Critical Points of Deep Linear Networks in $\mathbb{C}^N$

A Thesis submitted to the faculty of

San Francisco State University

In partial satisfaction of the

requirements for

the Degree

Master of Arts

in

Mathematics

by

Ayush Bharadwaj

San Francisco, California

Dec 2022

Certification of Approval

I certify that I have read Critical Points of Deep Linear Networks in $\mathbb{C}^N$ by Ayush Bharadwaj and that in my opinion this work meets the criteria for approving a thesis submitted in partial fulfillment of the requirement for the degree Master of Arts at San Francisco State University.

Serkan Hosten, Ph.D
Professor
Thesis Committee Chair

Tao He, Ph.D
Associate Professor

Henry Boateng, Ph.D
Assistant Professor

Abstract

In this thesis, we explore the critical points of the loss function of deep linear networks from an algebraic geometric perspective by viewing these critical points as solutions to a polynomial system associated with the network. We focus on 1-hidden layer networks, for which we identify a new upper bound, $\mathcal{B}_{\mathbb{C}^*}$, on the number of critical points that lie in $(\mathbb{C}^*)^N$, where N is the number of weights in the network. We show that this upper bound is smaller (in some cases, orders of magnitude smaller) than the BKK bound on the polynomial system associated with the network. We also identify an upper bound, $\mathcal{B}_{\mathbb{C}}$, on the total number of complex critical points of the network.

Further, we explore the structure within the critical points of linear networks. In particular, for 1-hidden layer networks, we prove that critical points that lie outside $(\mathbb{C}^*)^N$ must lie on particular coordinate subspaces. For larger networks, we demonstrate this through experiments.

Acknowledgments

I want to express my deepest thanks to Dr. Serkan Hosten for his invaluable guidance and encouragement; to Sneha for her unwavering support in all of life's pursuits; and to my friends for taking an interest in my journey.

Table of Contents

List of Tables

List of Figures

Introduction

Biologically inspired artificial neural networks have proved to be powerful function approximaters, able to approximate highly non-linear functions. This ability is a result of the non-linearity of the network's activation function. However, a non-linear activation function also means that the loss function, which is minimized in the process of training a network, tends to be highly non-convex. Consequently, finding all minima of the loss function becomes an important but challenging task. Even specifying a good upper bound on the number of minima can be hard.

A different approach is to turn the problem of training a neural network into a that of solving a system of polynomial equations. Doing so allows us to use known results from algebraic geometry to place upper bounds on the number of complex critical points of the loss function. Knowing these bounds then allows us to find all complex critical points of modest sized networks using homotopy continuation methods. This approach was proposed in [6] where it was employed to analyze the critical points of linear networks - networks with identity activation function.

In this thesis, we take a closer look at the specific case of linear networks with one hidden layer. In Chapter 1, we introduce the mathematical model of a single neuron as the building block of larger neural networks. We then examine the special case of linear networks and demonstrate, with examples, that the critical points of the loss function of a linear network are precisely the solutions to a polynomial system. We then provide motivation for analyzing this polynomial system, which we call the gradient polynomial system, using algebraic geometric methods.

In Chapter 2, we introduce basic concepts in algebraic geometry with the goal of better understanding the gradient polynomial system of a linear network. We cover well known results like the Classical Bezout Bound (Theorem 2.13) and the BKK bound (Theorem 2.22), that relate the monomial structure of a polynomial system to the number of its complex solutions. We outline homotopy continuation methods from computational algebraic geometry that use the preceding bounds to find the solutions to a polynomial system. In particular, we highlight the fact that knowing better upper bounds on the number of complex solutions leads to more efficient ways of solving the system.

In Chapter 3, we present our main results. For the 1-hidden layer linear network, we examine the case with one training example. For this case we note that the solutions (W_1, W_2) to the gradient polynomial system are very structured. More specifically, appearance of zeros in W_1 and W_2 follows a particular pattern (Propositions 3.1, 3.2, 3.3). In other words, the solutions with at least one zero entry lie on particular coordinate subspaces. For solutions with all non-zero entries, we give a new upper bound, $\mathcal{B}_{\mathbb{C}^*}$ (Proposition 3.6). Combining the preceding results, we give a general upper bound, $\mathcal{B}_{\mathbb{C}}$, on the total number of solutions to the gradient polynomial system. We demonstrate through experiments that the new bounds $\mathcal{B}_{\mathbb{C}^*}$ and $\mathcal{B}_{\mathbb{C}}$ are significantly smaller than the BKK bound for the gradient polynomial system (Table 3.1). Finally, we present experimental results for linear networks with more than one hidden layer and more than one training example.

Chapter 1

Neural Networks

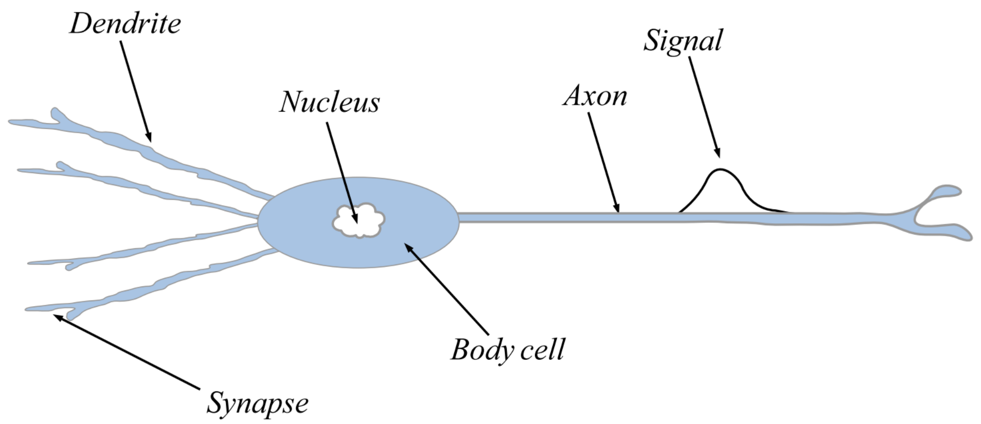
In this chapter, we examine how biologically-inspired mathematical models of neurons (and networks of neurons) are used to approximate functions. This process is often referred to as ”learning” or ”training”.

1.1 Biological Neuron

This section has been taken from [4] largely unmodified. A neuron is a cell which consists of the following parts: *dendrites*, the *axon*, and the *body-cell*, see Fig 1.1. The synapse is the connection between the axon of one neuron and the dendrite of another. The functions of each part is briefly described below:

- Dendrites are transmission channels that collect information from the axons of other neurons. This is described in Fig. 1.2 a). The signal traveling through an axon reaches

Figure 1.1: Anatomy of a biological neuron



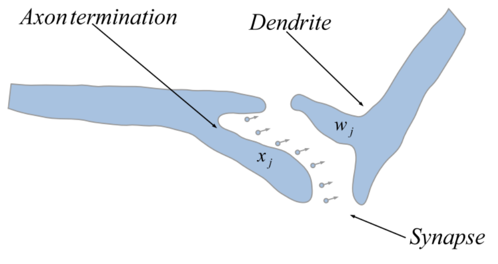
its terminal end and produces some chemicals which are liberated in the synaptic gap.

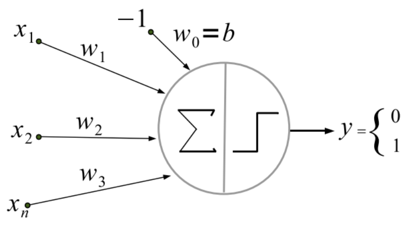
- The body-cell collects all signals received by its dendrites. Here the dendrites' activity adds up into a total potential and if a certain threshold is reached, the neuron fires a signal through the axon. The threshold depends on the sensitivity of the neuron and measures how easy it is to get the neuron to fire.
- The axon is the channel for signal propagation. The signal consists in the movement of ions from the body-cell towards the end of the axon. The signal is transmitted electrochemically to the dendrites of the next neuron, see Fig. 1.1.

1.2 Modeling the neuron

The biological neuron can be modelled mathematically as follows. Let x_j represent the signal received by the jth dendrite and let w_j represent the strength of the corresponding

Figure 1.2: Model of a single neuron





a **b**

a) Synaptic gap. b) Schematic representation of a neuron

synaptic connection (see Fig. 1.2 (a)). Then the total potential received by the neuron through all its dendrites is represented by the weighted sum $\sum_{j=1}^{n} w_j x_j$. Finally, if the total potential exceeds some threshold b, then the neuron outputs a signal 1, else it outputs a signal 0 (see Fig. 1.2(b)). This neuronal response can be represented as a function of the input as follows:

$$f_w(x_1, ..., x_n) = \phi\left(\sum_{j=1}^{n} w_j x_j - b\right) \tag{1.1}$$

where ϕ is the Heaviside function:

$$\phi(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{if } t \leq 0 \end{cases}$$

Renaming b to w_0 and appending $x_0 = -1$ to the input vector, (1.1) can be written as

$$f_w(x_0, ..., x_n) = \phi\left(\sum_{j=0}^{n} w_j x_j\right) \tag{1.2}$$

In the equation (1.2) above, f_w is called the *output function* of the neuron and ϕ is called the *activation function*.

We note that other activation functions can also be used to model the neuron output. Some examples are:

- The identity function: $\phi(x) = x$.
- The linear function: $\phi(x) = kx, k > 0$.
- The Rectified Linear Unit (ReLU) function: $\phi(x) = max\{x, 0\}$

As can be seen above, the activation function of choice may be continuous, discontinuous, differentiable or non-differentiable.

How does a neuron learn?

Given a target function y, our goal is to approximate it "as closely as possible" using the neuron's output function $f_w : \mathbb{R}^n \longrightarrow \mathbb{R}$ as defined in (1.2). In practice, the target function y is rarely known. Instead, we typically only have access to the values that y takes on a finite set of inputs. This finite set of inputs and the corresponding set of y values can be represented in matrix notation as follows

$$\begin{aligned} X &= \left[x^{(1)}, ..., x^{(m)}\right] \in \mathbb{R}^{n \times m} \\ Y &= \left[y^{(1)}, ..., y^{(m)}\right] \in \mathbb{R}^{p \times m} \end{aligned} \tag{1.3}$$

where $x^{(i)} \in \mathbb{R}^n$ are input points and $y^{(i)} \in \mathbb{R}^p$ are corresponding values of the target function (with $p = 1$ for scalar valued y). Then (X, Y) is called a *training set* and each $(x^{(i)}, y^{(i)})$ is called a *training example.*

Now, consider the function $\mathcal{L} : \mathbb{R}^n \longrightarrow \mathbb{R}$ that represents the square of the distance (in an L^2 sense) between f_w and y over the finite set X:

$$\begin{aligned}\mathcal{L}(w) &= \frac{1}{2}\sum_{i=1}^{m} \left(f_w(x^{(i)}) - y^{(i)}\right)^2 \\ &= \frac{1}{2}\sum_{i=1}^{m} \left(\phi\left(\sum_{j=0}^{n} w_j x_j\right) - y^{(i)}\right)^2\end{aligned} \tag{1.4}$$

We call $\mathcal{L}$ the *loss function*. Our goal is to find the weights $w \in \mathbb{R}^n$ that minimize $\mathcal{L}$. So, training the neuron is equivalent to solving the following unconstrained optimization problem:

$$\min_{w \in \mathbb{R}^n} \; \mathcal{L}(w) \tag{1.5}$$

A single neuron can only learn simple functions. In order to learn more complicated functions, we need a network of neurons working together. Such networks will be the subject of discussion in the next section.

1.3 Neural Networks

Neural Networks are grid-like structures comprising consecutive layers of neurons (see Fig. 1.3). Each layer receives some input and produces an output which is passed on as input to the next layer. Finally, the output of the last layer is considered the output of the entire network. As a matter of convention, the last layer of the network is called the *output layer*, while all preceding layers are called *hidden layers*.

Let us consider an example of a simple 2-layer network comprising a total of three neurons.

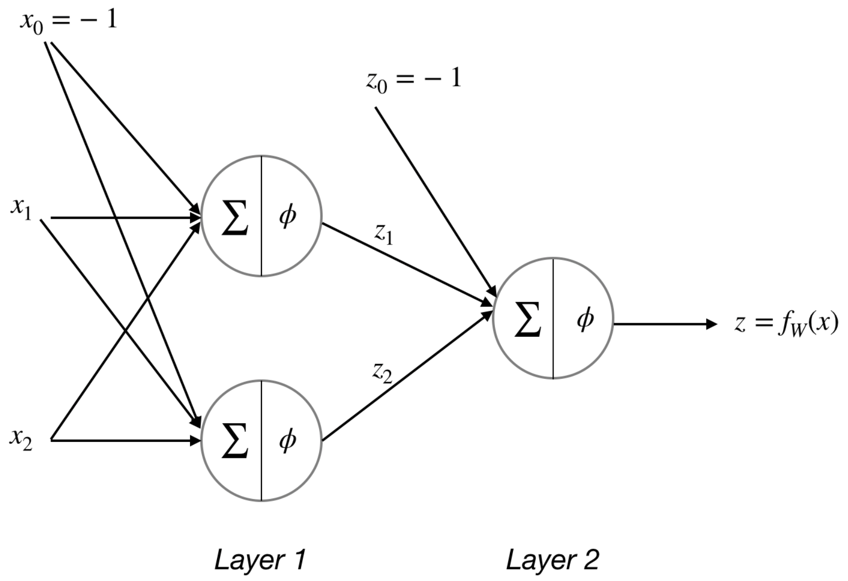


Figure 1.3: Neural Network Schematic

The network shown in Fig. 1.3 has one hidden layer and one output layer. The hidden layer contains two neurons while the output layer contains a single neuron. As we have seen in Section 1.2, each neuron has some weights associated with it and produces an output according to equation (1.2). Let w_{ijk} denote the kth weight associated with the jth neuron of the ith layer. Then, the first neuron of the first layer receives the vector $(-1, x_1, x_2)$ as its input and produces the scalar output

$$z_1 = \phi\left(\sum_{k=0}^{2} w_{11k}x_k\right)$$

The second neuron of the first layer also receives $(-1, x_1, x_2)$ as its input and produces the scalar output

$$z_2 = \phi\left(\sum_{k=0}^{2} w_{12k} x_k\right)$$

Finally, the only neuron in the output layer receives $(-1, z_1, z_2)$ as its input and, in turn, produces the scalar output

$$z = \phi\left(\sum_{k=0}^{2} w_{21k} z_k\right)$$

In order to write the above equations more succinctly, we introduce some notation.

Definition 1.1 (weight matrix of the ith layer)**.** Suppose the ith layer of a neural network contains r neurons, each neuron containing n weights. Then we define the weight matrix associated with the ith layer to be the matrix

$$\begin{pmatrix} w_{i11} & \cdots & w_{i1n} \\ \vdots & & \vdots \\ w_{ir1} & \cdots & w_{irn} \end{pmatrix}$$

where w_{ijk} denotes the kth weight of the jth neuron of the ith layer.

Now, suppose our network has H hidden layers and one output layer. Let W_i denote the weight matrix of the ith layer with $i = 1, ..., H$ for the hidden layers and $i = H + 1$ for the output layer. Further, let $z^{(i)}$ denote the output of the ith layer with the convention that $z^{(0)} = (-1, x)$, where $x \in \mathbb{R}^n$ is the input vector. Now we can recursively describe the output of each layer as:

$$z^{(i)} = \phi\Big(W_i \begin{bmatrix} -1 \\ \\ z^{(i-1)} \end{bmatrix}\Big)$$

where ϕ is applied component-wise. The output of the entire network as a function of the input x is given by:

$$f_W(x) = z^{(H+1)} = \phi\Big(W_{H+1} \begin{bmatrix} -1 \\ \\ z^{(H)} \end{bmatrix}\Big) \tag{1.6}$$

In practice, the same activation function is applied to all neurons in a network. We will adopt the same convention in this work.

1.4 Special case: linear networks

In order to keep things mathematically tractable, we introduce two simplifications in (1.6). First, we drop the bias term, i.e. we set $w_0 = 0$ in (1.2) for all neurons in the network. Doing so simplifies the form of the output function to:

$$f_W(x) = z^{(H+1)} = \phi\left(W_{H+1} z^{(H)}\right) \tag{1.7}$$

where ϕ is applied component wise. Next, we choose our activation function to be the identity function, $\phi(x) = x$. This further reduces (1.7) to

$$f_W(x) = W_{H+1} W_H \cdots W_2 W_1 x \tag{1.8}$$

The corresponding loss function is then given by:

$$\begin{aligned}\mathcal{L}(W) &= \frac{1}{2}\sum_{i=1}^{m} ||f_W(x^{(i)}) - y^{(i)}||^2 \\ &= \frac{1}{2}\sum_{i=1}^{m} ||W_{H+1}W_H...W_2W_1x^{(i)} - y^{(i)}||^2\end{aligned} \tag{1.9}$$

where $x^{(i)} \in X, y^{(i)} \in Y$.

We must note that the above mathematical tractability comes at a cost. Choosing $\phi = \mathbf{I}$ makes the output function $f_W(x)$ linear in x, which means our network may not be able to fit non-linear functions well. However, $\mathcal{L}(W)$ still remains non-convex in the entries of W, i.e. the non-convexity of the loss function - which is an important issue in practice - is still retained. As a result, we can still hope to make non-trivial observations about the landscape of the loss function. This makes $\phi = \mathbf{I}$ a compelling choice. We will make a final remark about our choice of the activation function at the end of Chapter 2.

1.5 Critical points of $\mathcal{L}$ as solutions to polynomial systems

Training a linear network is equivalent to finding values of the weights W that minimize the loss function given in (1.9). Since $\mathcal{L}$ is not convex, a unique minimum is not guaranteed. So, we would like to find all minima of $\mathcal{L}$. Instead, we first aim to find the set of all critical points of $\mathcal{L}$. A naive formulation of the problem we want to solve is:

$$\nabla\mathcal{L}(W) = 0 \tag{1.10}$$

Suppose

$$W_i = \begin{pmatrix} w_{i11} & w_{i12} & \dots & w_{i1n} \\ \vdots & \vdots & & \\ w_{im1} & w_{im2} & \dots & w_{imn} \end{pmatrix}$$

Define

$$\frac{\partial \mathcal{L}}{\partial W_i} \triangleq \begin{pmatrix} \frac{\partial \mathcal{L}}{\partial w_{i11}} & \frac{\partial \mathcal{L}}{\partial w_{i12}} & \dots & \frac{\partial \mathcal{L}}{\partial w_{i1n}} \\ \vdots & \vdots & \vdots & \\ \frac{\partial \mathcal{L}}{\partial w_{im1}} & \frac{\partial \mathcal{L}}{\partial w_{im2}} & \dots & \frac{\partial \mathcal{L}}{\partial w_{imn}} \end{pmatrix}$$

Then (1.10) is equivalent to the below system of equations:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial W_1} &= 0 \\ &\vdots \\ \frac{\partial \mathcal{L}}{\partial W_{H+1}} &= 0 \end{aligned} \tag{1.11}$$

As we will see shortly, (1.11) turns out to be a polynomial system. Note that (1.11) can be written equivalently as follows (see [6], page 3, equation (3)):

$$\frac{\partial \mathcal{L}}{\partial W_i} = U_i^T \left[W \left(\sum_{k=1}^{m} x^{(k)} x^{(k)T} \right) - \left(\sum_{k=1}^{m} y^{(k)} x^{(k)T} \right) \right] V_i^T \tag{1.12}$$

where $U_i^T = \prod_{j=i+1}^{H+1} W_j^T$, $V_i^T = \prod_{j=1}^{i-1} W_j^T$ for $i = 1, ..., H+1$ and $W = W_{H+1} ... W_1$. Let us look at a simple example.

Example 1.2 (A 4-weight network). Let us consider a 2-layer network with $W_1 = [\alpha_1, \alpha_2]$,

$W_2 = \left[\begin{smallmatrix}\beta_1\\ \beta_2\end{smallmatrix}\right]$, $X = \left(\begin{smallmatrix}1 & 2\\ 3 & 4\end{smallmatrix}\right)$, $Y = \left(\begin{smallmatrix}1 & 3\\ 2 & 4\end{smallmatrix}\right)$. Then substituting these values in (1.12), we get:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial W1} &= [\beta_1, \beta_2]\left[\begin{pmatrix}\alpha_1\beta_1 & \alpha_2\beta_1\\ \alpha_1\beta_2 & \alpha_2\beta_2\end{pmatrix}\begin{pmatrix}5 & 11\\ 11 & 25\end{pmatrix} - \begin{pmatrix}7 & 15\\ 10 & 22\end{pmatrix}\right]\\
&= [5\alpha_1\beta_1^2 + 5\alpha_1\beta_2^2 + 11\alpha_2\beta_1^2 + 11\alpha_2\beta_2^2 - 7\beta_1 - 10\beta_2,\\
&\qquad 11\alpha_1\beta_1^2 + 11\alpha_1\beta_2^2 + 25\alpha_2\beta_1^2 + 25\alpha_2\beta_2^2 - 15\beta_1 - 22\beta_2]\\
\frac{\partial \mathcal{L}}{\partial W2} &= \left[\begin{pmatrix}\alpha_1\beta_1 & \alpha_2\beta_1\\ \alpha_1\beta_2 & \alpha_2\beta_2\end{pmatrix}\begin{pmatrix}5 & 11\\ 11 & 25\end{pmatrix} - \begin{pmatrix}7 & 15\\ 10 & 22\end{pmatrix}\right]\begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix}\\
&= \begin{pmatrix}5\alpha_1^2\beta_1 + 22\alpha_1\alpha_2\beta_1 + 25\alpha_2^2\beta_1 - 7\alpha_1 - 15\alpha_2\\ 5\alpha_1^2\beta_2 + 22\alpha_1\alpha_2\beta_2 + 25\alpha_2^2\beta_2 - 10\alpha_1 - 22\alpha_2\end{pmatrix}
\end{aligned}$$

Substituting these expressions in (1.11), we get the below system:

$$\begin{aligned}
5\alpha_1\beta_1^2 + 5\alpha_1\beta_2^2 + 11\alpha_2\beta_1^2 + 11\alpha_2\beta_2^2 - 7\beta_1 - 10\beta_2 &= 0\\
11\alpha_1\beta_1^2 + 11\alpha_1\beta_2^2 + 25\alpha_2\beta_1^2 + 25\alpha_2\beta_2^2 - 15\beta_1 - 22\beta_2 &= 0\\
5\alpha_1^2\beta_1 + 22\alpha_1\alpha_2\beta_1 + 25\alpha_2^2\beta_1 - 7\alpha_1 - 15\alpha_2 &= 0\\
5\alpha_1^2\beta_2 + 22\alpha_1\alpha_2\beta_2 + 25\alpha_2^2\beta_2 - 10\alpha_1 - 22\alpha_2 &= 0
\end{aligned} \tag{1.13}$$

Thus we see that for our example, solving (1.10) is equivalent to solving the polynomial system (1.13).

For convenience, we will refer to polynomial systems arising out of the gradient equations of linear networks as *gradient polynomial systems*. We will have more to say about such systems in Chapter 2. For now, we make a few observations:

- (1.13) is a square system with four equations in four unknowns, viz. $\alpha_1, \alpha_2, \beta_1, \beta_2$. Squareness of the gradient polynomial systems holds true for linear networks in general.

- Each polynomial has degree 3. In general, the degree of each polynomial in a gradient polynomial system is given by $2H + 1$, where H is the number of hidden layers. In our case $H = 1$.

- Typically, the weights W of a linear network are assumed to be real. However, for solving a polynomial system like (1.13), it is often necessary to consider polynomials over $\mathbb{C}$, i.e. polynomials with complex coefficients and roots. We can always select the real solutions (if any) from the complex solutions of a polynomial system.

1.6 Motivation for an algebraic geometric view

In real analysis, there exist well understood and efficient methods that can be used to solve (1.10) algorithmically. However, this approach has some limitations:

1. Since $\mathcal{L}$ is not convex, it can often be hard to guarantee that the algorithm has found all minima.

2. It is hard to make precise assertions about the number and location of critical points beforehand, i.e. without actually solving the system.

Viewing (1.10) as a system of polynomial equations allows us to do away with the above limitations. As we will learn in Chapter 2, we can employ homotopy continuation methods from algebraic geometry to provably find all critical points of $\mathcal{L}$. Moreover, as we will learn in Chapter 3, we can make precise assertions about the number and locations of critical points of $\mathcal{L}$ for linear networks of modest size. We must note that applying algebraic geometric methods to solve 1.10 requires us to consider issues like singular and positive dimensional solutions. We will outline existing ways to tackle these issues in Chapter 3.

In order to better understand the above claims, we need to first lay some groundwork. We do this by introducing some fundamental algebraic geometry concepts in the following chapter.

Chapter 2

Algebraic Geometry Essentials

Algebraic geometry is concerned with systems of polynomial equations and their solution sets. In the following sections, we define some basic terms and use examples to illustrate the main concepts. In later chapters, we will use these concepts to analyze and solve polynomial systems arising from the training of deep linear networks. We begin by looking at some simple examples of systems of polynomials.

2.1 Polynomial Systems

Example 2.1 (real isolated solutions). Consider the polynomials $z^2 - 1$ and $w^2 - 4$ in $\mathbb{C}[z, w]$. Then

$$\begin{bmatrix} z^2 - 1 \\ w^2 - 4 \end{bmatrix} = 0$$

is a polynomial system with four real **isolated** solutions viz. $(1,2),(-1,2),(1,-2),(-1,-2)$ in $\mathbb{C}^2$.

Example 2.2 (mixed isolated solutions). Consider the system

$$\begin{bmatrix} z^3-1 \\ w^2-4 \end{bmatrix} = 0$$

The above system has two real isolated solutions $(1,2)$, $(1,-2)$ and four non-real isolated solutions $(\frac{-1}{2}+i\frac{\sqrt{3}}{2},2)$, $(\frac{-1}{2}+i\frac{\sqrt{3}}{2},-2)$, $(\frac{-1}{2}-i\frac{\sqrt{3}}{2},2)$, $(\frac{-1}{2}-i\frac{\sqrt{3}}{2},-2)$.

Example 2.3 (positive dimensional solutions). Consider the system

$$\begin{bmatrix} zw \\ z(w-1) \end{bmatrix} = 0$$

Note that the complex line $z=0$ (i.e. the w-axis) is a non-isolated, or **positive dimensional**, solution to the above polynomial system. In fact, it is the only solution to the above system.

Example 2.4 (isolated and non-isolated solutions). Unlike linear systems, polynomial systems can have solutions with different dimensions. Consider the below system

$$\begin{bmatrix} z(z-1) \\ z(w-1) \end{bmatrix} = 0$$

The above system has a (real) isolated solution, viz. $(1,1)$ and a positive dimensional (complex) solution, viz. the complex line $z=0$. In fact, $z=0$ is a **singular solution** [1] to the above system.

[1] A singular solution to a polynomial system is one over which the Jacobian is not full-rank. See [1] for more details.

Example 2.5 (multiplicity > 1). Consider the system

$$\begin{bmatrix} z^2 w \\ \\ z^2(w-1) \end{bmatrix} = 0$$

This system has the same solutions as the one in Example 2.3. However, in this case, the complex line $z = 0$ is a solution with **multiplicity** 2. [2]

We note in passing that solving polynomial systems with positive dimensional solutions, singular solutions or solutions with multiplicities greater than 1 presents numerical challenges. However, we will only concern ourselves with polynomial systems that have isolated, non-singular solutions with multiplicity one. We are now ready to make things a bit more formal.

Definition 2.6 (Algebraic set)**.** An algebraic set is the set of common solutions to a system of polynomial equations.

Let $f_1, ..., f_N \in \mathbb{C}[x_1, ..., x_N]$. Then the algebraic set associated with the system

$$\begin{bmatrix} f_1(x) \\ \vdots \\ f_N(x) \end{bmatrix} = 0$$

is the set of points in $\mathbb{C}^N$ at which every one of $f_1, ..., f_N$ vanishes. Note that the above system is a "square" system since the number of equations equals the number of indeterminates,

[2]because every point on $z = 0$ is a root with multiplicity 2 for both polynomials.

viz N. Unless stated otherwise, by a "polynomial system", we mean a "square polynomial system". For details on non-square systems, we refer the interested reader to [1].

We will now examine how the structure of a polynomial relates to the number of its roots. First, we introduce some terminology and notation. Consider the polynomial $f \in \mathbb{C}[x_1, x_2]$ given by

$$f(x_1, x_2) = c_1 x_1^0 x_2^1 + c_2 x_1^1 x_2^2$$

The polynomial f is a linear combination of two monomials, viz. $x_1^0 x_2^1$ and $x_1^1 x_2^2$, that can be identified by the points $(0, 1)$ and $(1, 2)$ respectively, in the lattice $\mathbb{Z}^2$. Let $\mathcal{A} = \{(0, 1), (1, 2)\}$. We say that the set $\mathcal{A}$ is the **monomial support** of the polynomial f. We can then express f more succinctly as

$$f(x) = \sum_{a \in \mathcal{A}} c_a x^a \tag{2.1}$$

where $\mathcal{A}$ is a fixed finite subset of $\mathbb{Z}^N_{\geq 0}$.

Definition 2.7 (dense polynomial)**.** Let f be a degree-d polynomial. We say that f is dense (equivalently, has full monomial support) if every possible monomial term of degree d or less appears in f.

Definition 2.8 (sparse polynomial)**.** We say a polynomial is sparse if it is not dense. In this case, we say that the polynomial does not have full monomial support.

Definition 2.9 (Zariski closed set)**.** Zariski closed sets are sets of the form $\bigcap_{i=1}^{K} V(f_i)$ where $V(f_i) = \{x \in \mathbb{C}^m : f_i(x) = 0\}$ with $f_i \in \mathbb{C}[x_1, ..., x_m]$ and $i = 1, ..., K$.

Note that Zariski closed sets are just algebraic sets. They are so called because they give rise to the Zariski topology which plays an important role in algebraic geometry. An important fact about Zariski closed sets is that they have zero Lebesgue measure [3]. We will make use of this fact in the upcoming results.

Definition 2.10 (Generic)**.** A polynomial system F is said to be generic with respect to a property P if and only if the set U (in the coefficient space) for which P fails to hold lies inside some Zariski closed set (in the coefficient space). Moreover, the coefficients in U^c are said to be generic with respect to P, or simply generic when P is obvious from the context.

We illustrate with an example. Consider the single polynomial system $F(x) = ax^2 + bx + c = 0$. Consider the property $P =$ "has two distinct roots". We know from the fundamental theorem of algebra that P fails to hold if and only if $b^2 - 4ac = 0$, and in this case F has exactly one root (with multiplicity 2). In other words, the set U on which P fails to hold is given by $U = \{(a, b, c) \in \mathbb{C}^3 : b^2 - 4ac = 0\}$. Note that this set is a Zariski closed set, viz. $U = V(b^2 - 4ac)$. Therefore we say that F is generic with respect to two distinct roots, or, equivalently, that F has two distinct roots for generic coefficients.

Remark 2.11. We use the statements "F is generic with respect to a property P" and "F has property P for generic coefficients" interchangeably.

Remark 2.12. Randomly chosen coefficients are generic with probability one. Suppose we define a probability space[7] on the coefficient space $\mathbb{C}^m$ of a polynomial system. Then

[3]Zariski closed sets form a lower-dimensional hyper-surface (see [1]).

we can define a probability density with respect to the Lebesgue measure on $\mathbb{C}^m$. It is then necessarily true that a zero Lebesgue measure set in $\mathbb{C}^m$ has zero probability measure. Now suppose we choose a point in $\mathbb{C}^m$ at random with respect to the defined density. The probability that the chosen point lies inside the pathological set U is zero because U is a subset of some zero-measure Zariski closed set in $\mathbb{C}^m$. Hence, U has probability measure zero. Therefore, a randomly chosen point lies outside U (equivalently, is generic) with probability one.

Now we state a classical theorem that connects the structure of polynomial systems to the number of their solutions.

Theorem 2.13 (Classical Bezout Bound). *Let $f_1, ..., f_N$ be polynomials in $\mathbb{C}[x_1, ..., x_N]$. Then the number of isolated solutions of the system $f_1(x) = ... = f_N(x) = 0$ is bounded above by the product $deg(f_1) \cdots deg(f_N)$.*

Note that the Classical Bezout Bound (or the CBB) in Theorem 2.13 gives an upper bound for arbitrary polynomials $f_1, ..., f_N$. However, if we assume the polynomials to be dense with generic coefficients, then this upper bound is attained.

Corollary 2.13.1. *Suppose $f_1, ..., f_N$ are dense polynomials in $\mathbb{C}[x_1, ..., x_N]$ with generic coefficients. Then the number of isolated solutions of the system $f_1(x) = ... = f_N(x) = 0$ equals the CBB.*

Remark 2.14. We note that it is not necessary for a polynomial system to be dense in order for the CBB to be attained. There are sparse systems for which the number of solutions equals the CBB (see example 2.1 and 2.2). As a matter of fact, these sparse systems have a special structure which accounts for the CBB being attained. We will learn more about this structure in Section 2.3. For now, we note that, in general, for sparse systems, CBB is a strict upper bound.

Example 2.15. [CBB is not attained] Recall the sparse gradient polynomial system (1.13). The CBB for this system is $3 \cdot 3 \cdot 3 \cdot 3 = 81$. However, this system has just 5 isolated solutions including the origin (see appendix .1 for solutions obtained using [3]).

So far we have made observations about the structure of a polynomial system and the number of its solutions. In the next section we turn to the problem of actually finding these solutions.

2.2 Homotopy Continuation

The main idea in solving a system A of polynomial equations is to find a "similar" system B with known solutions and then continuously deform B and its solutions to A and its solutions. This approach is called *homotopy continuation* and the continuous deformation is achieved via a parameterized family of equations called a *homotopy*. We make this idea more precise.

Let $f_1, ..., f_N$ be polynomials in $\mathbb{C}[x_1, ..., x_N]$. Suppose we want to solve the system

$$F(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_N(x) \end{bmatrix} = 0$$

Assume we know another polynomial system

$$G(x) = \begin{bmatrix} g_1(x) \\ \vdots \\ g_N(x) \end{bmatrix} = 0$$

which has at least as many isolated solutions as $F(x) = 0$, and that the solutions to $G(x) = 0$ are known to us beforehand. Then we can define a homotopy as follows:

$$H(x,t) = tG(x) + (1-t)F(x), \quad t \in [0,1] \tag{2.2}$$

Note that $H(x,t)$ yields a new polynomial system for each value of t, with $H(x, t = 1) \equiv G(x)$ and $H(x, t = 0) \equiv F(x)$. The homotopy described in 2.2, is called a *total degree homotopy* with $G(x) = 0$ as the *start system* and $F(x) = 0$ as the *target system.*

The key observation is that, as t goes from 1 to 0, the solutions of $H(x,t) = 0$ lie on continuous paths that connect the solutions of the start system to the solutions of the target system (see Fig. 2.1 below). Therefore, the task of solving $F(x) = 0$ can be reduced to that of tracking the paths from the solutions of the start system to the solutions of target system as t goes from 1 to 0.

Solving a polynomial system efficiently amounts to choosing a "good" start system. Roughly speaking, a good start system is one for which the number of isolated solutions is "as close as possible" to the number of isolated solutions of the target system. Now, we do not know the exact number of isolated solutions to the target system beforehand, so we choose an upper bound instead, when selecting a start system. A tighter upper bound implies a better start system in that there are fewer paths to track.

A naive choice of upper bound would be the CBB. However, as we have seen, CBB often dramatically overstates the number of isolated solutions to a system. As it turns out, we can often select a better bound than the CBB, leading to more efficient solutions. This will be the topic of investigation in the next section.

2.3 Sparse Systems and the BKK bound

Polynomial systems arising out of linear neural networks are sparse, like the one in (1.13). For such systems, we can place a tighter bound than the CBB on the number of solutions. This bound, known as the Bernstein–Khovanskii– Kushnirenko bound (or the BKK bound) allows us to solve sparse systems more efficiently by using more accurate start systems for homotopy continuation. Before we state the main result of this section, we need some preliminary definitions.

Definition 2.16 (dense and sparse systems)**.** We say that the polynomial system $f_1(x) = ... = f_N(x) = 0$ is dense if f_i is a dense polynomial for each $i = 1, ..., N$. Otherwise, we say

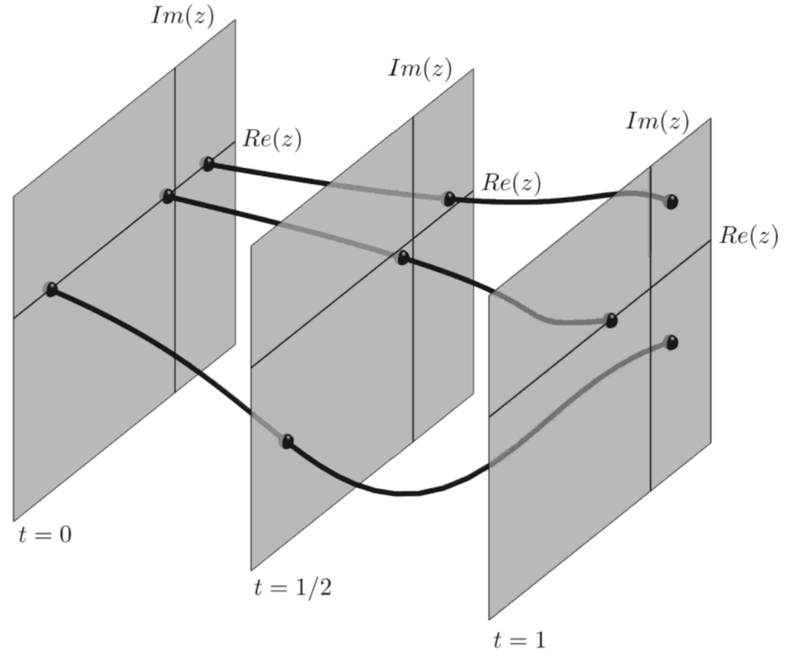


Figure 2.1: Solution paths sliced at three different values of t [1].

that the system is sparse.

Definition 2.17 (Newton polytope)**.** Let f be a polynomial with monomial support $\mathcal{A}$. The Newton polytope Q of f is defined to be

$$Q = conv(\mathcal{A})$$

where $conv(\mathcal{A})$ denotes the convex hull of $\mathcal{A}$ in $\mathbb{R}^N$.

Definition 2.18 (Scaled Minkowski sum of polytopes)**.** The scaled Minkowski sum of the polytopes $Q_1, ..., Q_N \subset \mathbb{R}^N$ is the set

$$\lambda_1 Q_1 + ... + \lambda_N Q_N = \{\lambda_1 x_1 + ... + \lambda_N x_N \in \mathbb{R}^N : x_i \in Q_i\}$$

where $\lambda_i \geq 0$.

Note that $\lambda_1 Q_1 + ... + \lambda_N Q_N$ is also a polytope in $\mathbb{R}^N$ and therefore has the usual Euclidean volume associated with it. Let us denote this volume by $vol(\lambda_1 Q_1 + ... + \lambda_N Q_N)$. For fixed polytopes $Q_1, ..., Q_N$ we can think of $vol(\lambda_1 Q_1 + ... + \lambda_N Q_N)$ as a function of λ_i. It is a known result in convex geometry that $vol(\lambda_1 Q_1 + ... + \lambda_N Q_N)$ is a homogeneous, degree-N polynomial in $\lambda_1, ..., \lambda_N$ [5]. We use this fact in the next definition.

Definition 2.19 (Mixed Volume)**.** The mixed volume of polytopes $Q_1, ..., Q_N$, denoted $\mathcal{M}(Q_1, ..., Q_N)$, is defined to be the coefficient of $\lambda_1 \cdots \lambda_N$ in the polynomial $vol(\lambda_1 Q_1 + ... + \lambda_N Q_N)$.

Definition 2.20 (Laurent polynomial)**.** If the monomials in a polynomial are allowed to have negative integral exponents, then the polynomial is called a Laurent polynomial. More precisely, a Laurent polynomial is of the form

$$f(x) = \sum_{a \in \mathcal{A}} c_a x^a$$

where $\mathcal{A}$ is a fixed finite subset of $\mathbb{Z}^N$.

Definition 2.21 ($\mathbb{C}^*$)**.** We define $\mathbb{C}^*$ to be the set of non-zero complex numbers, $\mathbb{C} \backslash \{0\}$.

We are now ready to state the main result of this chapter - the celebrated theorem due to Bernstein [2].

Theorem 2.22 (Bernstein's Theorem). *Let $f_1, ..., f_N \in \mathbb{C}[x_1, ..., x_N]$ be Laurent polynomials with Newton polytopes $Q_1, ..., Q_N$. The number of isolated solutions of the system $f_1(x) = ... = f_N(x) = 0$ in $(\mathbb{C}^*)^N$ is bounded above by the mixed volume $\mathcal{M}(Q_1, ..., Q_N)$.*

The upper bound given in Theorem 2.22 is called the Bernstein–Khovanskii–Kushnirenko bound or the **BKK bound**.

Corollary 2.22.1. *For generic coefficients, the BKK bound is attained.*

Proofs of Theorem 2.22 and Corollary 2.22.1 can be found in [2]. We make a few important observations here. First, every regular polynomial is also a Laurent polynomial. So, the BKK bound applies to regular polynomial systems. Second, in the case of sparse polynomial systems, the BKK bound is often much smaller than the CBB ([6], Table 1). Consequently, we can use much smaller start systems for homotopy continuation, resulting in significantly fewer paths to track [5].

Before moving on to the next chapter we make a final remark about our choice of the activation function. Recall that in Section 1.4, we chose $\phi = \mathbf{I}$ as our activation function. This choice allowed us to treat $\nabla\mathcal{L}(W) = 0$ as a polynomial system. We note that choosing ϕ to be a polynomial would also result in $\nabla\mathcal{L}(W) = 0$ being a polynomial system (see 1.7). However, the constituent polynomials would be of degree greater than $2H + 1$. This would result in larger Newton polytopes and solving the system would incur a greater computational cost. In the spirit of understanding smaller systems well before analyzing larger systems, and to not get distracted by computational issues, we choose to stick with $\phi = \mathbf{I}$.[4]

In the upcoming Chapter 3, we apply the algebraic geometry machinery developed in this chapter to gradient polynomial systems introduced in Chapter 1 and present the main results of our investigation.

[4]We must also note that other activation functions, like $ReLU$, can be approximated by polynomials. However, doing so will result in the same challenges as mentioned above.

Chapter 3

Main Results

We now take a closer look at gradient polynomial systems associated with deep linear networks. More specifically, we revisit the limitations posed at the end of Section 1.6.

First, we outline known methods and results for removing singular and positive dimensional solutions, as well as for provably finding all isolated solutions of the gradient polynomial systems. Then, we offer new insights into the geometry of the solutions of these systems. These insights allow us to tighten the currently known upper bounds on the number of complex critical points, as well as make *a priori* assertions about the locations of these critical points in $\mathbb{C}^N$.

Recall that for a linear network with H hidden layers, the weight matrices are denoted W_i with $i = 1, ..., H$ for the hidden layers and $i = H + 1$ for the output layer. The training set (X, Y) has m examples with input dimension d_x and output dimension d_y. We assume the training set to be real, i.e. $X \in \mathbb{R}^{d_x \times m}$ and $Y \in \mathbb{R}^{d_y \times m}$.

As noted at the end of Section 1.4, in order to view the gradient equation (1.10) as a polynomial system, we need to consider the weights W to be complex. Let d_i denote the number of neurons in layer i. Then matrix multiplication yields $W_1 \in \mathbb{C}^{d_1 \times d_x}$, $W_2 \in \mathbb{C}^{d_2 \times d_1}$, ..., $W_{H+1} \in \mathbb{C}^{d_y \times d_H}$. Moreover, the total number of weights in the network is given by $N = d_1 d_x + d_2 d_1 + ... + d_y d_H$. Note that this is also the total number of variables (and equations) in the corresponding polynomial system.

3.1 Eliminating singular and positive dimensional solutions

As mentioned in Section 1.6 and in Section 2.1, polynomial systems can have singular as well as positive dimensional solutions. Working with such solutions present numerical challenges. A particular technique that guarantees the elimination of singular and positive dimensional solutions is called Tikhonov regularization (see [6]). The Tikhonov-regularized gradient polynomial system is given by

$$\frac{\partial \mathcal{L}}{\partial W_i} = U_i^T \left[W \Big(\sum_{k=1}^{m} x^{(k)} x^{(k)T} \Big) - \Big(\sum_{k=1}^{m} y^{(k)} x^{(k)T} \Big) \right] V_i^T + \Lambda_i \circ W_i = 0 \tag{3.1}$$

for $i = 1, ..., H+1$, where Λ_i is a matrix of the same size as W_i with entries chosen uniformly from $[0,1]$, and $\Lambda_i \circ W_i$ denotes the Hadamard (entrywise) product. All gradient polynomial systems in the upcoming results will be Tikhonov-regularized.

3.2 Finding all critical points

Homotopy continuation methods outlined in Section 2.2 guarantee that all isolated solutions of the target systems are found. However, we recall that the success of these methods depends critically on the choice of the start system. Therefore, knowing the exact number of solutions to the target system, or even a good upper bound is extremely important.

3.3 Exploring the geometry of complex critical points

As noted in Chapter 2, the BKK bound gives an upper bound on the number of solutions in $(C^*)^N$ for a polynomial system. In the experimentally generated gradient polynomial systems, we observed that the actual number of solutions was much less than the BKK bound (see Table 3.1). Moreover, most solutions did not lie in $(C^*)^N$ for larger systems. Instead they seem to lie on coordinate subspaces (where at least one of the variables is zero), and so the BKK bound does not apply to such solutions. With the preceding observations as motivation to understand the geometry of the solutions, we now examine the gradient polynomial systems for small linear networks.

Networks with $H = 1$ and $m = 1$

For $H = 1, m = 1$, equation 3.1 yields the following system

$$W_2^T \left(W x x^T - y x^T \right) + \Lambda \circ W_1 = 0 \tag{3.2}$$

$$\left(Wxx^T - yx^T\right) W_1^T + \mu \circ W_2 = 0 \tag{3.3}$$

where $W = W_2 W_1$ and Λ, μ are Tikhonov regularization matrices of appropriate size. Let $\Lambda = \begin{pmatrix} \lambda_{1,1} & \cdots & \lambda_{1,n} \\ \vdots & & \vdots \\ \lambda_{d,1} & \cdots & \lambda_{d,n} \end{pmatrix}$ and $\mu = \begin{pmatrix} \mu_{1,1} & \cdots & \mu_{1,d} \\ \vdots & & \vdots \\ \mu_{p,1} & \cdots & \mu_{p,d} \end{pmatrix}$. Note that Λ, μ are generically chosen, constant matrices. We will write (3.2) and (3.3) in their equivalent polynomial system form to make it easier to analyze them. Note that

$$W = W_2 W_1 = \begin{pmatrix} \sum_{i=1}^d b_{1,i} a_{i,1} & \cdots & \sum_{i=1}^d b_{1,i} a_{i,n} \\ \vdots & & \vdots \\ \sum_{i=1}^d b_{p,i} a_{i,1} & \cdots & \sum_{i=1}^d b_{p,i} a_{i,n} \end{pmatrix}$$

So,

$$Wxx^T = \begin{pmatrix} \sum_{i=1}^d b_{1,i} a_{i,1} & \cdots & \sum_{i=1}^d b_{1,i} a_{i,n} \\ \vdots & & \vdots \\ \sum_{i=1}^d b_{p,i} a_{i,1} & \cdots & \sum_{i=1}^d b_{p,i} a_{i,n} \end{pmatrix} \begin{pmatrix} x_1 x_1 & \cdots & x_1 x_n \\ \vdots & & \vdots \\ x_n x_1 & \cdots & x_n x_n \end{pmatrix}$$

$$= \begin{pmatrix} \left[\sum_{i=1}^d b_{1,i} \left(\sum_{j=1}^n a_{i,j} x_j\right)\right] x_1 & \cdots & \left[\sum_{i=1}^d b_{1,i} \left(\sum_{j=1}^n a_{i,j} x_j\right)\right] x_n \\ \vdots & & \vdots \\ \left[\sum_{i=1}^d b_{p,i} \left(\sum_{j=1}^n a_{i,j} x_j\right)\right] x_1 & \cdots & \left[\sum_{i=1}^d b_{p,i} \left(\sum_{j=1}^n a_{i,j} x_j\right)\right] x_n \end{pmatrix}$$

Let $A_i := \sum_{j=1}^n a_{i,j} x_j$ for $i = 1, ..., d$. Then we can write

$$Wxx^T = \begin{pmatrix} \left(\sum_{i=1}^d b_{1,i} A_i\right) x_1 & \cdots & \left(\sum_{i=1}^d b_{1,i} A_i\right) x_n \\ \vdots & & \vdots \\ \left(\sum_{i=1}^d b_{p,i} A_d\right) x_1 & \cdots & \left(\sum_{i=1}^d b_{p,i} A_d\right) x_n \end{pmatrix}$$

So,

$$Wxx^T - yx^T = \begin{pmatrix} \left[\left(\sum_{i=1}^d b_{1,i}A_i\right) - y_1\right] x_1 & \cdots & \left[\left(\sum_{i=1}^d b_{1,i}A_i\right) - y_1\right] x_n \\ \vdots & & \vdots \\ \left[\left(\sum_{i=1}^d b_{p,i}A_d\right) - y_p\right] x_1 & \cdots & \left[\left(\sum_{i=1}^d b_{p,i}A_d\right) - y_p\right] x_n \end{pmatrix}$$

Let $R_k := \sum_{i=1}^d b_{k,i}A_i - y_k$ for $k = 1, ..., p$. Then we can write

$$Wxx^T - yx^T = \begin{pmatrix} R_1x_1 & \cdots & R_1x_n \\ \vdots & & \vdots \\ R_px_1 & \cdots & R_px_n \end{pmatrix} \tag{3.4}$$

Now we substitute the above expression for $Wxx^T - yx^T$ from (3.4) into (3.2) and get

$$\begin{pmatrix} b_{1,1} & \cdots & b_{p,1} \\ \vdots & & \vdots \\ b_{1,d} & \cdots & b_{p,d} \end{pmatrix} \begin{pmatrix} R_1x_1 & \cdots & R_1x_n \\ \vdots & & \vdots \\ R_px_1 & \cdots & R_px_n \end{pmatrix} + \begin{pmatrix} \lambda_{1,1}a_{1,1} & \cdots & \lambda_{1,n}a_{1,n} \\ \vdots & & \vdots \\ \lambda_{d,1}a_{d,1} & \cdots & \lambda_{d,n}a_{d,n} \end{pmatrix} = 0$$

which gives us the following set of equations

$$\begin{gathered} \left(\sum_{k=1}^p b_{k,1}R_k\right) x_j = -\lambda_{1,j}a_{1,j} \quad \forall\ j = 1, ..., n \\ \vdots \\ \left(\sum_{k=1}^p b_{k,d}R_k\right) x_j = -\lambda_{d,j}a_{d,j} \quad \forall\ j = 1, ..., n \end{gathered} \tag{3.5}$$

Next, we substitute the expression for $Wxx^T - yx^T$ from (3.4) into (3.3) and get

$$\begin{pmatrix} R_1x_1 & \cdots & R_1x_n \\ \vdots & & \vdots \\ R_px_1 & \cdots & R_px_n \end{pmatrix} \begin{pmatrix} a_{1,1} & \cdots & a_{d,1} \\ \vdots & & \vdots \\ a_{1,n} & \cdots & a_{d,n} \end{pmatrix} + \begin{pmatrix} \mu_{1,1}b_{1,1} & \cdots & \mu_{1,d}b_{1,d} \\ \vdots & & \vdots \\ \mu_{p,1}b_{p,1} & \cdots & \mu_{p,d}b_{p,d} \end{pmatrix} = 0$$

$$\begin{pmatrix} R_1\sum_{j=1}^n a_{1,j}x_j & \cdots & R_1\sum_{j=1}^n a_{d,j}x_j \\ \vdots & & \vdots \\ R_p\sum_{j=1}^n a_{1,j}x_j & \cdots & R_p\sum_{j=1}^n a_{d,j}x_j \end{pmatrix} + \begin{pmatrix} \mu_{1,1}b_{1,1} & \cdots & \mu_{1,d}b_{1,d} \\ \vdots & & \vdots \\ \mu_{p,1}b_{p,1} & \cdots & \mu_{p,d}b_{p,d} \end{pmatrix} = 0$$

$$\begin{pmatrix} R_1A_1 & \cdots & R_1A_d \\ \vdots & & \vdots \\ R_pA_1 & \cdots & R_pA_d \end{pmatrix} + \begin{pmatrix} \mu_{1,1}b_{1,1} & \cdots & \mu_{1,d}b_{1,d} \\ \vdots & & \vdots \\ \mu_{p,1}b_{p,1} & \cdots & \mu_{p,d}b_{p,d} \end{pmatrix} = 0$$

We obtain $R_kA_i + \mu_{k,i}b_{k,i} = 0$ for $i = 1, ..., d$ and $k = 1, ..., p$, which can be written as:

$$b_{k,i} = -\frac{R_kA_i}{\mu_{k,i}} \tag{3.6}$$

for all $i = 1, ..., d$ and for all $k = 1, ..., p$.

Note that the systems ((3.2),(3.3)) and ((3.5),(3.6)) are equivalent in the sense that every solution of one system is a solution of the other. We note once and for all that the regularization constants Λ, μ and the training data (X, Y) are chosen uniformly at random and are therefore non-zero with probability one. Hence, division by these constants is justified.

Proposition 3.1 (no stray zeros in W_1)**.** *Consider a linear network with $H = 1, m = 1, d_x = n, d_y = p$ and $d_1 = d$. Let $(W_1, W_2) = (\begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ & \vdots & \\ a_{d,1} & \cdots & a_{d,n} \end{pmatrix}, \begin{pmatrix} b_{1,1} & \cdots & b_{1,d} \\ & \vdots & \\ b_{p,1} & \cdots & b_{p,d} \end{pmatrix})$ denote a solution to the regularized gradient polynomial system (3.1). If $a_{i,j} = 0$, then $a_{i,s} = 0$ for all $s = 1, ..., n$.*

Proof. Recall that for $H = 1, m = 1$, (3.1) yields ((3.2),(3.3)) which is equivalent to ((3.5),(3.6)). Hence, (W_1, W_2) is a solution to ((3.5),(3.6)). Let us look at the first equation in (3.5). Note that $(\sum_{k=1}^{p} b_{k,1}R_k)$, i.e. the coefficient of x_j, is a constant complex number that does not depend on j. Therefore $\frac{-\lambda_{1,j}a_{1,j}}{x_j}$ is a constant. Hence, we have $\frac{-\lambda_{1,1}a_{1,1}}{x_1} = \frac{-\lambda_{1,2}a_{1,2}}{x_2} = ... = \frac{-\lambda_{1,n}a_{1,n}}{x_n}$. Equivalently, $a_{1,j} = \left(\frac{x_j}{\lambda_{1,j}}\frac{\lambda_{1,1}}{x_1}\right)a_{1,1}$ for all $j = 1, ..., n$. Repeating the same observation for all equations in (3.5) above, we get the following set of equations:

$$\begin{aligned} a_{1,j} &= \left(\frac{x_j}{\lambda_{1,j}}\frac{\lambda_{1,1}}{x_1}\right)a_{1,1} \quad \forall\ j = 1, ..., n \\ &\vdots \\ a_{d,j} &= \left(\frac{x_j}{\lambda_{d,j}}\frac{\lambda_{d,1}}{x_1}\right)a_{d,1} \quad \forall\ j = 1, ..., n \end{aligned} \tag{3.7}$$

Hence $a_{i,j} = 0 \implies a_{i,1} = 0 \implies a_{i,s} = 0$ for all $s = 1, ..., n$. This completes the proof. □

Proposition 3.2 (null rows of W_1 match null columns of W_2)**.** *Consider a linear network with* $H = 1$*,* $m = 1$*,* $d_x = n$*,* $d_y = p$ *and* $d_1 = d$*. Let* $(W_1, W_2) = (\begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ & \vdots & \\ a_{d,1} & \cdots & a_{d,n} \end{pmatrix}, \begin{pmatrix} b_{1,1} & \cdots & b_{1,d} \\ & \vdots & \\ b_{p,1} & \cdots & b_{p,d} \end{pmatrix})$ *denote a solution to the regularized gradient polynomial system (3.1). Then,*

$$a_{i,\cdot} = 0 \iff b_{\cdot,i} = 0$$

Proof. Recall that for $H = 1, m = 1$, (3.1) yields ((3.2),(3.3)) which is equivalent to ((3.5),(3.6)). Hence, (W_1, W_2) is a solution to ((3.5),(3.6)).

($\Longrightarrow$) Also recall that $A_i = \sum_{j=1}^{n} a_{i,j}x_j$. Substituting the expression for $a_{i,j}$ from (3.7), we get

$$A_i = \left(\sum_{j=1}^{n} \frac{x_j^2}{\lambda_{i,j}} \right) \frac{\lambda_{i,1}}{x_1} a_{i,1} \qquad i = 1, ..., d \tag{3.8}$$

Then, substituting the expressions for A_i from (3.8) into (3.6), we have

$$b_{k,i} = -\frac{R_k}{\mu_{k,i}} \left(\sum_{j=1}^{n} \frac{x_j^2}{\lambda_{i,j}} \right) \frac{\lambda_{i,1}}{x_1} a_{i,1}$$

for all $i = 1, ..., d$ and all $k = 1, ..., p$. Now, $a_{i,\cdot} = 0 \implies a_{i,1} = 0 \implies b_{k,i} = 0$ for all $k = 1, ..., p \implies b_{\cdot,i} = 0$. This proves the implication in one direction.

($\Longleftarrow$) For the implication in the other direction, we note that $b_{\cdot,i} = 0 \implies b_{k,i} = 0$ for $k = 1, ..., p \implies \sum_{k=1}^{p} b_{k,i} R_k = 0 \implies \left(\sum_{k=1}^{p} b_{k,i} R_k\right) x_j = 0$ for $j = 1, ..., n \implies \lambda_{i,j} a_{i,j} = 0$ for $j = 1, ..., n$ (see 3.5) $\implies a_{i,j} = 0$ for $j = 1, ..., n \implies a_{i,\cdot} = 0$ $\square$

Proposition 3.3 (no stray zeros in W_2)**.** *Consider a linear network with $H = 1, m = 1, d_x = n, d_y = p$ and $d_1 = d$. Let $(W_1, W_2) = (\begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ & \vdots & \\ a_{d,1} & \cdots & a_{d,n} \end{pmatrix}, \begin{pmatrix} b_{1,1} & \cdots & b_{1,d} \\ & \vdots & \\ b_{p,1} & \cdots & b_{p,d} \end{pmatrix})$ denote a solution to the regularized gradient polynomial system (3.1). Then,*

$$b_{k,i} = 0 \implies b_{\cdot,i} = 0$$

Proof. Fix i. Now suppose $b_{k,i} = 0$ for some k. Then, (3.6) implies $-\frac{R_k A_i}{\mu_{k,i}} = 0$ which implies $A_i = 0$ or $R_k = 0$. We claim that R_k is not 0. Recall that $R_k = b_{k,1} A_1 + ... + b_{k,d} A_d - y_k$.

Hence,

$$\begin{aligned}
&R_k = 0\\
\implies &b_{k,1}A_1 + ... + b_{k,d}A_d - y_k = 0\\
\implies &b_{k,1}A_1 + ... + b_{k,d}A_d = y_k\\
\implies &\frac{b_{k,1}}{y_k}A_1 + ... + \frac{b_{k,d}}{y_k}A_d = 1\\
\implies &\frac{b_{k,1}}{y_k}\left(\sum_{j=1}^{n}\frac{x_j^2}{\lambda_{1,j}}\right)\frac{\lambda_{1,1}}{x_1}a_{1,1} + ... + \frac{b_{k,d}}{y_k}\left(\sum_{j=1}^{n}\frac{x_j^2}{\lambda_{d,j}}\right)\frac{\lambda_{d,1}}{x_1}a_{d,1} = 1 \quad (see\ 3.8)\\
\implies &g_1a_{1,1} + ... + g_da_{d,1} = 1
\end{aligned}$$

where $g_i = \frac{b_{k,i}}{y_k}\left(\sum_{j=1}^{n}\frac{x_j^2}{\lambda_{i,j}}\right)\frac{\lambda_{i,1}}{x_1}$. Since each g_i is a polynomial in $b_{k,i}$ with no constant term, the L.H.S. of the last equation above cannot have a constant term. In particular the L.H.S cannot equal 1. Therefore we conclude that $g_1a_{1,1} + ... + g_da_{d,1} = 1$ cannot be true. Hence, R_k is nots 0. This means $A_i = 0$ must be true. Now, $A_i = 0 \implies a_{i,1} = 0$ (3.8) $\implies a_{i,\cdot} = 0$ (Prop. 3.1) $\implies b_{\cdot,i} = 0$ (Prop. 3.2). This completes the proof. □

Propositions 3.1 , 3.2 and 3.3 together state that if the (i, j) entry in W_1 is zero, then the entire ith row of W_1 and the entire ith column of W_2 must be zero. Similarly, if the (k, i) entry in W_2 is zero then the entire ith column of W_2 must be zero and the entire ith row of W_1 must be zero. The preceding result restricts the number of possible solutions that can lie outside $(\mathbb{C}^*)^N$. Next we examine solutions that lie inside $(\mathbb{C}^*)^N$.

Definition 3.4 (corner polynomial of degree d)**.** Let $f \in \mathbb{C}[x_1, ..., x_N]$ be a polynomial of degree d. We say that f is a corner polynomial of degree d if

- f contains the monomials $c_1x_1^d, c_2x_2^d, ..., c_Nx_N^d$, where $c_1, c_2, ..., c_N \in \mathbb{C}^*$.

- f contains a constant term

Lemma 3.5. *Let $f \in \mathbb{C}[x_1, ..., x_N]$ be a corner polynomial of degree d. Then the Newton polytope of f is the simplex*

$$conv\{0, d \cdot e_1, ..., d \cdot e_N\} \subset \mathbb{R}^N$$

where $e_1, ..., e_N$ are the standard basis vectors in $\mathbb{R}^N$.

The above lemma is easy to verify. We do not present a proof here. The next proposition deals with the number of solutions that lie inside $(\mathbb{C}^*)^N$.

Proposition 3.6 (upper bound on solutions in $(\mathbb{C}^*)^N$, $\mathcal{B}_{\mathbb{C}^*}$)**.** *Consider a linear network with $H = 1$, $m = 1$, $d_x = n$, $d_y = p$ and $d_1 = d$. Let $(W_1, W_2) = (\begin{pmatrix} a_{1,1} & \cdots & a_{1,n} \\ & \vdots & \\ a_{d,1} & \cdots & a_{d,n} \end{pmatrix}, \begin{pmatrix} b_{1,1} & \cdots & b_{1,d} \\ & \vdots & \\ b_{p,1} & \cdots & b_{p,d} \end{pmatrix})$ denote a solution to the gradient polynomial system (3.1). Then, there are at most*

$$\mathcal{B}_{\mathbb{C}^*} = (4p)^d$$

solutions for which $a_{1,1}, ..., a_{d,n} \in \mathbb{C}^*$ and $b_{1,1}, ..., b_{p,d} \in \mathbb{C}^*$.

Proof. Recall that for $H = 1, m = 1$, (3.1) yields ((3.2), (3.3)) which is equivalent to ((3.5), (3.6)). Hence, (W_1, W_2) is a solution to ((3.5), (3.6)).

The overall strategy for this proof is to first express the $b_{k,i}$ variables in terms of the $a_{i,j}$ variables using (3.6), and then substitute the expressions for the $b_{k,i}$ variables so obtained into (3.5). That will give us a polynomial system in just the $a_{i,j}$ variables. As we will see, the new polynomial system will have fewer variables than (3.5). We will then compute the BKK bound for this "reduced" system. That will give us an upper bound on the number of solutions in $(\mathbb{C}^*)^N$.

We first express $b_{k,2}$, ..., $b_{k,d}$ in terms of $b_{k,1}$. Recall that (3.8) gives $A_i = \left(\sum_{j=1}^{n} \frac{x_j^2}{\lambda_{i,j}}\right) \frac{\lambda_{i,1}}{x_1} a_{i,1}$. i.e. each A_i is equal to $a_{i,1}$ scaled by some constant. Since each $a_{i,j}$ is non-zero by assumption, each A_i is non-zero. Then (3.6) implies

$$\frac{b_{k,1}\mu_{k,1}}{A_1} = \frac{b_{k,2}\mu_{k,2}}{A_2} = ... = \frac{b_{k,d}\mu_{k,d}}{A_d}$$

Hence, $b_{k,2} = b_{k,1}\left(\frac{\mu_{k,1}}{\mu_{k,2}}\frac{A_2}{A_1}\right)$, ..., $b_{k,d} = b_{k,1}\left(\frac{\mu_{k,1}}{\mu_{k,d}}\frac{A_d}{A_1}\right)$. Recall that $R_k = b_{k,1}A_1 + ... + b_{k,d}A_d - y_k$. Substituting the preceding expressions for $b_{k,2}$ through $b_{k,d}$ into R_k, we get:

$$\begin{aligned} R_k &= b_{k,1}A_1 + b_{k,1}\left(\frac{\mu_{k,1}}{\mu_{k,2}}\frac{A_2}{A_1}\right)A_2 + ... + b_{k,1}\left(\frac{\mu_{k,1}}{\mu_{k,d}}\frac{A_d}{A_1}\right)A_d - y_k \\ &= b_{k,1}\left[A_1 + \left(\frac{\mu_{k,1}}{\mu_{k,2}}\frac{A_2^2}{A_1}\right) + ... + \left(\frac{\mu_{k,1}}{\mu_{k,d}}\frac{A_d^2}{A_1}\right)\right] - y_k \end{aligned}$$

Now, for $i = 1$, (3.6) implies $b_{k,1} = -\frac{R_k A_1}{\mu_{k,1}}$. Substituting the expression for R_k from above, we get

$$\begin{aligned} b_{k,1} &= -\left(b_{k,1}\left[A_1 + \left(\frac{\mu_{k,1}}{\mu_{k,2}}\frac{A_2^2}{A_1}\right) + ... + \left(\frac{\mu_{k,1}}{\mu_{k,d}}\frac{A_d^2}{A_1}\right)\right] - y_k\right)\frac{A_1}{\mu_{k,1}} \\ \implies \frac{y_k A_1}{\mu_{k,1}} &= b_{k,1}\left[1 + \frac{A_1^2}{\mu_{k,1}} + \frac{A_2^2}{\mu_{k,2}} + ... \frac{A_d^2}{\mu_{k,d}}\right] \\ \implies b_{k,1} &= \frac{y_k A_1}{\mu_{k,1}} \frac{1}{\left[1 + \frac{A_1^2}{\mu_{k,1}} + \frac{A_2^2}{\mu_{k,2}} + ... \frac{A_d^2}{\mu_{k,d}}\right]} \end{aligned}$$

Let $T_k := \frac{A_1^2}{\mu_{k,1}} + \frac{A_2^2}{\mu_{k,2}} + ... \frac{A_d^2}{\mu_{k,d}}$. Then we can write $b_{k,1} = \frac{y_k A_1}{\mu_{k,1}(1+T_k)}$. Repeating the above process for all values of i, we get:

$$b_{k,i} = \frac{y_k A_i}{\mu_{k,i}(1 + T_k)} \tag{3.9}$$

Note that the R.H.S. of the above equation does not contain any b variables. Now, we substitute the expressions for $b_{k,i}$ from (3.9) into the first equation of (3.5)

$$\begin{aligned}
-\frac{\lambda_{1,j}a_{1,j}}{x_j} &= \left(\sum_{k=1}^{p} b_{k,1}R_k\right) \\
&= \sum_{k=1}^{p} b_{k,1}(b_{k,1}A_1 + ... + b_{k,d}A_d - y_k) \\
&= \sum_{k=1}^{p} \frac{y_k A_1}{\mu_{k,1}(1+T_k)} \left[\frac{y_k A_1^2}{\mu_{k,1}(1+T_k)} + ... + \frac{y_k A_d^2}{\mu_{k,d}(1+T_k)} - y_k\right] \\
&= \sum_{k=1}^{p} \frac{y_k^2 A_1}{\mu_{k,1}(1+T_k)} \left[\frac{A_1^2}{\mu_{k,1}(1+T_k)} + ... + \frac{A_d^2}{\mu_{k,d}(1+T_k)} - 1\right] \\
&= \sum_{k=1}^{p} \frac{y_k^2 A_1}{\mu_{k,1}(1+T_k)^2} \left[\frac{A_1^2}{\mu_{k,1}} + ... + \frac{A_d^2}{\mu_{k,d}} - (T_k+1)\right] \\
&= \sum_{k=1}^{p} \frac{y_k^2 A_1}{\mu_{k,1}(1+T_k)^2} \left[T_k - (T_k+1)\right] \\
&= -\sum_{k=1}^{p} \frac{y_k^2 A_1}{\mu_{k,1}(1+T_k)^2}
\end{aligned}$$

Now, (3.7) implies $-\frac{\lambda_{1,j}a_{1,j}}{x_j} = -\frac{\lambda_{1,1}a_{1,1}}{x_1}$. Moreover, $A_1 = \left(\sum_{j=1}^{n} \frac{x_j^2}{\lambda_{1,j}}\right) \frac{\lambda_{1,1}a_{1,1}}{x_1}$. Substituting in the equation above, we get

$$\begin{aligned}
\frac{1}{\left(\sum_{j=1}^{n} \frac{x_j^2}{\lambda_{1,j}}\right)} &= \sum_{k=1}^{p} \frac{y_k^2}{\mu_{k,1}(1+T_k)^2} \\
\implies \frac{1}{\left(\sum_{j=1}^{n} \frac{x_j^2}{\lambda_{1,j}}\right)} &= \left[\frac{y_1^2}{\mu_{1,1}(1+T_1)^2} + \frac{y_2^2}{\mu_{2,1}(1+T_2)^2} + ... + \frac{y_p^2}{\mu_{p,1}(1+T_p)^2}\right]
\end{aligned}$$

which gives us the equation

$$\frac{1}{\left(\sum_{j=1}^{n} \frac{x_j^2}{\lambda_{1,j}}\right)} - \left[\frac{y_1^2}{\mu_{1,1}(1+T_1)^2} + \frac{y_2^2}{\mu_{2,1}(1+T_2)^2} + ... + \frac{y_p^2}{\mu_{p,1}(1+T_p)^2}\right] = 0 \qquad (3.10)$$

Multiplying both sides of the above equation by the product $(1+T_1)^2(1+T_2)^2\cdots(1+T_p)^2$ we get

$$\begin{aligned}\frac{(1+T_1)^2(1+T_2)^2\cdots(1+T_p)^2}{\left(\sum_{j=1}^{n}\frac{x_j^2}{\lambda_{1,j}}\right)} &- \frac{y_1^2}{\mu_{1,1}}(1+T_2)^2(1+T_3)^2\cdots(1+T_p)^2\\ &- \frac{y_2^2}{\mu_{2,1}}(1+T_1)^2(1+T_3)^2\cdots(1+T_p)^2\\ &\vdots\\ &- \frac{y_p^2}{\mu_{p,1}}(1+T_1)^2(1+T_2)^2\cdots(1+T_{p-1})^2 = 0\end{aligned} \tag{3.11}$$

Recall that $T_k := \frac{A_1^2}{\mu_{k,1}} + \frac{A_2^2}{\mu_{k,2}} + ... \frac{A_d^2}{\mu_{k,d}}$, and that each A_i is equal to $a_{i,1}$ scaled by some constant. Therefore, each T_k is a polynomial in the variables $a_{1,1},...,a_{d,1}$. Hence, (3.11) is a polynomial equation in the variables $a_{1,1},...,a_{d,1}$. We are interested in the Newton polytope of this polynomial.

Note that, the polynomial in (3.11) has the same monomial structure, and therefore the same Newton polytope, as the polynomial

$$\left(1+a_{1,1}^2+...+a_{d,1}^2\right)^{2p} - \left(1+a_{1,1}^2+...+a_{d,1}^2\right)^{2(p-1)}$$

Also note that the above polynomial is a corner polynomial of degree $4p$ and therefore, by Lemma 3.5, has the Newton polytope

$$Q = conv\{0, 4p\cdot e_1, ..., 4p\cdot e_d\} \tag{3.12}$$

Hence we conclude that our polynomial in (3.11) also has the above Newton polytope.

Recall that the polynomial equation (3.11) and the corresponding newton polytope (3.12) were obtained by substituting the expressions for $b_{k,i}$ from (3.6) into the first equation of

(3.5). Repeating this process for each of the d equations of (3.5) yields a polynomial equation similar to (3.11). Thus, we obtain the below new polynomial system with d equations in d unknowns

$$\begin{aligned} &\frac{(1+T_1)^2(1+T_2)^2\cdots(1+T_p)^2}{\left(\sum_{j=1}^{n}\frac{x_j^2}{\lambda_{1,j}}\right)} - \frac{y_1^2}{\mu_{1,1}}(1+T_2)^2(1+T_3)^2\cdots(1+T_p)^2 ... \\ &\qquad -\frac{y_p^2}{\mu_{p,1}}(1+T_1)^2(1+T_2)^2\cdots(1+T_{p-1})^2 = 0 \\ &\qquad \vdots \\ &\frac{(1+T_1)^2(1+T_2)^2\cdots(1+T_p)^2}{\left(\sum_{j=1}^{n}\frac{x_j^2}{\lambda_{d,j}}\right)} - \frac{y_1^2}{\mu_{1,d}}(1+T_2)^2(1+T_3)^2\cdots(1+T_p)^2 ... \\ &\qquad -\frac{y_p^2}{\mu_{p,d}}(1+T_1)^2(1+T_2)^2\cdots(1+T_{p-1})^2 = 0 \end{aligned} \tag{3.13}$$

It is easy to see that every polynomial in (3.13) has the same Newton polytope as (3.12). Hence the BKK bound for this system is equal to the mixed volume of d identical Newton polytopes, which is the same as the normalized Euclidean volume. Hence, we have

$$\mathcal{M}(Q,...,Q) = d!Vol(Q) = d!\cdot\frac{1}{d!}|det(4p\cdot e_1 \ \ ... \ \ 4p\cdot e_d)| = (4p)^d \tag{3.14}$$

By invoking Bernstein's theorem (Th. 2.22) we conclude that the reduced system (3.13) can have at most $(4p)^d$ solutions in $(\mathbb{C}^*)^d$.

Note that the reduced system (3.13) was obtained by simple algebraic operations on the original gradient polynomial system ((3.5), (3.6)). Hence, we expect the the BKK bound for the new system to be equal to the BKK bound for the original system. However, this is not always the case. Recall that we obtained (3.11) by multiplying (3.10) on both sides by the product $(1+T_1)^2(1+T_2)^2\cdots(1+T_p)^2$. Consequently, every solution of (3.10) is a solution of (3.11). However, the converse is not true. In particular, values of $a_{1,1},...,a_{d,1}$ that satisfy $(1+T_1)=...=(1+T_p)=0$ will form a solution to (3.11) but not a solution to (3.10). In other words, the reduced system (3.13) may have more solutions than the original system ((3.5), (3.6)) (see 3.7). Nevertheless, the BKK bound computed in (3.14) is still an upper bound on the number of solutions in $(\mathbb{C}^*)^N$ to the original system ((3.5), (3.6)). This completes the proof. $\square$

Example 3.7. [Extraneous solutions in the reduced system] We consider the case $H=1, m=1, d=2, d_x=1, d_y=2$. Let $(W_1,W_2)=(\left(\begin{smallmatrix} a_{1,1} \\ a_{2,1}\end{smallmatrix}\right), \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2}\end{pmatrix})$ be a solution to the gradient polynomial system (3.1) with $a_{1,1},a_{2,1},b_{1,1},b_{1,2},b_{2,1},b_{2,2}\in\mathbb{C}^*$. Let $X=(1), Y=\left(\begin{smallmatrix} 2 \\ 3\end{smallmatrix}\right), \Lambda=\left(\begin{smallmatrix} 4 \\ 5\end{smallmatrix}\right), \mu=\left(\begin{smallmatrix} 6 & 7 \\ 8 & 9\end{smallmatrix}\right)$. Then the original system ((3.5), (3.6)) yields 16 solutions in $(\mathbb{C}^*)^6$ while the reduced system (3.13) yields 18 solutions, including two non-real singular solutions, in $(\mathbb{C}^*)^2$.

Corollary 3.7.1 (solutions with r non-zero rows in W_1)**.** *Consider a linear network with $H=1$, $m=1$, $d_x=n$, $d_y=p$ and $d_1=d$. Let (W_1,W_2) denote a solution to the gradient polynomial system (3.1). Consider the solutions where W_1 contains r non-zero rows. Then there are at most $\binom{d}{r}(4p)^r$ such solutions.*

Proof. W.L.O.G. we can assume that the first r rows of W_1 are non-zero. Then for all

$r < i \leq d$, we have $a_{i,\cdot} = 0$ and $b_{\cdot,i} = 0$. Substituting these values in (3.5) we get

$$\begin{aligned} \left(\sum_{k=1}^{p} b_{k,1} R_k\right) x_j &= -\lambda_{1,j} a_{1,j} \quad \forall\ j = 1, ..., n \\ &\vdots \\ \left(\sum_{k=1}^{p} b_{k,r} R_k\right) x_j &= -\lambda_{r,j} a_{r,j} \quad \forall\ j = 1, ..., n \end{aligned} \tag{3.15}$$

Substituting in (3.6), we get

$$b_{k,i} = -\frac{R_k A_i}{\mu_{k,i}} \tag{3.16}$$

for all $i = 1, ..., r$ and for all $k = 1, ..., p$.

Recall that in the proof of Proposition 3.6, we applied a series of algebraic operations on the system ((3.5), (3.6)) to obtain a reduced polynomial system (3.13) with d equations in d unknowns, where each polynomial was a corner polynomial of degree $4p$. Applying Bernstein's theorem yielded a BKK bound of $(4p)^d$.

Applying the same process to the system ((3.15), (3.16)) above, we obtain a reduced polynomial system with r equations in r unknowns, where each polynomial is a corner polynomial of degree $4p$. Applying Bernstein's theorem yields a BKK bound of $(4p)^r$. Hence, there are at most $(4p)^r$ solutions where the first r rows of W_1 are non-zero. Finally, there are $\binom{d}{r}$ ways of selecting r non-zero rows out of d rows. Hence we conclude that there are at most $\binom{d}{r}(4p)^r$ solutions with r non-zero rows in W_1. This completes the proof. □

Corollary 3.7.2 (solutions with 1 non-zero row in W_1)**.** *Consider a linear network with $H = 1$, $m = 1$, $d_x = n$, $d_y = p$ and $d_1 = d$. Let (W_1, W_2) denote a solution to the gradient* *polynomial system (3.1). Consider the solutions where W_1 contains 1 non-zero row. Then there are exactly $4p \cdot d$ such solutions.*

Proof. W.L.O.G. we can assume the first row of W_1 to be non-zero. Then for all $1 < i \leq d$, we have $a_{i,\cdot} = 0$ and $b_{\cdot,i} = 0$. Substituting these values in (3.5), we get the single equation

$$\left(\sum_{k=1}^{p} b_{k,1} R_k \right) x_j = -\lambda_{1,j} a_{1,j} \qquad \forall\ j = 1, ..., n \tag{3.17}$$

Substituting in (3.6), we get

$$b_{k,1} = -\frac{R_k A_1}{\mu_{k,1}} \tag{3.18}$$

for all $k = 1, ..., p$. Once again, we apply the same algebraic operations to the system ((3.17), (3.18)) as we did to the system ((3.5), (3.6)) in the proof of Proposition 3.6. This yields a reduced polynomial system with one equation in one unknown, where the polynomial is a corner polynomial of degree $4p$. We write this polynomial explicitly below

$$\begin{aligned} \frac{(1+T_1)^2(1+T_2)^2 \cdots (1+T_p)^2}{\left(\sum_{j=1}^{n} \frac{x_j^2}{\lambda_{1,j}} \right)} &- \frac{y_1^2}{\mu_{1,1}}(1+T_2)^2(1+T_3)^2 \cdots (1+T_p)^2 \\ &- \frac{y_2^2}{\mu_{2,1}}(1+T_1)^2(1+T_3)^2 \cdots (1+T_p)^2 \\ &\vdots \\ &- \frac{y_p^2}{\mu_{p,1}}(1+T_1)^2(1+T_2)^2 \cdots (1+T_{p-1})^2 = 0 \end{aligned} \tag{3.19}$$

where $T_k = \frac{A_1^2}{\mu_{k,1}}$ and A_1 is equal to $a_{1,1}$ scaled by some constant. Clearly, (3.19) is degree $4p$ polynomial in $a_{1,1}$. The fundamental theorem of algebra states that (3.19) has exactly $4p$ roots. Finally, there are d ways to choose a row out of d rows. Hence, we conclude that the there are exactly $4p \cdot d$ solutions where W_1 contains 1 non-zero row. This completes the proof. $\square$

Remark 3.8. It is worth noting that the upper bounds introduced in Proposition 3.6 and Corollary 3.7.1 and the exact count introduced in Corollary 3.7.2 only depend on d_y, the length of the output vector. In particular, they do not depend on the length d_x of the input vector. This fact is consistent with experimental observations. For fixed values of d, H and m, changing d_x did not change the number of complex solutions $N_{\mathbb{C}}$. On the other hand, changing d_y did change the number of complex solutions (see Table 3.1).

We now present the main result of this section. Theorem 3.9 allows us to place an upper bound on the total number of complex critical points of a linear network.

Theorem 3.9 (upper bound on complex critical points, $\mathcal{B}_{\mathbb{C}}$)**.** *Consider a linear network with $H = 1$, $m = 1$, $d_x = n$, $d_y = p$ and $d_1 = d$. This network has at most*

$$\mathcal{B}_{\mathbb{C}} = (1 + 4p)^d$$

complex critical points.

Proof. The proof merely consists of applying Corollary 3.7.1 for different values of $r = 1, ..., d$ and adding up the corresponding upper bounds. Finally, we add 1 to account for the origin as a solution. We get $\mathcal{B}_{\mathbb{C}} = 1 + \sum_{r=1}^{d} \binom{d}{r} \cdot (4p)^r = (1 + 4p)^d$ $\square$

In Table 3.1, we present numerical results comparing various upper bounds with the actual number of complex solutions for the $H = 1, m = 1$ case. These results were obtained by solving randomly generated polynomial systems for various values of d, d_x and d_y. Each row in Table 3.1 contains the results from a sample of 100 randomly generated polynomial systems, each system corresponding to a particular choice of the regularization constants Λ_i, μ_i and training data (X, Y) (see 3.1). Each entry of Λ_i, μ_i, X, Y was chosen uniformly at random from $[0, 1]$.

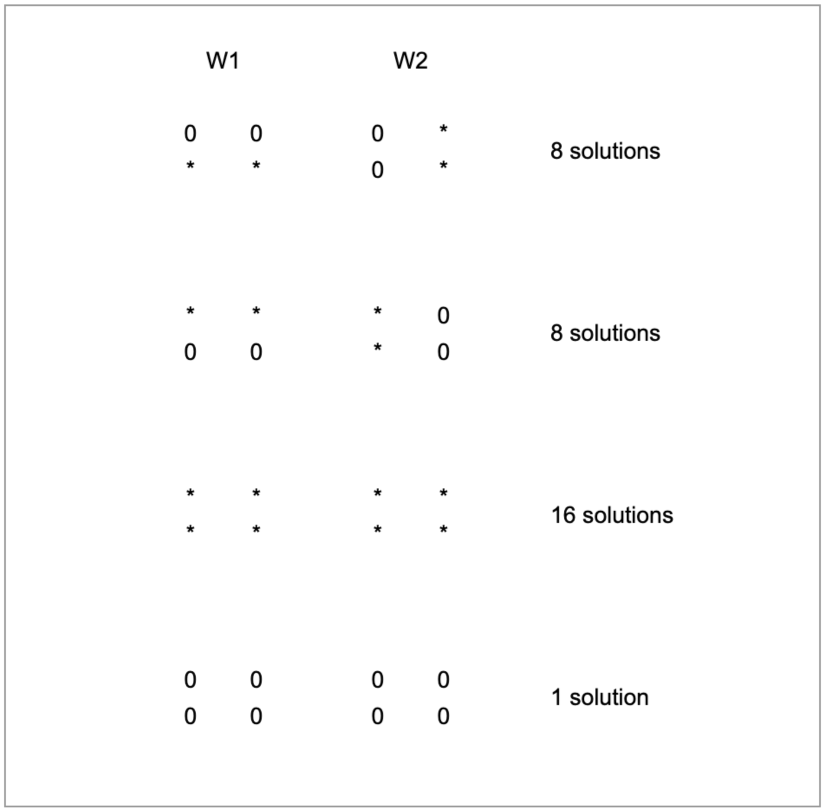
Recall that the bounds CBB, BKK, $\mathcal{B}_{\mathbb{C}}$ and $\mathcal{B}_{\mathbb{C}^*}$ only depend on the monomial structure of the polynomials and not on the coefficients. As a result these bounds are constant for each sample. Moreover, choosing the regularization constants and training examples randomly makes the coefficients of our polynomial systems generic. As a result, the number of complex solutions remains constant over each sample. However, the number of real solutions varies within in each sample. We report the maximum number of real solutions observed for each sample in Table 3.1.

In Figure 3.1, we show how the complex solutions of the gradient polynomial system are situated in $\mathbb{C}^N$ for the case $H = 1$, $m = 1$, $d = 2$, $d_x = 2$ and $d_y = 2$. We note that the results are consistent with Proposition 3.1 (no stray zeros in W_1), Proposition 3.2 (null rows of W_1 match null columns of W_2) and Proposition 3.3 (no stray zeros in W_2). Indeed, the solutions that lie outside $(\mathbb{C}^*)^N$ lie on particular coordinate subspaces.

In Figure 3.2, we show the frequency distribution of the number of real solutions for the particular case of $H = 1$, $m = 1$, $d = 1$, $d_x = 2$, $d_y = 3$. While there were no observations with 0 real solutions, there were 60 observations with 7 real solutions. The maximum real solution count observed within the sample was 27.

No	d	d_x	d_y	N	CBB	BKK	$\mathcal{B}_{\mathbb{C}}$	$\mathcal{B}_{\mathbb{C}^*}$	$N_{\mathbb{C}}$	$N_{\mathbb{C}^*}$	$max\{N_{\mathbb{R}}\}$
1	1	1	1	2	9	5	5	4	5	4	3
2	1	2	1	3	27	9	5	4	5	4	3
3	1	3	1	4	81	13	5	4	5	4	3
4	1	1	2	3	27	9	9	8	9	8	3
5	1	2	2	4	81	33	9	8	9	8	3
6	1	3	2	5	243	73	9	8	9	8	3
7	1	1	3	4	81	13	13	12	13	12	3
8	1	2	3	5	243	73	13	12	13	12	3
9	1	3	3	6	729	245	13	12	13	12	3
10	2	1	1	4	81	25	25	16	9	0	4
11	2	2	1	6	729	81	25	16	9	0	5
12	2	3	1	8	6561	169	25	16	9	0	5
13	2	1	2	6	729	81	81	64	33	16	9
14	2	2	2	8	6561	1089	81	64	33	16	9
15	2	3	2	10	59049	5329	81	64	33	16	9
16	2	1	3	8	6561	169	169	144	73	48	9
17	2	2	3	10	59049	5329	169	144	73	48	9
18	2	3	3	12	531441	60025	169	144	73	48	9
19	3	1	1	6	729	125	125	64	13	0	7
20	3	2	1	9	19683	729	125	64	13	0	7
21	3	3	1	12	531441	2197	125	64	13	0	7
22	3	1	2	9	19683	729	729	512	73	0	19
23	3	2	2	12	531441	35937	729	512	73	0	19
24	3	3	2	15	14348907	389017	729	512	73	0	19
25	3	1	3	12	531441	2197	2197	1728	245	64	27
26	3	2	3	15	14348907	389017	2197	1728	245	64	27

Table 3.1: Case: $H = 1, m = 1$. Comparison of upper bounds on the number of complex critical points of a linear network. d = number of neurons in each layer, d_x = input dimension and d_y = output dimension. N = total number of weights in the network. CBB and BKK refer to the classical Bezout bound and the BKK bound respectively. $\mathcal{B}_{\mathbb{C}}$ and $\mathcal{B}_{\mathbb{C}^*}$ refer to the new bounds on the number of critical points in $(\mathbb{C})^N$ and $(\mathbb{C}^*)^N$ respectively. $N_{\mathbb{C}}$ and $N_{\mathbb{C}^*}$ refer to the actual number of critical points in $(\mathbb{C})^N$ and $(\mathbb{C}^*)^N$ respectively. $max\{N_{\mathbb{R}}\}$ = maximum number of real solutions observed within each sample.



W1	W2	
$\begin{matrix} 0 & 0 \\ * & * \end{matrix}$	$\begin{matrix} 0 & * \\ 0 & * \end{matrix}$	8 solutions
$\begin{matrix} * & * \\ 0 & 0 \end{matrix}$	$\begin{matrix} * & 0 \\ * & 0 \end{matrix}$	8 solutions
$\begin{matrix} * & * \\ * & * \end{matrix}$	$\begin{matrix} * & * \\ * & * \end{matrix}$	16 solutions
$\begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}$	$\begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}$	1 solution

Figure 3.1: Case: $H = 1$, $m = 1$, $d = 2$, $d_x = 2$, $d_y = 2$ (corresponds to line no. 14 in Table 3.1. Number of solutions corresponding to each zero-pattern occurring in the weight matrices W_1 and W_2. * represents a non-zero entry.

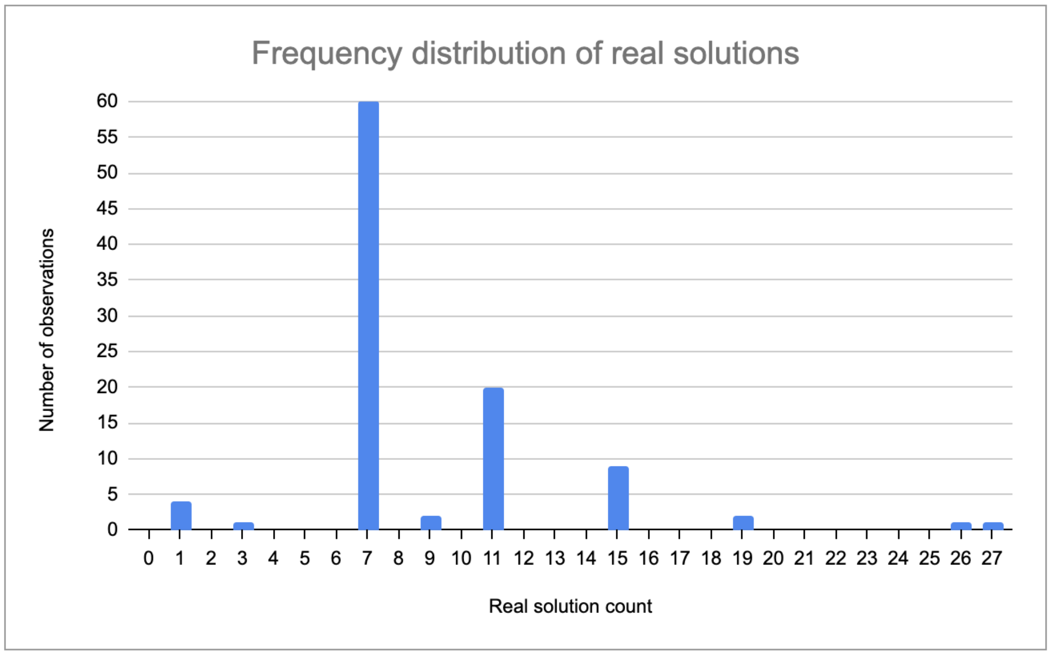


Figure 3.2: Case: $H = 1$, $m = 1$, $d = 3$, $d_x = 2$, $d_y = 3$ (corresponds to line no. 26 in Table 3.1). Frequency of occurrence of real solution counts within a sample of 100 observations. Each observation corresponds to a gradient polynomial system with a particular choice of regularization constants and training data. There were 60 observations with 7 real solutions. There were no observations with 0 real solutions.

Networks with $H > 1$, $m = 1$

In Table 3.2, we record the number of solutions for linear networks with 2-hidden layers. While we do not extend $\mathcal{B}_{\mathbb{C}^*}$ and $\mathcal{B}_{\mathbb{C}}$ to the $H = 2$ case, we do note that the CBB and BKK bounds are much higher than the actual number of solutions. Existence of tighter upper bounds seems plausible.

No.	d	d_x	d_y	N	CBB	BKK	$N_{\mathbb{C}}$	$N_{\mathbb{C}^*}$	$max\{N_{\mathbb{R}}\}$
1	1	1	1	3	125	17	16	16	8
2	1	2	1	4	625	33	17	16	8
3	1	3	1	5	3125	49	17	16	8
4	1	1	2	4	625	33	33	32	9
5	1	2	2	5	3125	129	33	32	9
6	1	3	2	6	15625	289	33	32	9
7	2	1	1	8	390625	1857	129	64	64
8	2	2	1	10	9765625	23937	129	64	64
9	2	3	1	12	244140625	109249	129	64	64
10	2	1	2	10	9765625	23937	641	384	64
11	2	2	2	12	244140625	780801	641	384	72
12	3	1	1	15	30517578125	667921	977	256	312

Table 3.2: Case: $H = 2$, $m = 1$. Comparison of classical upper bounds with the actual number of complex critical points of a linear network.

Chapter 4

Looking Ahead

In this thesis, we explored the complex critical points of the loss function of linear networks. In particular we showed that minimizing the loss function of these networks can be turned into a problem of solving systems of polynomial equations. For 1-hidden layer networks, we showed that the above change of perspective allows us to not only place good upper bounds on the number of complex critical points but also allows us to find them more efficiently. Furthermore, we also showed that there is structure in the way zeros appear in the weight matrices of 1-hidden layer linear networks. In particular, critical points that lie outside of $(\mathbb{C}^*)^N$ lie on particular coordinate subspaces of $\mathbb{C}^N$.

The above results lead to some interesting questions which may open up avenues for further research. We list some of these questions below:

- Are the bounds $\mathcal{B}_{\mathbb{C}^*}$ and $\mathcal{B}_{\mathbb{C}}$ attained if we increase the number of training examples?

- Conversely, can we show that $\mathcal{B}_{\mathbb{C}^*}$ and $\mathcal{B}_{\mathbb{C}}$ are not attained?
- Can we extend the bounds $\mathcal{B}_{\mathbb{C}^*}$ and $\mathcal{B}_{\mathbb{C}}$ to linear networks with $H > 1$?
- Can we show that the zero-patterns hold for $H > 1$?

Answering these questions will help us better understand the effect that m (number of training examples) and H (number of layers) have on the number and location of the critical points of linear networks in $\mathbb{C}^N$.

.1 Appendix A

Solutions to the system (1.13)

- ComplexF64[-58.74246608059022 + 33.99335474985693im,
  26.93220890741545 - 15.585251908406631im,
  -0.009292722892228591 - 0.005377554722239227im,
  0.006548760207464838 + 0.0037896660415771078im]
- ComplexF64[0.0154333736179604 - 0.023845351783789572im,
  -0.0014456112048643842 + 0.0022335432664228345im,
  33.88424645895807 + 52.352894237089174im,
  48.08191211462479 + 74.28901400250797im]
- ComplexF64[-2.6008099497868605e-42 - 4.842887492706568e-42im,
  1.210721873176642e-42 + 2.2420775429197073e-42im,
  -2.8698592549372254e-42 - 3.407957865237955e-42im,
  1.9730282377693424e-42 + 2.3317606446364956e-42im]
- ComplexF64[-0.001177087380266545 - 0.00400842863095992im,
  0.0005396702818728927 + 0.0018377818379529 83im,
  -49.14436810399029 + 167.35519848480632im,
  34.63297958982107 - 117.93842096230237im]
- ComplexF64[4.026936355450126 - 207.1820248386482im,
  -0.37719454351443216 + 19.406298581712793im,

```
    0.00016611312689544 + 0.008546361537572086im,

    0.00023571534276691612 + 0.012127328988917559im]
```

Bibliography

[1] Daniel J. Bates et al. *Numerically solving polynomial systems with Bertini.* Vol 25 of Software, Environments and Tools. SIAM, 2013. ISBN: 978-1-611972-69-6.

[2] D.N. Bernshtein. "The number of roots of a system of equations". In: *Functional Anal. Appl.* 9 (1975), pp. 1–4. DOI: `https://doi.org/10.1007/BF01075595`.

[3] Paul Breiding and Sascha Timme. "HomotopyContinuation.jl: A Package for Homotopy Continuation in Julia". In: *International Congress on Mathematical Software*. Springer. 2018, pp. 458–465.

[4] Ovidiu Calin. *Deep Learning Architectures - A Mathematical Approach.* Springer Series in the Data Sciences. Springer Cham, 2020. ISBN: 978-3-030-36720-6.

[5] Birkett Huber and Bernd Sturmfels. "A Polyhedral Method for Solving Sparse Polynomial Systems". In: *Mathematics of Computation* 64.212 (1995), pp. 1541–1555. URL: `https://www.jstor.org/stable/2153370`.

[6] D. Mehta et al. "The loss surface of deep linear networks viewed through the algebraic geometry lens". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.9 (2022), pp. 5664–5680. DOI: `10.1109/TPAMI.2021.3071289`.

[7] Rolf Steyer and Werner Nagel. *"Probability and conditional expectation: fundamentals for the empirical sciences"*. Wiley series in probability and statistics. Chichester, West Sussex, United Kingdom, 2017. ISBN: 9781119243526.