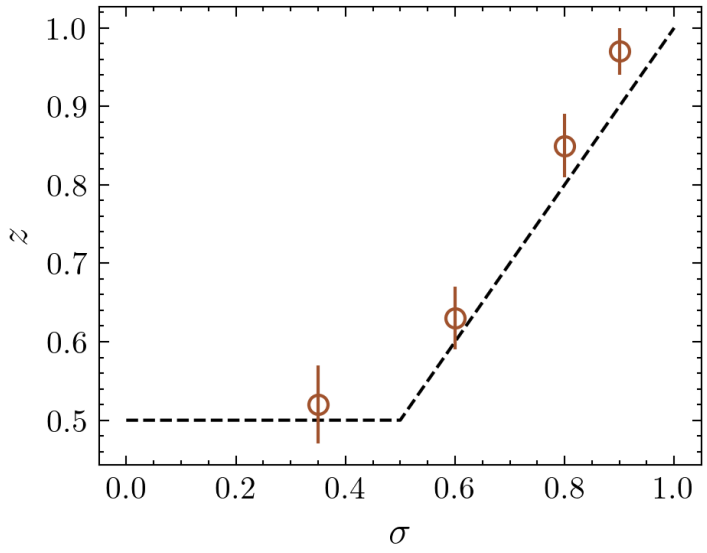

Figure 3.6: z exponent for a 1D spin chain. Our estimate of the z critical exponent for the values of σ simulated. The black dashed line represents the expectation coming from our arguments. It is clear that in the Mean Field region, the hypothesis $z = d/2 = 0.5$ is corroborated. In the Long Range one we observe a small perturbation to the free diffusive behavior in the Long Range one, which would lead to $z = \sigma$, and this perturbation gets bigger in the vicinity of the point $\sigma = 1$, where critical dynamics disappears.

σ	z extrapolated
0.35	0.52 ± 0.05
0.6	0.63 ± 0.04
0.8	0.85 ± 0.04
0.9	0.97 ± 0.03

Table 3.2: Numerical details of our z extrapolation for a 1-dimensional spin chain. We found good agreements with the picture we have outlined in the text. We observed that in the Mean Field region $z = d/2$, while in the Long Range region, this exponent is slightly perturbed from the value $z = \sigma$.

3.4.2 2D Ising spin lattice

The 2-dimensional case is more subtle for several reasons. The number of spins N scales squarely with the linear size L, increasing the computational cost. Because of this, we restricted L to be in the interval $[2^4, 2^9]$. Since we deal with smaller linear sizes with respect to the 1D case, the finite-size effects are more significant. Indeed, in simulating the 2D lattice, the furthest distance on which we can rely is $L_{max}^{(2D)}/2 = 256$, which is three orders of magnitude lower than that of the 1D case $L_{max}^{(1D)}/2 = 16384$. The issue of strong finite-size effects is well known in the literature, especially near the LR-SR crossover

[43, 22]. As was mentioned in previous sections, we managed these effects by choosing suitable boundary conditions and including second-order terms in the scaling relation (see Eq. (3.11)). Similarly to the 1D case, we will first show our study of the critical static exponent y and then of the dynamical one z.

3.4.2.1 Statics (y exponent)

We collect our results in Fig. 3.7 and Table 3.3. The brown circles in this figure, whose detailed values are shown in the second column of the table, represent our estimate using just the leading terms, i.e. Eq. (3.3). It is evident that those points deviate significantly from the expected values, represented in the figure by a black dashed line and this deviation is bigger near the LR-SR crossover, at $\sigma = 1.75$. We simulate the same linear sizes with the cluster algorithm and it reveals the same discrepancies, which implies they are attributable to finite-size effects rather than pathologies of the DLL algorithm. Indeed, the authors of [22], argued that by considering second-order terms, in the form of Eq. (3.11), one can recover the analytical predictions of Sak calculations. In particular, they found that a value of $\delta = 0.42$, accounts for the correct scaling at $\sigma = 1.75$. Guided by their results, we consider the same correction to scaling $\delta = 0.42$, and we find it is consistent also for the other values of σ. Indeed, the quantity $\chi L^{-(2-\eta)}$ displays a linear dependency on $L^{-\delta}$, as shown in Fig. 3.8 for σ near the LR-SR crossover, which validates the scaling corrections considered (we recall that in this region y is by definition equivalent to η). Interestingly, these straight lines, extrapolated with a linear fit, are parallel to each other, meaning they share the same coefficient b (see Eq. (3.11)). Nevertheless, they move toward the x-axis approaching the crossover $\sigma = 1.75$, which implies that finite-size effects are stronger near this point since the coefficient a becomes less significant. Indeed, we study the ratio a/b over the whole set of σ simulated, finding a sharp minimum at $\sigma = 1.75$. These ratios can be seen in the plot of Fig. 3.9, where the y-axis is represented in a logarithmic scale, to better highlight the sharpness of this minimum. We show our estimate of $2 - y$ with the $L^{-\delta}$ correction as blue stars in Fig. 3.7, whose details can be found in the last two columns of Table 3.3. Note that the errors in this case become bigger since it assumes larger scaling corrections.

Our analysis has shown that, by applying known finite-size effects strategies, our algorithm provides the expected equilibrium critical properties even for the two-dimensional case,

thus we can investigate the dynamical critical exponent z.

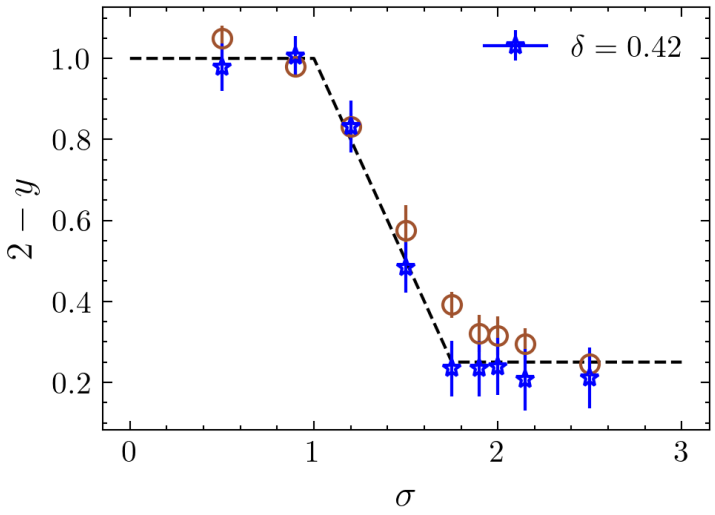

Figure 3.7: $2-y$ exponent for a 2D spin lattice. Here we collect our estimate of the y exponent, both with and without scaling corrections. The brown circles represent the results without considering second-order terms. The discrepancy from the expected values, shown with the black dashed line, grows bigger near the LR-SR crossover, at $\sigma = 1.75$. Nevertheless, by taking into account next to leading order corrections, our simulations show good agreements with the Sak scenario. We made use of the same correction $L^{-\delta}$ of [22], with $\delta = 0.42$ and we pictured these points as blue stars.

σ	$2-y$ (leading)	$\Delta(2-y)$	$2-y$ ($\delta = 0.42$)	$\Delta(2-y)$
0.5	0.95 ± 0.03	0.05 ± 0.03	1.022 ± 0.059	0.022 ± 0.059
0.9	1.019 ± 0.025	0.019 ± 0.025	0.995 ± 0.051	0.005 ± 0.051
1.2	1.168 ± 0.033	0.032 ± 0.033	1.169 ± 0.064	0.031 ± 0.064
1.5	1.424 ± 0.061	0.076 ± 0.061	1.516 ± 0.062	0.016 ± 0.062
1.75	1.608 ± 0.032	0.142 ± 0.032	1.766 ± 0.068	0.016 ± 0.068
1.9	1.679 ± 0.046	0.071 ± 0.046	1.766 ± 0.069	0.016 ± 0.069
2.0	1.685 ± 0.048	0.065 ± 0.048	1.761 ± 0.07	0.011 ± 0.07
2.15	1.705 ± 0.038	0.045 ± 0.038	1.793 ± 0.076	0.043 ± 0.076
2.5	1.754 ± 0.026	0.004 ± 0.026	1.789 ± 0.075	0.039 ± 0.075

Table 3.3: Numerical details about our estimate of $2-y$ in the 2D case. The second and third columns collect our results when neglecting second-order corrections to scaling. The discrepancies are greater near the point $\sigma = 1.75$. These discrepancies become compatible with zero when considering next-to-leading-order terms. In the last two columns are presented our estimate of $2-y$, when using a correction $L^{-\delta}$ as described by Eq. (3.11), with $\delta = 0.42$.

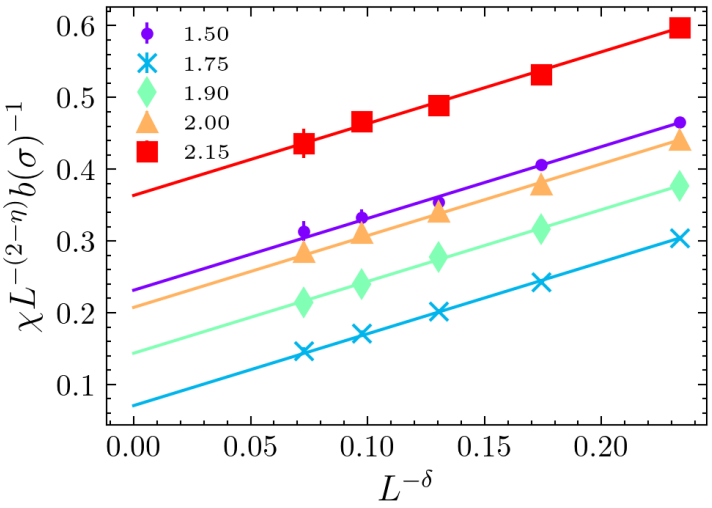

Figure 3.8: Linear dependency of $\chi L^{-(2-\eta)}$ respect to $L^{-\delta}$, with $\delta = 0.42$. Here we show only the σ near the LR-SR crossover. The straight lines were extrapolated using a linear fit of the points. We have checked that these fits gave a reduced χ^2 close to unity, validating the linear trend of the points. The lines are quite parallel to each other, meaning that the strength of the finite-size effects is due to the value of the intercept a, which is minimum near $\sigma = 1.75$, where it is known these effects are stronger.

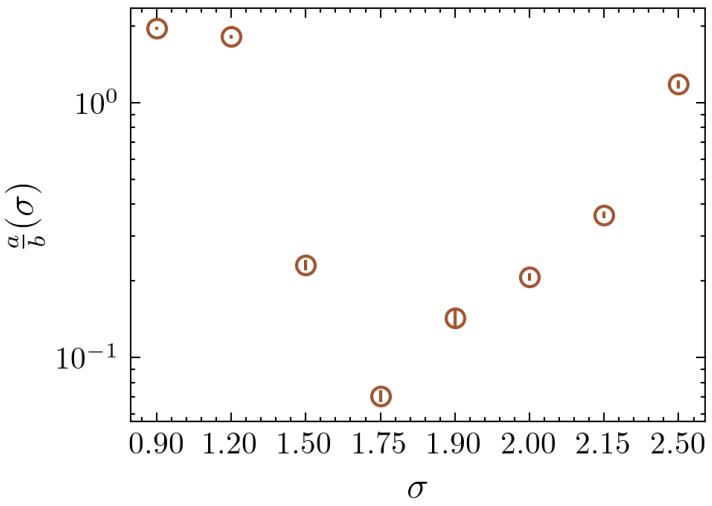

Figure 3.9: a/b ratio at various σ. The ratio of the linear fit coefficients is shown for the whole set of σ simulated. We choose to put the logarithmic scale on the y-axis to better highlight the sharpness of the minimum, which is found at $\sigma = 1.75$. This study validates the already formulated hypothesis that finite-size effects are much stronger near the LR-SR crossover.

3.4.2.2 Dynamics (z exponent)

We conclude our analysis by extrapolating the dynamical critical exponent z for a 2D spin lattice. We have already argued what are the expectations in Section 3.2. In the Mean

Field region, due to the disappearance of the geometry, the relaxation time is expected to scale as $L^{d/2}$, i.e. $z = 1$ for the 2D lattice. At $\sigma = d/2$, the model is Gaussian, meaning the system is in the free super-diffusive regime, which implies for Long Range systems $z = 2d/d_s = \sigma$. Then, in the Long Range region, we expect z to be slightly greater than σ, since the interactions perturb the free random walk behavior. Finally, in the Short Range region, z should be equal to the one of the corresponding short-range model. The most recent estimate of the z exponent in a Short Range 2D Ising model is $z = 2.14 \pm 0.02$, as taken from [45]. It is not necessarily true that Sak boundaries are valid also for the dynamical critical behavior. If this was the case, in the Long Range region z should deviate rapidly from the free super-diffusive regime $z = \sigma$, to recover the short-range value at $\sigma = 1.75$. However, our simulations invalidate this scenario, since they suggest the short-range regime is recovered smoothly at $\sigma = 2$. We collect our results in Fig. 3.10 and Table 3.4. In this picture, the black dashed line represents the expectation if the Sak scenario can be applied to dynamical properties, while the lighter one shows the behavior of z when this hypothesis is no longer valid. The dark horizontal line is the value of z with its error of the 2D Short Range Ising model, as found in [45]. By looking at the value of z for $\sigma = 1.75$, it is evident that the effect of interactions does not perturb strongly the free super-diffusive behavior, and from the value of z at $\sigma = 1.90$ Sak picture is clearly violated and the short-range behavior is recovered smoothly for $\sigma \geq 2$. From this picture we have just highlighted, more extensive simulations are needed to better understand the dynamical critical properties of the 2D Long Range Ising model in the range $\sigma \in (1.75, 2)$. This would also benefit from an analytical Renormalization Group argument, although it lies outside the scope of our current study.

σ	z extrapolated
0.5	1.01 ± 0.04
0.9	1.01 ± 0.04
1.2	1.2 ± 0.035
1.5	1.55 ± 0.04
1.75	1.76 ± 0.02
1.9	1.91 ± 0.03
2.0	2.016 ± 0.035
2.15	2.07 ± 0.06
2.5	2.12 ± 0.04

Table 3.4: Numerical details of our z extrapolation for a 2D spin lattice. In this table, we collect the numerical values of the points in Fig.3.10, which highlight the agreement with the expectations and the discovered behavior in the region $\sigma \in [1.75, 2]$.

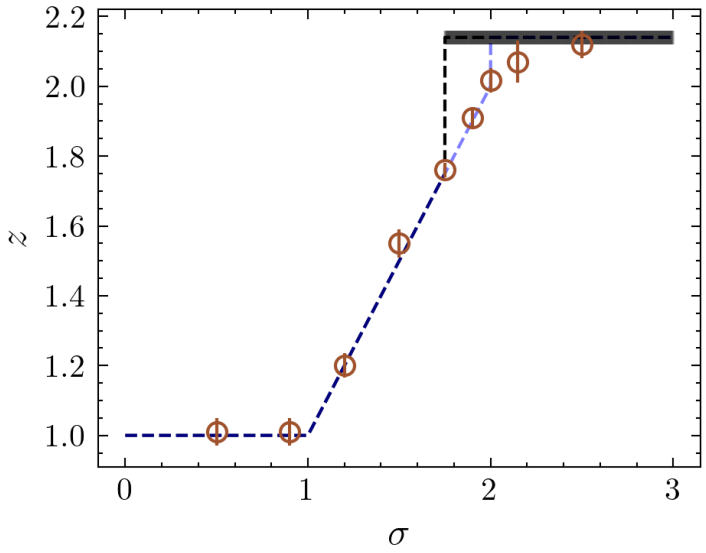

Figure 3.10: z exponent for a 2D spin lattice. The dashed lines represent the leading order of theoretical expectations. The black one shows the expected behavior if the Sak scenario is valid also for dynamical properties. When this is no longer true, we expect z to follow the lighter dashed line, for which the short-range behavior is recovered at $\sigma = 2$. The dark horizontal line is the value of z with its error of a 2D Short Range Ising model, as taken directly from [45]. We found good agreement both in the Mean Field region ($z = d/2$) and in the Long Range one, up to $\sigma = 1.75$, ($z = \sigma$). The value of z at $\sigma = 1.75$ suggests the long-range interactions poorly deviate the behavior of the system from a free super-diffusive regime. Moreover, the result of $\sigma = 1.90$ highlights the fact that the short-range regime is recovered smoothly at $\sigma = 2$, invalidating the Sak scenario for dynamical critical properties.

3.4.3 2D Long Range XY

We have collected evidence that our algorithm, the DLL, can numerically simulate the Long Range Ising model [46]. Moreover, due to the local nature of the DLL, it can be exploited to study various complex systems, for example, the Long Range XY model. In this final section, we will explain our methods and the early results of our preliminary work on this model.

In Section 2.2.2 we have seen that long-range interactions change the behavior of the XY model, and Renormalization Group calculations [14] highlight the presence of three regions of interest depending on σ:

- For $\sigma \in (0, 7/4)$ the system exhibits a ferromagnetic-paramagnetic phase transition. Moreover, it appears that the connected spatial correlation function assumes a power-law behavior in the whole ferromagnetic phase $\langle \mathbf{S}(0) \cdot \mathbf{S}(r) \rangle_c \equiv \langle \mathbf{S}(0) \cdot \mathbf{S}(r) \rangle - \langle \mathbf{S}(0) \rangle^2 \sim r^{\sigma-2}$.
- For $\sigma \in (7/4, 2)$ the system presents two phase transitions. Raising the temperature, the system passes from a magnetized phase to a BKT one, and eventually, it undergoes a BKT-paramagnetic phase transition.
- For $\sigma > 2$ the system behaves like the short-range model and a BKT-paramagnetic phase transition is present. In the BKT phase, the connected spatial correlation is again a power-law, with a temperature-dependent exponent $\eta(T) \propto T$.

In order to test these predictions, we choose to extrapolate the so-called Binder cumulant of the magnetization, a statistical observable that can serve as a phase marker. It is defined as

$$B(m) \equiv 2 - \frac{\langle m^4 \rangle}{\langle m^2 \rangle^2}\,, \tag{3.19}$$

where m is the magnetization per site. It can be shown that, in the thermodynamic limit, $B(m) = 1$ in the ferromagnetic phase, while $B(m) = 0$ in the paramagnetic one. In the BKT phase, it assumes a value between 0 and 1, depending on the temperature $B(m)_{BKT} = B(m, T)$. Numerical studies suggest that the value of the Binder cumulant at the BKT transition, T_{BKT}, is close to 1: $B(m, T_{BKT}) \approx 0.982$ [47]. For this reason, following the strategy designed in [34], we study a quantity derived from the Binder

cumulant, to better highlight the deviation from the value 1:

$$\tilde{B}(L) = \log\left(B(m,L)^{-1} - 1\right) , \tag{3.20}$$

where $B(m, L)$ denotes the value of B at the particular finite-size L of the system simulated. In this way, for the three phases, in the thermodynamic limit, we expect:

- In the paramagnetic phase, $\tilde{B} \to +\infty$, since $B \to 0$;
- In the ordered phase, $\tilde{B} \to -\infty$, since $B \to 1$;
- In the BKT phase, $\tilde{B} \to \tilde{B}^*(T)$, since B approach a temperature-dependent value $B^*(T)$.

In order to find the asymptotic behavior of $\tilde{B}$, we study the trend of its logarithmic discrete derivative, i.e. we compute how goes $|\tilde{B}(L) - \tilde{B}(2L)|$ with growing sizes. In the ferromagnetic (paramagnetic) phase, we expect this derivative to reach asymptotically a negative (positive) constant, so $\tilde{B}$ can diverge to $+\infty$ ($-\infty$). Meanwhile, in the BKT phase, the derivative should approach 0, since $\tilde{B}$ tends to be a constant in the thermodynamic limit.

For our preliminary experiments, we simulate three values of σ, one for each region of interest: $\sigma = 1.2, 1.8, 4$. The linear sizes considered are $L = 16, 32, 64, 128, 256$ for $\sigma = 1.2$ and 4, while we add also $L = 512$ and $L = 1024$ in the more interesting intermediate region.

In Fig. 3.11, we show our results for the ferromagnetic-paramagnetic region. It is evident that this system undergoes a phase transition at a temperature between $T = 0.68$ and $T = 1.19$. The two higher temperatures send $\tilde{B}$ to increasing positive values, while for the other temperatures tested this quantity seems to quickly decreasing. Indeed, by looking at the logarithmic derivative, we find it reaches a negative value already with small sizes. These are clear signals that for this value of σ, the system undergoes the expected magnetized-disordered phase transition.

In the short-range region, we simulate a system with $\sigma = 4$ and we collect the results in Fig. 3.12. Again, it is clear that there is some phase transition between the temperatures $T = 0.42$ and $T = 0.6$ since for the latter, $\tilde{B}$ is fast growing with the size. Meanwhile,

the lower temperatures seem to reach an asymptotic value. This can be checked from the derivative plot, where it is evident that the difference $|\tilde{B}(L) - \tilde{B}(2L)|$ approaches zero with respect to increasing sizes.

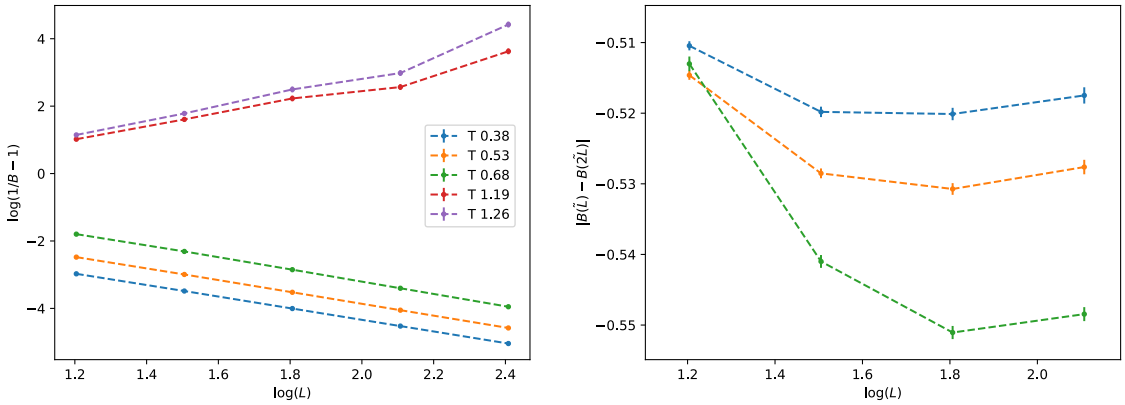

Figure 3.11: Behavior of the Binder cumulant at $\sigma = 1.2$. On the left, we show how $\tilde{B} \equiv \log\left(B(m, L)^{-1} - 1\right)$ behaves at different temperatures. It is quite clear that the two higher temperatures diverge positively when increasing sizes, while the others quickly decrease. This last phenomenon is verified by the behavior of the logarithmic derivative on the right, where it is pictured how they approach a negative constant value.

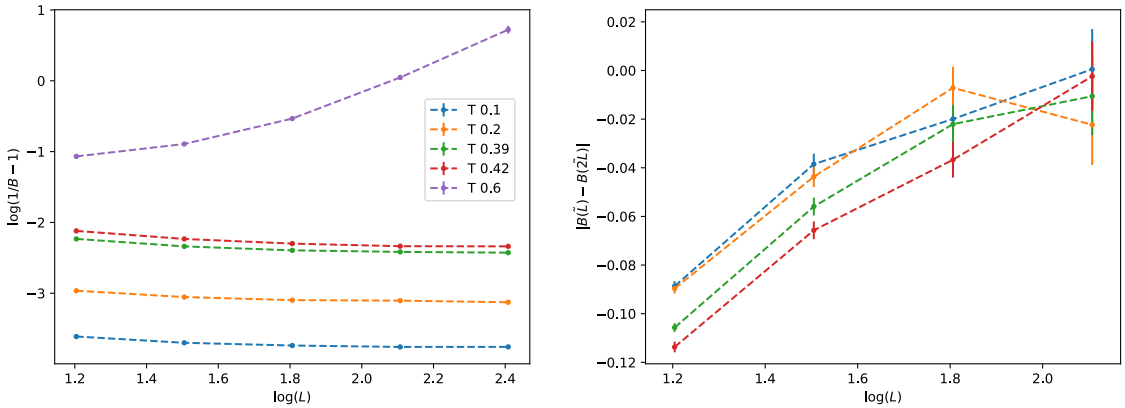

Figure 3.12: Behavior of the Binder cumulant at $\sigma = 4$. Here we show the trend of $\tilde{B}$ in the short-range region, for $\sigma = 4$. At high temperatures, this quantity diverges, highlighting the presence of a paramagnetic phase. For low temperature, it seems $\tilde{B}$ reaches an asymptotic value, suggested also by the behavior of the derivative, on the right. It is evident that the derivatives approaches quickly the zero value, meaning $\tilde{B}$ becomes constant in the thermodynamic limit.

There is empirical evidence that our algorithm is able to capture the expected behavior for a 2D XY Long Range model, in the regions $\sigma \notin (7/4, 2)$. For this reason, we move to

study the intermediate region $\sigma \in (7/4, 2)$. Here, we expect all the three behaviors already highlighted: some temperatures with $\tilde{B}$ positively and negatively diverging and some with a constant value of it. We simulated several temperatures, ranging three orders of magnitude, and in Fig. 3.13, we show a sample of them, $T = 1.0$, 0.72, 0.667, 0.023, 0.001. At $T = 1.0$, we find the paramagnetic behavior and we avoid simulating the size $L = 1024$ since it is computationally demanding. Then, we check that $T = 0.72$ is the highest temperature that does not present a positively diverging behavior of $\tilde{B}$. For every temperature below this one, we extrapolate a ferromagnetic-like behavior. It seems that the derivative starts from negative values and increases with the size, but then it reaches a plateau far from zero. With our experiments and sizes simulated, we can not still say if the absence of a BKT phase is due to strong finite-size effects or if it derives from some approximations done in Renormalization Group calculations. The same results were found in [34], where $\sigma = 1.875$ were simulated. Then, it is clear that the Binder strategy employed is not able to highlight the presence of a triple phase in the intermediate region.

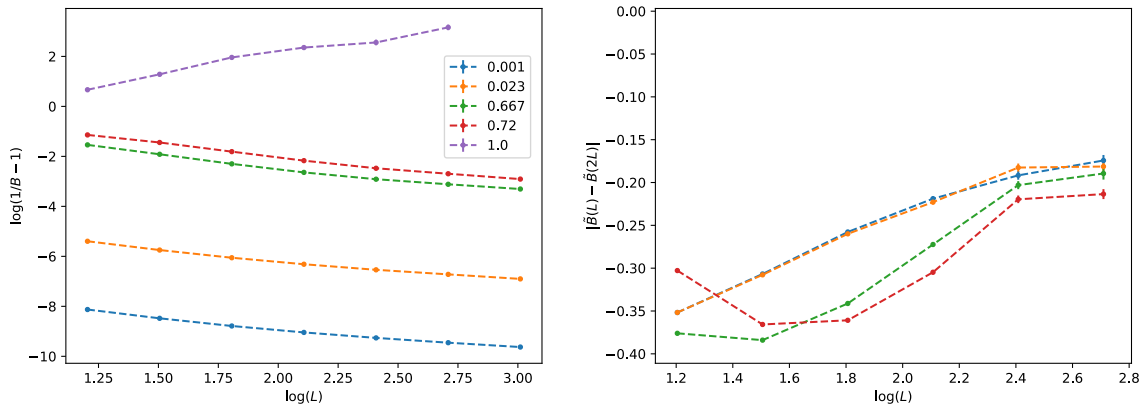

Figure 3.13: Behavior of the Binder cumulant at $\sigma = 1.8$. Here we show the behavior at some temperatures at $\sigma = 1.8$. At the highest temperature, $T = 1$, the system is clearly in the paramagnetic phase, and there was no need to simulate bigger sizes than $L = 512$. For all the temperatures below $T = 0.72$, we find an asymptotic behavior quite similar to a ferromagnetic one, as was already found in [34]. However, at this stage of our work, we can not exclude that the absence of a BKT phase is due to strong finite-size effects. In the picture on the right, we deliberately show a wide range of the y-axis, to better highlight the distance from the expected value 0 for a BKT phase.

Since the Binder strategy is not sufficient for gaining insights about the intermediate region, we are currently studying the critical exponent governing the connected spatial correlation function $G_c(r_{ij}) \equiv \langle \mathbf{S_i} \cdot \mathbf{S_j} \rangle_c \equiv \langle \mathbf{S_i} \cdot \mathbf{S_j} \rangle - \langle \mathbf{S_i} \rangle^2$, where the average is over both the Gibbs

ensemble and the lattice points, and r_{ij} is the distance between the two spins. In the ferromagnetic region, we recall we expect to recover the Sak scenario $G_c(r) \sim r^{-\eta} \sim r^{\sigma-2}$, while in the BKT phase, the exponent η depends linearly on temperature. Dealing with systems having finite linear size L, we want to study the scaling form of $G_c(r)$

$$G_c(r, L) = L^{-\eta} g\left(\frac{r}{L}\right) , \tag{3.21}$$

with $g(r/L)$ a scaling function. The goal is to extrapolate η, such that if we have $G_c(r)$ for different sizes and we rescale them properly, then they will collapse onto the same curve, as when we extrapolated the dynamical critical exponent z for the Long Range Ising model (see for example Fig. 3.3). In order to do so, we numerically compute $G_c(r, L)$ for every size, with r ranging from 1 to $L/2$. Then, we consider the values taken at the same ratio r/L and we perform a linear fit following the scaling law

$$\log(G_c(r(L), L)) = -\eta \log(L) + C \, , \tag{3.22}$$

where $r(L)$ denotes the fact we are choosing a suitable distance such that r/L is equal for any size, and C is a constant independent from L. Since we collect many distances, we extrapolate the value of η for many $r(L)$ and we consider the average and standard deviation over this sample of ηs. To get rid of finite-size effects, when fitting Eq. (3.22), we employ the same strategy already discussed in Section 3.3 (see Fig. 3.2).

In the region $\sigma \in (0, 7/4)$, we extrapolate η for two values, $\sigma = 0.5$, 1.2. We collect our results in Fig. 3.15. It is clear that in this region, we can find the expected Sak law $\eta = 2 - \sigma$. In Fig. 3.14, we show also how well the found exponent collapses the curves for the $\sigma = 1.2$ case.

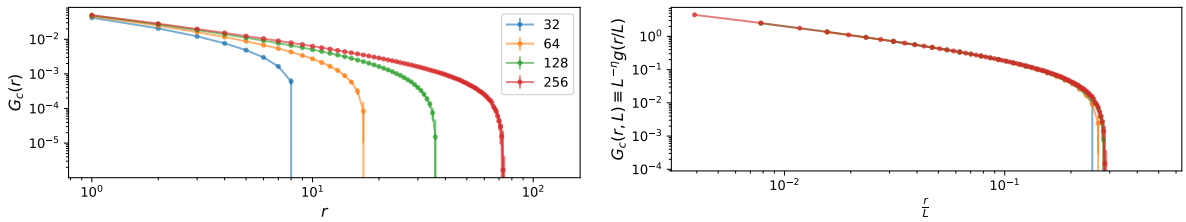

Figure 3.14: Collapsing the $G_c(r)$ curves with the η found with our strategy. In this example, we show that our method is able to find the correct exponent for collapsing the curves $G_c(r)$ at different linear sizes.

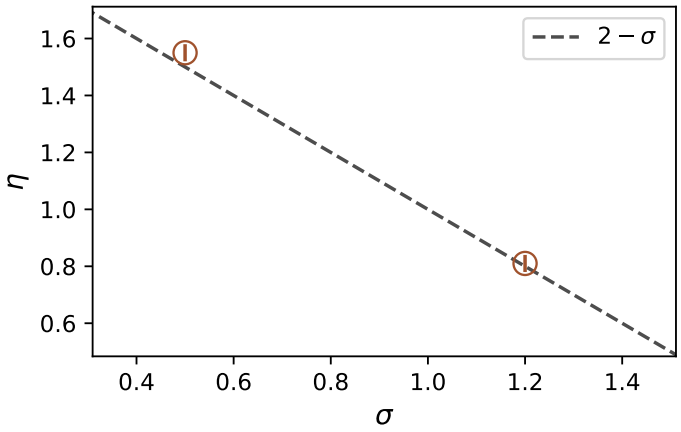

Figure 3.15: η extrapolation for the ferromagnetic-paramagnetic region $\sigma \in (0, 7/4)$. We extrapolate η for $\sigma = 0.5$ and $\sigma = 1.2$. We recover Sak law $\eta = 2-\sigma$, indicated in the plot with the black dotted line. The exponents found are $\eta(\sigma = 0.5) = 1.55 \pm 0.03$ and $\eta(\sigma = 1.2) = 0.81 \pm 0.03$

In the Short Range region $\sigma > 2$, we expect to recover the short-range XY behavior. If the system is in the BKT phase, then the exponent η should present a linear dependency on the temperature. In Fig. 3.16, we show our results for $\sigma = 2.5$ and it seems η follows the expected linear scaling.

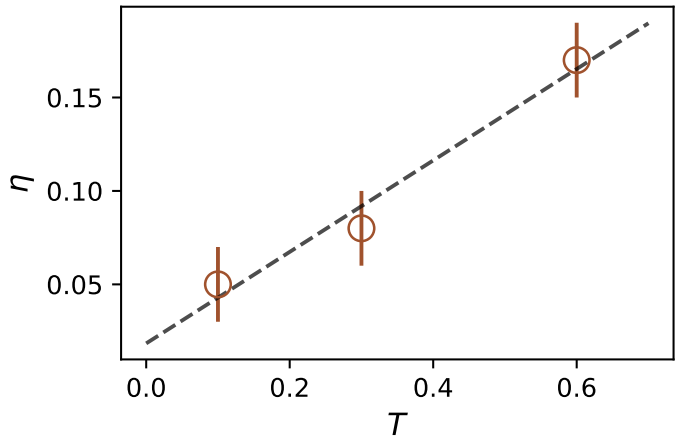

Figure 3.16: $\eta(T)$ at $\sigma = 2.5$. In the temperature region tested, the extrapolated exponent η seems to follow a linear dependency on T (the black dotted line represents the best linear fit we found).

Finally, we manage to test the intermediate region $\sigma \in (7/4, 2)$ and we extrapolate η for the same temperatures tested at $\sigma = 1.8$ (Fig. 3.13). Here, we should expect a different behavior depending on the temperature. At low temperatures, η should follow the usual

Sak law $\eta = 2 - \sigma$, while at higher temperatures (below T_{BKT}) η should be a linear function of T. We collect our results in Fig. 3.17. It is clear that from our preliminary analysis, both the expected behaviors are not recovered. Again, at this stage, we can not claim that the found discrepancies are not due to strong finite-size effects. Indeed, one can ask if a correction like the one in Fig. 3.7 is sufficient to get rid of these effects. Using the same correction with $\delta = 0.42$, the blue stars in the figure, we found a behavior that seems to be in line with theoretical expectations. At low temperatures, η seems constant and near to the value $2 - \sigma = 0.2$, while at higher temperatures a linear trend is suggested. For these reasons, we plan to investigate this region with more intensive simulations. Moreover, a more involved study on the correction δ should be performed.

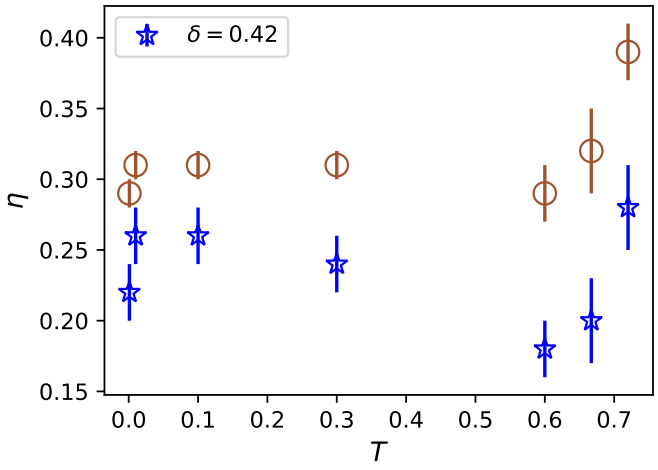

Figure 3.17: Behavior of η at different T at $\sigma = 1.8$. It is clear that neither the $\eta = 2 - \sigma$ nor the linear dependency is highlighted by our analysis. We conjecture these discrepancies are due to strong finite-size effects. Indeed, by employing the same strategy used for the Ising model (see Fig. 3.7), we observe a behavior quite similar to the one expected. We plan to investigate this region with more intensive numerical simulations.

Part II

Deep Neural Networks

4 Deep Learning theory in the Infinite-Width limit

In the contemporary era, the advent of deep learning models has marked a significant milestone in the realm of artificial intelligence. These models, characterized by their deep architectures and vast number of parameters, have demonstrated proficiency in numerous tasks, ranging from image recognition to natural language processing. Their success has not only reshaped industries but has also piqued the curiosity of the broader scientific community.

However, with this surge in popularity and application, there arises a pressing challenge: the intricacy of these models often renders them as "black boxes." The complexity inherent in their design, combined with the multitude of layers and connections, makes it daunting to decipher their inner workings. This opacity is not just a theoretical concern but has practical implications, especially when these models are employed in critical applications where transparency and interpretability are paramount.

It is worth noting that the landscape of machine learning is vast, encompassing a variety of learning problems. These can be broadly categorized into supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning, among others. Each of these paradigms addresses distinct challenges and is suited for specific types of tasks. For instance, unsupervised learning focuses on finding patterns in unlabeled data, while reinforcement learning is concerned with decision-making in dynamic environments.

In the context of this chapter, our primary focus will be on supervised learning. This paradigm, where models are trained using labeled data to make predictions or classifications, serves as the backbone for many of the most celebrated achievements of deep learning. As we delve deeper into the statistical mechanics of neural networks, we'll explore how the principles of this discipline can shed light on the behavior of supervised learning models, offering a clearer understanding of their strengths, limitations, and potential avenues for innovation.

4.1 Supervised learning

The starting point for defining a supervised learning problem is the **training set** $\mathcal{D}_P$. This set consists of a number P of input-output pairs, where the input represents some data or pattern, and the output represents a particular label associated with the corresponding input. From now on, for simplicity, we will consider vectors belonging to a high dimensional space as input, and binary outcomes as output. Then, the training set is defined as

$$\mathcal{D}_P = \{\mathbf{x}^\mu, y^\mu\}_{\mu=1}^P \ , \tag{4.1}$$

where $\mathbf{x}^\mu \in \mathbb{R}^{N_0}$, for some integer N_0, and $y^\mu \in Y$, where Y denotes a general set, which can be discrete for classification tasks or continuous for regression ones. Formally, we assume that the pairs $(\mathbf{x}, y)$ are random variables distributed according to an unknown joint distribution $P(\mathbf{x}, y)$, and the particular training set $\mathcal{D}_P$ is a random instance of this distribution. Then, the goal of any learning algorithm is to find a function $f : \mathbb{R}^{N_0} \to \{0, 1\}$ that approximate the conditional probability distribution $P(y|\mathbf{x})$, such that for a new pair $(\tilde{\mathbf{x}}, \tilde{y}) \sim P(\mathbf{x}, y)$ drawn from the same probability distribution of the training set, it predicts with high probability the correct output, i.e. $f(\tilde{\mathbf{x}}) = \tilde{y}$. In order to find this function, one has to restrict the search within a particular set of possible functions, also called the **hypothesis class**. For our scopes, we will focus on functions that implement artificial neural networks (ANN). ANNs are computational graphs made of neurons, which are a model of the biological neurons in the brain, in the sense that they receive a signal as input and they eventually propagate it when a certain threshold is exceeded. ANNs are usually defined by consecutive layers of neurons, where each one takes as input the output of the previous layer and applies a non-linear transformation. In general, these transformations depend on the neural network architecture chosen. In the following, we will describe the two types of architecture that are the focus of our studies, the Fully-Connected neural Network, and the Convolutional Neural Network.

4.1.1 Fully-Connected neural Network (FCN)

We now describe the easiest architecture one can think of, the Fully-Connected neural Network (FCN), also called Multi-Layer Perceptron (MLP). Each layer ℓ of an FCN

consists of N_ℓ neurons, associated with a real-valued number. The value carried by a neuron depends on the value of all the neurons in the previous layer, whose signal is transformed with the application of a non-linear affine transformation. This is defined by a matrix of **weights** $\mathbf{W}^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ and a vector $\mathbf{b}^{(\ell)} \in \mathbb{R}^{N_\ell}$ called **bias**, followed by a non-linear function $\sigma^{(\ell)}$ applied element-wise. In this way, the signal is propagated through the layers until the output layer is reached. In Fig.4.1 we show an example of how FCN architectures are usually represented.

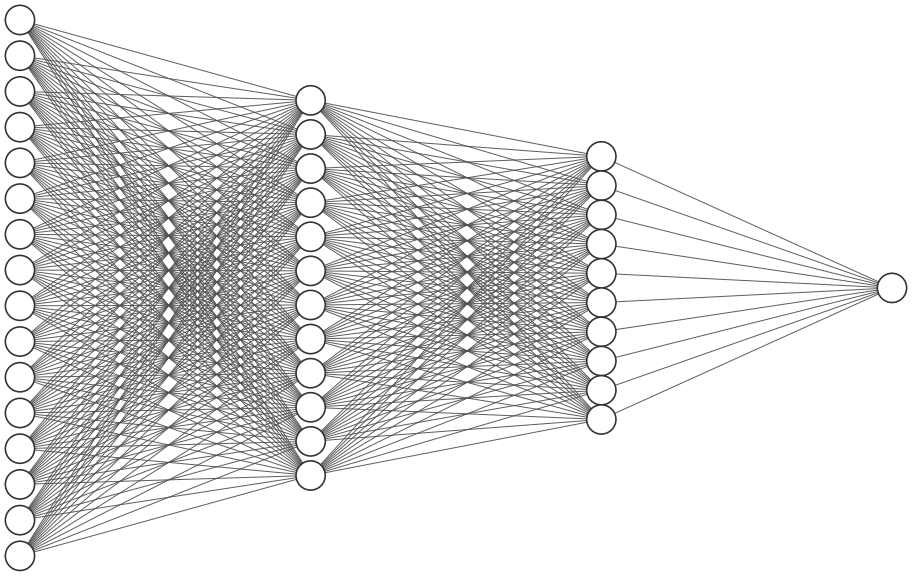

Figure 4.1: Example of a FCN architecture. In this picture is shown an example of a FCN architecture. The circles represent the neurons and the links between them are the affine transformations which depend on the weights and biases. In this example, the input is a 16-component vector, while the output is binary.

It is useful to define the **pre-activation** $\mathbf{h}^{(\ell)}$ of a layer as

$$h_i^{(\ell)}(\mathbf{x}) = \sum_{j=1}^{N_{\ell-1}} W_{ij}^{(\ell)} \phi_j^{(\ell-1)}(\mathbf{x}) + b_i^{(\ell)} \tag{4.2}$$

where $\phi(\mathbf{x})$ is the activation vector and is defined as $\phi_i^{(\ell)}(\mathbf{x}) \equiv \sigma^{(\ell)}\left(h_i^{(\ell)}(\mathbf{x})\right)$. We collect all the parameters in each layer in the set $\theta = \left\{(\mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})\right\}_{\ell=1}^{L}$, where L is the number of layers, also called network's **depth**. Finally, we can define the function which implements a FCN architecture as

$$f_\theta(\mathbf{x}) \equiv \sigma^{(L)} \circ \mathbf{A}^{(L)} \circ \sigma^{(L-1)} \circ \mathbf{A}^{(L-1)} \circ \cdots \circ \sigma^{(1)} \circ \mathbf{A}^{(1)}(\mathbf{x})\,, \tag{4.3}$$

where we have used a compact notation for the affine transformation $\mathbf{A}^{(\ell)}(\mathbf{x}) \equiv \mathbf{W}^{(\ell)} \cdot \mathbf{x} + \mathbf{b}^{(\ell)}$.

4.1.2 Convolutional Neural Network (CNN)

In the realm of artificial neural networks, Convolutional Neural Networks (CNNs) are a specialized kind designed for processing grid-like data structures, usually images. The very first idea about CNNs can be traced back to the work of Fukushima [48], where he introduced the concept of Neocognitron. However, it was Yann LeCun's work [49] in the late 1990s on the LeNet-5 architecture for digit recognition that truly characterized and started the development of CNNs.

The fundamental components of a CNN are the Convolutional layers. These layers perform a convolution operation, applying a filter to an input to produce a feature map. This operation involves a sliding filter (also called **mask**) over the input data and computing the dot product between the mask and the section of input it covers at each position. The mask slides a fixed number S of positions (or coordinates or pixels), referred to as **stride**. Mathematically, the convolution operation for a 2D input I and a filter K, considering $S = 1$, is given by:

$$(I * K)(x, y) = \sum_{i=-\lfloor \frac{M_x}{2} \rfloor}^{\lfloor \frac{M_x}{2} \rfloor} \sum_{j=-\lfloor \frac{M_y}{2} \rfloor}^{\lfloor \frac{M_y}{2} \rfloor} I(x - i, y - j) \cdot K(i, j)\,, \tag{4.4}$$

where the filter dimensions are taken to be $M_x \times M_y$. In Fig. 4.2 we show a visual representation of the convolution operation.

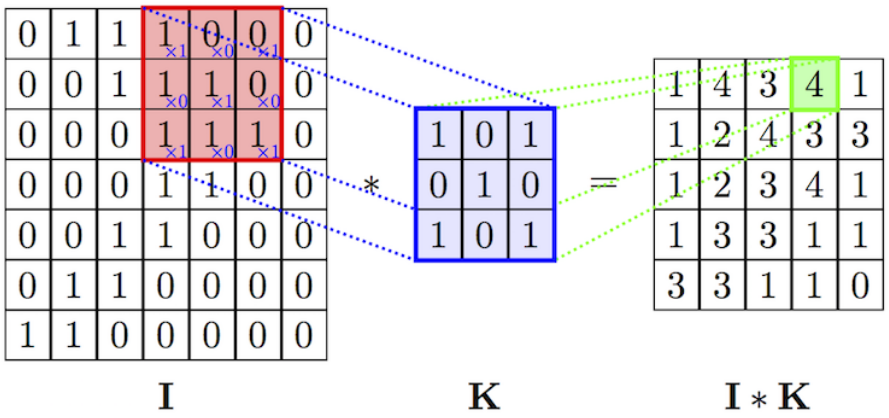

Figure 4.2: Example of convolution operation. This sketch is taken from https://anhvnn.wordpress.com/2018/02/01/deep-learning-computer-vision-and-convolutional-neural-networks/. It illustrates how the convolved image $I * K$ is built through the application of a filter K to an input data (an image) I. Each element (pixel) of $I * K$ is computed as the dot product between the filter and the corresponding portion of the input data.

The neurons on the mask interact only with a local neighborhood of neurons in the previous layer, implementing the so-called **local connectivity**. In the context of NNs, the convolution filter K is made of weights and biases that can be updated through the learning phase. Since the filter is sliding through the entire input data, the neurons of the feature map (the convolved image) share the same set of weights and biases. This principle drastically reduces the number of parameters in the network and it ensures that the same feature can be detected anywhere in the input (the convolution operation is said to be *equivariant* under shifts of the locations of input features). This fundamental property is known as **weight sharing**, and we will see how it is crucial. Finally, multiple feature maps are stacked on top of each other by applying a set of independent filters, each one called **channel**. Each channel has its own set of weights and in principle, it detects different features in the input. Finally, the feature maps are collected as single neurons in a one-dimensional layer, which serves as a hidden layer of an FCN. In this way, a final output can be produced. We schematize the most simple CNN architecture in Fig. 4.3: a single convolutional layer, followed by the collection of all features maps (Flatten operation) and finally a Fully-Connected (Dense) layer which produces the output.

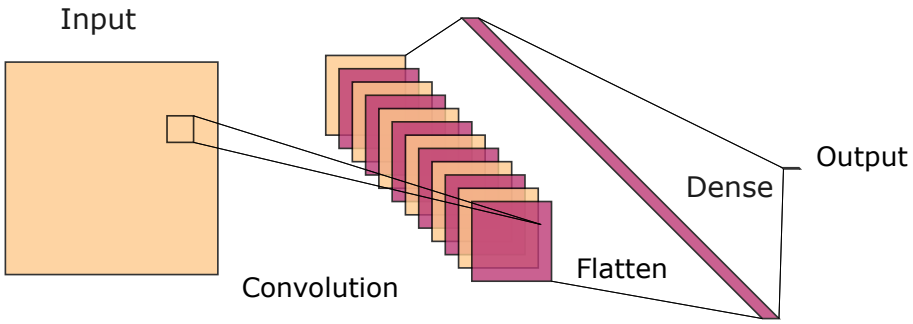

Figure 4.3: Example of the simplest CNN architecture. This picture shows the simplest CNN architecture: a convolution layer, followed by a flattened operation (the feature maps are collected in a one-dimensional layer of neurons), and finally a Fully-Connected (Dense) operation that produces the output (represented as a single neuron). In this example, the convolutional layer has 12 independent channels.

Note that, we can use the same notation of Eq. (4.3) for representing a CNN function, obviously changing the definition of the affine map.

4.1.3 Loss Function

It can be proved that a function in the form of f_θ (4.3), whatever the affine transformation is, satisfies the so-called **Universal Approximation Theorem** [50], which stands that for every function g defined on a compact space $K \subseteq \mathbb{R}^{N_0}$ and every $\epsilon > 0$, there is a particular choice of θ such that $\sup_{x \in K} |f_\theta(\mathbf{x}) - g(\mathbf{x})| < \epsilon$.

Now that the type of function that would solve the supervised problem is defined, the next step is to find the best function within the hypothesis class. This step actually defines the learning procedure and it often involves an optimization schedule. In order to find the best solution for a particular problem, a definition of *best* is required, which accounts for the evaluation of a hypothesis function. This is done through the definition of the so-called **Loss Function** $\mathcal{L} : Y \times Y \to \mathbb{R}$, which takes as input the labels of the set of data and the outputs of the network, and returns the distance between them. There are several choices of $\mathcal{L}$, depending on the kind of supervised problem. For our scopes we will focus only on **regression** problems, where the labels are a continuous real-value vectors $\mathbf{y} \in \mathbb{R}^k$, and typically the chosen Loss Function is the **Mean Squared Error**, defined as

$$\mathcal{L}_{MSE} \equiv \frac{1}{P} \sum_{\mu=1}^{P} \|\mathbf{y} - f_\theta(\mathbf{x})\|^2 . \tag{4.5}$$

Given a Loss Function $\mathcal{L}$ and an architecture f_θ, the goal is to find the set of parameter θ, which minimizes $\mathcal{L}$, i.e. the distance between original labels and network's predictions. One starts with a random initialization of the parameters θ and then evolves them. The evolution follows the gradient descent of the Loss Function and the dynamics of the parameters through the learning algorithm is described by the relation $\dot{\theta} = -\nabla_\theta \mathcal{L}$.

An efficient way for computing the gradient of $\mathcal{L}$ with respect to the weights is the **backpropagation algorithm**. Briefly, it computes the gradient one layer at a time and then iterates backwardly from the last layer to avoid redundant calculations, exploiting the chain rule of differentiation. For simplicity, in the following, we will describe this mechanism in the case of an FCN. It is easy to prove through induction that the dynamics of the weights $\mathbf{W}$ are described by the following relation

$$\dot{W_{ij}^{(\ell)}} = -\frac{\partial \mathcal{L}}{\partial W_{ij}^{(\ell)}} = q_i^{(\ell)} \phi_j^{(\ell)} \,, \tag{4.6}$$

where

$$\begin{aligned} q_i^{(\ell)} &= \sum_{j=1}^{N_{\ell+1}} q_j^{(\ell+1)} W_{ji}^{(\ell+1)} \sigma'(h_i^{(\ell)}) \,, \\ q_i^{(L)} &= \left(y^\mu - \phi^{(L)}(\mathbf{x}^\mu)\right) \sigma'(h_i^{(L)}) \,, \end{aligned} \tag{4.7}$$

recalling that $\mathbf{h}^{(\ell)}$ and $\phi^{(\ell)}$ are respectively the pre and post-activation of the ℓ-th layer (see Eq. (4.2)), and where we have considered the same activation function σ for every layer. The whole process of finding the minimum of the Loss Function by adjusting the parameters θ is referred to as the **training phase**.

When dealing with a supervised problem, it would be useful to obtain an inference model capable of good performance on new examples, which are not visible during the training phase. For this purpose, it is defined the **generalization error** ϵ_g, which is a measure of how accurately an algorithm is able to predict outcome values for previously unseen data. It is defined as the average distance between the label y_0 of a new unseen pattern $\mathbf{x}_0$ and the output of the trained neural network. The formal definition of ϵ_g reads

$$\epsilon_g(f_\theta) \equiv \int d\mathbf{x} dy \, P(\mathbf{x}, y) \mathcal{L}(y, f_\theta(\mathbf{x})) \,, \tag{4.8}$$

where $P(\mathbf{x}, y)$ is the unknown joint probability distribution for the pairs $(\mathbf{x}, y)$. Clearly, it would be useful to build a theory that can predict the minimum ϵ_g reachable, given a desired function f_θ. For this reason, studying the generalization error is the core of any learning problem and this will be the topic of the following section.

4.2 The problem of generalization in overparameterized neural networks

Any learning algorithm aims to build a model of inference, which is gaining knowledge and being able to make predictions from a set of data. We have already seen that the starting point is to search for a function f_θ under some assumptions, which define the space of possible functions, the hypothesis class. For every f_θ, we can then compute the generalization error as

$$\epsilon_g(f_\theta) \equiv \mathbb{E}_{\mathbf{x},y}\left[(f_\theta(\mathbf{x}) - y)^2\right] = \int d\mathbf{x}dy\,(f_\theta(\mathbf{x}) - y)^2 P(\mathbf{x}, y)\,, \tag{4.9}$$

where $\mathbb{E}_z$ represents the expected value of a quantity over the probability distribution $P(z)$ of the variable z, and we consider the MSE as Loss Function, since we are dealing with a regression problem. Note that, f_θ is a function whose parameters are adjusted to predict pairs of the dataset, then it depends on the particular choice of $\mathcal{D}_P$. However, $\mathcal{D}_P$ is itself drawn from the distribution $P(\mathbf{x}, y)^P$ and it must be taken into account if we want a full understanding of the generalization error. Then, in the following, we explain an argument to quantify the **expected generalization error**, i.e. $\mathbb{E}_{\mathcal{D}}\left[\epsilon_g(f_\theta)\right]$. The first step is to introduce the expected value of f_θ over the dataset distribution, which for the sake of notation we will call $\bar{f}_\theta \equiv \mathbb{E}_{\mathcal{D}}\left[f_\theta(\mathbf{x})\right]$:

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[\epsilon_g(f_\theta)\right] \equiv \mathbb{E}_{\mathcal{D},\mathbf{x},y}\left[f_\theta(\mathbf{x} - y)^2\right] &= \mathbb{E}_{\mathcal{D},\mathbf{x},y}\left[((f_\theta(\mathbf{x}) - \bar{f}_\theta) + (\bar{f}_\theta - y))^2\right] = \\ &= \mathbb{E}_{\mathcal{D},\mathbf{x}}\left[(f_\theta(\mathbf{x}) - \bar{f}_\theta)^2\right] + 2\,\mathbb{E}_{\mathcal{D},\mathbf{x},y}\left[(f_\theta(\mathbf{x}) - \bar{f}_\theta)(\bar{f}_\theta - y)\right] + \\ &+ \mathbb{E}_{\mathbf{x},y}\left[(\bar{f}_\theta - y)^2\right]\,.\end{aligned} \tag{4.10}$$

It is easy to check that the double product term, the second one in the second row, is zero. The term $\mathbb{E}_{\mathcal{D},\mathbf{x}}\left[(f_\theta(\mathbf{x}) - \bar{f}_\theta)^2\right]$ is the **variance** of the random variable f_θ, over the

distribution of $\mathcal{D}$. This quantity is a measure of how much f_θ varies when changing elements in the training set and it tells how much the function specializes in predicting the particular drawn $\mathcal{D}$ and fails in predicting unseen data. The variance is a direct measure of the phenomenon called **overfitting**, and heuristically it grows with the complexity of the model.

The other term which is not zero, $\mathbb{E}_{\mathbf{x},y}\left[(\bar{f}_\theta - y)^2\right]$, can be rewritten by inserting the expected label given a particular input, referred to as $\bar{y} \equiv \mathbb{E}_{y|\mathbf{x}}[y]$:

$$\begin{aligned}\mathbb{E}_{\mathbf{x},y}\left[(\bar{f}_\theta - y)^2\right] &= \mathbb{E}_{\mathbf{x},y}\left[((\bar{f}_\theta - \bar{y}) + (\bar{y} - y))^2\right] = \\ &= \mathbb{E}_{\mathbf{x}}\left[(\bar{f}_\theta - \bar{y})^2\right] + \mathbb{E}_{\mathbf{x},y}\left[(\bar{y} - y)^2\right] + 2\,\mathbb{E}_{\mathbf{x},y}\left[(\bar{f}_\theta - \bar{y})(\bar{y} - y)\right] . \end{aligned} \tag{4.11}$$

Again the last term, the one coming from the double product, is equal to zero. The second term, $\mathbb{E}_{\mathbf{x},y}\left[(\bar{y} - y)^2\right]$ is the **noise** of the label in the dataset. It is an inevitable aspect of the data and it can not be made smaller.

Finally, the term $\mathbb{E}_{\mathbf{x}}\left[(\bar{f}_\theta - \bar{y})^2\right]$ is the square of the so-called **bias**. It represents how much the best possible function within the chosen hypothesis class will inevitably differ from the real unknown function which describes the relation between the pairs in the dataset. This can be made smaller by taking more complex models, for example, in the case of ANNs, by taking more neurons or adding layers.

In summarizing, given a distribution of data and a hypothesis class from which we draw a predictive function, the expected value of the generalization error can be decomposed in three terms, each one characterizing the learning problem considered. This decomposition is called **bias-variance decomposition**:

$$\mathbb{E}_{\mathcal{D}}\left[\epsilon_g(f_\theta)\right] = \underbrace{\mathbb{E}_{\mathbf{x},y}\left[(\bar{y} - y)^2\right]}_{\text{noise}} + \underbrace{\mathbb{E}_{\mathcal{D},\mathbf{x}}\left[(f_\theta(\mathbf{x}) - \bar{f}_\theta)^2\right]}_{\text{variance}} + \underbrace{\mathbb{E}_{\mathbf{x}}\left[(\bar{f}_\theta - \bar{y})^2\right]}_{\text{bias}^2} . \tag{4.12}$$

For better understanding, we draw a picture of how the generalization error should behave when adding complexity to the model in Fig. 4.4.

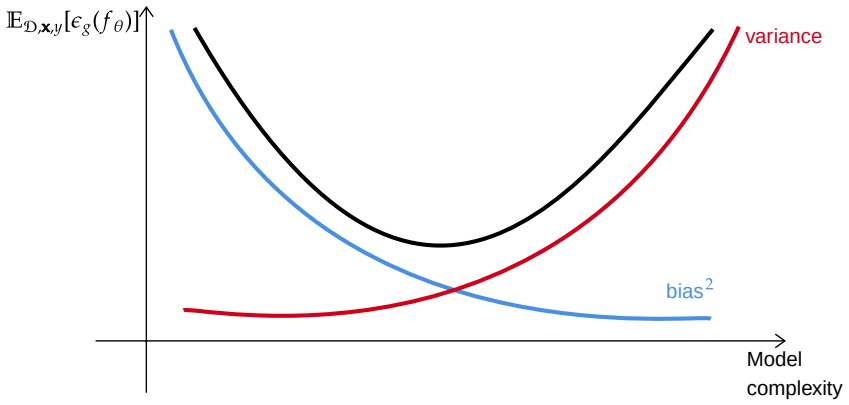

Figure 4.4: Bias-variance tradeoff. Sketch of the expected behavior of the generalization error trend with the model complexity. If we consider more complex models in the hypothesis space, the bias will reduce, because the possible functions will have more expressivity. Meanwhile, the variance will grow because of the *overfitting* phenomenon, i.e. the minimization of the Loss Function over the training set tends to select overspecialized functions, which fail to generalize on unseen data.

From the perspective of the bias-variance decomposition, for every learning problem, there should be an optimal choice of model complexity. In the case of neural networks, there should be a suitable number of layers and neurons, such that an optimal function can be learned. This function should achieve a small but non-zero error on the training set, to avoid overfitting, and should minimize the generalization error. However, in recent years, practitioners have been using increasingly larger neural networks for fitting datasets made of a large number of pairs, achieving errors on the training set near zero. Despite the high complexity of the functions and the near-perfect fit to training data, these big models often give accurate predictions on unseen data, breaking completely the previous picture. Indeed, empirical evidence shows that there is a certain value of complexity, called **interpolation threshold**, for which the model can perfectly interpolate the training data and the generalization error shows a maximum. However, after this point, ϵ_g seems to decrease monotonically, typically going below the minimum reached before the interpolation threshold. This phenomenon is called **double-descent**, and it is represented skeptically in Fig. 4.5.

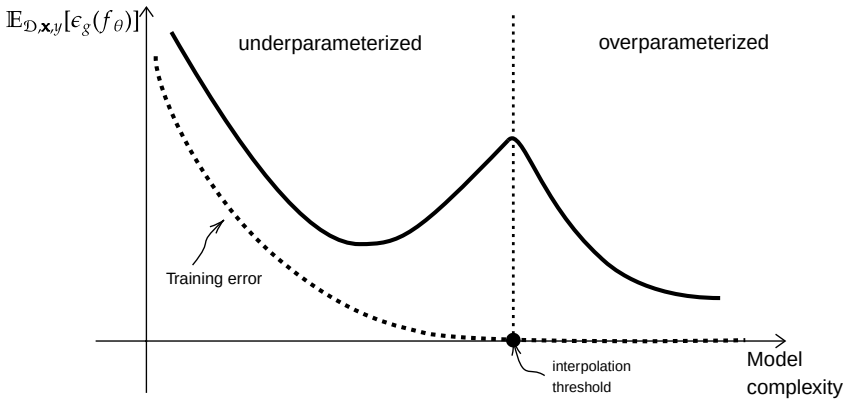

Figure 4.5: Double-descent. Sketch of the empirical behavior of the generalization error trend with the model complexity. There is a point, the interpolation threshold, which highlights two regimes, under and over-parameterized. Below this point, the training error is almost zero because the model is complex enough (it has a sufficient number of parameters) to perfectly interpolate the training data. Despite the bias-variance decomposition picture, adding complexity to the model has the effect of decreasing the generalization error, even to values lower than the minimum reached in the under-parameterized regime.

This peculiar phenomenon highlights the fact that there is something hidden underneath the simple picture of the bias-variance decomposition and a theory that explains with more details the behavior of the generalization is needed.

4.3 Kernel methods and Gaussian Processes

Many years before the rise of neural networks, practitioners have been focused on **kernel methods**, which represent a practical way for building a model of inference. Moreover, there is a particular case, the **Gaussian Processes**, from which the generalization error can be computed analytically and we will see they are also useful in describing overparameterized neural networks.

Basically, a kernel method is a way of performing a non-linear regression with the same cost as a linear one. In practice, the strategy is to build a learning algorithm for discovering linear patterns in a high-dimension space, where the inputs are embedded through a non-linear transformation. In order to explain kernel methods, we have to introduce first the concept of **linear regression**. Let us take the dataset $\mathcal{D}_P$, considering the targets $y \in \mathbb{R}$ and the input data $\mathbf{x} \in \mathbb{R}^{N_0}$. The goal of linear regression is to find a linear function $f(\mathbf{x}) = \sum_{i=1}^{N_0} W_i x_i$, that best interpolates the given training set. The strategy is to define

a Loss Function, which quantifies both the error committed by a particular function f and the norm of the weights $\mathbf{W}$. The standard Loss Function for linear regression is again the MSE, to which the contribution of the weights norm is added

$$\mathcal{L} = \frac{1}{2}\sum_{\mu=1}^{P}\left(y^{\mu} - \sum_{i=1}^{N_0} W_i x_i^{\mu}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{N_0} W_i^2\,, \tag{4.13}$$

where λ is a positive constant that controls the regularization, i.e. which one of the two terms is most considered in computations. In this way, we can select the function f that both minimizes the Loss Function and has the smallest weights. Finding the vector $\mathbf{W}$ which minimizes $\mathcal{L}$ is straightforward and can be done in two parallel ways. One is to directly solve the minimization problem and to find the suitable values of the weights:

$$\mathbf{W} = \left(\lambda\mathbb{1} + \tilde{\mathbf{C}}\right)^{-1}\mathbf{B}\,, \tag{4.14}$$

where $\tilde{\mathbf{C}}$ is defined as $\tilde{C}_{ij} \equiv \sum_{\mu=1}^{P} x_i^{\mu} x_j^{\mu}$, and the vector $\mathbf{B}$ is defined as $B_j \equiv \sum_{\mu=0}^{P} y_j^{\mu} x_j^{\mu}$. Note that, the role of the regularization term in $\mathcal{L}$ and the fact that $\lambda > 0$, ensures the invertibility of the matrix $\lambda\mathbb{1} + \mathbf{C}$, and this implies the existence of a solution. Solving this system of N_0 linear equations has a computational cost of $O(N_0^3)$. Another strategy is to consider linear combinations of the training examples. Indeed, it is easy to see that the weights can be expanded as $\mathbf{W} = \sum_{\mu=1}^{P} \alpha^{\mu}\mathbf{x}^{\mu}$, where the coefficients α are computed as $\alpha = (\lambda\mathbb{1} + \mathbf{C})^{-1}\mathbf{y}$, where $\mathbf{y}$ denotes the vector that collects all the labels. Here, $\mathbf{C}$ denotes the covariance matrix (also called Gram matrix) of the training set, i.e. $C_{\mu\nu} \equiv \sum_{i=1}^{N_0} x_i^{\mu} x_i^{\nu}$. In this case, finding α requires an $O(P^3)$ operations, and so, depending on the number and the dimensionality of the input data, one can choose the most efficient way. In any case, the key point of the linear regression problem is that finding the solution requires only the **inner product** between elements of the training set, and this concept will be extended in kernel methods.

This whole procedure is efficient and can give insights into the relations between the training set pairs, but in most cases, this is not sufficient. In general, these relations are highly non-linear and the approach of linear regression will lead to wrong predictions. Here come kernel methods. In few words, the strategy is to map through a non-linear function $\phi : \mathbb{R}^{N_0} \to \mathbb{R}^{N_\phi}$ the input data into a high dimensional space $\mathbb{R}^{N_\phi}$, called **feature**

space, such that the relations of the embedded dataset, whose elements are $(\phi(\mathbf{x}^\mu), y^\mu)$, are nearly linear. Thus, kernel methods aim to find a suitable embedding map ϕ and then to perform a linear regression in the feature space. To do that, one has to repeat the previous reasoning about linear regression, substituting any inner product with the so-called **kernel**. A kernel is defined as a positive semi-definite map $K : \mathbb{R}^{N_0} \times \mathbb{R}^{N_0} \to \mathbb{R}$, which represents the inner product after the transformation ϕ, i.e.

$$K(\mathbf{x}^\mu, \mathbf{x}^\nu) \equiv \sum_{i=1}^{N_\phi} \phi_i(\mathbf{x}^\mu)\phi_i(\mathbf{x}^\nu)\,. \tag{4.15}$$

Note that, by definition $K(\cdot,\cdot)$ is positive semi-definite. Thus, one has to search for a function in the form $f(\mathbf{x}) = \sum_{i=1}^{N_\phi} W_i\phi_i(\mathbf{x})$, which minimizes the Loss Function

$$\mathcal{L} = \frac{1}{2}\sum_{\mu=1}^{P}\left(y^\mu - \sum_{i=1}^{N_\phi} W_i\phi_i(\mathbf{x}^\mu)\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{N_\phi} W_i^2\,. \tag{4.16}$$

It can be shown that in this setting, but this is still true for classification problems and for other types of Loss Functions, the function f satisfies the Representer Theorem [51], which states that it can be expanded as a linear combination of kernels,

$$f(\mathbf{x}) = \sum_{\mu=1}^{P} \alpha_\mu K(\mathbf{x}^\mu, \mathbf{x})\,, \tag{4.17}$$

where α_μ are coefficients that can be easily computed. When the Loss Function is the one in (4.16), these coefficients can be found as $\alpha = (\mathbf{K} + \lambda\mathbb{1})^{-1}\,\mathbf{y}$, where $\mathbf{K}$ is the $P \times P$ kernel matrix of elements $K_{\mu\nu} = K(\mathbf{x}^\mu, \mathbf{x}^\nu)$. Therefore, kernel methods provide a clever way for inferring non-linear functions with the same computational cost of linear regression, once the kernel matrix is computed. A naive example of how kernel methods can be applied is represented in Fig. 4.6.

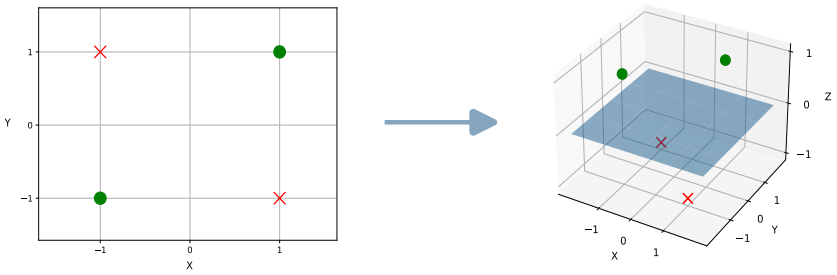

Figure 4.6: Example of kernel method. This sketch represents an example of how kernel methods can be applied to a problem that linear regression can not solve. Suppose to have a dataset where $\mathbf{x} \in \mathbb{R}^2$ and $y = XOR(\mathbf{x})$, i.e. $y = 1$ if the component of $\mathbf{x}$ share the sign, $y = -1$ otherwise. As shown on the left side, this dataset is not linearly separable (the green dots represent the $y = 1$ data, and the red crosses the $y = -1$ one). It is easy to see that the mapping $\phi(\mathbf{x}) = (x_1, x_2, x_1 x_2) : \mathbb{R}^2 \to \mathbb{R}^3$, makes the embedded dataset separable by a plane, the blue one on the right side of the figure. Thus, one has to compute only the kernel $\mathbf{K}$ and then perform a simple linear regression. Note that this is actually a classification problem, but in general, we can treat classification as regression on class labels.

In the realm of kernel methods, there is a particular case, which we will see is deeply connected with overparameterized neural networks, called **Gaussian Process** (GP). Let us start with the definition:

Definition 1. *A **Gaussian Process** (GP) is a collection of random variables, any finite number of which have joint Gaussian distributions.*

Any GP can be thought of as a distribution over functions, fully specified by the mean function μ and covariance function $\mathbf{\Sigma}$. Then, given a regression problem, the goal is to find the appropriate (μ, Σ), from which a good predictive function can be drawn. The peculiarity of GP regression is that it provides the posterior predictive distribution of the dataset, marginalizing the distribution over the parameter vector $\mathbf{W}$. In this sense, it provides a way for considering every possible model with every possible parameter, without concentrating on finding a particular suitable one. Formally, the goal is to find the distribution

$$P(y|\mathcal{D}, \mathbf{x}) = \int_{\mathbf{W}} P(y|\mathbf{x}, \mathbf{W}) P(\mathbf{W}|\mathcal{D}) d\mathbf{W} \propto \int_{\mathbf{W}} P(y|\mathbf{x}, \mathbf{W}) P(\mathcal{D}|\mathbf{W}) P(\mathbf{W}) d\mathbf{W}\,, \quad (4.18)$$

where the Bayes rule has been used on the conditional probability of $\mathbf{W}$ over the dataset $\mathcal{D}$.

The starting assumption is that the distribution $P(y|\mathbf{x}, \mathbf{W})$ is a Gaussian distribution with mean value $\mathbf{W} \cdot \phi(\mathbf{x})$ and covariance matrix proportional to the identity matrix. $P(\mathcal{D}|\mathbf{W})$ is simply the product of P distribution of the element of the dataset, $P(y^\mu|\mathbf{x}^\mu, \mathbf{W})$, and so it is again a Gaussian distribution. Finally, we can assume the prior distribution over the parameter vector $\mathbf{W}$ to be Gaussian (this assumption will be discussed in detail in the next chapters). Since we deal with the integral over a product of Gaussians, we can solve analytically Eq. (4.18) and we can marginalize the probability distribution over $\mathbf{W}$. This allows one to make predictions without selecting a particular model over the hypothesis space.

In order to solve the regression problem with GP, first the covariance matrix Σ must be computed. This is done through the application of a kernel function, which is the reason why GPs are considered as a type of kernel method. We remember that kernels are functions that quantify the similarities between data inputs. Then, one has to choose a kernel function, compute with the feature map the kernel matrix $\mathbf{K}$, and then consider this matrix as the covariance matrix of the GP. Instead, the mean μ is simply the vector whose components are the average values of the corresponding components of the input data. Once having constructed the GP, we can make predictions over new data $\mathbf{x}^*$, i.e. we can compute the probability distribution of $P(y^*|y^1, \cdots y^P, \mathbf{x}^1, \cdots \mathbf{x}^P, \mathbf{x}^*)$. It is easy to check, applying the rule of conditioning Gaussian distributions, that the prediction is drawn from the distribution

$$P(y^*|y^1, \cdots y^P, \mathbf{x}^1, \cdots \mathbf{x}^P, \mathbf{x}^*) = \mathcal{N}\left(\mathbf{K}^{*\top} (\mathbf{K} + \lambda \mathbb{1})^{-1} \mathbf{y}, K^{**} - \mathbf{K}^{*\top} (\mathbf{K} + \lambda \mathbb{1})^{-1} \mathbf{K}^*\right) , \tag{4.19}$$

where $K^*_\mu = K(\mathbf{x}^*, \mathbf{x}^\mu)$, $K^{**} = K(\mathbf{x}^*, \mathbf{x}^*)$, λ is a regularization term which takes in account the inherent noise over the training labels, and for simplicity we consider the input data having zero mean. Note that the mean value of the predictive distribution is equal to the solution of the kernel regression problem, Eq. (4.17). Moreover, the power of GPs stands to the fact they also provide the confidence of such a prediction, since its variance can be computed explicitly. This variance depends only on the value of the inputs and not on the labels $\mathbf{y}$.

In the following section, we will see how GPs are connected to neural networks, and how this connection can provide an estimate of the generalization error.

4.4 Bayesian Learning and Neural Network Gaussian Process (NNGP)

In this section, we introduce the concept of Bayesian Neural Network, a theoretical framework that allows to study of the behavior of ANNs (4.3). A Bayesian Neural Network is a type of neural network that incorporates Bayesian probability theory. Instead of having fixed weights, BNNs treat them as probability distributions. First, a prior distribution, before any type of training, is assumed. In the following, we will assume independent zero-mean Gaussian distributions as priors for the weights, and for simplicity, we will take all the biases $\mathbf{b}$ to zero. Indeed, one can map a system with non-zero biases in a zero-bias one increasing by one the dimensions of the input and of the activations at each layer. The original biases are then trivially mapped in the extra weights of the augmented system. We denotes the prior over $\mathbf{W}$ as $P(\mathbf{W}) = \mathcal{N}(0, \sigma^2_{\mathbf{W}})$.

In the Bayes view of NN, predictions for the output value y^* corresponding to a new input value $\mathbf{x}^*$ are made by integrating over the posterior distribution in weight space. Given the training set $\mathcal{D}_P$, this distribution is denoted as $P(\mathbf{W}|\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ and $\mathbf{y}$ denote the collection of inputs and labels. Then, the predictive distribution for the new unseen data is

$$P(y^*|\mathbf{X}, \mathbf{y}, \mathbf{x}^*) = \int \delta(y^* - f_{\mathbf{W}}(\mathbf{x}^*)) P(\mathbf{W}|\mathbf{X}, \mathbf{y}) d\mathbf{W}\,, \tag{4.20}$$

where $f_{\mathbf{W}}(\mathbf{x}^*)$ is the output of the NN having as parameter $\mathbf{W}$, when the input is $\mathbf{x}^*$. It can be shown [52, 53], by means of the Bayes theorem, that this can be viewed as making predictions over functions $z(\mathbf{x})$ rather than weights:

$$P(y^*|\mathbf{X}, \mathbf{y}, \mathbf{x}^*) = \int P(y^*|\mathbf{z}) P(\mathbf{z}|\mathbf{y}) d\mathbf{z}\,. \tag{4.21}$$

This distribution can be evaluated in many ways, for example by means of a Monte Carlo routine. However, there is a special case for which this distribution can be found analytically [52, 53, 54]. In the case of one-hidden layer (1HL) neural networks, if the number of neurons in the hidden layer is much bigger than the number of data, then this distribution is equivalent to the one of a particular GP. The regime just mentioned is

called the **infinite-width** (IW) limit of neural networks:

$$P \text{ fixed}\,, \;\; N_\ell \to \infty. \tag{4.22}$$

Let us suppose for simplicity, a 1HL FCN with binary output. Then, the output of this network is computed as

$$f_\theta(\mathbf{x}) = \frac{1}{\sqrt{N_1}} \sum_{i=1}^{N_1} v_i \sigma\left(\frac{\mathbf{W}_i \cdot \mathbf{x}}{\sqrt{N_0}}\right) = \frac{1}{\sqrt{N_1}} \sum_{i=1}^{N_1} v_i \sigma(h_i)\,, \tag{4.23}$$

where v_i are the weights of the last layer ($\mathbf{v} \equiv \mathbf{W}^{(L)}$) and where we have used the **standard normalization** (parameterization), which allows us to make use of the Central Limit Theorem (CLT), and the definition of pre-activations (4.2). We want to stress out that this is not the only possible normalization for defining a NN function. For example, the authors of [55] normalize with the factor $1/N_\ell$, thanks to which they build a mean-field theory for NNs with two layers. Nevertheless, in the following of the entire thesis, we will focus on the standard normalization.

Since the weights are taken to be i.i.d, then the components of the post-activations $\sigma(h)$ are independent. $f_\theta(\mathbf{x})$ is a sum of i.i.d. terms, and so from the CTL, it follows that in the IW limit f_θ will be Gaussian distributed. Moreover, from the multidimensional CLT, any finite collection of different f_θ will have a joint multivariate Gaussian distribution, which is exactly the definition of a GP (Def. (1)). Thus, f_θ is drawn from a GP with mean $\mu(\mathbf{x}) \equiv \mathbb{E}[f_\theta(\mathbf{x})] = 0$ (the parameter are taken with zero mean) and covariance function

$$K(\mathbf{x}, \mathbf{x}') \equiv \mathbb{E}[f_\theta(\mathbf{x}) f_\theta(\mathbf{x}')] = \sigma_{\mathbf{W}}^2 \mathbb{E}[\sigma(h(\mathbf{x}))\sigma(h(\mathbf{x}'))]\,, \tag{4.24}$$

which is called the Neural Network Gaussian Process (NNGP) kernel. This argument about the correspondence of a one hidden layer NN and a GP can be extended to deeper architectures, by induction. It is sufficient to take the hidden layer widths to be infinite in succession, to guarantee the presence of a GP as input in the layer under consideration, as was demonstrated in [54].

The expression of the prior distribution of the outputs of a wide network in the IW limit can be derived informally using the physics formalism. This is instructive to introduce the

reader to the method used in our work. Drawing parallels between BNNs and Statistical Mechanics, one can think of the weights in a BNN as analogous to particles in a physical system. The probability distributions over these weights can be linked to the Boltzmann distribution. The energy landscape of a neural network, defined by its Loss Function, is reminiscent of the potential energy landscapes in physical systems. Training a neural network can then be seen as finding low-energy configurations, much like how physical systems tend to states of lower energy. Then, if the NN function f_θ is fully defined by the set of its parameter θ, the aim for solving a statistical mechanics theory of NN, is to compute the partition function. Given a dataset $\mathcal{D}_P$, the partition function reads

$$Z = \int \mathcal{D}\theta \, e^{-\beta \mathcal{L}(\theta)} \,, \tag{4.25}$$

where we included the prior distribution of the weights in the differential $\mathcal{D}\theta$ and β is the inverse temperature.

We can show that a 1-hidden layer neural network is equivalent to a GP in the infinite-width limit, by integrating out the weights in Z. Note that, while the following is not a formal proof (which was previously mentioned in a few words), it is more transparent from the point of view of Statistical Mechanics.

First of all, we take the NN function in the form defined in Eq. (4.23). Note that in [52] the normalizing factor $\sqrt{N_\ell}$ is directly inserted in the weights prior, but our choice is equivalent. Indeed, we take as prior distribution for the weights Gaussian distributions with variance λ_0^{-1} and λ_1^{-1}, for $\mathbf{W}$ and $\mathbf{v}$ respectively and we incorporate them into the Loss Function, as regularization terms

$$\mathcal{L} = \underbrace{\frac{\lambda_0}{2\beta}\|\mathbf{W}\|^2 + \frac{\lambda_1}{2\beta}\|\mathbf{v}\|^2}_{\mathcal{L}_{reg}} + \frac{1}{2}\sum_{\mu=1}^{P} \left(y^\mu - f_\theta(\mathbf{x}^\mu)\right)^2 \,, \tag{4.26}$$

where $\|W\|^2$ is computed as the Frobenius norm for matrices. With these definitions, the partition function reads

$$Z = \int \prod_{i=1}^{N_1} dv_i \prod_{i,j=1}^{N_0,N_1} dW_{ij} \, e^{-\frac{\lambda_0}{2}\|\mathbf{W}\|^2 - \frac{\lambda_1}{2}\|\mathbf{v}\|^2 - \frac{\beta}{2}\sum_{\mu=1}^{P}(y^\mu - f_\theta(\mathbf{x}^\mu))^2} \,. \tag{4.27}$$

Following the GP reasoning, we have to integrate out the weights, to gain a prior over functions. For doing that we have to decouple them in f_θ, by defining auxiliary variables, the network output $s^\mu \equiv f_\theta(\mathbf{x}^\mu)$ and the preactivation $h_i^\mu \equiv (N_0)^{-1}\sum_j W_{ij}x_j^\mu$. We insert these new variables into Z as constraints through Dirac's deltas

$$Z = \int \mathcal{D}\mathbf{v}\mathcal{D}\mathbf{W} \prod_{\mu=1}^{P} ds^\mu \prod_{\mu,i=1}^{P,N_1} dh_i^\mu \, e^{-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^\mu - s^\mu)^2} \\ \times \prod_{\mu=1}^{P} \delta\left(s^\mu - \frac{1}{\sqrt{N_1}}\sum_{i=1}^{N_1} v_i \sigma(h_i^\mu)\right) \prod_{\mu,i=1}^{P,N_1} \delta\left(h_i^\mu - \frac{1}{\sqrt{N_0}}\sum_{j=1}^{N_0} W_{ij}x_j^\mu\right) , \tag{4.28}$$

where we have incorporated the weights prior into the differential form $\mathcal{D}\mathbf{v}\mathcal{D}\mathbf{W}$. In this way, the weights of different layers are decoupled, and we have to compute the average value of the deltas over the prior distributions

$$\left\langle \prod_{\mu=1}^{P} \delta\left(s^\mu - \frac{1}{\sqrt{N_1}}\sum_{i=1}^{N_1} v_i \sigma(h_i^\mu)\right)\right\rangle_{\mathbf{v}} , \left\langle \prod_{\mu,i=1}^{P,N_1} \delta\left(h_i^\mu - \frac{1}{\sqrt{N_0}}\sum_{j=1}^{N_0} W_{ij}x_j^\mu\right)\right\rangle_{\mathbf{W}} , \tag{4.29}$$

with $\langle O(\mathbf{v})\rangle_{\mathbf{v}} \equiv \int \mathcal{D}\mathbf{v} O(\mathbf{v})$. These two integrals are easily solvable once we introduce the Fourier representation of the deltas $\delta(x) \propto \int d\bar{x} e^{ix\bar{x}}$, and give the contribution

$$\langle\cdot\rangle_{\mathbf{W}} \langle\cdot\rangle_{\mathbf{v}} = \int \prod_{\mu=1}^{P} d\bar{s}^\mu \, e^{i\sum_\mu s^\mu \bar{s}^\mu} \prod_{i=1}^{N_1} \left[\mathcal{N}_{\mathbf{h}_i}(0,\mathbf{C}) e^{-\frac{1}{2\lambda_1 N_1}\left(\sum_\mu \bar{s}^\mu \sigma(h_i^\mu)\right)^2}\right] , \tag{4.30}$$

where we have already integrate out the Fourier variables $\bar{\mathbf{h}}$ and we remind that $\mathcal{N}_{\mathbf{x}}(0,\mathbf{C}) \equiv (\det(2\pi\mathbf{C}))^{-1/2}\exp\left(-\frac{1}{2}\sum_{\mu\nu} C^{-1}_{\mu\nu}x^\mu x^\nu\right)$. Here, $\mathbf{C}$ is the covariance matrix of the input data and it is defined as $C_{\mu\nu} \equiv (\lambda_0 N_0)^{-1}\sum_{i=1}^{N_0} x_i^\mu x_i^\nu$. We can now focus on the integrals over the preactivations $\mathbf{h}^\mu$, noting that we can treat them as N_1 equal integrals of the quantity in square brackets in Eq. (4.30). At this point, the partition function reads

$$Z = \int \prod_{\mu=1}^{P} ds^\mu d\bar{s}^\mu \, e^{-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^\mu - s^\mu)^2 + i\sum_\mu s^\mu \bar{s}^\mu} \left[\int d^P h \, \mathcal{N}_{\mathbf{h}}(0,\mathbf{C}) e^{-\frac{1}{2\lambda_1 N_1}\left(\sum_\mu \bar{s}^\mu \sigma(h_i^\mu)\right)^2}\right]^{N_1} . \tag{4.31}$$

From this step, we can make use of the infinite-width limit (4.22). Since P, the size of the training set is kept finite and N_1 is taken large, we can expand the exponential in the

square bracket quantity, obtaining up to leading order

$$
\begin{aligned}
[\cdots]^{N_1} &\approx \left[\int d^P h\, \mathcal{N}_{\mathbf{h}}(0,\mathbf{C})\left(1-\frac{1}{2N_1\lambda_1}\sum_{\mu\nu}\bar{s}^\mu\sigma(h^\mu)\sigma(h^\nu)\bar{s}^\nu\right)\right]^{N_1} \\
&= \left[1-\frac{1}{2N_1\lambda_1}\sum_{\mu\nu}\bar{s}^\mu\bar{s}^\nu\int d^P h\, \mathcal{N}_{\mathbf{h}}(0,\mathbf{C})\sigma(h^\mu)\sigma(h^\nu)\right]^{N_1} \\
&\equiv \left[1-\frac{1}{2N_1}\sum_{\mu\nu}\bar{s}^\mu K(\mathbf{C})_{\mu\nu}\bar{s}^\nu\right]^{N_1} \underset{\substack{P \text{ fixed}\\ N_1\to\infty}}{\longrightarrow} e^{-\frac{1}{2}\sum_{\mu\nu}\bar{s}^\mu K(\mathbf{C})_{\mu\nu}\bar{s}^\nu}\,,
\end{aligned}
\tag{4.32}
$$

where we have defined with $K(\mathbf{C})_{\mu\nu}$ the integral on $\mathbf{h}$, which can be regarded as a correlation function of a Gaussian field. It is easy to check that

$$
K(\mathbf{C})_{\mu\nu} = \lambda_1^{-1}\int dt_1 dt_2\, \mathcal{N}_{\mathbf{t}}(0,\tilde{\mathbf{C}})\sigma(t_1)\sigma(t_2)\,, \text{ where } \tilde{\mathbf{C}}_{\mu\nu} = \begin{pmatrix} C_{\mu\mu} & C_{\mu\nu} \\ C_{\mu\nu} & C_{\nu\nu}\end{pmatrix}\,. \tag{4.33}
$$

Inserting this result in the partition function (4.31), we are left with a last Gaussian integral on the variables $\bar{\mathbf{s}}$, and the final form of Z reads

$$
Z = \int\prod_{\mu=1}^{P} ds^\mu e^{-\frac{\beta}{2}\sum_\mu (y^\mu - s^\mu)^2} P_{\text{prior}}(\mathbf{s})\,, \text{where } P_{\text{prior}}(\mathbf{s}) \equiv \left(\det\left(2\pi\mathbf{K}\right)\right)^{-\frac{1}{2}} e^{-\frac{1}{2}\mathbf{s}^\top\mathbf{K}\mathbf{s}}\,. \tag{4.34}
$$

This is exactly the form of a Gaussian Process, where we have a prior distribution over function which is Gaussian and depends on the kernel $\mathbf{K}$, which is equal to the one in (4.24), the NNGP kernel. As was mentioned earlier, this argument can be extended to deeper networks and we can write the following recurrence relation for $K^{(\ell)}$, the NNGP kernel of the corresponding GP process of the ℓ-th layer:

$$
\begin{cases}
K^{(\ell)}_{\mu\nu} = \lambda_\ell^{-1}\int dt_1 dt_2\, \mathcal{N}_{\mathbf{t}}\left(0,\tilde{\mathbf{K}}^{(\ell-1)}\right)\sigma(t_1)\sigma(t_2)\,, \text{where } \tilde{\mathbf{K}}^{(\ell-1)} \equiv \begin{pmatrix} K^{(\ell-1)}_{\mu\mu} & K^{(\ell-1)}_{\mu\nu} \\ K^{(\ell-1)}_{\mu\nu} & K^{(\ell-1)}_{\nu\nu}\end{pmatrix} \\
K^{(0)}_{\mu\nu} = \frac{\mathbf{x}^\mu\cdot\mathbf{x}^\nu}{\lambda_0 N_0} \equiv C_{\mu\nu}
\end{cases}
\tag{4.35}
$$

The Statistical Mechanics framework we have introduced, from one point gives a new way for proving the convergence of an NN to a GP in the infinite-width limit, and on the other hand, it is a useful tool for computing expectation values. For example, we can

compute the expected generalization error, made on a new unseen data pair $(\mathbf{x}^*, y^*)$, as the expected value of $(y^* - f_\theta(\mathbf{x}^*))^2$, in the zero-temperature limit, which correspond to put the system in the minimum of the energy landscape, i.e. selecting the weights that minimize the training Loss Function:

$$\begin{aligned}\langle \epsilon_g(*) \rangle &= \left\langle (y^* - f_\theta(\mathbf{x}^*)^2 \right\rangle \\ &\underset{\beta \to \infty}{\to} \underbrace{\left(y^* - \sum_{\mu\nu} K_\mu(\mathbf{x}^*) K^{-1}_{\mu\nu} y_\nu \right)^2}_{\text{Bias}^2} + \underbrace{\left(K_*(\mathbf{x}^*) - \sum_{\mu\nu} K_\mu(\mathbf{x}^*) K^{-1}_{\mu\nu} K_\nu(\mathbf{x}^*) \right)}_{\text{Variance}}, \quad (4.36)\end{aligned}$$

where $K_\mu(\mathbf{x}^*) \equiv K(\mathbf{x}_\mu, \mathbf{x}^*)$ and $K_*(\mathbf{x}^*) \equiv K(\mathbf{x}^*, \mathbf{x}^*)$. By recalling the solution of the GP (4.19), taken with no label noise $\lambda = 0$, it is evident that this equation corresponds to the same bias-variance decomposition of the predictor statistics of a GP having as kernel $\mathbf{K}$.

We conclude this section by mentioning the fact that the authors in [56] have proven that also a CNN with infinitely many channels (having number P of pattern fixed) is indeed a Gaussian Process, whose NNGP kernel reads

$$K^{(CNN)}_{\mu\nu} \equiv \frac{1}{\lambda_1 \lfloor N_0/S \rfloor} \sum_i K^{ii}_{\mu\nu}\,, \quad (4.37)$$

where $K^{ii}_{\mu\nu}$ is defined in a way analogously of the one in (4.33), with the introduction of an extended covariance matrix, which incorporates also the spatial correlations. In the next chapter, we will define and study it in detail (see Eq. (6.11) for anticipations).

4.5 Gradient Descent and Neural Tangent Kernel

We conclude this chapter by briefly explaining how one can study the training dynamics of a wide NN. Indeed, the authors of [57], found that not only NNs at initialization are equivalent to GP, but also their evolution during training can be described by a kernel, the Neural Tangent Kernel (NTK). In the following, we will show with a heuristic argument that this implies that in the infinite-width limit, the weights of the network move slightly compared to initialization.

In the first section, we have seen that one way for finding the best values of the parameters, collected in the vector θ, is to follow the gradient of the Loss Function, and this is done

operatively with the backpropagation algorithm. Indeed, the dynamics of the weights is described by the formula $\dot{\theta} = -\nabla_\theta \mathcal{L}$. Then, let us write this expression explicitly for the weights of a 1HL NN, whose function was described in Eq. (4.23), and in the case of MSE as Loss Function:

$$\begin{cases} \dot{v}_i = \frac{1}{\sqrt{N_1}} \sum_{\mu=1}^{P} \left(y^\mu - f_\theta(\mathbf{x}^\mu)\right) \sigma\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\mu}{\sqrt{N_0}}\right) \\ \dot{W}_{ij} = \frac{1}{\sqrt{N_1}} \sum_{\mu=1}^{P} (y^\mu - f_\theta(\mathbf{x}^\mu)) v_i \sigma'\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\mu}{\sqrt{N_0}}\right) \frac{x_j^\mu}{\sqrt{N_0}} \end{cases} , \tag{4.38}$$

where $\sigma'(\cdot)$ is the derivative of the activation function σ. Note that, both these derivatives are the sum of P random variables, and this implies that they are of order $O(\frac{\sqrt{P}}{\sqrt{N_1}})$. This means that, in the infinite-width limit, the weights slowly evolve from the initialization state. For this reason, let us see how the outputs of the network evolve, the already defined variables $s^\mu \equiv f_\theta(\mathbf{x}^\mu)$:

$$\dot{s^\mu} = \dot{f}_\theta(\mathbf{x}^\mu) = \sum_{i=1}^{N_1} \dot{v}_i \frac{\partial f_\theta(\mathbf{x}^\mu)}{\partial v_i} + \sum_{i,j=1}^{N_1,N_0} \dot{W}_{ij} \frac{\partial f_\theta(\mathbf{x}^\mu)}{\partial W_{ij}} = \sum_{\nu=1}^{P} (y^\nu - s^\nu) K^{\mathrm{NTK}}_{\mu\nu}(\theta_t) \,, \tag{4.39}$$

where θ_t denotes the value of the parameters at time t and where we have introduced the so-called Neural Tangent Kernel

$$\begin{aligned} K^{\mathrm{NTK}}_{\mu\nu}(\theta_t) =& \frac{1}{N_1} \sum_{i=1}^{N_1} \sigma\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\mu}{\sqrt{N_0}}\right) \sigma\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\nu}{\sqrt{N_0}}\right) + \\ &+ \frac{1}{N_1} \sum_{i=1}^{N_1} \frac{\mathbf{x}^\mu \cdot \mathbf{x}^\nu}{N_0} v_i^2 \sigma'\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\mu}{\sqrt{N_0}}\right) \sigma'\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\nu}{\sqrt{N_0}}\right) , \end{aligned} \tag{4.40}$$

where it is implied that the weights are taken at time t. The form of Eq. (4.39) highlights that the gradient descent of a 1HL NN is equivalent to a kernel gradient descent with respect to $K^{\mathrm{NTK}}_{\mu\nu}(\theta_t)$. By repeating the previous reasoning, it is clear that the kernel $K^{\mathrm{NTK}}_{\mu\nu}(\theta_t)$ is of order $O(1)$, and that implies the order of $\dot{s}^\mu$ being $O(\sqrt{P})$. This means that, in the infinite-width limit, on the same evolving time scale of s^μ the weights stay basically still and do not evolve from initialization. Then, when studying the dynamics of the outputs, we can take the NTK at initialization $\dot{s}^\mu \approx \sum_{\nu=1}^{P} (y^\nu - s^\nu) K^{\mathrm{NTK}}_{\mu\nu}(\theta_0)$ and we

can write

$$\lim_{N_1\to\infty} K^{\mathrm{NTK}}_{\mu\nu}(\theta_0) = \left\langle \sigma\left(\frac{\mathbf{W}\cdot\mathbf{x}^\mu}{\sqrt{N_0}}\right)\sigma\left(\frac{\mathbf{W}\cdot\mathbf{x}^\nu}{\sqrt{N_0}}\right)\right\rangle_{\theta_0} + \\ + \frac{\mathbf{x}^\mu\cdot\mathbf{x}^\nu}{N_0}\left\langle \mathbf{v}^2\sigma'\left(\frac{\mathbf{W}\cdot\mathbf{x}^\mu}{\sqrt{N_0}}\right)\sigma'\left(\frac{\mathbf{W}\cdot\mathbf{x}^\nu}{\sqrt{N_0}}\right)\right\rangle_{\theta_0}, \tag{4.41}$$

where the averages are taken respect the probability distribution $P(\mathbf{v}, \mathbf{W})$ of the weights at initialization. Note that, if we take as the prior distribution of the weights a Gaussian, the first expected value is the definition of the NNGP kernel.

The fact that in the infinite-width limit, the NTK converges to an explicit limiting kernel and stays constant during training, makes it possible to study the training of ANNs in function space instead of parameter space. Moreover, this is a clear indication of the fact that NNs operating in the infinite-width limit do not evolve significantly from the initialization. In the following chapter, we will argue that this fact suggests that in this regime there can not be any type of **feature learning**.

5 Beyond the Infinite-Width limit

Let us briefly recap what we have said so far. The statistical mechanics framework is naturally applicable to the so-called Bayesian Neural Networks (BNN), a type of NN where the weights are not fixed by the training phase, but they are considered as degrees of freedom of a statistical system. Indeed, they evolve and eventually reach the stationary state, following a Hamiltonian, which is described in terms of the Loss Function.

The statistical mechanics of NNs it not yet fully understood nor analytically tractable, due to the presence of non-linearities. However, there is a particular setting, the so-called infinite-width (IW) limit, for which a Gaussian equivalence can be found. Indeed, in the case of a one-hidden layer neural network (1HL NN), when the number of the neurons in the hidden layer N_1 is sent to infinite while keeping the number of training pairs P finite, the predictor statistics of the NN is equivalent to the one of a Gaussian Process (GP), having as kernel the so-called Neural Network Gaussian Process (NNGP) kernel, described in Eq. (4.33). This reasoning can be extended to an NN with L layer by induction, and we have already mentioned the recurrence relation of the NNGP kernel in Eq. (4.35). One can exploit this Gaussian equivalence to compute expectation values of quantities, like for example the generalization error (4.36). Since in the IW limit, the number of parameters, that scales as $O(L \times N_\ell^2)$, is much bigger than the number of patterns P, the NN is found in the overparameterized regime (see Fig. 4.5), where we have seen the generalization error ϵ_g behaves unexpectedly.

Recently, it has been studied also the dynamics under gradient descent of NNs in the IW limit (see Sec. 4.5). It was found [57] that the evolution of an NN during the training phase is described by a kernel, the Neural Tangent Kernel. In the IW limit, this kernel converges to a limiting kernel and it stays constant. This implies that in IW architecture, there can not be any type of **feature learning**, since the corresponding kernel does not change from the one at initialization.

5.1 Finite-width networks: empirical evidence and theoretical approaches

In the search for insights about feature learning mechanisms, the authors of [54, 58, 59] conducted a large-scale empirical study comparing finite-width FCNs, CNNs with a finite number of channels and their infinite-width limit kernel counterparts, exploring a plenitude of possible settings to improve generalization. For better understanding, we report directly one plot of their work in Fig. 5.1.

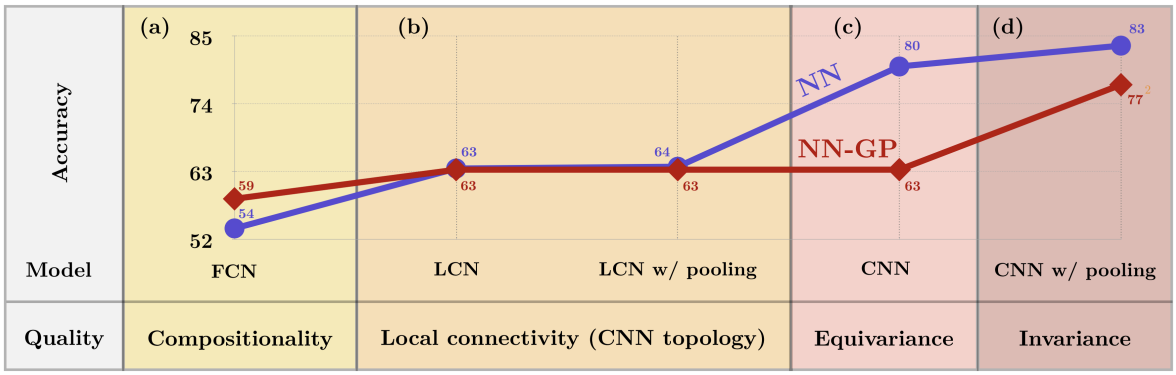

Figure 5.1: Performance on the test set for different architectures, compared with the corresponding NNGP kernel method. This plot is taken directly from [56]. They performed an intensive numerical simulation on a classification task (in particular using the CIFAR10 [60] dataset), and for this reason, the quantity that quantifies the performance is the test accuracy. The accuracy is simply the percentage of correct classifications of the elements in the test set. The blue line with the superscript "NN" refers to the accuracy found in training neural networks through stochastic gradient descent, while the red one ("NN-GP") is the accuracy of the corresponding infinite-width model (i.e. the corresponding GP). It is clear that, in FCN the infinite-width network outperforms the finite one, while it happens the contrary for CNNs. Meanwhile, for LCNs (which are CNNs without weight sharing) it is not clear which model performs better.

Their analysis prompts a few striking empirical observations:

(i) Infinite-width kernels systematically outperform their finite-width counterpart in the case of fully-connected deep neural networks;

(ii) CNNs with a finite number of channels often outperform their corresponding infinite-width kernel performance.

Moreover, in one of the seminal papers dealing with the infinite-width limit [54], they observe that increasing the hidden layers size leads to optimal test accuracy on deep

FCN architectures trained on MNIST and CIFAR10, two of the benchmark datasets for computer vision learning problems. Such evidence is further corroborated by a difference in the behavior of the test accuracy as a function of the size of the hidden layers: whereas for FCNs the accuracy monotonically approaches the kernel performance from below, in CNNs one often observes a non-monotonic behavior with a region, at finite-width, where the network is capable of outperforming the CNN GP. Similar results were found by the same authors in [58]. We report an example of their experiment in Fig. 5.2.

Several analogous observations have been reported in the literature: very recently, the authors of [61] pointed out that infinite-width deep FCNs eventually outperform their finite-width counterparts as the size of the training set grows. The interplay between the lazy training regime and the mean-field limit [62] has been the subject of a thorough investigation in [63, 64].

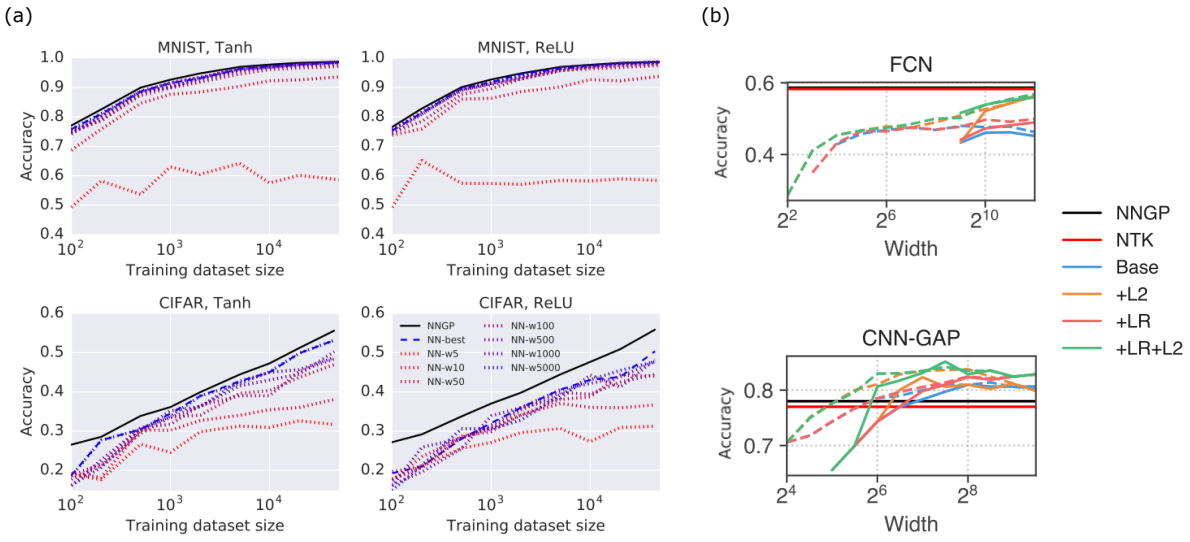

Figure 5.2: Performance behavior with increasing hidden layer size for different architectures. These two plots are taken from [54, 59]. In (a) the accuracy on the test set is shown against the size P of the training set, for FCNs with various sizes of the hidden layer (from 5 to 5000). The solid black line represents the accuracy of the corresponding NNGP. It is evident that this one outperforms any other finite-width architectures tested. Similarly, in (b) they show the performance behavior when increasing the width N_1, for various settings (learning rate LR, regularization L2), and both FCNs and CNNs. The acronym "GAP" means that a global average pooling is performed after the convolutional layer (a standard practice, which is an average of pixels in the feature maps). The performance of FCNs reaches the NNGP one from below, while this is not true for CNNs, where the best performance is found for finite-width architectures.

These observations suggest that CNNs leverage a better feature-learning mechanism at

finite width than FCNs and LCNs, and prompt at least two conceptual questions:

(i) Why is it ultimately convenient to employ large-width architectures when only fully-connected layers are available?

(ii) Why is this not the case when convolutional layers are employed? And how does a CNN operatively exploit the finite-width regime for efficient feature learning?

To answer these questions, it is needed a more enlightened theory that can incorporate also the finite-width limit of NNs, since this kind of network is mostly used in practice and can exhibit feature learning. For this reason, the effort of the community working in deep learning theory is now devoted to understanding how to move beyond the elegant and mathematically rigorous framework of infinite-width networks. Research lines in this direction include:

- standard perturbation theory in $1/N_\ell$ at finite size of the training set P [65, 66];
- proportional limits where both the size of the layers N_ℓ and the depth L are taken to infinity keeping their ratio fixed [67];
- thermodynamic descriptions to capture finite-width effects based on separation of scales [68, 69];
- analytically solvable toy models such as deep linear or globally-gated deep linear networks [70, 71, 72].

Moreover, the authors of Refs. [73, 74] analytically investigated the advantages of employing convolutional neural tangent kernels in the infinite-width limit and understood why infinite-width FCNs perform worse in the mean field regime than in the lazy-training one [75]. Recently, the authors of [76] built a general theory from which one can derive the infinite-width limit of a NN, starting from any parameterization, and they started a strategy for analytically computing these limits, which they called *Tensor program* [77, 78, 79, 80]. We have already seen that our study focused on the so-called standard (or NTK) parameterization (Eq. (4.23)), where the pre-activations are rescaled with $(N_{\ell-1})^{-1/2}$ and the post-activation by $(N_\ell)^{-1/2}$, and in this setting the NTK does not evolve in the IW limit, implying no feature learning. On the contrary, the authors of the Tensor program claim that there is one particular parameterization that allows the

network to deviate from the kernel regime and gain feature learning in the IW limit, even for an FCN architecture. Therefore, as we will see, their weight normalization is not yet directly applicable to our methods, and determining how to incorporate the regime they investigate will be a topic for future investigations.

Our work, carried out together with some of the authors of [81], fits into this vast community but it focuses on another point of view. Indeed, our approach relies on studying the statistical mechanics of NNs in the thermodynamics limit where both P and N_ℓ are sent to infinity, but their ratio is kept constant:

$$P, N_\ell \to \infty \,, \text{ such that } \alpha_\ell \equiv \frac{P}{N_\ell} = const \,. \tag{5.1}$$

We will refer to this regime as the **proportional limit** or **finite-width limit**. Note that in this setting, the NN is still in the overparameterized regime, where the training set can be exactly interpolated, since the number of parameter $L \times N_\ell^2 \gg P \quad \forall \ell$.

Let us set again the problem. Our theory applies to a supervised regression problem. The numerical parameters will be:

- P is the size of the training set, i.e. the number of pattern-label pairs;
- N_ℓ will be the number of neurons in the ℓ-th layer of the neural network architecture, $\ell = 1, \cdots, L$;
- N_0 will be the size of the input.

The training set is defined as $\mathcal{D}_P = \{\mathbf{x}^\mu, y^\mu\}$, where each $\mathbf{x}^\mu \in \mathbb{R}^{N_0}$ and we will consider one-dimensional output label, $y^\mu \in \mathbb{R}$. From now on, for the sake of clarity, we will denote vectors and matrices with bold characters, using lowercase letters for the former, $\mathbf{v}$, and capital letters for the latter, $\mathbf{M}$. In general, we can define a neural network as a function $f_\theta : \mathbb{R}^{N_0} \to \mathbb{R}$, which depends on a set of trainable parameters θ, also called *weights*. These parameters can evolve from the initialization to complete a particular task. In the following, we will make use of a standard notation for θ. The weights of the last layer L will be denote by $\mathbf{v} \in \mathbb{R}^{N_L}$ and the weights of the ℓ-th hidden layer by $\mathbf{W}^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$.

We consider a supervised regression problem with a quadratic loss function

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{\mu=1}^{P}\left[y^{\mu} - f_{\theta}(\mathbf{x}^{\mu})\right]^2 + \mathcal{L}_{reg}\,, \tag{5.2}$$

where we have made explicit the dependence on the network parameters and $\mathcal{L}_{reg}$ refers to a regularization term, which is inserted as standard practice when training a neural network. In our case, we will focus on the so-called L^2 regularization, which is defined as

$$\mathcal{L}_{reg} = \frac{1}{2\beta}\left(\sum_{\ell=1}^{L}\lambda_{\ell}\|W^{(\ell)}\|^2 \; + \lambda_L\|v\|^2\right)\,, \tag{5.3}$$

where $\|v\|$ is the norm of the weight vector of the last layer, $\|W^{(\ell)}\|$ is the Frobenius norm of the hidden layer weight matrices, λ_ℓ are coefficients which can be regarded as Gaussian priors over the weights of each layer, and β is the inverse temperature. Later, we will discuss in detail the meaning and the properties of the Gaussian priors λ_ℓ. The L^2 regularization will favor weights with a small L_2 norm. In order to follow a statistical mechanics approach to our Deep Learning problem, the loss function is regarded as an energy function, i.e. as the Hamiltonian of a system, whose degrees of freedom are the weights. In this way, having inserted the factor $1/\beta$, the $\mathcal{L}_{reg}$ acts as an entropic term.

As standard practice in Statistical Mechanics, we define the partition function

$$Z = \int \mathcal{D}\theta\, e^{-\beta\mathcal{L}(\theta)}\,, \tag{5.4}$$

where $\int \mathcal{D}\theta$ denotes the integration over the whole set of weights θ. In this manner, for $\beta \to \infty$, the first term of $\mathcal{L}$, the mean-squared error, will enforce minimization of the training error, while the entropic term will increase the chance of having configurations with a small norm.

The partition function allows us to compute several statistical quantities. The **generalization error** is one of them, whose definition we recall being

$$\left\langle \epsilon_g(\mathbf{x}^0, y^0)\right\rangle = \int \mathcal{D}\theta\left[y^0 - f_{\theta}(\mathbf{x}^0)\right]^2 e^{-\beta\mathcal{L}(\theta)}\,, \tag{5.5}$$

where the pair $(\mathbf{x}^0, y^0)$ belongs to the test set (the set of data not used during the training

phase), and the average is performed over the Gibbs distribution of θ. Another quantity, which we studied in detail in our work and we will explain in the next chapter is the **similarity matrix**, i.e. the average correlation matrix of internal representations

$$O^{(l)}_{\mu\nu} = \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} \phi_i^{(\ell)}(\mathbf{x}^\mu)\phi_i^{(\ell)}(\mathbf{x}^\nu)\,, \tag{5.6}$$

where $\phi_i^{(\ell)}(\mathbf{x}^\mu)$ is the i-th component of the internal representation on the ℓ-th layer of the μ-th pattern of the training set, i.e. the value of the i-th neuron of the ℓ-th layer when the network is fed with $\mathbf{x}^\mu$ as input. The importance of these matrices relies on the fact they give information about how features of the input stimulate the neurons, and how the input is represented and transformed through the layers of the network.

In order to calculate these quantities, one has to solve the theory, i.e. to compute the partition function. This task depends on the form of the neural network and the architecture chosen, and so on f_θ. Recently, Li and Sompolinksy [70] found a strategy for doing so in the presence of a Fully-connected neural network with linear activation function, which they called Back-Propagating Kernel Renormalization. With their method, they were able to analytically evaluate properties of a finite-width deep linear networks, trained on a generic fixed training set. Recently, some of our collaborators found a way to solve the partition function in the presence of a non-linear activation function. In their work [81], they focused on solving the theory for a Fully-Connected Network (FCN), finding an interesting behavior of the generalization error, and its dependency from the Gaussian priors. In this perspective, our work is directed to extend the same reasoning for solving a more interesting architecture, a Convolutional Neural Network (CNN).

In the following, we will give a brief review of what they found about FCNs, recalling the main mathematical aspects, which will serve as a playground for our calculations.

5.2 Bayesian effective action for FCNs in the proportional limit

The starting point is to define the neural network function f_θ. In the following, we will drop the θ subscript and we will replace it with the acronym of the corresponding architecture.

Then, the function for a shallow (which means one hidden layer) Fully-Connected (FCN) architecture reads

$$f_{FCN}(\mathbf{x}^\mu) = \frac{1}{\sqrt{N_1}} \sum_{i=1}^{N_1} v_i \sigma \left(\sum_{j=1}^{N_0} \frac{W_{ij} x_j^\mu}{\sqrt{N_0}} \right) . \tag{5.7}$$

The choice of inserting the square root of the layer length as normalization will allow us to apply an informally justified Gaussian equivalence through the Breuer-Major theorem [82].

Having defined the neural network function, the partition function (5.4) becomes

$$Z_{FCN} = \int \prod_{i=1}^{N_1} dv_i \prod_{i,j=1}^{N_1,N_0} dW_{ij}\, e^{-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^\mu - f_{FCN}(\mathbf{x}^\mu))^2 - \frac{\lambda_0}{2}\|w\|^2 - \frac{\lambda_1}{2}\|v\|^2} , \tag{5.8}$$

where we have added the FCN subscript to remark the fact it works for Fully-Connected Networks. The strategy is to integrate out the weights of the network **forwardly**, from the input weights to the output, as we have already done in Section 4.4 when we proved the NN-GP equivalence. For doing so, we introduce two sets of new variables, the *network outputs* s^μ and the *pre-activations* $\mathbf{h}^\mu$, defined as

$$h_i^\mu \equiv \sum_{j=1}^{N_0} \frac{W_{ij} x_j^\mu}{\sqrt{N_0}} , \tag{5.9}$$

$$s^\mu \equiv \sum_{i=1}^{N_1} \frac{v_i \sigma\left(h_i^\mu\right)}{\sqrt{N_1}} . \tag{5.10}$$

These new sets of variables allow us to decouple the weights of different layers and then integrate them out. Thus, we insert two families of Dirac's delta represented in Fourier transform, exploiting the identities

$$1 = \int \prod_{\mu=1}^{P} ds^\mu\, \delta\left(s^\mu - \sum_{i=1}^{N_1} \frac{v_i \sigma\left(h_i^\mu\right)}{\sqrt{N_1}} \right) \equiv \int \prod_{\mu=1}^{P} \frac{ds^\mu d\bar{s}^\mu}{2\pi}\, e^{i\sum_\mu s^\mu \bar{s}^\mu - \frac{i}{\sqrt{N_1}} \sum_\mu \bar{s}^\mu \sum_{i=1}^{N_1} v_i \sigma\left(h_i^\mu\right)} \tag{5.11}$$

$$1 = \int \prod_{i,\mu=1}^{N_1,P} dh_i^\mu\, \delta\left(h_i^\mu - \sum_{j=1}^{N_0} \frac{W_{ij} x_j^\mu}{\sqrt{N_0}} \right) \equiv \int \prod_{i,\mu=1}^{N_1,P} \frac{dh_i^\mu d\bar{h}_i^\mu}{2\pi}\, e^{i\sum_{i,\mu} h_i^\mu \bar{h}_i^\mu - \frac{i}{\sqrt{N_0}} \sum_{i,\mu} \bar{h}_i^\mu \sum_j W_{ij} x_j^\mu} , \tag{5.12}$$

$$\tag{5.13}$$

where we have inserted the correct coefficients 2π, which will be omitted in the following since they contribute to Z with an irrelevant constant term. Moreover, for the sake of notation and readability we will often drop the extremes of summations, referring with Greek letters μ, ν to the index over the training set and with Latin one i, j over the neurons.

In this way, the partition function can be written as

$$Z_{FCN} = \int \prod_{i,\mu} dh_i^\mu d\bar{h}_i^\mu \prod_\mu ds^\mu d\bar{s}^\mu \, e^{-\frac{\beta}{2}\sum_\mu (y^\mu - s^\mu)^2 + i\sum_\mu s^\mu \bar{s}^\mu + i\sum_{\mu i} h_i^\mu \bar{h}_i^\mu} \times \left\langle e^{-i\sum_\mu \bar{s}^\mu \left(\frac{1}{\sqrt{N_1}}\sum_i v_i \sigma(h_i^\mu)\right)} \right\rangle_{\mathbf{v}} \times \left\langle e^{-i\sum_{i,\mu} \bar{h}_i^\mu \left(\frac{1}{\sqrt{N_0}}\sum_j W_{ij} x_j^\mu\right)} \right\rangle_{\mathbf{W}} , \tag{5.14}$$

where the operation $\langle \cdot \rangle_{\mathbf{v}}$ means the mean value over the probability distribution of $\mathbf{v}$, i.e. $\langle O \rangle_{\mathbf{v}} \equiv \int dP(\mathbf{v}) O(\mathbf{v})$. These two mean values are simply Gaussian integrals and by solving them we remain with the following partition function

$$Z_{FCN} = \int \prod_\mu ds^\mu d\bar{s}^\mu e^{-\frac{\beta}{2}\sum_\mu (y^\mu - s^\mu)^2 + i\sum_\mu s^\mu \bar{s}^\mu} \times \left\{ \int \prod_\mu dh^\mu d\bar{h}^\mu e^{i\sum_\mu h^\mu \bar{h}^\mu - \frac{1}{2\lambda_1 N_1}\left[\sum_\mu \bar{s}^\mu \sigma(h^\mu)\right]^2 - \frac{1}{2\lambda_0 N_0}\sum_i \left[\sum_\mu \bar{h}^\mu x_i^\mu\right]^2} \right\}^{N_1} , \tag{5.15}$$

where we have dropped the index i on h_i^μ and collected the N_1 identical integrals under the curl brackets. At this point, the integral on $\bar{\mathbf{h}}$ is again Gaussian and it leaves the term

$$\frac{e^{-\frac{1}{2}\sum_{\mu\nu} h^\mu C^{-1}_{\mu\nu} h^\nu}}{\sqrt{(2\pi)^P \det C}} , \tag{5.16}$$

where $\mathbf{C}$ is the covariance matrix of the training set, defined as $C_{\mu\nu} = \frac{1}{\lambda_0 N_0}\sum_i x_i^\mu x_i^\nu$. Thus, we are in a situation similar to the one present in Eq. (4.31). In that case, we expanded the exponential and then we made use of the CLT to gain a Gaussian equivalence. However, here we can not repeat this strategy, because the quantity $(N_1)^{-1}(\sum_\mu \bar{s}^\mu h^\mu)^2$ is of order $\alpha = P/N_1$, which is finite in the finite-width limit. Moreover, the presence of the non-linear function σ does not make this integral analytically solvable through any kind of transformation, like for example the Hubbard-Stratonovich one (this would be the case if $\sigma(x) = x$). However, there is a theorem that can be invoked to regain some sort of Gaussian equivalence. Note that Gaussian equivalence principles have been already

employed for random and generic feature models [55, 83, 84, 85].

Let us describe briefly what states the Breuer-Major theorem.

The Breuer-Major theorem

The Breuer-Major (BM) theorem and its extensions deal with sequences of random variables in the form of

$$S_N = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} c_i F(x_i) \tag{5.17}$$

where $N \geq 1$, c_i are coefficient and x_i denotes some variables that can be collected in the vector $\mathbf{x}$. It is clear that if the x_i are distributed independently, i.e. the covariance matrix $\mathbf{C} = N^{-1}\mathbf{x}^\top \mathbf{x} = \mathbb{1}$ and F is the identity function or a well-behaved non-linearity, the random variable $S = \lim_{N\to\infty} S_N$ is normally distributed, providing that the mean and variance of x_i are finite and the coefficients c_i satisfy the Linderberg's condition. The BM theorem extends this reasoning to Gaussian Processes, providing the conditions that guarantee the convergence of S_N to a Gaussian distribution. Let us state the theorem as it is reported in [86], recalling the Hermite rank of a function F is the smallest positive integer that appears in the decomposition over the Hermite polynomials:

Theorem 2 (Breuer and Major,1983)**.** *Let $x = (x_k)_{k\in\mathbb{Z}}$ be a stationary unidimensional GP with covariance $C(i-j)$. Let $\mathbb{E}[F(x_1)] = 0$ and $\mathbb{E}[F^2(x_1)] < \infty$ and assume the function F has Hermite rank $R \geq 1$. Suppose that*

$$\sum_{j\in\mathbb{Z}} |C_{1j}|^R < \infty\,. \tag{5.18}$$

Then $\sigma^2 \equiv \mathbb{E}[F^2(x_1)] + 2\sum_{j=1}^{\infty} \mathbb{E}[F(x_1)F(x_j)]$ is finite. Moreover, the sequence of random variables

$$S_N = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} F(x_i) \;\; N \geq 1 \tag{5.19}$$

converges in distribution to a Gaussian distribution with zero mean and variance σ^2, i.e. to $\mathcal{N}(0, \sigma^2)$.

For the aim of the calculations, we need stronger statements than the hypothesis. Indeed, we would like to exploit the BM theorem with the quantity $q \equiv (\lambda_1 N_1)^{-1/2} \sum_\mu \bar{s}^\mu \sigma(h^\mu)$. There are two concerns for doing that. First, we have to consider a non-stationary

covariance matrix, because for the vast majority of real datasets, this is no longer true. However, this condition can be relaxed and substituted by the weaker one

$$\sum_{j\in\mathbb{Z}} |C_{ij}|^R < B_0 \;\; \forall i \in \mathbb{Z}, \tag{5.20}$$

where B_0 is a positive constant, and this requires a uniform convergence of the elements of the covariance. The second concern is the fact we will need to consider a more general sequence of nonlinear function $c_i F(x_i)$, with weights $c_i \neq 1$. This statement has been addressed recently in [87] under some technicality. While these conditions are not mathematically guaranteed, in the case of real datasets and most used activation functions, it is reasonable to take them as assumptions, at least at the physics level of rigor.

Let us return to the computation of Z_{FCN}. We have to calculate N_1 identical integrals in the form

$$\int d^P h \, \mathcal{N}_{\mathbf{h}}(0, \mathbf{C}) e^{-\frac{1}{2}\left[\frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu \bar{s}^\mu \sigma(h^\mu)\right]^2} . \tag{5.21}$$

In order to do this, we include in the partition function the variable q, previously defined in text right after the BM theorem, inserting a Dirac's delta, leaving to the problem of finding its probability distribution $P(q)$:

$$P(q) = \int d^P h \, \mathcal{N}_{\mathbf{h}}(0, \mathbf{C}) \delta \left(q - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu \bar{s}^\mu \sigma(h^\mu) \right) . \tag{5.22}$$

In the proportional limit (5.1) and making the assumptions for which the extension statements are guaranteed, q satisfies the hypothesis of the BM theorem, and so its probability distribution converges to a Gaussian, $P(q) \to \mathcal{N}(0, Q(\bar{\mathbf{s}}, \mathbf{C}))$, where $Q(\bar{\mathbf{s}}, \mathbf{C})$ is defined by the theorem as

$$Q(\bar{\mathbf{s}}, \mathbf{C}) \equiv \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^\mu \left[\int d^P h \, \mathcal{N}_{\mathbf{h}}(0, \mathbf{C}) \sigma(h^\mu) \sigma(h^\nu) \right] \bar{s}^\nu \equiv \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^\mu K_{\mu\nu} \bar{s}^\nu , \tag{5.23}$$

where we have introduced the matrix $K_{\mu\nu}$, which turns out to be exactly the NNGP kernel, defined in (4.33), apart from the factor λ_1, that it is deliberately kept explicit.

It is crucial to note that, there exist special configurations $\bar{\mathbf{s}}$ in the domain of integration for which it is not allowed to invoke such a Gaussian equivalence, for example, $\bar{\mathbf{s}} = (1, 0, \cdots, 0)$

(remember that the components of $\bar{\mathbf{s}}$ here play the role of the coefficients c_i previously mentioned). In our derivation, we are assuming that the contribution of these special configurations is negligible in the thermodynamic limit. Moreover, for simplicity here we have assumed that the variable q has zero mean, a condition verified as long as

$$\int d^P h \, \mathcal{N}_{\mathbf{h}}(0, \mathbf{C}) \sigma(h^\mu) = 0 \,, \tag{5.24}$$

which means σ has to be an odd-function. The extension to even activation functions requires further calculations that can be found in [81].

Now, from the N_1 identical integrals in curl bracket in Eq. (5.15), it remains to integrate only the variable q, and this gives

$$\{\cdots\}^{N_1} = \left\{ \int \frac{dq \, e^{-\frac{q^2}{2} - \frac{q^2}{2Q(\bar{\mathbf{s}}, \mathbf{C})}}}{\sqrt{2\pi Q(\bar{\mathbf{s}}, \mathbf{C})}} \right\}^{N_1} = \{Q(\bar{\mathbf{s}}, \mathbf{C}) + 1\}^{-\frac{N_1}{2}} \,. \tag{5.25}$$

For completing the calculation, we have to introduce a new set of variable $Q, \bar{Q}$, again inserting a Dirac's delta, to deal with the implicit dependency of $Q(\bar{\mathbf{s}}, \mathbf{C})$ on the $\bar{\mathbf{s}}$. In this way, the integrals on $\bar{\mathbf{s}}$ and $\mathbf{s}$ are Gaussians and easy to compute. The final form of the partition function, after dropping constant terms which do not affect the theory, reads

$$Z_{FCN} = \int dQ d\bar{Q} \, e^{-\frac{N_1}{2} S_{FCN}(Q, \bar{Q})} \,, \tag{5.26}$$

where $S_{FCN}(Q, \bar{Q})$ is an effective action defined as

$$\begin{aligned} S_{FCN}(Q, \bar{Q}) = & - Q\bar{Q} + \log(1 + Q) + \frac{\alpha_1}{P} \mathrm{Tr} \log \left[\beta \left(\frac{\mathbb{1}}{\beta} + \frac{\bar{Q}\mathbf{K}}{\lambda_1} \right) \right] + \\ & + \frac{\alpha_1}{P} \mathbf{y}^\top \left(\frac{\mathbb{1}}{\beta} + \frac{\bar{Q}\mathbf{K}}{\lambda_1} \right)^{-1} \mathbf{y} \,, \end{aligned} \tag{5.27}$$

where we denote $\mathbf{v}^\top \mathbf{M} \mathbf{v} \equiv \sum_{\mu\nu} M_{\mu\nu} v_\mu v_\nu$. We can think of the quantity $\bar{Q}\mathbf{K}$ as a rescaling of the NNGP kernel, due to the effect of training on an extensive (respect to N_1) number of data, i.e. as a **Kernel Renormalization**.

Since $N_1 \to \infty$, one can exploit the form (5.26) of Z_{FCN} to perform the saddle-point (SP) method. By taking the zero-temperature limit $\beta \to \infty$, the solution of the SP equations

$\partial_Q S_{FCN} = 0$ and $\partial_{\bar{Q}} S_{FCN} = 0$ is

$$\bar{Q} = (1+Q)^{-1}\,, \quad Q = \frac{\alpha_1}{\bar{Q}} - \frac{\alpha_1}{\bar{Q}^2}\frac{1}{P}\mathbf{y}^\top \left(\frac{\mathbf{K}}{\lambda_1}\right)^{-1}\mathbf{y}\,, \tag{5.28}$$

from which the unique solution (given the condition $Q \geq -1$ which derives from (5.25)) reads

$$\bar{Q}^* = \frac{1}{2}\left(\sqrt{(\alpha_1-1)^2 + 4\alpha_1 P^{-1}\mathbf{y}^\top \left(\frac{\mathbf{K}}{\lambda_1}\right)^{-1}\mathbf{y}} - (\alpha_1 - 1)\right)\,. \tag{5.29}$$

We focus on the zero-temperature limit because in this regime the system described by (5.4) is forced to configurations that minimize the energy, i.e. the training loss, and this implies we can consider the neural network in the regime after the interpolation threshold (see Fig. 4.5). However, the whole framework provides analytical results for any non-zero temperature, and one can manage to compute deviation from the ideal zero-loss situation.

Moreover, this theoretical framework allows us to compute observables, like the generalization error. Indeed, following the same reasoning for finding the effective action (5.27), one can easily recover the usual bias-variance decomposition:

$$\begin{aligned}
\left\langle \epsilon_g(\mathbf{x}^0, y^0)\right\rangle &= \left(y^0 - \Gamma_1\right)^2 + \sigma_1^2\,,\\
\Gamma_1 &= \frac{\bar{Q}}{\lambda_1}\sum_{\mu\nu} K_\mu(\mathbf{x}^0)\left(\frac{\mathbb{1}}{\beta} + \frac{\bar{Q}\mathbf{K}}{\lambda_1}\right)^{-1} y_\nu\,,\\
\sigma_1^2 &= \frac{\bar{Q}}{\lambda_1}\left[K_0(\mathbf{x}^0) - \frac{\bar{Q}}{\lambda_1}\sum_{\mu\nu} K_\mu(\mathbf{x}^0)\left(\frac{\mathbb{1}}{\beta} + \frac{\bar{Q}\mathbf{K}}{\lambda_1}\right)^{-1} K_\nu(\mathbf{x}^0)\right]\,,
\end{aligned} \tag{5.30}$$

where $K_\mu(\mathbf{x})$ and $K_0(\mathbf{x})$ can be computed from the functional definition of the NNGP kernel $\mathbf{K}$:

$$\begin{aligned}
K_\mu(\mathbf{x}^0) \equiv \mathbf{K}(\mathbf{x}^\mu, \mathbf{x}^0) &= \int \frac{dt_1 dt_2}{\sqrt{\det(2\pi\tilde{\mathbf{C}}_\mu)}}\mathcal{N}_{\mathbf{t}}(0, \tilde{\mathbf{C}}_\mu)\sigma(t_1)\sigma(t_2)\,,\\
K_0(\mathbf{x}^0) \equiv \mathbf{K}(\mathbf{x}^0, \mathbf{x}^0) &= \int \frac{dt}{\sqrt{2\pi C_{00}}}\mathcal{N}_t(0, C_{00})\sigma(t)^2\,,
\end{aligned} \tag{5.31}$$

where

$$\tilde{\mathbf{C}}_\mu = \begin{pmatrix} C_{\mu\mu} & C_{\mu 0} \\ C_{\mu 0} & C_{00}\end{pmatrix}\,,\; C_{\mu 0} = (\lambda_0 N_0)^{-1}\sum_i x_i^\mu x_i^0\,,\; C_{00} = (\lambda_0 N_0)^{-1}\sum_i (x_i^0)^2\,. \tag{5.32}$$

In the zero-temperature limit, the form of the bias and variance is quite similar to the one found for the predictor statistics of Bayesian Neural Network in the IW limit (4.36), i.e. when the BNN is equivalent to a Gaussian Process. The difference stands in the variance because in the finite-width limit, it is scaled from the NNGP one by a factor $\bar{Q}^*$, where $\bar{Q}^*$ is the solution of the SP equation in the zero-temperature limit (5.29) (note that the Gaussian prior λ_1 is made explicit here, while included in the definition of $\mathbf{K}$ for the NNGP kernel). The exact form of the NNGP predictor statistics is correctly recovered in the IW limit, i.e. for $\alpha_1 = 0$, since in this case $\bar{Q}^* = 1$. Then, in the proportional limit (5.1), the predictor statistics of an FCN is the same as a GP, where the kernel $\mathbf{K}$ is renormalized, naively by rescaling it with a constant, even though its posterior distribution is not equivalent to the one of a GP.

Moreover, by performing a Taylor expansion around the IW limit $\alpha_1 = 0$, we can compute the corrections to the IW of the generalization error, which at the leading order reads

$$\Delta\epsilon_g \equiv \epsilon_g - \epsilon_g^{\text{IW}} \propto \alpha_1 \left(\frac{1}{P} \mathbf{y}^\top \mathbf{K}^{-1} \mathbf{y} - 1 \right) . \tag{5.33}$$

This means that the finite-width network will outperform its IW counterpart whenever $\frac{1}{P}\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y} < 1$. Surprisingly, it seems there is a hidden relation with Student's t-processes. Indeed, the authors in [88] showed that whenever this quantity is smaller than one then the Student's t-process having as kernel $\mathbf{K}$ has a smaller variance with respect to the corresponding GP having the same kernel. This is connected directly to the fact that, in the bias-variance decomposition, the variance of the finite-width network is lower than the IW counterpart.

This effective theory for a 1HL FCN was corroborated by intensive numerical simulations, performed in [81]. In Fig. 5.3, we report one of their checks, whose details can be found in the article. When taking Eq. (5.30) at $\beta = \infty$ and at the saddle-point, one realizes that the bias term remains constant and it is independent on both the ratio $\bar{Q}^*/\lambda_1$ and N_1. Instead, the variance depends on N_1 and decreases as $1/\sqrt{\lambda_1}$ (see Eq. (5.29)). For these reasons, at fixed P the generalization error should approach monotonically the IW one from above or below, depending on the value of the observable $\frac{1}{P}\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}$ (see Eq. (5.33)). Moreover, increasing the magnitude of the Gaussian prior λ_1 should improve the generalization error for any hidden-layer size and the dependence on N_1 should disappear

in the λ_1-large limit.

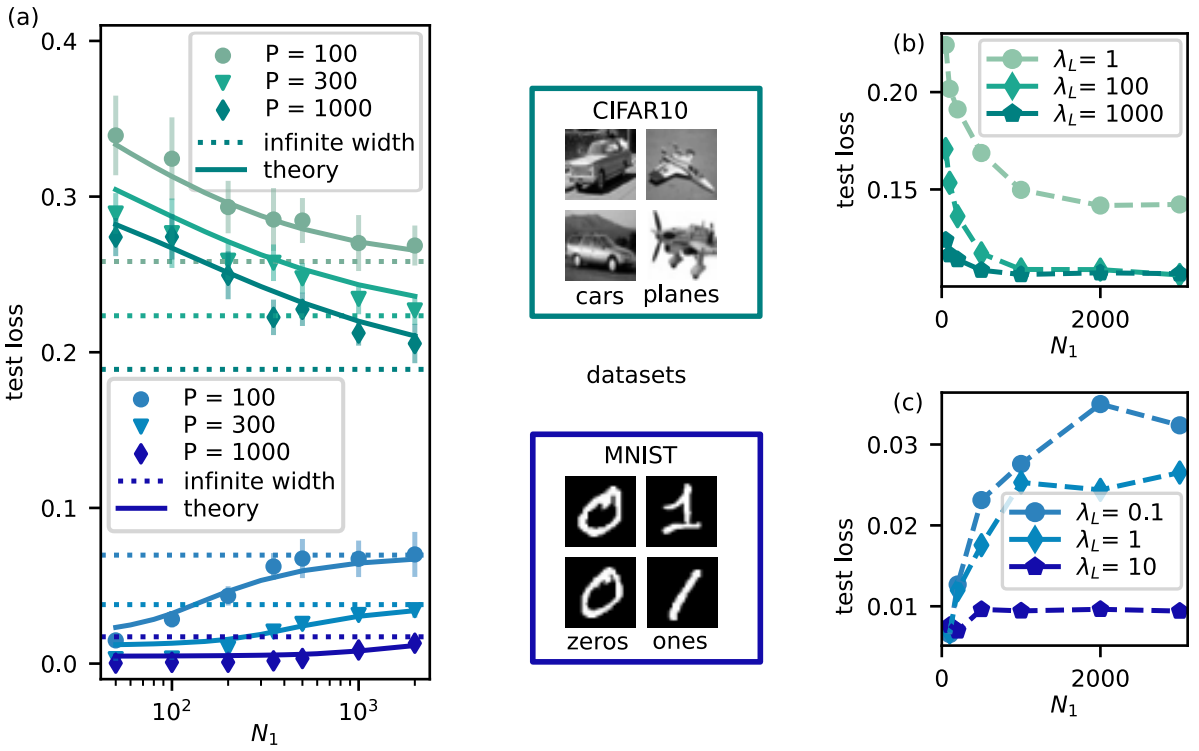

Figure 5.3: Testing the prediction of the effective theory of a FCN. This picture is a revised version of one of the figures in [81]. With their experiments, they show (a) that the experimental test loss (points) is in good agreement with the theoretical expectation (solid lines), computed from Eq. (5.30) taken at $\beta = \infty$. The generalization curves are monotonically decreasing or increasing with N_1 depending on the fact that the observable $\frac{1}{P}\mathbf{y}^\top\mathbf{K}^{-1}\mathbf{y}$ is larger or smaller than 1, respectively for the dataset CIFAR10 or MNIST. Moreover, panels (b) and (c) show that the generalization loss decreases for any N_1 when the magnitude of the Gaussian prior of the last layer λ_1 grows and the dependence of the layer size disappears in the λ_1-large limit.

6 Feature Learning in finite-width networks

In this chapter, we will exploit the theoretical framework previously defined for FCNs, to formulate an analogous effective theory in the presence of a convolutional layer, which was part of our work [89]. For simplicity, we will restrict to the case of one-dimensional convolutions, but the formalism can be easily generalized to higher dimensions. These calculations will allow us to study the behavior of the generalization error and to compare FCNs and CNNs. Moreover, we will show how to compute the expected value of the similarity matrix in both architectures, highlighting the major differences. Finally, we will present numerical experiments to test our theory.

6.1 Bayesian effective action for CNNs in the proportional limit

Let us collect the weights of a mask of a particular channel, from now on indexed with the letter a, in a $N_c \times M$ matrix $\mathbf{W}$, where N_c is the number of channels, while M is the dimension of the mask. Following the reasoning on the FCNs, we first define the pre-activations as

$$h_i^a(\mathbf{x}) \equiv \frac{1}{\sqrt{M}} \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} W_m^a x_{Si+m} \,, \tag{6.1}$$

where S is the stride of the filter (the mask is convolved every S pixels) and the spatial index i runs over the input coordinate from 1 to $\lfloor N_0/S \rfloor$. For simplicity, we define periodic boundary conditions over the input coordinates and we consider odd values of M (the floor operation $\lfloor \cdot \rfloor$ takes the integer part of a rational number). In this way, the N_c feature maps are vectors of length $\lfloor N_0/S \rfloor$. This means that the "flatten" layer, where we collect all the neurons in all the feature maps, will have $N_c \times \lfloor N_0/S \rfloor$ neurons. Then, we collect the weights of the last layer, the one that produces the output, in the matrix whose elements are v_i^a. Thus, the function which implements a shallow CNN architecture is given by

$$f_{CNN}(\mathbf{x}) = \frac{1}{\sqrt{N_c \lfloor N_0/S \rfloor}} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sum_{a=1}^{N_c} v_i^a \sigma(h_i^a(\mathbf{x})) \,. \tag{6.2}$$

Our goal is to derive an effective action as was done for FCNs in [81], considering as thermodynamic limit the conditions

$$P, N_c \to \infty \,, \text{ such that } \alpha_c \equiv \frac{P}{N_c} = const \,. \tag{6.3}$$

The strategy and computation are quite similar to the FCN case but with few important differences. The starting point is the definition of the partition function, which is the same as Eq. (5.8), apart from the differentials and NN function

$$Z_{CNN} = \int \prod_{i,a=1}^{\lfloor N_0/S \rfloor, N_c} dv_i^a \prod_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} \prod_{a=1}^{N_c} dW_m^a \, e^{-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^\mu - f_{CNN}(\mathbf{x}^\mu))^2 - \frac{\lambda_0}{2}\|W\|^2 - \frac{\lambda_1}{2}\|v\|^2} \,. \tag{6.4}$$

Similarly to the FCN case, we introduce two sets of new variables, the pre-activations h_{ia}^μ and the outputs s^μ, inserting families of Dirac's deltas in Fourier representation, gaining

$$\begin{aligned} Z_{CNN} = \int \prod_{i,a,\mu} dh_{ia}^\mu d\bar{h}_{ia}^\mu \prod_\mu ds^\mu d\bar{s}^\mu \, e^{-\frac{\beta}{2}\sum_\mu (y^\mu - s^\mu)^2 + i\sum_\mu s^\mu \bar{s}^\mu + i\sum_{\mu ia} h_{ia}^\mu \bar{h}_{ia}^\mu} \\ \times \left\langle e^{-i\sum_\mu \bar{s}^\mu \left(\frac{1}{\sqrt{N_1}} \sum_{ia} v_i^a \sigma(h_{ia}^\mu) \right)} \right\rangle_{\mathbf{v}} \times \left\langle e^{-i\sum_{ia\mu} \bar{h}_{ia}^\mu \left(\frac{1}{\sqrt{M}} \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} W_m^a x_{Si+m}^\mu \right)} \right\rangle_{\mathbf{W}} \,, \end{aligned} \tag{6.5}$$

where we have defined the number of neurons in the last layer as $N_1 \equiv N_c \lfloor N_0/S \rfloor$, and for the sake of notation we dropped any extremes of production and summation (remember that $a = 1, \cdots, N_c$, $i = 1, \cdots, \lfloor N_0/S \rfloor$ and $\mu = 1, \cdots, P$). The pre-activation h_{ia}^μ is the one defined in Eq. (6.1), with $\mathbf{x}^\mu$ as input, and thus we consider it as a three-dimensional matrix of dimensions $P \times N_c \times \lfloor N_0/S \rfloor$. The two average values over the prior distributions (which are Gaussian) of the weights are both multivariate Gaussian integrals and give the contributions

$$\prod_a \exp\left\{ -\frac{1}{2\lambda_1 N_1} \sum_i \left[\sum_\mu \bar{s}^\mu \sigma(h_{ia}^\mu) \right]^2 \right\} \,, \quad \prod_a \exp\left\{ \frac{1}{2} \sum_{ij} \sum_{\mu\nu} \bar{h}_{ia}^\mu C_{\mu\nu}^{ij} \bar{h}_{ja}^\nu \right\} \,, \tag{6.6}$$

respectively for the integrals on $\mathbf{v}$ and $\mathbf{W}$, and we recall λ_0 and λ_1 being the variance of the Gaussian priors of the convolution and last layer respectively. Here, we have already highlighted the fact we can factorize over the channel index a and we have defined the

local covariance matrix C, whose entries are

$$C^{ij}_{\mu\nu} \equiv \frac{1}{\lambda_0 M} \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} x^{\mu}_{Si+m} x^{\nu}_{Sj+m} \,. \tag{6.7}$$

Then, the integrals on the $\bar{\mathbf{h}}$ variable can be done since Gaussian, and we get

$$\begin{aligned} Z_{CNN} = \int \prod_{\mu} ds^{\mu} d\bar{s}^{\mu} e^{-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^{\mu}-s^{\mu})^2 + i\sum_{\mu=1}^{P} s^{\mu}\bar{s}^{\mu}} \\ \times \left\{ \int \prod_{\mu i} dh^{\mu}_{i} \, \mathcal{N}_{\mathbf{h}}(0,\mathbf{C}) e^{-\frac{1}{2\lambda_1 N_1}\sum_{i}\left[\sum_{\mu=1}^{P}\bar{s}^{\mu}\sigma(h^{\mu}_{i})\right]^2} \right\}^{N_c} . \end{aligned} \tag{6.8}$$

As was done for the FCN case, to deal with the non-linear term, we define a collection of $\lfloor N_0/S \rfloor$ deltas:

$$1 = \int \prod_{i} dq_i \delta \left(q_i - \frac{1}{\lambda_1 N_1} \sum_{\mu} \bar{s}^{\mu} \sigma(h^{\mu}_{i}) \right) . \tag{6.9}$$

Analogously to the FCN case, to continue the calculation, we need to invoke a multivariate version of the B-M theorem (Th. 2). Despite the fact there are no tracks of an extended version of this theorem, it seems reasonable to assume its validity. In this way, we retrieve an informally justified Gaussian equivalence on the variables q_i. This implies that their joint distribution converges to a multivariate Gaussian distribution with zero mean and covariance matrix $\mathbf{Q}$, whose elements, which depend on $\bar{\mathbf{s}}$ and $\mathbf{C}$, are given by

$$Q_{ij}\left(\bar{\mathbf{s}}, \mathbf{C}\right) \equiv \left\langle q_i q_j \right\rangle_{\mathbf{h}} = \frac{1}{\lambda_1 N_1} \sum_{\mu\nu} \bar{s}^{\mu} \left\langle \sigma(h^{\mu}_{i}) \sigma(h^{\nu}_{j}) \right\rangle_{\mathbf{h}} \bar{s}^{\nu} \equiv \frac{1}{\lambda_1 N_1} \sum_{\mu\nu} \bar{s}^{\mu} K^{ij}_{\mu\nu} \bar{s}^{\nu} \,, \tag{6.10}$$

where the average operation is done over the distribution of $\mathbf{h}$, which is the Gaussian $\mathcal{N}_{\mathbf{h}}\left(0, \mathbf{C}\right)$, and where we have defined the **local kernel K** to be the four-indices matrix, of dimensions $\lfloor N_0/s \rfloor^2 \times N_c^2$, whose entries are computed via the same functional relations of the FCN case:

$$\begin{aligned} K^{ij}_{\mu\nu} &= \int d^2 t \, \mathcal{N}_{\mathbf{t}} \left(0, \tilde{\mathbf{C}}^{ij}_{\mu\nu}\right) \sigma(t_1)\sigma(t_2) \,, \\ \tilde{\mathbf{C}}^{ij}_{\mu\nu} &= \begin{pmatrix} C^{ii}_{\mu\mu} & C^{ij}_{\mu\nu} \\ C^{ij}_{\mu\nu} & C^{jj}_{\nu\nu} \,. \end{pmatrix} . \end{aligned} \tag{6.11}$$

In this way, the integral in the curl brackets in (6.8) is Gaussian and reads

$$\{\cdots\}^{N_c} = \left\{ \int \prod_i dq_i \, \frac{e^{-\frac{1}{2}\sum_{ij} q_i \left(\delta_{ij} + Q_{ij}^{-1}\right) q_j}}{\sqrt{\det(2\pi \mathbf{Q})}} \right\}^{N_c} = \left\{ \det\left(\mathbb{1} + \mathbf{Q}\right) \right\}^{-\frac{N_c}{2}} = e^{-\frac{N_c}{2} \operatorname{Tr} \log(\mathbb{1} + \mathbf{Q})} \,, \tag{6.12}$$

where we have omitted the dependency of $\mathbf{Q}$ on $\bar{\mathbf{s}}$ and $\mathbf{C}$. At this stage, as was the case when dealing with the FCN, we have to introduce the variable $\mathbf{Q}$ to deal with the implicit dependency on $\bar{\mathbf{s}}$:

$$\begin{aligned} 1 &= \int \prod_{ij} dQ_{ij} \, \delta \left(Q_{ij} - \frac{1}{\lambda_1 N_1} \sum_{\mu\nu} \bar{s}^\mu K_{\mu\nu}^{ij} \bar{s}^\nu \right) \\ &\propto \int \prod_{ij} dQ_{ij} d\bar{Q}_{ij} \, e^{i \sum_{ij} Q_{ij} \bar{Q}_{ij} - \frac{i}{\lambda_1 N_1} \sum_{ij} \bar{Q}_{ij} \sum_{\mu\nu} \bar{s}^\mu K_{\mu\nu}^{ij} \bar{s}^\nu} \,, \end{aligned} \tag{6.13}$$

where we have shown explicitly the Fourier representation for clarity. Indeed, Eq. (6.13) shows the main difference between the FCN and CNN, which stands in the renormalization of the kernel. In the FCN case, the kernel $\mathbf{K}$ is globally renormalized, in the sense that each element of the matrix is rescaled by the same scalar factor $\bar{Q}$. Instead, in the CNN case, the kernel is a 4-indices matrix and the renormalization acts locally, in the sense that $\mathbf{K}$ is rescaled through a matrix multiplication with $\bar{\mathbf{Q}}$ in the subspace regarding the pixel positions. We then define the **local kernel renormalization** of $\mathbf{K}$ to be:

$$\left[K_{CNN}^{(\mathrm{R})}(\bar{\mathbf{Q}}) \right]_{\mu\nu} \equiv \frac{1}{\lambda_1 \lfloor N_0/S \rfloor} \sum_{ij=1}^{\lfloor N_0/S \rfloor} \bar{Q}_{ij} K_{\mu\nu}^{ij} \,, \tag{6.14}$$

where the choice of the coefficient will allow us to factorize the term N_c in the effective action. At this point, we are left with the integrals on the variables $\mathbf{s}$ and $\bar{\mathbf{s}}$, which are both Gaussian, and the remaining terms form the effective action for a shallow CNN architecture

$$Z_{CNN} = \int \prod_{ij} dQ_{ij} d\bar{Q}_{ij} \, e^{-\frac{N_c}{2} S_{CNN}(\mathbf{Q}, \bar{\mathbf{Q}})} \,, \tag{6.15}$$

where the action reads

$$\begin{aligned} S_{CNN}\left(\mathbf{Q}, \bar{\mathbf{Q}}\right) = &- \operatorname{Tr} \mathbf{Q}\bar{\mathbf{Q}} + \operatorname{Tr} \log\left(\mathbb{1} + \mathbf{Q}\right) + \\ &+ \frac{\alpha_c}{P} \operatorname{Tr} \log \beta \left(\frac{\mathbb{1}}{\beta} + \mathbf{K}_{CNN}^{(\mathrm{R})}(\bar{\mathbf{Q}}) \right) + \frac{\alpha_c}{P} \mathbf{y}^\top \left(\frac{\mathbb{1}}{\beta} + \mathbf{K}_{CNN}^{(\mathrm{R})}(\bar{\mathbf{Q}}) \right)^{-1} \mathbf{y} \,, \end{aligned} \tag{6.16}$$

where the first two traces are over $\lfloor N_0/S \rfloor \times \lfloor N_0/S \rfloor$ operators, while the operator $\frac{\mathbb{1}}{\beta} + \mathbf{K}^{(\mathrm{R})}_{CNN}(\bar{\mathbf{Q}})$ is of dimension $P \times P$.

6.1.1 Local Kernel Renormalization as a mechanism of feature learning

Having defined an effective action, we can now analyze the relative saddle-point equations, $\partial_{Q_{ij}} S_{CNN}\left(\mathbf{Q}, \bar{\mathbf{Q}}\right) = 0$ and $\partial_{\bar{Q}_{ij}} S_{CNN}\left(\mathbf{Q}, \bar{\mathbf{Q}}\right) = 0$, for all ij pairs, in the zero-temperature limit. Exploiting the so-called Jacobi's formula we find

$$\bar{Q}_{ij} = (\mathbb{1} + \mathbf{Q})^{-1}_{ij} \tag{6.17}$$

$$Q_{ij} = \frac{\alpha_c}{P} \sum_{\mu\nu} K^{ij}_{\mu\nu} \left[K^{(R)}_{\mathrm{CNN}}(\bar{\mathbf{Q}})\right]^{-1}_{\mu\nu} + \\ - \frac{\alpha_c}{P} \sum_{\mu\nu} y^\mu \left(\sum_{\omega\epsilon} \left[K^{(R)}_{\mathrm{CNN}}(\bar{\mathbf{Q}})\right]^{-1}_{\mu\omega} K^{ij}_{\omega\epsilon} \left[K^{(R)}_{\mathrm{CNN}}(\bar{\mathbf{Q}})\right]^{-1}_{\epsilon\nu} \right) y^\nu \,. \tag{6.18}$$

In contrast to the FCN case, we can not obtain these matrix equations in closed form. However, we can manage to compute the perturbation around the IW limit $\alpha_c \to 0$. We can parameterized the solution of the first equation as $\bar{Q}_{ij} = \delta_{ij} + \alpha_c \delta\bar{Q}_{ij} + o(\alpha_c)$, where the perturbation reads

$$\delta\bar{Q}_{ij} = \frac{1}{P} \sum_{\mu\nu} y^\mu \left(\sum_{\omega\epsilon} \bar{K}^{-1}_{\mu\omega} K^{ij}_{\omega\epsilon} \bar{K}^{-1}_{\epsilon\nu} \right) y^\nu - \frac{1}{P} \sum_{\mu\nu} K^{ij}_{\mu\nu} \bar{K}^{-1}_{\mu\nu} \,, \tag{6.19}$$

where we have introduced the IW CNN averaged kernel defined as

$$\bar{K}_{\mu\nu} \equiv \frac{1}{\lambda_1 \lfloor N_0/S \rfloor} \sum_i K^{ii}_{\mu\nu} \,. \tag{6.20}$$

It is worth noting that this averaged kernel has been already found in [56]. In this work, the authors found that a CNN with infinitely many channels, $N_c \gg P$, is equivalent to a GP having as kernel the one defined in Eq. (6.20). In our theoretical framework, we recover this result solving the SP equations in the IW limit $\alpha_c = 0$, for which $\bar{Q}_{ij} = \delta_{ij}$ and so the renormalized local kernel (6.14) becomes equal to the averaged one (6.20).

From Eq. (6.19), we can gain some important insights about the role of the local kernel renormalization, and here we explain our claim about how it is related to the characteristic

feature learning in CNN architectures. In our framework, the effect of training leads back to the optimization of the variables that describe the effective action of the architecture, namely the elements of the matrix $\bar{\mathbf{Q}}$ for the shallow CNN case. These elements are in one-to-one correspondence with the combinations of pairs of the $\lfloor N_0/S \rfloor$ patches of the local covariance matrix. The first term in Eq. (6.19) is a measure of the relative importance of the pairwise spatial correlations in the input dataset w.r.t. the labels. Indeed, if the CNN local kernel $K^{ij}_{\mu\nu}$ at the spatial position (i, j) will have a significant overlap with the effective label $\sum_\omega \bar{K}^{-1}_{\mu\omega} y^\omega$, the element $\bar{Q}_{ij}$ will differ from one. In this sense, we can interpret the matrix $\bar{\mathbf{Q}}$ as a data-dependent feature matrix, which gives information about how a given component of the local kernel contributes to the renormalized one $\mathbf{K}^{(\mathrm{R})}_{CNN}$.

At this point, one can wonder what are the characteristics of the CNNs that lead to this peculiar local kernel renormalization. Certainly, the first difference between FCNs and CNNs resides in the local connectivity. In FCNs, all the neurons of a layer interacts with all the neurons in the next layer, while in CNNs a patch of neuron in the convolutional layer interacts only with a single portion of the input data at a time. However, we find this ingredient to be not sufficient, since at the core of the local kernel renormalization seems to be the *weight sharing*. Indeed, with our framework we study a particular type of CNN, called Locally Connected Network (LCN) and we find that in this case, the local renormalization does not occur. LCNs are CNN-like structures, with the difference that the mask of a channel is not shared with the whole input data. Instead, each local patch interacts with a different set of neurons in the convolutional layer. In the CNNs, the weights are described by a $N_c \times M$ matrix W^a_m, while in the LCNs these are collected in a $N_c \times M \times \lfloor N_0/S \rfloor$ matrix W^a_{mi}. The absence of weight sharing leads to a different kernel renormalization, which reads

$$K^{(\mathrm{R})}_{LCN} = \frac{1}{\lambda_1 \lfloor N_0/S \rfloor} \sum_{i=1}^{\lfloor N_0/s \rfloor} \bar{Q}_i K^{ii}_{\mu\nu} \,. \tag{6.21}$$

Then, for LCNs, i.e. for CNNs without weight sharing, the effect of training is to renormalize only the diagonal components of the local kernel $K^{ij}_{\mu\nu}$ through the multiplication with a vector $\bar{Q}_i$, and this implies these networks exploit only the correlation of a single local patch with itself through the dataset. Meanwhile, in CNNs, the presence of weight sharing allows to gain information about correlation between different local

Architecture	Kernel type	Dim.	IW Kernel	Renormalized Kernel
FCN	$K_{\mu\nu}$	$P \times P$	$K_{\mu\nu}$	$\bar{Q}K_{\mu\nu}$
LCN	$K^{ii}_{\mu\nu}$	$\lfloor \frac{N_0}{S} \rfloor \times P \times P$	$\sum_i K^{ii}_{\mu\nu}$	$\sum_i \bar{Q}_i K^{ii}_{\mu\nu}$
CNN	$K^{ij}_{\mu\nu}$	$\lfloor \frac{N_0}{S} \rfloor \times \lfloor \frac{N_0}{S} \rfloor \times P \times P$	$\sum_i K^{ii}_{\mu\nu}$	$\sum_{ij} \bar{Q}_{ij} K^{ij}_{\mu\nu}$

Table 6.1: Summary of the analytical results for 1HL networks. We identify three different types of kernels in the Bayesian effective action for FCN, LCN, and convolutional networks: a global kernel in the FCN case, a local diagonal kernel in LCNs, and a local (with additional non-diagonal components) in CNNs. The infinite-width kernel that describes LCNs and CNNs is the same, as shown in [56]. In the proportional limit, the renormalization of the FCN kernel is scalar, i.e. the NNGP kernel is only globally rescaled by a data-dependent number $\bar{Q}$. In LCNs, we can adjust a vector of components $\bar{Q}_i$, which determines how much the self-correlations of the i-th patch of the diagonal local kernel contribute to the final prediction. In CNNs, due to locality and weight-sharing, we can fine-tune a full matrix $\bar{Q}_{ij}$, which controls how much the local correlations of patches i and j will enter in the effective kernel learned by the network after training. This is what we call *local kernel renormalization*.

patches at different locations through the dataset. Details about the computation of the effective action for an LCN can be found in Appendix A1. We summarize the analytical results for shallow networks in Table 6.1, where we collect the kernel behavior under the different architectures, how they are affected by the training in the proportional limit (finite width), and the form they assume in the infinite-width limit. In order to visualize the different types of kernel in FCNs and CNNs, in Fig. 6.1 we show an example of kernel matrix for a subsample of the CIFAR10 dataset [60].

6.2 Computing observables: the *similarity matrix*

The formalism we have employed allowed us to integrate over the weights of the NN and to describe the effect of training with an effective action. Moreover, we can exploit standard methods of Statistical Mechanics to predict statistical values of observables of the theory, i.e. we can compute expected values as $\langle O \rangle \equiv \int \mathcal{D}\theta \, e^{-S(\theta)} O(\theta)$. One interesting quantity to observe is the so-called **similarity matrix**, which is the covariance matrix of the hidden representations of the training set. This quantity is related to the inner product of the data in the latent space (the one on which the data are described in the internal layers of the network) and so to the renormalized kernel of the architecture under study. Indeed, at the moment, we do not have a way to extract the renormalized kernel

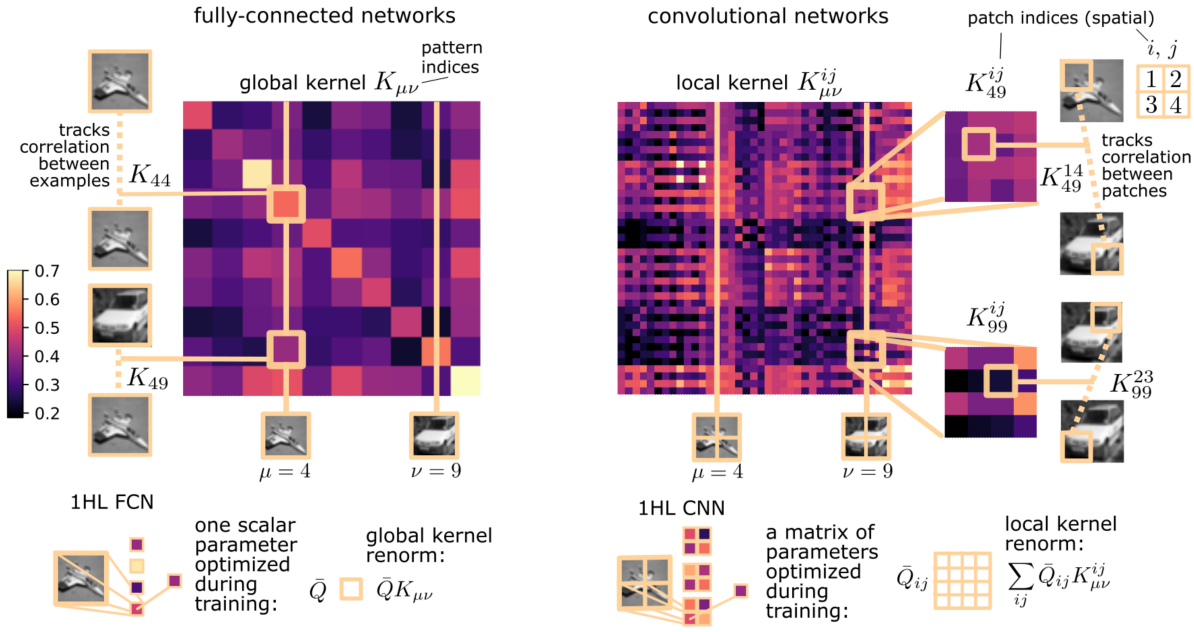

Figure 6.1: Global and local kernels in fully-connected and convolutional one hidden layer networks. Our calculations in the proportional limit $P, N_1 \to \infty$ at fixed $\alpha_1 = P/N_1$ reveal a striking difference between FCNs and CNNs. In the first case, the NNGP kernel $K_{\mu\nu}$ that enters into the Bayesian effective action is a global one, which is only capable of tracking global correlations between different training patterns (μ, ν) (left). An FCN in this regime will use the information contained in the dataset to fine-tune a single parameter, $\bar{Q}$, that will globally rescale the kernel matrix $K_{\mu\nu}$. On the other hand, in the CNN case, we identify a local kernel with two additional (spatial) indices. If we consider $2d$ images, the effect of the local components is to track local correlations between different patches i and j of (possibly) different pairs of images (μ, ν) (right). The spatial information contained in these correlations allows the CNN to optimize a matrix of parameters $\bar{Q}_{ij}$, which will renormalize the kernel matrix $K^{ij}_{\mu\nu}$ in a non-trivial way. In the figure, we display the four-index local kernel $K^{ij}_{\mu\nu}$ as a matrix using the multi-index notation $K_{(\mu,i),(\nu,j)}$ and choosing an ordering for the multi-index (μ, i). Global and local kernels are computed for a subset of $P = 10$ gray-scaled CIFAR10 images down-sized to $N_0 = 784$. The local kernels are empirically evaluated for $2d$ convolutions with the linear size of the filter mask $M = 14$ and stride $S = 14$ (non-overlapping filters).

directly from the measure of a physical observable. The similarity matrix can represent one indirect experimental evidence of the fact the kernel renormalizes differently in FCNs and CNNs.

The definition of the similarity matrix, introduced in Eq. (5.6), depends on the architecture, and specializing to FCN and CNNs, it reads

$$O^{\mathrm{FCN}}_{\mu\nu} = \frac{1}{N_1}\sum_{i=1}^{N_1}\sigma\left(h_i^\mu\right)\sigma\left(h_i^\nu\right), \qquad h_i^\mu \equiv \frac{\mathbf{W_i}\cdot\mathbf{x}^\mu}{\sqrt{N_0}} \tag{6.22}$$

$$O^{\mathrm{CNN}}_{\mu\nu} = \frac{1}{N_c}\sum_{a=1}^{N_c}\frac{1}{\lfloor\frac{N_0}{S}\rfloor}\sum_{i=1}^{\lfloor N_0/S\rfloor}\sigma(h^\mu_{ia})\sigma(h^\nu_{ia}), \qquad h^\mu_{ia} \equiv \frac{1}{\sqrt{M}}\sum_{m=-\lfloor M/2\rfloor}^{\lfloor M/2\rfloor} W^a_m x^\mu_{Si+m}\,. \tag{6.23}$$

The average FCN similarity matrix turns out to have a simple form, and it depends only on the NNGP kernel $K_{\mu\nu}$ and on a naive combination of the labels y^μ:

$$\left\langle O^{\mathrm{FCN}}_{\mu\nu}\right\rangle = \left(1-\frac{1}{N_1}\right)K_{\mu\nu} + \frac{\lambda_1}{N_1\bar{Q}}y^\mu y^\nu. \tag{6.24}$$

Note that our formalism allows us to retrieve this observable for deep linear networks, consistently finding the same result of [70], that is the same expression with the replacement of the NNGP kernel with the data covariance matrix $C_{\mu\nu}$.

The CNN case turns out to be more complicated. Indeed, there is again a dependency on the NNGP kernel, but the convolutional architecture mixes different labels in a non-trivial way:

$$\begin{aligned}\left\langle O^{\mathrm{CNN}}_{\mu\nu}\right\rangle =& \frac{1}{\lfloor N_0/S\rfloor}\sum_{i=1}^{\lfloor N_0/S\rfloor}K^{ii}_{\mu\nu} + \\ &- \frac{1}{\lambda_1 N_c}\frac{1}{\lfloor N_0/S\rfloor^2}\sum_{i,j=1}^{\lfloor N_0/S\rfloor}\sum_{\lambda,\rho=1}^{P}\bar{Q}_{ij}P^{ij}_{\mu\lambda\nu\rho}\left[(K^{(\mathrm{R})})^{-1}_{\lambda\rho} - \sum_{\epsilon,\omega=1}^{P}(K^{(\mathrm{R})})^{-1}_{\lambda\epsilon}(K^{(\mathrm{R})})^{-1}_{\rho\omega}y^\epsilon y^\omega\right],\end{aligned} \tag{6.25}$$

where $P^{ij}_{\mu\lambda\nu\rho} \equiv \frac{1}{2}\sum_k\left[K^{ik}_{\mu\lambda}K^{kj}_{\nu\rho} + K^{ki}_{\mu\lambda}K^{jk}_{\nu\rho}\right]$. In this case, the labels are mixed through the application of the renormalized kernel, incorporating the effect of training. The calculations that lead to these expressions can be found in Appendix A2, and in the next section, we will show our numerical experiments for checking these predictions.

6.3 FCNs vs CNNs

In the previous sections, we have seen how the theoretical framework introduced in [81] made it possible to gain insights about the hidden behavior of deep non-linear networks. In this framework, NNs are treated as a complex system described by the Loss Function as Hamiltonian, whose degrees of freedom are the weights. In order to solve the theory, i.e. to gain information from the partition function, the paradigm is to switch to other variables, the pre-activation and the network's output, and to integrate out the weights, as it is done in the Bayesian view of NNs. If one carries out the calculations, the partition function reduces in the integration over new variables Q and $\bar{Q}$, treated as *order parameter*, and the system is described by a new effective action $S(Q, \bar{Q})$. We report here the one for an FCN found in [81] and the one for a CNN, which is part of our work [89]:

$$\begin{aligned} S_{FCN}(Q,\bar{Q}) =& - Q\bar{Q} + \log(1+Q) + \frac{\alpha_1}{P}\mathrm{Tr}\log\left[\beta\left(\frac{\mathbb{1}}{\beta} + \mathbf{K}^{(\mathrm{R})}_{FCN}(\bar{Q})\right)\right] + \\ &+ \frac{\alpha_1}{P}\mathbf{y}^\top\left(\frac{\mathbb{1}}{\beta} + \mathbf{K}^{(\mathrm{R})}_{FCN}(\bar{Q})\right)^{-1}\mathbf{y}\,, \end{aligned} \tag{6.26}$$

$$\begin{aligned} S_{CNN}\left(\mathbf{Q},\bar{\mathbf{Q}}\right) =& - \mathrm{Tr}\,\mathbf{Q}\bar{\mathbf{Q}} + \mathrm{Tr}\log\left(\mathbb{1}+\mathbf{Q}\right) + \frac{\alpha_c}{P}\mathrm{Tr}\log\left[\beta\left(\frac{\mathbb{1}}{\beta} + \mathbf{K}^{(\mathrm{R})}_{CNN}(\bar{\mathbf{Q}})\right)\right] + \\ &+ \frac{\alpha_c}{P}\mathbf{y}^\top\left(\frac{\mathbb{1}}{\beta} + \mathbf{K}^{(\mathrm{R})}_{CNN}(\bar{\mathbf{Q}})\right)^{-1}\mathbf{y}\,. \end{aligned} \tag{6.27}$$

Note that these actions share the same functional form, except for the two facts; the variables $\mathbf{Q}$, $\bar{\mathbf{Q}}$ in the CNN case are $\lfloor N_0/S \rfloor \times \lfloor N_0/S \rfloor$, while they are scalars for FCNs, and the definition of *kernel renormalization* is quite different since they read

$$\left[K^{(\mathrm{R})}_{FCN}(\bar{Q})\right]_{\mu\nu} \equiv \frac{\bar{Q}}{\lambda_1}K_{\mu\nu}\,, \quad \left[K^{(\mathrm{R})}_{CNN}(\bar{\mathbf{Q}})\right]_{\mu\nu} \equiv \frac{1}{\lambda_1\lfloor N_0/S\rfloor}\sum_{ij=1}^{\lfloor N_0/S\rfloor}\bar{Q}_{ij}K^{ij}_{\mu\nu}\,, \tag{6.28}$$

where $K_{\mu\nu}$ is the NNGP kernel (4.33), and $K^{ij}_{\mu\nu}$ is its CNN extension defined in (6.11). It is clear that these two architectures FCN and CNN have important differences, and one of our scope is to gain insights about these. This final section is devoted to briefly presenting what is already known empirically and explaining how our theory can capture these phenomena. Moreover, we show the results of our numerical experiments for testing and corroborating the predictions of the theory.

6.3.1 Generalization error through the predictor statistics

As was already mentioned in the introduction of this chapter, recently, the authors of [56] carried out an intensive numerical analysis of the different behavior of the generalization error for different architectures and regimes. For every network type, they compared the generalization error of a finite-width NN trained with stochastic gradient descent and the corresponding infinite-width GP, i.e. a Bayesian NN having as kernel the NNGP kernel. They found that the infinite-width FCN slightly outperforms the finite-width one. Moreover, for the same task, which is the classification of the CIFAR10 [60] dataset, the CNNs, both finite and infinite wide, outperform the FCN. In the infinite/finite comparison, they found that CNNs can outperform the corresponding NNGP, but only in the presence of weight sharing. Indeed, it seems LCNs perform equally in both regimes. Let us see how these facts can be explained in our theoretical framework.

Our theory allows us to compute the predictor statistics of a NN, and the expected value of the generalization error over a new sample $(\mathbf{x}_0, y_0)$ reads

$$\left\langle \epsilon_{\mathrm{g}}(\mathbf{x}^0, y^0) \right\rangle = (y^0 - \Gamma)^2 + \sigma^2 \tag{6.29}$$

with:

$$\Gamma = \sum_{\mu\nu=1}^{P} K_{\mu}^{(\mathrm{R})} \left(\frac{\mathbb{1}}{\beta} + \mathbf{K}^{(\mathrm{R})}(\bar{Q}) \right)^{-1}_{\mu\nu} y_{\nu}, \tag{6.30}$$

$$\sigma^2 = K_0^{(\mathrm{R})} - \sum_{\mu\nu=1}^{P} K_{\mu}^{(\mathrm{R})} \left(\frac{\mathbb{1}}{\beta} + \mathbf{K}^{(\mathrm{R})}(\bar{Q}) \right)^{-1}_{\mu\nu} K_{\nu}^{(\mathrm{R})} \,. \tag{6.31}$$

These expressions are valid both for FCNs and CNNs, considering the corresponding kernel $\mathbf{K}^{(\mathrm{R})}(\bar{Q})$. We recall K_0 and K_μ being the extended kernel functional in the presence of a new sample, and defined in Eq. (5.31). In the case of CNNs, these quantities are computed in the same way, starting from the local covariance matrix $C^{ij}_{\mu\nu}$. These terms undergo the same type of renormalization as the correspondent kernel matrices (see Table

6.1), that is:

$$\left[K^{(\mathrm{R})}_{FCN}\right]_{\mu} = \frac{\bar{Q}}{\lambda_1} K_{\mu}\,, \tag{6.32}$$

$$\left[K^{(\mathrm{R})}_{\mathrm{CNN}}\right]_{\mu} = \frac{1}{\lambda_1 \lfloor N_0/S \rfloor} \sum_{ij=1}^{\lfloor N_0/S \rfloor} \bar{Q}_{ij} K^{ij}_{\mu}\,, \tag{6.33}$$

First, let us focus on the form of the biases Γ. The $\beta \to \infty$ limit of Eq. (6.30) yields different behaviors in the two cases:

$$\Gamma_{\mathrm{FCN}} = \sum_{\mu\nu=1}^{P} K_{\mu} K^{-1}_{\mu\nu}\, y_{\nu}\,, \tag{6.34}$$

$$\Gamma_{\mathrm{CNN}} = \sum_{\mu\nu=1}^{P} \left[K^{(\mathrm{R})}_{\mathrm{CNN}}\right]_{\mu} \left(\mathbf{K}^{(\mathrm{R})}_{\mathrm{CNN}}(\bar{\mathbf{Q}})\right)^{-1}_{\mu\nu} y_{\nu}\,. \tag{6.35}$$

From these expressions, it is clear (as was discussed in [81]) that the FCN bias at finite-width is independent of $\bar{Q}$, and then it is identical to the infinite-width case (4.36) for any hidden layer size. This implies that the only difference in the generalization error between the finite and infinite width case resides in the variance term. In the zero-temperature limit and for a FCN, the variance reads

$$\sigma^2 = \frac{\bar{Q}^*}{\lambda_1}\left[K_0 - \sum_{\mu\nu} K_{\mu} K^{-1}_{\mu\nu} K_{\nu}\right]\,, \tag{6.36}$$

where $\bar{Q}^*$ is the solution of the saddle-point equation (5.29). This expression clearly states that the only difference between the variance of a finite-width FCN and the one of the predictor statistics of an NNGP is the proportional term $\bar{Q}^*/\lambda_1$ in front of the square brackets. Once evaluated on the saddle-point, $\bar{Q}^*$ is a data-dependent scalar. This implies that, once the hyperparameters (P, σ,...) are fixed, it is always possible to re-obtain the performance of any finite-width network by carefully fine-tuning the Gaussian prior λ_1 in the corresponding infinite-width kernel. Therefore, *the generalization performance of any finite-width one hidden layer FCN is bounded by the one of a suitable infinite-width kernel with optimal choice of the Gaussian prior.*

Moreover, from Eqs. (6.36) and (5.29) we see that the variance goes to zero as $1/\sqrt{\lambda_1}$ in the FCN case, while this is not obvious in CNNs. However, we can always factorize a global

$1/\lambda_1$ term (see the form the the renormalized kernel (6.33)). This means that training at large values of the Gaussian prior of the last layer should improve generalization at any finite-width since only the bias term remains as leading factor for the error. Furthermore, when $\lambda_1 \gg 1$ the analytical criterion (5.33) to predict whether a finite-width FCN will outperform its IW counterpart, is never satisfied, since the term $P^{-1}\mathbf{y}^\top\mathbf{K}^{-\mathbf{1}}\mathbf{y}$ present the term λ_1 in the numerator and so it can not be made lower than 1. These observations lead to the conclusion that **for any finite-width FCN there exists a fully-connected NNGP that outperforms it**. We have seen in Section 4.5 that in infinite-width FCNs the kernel does not evolve from the initialization, then we can conclude that feature learning is not particularly effective in networks with fully-connected layers alone at finite-width.

We stress here that the aforementioned observations are not ruling out all other possible forms of feature learning in FCN architectures: (i) Renormalization of the infinite-width kernel is not the only source of feature learning possible. Higher order kernels, irrelevant in the infinite-width limit, may play a role in feature learning, especially as long as one considers the case where the size of the dataset P roughly scales as the number of parameters $\sim O(L \times N^2)$ ($N_\ell = N\ \forall \ell$) of the fully-connected deep neural network. The recent work [90] is a notable example of how an FCN can learn a convolutional structure in special settings; (ii) Our result holds for Bayesian learning, i.e. when the weights of the networks are sampled from the canonical Gibbs ensemble. This is only obtained if training is governed by a Markov chain Monte Carlo. Practical learning algorithms with state-of-the-art optimizers may behave differently and possibly activate alternative forms of feature learning; (iii) We are limiting our analysis to the standard parametrization setting where the last layer is normalized as $1/\sqrt{N_1}$, and we cannot describe the mechanism for feature learning that may occur in the mean field regime [62] where the last layer is normalized as $1/N_1$ (we note that the framework of [69] seems more suitable to deal with this case). Nonetheless, it must be also said that the empirical evidence provided in Refs. [61, 54, 56, 59, 64] is quite against the fact that these forms of feature learning can play a significant role in improving the generalization performance of FCNs.

All the observations we have made for the FCN case, derive from the fact that the kernel renormalization with global scalar constant implies $\sum_\mu \left[K^{(\mathrm{R})}_{FCN}\right]_\mu \left[K^{(\mathrm{R})}_{FCN}\right]^{-1}_{\mu\nu} = \sum_\mu K_\mu K^{-1}_{\mu\nu}$.

This is no longer true for the CNN case, where the renormalization of the kernel is not trivial. The fact that the kernel renormalizes locally, with the application of a matrix $\bar{Q}_{ij}$, implies that the bias Γ_{CNN} depends on it and can be reduced through the optimization of this matrix. This means that, in principle **a finite-width CNN can select a suitable set of parameters, far from the initialization, such that the generalization error is lower than the one of the NNGP counterpart**, i.e. the CNN with infinitely many channels.

To test these predictions, we perform numerical simulations with $2d$ CNNs trained on CIFAR10 examples, close to the zero-temperature limit and in the bias-dominated setting where we let the Gaussian prior λ_1 be large. As shown in Fig. 6.2, the test loss of the FCN is a monotonically decreasing function of the hidden layer size N_1 and it does reach its asymptotic (best) performance already for $\alpha \sim 1$. Compatibly to local kernel renormalization, CNNs with a finite number of channels beat their infinite-width counterpart, already for large filter sizes ($M = 14, 7$), and ultimately approach the infinite-width limit performance from below. It is worth stressing that we do not expect that local kernel renormalization will lead to improved generalization performance for generic learning tasks, but only in those cases where the training patterns exhibit local spatial correlations, as it occurs in computer vision problems.

6.3.2 Differences in the similarity matrix

We have already seen that in our formalism one can compute the expected value of observables. It would be nice to extract the renormalized kernel from a trained network, however, at the moment, we have no way to directly access it in numerical experiments. One indirect experimental blueprint of the form of kernel renormalization at finite width in different architectures is given by the *similarity matrix* of the internal representations before and after training. We have already defined it in Section 6.2:

$$O_{\mu\nu} \equiv \frac{1}{N_1} \sum_{i=1}^{N_1} \sigma\left(h_i^\mu\right) \sigma\left(h_i^\nu\right) \,, \tag{6.37}$$

where N_1 denotes the number of neurons in the last layer (for CNNs we have $N_1 = N_c \lfloor N_0/S \rfloor$). We can track the effect of training by taking the difference between the

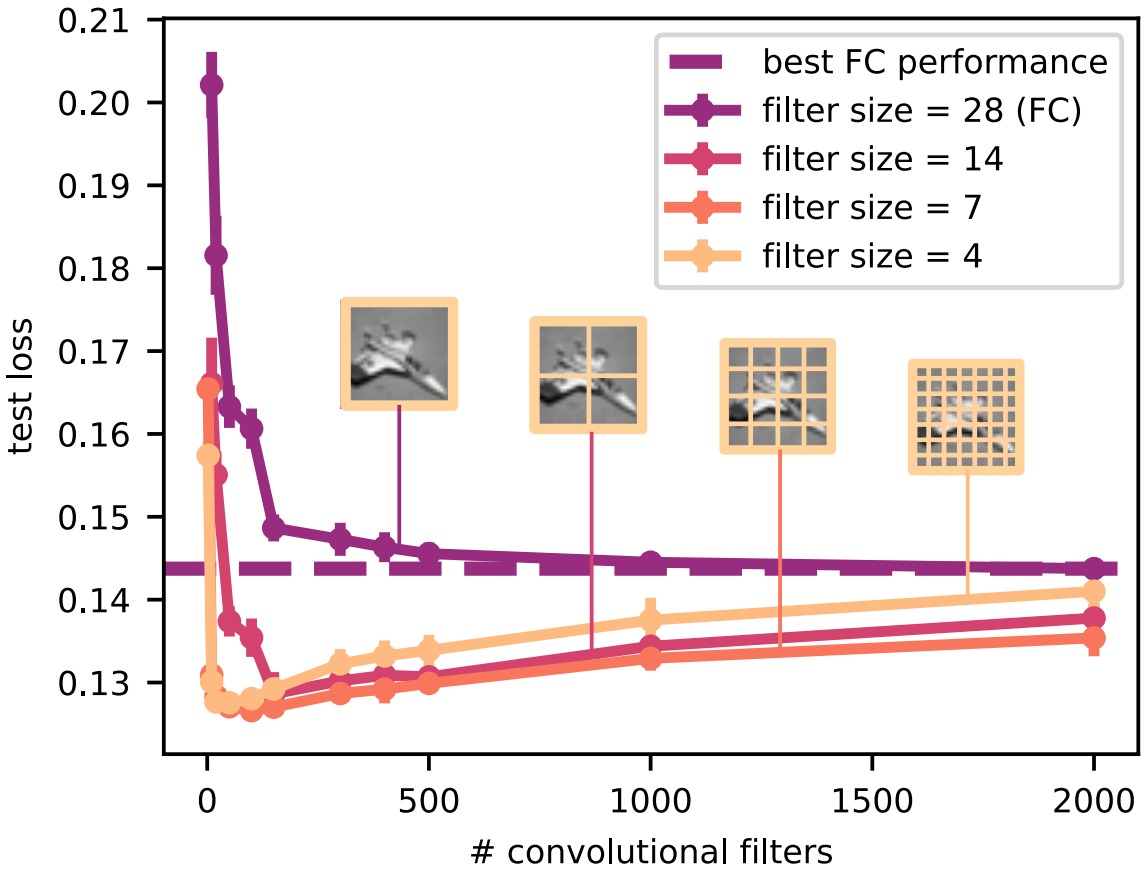

Figure 6.2: Test loss vs number of $2d$ convolutional filters for one hidden layer CNNs with different filter sizes M. All the networks are trained on the same binary regression task with $P = 500$ examples from two classes of the CIFAR10 dataset, "cars" and "planes", respectively labeled 0 and 1. Error bars represent one standard deviation. The images are gray-scaled and coarse-grained to $N_0 = 28 \times 28$. We choose the filter size M to be an integer divisor of 28, and the filters to be non-overlapping (taking stride $S = M$). A sketch of how the images are parsed by filters is reported for each network. With this choice, the FCN is retrieved setting $M = 28$. The nets are trained with a zero temperature Langevin dynamics (see Appendix A3), and with large Gaussian priors λ_1. In this bias-dominated regime, we observe that: (i) In FCNs, the best possible performance is the infinite-width one. Here the variance is monotonically decreasing with the width N_c, while the bias is a constant. Therefore, the test loss approaches its minimum monotonically from above, as predicted by our theory. (ii) Finite-width CNNs can outperform their infinite-width counterpart. The local kernel renormalization mechanism allows the CNN to optimize both bias and variance at finite width, making it even more convenient to have a small number of filters (the minimum is reached for $N_c \sim 50$). Eventually, the CNNs reach their asymptotic performance from below.

similarity matrix at initialization, which is by definition the NNGP kernel (up to the Gaussian prior λ_1), and the same quantity after training: $\Delta K_{\mu\nu} \equiv \langle O_{\mu\nu} \rangle - K_{\mu\nu}$, where the average is done over the Gibbs ensemble of the weights. In this way, we can gain insights into how the training phase affects the internal representation of the training set. We have analytically computed this observable for FCNs and CNNs, as shown in Appendix A2. The final result in the two cases reads (at zero temperature):

$$\Delta K^{\mathrm{FCN}}_{\mu\nu} = -\frac{1}{N_1}\left[K_{\mu\nu} - \frac{\lambda_1}{\bar{Q}} y^\mu y^\nu\right]; \tag{6.38}$$

$$\begin{aligned}\Delta K^{\mathrm{CNN}}_{\mu\nu} = &-\frac{1}{\lambda_1 N_c}\frac{1}{\lfloor N_0/S \rfloor^2}\sum_{ij,\lambda\rho=1}^{\lfloor N_0/S \rfloor, P} \bar{Q}_{ij} P^{ij}_{\mu\lambda\nu\rho} \\ &\times \left[\left(K^{(\mathrm{R})}_{\mathrm{CNN}}\right)^{-1}_{\lambda\rho} - \sum_{\epsilon\omega=1}^{P}\left(K^{(\mathrm{R})}_{\mathrm{CNN}}\right)^{-1}_{\lambda\epsilon}\left(K^{(\mathrm{R})}_{\mathrm{CNN}}\right)^{-1}_{\rho\omega} y^\epsilon y^\omega\right],\end{aligned} \tag{6.39}$$

where $P^{ij}_{\mu\lambda\nu\rho} \equiv \frac{1}{2}\sum_k \left[K^{ik}_{\mu\lambda}K^{kj}_{\nu\rho} + K^{ki}_{\mu\lambda}K^{jk}_{\nu\rho}\right]$. Let us now highlight a few interesting physical implications that directly follow from these formulas and can be checked with numerical experiments:

(i) The difference between the trained and untrained similarity matrix $\Delta K^{\mathrm{FCN}}_{\mu\nu}$ converges to zero in the thermodynamic limit $P, N_1 \to \infty$ at fixed $\alpha_1 = P/N_1$, as long as the terms in the square brackets are of order $o(N_1)$. This means that in the proportional regime under consideration the effect of training on the internal representations is just a finite-size correction $O(1/N_1)$ and the trained similarity matrix stays very close to its value at initialization. In particular, we expect that the empirical distribution of the elements of the matrix $\Delta K^{\mathrm{FCN}}_{\mu\nu}$ will be increasingly peaked around zero in finite-size scaling experiments where we consider increasing values of P and N_1 keeping their ratio $\alpha_1 = P/N_1$ fixed;

(ii) By choosing a learning task with labels $y^\mu = 0, 1$ and input patterns x^μ such that the matrix elements of the kernel $K_{\mu\nu}$ are on average centered in zero, the distribution of the matrix elements of the block $\Delta K^{\mathrm{FCN}}_{\mu\nu}|_{11}$ (corresponding to the subset of training patterns with label one) should drift towards zero at a rate $1/N_1$, if one performs experiments at constant α_1 while increasing P and N_1. Instead, the remaining blocks of the matrix $\Delta K^{\mathrm{FCN}}_{\mu\nu}$ should not be affected by this drift;

(iii) The difference between the trained and untrained convolutional similarity matrix $\Delta K^{\mathrm{CNN}}_{\mu\nu}$ displays a different behavior, due to local kernel renormalization. For instance, even if we restrict our analysis to the learning setting outlined in bullet (ii), the elements of the block $\Delta K^{\mathrm{CNN}}_{\mu\nu}|_{01}$ (or $\Delta K^{\mathrm{CNN}}_{\mu\nu}|_{00}$) should not identically converge to zero in the thermodynamic proportional limit. A straightforward scaling analysis shows that the contribution due to the term in the square bracket does not vanish in the proportional limit.

In order to check the aforementioned predictions we set a series of numerical experiments. For each architecture, we train a sample of one hidden layer neural networks on the same task and we collect the similarity matrix, averaged over this sample, before and after the training. We train the networks on a synthetic dataset composed of random Gaussian patterns whose labels were given by a linear teacher function, i.e. $y = \frac{1}{2}\left[1 + \mathrm{sign}(\mathbf{t} \cdot \mathbf{x})\right]$, where t is an N_0-dimensional vector with unitary entries. The numerical details of the simulations are shown in Appendix A3, but we want to stress here that these experiments are carried out using gradient descent and the so-called ADAM optimizer [91]. Interestingly, even though our theory should be valid only in the Bayesian setting, it turns out to be also predictive for a learning algorithm that does not explicitly sample from the Gibbs posterior distribution. We note that the clean signature of the differences between FCNs and CNNs ((6.38), (6.39)) derives also from our choice of working close to zero temperature (where the architectures can precisely fit the training data). To reach such a regime, we set a threshold value close to zero ($O(10^{-10})$) and we train the networks until the training loss is lower than this threshold (thanks to Adam, we have reached very often values of the training loss of $O(10^{-14})$ with a small number of epochs). Overall, the fact that the change in the internal representations of FCNs is a finite-size effect is confirmed by the experiments. On the other hand, the same numerical simulations with CNNs clearly identify a genuine change in the internal representations that occurs also in the thermodynamic proportional limit. In the following, we will show our results.

6.3.2.1 FCN at $\alpha_1 = 1$

In Fig. 6.3, we show the numerical experiments using an FCN as architecture, with a fixed value of $\alpha_1 = P/N_1 = 1$ and increasing value of the number P of elements in the training set and the number N_1 of neurons in the hidden layer. In particular, we train

architectures with a single hidden fully-connected layer with tanh activation (see details in Appendix A3) on a learning task with random Gaussian inputs and zero/one labels given by a linear teacher. In the picture, we show the distribution of the elements of the difference similarity matrix, in two subspaces: the one where both the labels are $y^\mu = y^\nu = 1$, indicated as subspace "11", and the one with discordant labels, the subspace "01". Our experiments show a good agreement with the predictions derived from Eq. (6.38). In the subspace "11", we observe that the distribution of the matrix elements shrinks while drifting towards zero at a rate $\sim 1/N_1$. Instead, in the subspace "01", we find that the distributions are peaked around zero, and their variance rapidly converges to zero as N_1 grows, in particular, we observe it goes to zero as $\sim N_1^{-2}$ (the results about the scaling are shown as linear fit in log-log scale in panel (c)). Overall, our numerical analysis suggests that all the differences before and after training in the similarity matrix are due to finite-size effects, and they should disappear in the thermodynamic limit. This fact suggests that in FCNs the effects of training are not visible in the internal representation of the network, even in the finite-width limit (we recall that in the IW limit, the weights do not evolve from the initialization and so the similarity matrix).

6.3.2.2 CNN at $\alpha_1 = 1$

In Fig. 6.4 we repeat the previous analysis for the CNNs case, using the same training setting, fixing the ratio between the number of patterns and the number of channel $\alpha_c = P/N_c = 1$ and using $1d$ convolutional filters. Again, we show the distribution of the elements in the subspaces "11" and "01" and we study their scaling with the size. The main difference with respect to the FCN case can be deduced from Eq. (6.39). Indeed, the label term $y^\epsilon y^\omega$ enters in the expression of $\Delta K_{\mu\nu}$ in a non-trivial way, and also in the subspace "00" the matrix depends on the mixing between labels and the renormalized kernel. Moreover, due to the form of the local kernel renormalization, the renormalized terms remain explicit. These facts suggest that in CNNs the effect of training could be visible in the difference between the similarity matrix before and after the training. Indeed, we observe that, in both the subspaces, the variance of the empirical distribution of the matrix elements does not shrink to zero as N_c grows and α_c is kept fixed (as can be seen from the fit in the upper plot in the panel (c)). Interestingly, the drift towards zero in the subspace "11" seems to be present, with the same scaling law of the FCN case (see

the lower plot in panel (c)), although these peaks are small with respect to the one in the FCN case for at least one order of magnitude. Overall, our experiments make evident that the distribution of the elements of $\Delta K^{CNN}_{\mu\nu}$ does not disappear in the thermodynamic limit, suggesting the training phase has some effects on the internal representation.

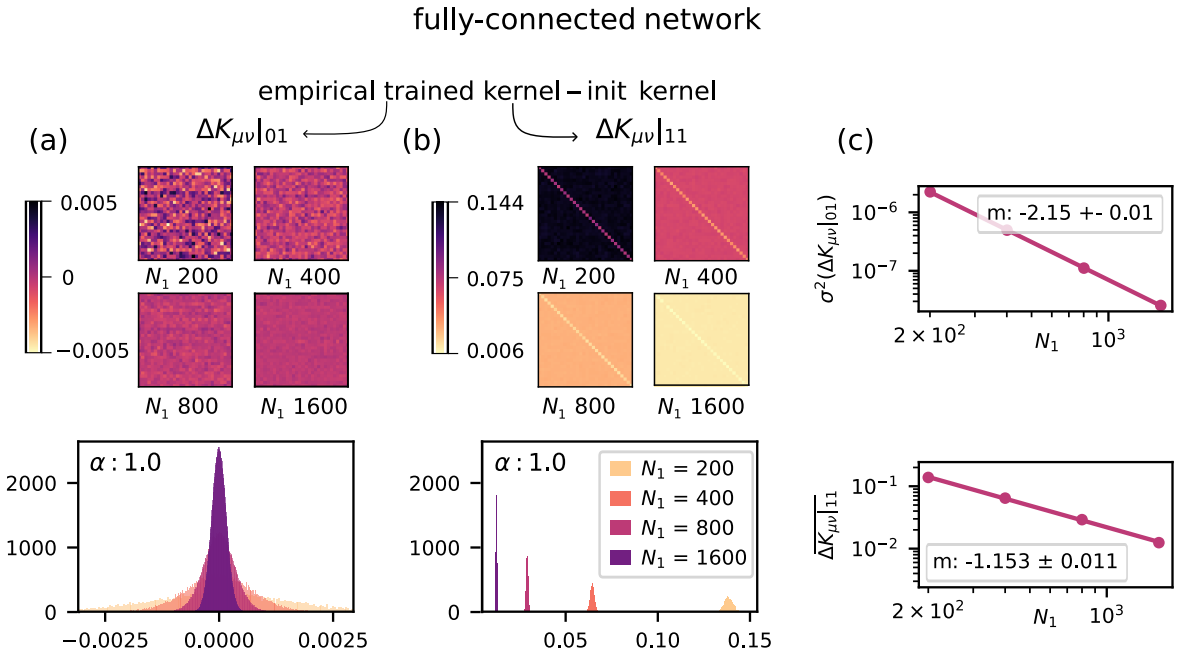

Figure 6.3: Predicting the effect of global kernel renormalization on the internal representations of FCNs. A finite-size scaling analysis. Here we show the results of numerical experiments performed to check the predictions of Eq. (6.38) for the difference $\Delta K_{\mu\nu}$ between the similarity matrix after and before training. We perform a finite-size scaling analysis by keeping $\alpha_1 = 1$ and choosing increasing values of P, N_1. In panel (a) we show the zero-one block $\Delta K^{\text{FCN}}_{\mu\nu}|_{01}$. At the top, there is a subsample of this submatrix for every value of N_1 tested (note that $P = N_1$ for $\alpha_1 = 1$), while the distribution of the elements is represented in the histogram on the bottom. It is clear that this distribution becomes more peaked around the value 0 when increasing the size, which suggests that in the thermodynamic limit, the similarity matrices after and before training become more and more similar. The shrinking is also evident from the upper plot in panel (c), where the variance of these distributions is shown against the size, and it goes to zero as $\sim N_1^{-2}$ (m reported is the value founded with a linear fit). Analogously, in panel (b) we show the same analysis for the one-one block $\Delta K^{\text{FCN}}_{\mu\nu}|_{11}$. Here, it is evident that the distributions (bottom) shrink while drifting towards zero. From the bottom side of panel (c), it is clear that they drift at a rate $\sim 1/N_1$ (see the m reported in the plot), as expected from the theory. Overall, this analysis represents a clear indication that the change in the after-training internal representations in one hidden layer FCNs is a finite-size effect.

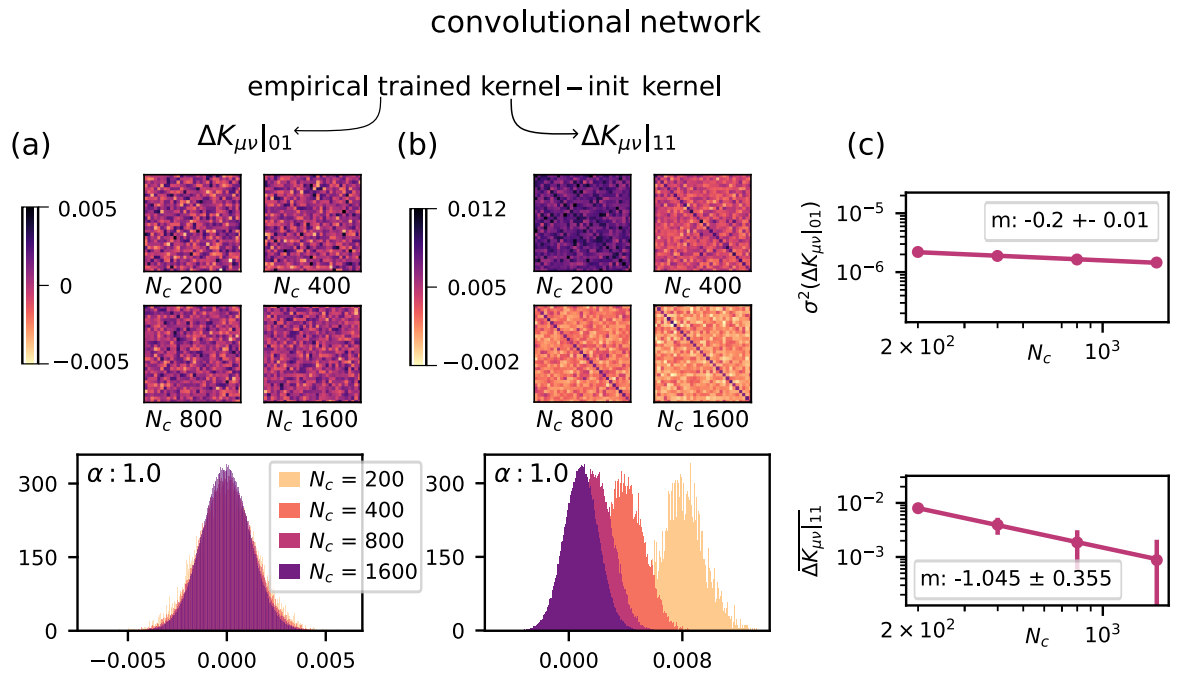

Figure 6.4: Predicting the effect of global and local kernel renormalization on the internal representations of CNNs. A finite-size scaling analysis. Analogously to the previous picture, here we show the results of numerical experiments on the CNNs, to check the predictions of Eq. (6.39). Again, in panels (a) and (b) we show the distribution (bottom) of the matrix elements in the subspaces "01" and "11" respectively, and an array plot representing a subsample in these subspaces (up). It is clear that in this case, the distributions do not shrink as the size N_c grows, as we can see also from the linear fit of the variance in the upper part of the panel (c). Surprisingly, there is a drift towards zero in the "11" subspace, although the distributions are peaked around small values with respect to the one found in FCNs. Overall, it is evident that these distributions do not disappear in the thermodynamic limit and the effect of training should remain visible in the internal representation.

6.4 The predictive power of the theory at non-zero temperature

This ending section is devoted to introducing our ongoing work, in collaboration with the Lattice Group here in Parma (Dott. P. Baglioni and Prof. F. Di Renzo). This project aims to perform intensive numerical simulations to test our theory on non-zero temperature regimes. Our strategy is to train a Fully Connected Network on the CIFAR10 and MNIST datasets using a discretized Langevin dynamics, as it is explained in Appendices, Eq. (A3.2). This is done to sample configurations of the weights from the correct Gibbs ensemble. In this way, we can perform a standard Monte Carlo routine that allows us

to extrapolate average values of observables, in particular the generalization error, at non-zero temperatures. We are exploring the predictive power of the theory for an FCN as a function of the parameters: the temperature $1/\beta$, the magnitude of the Gaussian priors λ_1, λ_0, the size of the hidden layer N_1 and the size of the training set P. In the following, we present our preliminary results on the CIFAR10 dataset. The lines representing the theoretical expectations are computed by numerically solving the saddle-point equations.

In Fig. 6.5, we extrapolate the generalization error dependency on N_1 at fixed $\alpha = P/N_1 = 0.25$, for three temperatures $T = 1.0$, 0.5, and 0.01 (fixed Gaussian priors $\lambda_0 = 1$ and $\lambda_1 = 1$). For every point simulated, we take a new subsample of the entire dataset, which leads to small fluctuations between points at different N_1.

In Fig 6.6, we test our predictions about the dependency of the generalization to the Gaussian prior λ_1, at fixed $\lambda_0 = 1.0$ and $P = 1000$, for two values of N_1.

Finally, in Fig. 6.7 we present our experiments when keeping fixed all the parameters ($P = 1000$, $\lambda_0 = 1$, $\lambda_1 = 1$) except the hidden size N_1, at different T. For the CIFAR10 case, the best generalization error is found for non-zero temperatures. Indeed, our theory predicts a crossing point of the generalization error curves between $T = 0.1$ and $T = 0.03$.

From all these preliminary results, it is clear that our theory is in good agreement with the numerical experiments. Quite remarkably, it seems to capture also the small fluctuations due to changing examples in the dataset. We observe little deviations at small values of N_1, that quickly disappear for larger sizes, suggesting they are caused by finite size effects. Our ongoing research will be focused on studying these deviations in detail. Moreover, we want also to explore other combinations of the hyperparameters and then we are going to extrapolate with more accuracy another observable, already discussed, the similarity matrix. Finally, we will investigate the predictive power of the effective theory for a convolutional neural network.

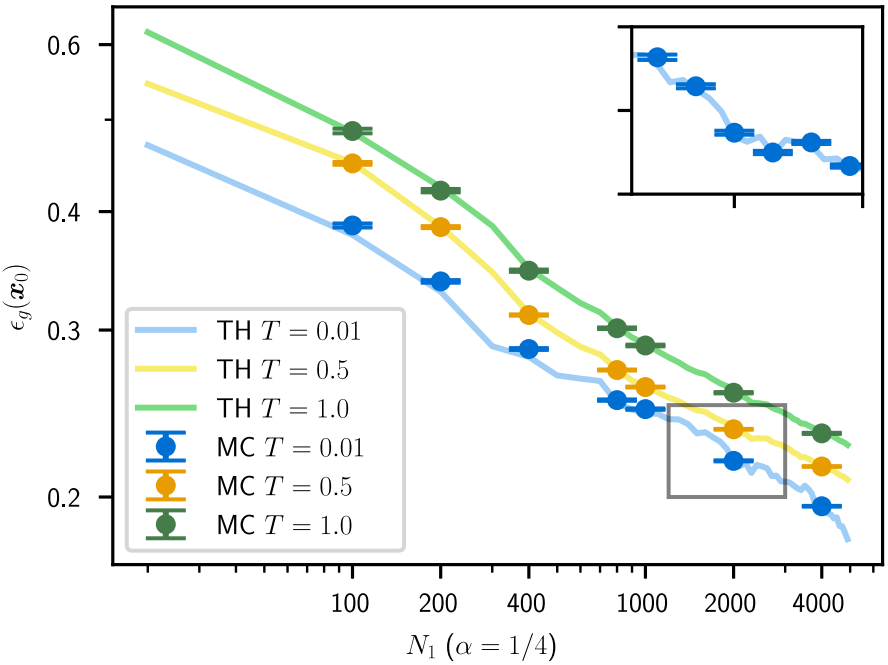

Figure 6.5: Predicting the generalization error dependency on N_1 at $\alpha = 0.25$ fixed. We show the predictive power of our theory (TH). The lines are extrapolated solving the saddle-point equations, while the points are the result of intensive Monte Carlo (MC) simulations. The inset on the upper right shows the good agreement between the experiments and the theory, even in the presence of small fluctuations due to the choice of different examples in the dataset.

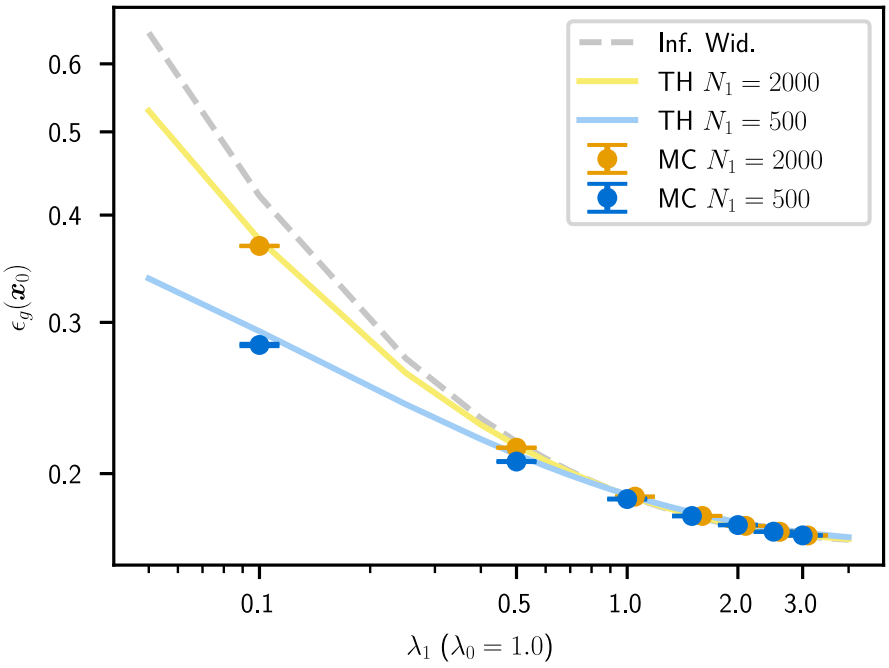

Figure 6.6: Predicting the generalization error dependency on λ_1. Here we test the prediction of our theory about the dependency of ϵ_g on the Gaussian prior of the last layer λ_1. The dotted line represents the generalization error of the correspondent infinite-width network. The discrepancies quickly disappear considering larger widths of the hidden layer, suggesting they are due to finite-size effects.

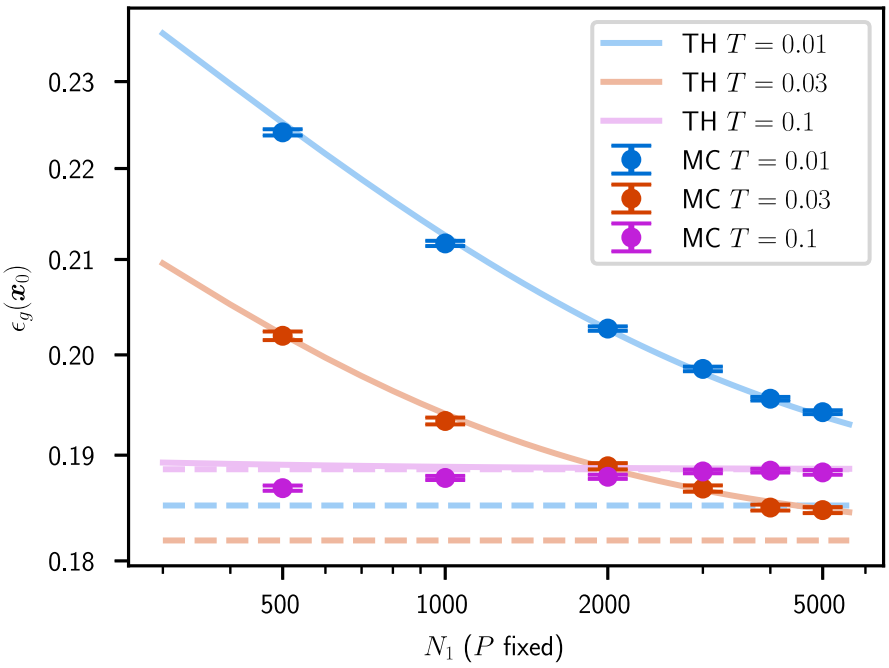

Figure 6.7: Predicting the generalization error dependency on N_1 at fixed $P = 1000$. In this plot, we show that our theory can predict also particular behaviors of the generalization error, as the crossing point between different temperatures. Indeed, for the dataset considered (CIFAR10), it seems the best generalization error can be found at a non-zero temperature and our experiments are in good agreement with this observation. Again, it seems the deviations are due to finite-size effects. The dotted lines represent the generalization error of the correspondent infinite-width networks.

References

[1] A. Campa, T. Dauxois, D. Fanelli, and S. Ruffo. *Physics of Long-Range Interacting Systems*. Oxford University PressOxford, 1 edition, August 2014.

[2] Nicolò Defenu, Tobias Donner, Tommaso Macrì, Guido Pagano, Stefano Ruffo, and Andrea Trombettoni. Long-range interacting quantum systems, 2021. arXiv.2109.01063.

[3] R Burioni and D Cassi. Random walks on graphs: ideas, techniques and results. *Journal of Physics A: Mathematical and General*, 38(8):R45, feb 2005.

[4] Davide Cassi. Phase transitions and random walks on graphs: A generalization of the mermin-wagner theorem to disordered lattices, fractals, and other discrete structures. *Physical Review Letters*, 68(24):3631–3634, June 1992.

[5] Raffaella Burioni, Davide Cassi, and Alessandro Vezzani. Inverse mermin-wagner theorem for classical spin models on graphs. *Physical Review E*, 60(2):1500–1502, August 1999.

[6] Raffaella Burioni and Davide Cassi. Geometrical Universality in Vibrational Dynamics. *Modern Physics Letters B*, 11(25):1095–1101, October 1997.

[7] Michael E. Fisher, Shang-keng Ma, and B. G. Nickel. Critical Exponents for Long-Range Interactions. *Physical Review Letters*, 29(14):917–920, October 1972.

[8] Kenneth G. Wilson and Michael E. Fisher. Critical exponents in 3.99 dimensions. *Phys. Rev. Lett.*, 28:240–243, Jan 1972.

[9] J.L. Cardy. One-dimensional models with 1/r2 interactions. *J. Phys. A: Math. Gen.*, 14(6):1407, jun 1981.

[10] J. Sak. Recursion Relations and Fixed Points for Ferromagnets with Long-Range Interactions. *Physical Review B*, 8(1):281–285, July 1973.

[11] N. D. Mermin and H. Wagner. Absence of ferromagnetism or antiferromagnetism in one- or two-dimensional isotropic heisenberg models. *Physical Review Letters*, 17(22):1133–1136, November 1966.

[12] V. L. Berezinsky. Destruction of long range order in one-dimensional and two-dimensional systems having a continuous symmetry group. I. Classical systems. *Sov. Phys. JETP*, 32:493–500, 1971.

[13] J M Kosterlit. Ordering, metastability and phase transitions in two-dimensional systems.

[14] Guido Giachetti, Nicolò Defenu, Stefano Ruffo, and Andrea Trombettoni. Berezinskii-Kosterlitz-Thouless Phase Transitions with Long-Range Couplings. *Physical Review Letters*, 127(15):156801, October 2021.

[15] Guido Giachetti, Andrea Trombettoni, Stefano Ruffo, and Nicolò Defenu. Berezinskii-kosterlitz-thouless transitions in classical and quantum long-range systems. *Physical Review B*, 106(1), July 2022.

[16] Ulli Wolff. Collective Monte Carlo Updating for Spin Systems. *Physical Review Letters*, 62(4):361–364, January 1989.

[17] Erik Luijten and Henk W.J. Blöte. Monte carlo method for spin models with long-range interactions. *International Journal of Modern Physics C*, 06(03):359–370, June 1995.

[18] Erik Luijten and Henk W. J. Blöte. Finite-size scaling and universality above the upper critical dimensionality. *Phys. Rev. Lett.*, 76:1557–1561, Mar 1996.

[19] Erik Luijten and Henk W. J. Blöte. Classical critical behavior of spin models with long-range interactions. *Physical Review B*, 56(14):8945–8958, October 1997.

[20] Erik Luijten and Henk W. J. Blöte. Boundary between Long-Range and Short-Range Critical Behavior in Systems with Algebraic Interactions. *Physical Review Letters*, 89(2):025703, June 2002.

[21] Kouki Fukui and Synge Todo. Order-N cluster Monte Carlo method for spin systems with long-range interactions. *Journal of Computational Physics*, 228(7):2629–2642, April 2009.

[22] Maria Chiara Angelini, Giorgio Parisi, and Federico Ricci-Tersenghi. Relations between short-range and long-range Ising models. *Physical Review E*, 89(6):062120, June 2014.

[23] G. Gori, M. Michelangeli, N. Defenu, and A. Trombettoni. One-dimensional long-range percolation: A numerical study. *Phys. Rev. E*, 96:012108, Jul 2017.

[24] Henrik Christiansen, Suman Majumder, and Wolfhard Janke. Phase ordering kinetics of the long-range Ising model. *Physical Review E*, 99(1):011301, January 2019.

[25] Henrik Christiansen, Suman Majumder, Malte Henkel, and Wolfhard Janke. Aging in the Long-Range Ising Model. *Physical Review Letters*, 125(18):180601, October 2020.

[26] Ramgopal Agrawal, Federico Corberi, Eugenio Lippiello, Paolo Politi, and Sanjay Puri. Kinetics of the two-dimensional long-range ising model at low temperatures. *Phys. Rev. E*, 103:012108, Jan 2021.

[27] Ramgopal Agrawal, Federico Corberi, Ferdinando Insalata, and Sanjay Puri. Asymptotic states of ising ferromagnets with long-range interactions. *Phys. Rev. E*, 105:034131, Mar 2022.

[28] L. Leuzzi, G. Parisi, F. Ricci-Tersenghi, and J. J. Ruiz-Lorenzo. Dilute One-Dimensional Spin Glasses with Power Law Decaying Interactions. *Physical Review Letters*, 101(10):107203, September 2008.

[29] L. Leuzzi, G. Parisi, F. Ricci-Tersenghi, and J. J. Ruiz-Lorenzo. Ising spin-glass transition in a magnetic field outside the limit of validity of mean-field theory. *Phys. Rev. Lett.*, 103:267201, Dec 2009.

[30] Helmut G. Katzgraber, Derek Larson, and A. P. Young. Study of the de almeida–thouless line using power-law diluted one-dimensional ising spin glasses. *Phys. Rev. Lett.*, 102:177205, Apr 2009.

[31] Miguel Ibáñez Berganza and Luca Leuzzi. Critical behavior of the X Y model in complex topologies. *Physical Review B*, 88(14):144104, October 2013.

[32] Ana P. Millán, Giacomo Gori, Federico Battiston, Tilman Enss, and Nicolò Defenu. Complex networks with tuneable spectral dimension as a universality playground. *Phys. Rev. Res.*, 3:023015, Apr 2021.

[33] Giacomo Bighin, Tilman Enss, and Nicolò Defenu. Universal scaling in fractional dimension, November 2022. arXiv:2211.13302 [cond-mat].

[34] Fabiana Cescatti, Miguel Ibáñez Berganza, Alessandro Vezzani, and Raffaella Burioni. Analysis of the low-temperature phase in the two-dimensional long-range diluted xy model. *Phys. Rev. B*, 100:054203, Aug 2019.

[35] Arkadiusz Jędrzejewski, Anna Chmiel, and Katarzyna Sznajd-Weron. Oscillating hysteresis in the q-neighbor ising model. *Phys. Rev. E*, 92:052105, Nov 2015.

[36] A. Chmiel, T. Gradowski, and A. Krawiecki. q-Neighbor Ising model on random networks. *International Journal of Modern Physics C*, 29(06):1850041, June 2018.

[37] Nicholas Metropolis and S. Ulam. The Monte Carlo Method. *Journal of the American Statistical Association*, 44(247):335–341, September 1949.

[38] Roy J. Glauber. Time-Dependent Statistics of the Ising Model. *Journal of Mathematical Physics*, 4(2):294–307, February 1963.

[39] Jong-Min Park and Jae Dong Noh. Tricritical behavior of nonequilibrium Ising spins in fluctuating environments. *Physical Review E*, 95(4):042106, April 2017.

[40] Sage Alexander and Raymond Orbach. Density of states on fractals: fractons. *J. Physique Lett.*, 43:625–631, 09 1982.

[41] A. Sokal. Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms. In *Functional Integration*, volume 361 of *NATO ASI Series*, pages 131–192. Springer US, 1997.

[42] Alan M. Ferrenberg and Robert H. Swendsen. New monte carlo technique for studying phase transitions. *Physical Review Letters*, 61(23):2635–2638, December 1988.

[43] Marco Picco. Critical behavior of the Ising model with long range interactions, July 2012. arXiv:1207.1018 [cond-mat, physics:hep-th].

[44] Martin Hasenbusch. Dynamic critical exponent z of the three-dimensional ising universality class: Monte carlo simulations of the improved blume-capel model. *Phys. Rev. E*, 101:022126, Feb 2020.

[45] L.Ts. Adzhemyan, D.A. Evdokimov, M. Hnatič, E.V. Ivanova, M.V. Kompaniets, A. Kudlis, and D.V. Zakharov. The dynamic critical exponent z for 2d and 3d ising models from five-loop ϵ expansion. *Physics Letters A*, 425:127870, February 2022.

[46] Riccardo Aiudi, Raffaella Burioni, and Alessandro Vezzani. Critical dynamics of long-range models on dynamical lévy lattices. *Physical Review B*, 107(22), June 2023.

[47] Martin Hasenbusch. The binder cumulant at the kosterlitz–thouless transition. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(08):P08003, August 2008.

[48] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, April 1980.

[49] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[50] Michael A. Nielsen. Neural networks and deep learning, 2018.

[51] George Kimeldorf and Grace Wahba. Some results on tchebycheffian spline functions. *Journal of mathematical analysis and applications*, 33(1):82–95, 1971.

[52] Radford M. Neal. *Priors for Infinite Networks*, pages 29–53. Springer New York, New York, NY, 1996.

[53] Christopher K. I. Williams. Computing with infinite networks. In *Proceedings of the 9th International Conference on Neural Information Processing Systems*, NIPS'96, page 295–301, Cambridge, MA, USA, 1996. MIT Press.

[54] Jaehoon Lee, Jascha Sohl-dickstein, Jeffrey Pennington, Roman Novak, Sam Schoenholz, and Yasaman Bahri. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

[55] Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, 2019.

[56] Roman Novak, Lechao Xiao, Yasaman Bahri, Jaehoon Lee, Greg Yang, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-dickstein. Bayesian deep convolutional networks with many channels are gaussian processes. In *International Conference on Learning Representations*, 2019.

[57] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[58] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[59] Jaehoon Lee, Samuel Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein. Finite versus infinite neural networks: an empirical study. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15156–15172. Curran Associates, Inc., 2020.

[60] Alex Krizhevsky. Learning multiple layers of features from tiny images. pages 32–33, 2009.

[61] Alexander Atanasov, Blake Bordelon, Sabarish Sainathan, and Cengiz Pehlevan. The

onset of variance-limited behavior for networks in the lazy and rich regimes. *arXiv preprint arXiv:2212.12147*, 2022.

[62] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[63] Mario Geiger, Leonardo Petrini, and Matthieu Wyart. Landscape and training regimes in deep learning. *Physics Reports*, 924:1–18, 2021. Landscape and training regimes in deep learning.

[64] Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(11):113301, nov 2020.

[65] Daniel A. Roberts, Sho Yaida, and Boris Hanin. *The Principles of Deep Learning Theory*. Cambridge University Press, 2022. https://deeplearningtheory.com.

[66] Boris Hanin. Random fully connected neural networks as perturbatively solvable hierarchies, 2023.

[67] Boris Hanin. Correlation functions in random fully connected neural networks at finite width. *arXiv preprint arXiv:2204.01058*, 2022.

[68] Gadi Naveh and Zohar Ringel. A self consistent theory of gaussian processes captures feature learning effects in finite cnns. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 21352–21364. Curran Associates, Inc., 2021.

[69] Inbar Seroussi, Gadi Naveh, and Zohar Ringel. Separation of scales and a thermodynamic description of feature learning in some cnns. *Nature Communications*, 14(1):908, 02 2023.

[70] Qianyi Li and Haim Sompolinsky. Statistical mechanics of deep linear neural networks: The backpropagating kernel renormalization. *Phys. Rev. X*, 11:031059, Sep 2021.

[71] Qianyi Li and Haim Sompolinsky. Globally gated deep linear networks. *arXiv preprint arXiv:2210.17449*, 2022.

[72] Boris Hanin and Alexander Zlokapa. Bayesian interpolation with deep linear networks. *Proceedings of the National Academy of Sciences*, 120(23):e2301345120, 2023.

[73] Francesco Cagnetta, Alessandro Favero, and Matthieu Wyart. What can be learnt with wide convolutional neural networks?, 2023.

[74] Alessandro Favero, Francesco Cagnetta, and Matthieu Wyart. Locality defeats the curse of dimensionality in convolutional teacher-student scenarios. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[75] Leonardo Petrini, Francesco Cagnetta, Eric Vanden-Eijnden, and Matthieu Wyart. Learning sparse features can lead to overfitting in neural networks. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[76] Greg Yang and Edward J. Hu. Feature learning in infinite-width neural networks, 2020.

[77] Greg Yang. Tensor programs i: Wide feedforward or recurrent neural networks of any architecture are gaussian processes, 2019.

[78] Greg Yang. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation, 2019.

[79] Greg Yang. Tensor programs ii: Neural tangent kernel for any architecture, 2020.

[80] Greg Yang. Tensor programs iii: Neural matrix laws, 2020.

[81] R. Pacelli, S. Ariosto, M. Pastore, F. Ginelli, M. Gherardi, and P. Rotondo. A statistical mechanics framework for bayesian deep neural networks beyond the infinite-width limit, 2022.

[82] Péter Breuer and Péter Major. Central limit theorems for non-linear functionals of gaussian fields. *Journal of Multivariate Analysis*, 13(3):425–441, 1983.

[83] Sebastian Goldt, Bruno Loureiro, Galen Reeves, Florent Krzakala, Marc Mézard, and Lenka Zdeborová. The gaussian equivalence of generative models for learning with shallow neural networks. *arXiv preprint arXiv:2006.14709*, 2020.

[84] Federica Gerace, Bruno Loureiro, Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Generalisation error in learning with random features and the hidden manifold model. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124013, dec 2021.

[85] Bruno Loureiro, Cedric Gerbelot, Hugo Cui, Sebastian Goldt, Florent Krzakala, Marc Mezard, and Lenka Zdeborová. Learning curves of generic features maps for realistic datasets with a teacher-student model. *Advances in Neural Information Processing Systems*, 34, 2021.

[86] Ivan Nourdin, Giovanni Peccati, and Mark Podolskij. Quantitative Breuer-Major theorems, 2010.

[87] Jean-Marc Bardet and Donatas Surgailis. Moment bounds and central limit theorems for gaussian subordinated arrays. *Journal of Multivariate Analysis*, 114:457–473, February 2013.

[88] Brendan D. Tracey and David Wolpert. Upgrading from gaussian processes to student's-t processes. In *2018 AIAA Non-Deterministic Approaches Conference*. American Institute of Aeronautics and Astronautics, jan 2018.

[89] R. Aiudi, R. Pacelli, A. Vezzani, R. Burioni, and P. Rotondo. Local kernel renormalization as a mechanism for feature learning in overparametrized convolutional neural networks, 2023.

[90] Alessandro Ingrosso and Sebastian Goldt. Data-driven emergence of convolutional structure in neural networks. *Proceedings of the National Academy of Sciences*, 119(40):e2201854119, 2022.

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

[92] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

Appendix

A1 LCN effective action derivation

Here we present the explicit computation of the effective action for one hidden layer LCNs. For simplicity, we consider one-dimensional local interaction, but the calculation can be generalized to a d-dimensional setting. For LCNs, the pre-activations of the hidden layer are given by:

$$h_i^a(x) = \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} \frac{W_{im}^a x_{Si+m}}{\sqrt{M}}\,, \tag{A1.1}$$

where M indicates the size of the filter and S denotes the value of the stride. The index $a = 1 \dots N_c$ runs over the filters. From the pre-activations, we can define the output of the LCN as:

$$f_{\text{LCN}}(x) = \frac{1}{\sqrt{N_c \lfloor N_0/S \rfloor}} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sum_{a=1}^{N_c} v_i^a \sigma\left(h_i^a\right). \tag{A1.2}$$

We recall the notation for the total number of weights in the last layer $N_1 = N_c \lfloor N_0/S \rfloor$. Note that in the special case $S = N_0$ we recover the FCN architecture.

Our objective is to construct the data-dependent partition function for this learning problem in the proportional limit:

$$Z = \int \prod_{i,a} dv_i^a \prod_{i,m,a} dW_{im}^a \, \exp\Big\{ -\frac{\lambda_1}{2}\|v\|^2 - \frac{\lambda_0}{2}\|W\|^2 - \frac{\beta}{2}\sum_{\mu=1}^{P}\left[y^\mu - f_{\text{LCN}}(x^\mu)\right]^2 \Big\}. \tag{A1.3}$$

Here, we employ the same notation of the main text, where λ_0 and λ_1 are respectively the Gaussian priors of the hidden and last layer. We introduce two sets of Dirac deltas, corresponding to the pre-activations of the hidden layer and the output of the network. We denote these new degrees of freedom as h_{ia}^μ and s^μ respectively:

$$\prod_{i,a,\mu} \delta\Big(h_{ia}^\mu - \frac{1}{\sqrt{M}} \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\lfloor \frac{M}{2} \rfloor} W_{im}^a x_{Si+m}^\mu\Big)\,, \qquad \prod_{\mu} \delta\Big(s^\mu - \frac{1}{\sqrt{N_1}} \sum_{i,a=1}^{\lfloor \frac{N_0}{S} \rfloor, N_c} v_i^a \sigma(h_{ia}^\mu)\Big)\,. \tag{A1.4}$$

Expressing the deltas directly in their standard Fourier representation, we have:

$$Z = \int \prod_{i,a,\mu} dh^{\mu}_{ia} d\bar{h}^{\mu}_{ia} \prod_{\mu} ds^{\mu} d\bar{s}^{\mu} e^{-\frac{\beta}{2}\sum_{\mu}(y^{\mu}-s^{\mu})^2 + i\sum_{\mu} s^{\mu}\bar{s}^{\mu} + i\sum_{\mu,i} h^{\mu}_{ia}\bar{h}^{\mu}_{ia}} \times \left\langle e^{-i\sum_{\mu}\bar{s}^{\mu}\left(\frac{1}{\sqrt{N_1}}\sum_i v^a_i \sigma(h^{\mu}_{ia})\right)} \right\rangle_v \left\langle e^{-i\sum_{\mu,i,a}\bar{h}^{\mu}_{ia}\left(\frac{1}{\sqrt{M}}\sum_m W^a_{im} x^{\mu}_{Si+m}\right)} \right\rangle_W . \tag{A1.5}$$

where the average is over the distribution of the weights. Performing these integrals yield a simple expression for the quantities in brackets. Respectively:

$$\prod_{i,a} \exp\left\{ - \frac{\left[\sum_{\mu=1}^{P} \bar{s}^{\mu}\sigma(h^{\mu}_{ia})\right]^2}{2\lambda_1 N_1} \right\}, \qquad \prod_{i,a} \exp\left\{ - \frac{\sum_{\mu,\nu=1}^{P} \bar{h}^{\mu}_{ia} C^{ii}_{\mu\nu} \bar{h}^{\nu}_{ia}}{2} \right\}, \tag{A1.6}$$

where the matrix $C^{ii}_{\mu\nu}$ is the diagonal part of the local covariance matrix defined in Eq. (6.7), i.e.

$$C^{ii}_{\mu\nu} = \frac{1}{\lambda_0 M} \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} x^{\mu}_{Si+m} x^{\nu}_{Si+m}. \tag{A1.7}$$

This quantity provides information on the self-correlation of the local patch in position Si for any pair of training patterns μ, ν in the dataset. These steps allow to factorize the integrals on the h-$\bar{h}$ variables over the patch indices i and channel one a, leading to the following equation for the partition function

$$Z = \int \prod_{\mu} ds^{\mu} d\bar{s}^{\mu} e^{-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^{\mu}-s^{\mu})^2 + i\sum_{\mu=1}^{P} s^{\mu}\bar{s}^{\mu}} \times \left\{ \int \prod_i d^P h_i \, d^P \bar{h}_i \, e^{-\frac{1}{2}\sum_{\mu\nu i}\bar{h}^{\mu}_i C^{ii}_{\mu\nu}\bar{h}^{\nu}_i + i\sum_{\mu i} h^{\mu}_i \bar{h}^{\mu}_i - \frac{1}{2\lambda_1 N_1}\sum_i [\sum_{\mu=1}^{P}\bar{s}^{\mu}\sigma(h^{\mu}_i)]^2} \right\}^{N_c} . \tag{A1.8}$$

After the integration over the $\bar{h}$-variables, the quantity in curly brackets becomes:

$$\prod_i \int d^P h_i \, P^i_1(\{h^{\mu}\}) e^{-\frac{1}{2\lambda_1 N_1}[\sum_{\mu=1}^{P}\bar{s}^{\mu}\sigma(h^{\mu}_i)]^2}, \tag{A1.9}$$

where the probability distribution $P^{(i)}_1$ is given by:

$$P^{(i)}_1(\{h^{\mu}\}) \equiv \frac{e^{-\frac{1}{2}h_i^{\top}(C^{ii})^{-1}h_i}}{\sqrt{(2\pi)^P \det C^{ii}}}, \tag{A1.10}$$

To proceed in the calculation, we introduce a new set of variables q_i through Dirac's δ identities. In this way Eq. (A1.9) becomes:

$$\prod_i \int dq_i e^{-\frac{1}{2}q_i^2} P(q_i)\,, \qquad P(q_i) = \int d^P h_i P_1^i(\{h^\mu\}) \delta\Big(q_i - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_{\mu=1}^{P} \bar{s}^\mu \sigma(h_i^\mu)\Big) \tag{A1.11}$$

Similarly to the FCN and CNN case, we perform a Gaussian approximation on the limiting distribution of $P(q_i)$, heuristically justified by the Breuer-Major theorem [82]:

$$P(q_i) \to \mathcal{N}_{q_i}(0, Q_i), \tag{A1.12}$$

where the variances are:

$$Q_i(\bar{s}, C^i) = \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^\mu \Big[\int d^P h P_1^i(\{h^\mu\}) \sigma(h^\mu)\sigma(h^\nu) \Big] \bar{s}^\nu \equiv \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^\mu K^i_{\mu\nu} \bar{s}^\nu\,, \tag{A1.13}$$

here $K^i_{\mu\nu} \equiv K^{ii}_{\mu\nu}$ is a compact notation for the diagonal part (in the spatial indices) of the kernel matrix defined in (6.11). After performing the (Gaussian) integrals in the q_i variables, we have:

$$\Big\{ \prod_i (1 + Q_i(\bar{s}, C^i))^{-\frac{1}{2}} \Big\}^{N_c} = e^{-\frac{N_c}{2} \sum_i \log\left(1 + Q_i(\bar{s}, C^i)\right)}. \tag{A1.14}$$

We insert a new set of deltas to handle the $\bar{s}$ dependency in $Q_i(\bar{s}, C^i)$):

$$1 = \int dQ_i\, \delta\Big(Q_i - \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^\mu K^i_{\mu\nu} \bar{s}^\nu\Big). \tag{A1.15}$$

In this way, we can perform the Gaussian integration on the $\bar{s}$ variable, after a rescaling

$\bar{Q} \to -i\frac{N_c}{2}\bar{Q}$, obtaining:

$$Z = \int \prod_i dQ_i d\bar{Q}_i \prod_\mu ds^\mu \, \exp(-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^\mu - s^\mu)^2) \\ \times \exp\left(-\frac{1}{2}s^\top \left[K^{(\mathrm{R})}_{\mathrm{LCN}}\right]^{-1} s - \frac{1}{2}\log\det\left[K^{(\mathrm{R})}_{\mathrm{LCN}}\right] - \frac{N_c}{2}\sum_i \left(Q_i\bar{Q}_i - \log(1+Q_i)\right)\right), \tag{A1.16}$$

where we have identified the renormalized kernel for the LCN:

$$\left[K^{(\mathrm{R})}_{\mathrm{LCN}}(\bar{Q})\right]_{\mu\nu} \equiv \frac{1}{\lfloor N_0/S \rfloor}\sum_{i=1}^{\lfloor N_0/S \rfloor} \bar{Q}_i K^i_{\mu\nu}\,. \tag{A1.17}$$

which contains only the diagonal elements of the local kernel matrix.

Finally, we perform this last Gaussian integral and we are left with a partition function that depends only on Q and $\bar{Q}$:

$$Z = \int \prod_i dQ_i d\bar{Q}_i e^{-\frac{N_c}{2}S_{\mathrm{LCN}}(Q,\bar{Q})}, \tag{A1.18}$$

where the LCN effective action S_{LCN} is given by:

$$S_{\mathrm{LCN}} \equiv -\sum_i (Q_i\bar{Q}_i - \log(1+Q_i)) + \frac{\alpha_c}{P}\mathrm{Tr}\log\beta\Big(\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})}_{\mathrm{LCN}}\Big) + \frac{\alpha_c}{P}y^\top\Big(\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})}_{\mathrm{LCN}}\Big)^{-1}y. \tag{A1.19}$$

We recall that $\left(\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})}_{\mathrm{LCN}}\right)$ is a $P \times P$ matrix. Note that this action shares the same functional form as the one for FCNs and CNNs.

A2 Similarity matrix

In the main text, we have already said that our formalism allows us to compute physical observables. One interesting observable, which gives insights about the internal representation of the network and which is deeply connected with the NN kernel, is the similarity matrix, whose definition can be found in Eq. (5.6). In this appendix, we will show how our theoretical framework can be exploited to analytically compute this quantity,

at least in the zero-temperature regime.

A2.1 FCN averaged similarity matrix

This subsection is devoted to showing our strategy for computing the expected similarity matrix for an FCN architecture. Then, we want to calculate the quantity

$$\langle O_{\mu\nu}^{\mathrm{FCN}} \rangle \equiv \left\langle \frac{1}{N_1} \sum_{i=1}^{N_1} \sigma\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\mu}{\sqrt{N_0}} \right) \sigma\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\nu}{\sqrt{N_0}} \right) \right\rangle = \frac{1}{Z_{FCN}} \int \mathcal{D}\theta e^{-\beta \mathcal{L}(\theta)} O_{\mu\nu}^{\mathrm{FCN}} \,. \quad \text{(A2.1)}$$

Note that this quantity depends on β, but since we are interested in the regime where the network perfectly interpolates the training data, we will consider the $\beta \to \infty$ limit. Moreover, the mean operation over the neuron index i commutes with the average operation over the Gibbs ensemble, then we have to compute only the expected value for a particular index. Thus, we study the expected value of the inner product over the i-th neuron,

$$\frac{1}{Z_{FCN}} \int \mathcal{D}\theta e^{-\beta \mathcal{L}(\theta)} \sigma\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\mu}{\sqrt{N_0}} \right) \sigma\left(\frac{\mathbf{W}_i \cdot \mathbf{x}^\nu}{\sqrt{N_0}} \right) , \quad \text{(A2.2)}$$

and then we compute the mean value over all the neurons in the layer. We will see that the quantity in (A2.2) does not depend on the particular choice of the neuron since the variables $h_i^\mu \equiv \frac{\mathbf{W}_i \cdot \mathbf{x}^\mu}{\sqrt{N_0}}$ are statistically identical. This integral is not directly solvable through the reasoning applied when the effective action was found, but we need a subtle trick. In the following, we explain our strategy. In order to fix the idea, we will consider the neuron indexed with $i = 1$.

As a standard practice in statistical mechanics, we need to define a modified partition function adding a source term to one of the identical N_1 neurons of the last layer: $\frac{1}{2N_1} \left(\sum_\mu J_\mu \sigma(h_1^\mu) \right)^2$. The extended partition function reads:

$$Z_J^{(\mathrm{FCN})} = \int P(\theta) \mathrm{d}\theta \, \exp\left(-\frac{\beta}{2} \sum_{\mu=1}^{P} [y^\mu - f_{\mathrm{FCN}}(x^\mu)]^2 + \frac{1}{2N_1} \left(\sum_\mu J_\mu \sigma(h_1^\mu) \right)^2 \right) \quad \text{(A2.3)}$$

where we have reabsorbed the Gaussian priors λ_0, λ_1 in the term $P(\theta) \equiv P(v, W)$. In this way, the expected value (A2.2) is easily computed from the derivatives of the partition

function with respect to the source J_μ:

$$\langle \sigma(h_1^\mu)\sigma(h_1^\nu)\rangle = -\lambda_1 N_1 \frac{1}{Z_J^{(\mathrm{FCN})}} \frac{\partial^2 Z_J^{(\mathrm{FCN})}}{\partial J^\mu \partial J^\nu}\Big|_{\mathbf{J}=0} . \tag{A2.4}$$

Let's take the FCN partition function at the stage where we have already integrated over $\mathbf{v}$, $\mathbf{W}$ and $\bar{\mathbf{h}}$, but considering the source term:

$$\begin{aligned} Z_J^{(\mathrm{FCN})} = \int \prod_\mu ds^\mu d\bar{s}^\mu e^{-\frac{\beta}{2}\sum_\mu (y^\mu - s^\mu)^2 + i\sum_\mu s^\mu \bar{s}^\mu} \left\{ \int \prod_\mu dh^\mu \, \mathcal{N}_{\mathbf{h}}\left(0, \mathbf{C}\right) e^{-\frac{1}{2\lambda_1 N_1}\left[\sum_\mu \bar{s}^\mu \sigma(h^\mu)\right]^2} \right\}^{N_1 - 1} \\ \times \int \prod_\mu dh_1^\mu \, \mathcal{N}_{\mathbf{h}}\left(0, \mathbf{C}\right) \, e^{-\frac{1}{2\lambda_1 N_1}[\sum_\mu \bar{s}^\mu \sigma(h_1^\mu)]^2} e^{-\frac{1}{2\lambda_1 N_1}[\sum_\mu J^\mu \sigma(h_1^\mu)]^2}, \end{aligned} \tag{A2.5}$$

where we have separated the $\mathbf{h}_1$ integral, but we can immediately drop the dummy index $i = 1$, that was reported for the sake of clarity. The integral in curly brackets is the same found in the calculation of the FCN partition function in [81], and the result is the same as the one of Eq. (5.25), with the substitution $N_1 \to N_1 - 1$.

The same reasoning can be applied to the integral on $\mathbf{h}_1$, but with an arrangement. First, we need to introduce two more delta function identities:

$$1 = \int dq_1 \, \delta\left(q_1 - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu \bar{s}^\mu \sigma(h^\mu)\right) , \quad 1 = \int dq_2 \, \delta\left(q_2 - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu J^\mu \sigma(h^\mu)\right) , \tag{A2.6}$$

where we have already dropped the $i = 1$ index. Identifying a collective variable $\mathbf{q} \equiv (q_1, q_2)$, the integral in the last row of (A2.5) can be rewritten as:

$$\int d^2 q \, P(\mathbf{q}) \, e^{-\frac{1}{2}\|q\|^2} ,$$

where

$$P(\mathbf{q}) = \int \prod_\mu dh_1^\mu \, \mathcal{N}_{\mathbf{h}}\left(0, \mathbf{C}\right) \, \delta\left(q_1 - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu \bar{s}^\mu \sigma(h^\mu)\right) \delta\left(q_2 - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu J^\mu \sigma(h^\mu)\right) \tag{A2.7}$$

If the BM theorem's hypotheses hold for q_1 and q_2, we can use it again to justify a

Gaussian equivalence for the distribution $P(\mathbf{q})$:

$$P(\mathbf{q}) \to \mathcal{N}_{\mathbf{q}}(0, \mathbf{\Sigma}(\bar{\mathbf{s}})) \equiv \frac{e^{-\frac{1}{2}\mathbf{q}^\top \mathbf{\Sigma}^{-1}\mathbf{q}}}{\sqrt{(2\pi)^2 \det(\mathbf{\Sigma})}}, \tag{A2.8}$$

where $\mathbf{\Sigma}(\bar{\mathbf{s}})$ is a 2×2 covariance matrix defined as:

$$\mathbf{\Sigma}(\bar{\mathbf{s}}) = \frac{1}{\lambda_1 N_1} \begin{pmatrix} \bar{\mathbf{s}}^\top \mathbf{K} \bar{\mathbf{s}} & \bar{\mathbf{s}}^\top \mathbf{K} \mathbf{J} \\ \bar{\mathbf{s}}^\top \mathbf{K} \mathbf{J} & \mathbf{J}^\top \mathbf{K} \mathbf{J} \end{pmatrix}, \tag{A2.9}$$

where $\mathbf{K}$ is the NNGP kernel matrix, whose components we recall being $K_{\mu\nu} = \int d^P h \, \mathcal{N}_{\mathbf{h}}(0, \mathbf{C}) \sigma(h^\mu) \sigma(h^\nu)$. In this way, the integral on $\mathbf{q}$ is Gaussian and Eq. (A2.5) becomes:

$$Z_J^{(\mathrm{FCN})} = \int \prod_\mu ds^\mu d\bar{s}^\mu e^{-\frac{\beta}{2}\sum_\mu (y^\mu - s^\mu)^2 + i \sum_\mu s^\mu \bar{s}^\mu} \left[Q(\bar{\mathbf{s}}) + 1\right]^{-\frac{N_1 - 1}{2}} \left[\det\left(\mathbb{1} + \mathbf{\Sigma}(\bar{\mathbf{s}})\right)\right]^{-\frac{1}{2}} . \tag{A2.10}$$

Let us focus on the determinant in the equation above, which is the only term that depends on the source J^μ. Firstly we point out that, when the source is vanishing, equation (A2.10) correctly reduces to the FCN partition function already computed:

$$\det(\mathbb{1} + \mathbf{\Sigma})\Big|_{\mathbf{J}=0} = Q(\bar{\mathbf{s}}) + 1, \tag{A2.11}$$

To avoid heavy notation, we have dropped the explicit dependence $\mathbf{\Sigma}(\bar{\mathbf{s}})$, which will be understood implicitly in the following. The derivatives with respect to J^μ and J^ν can be computed using the Jacobi's identities:

$$\begin{aligned} \partial_{J^\mu} \partial_{J^\nu} \left[\det(\mathbb{1} + \mathbf{\Sigma})\right]^{-\frac{1}{2}} = \left[\det(\mathbb{1} + \mathbf{\Sigma})\right]^{-\frac{1}{2}} \Bigg\{ & \frac{1}{4} \mathrm{Tr}\Big((\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\nu} \mathbf{\Sigma}\Big) \mathrm{Tr}\Big((\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\mu} \mathbf{\Sigma}\Big) + \\ & - \frac{1}{2} \mathrm{Tr}\Big((\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\mu} \mathbf{\Sigma} (\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\nu} \mathbf{\Sigma}\Big) + \frac{1}{2} \mathrm{Tr}\Big((\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\mu} \partial_{J^\nu} \mathbf{\Sigma}\Big) \Bigg\}, \end{aligned} \tag{A2.12}$$

$$\frac{\partial \mathbf{\Sigma}}{\partial J^\mu}\Big|_{\mathbf{J}=0} = \frac{1}{\lambda_1 N_1} \begin{pmatrix} 0 & (\mathbf{K}\bar{\mathbf{s}})^\mu \\ (\mathbf{K}\bar{\mathbf{s}})^\mu & 0 \end{pmatrix}, \qquad \frac{\partial^2 \mathbf{\Sigma}}{\partial J^\mu \partial J^\nu}\Big|_{\mathbf{J}=0} = \frac{2}{\lambda_1 N_1} \begin{pmatrix} 0 & 0 \\ 0 & K_{\mu\nu} \end{pmatrix}, \tag{A2.13}$$

where $(\mathbf{K}\bar{\mathbf{s}})^\mu \equiv \sum_\rho K_{\mu\rho}\bar{s}^\rho$. One can see that the first term in curly brackets in Eq. (A2.12) vanishes when $J = 0$, since the matrix

$$(\mathbb{1} + \mathbf{\Sigma})^{-1}\partial_{J^\mu}\mathbf{\Sigma}\Big|_{\mathbf{J}=0} = \frac{1}{\lambda_1 N_1}\begin{pmatrix} 0 & (1+Q(\bar{\mathbf{s}}))^{-1}(\mathbf{K}\bar{\mathbf{s}})^\mu \\ (\mathbf{K}\bar{\mathbf{s}})^\mu & 0 \end{pmatrix}, \tag{A2.14}$$

has zero trace. The two non-zero contributions give:

$$-\frac{1}{2}\mathrm{Tr}\Big((\mathbb{1} + \mathbf{\Sigma})^{-1}\partial_{J^\mu}\mathbf{\Sigma}(\mathbb{1} + \mathbf{\Sigma})^{-1}\partial_{J^\nu}\mathbf{\Sigma}\Big) = -\frac{1}{(\lambda_1 N_1)^2}\frac{(\mathbf{K}\bar{\mathbf{s}})^\mu(\mathbf{K}\bar{\mathbf{s}})^\nu}{1+Q(\bar{\mathbf{s}})}, \tag{A2.15}$$

$$\frac{1}{2}\mathrm{Tr}\Big((\mathbb{1} + \mathbf{\Sigma})^{-1}\partial_{J^\mu}\partial_{J^\nu}\mathbf{\Sigma}\Big) = K_{\mu\nu}/(\lambda_1 N_1), \tag{A2.16}$$

Collecting what we have computed so far, we have:

$$-\lambda_1 N_1 \partial_{J^\mu}\partial_{J^\nu}(\det(\mathbb{1} + \mathbf{\Sigma}))^{-\frac{1}{2}}\Big|_{\mathbf{J}=0} = (1+Q(\bar{\mathbf{s}}))^{-\frac{1}{2}}\left\{K_{\mu\nu} - \frac{1}{\lambda_1 N_1}\frac{(\mathbf{K}\bar{\mathbf{s}})^\mu(\mathbf{K}\bar{\mathbf{s}})^\nu}{1+Q(\bar{\mathbf{s}})}\right\}. \tag{A2.17}$$

The next step is to put back this result in Eq.(A2.4), having already set $\mathbf{J} = 0$, and to apply the same strategy adopted for computing the effective action. Thus, we insert the Dirac's delta for the $Q(\bar{\mathbf{s}})$. The integral on $\mathbf{s}$ is Gaussian, and dropping constant terms we get

$$\langle\sigma(h^\mu)\sigma(h^\nu)\rangle = \frac{1}{Z_{FCN}}\int dQ d\bar{Q}\, e^{-\frac{N_1}{2}\log(1+Q)+\frac{N_1}{2}Q\bar{Q}} \tag{A2.18}$$

$$\times \int \prod_\mu d\bar{s}^\mu e^{-\frac{1}{2\beta}\sum_\mu(\bar{s}^\mu)^2 + i\sum_\mu y^\mu\bar{s}^\mu - \frac{\bar{Q}}{2\lambda_1}\bar{\mathbf{s}}^\top\mathbf{K}\bar{\mathbf{s}}}\Big[K_{\mu\nu} - \frac{1}{\lambda_1 N_1}\frac{(\mathbf{K}\bar{\mathbf{s}})^\mu(\mathbf{K}\bar{\mathbf{s}})^\nu}{1+Q}\Big]. \tag{A2.19}$$

The integral on $\bar{\mathbf{s}}$ is again Gaussian and can be written as

$$e^{-\frac{1}{2}\mathbf{y}^\top[\tilde{\mathbf{K}}^{(\mathrm{R})}]^{-1}\mathbf{y}}\int d^P\bar{s}\; e^{-\frac{1}{2}\left(\bar{\mathbf{s}}+i[\tilde{\mathbf{K}}^{(\mathrm{R})}]^{-1}\cdot\mathbf{y}\right)^\top\tilde{\mathbf{K}}^{(\mathrm{R})}\left(\bar{\mathbf{s}}+i[\tilde{\mathbf{K}}^{(\mathrm{R})}]^{-1}\cdot\mathbf{y}\right)}\left[K_{\mu\nu} - \frac{1}{\lambda_1 N_1}\frac{(\mathbf{K}\bar{\mathbf{s}})^\mu(\mathbf{K}\bar{\mathbf{s}})^\nu}{1+Q}\right], \tag{A2.20}$$

where we call $\tilde{\mathbf{K}}^{(\mathrm{R})} \equiv \left(\frac{\mathbb{1}}{\beta} + \mathbf{K}^{(\mathrm{R})}\right)$, with $\mathbf{K}^{(\mathrm{R})} \equiv \bar{Q}\mathbf{K}/\lambda_1$ the *renormalized* kernel matrix. Note that, in the zero-temperature limit, $\tilde{\mathbf{K}}^{(\mathrm{R})}$ reduces to $\mathbf{K}^{(\mathrm{R})}$. The first term in the square brackets of Eq. (A2.20) does not depend on $\bar{\mathbf{s}}$ and so it will give us a term $K_{\mu\nu}Z_{FCN}$, proportional to the partition function. After the transformation $\bar{s}^\mu \to \bar{s}^\mu - i\left([\tilde{\mathbf{K}}^{(\mathrm{R})}]^{-1}\cdot\mathbf{y}\right)^\mu$,

we have a zero-mean Gaussian integral, whose solution reads:

$$-\frac{[\det(\tilde{\mathbf{K}}^{(\mathrm{R})})]^{-\frac{1}{2}}}{\lambda_1 N_1(1+Q)}\sum_{\rho\lambda}K_{\mu\rho}K_{\nu\lambda}\left[[\tilde{K}^{(\mathrm{R})}]^{-1}_{\rho\lambda}-([\tilde{\mathbf{K}}^{(\mathrm{R})}]^{-1}\cdot\mathbf{y})^{\rho}([\tilde{\mathbf{K}}^{(\mathrm{R})}]^{-1}\cdot\mathbf{y})^{\lambda}\right]. \tag{A2.21}$$

Then, we can write

$$\begin{aligned}\langle\sigma(h^\mu)\sigma(h^\nu)\rangle = K_{\mu\nu}+\frac{1}{Z}\int dQd\bar{Q}\,e^{-\frac{N_1}{2}S_{\mathrm{FCN}}(Q,\bar{Q})}\\ \times\frac{1}{\lambda_1 N_1(1+Q)}\Big[-\sum_{\lambda\rho}K_{\mu\lambda}K_{\nu\rho}[\tilde{K}^{(\mathrm{R})}]^{-1}_{\lambda\rho}+\\ +\sum_{\lambda\rho\epsilon\omega}K_{\mu\lambda}K_{\nu\rho}[\tilde{K}^{(\mathrm{R})}]^{-1}_{\lambda\epsilon}[\tilde{K}^{(\mathrm{R})}]^{-1}_{\rho\omega}\,y^{\epsilon}y^{\omega}\Big],\end{aligned} \tag{A2.22}$$

where we recall the summations over Greek indexes are in the range $(0, P)$. Finally, we can take the saddle point solution of the effective action, for which $(1+Q)^{-1}=\bar{Q}$ and in the zero-temperature limit $\beta\to\infty$, we get

$$\langle\sigma(h^\mu)\sigma(h^\nu)\rangle=\Big(1-\frac{1}{N_1}\Big)K_{\mu\nu}+\frac{\lambda_1}{N_1\bar{Q}}y^\mu y^\nu\,. \tag{A2.23}$$

Since this result is independent on the index i, the average of the observable in (6.22) is trivially retrieved:

$$\left\langle O^{\mathrm{FCN}}_{\mu\nu}\right\rangle=\left\langle\frac{1}{N_1}\sum_{i=1}^{N_1}\sigma\left(h_i^\mu\right)\sigma\left(h_i^\nu\right)\right\rangle\equiv\frac{1}{N_1}\sum_{i=1}^{N_1}\left\langle\sigma\left(h_i^\mu\right)\sigma\left(h_i^\nu\right)\right\rangle=\left\langle\sigma\left(h^\mu\right)\sigma\left(h^\nu\right)\right\rangle. \tag{A2.24}$$

A2.2 CNN averaged similarity matrix

In this subsection, we extend the previous calculation to CNNs. We want to compute the average $\left\langle O^{\mathrm{CNN}}_{\mu\nu}\right\rangle$, where the observable $O^{\mathrm{CNN}}_{\mu\nu}$ is defined in (6.23). In this case, we will compute the expectation value for only one particular channel, whose index is a, since they are all statistically independent. Thus, in the following, we will drop the channel suffix.

Similarly to the previous section, we add a source term to the CNN's partition function $\frac{1}{\lambda_1 N_1}\sum_i\left[\sum_\mu J^\mu\sigma(h_i^\mu)\right]^2$. Here $N_1=N_c\lfloor N_0/S\rfloor$ denotes the number of neurons in the last

layer of the CNN. The modified partition function reads:

$$Z_J^{(\mathrm{CNN})} = \int \prod_\mu ds^\mu d\bar{s}^\mu e^{-\frac{\beta}{2}\sum_\mu (y^\mu - s^\mu)^2 + i\sum_\mu s^\mu \bar{s}^\mu} \left\{ \int dP(h_i^\mu)\, e^{-\frac{1}{2\lambda_1 N_1}\sum_i [\sum_\mu \bar{s}^\mu \sigma(h_i^\mu)]^2} \right\}^{N_c - 1}$$
$$\times \int dP(h_i^\mu)\, e^{-\frac{1}{2\lambda_1 N_1}\sum_i [\sum_\mu \bar{s}^\mu \sigma(h_i^\mu)]^2} e^{-\frac{1}{2\lambda_1 N_1}\sum_i [\sum_\mu J^\mu \sigma(h_i^\mu)]^2}, \tag{A2.25}$$

where:

$$dP(h_i^\mu) \equiv \prod_{\mu i} \frac{dh_i^\mu}{\sqrt{\det 2\pi C}} e^{-\frac{1}{2}\sum_{\mu\nu ij} h_i^\mu [C^{-1}]_{\mu\nu}^{ij} h_j^\nu}. \tag{A2.26}$$

Analogously to the FCN case, one can check that:

$$\left\langle \frac{1}{\lfloor \frac{N_0}{S} \rfloor} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sigma(h_i^{a\mu}) \sigma(h_i^{a\nu}) \right\rangle = -\lambda_1 N_c \frac{1}{Z_J^{(\mathrm{FCN})}} \frac{\partial^2 Z_J^{(\mathrm{FCN})}}{\partial J^\mu \partial J^\nu}\Big|_{\mathbf{J}=0}. \tag{A2.27}$$

Again, we need to find a strategy to compute the integral in the last row of (A2.25). To do that, we repeat the resoning made before for the FCN case and we insert two families of Dirac's δ's:

$$\int \prod_i dq_{\bar{s}}^i \delta\big(q_{\bar{s}}^i - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu \bar{s}^\mu \sigma(h_i^\mu)\big)\,, \quad \int \prod_i dq_J^i \delta\big(q_J^i - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu \bar{J}^\mu \sigma(h_i^\mu)\big)\,, \tag{A2.28}$$

and so we are left to the problem of finding the joint distribution $P(\mathbf{q}_{\bar{\mathbf{s}}}, \mathbf{q}_{\mathbf{J}})$ of the new variables. We employ a multivariate version of the BM theorem to make the usual Gaussian approximation:

$$P(\mathbf{q}_{\bar{\mathbf{s}}}, \mathbf{q}_{\mathbf{J}}) \to (\det 2\pi\boldsymbol{\Sigma})^{-1/2} \exp\left(-\frac{1}{2} (\mathbf{q}_{\bar{\mathbf{s}}} \,|\, \mathbf{q}_{\mathbf{J}})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{q}_{\bar{\mathbf{s}}} \,|\, \mathbf{q}_{\mathbf{J}}) \right), \tag{A2.29}$$

where we made use of the following notation for the concatenation of two vectors:

$$(\mathbf{q}_{\bar{\mathbf{s}}} \,|\, \mathbf{q}_{\mathbf{J}}) \equiv (q_{\bar{s}}^{(1)}, \cdots, q_{\bar{s}}^{(\lfloor N_0/S \rfloor)}, q_J^{(1)}, \cdots, q_J^{(\lfloor N_0/S \rfloor)}). \tag{A2.30}$$

Furthermore, the covariance matrix of this joint distribution is a block matrix of the form

$$\boldsymbol{\Sigma} = \frac{1}{\lambda_1 N_1} \left(\begin{array}{c|c} \bar{\mathbf{s}}^\top \mathbf{K}^{ij} \bar{\mathbf{s}} & \bar{\mathbf{s}}^\top \mathbf{K}^{ij} \mathbf{J} \\ \hline \bar{\mathbf{s}}^\top \mathbf{K}^{ij} \mathbf{J} & \mathbf{J}^\top \mathbf{K}^{ij} \mathbf{J} \end{array} \right), \tag{A2.31}$$

where $\bar{\mathbf{s}}^\top \mathbf{K}^{ij} \bar{\mathbf{s}}$ represents the $\lfloor N_0/S \rfloor \times \lfloor N_0/S \rfloor$ matrix which remains after contracting $K^{ij}_{\mu\nu}$ with two vectors on the pattern indices μ and ν, i.e. $\bar{\mathbf{s}}^\top \mathbf{K}^{ij} \bar{\mathbf{s}} \equiv \sum_{\mu\nu} K^{ij}_{\mu\nu} \bar{s}^\mu \bar{s}^\nu$. From now on, every block in the following matrices will be $\lfloor N_0/S \rfloor \times \lfloor N_0/S \rfloor$ matrices. The integral on the $\mathbf{q}$ variables is Gaussian and gives the term $\det(\mathbb{1} + \mathbf{\Sigma})^{-1/2}$. Then, we have to compute the second derivative of this term, analogously as was done for the FCN case, recovering Eq. (A2.12), obviously with a different definition of the matrix $\mathbf{\Sigma}$. Then, for the CNN case, we have

$$(\mathbb{1} + \mathbf{\Sigma})^{-1}\Big|_{\mathbf{J}=0} = \left(\begin{array}{c|c} (\mathbb{1} + \mathbf{Q}(\bar{\mathbf{s}}))^{-1} & 0 \\ \hline 0 & \mathbb{1} \end{array}\right)$$

$$(\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\mu} \mathbf{\Sigma}\Big|_{\mathbf{J}=0} = \frac{1}{\lambda_1 N_1} \left(\begin{array}{c|c} 0 & \sum_k (\mathbb{1} + \mathbf{Q}(\bar{\mathbf{s}}))^{-1}_{ik} (\mathbf{K}^{kj} \bar{\mathbf{s}})^\mu \\ \hline (\mathbf{K}^{ij} \bar{\mathbf{s}})^\mu & 0 \end{array}\right)$$

$$\begin{aligned} &(\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\mu} \mathbf{\Sigma} (\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\nu} \mathbf{\Sigma}\Big|_{\mathbf{J}=0} = \\ &= \frac{1}{(\lambda_1 N_1)^2} \left(\begin{array}{c|c} \sum_{kl} (\mathbb{1} + \mathbf{Q}(\bar{\mathbf{s}}))^{-1}_{ik} (\mathbf{K}^{kl} \bar{\mathbf{s}})^\mu (\mathbf{K}^{lj} \bar{\mathbf{s}})^\nu & 0 \\ \hline 0 & \sum_{kl} (\mathbf{K}^{ik} \bar{\mathbf{s}})^\mu (\mathbb{1} + \mathbf{Q}(\bar{\mathbf{s}}))^{-1}_{kl} (\mathbf{K}^{lj} \bar{\mathbf{s}})^\nu \end{array}\right) \end{aligned}$$

$$(\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\mu} \partial_{J^\nu} \mathbf{\Sigma}\Big|_{\mathbf{J}=0} = \frac{2}{\lambda_1 N_1} \left(\begin{array}{c|c} 0 & 0 \\ \hline 0 & K^{ij}_{\mu\nu} \end{array}\right) , \tag{A2.32}$$

where we have defined $[Q(\bar{\mathbf{s}})]_{ij} \equiv \frac{1}{\lambda_1 N_1} \bar{\mathbf{s}}^\top \mathbf{K}^{ij} \bar{\mathbf{s}}$ and $(\mathbf{K}^{ij} \bar{\mathbf{s}})^\mu \equiv \sum_\nu K^{ij}_{\mu\nu} \bar{s}^\nu$. Note that the four blocks in each matrix are a $\lfloor N_0/S \rfloor \times \lfloor N_0/S \rfloor$ matrix, whose indices, when made explicit, are indicated with Latin letters (ij). In these equations, the second one has zero trace, while the trace of the third reads

$$\mathrm{Tr}\Big((\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\mu} \mathbf{\Sigma} (\mathbb{1} + \mathbf{\Sigma})^{-1} \partial_{J^\nu} \mathbf{\Sigma}\Big|_{\mathbf{J}=0} \Big) = \sum_{i,j=1}^{\lfloor N_0/S \rfloor} \sum_{\lambda,\rho=1}^{P} \frac{2}{(\lambda_1 N_1)^2} (\mathbb{1} + \mathbf{Q}(\bar{\mathbf{s}}))^{-1}_{ij} \bar{s}^\lambda \bar{s}^\rho P^{ij}_{\mu\lambda\nu\rho} , \tag{A2.33}$$

where, for the sake of notation, we have defined the six-indices matrix:

$$P^{ij}_{\mu\lambda\nu\rho} = \frac{1}{2}\left(\sum_{k=1}^{\lfloor N_0/S\rfloor} K^{ik}_{\mu\lambda}K^{kj}_{\nu\rho} + K^{ki}_{\mu\lambda}K^{jk}_{\nu\rho}\right) . \tag{A2.34}$$

Now we are able to solve Eq. (A2.12) and we get

$$\begin{aligned}-\lambda_1 N_c \partial_{J^\mu}\partial_{J^\nu}(\det(\mathbb{1}+\mathbf{\Sigma}))^{-\frac{1}{2}} =& \frac{1}{\lfloor N_0/S\rfloor}\sum_{i=1}^{\lfloor N_0/S\rfloor} K^{ii}_{\mu\nu} + \\ &- \frac{1}{\lambda_1 N_c}\frac{1}{\lfloor N_0/S\rfloor^2}\sum_{i,j=1}^{\lfloor N_0/S\rfloor}\sum_{\lambda,\rho=1}^{P}(\mathbb{1}+\mathbf{Q}(\bar{\mathbf{s}}))^{-1}_{ij}\bar{s}^\lambda\bar{s}^\rho P^{ij}_{\mu\lambda\nu\rho},\end{aligned} \tag{A2.35}$$

where $\sum_{i=1}^{\lfloor N_0/S\rfloor} K^{ii}_{\mu\nu}$ is the trace of K along the spatial indices (i,j). Since it is a constant we can take it outside the integral. Thus, let's compute the second contribution. For the sake of notation and since the only non-contracted indices are (μ,ν), we will refer to it as $P_{\mu\nu}(\bar{\mathbf{s}})$.

First, we insert the deltas for the $\mathbf{Q}(\bar{\mathbf{s}})$ variables, to get rid of the explicit dependency on $\bar{\mathbf{s}}$, and we make use again of the Fourier representation of these deltas, with the variables $\bar{\mathbf{Q}}$. Since the new term depends only on $\bar{\mathbf{s}}$, we can easily integrate on $\mathbf{s}$, obtaining (we write only the $\bar{\mathbf{s}}$ integral)

$$\begin{aligned}&\int\prod_\mu d\bar{s}^\mu\, e^{-\frac{1}{2\beta}\sum_\mu \bar{s}^{\mu 2}+i\sum_\mu y^\mu\bar{s}^\mu-\frac{1}{2}\sum_{\mu\nu}K^{(\mathrm{R})}_{\mu\nu}\bar{s}^\mu\bar{s}^\nu}P_{\mu\nu}(\bar{\mathbf{s}}) = \\ &e^{-\frac{1}{2}\mathbf{y}^\top\tilde{\mathbf{K}}^{(\mathrm{R})}\mathbf{y}}\int\prod_\mu d\bar{s}^\mu\, e^{-\frac{1}{2}(\bar{\mathbf{s}}+i(\tilde{\mathbf{K}}^{(\mathrm{R})})^{-1}\mathbf{y})^\top\mathbf{K}^{(\mathrm{R})}(\bar{\mathbf{s}}+i(\tilde{\mathbf{K}}^{(\mathrm{R})})^{-1}\mathbf{y})}P_{\mu\nu}(\bar{\mathbf{s}}) ,\end{aligned} \tag{A2.36}$$

where $\mathbf{K}^{(\mathrm{R})}$ is the renormalized local kernel defined in (6.14) and $\tilde{\mathbf{K}}^{(\mathrm{R})} \equiv \frac{\mathbb{1}}{\beta}+\mathbf{K}^{(\mathrm{R})}$. Following an analogous strategy already performed in the FCN case, this integral can be solved performing the transformation $\bar{s}^\mu \to \bar{s}^\mu - i\sum_\epsilon[\tilde{K}^{(\mathrm{R})}]^{-1}_{\mu\epsilon}y^\epsilon$, which implies that $P_{\mu\nu}(\bar{\mathbf{s}})$ becomes

$$P_{\mu\nu}(\bar{\mathbf{s}}) \to \frac{1}{\lambda_1 N_c}\frac{1}{\lfloor N_0/S\rfloor^2}\sum_{i,j=1}^{\lfloor N_0/S\rfloor}\sum_{\lambda,\rho=1}^{P}(\mathbb{1}+\mathbf{Q}(\bar{\mathbf{s}}))^{-1}_{ij}P^{ij}_{\mu\lambda\nu\rho}\left[\bar{s}^\lambda\bar{s}^\rho - \sum_{\epsilon,\omega=1}^{P}(\tilde{K}^{(\mathrm{R})})^{-1}_{\lambda\epsilon}(\tilde{K}^{(\mathrm{R})})^{-1}_{\rho\omega}y^\epsilon y^\omega\right] . \tag{A2.37}$$

With this change of variables, we can perform the Gaussian integral, and we are left only with the CNN partition function having only the $\mathbf{Q}, \bar{\mathbf{Q}}$ terms. Finally, we take all the contributions at the saddle-point solution and at the zero-temperature regime, which means

$$\begin{aligned}(\mathbb{1} + \mathbf{Q})^{-1} &\to \bar{\mathbf{Q}}, \\ \tilde{\mathbf{K}}^{(\mathrm{R})} &\to \mathbf{K}^{(\mathrm{R})}.\end{aligned} \tag{A2.38}$$

Collecting what we have achieved so far and noting that Eq. (A2.27) is independent wrt to the channel index, we get the final form of the similarity matrix for a CNN shallow architecture

$$\begin{aligned}\langle O^{\mathrm{CNN}}_{\mu\nu} \rangle = \left\langle \frac{1}{N_c} \sum_{a=1}^{N_c} \frac{1}{\lfloor \frac{N_0}{S} \rfloor} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sigma(h_i^{a\mu}) \sigma(h_i^{a\nu}) \right\rangle \equiv \frac{1}{N_c} \sum_{a=1}^{N_c} \left\langle \frac{1}{\lfloor \frac{N_0}{S} \rfloor} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sigma(h_i^{a\mu}) \sigma(h_i^{a\nu}) \right\rangle = \\ = \frac{1}{\lfloor N_0/S \rfloor} \sum_{i=1}^{\lfloor N_0/S \rfloor} K^{ii}_{\mu\nu} + \\ - \frac{1}{\lambda_1 N_c} \frac{1}{\lfloor N_0/S \rfloor^2} \sum_{i,j=1}^{\lfloor N_0/S \rfloor} \sum_{\lambda,\rho=1}^{P} \bar{Q}_{ij} P^{ij}_{\mu\lambda\nu\rho} \left[(K^{(\mathrm{R})})^{-1}_{\lambda\rho} - \sum_{\epsilon,\omega=1}^{P} (K^{(\mathrm{R})})^{-1}_{\lambda\epsilon} (K^{(\mathrm{R})})^{-1}_{\rho\omega} y^{\epsilon} y^{\omega} \right].\end{aligned} \tag{A2.39}$$

A3 Beyond the IW limit: numerical subtleties

We perform numerical experiments both with shallow FCNs, and CNNs with 1d and 2d convolutions. The networks are trained on two different regression tasks. Respectively: (i) a synthetic random dataset, whose elements have entries sampled from a Gaussian distribution $\mathcal{N}(0, 1)$, with labels given by a linear teacher function with unitary weights $t = \{1 \dots 1\}$:

$$y^{\mu} = \frac{1}{2} \left(1 + \mathrm{sign}\left(t \cdot x^{\mu}\right)\right) \tag{A3.1}$$

(ii) a computer vision task: we use the 0 and 1 classes of the CIFAR10 datasets, respectively corresponding to the labels "cars" and "planes". The images are coarse-grained to $N_0 = 28 \times 28 = D^2$ pixels and converted to grayscale.

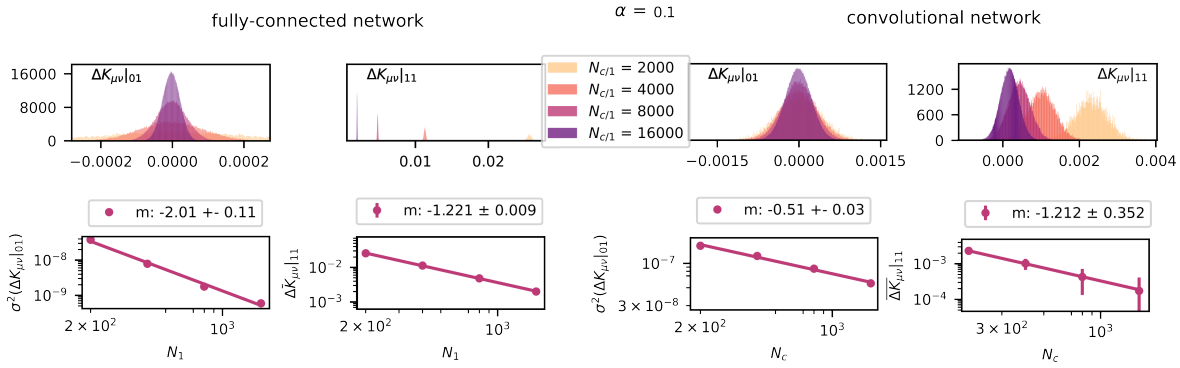

Figure A3.1: Additional finite-size scaling experiments on the similarity matrix, $\alpha = 0.1$.

A3.1 Experiments with 1d convolutions

The finite-width analysis presented in Figs. 6.3 and 6.4 is carried out by training a one hidden layer network with 1d convolutional filters on the synthetic dataset described in the previous section. The network implements exactly the function reported in Eq. (6.2), with activation function $\sigma(x) = \tanh(x)$. The FCN architecture is retrieved by setting both the filter size M and the stride S equal to the input dimension $N_0 = M = S$. This correctly implements the function given in Eq. (5.7). Here the networks are trained using full-batch gradient descent, with ADAM optimizer, implemented with TensorFlow (TF) [92]. We train using a large value of the last layer Gaussian prior λ_1, until the loss reaches a value of $O(10^{-10})$. We employ a scheduler for the learning rate, reducing its value from 10^{-3} to 10^{-7} with the so-called ReduceLROnPlateau scheduler of TF. The experiments shown in Figs. 6.3 and 6.4 are performed fixing the value of $\alpha_1 = P/N_1 = 1$ with increasing values of N_1. The input size is set to $N_0 = 6400$ and we choose non-overlapping convolutional filters, taking the size of the mask and the stride equal to $M = S = 400$. We build a statistical sample of $O(10^2)$ networks, trained independently, over which we averaged each result. In Fig. A3.2 and A3.1 we show the result for $\alpha_1 = 10$ and $\alpha_1 = 0.1$, respectively. In order to test the consistency of our results, we carried out the same experiments for a smaller value of N_0, in particular, we chose $N_0 = 1600$ and the same values of P and N_1. We find similar results, also in this regime where N_0 is comparable with N_1. We collect all the numerical fit of every experiment in Table A3.1.

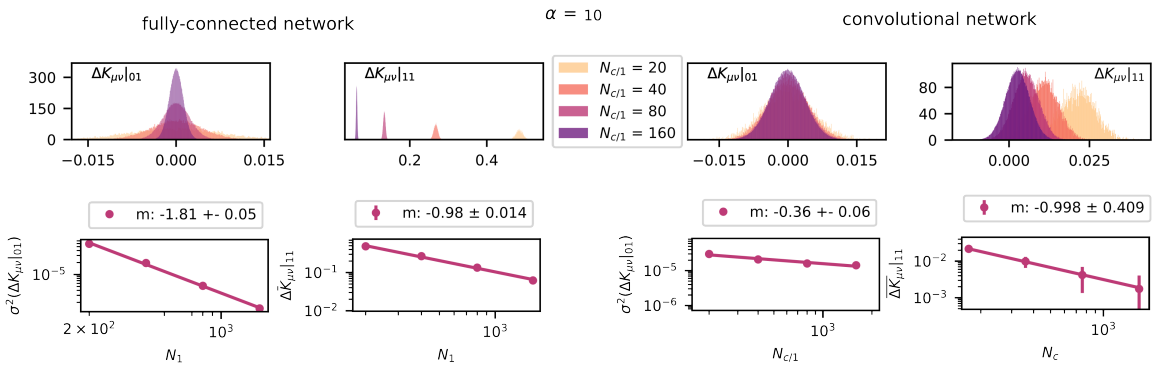

Figure A3.2: Additional finite-size scaling experiments on the similarity matrix, $\alpha = 10$.

N_0	$\alpha_{1/c} = P/N_{1/c}$	Fully-connected network		Convolutional network	
		$\sigma^2(\Delta K_{\mu\nu}\|_{01})$	$\overline{\Delta K_{\mu\nu}}\|_{11}$	$\sigma^2(\Delta K_{\mu\nu}\|_{01})$	$\overline{\Delta K_{\mu\nu}}\|_{11}$
6400	0.1	-2.01 ± 0.11	1.221 ± 0.009	-0.51 ± 0.03	-1.212 ± 0.352
	1	-2.15 ± 0.01	1.153 ± 0.011	-0.2 ± 0.01	-1.045 ± 0.355
	10	-1.81 ± 0.05	-0.980 ± 0.014	-0.36 ± 0.06	0.998 ± 0.409
1600	0.1	-1.34 ± 0.23	-1.278 ± 0.018	-0.014 ± 0.046	-0.9 ± 2.9
	1	-1.43 ± 0.22	-1.154 ± 0.017	0.008 ± 0.051	-0.9 ± 2.8
	10	-1.35 ± 0.13	-1.003 ± 0.021	-0.087 ± 0.029	-0.9 ± 2.8

Table A3.1: Additional finite-size scaling experiments on the similarity matrix. In this table, we summarize the results of the various fits for $\alpha_{1/c} = 0.1, 1, 10$, with two choices of $N_0 = 6400, 1600$. The values of P used in all experiments are $P = 200, 400, 800, 1600$. The column denoted by $\sigma^2(\Delta K_{\mu\nu}|_{01})$ shows the trend of the variance of the elements in the submatrix with labels $0, 1$, while in the one denoted by $\overline{\Delta K_{\mu\nu}}|_{11}$ we put the fit of the mean value of the elements in the submatrix with labels 11.

A3.2 Experiments with 2d convolutions

The results shown in Fig. 6.2 are obtained by training a one hidden layer architecture with Erf activation and $2d$ non-overlapping convolutional filters on the CIFAR10 binary task discussed in the main text. This is achieved by setting the stride S to be equal to the filter mask size M. To avoid information loss, we choose M to be an integer divisor of the linear input size $d = 28$. To ensure convergence of the posterior weights distribution to the Gibbs ensemble, we train our networks using a discretized Langevin dynamics, similarly to what is done in [70, 69, 81]. At each training step t the parameters $\theta = \{W, v\}$ are updated according to:

$$\theta(t+1) = \theta(t) - \eta \nabla_\theta \mathcal{L}(\theta(t)) + \sqrt{2T\eta}\epsilon(t) \tag{A3.2}$$

where $T = 1/\beta$ is the temperature, η is the learning rate, $\epsilon(t)$ is a white Gaussian noise vector with entries drawn from a standard normal distribution, and the loss is the one defined in equation (5.2). We employ $T = \eta = 2 \cdot 10^{-3}$ throughout all these experiments. This is sufficient to approximate the $T = 0$ dynamics in the regime we are considering. This dynamics requires $\sim 10^6$ steps to reach thermalization, in particular, we run the experiment for $5 \cdot 10^6$ epochs. When possible, we extract the generalization loss within a single run: after the training, the error has reached its minimum and the test loss is thermalized, we average test loss values every 10^3 epochs. In the case of FCN architecture in Fig. 6.2, we averaged over $n = 3$ samples to reduce the error.