

Neurons on Amoebae

Jiakang Bao,^a Yang-Hui He,^{b,a,c,d} Edward Hirst^a

^a*Department of Mathematics, City, University of London, EC1V 0HB, UK*

^b*London Institute for Mathematical Sciences, Royal Institution of GB, W1S 4BS, UK*

^c*Merton College, University of Oxford, OX1 4JD, UK*

^d*School of Physics, NanKai University, Tianjin, 300071, P.R. China*

E-mail: jiakang.bao@city.ac.uk, hey@maths.ox.ac.uk,
edward.hirst@city.ac.uk

ABSTRACT: We apply methods of machine-learning, such as neural networks, manifold learning and image processing, in order to study 2-dimensional amoebae in algebraic geometry and string theory. With the help of embedding manifold projection, we recover complicated conditions obtained from so-called lopsidedness. For certain cases it could even reach $\sim 99\%$ accuracy, in particular for the lopsided amoeba of F_0 with positive coefficients which we place primary focus. Using weights and biases, we also find good approximations to determine the genus for an amoeba at lower computational cost. In general, the models could easily predict the genus with over 90% accuracies. With similar techniques, we also investigate the membership problem, and image processing of the amoebae directly.

Contents

1	Introduction	1
2	Rudiments of Amoebae	2
2.1	Amoebae and Lopsidedness	5
2.2	Amoebae from D-branes	6
3	ML Amoebae from Coefficients	8
3.1	Lopsided Amoebae: $n = 1$	9
3.1.1	Example 1: F_0	9
3.1.2	Example 2: $L^{3,3,2}$	13
3.1.3	Example 3: $\mathcal{C}/(\mathbb{Z}_2 \times \mathbb{Z}_4)$	15
3.1.4	Example 4: $K^{4,5,3,2}$	17
3.2	Reproduction of the Genus: Interpretable ML	21
3.2.1	From Projections	21
3.2.2	Learning from Weights	22
3.3	Lopsided Amoebae: $n > 1$	25
3.3.1	Example: F_0	25
3.3.2	Reproduction of the Genus	28
3.4	The Membership Problem	30
3.4.1	Crawling of Amoebae	31
4	Image Processing Amoebae	33
4.1	Amoeba Image Dataset	33
4.2	Image Processing Results	35
4.3	Amoeba Image Misclassifications	36
5	Conclusions and Outlook	40
A	Transformations of Amoebae	41
B	Manifold Learning	42
C	Lopsidedness and Cyclic Resultants	43
C.1	Genus for Lopsided Amoeba	43
C.2	Cyclic Resultants for F_0	46

1 Introduction

Amoebae of algebraic varieties along with closely related tropical geometry constitutes a lively branch of contemporary mathematics [1–3]. One notable direction is the relation to dimer models, which bring together the combinatorics of toric varieties, the geometry of complex manifolds, the theory of partitions and statistical mechanics [4, 5]. In string theory and supersymmetric quantum field theory, amoebae and dimers naturally encode quiver gauge theories whose space of vacua are affine toric Calabi-Yau manifolds [6] as well as topological strings on these geometries [7–10]. The reader is referred to [3, 11–16] for various reviews, directed to mathematicians and physicists alike.

More recently, there has been much activity on using techniques of modern data science, especially machine learning (ML), to analyse pure mathematical data, as tools of conjecture formulation and aids in computation and proof. Whilst this arose from the investigation of the string theory/algebraic geometry landscape [17–22], one is led to see how such a methodology can be extended to different disciplines of mathematics (q.v. reviews in [23, 24]).

Given these two skeins of enquiry, it is only natural to ask how one might machine-learn pertinent features of amoeba geometry. As outlined in [24], whatever sub-field of mathematics, there often emerges theoretical data representable as tensors, labelled or unlabelled. This is particularly pronounced due to the recent vast growth in computing power and experimental mathematics. One is thence encouraged to immediately turn to a neural network classifier or regressor to treat such data. Algebraic geometry is certainly a discipline rich in data of this type, for example varieties can be computationally recorded by lists of its polynomial coefficients; and its morphism maps in cohomology are integer matrices.

As we will see, amoebae and tropical geometry are endowed with the perfect data structure for machine learning: they are, by nature, graphical, and image-processing is ML’s forte. The purpose of this paper is to initiate the study of ML of tropical geometry, much like how [17] attempted to learn complex algebraic geometry or how [25] discrete geometry. We will see how otherwise difficult problems, such as deciding membership and computing genus, can be solved to high accuracy with neural networks (NNs). In fact, the problem can be effectively redescribed as the identification of bounding hypersurfaces in the amoebae coefficient abstract space — a feat NNs famously perform exceptionally well at. We should emphasize what we are doing here is

paradigmatically different from [26, 27], which finds tropical-geometric formulations of deep ReLU networks. Our emphasis is the use on ML to study tropical geometry, as opposed to the use of tropical geometry to study ML; and here we focus in on learning the genus of 2-dimensional amoebae within tropical geometry. It is worth noting that this work puts particular focus on the prototypical F_0 example, with other examples considered briefly. Training primarily uses lopsided amoebae, which act as approximations to the true amoebae, where the limit of convergence to the true amoebae considered in specific cases.

Outline The paper is organized as follows. In §2, we briefly introduce amoebae, lopsidedness, and relevant concepts in mathematics and physics. In §3, we move onto the ML investigations. Starting with the simplest lopsided amoebae in §3.1 where NNs are applied to learning the genus for various examples. From manifold learning and the neural network structure, we will try to recover or approximate the conditions that determine the genus. We then conduct similar study for more subtle lopsidedness in §3.3. In particular, we will focus on the zeroth Hirzebruch surface F_0 as an illustration. In §3.4, we will briefly comment on the membership problem. Finally, in §4 convolutional neural networks (CNNs) learn to identify the genus for F_0 from Monte Carlo generated amoebae at varying image resolutions. More background and related calculations for amoebae can be found in the appendices. One thing to note is that in Monte Carlo, sampling of points is an important issue, which we discuss in Appendix A.

This paper is a contribution to the special volume “Algebraic Geometry and Machine Learning” in the Journal of Symbolic Computation [28], Springer 2021.

2 Rudiments of Amoebae

In this section, we review some basics of amoebae, with a view toward how they arise in physics. Suppose we are given a multi-complex-variable polynomial

$$P(\mathbf{z}) = \sum_{\mathbf{p}} c_{\mathbf{p}} \mathbf{z}^{\mathbf{p}} = \sum_{i_1, \dots, i_r} c_{i_1, \dots, i_r} z_1^{i_1} \dots z_r^{i_r}, \quad \mathbf{z} \in (\mathbb{C}^*)^r. \quad (2.1)$$

As is customary, we use bold-face to denote the multi-index notation for monomials. This defines an affine algebraic variety as a hyper-surface $P(\mathbf{z}) = 0$ in $(\mathbb{C}^*)^r$. Then, we have the following.

Definition 2.1. *Writing the coordinates as*

$$(z_1, \dots, z_r) := (\exp(s_1 + i\theta_1), \dots, \exp(s_r + i\theta_r)), \quad (2.2)$$

where $s_i \in \mathbb{R}$ and $\theta_i \in [0, 2\pi)$, the **amoeba** \mathcal{A}_P is the image under the projection

$$\text{proj} : (z_1, \dots, z_r) \mapsto (s_1, \dots, s_r) . \quad (2.3)$$

Equivalently, proj is the map

$$(z_1, \dots, z_r) \mapsto (\log |z_1|, \dots, \log |z_r|) . \quad (2.4)$$

We would often abbreviate this as $\text{Log}|\mathbf{z}|$ in our multi-index notation \mathbf{z} .

While the above definition is general, we will exclusively focus on $P(\mathbf{z})$ being a Newton polynomial, which we recall as follows:

Definition 2.2. Let Δ be a convex lattice polytope, which we will also refer to as a **toric diagram**¹, with lattice points $\{\mathbf{p} \in \Delta\}$. We can always write a polynomial, called the **Newton polynomial**, each of whose monomial terms is $\mathbf{z}^{\mathbf{p}}$. The coefficient is an arbitrary non-zero complex number and constitutes one of the moduli. Conversely, for any polynomial in (2.1), one can construct a Newton polytope by forming the convex hull of all the powers which appear in the monomials.

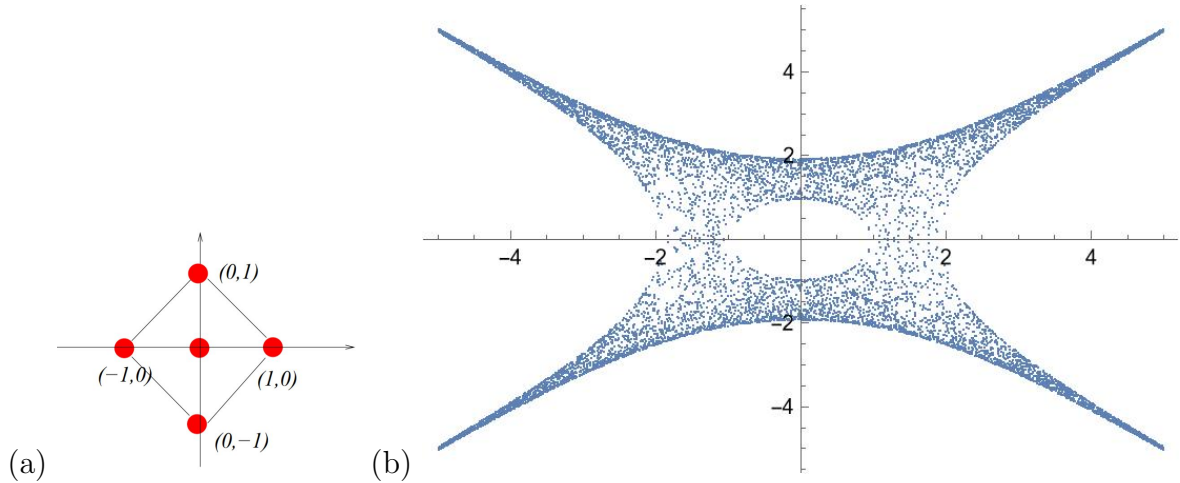


Figure 2.1: (a) A lattice polygon (polytope in 2-D), giving the Newton polynomial $P(z_1, z_2) = c_{(0,0)} + c_{(1,0)}z_1 + c_{(-1,0)}z_1^{-1} + c_{(0,1)}z_2 + c_{(0,-1)}z_2^{-1}$. (b) Taking the coefficients $c_{(0,0)} = 5$ and all others to be 1, the amoeba is the region in blue (the figure is obtained by Monte Carlo).

Let us consider an example. Take the 2-dimensional polytope in Fig. 2.1 (a). The lattice points contained therein are $(0,0)$, $(1,0)$, $(0,1)$, $(-1,0)$, $(0,-1)$. Thus, the

¹We refer the reader to the modern classic [29] for toric varieties and convex polytopes.

Newton polynomial is $P(z_1, z_2) = c_{(0,0)} + c_{(1,0)}z_1 + c_{(-1,0)}z_1^{-1} + c_{(0,1)}z_2 + c_{(0,-1)}z_2^{-1}$, where the five c 's are complex coefficients. Note that strictly speaking, the Newton polynomial is a Laurent polynomial since negative powers appear; these can be removed by shifting the origin of the polytope. The amoeba for the choice of coefficients $5 + z_1 + z_1^{-1} + z_2 + z_2^{-1}$ is shown in Fig. 2.1 (b). This star-shaped region with a hole looks like an amoeba from biology, hence the name. The hole in the middle, along with the four unbounded complementary regions are real points (s_1, s_2) , where no complex numbers (z_1, z_2) satisfying $P(z_1, z_2) = 0$ map to them under the projection 2.1. One may also notice there are four *tentacles* of this amoeba that go to infinity exponentially; these four asymptotic lines of the tentacles are known as the *spines*.

Since the amoeba is obtained from the solutions to $P(\mathbf{z}) = 0$, it is a subset in \mathbb{R}^r . Thus, for one thing, this half-reduction from complex dimension r to real dimension r is good for *visualization*. In particular, for $r = 2$ (where the Newton polytope is a convex lattice polygon), the amoeba is a great way to picture Riemann surfaces. We will exclusively work with $r = 2$. For these, we have some important definitions.

Definition 2.3. *Consider the amoeba \mathcal{A} of the complex curve/Riemann surface $P(z_1, z_2) = 0$.*

- *There are (potentially separated) regions complement to \mathcal{A} , which may or may not be bounded. We shall call the number of such bounded complementary regions the **genus**² of \mathcal{A} . For instance, the amoeba in Fig. 2.1 (b) is genus 1.*
- *The genus counts the number of “holes” in \mathcal{A} , and it explicitly depends on the choice of the coefficients in the Newton polynomial. As we will focus on spectral curves $P(z_1, z_2) = 0$ with real coefficients, these curves are real plane curves in \mathbb{RP}^2 . In particular, the curves for which the numbers of holes (as well as components) are maximal amongst the choices of coefficients are called the **Harnack curves**.*

Membership Problem: Given an amoeba, it is natural to ask how one can determine whether a point in the Log space belongs to the amoeba. This is known as the *membership problem*. From [30–32], we learn that the membership problem can be solved in polynomial time for a fixed dimension. However, to determine the full boundaries of a general amoeba, the typical approach would be approximations using lopsidedness as we shall explain now. The algorithm is more time-consuming and the fastest algorithm so far to our best knowledge was proposed in [33] (see Footnote 4).

²It is curiously appealing that the phrase “the genus of an amoeba” makes sense both in biology and in mathematics! Perhaps the most common species, *Amoeba proteus*, should be associated to a particular class of Riemann surfaces.

2.1 Amoebae and Lopsidedness

To learn the complementary regions of amoebae — and hence its boundaries, genus, and membership decisions — it is useful to introduce the concept of lopsidedness [34].

Definition 2.4. Let $f \in \mathbb{C}[z_1, z_1^{-1}, \dots, z_r, z_r^{-1}]$ be a sum of (Laurent) monomials m_i as $f(\mathbf{z}) = m_1(\mathbf{z}) + \dots + m_k(\mathbf{z})$. For $\mathbf{x} \in \mathbb{R}^r$, define the list of positive numbers

$$f\{\mathbf{x}\} := \{|m_1(\text{Log}^{-1}(\mathbf{x}))|, \dots, |m_k(\text{Log}^{-1}(\mathbf{x}))|\}. \quad (2.5)$$

We say a list of positive numbers is **lopsided** if one of the numbers is greater than the sum of all the others. This definition can then be applied to $f\{\mathbf{x}\}$.

Importantly, a list of positive numbers $\{b_i\}$ is *not lopsided* if one could find a list of phases $\{\phi_i\} \subset \mathbb{C}$ with $|\phi_i| = 1 \forall i$ such that $\sum_i \phi_i b_i = 0$; this follows from the triangular inequality. These exact phases allow a real point on the plane projected to, to have a Log^{-1} lift into complex space that satisfies the amoebae equation $P(\mathbf{z}) = 0$, thus making this real point a member of the amoeba. It is here how not-lopsidedness connects to amoebae membership.

It is then natural to define the following [33]:

Definition 2.5. Given a Newton polynomial P , the **lopsided amoeba**³ is

$$\mathcal{L}\mathcal{A}_P := \{\mathbf{x} \in \mathbb{R}^r \mid P\{\mathbf{x}\} \text{ is not lopsided}\}. \quad (2.6)$$

For some cases, such as $P = z_1 + z_2 + 1$, $\mathcal{L}\mathcal{A}_P = \mathcal{A}_P$. However, in general, they do not need to coincide. Nevertheless,

$$\mathcal{A}_P \subseteq \mathcal{L}\mathcal{A}_P, \quad (2.7)$$

so that $\mathcal{L}\mathcal{A}_P$ can be constructed as a crude approximation to \mathcal{A}_P . This can be made precise as follows.

Let n be a positive integer, $\mathbf{x} \in \mathbb{R}^r$, and $P(\mathbf{x})$ a (Newton) polynomial, define \tilde{P}_n to be⁴

$$\tilde{P}_n(\mathbf{x}) := \prod_{k_1=0}^{n-1} \dots \prod_{k_r=0}^{n-1} P(e^{2\pi i k_1/n} x_1, \dots, e^{2\pi i k_r/n} x_r). \quad (2.8)$$

Clearly, $\tilde{P}_1 = P$. Such \tilde{P}_n is in fact a *cyclic resultant*

$$\tilde{P}_n = \text{res}_{u_r} \left(\text{res}_{u_{r-1}} \left(\dots \text{res}_{u_1} \left(P(u_1 x_1, \dots, u_r x_r), u_1^n - 1 \right) \dots, u_{r-1}^n - 1 \right), u_r^n - 1 \right) \quad (2.9)$$

³The appellation might seem a bit confusing since the lopsided amoeba is the set where the $P\{\mathbf{x}\}$ is *not* lopsided.

⁴In [33], a faster algorithm was proposed to compute $\mathcal{L}\mathcal{A}_{\tilde{P}_n}$ at level k where $n = 2^k$ using the properties of cyclic resultants. The time complexity is $\mathcal{O}(kd^2)$ with d being the degree of $P(z_1, z_2)$.

where $\text{res}_u(f, g)$ is the resultant of f, g with respect to the variable u .

The lopsided amoeba $\mathcal{L}\mathcal{A}_{\tilde{P}_n}$ for \tilde{P}_n approximates \mathcal{A}_P itself [34]:

Theorem 2.1. *For an r -dimensional Newton polytope $\Delta(P)$, with polytope coordinates p_i for each i^{th} direction in the \mathbb{Z}^r lattice which the polytope is defined in, one defines $c_i := \max(p_i) - \min(p_i)$ over the polytope vertices; then $c = \max(c_i)$. Suppose $\mathbf{x} \in \mathbb{R}^r \setminus \mathcal{A}_P$ is a point in the amoeba complement whose distance from \mathcal{A}_P is at least $\epsilon > 0$. If n is large enough so that*

$$n\epsilon \geq (r-1)\log n + \log((r+3)2^{r+1}c), \quad (2.10)$$

then $\tilde{P}_n\{\mathbf{x}\}$ is lopsided⁵ and $\mathcal{L}\mathcal{A}_{\tilde{P}_n}$ converges uniformly to \mathcal{A}_P as $n \rightarrow \infty$.

A consequence of this is one way to solve the membership problem:

Theorem 2.2. *Let $I \subset \mathbb{C}[z_1, z_1^{-1}, \dots, z_r, z_r^{-1}]$ be an ideal. The point $\mathbf{x} \in \mathbb{R}^r$ is in the amoeba \mathcal{A}_I if and only if $f\{\mathbf{x}\}$ is not lopsided for every $f \in I$.*

Therefore, to fully determine the boundary of an amoeba, we need to consider all the Laurent polynomials in the ideal generated by our Newton polynomial. Equivalently, we need to take $n \rightarrow \infty$ for the cyclic resultant. As a result, we often approximate the boundary with some finite large n in practice. This would also be our basic strategy to study the genus using neural networks later, although unlike finding the boundary, sometimes we can count the genus in other ways, as we will see.

Example For instance, consider the Newton polynomial $P = z^3 + w^3 + 2zw + 1$ whose amoeba is plotted red in Figure 2.2. The dark blue points (plus the red ones) form the lopsided amoeba $\mathcal{L}\mathcal{A}_{\tilde{P}_{16}}$ while the chartreuse points (plus the red and dark blue ones) give the region of $\mathcal{L}\mathcal{A}_{\tilde{P}_8}$.

2.2 Amoebae from D-branes

As mentioned in the introduction, amoebae are also interesting objects from the viewpoints of physics. Historically, amoebae have a strong connection to dimer models where $P = 0$ acts as the spectral curve of the bipartite graph [4, 5]. In particular, for a dimer with positive weights on its edges, $P = 0$ is a *Harnack curve*. D-branes probing toric Calabi-Yau (CY) manifolds has been very well studied in the past few decades especially in the context of quiver⁶ gauge theories [36–38]. Brane tilings are precisely dimer models that successfully show how one can connect, for example, the

⁵In this paper, as r is always 2, we have $n\epsilon \geq \log n + \log(8c)$.

⁶q.v. [35] for a recent ML treatment of quiver gauge theories.

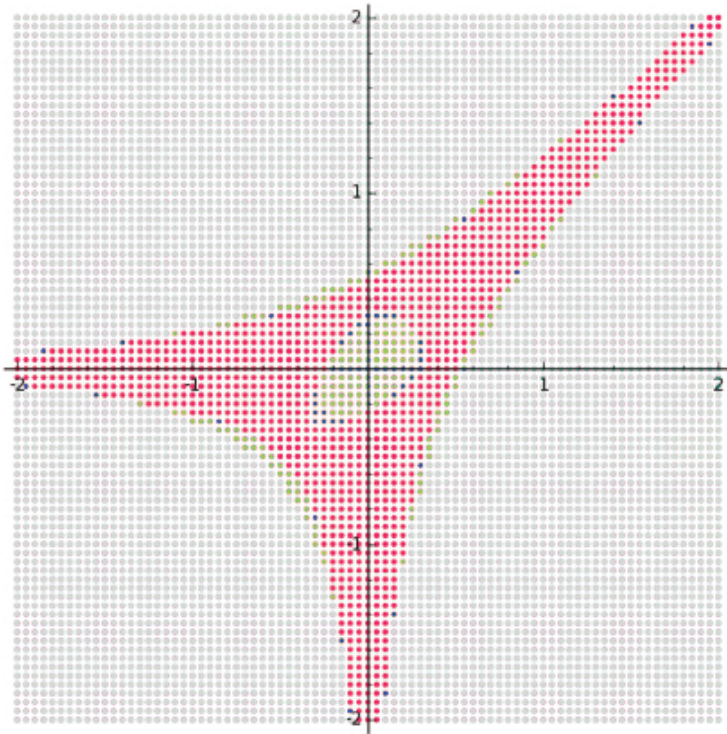


Figure 2.2: It is clear from this example that the lopsided amoeba contains the amoeba as a subset. Figure taken from [33, Figure 1].

world-volume theories of a stack of D3-branes and the toric geometry of CY threefolds [39–41]. It is also well-known that under mirror symmetry, this gives rise to a description in terms of D6-branes wrapping the \mathbb{T}^3 fibres [42, 43].

Briefly, as we are considering 2d lattice polygons, we will denote the Newton polytope as $P(z, w)$ with $z, w \in \mathbb{C}^*$, note this is the convention where the surface is in $(\mathbb{C}^*)^2$ such that subsequent use of (z, w) over (z_1, z_2) implies this. As shown in [42], the mirror geometry of a CY_3 whose toric diagram is $\Delta(P)$ is the local threefold $P(z, w) = uv$ where u, v are complex variables. As discussed in [6], the curve $P(z, w) = 0$ plays a crucial role in geometry and quiver theories. This is a punctured Riemann surface Σ of genus g where g equals the number of interior points of the Newton polygon.

Furthermore, the (p, q) -webs are also related to the above CY_3 and the Riemann surface Σ . If we wrap the 5-branes on Σ and compactify the theory on a torus, we can get the theory for D3-branes probing CY_3 after performing T-dualities on the two directions of the torus. In fact, the toric diagram reveals a simple connection: the dual graph of this toric polygon is exactly the (p, q) -web diagram. The number of the boundary points of the toric diagram is equal to the number of NS5 cycles in the brane

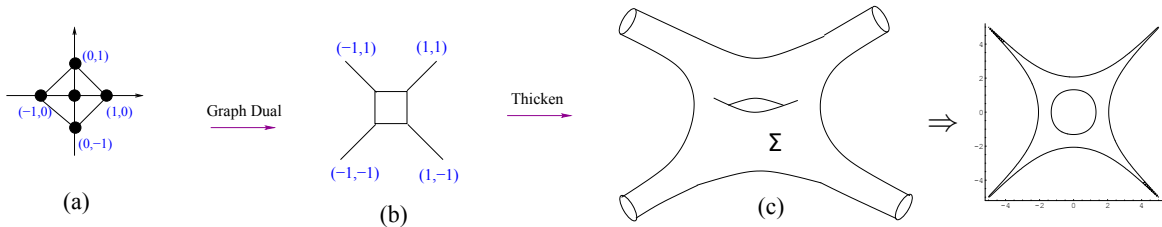


Figure 2.3: Figure taken from [6] (figure 3): (a) The toric diagram of F_0 . (b) The (p, q) -web is the dual graph of the toric diagram. (c) The holomorphic Riemann surface Σ is a thickening of the web diagram. To the right we also include the amoeba for reference, this is a deformation retract of Σ .

system. As shown in [6], the Riemann surface Σ can also be thought of as the thickening of the web diagram; conversely, the web is a deformation retract of the amoeba.

As an example, the lattice polygon in Figure 2.1 (a) is actually the toric diagram of the affine Calabi-Yau cone over the zeroth Hirzebruch surface $F_0 \cong \mathbb{P}^1 \times \mathbb{P}^1$. We summarize the above concepts in Figure 2.3, using this example. How one can obtain the brane tiling on \mathbb{T}^2 and the intersection locus of D6s on Σ was studied in [6]. Following Definition 2, we can take the projection from $(\mathbb{C}^*)^2$ to \mathbb{R}^2 to obtain the corresponding amoeba and plot it on the Log plane.

We can see our familiar concepts such as genus, tentacles, spine, and thickening from the figure. If this thickening is thick enough, the hole in the above example might disappear. For any general amoeba, the number of its genus is controlled by the coefficients of the Newton polynomial. For instance, the amoeba in Figures 2.1 and 2.3 would become genus 0 if $c \leq 4$ for $P(z, w) = z + w + 1/z + 1/w + c$.

Therefore, the coefficients in front of the monomials in the Newton polynomial are of particular interest in the study of amoebae as they determine the amoebae boundaries. In general, the coefficients can be any complex numbers and they are mirror to the Kähler moduli of the toric geometry. In dimer models, we are mainly interested in positive integer coefficients since they count perfect matchings [41]. For more details, one is referred to [6] as well as the review [11]. In this paper, we will consider real coefficients as our moduli.

3 ML Amoebae from Coefficients

Having introduced all the basics and hopefully having motivated the readers, we shall study amoebae from the novel perspectives of machine learning. We will consider the problems of computing genus and finding membership using feed-forward and convolutional neural networks, as well as using manifold learning. Unless specified, the architectures are as follows:

MLP The multilayer perceptrons (MLPs) or more commonly known today as feed-forward NN, all have the structure with one hidden layer with 100 perceptrons/single-neurons. We will always use ReLU as the activation function⁷ for MLPs.

CNN The CNNs consist of four 1d convolutional layers, each followed by a LeakyReLU layer then a 1d MaxPooling layer.

For all our NNs, the *learning rates* are all 0.001 and we always use the Adam optimizer [45]. All computation time is matter of a few minutes even though the structure is not minimized. Notice that it is possible to further reduce the structures for these MLPs and CNNs. In general, a not-carefully-tuned network can still give similar good performance. Therefore, the structures mentioned here are not so important. In the following, we will reduce the network to find the minimal structure whenever necessary. This independence of detailed architecture is again consistent with [24] which observed that, generically across disciplines, the robustness and absence of noise in pure mathematical data tend to give similar accuracies for rather different methods of NNs and classifiers.

3.1 Lopsided Amoebae: $n = 1$

As discussed above, we will use lopsided amoebae since they are more amenable to computation. We begin with the simplest case of $n = 1$ where $\tilde{P}_1 = P$ by definition, so that $\mathcal{L}\mathcal{A}_{\tilde{P}_1} = \mathcal{L}\mathcal{A}_P$. Can one predict the genus without explicit computation?

For a fixed Newton polynomial $P(z, w) = \sum c_k z^i w^j$, the input is the vector composed of the coefficients, i.e., $\{c_1, c_2, \dots, c_n\}$, and the output is the genus. In other words, we have labelled data of the form

$$\{c_1, c_2, \dots, c_n\} \longrightarrow g . \quad (3.1)$$

Let us use some standard classifiers on this problem, in the spirit of [24].

3.1.1 Example 1: F_0

We start with our simple running example F_0 whose toric diagram was given in Figure 2.1 (a). The Newton polynomial is

$$P(z, w) = c_1 z + c_2 w + c_3 z^{-1} + c_4 w^{-1} + c_5 . \quad (3.2)$$

⁷The rectified linear unit (ReLU) function is defined as $\text{ReLU}(q) := \max(q, 0)$. Similarly, $\text{LeakyReLU}(q)$ is defined to be αq when $q < 0$ for some constant α and q when $q \geq 0$. In our CNN models, we will take $\alpha = 0.1$. Inspired by [44] single-layer ReLU NNs were selected as universal function approximators.

Hence, our input is $\{c_1, c_2, c_3, c_4, c_5\}$. Since the resulting lopsided amoeba could have at most one genus (corresponding to its single interior point), this is a binary classification where the output is either $g = 0$ or $g = 1$. For this simple example, one can analytically derive (see Appendix C) the genus g as a function of the coefficients:

$$g = \begin{cases} 0, & |c_5| \leq 2|c_1c_3|^{1/2} + 2|c_2c_4|^{1/2} \\ 1, & |c_5| > 2|c_1c_3|^{1/2} + 2|c_2c_4|^{1/2} \end{cases} \quad (3.3)$$

Nevertheless, we can see from the RHS that the boundaries of decision are already non-trivial even for this simplest of examples.

Real Coefficients in Newton Polynomial: We generate a balanced dataset with ~ 2000 random samples, with $c_{1,2,3,4} \in [-5, 5]$ and $c_5 \in [-20, 20]$. Our forward-feed NN (MLP) can easily reach $0.957(\pm 0.005)$ accuracy (with 95% confidence interval) for a 5-fold cross validation⁸.

To see how well the NN is learning the above equation for g , we can perform principal component analysis (PCA) projections, as visualized in Figure 3.1(a). With the help of the `Yellowbrick` package [46], we can use multi-dimensional scaling (MDS) manifold projection to make the two types of data points further separated as in Figure 3.1(b). For a brief introduction to different methods of manifold learning, see Appendix B.

To get an idea about how these data points are distributed in the projection, let us define⁹ $x \equiv c_5$ and $|y| \equiv 2|c_1c_3|^{1/2} + 2|c_2c_4|^{1/2}$. Then the inequalities in (3.3) has the boundary lines $y = \pm x$. As depicted in Figure 3.1(c), the two lines divide the projection plane into different regions, where in the blue region we have genus zero while in the green region we have genus one. The equivalent separation coming from our NN in Figure 3.1(a,b) is represented by the red lines in (c). We can see that they do not intersect at the origin and the $g = 0$ region would occupy some of the $g = 1$ parts. This could account for the 0.95 accuracy (rather than 100%).

Why is there a shift of the boundary lines? We believe that this is due to the non-trivial expression of y (especially the square roots therein). Although it is possible for an NN to learn such expression in principle, it could still be too complicated for a simple NN to fully simulate this.

⁸We have also tried some other classification models such as support vector machine, random forests etc. It turns out that these models would have nice performance with accuracies around $0.900(\pm 0.010)$ for 5-fold cross validation although the MLP behaves slightly better. Henceforth, we shall always stick to MLP.

⁹As discussed in Appendix C, (3.3) works for all complex coefficients. However, we are only using real input vectors here, and more importantly, only the absolute values would matter in the condition. Hence, we can set $c_5 = x \in \mathbb{R}$ in the projection.

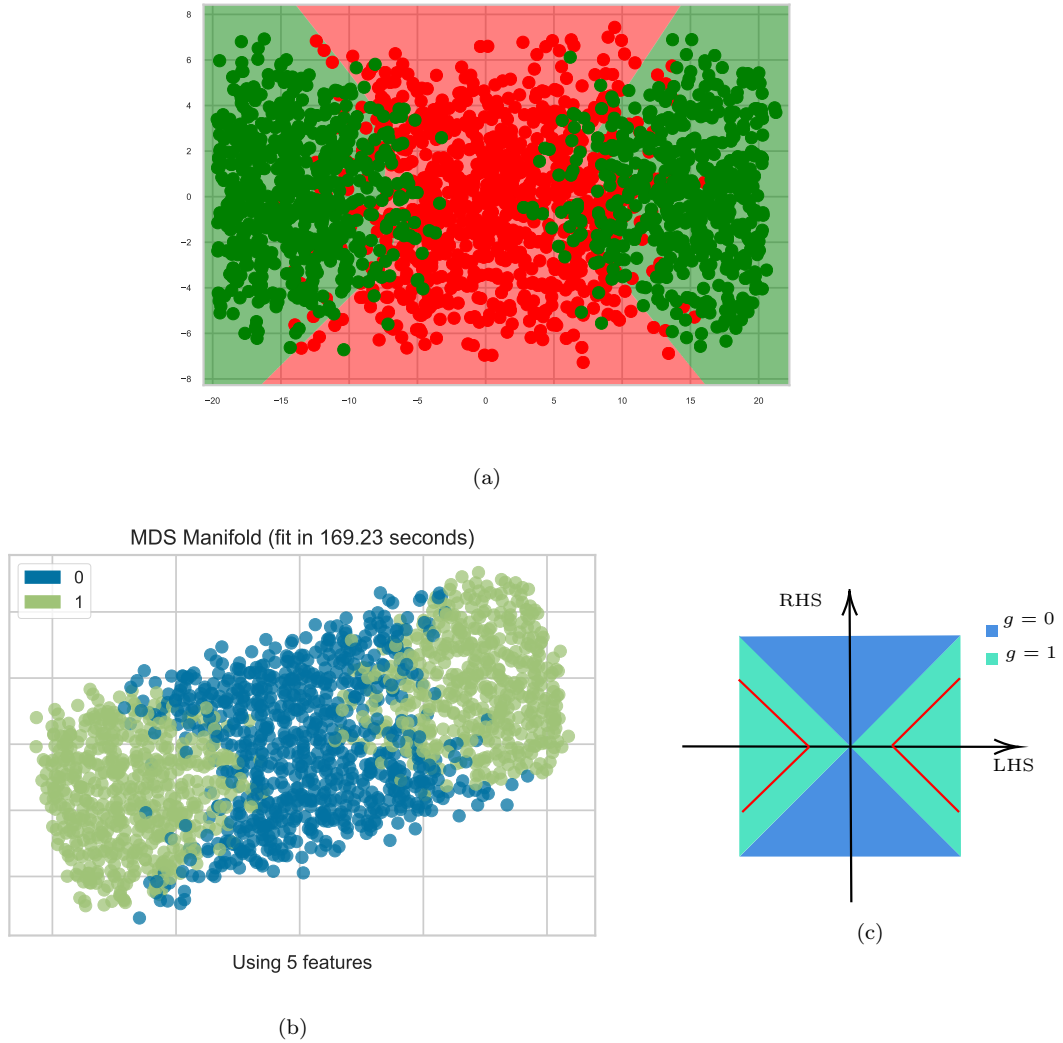


Figure 3.1: (a) The kernel PCA projection for an NN. (b) The MDS manifold projection which gives a better separation of the two classes of points. (c) Ideally, the blue and green regions would be separated by $y = \pm x$. In practice, due to the complication of square roots, the NN would get shifted. This shift, i.e., the actual separation of the blue and green points, is indicated by the red lines here.

There is another useful projection in this example, that is, the spectral embedding visualization [46] as shown in Figure 3.2(a). As we can see, this gives a distribution in a parabola shape. We argue that this comes from squaring $y = \pm x$, i.e., $Y(\equiv y^2) = x^2$. However, again due to the complicated expression, it would be very hard for a simple NN to fully recover Y . Hence, there could still be some mixing at the boundary parts. As a validation, we generate a dataset whose binary classes are separated by the bound

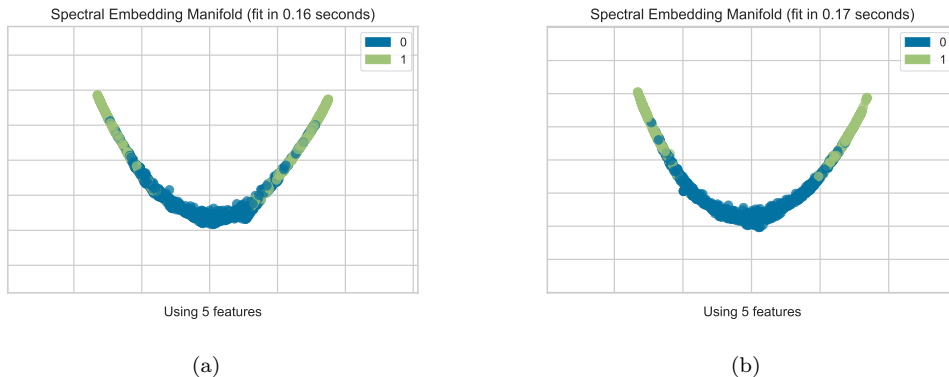


Figure 3.2: (a) The spectral embedding manifold projection of the dataset for F_0 . For the input vectors, $c_{1,2,3,4}$ range from -5 to 5 while c_5 ranges from -20 to 20 . (b) To verify our explanation for (a), we generate a dataset whose two classes are separated by $c_5^2 = 4|c_1c_3| + 4|c_2c_4| + c_0$ (with c_i' the same range as c_i), where $c_0 = 4 \times 2 \times 2.5^2$ as 2.5 is the average of the possible values for $c_{1,2,3,4}$.

$c_5^2 = 4|c_1c_3| + 4|c_2c_4| + 50$ and plot its spectral embedding as in Figure 3.2(b). Indeed, we obtain a similar parabola-shaped projection. Moreover, since the square root part is replaced by the constant 50 , NN could also give higher accuracy (~ 0.97 at same training percentage).

From (3.3), we also learn that only the absolute values of the coefficients would matter. In fact, this could be reflected in machine learning as well. We can use the same dataset, but with $|c_i|$'s as input. Indeed, this helps the model to improve its performance and a 5-fold cross validation now gives $0.987(\pm 0.007)$ accuracy.

Integer Coefficients in Newton Polynomial: As discussed in §2.2, the coefficients c_i have significance: they count perfect matching in dimer models and so-called GLSM quantum fields in brane tilings in string theory. Thus, positive integer coefficients are of particular interest. We therefore generate ~ 2000 data points with only positive integer inputs ($c_{1,2,3,4} \in [1, 5]$, $c_5 \in [1, 20]$) for machine learning as well. It turns out that the accuracy is further improved to $0.992(\pm 0.004)$ for 5-fold cross validation. We summarize the results, contrasting the input type, in Table 3.1. We also

Input type	\mathbb{R}^5	$(\mathbb{R}^+)^5$	$(\mathbb{Z}^+)^5$
Accuracy	$0.957(\pm 0.005)$	$0.987(\pm 0.007)$	$0.992(\pm 0.004)$

Table 3.1: The performance for MLP using 5-fold cross validation with 95% confidence interval. The input is a five vector of the coefficients c_i in the Newton polynomial for F_0 , and we contrast the 3 different types of choices for c_i .

plot the MDS projections for the positive inputs in Figure 3.3.

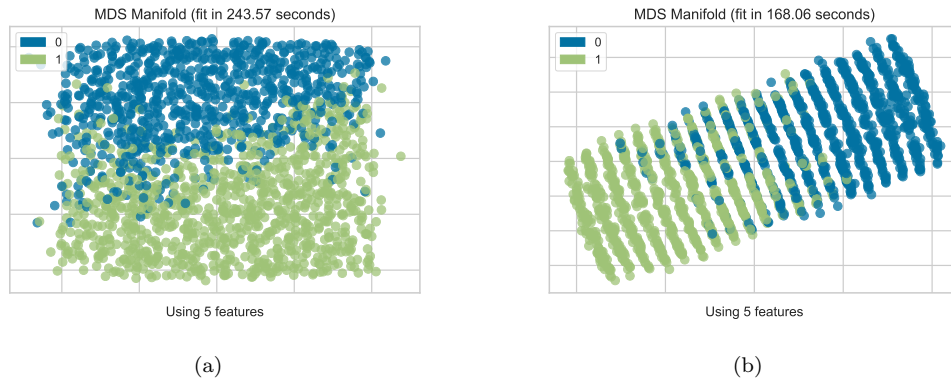


Figure 3.3: (a) The MDS manifold projection for positive real input. (b) The MDS manifold projection for positive integer input. Since we are taking positive values, the plots would correspond to the first quadrant in Figure 3.1.

3.1.2 Example 2: $L^{3,3,2}$

Let us consider a non-reflexive example with two interior points ¹⁰, viz., $L^{3,3,2}$ whose toric diagram is



The Newton Polynomial is

$$P(z, w) = c_1 z + c_2 w + c_3 z^{-1} + c_4 w^{-1} + c_5 z^2 + c_6. \quad (3.5)$$

Hence, our input is $\{c_1, c_2, c_3, c_4, c_5, c_6\}$. Since the resulting lopsided amoeba could have at most two holes (corresponding to its two interior points), this is a ternary classification where the output can be $g = 0, 1, 2$.

As with F_0 , we can analytically find the boundary decisions, though the expressions are much more complicated. The details are presented in Appendix C, and we summarize them here:

$$g = \begin{cases} 0, & |c_1| \leq a_1 \text{ and } |c_6| \leq a_2 \\ 1, & (|c_1| > a_1 \text{ and } |c_6| \leq a_2) \text{ or } (|c_1| \leq a_1 \text{ and } |c_6| > a_2), \\ 2, & |c_1| > a_1 \text{ and } |c_6| > a_2 \end{cases} \quad (3.6)$$

¹⁰The 16 reflexive polygons were considered in the context of brane-tiling in [47, 48], and the 45 non-reflexives with 2 interior points, in [49].

where $a_1 := |c_2|w_0/z_0 + |c_3|/z_0^2 + |c_4|/(z_0w_0) + |c_5|z_0 + |c_6|/z_0$ and $a_2 := |c_1|z'_0 + |c_2|w_0 + |c_3|/z'_0 + |c_4|/w_0 + |c_5|z_0'^2$ such that $z_0 := \sqrt[3]{-\frac{q}{2} + \sqrt{\Delta}} + \sqrt[3]{-\frac{q}{2} - \sqrt{\Delta}}$, with $\Delta := \left|\frac{c_3}{c_5}\right|^2 - \frac{(2|c_2c_3|^{1/2} + |c_6|)^3}{27|c_5|^3}$, and z'_0 is the positive root of $2|c_5|z_0'^3 + |c_1|z_0'^2 - |c_3| = 0$.

We generate a balanced set with only ~ 9000 random samples. A 5-fold cross validation for MLP gives $0.912(\pm 0.006)$ accuracy (with 95% confidence interval).

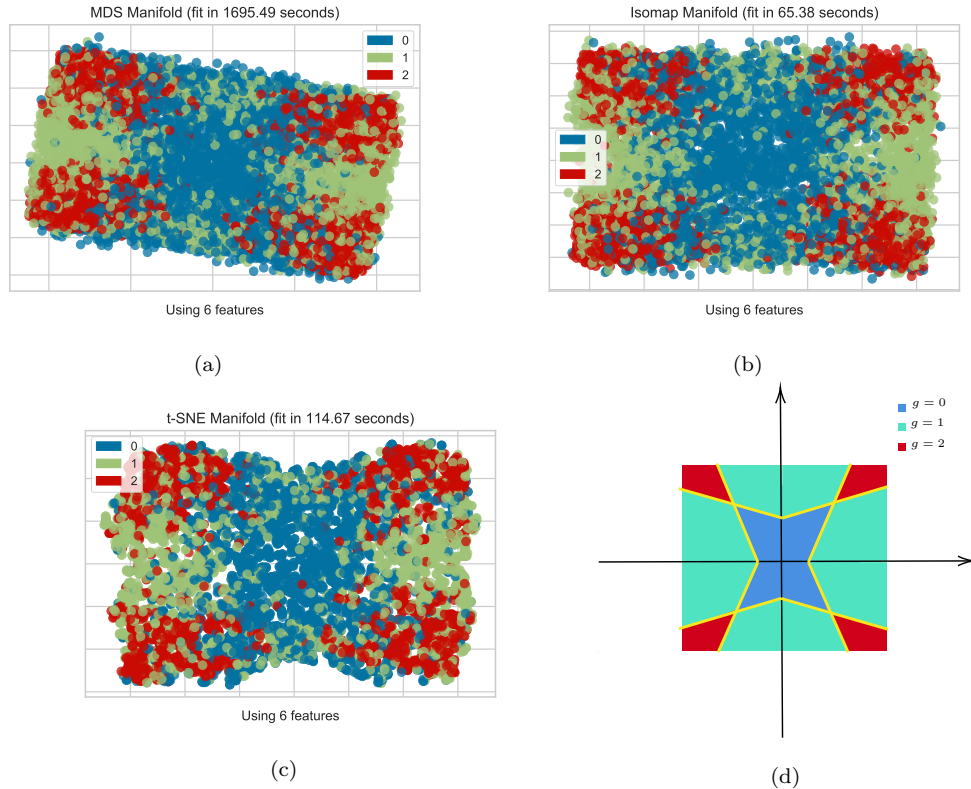


Figure 3.4: (a) The MDS embedding manifold projection of the dataset for $L^{3,3,2}$. (b) The Isomap embedding manifold projection. (c) The t-SNE embedding manifold projection. For instructions on these manifold projections, one is referred to [46]. (d) The sketch of an ideal separation of the data points.

Again, we can plot different manifold projections to visualize how the NN is simulating the above condition as shown in Figure 3.4(a,b,c). We can see that different projections give similar data separations. To understand such decision regions, let us first write the two bounds as

$$\begin{aligned} |c_1| &= a_1 \equiv |c_6|/z_0 + b_1, \\ |c_6| &= a_2 \equiv |c_1|z'_0 + b_2, \end{aligned} \tag{3.7}$$

where $b_i > 0$. In other words, when $g = 2$, the region is bounded by $|c_6|/z_0 + b_1 < |c_1| < |c_6|/z'_0 - b_2$. This corresponds to the red region in Figure 3.4(d) whose horizontal and vertical axes are $x \equiv \pm|c_6|$ and $y \equiv \pm|c_1|$ respectively. Notice that z_0, z'_0, b_i are not constants, but we can always draw some boundary lines (in yellow) as a sketch (assuming that the changes of the boundary lines are small compared to different coloured regions¹¹).

Likewise, it is straightforward to find the regions for $g = 0$ (in blue) and $g = 1$ (in green). Indeed, this is the decision regions we get from those different projections. As z_0, z'_0 and b_i are not really constants and have some complicated expressions, it is natural to see some mixings in the projections. With the same dataset, we can also take the absolute values of the coefficients as input since only $|c_i|$ is relevant for counting the genus. For 5-fold cross validation, the accuracy is significantly improved to $0.968(\pm 0.004)$.

As before, we may also generate a dataset of similar size with only positive integer coefficients. For 5-fold cross validation, the accuracy is improved to $0.990(\pm 0.003)$. We summarize the results in Table 3.2. We also plot the MDS projections as an example

Input type	\mathbb{R}^6	$(\mathbb{R}^+)^6$	$(\mathbb{Z}^+)^6$
Accuracy	$0.912(\pm 0.006)$	$0.968(\pm 0.004)$	$0.990(\pm 0.003)$

Table 3.2: The performance for MLP using 5-fold cross validation with 95% confidence interval.

for the positive inputs in Figure 3.5.

3.1.3 Example 3: $\mathcal{C}/(\mathbb{Z}_2 \times \mathbb{Z}_4)$

We now contemplate an example with three interior points, that is, $\mathcal{C}/(\mathbb{Z}_2 \times \mathbb{Z}_4)$ with actions $(1, 0, 0, 1); (0, 1, 3, 0)$, an Abelian quotient of the conifold \mathcal{C} . Its toric diagram is



$$(3.8)$$

and the Newton polynomial is

$$P(z, w) = c_0 + c_1z + c_2z^2 + c_3z^3 + c_4z^4 + c_5w + c_6zw + c_7z^2w + c_8z^3w + c_9z^4w + c_{10}w^2 + c_{11}zw^2 + c_{12}z^2w^2 + c_{13}z^3w^2 + c_{14}z^4w^2. \quad (3.9)$$

¹¹For instance, here $c_{1,6}$ are generated in the range $[-20, 20]$, and the means are $\mu(1/z_0) = 0.31, \mu(1/z'_0) = 2.01, \mu(b_1) = 6.48, \mu(b_2) = 9.18$ with standard deviations $\sigma(1/z_0) = 0.18, \sigma(1/z'_0) = 0.77, \sigma(b_1) = 3.72, \sigma(b_2) = 3.07$. We will not restate this explicitly for the examples discussed below. The complicated expressions for b_i and the standard deviations could account for the misclassifications in machine learning.

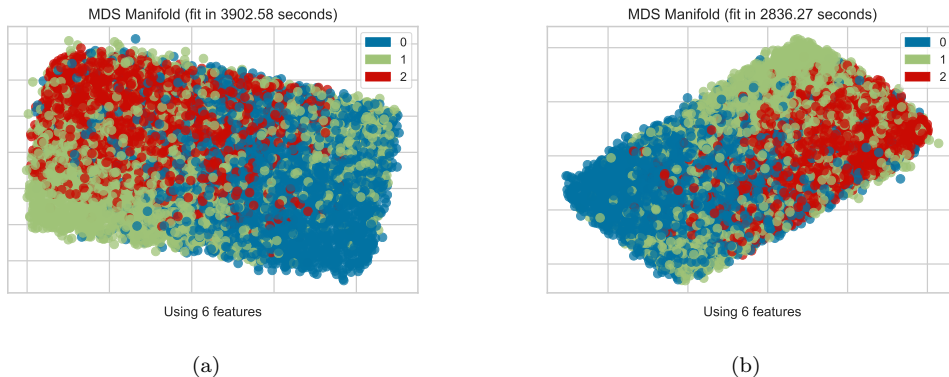


Figure 3.5: (a) The MDS manifold projection for positive real input. (b) The MDS manifold projection for positive integer input. Since we are taking positive values, the plots would correspond to the first quadrant in Figure 3.4.

Hence, our input is $\{c_0, c_1, c_2, \dots, c_{14}\}$. Since the resulting lopsided amoeba could have at most three genera (corresponding to its three interior points), this is a classification where the output can be $g = 0, 1, 2, 3$.

We generate ~ 12000 random samples with $c_i \in [-30, 30]$, using the coefficient vectors and their absolute values respectively as inputs for two experiments. We also generate ~ 12000 random coefficient vectors with only positive integer entries. It turns out that a CNN has a better performance than MLP. We list the results for 5-fold cross validation in Table 3.3.

Input type		\mathbb{R}^{15}	$(\mathbb{R}^+)^{15}$	$(\mathbb{Z}^+)^{15}$
Accuracy	MLP	0.792(± 0.006)	0.856(± 0.005)	0.890(± 0.005)
	CNN	0.803(± 0.009)	0.909(± 0.007)	0.907(± 0.002)

Table 3.3: The performances for MLP and CNN using 5-fold cross validation with 95% confidence interval.

To understand how the model does the classification, we plot the Isomap embedding manifold projection in Figure 3.6. Likewise, we can write down the set of the monomials of $P(z, w)$ and check different lopsided cases to find the regions for different g though there is no general formulae to solve the equations and find analytic expressions for different regions in terms of the coefficients as variables. As a sketch, the lopsidedness property leads to the following inequalities:

$$\begin{aligned}
 |c_8| &< a_1|c_6| - a_2|c_7| - a_3, \\
 |c_8| &< -b_1|c_6| + b_2|c_7| - b_3, \\
 |c_8| &> d_1|c_6| + d_2|c_7| + d_3,
 \end{aligned} \tag{3.10}$$

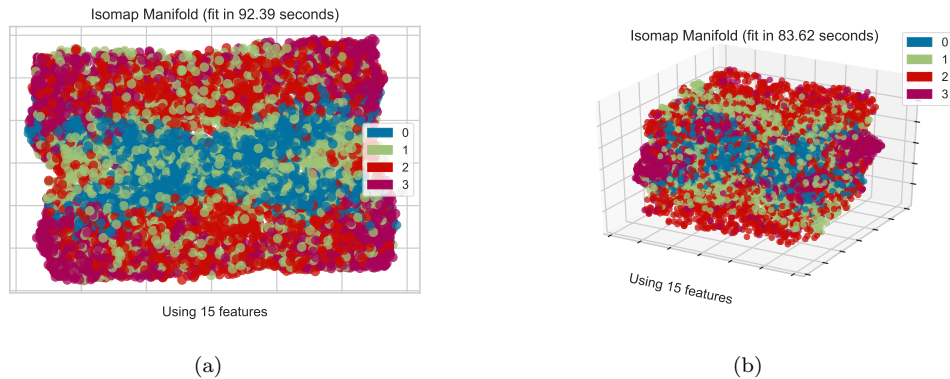


Figure 3.6: (a) The Isomap manifold projection for real input. (b) The Isomap manifold 3d projection for the same input.

where a_i, b_i, d_i are complicated *positive* expressions in terms of $|c_i|$.

Then for fixed coefficients, g is equal to the number of these inequalities being satisfied. As our data is generated with coefficients in certain finite ranges, we can still assume that the changes of a_i, b_i, d_i are small compared to the coloured regions. The schematic is depicted in Figure 3.7, where for convenience only the first quadrant is shown. As we can see, the 2d projection in Figure 3.6(a) is actually viewing the region from the “side” (rather than “front” or “back”)¹² while the 3d projection in Figure 3.6(b) is also consistent with Figure 3.7.

For reference, we also plot the Isomap manifold projection for positive real and positive integer coefficients in Figure 3.8 (a) and (b) respectively. Again, the machine is viewing the different regions from “the side”. This agrees with the sketch in Figure 3.9, and we see that decision regions are better simulated than before, which therefore gives higher accuracies, as the coefficients are all positive (or equivalently, $|c_i|$).

3.1.4 Example 4: $K^{4,5,3,2}$

For more complicated cases with more interior points, it is harder to analyze the visualizations of different bounds for the genus. Here we briefly discuss this with an example, the so-called $K^{4,5,3,2}$ space whose toric diagram is¹³

$$(3.11)$$

¹²One consequence of viewing from the “side” is that $g = 0$ and $g = 3$ are very well separated. Indeed, if we only use data with $g = 0, 3$ for binary classification. The accuracy could easily reach over 0.99.

¹³For convenience, the nomenclature for $K^{a,b,c,d}$ follows [49].

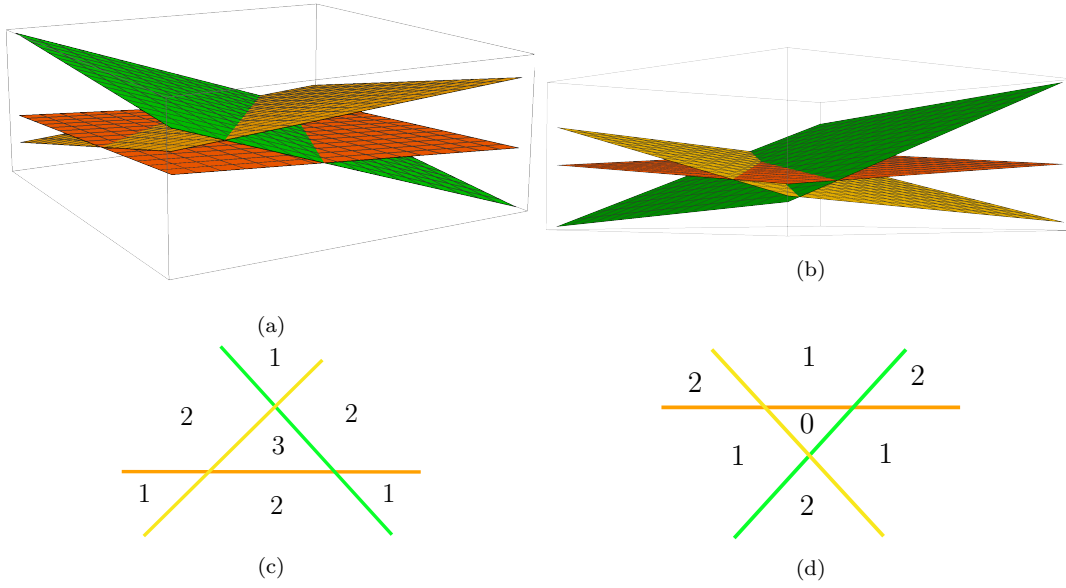


Figure 3.7: The bounds for different g 's in $|c_{6,7,8}|$ coordinates. (a) The bounds for different regions viewed from the “front”. (b) The bounds for different regions viewed from the “back”. The orange plane corresponds to $|c_8| = d_1|c_6| + d_2|c_7| + d_3$. The green plane is $|c_8| = a_1|c_6| - a_2|c_7|z - a_3$. The yellow one is $|c_8| = -b_1|c_6| + b_2|c_7| - b_3$. (c) The 2d cross section of (a) viewed from the “front”. (d) The 2d cross section of (b) viewed from the “back”. The number in each region is the genus.

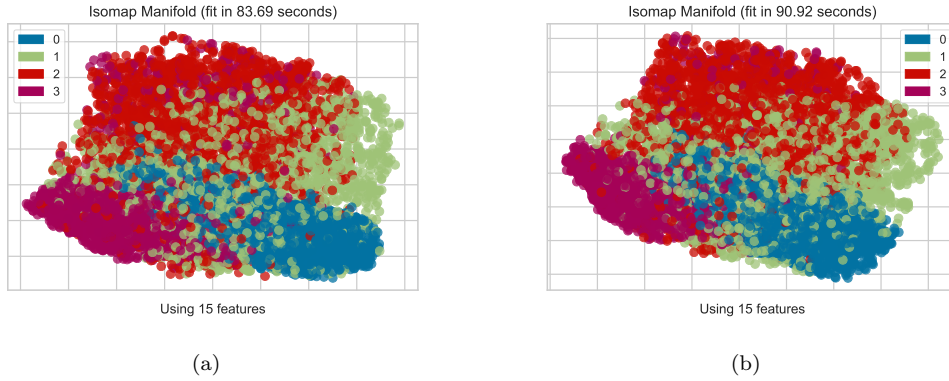


Figure 3.8: (a) The Isomap manifold projection for positive real input. (b) The Isomap manifold projection for positive integer input.

The Newton polynomial is

$$P(z, w) = c_0 + c_1z + c_2z^2 + c_3w + c_4zw + c_5z^2w + c_6w^2 + c_7zw^2 + c_8z^2w^2 + c_9z^3w^2 + c_{10}z^2w^3. \tag{3.12}$$

Hence, the input is $\{c_0, c_1, \dots, c_{10}\}$, and the classification output is $g = 0, 1, 2, 3, 4$. We generate a balanced data-set with ~ 25000 random samples with $c_i \in [-30, 30]$. For

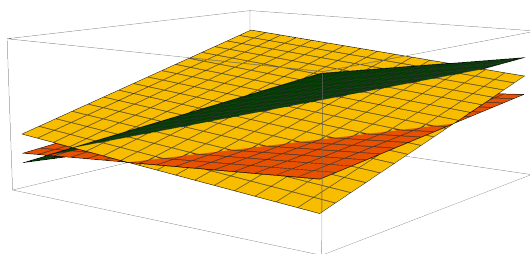


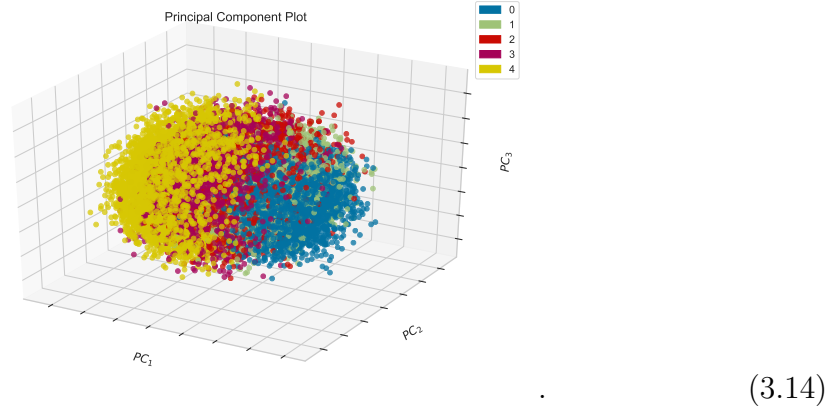
Figure 3.9: The bounds for $\mathcal{C}/(\mathbb{Z}_2 \times \mathbb{Z}_4)$, viewed from “the side”. This is exactly what the NN gets in Figure 3.8 (notice that some red and green points are covered by blue and purple points near the middle as the projection is 2d).

real coefficients, MLP would only give $0.606(\pm 0.003)$ accuracy for 5-fold cross validation while CNN would get $0.671(\pm 0.013)$ accuracy. However, if we restrict the coefficients to be positive (or equivalently, taking their absolute values), the accuracy for MLP would be increased to $0.706(\pm 0.006)$. Moreover, CNN could reach $0.815(\pm 0.006)$ accuracy. It would also be reasonable to expect further improvements of the performance with larger datasets, as well as optimisation over the NN hyperparameters (including the consideration of higher depth networks which can exhibit faster convergence).

As before, we can write the conditions for the genus as

$$\begin{aligned}
 |c_4| &> a_1|c_5| + a_2|c_7| + a_3|c_8| + a_4, \\
 |c_5| &> b_1|c_4| + b_2|c_7| + b_3|c_8| + b_4, \\
 |c_7| &> d_1|c_4| + d_2|c_5| + d_3|c_7| + d_4, \\
 |c_8| &> e_1|c_4| + e_2|c_5| + e_3|c_7| + e_4,
 \end{aligned}
 \tag{3.13}$$

where a_i, b_i, d_i, e_i are again complicated expressions in c_i . For our finite data-set, we may assume that they have small changes compared to the regions of data points and hence think of them as hyperplanes in the 4-dimensional system. Then for fixed $|c_i|$, g is equal to the number of these inequalities being satisfied. It is difficult to visualize the projections as in the previous subsections. Nevertheless, let us still use PCA to project the input vectors to 3d.



Although it is hard to tell what the three principal components are, we can still see that points in different colours are arranged from left to right (with mixings).

Another way to visualize in 3d is to consider 3-dimensional slices in the 4-dimensional system. For instance, we can fix $|c_8|$ and different plots should correspond to different cross sections of the 4d “plot”. In Figure 3.10, we plot several slices by restricting $|c_8|$ to small ranges as our data consists of vectors with random real entries. We find that the distribution of coloured points in each slice looks similar to (3.14), where the numbers of points with larger g decrease and those with smaller g increase when we decrease the value of $|c_8|$.

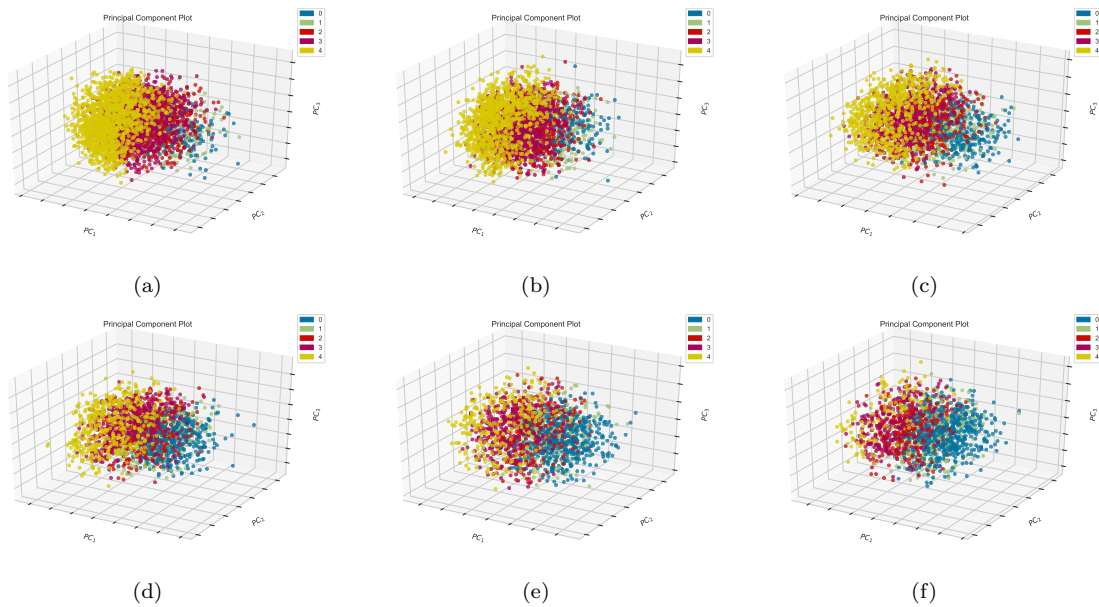


Figure 3.10: The PCAs of different slices for (a) $27 < |c_8| \leq 30$, (b) $24 < |c_8| \leq 27$, (c) $21 < |c_8| \leq 24$, (d) $18 < |c_8| \leq 21$, (e) $15 < |c_8| \leq 18$ and (f) $12 < |c_8| \leq 15$.

Of course, it is also possible to project an input vector to one with 4 components. We will discuss this in §3.2.1.

3.2 Reproduction of the Genus: Interpretable ML

Encouraged by the success of the above experiments, it is natural to ask whether we can recover the bounds/conditions for genus using the machine-learning models we built. The analogous situation in computing cohomology groups for bundles over algebraic surfaces was considered in [50], where polynomial boundaries in the space of bundle degrees were detected with NNs. Likewise, hypersurfaces which separate the space of simple finite groups from the non-simple were detected in [51]. Can our complicated hole-boundaries of amoebae be detected by ML?

3.2.1 From Projections

In §3.1, we presented many plots where the input data is projected to some lower dimensional representations under certain manifold embeddings. For instance, let us take Figure 3.1(b) for F_0 where MDS manifold projection was applied. Each point (x, y) in the plane represents one input vector as $(c_1, c_2, c_3, c_4, c_5) \mapsto (x, y)$. By checking the coordinates of (x, y) , it is possible to recover the conditions in (3.3), i.e.,

$$g = \begin{cases} 0, & |c_5| \leq a \\ 1, & |c_5| > a \end{cases} \text{ with } a := |c_1 c_3|^{1/2} + 2|c_2 c_4|^{1/2}, \text{ as follows.}$$

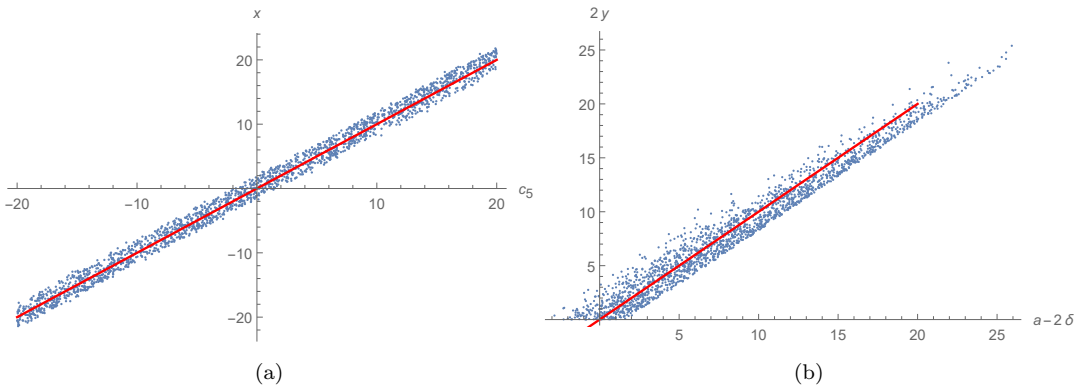


Figure 3.11: For the projection $(c_1, c_2, c_3, c_4, c_5) \mapsto (x, y)$ for F_0 , we plot (a) x versus c_5 and (b) $2|y|$ versus $a - 2\delta$.

We plot x against c_5 in Fig. 3.11(a) and find that the average of $(c_5 - x)$ is -0.172 ± 1.054 . We also draw the line $x = c_5$ in red to show a good fit. Indeed, we see that x

is essentially recovering the left hand side of the above inequalities. For y , it is more complicated and we find that there is a nice fit with $|c_{1,2,3,4}|$:

$$|y_{\text{fit}}| = 0.141|c_1c_3| + 0.166|c_2c_4| + 2.411. \quad (3.15)$$

In fact, this fit actually contains a typical linear approximation of square roots, $\sqrt{m} \approx (0.1k+1.2) \times 10^n$ for any real $k \in [1, 100)$ and $n \in \mathbb{Z}$ such that $m = k \times 10^{2n}$. Here, $c_{1,2,3,4}$ are random reals generated in the range $[-5, 5]$ and the right hand side is approximately $(0.1|c_1c_3| + 1.2 + 0.1|c_2c_4| + 1.2)$. This agrees with the linear approximation for square roots when $m = k$ and $n = 0$. In other words,

$$|y| \approx \sqrt{|c_1c_3|} + \sqrt{|c_2c_4|} = a/2. \quad (3.16)$$

We show this in Fig. 3.11(b). The blue points are the pairs $(a - 2\delta, 2|y|)$ and the line $2|y| = a - 2\delta$ is in red, where we have defined $\delta := |y_{\text{fit}}| - |y|$. Comparing the fitted results of (x, y) and Figure 3.1, we find that this indeed gives the bound $|c_5| = a$ for F_0 .

Let us now consider a more complicated example, that is, $K^{4,5,3,2}$ discussed in §3.1.4. Previously, we projected the input vectors to 3d in order to visualize the data. Here, we use PCA to project the 11-component vectors to 4-component ones and see how well this realizes (3.13). Recall that our input is $(c_0, c_1, \dots, c_{10})$. The projection then maps it to the vector (x_1, x_2, x_3, x_4) . The simple fit for each x_i is shown in Figure 3.12. In each plot, the horizontal axis is k_1c_j and the vertical axis is $k_2x_i + k_3$ for some constants $k_{1,2,3}$. We also plot the line $Y = |X|$ in red. It is thus not hard to see that we have the fit $k_2x_i + k_3 = |k_1c_j|$; which agrees with the four components/variables $|c_{4,5,7,8}|$ in (3.13). Since the parameters a_i, b_i, d_i, e_i in (3.13) are complicated expressions of $|c_i|$ which are approximately viewed as “constants”, the V-shape distributions of the points $(k_1c_j, k_2x_i + k_3)$ are thickened strips of the red line.

3.2.2 Learning from Weights

From PCA let us return to NNs. Given the good performance of latter, can we extract information from the NN (hyper-)parameters? In this subsection, we show that the bounds/conditions for the genus can be very well approximated by the weight matrices and biases in the NN structure. Indeed, an MLP with n hidden layers with activation function $f_{i+1}(W_i\mathbf{x}_i + \mathbf{b}_i)$ at the $(i + 1)^{\text{th}}$ layer, where \mathbf{x}_i is the output from the i^{th} layer and W_i, \mathbf{b}_i are the weight matrix and bias respectively. Then the composition of f_i 's would give an approximate expression (in the spirit of the universal approximation theorems of NNs) in terms of the coefficients c_j with low calculation cost to compute g .

As an example, let us again illustrate with F_0 . With input of the form $\{c_1, c_2, c_3, c_4, c_5\}$, it turns out that even an MLP with two hidden layers and three neurons at each layer

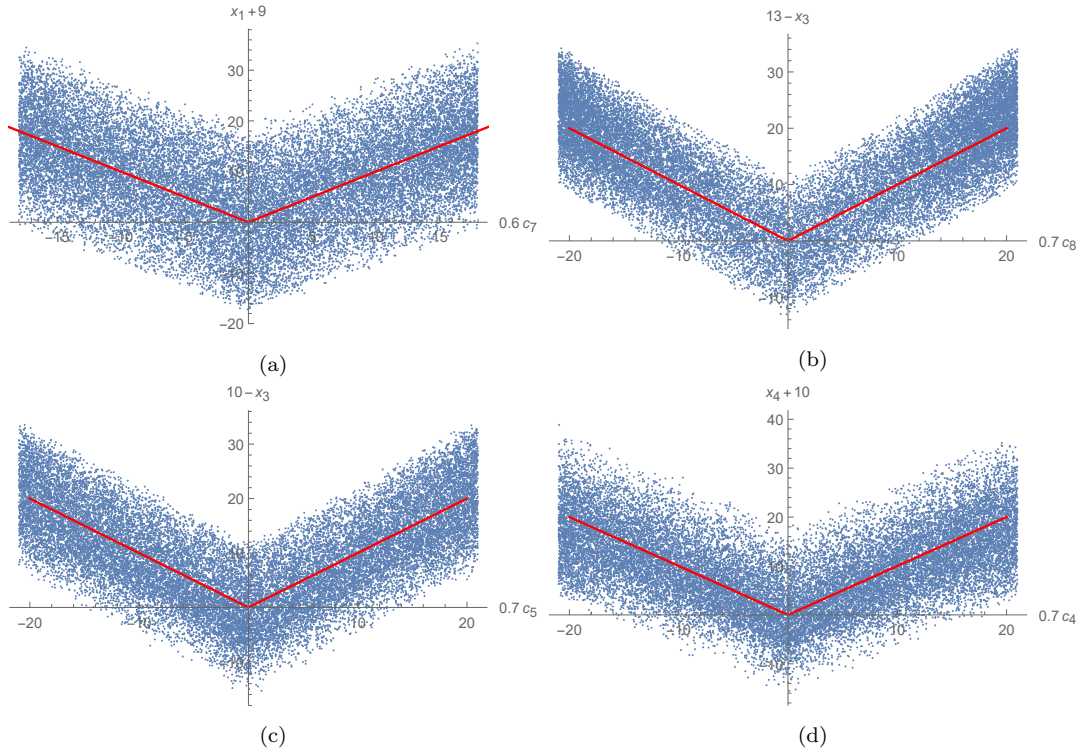


Figure 3.12: The fits of x_i 's in terms of c_j 's.

would give over 95% accuracy. It turns out that $f_{\text{output}}(W_2 \mathbf{x}_2 + \mathbf{b}_2)$ would always give a non-negative number. When this number is zero, g is classified to be 0, and g is 1 for the number being positive, as we will now see.

Now that we have low dimensional projection of the data, we can use this projected coordinates (x, y) as input and analyze the structure of MLP. This allows us to further simplify the MLP such that only one hidden layer with four neurons can give over 95% accuracy. The precise entries of weight matrices and bias vectors may vary every time we train a new model, but these values are quite stable and have very small changes. For instance, we have

$$W_1 = \begin{pmatrix} 0.4667 & 1.4986 & -0.3313 & -1.3031 \\ -1.3543 & -1.0408 & -0.5699 & -1.5610 \end{pmatrix}^T \quad (3.17)$$

and

$$W_2 = (-1.1199 \ 0.6904 \ 0.6216 \ 0.2827) \quad (3.18)$$

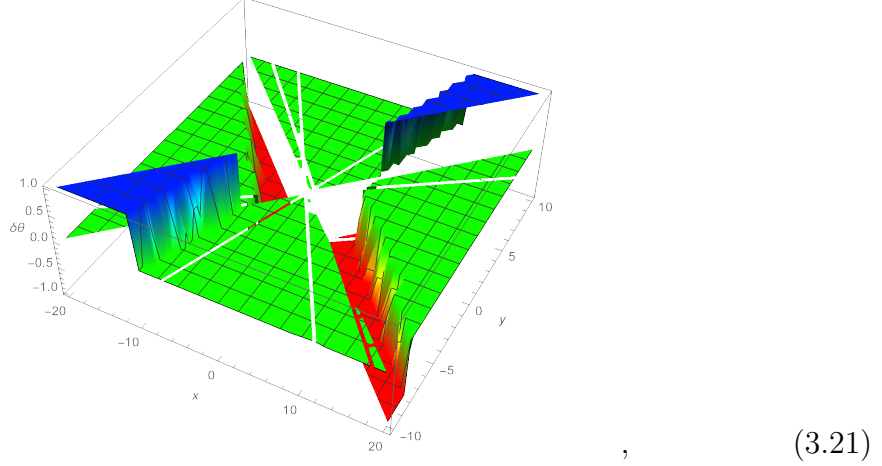
for the weight matrices. The bias vectors are

$$\mathbf{b}_1 = (-0.0349, 0.0439, 0.3044, -0.5884)^T, \quad \mathbf{b}_2 = (-4.1003). \quad (3.19)$$

Again, for some input (x, y) , this would give a non-negative number

$$p = W_2 \cdot \text{ReLU}(W_1 \cdot (x, y)^T + \mathbf{b}_1) + \mathbf{b}_2 \quad (3.20)$$

with $g = 0$ only when $p \leq 0$. In other words, $g = \theta(p)$ where θ is the Heaviside function. This can actually be visualized as



where the two horizontal axes are (x, y) and the vertical axis is $\delta\theta := \theta(p) - \theta(|x| - 2|y|)$. Notice that $\theta(|x| - 2|y|) = |x| - 2|y| = g$ when c_i 's are perfectly projected to $|c_5|$ and $(\sqrt{|c_1 c_3|} + \sqrt{|c_2 c_4|})$. As we can see, this gives a precise prediction for g in the green area (assuming that x and y are perfect projections). There are only four slices in blue or red which are not well approximated.

Incidentally, a layer with four neurons is possible to always give $\delta\theta = 0$ in principle. This can be realized, for example, when

$$W'_1 = \begin{pmatrix} 0 & 1 & 0 & -1 \\ -1 & 0 & 1 & 0 \end{pmatrix}^T, \quad W'_2 = (-2 \ 1 \ -2 \ 1) \quad (3.22)$$

and $\mathbf{b}'_{1,2}$ vanish since $|q| = \text{ReLU}(q, 0) + \text{ReLU}(-q, 0)$ for any real q . Nevertheless, $W_{1,2}$ and $\mathbf{b}_{1,2}$, together with the projection to (x, y) , can already give a nice approximated expression for g :

$$\begin{aligned} g &= \theta(p), \quad p = W_2 \cdot \text{ReLU}(W_1 \cdot (x, y)^T + \mathbf{b}_1) + \mathbf{b}_2, \\ |x| &= |c_5|, \quad |y| = 0.1|c_1 c_3| + 0.1|c_2 c_4| + 2.4. \end{aligned} \quad (3.23)$$

We can also use $(c_1, c_2, c_3, c_4, c_5)$ as input directly without any projections. This would give some approximated expression with certain W_i and \mathbf{b}_i (though it would be hard to visualize as above), and from the results in §3.1, we know that this expression approximates g with high accuracy. Likewise, for much more complicated cases, it is also possible to have well-approximated expressions for genus with low calculation cost given sufficient layers and neurons.

3.3 Lopsided Amoebae: $n > 1$

In the previous section we have shown how ML works successfully with lopsided amoebae for $n = 1$ where $\tilde{P}_1 = P$ and we directly addressed the amoeba \mathcal{A}_P . Recall that in practice, it is more efficient to consider $\mathcal{L}\tilde{P}_n$ which approximates \mathcal{A}_P for large enough n . What can ML say about the $n > 1$ cases?

For any finite n , $\mathcal{L}\mathcal{A}_{\tilde{P}_n}$ is always a superset of \mathcal{A}_P which further includes points of distance $\epsilon < (\log(n) + \log(8c))/n$ to the boundary of \mathcal{A}_P . In particular, for $n = 1$ discussed in the previous section, the genus would be different from the one for \mathcal{A}_P if the “size” of the genus is smaller than $\log(8c)$. For example, the amoeba for $P = z + w - z^{-1} - w^{-1} + 1$ is of genus one as shown in Fig. 2.1. However, $\mathcal{L}\mathcal{A}_{\tilde{P}_1}$ would have genus zero as can be seen either from $\epsilon = 4 \log 2 \approx 1.204$ and the figure, or from checking lopsidedness. Therefore, we now consider larger n for better approximations of the amoebae. Since the basic approach is similar to what we have discussed in §3.1, let us again illustrate this with F_0 as an example.

3.3.1 Example: F_0

In general, for any toric polygon Δ , the cyclic resultant \tilde{P}_n also corresponds to a Newton polytope which is $(n^2\Delta) \cap (n\mathbb{Z})^2$. For instance, we plot the first several Newton polytopes for \tilde{P}_n of F_0 in Figure 3.13. In general, $(n^2\Delta) \cap (n\mathbb{Z})^2$ would have $(2n^2 + 2n + 1)$

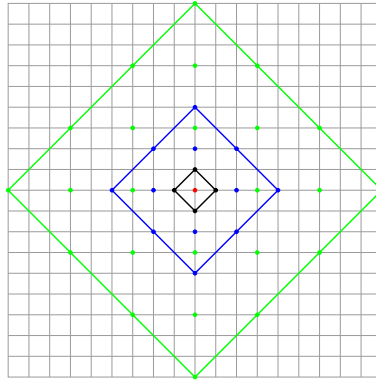


Figure 3.13: The polygons $(n^2\Delta_{F_0}) \cap (n\mathbb{Z})^2$ in the lattice \mathbb{Z}^2 . The black one is F_0 , i.e., $n = 1$. The blue and green ones correspond to $n = 2, 3$ respectively. In particular, the centre point in red, which is also the polygon with $n = 0$, belongs to $(n^2\Delta_{F_0}) \cap (n\mathbb{Z})^2$ for all n .

lattice points (or equivalently, monomials in \tilde{P}_n), among which $4n$ are boundary points.

Using \tilde{P}_n , which are listed in Appendix C.2 by explicit computation of the cyclic resultant, we can get the conditions for the genus (though they are much more complicated than the case of $n = 1$). As $n \rightarrow \infty$, we would recover the amoeba \mathcal{A}_P . To generate a data-set for \mathcal{A}_P , the following trick is used. We check whether the centre

point (which is derived in Appendix C.1) lives in the amoeba, i.e., whether there exists a solution to $P(z, w) = 0$ such that

$$(\text{Log}|z|, \text{Log}|w|) = \left(\frac{1}{2} \text{Log} \left| \frac{c_3}{c_1} \right|, \frac{1}{2} \text{Log} \left| \frac{c_4}{c_2} \right| \right). \quad (3.24)$$

Again, we use our MLP classifier with input (c_1, \dots, c_5) and output g . We also take $(|c_1|, \dots, |c_5|)$ as input to perform the same test. For each test, we generate a balanced dataset with ~ 2000 samples. The results are listed in Table 3.4.

Input	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n \rightarrow \infty$
c_i	0.957(± 0.005)	0.956(± 0.007)	0.960(± 0.008)	0.933(± 0.009)	0.915(± 0.016)
$ c_i $	0.987(± 0.007)	0.912(± 0.011)	0.905(± 0.013)	0.903(± 0.009)	0.753(± 0.056)

Table 3.4: The accuracies for MLP using 5-fold cross validation with 95% confidence interval. We use $n \rightarrow \infty$ to denote the case for \mathcal{A}_P . The result of $n = 1$ from the previous section is also listed here for reference.

As we can see, unlike $n = 1$, for other cases, the performance using c_i as input is always better than the one using $|c_i|$. In fact, this is very reasonable since only $n = 1$ has sets $P\{\mathbf{z}\}$ with elements $|c_i|$ (multiplied by certain $z^i w^j$) while for larger n the elements can be complicated expressions of c_i 's. For instance, as shown in Appendix C.2, there exists an element $|(4c_1^3 c_3 - 4c_1^2 c_2 c_4 - 2c_1^2 c_5^2) w^4 z^6|$ in $\tilde{P}_2\{\mathbf{z}\}$ for F_0 . Therefore, taking absolute values is not equivalent to restricting to positive coefficients and could mislead the machine.

We may still use the MDS manifold embedding to project the inputs c_i to 2d. This is visualized in Figure 3.14.

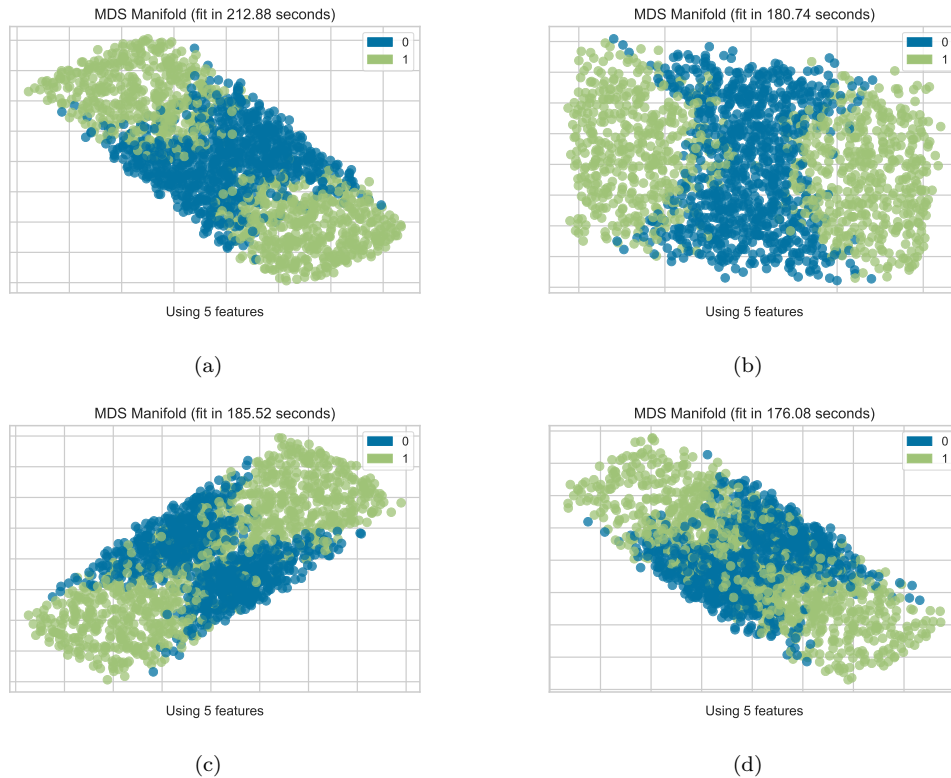


Figure 3.14: The MDS manifold projections for the datasets with (a) $n = 2$, (b) $n = 3$, (c) $n = 4$ and (d) $n \rightarrow \infty$.

Although it would be hard to tell what the two components are as the conditions from lopsidedness are much more complicated, it is interesting to see that these distributions of blue and green points have similar shapes/regions compared to $n = 1$ ¹⁴.

Now that taking absolute values is completely different from taking non-negative coefficients, we generate a dataset for non-negative coefficients with ~ 2000 samples for each n and perform the same tests as above. The result is in Table 3.5.

Input	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n \rightarrow \infty$
$c_i \in \mathbb{R}^+$	0.987(± 0.007)	0.983(± 0.008)	0.985(± 0.002)	0.976(± 0.006)	0.930(± 0.008)

Table 3.5: The accuracies for MLP using 5-fold cross validation with 95% confidence interval.

¹⁴One may also project the inputs $|c_i|$ to 2d as before. It turns out that the shapes/regions of distributions are also similar to case with $n = 1$ but with more mixings at the boundary of two regions. However, we are not really interested in this as taking absolute values is not helpful at all here.

Compared to the results for real coefficients in Table 3.4, we see that the performance all get improved. In particular, the cases for finite n listed here all give almost perfect results while the accuracy for $n \rightarrow \infty$ is also slightly improved. As plotted in Figure 3.15, the MDS manifold projections for positive coefficients also have similar distributions compared to the one for positive coefficients with $n = 1$.

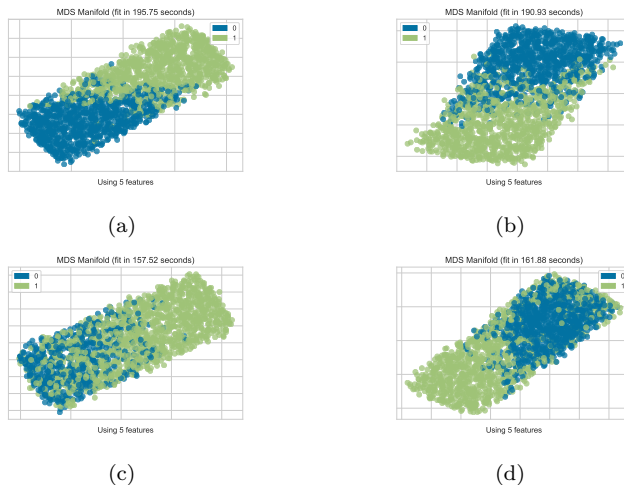


Figure 3.15: The MDS manifold projections for the datasets with (a) $n = 2$, (b) $n = 3$, (c) $n = 4$ and (d) $n \rightarrow \infty$. Here the coefficients c_i are restricted to be positive.

3.3.2 Reproduction of the Genus

We may also use the weight matrices and bias vectors of the MLP to get nice approximated expressions of g for different n 's as what we previously did for $n = 1$. Here, for F_0 , we present the weight matrices and bias vectors for the amoeba \mathcal{A}_P (viz, $n \rightarrow \infty$) as an example.

It turns out an MLP of one hidden layer with 20 perceptrons could already give high accuracy as in Table 3.4. We still use ReLU as activation functions. The weight matrices are

$$W_1 = \begin{pmatrix} 0.597 & 0.911 & -0.981 & -0.458 & -0.359 & -0.143 & -1.183 & -0.559 & -0.864 & -0.378 & -1.076 & -0.046 & 0.624 & -0.804 & -0.656 & -0.067 & 0.274 & -0.516 & -1.021 \\ -1.031 & 0.910 & 0.444 & 0.887 & 0.898 & 1.117 & -0.530 & 0.182 & -0.355 & 1.347 & -0.435 & 0.769 & 1.379 & 1.558 & -0.384 & -0.779 & 1.054 & 0.747 & -0.346 \\ 0.717 & 1.528 & -2.440 & -0.208 & -1.299 & 0.285 & 0.417 & -1.318 & -0.644 & -0.794 & -1.005 & 1.408 & 0.228 & 0.870 & 1.738 & -0.166 & 0.585 & -0.496 & -1.869 \\ -0.596 & 0.757 & 0.284 & 0.814 & -0.940 & -0.802 & -1.363 & -0.550 & 0.375 & 1.116 & 0.730 & 1.308 & -1.127 & 0.834 & -0.373 & -2.297 & 1.203 & 0.780 & 0.630 \\ -0.103 & -0.556 & -0.035 & -0.811 & -0.503 & 0.475 & -0.475 & 1.062 & 0.360 & 0.156 & -0.550 & 0.552 & 0.165 & 0.643 & 0.207 & -0.096 & 1.380 & 0.455 & -1.287 \\ & & & & & & & & & & & & & & & & & & & 0.817 \end{pmatrix}^T \quad (3.25)$$

and

$$W_2 = (0.144 \quad -0.958 \quad -1.435 \quad 1.160 \quad 0.629 \quad -1.311 \quad 0.183 \quad 1.193 \quad 0.701 \quad -1.597 \quad 0.331 \quad 1.064 \quad 1.394 \quad -1.296 \quad 1.108 \quad -1.078 \quad 1.262 \quad 0.891 \quad 0.511 \quad -1.776). \quad (3.26)$$

The bias vectors are

$$\mathbf{b}_1 = (-0.044, 0.320, 0.234, 0.421, 0.221, 0.057, 0.088, -0.317, 0.387, -0.054, -0.154, -0.296, 0.426, 0.051, 0.368, 0.162, 0.006, -0.004, -0.394, -0.072)^T \quad (3.27)$$

and

$$\mathbf{b}_2 = (-0.943). \quad (3.28)$$

This gives a non-negative number p with $g = 0$ only when $p = 0$:

$$p = W_2 \cdot \text{ReLU}(W_1 \cdot (c_1, \dots, c_5)^T + \mathbf{b}_1) + \mathbf{b}_2 \Rightarrow g = \theta(p). \quad (3.29)$$

Recall that computing the genus for F_0 is, in fact, finding a solution to

$$c_1 z + c_2 w + c_3 z^{-1} + c_4 w^{-1} + c_5 = 0 \quad (3.30)$$

with conditions

$$|z| = \left| \frac{c_3}{c_1} \right|^{1/2}, \quad |w| = \left| \frac{c_4}{c_2} \right|^{1/2}. \quad (3.31)$$

If there is no such solution, then $g = 1$. Otherwise, $g = 0$. Now, we see that this is translated to (3.29)¹⁵.

¹⁵More strictly, this approximate expression is valid in the ranges of the coefficients in our data-set, which are $c_{1,2,3,4} \in [-5, 5]$ and $c_5 \in [-20, 20]$. Of course, if we have larger ranges for the coefficients, it is still possible to obtain certain approximations following the same recipe.

3.4 The Membership Problem

So far, we have focused on detecting edges of holes and determining the genus of the amoeba. We can also use machine learning to study another important problem, that of membership. We introduced this in §2: given the Newton polynomial (always bivariate in this paper), how does one decide whether a point in the plane belongs to the amoeba? Thus, our input is of form $\{c_1, \dots, c_m, x_1, x_2\}$, where c_i are the coefficients of the Newton polynomial as before and (x_1, x_2) is a point on the Log plane. If (x_1, x_2) belongs to the amoeba, then the output is 1. Otherwise, the output is 0. This is a binary classification problem.

Again, we illustrate with our running example of F_0 . Here, the input vectors are of the form $\{c_1, \dots, c_5, x_1, x_2\}$. For a balanced dataset with ~ 5000 samples, we find that our MLP with one hidden layer of 8 perceptrons already gives 0.944(± 0.003) accuracy for 5-fold cross validation with 95% confidence interval¹⁶. Therefore, the weight matrices and bias vectors could provide good approximated expressions to determine the regions for amoebae.

The weight matrices are

$$W_1 = \begin{pmatrix} 0.246 & -0.401 & 0.287 & -0.105 & 0.006 & 0.553 & 0.700 & -0.006 \\ -0.123 & -0.039 & 0.088 & -0.139 & 0.002 & 0.147 & -0.031 & 0.008 \\ 0.198 & -0.104 & 0.040 & 0.058 & 0.000 & 0.066 & 0.208 & 0.009 \\ 0.134 & -0.216 & 0.157 & 0.158 & 0.000 & 0.024 & 0.292 & 0.018 \\ -0.067 & 0.106 & -0.058 & -0.014 & -0.002 & -0.117 & -0.199 & -0.012 \\ -0.011 & -2.796 & -3.378 & -2.026 & -3.72 & -1.365 & 3.522 & -5.367 \\ -2.396 & 0.254 & -2.870 & 3.128 & 3.763 & 3.297 & 1.409 & -5.014 \end{pmatrix}^T \quad (3.32)$$

and

$$W_2 = (1.273 \ 3.525 \ 2.168 \ 2.275 \ -4.418 \ 1.993 \ -1.313 \ -6.769). \quad (3.33)$$

The bias vectors are

$$\mathbf{b}_1 = (0.337, -3.304, 3.064, -0.646, -2.778, 0.453, 2.778, -1.000)^T; \mathbf{b}_2 = -1.526. \quad (3.34)$$

This gives a non-negative number

$$p = W_2 \cdot \text{ReLU}(W_1 \cdot (c_1, \dots, c_5, x_1, x_2)^T + \mathbf{b}_1) + \mathbf{b}_2. \quad (3.35)$$

¹⁶This is probably be the simplest MLP to give the best performance. We also checked MLPs with fewer perceptrons in the hidden layer, but all of them were inferior. On the other hand, if we include more neurons, the performance hardly improved (sometimes even worsened).

Then (x_1, x_2) is in the complement of amoeba only when $p \leq 0$. In other words,

$$(x_1, x_2) \in \mathcal{A}_P \text{ if } \theta(p) = 1. \quad (3.36)$$

As we can see, the membership problem is now transformed to (3.35) and (3.36) with lower calculation cost¹⁷. Of course, it is natural to expect that we can get better approximations with more training data and a more complicated neural network. Such method can also be applied to c_i and (x_1, x_2) with larger ranges and amoebae for other Newton polytopes.

3.4.1 Crawling of Amoebae

Let us end this section by a quick comment on the boundaries of amoebae. For a given Newton polynomial, if we vary the coefficients continuously, then the shape/boundary of the amoeba should also change continuously. We shall refer to this as the “crawling” of amoebae. Again, let us use F_0 as our example. In Figure 3.16, we show how the amoeba crawls for $z + w + z^{-1} + w^{-1} + c_5 = 0$ with $c_5 \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$.

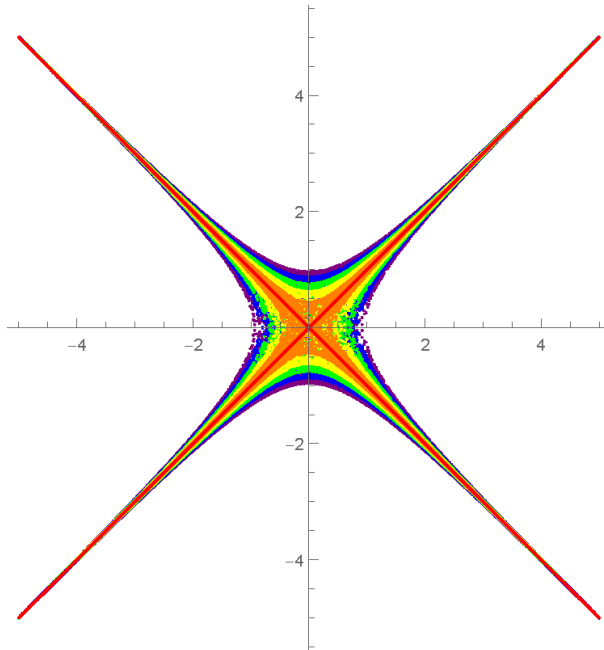


Figure 3.16: How the amoeba “crawls” for $z + w + z^{-1} + w^{-1} + c_5 = 0$, where $c_5 = 0, 0.2, 0.4, 0.6, 0.8, 1$ are in red, orange, yellow, green, blue and purple respectively.

We may also plot this crawling in 3d with c_5 as the third axis. We vary c_5 from 0 to 5 with step 0.2 in Figure 3.17.

¹⁷To be strict, such approximated expression should be valid in the ranges of the coefficients in our dataset, which are $c_i, x_i \in [-5, 5]$.

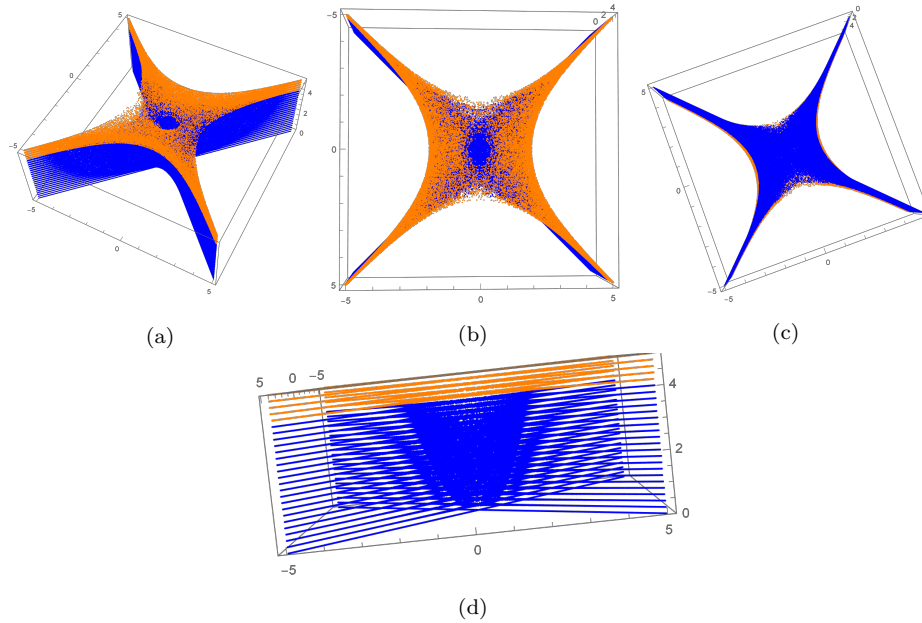


Figure 3.17: (a) The 3d plot of amoeba varying c_5 . (b) The same plot viewed from the top. (c) The same plot viewed from the bottom. (d) The same plot viewed from the side. The two colours are chosen based on whether the corresponding lopsided amoeba $\mathcal{L}\mathcal{A}_P$ has genus 0 or 1. Hence, some blue slices near the orange part should also have genus 1.

Likewise, in Figure 3.18, we vary c_1 while keeping the other coefficients fixed. The Newton polynomial now reads $c_1z + w + z^{-1} + w^{-1} + 4 = 0$, where c_1 is from -2 to 2 with step 0.2 .

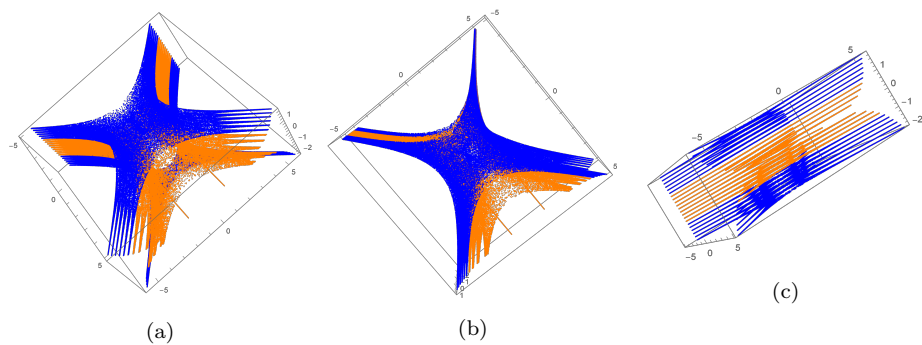


Figure 3.18: (a) The 3d plot of amoeba varying c_1 . (b) The same plot viewed from the top. Viewing from the bottom would be the same due to the symmetry of the polynomial. (c) The same plot viewed from the side. The two colours are chosen based on whether the corresponding lopsided amoeba $\mathcal{L}\mathcal{A}_P$ has genus 0 or 1. Hence, some blue slices near the orange part should also have genus 1. Notice that when $c_1 = 0$ there is no genus due to degeneracy.

Here, we only vary one coefficient and restrict the coefficients to be real so as

to visualize a specific crawling of the amoebae. In general, one can vary multiple coefficients and consider the coefficients to be any complex numbers. If we know how an amoeba crawls when varying any combination of the coefficients (though not easy), we can determine the boundaries of all amoebae for a given Newton polytope/polynomial once we know the boundary information of one amoeba.

In fact, from the above discussion, we may further conjecture that when varying the coefficients, the (change of) boundary of an amoeba (i.e., the crawling of it) forms some (almost) manifold. In other words, it is smooth except when there is degeneracy. By degeneracy, we mean that some of the coefficients vanish in $P(z, w)$. For instance, the amoeba degenerates to $\text{Log}|w| = \pm \text{Log}|z|$ when $c_5 = 0$ in Figure 3.17 while the amoeba degenerates to the amoeba of $\mathbb{C}^3/\mathbb{Z}_2(1, 1, 1)$ when $c_1 = 0$ in Figure 3.18.

Overall, since the machine learning models here are aimed to show that a minimal number of data points and a minimal structure with very simple weights and bias could already perform well for the membership problem, and since the input is composed of varying coefficients and (x_1, x_2) rather than (x_1, x_2) only, the approximations of the boundaries is still not comparable to the approximations by lopsidedness or plotting directly using Monte Carlo method. However, it is reasonable to expect a more complicated neural network with more training data would refine such approximations, and we hope that the tests and observations here could shed light on the study of amoebae and their complementary regions.

4 Image Processing Amoebae

In addition to machine learning the abstract space of amoebae coefficients, it is interesting to test the success of ML on the amoebae images directly. For this investigation F_0 is used again as our prototypical example, with the database consisting of Monte Carlo generated images of $n = 1$ lopsided amoebae.

CNNs find their most common uses in image related tasks, due to their convolutional action over the local structure of the images. Since this is the behaviour used in collecting Monte Carlo generated points into a full amoeba, this ML architecture is the most natural choice. To fully investigate their success in identifying genus from these images, varying image resolutions are used for the CNN input. Determining how the learning measures varied as the amoebae images varied in quality is the main focus of this investigation.

4.1 Amoeba Image Dataset

Here, our database consists of Monte Carlo generated **images** of the F_0 amoebae for varying coefficients such that the full database consisted of 1000 genus 0 amoebae, and

1000 genus 1 amoebae. The amoebae genus labelling used the $n = 1$ lopsided amoeba approximation, as in equation 3.3. Monte Carlo image generation is commonly used for more complicated amoebae where analytical expressions are hard to compute and then plot. Therefore examining the success of a convolutional neural network (CNN) in identifying genus from these plots is a particularly relevant task. We now undertake this task of the CNN ML of the labelled data-set

$$\text{Image}(\mathcal{A}_{P(F_0)}) \longrightarrow \text{genus } \{0 \text{ or } 1\}.$$

The amoebae images in the dataset are of varying shape, to create consistency across the dataset (as needed for the CNN tensor inputs), the images are resized such that there is always an equal number of pixels in each dimension. The images contained the real plane axes since the identification of a plot’s origin is useful in determining genus for F_0 amoebae by eye, hence giving the CNN realistic information for its learning. Images are reformatted to greyscale such that inclusion in the amoeba is the only relevant data at each resolution. The resolutions learnt on varied logarithmically with base 2, choosing this base such that the computational learning algorithms worked most efficiently.

An example of a genus 1 amoeba from the dataset is given in Figure 4.1, this amoebae had randomly generated \mathbb{R}^+ coefficients:

$$(c_1, \dots, c_5) \sim \{0.6104, 1.8940, 0.4989, 2.9777, 6.9871\}. \quad (4.1)$$

In the figure the image at the varying resolutions considered is shown, the reshaping to square from the original is clear, and the binary nature of the data is emphasised by the greyscaling (here plotted with a blue-yellow colour scheme).

CNN Architecture: The `Tensorflow` library `Keras` is used for implementing the CNN architecture [52]. The CNNs trained have a consistent layer structure: $3 \times 2d$ convolutional layers of size matching the image input size (each followed by a 2d max-pooling layer and a dropout layer (factor 0.01)); following these is a flattening layer, then a dense layer of size equal to the number of pixels in one dimension of the input image, and then a final output dense layer with 1 neuron. All layers used a LeakyReLU activation (with factor 0.01), except the final output Dense layer which used sigmoid activation to better map to the required binary classification. The convolutional and max-pooling layers used the ‘same’ padding regime, and the convolutional layers used a 3×3 kernel size.

The CNN architecture used for the investigation at each image resolution is trained over 20 epochs of the data on batches of 32, in a 5-fold cross-validation scheme. The Adam optimiser [45] is used to minimise the binary cross-entropy loss function for

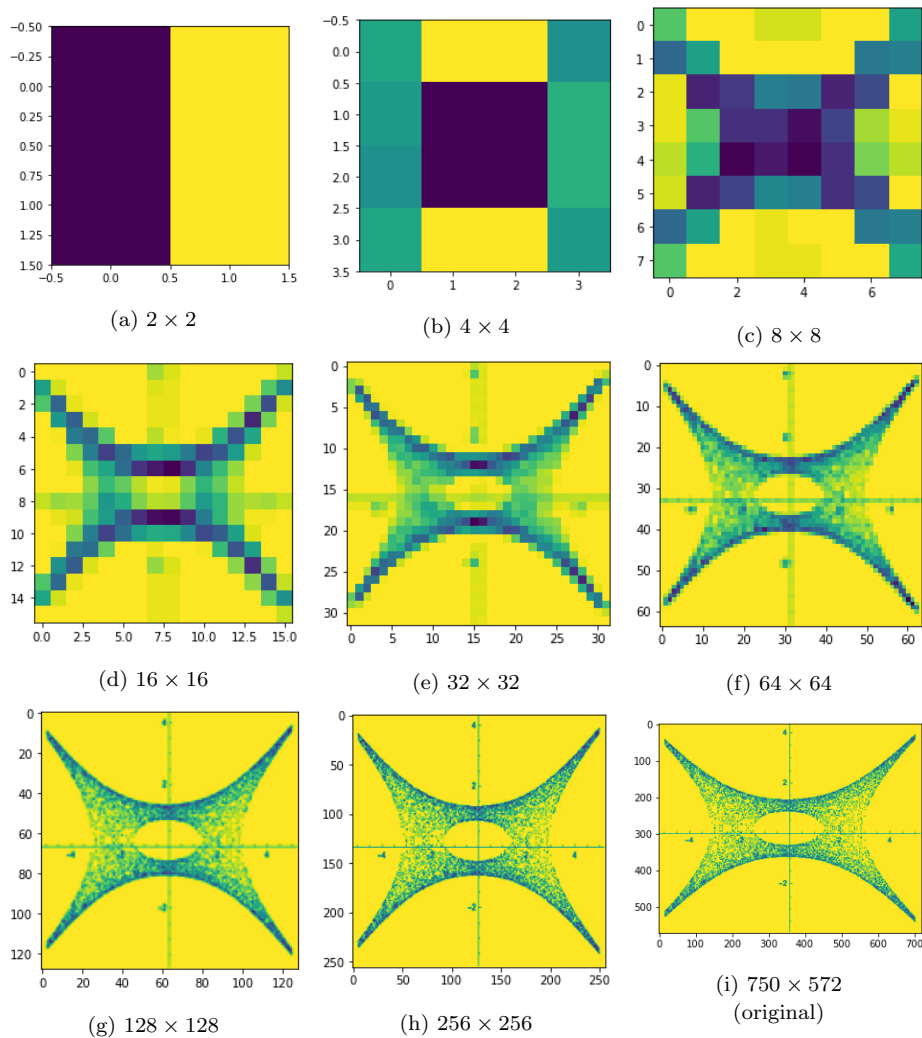


Figure 4.1: An example F_0 amoeba image of genus 1 at varying resolutions. Each subcaption denotes the respective number of pixels in the resizing, the original image is reformatted to a square shape for consistency across all amoebae.

predicting genus 0 or genus 1 respectively. The learning is measured using the metrics: accuracy and Matthews' Correlation Coefficient (MCC)¹⁸, which are averaged over the 5 runs in the 5-fold cross-validation.

4.2 Image Processing Results

Results for the CNN ML over varying image resolutions are given in Table 4.1. The accuracies increase as image resolution improves up to some *optimal value*, around

¹⁸The MCC is equal to the chi-square of the 2×2 confusion matrix in the classification.

32×32 pixels where it then falls off; this behaviour is further shown in Figure 4.2. The high accuracies at optimal resolution (> 0.98) show the success of CNNs in identifying genus from Monte Carlo generated amoebae images.

The change in accuracy as image resolution decreases from the largest size considered may be explained by initial resolution loss causing averaging over the Monte Carlo generated points to produce a connected amoeba component with a clearer hole structure in genus 1 amoebae (Figure 4.1e). Then further decrease in resolution loses more of the image information combining any holes into the amoebae themselves until there is no sensible information in the image remaining (Figure 4.1a). Additionally any poorly sampled parts of the amoeba will be averaged over the resolution decrease to be effectively removed, perhaps causing the amoebae to appear disconnected.

Similar behaviour can be seen over the MCC values where calculable. Although MCC as a measure is less susceptible to bias in the data making it generally preferential, there are issues of incalculability where learning fails and the same class is predicted for all test data. Where this happened for some of the 5 runs in the 5-fold cross-validation, the MCC average and standard error is calculated over the calculable values, and are denoted with a ‘*’ (note this inflates the measures’ value as the cases of no learning are ignored). Where the MCC is incalculable for all runs an averaged MCC is then incalculable also, these scenarios are denoted with ‘nan’ in the results in Table 4.1.

Learning Measures	Image Resolution							
	2×2	4×4	8×8	16×16	32×32	64×64	128×128	256×256
Accuracy	0.484	0.485	0.746	0.972	0.987	0.986	0.803	0.754
	0.005	0.006	0.064	0.008	0.005	0.004	0.116	0.055
MCC	nan	nan	0.620*	0.944	0.974	0.971	0.639	0.646*
	nan	nan	0.012*	0.017	0.010	0.008	0.212	0.032*

Table 4.1: Learning results for CNN binary classification of F_0 $n = 1$ lopsided amoebae images, determining genus 0 or 1 over balanced dataset of 2000 images at varying image resolutions. Resolution is given in terms of the number of pixels. Measures are averaged accuracies and MCCs over the 5-fold cross-validations run, with standard errors. MCCs calculated over less than 5 of the cross-validations are denoted with a ‘*’, for those completely incalculable ‘nan’ is given.

4.3 Amoeba Image Misclassifications

Examining the image resolution 32×32 which leads to the most successful CNN classification, the misclassifications where the CNN disagreed with the images labelled genus provide interesting further insight into the learning. In one of the 5-fold cross-validation runs 3 of the 400 images in the testing dataset are misclassified such that

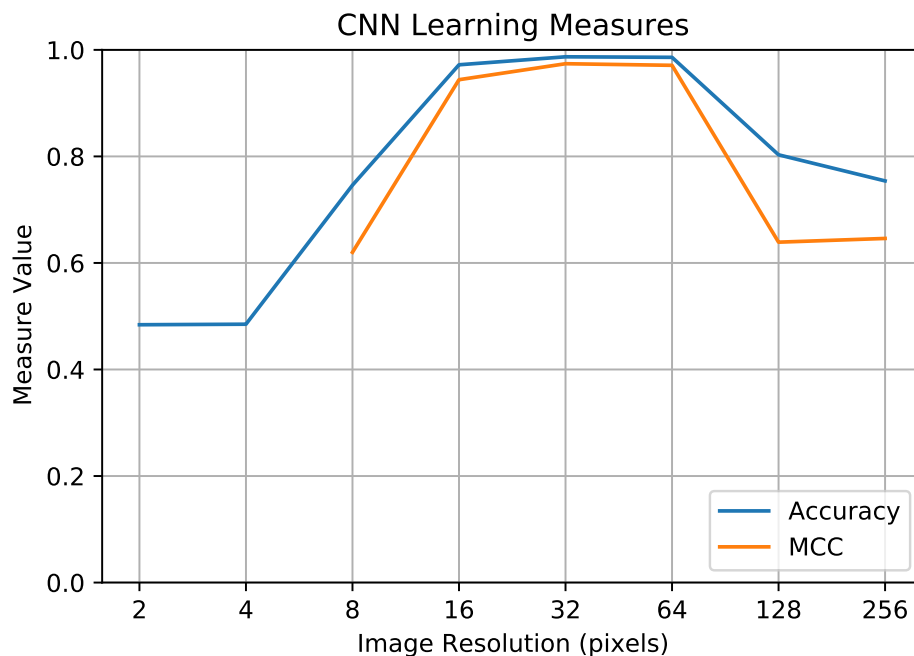


Figure 4.2: Averaged accuracy and MCC learning measures for the CNNs trained over F_0 amoebae images at varying resolutions (denoted by number of pixels in one dimension of the square images). MCC values were incalculable at certain resolutions, and may be artificially high at resolutions $\{8, 256\}$.

the predicted amoeba genus differed from the labelled genus. These images are shown at full resolution as well as the 32×32 resolution in Figure 4.3.

The first image misclassified, shown in full and lowered resolutions in Figures 4.3a & 4.3d respectively, is labelled as genus 0, but misclassified to genus 1. The original image shows the true genus 0 amoeba, however the Monte Carlo sampling of amoeba points shows a particularly low density in the amoeba’s centre, this leads to a poorer collation of points as the resolution drops, shown by fainter parts of the amoebas centre in the lower resolution image perhaps misleading the CNN to predict a non-zero genus.

The second misclassified image, shown in full and lowered resolutions in Figures 4.3b & 4.3e respectively, is labelled as genus 1, but misclassified to genus 0. In the original image in particular, the issues with the Monte Carlo image generation become clear, the amoeba looks disconnected into two parts, where the hole in the centre dominates the amoeba. This leads to part of the hole’s boundary being lost in the lower resolution image, making the amoeba appear to be genus 0, misleading the CNN.

The third and final misclassified image in this run, shown in full and lowered resolutions in Figures 4.3c & 4.3f respectively, is also labelled as genus 1, but misclassified

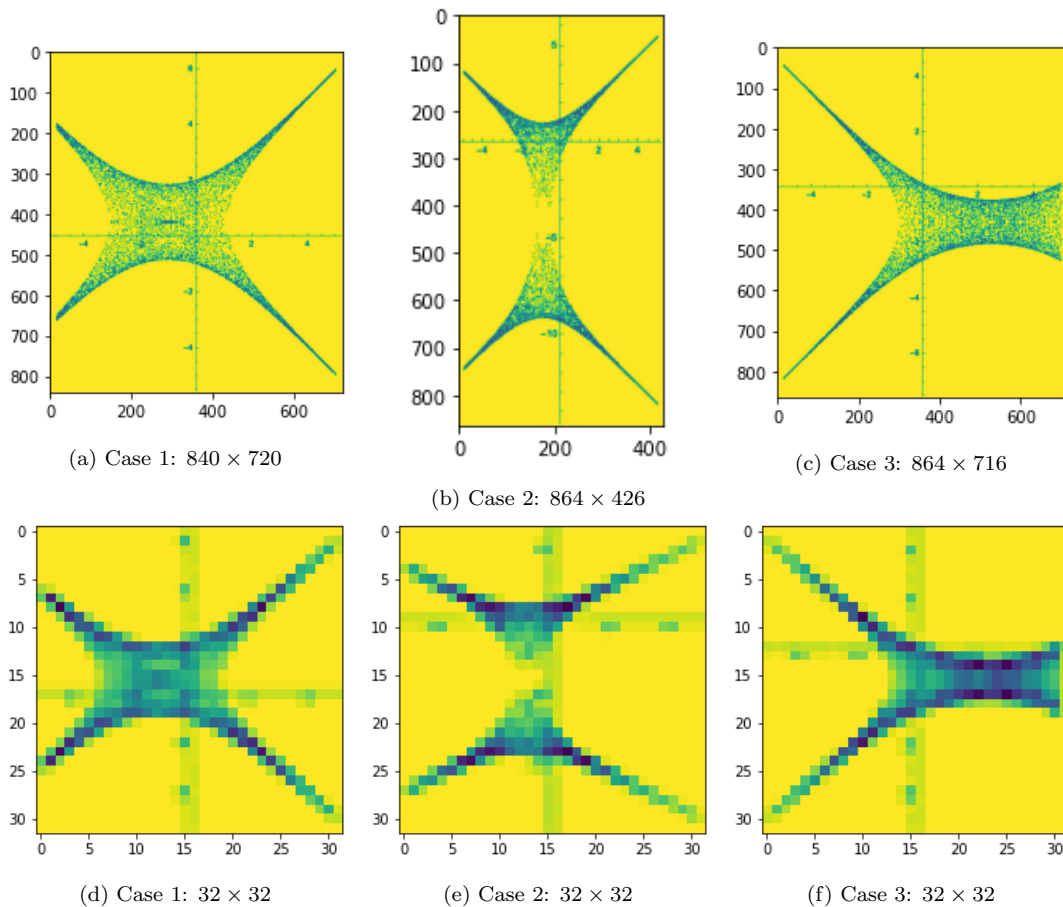


Figure 4.3: Example CNN misclassifications during model testing, images show the original image and the image at the 32×32 resolution considered. Case 1, (images a & d) was labelled as genus 0 with the CNN predicting genus 1; cases 2 & 3 (images b,e & c,f) were labelled genus 1 with CNN predictions of genus 0.

to genus 0. Here the original image shows a genus 0 amoeba, interestingly this scenario is a rare occurrence where the $n = 1$ lopsided amoeba (used to produce the genus labels for the dataset as in 3.3) has a different genus to the true amoeba. Therefore although this is considered by the CNN as a misclassification, the CNN has actually managed to predict the true amoeba genus from the image, whilst at the same time highlight a case where the match-up to lopsided amoebae fails.

The three misclassification examples from this run demonstrate the subtleties of the Monte Carlo image generation for amoeba, as well as the issues with relying on lopsided amoeba approximation for genus prediction.

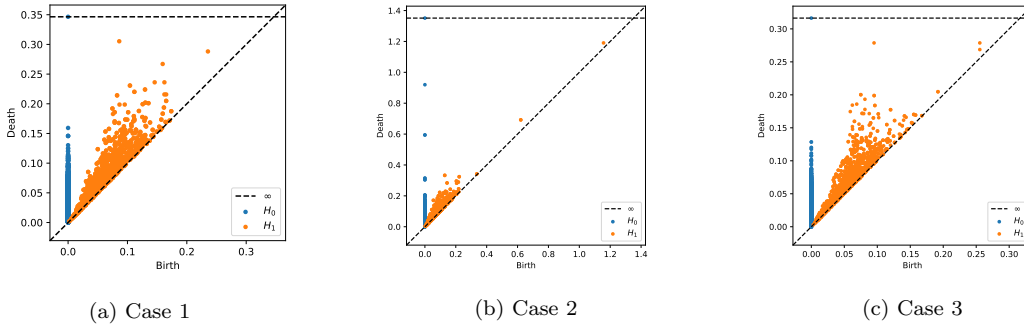


Figure 4.4: Persistence diagrams for the 3 misclassification cases considered, showing the H_0 and H_1 persistent homology of the Monte Carlo sampled amoebae points.

Persistent Homology Topological data analysis provides an alternative technique for describing the homotopy of data manifolds. Within this field, identifying genus of surfaces is often well performed by the technique of persistent homology.

In the persistent homology computation of this image classification problem, the 2-dimensions Monte Carlo generated amoeba points would have 2-dimensional discs of radius r drawn about each of them. As the radius is varied through the range: $0 \mapsto \infty$ the discs begin to intersect, at each new radius where there is a new intersection of n discs the simplicial complex formed from the 0-simplex points has an n -simplex drawn between those points (up to the maximum allowed simplex dimensionality: 2). The discrete updating of the complex as r increases creates a chain complex within which the persistent homology can then be analysed.

To identify the genus, one must analyse the occurrence of 2-dimensional holes; the H_1 homology should then have persistent features for each hole contributing to the genus count. For these examples where genus is 1 there should hence be a single persistent feature which is far from the persistence diagram diagonal — corresponding to a hole whose boundary is connected quickly and fills in much later. Conversely for 0 genus there should be no such feature.

Additionally, the H_0 homology keeps track of the number of connected components in the complex, for well sampled amoeba points all the features should hence ‘die’ quickly as all the amoeba points become connected to nearby points and form the amoeba as a single component.

For computation of the persistence diagrams of the 3 misclassification cases considered here the python library `ripser` was used [53]. The persistence diagrams for each of these cases are plotted in Figure 4.4.

The H_0 homology of cases 1 and 3 show the amoeba is connected quickly, with no gaps in the H_0 feature line. However for case 2 there are gaps in the line, indicating

the sampling is poor and the amoeba appears as disconnected components, agreeing with previous analysis.

For the H_1 homology all cases have no significant features far from the diagonal, indicative of a 0 genus amoebae. Any significant features should be a distance from the diagonal comparable to the scale of the amoeba. Since the amoeba centres have size ~ 2 and all H_1 features die for $r \lesssim 0.3$, despite there being some points slightly further from the diagonal they are not persistent enough to indicate a significant hole and genus 1. For case 1 the persistent homology now correctly predicts the genus, outperforming the CNN. However for cases 2 and 3 where the genus is truly 1 the homology prediction is incorrect, likely due to the same reasons as the CNN — the poor sampling means the case 2 boundary isn't connected properly, and for case 3 the lopsided amoeba boundary is correctly identified even though it does not match the true amoeba.

Both the H_0 and H_1 data corroborate the CNN results, however the persistent homology performs better with case 1 correctly identifying genus 0.

5 Conclusions and Outlook

In this paper, we initiated the study of applying neural networks to the analysis of amoebae, especially their genus and membership, which are questions of interest to algebraic geometers and theoretical physicists alike. We found that a simple MLP or CNN can tell the genus for amoebae with a given Newton polytope with rather high accuracy, from a vector composed of the coefficients of the Newton polynomial P . In particular, our approach used the amoeba \mathcal{A}_P and the lopsided amoeba $\mathcal{L}\mathcal{A}_{\tilde{P}_n}$ and showed that ML is able to count the genus with good performance for different n 's.

Importantly, using manifold ML, We showed that one can reproduce the complicated conditions of genus in terms of lopsidedness and the coefficients. These results are in line with the interpretability of the ML, which is currently one of the biggest challenges in AI. In parallel, we saw how one can identify the genus from the amoebae directly as an image processing problem using a CNN. Here, an optimal image resolution was found which can improve the efficacy on the Monte Carlo generated images; whilst misclassifications highlight subtleties in this process.

As we trained the models with coefficients of the Newton polynomial, the characteristics of the NNs allow one to write down certain approximations to the conditions for deciding the genus. It would be interesting to compare this with the approximations from lopsidedness in future work and find more refined approximations using ML. From our extensive appendices we saw how complicated the decision boundaries of the holes are, and the interpretability of our NN allowed for this approximation.

While we mainly focused on amoebae obtained from the projection onto \mathbb{R}^2 , one may also take the projection from $(\mathbb{C}^*)^2$ to \mathbb{T}^2 and plot the (θ, ϕ) on the real plane. This is known as the *alga* or *coamoeba*, and it is possible to recover brane tilings in string theory therefrom [6]. A similar study on algae would be an intriguing direction to explore. Besides, we only studied two-dimensional polygons and their associated amoebae in the paper as they are intimately related to D3 branes probing toric CY 3-folds and the mirror geometry involving D6 branes. It is possible to generalize the analysis here to higher dimensions for lattice polytopes and amoebae. More broadly, tropical geometry has been connected to machine learning architecture in [26, 27]. The piecewise linearity as a common feature for both tropical geometry and neural networks would naturally lead to future communications between the two areas.

Acknowledgement

JB is supported by a CSC scholarship. YHH would like to thank STFC for grant ST/J00037X/1. EH would like to thank STFC for the PhD studentship. We would like to thank Thomas Fink and Forrest Sheldon for stimulating discussion.

A Transformations of Amoebae

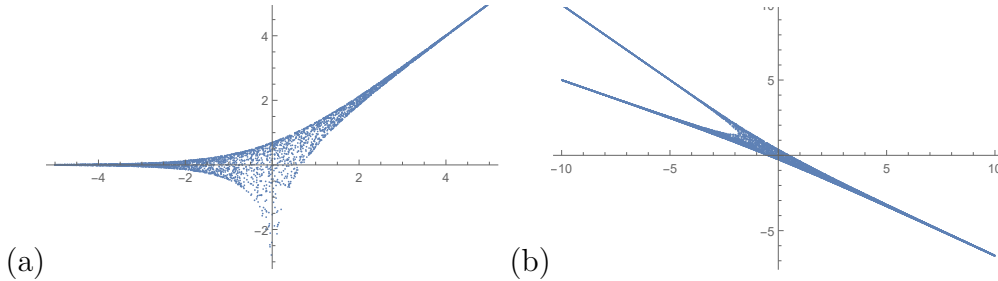


Figure A.1: The two amoebae have the same shape. (a) The amoeba given by $z + w + 1$. (b) The amoeba given by $z^2w^3 + zw^2 + 1$.

When drawing amoebae, it is often much more efficient to use Monte Carlo method than to plot them analytically. However, due to the arguments of z and w , the sample points may distribute unevenly on the Log plane. For instance, even for one of the simplest cases, \mathbb{C}^2 , the amoeba plotted from Monte Carlo method is shown in Figure A.1(a). We can see that one of the tentacles has fewer points compared to the other two.

As it is often not easy to find suitable distributions for $\arg(z)$ and $\arg(w)$, one way to resolve this is to consider certain transformations of amoebae [13]:

Theorem A.1. For $(\alpha, M) \in (\mathbb{C}^*)^2 \rtimes \text{GL}(2, \mathbb{Z})$ and $P(\mathbf{z}) \equiv P(z, w) \in \mathbb{C}[z, w, z^{-1}, w^{-1}]$, the map $\Psi : (\mathbb{C}^*)^2 \rtimes \text{GL}(2, \mathbb{Z}) \rightarrow \text{Aut}(\mathbb{C}[z, w, z^{-1}, w^{-1}])$ defined by $\Psi(\alpha, M)(P(\mathbf{z})) = P(\alpha \cdot \mathbf{z}^M)$ is an isomorphism. Moreover, their Newton polytopes satisfy $\Delta(\Psi(P)) = M \cdot \Delta(P)$. Denote the amoeba of P as \mathcal{A}_P , then for $\det(M) \neq 0$, we have $\mathcal{A}_P = M\mathcal{A}_{\Psi(P)} - \text{Log}(\alpha)$.

Let us take $M = \begin{pmatrix} 2 & 1 \\ 3 & 2 \end{pmatrix}$ and $\alpha = (1, 1)$. This sends $z+w+1$ to $z^2w^3+zw^2+1$ whose amoeba is plotted in Figure A.1(b). We can see that the points are distributed more evenly. The price is that this would slow the calculations due to the higher degrees.

B Manifold Learning

In this appendix, we give a quick introduction to different methods in manifold visualizations, including isomap, t-SNE, MDS and spectral embedding. For more details including other ways of projection, one is referred to the manual of `Yellowbrick` [46].

Isomap The isometric mapping, also known as isomap, projects the data onto a lower dimensional space while preserving the geodesic distances among all data points. It first applies nearest neighbour search and shortest-path graph search. Then the embedding is encoded in the eigenvectors of the largest eigenvalues of the isomap kernel matrix. See [54] for a detailed study on this.

MDS The multi-dimensional scaling analyzes the similarity of the data, and the data points that are close in the higher dimensional space are also near to each other in the embedding. For the (metric) MDS applied in our main context, it minimizes the cost function known as the stress defined as $\left(\sum_{i,j} (d_{ij} - \|x_i - x_j\|)^2 \right)^{1/2}$ where d_{ij} is the Euclidean distance between the data points x_i and x_j . There are also other implementations of MDS. See for example [55].

Spectral embedding The spectral embedding is a discrete approximation of the low dimensional manifold which uses a graph representation known as the spectral decomposition. The algorithm is known as the Laplacian eigenmaps which transform the data into some graph representation. Then it constructs the graph Laplacian matrix $L = D - A$ where A is the adjacency matrix and D is the degree matrix of the graph. Finally, it applies eigenvalue decomposition on the graph Laplacian. Further discussions can be found in [56].

t-SNE The t-distributed stochastic neighbour embedding converts the similarity of data points into probabilities. The similarity in the higher dimensional space is represented by Gaussian probabilities while the lower dimensional embedding uses student's t-distributions. Then gradient descent is applied to minimize the relative entropy (aka Kullback-Leibler divergence) of the joint probabilities in the two spaces. In general, t-SNE can group samples and extract local clusters of the data points. Such technique was first proposed and explored in [57].

C Lopsidedness and Cyclic Resultants

In this appendix, we give some examples for lopsidedness and cyclic resultants.

C.1 Genus for Lopsided Amoeba

Using lopsidedness, we can write the conditions for the number of genus for any $\mathcal{LA}_{\bar{P}_n}$. We now derive such conditions for some lopsided amoebae where $n = 1$.

Example 1: F_0 As one of the simplest examples, let us determine the genus of the lopsided amoeba for F_0 with $P(z, w) = c_1z + c_2w + c_3z^{-1} + c_4w^{-1} + c_5$ which could have at most one genus corresponding to its sole interior point¹⁹. Straight away, we can find the centre of the amoeba, which always lie in the hole if $g = 1$. We can find the spectral curve (spines) by considering the asymptotic behaviour as

- $z, w \rightarrow \infty, z/w \sim \mathcal{O}(1)$: this yields $\text{Log}|w| = \text{Log}|z| - \text{Log}\left|\frac{c_2}{c_1}\right|$.
- $1/z, 1/w \rightarrow \infty, z/w \sim \mathcal{O}(1)$: this yields $\text{Log}|w| = \text{Log}|z| + \text{Log}\left|\frac{c_4}{c_3}\right|$.
- $z, 1/w \rightarrow \infty, zw \sim \mathcal{O}(1)$: this yields $\text{Log}|w| = -\text{Log}|z| - \text{Log}\left|\frac{c_1}{c_4}\right|$.
- $1/z, w \rightarrow \infty, zw \sim \mathcal{O}(1)$: this yields $\text{Log}|w| = -\text{Log}|z| + \text{Log}\left|\frac{c_3}{c_2}\right|$.

In particular, the first two lines are parallel to each other, and so are the other two. The remaining 4 pairs give rise to 4 intersection points (which may or may not coincide). In other words, we have obtained the equations for the four spines of the amoeba and how they surround a rectangle with these 4 intersection points as vertices. One can check that this rectangle is centred at $\left(\frac{1}{2}\text{Log}\left|\frac{c_3}{c_1}\right|, \frac{1}{2}\text{Log}\left|\frac{c_4}{c_2}\right|\right)$.

¹⁹Here, all the coefficients can be any complex numbers. To avoid degenerate cases, we also require $c_{1,2,3,4} \neq 0$.

In general, to determine the genus, we should find all possibilities for the lopsided lists. Here, using (2.5), we have that $P\{x_1, x_2\} = \{|c_1z|, |c_2w|, |c_3/z|, |c_4/w|, |c_5|\}$. Suppose $|c_5|$ is the largest number, then we have the lopsided condition:

$$|c_5| > |c_1z| + |c_2w| + |c_3/z| + |c_4/w|. \quad (\text{C.1})$$

However, the right hand side reaches a minimum when $|c_1z| = |c_3/z|, |c_2w| = |c_4/w|$, i.e., $|z| = |c_3/c_1|^{1/2}, |w| = |c_4/c_2|^{1/2}$, which is exactly the aforementioned centre of the amoeba. Therefore, $|c_5|$ should at least be greater than this minimum for genus 1, and this bound is precisely $|c_5| > a := 2|c_1c_3|^{1/2} + 2|c_2c_4|^{1/2}$. In other words,

$$g = \begin{cases} 0, & |c_5| \leq a \\ 1, & \text{otherwise} \end{cases}. \quad (\text{C.2})$$

In particular, the centre point is precisely the point where the right hand side of (C.1) reaches its minimum.

For completeness, there are four more possibilities for $P\{x_1, x_2\}$ to be lopsided, but we can see that they would not lead to a non-zero genus by the same argument. For example, suppose the largest is

$$|c_1z| > |c_2w| + |c_3/z| + |c_4/w| + |c_5|. \quad (\text{C.3})$$

Let us now fix $|w|$, viz, contemplating a horizontal line on the Log plane. If we keep increasing $|z|$ (or equivalently $\text{Log}|z|$), this inequality would always hold. Therefore, this region, as a complementary component of the amoeba on the Log plane, would go to infinity. Hence, it is not bounded and cannot be a hole of the amoeba. Likewise, the other three inequalities would not give a hole either by considering the asymptotic behaviour of $|w|$ (or $|z|$) going to infinity or zero while keeping $|z|$ (or $|w|$) fixed. This also verifies that the lopsided amoeba for F_0 can have at most genus 1.

Example 2: $L^{3,3,2}$ Let us now consider $L^{3,3,2}$ as it is a non-reflexive polytopes (see [49]) and hence has more interior points. Its Newton polynomial is $P = c_1z + c_2w + c_3z^{-1} + c_4w^{-1} + c_5z^2 + c_6 = 0$. Therefore, $P\{\mathbf{x}\} = \{|c_1z|, |c_2w|, |c_3/z|, |c_4/w|, |c_5z^2|, |c_6|\}$. One possibility for these numbers to be lopsided is

$$|c_1z| > |c_2w| + |c_3/z| + |c_4/w| + |c_5z^2| + |c_6|. \quad (\text{C.4})$$

As $|z|$ cannot be zero, we can divide both sides by $|z|$ and then find the minimum for the right hand side. For $|w|$, it is easy to see that this requires $|w| = w_0 \equiv (|c_4/c_2|)^{1/2}$. Then for $|z|$, we have the cubic equation

$$|c_5||z|^3 - (2|c_2c_4|^{1/2} + |c_6|)|z| - 2|c_3| = 0. \quad (\text{C.5})$$

Write

$$p = -\frac{(2|c_2c_4|^{1/2} + |c_6|)}{|c_5|}, \quad q = -2\left|\frac{c_3}{c_5}\right|, \quad (C.6)$$

$$\Delta = \left(\frac{q}{2}\right)^2 + \left(\frac{p}{3}\right)^3 = \left|\frac{c_3}{c_5}\right|^2 - \frac{(2|c_2c_3|^{1/2} + |c_6|)^3}{27|c_5|^3}.$$

Based on the sign of the discriminant, we have three different cases. If $\Delta > 0$, or equivalently, $27|c_3^2c_5| > (2|c_2c_4|^{1/2} + |c_6|)^3$, then there is only one real root to this equation:

$$z_0 = \sqrt[3]{-\frac{q}{2} + \sqrt{\Delta}} + \sqrt[3]{-\frac{q}{2} - \sqrt{\Delta}}. \quad (C.7)$$

Since $q < 0$, z_0 is always positive. If $\Delta = 0$, there are three real roots. Again, due to negative q , we always have a positive root

$$z_0 = -2\sqrt[3]{\frac{q}{2}}. \quad (C.8)$$

If $\Delta < 0$, then we would have three distinct roots, $z_{1,2,3}$. Since $z_1 + z_2 + z_3 = 0$, there must be at least one positive root, which we shall still call z_0 . Hence, there would be (at least) one hole if

$$|c_1| > a_1 := |c_2|w_0/z_0 + |c_3|/z_0^2 + |c_4|/(z_0w_0) + |c_5|z_0 + |c_6|/z_0. \quad (C.9)$$

It is also possible that these numbers are lopsided as

$$|c_6| > |c_1z| + |c_2w| + |c_3/z| + |c_4/w| + |c_5z^2|. \quad (C.10)$$

Likewise, the right hand side reaches its minimum when $|w| = w_0 = (|c_4/c_2|)^{1/2}$ and $|z| = z'_0$ where z'_0 is a positive number satisfying²⁰

$$2|c_5|z_0'^3 + |c_1|z_0'^2 - |c_3| = 0. \quad (C.11)$$

Then there would be (at least) one hole if

$$|c_6| > a_2 := |c_1|z_0' + |c_2|w_0 + |c_3|/z_0' + |c_4|/w_0 + |c_5|z_0'^2. \quad (C.12)$$

²⁰One can show that there is always a positive root for this cubic equation. A quick way to see this is to consider the function $y = x^2(2|c_5|x + |c_1|)$, which is a cubic curve tangent to the x -axis at the origin (and always increasing for positive x). It also crosses the negative x -axis once while increasing. Then we can simply move this curve down along the y -axis to get $y = x^2(2|c_5|x + |c_1|) - |c_3|$. Hence, this would always give a positive root. Using this method, one can also check that both z_0 and z'_0 give local minima in the two cases.

One may check that other ways for $P\{\mathbf{x}\}$ to be lopsided would lead to unbounded complementary regions. To summarize²¹,

$$g = \begin{cases} 0, & |c_1| \leq a_1 \text{ and } |c_6| \leq a_2 \\ 1, & (|c_1| > a_1 \text{ and } |c_6| \leq a_2) \text{ or } (|c_1| \leq a_1 \text{ and } |c_6| > a_2) . \\ 2, & |c_1| > a_1 \text{ and } |c_6| > a_2 \end{cases} \quad (\text{C.13})$$

The punchline is that in this example, we are dealing with cubic (and quadratic) equations. Hence, we can always write down a full analytic condition for the genus. In general, for most of the polygons (even including reflexive ones) as well as \tilde{P}_n , we can always write certain equations to determine the genus for any coefficients, but there may not be general formulae to solve them analytically.

C.2 Cyclic Resultants for F_0

In this subsection, for reference, we list the cyclic resultants \tilde{P}_n for F_0 for $n = 1, 2, 3$:

$$\tilde{P}_1 = P = c_1 w z^2 + c_2 w^2 z + c_3 w + c_4 z + c_5 w z, \quad (\text{C.14})$$

$$\begin{aligned} \tilde{P}_2 = & c_2^4 w^8 z^4 - 2c_1^2 c_2^2 w^6 z^6 - 2c_2^2 c_3^2 w^6 z^2 + (4c_4 c_2^3 - 2c_5^2 c_2^2 - 4c_1 c_3 c_2^2) w^6 z^4 + c_1^4 w^4 z^8 \\ & + (4c_3 c_1^3 - 2c_5^2 c_1^2 - 4c_2 c_4 c_1^2) w^4 z^6 + (4c_1 c_3^3 - 2c_5^2 c_3^2 - 4c_2 c_4 c_3^2) w^4 z^2 \\ & + (c_5^4 - 4c_1 c_3 c_5^2 - 4c_2 c_4 c_5^2 + 6c_1^2 c_3^2 + 6c_2^2 c_4^2 - 8c_1 c_2 c_3 c_4) w^4 z^4 - 2c_1^2 c_4^2 w^2 z^6 \\ & - 2c_3^2 c_4^2 w^2 z^2 + (4c_2 c_4^3 - 2c_5^2 c_4^2 - 4c_1 c_3 c_4^2) w^2 z^4 + c_3^4 w^4 + c_4^4 z^4, \end{aligned} \quad (\text{C.15})$$

²¹If different holes combined with each other, then there would be a point (i.e., fixed $|z|, |w|$) in the hole satisfying more than one inequality. However, this is not possible for fixed $|z|, |w|$.

$$\begin{aligned}
\tilde{P}_3 = & z^9 c_2^9 w^{18} + 3z^{12} c_1^3 c_2^6 w^{15} + 3z^6 c_2^6 c_3^3 w^{15} + z^9 (-9c_4 c_5 c_2^7 + 3c_5^3 c_2^6 + 18c_1 c_3 c_5 c_2^6) w^{15} \\
& + 3z^3 c_2^3 c_3^6 w^{12} + 3z^{15} c_1^6 c_2^3 w^{12} + z^{12} (9c_2^3 c_3 c_5 c_1^4 - 21c_2^3 c_5^3 c_1^3 + 9c_2^4 c_4 c_5 c_1^3) w^{12} \\
& + z^6 (9c_3^3 c_4 c_5 c_2^4 - 21c_3^3 c_5^3 c_2^3 + 9c_1 c_3^4 c_5 c_2^3) w^{12} + z^9 (3c_4^3 c_2^6 - 27c_1 c_3 c_4^2 c_2^5 + 27c_4^2 c_5^2 c_2^5 \\
& - 18c_4 c_5^4 c_2^4 - 54c_1 c_3 c_4 c_5^2 c_2^4 + 54c_1^2 c_3^2 c_4 c_2^4 + 3c_5^6 c_2^3 + 9c_1 c_3 c_5^4 c_2^3 - 21c_1^3 c_3^3 c_2^3 \\
& + 27c_1^2 c_3^2 c_5^2 c_2^3) w^{12} + z^{18} c_1^9 w^9 + c_3^9 w^9 + z^{15} (-9c_3 c_5 c_1^7 + 3c_5^3 c_1^6 + 18c_2 c_4 c_5 c_1^6) w^9 \\
& + z^3 (-9c_1 c_5 c_3^7 + 3c_5^3 c_3^6 + 18c_2 c_4 c_5 c_3^6) w^9 + z^{12} (3c_3^3 c_1^6 + 27c_2^2 c_5^2 c_1^5 - 27c_2 c_3^2 c_4 c_1^5 \\
& - 18c_3 c_4^4 c_1^4 + 54c_2^2 c_3 c_4^2 c_1^4 - 54c_2 c_3 c_4 c_5^2 c_1^4 + 3c_5^6 c_1^3 + 9c_2 c_4 c_5^4 c_1^3 - 21c_2^3 c_4^3 c_1^3 \\
& + 27c_2^2 c_4^2 c_5^2 c_1^3) w^9 + z^6 (3c_1^3 c_3^6 + 27c_1^2 c_5^2 c_3^5 - 27c_1^2 c_2 c_4 c_3^5 - 18c_1 c_5^4 c_3^4 + 54c_1 c_2^2 c_4^2 c_3^4 \\
& - 54c_1 c_2 c_4 c_5^2 c_3^4 + 3c_5^6 c_3^3 + 9c_2 c_4 c_5^4 c_3^3 - 21c_2^3 c_4^3 c_3^3 + 27c_2^2 c_4^2 c_5^2 c_3^3) w^9 + z^9 (c_5^9 - 9c_1 c_3 c_5^7 \\
& - 9c_2 c_4 c_5^7 + 27c_1^2 c_3^2 c_5^5 + 27c_2^2 c_4^2 c_5^5 + 27c_1 c_2 c_3 c_4 c_5^5 - 21c_1^3 c_3^3 c_5^5 - 21c_2^3 c_4^3 c_5^5 \\
& - 27c_1 c_2^2 c_3 c_4^2 c_5^5 - 27c_1^2 c_2 c_3^2 c_4 c_5^5 - 18c_4^4 c_3^4 c_5 - 18c_2^4 c_4^4 c_5 + 117c_1 c_2^3 c_3 c_4^3 c_5 \\
& - 162c_1^2 c_2^2 c_3^2 c_4^2 c_5 + 117c_1^3 c_2 c_3^3 c_4 c_5) w^9 + 3z^{15} c_1^6 c_4^3 w^6 + 3z^3 c_3^6 c_4^3 w^6 + z^{12} (9c_3 c_4^3 c_5 c_1^4 \\
& - 21c_4^3 c_5^3 c_1^3 + 9c_2 c_4^4 c_5 c_1^3) w^6 + z^6 (9c_1 c_4^3 c_5 c_3^4 - 21c_4^3 c_5^3 c_3^3 + 9c_2 c_4^4 c_5 c_3^3) w^6 + z^9 (3c_2^3 c_4^6 \\
& + 27c_2^2 c_5^2 c_4^5 - 27c_1 c_2^2 c_3 c_4^5 - 18c_2 c_5^4 c_4^4 + 54c_1^2 c_2 c_3^2 c_4^4 - 54c_1 c_2 c_3 c_5^2 c_4^4 + 3c_5^6 c_4^3 + 9c_1 c_3 c_5^4 c_4^3 \\
& - 21c_1^3 c_3^3 c_4^3 + 27c_1^2 c_3^2 c_5^2 c_4^3) w^6 + 3z^{12} c_1^3 c_4^6 w^3 + 3z^6 c_3^3 c_4^6 w^3 + z^9 (-9c_2 c_5 c_4^7 + 3c_5^3 c_4^6 \\
& + 18c_1 c_3 c_5 c_4^6) w^3 + z^9 c_4^9,
\end{aligned} \tag{C.16}$$

References

- [1] I. Gelfand, M. Kapranov, and A. Zelevinsky, *Discriminants, Resultants, and Multidimensional Determinants*. Modern Birkhäuser Classics. Birkhäuser Boston, 2008. <https://books.google.co.uk/books?id=2zgxQVU1hFAC>.
- [2] I. Itenberg, G. Mikhalkin, and E. Shustin, *Tropical Algebraic Geometry*. Oberwolfach Seminars. Birkhäuser Basel, 2009. <https://books.google.co.uk/books?id=Se64NAEACAAJ>.
- [3] M. Gross, “Amoebas of complex curves and tropical curves,” in *UK-Japan winter school 2004—Geometry and analysis towards quantum theory*. University of Durham, UK, Jan 6–9, 2004, pp. 24–36. Yokohama: Keio University, 2004.
- [4] R. Kenyon, A. Okounkov, and S. Sheffield, “Dimers and amoebae,” *Ann. Math. (2)* **163** no. 3, (2006) 1019–1056, [math-ph/0311005](https://arxiv.org/abs/math-ph/0311005).
- [5] R. Kenyon and A. Okounkov, “Planar dimers and Harnack curves,” [arXiv:math/0311062](https://arxiv.org/abs/math/0311062).

- [6] B. Feng, Y.-H. He, K. D. Kennaway, and C. Vafa, “Dimer models from mirror symmetry and quivering amoebae,” *Adv. Theor. Math. Phys.* **12** no. 3, (2008) 489–545, [arXiv:hep-th/0511287](#).
- [7] J. J. Heckman and C. Vafa, “Crystal Melting and Black Holes,” *JHEP* **09** (2007) 011, [arXiv:hep-th/0610005](#).
- [8] H. Ooguri and M. Yamazaki, “Crystal Melting and Toric Calabi-Yau Manifolds,” *Commun. Math. Phys.* **292** (2009) 179–199, [arXiv:0811.2801 \[hep-th\]](#).
- [9] M. Cirafici, A.-K. Kashani-Poor, and R. J. Szabo, “Crystal melting on toric surfaces,” *J. Geom. Phys.* **61** (2011) 2199–2218, [arXiv:0912.0737 \[hep-th\]](#).
- [10] A. Zahabi, “Quiver asymptotics and amoeba: Instantons on toric Calabi-Yau divisors,” *Phys. Rev. D* **103** no. 8, (2021) 086024, [arXiv:2006.14041 \[hep-th\]](#).
- [11] M. Yamazaki, “Brane Tilings and Their Applications,” *Fortsch. Phys.* **56** (2008) 555–686, [arXiv:0803.4474 \[hep-th\]](#).
- [12] M. Yamazaki, “Geometry and Combinatorics of Crystal Melting,” *RIMS Kokyuroku Bessatsu B* **28** (2011) 193, [arXiv:1102.0776 \[math-ph\]](#).
- [13] M. Bogaard, “Introduction to amoebas and tropical geometry,” *Masters thesis, U. Amsterdam* (2015) . <https://scripties.uba.uva.nl/download?fid=564840>.
- [14] Y.-H. He, “Calabi-Yau Varieties: from Quiver Representations to Dessins d’Enfants,” in *Proc. of Workshop on Grothendieck-Teichmüller Theories, Nankai, 2016*. 11, 2016. [arXiv:1611.09398 \[math.AG\]](#).
- [15] D. Maclagan and B. Sturmfels, *Introduction to Tropical Geometry*. Graduate Studies in Mathematics. American Mathematical Society, 2015. <https://books.google.co.uk/books?id=zFsoCAAAQBAJ>.
- [16] J. Bao, Y.-H. He, E. Hirst, and S. Pietromonaco, “Lectures on the Calabi-Yau Landscape,” [arXiv:2001.01212 \[hep-th\]](#).
- [17] Y.-H. He, “Deep-Learning the Landscape,” *Phys. Lett. B* **774** (6, 2017) 564–568, [arXiv:1706.02714 \[hep-th\]](#).
- [18] D. Krefl and R.-K. Seong, “Machine Learning of Calabi-Yau Volumes,” *Phys. Rev. D* **96** no. 6, (2017) 066014, [arXiv:1706.03346 \[hep-th\]](#).
- [19] F. Ruehle, “Evolving neural networks with genetic algorithms to study the String Landscape,” *JHEP* **08** (2017) 038, [arXiv:1706.07024 \[hep-th\]](#).
- [20] J. Carifio, J. Halverson, D. Krioukov, and B. D. Nelson, “Machine Learning in the String Landscape,” *JHEP* **09** (2017) 157, [arXiv:1707.00655 \[hep-th\]](#).
- [21] Y.-H. He, E. Hirst, and T. Peterken, “Machine-learning dessins d’enfants: explorations

- via modular and Seiberg–Witten curves,” *J. Phys. A* **54** no. 7, (2021) 075401, [arXiv:2004.05218 \[hep-th\]](#).
- [22] J. Bao, Y.-H. He, E. Hirst, J. Hofscheier, A. Kasprzyk, and S. Majumder, “Hilbert Series, Machine Learning, and Applications to Physics,” [arXiv:2103.13436 \[hep-th\]](#).
- [23] Y.-H. He, “The Calabi-Yau Landscape: from Geometry, to Physics, to Machine-Learning; to appear, Springer LNM,” [arXiv:1812.02893 \[hep-th\]](#).
- [24] Y.-H. He, “Machine-Learning Mathematical Structures,” [arXiv:2101.06317 \[cs.LG\]](#).
- [25] Y.-H. He and S.-T. Yau, “Graph Laplacians, Riemannian Manifolds and their Machine-Learning,” [arXiv:2006.16619 \[math.CO\]](#).
- [26] L. Zhang, G. Naitzat, and L.-H. Lim, “Tropical geometry of deep neural networks,” [arXiv:1805.07091 \[cs.LG\]](#).
- [27] V. Charisopoulos and P. Maragos, “A tropical approach to neural networks with piecewise linear activations,” [arXiv:1805.08749 \[stat.ML\]](#).
- [28] J. Hauenstein, Y.-H. He, I. Kotsireas, D. Mehta, and T. Tang, “Algebraic Geometry and Machine Learning,” in *Journal of Symbolic Computation*. Springer, 2021.
- [29] D. Cox, J. Little, and H. Schenck, *Toric Varieties*. Graduate studies in mathematics. American Mathematical Society, 2011.
<https://books.google.co.uk/books?id=AoSDAwAAQBAJ>.
- [30] G. E. Collins, “Quantifier elimination for real closed fields by cylindrical algebraic decomposition,” in *Automata theory and formal languages*, pp. 134–183. Springer, 1975.
- [31] M. Ben-Or, D. Kozen, and J. Reif, “The complexity of elementary algebra and geometry,” in *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pp. 457–464. 1984.
- [32] T. Theobald, “Computing amoebas,” *Experimental Mathematics* **11** no. 4, (2002) 513–526.
- [33] J. Forsgård, L. F. Matusevich, N. Mehlhop, and T. de Wolff, “Lopsided approximation of amoebas,” [arXiv:1608.08663 \[cs.SC\]](#).
- [34] K. Purbhoo, “A nullstellensatz for amoebas,” [arXiv:math/0603201](#).
- [35] J. Bao, S. Franco, Y.-H. He, E. Hirst, G. Musiker, and Y. Xiao, “Quiver Mutations, Seiberg Duality and Machine Learning,” *Phys. Rev. D* **102** no. 8, (2020) 086013, [arXiv:2006.10783 \[hep-th\]](#).
- [36] B. Feng, A. Hanany, and Y.-H. He, “D-brane gauge theories from toric singularities and toric duality,” *Nucl. Phys. B* **595** (2001) 165–200, [arXiv:hep-th/0003085](#).
- [37] Y. He, “D-brane gauge theory and toric singularities,” *Int. J. Mod. Phys. A* **16S1C** (2001) 981–983.

- [38] B. Feng, Y.-H. He, and F. Lam, “On correspondences between toric singularities and (p,q) webs,” *Nucl. Phys. B* **701** (2004) 334–356, [arXiv:hep-th/0403133](#).
- [39] B. Feng, A. Hanany, Y.-H. He, and A. M. Uranga, “Toric duality as Seiberg duality and brane diamonds,” *JHEP* **12** (2001) 035, [arXiv:hep-th/0109063](#).
- [40] A. Hanany and K. D. Kennaway, “Dimer models and toric diagrams,” [arXiv:hep-th/0503149](#).
- [41] S. Franco, A. Hanany, K. D. Kennaway, D. Vegh, and B. Wecht, “Brane dimers and quiver gauge theories,” *JHEP* **01** (2006) 096, [arXiv:hep-th/0504110](#).
- [42] K. Hori and C. Vafa, “Mirror symmetry,” [arXiv:hep-th/0002222](#).
- [43] K. Hori, A. Iqbal, and C. Vafa, “D-branes and mirror symmetry,” [arXiv:hep-th/0005247](#).
- [44] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, “The expressive power of neural networks: A view from the width,”.
- [45] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2017.
- [46] B. Bengfort and R. Bilbro, “Yellowbrick: Visualizing the Scikit-Learn Model Selection Process,”. <http://joss.theoj.org/papers/10.21105/joss.01075>.
- [47] A. Hanany and R.-K. Seong, “Brane Tilings and Specular Duality,” *JHEP* **08** (2012) 107, [arXiv:1206.2386 \[hep-th\]](#).
- [48] Y.-H. He, R.-K. Seong, and S.-T. Yau, “Calabi–Yau Volumes and Reflexive Polytopes,” *Commun. Math. Phys.* **361** no. 1, (2018) 155–204, [arXiv:1704.03462 \[hep-th\]](#).
- [49] J. Bao, G. Beane Colverd, and Y.-H. He, “Quiver Gauge Theories: Beyond Reflexivity,” *JHEP* **06** (2020) 161, [arXiv:2004.05295 \[hep-th\]](#).
- [50] C. R. Brodie, A. Constantin, R. Deen, and A. Lukas, “Machine Learning Line Bundle Cohomology,” *Fortsch. Phys.* **68** no. 1, (2020) 1900087, [arXiv:1906.08730 \[hep-th\]](#).
- [51] Y.-H. He and M. Kim, “Learning Algebraic Structures: Preliminary Investigations,” [arXiv:1905.02263 \[cs.LG\]](#).
- [52] M. Abadi *et al.*, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. <http://tensorflow.org/>. Software available from tensorflow.org.
- [53] C. Tralie, N. Saul, and R. Bar-On, “Ripser.py: A lean persistent homology library for python,” *The Journal of Open Source Software* **3** no. 29, (Sep, 2018) 925. <https://doi.org/10.21105/joss.00925>.
- [54] J. B. Tenenbaum, V. De Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *science* **290** no. 5500, (2000) 2319–2323.
- [55] I. Borg and P. J. Groenen, *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

- [56] M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural computation* **15** no. 6, (2003) 1373–1396.
- [57] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.” *Journal of machine learning research* **9** no. 11, (2008) .