AdS/CFT Integrability and Machine Learning Combinatorial Structures

Dmitrii Riabchenko

Thesis submitted in fulfilment of the requirements for the award of the degree Doctor of Philosophy

Department of Mathematics
City St George's, University of London
August 2025

Declaration

I, Dmitrii Riabchenko, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Throughout the course of this PhD, I have co-authored two published papers and one prepared to be published, which became the basis of this thesis:

- Chen, S., Dechant, P.P., He, YH. et al. Machine Learning Clifford Invariants of ADE Coxeter Elements. Adv. Appl. Clifford Algebras 34, 20 (2024). https://doi.org/10.1007/s00006-024-01325-y
- Ohlsson Sax, O., Riabchenko, D., Stefański Jr., B. Worldsheet kinematics, dressing factors and odd crossing in mixed-flux AdS3 backgrounds. J. High Energ. Phys. 2024, 132 (2024). https://doi.org/10.1007/JHEP09(2024)132
- He, YH., Kasprzyk, A., Le, Q, Riabchenko, D. Machine Learning Discovers New Champion Codes. Preprint (Version 1) available at Research Square (2025). https://doi.org/10.21203/rs.3.rs-7196769/v1

Abstract

This thesis explores two distinct but powerful approaches used in theoretical physics and mathematics to study complex systems: AdS/CFT integrability and Machine Learning.

The first part starts with an introduction to AdS/CFT integrability, reviewing the integrability in $AdS_5 \times S^5$ and $AdS_3 \times S^3 \times T^4$ backgrounds, which is then used to present the novel kinematical structure of the mixed-flux $AdS_3 \times S^3 \times T^4$ background and a previously unknown solution for the odd part of the dressing factor of the exact worldsheet S-matrix.

The second part applies Machine Learning to two combinatorial problems in Clifford algebras and error-correcting codes. After a brief introduction of Machine Learning methods, we explore the potential of network classification and principal component analysis in the study of Clifford geometric invariants of Coxeter elements in the root systems of ADE algebras, demonstrating the viability of the computational approach and the potential for further analytical exploration. We then address the problem of searching for new champion codes of generalised toric codes. We develop a pipeline that employs a Transformer Deep Learning architecture and Genetic Algorithm to find new champion codes for $\mathbb{F}_8$ generalised toric codes.

Contents

List of Figures

List of Tables

Acknowledgements

First, I am grateful to my supervisor, Professor Bogdan Stefanski, for his mentorship, insightful critiques, and constant support. I also wish to thank my co-supervisor, Professor Yang-Hui He, who introduced me to the field of Machine Learning and involved me in many fruitful collaborations. Their wisdom and kindness have taught me many valuable lessons and helped me to grow, both scientifically and personally.

I want to thank my examiners, Dr Evgeny Sobko and Dr Alessandro de Martino, and the viva chair, Dr Maud Visscher, for their time and thoughtful reading of my thesis and making the viva a very smooth and enjoyable experience. It was a pleasure to take part.

I would also like to thank my many collaborators: Suvajit Majumder, Olof Ohlsson Sax, Edward Hirst, Elli Heyes, Siqi Chen, Pierre-Philippe Dechant, Le Q and Alexander Kasprzyk. It was a pleasure and an honour to know and work with them.

I sincerely thank my colleagues and fellow PhD students in the Department of Mathematics at City, University of London. It was a pleasure to share lunches and worries with Hasan Haq, Fabio Sailis, Michele Mazzoni, Bethan Turner, Elli Heyes, Vaselin Manojlovic, Charles Thull and many others whom I met there. A special thanks goes to my office buddies, Aman Anand and Manjyot Singh Bedi, for the unforgettable time together. Furthermore, I greatly admire Suvajit Majumder and Jiakang Bao and enjoyed the time we spent discussing physics and beyond.

I am grateful for the financial support from the School of Science and Technology at City, University of London and the UK Science and Technology Facilities Council (STFC).

Finally, I am deeply thankful to my parents, Vadim and Svetlana, who instilled in me a love for mathematics and physics and supported my scientific aspirations. I would especially like to thank the love of my life, my incredible girlfriend Valeriia, who became my wife during the PhD and who has supported me unwaveringly throughout these years.

Chapter 1

Introduction

The holographic principle is based on the idea that combining quantum mechanics and gravity requires the universe to be an image of data that can be stored on a boundary with one less dimension. It was originally proposed by Gerard 't Hooft [1], promoted by Leonard Susskind [2] and achieved the most successful realisation in a study by Juan Maldacena in 1997 [3].

Maldacena proposed a concrete conjecture that relates the gravity theory in asymptotically Anti-de Sitter spacetimes (AdS) to conformal field theory (CFT) on the boundary (thus another name of it – AdS/CFT correspondence). The conjecture has two essential features: it is a duality between conformal field theory and string theory, and it is a strong-weak duality. The concept of duality in this context refers to the equivalence between specific pairs of models such that all observable quantities in one theory are the same as those in another. They describe the same physics, but the languages used to describe these models can be very different. It is important to note, however, that despite the huge success in many areas and the plethora of checks conducted over the last 25 years, this duality is still a conjecture.

Maldacena's conjecture offers a novel approach in the pursuit of quantum gravity. Historically, the unification of all quantum theories of forces into a single framework, the Standard Model, has been very successful, except for one ingredient: gravity. There were many attempts to quantise gravity, starting from the works of Rosenfeld in 1930 [4] and canonical quantisation by DeWitt in 1967 [5], and it is still very much in development. In this case, AdS/CFT can provide access to previously inaccessible regimes and more insights through string theory as a theory of gravitation. For instance, it helped better understand the quantum nature of black holes [6] and symmetries in gravity [7].

Another application of the AdS/CFT correspondence comes from its strong-weak duality property. This is the property that the results, which are very difficult to compute on one side of the duality, can be obtained more easily on the other side, and vice versa. Such a feature was used to investigate the properties of quantum gravity by considering non-gravitational counterparts, but also in the opposite direction. Focusing on the small coupling regimes of string theory, where it becomes classical gravity, it is possible to study strongly coupled theories, such as $\mathcal{N} = 4$ SYM. The latter is one of the simplest examples of four-dimensional gauge theory. In turn, a better understanding of $\mathcal{N} = 4$ SYM may help in understanding other gauge theories, such as Quantum Chromodynamics (QCD). Developed in 1970 to describe strong interactions, QCD is very complicated because of its large coupling constant at low energies, which renders most standard quantum field theory methods ineffective because they are perturbative. Unlike $\mathcal{N} = 4$ SYM, QCD is not conformal. But it is asymptotically free, which means that at high energies, it is close to being conformal

so that features such as high-energy gluon scattering can be understood by studying gauge boson amplitudes in $\mathcal{N} = 4$ SYM [8].

One more powerful idea that has the potential to help with the understanding of many physical theories is integrability[1]. This is a property of certain theories, such as $\mathcal{N} = 4$ SYM, which allows one to solve them exactly. Roughly speaking, such theories possess an infinite number of conserved charges, which can be used to describe the behaviour of a model without actually solving the equations of motion. Therefore, it can be used to study theories in the nonperturbative regime, as the results obtained by integrability methods hold for all energy scales. Additionally, it may be compatible with the AdS/CFT correspondence, so that if one of the dual theories is integrable, the other will be as well. This is potentially very useful as it allows verification of the correctness of the AdS/CFT correspondence, as it is a strong-weak or strong-strong duality. In this case, to check that duality really holds, one needs to compare the results on both sides. The computation of observables on the side with strong coupling is difficult, and thus, integrability can be employed.

Although these analytical tools, duality and integrability, are powerful, they have limitations. When the solution is derived, it may be too complex, or the solution space may be too vast. For instance, in string theory, the landscape of possible vacua is enormous [11], [12] and so is intractable with analytical methods. In such cases, computational approaches are more feasible. Recently, Machine learning (ML) has emerged as a tool in theoretical sciences to generate insights where traditional analytical or numerical techniques fall short.

ML technique applications started from investigations of string theory [12], [13], [14], [15], [16] but quickly spread to other areas of mathematics and theoretical physics. These methods benefit from both the abundance of data naturally generated in physical and mathematical models and the increasing availability of hardware and software for handling large-scale computations.

In particular, ML techniques have been applied to study theoretical topics such as the symmetries of physical systems [17], [18], [19], [20] and integrable systems [21], [22], [23], [24], and even to search for new integrable models [25]. However, the range of applications is much wider, as one of the distinct features of modern ML techniques is their universality. ML techniques have been applied in studies ranging from particle detection at the Large Hadron Collider to cosmological observations [26]. As a result, advancing the methods applied in one field often leads to progress in other fields.

A particular class of problems that is suitable for ML applications can be coarsely named as combinatorial. Combinatorics is a vast field with applications ranging from computer science to statistical physics and logic. There is no formal definition of it, so it is sometimes defined as a field of mathematics concerned with the study of the arrangement of elements into discrete, finite sets [27]. These problems are challenging because they typically involve a large number of configurations, and the space of potential solutions for a given problem is at least exponential in the size of the element set in the problem [28]. However, this same feature makes it an ideal playground for ML because of the large dataset requirements.

This thesis presents the necessary background material as well as the author's contributions in both of these domains. The first part is devoted to AdS/CFT integrability and culminates in the presentation of a novel kinematical structure and the solution of the odd part of massive crossing for string theory in $\mathrm{AdS}_3 \times \mathrm{S}^3 \times \mathrm{T}^4$ background supported by mixed flux. In the second part, we consider the application of Machine Learning to study combinatorial structures. First, we establish the viability of Machine Learning as a tool for experimental mathematics for analysing and classifying Clifford geometric invariants of Coxeter elements

[1] For introduction see [9] and [10].

in ADE root systems. Second, we construct an ML pipeline for discovering champion generalised toric codes, which is potentially applicable to other codes.

Outline

In Chapter 2, we begin with a general review of AdS/CFT integrability. We first consider the AdS/CFT correspondence conjecture and construct the dualities arising from a stack of D3-branes and D1-D5 brane systems related to string theory in $\mathrm{AdS}_5 \times \mathrm{S}^5$ and $\mathrm{AdS}_3 \times \mathrm{S}^3 \times \mathrm{T}^4$ backgrounds, respectively. Then, we consider the integrability of $\mathrm{AdS}_5 \times \mathrm{S}^5$ string theory and derive its exact worldsheet excitation S-matrix, with a focus on the dressing factor. This is followed by an integrability in $\mathrm{AdS}_3 \times \mathrm{S}^3 \times \mathrm{T}^4$ with pure Ramond-Ramond flux, where we also consider the construction of the exact S-matrix and its dressing factor. Building on this foundation, we present original results for $\mathrm{AdS}_3 \times \mathrm{S}^3 \times \mathrm{T}^4$ mixed Ramond-Ramond and Neveu–Schwarz-Neveu–Schwarz flux theory in Chapter 3, where we discuss the novel kinematics and construct a previously unknown solution for the odd part of the dressing factor, as published in [29].

Chapter 4 introduces the background material on Machine Learning for further application to combinatorial problems. We review the necessary techniques, such as Principal Component Analysis, Neural Networks and Sequence Modelling architectures, and the Genetic Algorithm.

In Chapter 5, we apply Machine Learning to the study of Clifford geometric invariants, based on the publication [19]. We begin by creating a dataset of Clifford invariants for Coxeter elements in the A_8, D_8, and E_8 root systems. Setting the goal to test an experimental mathematics approach, we perform the analysis including identifying structural patterns in the invariants, neural network classification and regression of them, followed by gradient saliency analysis and principal component analysis. The results prepare the background for further theoretical study.

In Chapter 6, we present a method for discovering new champion generalised toric codes using Machine Learning, based on work prepared for publication [30]. We propose a Transformer-based model for the minimum Hamming distance prediction of a given code. It is then used in a Genetic Algorithm pipeline to efficiently search the space of codes and identify new champion codes over $\mathbb{F}_8$.

Chapter 7 concludes the thesis by summarising the main findings and discussing potential directions for future research.

Chapter 2

AdS/CFT Integrability

2.1 AdS/CFT Correspondence

The AdS/CFT correspondence relates string theory on asymptotically Anti-de Sitter spacetimes[1] to certain conformal field theories. Examples include $AdS_5 \times S^5$[2] strings and $D = 4$ $\mathcal{N} = 4$ supersymmetric Yang–Mills (SYM) theory [31], $AdS_4 \times CP^3$[3] superstring and ABJM Chern-Simons theories [32] and many more [33].

We will use the example of $AdS_5 \times S^5$ to show its key features. The strongest form of the AdS_5/CFT_4 correspondence (or duality) states that $D = 4$ $\mathcal{N} = 4$ Super Yang-Mills (SYM) theory with gauge group $SU(N)$ and Yang-Mills coupling constant g_{YM} is dynamically equivalent to the type IIB superstring theory with string length $l_s = \sqrt{\alpha'}$[4] and the coupling constant g_s on $AdS_5 \times S^5$ with radius of curvature L and N units of $F_{(5)}$ flux on S^5. The two free parameters on the field theory side, i.e. g_{YM} and N, are mapped to the free parameters g_s and $L/\sqrt{\alpha'}$ on the string theory side by

$$g_{\text{YM}}^2 = 2\pi g_s\,, \qquad 2g_{\text{YM}}^2 N = L^4/\alpha'^2\,. \tag{2.1.1}$$

The duality is difficult to prove for the general values of the free parameters g_{YM} (g_s) and N ($L/\sqrt{\alpha'}$). However, we can take $g_s \ll 1$ with $L/\sqrt{\alpha}$ kept constant, that is, working with perturbative string theory. On the 'CFT' side, this maps to $g_{\text{YM}} \ll 1$ with $\lambda = g_{\text{YM}}^2 N$ kept constant, that is, the large N limit for fixed λ (t'Hooft limit). This is a strong form of duality.

The remaining free parameters of both sides are related by

$$2\lambda = \frac{L^4}{\alpha'^2}\,. \tag{2.1.2}$$

Then, in the weak form of duality, we consider the limit $\lambda \to \infty$, which is a strong-coupling field theory, and $\sqrt{\alpha'}/L \to 0$, which is the small curvature limit of string theory, that is, supergravity.

In the next section, we consider these two limits, which are the weak form of duality. This is the easiest path to motivate the conjecture using the arguments of [3]. For a more detailed discussion, see the review [33] and book [34].

[1] Anti-de Sitter spacetime is a maximally symmetric Lorentzian manifold, an exact solution of the Einstein field equations for an empty universe with negative cosmological constant.

[2] Product space of 5-dimensional Anti-de Sitter spacetime and 5-dimensional sphere.

[3] Product space of 4-dimensional Anti-de Sitter spacetime and 3-dimensional complex projective space.

[4] α' is the "slope parameter" which appears in front of Nambu-Goto action and determines the string length or string tension parameters.

2.1.1 AdS_5 from D3 Branes

In Maldacena's conjecture, string theory is defined on the product spacetime of five-dimensional Anti-de Sitter space and a five-dimensional sphere, usually denoted as $AdS_5 \times S^5$[5], and the dual-field theory is a conformally invariant theory residing in a flat (3+1)-dimensional spacetime.

The duality can be seen arising within the framework of superstring theory by considering type IIB strings in flat space with a stack of N coincident D3-branes. For convenience, we assume that the stack of branes extends along the spacetime dimensions x^0, x^1, x^2, x^3, with Neumann boundary conditions along these and Dirichlet boundary conditions along the rest. This configuration breaks half of the thirty-two supercharges of the IIB string theory in flat spacetime.

This stack of D-branes can be viewed from the perspective of either closed or open strings.

Open string perspective

In the open string perspective, D-branes may be viewed as higher-dimensional objects at which open strings end. While string interactions with a single D-brane are described by g_s, the interaction with a stack of N D-branes has an effective coupling constant given by $g_s N$. For the current perspective to be valid, the strings should not perturb the background too much. Therefore, $g_s N \ll 1$. Additionally, because we consider a weak form of duality, energies are low $E \ll \alpha'^{-1/2}$.

The low-energy limit means that we can only consider massless excitations. For the open string, these excitations can be grouped into a four-dimensional $\mathcal{N} = 4$ supermultiplet, which consists of a gauge field A_μ, six real scalar fields ϕ^i and their fermionic superpartners. The fact that we have a stack of N coincident D3-branes results in the $U(N)$ gauge group for the A_μ field because open strings interacting with D-branes are also labelled by $U(N)$ Chan-Paton factors.

The complete effective action for all massless string modes is

$$S = S_{\text{closed}} + S_{\text{open}} + S_{\text{int}} \,. \tag{2.1.3}$$

The S_{closed} part is the action of ten-dimensional supergravity plus higher derivative terms which are suppressed by powers of $\alpha'^{1/2} E \ll 1$. If we only preserve the lowest-order contribution in α' to metric fluctuations in the action, we get

$$S_{\text{closed}} = -\frac{1}{2} \int d^{10}x \partial_M h \partial^M h + \mathcal{O}(\kappa) \,, \tag{2.1.4}$$

where h is given by $g = \eta + \kappa h$ and κ is $2\kappa^2 = (2\pi)^7 \alpha'^4 g_s^2$.

The other two terms in (2.1.3), S_{open} and S_{int}, are derived from the Dirac-Born-Infeld ac-

[5]This spacetime is a supergravity solution. In supergravity, both AdS_5 and S^5 parts have the same radius L, although in string theory, they might have different radii.

tion [35]. At the lowest order in α', the Dirac-Born-Infeld action can be decomposed into

$$\begin{aligned} S_{\text{open}} &= -\frac{1}{2\pi g_s}\int dx^4 \text{Tr}\left(\frac{1}{4}F^a_{\mu\nu}F^{a\mu\nu}+\frac{1}{2}\eta^{\mu\nu}D_\mu\phi^i D_\nu\phi^i \right. \\ &\qquad\qquad \left. -\frac{1}{4}\sum_{i,j}[\phi^i,\phi^j]^2+\mathcal{O}(\alpha')\right), \\ S_{\text{int}} &= 0+\mathcal{O}(\alpha'), \end{aligned} \tag{2.1.5}$$

where $F_{\mu\nu}$ is the usual field strength tensor of the non-abelian gauge theory of A^a_μ, D_μ is the covariant derivative associated with A_μ, $A_\mu = A^a_\mu T_a$ and $\phi^i = \phi^{ia}T_a$ are algebra-valued fields. In the above expressions, all fermionic fields are omitted.

In the low-energy limit $S_{\text{int}} \sim 0$, so closed and open strings decouple. Thus, we end up with the free supergravity in (9+1)-dimensional Minkowski spacetime from S_{closed} and $\mathcal{N}=4$ SYM theory from S_{open} provided we identify

$$2\pi g_s = g^2_{\text{YM}}. \tag{2.1.6}$$

If we were to separate one of the branes from the others in x^9 direction such that all the branes are at $x^9=0$ and the last one at $x^9=r$, we would get a system in a Higgs phase with an expectation value for one of the scalar fields being $\langle\phi\rangle = \frac{r}{2\pi\alpha'}$ [3], [36]. In this setup, if we decide to take the limit $\alpha'\to 0$, we should ensure that all the physical quantities remain finite, including the expectation value of the scalar fields. Thus, we should be more careful when taking this limit by demanding

$$\alpha'\to 0 \quad \text{with} \quad U=\frac{r}{\alpha'} \quad \text{fixed}. \tag{2.1.7}$$

Closed string perspective

The closed-string perspective is valid when the coupling constant g_sN is large. Additionally, we consider low energies $E \ll {\alpha'}^{-1/2}$.

In this picture, the D-branes can be viewed as solitonic solutions of the supergravity. The stack of coincident D-branes curves the surrounding background with a characteristic length L. To ensure the weak curvature and validity of the supergravity approximation, $L/\sqrt{\alpha'}$ should be large, and because $L^4/{\alpha'}^2 \sim g_sN$, we should also take $g_sN \gg 1$.

The supergravity solution of N D3-branes preserving $\text{SO}(3,1)\times\text{SO}(6)$ isometries of $\mathbb{R}^{9,1}$ and half of the supercharges of type IIB supergravity are given by

$$\begin{aligned} ds^2 &= H(r)^{-1/2}\left(-dt^2+\sum_{i=1}^3 dx_i^2\right)+H(r)^{1/2}\left(dr^2+r^2d\Omega_5^2\right), \\ e^{2\phi(r)} &= g_s^2, \\ C_{(4)} &= \left(1-H(r)^{-1}\right)dx^0\wedge dx^1\wedge dx^2\wedge dx^3+\dots, \end{aligned} \tag{2.1.8}$$

where $\mu,\nu = 0\dots3$; $i,j=4\dots9$ and $r^2=\sum_{i=4}^9 x_i^2$

The type IIB supergravity equation of motion can be solved to find $H(r)$ and get the following supergravity metric for the stack of D3-branes [37]

$$ds^2 = \frac{1}{\left(1+\frac{L^4}{r^4}\right)}\left(-dt^2+\sum_{i=1}^3 dx_i^2\right)+\left(1+\frac{L^4}{r^4}\right)\left(dr^2+r^2d\Omega_5^2\right), \tag{2.1.9}$$

where $d\Omega_n^2$ is the round metric in n dimensions, and r is the associated radial coordinate.

We also know that the flux of $F_{(5)}$ through the S^5 should be quantised and should depend on the number of coincident D3-branes

$$Q = \int_{S^5} *F_{(5)} = N \,. \tag{2.1.10}$$

This integral can be evaluated using the solution 2.1.8 and the fact that $F_{(5)} = dC_{(4)}$, to get

$$L^4 = 4\pi g_s N \alpha'^2 \,. \tag{2.1.11}$$

Now, looking at (2.1.9), we can identify two regions in this background: small r and large r. If $r \gg L$, then $H(r) \sim 1$ and the metric describes a flat Minkowski spacetime. Thus, when $r \gg L$, we obtain a theory of closed strings on a flat 10-dimensional background. However, if $r \ll L$, then $H(r) \sim L^4/r^4$ is large such that we can neglect 1 in $H(r)$. This metric describes the near-horizon region (or throat) of the stack of D3-branes

$$ds^2 = \frac{L^2}{z^2}\left(\eta_{\mu\nu}dx^\mu dx^\nu + dz^2\right) + L^2 ds^2_{S^5} \tag{2.1.12}$$

where we introduced the coordinate $z = L^2/r$ and spherical coordinates (r, Ω_5) with $ds^2_{S^5}$ being the metric on S^5 sphere. Therefore, when $r \ll L$, we obtain a theory of closed strings on $AdS_5 \times S^5$ background.

Finally, we take a low-energy limit $E \ll \alpha'^{-1/2}$ and strings in both regions should decouple. This can be observed by examining the energies of the excitations. The energy E of an excitation at distance r from the horizon is related to the energy of a distant observer by the redshift factor

$$E_\infty = \sqrt{-g_{00}} E_r \,. \tag{2.1.13}$$

If we compare energies at infinity and the throat, we find

$$E_\infty = \lim_{r\to 0} \frac{r^2}{L^2} E_r \,. \tag{2.1.14}$$

Therefore, excitations at infinity have very small energies compared to the excitation at the throat. Therefore, the observer at infinity sees two different low-energy regions: the supergravity modes on a flat Minkowski space-time at infinity and string excitations on $AdS_5 \times S^5$ near the stack of branes.

Comparing perspectives

By comparing the two perspectives, we can conjecture that because the 10-dimensional supergravity part is the same on both sides, type IIB supergravity on $AdS_5 \times S^5$ is equivalent to $\mathcal{N} = 4$ SYM theory [3].

The argument above shows only the correspondence of classical supergravity and strongly coupled $\mathcal{N} = 4$ SYM. However, duality is believed to hold more generally. In particular, if we relax the low-energy condition, the classical type IIB string theory in $AdS_5 \times S^5$ should be equivalent to $\mathcal{N} = 4$ SYM theory in (3+1)-dimensions in the planar limit. Moreover, we can relax the condition $N \gg 1$ to obtain the strongest form of the duality: $\mathcal{N} = 4$ SYM theory at arbitrary N and λ should be equivalent to IIB theory quantum string theory on $AdS_5 \times S^5$ background with arbitrary g_s and α'/L^2.

This is the strong form of duality that will be useful in the discussion of integrability in section 2.2.

2.1.2 AdS_3 from D1-D5 Branes

In this section, we consider the D1-D5 brane system for AdS_3/CFT_2 correspondence, which was proposed in [3] and further discussed in [33]. Arguments, similar to those for the strings in $AdS_5 \times S^5$ presented above, will reveal that D1-D5 brane system gives raise to $AdS_3 \times S^3 \times T^4$ supergravity background with pure Ramond-Ramond (R-R) 3-form flux[6]. For more detailed discussion, see [3], [33] or lecture notes [38].

We consider a system of N_1 coincident D1-branes together with N_2 coincident D5-branes. The D1-branes extend along two of the dimensions, one in time and one in space, which the D5-branes span. The other four dimensions along which the D5-branes extend are compactified on a T^4. This system is described by a metric [39]

$$ds^2 = \frac{1}{\sqrt{f_1 f_5}}(-dt^2 + dx_1^2) + \sqrt{f_1 f_5}(dr^2 + r^2 d\Omega_3^2) + \sqrt{\frac{f_1}{f_5}} \sum_{i=6}^{9} dx_i^2 \,, \tag{2.1.15}$$

where $f_1 = 1 + \frac{g_s \alpha' N_1}{v r^2}$, $f_5 = 1 + \frac{g_s \alpha' N_5}{r^2}$, $v = \frac{V_{T^4}}{(2\pi)^4 \alpha'^2}$, V_{T^4} – volume of T_4.

The D1-D5 system couples magnetically and electrically to the R-R field: 3-form flux $F^{(3)}$. In the near-horizon limit, it is proportional to the volume forms on AdS_3 and S^3 [40].

In line with the procedure for the D3-branes stack, we take the supergravity limit $g_s N_1 \gg 1$, $g_s N_5 \gg 1$. Then, when taking the near-horizon limit $\alpha' \to 0$, we should ensure all the physical quantities stay constant, which leads to the following constraints

$$\frac{r}{\alpha'} = \text{fixed}, \quad v = \frac{V_{T^4}}{(2\pi)^4 \alpha'^2} = \text{fixed}, \quad g_6 = \frac{g_s}{\sqrt{v}} = \text{fixed}. \tag{2.1.16}$$

After defining new variables

$$U = \frac{r}{\alpha'} \quad \text{and} \quad \tilde{x}_i = \frac{x_i}{\alpha} \,, \tag{2.1.17}$$

and taking $r \ll L$ and low-energy limit, the D1-D5 system metric reduces to

$$ds^2 = \alpha' \left[\frac{U}{R^2}(-dt^2 + dx_1^2) + \frac{R^2}{U^2} dU^2 + R^2 d\Omega_3^2 + \sqrt{\frac{N_1}{N_5}} \sqrt{v} \sum_{i=6}^{9} d\tilde{x}_i^2 \right] \,, \tag{2.1.18}$$

which is the $AdS_3 \times S^3 \times T^4$ metric, with $R_{AdS_3} = R_{S^3} = R = g_s \sqrt{N_1 N_5}$, and the volume of T^4 proportional to $\frac{N_1}{N5}$.

Similarly to the discussion in Section 2.1.1, if we look into the bulk by taking the limit $\alpha' \to \infty$, we find a 10-dimensional supergravity in the Minkowski spacetime.

More generally, one can obtain a background supported by a mixture of R-R and NS-NS fluxes. Namely, while the pure R-R flux background comes from the D1-D5 brane system and the pure NS-NS background comes from the NS5-F1 brane system, the mixed-flux background can be obtained by considering the near-horizon limit of combination of NS5-F1 and D1-D5 branes[7] [41]. When the AdS_3 radius is set to 1, the fluxes can be related to the parameters of the background [40] as

$$F^{(3)} = \tilde{q}(\text{Vol}(AdS_3) + \text{Vol}(S^3)) \qquad H^{(3)} = q(\text{Vol}(AdS_3) + \text{Vol}(S^3)) \,, \tag{2.1.19}$$

[6] In general, the fields in the type II string theory action can be grouped into Ramond–Ramond (R–R) and Neveu–Schwarz–Neveu–Schwarz (NS–NS) sectors. By "Pure Ramond-Ramond" we mean that the NS-NS charge associated with $B_{\text{NS-NS}}$ is put to zero.

[7] In fact, NS5-F1 brane system is S-dual of the D1–D5 system.

where $q^2 + \tilde{q}^2 = 1$. Therefore, it is a one-parameter family of supersymmetric backgrounds.

Then, two important parameters can be defined. The mixed-flux coupling constant h [40], [42]

$$h(\lambda, q) = \frac{\tilde{q}\sqrt{\lambda}}{2\pi} + \mathcal{O}(\frac{1}{\sqrt{\lambda}}), \tag{2.1.20}$$

which enters as an overall normalisation of the central charge, and coupling k of the WZW model [42]

$$k = q\sqrt{\lambda}, \quad k \in \mathbb{Z}. \tag{2.1.21}$$

Because $q^2 + \tilde{q}^2 = 1$, we can say that the parameter q controls the relative contribution of R-R and NS-NS fluxes to the AdS_3 background.

Identifying the dual conformal theory to the AdS_3 background is a more difficult problem than in the case of AdS_5. Unlike the $\mathcal{N} = 4$ SYM theory, where we only have two free parameters N and λ, various scalars arise from the open string dynamics on the D1-D5 system which have non-zero expectation values [3].

When $q = 1$, the AdS_3 background is supported by pure NS-NS three-form flux, and the worldsheet sigma model can be described within the RNS or hybrid formalism, where the background can be realised as a level-k supersymmetric WZW model [43], [44] (for $k > 1$). In that case, the string theory is not dual to a specific CFT but rather to a grand-canonical ensemble of superconformal field theories (SCFTs) [45], [46]. For $k = 1$, that is the string theory on $AdS_3 \times S^3 \times T^4$ with (minimal) one unit of NS-NS flux, the spectrum matches the large N limit of the free symmetric product orbifold CFT $\mathrm{Sym}^N(T_4)$[47], [48].

The $q = 0$ point is the pure R-R flux background. It was conjectured long ago that the dual CFT for this setup should be the symmetric product orbifold CFT $\mathrm{Sym}^N(T_4)$ [3] which has a 20-parameter moduli space [49].

The $0 < q < 1$ case is the background supported by a mixture of NS-NS and R-R fluxes. This string theory also has a 20-parameter moduli space [50]. The general argument of [3] suggests that the dual CFT is some 2d $N = (4, 4)$ superconformal field theory (SCFT). The recent work [51] proposes that the string theory at $k = 1$ with a small amount of R-R flux (whose strength is indicated by h) switched on ($h \ll 1$, deforming the theory away from the tensionless point) should be dual to a perturbation of the symmetric orbifold. In this work, anomalous dimensions of w-cycle twisted sector operators were computed for the perturbation of the symmetric orbifold with the operator Φ [8] which potentially allows the comparison of the spectrum of both sides.

2.2 Integrability in $AdS_5 \times S^5$

Although duality is believed to hold beyond the original example [52], both sides are studied mainly within the 't Hooft limit. In this limit, the rank N of the gauge group $U(N)$ is taken $N \to \infty$, keeping $\lambda = Ng^2$ fixed, so only planar diagrams dominate (thus the other name, the planar limit). They correspond to string theory diagrams in the limit where the string coupling constant $g_s = \lambda/4\pi N \to 0$ and string tension $g = \sqrt{\lambda}/2\pi$.

One of the reasons this region is well studied is the integrability of the underlying theories, which significantly facilitates the analysis. Integrability is a property of theories with a large amount of symmetry, making analytical calculations of many physical quantities possible. As will be explained below, both $AdS_5 \times S^5$ string theory and $\mathcal{N} = 4$ SYM are integrable

[8] Φ is the singlet or 'blow-up' mode of the orbifold from $\mathbb{Z}_2$ twisted sector.

in the planar limit, allowing us to perform calculations at a strong coupling and test many consequences of the duality. There is some hope that integrability is present beyond this limit [53], but well-established results exist mostly within this limit.

In this section, we review integrability in $\mathcal{N}=4$ SYM and $\text{AdS}_5/\text{CFT}_4$. For more details, see the general review [31], and [8] for $\mathcal{N}=4$ SYM integrability in particular.

2.2.1 Integrability in $\mathcal{N}=4$ SYM

The Super Yang-Mills (SYM) theory has the symmetry group $\text{PSU}(2,2|4)$ – $\mathcal{N}=4$ superconformal group [54] (see [55] for a review and [56], [57] for a general introduction). The corresponding superalgebra $\mathfrak{psu}(2,2|4)$ has 16 supercharges. Its bosonic subalgebra is $\mathfrak{su}(2,2)\oplus\mathfrak{su}(4)\equiv\mathfrak{so}(4)\oplus\mathfrak{so}(6)$, where the first part is the conformal subalgebra and the second is the R-charge. This is in agreement with the string theory symmetries: the $\text{AdS}_5\times\text{S}^5$ spacetime has an isometry group $\text{SO}(2,4)\times\text{SO}(6)$.

In SYM, all operators belong to some representations of the global symmetry group and are labelled by 6 Casimir operators

$$(\Delta, S_1, S_2, J_1, J_2, J_3)\ , \tag{2.2.1}$$

where J_i are the three angular momenta of $\text{SO}(6)$, S_i are the spins of $\text{SO}(2,4)$ and Δ is the scaling dimension. The latter is also an eigenvalue of the dilatation operator $\mathcal{D}$. In unitary quantum field theories, all operators (apart from identity) have positive scaling dimensions. Operators with the lowest dimension are called *primary*. By applying the elements of the symmetry algebra, we can obtain other operators (descendants). The primary operator and its descendants make up an irreducible representation of $\text{PSU}(2,2|4)$, which is infinite-dimensional. An important subclass of these operators is the chiral primary operators or BPS operators. They commute with half of the supercharges, and one can show that their scaling dimensions are protected from quantum corrections (they cannot have an anomalous dimension). See [58] for a detailed review.

The scaling dimension mentioned above also determines the two-point correlator, the form of which is fixed by the conformal symmetry

$$\langle\mathcal{O}(x)\mathcal{O}'(y)\rangle\propto\frac{1}{|x-y|^{2\Delta}}\ , \tag{2.2.2}$$

where $\Delta=\Delta_0+\gamma(\lambda)$, Δ_0 is the bare (classical) dimension, γ is the anomalous scaling dimension. The anomalous scaling dimension γ can be computed perturbatively ($\lambda\ll 1$) using the Feynman diagrams for these correlators. From (2.2.2), for $\gamma\ll\Delta_0$, one can get

$$\langle\mathcal{O}(x)\mathcal{O}'(y)\rangle\propto\frac{1}{|x-y|^{2\Delta_0}}(1-\gamma\ln\Lambda^2|x-y|^2+...)\,, \tag{2.2.3}$$

where Λ is the cutoff scale. However, at one loop, quantum corrections introduce operator mixing and effectively make the dilatation operator non-diagonal. Since operator mixing preserves the $\text{PSU}(2,2|4)$ symmetry group (and its subgroups), mixing only occurs between operators with the same R- and Lorentz charges and the same bare dimensions. However, sometimes we can group operators into smaller sectors, which are completely closed under the action of dilatation, such as the $\text{SU}(2)$ sector, or only up to one-loop, such as the $\text{SO}(6)$ sector. This property will be useful later.

In general, CFTs are fully determined by the scaling dimensions of the primary operators and three-point correlators. Since SYM is supersymmetric, we should consider primary

operators, which are superconformal primaries and gauge-invariant. All other observables can be computed using the operator product expansion (OPE). The collection of all primary operators O_p, characterised by their scaling dimensions Δ_p and representations ρ_p, is the *spectrum* of the theory. Together with the set of OPE coefficients (determining the form of the three-point functions), they constitute the *CFT data*. The way to construct the gauge-invariant operators is to take products of traces of the fields that transform covariantly under the gauge group

$$\mathcal{O} = \text{tr}[\chi_1 ... \chi_n]\text{tr}[\chi_{n+1} ...] ... \,, \tag{2.2.4}$$

where χ_i's are any covariant fields, and these are then used to compute various correlation functions

$$\langle \mathcal{O}_1 ... \mathcal{O}_n \rangle \,. \tag{2.2.5}$$

In the large N limit, such correlation functions factorise

$$\langle \text{tr}[\chi_1 ... \chi_n]\text{tr}[\chi_{n+1} ...] ... \rangle = \langle \text{tr}[\chi_1 ... \chi_n] \rangle \langle \text{tr}[\chi_{n+1} ...] ... \rangle + O(1/n^2) \,. \tag{2.2.6}$$

Thus, the dimensions of the product of single-trace operators are given by the sum of their dimensions; therefore, it is sufficient to study the spectrum of single-trace operators. Moreover, if we first construct the single-trace primaries, we will not need to consider quantum corrections and can systematically assemble the other operators from these. Therefore, we can understand this class of operators better if the conformal dimensions of single-trace primaries are known.

This is where one can use the integrability of $\mathcal{N} = 4$ SYM. For example, an equivalence between the one-loop spectral problem of single-trace operators in the SO(6) sector (the full set of scalar operators) and an integrable spin chain with nearest-neighbour interactions can be established [59]. In this case, the one-loop dilation operator can be identified with the Hamiltonian of an integrable SO(6) spin-chain. When the operators under the trace are composed of scalar fields with no covariant derivatives, the length L of the spin chain is given by the total bare dimension of the operators under the trace (or the number of such operators).

Later, more evidence was found that integrability extends to higher loops[9], to the full $\mathfrak{psu}(2,2|4)$ symmetry algebra, and arbitrary loop order [61], [62].

A simpler example of this equivalence can be shown for the SU(2) sector, where operators are made of two types of scalar fields, Z and X, which have the charges (1,0,0,1,0,0) and (1,0,0,0,1,0), respectively. The solutions for this spin chain can be obtained using the Bethe equations [63].

As an example, we can label the X field as spin up $(\uparrow)$ and the Z field as spin down $(\downarrow)$

$$\mathcal{O} = \text{tr}(XXZX...) \leftrightarrow |\psi\rangle = |\uparrow\uparrow\downarrow\uparrow ...\rangle \,. \tag{2.2.7}$$

The SU(2) sector has the Hamiltonian

$$H = \frac{\lambda}{8\pi^2} \sum_{l=1}^{L} (1 - P_{l,l+1}) \,, \tag{2.2.8}$$

or in terms of spin operators

$$H = \frac{\lambda}{8\pi^2} \sum_{l=1}^{L} \left(1 - \vec{S}_l \cdot \vec{S}_{l+1}\right) \,, \tag{2.2.9}$$

[9]The interaction range of the corresponding spin chain Hamiltonian increases with loop order [60]

which is identical to the Heisenberg spin chain with L lattice sites. Because of the sign of the $\vec{S}_l \cdot \vec{S}_{l+1}$ term, the spin-chain is ferromagnetic, and the ground state has all spins aligned, with total spin L/2

$$|0\rangle = |\uparrow\uparrow\uparrow ... \uparrow\rangle\,, \tag{2.2.10}$$

which corresponds to a chiral primary $\Psi_L = \mathrm{tr}[X^L]$.

Now, let us consider the state with one spin down and act on it with the Hamiltonian

$$H|\uparrow ... \overset{l}{\uparrow\downarrow}\uparrow ... \uparrow\rangle = \frac{\lambda}{8\pi^2}\left(2|\uparrow ... \overset{l}{\uparrow\downarrow}\uparrow ... \uparrow\rangle - |\uparrow ... \overset{l-1}{\downarrow\uparrow}\uparrow ... \uparrow\rangle - |\uparrow ... \overset{l+1}{\uparrow\downarrow} ... \uparrow\rangle\right). \tag{2.2.11}$$

From this, one can guess the form of the eigenstate

$$|p\rangle = \frac{1}{\sqrt{L}}\sum_{l=1}^{L} e^{ipl}|\uparrow ... \overset{l}{\uparrow\downarrow}\uparrow ... \uparrow\rangle\,, \tag{2.2.12}$$

which turns out to have a nice energy eigenvalue

$$\epsilon(p) = \frac{\lambda}{2\pi^2}\sin^2\frac{p}{2}\,. \tag{2.2.13}$$

With a condition on invariance under shifts $l \to l + L$

$$e^{ipL} = 1 \Rightarrow p = \frac{2\pi n}{L}\,. \tag{2.2.14}$$

This state is called a *single-magnon state*. Because we used a trace operation in the construction of operators, we also demand the invariance of this spin chain under $l \to l + 1$ shift, which leaves only the physical but trivial value of the momentum $p = 0$. The resulting state is a descendant of a vacuum, given by $\mathrm{tr}[X^L]$, and is in a BPS multiplet which does not receive a quantum correction to its bare dimension.

The next state that can be considered has two spins down, and it is the first nontrivial state. It is not a descendant of the vacuum and thus may have a nontrivial anomalous dimension. To construct it, we first need to take an infinitely long chain $L \to \infty$. This allows us to separate two asymptotically separated magnon states and consider their scattering

$$|p_1, p_2\rangle = \sum_{l_1<l_2} e^{ip_1l_1+ip_2l_2}|... \overset{l_1}{\downarrow} ... \overset{l_2}{\downarrow} ...\rangle + e^{i\phi}\sum_{l_1>l_2} e^{ip_1l_1+ip_2l_2}|... \overset{l_2}{\downarrow} ... \overset{l_1}{\downarrow} ...\rangle\,, \tag{2.2.15}$$

where we assume $p_1 > p_2$. The physical interpretation is that one magnon overtakes another with the phase $e^{i\phi}$, which is the S-matrix for scattering. For $|p_1, p_2\rangle$ to be an eigenstate with the eigenvalue being the sum of the energies of the two magnons, we need to impose an additional constraint

$$S_{12} = e^{i\phi} = -\frac{e^{ip_1+ip_2} - 2e^{ip_2} + 1}{e^{ip_1+ip_2} - 2e^{ip_1} + 1}\,. \tag{2.2.16}$$

When we make the spin chain of finite length L again, we should impose both the periodicity $l \to l+L$ and trace $l \to l+1$ conditions. The latter restricts the momenta to $p_1+p_2 = 0$. When applying the periodicity condition, we should be able to circulate an individual magnon around the chain and through another one, which gives an additional phase $e^{i\phi}$. For the state to be invariant, we demand

$$e^{ip_1L}S_{12} = 1\,. \tag{2.2.17}$$

Together with $p_1 = -p_2$ condition, we find that $p_1 = 2\pi n/(L-1)$ and possible eigenvalues for this state are

$$\gamma = \frac{\lambda}{\pi^2} \sin^2 \frac{\pi n}{L-1} \,. \tag{2.2.18}$$

However, the spin chain we considered corresponds to the dilatation operator for SYM, such that the spin-chain energy eigenvalues correspond to the anomalous dimension of single-trace operators.

It is possible to solve this problem for an arbitrary number of spin-down states

$$|p_1 p_2 ... p_l\rangle = \sum_{l_1<l_2<...<l_M} e^{ip_1 l_1 + ip_2 l_2 + ... + ip_M l_M} |... \overset{l_1}{\downarrow} ... \overset{l_2}{\downarrow} ... \cdots \overset{l_M}{\downarrow} ...\rangle + ... \,, \tag{2.2.19}$$

with $p_1 > p_2 > ... > p_M$. The last set of dots represents the other possible orderings of magnons with appropriate phase factors. It is convenient to define the rapidity variable

$$e^{ip_k} = \frac{u_k + i/2}{u_k - i/2} \,. \tag{2.2.20}$$

Then one can find a simple form for the S-matrix

$$S_{jk} = \frac{u_j - u_k - i}{u_j - u_k + i} \,, \tag{2.2.21}$$

and define the dispersion relation for individual magnons

$$\epsilon(u) = \frac{\lambda}{8\pi^2} \frac{1}{u^2 + 1/4} \,. \tag{2.2.22}$$

By putting the magnons on a circle of length L, we find the quantisation condition

$$\left(\frac{u_j + i/2}{u_j - i/2} \right)^L = \prod_{k \neq j}^{M} \frac{u_j - u_k + i}{u_j - u_k - i} \,, \tag{2.2.23}$$

and the trace (zero momentum) condition for the total momentum

$$\prod_{j=1}^{M} \frac{u_j + i/2}{u_j - i/2} = 1 \,. \tag{2.2.24}$$

Together, this gives the energy eigenvalue for this state in the form

$$\gamma = \sum_{k=1}^{M} \epsilon(u_k) \,, \tag{2.2.25}$$

which is also the 1-loop anomalous dimension of a single-trace operator composed of fields Z and X.

The $AdS_5 \times S^5$ string theory, which is dual to the SYM theory described above, is also integrable. However, the approach taken to demonstrate this is different, with the worldsheet S-matrix serving as a key ingredient.

2.2.2 Integrability in $\mathbf{AdS}_5 \times \mathbf{S}^5$

The S-matrix is a powerful tool for studying integrable theories. In particular, one can use the factorised scattering theory approach to construct an S-matrix and then use integrability techniques to study the spectrum of the theory. We illustrate this approach using the example of $\text{AdS}_5 \times \text{S}^5$ strings [64].

String theory in $\text{AdS}_5 \times \text{S}^5$ is a maximally supersymmetric type IIB string theory with 32 supercharges. As a dual to $\mathcal{N} = 4$ SYM, it also has a group of isometries $\text{PSU}(2,2|4)$.

This can be seen explicitly in the Metsaev-Tseytlin formulation [65], where it is written as a sigma-model on the coset superspace

$$\frac{\text{PSU}(2,2|4)}{\text{SO}(1,4) \times \text{SO}(5)} \,. \tag{2.2.26}$$

Because the bosonic part of $\text{PSU}(2,2|4)$ is $\text{SO}(2,4) \times \text{SO}(6)$, we can write that

$$\frac{\text{SO}(2,4)}{\text{SO}(1,4)} \times \frac{\text{SO}(6)}{\text{SO}(5)} \simeq \text{AdS}_5 \times \text{S}^5 \,. \tag{2.2.27}$$

It is known that symmetric space coset sigma-models are classically integrable [66], and so is the string theory in $\text{AdS}_5 \times \text{S}^5$ [67]. The key property that allows integrability is $\mathbb{Z}_4$ grading of the superalgebra $\mathfrak{psu}(2,2|4)$ [68].

While it is known how to rigorously formulate the integrability condition at the classical level, this is not the case for quantum systems. Therefore, to make progress, one assumes quantum integrability and investigates the consequences.

The first indications that integrability is present at the quantum level were found on the gauge theory side in terms of a spin chain [59]. However, it was later realised that an S-matrix can be a very useful tool to study that [64]: integrability in the large-N AdS/CFT system is equivalent to factorised scattering of the corresponding hidden elementary excitations. The S-matrix was found on the CFT side [62], [64], [69], and later on the string side [70] using the Zamolodchikov-Faddeev (ZF) algebra [71][10].

The correspondence was verified at the level of solitonic solutions in the limit of large spins J for spinning strings [72], yielding the first result interpolating between the weak and strong-coupling regimes. The analysis relied on the all-loop asymptotic Bethe ansatz (ABA) framework [73], [74].

One begins with the Green-Schwarz action and quantises it in a uniform light-cone gauge [75], [76] which results in an integrable Non-linear Sigma Model (NLSM). By analysing the light-cone Hamiltonian, one can find that although the full superisometry algebra of the string sigma-model is $\mathfrak{psu}(2,2|4)$, only the $\mathfrak{psu}(2|2) \otimes \mathfrak{psu}(2|2)$ subalgebra leaves the Hamiltonian invariant [77]. The direct quantisation of the NLSM action is beyond current methods and is performed perturbatively by expanding around a given classical solution in powers of the effective string tension $\sqrt{\lambda}$. This theory can be considered as a theory on a cylinder of finite circumference proportional to the light-cone momentum with an infinite time. Working with it is still hard, but one can consider the decompactification limit, where the light-cone momentum P_+ goes to infinity while keeping the string tension fixed, so that the cylinder becomes a plane. When it is done, one can apply factorised scattering theory and the matrix structure of this S-matrix and dispersion relation [69] are uniquely fixed by

[10] Under the assumption of 2-particle irreducible representations having Hopf algebra structure.

the centrally extended $\mathfrak{psu}(2|2) \oplus \mathfrak{psu}(2|2)$[11] symmetry algebra, the Yang–Baxter equation and the generalised physical unitarity condition [69], [70], [78] up to an overall scalar function of particle momenta (it is called dressing factor and will be discussed later). Finally, one can use the resulting S-matrix and techniques such as the Bethe Ansatz to determine the worldsheet spectrum.

To work with non-decompactified theory, one needs to consider finite-size corrections. They are related to the wrapping interactions arising from the dynamics of virtual particles (in finite volume). In particular, the spectrum of the theory on a cylinder is different from that in the decompactification limit because of the wrapping effects [79]. In the $AdS_5 \times S^5$ theory , they are exponentially suppressed, and one can use the Lüscher approach for wrapping corrections to account for them [80], [81]. In the $AdS_3 \times S^3$ case, Lüscher are not exponentially suppressed due to the presence of massless particles in the spectrum [82]. See [83] for a more detailed review.

Another way to work with non-compactified theory is to perform a double Wick rotation in the NLSM theory. Then, the problem of determining the spectrum of this QFT (which is integrable, as mentioned before) is reduced to the study of the mirror theory obtained from the original NLSM. By doing so, we trade the finite volume (and infinite time direction) for a finite temperature (and infinite spatial direction). The S-matrix of the new theory is integrable because it is obtained from the old one by analytic continuation. Then TBA techniques can be used to determine the spectrum [84].

Finally, the state-of-the-art technique for obtaining a full non-perturbative spectrum is *Quantum Spectral Curve* (QSC), which generalises integrability-based approaches like Bethe Ansatz, TBA and Y-system [85], [86]. In this approach, the spectrum is encoded into a set of analytic functions called Q-functions, which satisfy a finite number of certain functional relations and analyticity conditions.

2.2.3 Symmetry Algebra

We begin with an overview of the symmetries of $AdS_5 \times S^5$ strings, which will allow us to derive the dispersion relation and will be useful in the discussion of the S-matrix. A more detailed discussion can be found in [87].

The full symmetry algebra of type IIB superstrings in the $AdS_5 \times S^5$ background in $\mathfrak{psu}(2,2|4)$ however, the use of light-cone gauge-fixing breaks it to $\mathfrak{psu}(2|2) \otimes \mathfrak{psu}(2|2)$ [77]. The letter $\mathfrak{p}$ in $\mathfrak{psu}$ stands for projected which removes $\mathfrak{u}(1)$ factor from $\mathfrak{su}(2|2)$ making it a simple algebra. In what follows, we will work with just $\mathfrak{su}(2|2) \otimes \mathfrak{su}(2|2)$ as it does not change the results for physics.

In a light-cone gauge, string theory becomes an NLSM living on a cylinder of circumference P_+ (the light-cone momentum). Additionally, one should impose the Virasoro level-matching condition, which restricts the allowed physical states: the total world-sheet momentum p_{ws} carried by a state must vanish.

To introduce the concept of world-sheet excitations, one needs to relax the level-matching condition (i.e. consider "off-shell" theory). Then, to define the asymptotic states and the world-sheet S-matrix, one unwraps the cylinder of circumference P_+ to the plane by taking the limit $P_+ \to \infty$.

If the level-matching condition holds, the $\mathfrak{su}(2|2) \oplus \mathfrak{su}(2|2)$ algebra is spanned by the usual generators which commute with the worldsheet Hamiltonian. Giving up the level-matching

[11] Generally, there are several ways to construct a central extension. Here, we mean a certain central extension of algebra, where each of the $\mathfrak{psu}(2|2)$ factors is centrally extended, which is explained in the next section.

condition leads to a modification of the $\mathfrak{su}(2|2) \oplus \mathfrak{su}(2|2)$ algebra: the algebra receives two central charges in addition to the Hamiltonian, which are functions of the momentum carried by the one-particle excitations[12][13] .

We now explain what we mean by central extension precisely. The symmetry algebra we consider is equivalent to two copies of the centrally extended $\mathfrak{su}(2|2)$ algebra, with both copies sharing the same central element, which is a world-sheet light-cone Hamiltonian. First, we consider the representation of a single $\mathfrak{su}(2|2)$, and then construct the desired product representation.

The centrally extended $\mathfrak{su}(2|2)$ algebra, which we will denote $\mathfrak{su}(2|2)_{c.e.}$, consists of the rotation generators $\mathbf{L}_{ab}$, $\mathbf{R}_{\alpha\beta}$ of the two $\mathfrak{su}(2)$ bosonic subalgebras, the supersymmetry generators $\mathbf{Q}_{\alpha}^{\ a}$, $\overline{\mathbf{Q}}_{a}^{\ \alpha}$ and three central elements $\mathbf{H}$, $\mathbf{C}$ and $\overline{\mathbf{C}}$. Latin indices a, b take values $\{1, 2\}$, while Greek indices α, β take values $\{3, 4\}$. The action of bosonic generators reads

$$\begin{aligned} \left[\mathbf{L}_{a}^{\ b}, \mathbf{J}_{c}\right] &= \delta_{c}^{b}\mathbf{J}_{a} - \frac{1}{2}\delta_{a}^{b}\mathbf{J}_{c}\,, \quad \left[\mathbf{R}_{\alpha}^{\ \beta}, \mathbf{J}_{\gamma}\right] = \delta_{\gamma}^{\beta}\mathbf{J}_{\alpha} - \frac{1}{2}\delta_{\alpha}^{\beta}\mathbf{J}_{\gamma}\,, \\ \left[\mathbf{L}_{a}^{\ b}, \mathbf{J}^{c}\right] &= -\delta_{a}^{c}\mathbf{J}^{b} + \frac{1}{2}\delta_{a}^{b}\mathbf{J}^{c}\,, \quad \left[\mathbf{R}_{\alpha}^{\ \beta}, \mathbf{J}^{\gamma}\right] = -\delta_{\alpha}^{\gamma}\mathbf{J}^{\beta} + \frac{1}{2}\delta_{\alpha}^{\beta}\mathbf{J}^{\gamma}\,, \end{aligned} \tag{2.2.28}$$

where $\mathbf{J}_{*}$ ($\mathbf{J}^{*}$) is any generator with the Latin or Greek lower or upper index, e.g. $\overline{\mathbf{Q}}_{a}^{\ \alpha}$ would be acted according to the very first commutation relation. The centrally extended algebra differs from the non-centrally-extended one by fermionic commutators [69], [70]

$$\begin{aligned} &\{\mathbf{Q}_{\alpha a}, \overline{\mathbf{Q}}_{b\beta}\} = \epsilon_{ba}\mathbf{R}_{\alpha\beta} + \epsilon_{\alpha\beta}\mathbf{L}_{ba} + \epsilon^{\beta\alpha}\epsilon_{ba}\mathbf{H}\,, \\ &\{\mathbf{Q}_{\alpha}^{a}, \mathbf{Q}_{\beta}^{b}\} = \epsilon_{\alpha\beta}\epsilon^{ab}\mathbf{C}, \qquad \{\overline{\mathbf{Q}}_{a}^{\alpha}, \overline{\mathbf{Q}}_{b}^{\beta}\} = \epsilon_{ab}\epsilon^{\alpha\beta}\overline{\mathbf{C}}, \end{aligned} \tag{2.2.29}$$

The central element $\mathbf{H}$ is hermitian and is identified with the world-sheet light-cone Hamiltonian, and $\mathbf{C}$ and $\overline{\mathbf{C}}$ are two additional central charges.

It is possible to write the conserved currents and charges in terms of bosonic and fermionic fields of the string sigma-model action in the light-cone gauge and determine the above supersymmetry algebra commutation relations by studying the Poisson bracket of the Noether charges of the string sigma-model in the light-cone gauge, which was done in [77]. Additionally, one can find that $\mathbf{C}$ can be expressed through the world-sheet momentum $\mathbf{P}$ as follows

$$\mathbf{C} = +\frac{ig}{2}\zeta(e^{+i\mathbf{P}} - 1)\,, \qquad \overline{\mathbf{C}} = -\frac{ig}{2}\bar{\zeta}(e^{-i\mathbf{P}} - 1)\,, \tag{2.2.30}$$

where ζ is a phase that can be an arbitrary function of the central elements. This is the reflection of the fact that the algebra has $\mathrm{U}(1)$ automorphism

$$\mathbf{Q} \to e^{i\xi}\mathbf{Q}, \qquad \mathbf{C} \to e^{2i\xi}\mathbf{C}\,, \tag{2.2.31}$$

which can also be referred to as a choice frame (choice of basis).

Before moving further, we need to introduce a representation space where the $\mathfrak{su}(2|2)_{c.e.}$ symmetry algebra acts. We can use a basis of the four-dimensional fundamental representation, which is constructed from fundamental representations of the two $\mathfrak{su}(2)$ algebras [70]

$$|e_{M}\rangle = \begin{cases} |e_{a}\rangle\ , \\ |e_{\alpha}\rangle\ , \end{cases} \tag{2.2.32}$$

[12] This was first argued by Beisert in [69] on the gauge theory side.

[13] In [77], the enlargement of the $\mathfrak{su}(2|2) \oplus \mathfrak{su}(2|2)$ algebra by a common central element (central charge) was shown by expanding all supersymmetry generators in powers of fields (equivalently in the inverse string tension $2\pi/\sqrt{\lambda}$) for strings in $\mathrm{AdS}_5 \times \mathrm{S}^5$.

with the following action of bosonic generators

$$\begin{aligned} \mathbf{L}_a^{\ b}\,|e_c\rangle &= \delta_c^b\,|e_a\rangle - \frac{1}{2}\delta_a^b\,|e_c\rangle \qquad & \mathbf{R}_\alpha^{\ \beta}\,|e_a\rangle &= 0 \\ \mathbf{L}_a^{\ b}\,|e_\alpha\rangle &= 0 & \mathbf{R}_\alpha^{\ \beta}\,|e_\gamma\rangle &= \delta_\gamma^\beta\,|e_\alpha\rangle - \frac{1}{2}\delta_\alpha^\beta\,|e_\gamma\rangle\ , \end{aligned} \tag{2.2.33}$$

and supersymmetry generators

$$\begin{aligned} \mathbf{Q}_\alpha^{\ a}\,|e_b\rangle &= a\delta_b^a\,|e_\alpha\rangle \qquad & \overline{\mathbf{Q}}_a^{\ \alpha}\,|e_b\rangle &= c\epsilon_{ab}\epsilon^{\alpha\beta}\,|e_\beta\rangle \\ \mathbf{Q}_\alpha^{\ a}\,|e_\beta\rangle &= b\epsilon_{\alpha\beta}\epsilon^{ab}\,|e_b\rangle & \overline{\mathbf{Q}}_a^{\ \alpha}\,|e_\beta\rangle &= d\delta_\beta^\alpha\,|e_a\rangle\ , \end{aligned} \tag{2.2.34}$$

where coefficients a, b, c and d are the parameters of the representation.

This allows us to find the values of the central charges expressed through the representation parameters a, b, c, d in the following way

$$\mathbf{H}\,|e_M\rangle = (ad+bc)\,|e_M\rangle\,, \quad \mathbf{C}\,|e_M\rangle = ab\,|e_M\rangle\,, \quad \overline{\mathbf{C}}\,|e_M\rangle = cd\,|e_M\rangle\ . \tag{2.2.35}$$

For generic values of the parameters a, b, c, d the representation is non-unitary. To obtain a unitary representation, one should impose the conditions $d=\bar{a}$ and $c=\bar{b}$.

One can check that the relations (2.2.28) and (2.2.29) are consistent with this representation if

$$ad-bc=1\,, \tag{2.2.36}$$

which implies

$$\mathbf{H}^2 - 4\mathbf{C}\overline{\mathbf{C}} = 1\,. \tag{2.2.37}$$

That is the *shortening condition* that defines the short multiplet of $\mathfrak{su}(2|2)_{c.e.}$ [88].

By plugging (2.2.30) into (2.2.37), one can derive the dispersion relation

$$H = \sqrt{1+4g^2\sin\left(\frac{p}{2}\right)^2}\,, \tag{2.2.38}$$

which has a square-root form. A way to make it a non-square root type is to introduce a parameterisation in terms of Zhukovsky variables $x^\pm$ [70]

$$a=\sqrt{h}\eta, \quad b=\sqrt{h}\frac{i\zeta}{\eta}\left(\frac{x^+}{x^-}-1\right), \quad c=-\sqrt{h}\frac{\eta}{\zeta x^+} \quad d=\sqrt{h}\frac{x^+}{i\eta}\left(1-\frac{x^-}{x^+}\right)\,, \tag{2.2.39}$$

where

$$x^+ + \frac{1}{x^+} - x^- - \frac{1}{x^-} = \frac{i}{h}, \quad \frac{x^+}{x^-} = e^{ip}, \quad \eta = e^{ip/4}\sqrt{i(x^- - x^+)}\,, \tag{2.2.40}$$

so that we get

$$H = \frac{ih}{2}\left(x^- - \frac{1}{x^-} - x^+ + \frac{1}{x^+}\right)\,. \tag{2.2.41}$$

2.2.4 The Exact S-matrix

The symmetry algebra of the Hamiltonian implies certain restrictions on the S-matrix. Let us denote by $\mathbf{J}^a$ the operators which generate the symmetry algebra $\mathcal{A}$

$$[\mathbf{J}^a, \mathbf{H}] = 0, \quad a=1,...,\dim\mathcal{A}\,, \tag{2.2.42}$$

and let them act on some representation $\mathcal{V}$. In addition to $\mathbf{H}$, the symmetry generators commute with the total momentum $\mathbf{P}$ and number of particles $\mathbf{N}$ operators, and all the higher conserved charges. The Hilbert space is created by the Zamolodchikov–Faddeev (ZF) operators and carries a linear representation of $\mathcal{A}$. Later, we will define multiparticle states via the tensor product of $\mathcal{V}$'s.

First, we introduce a vacuum state $|\Omega\rangle$ which is invariant under the algebra. Then, introduce a ZF creation operator $A_i^\dagger(p)$ which creates a particle out of the vacuum with momentum p and flavour denoted by i.

$$A_i^\dagger(p)\,|\Omega\rangle = |p\rangle_i \tag{2.2.43}$$

The hermitian conjugate $A_i(p)$ is the ZF annihilation operator such that

$$A_i(p)\,|\Omega\rangle = 0\,. \tag{2.2.44}$$

The non-Abelian symmetry algebra $\mathcal{A}$ acting on the spectrum of the Hamiltonian implies a non-trivial constraint on the scattering matrix

$$\mathbf{J}^a \cdot A_i^\dagger(p_1)A_j^\dagger(p_2)|\Omega\rangle = S_{ij}^{kl}(p_1,p_2)\mathbf{J}^a \cdot A_l^\dagger(p_2)A_k^\dagger(p_1)|\Omega\rangle\,, \tag{2.2.45}$$

and the invariance condition can be written as

$$S_{12}(p_1,p_2)\mathbf{J}_{12}^a(p_1,p_2) = \mathbf{J}_{21}^a(p_2,p_1)S_{12}(p_1,p_2)\,, \tag{2.2.46}$$

where $\mathbf{J}_{12}^a$ should act on a tensor product of two one-particle states. If $\mathcal{A}$ is a simple Lie superalgebra with (momentum-independent) structure constants in the one-particle representation, then the two-particle states can be identified with the graded tensor product of two one-particle states. The two-particle symmetry generators are given by the graded tensor product[14]

$$\mathbf{J}_{12}^a = \mathbf{J}^a \otimes \mathbb{1} + \mathbb{1}^g(\mathbb{1} \otimes \mathbf{J}^a)\mathbb{1}^g\,, \tag{2.2.47}$$

where $\mathbb{1}^g$ is the graded identity defined to account for the correct statistics of operators

$$\mathbb{1}^g = (-1)^{\epsilon_i\epsilon_j}E_i^i \otimes E_j^j\,, \tag{2.2.48}$$

and ϵ_i is parity. It is equal to zero or one, depending on whether the value of i corresponds to a bosonic or fermionic state, respectively.

However, the symmetry algebra $\mathcal{A}$ of the light-cone string sigma model is not of this type, as it has a non-trivial centre. Now, any representation of $\mathbf{J}$ is parameterised by the momentum of the particle and the values of the Lie algebra central elements. We denote the generators of $\mathbf{J}$ in some representation $\mathcal{V}$ as $\mathbf{J}_a(p;c)$, where c denotes the values of the central elements. When we identify $\mathcal{V}$ with a one-particle representation in the Fock space, we must prescribe a fixed value for c, as only the one-particle representation should be characterised by the particle momentum. In general, we can modify (2.2.47) for the operators in the two-particle representation

$$\mathbf{J}_{12}^a = \mathbf{J}^a(p_1;c_1) \otimes \mathbb{1} + \mathbb{1}^g(\mathbb{1} \otimes \mathbf{J}^a(p_2;c_2))\mathbb{1}^g \tag{2.2.49}$$

with arbitrary c_1 and c_2 depending on the particle momenta p_1 and p_2.

When we consider how the central charges act on two-particle states, we find that the result does not give the correct value of the central charge if we consider a trivial coproduct

[14]Two terms $\mathbb{1}$ are included because the tensor product sign $\otimes$ we are using denotes an ordinary tensor product, while we need to construct a graded tensor product.

and set $\zeta = 1$ everywhere. It is expected that the central charge of the tensor product of fundamental representations should be equal to the sum of their charges

$$\begin{aligned}\mathbf{C}|A^\dagger_{M_1}(p_1)A^\dagger_{M_2}(p_2)\rangle &= \frac{ig}{2}(e^{i\mathbf{P}}-1)|A^\dagger_{M_1}(p_1)A^\dagger_{M_2}(p_2)\rangle \\ &= \frac{ig}{2}(e^{i(p_1+p_2)}-1)|A^\dagger_{M_1}(p_1)A^\dagger_{M_2}(p_2)\rangle\,. \quad (2.2.50)\end{aligned}$$

However, if we consider the action of the central charge on the tensor product of the fundamental representation explicitly

$$\begin{aligned}\mathbf{C}_{12} &= c_1\mathbb{1}\otimes\mathbb{1} + \mathbb{1}^g(\mathbb{1}\otimes c_2\mathbb{1})\mathbb{1}^g \\ &= (c_1+c_2)\mathbb{1}\otimes\mathbb{1} = \frac{ig}{2}((e^{ip_1}-1)+(e^{ip_2}-1))\mathbb{1}\otimes\mathbb{1}\,, \quad (2.2.51)\end{aligned}$$

which does not agree with (2.2.50).

To fix this, we should not fix $\zeta = 1$ everywhere but rather choose [70]

$$\{\zeta_1 = 1, \zeta_2 = e^{ip_1}\} \qquad \text{or} \qquad \{\zeta_1 = e^{ip_2}, \zeta_2 = 1\}\,, \quad (2.2.52)$$

which leads to the modification of (2.2.51)

$$\begin{aligned}\mathbf{C}_{12} &= c_1\mathbb{1}\otimes\mathbb{1} + \mathbb{1}^g(\mathbb{1}\otimes c_2\mathbb{1})\mathbb{1}^g = (c_1+c_2)\mathbb{1}\otimes\mathbb{1} \\ &= \frac{ig}{2}(\zeta_1(e^{ip_1}-1)+\zeta_2(e^{ip_2}-1))\mathbb{1}\otimes\mathbb{1} = \frac{ig}{2}(e^{ip_1+p_2}-1)\mathbb{1}\otimes\mathbb{1}\,. \quad (2.2.53)\end{aligned}$$

This means that we must ifer the value of c_1 and c_2 from these solutions when considering the tensor product of one-particle states.

Now, we can modify (2.2.45) for the case of algebra with a non-trivial centre and coproduct [70]. The S-matrix commutation relation takes the following form with bosonic

$$S_{12}(p_1,p_2)\big(\mathbf{J}\otimes\mathbb{1}+\mathbb{1}\otimes\mathbf{J}\big) = \big(\mathbf{J}\otimes\mathbb{1}+\mathbb{1}\otimes\mathbf{J}\big)S_{12}(p_1,p_2)\,, \quad (2.2.54)$$

and fermionic generators

$$\begin{aligned}&S_{12}(p_1,p_2)\big(\mathbf{J}(p_1;1)\otimes\mathbb{1}+\Sigma\otimes J(p_2;e^{ip_1})\big) \\ &\qquad = \big(\mathbf{J}(p_1;e^{ip_2})\otimes\Sigma+\mathbb{1}\otimes\mathbf{J}(p_2;1)\big)S_{12}(p_1,p_2)\,, \quad (2.2.55)\end{aligned}$$

where Σ is the grading matrix $\Sigma = (-1)^{\epsilon_i}E^i_i$. These constraints on the S-matrix are sufficient to bootstrap its matrix form up to an overall scalar function, the dressing factor, which will be discussed in the next section.

2.2.5 Dressing Factors

The S-matrix is determined up to an overall scalar function $\sigma(p1,p2)$, the so-called dressing factor [89], which is not constrained by the conditions imposed by the symmetry algebra, the Yang–Baxter equation, and the generalised physical unitarity condition. It is anticipated that further physical requirements will allow for the complete determination of this factor.

In Lorentz-invariant theories, this is done using Lorentz invariance together with crossing symmetry (exchanging particles with antiparticles) [71]. The light-cone gauge-fixed

sigma model is not Lorentz invariant; however, there is an analogue of crossing symmetry which can be motivated by considering the process of scattering particles and the same process with a particle changed to an antiparticle [90].

In Lorentz invariant theories, one can use the argument of [88] to construct a singlet state

$$|1\rangle \equiv \sum_h |\{h,\theta\},\{h,\theta-i\}\rangle\,, \tag{2.2.56}$$

where h denotes the theory degrees of freedom, θ is the rapidity variable, and $\theta - i$ denotes to particle to anti-particle transformed state. Therefore, the total momentum, energy, and flavour should be zero. The scattering of a physical particle through such a spurious state is trivial. This gives us an additional constraint on an S-matrix

$$\hat{\mathrm{S}}(\theta)\hat{\mathrm{S}}(\theta-i) = \mathbb{1}\,, \tag{2.2.57}$$

which can be turned into the condition on the scalar prefactor of the S-matrix

$$\sigma(\theta)\sigma(\theta-i) = f(\theta)\,, \tag{2.2.58}$$

with some function $f(\theta)$ determined from the matrix elements of the S-matrix. Although the string theory in the light-cone gauge is not relativistic, the same reasoning can be applied to the derivation of crossing (as done in [69]).

An equivalent approach is to employ an underlying Hopf algebra structure to derive the crossing condition for the nonrelativistic S-matrices relevant to integrable holography, which was first suggested in [90]. To promote the symmetry algebra $\mathcal{A}$ to a Hopf algebra, we need to introduce a coproduct $\Delta : \mathcal{A} \to \mathcal{A}\otimes\mathcal{A}$ and an antipode $\mathbb{S} : \mathcal{A} \to \mathcal{A}$[15].

Concrete realisation of the Hopf algebra coproduct action on the symmetry algebra $\mathcal{A}$ have the following form [91]

$$\begin{aligned}
\Delta(\mathbf{J}) &= \mathbf{J}\hat{\otimes}\mathbb{1} + \mathbb{1}\hat{\otimes}\mathbf{J}\,,\\
\Delta(\mathbf{Q}^a_\alpha) &= \mathbf{Q}^a_\alpha\hat{\otimes}\mathbb{1} + e^{\frac{i}{2}\mathbf{P}}\hat{\otimes}\mathbf{Q}^a_\alpha\,,\\
\Delta(\overline{\mathbf{Q}}^a_\alpha) &= \overline{\mathbf{Q}}^a_\alpha\hat{\otimes}\mathbb{1} + e^{-\frac{i}{2}\mathbf{P}}\hat{\otimes}\overline{\mathbf{Q}}^a_\alpha\,,
\end{aligned} \tag{2.2.59}$$

where $\mathbf{J}$ denotes any even generator[16]. The sign $\hat{\otimes}$ is the graded tensor product having property that for any $a,b,c,d \in \mathcal{A}$

$$(a\hat{\otimes}b)(c\hat{\otimes}d) = (-1)^{\epsilon_b\epsilon_c}(ac\hat{\otimes}bd)\,, \tag{2.2.61}$$

where $\epsilon_a = 0$ if a is even, and $\epsilon_a = -1$ if a is odd. A very useful property of the construction in (2.2.59) is that it preserves the structure of the algebra in (2.2.28). One can notice the structural similarity of (2.2.59) to what we have seen before in (2.2.49). Additionally, there is a factor $e^{-\frac{i}{2}\mathbf{P}}$ which plays a similar role as $e^{i\xi}$ in (2.2.52).

The antipode of a Hopf algebra is analogous to charge conjugation and has a physical interpretation of a particle-to-antiparticle transformation realised as

$$\mathbb{S}(J) = -J, \quad \mathbb{S}(Q_\alpha{}^a) = -e^{-i/2\mathbf{P}}Q_\alpha{}^a, \quad \mathbb{S}(\overline{\mathbf{Q}}_a{}^\alpha) = -e^{i/2\mathbf{P}}\overline{\mathbf{Q}}_a{}^\alpha\,. \tag{2.2.62}$$

[15]Plus some additional structures associated with a coalgebra and a Hopf algebra. See [91] for more details.

[16]There is a caveat that these relations hold for generators on the two-particle states *in states*. For the *out states*, we need to use

$$\Delta^{op} \equiv P^g\Delta\,, \tag{2.2.60}$$

where P^g is the graded permutation operator.

These can be determined using the consistency relations for the Hopf Algebra (see [87], [91]).

It is possible to reformulate the action of the antipode on algebra elements in terms of the transformation of specific representations $\mathcal{V} \otimes \mathcal{V}$ by introducing the charge conjugation matrix C.

$$\pi(\mathbb{S}(\mathcal{A})) = C^{-1}\bar{\pi}(\mathcal{A})^{str}C\,, \tag{2.2.63}$$

where $\pi/\bar{\pi}$ is the particle/antiparticle representation of $\mathcal{A}$ (s.t. $\pi(\mathcal{A}) = \mathcal{A}$), and str denotes the supertransposition.

Finally, for particle-to-antiparticle transformation to be a symmetry of S-matrix, one can use the Hopf-algebraic conditions for the R-matrix

$$(\mathbb{S} \otimes \mathbb{1})R = R^{-1}, \quad (\mathbb{1} \otimes \mathbb{S}^{-1})R = R^{-1}\,. \tag{2.2.64}$$

These can be rewritten for the S-matrix (found in [90]) so the following condition need to be satisfied

$$\begin{aligned}(C \otimes \mathbb{1})\mathsf{S}_{12}^{t_1}(\bar{p}_1, p_2)(C^{-1} \otimes \mathbb{1})\mathsf{S}_{12}(p_1, p_2) = \mathbb{1} \otimes \mathbb{1}\,,\\ (\mathbb{1} \otimes C)\mathsf{S}_{12}^{t_2}(p_1, \bar{p}_2)(\mathbb{1} \otimes C^{-1})\mathsf{S}_{12}(p_1, p_2) = \mathbb{1} \otimes \mathbb{1}\,,\end{aligned} \tag{2.2.65}$$

where the bar over p_1 indicates that the first particle should be in the anti-particle representation. One can interpret the first/second equation as one in which we replace the first/second particle by an antiparticle by performing the crossing transformation. Under this transformation, both the Hamiltonian and the momentum change their sign, which can be considered as the positive energy branch of the dispersion relation (2.2.38) transforming into a negative one.

By looking at the parametrisation of the dispersion relation (2.2.41) and momenta (2.2.40) in terms of Zhukowski variables, we can notice how Zhukowski variables should change under crossing transformation

$$x^{\pm}_{anti-particle} = \frac{1}{x^{\pm}_{particle}}\,. \tag{2.2.66}$$

However, to correctly implement the action of crossing on the S-matrix, we need to consider one more subtle point. The S-matrix has branches corresponding to the branches of the dispersion relation and thus should be treated as a function of both the particle momenta and the particle energies $S(p_1, H_1; p_2, H_2)$. Alternatively, it can be parametrised as a function of Zhukovsky variables $S(x_1^+, x_1^-; x_2^+, x_2^-)$. The subtlety is that the crossing transformation should relate the values of the S-matrix on the different branches, so we need to specify the path γ under which $x^{\pm} = \frac{1}{x^{\pm}}$ to properly define the crossing transformation. We will skip the details of how to choose this path for the AdS_5 case, but will consider AdS_3 in more detail in the following.

Using the parameterisation of S-matrix elements and normalisations choice of [73], it is possible rewrite (2.2.65) as a condition on dressing factor similar to (2.2.58)

$$\sigma(\tilde{x}, y)\sigma(x, y) = \frac{1 + \frac{1}{x^+y^+}}{1 - \frac{x^-}{y^-}}\frac{1 - \frac{x^-}{y^+}}{1 - \frac{1}{x^+y^-}} \tag{2.2.67}$$

where $\tilde{x}$ denotes that we should perform a crossing transformation of the first particle in the corresponding dressing factor.

Perturbative Dressing factor

Initially, the dressing factor was proposed in [89] to fix the discrepancy between the all-loop asymptotic Bethe ansatz of the gauge and string theory. The functional form of the dressing phase in terms of the local conserved charges of the model was conjectured [89] by discretising the finite-gap solutions. The most general form was suggested in [92]

$$\theta(x_1^+, x_1^-, x_2^+, x_2^-) = \sum_{r=2}^{\infty} \sum_{\substack{s>r \\ r+s=\text{odd}}}^{\infty} c_{r,s}(g) \Big[q_r(x_1^\pm) q_s(x_2^\pm) - q_r(x_2^\pm) q_s(x_1^\pm) \Big] , \tag{2.2.68}$$

where $\sigma(x_1^\pm, x_2^\pm) = e^{i\theta(x_1^\pm, x_2^\pm)}$, q_r are conserved charges of the corresponding spin-chain and are given by

$$q_r(x_k^-, x_k^+) = \frac{i}{r-1} \left[\left(\frac{1}{x_k^+} \right)^{r-1} - \left(\frac{1}{x_k^-} \right)^{r-1} \right] . \tag{2.2.69}$$

This, in turn, implies that there should be a so-called χ-decomposition of it [93]

$$\theta(x_1^+, x_1^-, x_2^+, x_2^-) = \chi(x_1^+, x_2^+) + \chi(x_1^-, x_2^-) - \chi(x_1^+, x_2^-) - \chi(x_1^-, x_2^+) . \tag{2.2.70}$$

This will be useful when we formulate the solution in terms of χ's, as it simplifies the structure of the solution considerably. Additionally, it tells us that functions $\chi(x, y)$ which differ by a constant of a function of x or y only will lead to the same phase θ. We will rely heavily on this property in Section 3.3.

The dressing factor explicitly depends on the string tension g and admits a strong-coupling expansion in powers of $1/g$ corresponding to an asymptotic perturbative expansion of the string sigma model. In [89], a solution to (2.2.67) was found in the strong-coupling expansion only at the leading g^1 order (AFS-order). The first quantum correction g^0 order (HL-order) was discovered in [94]. The full asymptotic series was proposed in [95]. The authors of [95] also discovered that the odd crossing relation[17] is solved by an HL-order solution (whose asymptotic series can be summed up), and suggested a solution for the remaining even part of the crossing equation.

In contrast to the strong-coupling expansion, the gauge theory perturbative expansion of the dressing factor is in powers of g and has a finite radius of convergence. The dressing factor suitable for the weak coupling expansion has been found by assuming a certain analytic continuation of its strong-coupling counterpart, and has the name Beisert-Eden-Staudacher (BES) dressing factor [73]. This is in agreement with the explicit two-loop sigma model results [96], [97]. It can be shown that the BES dressing factor satisfies crossing equations for finite values of the string coupling [98]. Then Dorey, Hofman and Maldacena (DHM) [99] gave the most useful representation of this phase in terms of a double integral

$$\chi(x, y) = \oint \frac{dz}{2\pi i} \oint \frac{dz'}{2\pi i} \frac{1}{z-x} \frac{1}{z'-y} \log \frac{\Gamma[1 + \frac{ih}{2}(z + 1/z - z' - 1/z')]}{\Gamma[1 - \frac{ih}{2}(z + 1/z - z' - 1/z')]} . \tag{2.2.71}$$

A relatively short and very illustrative way to derive the DHM representation of the solution of crossing (2.2.67) is to use a half-crossing procedure and shift operators [100].

Additionally, there might be a homogeneous part of the crossing which cannot be determined from the crossing equation, as it is a solution of the crossing equation with a trivial RHS (identity) [95], [100]. However, we can demand that only certain poles and zeros, corresponding to the expected bound states, and some higher-order poles can be present (we will discuss this later).

[17] This will be discussed later in section 2.3.4.

2.2.6 General S-matrix properties

Before we switch to the discussion of the AdS_3 case, it is worth summarising the discussion above into a set of general principles which can be used to construct an S-matrix.

There are three objects which we will be working with, depending on convenience

$$\check{\mathrm{S}}(p,q) = \mathbb{1}^g \mathrm{S}(p,q) = \Pi\mathbb{R}(p,q)\,, \tag{2.2.72}$$

where Π is a permutation, $\mathbb{1}^g$ is the graded identity, $\mathrm{S}(p,q)$ is the standard S-matrix, $\check{\mathrm{S}}(p,q)$ is the graded S-matrix, and $\mathbb{R}(p,q)$ is the R-matrix. Some expressions will be simpler with one object and lengthier with others, so we might change between them in the text.

One may identify a general set of symmetries which the S-matrix should obey. These include

1. Full off-shell symmetry algebra $\mathcal{A}$

$$\check{\mathrm{S}}_{12}(p,q)\,\mathbf{J}_{12}(p,q) - \mathbf{J}_{12}(q,p)\,\check{\mathrm{S}}_{12}(p,q) = 0\,, \tag{2.2.73}$$

   where $\mathbf{J}_{12}(p,q) \in \mathcal{A}$.

2. Physical unitarity. The usual concept of unitarity for an S-matrix

$$\big(\check{\mathrm{S}}_{(12)}(p,q)\big)^{\dagger}\check{\mathrm{S}}_{(12)}(p,q) = \mathbf{I}\,,$$

3. Braiding unitarity is the consistency relation for factorised two–body scattering

$$\check{\mathrm{S}}_{(12)}(q,p)\check{\mathrm{S}}_{(12)}(p,q) = \mathbf{I}\,.$$

   Braiding unitarity supplements the usual physical unitarity condition. The latter states that S-matrix should be a unitary matrix on the physical sheet. However, the former is the statement that the S-matrix is exchanging two excitations and thus exchanging once with $\check{\mathrm{S}}_{(12)}(p,q)$ and then back with $\check{\mathrm{S}}_{(12)}(q,p)$ must give the identity.

4. Yang-Baxter equation. This is a manifestation of integrability at the quantum level (assuming that integrable structures extend to the quantum level).

$$\check{\mathrm{S}}_{(23)}(q,r)\check{\mathrm{S}}_{(13)}(p,r)\check{\mathrm{S}}_{(12)}(p,q) = \check{\mathrm{S}}_{(12)}(p,q)\check{\mathrm{S}}_{(13)}(p,r)\check{\mathrm{S}}_{(23)}(q,r)\,.$$

Together, braiding unitarity and the Yang–Baxter equation ensure multi-particle scattering is well-defined when the multi-particle S-matrix is built by factorisation from the two-body S-matrix. This is efficiently encoded in the Zamolodchikov–Faddeev (ZF) algebra, where both of them serve as consistency (associativity) conditions for the ZF algebra we introduced in 2.2.4.[18]

Additionally, there is a list of assumptions on the analytical structure of σ as a function of complex arguments that fix its functional form.

- Bound states and simple poles/zeros. The position of simple poles in the S-matrix should correctly reproduce the structure of the bound states. This is usually accounted for in the normalisation of the S-matrix elements, and the $\sigma(x,y)$ should not have simple poles. Unitarity also excludes simple zeros.

[18] See [101] for a review in the AdS_3/CFT_2 context.

- Poles/zeros of higher degree. The only allowed poles of $\sigma(x, y)$ are double poles from the exchange of pairs of composite states discovered by Dorey, Hofman, and Maldacena [99]. By unitarity, there should be the same number of double zeroes.

- Branch points. We require that the structure of the branch points be as simple as possible so that only the branch points required by the crossing equations are allowed.

- χ-decomposition. This is a direct consequence of the decomposition of $\theta(x, y)$ in terms of the higher conserved charges. This form is expected for general long-range integrable spin chains [92].

- Asymptotics at infinity. Since the infinite x limit corresponds to zero momenta, $\lim_{x\to\infty} \sigma(x, y) = 1$ as we expect particles to scatter trivially with all zero-momentum particles.

- Analyticity of $\chi(x, y)$ in the physical domain. There should be no poles/branch cuts for χ in the physical region. Otherwise, they would lead to unphysical singularities in the description of the scattering of bound states.

We will now switch to the discussion of integrability in strings in AdS_3. Many of the features will be shared with the AdS_5, but there will also be some notable differences.

2.3 Integrability in $AdS_3 \times S^3$

The existence of holography in AdS_3/CFT_2 systems has long been known [3], but only recently has more progress been made in understanding unprotected quantities.

String theories on $AdS_3 \times S^3 \times \mathcal{M}_4$, where $\mathcal{M}_4$ equal to K_3, $S^3 \times S^1$[19] or T^4[20], are classically integrable [102], [103][21]. One of the hints that this is indeed the case is the fact that some giant magnon solutions in $AdS_5 \times S^5$ are contained in $\mathbb{R} \times S^2$, which can be embedded in $AdS_3 \times S^3 \times \mathcal{M}^4$ [104]. These findings gave hope that it is possible to generalise what was done for the $AdS_5 \times S^5$ string to the AdS_3 backgrounds.

However, there are two features that make string theory in $AdS_3 \times S^3 \times \mathcal{M}^4$ different from previous cases. First, string theory in $AdS_5 \times S^5$ can be supported by Ramon-Ramon (R-R) charge fields only, while AdS_3 backgrounds can be supported by a Neveu-Schwarz-Neveu-Schwarz (NS-NS) charge or by a combination of NS-NS and R-R charges. The resulting NLSM remains integrable in all the cases[22] [105]. Second, on AdS_3 backgrounds (unlike the two very first examples in $AdS_5 \times S^5$ and $AdS_4 \times CP^3$) there are not only massive but also massless modes . Therefore, the application of integrability techniques developed in AdS_5/CFT_4 is not straightforward.

In what follows, we will focus on the $AdS_3 \times S^3 \times T^4$ background and discuss its symmetry algebra construction, representations, and its application to crossing.

[19] The radii R_+ and R_- of the two S^3's in this background are related to the radius R of AdS_3 as $\frac{1}{R^2} = \frac{1}{R_+^2} + \frac{1}{R_-^2}$.

[20] When $R^2_{AdS_3} = R^2_{S^3}$

[21] $\mathcal{M}_4$ can also be a Z_n orbifold of T^4, which gives a blown-down K_3, but integrability is less understood and depends on the moduli.

[22] Notice that not all combinations of NS-NS and R-R fluxes are allowed, but only those whose coefficients satisfy a certain constraint.

2.3.1 Symmetry Algebra

The theory can also be written in a super-coset description

$$\frac{\mathrm{PSU}(1,1|2)_L \times \mathrm{PSU}(1,1|2)_R}{\mathrm{SO}(1,2) \times \mathrm{SO}(3)}\,, \tag{2.3.1}$$

for the pure R-R case. However, the coset action requires the use of a particular kappa gauge, which does not allow for a straightforward quantisation of the massless modes [106]. Therefore, the Green-Schwarz formalism must be used [107], [108], [109].

The quantisation procedure for the Green-Schwarz action follows the lines for AdS_5, where one quantises it in a certain light-cone gauge. The resulting NLSM can be considered as describing a theory on a cylinder of finite circumference proportional to the light-cone momentum with infinite time. One then considers the decompactification limit (light-cone momentum $P_+ \to \infty$), where the factorised scattering theory can be applied.

The full superisomtery algebra of the background is [110]

$$\mathfrak{psu}(1,1|2)_L \oplus \mathfrak{psu}(1,1|2)_R \oplus \mathfrak{u}(4)^4\,, \tag{2.3.2}$$

where the four $\mathfrak{u}(1)$ factors originate from the four T^4 directions.

The symmetry algebra of the Green–Schwarz action can be computed by considering the superalgebra spanned by supercurrents truncated at the quadratic order in the fields. The results are identical to those of the computation in AdS_5 for the massive sector [77]. Additionally, there is a massless sector, the computation of which was performed in [110].

In the decompactification limit, $\mathfrak{u}(1)$ factors are enhanced to $\mathfrak{so}(4)$, so one can find the symmetry algebra[23]

$$\mathfrak{psu}(1|1)^4_{c.e.} \oplus \mathfrak{so}(4)\,. \tag{2.3.3}$$

However, the last $\mathfrak{so}(4)$ factor is only an apparent symmetry of massive excitations in the quadratic gauge fixed action or decompactification limit, but it is not the true symmetry of the quantum theory. The algebra of the general quantised sigma model after light-cone gauge fixing is

$$\mathfrak{psu}(1|1)^4{}_{c.e.}\,. \tag{2.3.4}$$

One can see that by computing the higher than quadratic gauge-fixed action where $\mathfrak{so}(4)$ will no longer be a symmetry [110].

The $\mathfrak{su}(1|1)^2{}_{c.e.}$ symmetry algebra and its representations

In this section, we will construct the symmetry algebra explicitly from simple components[24].

The full symmetry algebra $\mathfrak{psu}(1|1)^4{}_{c.e.}$ can be constructed as a tensor product of fundamental representations of the smaller algebra $\mathfrak{su}(1|1)^2{}_{c.e.}$. In turn, $\mathfrak{su}(1|1)^2_{c.e.}$ is equivalent to centrally extended $\mathfrak{su}(1|1)_{\mathrm{L}} \oplus \mathfrak{su}(1|1)_{\mathrm{R}}$

$$\{\mathbf{q}_{\mathrm{L}}, \overline{\mathbf{q}}_{\mathrm{L}}\} = \mathbf{h}_{\mathrm{L}}\,, \quad \{\mathbf{q}_{\mathrm{R}}, \overline{\mathbf{q}}_{\mathrm{R}}\} = \mathbf{h}_{\mathrm{R}}\,, \tag{2.3.5}$$

which are coupled via the Hamiltonian and an angular momentum

$$\mathbf{h} = \mathbf{h}_{\mathrm{L}} + \mathbf{h}_{\mathrm{R}}\,, \quad \mathbf{m} = \mathbf{h}_{\mathrm{L}} - \mathbf{h}_{\mathrm{R}}\,, \tag{2.3.6}$$

[23] Many methods exist for constructing central extensions. What we mean here by $_{c.e.}$ is a central extension between Left and Right copies of $\mathfrak{su}(1|1)$, which are then taken twice to get $\mathfrak{psu}(1|1)^2{}_{c.e.} \oplus \mathfrak{psu}(1|1)^2{}_{c.e.} = \mathfrak{psu}(1|1)^4{}_{c.e.}$.

[24] More details can be found in [110].

and have a central extension of the form

$$\{\mathbf{q}_{\text{L}}, \mathbf{q}_{\text{R}}\} = \mathbf{c}\,, \quad \{\overline{\mathbf{q}}_{\text{L}}, \overline{\mathbf{q}}_{\text{R}}\} = \overline{\mathbf{c}}\,. \tag{2.3.7}$$

Here $\mathbf{h}$ corresponds to the Hamiltonian and $\mathbf{m}$ is a combination of angular momenta in $\text{AdS}_3 \times \text{S}^3$. Supercharges are denoted by $\mathbf{q}$. The labels L or R are inherited from the supersymmetry algebra $\mathfrak{psu}(1,1|2)_{\text{L}} \oplus \mathfrak{psu}(1,1|2)_{\text{R}}$.

We then consider the tensor product of two copies of the above algebra. We add the $\mathfrak{su}(2)$ index "1" to denote that an operator sits in the first space in a tensor product, and the index "2" if it is in the second space[25]

$$\begin{aligned} &\mathbf{Q}_{\text{L}}{}^{1} = \mathbf{q}_{\text{L}} \otimes \mathbb{1}, \qquad \overline{\mathbf{Q}}_{\text{L}1} = \overline{\mathbf{q}}_{\text{L}} \otimes \mathbb{1}, \qquad \mathbf{Q}_{\text{L}}{}^{2} = \Sigma \otimes \mathbf{q}_{\text{L}}, \qquad \overline{\mathbf{Q}}_{\text{L}2} = \Sigma \otimes \overline{\mathbf{q}}_{\text{L}}, \\ &\mathbf{Q}_{\text{R}1} = \mathbf{q}_{\text{R}} \otimes \mathbb{1}, \qquad \overline{\mathbf{Q}}_{\text{R}}{}^{1} = \overline{\mathbf{q}}_{\text{R}} \otimes \mathbb{1}, \qquad \mathbf{Q}_{\text{R}2} = \Sigma \otimes \mathbf{q}_{\text{R}}, \qquad \overline{\mathbf{Q}}_{\text{R}}{}^{2} = \Sigma \otimes \overline{\mathbf{q}}_{\text{R}}. \end{aligned} \tag{2.3.8}$$

The matrix Σ is defined as a diagonal matrix taking the value $+1$ for bosons and -1 for fermions. The same should be done to the central elements

$$\begin{aligned} &\mathbf{H}_{\text{L}}{}^{1} = \mathbf{h}_{\text{L}} \otimes \mathbb{1}, \qquad \mathbf{H}_{\text{L}}{}^{2} = \mathbb{1} \otimes \mathbf{h}_{\text{L}}, \qquad \mathbf{C}^{1} = \mathbf{c} \otimes \mathbb{1}, \qquad \mathbf{C}^{2} = \mathbb{1} \otimes \mathbf{c}, \\ &\mathbf{H}_{\text{R}}{}^{1} = \mathbf{h}_{\text{R}} \otimes \mathbb{1}, \qquad \mathbf{H}_{\text{R}}{}^{2} = \mathbb{1} \otimes \mathbf{h}_{\text{R}}, \qquad \overline{\mathbf{C}}^{1} = \overline{\mathbf{c}} \otimes \mathbb{1}, \qquad \overline{\mathbf{C}}^{2} = \mathbb{1} \otimes \overline{\mathbf{c}}. \end{aligned} \tag{2.3.9}$$

To reproduce the property that central charges are not charged under the $\mathfrak{su}(2)$ algebra, we make identifications

$$\mathbf{H}_{\text{L}} \equiv \mathbf{H}_{\text{L}}{}^{1} = \mathbf{H}_{\text{L}}{}^{2}, \quad \mathbf{H}_{\text{R}} \equiv \mathbf{H}_{\text{R}}{}^{1} = \mathbf{H}_{\text{R}}{}^{2}, \qquad \mathbf{C} \equiv \mathbf{C}^{1} = \mathbf{C}^{2}, \quad \overline{\mathbf{C}} \equiv \overline{\mathbf{C}}^{1} = \overline{\mathbf{C}}^{2}\,, \tag{2.3.10}$$

so we can omit indices ^{1} and ^{2} in the following. After that, we get the following central extension

$$\begin{aligned} &\{\mathbf{Q}_{\text{L}}{}^{\dot{a}}, \overline{\mathbf{Q}}_{\text{L}\dot{b}}\} = \frac{1}{2}\delta^{\dot{a}}_{\ \dot{b}}\,(\mathbf{H} + \mathbf{M}), \qquad \{\mathbf{Q}_{\text{L}}{}^{\dot{a}}, \mathbf{Q}_{\text{R}\dot{b}}\} = \delta^{\dot{a}}_{\ \dot{b}}\,\mathbf{C}, \\ &\{\mathbf{Q}_{\text{R}\dot{a}}, \overline{\mathbf{Q}}_{\text{R}}{}^{\dot{b}}\} = \frac{1}{2}\delta_{\dot{a}}^{\ \dot{b}}\,(\mathbf{H} - \mathbf{M}), \qquad \{\overline{\mathbf{Q}}_{\text{L}\dot{a}}, \overline{\mathbf{Q}}_{\text{R}}{}^{\dot{b}}\} = \delta_{\dot{a}}^{\ \dot{b}}\,\overline{\mathbf{C}}\,, \end{aligned} \tag{2.3.11}$$

where $\dot{a}, \dot{b} \in \{1, 2\}$, $\mathbf{H} = \mathbf{H}_{\text{L}} + \mathbf{H}_{\text{R}}$ and $\mathbf{M} = \mathbf{H}_{\text{L}} - \mathbf{H}_{\text{R}}$.

Therefore, the central charges are given by the Hamiltonian $\mathbf{H}$, the angular momentum $\mathbf{M}$, and the central elements $\mathbf{C}$ and $\mathbf{C}^\dagger$. One can check the expressions for the charges at the quadratic order in fields explicitly [77], [110]. Additionally, one could find how to express central charges $\mathbf{C}$ and $\overline{\mathbf{C}}$ via the worldsheet momentum

$$\mathbf{C} = +\frac{i\xi}{2}\left(e^{+i\mathbf{P}} - 1\right), \quad \overline{\mathbf{C}} = -\frac{i\xi}{2}\left(e^{-i\mathbf{P}} - 1\right)\,. \tag{2.3.12}$$

Fundamental excitations are expected to transform into two distinct representations for massive and massless particles, each being 4-dimensional. These representations must be short[26], so the following shortening condition must be satisfied

$$\mathbf{H}_{\text{L}}\mathbf{H}_{\text{R}} = \mathbf{C}\overline{\mathbf{C}}\,, \tag{2.3.13}$$

which one can recast in the form of a dispersion relation

$$\mathbf{H}^2 = \mathbf{M}^2 + 4\mathbf{C}\overline{\mathbf{C}}\,, \tag{2.3.14}$$

[25] Notice that Left supercharges have an upper $\mathfrak{su}(2)$ index, while for Right supercharges the index is lower. This is the reflection of the convention that Left supercharges transform in the anti-fundamental representation of $\mathfrak{su}(2)_\bullet$, and Right supercharges transform in the fundamental.

[26] Long representations are at least 16-dimensional.

where the eigenvalues of $\mathbf{M}$ play the role of a mass term. This relation immediately allows solving for the eigenvalue E_p of the Hamiltonian $\mathbf{H}$ using (2.3.12)

$$E_p = \sqrt{m^2 + 4h^2 \sin^2 \frac{p}{2}}\,. \tag{2.3.15}$$

Bi-fundamental representations

To construct the physical representations, it is sufficient to consider four inequivalent representations of $\mathfrak{su}(1|1)^2_{c.e}$ (found in [111]): ρ_{L}, ρ_{R}, $\tilde{\rho}_{\text{L}}$ and $\tilde{\rho}_{\text{R}}$. Each is based on two states, one bosonic $|\phi^a_p\rangle$ and one fermionic $|\psi^a_p\rangle$, where p denotes the physical momentum of the excitation and $a \in \{L, R, \tilde{L}, \tilde{R}\}$. Representations with and without the tilde differ by the choice of the highest weight state.

Representation ρ_{L} is constructed from ϕ^{L}_p and ψ^{L}_p, ρ_{R} – from ϕ^{R}_p and ψ^{R}_p, $\tilde{\rho}_{\text{L}}$ – from $\tilde{\phi}^{\text{L}}_p$ and $\tilde{\psi}^{\text{L}}_p$, $\tilde{\rho}_{\text{R}}$ – from $\tilde{\phi}^{\text{R}}_p$ and $\tilde{\psi}^{\text{R}}_p$.

The action of $\mathfrak{su}(1|1)^2_{c.e}$ generators on ρ_{L} and ρ_{R} states (the most interesting ones for us)

$$\begin{array}{cccc}
\overbrace{\qquad\qquad\qquad\qquad}^{\rho_{\text{L}}} & & & \overbrace{\qquad\qquad\qquad\qquad}^{\rho_{\text{R}}} \\
\mathbf{Q}_{\text{L}}\,|\phi^{\text{L}}_p\rangle = a_p\,|\psi^{\text{L}}_p\rangle\,, & \mathbf{Q}_{\text{L}}\,|\psi^{\text{L}}_p\rangle = 0, & \mathbf{Q}_{\text{R}}\,|\phi^{\text{R}}_p\rangle = a_p\,|\psi^{\text{R}}_p\rangle\,, & \mathbf{Q}_{\text{R}}\,|\psi^{\text{R}}_p\rangle = 0, \\
\overline{\mathbf{Q}}_{\text{L}}\,|\phi^{\text{L}}_p\rangle = 0, & \overline{\mathbf{Q}}_{\text{L}}\,|\psi^{\text{L}}_p\rangle = \bar{a}_p\,|\phi^{\text{L}}_p\rangle\,, & \overline{\mathbf{Q}}_{\text{R}}\,|\phi^{\text{R}}_p\rangle = 0, & \overline{\mathbf{Q}}_{\text{R}}\,|\psi^{\text{R}}_p\rangle = \bar{a}_p\,|\phi^{\text{R}}_p\rangle\,, \\
\mathbf{Q}_{\text{R}}\,|\phi^{\text{L}}_p\rangle = 0, & \mathbf{Q}_{\text{R}}\,|\psi^{\text{L}}_p\rangle = b_p\,|\phi^{\text{L}}_p\rangle\,, & \mathbf{Q}_{\text{L}}\,|\phi^{\text{R}}_p\rangle = 0, & \mathbf{Q}_{\text{L}}\,|\psi^{\text{R}}_p\rangle = b_p\,|\phi^{\text{R}}_p\rangle\,, \\
\overline{\mathbf{Q}}_{\text{R}}\,|\phi^{\text{L}}_p\rangle = \bar{b}_p\,|\psi^{\text{L}}_p\rangle\,, & \overline{\mathbf{Q}}_{\text{R}}\,|\psi^{\text{L}}_p\rangle = 0. & \overline{\mathbf{Q}}_{\text{L}}\,|\phi^{\text{R}}_p\rangle = \bar{b}_p\,|\psi^{\text{R}}_p\rangle\,, & \overline{\mathbf{Q}}_{\text{L}}\,|\bar{\psi}_p\rangle = 0.
\end{array}$$

Eventually, the Left-massive (mass $m = 1$), the Right-massive (mass $m = -1$) and the massless (mass $m = 0$) modules will be constructed as the following tensor products of representations[27][28]

$$\text{Left}: \ \rho_{\text{L}} \otimes \rho_{\text{L}}, \qquad \text{Right}: \ \rho_{\text{R}} \otimes \rho_{\text{R}}, \qquad \text{massless}: \ (\rho_{\text{L}} \otimes \tilde{\rho}_{\text{L}})^{\oplus 2}\,. \tag{2.3.16}$$

These bi-fundamental representations of $\mathfrak{su}(1|1)^2_{c.e}$ have the following identifications for the massive states

$$\begin{array}{cccc}
Y^{\text{L}} = \phi^{\text{L}} \otimes \phi^{\text{L}}, & \eta^{\text{L}1} = \psi^{\text{L}} \otimes \phi^{\text{L}}, & \eta^{\text{L}2} = \phi^{\text{L}} \otimes \psi^{\text{L}}, & Z^{\text{L}} = \psi^{\text{L}} \otimes \psi^{\text{L}}, \\
Y^{\text{R}} = \phi^{\text{R}} \otimes \phi^{\text{R}}, & \eta^{\text{R}}_1 = \psi^{\text{R}} \otimes \phi^{\text{R}}, & \eta^{\text{R}}_2 = \phi^{\text{R}} \otimes \psi^{\text{R}}, & Z^{\text{R}} = \psi^{\text{R}} \otimes \psi^{\text{R}},
\end{array} \tag{2.3.17}$$

and the massless ones as

$$T^{1a} = \left(\psi^{\text{L}} \otimes \tilde{\psi}^{\text{L}}\right)^a, \quad \widetilde{\chi}^a = \left(\psi^{\text{L}} \otimes \tilde{\phi}^{\text{L}}\right)^a, \quad \chi^a = \left(\phi^{\text{L}} \otimes \tilde{\psi}^{\text{L}}\right)^a, \quad T^{2a} = \left(\phi \otimes \tilde{\phi}^{\text{L}}\right)^a. \tag{2.3.18}$$

In the following, we will focus on massive representations. These can be acted on by the supercharges, as shown in Figure 2.1. The action can be expressed using two complex coefficients, a_p and b_p, and their complex conjugates, $\bar{a}_p$ and $\bar{b}_p$. These coefficients depend on the momentum p of the excitation and the eigenvalue m of the central charge $\mathbf{M}$ in the

[27] In fact, massless representation can be constructed in four different ways by combining ρ and $\tilde{\rho}$ with either L or R indices, but all of them are equivalent.

[28] Notice that for the massless module one has to consider two copies of $\rho_{\text{L}} \otimes \rho_{\text{L}}$, hence the symbol $\oplus 2$, and twice as many particles compared to massive Left or Right modules.

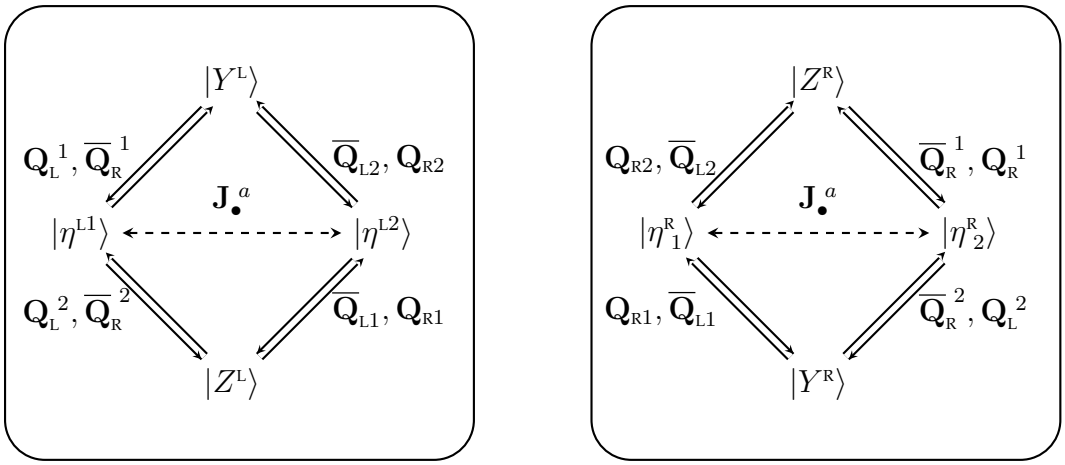


Figure 2.1: The action of $\mathfrak{psu}(1|1)^4{}_{c.e.}$ generators in left and right multiplets of two bosons $Y^{\rm L,R}$, $Z^{\rm L,R}$ and of two fermions $\eta^{\rm L}_{\dot a}$, $\eta^{\rm R}_{\dot a}$.

specific module. By computing anti-commutators of supercharges, we can write the relation between these coefficients and the eigenvalue of the central charges

$$\begin{aligned}
\mathbf{M}: &\qquad a_p\bar a_p - b_p\bar b_p = |m|\,,\\
\mathbf{H}: &\qquad a_p\bar a_p + b_p\bar b_p = \sqrt{m^2 + 4h^2\sin^2\frac{p}{2}}\,,\\
\mathbf{C}: &\qquad\qquad a_p b_p = h\,\frac{i}{2}(e^{ip}-1)\,\zeta\,.
\end{aligned} \tag{2.3.19}$$

Here, $\zeta = e^{2i\,\xi}$ is a free parameter of the representation. For one-particle states, it can be taken to be 1, but, similar to the case of AdS_5, it can take only specific values for the multiparticle states.

A way to solve (2.3.19) is to introduce the Zhukovski parameters $x^\pm$, which satisfy

$$x_p^+ + \frac{1}{x_p^+} - x_p^- - \frac{1}{x_p^-} = \frac{2i\,|m|}{h}, \qquad \frac{x_p^+}{x_p^-} = e^{ip}. \tag{2.3.20}$$

Then, one can take the representation coefficients to be [111], [112]

$$a_p = \eta_p e^{i\xi}, \quad \bar a_p = \eta_p\left(\frac{x_p^+}{x_p^-}\right)^{-1/2} e^{-i\xi}, \quad b_p = -\frac{\eta_p}{x_p^-}\left(\frac{x_p^+}{x_p^-}\right)^{-1/2} e^{i\xi}, \quad \bar b_p = -\frac{\eta_p}{x_p^+}e^{-i\xi}, \tag{2.3.21}$$

where we have introduced the function

$$\eta_p = \left(\frac{x_p^+}{x_p^-}\right)^{1/4}\sqrt{\frac{ih}{2}(x_p^- - x_p^+)}\,. \tag{2.3.22}$$

This parametrisation is convenient in the sense that it eliminates the square root in the dispersion relation

$$E(x^\pm) = \frac{h}{2i}\left(x_p^+ - \frac{1}{x_p^+} - x_p^- + \frac{1}{x_p^-}\right). \tag{2.3.23}$$

The constraints (2.3.20) on $x^\pm$ can be resolved by taking

$$x_p^\pm = \frac{(|m| + E_p)}{2h\sin\left(\frac{p}{2}\right)}e^{\pm\frac{ip}{2}}\,, \tag{2.3.24}$$

where the branch of the square root has been chosen such that $|x_p^\pm| > 1$ for real values of the momentum p, when we consider massive states $|m| > 0$. For massless states, we have simply

$$x_p^\pm = \mathrm{sgn}(\sin\tfrac{p}{2})\, e^{\pm\frac{ip}{2}}, \qquad E_p = 2h\left|\sin\frac{p}{2}\right|. \tag{2.3.25}$$

Two-particle states in pure R-R

When constructing two-particle states, we closely follow Section 2.2.5. The standard method for defining two-particle states is to use a co-product. Given a one-particle state $|\mathcal{X}\rangle$ and a charge $\mathbf{J}$ with an action

$$\mathbf{J}\,|\mathcal{X}\rangle = |\mathcal{Y}\rangle\ , \tag{2.3.26}$$

its action on a two-particle state should be

$$\mathbf{J}_{12}\,|\mathcal{X}_1\mathcal{X}_2\rangle = (\mathbf{J}\otimes\mathbb{1} + \mathbb{1}\otimes\mathbf{J})\,|\mathcal{X}_1\mathcal{X}_2\rangle = |\mathcal{Y}_1\mathcal{X}_2\rangle + |\mathcal{X}_1\mathcal{Y}_2\rangle\ . \tag{2.3.27}$$

However, as already mentioned after formula (2.3.19), when constructing two-particle states, we cannot take just the tensor product of one-particle states. One way to see why we need a deformation is to use the definition of the central charge $\mathbf{C} = \frac{ih}{2}(e^{+i\mathbf{P}} - 1)$. We expect

$$\mathbf{C}\,|\mathcal{X}_1\mathcal{X}_2\rangle = \frac{ih}{2}(e^{+i(p_1+p_2)} - 1)\,|\mathcal{X}_1\mathcal{X}_2\rangle\ . \tag{2.3.28}$$

However, with the standard co-product, we get

$$\mathbf{C}\,|\mathcal{X}_1\mathcal{X}_2\rangle = \frac{ih}{2}\left(e^{2i\xi_1}(e^{+ip_1} - 1) + e^{2i\xi_2}(e^{+ip_2} - 1)\right)|\mathcal{X}_1\mathcal{X}_2\rangle\ . \tag{2.3.29}$$

Thus, we cannot set $e^{i\xi_1} = e^{i\xi_2} = 1$ and need to consider other values of the parameters of the representations. To make two equations above agree, we should set $\{e^{2i\xi_1} = 1, e^{2i\xi_2} = e^{ip_1}\}$ or $\{e^{2i\xi_1} = e^{ip_2}, e^{2i\xi_2} = 1\}$.

Another way to think about this is from the point of view of the Hopf algebra described in Section 2.2.5: the product needs to be deformed relative to the standard product to reproduce the symmetry algebra at the level of the bialgebra.

The conclusion we have seen for the AdS_5 case will also hold for AdS_3: we need to deform the action of all odd generators in the following way

$$\begin{aligned}\mathbf{Q}(p_1,p_2) &= \mathbf{Q}(p_1)\otimes\mathbb{1} + e^{+\frac{i}{2}\mathbf{P}}\Sigma\otimes\mathbf{Q}(p_2)\,,\\ \overline{\mathbf{Q}}(p_1,p_2) &= \overline{\mathbf{Q}}(p_1)\otimes\mathbb{1} + e^{-\frac{i}{2}\mathbf{P}}\Sigma\otimes\overline{\mathbf{Q}}(p_2)\,,\end{aligned} \tag{2.3.30}$$

as well as the central charges $\mathbf{C}$ and $\overline{\mathbf{C}}$

$$\begin{aligned}\mathbf{C}(p_1,p_2) &= \mathbf{C}(p_1)\otimes\mathbb{1} + e^{+i\mathbf{P}}\otimes\mathbf{C}(p_2)\,,\\ \overline{\mathbf{C}}(p_1,p_2) &= \overline{\mathbf{C}}(p_1)\otimes\mathbb{1} + e^{-i\mathbf{P}}\otimes\overline{\mathbf{C}}(p_2)\,.\end{aligned} \tag{2.3.31}$$

The action of the rest of the generators, such as Hamiltonian $\mathbf{H}$ or angular momentum M, remains the same.

2.3.2 The Exact S-matrix

General properties of the S-matrix outlines in Section 2.2.6 remain true in AdS_3 as well. Symmetry algebra $\mathcal{A}$ now becomes $\mathfrak{psu}(1|1)^4_{c.e.}$.

Additionally, we will see that there are algebra labels L and R which lead to a new symmetry:

- Left-right symmetry. Relabelling $L \leftrightarrow R$ should be indistinguishable. In particular, this relates the LL dressing factor to the R-R one and the LR dressing factor to the RL one.

We will use the fact that fundamental representations of $\mathfrak{psu}(1|1)^4_{c.e.}$ can be understood as bi-fundamental representations of $\mathfrak{psu}(1|1)^2_{c.e.}$. This allows us to rewrite an S-matrix invariant under $\mathfrak{psu}(1|1)^4_{c.e.}$ algebra as a tensor product of two copies of $\mathfrak{psu}(1|1)^2_{c.e.}$-invariant S-matrices.

Due to the invariance under the action of the generators $\mathbf{M}$ and $\mathbf{H}$ and integrability, one can split the $\mathfrak{psu}(1|1)^2_{c.e.}$-invariant S-matrix into massive, massless, and mixed mass sectors.

In Section 2.3.1 we presented the fundamental representations of $\mathfrak{su}(1|1)^2_{c.e.}$ which we called ρ_{L}, ρ_{R}, $\tilde{\rho}_{\mathrm{L}}$ and $\tilde{\rho}_{\mathrm{L}}$, which were then combined into massive left, right and massless representation. However, we are interested in the scattering of massive excitations and thus consider the corresponding massive S-matrix below.

In the left-left (LL) sector, we examine the two particles from ρ_{L} representations. Their scattering is restricted by symmetry algebra [111], [113] and is described by $\check{\mathrm{S}}^{\mathrm{LL}}$. One can argue [71] that having enough conserved charges (which is the case in integrable theories) leads to scattering of transmission and reflection type only, which results in the following non-zero matrix elements (we are sticking to normalisations of [111])

$$\begin{aligned}
\check{\mathrm{S}}^{\mathrm{LL}}\,|\phi^{\mathrm{L}}_p\phi^{\mathrm{L}}_q\rangle &= A^{\mathrm{LL}}_{pq}\,|\phi^{\mathrm{L}}_q\phi^{\mathrm{L}}_p\rangle\ , & \check{\mathrm{S}}^{\mathrm{LL}}\,|\phi^{\mathrm{L}}_p\psi^{\mathrm{L}}_q\rangle &= B^{\mathrm{LL}}_{pq}\,|\psi^{\mathrm{L}}_q\phi^{\mathrm{L}}_p\rangle + C^{\mathrm{LL}}_{pq}\,|\phi^{\mathrm{L}}_q\psi^{\mathrm{L}}_p\rangle\ ,\\
\check{\mathrm{S}}^{\mathrm{LL}}\,|\psi^{\mathrm{L}}_p\psi^{\mathrm{L}}_q\rangle &= F^{\mathrm{LL}}_{pq}\,|\psi^{\mathrm{L}}_q\psi^{\mathrm{L}}_p\rangle\ , & \check{\mathrm{S}}^{\mathrm{LL}}\,|\psi^{\mathrm{L}}_p\phi^{\mathrm{L}}_q\rangle &= D^{\mathrm{LL}}_{pq}\,|\phi^{\mathrm{L}}_q\psi^{\mathrm{L}}_p\rangle + E^{\mathrm{LL}}_{pq}\,|\psi^{\mathrm{L}}_q\phi^{\mathrm{L}}_p\rangle\ .
\end{aligned} \tag{2.3.32}$$

All coefficients are determined up to an overall factor. The standard convention is to normalise $A^{\mathrm{LL}}_{pq} = 1$. By requiring the compatibility of the S-matrix with the symmetry algebra (2.2.73), one can fully fix the values of all other matrix elements

$$\begin{aligned}
A^{\mathrm{LL}}_{pq} &= 1, & B^{\mathrm{LL}}_{pq} &= \left(\frac{x^-_p}{x^+_p}\right)^{1/2}\frac{x^+_p - x^+_q}{x^-_p - x^+_q}\,,\\
C^{\mathrm{LL}}_{pq} &= \left(\frac{x^-_p}{x^+_p}\frac{x^+_q}{x^-_q}\right)^{1/2}\frac{x^-_q - x^+_q}{x^-_p - x^+_q}\frac{\eta_p}{\eta_q}\,, & D^{\mathrm{LL}}_{pq} &= \left(\frac{x^+_q}{x^-_q}\right)^{1/2}\frac{x^-_p - x^-_q}{x^-_p - x^+_q}\,,\\
E^{\mathrm{LL}}_{pq} &= \frac{x^-_p - x^+_p}{x^-_p - x^+_q}\frac{\eta_q}{\eta_p}, & F^{\mathrm{LL}}_{pq} &= -\left(\frac{x^-_p}{x^+_p}\frac{x^+_q}{x^-_q}\right)^{1/2}\frac{x^+_p - x^-_q}{x^-_p - x^+_q}\,.
\end{aligned} \tag{2.3.33}$$

The result is written in terms of the Zhukovski variables introduced in Section 2.3.1. It must hold for any value of the masses $|m|$ that appear in Zhukovski variables, and is also valid for the scattering of particles of different masses.

In the Right-Right (RR) sector, we would find exactly the same formulas owing to LR symmetry, which implies that parameterisations coincide in the LL and RR sectors.

For the case of left–right scattering of two particles with equal masses, it turns out that requiring just invariance under the symmetry algebra together with the unitarity gives two

physically distinct solutions. One solution gives

$$\text{(T):}\quad \langle \mathcal{X}^{\rm L}\mathcal{Y}^{\rm R}|\,\check{\rm S}_{pq}\,|\mathcal{X}^{\rm L}\mathcal{Y}^{\rm R}\rangle = 0 \qquad \textbf{and} \qquad \langle \mathcal{X}^{\rm L}\mathcal{Y}^{\rm R}|\,\check{\rm S}_{pq}\,|\mathcal{Y}^{\rm R}\mathcal{X}^{\rm L}\rangle \neq 0\,, \tag{2.3.34}$$

and similarly for $L \leftrightarrow R$, whereas the other gives

$$\text{(R):}\quad \langle \mathcal{X}^{\rm L}\mathcal{Y}^{\rm R}|\,\check{\rm S}_{pq}\,|\mathcal{X}^{\rm L}\mathcal{Y}^{\rm R}\rangle \neq 0 \qquad \textbf{and} \qquad \langle \mathcal{X}^{\rm L}\mathcal{Y}^{\rm R}|\,\check{\rm S}_{pq}\,|\mathcal{Y}^{\rm R}\mathcal{X}^{\rm L}\rangle = 0\,, \tag{2.3.35}$$

with $\mathcal{X}$ and $\mathcal{X}$ being two generic excitations. The case indicated by (T) corresponds to *pure-transmission* of the target-space chirality, while (R) corresponds to *pure-reflection* [111]. Compatibility with perturbative results then forces us to choose the pure-transmission S-matrix[29]

$$\begin{aligned}
\check{\rm S}^{\rm LR}\,|\phi_p^{\rm L}\phi_q^{\rm R}\rangle &= A^{\rm LR}_{pq}\,|\phi_q^{\rm R}\phi_p^{\rm L}\rangle + B^{\rm LR}_{pq}\,|\psi_q^{\rm R}\psi_p^{\rm L}\rangle\,, & \check{\rm S}^{\rm LR}\,|\phi_p^{\rm L}\psi_q^{\rm R}\rangle &= C^{\rm LR}_{pq}\,|\psi_q^{\rm R}\phi_p^{\rm L}\rangle\,,\\
\check{\rm S}^{\rm LR}\,|\psi_p^{\rm L}\psi_q^{\rm R}\rangle &= E^{\rm LR}_{pq}\,|\psi_q^{\rm R}\psi_p^{\rm L}\rangle + F^{\rm LR}_{pq}\,|\phi_q^{\rm R}\phi_p^{\rm L}\rangle\,, & \check{\rm S}^{\rm LR}\,|\psi_p^{\rm L}\phi_q^{\rm R}\rangle &= D^{\rm LR}_{pq}\,|\phi_q^{\rm R}\psi_p^{\rm L}\rangle\,,
\end{aligned} \tag{2.3.36}$$

where the scattering elements are parametrised in terms of Zhukovski variables

$$\begin{aligned}
A^{\rm LR}_{pq} &= \zeta_{pq}\left(\frac{x_p^+}{x_p^-}\right)^{1/2}\frac{1-\frac{1}{x_p^+x_q^-}}{1-\frac{1}{x_p^-x_q^-}}\,, & B^{\rm LR}_{pq} &= -\frac{2i}{h}\left(\frac{x_p^-}{x_p^+}\frac{x_q^+}{x_q^-}\right)^{1/2}\frac{\eta_p\eta_q}{x_p^-x_q^+}\frac{\zeta_{pq}}{1-\frac{1}{x_p^-x_q^-}}\,,\\
C^{\rm LR}_{pq} &= \zeta_{pq}\,, & D^{\rm LR}_{pq} &= \zeta_{pq}\left(\frac{x_p^+}{x_p^-}\frac{x_q^+}{x_q^-}\right)^{1/2}\frac{1-\frac{1}{x_p^+x_q^+}}{1-\frac{1}{x_p^-x_q^-}}\,,\\
E^{\rm LR}_{pq} &= -\zeta_{pq}\left(\frac{x_q^+}{x_q^-}\right)^{1/2}\frac{1-\frac{1}{x_p^-x_q^+}}{1-\frac{1}{x_p^-x_q^-}}\,, & F^{\rm LR}_{pq} &= \frac{2i}{h}\left(\frac{x_p^+}{x_p^-}\frac{x_q^+}{x_q^-}\right)^{1/2}\frac{\eta_p\eta_q}{x_p^+x_q^+}\frac{\zeta_{pq}}{1-\frac{1}{x_p^-x_q^-}}\,.
\end{aligned} \tag{2.3.37}$$

Here, we have introduced a convenient factor

$$\zeta_{pq} = \left(\frac{x_p^+}{x_p^-}\right)^{-1/4}\left(\frac{x_q^+}{x_q^-}\right)^{-1/4}\left(\frac{1-\frac{1}{x_p^-x_q^-}}{1-\frac{1}{x_p^+x_q^+}}\right)^{1/2}\,. \tag{2.3.38}$$

Similarly, an S-matrix $\check{\rm S}^{\rm RL}$ can be obtained by swapping the labels L and R in (2.3.36). Using LR-symmetry, one arrives at the parametrisation which is the same as above with $A^{\rm RL}_{pq} = A^{\rm LR}_{pq}$ and so on.

The pure R-R S-matrix as a tensor product

From previous sections we know that we can write $\mathfrak{psu}(1|1)^4_{c.e.}$ S-matrix as a tensor product of two $\mathfrak{psu}(1|1)^2_{c.e.}$

$$\check{\rm S}^{\rm I\,J}_{\mathfrak{psu}(1|1)^4} = S_0^{\rm I\,J}\check{\rm S}^{\rm I\,J}_{\mathfrak{su}(1|1)^2}\check{\otimes}\check{\rm S}^{\rm I\,J}_{\mathfrak{su}(1|1)^2}\,, \tag{2.3.39}$$

where ${\rm I},{\rm J} = L$ or R, and $\check{\otimes}$ is the graded tensor product

$$(\mathcal{A}\check{\otimes}\mathcal{B})^{KK',LL'}_{MM',NN'} = (-1)^{\epsilon_{M'}\epsilon_N+\epsilon_L\epsilon_{K'}}\mathcal{A}^{KL}_{MN}\mathcal{B}^{K'L'}_{M'N'}\,, \tag{2.3.40}$$

where ϵ takes the value of 1 for fermions and 0 for bosons. The term $S_0^{\rm I\,J}$ is a scalar.

[29] Moreover, only this choice gives an S-matrix that satisfies the Yang-Baxter equation.

Generally, there might be four different $S_0^{\mathrm{I\,J}}$. However, using LR-symmetry we can deduce that $S_0^{\mathrm{LL}} = S_0^{\mathrm{RR}}$ and $S_0^{\mathrm{LR}} = S_0^{\mathrm{RL}}$ so we have just two independent factors[30].

It is common to normalise these scalar factors. One of the choices is

$$S_0^{\mathrm{LL}}(x_p^{\pm}, x_q^{\pm}) = \frac{x_p^+}{x_p^-}\,\frac{x_q^-}{x_q^+}\,\frac{x_p^- - x_q^+}{x_p^+ - x_q^-}\,\frac{1-\frac{1}{x_p^- x_q^+}}{1-\frac{1}{x_p^+ x_q^-}}\,\frac{1}{\left(\sigma_{pq}^{\bullet\bullet}\right)^2}, \tag{2.3.41}$$

where $\sigma_{pq}^{\bullet\bullet}$ is called *dressing factor*. It cannot be fixed by the symmetry algebra but the crossing equations, as in the AdS $-$ 5 case in Section 2.2.5. The idea behind this particular normalisation is to account for the bound states expected in the spectrum. The existence of a bound state can be predicted by investigating when the generic long multiparticle representation becomes short. For $\mathfrak{psu}(1|1)^4_{c.e.}$ this occurs for momenta p and q such that $x_p^+ = x_q^-$ or $x_p^- = x_q^+$. In the first case, one can find that $|Y_p^L Y_q^L\rangle$ survives in the tensor product of two left massive modules. In the second, – $|Z_p^L Z_q^L\rangle$. We expect only one of them to appear in the spectrum, and we choose it to be $|Y_p^L Y_q^L\rangle$. Thus, we include $x_p^+ - x_q^-$ to produce a pole and $x_p^- - x_q^+$ to produce a zero in a normalisation. Owing to the LR symmetry, the case of two right massive excitations is equivalent.

For the scattering of a Left excitation with a Right one, the preferred normalisation is

$$S_0^{\mathrm{LR}}(x_p^{\pm}, x_q^{\pm}) = \left(\frac{x_p^+}{x_p^-}\right)^{1/2}\left(\frac{x_q^+}{x_q^-}\right)^{-1/2}\frac{1-\frac{1}{x_p^- x_q^+}}{1-\frac{1}{x_p^+ x_q^-}}\,\frac{1}{\left(\tilde{\sigma}_{pq}^{\bullet\bullet}\right)^2}\,, \tag{2.3.42}$$

where a new dressing factor $\tilde{\sigma}_{pq}^{\bullet\bullet}$ is introduced, and again, due to the LR symmetry, the right-left sector should be the same.

Finally, Unitarity and the Yang-Baxter equation put some constraints on the form of introduced factors $\sigma_{pq}^{\bullet\bullet}$ and $\tilde{\sigma}_{pq}^{\bullet\bullet}$

$$\sigma_{qp}^{\bullet\bullet} = \left(\sigma_{pq}^{\bullet\bullet}\right)^* = \frac{1}{\sigma_{pq}^{\bullet\bullet}}\,, \qquad \tilde{\sigma}_{qp}^{\bullet\bullet} = \left(\tilde{\sigma}_{pq}^{\bullet\bullet}\right)^* = \frac{1}{\tilde{\sigma}_{pq}^{\bullet\bullet}}\,. \tag{2.3.43}$$

We see that the dressing factors in the massive case can be written as exponentials of anti-symmetric functions of the two momenta, and for physical momenta, they take values on the unit circle.

2.3.3 Crossing Symmetry and Dressing Factors

Physical region

Before moving further, it is worth briefly considering how crossing transformation transforms the S-matrix in the relativistic case[31]. In two-dimensional Lorentz-invariant models, it is possible to express the particle momentum and energy through a rapidity variable θ parametrising energy and momentum

$$p = m\sinh\theta\,, \quad H = m\cosh\theta\,, \tag{2.3.44}$$

[30]The two double dots in $\sigma_{pq}^{\bullet\bullet}$ and $\tilde{\sigma}_{pq}^{\bullet\bullet}$ denote that this dressing factor belongs to the S-matrix responsible for scattering two massive excitations (massive-massive). One can also define massive-massless $\sigma_{pq}^{\bullet\circ}$ or massless-massless $\sigma_{pq}^{\circ\circ}$ factors, but we will not consider them here.

[31]See [114] for a more detailed discussion.

so that the relativistic dispersion relation is trivially satisfied

$$H^2 - p^2 = m^2 \,. \tag{2.3.45}$$

For two-particle scattering, the S-matrix is a function of the momenta of two particles, $\mathsf{S}(p_1, p_2)$. The Lorentz invariance condition forces the S-matrix to be of the difference form when expressed in terms of rapidities

$$\mathsf{S}(p_1, p_2) = \mathsf{S}(\theta_1 - \theta_2) \,. \tag{2.3.46}$$

Alternatively, it can be written in terms of the Mandelstam variable

$$s = (p_1 + p_2)^2 = 2m^2(1 + \cosh(\theta_1 - \theta_2)) \,. \tag{2.3.47}$$

In physical processes, s is real and $s > 4m$, but it is possible to analytically continue the S-matrix to the whole complex plane except cuts along $s > 4m$ and $s < 0$[32]. Furthermore, we can consider an analytic continuation through the cuts to the next Riemann sheet, which turns out to correspond to anti-particles[33]. The region that describes the physical particles is conventionally chosen to be constrained to the first Riemann sheet by branch cuts. However, we are free to exchange particle and anti-particle regions without any change of physics, as they are equivalent mathematically.

Now, the intuition from the Lorentz-invariant theory example should be applied to the theory at hand.

First, one can notice periodicity in p in all the formulas (2.3.19) and (2.3.15). Therefore, it is natural to restrict the allowed region for p as $0 \leq p < 2\pi$. If we look at the plot of the Zhukovski variables for $m = 0$, given in figure 2.2, we notice that the $m = 0$ contour is a circle, which naturally captures the periodicity of the momentum.

Another observation we can make is that the transformation $x^\pm \to 1/x^\pm$ changes the sign of the energy (2.3.23) and momentum (2.3.20)

$$p \to \bar{p} = -p, \quad E \to \bar{E} = -E \,, \tag{2.3.48}$$

but leaves the constraint on the Zhukovski variables (2.3.20) intact. This suggests that the contour for $m = 0$ excitations is a branch cut of the S-matrix. Now, there is a choice similar to the one in Lorentz invariant theory: we can either choose the physical amplitudes for $m = 0$ excitations to lie just outside or inside the circle. If we decide to use the former, it is natural to extend the physical region to the entire complex plane outside the circle.

Under the map $x \to \frac{1}{x}$, the region inside the circle is entirely equivalent to the region outside. Therefore, designating one of them as a physical region is merely a matter of convention. The region, dual under this map, is the *crossed* region.

Crossing transformation

We have seen that under the transformation

$$x_p^\pm \to x_{\bar{p}}^\pm = \frac{1}{x_p^\pm} \,, \tag{2.3.49}$$

[32]The cut $s < 0$ comes from the equivalence of the scattering in s- and t-channels relating $\mathsf{S}(s) = \mathsf{S}(4m^2 - s)$.

[33]In theories with more particle scattering, there may be more than two Riemann sheets. Furthermore, more general theories (e.g. for the sine-Gordon model) may have infinitely many sheets.

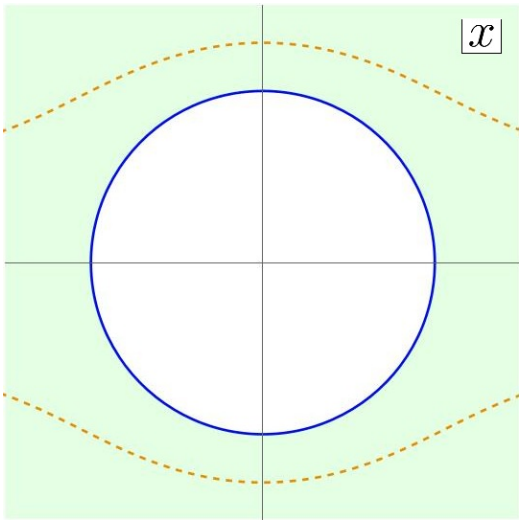


Figure 2.2: Plots of Zhukovski variables for $m = 0$ (solid) and $m = 1$ (dashed) for real momenta $p \in [0, 2\pi]$. The variable x^+ has positive real part, and x^+ has negative real part. The region shaded in green is the physical region.

momentum and energy transform as

$$p \to \bar{p} = -p, \quad E \to \bar{E} = -E\,, \tag{2.3.50}$$

which corresponds to the particle-to-anti-particle transformation, or in other words, the *crossing transformation*. Technically, if we want to perform a crossing transformation on an expression, we need to analytically continue it in the Zhukovski variable from x to $1/x$. In this process, we might cross different cuts and discontinuities. In particular, for the S-matrix, we expect cuts along massless Zhukovski contours. One of the choices to go from the physical region outside the circle to the crossed region inside the circle is depicted in Figure 2.3. Thus, to be consistent, we need to choose a particular crossing path, so we choose to use the one in figure 2.3.

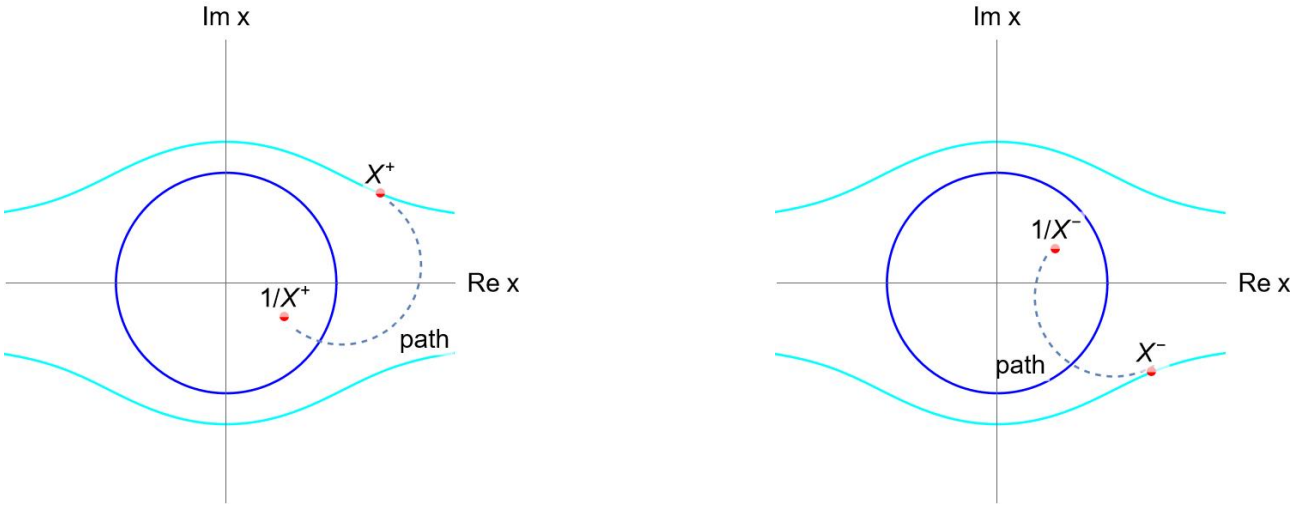


Figure 2.3: The crossing transformation path (dashed) for massive variables in the $x^\pm$ planes. Blue contour corresponds to $m = 0$ Zhukovski, Cyan contour – $m = 1$ x^+/x^- Zhukovski.

For the theory where particles and antiparticles coincide, elements of the S-matrix related by a particle-to-antiparticle transformation are equivalent up to performing an analytic con-

tinuation. This transformation exchanges the branches of the dispersion relation, in other words, changes the sign of energy and momentum. However, this is not true for the theory we consider. In our case, we should supplement analytic continuation with a map which sends the particle representation into the antiparticle representation. We have seen this for the AdS_5 strings in section 2.2.5.

To define the crossing equation, such as (2.2.65), we need to define a charge-conjugation matrix $\mathcal{C}$, which exchanges the Left and Right representations in the $\mathfrak{su}(1|1)^2_{c.e.}$ basis $B = (\phi^{\mathrm{L}}, \psi^{\mathrm{L}}, \phi^{\mathrm{R}}, \psi^{\mathrm{R}})$, $\mathcal{C}$ can be chosen as

$$\mathcal{C} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -i \\ 1 & 0 & 0 & 0 \\ 0 & -i & 0 & 0 \end{pmatrix},$$

where the matrix Σ is defined as a diagonal matrix taking the value $+1$ for bosons and -1 for fermions. In this way, $\mathcal{C}^2 = \Sigma$ and $\mathcal{C}^{-1} = \mathcal{C}^\dagger$.

The form of the charge conjugation matrix is not unique. There are several degrees of freedom which can be reflected in the form of charge conjugation matrix. First, each of the $\mathfrak{su}(1|1)_{\mathrm{L/R}}$ factors in $\mathfrak{su}(1|1)^2_{\mathrm{c.e.}}$ admits a continuous one-parameter family of automorphisms, that is a $\mathrm{U}(1)$ phase rotation acting on the supercharges:

$$\mathbf{q} \to e^{i\phi}\,\mathbf{q}\,, \quad \overline{\mathbf{q}} \to e^{-i\phi}\,\overline{\mathbf{q}}\,.$$

Physically, this $\mathrm{U}(1)$ symmetry corresponds to a global phase freedom in defining the fermionic generators. This freedom can be reflected in the charge conjugation matrix $\mathcal{C}$ [101]

$$\mathcal{C} = \begin{pmatrix} 0 & 0 & \xi^{\mathrm{RL}} & 0 \\ 0 & 0 & 0 & -i\xi^{\mathrm{RL}} \\ \xi^{\mathrm{LR}} & 0 & 0 & 0 \\ 0 & -i\xi^{\mathrm{LR}} & 0 & 0 \end{pmatrix},$$

where $\xi^{\mathrm{RL}} = e^{i\phi}$ and $\xi^{\mathrm{LR}} = e^{-i\phi}$. The standard choice is to fix $\xi^{\mathrm{LR}} = \xi^{\mathrm{RL}} = 1$ so that $\mathcal{C}^2 = \Sigma$ and $C^{-1} = C^\dagger$. Second, there is a basis freedom in $\mathfrak{su}(1|1)^2_{\mathrm{c.e.}}$ representations ρ_{L}, ρ_{R}, $\tilde{\rho}_{\mathrm{L}}$ and $\tilde{\rho}_{\mathrm{R}}$. For instance, we are free to rescale the basis states as $|\phi\rangle \to \lambda\,|\phi\rangle$ and $|\psi\rangle \to \mu\,|\psi\rangle$. In fact, this basis freedom can offset the action of the $\mathrm{U}(1)$ automorphism we have seen above. Finally, there is a choice of the overall normalisation of $\mathcal{C}$.

However, all of these freedoms are not physical and result in the same crossing equations. In the basis of $\mathfrak{psu}(1|1)^4_{c.e.}$ representation

$$\{Y^{\mathrm{L}}, \eta^{\mathrm{L}1}, \eta^{\mathrm{L}2}, Z^{\mathrm{L}}\} \oplus \{Y^{\mathrm{R}}, \eta^{\mathrm{R}1}, \eta^{\mathrm{R}2}, Z^{\mathrm{R}}\}\,, \tag{2.3.51}$$

the charge conjugation matrix takes the form

$$\mathscr{C}_p = \left(\begin{array}{cccc|cccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -i & 0 \\ 0 & 0 & 0 & 0 & 0 & i & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & i & 0 & 0 & 0 & 0 & 0 \\ 0 & -i & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right). \tag{2.3.52}$$

Following [112], crossing equation can be derived using $\mathbb{R}_{pq} = \Pi\check{\mathbb{S}}_{pq}$ if we notice that the objects

$$\mathbb{R}^{-1}_{pq} \quad \textbf{and} \quad \mathscr{C}_{p(1)}\mathbb{R}^{t_1}_{\bar{p}q}\mathscr{C}^{\dagger}_{p(1)} \tag{2.3.53}$$

where ^{t_1} denotes transposition on the first space, have the same invariance property with respect to all supercharges $\mathbf{q}$. We can therefore conclude

$$\mathscr{C}_{p(1)}\mathbb{R}^{t_1}_{\bar{p}q}\mathscr{C}^{\dagger}_{p(1)} = \mathbb{R}^{-1}_{pq}\,, \tag{2.3.54}$$

which is a *crossing equation*, equivalent to (2.2.65), obtained from the Hopf algebra considerations. This requirement means that the scalar factors $\sigma^{\bullet\bullet}_{pq}$ and $\tilde{\sigma}^{\bullet\bullet}_{pq}$ are not arbitrary, but satisfy the analyticity requirements coming from (2.3.54) or its analogue for crossing for q (by transposing $\mathbb{R}$ in the second space). The whole set of equations reduces to equations just for the dressing factors

$$\begin{aligned}
\left(\sigma^{\bullet\bullet}_{pq}\right)^2 \left(\tilde{\sigma}^{\bullet\bullet}_{\bar{p}q}\right)^2 &= \left(\frac{x^-_q}{x^+_q}\right)^2 \frac{(x^-_p - x^+_q)^2}{(x^-_p - x^-_q)(x^+_p - x^+_q)} \frac{1-\frac{1}{x^-_p x^+_q}}{1-\frac{1}{x^+_p x^-_q}}, \\
\left(\sigma^{\bullet\bullet}_{\bar{p}q}\right)^2 \left(\tilde{\sigma}^{\bullet\bullet}_{pq}\right)^2 &= \left(\frac{x^-_q}{x^+_q}\right)^2 \frac{\left(1-\frac{1}{x^+_p x^+_q}\right)\left(1-\frac{1}{x^-_p x^-_q}\right)}{\left(1-\frac{1}{x^+_p x^-_q}\right)^2} \frac{x^-_p - x^+_q}{x^+_p - x^-_q}\,.
\end{aligned} \tag{2.3.55}$$

However, there is still some freedom in choosing dressing factors: it is possible to add a CDD factor – solution to the homogeneous crossing equation [115]

$$\sigma^{\text{CDD}}_{pq}\tilde{\sigma}^{\text{CDD}}_{\bar{p}q} = 1\,, \quad \sigma^{\text{CDD}}_{\bar{p}q}\tilde{\sigma}^{\text{CDD}}_{pq} = 1\,. \tag{2.3.56}$$

However, such factors might modify the pole structure of the S-matrix. This is why the requirements discussed at the end of section 2.2.5 are important; they allow one to fix this ambiguity. Ultimately, the validity of the assumptions must be tested by comparing the S-matrix with perturbative calculations.

Originally, a proposal for pure R-R dressing factors was made and checked against perturbative computation in [116]. However, it was later modified in [29], [117]. We provide more details in the next section.

2.3.4 Solution of the crossing equations

In this section, we provide an overview of the crossing equation solution in the pure R-R case based on [116].

In (2.3.55), we introduce functions g_{12} and $\tilde{g}_{12}$ which will be useful later

$$g(x_1, x_2) = g_{12} = \left(\frac{x^-_2}{x^+_2}\right)^2 \frac{(x^-_1 - x^+_2)^2}{(x^-_1 - x^-_2)(x^+_1 - x^+_2)} \frac{1-\frac{1}{x^-_1 x^+_2}}{1-\frac{1}{x^+_1 x^-_2}}, \tag{2.3.57}$$

$$\tilde{g}(x_1, x_2) = \tilde{g}_{12} = \left(\frac{x^-_2}{x^+_2}\right)^2 \frac{(1-\frac{1}{x^+_1 x^+_2})(1-\frac{1}{x^-_1 x^-_2})}{(1-\frac{1}{x^+_1 x^-_2})^2} \frac{x^-_1 - x^+_2}{x^+_1 - x^-_2}. \tag{2.3.58}$$

Then we can define two new functions, called Sum and Difference phases

$$\sigma^+(x_1, x_2) = \sigma^{\bullet\bullet}(x_1, x_2)\tilde{\sigma}^{\bullet\bullet}(x_1, x_2), \quad \sigma^-(x_1, x_2) = \frac{\sigma^{\bullet\bullet}(x_1, x_2)}{\tilde{\sigma}^{\bullet\bullet}(x_1, x_2)}, \tag{2.3.59}$$

as well as two more functions, called BES and HL phases, defined by the equations

$$\sigma^{BES}(x_1,x_2)\sigma^{BES}(\bar{x}_1,x_2)=h(x_1,x_2)\equiv h_{12}\equiv\frac{x_2^-}{x_2^+}\frac{x_1^--x_2^+}{x_1^--x_2^-}\frac{1-\frac{1}{x_1^+x_2^+}}{1-\frac{1}{x_1^+x_2^-}} \tag{2.3.60}$$

for the BES phase, and

$$\sigma^{HL}(x_1,x_2)\sigma^{HL}(\bar{x}_1,x_2)=\sqrt{\frac{h_{12}}{h_{\bar{1}2}}}, \tag{2.3.61}$$

for the HL phase. Here, we introduce another function $h_{\bar{1}2}=h(\bar{x}_1,x_2)$.

This allows us to get an equation for σ^+ by using definition of σ^+ (2.3.62) and taking the product the of two lines of (2.3.55)

$$\sigma^+(x_1,x_2)^2\sigma^+(\bar{x}_1,x_2)^2=g_{12}\tilde{g}_{12}. \tag{2.3.62}$$

Then we notice that using the following equality[34]

$$\frac{1-\frac{1}{x^+y^+}}{x^--y^-}=\frac{1-\frac{1}{x^-y^-}}{x^+-y^+}, \tag{2.3.63}$$

using which one can check that

$$g_{12}\tilde{g}_{\bar{1}2}=\frac{(h_{12})^3}{(h_{\bar{1}2})}\,. \tag{2.3.64}$$

This fact allows us to write the solution for (2.3.62)

$$\sigma^+_{12}=\frac{(\sigma^{BES}_{12})^2}{\sigma^{HL}_{12}}, \tag{2.3.65}$$

that is, σ^+ is given in terms of the BES and HL phases (see [116]).

If we take the ratio of the two lines of (2.3.55), we get an equation for σ^-

$$\frac{\sigma^-(x_1,x_2)}{\sigma^-(\bar{x}_1,x_2)^2}=\frac{\tilde{g}_{12}}{g_{12}}=\frac{l^-(x_1^+,x_2^-)l^-(x_1^-,x_2^+)}{l^-(x_1^+,x_2^+)l^-(x_1^-,x_2^-)}\,, \tag{2.3.66}$$

where $l^-(x,y)=(x-y)(1-\frac{1}{xy})$. Therefore, σ^- can be written in terms of χ^- (see [116]).

It is convenient to define the *dressing phase* instead of the dressing factor

$$\sigma(x,y)=e^{i\theta(x^\pm,y^\pm)}\quad\tilde{\sigma}(x,y)=e^{i\tilde{\theta}(x^\pm,y^\pm)}\,. \tag{2.3.67}$$

Similarly to AdS_5 case (2.2.70), dressing phases in AdS_3 have a very useful representation

$$\theta(x^\pm,y^\pm)=\chi(x^+,y^+)+\chi(x^-,y^-)-\chi(x^+,y^-)-\chi(x^-,y^+)\,, \tag{2.3.68}$$

$$\tilde{\theta}(x^\pm,y^\pm)=\tilde{\chi}(x^+,y^+)+\tilde{\chi}(x^-,y^-)-\tilde{\chi}(x^+,y^-)-\tilde{\chi}(x^-,y^+)\,. \tag{2.3.69}$$

[34]This is derived from the shortening condition for Zhukovski, see [116].

Odd part of a dressing phase

To reduce (2.3.55) to something simpler, we can perform the following trick.

In (2.3.55), one can perform crossing one more time to obtain additional equations for the dressing phase

$$\left(\frac{\sigma(\bar{\bar{x}},y)}{\sigma(x,y)}\right)^2=\left(\frac{x^+-y^+}{x^+-y^-}\frac{x^--y^-}{x^--y^+}\right)^2, \tag{2.3.70}$$

and

$$\left(\frac{\tilde{\sigma}(\bar{\bar{x}},y)}{\tilde{\sigma}(x,y)}\right)^2=\left(\frac{1-1/(x^-y^+)}{1-1/(x^+y^+)}\frac{1-1/(x^+y^-)}{1-1/(x^-y^-)}\right)^2. \tag{2.3.71}$$

If we separate dressing phases in the following way

$$\sigma(x,y)=\sigma^{even}(x,y)\sigma^{odd}(x,y),\quad \tilde{\sigma}(x,y)=\tilde{\sigma}^{even}(x,y)\tilde{\sigma}^{odd}(x,y), \tag{2.3.72}$$

where σ^{even} means that this function is even under double-crossing, i.e. $\sigma^{even}(\bar{\bar{x}},y)=\sigma^{even}(x,y)$, and σ^{odd} is odd under double-crossing, i.e. $\sigma^{odd}(\bar{\bar{x}},y)\neq\sigma^{odd}(x,y)$; this allows us to write crossing equation for odd part of the dressing phase specifically

$$\sigma^{odd}(x,y)^2\tilde{\sigma}^{odd}(\bar{x},y)^2=\left(\frac{x^+-y^-}{x^+-y^+}\frac{x^--y^+}{x^--y^-}\right), \tag{2.3.73}$$

$$\sigma^{odd}(\bar{x},y)^2\tilde{\sigma}^{odd}(x,y)^2=\left(\frac{1/x^+-y^+}{1/x^+-y^-}\frac{1/x^--y^-}{1/x^--y^+}\right). \tag{2.3.74}$$

Such a representation in terms of even and odd phases is especially important for the mixed-flux case considered in the next chapter.

The full solution

We combine all these pieces to get the all-loop expression for dressing phases in terms of χ

$$\chi=\chi^{BES}+\frac{1}{2}(-\chi^{HL}+\chi^-) \tag{2.3.75}$$

$$\tilde{\chi}=\chi^{BES}+\frac{1}{2}(-\chi^{HL}-\chi^-) \tag{2.3.76}$$

where χ^{BES} solves $AdS_5\times S^5$ crossing and has a strong-coupling limit expansion ($h\to\infty$):

$$\chi^{BES}=\underbrace{\chi^{AFS}}_{\sim h^0}+\underbrace{\chi^{HL}}_{\sim 1/h}+\dots, \tag{2.3.77}$$

where χ^{HL} is the odd part of the crossing in $AdS_5\times S^5$. To separate an odd part of AdS_3 crossing, we subtract χ^{HL} from χ^{BES}

$$\chi=\chi^{BES}-\chi^{HL}+\underbrace{\frac{1}{2}(\chi^{HL}+\chi^-)}_{\text{odd part}} \tag{2.3.78}$$

$$\tilde{\chi}=\chi^{BES}-\chi^{HL}+\underbrace{\frac{1}{2}(\chi^{HL}-\chi^-)}_{\text{odd part}} \tag{2.3.79}$$

where

$$\begin{aligned}\chi^{HL} &= (\oint\limits_{\curvearrowleft} - \oint\limits_{\curvearrowright})\frac{dw}{4\pi}\frac{1}{x-w}\left(\log(y-w)-\log(y-\frac{1}{w})\right),\\ \chi^{-} &= (\oint\limits_{\curvearrowleft} - \oint\limits_{\curvearrowright})\frac{dw}{8\pi}\frac{1}{x-w}\left(\log(y-w)+\log(1-\frac{1}{yw})\right)-(x\leftrightarrow y)\,.\end{aligned} \tag{2.3.80}$$

To sum up, σ^{BES}, σ^{HL} and σ^{-} together fully define a solution (2.3.55) according to [116]. However, as mentioned before, this was later modified in [29], [117].

In [117], a new set of variables γ, with properties resembling relativistic rapidity variables, was introduced, and parity as a symmetry of the dressing factor, which was overlooked in [116], was considered. The resulting dressing phase differed from the original by a CDD factor.

A simple modification of (2.3.80) can be made by adding the following term[35]

$$\delta\phi = \frac{1}{16\pi}\log\frac{(x+1)^2}{x}\log\frac{(y-1)^2}{y} - \frac{1}{16\pi}\log\frac{(x-1)^2}{x}\log\frac{(y+1)^2}{y}\,, \tag{2.3.81}$$

which cancels out the unphysical branch cuts of the odd part of the solution $\chi_{odd} = \frac{1}{2}(\chi^{HL} + \chi^{-})$ and $\tilde{\chi}_{odd} = \frac{1}{2}(\chi^{HL} - \chi^{-})$. This modification was presented in [29]. The result leaves the contributions crossing unchanged but restores the parity symmetry and makes the entire solution agree with [117]. It is the odd part of the solution (2.3.78) which gets modified

$$\begin{aligned}\chi^{(\text{mod})}_{\text{odd}} &= \chi_{\text{odd}} + \delta\phi\,,\\ \tilde{\chi}^{(\text{mod})}_{\text{odd}} &= \tilde{\chi}_{\text{odd}} - \delta\phi\,.\end{aligned} \tag{2.3.82}$$

We can further rewrite in the following form

$$\begin{aligned}\chi^{(\text{mod})}_{\text{odd}}(x,y) &= +\oint\frac{dz}{8\pi}\frac{\log\frac{(y-z)^2}{y}}{x-z} + \frac{1}{16\pi}\log\frac{(x+1)^2}{(x-1)^2}\Big(\log\frac{(y+1)^2}{y}+\log\frac{(y-1)^2}{y}\Big),\\ \tilde{\chi}^{(\text{mod})}_{\text{odd}}(x,y) &= -\oint\frac{dz}{8\pi}\frac{\log\frac{(y-z)^2}{y}}{\frac{1}{x}-z} - \frac{1}{16\pi}\log\frac{(x+1)^2}{(x-1)^2}\Big(\log\frac{(y+1)^2}{y}+\log\frac{(y-1)^2}{y}\Big),\end{aligned} \tag{2.3.83}$$

where integration is taken along the upper and lower parts unit circle, clockwise for the lower part and anti-clockwise for the upper part.

For both $|x|, |y| > 1$, we can deform the circular contours to the real line $z \in [-1, 1]$ and get the following expressions

$$\begin{aligned}\chi^{(\text{mod})}_{\text{odd}}(x,y) &= +\int_1^{-1}\frac{dz}{2\pi}\frac{\log(y-z)}{x-z} + \frac{1}{4\pi}\log\frac{x+1}{x-1}\Big(\log(y+1)+\log(y-1)\Big)\,,\\ \tilde{\chi}^{(\text{mod})}_{\text{odd}}(x,y) &= -\int_1^{-1}\frac{dz}{2\pi}\frac{\log(y-z)}{\frac{1}{x}-z} - \frac{1}{4\pi}\log\frac{x+1}{x-1}\Big(\log(y+1)+\log(y-1)\Big)\\ &\quad -\frac{i}{2}\,\text{sign}(\text{Im}\,x)\log\big(y-\frac{1}{x}\big)\,.\end{aligned} \tag{2.3.84}$$

Notice the additional term at the end of the expression for $\tilde{\chi}^{(\text{mod})}$. The source for it is the pole we pick up at $1/x$ whose residue depends on which half-plane x is in.

[35] This was suggested in an unpublished work by Andrea Cavaglia and Simon Ekhammer.

The form of the solution may differ. For instance, the solution in [117] is equivalent to the one above but has a significantly different form. One of the ways to see their equivalence is to look at large x and y expansions, which should be of the form (2.2.68). If the solutions are the same, the coefficients in this expansion should be the same. One can check that the expansion of (2.3.84)[36] has the following form

$$\begin{aligned}
\chi_{\text{odd}}^{(\text{mod})} &= \frac{1}{4\pi}\sum_{m,n=1}^{\infty}\frac{x^{-m}y^{-n}}{mn}\left[\frac{m-n}{m+n}\big(1-(-1)^{n+m}\big)+(-1)^m-(-1)^n\right], \\
\tilde{\chi}_{\text{odd}}^{(\text{mod})} &= +\frac{1}{4\pi}\sum_{\substack{m,n=1\\ m\neq n}}^{\infty}\frac{x^{-m}y^{-n}}{mn}\left[\frac{n+m}{n-m}\big(1-(-1)^{m-n}\big)+(-1)^n-(-1)^m\right] \\
&\quad+\frac{1}{2\pi}\sum_{n=1}^{\infty}\frac{1-(-1)^{-n}}{n^2}y^{-n}\,,
\end{aligned} \tag{2.3.85}$$

which agrees with that of [117] up to terms not relevant in the full phase (2.2.70).

Now, we shall generalise it for the mixed-flux case in the next chapter.

[36] This was computed by Suvajit Majumder and Bogdan Stefanski in an unpublished work.

Chapter 3

Integrability in $\mathbf{AdS}_3 \times \mathbf{S}^3 \times \mathbf{T}^4$ with mixed-flux

In this chapter, we discuss the worldsheet S-matrix for $\mathrm{AdS}_3 \times \mathrm{S}^3 \times \mathrm{T}^4$ with mixed flux, focusing on its dressing factors. We start from the known facts about the symmetry algebra and its representations, which is then followed by a presentation of elements of the novel mixed-flux kinematics and construction of the previously unknown odd part of the massive dressing phase[1].

3.1 The $\mathfrak{su}(1|1)^2_{c.e.}$ Symmetry Algebra for the mixed-flux

In this section, we briefly consider how the symmetry algebra and its representations are modified compared to the pure R-R theory, and how it changes the kinematics of the theory.

The symmetry algebra for the mixed-flux theory can be obtained in the near-plane-wave limit from supercurrents truncated at the quadratic order in the fields. It has the same form as the pure R-R one obtained in section 2.3.1, but the coefficients in the supercurrent expressions are deformed. The construction of $\mathfrak{su}(1|1)^2_{c.e.}$ representations, and then the product of two $\mathfrak{su}(1|1)^2_{c.e.}$ representations into $\mathfrak{psu}(1|1)^4_{c.e.}$ one mostly repeats the case of pure R-R.

In the pure R-R case, the left and right representations are equivalent. This is not the case for the mixed-flux theory, where the left and right representations are different. Namely, the representation coefficients used in the expressions for the central charges (2.3.19) are modified [40]. Additionally, representation coefficients are no longer the same for the left and right $\mathfrak{su}(1|1)^2_{c.e.}$ representation, so we need to keep indices L and R everywhere. However, there is still a symmetry in choosing the label for the left or right representation, meaning we can interchange all left labels L with right labels R without the change of physics[2]. This is usually referred to as left–right symmetry (LR symmetry).

Central charges have the following expressions [40]

$$\mathbf{M} = \begin{cases} m + \bar{k}p & \text{left}, \\ -m + \bar{k}p & \text{right}, \\ \bar{k}p & \text{massless}, \end{cases} \tag{3.1.1}$$

[1]The material in this section is based on the collaborative study [29]. The author of the thesis is the main contributor to the analysis presented in section 3.3.

[2]Provided we change $\mathbf{M} \to -\mathbf{M}$ where needed.

$$\mathbf{H} = \begin{cases} \sqrt{(m + \bar{k}p)^2 + 4h^2 \sin^2(\frac{p}{2})} & \text{left,} \\ \sqrt{(m - \bar{k}p)^2 + 4h^2 \sin^2(\frac{p}{2})} & \text{right,} \\ \sqrt{\bar{k}^2 p^2 + 4h^2 \sin^2(\frac{p}{2})} & \text{massless,} \end{cases} \tag{3.1.2}$$

$$\begin{aligned} \mathbf{C} &= \zeta \frac{ih}{2}(e^{ip} - 1)\,, \\ \overline{\mathbf{C}} &= -\zeta \frac{ih}{2}(e^{-ip} - 1)\,, \end{aligned} \tag{3.1.3}$$

where $\bar{k} = \frac{k}{2\pi}$. It will be useful sometimes to use $\bar{k}_{\mathrm{I}}$ defined as $\bar{k}_{\mathrm{L}} = \bar{k}$ and $\bar{k}_{\mathrm{R}} = -\bar{k}$.

This leads to modifications of the dispersion relation derived from the shortening condition (2.3.14)

$$E_{\mathrm{I}}(p) = \sqrt{(m + \bar{k}_{\mathrm{I}} p)^2 + 4h^2 \sin^2\big(\frac{p}{2}\big)}\,, \quad I \in \{L, R\}\,, \tag{3.1.4}$$

The shortening condition of Zhukovski also changes

$$\frac{x^+_{\mathrm{I}p}}{x^-_{\mathrm{I}p}} = e^{ip}, \qquad x^+_{\mathrm{I}p} + \frac{1}{x^+_{\mathrm{I}p}} - x^-_{\mathrm{I}p} - \frac{1}{x^-_{\mathrm{I}p}} = \frac{2i\,(|m| + \bar{k}_{\mathrm{I}} p)}{h}\,, \quad I \in \{L, R\}\,. \tag{3.1.5}$$

These equations can be solved for $x^{\pm}_{\mathrm{I}}$ by setting

$$x^{\pm}_{\mathrm{I}p} = \frac{(|m| + \bar{k}_{\mathrm{I}} p) + E_{\mathrm{I}p}}{2h \sin(\frac{p}{2})} e^{\pm \frac{i}{2} p}\,, \quad I \in \{L, R\}\,, \tag{3.1.6}$$

with $m = \pm 1, 0$.

In terms of Zhukovski variables, the dispersion relation takes the form

$$E_I(p) = -\frac{ih}{2}\Big(x^+_I - \frac{1}{x^+_I} - x^-_I + \frac{1}{x^-_I}\Big)\,, \quad I \in \{L, R\}\,. \tag{3.1.7}$$

The major difference from the pure R-R case is that p is no longer confined to the region $(0, 2\pi)$ because the dispersion relation in (3.1.4) is not periodic in p. Therefore, there is no reason not to take $p \in \mathbb{R}$. The plot x^+_{L} for different values of m is shown in Figure 3.1[3]. We label the contours by the value of charge $\mathbf{M}$, which we call mass m, and the number of momentum regions, which can be $p \in (2\pi n, 2\pi(n+1))$, for $n \in \mathbb{Z}$.

One can notice that the dispersion relation (3.1.4) has the following periodicity property

$$m \to m + k \Longleftrightarrow p \to p \pm 2\pi \tag{3.1.8}$$

for the left and right representations, respectively. This periodicity property was first considered in [118] for the relativistic limit of the theory and later discussed in detail for the full theory in [29].

In the context of Zhukovsky variables, such periodicity is realised as equivalence between every line of mass $m + k$ with momentum $p \in [0, 2\pi]$ and every line with mass m and momentum $p \in [2\pi, 4\pi]$.

On a physical level, this feature can be interpreted as the existence of a singlet state, which implies the equivalence of the $m = k + 1$ bound states with momentum p and $m = 1$ states with momentum $p + 2\pi$. It can be demonstrated that the trivial scattering of such states

[3] We are using parameters $\bar{k} = \frac{5}{2\pi}$ and $h = 2$ for the plots everywhere in this work.

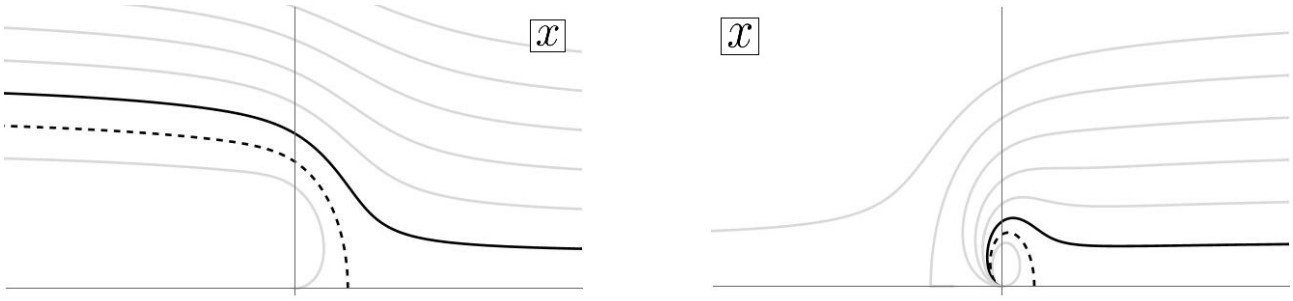


Figure 3.1: Plot of $x_{\rm L}^+$ (on the left) and $x_{\rm R}^+$ (on the right) contours for different values of masses and momenta in the $[0, 2\pi]$ region. The dashed line is the $m = 0$ contour, the solid line is for $m = 1$, the light grey line below the dashed $m = 0$ line is for $m = -1$, and the light grey lines above $m = 1$ are $m = 2, 3, \dots$ contours.

leads to crossing equations. Additionally, the existence of such a singlet state ensures that once we continue the momenta to the entire real line, the number of physical excitations is not inflated by an infinite number of identically charged states [29].

Furthermore, looking at (3.1.4), we can see that from the perspective of representations, the $m = k - 1$ left bound state at momentum $p - 2\pi$ is indistinguishable from $m = 1$ right excitation with momentum p. Thus, we can choose the left or right description depending on which is more convenient.

3.2 The Crossing Transformation

The application of the crossing transformation should change signs of the central charges (3.1.1)-(3.1.3). This can be achieved by

$$p \to \bar{p} = -p, \quad E_p \to E_{\bar{p}} = -E_p \,, \tag{3.2.1}$$

together with switching L and R indices. By looking at (3.1.7), one can notice how Zhukovski should transform

$$x_{\rm I}(p) \to \frac{1}{x_{\rm I}(\bar{p})} = \frac{1}{x_{\rm I}(-p)} = \frac{1}{x_{\bar{\rm I}}(p)}, \tag{3.2.2}$$

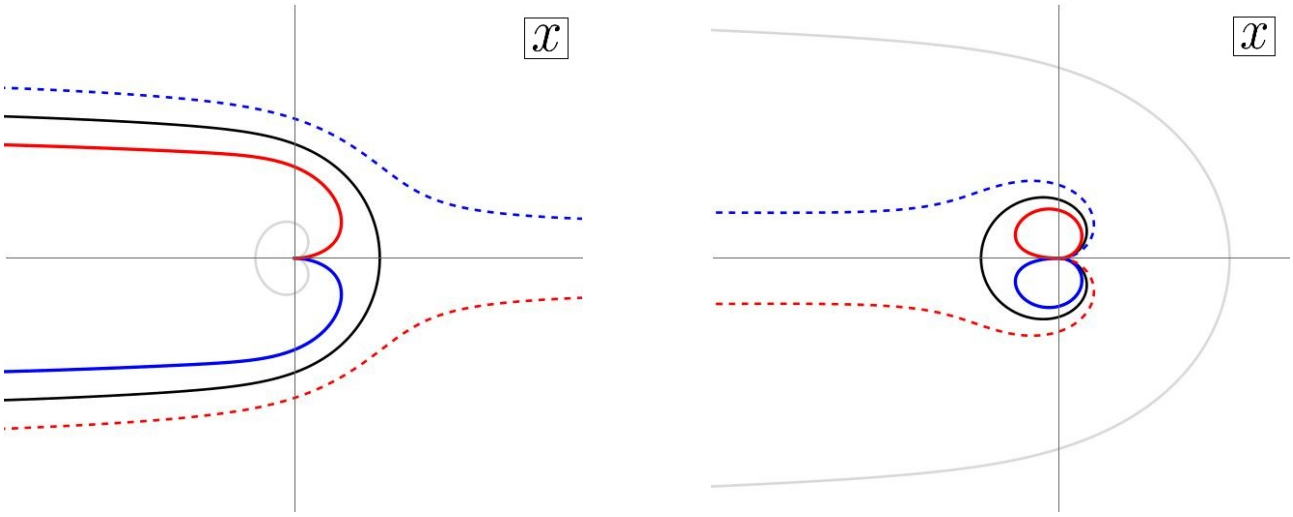
where $I = L, R$, and the bar over I means we should change L to R and vice versa. In the last equality we used (3.1.6) to replace $x_{\rm I}(-p)$ with $x_{\bar{\rm I}}(p)$.

To completely define a crossing transformation, we also need to identify the physical region.

However, identifying the physical region for the general value of momentum is more difficult than in the R-R case. If previously we choose the physical region outside the unit circle, which is a region where the Zhukovski variable can take values for physical $m = 0$ excitation, now, figure 3.1 contains only a part of the $m = 0$ contour with $p \in (0, 2\pi)$. If we continue it to $p > 2\pi$, we should end up on another sheet with $p \in (2\pi, 4\pi)$ where $m = 0$ contour will be at the same location as $m = k$ contour ($m = 2$ for figure 3.1) from $p \in (0, 2\pi)$. For $p \in (4\pi, 6\pi)$, $m = 0$ contour will be at the same location as $m = 2k$ contour ($m = 4$ for figure 3.1) and so on. The same process can be repeated if moving in the opposite direction in the momentum or considering Right Zhukovski instead of Left. Different sheets corresponding to different momenta regions are connected to each other via the log branch cut along the negative real axis (see [29] for more detailed discussion). Overall, there should

be an infinite number of separate sheets of $x_{\text{L}}^{\pm}/x_{\text{R}}^{\pm}$ with momentum $p \in [2\pi n, 2\pi(n+1)$ labelled by $n \in \mathbb{Z} \cup \{0\}$, each with its own Zhukovski contours.

Using the intuition from the pure R-R case, the contours $x_{\text{L}}^{\pm}/x_{\text{R}}^{\pm}$ with $m = 0$ should create a natural boundary for the physical and crossed regions. For instance, figure 3.2 shows $x_{\text{L}}^{\pm}$ with $m = 1$ in $p \in (0, 2\pi)$ and its crossed counterpart in subfigure (a); $x_{\text{L}}^{\pm}$ with $m = 1$ in $p \in (-2\pi, 0)$ and its crossed counterpart in subfigure (b). Other values of masses ($m > 1$) and other momentum regions give a similar picture.



(a) Momentum region $p \in (0, 2\pi)$.

(b) Momentum region $p \in (-2\pi, 0)$.

Figure 3.2: Plot of Zhukovski variable x_{L}^{+} (blue dashed) and x_{L}^{-} (red dashed), and the crossed Zhukovski variable $\frac{1}{x_{\text{R}}^{+}}$ (blue) and $\frac{1}{x_{\text{R}}^{-}}$ (red). We refer to the solid black contour in the left subfigure as the *Scallion* (S_{L}), and the solid black contour on the right subfigure as the *Kidney* (K_{L}).

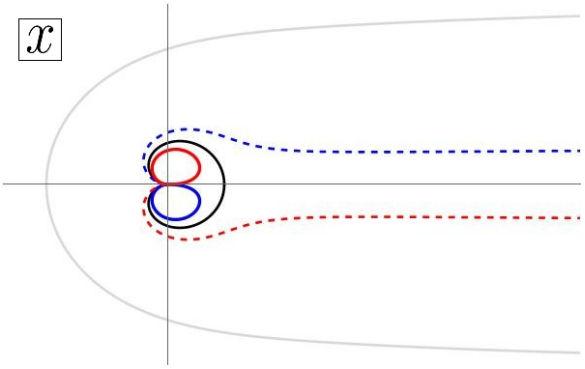


Figure 3.3: Plot of Zhukovski variable x_{R}^{+} (blue dashed) and x_{R}^{-} (red dashed), and the crossed Zhukovski variable $\frac{1}{x_{\text{L}}^{+}}$ (blue) and $\frac{1}{x_{\text{L}}^{-}}$ (red) in momentum region $p \in (0, 2\pi)$. We refer to the solid black contour as the *Kidney* (K_{R}).

The choice of the physical region in the regions with $p > 0$ and $p < -2\pi$ is rather simple. We can choose it to be the outside of the left scallion for $p > 0$ and outside the right scallion for $p < -2\pi$. The region $-2\pi < p < 0$ is more subtle. It can be observed that the composite states can have their constituent states crossed for some values of momenta (see discussion in [29]). If this occurs, the crossed constituents will decrease the composite state energy to such an extent that it becomes lighter than the fundamental excitation[4] in other regions.

[4]By fundamental excitation, we mean the lightest state from which every other state can be constructed.

Therefore, there is no obvious choice for a fundamental excitation in the $-2\pi < p < 0$ region. This is natural because the fundamental states for $p > 0$ and $p < -2\pi$ regions are the Left $m = 1$ state and Right $m = 1$ states, respectively, so that region $-2\pi < p < 0$ interpolates between them. However, this raises the question of which part of this region is physically obscured (see [29] for more details).

In the momenta region $p \in [0, 2\pi]$, we can choose the crossing path to be the one depicted in figure 3.4 (resembling the one we have seen for pure R-R case, see figure 2.3). Importantly, this contour should also cross one of the branch cuts of $E_{\text{L/R}}(p)$ in the p-plane.

The crossing transformation in other regions can be defined in the following way: one need to continue $x^{\pm}_{\text{L}}/x^{\pm}_{\text{R}}$ to the momenta region $p \in [0, 2\pi]$, perform crossing as defined above, and then go back to the original momenta region.

Transition between different regions is done by crossing the $m = 0$ contours and log-cuts along the negative real axis. For example, the transition form region $p \in [0, 2\pi]$ to $p \in [2\pi, 4\pi]$ for $x^{\pm}_{\text{L}}$ can be done by taking $x^{\pm}_{\text{L}}$ inside the scallion, through the log cut and then moving out of the scallion again.

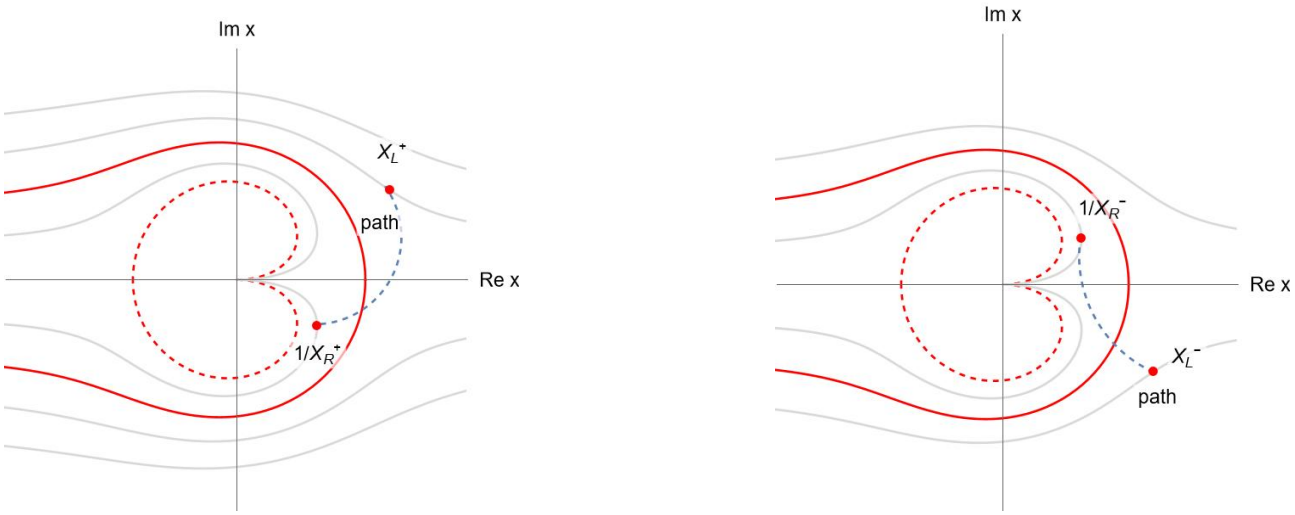


Figure 3.4: The crossing transformation path (blue dashed) for massive variables in the $x^{\pm}_{\text{L}}$ planes for the momentum range $p \in [0, 2\pi]$. The red contour corresponds to $m = 0$ Left Zhukovski contour in $[0, 2\pi]$ momenta, and the red dashed contour corresponds to the $m = 0$ Left Zhukovski contour in $[-2\pi, 0]$. Light grey contours correspond to $x^{+}_{\text{L}}/x^{-}_{\text{L}}$ and $\frac{1}{x^{+}_{\text{R}}}/\frac{1}{x^{-}_{\text{R}}}$ in momenta region $[0, 2\pi]$.

In the later section, we will construct the mixed-flux dressing factors by generalising the pure R-R solution, which is given by the odd part (2.3.78) and the even part in terms of the DHM integral (2.2.71). Both parts are expressed in terms of an integral over the boundary of the physical region, which is a circle in the pure R-R case. The expectation is that the expression for the mixed flux should also have an integration over the boundary of the physical region.

However, this creates a problem if one tries to write the solution in the form (2.2.71) [5] or (2.3.78). If one naively includes massless Zhukovski contours for all possible values of momenta, the resulting dressing phases would have an infinite number of branch cuts in the Zhukovski region outside the scallion (which is chosen as the physical region).

We propose to use the minimal integration contour consisting of the Scallion and the Kidney contours, depicted in figure 3.2. Both the Scallion and the Kidney have a branch

[5] BES phase, part of the full dressing phase which is believed to be universal and contains double poles, which is a common property of string theories in AdS_5 and AdS_3 [99].

point on the real line, whose location we denote as s and $-1/s$, which will play an important role later.

3.3 Solution of the crossing equations

The construction of the physical representation of $\mathfrak{psu}(1|1)^4_{\text{c.e.}}$ follows the same way as the pure R-R one. Furthermore, because the symmetry algebra is the same, the matrix part of the exact S-matrix is unchanged. Therefore, we can proceed to the unknown part – dressing factors.

Crossing equations for the massive sector are the same as for the pure R-R case, except for the addition of L and R indices [40]

$$\left(\sigma^{\bullet\bullet}_{\text{LL}}(x,y)\right)^2\left(\tilde{\sigma}^{\bullet\bullet}_{\text{RL}}(\bar{x},y)\right)^2=\left(\frac{y^-_{\text{L}}}{y^+_{\text{L}}}\right)^2\frac{(x^-_{\text{L}}-y^+_{\text{L}})^2}{(x^-_{\text{L}}-y^-_{\text{L}})(x^+_{\text{L}}-y^+_{\text{L}})}\frac{1-\frac{1}{x^-_{\text{L}}y^+_{\text{L}}}}{1-\frac{1}{x^+_{\text{L}}y^-_{\text{L}}}}\,, \tag{3.3.1}$$

$$\left(\tilde{\sigma}^{\bullet\bullet}_{\text{LR}}(x,y)\right)^2\left(\sigma^{\bullet\bullet}_{\text{RR}}(\bar{x},y)\right)^2=\left(\frac{y^-_{\text{R}}}{y^+_{\text{R}}}\right)^2\frac{(1-\frac{1}{x^+_{\text{L}}y^+_{\text{R}}})(1-\frac{1}{x^-_{\text{L}}y^-_{\text{R}}})}{(1-\frac{1}{x^+_{\text{L}}y^-_{\text{R}}})^2}\frac{x^-_{\text{L}}-y^+_{\text{R}}}{x^+_{\text{L}}-y^-_{\text{R}}}\,, \tag{3.3.2}$$

where a bar over x denotes that we crossed the expression in that variable.

We can identify even and odd parts of the full dressing phase in the same manner we did in section 2.3.4

$$\sigma^{odd}_{\text{LL}}(x,y)^2\sigma^{odd}_{\text{RL}}(\bar{x},y)^2=\left(\frac{x^+_{\text{L}}-y^-_{\text{L}}}{x^+_{\text{L}}-y^+_{\text{L}}}\frac{x^-_{\text{L}}-y^+_{\text{L}}}{x^-_{\text{L}}-y^-_{\text{L}}}\right), \tag{3.3.3}$$

$$\sigma^{odd}_{\text{LL}}(\bar{x},y)^2\sigma^{odd}_{\text{RL}}(x,y)^2=\left(\frac{1/x^+_{\text{R}}-y^+_{\text{L}}}{1/x^+_{\text{R}}-y^-_{\text{L}}}\frac{1/x^-_{\text{R}}-y^-_{\text{L}}}{1/x^-_{\text{R}}-y^+_{\text{L}}}\right). \tag{3.3.4}$$

In this work, we restricted our attention to the odd part of the massive mixed-flux dressing phase.

One of the simplest guesses for the mixed-flux solution is to write it to be very similar to the pure R-R one (2.3.84), with only contour integration modified

$$\begin{aligned}\chi^{\text{odd}}_{\text{LL}}(x,y)&=+\int_{S_{\text{L}}+K_{\text{L}}}\frac{dz}{2\pi}\frac{\log(y-z)}{x-z}\\&\quad+\frac{1}{4\pi}\log\frac{x+1/s}{x-s}\Big(\log(y+1/s)+\log(y-s)\Big)\,,\\\chi^{\text{odd}}_{\text{RL}}(x,y)&=-\int_{S_{\text{R}}+K_{\text{R}}}\frac{dz}{2\pi}\frac{\log(y-z)}{1/x-z}\\&\quad-\frac{1}{4\pi}\log\frac{x+s}{x-1/s}\Big(\log(y+1/s)+\log(y-s)\Big)\,,\end{aligned} \tag{3.3.5}$$

where we integrate over the Scallion and the Kidney contours (see figures 3.2 and 3.3), which are natural boundaries that generalise a circular contour from the pure R-R case.

However, the expressions in (3.3.5) are merely guesses at this point. Due to the complex kinematics of the theory, it does not seem possible to solve the crossing equation directly following the procedures developed for AdS_5 or pure R-R AdS_3 such as in [100]. However, in addition to the crossing equations (3.3.1), we can use various consistency conditions to construct the solution:

- general properties expected from the dressing phase described in 2.2.5;
- agreement of the possible solution with the pure R-R limit solution (2.3.84);
- agreement of the possible solution with the relativistic limit described in [118];

We have seen in the previous chapters how AdS_5 solution, given in terms of the BES phase [73], was modified with $(\chi^{HL} - \chi^{-})$ term (2.3.78) to produce the solution for the dressing factor of AdS_3 with pure R-R [116]. This was then modified to account for parity with the Cavaglia-Ekhammer term (2.3.81), which can be motivated by the fact that the original solution [116] had unphysical branch cuts. Following this approach, we had looked for a modification of the 'guess' solution (3.3.5) and tested many aspects of it, including consistency with the crossing equation, pure R-R and relativistic limits, strong-coupling expansion, absence of unphysical branch cuts and others.

The most useful of all was the strong-coupling expansion, which we consider in the next section.

3.3.1 Strong-coupling expansion

As argued in section 2.2.5, one of the important properties of the dressing phase is the decomposition in terms of local charges (2.2.68) (expansion in terms of powers of x and y). This is expected to hold for general integrable spin chains [92], for strings in AdS_5 [89], and is expected to hold in AdS_3.

To take this limit, we rescale the particle momenta $p \to p/h$ and take $h \to \infty$. This corresponds to taking large h and large Zhukovski variables x and y (so it is sometimes called large x and y expansion).

In this section, we focus on the large x and y expansion of the massive Left-Left odd dressing phase $\sigma^{\bullet\bullet}_{\text{LL}}$. This allows us to motivate the modification of the pure-R-R-inspired guess (3.3.5).

It is known that the leading term in the expansion should be proportional to $\delta_{r+1,s}$, while the subleading term should reproduce the perturbative S-matrix (see [119] for a discussion of the AdS_3 case). In the current form, there is a problem with (3.3.5). As we will see later, it does not have a valid large x and y expansion.

We start with a simplified analysis of (3.3.5). Up to some not important functions of x or y only, the integral over the Left Scallion can be simplified to

$$\int_{S_{\text{L}}} \frac{dz}{4\pi} \frac{1}{x-z} \log(y-z)\,. \tag{3.3.6}$$

The crucial difference compared to the pure R-R case is the integration over the Scallion S_{L}, which extends to infinity[6]. Upon inspection, we find that there are terms which do not have a well-defined large x and y expansion. Not well-defined in this case means that apart from the combination of powers of $\frac{1}{x}$ and $\frac{1}{y}$ (which are expected from the general form of the expansion (2.2.68)), we get terms of the form $\frac{x}{y}$. These terms also raise the question of the overall convergence of the expansion, which will be dependent on the relative magnitude of x and y.

[6]Such an infinite integration contour makes the integral divergent compared to the pure R-R case. However, the divergent part of the expression can be regularised, and the resulting divergent terms depend solely on either x or y, but not both. Consequently, they are cancelled in the full phase (2.3.68).

One of the ways to produce a large x and y expansion is to first expand the integrand and then separately integrate the terms of the resulting series. Ultimately, we want to consider a large (but still finite) x and y expansion. In pure R-R, it was sufficient to expand the integrand under this assumption and perform integration. However, in mixed flux, we need to consider the contour which extends to complex infinity, which makes the analysis more subtle.

For the analysis in this section, we assume that both x and y are fixed and located outside the integration contour, such that we can deform it to the real line $\int_{s+i\epsilon}^{-\infty+i\epsilon}$. Here s is the endpoint of the contour S_{L} on a real line which can be obtained from (3.1.6) by taking $m = 0$ and the limit $p \to 0^+$

$$s = \frac{\bar{k} + \sqrt{h^2 + \bar{k}^2}}{h} \,. \tag{3.3.7}$$

To expand the integrand, it is necessary to split the integration contour into three regions

- large $z \Rightarrow \left|\frac{x}{z}\right| , \left|\frac{y}{z}\right| < 1$,
- small $z \Rightarrow \left|\frac{z}{x}\right| , \left|\frac{z}{y}\right| < 1$,
- intermediate $z \Rightarrow \left|\frac{x}{z}\right| < 1, \left|\frac{y}{z}\right| > 1$ for $|x| < |y|$ or $\left|\frac{x}{z}\right| > 1, \left|\frac{y}{z}\right| < 1$ for $|x| > |y|$.

For convenience, we define z_x as a point on the integration contour S_{L} where $|z| = |x|$ and z_y as a point on the integration contour S_{L} where $|z| = |y|$, which we will call *intermediate points*. If $|x| < |y|$, we have

$$\begin{aligned}\int_{S_{\text{L}}} dz \frac{1}{z-x} \log(z-y) = \\ \int_s^{z_x} dz \frac{-1}{x} \frac{1}{1-z/x} [\log(-y) + \log(1-z/y)] + \int_{z_x}^{z_y} dz \frac{1}{z} \frac{1}{1-x/z} [\log(-y) + \log(1-z/y)] \\ + \int_{z_y}^{-\Lambda} dz \frac{1}{z} \frac{1}{1-x/z} [\log(z) + \log(1-y/z)] \,,\end{aligned} \tag{3.3.8}$$

where Λ is a regularisation parameter that we can take to infinity at the end.

First, we need to determine whether this expression depends on the intermediate points z_x and z_y. It is unclear *a priori*, however, we do not expect this dependence to be present in the final expression. For now, we discard the corresponding terms and investigate them in the next section. Without these, we have

$$\begin{aligned}- \log y \sum_{n=0}^{\infty} \frac{z^{n+1}}{(n+1)x^{n+1}} \Big|_s^{\dots} + \sum_{n=0,m=1}^{\infty} \frac{1}{m(n+m+1)} \frac{z^{n+m+1}}{x^{n+1} y^m} \Big|_s^{\dots} \\ + (\log z)^2 \Big|_{\dots}^{-\infty} + \sum_{n=1}^{\infty} x^n \left[-\frac{1}{n} \frac{\log z}{z^n} \Big|_{\dots}^{-\infty} + \frac{1}{n z^{n+1}} \Big|_{\dots}^{-\infty} \right] + y \log z \Big|_{\dots}^{-\infty} + \sum_{n=1}^{\infty} \frac{-1}{n} \frac{y}{z^n} \Big|_{\dots}^{-\infty} \\ + \sum_{m=2}^{\infty} \frac{-1}{m(m-1)} \frac{y^m}{z^{m-1}} \Big|_{\dots}^{-\infty} + \sum_{n=1,m=2}^{\infty} \frac{-1}{m(m+n-1)} \frac{x^n y^m}{z^{n+m-1}} \Big|_{\dots}^{-\infty}\end{aligned} \tag{3.3.9}$$

where the first line comes from the first integral of (3.3.8), and the other two from the last one of (3.3.8). Dotted points denote terms which depend on z_x and z_y and are to be considered later.

Second, we need to check the convergence of the expressions. The last two lines are either 0 in the limit $\Lambda \to -\infty$, or diverge. The divergent terms are functions of only one of the variables x or y and thus drop out in the full phase (2.3.68).

Finally, one can check that the first line of (3.3.9) reproduces a part of the large x and y expansion of [116] if $s = 1$, as expected.

Intermediate points contributions

Let us now look at the contributions in the integral (3.3.8) from the intermediate points z_x and z_y. There are two cases to consider: $|x| > |y|$ or $|x| < |y|$. We will consider the case $|x| < |y|$ in detail and only state the results for the other case, as the derivation is very similar.

In the integral (3.3.8), the following terms depend on z_x or z_y

$$\begin{aligned} S_{\text{interm}} = &- \log(-y) \sum_{n=1}^{\infty} \frac{z^n}{n x^n} + \sum_{n=1,m=1}^{\infty} \frac{1}{m(n+m)} \frac{z^{n+m}}{x^n y^m} \Big|_{\dots}^{z_x} \\ &+ \log(-y)\log(z-x) - \sum_{\substack{n=0 \\ m\neq n}}^{\infty} \sum_{m=1}^{\infty} \frac{1}{m(m-n)} \left(\frac{x}{z}\right)^n \left(\frac{z}{y}\right)^m + \log(z)\log(1-\frac{x}{y}) \Big|_{z_x}^{z_y} \\ &- \frac{1}{2}(\log z)^2 + \log z \log(z-x) + \sum_{n=1}^{\infty} \frac{1}{n^2} \left(\frac{y}{z}\right)^n - \sum_{n=1}^{\infty} \frac{1}{n^2} \left(\frac{x}{z}\right)^n + \sum_{n=1,m=1}^{\infty} \frac{1}{m(m+n)} \frac{x^n y^m}{z^{n+m}} \Big|_{z_y}^{\dots} \end{aligned} \tag{3.3.10}$$

where the dotted integration limits denote other parts of the expression that do not depend on z_x and z_y.

We expect cancellations of the terms containing z_x and z_y to occur, but this is not obvious at present. Computing these sums is non-trivial, so we differentiate the expressions under the sum sign with respect to y and evaluate the sums. After that procedure, we find

$$\partial_y S_{\text{interm}} = \frac{\log(-x) - \log(-y)}{x - y}\,, \tag{3.3.11}$$

which can be integrated back with respect to y (using identities for dilogarithm along the way)

$$S_{\text{interm}} = \log\left(\frac{x-y}{x}\right)(\log(-y) - \log(-x)) - \text{Li}_2\left(\frac{x}{y}\right) - \frac{1}{6}\pi^2 - \frac{1}{2}\log\left(-\frac{x}{y}\right). \tag{3.3.12}$$

We see that all z_x and z_y dependent terms are cancelled. There could be other constants or divergent terms dependent only on x or y, but these should also cancel at the level of the full dressing phase.

There is one more caveat here. In the above, when we differentiate with respect to y, we should also consider the y dependence of z_y. This is equivalent to adding two new group of terms: a z derivative of the second line of (3.3.10) (with a plus sign because of the z_y in the upper limit) and a z derivative third line of (3.3.10) (with a minus sign because of the z_y in the lower limit), both multiplied by $\frac{dz_y}{dy}$. It is easy to check that the resulting terms can be re-summed and the result is zero (up to functions of either x or y only, which should cancel in the full phase).

Similar computations can be performed for the case $|x| > |y|$. The result will again be independent of z_x and z_y, and the form of the expression will be the same up to a constant[7].

Combining all parts of (3.3.8) and dropping all constants and divergent terms yields

$$\int_{S_L} dz \frac{1}{z-x} \log(z-y) = \underbrace{\log y \sum_{n=1}^{\infty} \frac{s^n}{n x^n} - \sum_{n,m=1}^{\infty} \frac{1}{m(n+m)} \frac{s^{n+m}}{x^n y^m}}_{\text{same as pure R-R if } s=1}$$
$$- \log(x)\log(-y) - \log\left(1-\frac{x}{y}\right)\log\left(\frac{x}{y}\right) - \mathrm{Li}_2\left(\frac{x}{y}\right), \quad (3.3.13)$$

where the last line contains all terms (relevant in the full phase) arising from the intermediate point contributions in (3.3.8).

The derivation presented above is complicated if one tries to generalise it to complex contours and complex x and y. However, this is necessary if we want to work with the dressing phase. To verify the validity of the results, we regularise the integral (3.3.8) by introducing a cutoff and then check the expressions numerically.

Regularisation

Assuming that both x and y are fixed and located outside the integration contour, we can deform it to the real line $\int_{s+i\epsilon}^{-\infty+i\epsilon}$ and introduce a cut-off Λ

$$\int_s^{-\Lambda} dz \frac{1}{z-x} \log(z-y)\,. \quad (3.3.14)$$

We still need to evaluate the intermediate point contributions, so we demand Λ is large but finite such that $\Lambda > |x|, |y|$. Our approach again relies on differentiating with respect to y to evaluate the sums first and then integrating them back. Therefore, the results are derived up to constants and functions of x or y only.

As a result, we do not obtain all the necessary terms and are forced to guess (by trial and error numerically) the missing terms. The resulting expression is the following

$$\int_s^{-\Lambda} dz \frac{1}{z-x} \log(z-y) = S_1 + S_2 + S_3 + S_4\,, \quad (3.3.15)$$

where

$$S_1 = \frac{1}{2}\log\Lambda\log\Lambda - \log(-1)\log\Lambda + \frac{1}{2}\log(-1)^2 + \log\Lambda\log\big(1-\frac{x}{-\Lambda}\big)$$
$$- \log(-1)\log\big(1-\frac{x}{-\Lambda}\big)\,,$$

$$S_2 = -\mathrm{Li}_2\big(\frac{x}{-\Lambda}\big) + \sum_{m,n=1}^{\infty} \frac{1}{m(n+m)} \left(\frac{x}{-\Lambda}\right)^n \left(\frac{y}{-\Lambda}\right)^m + Li_2\left(\frac{y}{-\Lambda}\right),$$

[7]Some sums are convergent for $|x| < |y|$, but divergent for $|x| < |y|$. E.g. $\sum_{n=1}^{\infty} \frac{1}{n}\frac{x^n}{y^n} = -\log(1-\frac{x}{y})$ is well-defined for $|x| < |y|$, but not for $|x| < |y|$. Nevertheless, we still formally write the sum in the latter case as $-\log(1-\frac{x}{y})$ to demonstrate that the result has the same form.

$$S_3 = + \left(- \log(-y) \log(1 - s/x) - \sum_{m,n=1}^{\infty} \frac{1}{m(n+m)} \frac{s^{n+m}}{x^n y^m} \right) ,$$

$$S_4 = \pi^2/6 + 1/2 \log(-y)^2 - \log(-x)\log(-y) + \log(1 - x/y)\log(-y) \\ - \log(1 - x/y)\log(-x) - \mathrm{Li}_2(x/y) .$$

The terms grouped into S_1 diverge in the $\Lambda \to \infty$ limit. However, all of them are functions of either x or y only or reduced to these in the $\Lambda \to \infty$ limit. By requiring the solution to be evaluated in the principal value prescription, we can make these cancel between χ's in the full θ (2.3.68). The terms grouped into S_2 vanish in the $\Lambda \to \infty$ limit and can be discarded. The terms grouped into S_3 are part of the large x and y expansion, and reduce to the known pure R-R large x and y expansion in the limit $s \to 1$. The terms grouped into S_4, which are intermediate point contribution terms, are equivalent to those in the second line of (3.3.13), up to terms irrelevant in the full phase θ.

Numerical checks were performed in multiple ways, evaluating x or y derivatives of the expressions to obtain missing terms. We compared expression in terms of sums, complex contour integrals and integrals with contours deformed to the real line. To compute the contour integrals numerically, we used the *NIntegrate* function in Wolfram Mathematica.

Overall, the numerical tests show good agreement between the expressions, matching almost everywhere up to 10^{-10} when restricted to the first 40 terms in the sums. Figures 3.6 and 3.5 demonstrate the difference between expressions when compared in different ways[8]. For the region inside the contour, one should compare analytically continued expressions.

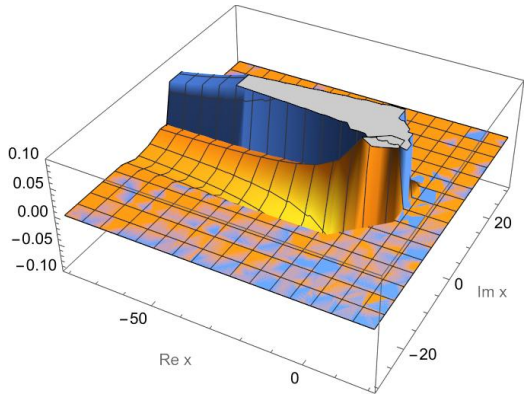


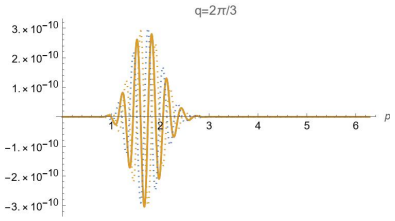
Figure 3.5: The difference between the numerical integral over S_{L} contour and its expansion where x varied in the complex plane and y fixed. The real part is shown in orange, the complex part is shown in blue.

One should be careful with the branch cuts of log terms. In particular, terms like the difference of two logarithms can sometimes be valid for x and y in the upper or lower half plane only. For other regions, we need to add i, $-i$ between the logarithms

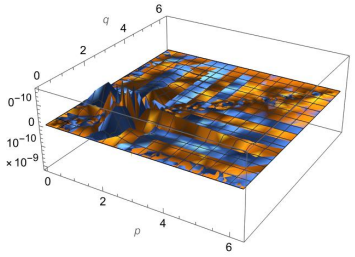
$$\log(a(s-x)) - \log(a(s-y)) , \tag{3.3.16}$$

where a can be -1,i or $-i$. The choice of a is dictated by the requirement of the absence of the log cut in the desired region for x and y. The way to achieve that is to rotate the $\log$-branch cut (by choosing a) and make sure that $\arg(x) - \arg(y)$ never crosses the log cut when x or y is varied in the desired region.

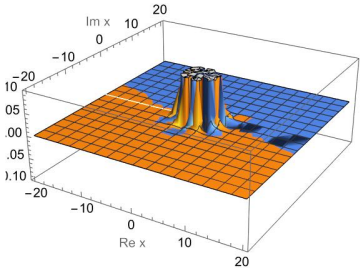
[8]The exception is the region of x (analogously for y) close to zero, depicted in 3.6(c), where the large Zhukovski expansion in x and y is no longer valid.



(a) $y(q)$ fixed with $q = 2\pi/3$ and $x(p)$ is varied along the $m = 1$ Zhukovski contour.



(b) $x(p)$,$y(q)$ varied on $m = 1$ Zhukovskis.



(c) x varied in the complex plane and $y(q)$ fixed at $q = \pi/3, m = 2$.

Figure 3.6: The difference between the large x/y expansion and the corresponding linear contour. The real part is depicted in orange, and the complex part in blue.

3.3.2 Modification of the mixed-flux solution.

Once we understood the problem with the strong-coupling expansion, we can cure the issue with the strong-coupling expansion by subtracting the unwanted terms in S_4 above

$$\begin{aligned}\chi_{\rm LL}^{\rm odd}(x,y) = &+\int_{S_{\rm L}+K_{\rm L}} \frac{dz}{2\pi}\frac{\log(y-z)}{x-z} + \frac{1}{4\pi}\log\frac{x+s}{x-s^{-1}}\Big(\log(y+s^{-1})+\log(y-s)\Big)\\ &+2\frac{1}{2\pi}\left(\log(-x)\log(-y)+\log\left(1-\frac{x}{y}\right)\log\left(\frac{x}{y}\right)+{\rm Li}_2\left(\frac{x}{y}\right)\right),\end{aligned} \tag{3.3.17}$$

where the last line has a factor of 2, as the same term comes from $S_{\rm L}$ from both x^+ and x^- parts. We assume that all expressions written in terms of integrals over the Left or the Right Scallion $S_{\rm L/R}$ and Kidney $K_{\rm L/R}$ are valid for momenta in the region outside the Left and Right Kidneys.

The above expression can be rewritten in a more concise form. First, we can transform the present integration contours using the relations

$$\begin{aligned}\int_{S_{\rm L}}+\int_{K_{\rm L}} &\to 2\int_s^{-1/s} -2\int_{-\infty}^0 \qquad \text{for } \chi_{\rm LL}\,,\\ \int_{S_{\rm R}}+\int_{K_{\rm R}} &\to 2\int_{1/s}^{-s} -2\int_0^{\infty} \qquad \text{for } \chi_{\rm RL}\,,\end{aligned} \tag{3.3.18}$$

where the contours on the right are deformed to be placed on the real line. Second, we can rewrite the subtracted terms in (3.3.17) using the identity

$$\int_{-\infty}^0 dz\frac{\log(y-z)}{x-z} = \log(-x)\log(-y)+\log\left(1-\frac{x}{y}\right)\log\left(\frac{x}{y}\right)+{\rm Li}_2\left(\frac{x}{y}\right). \tag{3.3.19}$$

Combining the two together, we rewrite (3.3.17)

$$\begin{aligned}\chi_{\rm LL}^{\rm odd}(x,y) = &+\int_s^{-1/s}\frac{dz}{2\pi}\frac{\log(y-z)}{x-z}\\ &+\frac{1}{4\pi}\log\frac{x+s}{x-s^{-1}}\Big(\log(y+s^{-1})+\log(y-s)\Big).\end{aligned} \tag{3.3.20}$$

Similar process can be performed for $\chi_{\text{RL}}^{\text{odd}}$

$$\begin{aligned}\chi_{\text{RL}}^{\text{odd}}(x,y) = &-\int_{1/s}^{-s}\frac{dz}{2\pi}\frac{\log(y-z)}{\frac{1}{x}-z}\\ &-\frac{1}{4\pi}\log\frac{x+s}{x-s^{-1}}\Big(\log(y+s^{-1})+\log(y-s)\Big)\\ &-\frac{i}{2}\operatorname{sign}(\operatorname{Im}x)\log\big(y-\frac{1}{x}\big)\,.\end{aligned} \tag{3.3.21}$$

The only procedural difference for $\chi_{\text{RL}}^{\text{odd}}$ is the inclusion of the term on the last line arising from the pole at $1/x$ during the deformation of the Scallion and Kidney contours to the real line.

Finally, expressions for $\chi_{\text{LR}}^{\text{odd}}$ and $\chi_{\text{RR}}^{\text{odd}}$ can be obtained from those above using LR symmetry. All of these expressions are valid for momenta in the region outside the left and right kidneys, that is, for $p_{\text{L}} \in (0, 2\pi)$ and $p_{\text{R}} \in (-2\pi, 0)$.

We can now check that the large x and y expansions of the above have the desired properties, namely, they contain only terms with inverse powers of x and y as intended

$$\begin{aligned}\chi_{\text{LL}} &= \frac{1}{4\pi}\sum_{m,n=1}^{\infty}\frac{x^{-m}y^{-n}}{mn}\left[\frac{m-n}{m+n}\big(s^{n+m}-(-s^{-1})^{n+m}\big)+(-1)^m s^{n-m}-(-1)^n s^{m-n}\right],\\ \chi_{\text{RL}} &= \frac{\log s}{\pi}\sum_{n=1}^{\infty}\frac{1}{n}y^{-n}x^{-n}+\frac{1}{4\pi}\sum_{n=1}^{\infty}\frac{(-1)^n\big(s^{-2n}-s^{2n}\big)}{n^2}y^{-n}x^{-n}\\ &\quad+\frac{1}{4\pi}\sum_{\substack{m,n=1\\ m\neq n}}^{\infty}\frac{x^{-m}y^{-n}}{mn}\left[\frac{n+m}{n-m}\big(s^{n-m}-(-s)^{m-n}\big)+(-1)^n s^{-n-m}-(-1)^m s^{n+m}\right],\end{aligned} \tag{3.3.22}$$

where we omitted terms which are irrelevant in the full phase. For χ_{RL}, we first deformed the $[1/s, -s]$ integration contour to a semicircular one, depicted in Figure 3.7, which is outside the Right Kidney. This allowed us to eliminate the term proportional to $\operatorname{sign}(\operatorname{Im}x)$ and take large x and y limits safely.

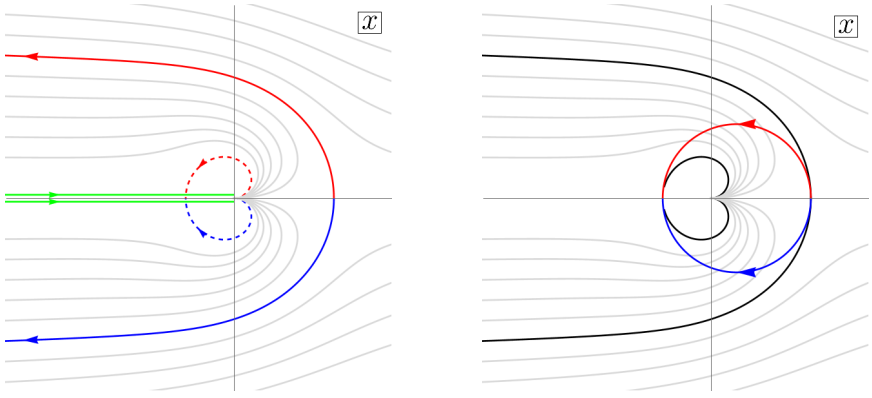


Figure 3.7: Mixed-flux odd dressing phase contours.

We shall now look back at the rest properties outlined in Section 3.3 and ensure that they are satisfied for (3.3.20) and (3.3.21).

- Agreement with pure R-R solution can be established by comparing the large x and y expansions (2.3.85) and (3.3.22) and taking the $s \to 1$ limit in the latter.

- Relativistic limit studied in [118] can be obtained by taking the limit $h \to 0$ and expanding the momentum around

$$p_{\text{L}} = -\frac{2\pi m}{k} + \epsilon(h)\sinh\theta\,, \quad p_{\text{R}} = \frac{2\pi m}{k} + \epsilon(h)\sinh\theta\,,$$

  where $\epsilon = \frac{4\pi h}{k}\left|\sin\left(\frac{m\pi}{k}\right)\right|$ and θ is the rapidity parameter.

  To check the agreement of (3.3.20) and (3.3.21) with the relativistic limit [118], it is necessary to continue left momenta to $(-2\pi, 0)$ region and right momenta to $(0, 2\pi)$ region. Then, assuming that the even phase trivialises in the relativistic limit, it is possible to match the analytical expressions for the derivative of the relativistic dressing phases with respect to the rapidity parameter θ of [118] (written in terms of the Barnes G-function) and the same θ derivative of our proposal. Alternatively, it is easy to compare both expressions numerically, similar to what we did in Section 3.3.1. In fact, it is the intermediate point contribution terms that are matched with the expressions of [118] while the rest is either cancelled in the full phase θ or trivialised in the relativistic limit.

- Similarly, our odd dressing phase can be matched numerically to the odd part of the phase of [120] (published shortly after our work [29]), where a full solution was proposed. Their solution was formulated as a deformation of the DHM-type integral for the even part and as a generalisation of the Barnes-G-function expression from [118] for the odd part.

- A property found for the pure R-R QSC [121], which is expected to hold for the mixed flux as well, is an absence of the monodromy around infinity. In our case, it is the property of the solution that if one completes a circle in the x-plane far enough from the origin (avoiding any cuts), one should not change the solution. The final form of the odd dressing phase (3.3.20) and (3.3.21) is formulated in terms of the finite contour integral. Additionally, the non-integral term cancels the branch cuts that would be present in the expression without it. Therefore, our solution satisfies the absence of the monodromy property.

Finally, it is easy to check that the phases (3.3.20) and (3.3.21) solve the odd part of the crossing equations (3.3.3) and (3.3.4). For example, we simply pick up a Sokhotski–Plemelj term when crossing the integration contours (figure 3.7) from outside the Scallion to inside the Kidney.

3.4 Summary and Outlook

In this chapter, we have focused on the S-matrix massive sector of $\text{AdS}_3 \times \text{S}^3 \times \text{T}^4$ with a mixed R-R and NS-NS flux. Using the results for the pure R-R case from Section 2.3, we demonstrated how the representation of the $\mathfrak{su}(1|1)^2_{\text{c.e.}}$ algebra should be deformed in the mixed-flux case. This led to very different kinematics with non-periodic momenta for the excitations and an infinite number of sheets for the Zhukovski variables.

Such a kinematical structure makes finding dressing factors distinct from the AdS_5 and AdS_3 pure R-R cases. We analysed the structure of Zhukovski sheets to define the crossing procedure. Then, starting from the ansatz inspired by the pure R-R solution and using the requirements of a well-behaved strong-coupling expansion, we constructed the solution for

the odd part of the massive dressing phase for mixed flux. The resulting solution agrees with the known solutions in the pure R-R and relativistic limits.

The dressing phase solution for the odd part of the massive dressing factor we obtained agrees with the odd part of the solution presented in [120], [122] published later. However, there are several points for disagreement for with our work. First, kinematics, in particular the identification of the physical region, is noticeably different. Second, a CDD factor required for the fusion procedure proposed in our [29] (not discussed here) is rejected in [122]. Thirdly, the final form of the solution in [122] does not have the absence of monodromy property discussed above. All of them require further investigation.

Once the full solution for massive modes is found, the next step is to find the dressing phases for the massless modes, which were also proposed recently [123].

When the full dressing factor is known, one can use the resulting S-matrix to understand the spectrum of the theory using techniques such as the Thermodynamic Bethe Ansatz [84] or Quantum Spectral Curve (QSC) [86]. While the TBA was constructed [123], the QSC remains to be derived.

The mixed flux strings in AdS_3 should interpolate between two very different regimes of string theory, having a WZW description for the pure NS-NS and an RNS description for pure R-R limits, and thus have the potential to bridge the gap between the two descriptions.

Another interesting research direction is the study of the $k = 1$ and small h limits, where it is possible to compare with the results of [47], [48], [51] to better understand the dual symmetric-product orbifold CFT of T^4. Furthermore, the generalisation of the considered integrability approach to other AdS_3 backgrounds might be possible. These include a long-known $AdS_3 \times S^3 \times S^3 \times S^1$ background and more recent cases of integrable deformations of $AdS_3 \times S^3 \times T^4$[9].

[9] See [124] for a recent review.

Chapter 4

Machine Learning and Optimisation Methods

The rapid growth in computational power over the last few decades has made many resource-intensive statistical methods practical for implementation. This has allowed the development and application of a diverse set of techniques known as *Machine Learning* (ML).

Machine Learning is a subfield of *artificial intelligence* (AI) that enables computers to learn from data and improve their performance over time. More intuitively, learning can be described as a process of fitting a mathematical model to the observed data.

ML methods have proven to be effective across a diverse range of fields. Within mathematics and mathematical physics research, a growing number of ML techniques have recently appeared, yet many opportunities for new applications and insights remain. An overview of the potential of ML in theoretical sciences can be found in [125]. Furthermore, reviews [126], [127] illustrate how AI can be used to discover patterns and relationships between mathematical objects, leading to new structures, conjectures, and theorems. Another significant ML application, situated between geometry and mathematical physics, is the study of the string theory landscape. This field adopted the application of ML first and continues to see active development [128].

The following chapters of the thesis explore the application of ML to complex problems in combinatorics, where, on the one hand, vast search spaces make traditional approaches less effective or intractable, but on the other hand, make the application of ML possible. In this chapter, we review some ML and optimisation techniques that will be subsequently applied to two specific challenges: first, the analysis and classification of Coxeter invariants of Clifford algebras, and second, the search for new champion generalised toric codes.

4.1 Genetic Algorithms

Genetic Algorithms (GA) are stochastic optimisation algorithms developed in the 1950s [129], [130], and later formalised in [131], [132]. This method is still in active use, and a review of modern techniques can be found in [133]. It is an optimisation and search algorithm that uses biologically inspired operations, such as selection, crossover, and mutation, which conceptually imitate the corresponding processes in nature.

The overall goal is to find a maximum x_0 of a function $f(x)$ defined on a space X. In many practical problems, this is almost impossible because of features such as combinatorially large search spaces or non-convexity of the function f. However, a near-optimal solution x is often sufficient in practice. To perform the search, one assumes that there exists a space

of strings S and an injective map $c : X \to S$ (encoding function). An inverse map $c^{-1} : S \to X$ (decoding function) can be defined, but it does not necessarily need to be injective, as multiple strings can sometimes decode to the same solution.

In this setting, a string s is referred to as an individual, and the corresponding $x = c^{-1}(s)$ is a candidate solution. Map c encodes a candidate solution as a string of characters (genes) from a finite alphabet, and c^{-1} performs the inverse transformation. Function f is called a fitness function and maps a solution to a fitness score, indicating the quality of the solution.

The GA is initialised by creating an initial population P_0 of size n_{pop} often drawn randomly. The initial population is iteratively updated $P_i \to P_{i+1}$, for $i = 0, 1, ..., n_{gen} - 1$, using the evolutionary process (described below).

The process is terminated when a specified criterion is met or the maximum number of iterations (sometimes referred to as generations) n_{gen} is completed.

The evolutionary process $P_i \to P_{i+1}$ consists of the following procedures:

- **Selection**: select individuals for reproduction from P_i;
- **Crossover**: create offspring from selected individuals;
- **Mutation**: randomly change some individuals;

Selection is a mechanism for selecting individuals (strings) for reproduction according to their fitness (objective function value). Usually, a probability function is defined for selecting individuals. The outcome of the selection should be random, however, individuals with a high fitness score are more likely to be selected. It is common to draw n_{pop} parents from the population, which are then randomly paired for further crossover[1]. This selection process is typically performed with replacement so that an individual can be selected several times and paired with several other individuals. Further details of the selection process depend on the exact approach used.

Crossover is a procedure of combining two selected individuals (parents) into two new individuals (offspring). Most often, this is performed by exchanging the subparts of the strings of the parents. For each pair selected at the selection stage, a pair of offspring is created, resulting in n_{pop} individuals which form a temporary population $\tilde{P}_{i+1}$.

Mutation is a way to diversify the resulting population by randomly changing some genes in individuals. A common strategy is to consider all genes in $\tilde{P}_{i+1}$ independently to have some probability r of being altered. The resulting population forms the next generation P_{i+1}.

A modification, which we will see used later, called **elitism**, can be added to ensure that the best solution quality does not decrease throughout the evolution process. When this modification is used, some number n_{elite} of individuals are carried over directly to the next generation, while the remaining $n_{pop} - n_{elite}$ members of the new generation are formed by applying selection, crossover, and mutation steps to the parent population P_i.

In Chapter 6, we will use a GA in conjunction with a neural network model to search for champion toric codes. The next section defines the neural network and reviews other relevant concepts.

[1] Assuming two offsprings per a pair of parents.

4.2 Machine Learning

In this section, we review the relevant concepts from the vast field of Machine Learning. For a general introduction to the subject, standard textbooks can be used [134], [135], or a more concise introduction to ML for physicists can be found in [136].

In contrast to GA, which is a general search algorithm, ML focuses on the different task of fitting mathematical models to observed data. It can be categorised into Supervised, Unsupervised, and Reinforcement learning, depending on the type of data available.

Supervised learning operates on labelled data, which contains both the input $\mathbf{x}$ and output $\mathbf{y}$ we expect the model to learn. The task is to learn a mapping $f : \mathbf{x} \to \mathbf{y}$ or a probability distribution $p(\mathbf{y}|\mathbf{x})$ (if $\mathbf{x}$ and $\mathbf{y}$ are treated as random variables). This type of learning can be subdivided into two primary tasks: classification, which involves predicting a discrete label, and regression, which involves predicting continuous values. One or the other is chosen depending on the nature of $\mathbf{y}$. Feedforward neural networks and recurrent neural networks are two prominent examples of supervised ML algorithms.

Unsupervised learning operates on unlabelled data where the objective is to discover patterns and relationships within the data. Common goals include clustering (grouping data) and dimensionality reduction (finding a more compact representation of the data). Principal Component Analysis (PCA) is an example of the latter.

Reinforcement learning works with data in the form of feedback, such as rewards and penalties generated by an environment. In this case, the goal of the ML model is to learn the optimal actions to maximise rewards and minimise penalties in the long term. The learned mapping from a state s in the set of states S to an action a in the action space A is known as a policy and is usually denoted as $\pi(a|s)$. This last type of learning will not be used in the present study.

We now proceed to review the relevant supervised and unsupervised learning techniques which will be encountered in the subsequent sections.

4.2.1 Principal Component Analysis

Principal Component Analysis (PCA) is an unsupervised ML technique, a linear dimensionality reduction method. This is achieved by learning the orthogonal linear transformation of data, where it is possible to find high-variance directions, which in most cases represent the most relevant information in the data.

Consider a set of points $\vec{x}^{(i)} \in \mathbb{R}^N, i = 1 \ldots, M$, which can be written as a matrix $\mathbf{x} \in \mathbb{R}^{M\times N}$. Without loss of generality, we can assume that they have a zero empirical mean $\sum_{i=1}^{M} \vec{x}^{(i)}/M = 0$.

The sample covariance matrix associated with $\mathbf{x}$ is

$$\mathrm{Var}[\mathbf{x}] = \frac{1}{M-1}\mathbf{x}^T\mathbf{x}\,. \tag{4.2.1}$$

Then, we can find a representation of the data $\mathbf{c} = \mathbf{D}^T\mathbf{x}$, such that $\mathrm{Var}[\mathbf{c}]$ is diagonal, and find directions associated with the highest variance.

This task is usually performed using Singular Value Decomposition (SVD). Recall that a real matrix $\mathbf{x}$ can be decomposed into $U\Sigma W^T$, where $U \in \mathbb{R}^{M\times M}$ and $W \in \mathbb{R}^{N\times N}$ are orthogonal matrices, $\Sigma \in \mathbb{R}^{M\times N}$ is a rectangular diagonal matrix with non-negative numbers

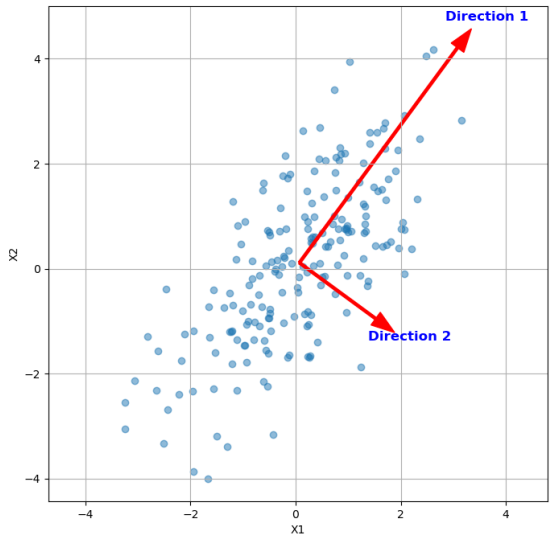


Figure 4.1: An example of application of PCA to two-dimensional data. Red arrows denote two principal directions.

on the diagonal. Then

$$\text{Var}[\mathbf{x}] = \frac{1}{M-1}\mathbf{x}^T\mathbf{x} = \frac{1}{M-1}\left(U\Sigma W^T\right)^T U\Sigma W^T = \tag{4.2.2}$$

$$W\left(\frac{\Sigma^2}{M-1}\right)W^T = W\Lambda W^T\,, \tag{4.2.3}$$

where Λ is a diagonal matrix with eigenvalues λ_i. From here, we can see that the right singular vectors of $\mathbf{x}$ (i.e. the columns of W) are the principal directions of $\text{Var}(\mathbf{x})$, while the singular values of $\mathbf{x}$ are related to the non-zero elements s_i of matrix Σ as $\lambda_i = s_i^2/(M-1)$.

Figure 4.1 depicts the result of applying PCA to two-dimensional data described by a skewed Gaussian distribution. Direction 1 is the principal direction with the largest variance, whereas direction 2 has a smaller variance and is thus less important.

To reduce the dimensionality of the data from N to $N' < N$, we can construct a matrix W' by selecting the principal components (columns of W) corresponding to the N' largest singular values λ_i. Matrix $\mathbf{W}'$ can be thought of as performing the encoding of data $\mathbf{x}' = \mathbf{xW'}$. The inverse transformation should be given by $\mathbf{x} \simeq \mathbf{x'W'}^T$. The approximate equivalence sign indicates that the decoding is not exact.

One can consider a reconstruction function $r(\mathbf{x}) = \mathbf{xW'W'}^T$. It can be shown that the encoding matrix $\mathbf{W'}$ can also be chosen by minimising the reconstruction error measured by the Frobenius norm

$$\mathbf{D}^* = \underset{D}{\text{argmin}} \sqrt{\sum_{i,j}\left(x_j^{(i)} - r(x^{(i)})_j\right)}, \text{ subject to } \mathbf{D}^T\mathbf{D} = I. \tag{4.2.4}$$

4.2.2 Neural Networks

A neural network (NN) is a versatile computational model used in all three subfields of ML problems. It consists of connected nodes called neurones, which loosely model the brain's

neurones. It can be represented as a function that maps an input $\vec{x} \in \mathbb{R}^{d_{in}}$ to an output $\vec{y} \in \mathbb{R}^{d_{out}}$, where d_{in} and d_{out} are the dimensions of the input and output, respectively. We denote the components of $\vec{x}$ as $x_i, i = 1, \ldots, d_{in}$, and the components of $\vec{y}$ as $y_j, j = 1, \ldots, d_{out}$.

The building block of an NN is a neuron. It is a function that takes an input $\vec{x}$, which is acted upon linearly by a vector of weights $\vec{W}$ and a bias b. The resulting number is then passed to a non-linear 'activation' function σ, producing the neuron output

$$z = \sigma\left(\vec{W} \cdot \vec{x} + b\right) , \tag{4.2.5}$$

In practical applications, σ is commonly chosen to be *hyperbolic tangent* (tanh) or *ReLU*

$$\sigma_{\text{ReLU}}(x) = \max(0, x) , \qquad \sigma_{\text{tanh}}(x) = \tanh(x) , \tag{4.2.6}$$

in regression tasks, while *sigmoid* and *softmax*

$$\sigma_{\text{softmax}}(x)_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)} , \qquad \sigma_{\text{sigm}}(x) = \frac{1}{1 + \exp(-x)} , \tag{4.2.7}$$

are frequent in classification problems. It is important to distinguish that *sigmoid*, *tanh*, or *ReLU* are applied element-wise, whereas *softmax* operates on an entire vector.

An individual neuron is a function from $\mathbb{R}^{d_{in}}$ to $\mathbb{R}$. Neurons can be arranged into a layer such that the output of each neuron in the layer gives a component of the output vector $\vec{y}$. This can be written in matrix notation if we collect the weight vectors of all the neurons $\vec{W}$ into a matrix $\mathbf{W}$ and the bias terms into a vector of biases $\vec{b}$. Then

$$\vec{z} = \sigma\left(\mathbf{W}\vec{x} + \vec{b}\right) . \tag{4.2.8}$$

These layers can be stacked by treating the output of the previous layer as the input for the next layer. That is, the input vector $\vec{x}$ is passed to each neuron in the first layer (called the *input layer*), then the output of each neuron at the first layer is combined into the vector $\vec{h}^{(1)}$ which is passed to the second layer and so on until the final layer (called the *output layer*) produces the output.

Let us denote the parameters of kth out of n_l neurons in layer l as $\mathbf{W}^{(l)}$ and $\vec{b}^{(l)}$, let the input be $\vec{x} = \vec{h}^{(0)}$ and the output be $\vec{y} = \vec{h}_i^{(L)}$. Then, for the layers $l = 1, \ldots, L$

$$\vec{h}^{(l)} = \sigma\big(\mathbf{W}^{(l)}\vec{h}^{(l-1)} + \vec{b}^{(l)}\big) . \tag{4.2.9}$$

In other words, the transformation performed by the NN on the input is equivalent to the iterative application of the same function (with different parameters $\mathbf{W}$ and $\vec{b}$) many times.

There is a lot of flexibility in this construction as one can vary the input $\vec{x}$ and output $\vec{y}$ dimensions, the number of layers in between the input and output layer (such layers are called *hidden layers*), and many choices for the functions σ can be made. Values $\mathbf{W}^{(l)}$ and $\vec{b}^{(l)}$ are parameters of the NN that define what function it represents. In the following, we will collectively denote the parameters $\{\mathbf{W}^{(l)}, \vec{b}^{(l)} \text{ for } l = 1, \ldots, L\}$ of the NN as θ, and the function that the NN approximates as f, so we can write the action as

$$\vec{y} = f(\vec{x}; \theta) . \tag{4.2.10}$$

Figure 4.2 depicts an example of a simple neural network, a *feed-forward neural network* (FFNN), also known as a multi-layer perceptron (MLP). Each circle in the input layer denotes

a component of an input vector $x_i = (x_1, x_2)$, which is taken to produce values fed into the i-th neuron in the hidden layer

$$\vec{h} = \sigma\left(\mathbf{W}^{(1)}\vec{x} + \vec{b}^{(1)}\right) . \tag{4.2.11}$$

After that, $\vec{h}$ is used to compute the output

$$\vec{y} = \sigma\left(\mathbf{W}^{(2)}\vec{h} + \vec{b}^{(2)}\right) . \tag{4.2.12}$$

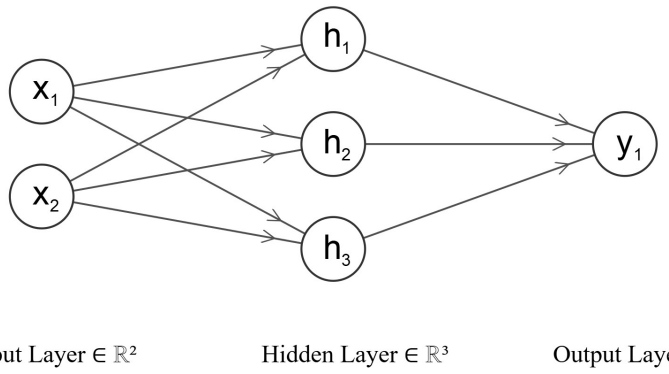


Figure 4.2: Scheme of a one hidden layer neural network.

NNs are one of the most powerful tools for supervised, unsupervised, and reinforcement learning problems. They can capture complex nonlinear dependencies, which is ensured by their versatility as universal approximations. The latter is an exact mathematical statement – a width version of the universal approximation theorem, that proves that for any continuous function, there exists a network with one hidden layer that can approximate any continuous function on a compact subset of $\mathbb{R}^n$ to arbitrary accuracy [137], [138][2]. However, finding the parameters of an NN which will fit a given taks is nontrivial. We will discuss this in the case of supervised learning problems in the next section.

4.2.3 Loss Function and Training

The training dataset in supervised learning consists of some number N input/output pairs $\{x^{(i)}, y^{(i)}$ for $i = 1, \ldots, N\}$. Here, $x^{(i)}$ and $y^{(i)}$ may be vectors or scalars, depending on the problem. To quantify how well the NN approximates the relationship between the input and output data, one can introduce the loss function $L(\theta)$. Typically, a lower loss indicates a better approximation. One of the most popular loss functions for regression tasks is *Mean Squared Error* (MSE) loss

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(f(x^{(i)};\theta) - y^{(i)}\right)^2 , \tag{4.2.13}$$

[2]There is also a depth version of the universal approximation theorem stating that there exists a network with ReLU activation functions and at least D_i+4 hidden units in each layer that can approximate any specified D_i-dimensional Lebesgue integrable function to arbitrary accuracy if it has enough hidden layers.

while in (binary) classification tasks ($y^{(i)} \in \{0, 1\}$) *binary cross-entropy loss* is common

$$L(\theta) = \sum_{i=1}^{N} \big[-(1 - y^{(i)}) \log \big(1 - \sigma_{\text{sigm}}(f(x^{(i)}; \theta))\big) \\ -y^{(i)} \log \big(\sigma_{\text{sigm}}(f(x^{(i)}; \theta))\big)\big] \,. \quad (4.2.14)$$

Both functions above, along with most others used in practice, have a minimum reaching which would indicate that the model learned the dataset perfectly.

Initially, the parameters of the network are initialised with random numbers sampled from uniform or normal distributions. Then, by gradually changing the parameters of the model according to some optimisation algorithm, the loss is decreased. This process is termed model *fitting*, *training*, and *learning*.

The goal of an optimisation algorithm is to find the parameters θ that minimise the loss:

$$\hat{\theta} = \text{argmin}\, L(\theta) \,. \quad (4.2.15)$$

There are many families of optimisation algorithms, but most are based on the concept of *gradient descent* (GD). The idea is to iteratively update the parameters using the derivative of the loss function

$$\theta_{\text{new}} = \theta - \eta \frac{\partial L}{\partial \theta} \,, \quad (4.2.16)$$

where η is the parameter called the learning rate, which defines how fast we change the model's parameters. However, there is no practical way to choose it for a particular problem analytically and Therefore, it must be selected empirically through a random or grid search.

The sizes of modern datasets are so large that it becomes computationally expensive to compute the full loss function. One of the simplest modifications of GD is the *stochastic gradient descent* (SGD). The idea is to use only a small random fraction, called a *batch*, of data entries at each training step in the sum of the loss function

$$\theta_{t+1} = \theta_t - \eta \frac{\partial L_{\text{batch}}}{\partial \theta_t} \,. \quad (4.2.17)$$

Each batch is obtained by randomly sampling from the full training dataset without replacement, so that the dataset is split into groups of data points (batches) that do not overlap. One full pass through the entire training dataset is referred to as *epoch*. Once all the batches are used once during one epoch, they might be re-sampled to be used during the next epoch.

Another difficulty is that the loss functions for most nonlinear models are non-convex. For instance, there might be numerous local minima to which the optimisation algorithm can be attracted. Alternatively, the loss function may have saddle points, which can significantly slow down the optimisation process. This motivates a modification of the stochastic gradient descent called the *stochastic gradient descent with momentum* (SGDM) [139], [140]. This method updates the parameters with a weighted combination of the gradient computed from the current batch and an exponentially decaying moving average of the past gradients as follows:

$$m_{t+1} = \beta \cdot m_t + (1 - \beta) \frac{\partial L_{\text{batch}}}{\partial \theta_t} \,, \quad (4.2.18)$$

$$\theta_{t+1} = \theta_t - \eta \cdot m_{t+1} \,, \quad (4.2.19)$$

where β controls the degree to which the gradient is smoothed over time, and m_t is the term that accumulates past gradients.

Modern optimisation algorithms such as AdaGrad, RMSProp, Adam and AdamW implement SGDM, as well as other techniques such as an adaptive learning rate and weight decay. Although there is no universal method that works best for all tasks, AdamW [141] is considered the standard, and we will mostly employ it in the experiments.

To approximate more complicated functions, the number of parameters can be increased by adding more hidden layers, increasing the *depth* of the network, or by increasing the number of neurons in the hidden layers, increasing the *width* of the network. In practice, moderately deep networks have an advantage in representation efficiency over shallow networks [142]. That is, deeper networks can model more complicated relations while having the same or fewer numbers of parameters compared to 'shallow' networks. However, as we make the network deeper, that is, add more hidden layers, training becomes more difficult, and problems such as dying/exploding gradients, overfitting, and others arise. To mitigate these issues, more complicated architectures than FFNNs are used. Examples include Convolutional Neural Networks (CNNs) for image and spatial data and Recurrent Neural Networks (RNNs) for sequential data (e.g. time series). The latter is relevant to this discussion and is described later.

4.2.4 Sequence Modelling

A particular type of data used in supervised learning is sequential data. It consists of an ordered sequence of input/output pairs $x_t, y_t, t \in [0, N]$. An example of such a sequence could be stock market prices in dollars at various moments in time t, which could be predicted in the future.

A sequence model in ML is a model which can work with sequential data, such as audio and text. Depending on the number of input and output elements, one can distinguish many-to-many, many-to-one, one-to-many, and other types of models.

One of the simplest approaches to handling sequential data is Recurrent Neural Network (RNN). Unlike a feed-forward network, it maintains a hidden state that is passed forward through time, allowing it to "remember" information from previous time steps. Formally, at each time step t, the hidden state h_t depends on the current input x_t as well as the previous hidden state h_{t-1}. The simplest realisation of this idea is the (Elman) Recurrent Neural Network (RNN). Given a sequence of inputs $(x_1, .., x_T)$, it computes an auxiliary sequence $h_1, ..h_T$ of hidden states

$$h_t = \sigma\left(W^{ih}x_t + W^{hh}h_{t-1}\right) , \tag{4.2.20}$$

which is then used to produce an output y_t

$$y_t = W^{ho}h_t , \tag{4.2.21}$$

for a many-to-many form, or

$$y = W^{ho}h_T , \tag{4.2.22}$$

in a many-to-one form. Here, h_t is the hidden state at step t, and W^{ih}, W^{hh}, W^{ho} are the input-to-hidden, hidden-to-hidden, and hidden-to-output weight matrices, respectively. This recursive use of the hidden state provides RNNs with the ability to capture sequential patterns in data.

However, this architecture has poor performance when used with long sequences. The repeated multiplication of the same weight matrix W^{hh} many times to process an output

y_t causes gradients to vanish or explode during backpropagation, which hinders the ability of the network to learn long-term dependencies. To address such gradient issues, a special type of RNN, *Long-Short Term Memory* (LSTM) [143], was proposed. LSTMs capture longer dependencies more effectively by introducing a cell state that can carry information over many time steps with minimal modifications via three gates: the forget gate, which determines what information is no longer relevant and should be discarded; the input gate, which selects which new information will be added to the cell state; and the output gate, which determines the part of the cell state that is propagated to the hidden state. Figure 4.3 shows a schematic representation of the LSTM.

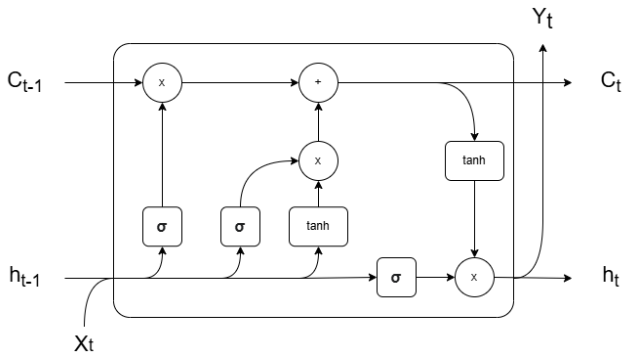


Figure 4.3: Structural scheme of LSTM cell where X_t: input at time step t, h_t: output, C_t: cell state. Operations inside circles are pointwise.

LSTMs have been successfully applied in language modelling and machine translation [144], [145], [146]. However, the sequential approach to data processing is a major limitation in terms of processing time. Moreover, despite significant improvements over RNNs, LSTMs still have the same issues with gradients, but now with longer sequences. This eventually led to the development of another technique, Transformers, which we discuss in the next section.

4.2.5 Transformers

One of the most powerful architectures currently available in ML is the Transformer architecture[147], which is the state-of-the-art approach for sequence modelling tasks used in Natural Language Processing (NLP) tasks such as chatbots [148], sentiment analysis [149] and others.

The key component of the transformer architecture is the attention mechanism. It was originally proposed [145] to improve RNN on language translation of long sentences, and only later it was realised that attention can be used on its own [147].

We shall now discuss the main elements of the Transformer when it is used in NLP tasks, which will be generalised later to other tasks.

Attention

A self-attention block[3], which we will denote $\mathbf{sa}(...)$, takes a sequence of N inputs $\vec{x}_1, ..., \vec{x}_N$, and outputs N vectors of the same size. Its action can be divided into several parts.

[3]A version of attention we describe is called self-attention. A modification of it called *cross-attention*, in which queries (to be described below) are computed using another sequence. We will not consider it in this work.

First, a set of *values*, *queries*, and *keys* are computed for each input using a linear transformation

$$\vec{v}_m = \mathbf{W}_v \vec{x}_m + \vec{b}_v \,, \tag{4.2.23}$$

$$\vec{q}_m = \mathbf{W}_q \vec{x}_m + \vec{b}_q \,, \tag{4.2.24}$$

$$\vec{k}_m = \mathbf{W}_k \vec{x}_m + \vec{b}_k \,, \tag{4.2.25}$$

where $\{\mathbf{W}_v, \vec{b}_v\}$, $\{\mathbf{W}_q, \vec{b}_q\}$, $\{\mathbf{W}_k, \vec{b}_k\}$ are the weight matrix and bias vector for *values*, *queries* and *keys*, respectively, which are the same for all inputs.

To produce an output, a weighted sum of the value vectors $\vec{v}_1, ..., \vec{v}_N$ is computed. The n^{th} output of the attention layer is

$$\mathbf{sa}_n(\vec{x}_1, ..., \vec{x}_N) = \sum_{m=1}^{N} a(\vec{x}_m, \vec{x}_n)\vec{v}_m \,, \tag{4.2.26}$$

where $a(\vec{x}_m, \vec{x}_n)$ is a scalar factor which depends on the inputs $\vec{x}_m$ and $\vec{x}_n$. These factors sum to one and represent the relative importance of the input vector $\vec{x}_m$ to $\vec{x}_n$, or vice versa.

Finally, the computation of the factors $a(\vec{x}_m, \vec{x}_n)$ is done by computing a dot product between the *query* and *key*, and the results is passed through a softmax function

$$a(\vec{x}_m, \vec{x}_n) = \text{softmax}'\big(\vec{k}_m^T \vec{q}_n\big) = \tag{4.2.27}$$

$$\frac{\exp\big(\vec{k}_m^T \vec{q}_n\big)}{\sum_{m'=1}^{N} \exp\big(\vec{k}_m^T \vec{q}_n\big)} \,, \tag{4.2.28}$$

where softmax$'$ is computed only along the dimension denoted by index m, which is explicitly written in the second line of (4.2.27). The coefficients $a(\vec{x}_\bullet, \vec{x}_n)$ can be interpreted as the relative similarities between the n^{th} query and all keys, and we will refer to them as attention weights.

Assuming that the input vectors $\vec{x}_n$ have dimension d_{in}, we can group them into $d_{in} \times N$ matrix $\mathbf{x}$. Doing the same for *values*, *queries* and *keys*, we get $\mathbf{v}$, $\mathbf{q}$ and $\mathbf{k}$, which can be used to write down the action of the attention layer as

$$\mathbf{Sa}(\mathbf{x}) = V(\mathbf{x}) \cdot \text{softmax}'\big(\mathbf{k}[\mathbf{x}]^T \mathbf{q}[\mathbf{x}]\big) \,. \tag{4.2.29}$$

In this formula, we explicitly include the dependence of the values, queries, and keys on the input $\mathbf{x}$, which makes it clear that the output is a non-linear function of the inputs despite the absence of non-linear activation.

Thus, the attention mechanism allows the exchange of information between elements of the sequence, allowing the output tokens in different parts of the sequence to be dependent on each other.

Compared to the original realisation, modern versions have several modifications:

- The dot products in the attention are scaled by the factor of $1/\sqrt{D_q}$, where D_q is the dimension of queries and keys such that

$$\mathbf{Sa}(\mathbf{x}) = V(\mathbf{x}) \cdot \text{softmax}'\big(\frac{\mathbf{k}[\mathbf{x}]^T \mathbf{q}[\mathbf{x}]}{\sqrt{D_q}}\big) \,. \tag{4.2.30}$$

  This is done to scale the product so that it has a unit standard deviation, assuming that the queries and keys also have a unit standard deviation.

- In the original formulation (4.2.29), no information about the position of the input element $\vec{x}_n$ is preserved. To bring this information back, a matrix $\mathbf{\Pi}$ is added to $\mathbf{x}$. Matrix $\mathbf{\Pi}$ can be chosen manually or learned during training. A version that we will see used later is *sinusoidal positional encoding* [147]. Matrix $\mathbf{\Pi}$ has the same dimensions as $\mathbf{x}$ and have the following components

$$\Pi_{2i,p} = \sin\left(\frac{p}{10000^{2i/d_{in}}}\right), \tag{4.2.31}$$

$$\Pi_{2i+1,p} = \cos\left(\frac{p}{10000^{2i/d_{in}}}\right). \tag{4.2.32}$$

- Usually, multiple self-attention mechanisms are applied in parallel. This is called *multi-head self-attention*. Each head h out of H heads has its own set of weights and biases to compute the value, query, and key, which are then combined using (4.2.29) to produce $\mathbf{Sa}_h(\mathbf{x})$. They are concatenated into the attention layer output $\mathbf{Sa}(\mathbf{x})$ in the end.

Transformer mechanism

The attention layer is only one part of the transformer mechanism suggested in [147]. The actual model contains a multi-head self-attention layer, followed by a fully connected NN. The intuition is that after the important information is extracted in the self-attention layer, it should be further transformed in a task-specific way by an NN.

In addition, there are two more components used to make the training of the transformer more robust

- To keep the gradients during backpropagation finite, after both the attention layer and a following NN, a *layer normalisation* (LayerNorm) is applied. It computes the empirical mean m and the standard deviation s of the output sequence, which are then used to rescale and shift the output h

$$h \to h' = \frac{h - m}{s + \epsilon}, \tag{4.2.33}$$

  where ϵ is a small constant that prevents division by zero. Then, the normalised sequence is rescaled using two learnable parameters γ and δ

$$h' \to \gamma h' + \delta. \tag{4.2.34}$$

- LayerNorm alone is not enough for stable training of deep networks. Another common technique is *residual or skip connections* [150]. It is achieved by adding the input to each network layer $f(\dots)$ back to the output

$$h_{n+1} = h_n + f_n(h_n). \tag{4.2.35}$$

  That way, the output of the network changes the input additively

$$X_n = X_{n-1} + F_{n-1}(X_{n-1}) = \tag{4.2.36}$$

$$= X_1 + \sum_{k=1}^{n-1} F_k(X_k), \tag{4.2.37}$$

  therefore, even if some layers have vanishing gradients, the entire network training will not be spoiled.

Tokenisation

Working with text data requires first transforming it into a form suitable for mathematical operations. This can be done by splitting it into small pieces, which can be anything from individual letters to words that form a vocabulary. Each item in the vocabulary is assigned a number-position in the vocabulary. This process is called *tokenisation*, and the chosen building blocks of the text are referred to as *tokens*.

It was found that the most efficient way to tokenise text is to use sub-word size groups of letters, which can sometimes be identified with linguistic morphemes. This way, the size of the vocabulary is reduced compared to using the whole words as tokens, while these larger chunks of words carry more information than individual letters.

Embeddings

Tokens naturally have structure in the sense that they carry some meaning as parts of the words and also have some notion of closeness, as some of them might be related via concepts. Therefore, their representation in terms of a single number is not useful.

One way to introduce more structure into the representation of tokens is to map the elements of the input sequence to vectors called *embeddings*. Such representation allows us to use concepts from vector algebra, such as dot product, orthogonality, or projection to operate with the tokens. For example, the closeness of token meaning can be measured by the dot product.

Pooling layer

In the standard transformer architecture, the number of elements in the output sequence is equal to the number of elements in the input sequence. However, for classification tasks, a single fixed-size output vector is required for each sequence. This can be considered as eliminating the time dimension T from the output, which can be achieved by a *pooling layer*. There are many variants in the literature, but the simplest are as follows:

- BERT-like pooling – introduce the special classification token `[CLS]` in the input, and then use the output at `[CLS]` for classification,
- average pooling – simply take the average of all outputs along the T dimension and use for classification,
- attention pooling – applying an attention mechanism to compute a weighted sum of sequence elements.

It was shown [151] that the latter two are somewhat better than the first approach and thus will be used later in this work.

Output

Before the output, there is usually a linear classification layer that projects from the embedding space to the output vector. Subsequently, a softmax function is applied to produce the probabilities of classes $\vec{p}$, which are used to sample the output tokens.

Loss functions

Because the output should be a label of the next token in a sequence, we need a way to measure the error between the predicted class and the ground truth label. This can be done by using a generalisation of the binary cross-entropy loss from (4.2.14) information – a Cross-Entropy loss

$$L = -\sum_{c=1}^{M} y_c \log(p_c)\,, \tag{4.2.38}$$

where M is the number of classes, y_c is a binary indicator (0 or 1) if class c is the correct class, and p_c is the predicted probability for class c.

Overall structure

The above-described elements are combined into a transformer model in many ways, depending on the task.

Figure 4.4 visualises these elements as a block scheme.

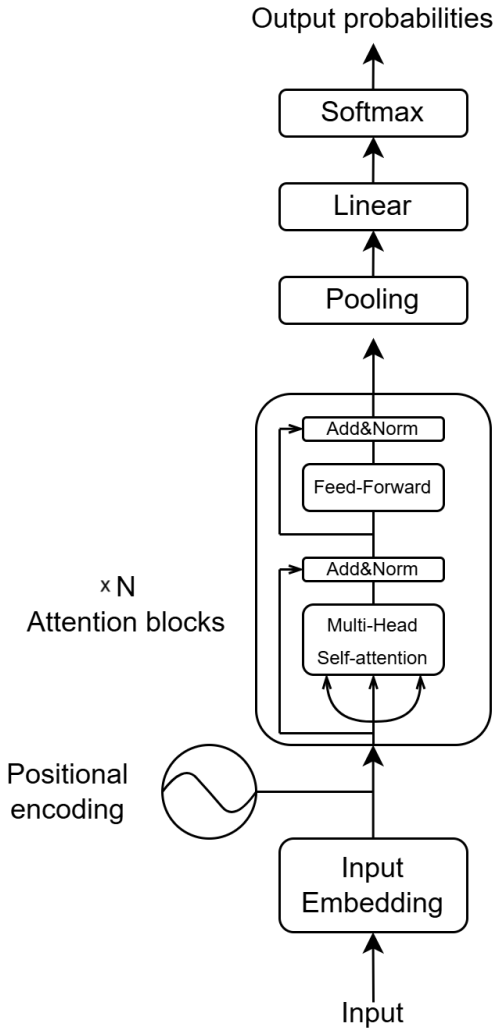


Figure 4.4: Transformer architecture.

Chapter 5

Machine Learning Clifford Invariants of ADE Coxeter Elements

This chapter presents a study of Clifford geometric invariants of Coxeter transformations in the A_8, D_8 and E_8 root systems in the experimental mathematics approach. The combinatorially large number of such invariants makes the application of Machine Learning and Data Science essential. We therefore employ these techniques to analyse the structure of the invariants, perform neural network classification and regression of them to provide an empirical foundation for further theoretical investigation[1].

5.1 Motivation and summary

Orthogonal transformations, such as rotations, and their invariants have important applications in engineering, such as moving cameras, robots and so on. Typically, such linear transformations are described by matrices, and their invariants are given by the determinant and trace, which appear in the highest and lowest coefficients of the characteristic polynomial.

An alternative description of orthogonal transformations can be provided using Clifford algebras introduced in [152]. In Clifford algebras, algebraic objects have a clearer geometric interpretation than in the standard matrix approach.

A particular type of invariants arising in this setup are Clifford geometric invariants. They are systematically related to geometric invariant spaces of the linear transformation and the coefficients in the characteristic polynomial, and the Cayley-Hamilton theorem. Calculating Clifford geometric invariants, which are expressed as multivectors, can be done systematically via simplicial derivatives, which we describe later. Recent works studied them from a practical [153], [154] and theoretical [154], [155], [156], [157] points of view. These types of transformations are also rotations and are particularly interesting because of their symmetry structures. In this work, we will study Clifford geometric invariants in the context of root systems[2], focusing on so-called called 'Coxeter elements' or 'Coxeter transformations'. These are a particular type of orthogonal transformation in reflection/Coxeter groups [159]. They are the group elements of the highest order (called the 'Coxeter number', h) and are all conjugate to each other. High-dimensional root systems are notoriously difficult to visualise, and projection into a distinguished plane (called a 'Coxeter plane') is a

[1]The work presented in this chapter is based on the collaborative study in [19]. The author of the thesis is responsible for the frequency analysis and ML classification work presented in Sections 5.3.2 and 5.5.1.

[2]See [158] for a detailed introduction to root systems and Clifford algebras.

common way of visualising the geometry. In these planes, the Coxeter elements just act by h-fold rotations. Root systems are determined by a subset called the 'simple roots', which act as a basis for the vector space and each determines a 'simple reflection' in the hyperplane to which they are orthogonal[3]. A Coxeter element is then just given by multiplying each of the simple reflections once in some permutation order, which at the versor level is just encoded by multiplying together the root vectors in the Clifford algebra directly, doubly covering the orthogonal transformation. This set of permutations giving rise to a set of Coxeter versors will be the focus of this paper, as these allow us to calculate the full set of invariants from them, which we will refer to as the set of characteristic multivectors (SOCM).

In this work, an experimental mathematics approach was adopted: first, a dataset of algebraic data was generated using computational algebra techniques and high-performance computing (HPC); this dataset was then mined by applying the standard data science toolkit in order to find patterns that were not obvious from an analytical perspective. This approach was previously explored in works [160], [161], [162], [163], [164], [165]. Mathematicians often calculate examples of interest by hand to formulate or test hypotheses. Essentially, this computational algebra approach automates and scales up such an approach, and turns the problem into a data analysis task[4]. One can either calculate a very large number of examples and analyse these statistically via 'data analysis'; or in other cases of interest, it may be possible to calculate *all* the cases exhaustively and analyse the patterns that emerge, which can help with hypothesis formulation and theorem proving.

We consider the three 8-dimensional root systems A_8, D_8 and E_8, which are of wider interest in terms of exceptional E_8 and ADE patterns [167], [168], [169]. The number of Coxeter elements grows factorially with the dimension of the root system. Eight dimensions allow $8! = 40320$ permutations of the simple roots, which can all be explicitly calculated with HPC, this is large enough for data science techniques to be practically effective. They give rise to linear transformations (Coxeter transformations) whose simplicial derivatives can be taken in an automated way, and whose characteristic multivectors are computed as the output. Of course, the actual number of Coxeter versors will be fewer, firstly because of the fact that the simple roots can be decomposed into two sets that are mutually orthogonal within sets (leading to a $k!$ reduction whenever k orthogonal simple roots are grouped together), but secondly because such roots are also more widely orthogonal to other roots outside of the sets, as given by the adjacency in the Dynkin diagram such that two roots are orthogonal if there exists no edge between them. So in practice, there will be some degeneracy in the mapping from the permutations to the Coxeter versors, which will result in a significantly reduced set of invariants.

We organise this work as follows. In Section 5.2, we introduce some details on the aforementioned Clifford simplicial derivatives and invariants, as well as root systems and Coxeter transformations. We then discuss in Section 5.3 what datasets we are mining, and how they were generated using computational algebra. This section also contains an exploratory data analysis around the number of distinct invariants as well as the connectivity structure of the bivector invariants. We then move on to Machine Learning in Section 5.5; in particular, we discuss predictive performance as well as ternary classification tasks, before moving on to gradient saliency sensitivity on the input and Principal Component Analysis. We con-

[3]The existence of the Coxeter plane relies on the simple roots admitting a separation into two sets that are mutually orthogonal within each set, often visualised as a bipartite (alternating) colouring of the corresponding Dynkin diagram.

[4]Clifford algebra multivector computations can easily be performed in such a `python` HPC setup using the `galgebra` package [166].

clude in Section 5.7. Computer code scripts and data can be found on GitHub[5].

5.2 Background

A detailed introduction to root systems and Clifford algebras is available in [158], so here we will be brief. A root system lives in the arena of a vector space with a scalar product (which immediately allows one to consider the corresponding Clifford algebra). It is a collection of vectors (called 'roots', and customarily denoted α) in that vector space which is invariant under all the reflections in the hyperplanes to which the root vectors are perpendicular. We will only consider root systems with roots of the same length, which can be assumed to be normalised[6]. Such reflections in the normal hyperplanes are given by $x \to x - 2(x \cdot n)n$, where x is the vector to be transformed and n is a unit normal to the hyperplane.

A subset called 'simple roots' is sufficient to write all roots as (in our case) integer linear combinations of this basis of simple roots, whilst their corresponding reflections, the 'simple reflections', generate the reflection group. Taking these simple reflections all exactly once leads to interesting types of group elements called 'Coxeter elements'. They are of the same order h (the 'Coxeter number') and have invariant planes, called 'Coxeter planes', which are useful for visualising root systems in any dimension (via projection into these planes). These reflection groups have interesting integer – in fact, prime – invariants that are characteristic of the geometry, called 'exponents' m. This name derives from the fact that Coxeter elements act on different invariant planes by h-fold rotations by m times $2\pi/h$, which is usually interpreted as a complex eigenvalue of the Coxeter element (even though we are, by assumption, in a real vector space). The root system geometry can also be encoded in diagrammatic form (called 'Coxeter-Dynkin diagram'), where each simple root corresponds to a node and orthogonal nodes are not linked, whilst roots at $2\pi/3$ angles are connected with a link (we will only be considering such 'simply-laced' examples, see Figure 5.1). Likewise, our simply-laced examples are tree-like and admit an alternate colouring (or 'bipartite', e.g. black and white). This effectively means that all black roots are orthogonal to each other, and likewise for the white roots. This colouring means that there are distinguished types of Coxeter elements where first all the black reflections are taken, and then all the white (or the other way round). We will call these 'bipartite' Coxeter elements. This bipartite colouring also implies the existence of the Coxeter plane via a more complex argument, the details of which we will omit here, but which relies on the adjacency matrix of the Dynkin diagram having a distinguished largest eigenvalue and corresponding eigenvector, the Perron-Frobenius eigenvector (which *will* make an appearance below). In our labelling of the 8 simple roots for A_8, D_8 and E_8, α_1 to α_7 make one long string. The different diagrams arise depending on where the 8th root α_8 attaches: at the terminal node α_7 for A_8 (leading to bilateral symmetry), at the penultimate node α_6 for D_8 (leading to permutation symmetry of the terminal nodes), or α_5 for E_8[7].

As mentioned above, Clifford algebras can be constructed when one is working in an n-dimensional vector space with an inner product, giving rise to a 2^n-dimensional algebra of 'multivectors'. The scalar product is given as the symmetric part of the geometric product, i.e. $a \cdot b = \frac{1}{2}(ab+ba)$[8]. Substituting this in the reflection formula above results in a cancellation

[5] `https://github.com/DimaDroid/ML_Clifford_Invariants.git`

[6] Note this is different from the normalisation convention used in Lie theory.

[7] Note that attaching to other roots is symmetry-equivalent to the options just mentioned except attaching to α_4, which leads to something called affine E_7, or $\tilde{E}_7$.

[8] The outer product $a \wedge b = \frac{1}{2}(ab - ba)$ is the antisymmetric part, is a bivector and determines the plane that

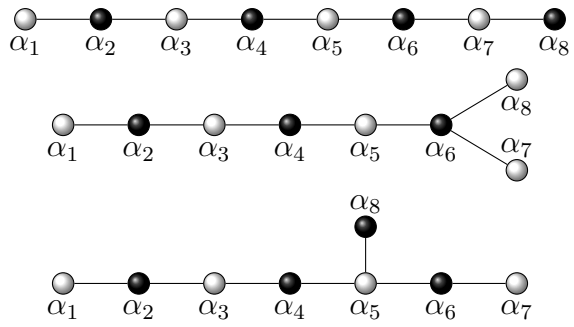


Figure 5.1: The diagrams of the 8-dimensional simply-laced root systems A_8, D_8 and E_8 (vertically downwards respectively), along with our labelling for the simple roots and a bipartite colouring.

which leads to the uniquely simple 'sandwiching' reflection formula in Clifford algebras

$$x \to x - 2(x \cdot n)n = -nxn. \tag{5.2.1}$$

Both n and $-n$ doubly cover the same reflection. Via the Cartan-Dieudonné theorem, orthogonal transformations are just products of such reflections, so that one can build up

$$x \to \pm n_k \cdots n_1 x n_1 \cdots n_k = \pm \tilde{A} x A \tag{5.2.2}$$

such transformations via defining multivectors that are the products of normal vectors which encode the reflection hyperplanes, $A = n_1 \cdots n_k$ (called 'versors'), and a tilde denotes reversing the order of these vectors in the product. These versors again doubly cover the transformation.

We discuss here for a moment how this applies when the orthogonal transformation is a Coxeter element. In traditional root system notation, the simple reflections are denoted s_i such that a Coxeter element is denoted $w = s_1 \cdots s_n$. In the above versor framework, the reflections are encoded by the root vectors themselves (as a double cover), whilst the multivectors W that one gets from multiplying the simple roots together $\alpha_1 \cdots \alpha_n$ doubly cover w

$$wx \to \pm \alpha_k \cdots \alpha_1 x \alpha_1 \cdots \alpha_k = \pm \tilde{W} x W. \tag{5.2.3}$$

We return now to the setting of linear transformations in Clifford algebras more generally. Let us denote this linear transformation by $f(x)$. In order to calculate the desired invariants of this linear transformation (the SOCM), we define the concept of 'simplicial derivatives'.

First, let $\{a_k\}, k = 1, \ldots, n$ denote a frame, i.e. a basis. Often we use either a Euclidean basis e_i or the basis of simple roots, α_i. We denote by $\{a^k\}$ its reciprocal frame such that $a^i \cdot a_j = \delta^i_j$. In a Euclidean basis, this is effectively the basis itself; for a basis of simple roots, the reciprocals are more commonly known as co-roots (up to a different conventional normalisation factor). We also define $b_k = f(a_k)$ as the transformation acting on the basis frame vectors. The rth simplicial derivative is then essentially defined as a combinatorial object

$$\partial_{(r)} f_{(r)} = \sum (a^{j_r} \wedge \cdots \wedge a^{j_1})(b_{j_1} \wedge \cdots \wedge b_{j_r}) \tag{5.2.4}$$

two vectors generically span.

with sum over $0 < j_1 < \cdots < j_r \leq n$ [9]. These simplicial derivatives are invariants of the linear transformation and are therefore 'characteristic multivectors' with geometric significance.

Now [152] showed that it is the scalar parts of these geometric invariants (denoted by $\partial_{(s)} * f_{(s)}$) that constitute the coefficients in the Cayley-Hamilton theorem

$$C_f(\lambda) = \sum_{s=0}^{m} (-\lambda)^{m-s} \partial_{(s)} * f_{(s)}$$

(where $\partial_{(0)} * f_{(0)}$ is interpreted as 1) and the characteristic polynomial

$$\sum_{s=0}^{m} (-1)^{m-s} \partial_{(s)} * f_{(s)} f^{m-s}(a) = 0$$

for any vector a (where $f^0(a)$ is interpreted as a).

One can explicitly verify this for our examples. Using the `galgebra` package, one can perform calculations in the 256-dimensional multivector algebra, calculating Coxeter versors from permutations of the simple roots, and from that simplicial derivatives and geometric invariants. We will refer to the simplicial derivatives $\partial_{(r)} f_{(r)}$ as the invariant of order r or Inv_r (and to the full set as SOCM).

Since we are considering an even orthogonal transformation in an 8-dimensional space, we get some interesting structure in these invariants (see Table 5.1): firstly, we note that only even multivectors occur (in principle, this allows us to reduce the length of the 256-dimensional multivectors by half). Secondly, the lowest order invariant only has a scalar part (trivially), the next picks up a bivector term, the next one a quadrivector term, the next a sextivector, till finally Inv_4 (generically) has a pseudoscalar term. Then it decreases again. Thirdly, in our case, we have a certain 'mirror symmetry', where the top half of the Table is equal to the bottom half, though this is not generally the case. In fact, all these pieces, which we could denote by Inv_r^k, are separately invariant under the Coxeter versor: $\tilde{W}\,\mathrm{Inv}_r^k\,W = \mathrm{Inv}_r^k$. So these Inv_r^k are eigenmultivectors of the Coxeter element of grade k, but they do not have to be k-blades (i.e. be able to be written as the outer product of k vectors[10]).

	Subinvariant				
Invariants by Order	scalar	bivector	quadrivector	sextivector	pseudoscalar
Inv_0	X				
Inv_1	X	X			
Inv_2	X	X	X		
Inv_3	X	X	X	X	
Inv_4	X	X	X	X	X
Inv_5	X	X	X	X	
Inv_6	X	X	X		
Inv_7	X	X			
Inv_8	X				

Table 5.1: Structure of the characteristic multivectors: non-zero grades are indicated by an X.

[9] This is due to the original notion of a multivector derivative essentially being equivalent to a projection.
[10] This was also noticed in the example in [153].

So among other multivector components, e.g. for E_8, we in particular have 4 invariant bivectors from the invariants. It turns out that these are orthogonal. The Coxeter element also acts on 4 invariant orthogonal bivectors (giving planes, and they are blades by construction) via the Coxeter plane construction, so there is an immediate question of how our characteristic multivectors relate to exponents and degrees. In fact, we will say here already that for E_8 one can show that the two sets of 4 orthogonal eigenvectors (from the SOCM and the Coxeter construction) span the same 4d-subspace of the 28d bivector space. Reflection groups can also have other interesting invariant subspaces, such as two H_4-invariant subspaces in E_8 [170], [171].

5.3 Datasets

We choose dimension 8 because of the following compromise: $8! = 40320$ gives us something resembling 'big data' which is accessible to data science techniques, whilst being computationally tractable[11]. It is also the last dimension in which there are three simply-laced root systems, with the exceptional E_8 adding some variety to the A_n and D_n families that exist in arbitrary dimensions. So we select A_8, D_8 and E_8, as this gives us scope for three-way (ternary) classification tasks, and ADE patterns are of course of wider interest.

The input vectors are the set of permutations in 8 elements, e.g. (0,1,2,3,4,5,6,7), labelling the simple roots α_1 through to α_8 and encoding in which order the simple roots are taken in for computing the Coxeter versor. The outputs are the 9 invariants $\{\mathrm{Inv}_0, \ldots, \mathrm{Inv}_8\}$ as multivectors.

$$\text{input} = (0,1,2,3,4,5,6,7) = \alpha_1 \ldots \alpha_8 \rightarrow \{\underset{0}{\mathrm{Inv}}, \ldots, \underset{8}{\mathrm{Inv}}\} = \text{SOCM} = \text{output} \tag{5.3.1}$$

In 8 dimensions, the multivector invariants have 256 components (some of which are trivial[12]).

5.3.1 Data Generation

The computational algebra approach followed here used `python` with the `galgebra` package for multivector computations [166]. Exploratory analysis for single permutations was performed in Jupyter notebooks, but once parallelised, the computations were run on clusters at Queen Mary, University of London, City, University of London, and University of Leeds. Data and Code can be found on GitHub. We performed computations both in a Euclidean basis[13], which is a bit more straightforward, and the basis of simple roots via the multivector basis that it induces, which is more meaningful geometrically and less dependent on the choice of simple roots in the Euclidean basis. According to our earlier discussion around Table 5.1, we can also extract different grades of these invariants (e.g. scalar, bivector, etc), which we refer to as 'subinvariants' Inv_r^k, from the full set (SOCM).

[11] With the caveat that there is degeneracy in the permutations leading to the same or similar Coxeter elements and thus invariants, reducing the true number of different output vectors. Although it was not obvious from the beginning, especially for E_9 and D_8.

[12] Since the odd components are typically 0, one could reduce this if needed, but for completeness and generalizability, we haven't.

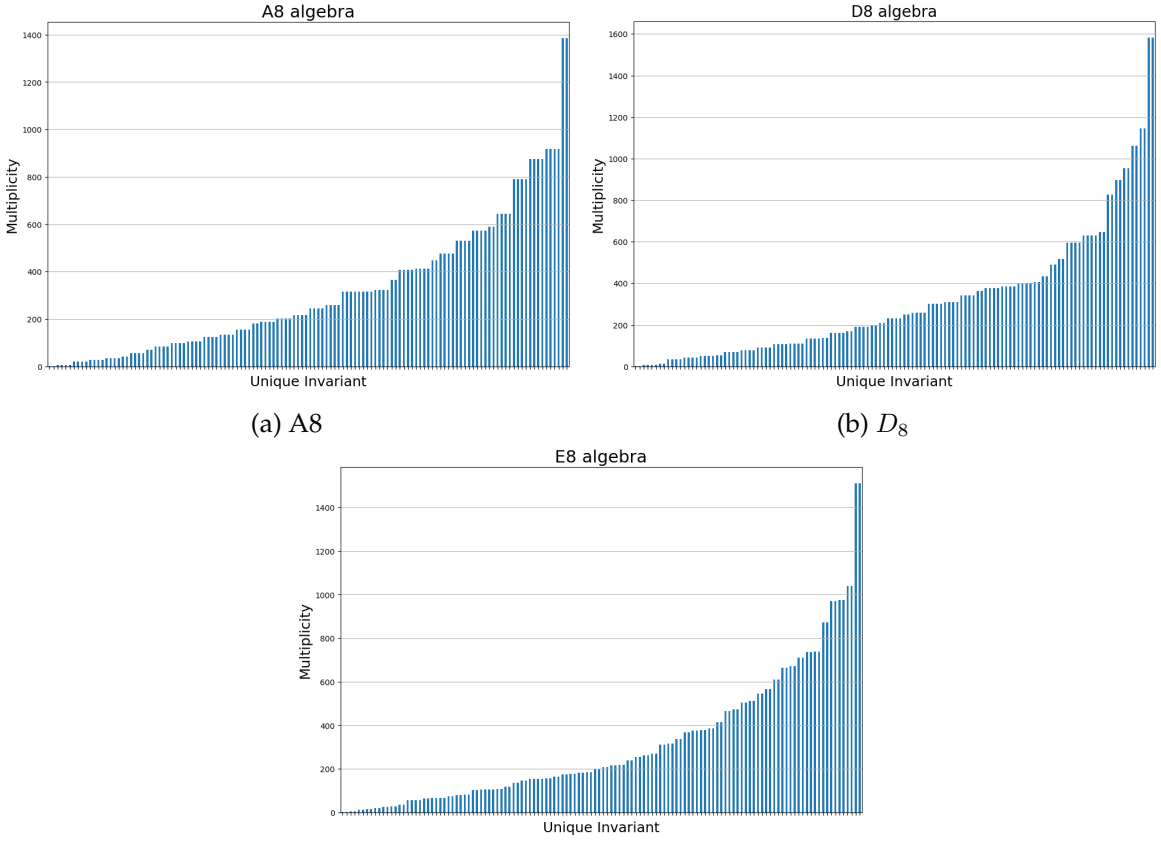
[13] Roots in the root system are often defined as columns of components in the Euclidean orthonormal basis in some higher-dimensional space. However, since the set of simple roots can generate the root system via addition, we can also take simple roots as a basis, although not orthonormal. While this may seem more complicated, it is more meaningful geometrically because everything we compute in geometric algebra using simple roots can be eventually expressed in the basis of simple roots.

5.3.2 Frequency Analysis

As was mentioned previously, for each of the A_8, D_8 and E_8 root systems, there are $8! = 40320$ permutations of 8 root vectors from which we can construct the corresponding Coxeter elements and the 9 geometric invariants.

Each invariant is a sum of 8 subinvariants written in terms of the wedge product of a different number of basis vectors for the 8-dimensional vector space. These 8 subinvariants are scalar, vector, bivector, trivector, quadrivector, and so on up to 8-vector pseudoscalar. The components for each of the subinvariants can be written down in a chosen basis, e.g. in terms of simple roots. This allows us to compare SOCMs corresponding to different Coxeter elements, or invariants of a chosen order, or focus on subinvariants within the invariant of a chosen order.

These exhaustive computational algebra calculations already show interesting results. On the highest level, we compare components of SOCMs and find that, although there are 40320 SOCMs we can construct, only 128 are distinct, and this is the same number for each of the algebras. We can think of these as 'classes' of SOCMs with the same components. Each class has a certain number of representatives[14] in it, which we call frequency. Although the number of classes for each of the algebras is the same, the frequencies of individual classes differ, see Figure 5.2.



(a) A8

(b) D_8

(c) E8

Figure 5.2: Sorted multiplicities of the 128 unique SOCMs, for each root system considered: A_8, D_8, E_8 respectively. A_8 is mostly quadruplets, E_8 mostly doublets and D_8 half-and-half.

In the following, we will be using two types of operations on permutations:

[14] They have a different order of roots in a permutation encoding Coxeter versor.

- Inversion – we say that two permutations are related by inversion if the order of simple roots for these permutations is reversed relative to each other, e.g. (0,1,2,3,4,5,6,7) and (7,6,5,4,3,2,1,0).

- $B \leftrightarrow W$ – following the bipartite colouring of black roots and white roots, we can reduce (with some degeneracy) a permutation to a black and white 'barcode'. For example, (2,4,6,8,1,3,5,7) becomes '$\bullet\bullet\bullet\bullet\circ\circ\circ\circ$'. We say that two permutations are related by $B \leftrightarrow W$ if we replace black roots with white roots and vice versa.

There are some common features among all three algebras:

- The frequency values come in groups of 2, 4 or 8 classes, which have the same frequency. We call these groups of classes with the same frequency as Doublets, Quadruplets and Octuplets, respectively;

- The highest frequencies appear in Doublets;

- For Doublets, permutations of the class elements in one class are related by inversion to permutations of the class elements in another class. Quadruplets are essentially two Doublets with the same frequency, and Octuplets are two Quadruplets with the same frequency.

Some other features which are different:

- For A_8:

  - Frequencies of all classes are odd numbers;
  - There is a Doublet with the lowest frequency equal to 1 (i.e. two unique invariants). Classes in this doublet are represented by permutations (0,1,2,3,4,5,6,7) and (7,6,5,4,3,2,1,0)[15];
  - There is a Doublet with the highest frequency equal to 1385. This Doublet consists of two invariants which are given by bipartite Coxeter elements: one of them is given by a Coxeter element with first 4 black roots with increased root number and then 4 white ones with increased root number as well; the second one is very similar and has first 4 white roots and then 4 black ones with increased root number in both subsets;
  - For Quadruplets, there are pairs of classes that are related by inversion. In addition, these pairs within the Quadruplet are related to each other by $B \leftrightarrow W$. Presumably, having inverse barcodes signifies similar combinatorial properties that result in the same frequency.
  - As was mentioned before, in Doublets, two classes within it are related by inversion. In addition, the two classes are related to each other by $B \leftrightarrow W$ symmetry: if we assign a black or white colour to every simple root according to Figure 5.1, we get a black and white 'barcode' for a permutation encoding a Coxeter element. One can check that the barcode for the first class is the inversion (change black to white and vice versa) of the barcode for the other class within the Doublet, i.e. the Doublets are self-dual under $B \leftrightarrow W$.

[15] One might call them 'maximally non-commuting permutations', where none of the roots adjacent in the permutation are orthogonal.

  - In some Quadruplets, there is even more $B \leftrightarrow W$ inversion symmetry: $B \leftrightarrow W$ symmetry between the pairs of classes that are related by inversion is enriched by the $B \leftrightarrow W$ symmetry within the pairs. This is because the barcode mapping is degenerate, i.e. non-equivalent permutations can give rise to the same barcode;
  - An Octuplet appears as two Quadruplets with the same frequency;
  - There are 8 Doublets, 26 Quadruplets and 1 Octuplet in total.

- For D_8:
  - All frequencies are even numbers;
  - There are no unique invariants, a Doublet with a frequency equal to 2, and a Doublet with the highest frequency equal to 1582;
  - No signs of $B \leftrightarrow W$ symmetry;
  - There are 20 Doublets and 22 Quadruplets.

- For E_8:
  - All frequencies are odd numbers;
  - There are no unique invariants, a Doublet with the lowest frequency equal to 3 and a Doublet with the highest frequency equal to 1511;
  - No signs of $B \leftrightarrow W$ symmetry;
  - There are 58 Doublets and 3 Quadruplets.

On the level of subinvariants within the invariants, one can perform the same analysis and find frequencies given in Tables 5.2 and 5.3. The tables for A_8 and E_8 are identical; all three groups have the same frequencies for bivector and quadrivector subinvariants. Empty cells denote the fact that all subinvariants for this order of invariants are trivially zero (some are also less-trivially zero).

Another interesting thing to look at is the frequencies of invariants and subinvariants with the identification of objects that differ up to an overall minus sign. We find that this modification does not alter the frequencies of the full invariants, however, it does change the frequencies of subinvariants, see Tables 5.4, 5.5 and 5.6. We see that frequencies for bivector and sextuvector (and pseudoscalar for A_8) subinvariants for A_8 and D_8 are halved, meaning that half of these subinvariants differ from the other half by a minus sign. At the same time, scalar and quadrivector subinvariants are unchanged. The picture is different for E_8, where bivector, quadrivector and sextuvector frequencies change non-trivially under sign identification.

The idea behind these observations is to understand the symmetries of the root system. The frequency of multiplicities for the three algebras should be determined by the permutation of these symmetries. It is rather simple to determine the number of unique invariants and explain the existence of Doublets and Quadruplets for the A_8 algebra due to its simple Dynkin diagram, but much harder for the D_8 and E_8 algebras.

	Subinvariant				
Invariant Order	scalar	bivector	quadrivector	sextuvector	pseudoscalar
Inv_0	1				
Inv_1	1	128			
Inv_2	1	128	64		
Inv_3	1	128	64	128	
Inv_4	1	128	64	128	2
Inv_5	1	128	64	128	
Inv_6	1	128	64		
Inv_7	1	128			
Inv_8	1				

Table 5.2: Frequencies of subinvariants for A_8/E_8 group. Empty cells denote the fact that all subinvariants for this order of invariants are trivially zero.

	Subinvariant				
Invariant Order	scalar	bivector	quadrivector	sextuvector	pseudoscalar
Inv_0	1				
Inv_1	1	128			
Inv_2	0	128	64		
Inv_3	0	128	64	32	
Inv_4	0	128	64	32	0
Inv_5	0	128	64	32	
Inv_6	0	128	64		
Inv_7	1	128			
Inv_8	1				

Table 5.3: Frequencies of subinvariants for D_8 group. Empty cells denote the fact that all subinvariants for this order of invariants are trivially zero. But there are also some non-trivial zeroes to do with the D_8 geometry, in which the factorisation of the Coxeter element into orthogonal eigenspaces contains two true reflections, also signalled by having two exponents of $h/2$.

5.4 Bivector Subinvariants

Now we are restricting our focus to the bivector parts of the invariants, which, as subinvariants, are of particular interest since bivectors generate planes for rotation (such as the Coxeter plane central to the study of these root systems). Each of the bivector subinvariants has 28 entries, corresponding to the $\binom{8}{2}$ combinations that form a basis for the bivector subspace, whether this is in a Euclidean basis or the basis of simple roots. Here, we will be working in the basis of simple roots. Each bivector subinvariant hence takes the form: $\sum_{i,j=1|i<j}^{8} c_{ij}(\alpha_i \wedge \alpha_j)$, for the 8 simple root basis vectors α_i, and general coefficients c_{ij}, which turn out to be even integers. This is motivated by the observation that, rather intriguingly, the bivector part of the bipartite E_8 Coxeter element gives precisely rise to the E_8 diagram, etc.

	Subinvariant				
Invariant Order	scalar	bivector	quadrivector	sextuvector	pseudoscalar
Inv_0	1				
Inv_1	1	64			
Inv_2	1	64	64		
Inv_3	1	64	64	64	
Inv_4	1	64	64	64	1
Inv_5	1	64	64	64	
Inv_6	1	64	64		
Inv_7	1	64			
Inv_8	1				

Table 5.4: Frequencies of subinvariants for A_8 group up to an overall minus sign. Empty cells denote the fact that all subinvariants for this order of invariants are trivially zero.

	Subinvariant				
Invariant Order	scalar	bivector	quadrivector	sextuvector	pseudoscalar
Inv_0	1				
Inv_1	1	40			
Inv_2	1	40	64		
Inv_3	1	52	48	36	
Inv_4	1	64	64	64	1
Inv_5	1	52	48	36	
Inv_6	1	40	64		
Inv_7	1	40			
Inv_8	1				

Table 5.5: Frequencies of subinvariants for E_8 group up to an overall minus sign. Empty cells denote the fact that all subinvariants for this order of invariants are trivially zero.

	Subinvariant				
Invariant Order	scalar	bivector	quadrivector	sextuvector	pseudoscalar
Inv_0	1				
Inv_1	1	64			
Inv_2	0	64	64		
Inv_3	0	64	64	16	
Inv_4	0	64	64	16	0
Inv_5	0	64	64	16	
Inv_6	0	64	64		
Inv_7	1	64			
Inv_8	1				

Table 5.6: Frequencies of subinvariants for D_8 group up to an overall minus sign. Empty cells denote the fact that all subinvariants for this order of invariants are trivially zero. But there are also some non-trivial zeroes to do with the D_8 geometry, in which the factorisation of the Coxeter element into orthogonal eigenspaces contains two true reflections, also signalled by having two exponents of $h/2$.

5.4.1 Interpretation as Graphs

From each bivector subinvariant, one can construct a graph. This is done by associating a vertex to each simple root, and including the edge between vertices i and j if $c_{ij} \neq 0$. This construction method manifestly creates undirected unweighted simple graphs (with no loops as $c_{ii} = 0 \; \forall i$, and at most one edge between any pair of vertices). The generated graph is practically constructed via a symmetric adjacency matrix, with binary entries, such that c_{ij} is taken as the upper triangle of a symmetric matrix with any non-zero entries converted to 1. Since the adjacency matrices are symmetric their eigenvalues are all real, and one can begin to analyse their eigenspectra[16].

The Perron-Frobenius theorem [172], [173] asserts that square matrices with positive integer coefficients have a unique largest real eigenvalue. In particular, for undirected graph adjacency matrices, this maximum eigenvalue takes value in the range $(0, n-1]$, for graphs with n vertices[17]. Furthermore, it turns out that the A, D, & E Dynkin diagrams are particularly special in the space of undirected graphs, in that they are the *only* connected graphs whose maximum eigenvalue < 2 by Smith's theorem [174]. In fact, the Coxeter elements of bipartite form were observed to just give the Coxeter-Dynkin diagrams of the A_8, D_8 and E_8 root systems. Since bivector graphs can be more generally induced by all forms of Coxeter elements, this motivates the study of the maximum eigenvalues for all the respective undirected graphs.

Returning to our databases, except for the trivial zero invariant formed from the bivector subinvariant of Inv_0 and Inv_8, an initial unanticipated observation is that each of the graphs constructed from each of the bivector subinvariants across the 3 databases is connected. In addition to this, there is no overlap of bivector subinvariants between algebras (excluding the trivial zero invariant). Also, there is no overlap of bivector subinvariants between the orders of invariants they come from, within each of the algebras. However, there is a small repetition of adjacency matrices (i.e. after reducing non-zero entries to 1), and also graphs. Specifically, there are A_8: (0, 38, 12), D_8: (0, 1, 6), E_8: (0, 0, 25) repeated (subinvariants, adjacencies, graphs) between orders 1 to 4, for each algebra respectively.

Analysing the multiplicities of the bivector subinvariants, for (A_8, D_8, E_8), there are (513, 513, 513) distinct subinvariants across all orders for each algebra, respectively. When considering the undirected adjacency matrices constructed from these (out of $2^{28} \sim 2.7 \times 10^9$ possible undirected adjacency matrices), these bivector subinvariants reduce to (219, 256, 251) distinct adjacency matrices respectively. These matrices then further reduce to respectively (88, 144, 137) non-isomorphic graphs (out of 11117 possible non-isomorphic graphs [175])[18].

Now, in examining the distribution of the maximum eigenvalues, we first note that the trivial zero invariant, which is equivalent to the empty graph, has all eigenvalues zero, so it is omitted in the following analysis. To set a baseline for comparison, we sample as many random connected adjacency matrices as non-zero bivector subinvariants occur in each dataset (282240), compute their maximum eigenvalues, and plot the respective histogram of multiplicities in Figure 5.3. The maximum eigenvalues for the adjacency matrices constructed from each bivector subinvariant were then computed for each algebra's dataset,

[16] Note that one may also create a directed weighted graph by setting the adjacency matrix upper triangle to be c_{ij}; however as the matrix is antisymmetric, eigenvalues are complex, and hence cannot be sorted sensibly for analysis.

[17] The upper bound is saturated by the complete graph on n vertices.

[18] Noting that the trivial zero invariant (all 28 coefficients $c_{ij} = 0$) contributes a subinvariant, an adjacency matrix, and an empty graph to the counts for each algebra.

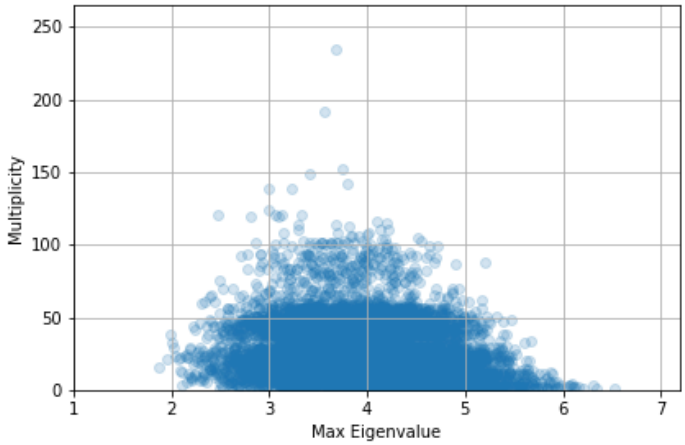
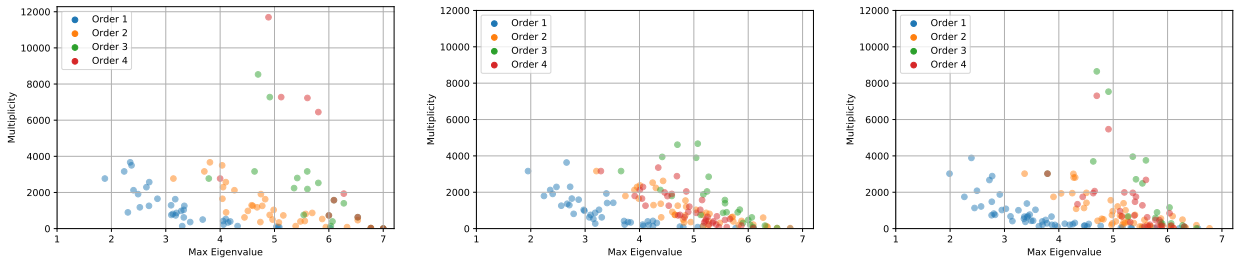


Figure 5.3: Distributions of the maximum eigenvalues for 282240 random connected matrices (of which 282086 are unique matrices, overall having 9741 unique eigenvalues).



(a) A_8 Subinvariants (b) D_8 Subinvariants (c) E_8 Subinvariants

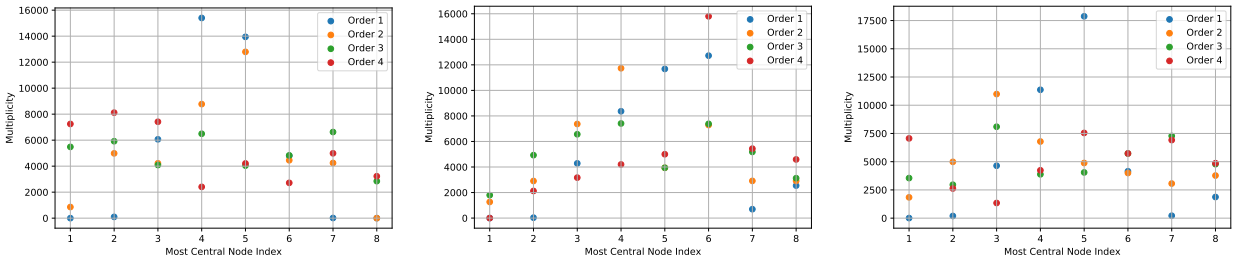
Figure 5.4: Distributions of the maximum eigenvalues for each of the bivector subinvariants for each of the considered algebras: A_8, D_8, E_8, respectively. Data includes all 282240 non-empty bivector subinvariants, coloured according to which order invariant they correspond to.

and histograms of their distributions for each algebra are shown in Figure 5.4, coloured according to the invariant order that they came from. In each class corresponding to invariant orders 1 to 4 (i.e. Inv_1 to Inv_4), there are A_8: [36, 34, 18, 11], D_8: [43, 41, 24, 36], E_8: [54, 49, 18, 30] distinct eigenvalues respectively with multiplicities as shown in the plots. Note that all these multiplicities reduce to 1 when considering the unique non-isomorphic graphs at each order.

From the plots, it can be seen that the distributions do appear to roughly follow a partition according to the order of the invariant they come from. This is perhaps hinting at how these different order subinvariants span different subspaces of the full space of subinvariants, as dictated by their eigendecompositions. Additionally, the actual A_8, D_8, and E_8 graphs occur as bivector subinvariant graphs for a large number of the Inv_1's *only* in each respective root system's dataset (the point with maximum eigenvalue below 2 in the A_8 plot is the A_8 graph of Figure 5.1, etc). Presumably, these are due to Coxeter elements in bipartite form and are in accordance with Smith's theorem.

5.4.2 Eigenvector Centrality

The maximum eigenvalue of a non-negative matrix has a corresponding eigenvector with



(a) A_8 Subinvariants (b) D_8 Subinvariants (c) E_8 Subinvariants

Figure 5.5: The multiplicities that each of the 8 graph nodes (i.e. simple roots α_i) exists as the most central node in a bivector subinvariant graph, for all graphs across all invariant orders for each of the considered root systems: A_8, D_8, E_8 respectively.

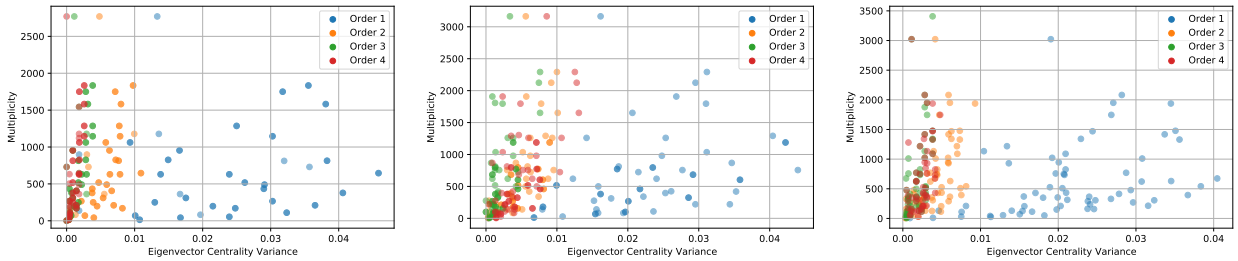
exclusively non-negative entries, as also dictated by the Perron-Frobenius theorem. One can then associate each of these normalised non-negative entries with a centrality score for the graph node with the corresponding index. This is known as *eigenvector centrality* [176].

For these bivector subinvariant graphs, there are 8 nodes (corresponding to the simple roots) and hence 8 respective centrality scores which can be computed for each graph for each invariant order across each root system. Since the centrality scores are normalised, when examining the distribution of measures across the nodes, it is most interesting to consider: (1) the most central node with the highest score; as well as (2) the score distribution variance. Respectively, these then indicate which parts of the graph are most important to the connectivity (and hence the most significant bivector contribution to its graph structure); and the extent to which this significance is polarised towards the requirement on this most central node to ensure connectivity. For instance, for the D_8 and E_8 Dynkin diagrams, the triply connected simple root is 6 and 5 respectively, so we would expect these to have the highest centrality. Likewise, the middle roots 4 and 5 in A_8 should be the most central. But these Dynkin diagrams are in some way minimal, and other bivector diagrams will be 'more fully connected', so we expect centrality of different roots to change for more general Coxeter elements.

To examine this behaviour in the root systems considered, the eigenvector corresponding to the previously studied largest eigenvalue was computed for each bivector subinvariant across all invariant orders for each root system. The node index of the most central node was then identified, and the variance of the centrality measure distribution was calculated. The multiplicity distributions of these centrality distribution measures are shown in Figure 5.5 for the node index of the most central node, and Figure 5.6 for the variance in centrality scores across each bivector subinvariant.

The results in Figure 5.5 show some consistency in which nodes are the most central between the root systems. In all cases, the order 1 invariants (Inv_1) have the most skewed distributions, with the first 2 nodes never being the most central, whilst the last nodes are the most central infrequently[19]. From this one can deduce that the basis invariants $\{a_4, a_5, a_6\}$ are most significant to the Inv_1's graph's connectivity, as in our labelling of the simple roots $\{a_1, a_2, a_3\}$ are the start of a long string, and are thus somewhat less central. But the relative multiplicities between the nodes can be used to differentiate the root systems. Within the set of each root system's Inv_1 invariants, the Dynkin diagrams themselves are included as graphs. It is hence not surprising to see that the D_8 and E_8 most central nodes have indeed maximum multiplicity as the most central node for a_6 and a_5 respectively, where the Dynkin

[19]We note here that graph nodes have no intrinsic order; the one chosen here matches the basis order.



(a) A_8 Subinvariants (b) D_8 Subinvariants (c) E_8 Subinvariants

Figure 5.6: The multiplicities of the variances across the distribution of eigenvector centrality scores for each bivector subinvariant graph. These variances were computed for each graph across all invariant orders for each of the considered root systems: A_8, D_8, E_8 respectively.

diagram nodes have degree 3. Equivalently, the A_8 Inv_1 invariants have a more symmetric distribution of the most central node, with the highest multiplicity for a_4 (and with some numerical differences, a_5) matching the A_8 Dynkin diagram. These results corroborate nicely the similarity of the graphs within each order and our earlier assertion that the graphs for higher order invariants 'become more connected'.

Considering the higher-order bivector subinvariant graphs, the range of multiplicities is noticeably lower. However, range does not strictly decrease as order increases. Order 2 graphs have a similarly noticeable skew towards more nodes closer to the middle of the basis order being more central, which either does not occur or is not significant enough to conclude for orders 3 and 4. For A_8 and E_8, the order 3 and 4 graphs have a somewhat consistent distribution of which node is the most central.

In a similar manner, the variances of the centrality measures shown in Figure 5.6 are larger for the order 1 bivector subinvariants Inv_1^2, extending the analysis of Figure 5.5 to the consideration of all the nodes' centrality scores (not just the most central one). The overlap in variance values for the higher orders indicates that their respective graphs have similar connectivity properties, although there are potential bounds which could separate some order 2 invariants (Inv_2), particularly for the A_8 and E_8 algebras.

Overall, the analysis of the bivector subinvariant graphs' eigenvector centralities indicates that the order 1 graphs are distinctly different to those coming from higher order invariants. The Inv_1 invariants tend to be more consistently structured (with the same basis elements creating the most central node) and more skewed in centrality, with central nodes more dominantly central. Generally, going to higher-order invariants increases the connectivity, at least for A_8 and E_8, with some reasonably well-separated clustering of the orders. For D_8, the Inv_3 invariants seem the most connected; this is likely due to the unique geometry of D_8, manifested e.g. also by the non-trivially zero scalar and pseudoscalar terms in Table 5.3.

5.5 Machine Learning

5.5.1 Binary Classification of Invariants: real vs fake data

It is expected that geometric invariants can be constructed, via some formula, from Coxeter elements that carry information about the root system. So one might hope to find specific features of invariants that depend on the root system/Lie algebra. In this section, we train

a neural network classifier to check whether there are such features which the a network is able to learn. High performance of the model would indicate a presence of such features.

The dataset used consists of 3 subsets, one for each of the algebras $A_8/E_8/D_8$, with 40320 entries and 2304 components in each subset. To further enlarge the amount of data, we generated a 'fake' dataset for each of the $A_8/E_8/D_8$ algebras. In the first iteration, 'fake' datasets were generated by constructing empirical distributions for each of the 2304 components from all available invariants for the corresponding algebra and then sampling from this distribution to create 40,000 unique elements in new datasets. One can notice that in the 'real' data, the number of zeros in each entry is constant and specific for different algebras[20]. We implemented this feature in the fake datasets we used by eliminating fake data entries which do not satisfy this condition.

We started with one of the simplest supervised learning approaches, binary classification by a dense neural network, with the idea to train 3 NNs to distinguish invariants for one of the algebras from other algebras and 'fake' invariants. Overall, we have 6 datasets of geometric invariant components for A_8, D_8, and E_8 algebras, as well as 'fake' datasets for each. From these, three final training/test datasets were constructed, one for each algebra, where we labelled 'real' A_8 or D_8 or E_8 algebra invariant components with 1, two remaining 'real' algebras invariant components with 0, and 3 'fake' datasets labelled as 0 as well. This is what we will imply when we say that we create training/test datasets to distinguish one of the A_8, D_8, and E_8 invariants from other 'real' invariants and the 'fake' ones.

One should notice, however, that training NNs to distinguish one of the A_8,D_8 or E_8 invariants from others and fake invariants using all datasets is invalid. For all three of them, the prediction on test datasets would be 100% accurate because, as described in section 5.3.2, there are only 128 unique invariants for each of the ADE datasets, leading to a large repetition of them (although with unequal frequencies). Hence, there is a high chance that in a randomly chosen training subset, we would find all 128 unique invariants. This would make the test stage biased, as it likely contains repetitions of the training data. Effectively, these NNs are learning to reproduce this dataset of invariants perfectly, but would not be able to generalise beyond it.

A more meaningful problem is to remove degeneracy in datasets for 'real' A_8,D_8 or E_8 invariants, leaving just 128 elements in each. Then, we can mix original and fake invariants, which ensures that we have some real invariants in the training set and others in the test set only. The proportion of data in training and test datasets was again set to 80% and 20%. However, this makes the whole dataset skewed as there are around 40000 fake invariants and only 3×128 real invariants. We kept data unbalanced in the training set, but in the test set, to make it easier to interpret results, we cut the number of fake invariants to be the same as the number of real invariants, meaning we had $3 \times \{128/k$ real and $128/k$ fake$\}$ invariants.

In this setup, we tried NN with architectures varying from 1 hidden layer with 32 units to 2 hidden layers with 256 units in each. In all of them, the ReLU activation function was used. During training, we used the Adam optimiser (with a learning rate of 0.001) to optimise the log-loss function. The train and test datasets represented an 80% / 20% split of the total data. The best performance was demonstrated by 1 hidden layer 64, 128 and 256 units NNs with accuracies in the range of 0.90-0.92. From this, one might speculate that there should be some relatively simple invariant quantity (similar to the genus of a surface) that was learned by the NNs to distinguish real invariants coming from different algebras and fake invariants.

[20] 1942 for E_8, 2083 for D_8 and 1805 for A_8.

5.5.2 Regressing Invariants from Permutations

As introduced in Section 5.2, Clifford algebras and the simplicial derivatives/characteristic multivectors provide us with a systematic way of computing the geometric invariants (SOCM) occurring, e.g. in the Cayley-Hamilton theorem. In particular, this depends on the permutation order of the simple reflections in a Coxeter element. It is, therefore, expected that there is an analytical formula that directly predicts the corresponding geometric invariant from the order of the permutation of the simple reflections. Since making conjectures or derivations from scratch is challenging in this task, in the spirit of experimental mathematics, we hope that the use of supervised learning algorithms, which train on labelled datasets to make predictions, can shed some light on this expected relation.

In this paper, we used dense NNs coded in `python` where the input is the (one-hot encoded [177]) permutation and the output is the coefficients of the invariants. For clearer results, we first partitioned each dataset for A_8, D_8, E_8 into 9 subdatasets for each of the 9 invariant orders, and then looked at the coefficients of both the full invariant and each subinvariant for each order subdataset, i.e., the scalar, bivector, quadrivector, sextivector, and pseudoscalar for each invariant order Inv_0-Inv_8 (SOCM). The NN model includes four dense layers of 256 units with ReLU activation function. In training, we used the Adam optimiser (with learning rate 0.001) to minimise a mean-squared error loss. 5-fold cross-validation was also used, with the test data subset being 20% of the full dataset. To calculate accuracy, predictions were rounded to the nearest integer, and a prediction was considered correct if all of its coefficients were predicted correctly after rounding.

Our ML results are summarised in Tables 5.7 to 5.9 for the A_8, D_8, E_8 data in the simple root basis. The results show near-perfect prediction of all invariants and subinvariants across all algebras. The trivial scalar invariants are unsurprisingly all learnt perfectly, but in many other cases, there is perfect learning also. The lowest performance occurs for the sextivectors, where there is less data to learn from. These results indicate that the NNs are capable of well approximating the complicated algorithm carried out to compute these invariants, as well as accommodating for the basis permutations.

5.5.3 Gradient Saliency Analysis

To better interpret the decision-making of our NN models – which are black-box models – we also performed gradient saliency analysis [178]. In general, the magnitude of the elements in a weight vector in the model tells us the importance of the corresponding input element for a particular output. We can extend this to consider the sensitivity of the entire NN function to the inputs by computing the gradient of a given output with respect to the input via backpropagation. The magnitude of the gradient indicates how sensitive the output is to a change in the input variable. The results for the average gradient magnitudes across the test sets (and 100 cross-validation runs) are shown in Figure 5.7 for the multi-classification investigation equivalent to Section 5.5.1 but without considering the fake data (performance was equivalently perfect), and Tables A.1, A.2, A.3 in appendix A for the subinvariant regression in Section 5.5.2.

The Figure 5.7 gradient saliency barcode maps show the relative importance of the subinvariant inputs for algebra classification; hence the bivector $\binom{8}{2} = 28$ ($= \binom{8}{6}$ for the sextivector also) coefficients are represented by 28 vertical lines, whilst the quadrivector $\binom{8}{4} = 70$ coeffi-

	$\text{Acc}(\text{Inv}_i)$	$\text{Acc}(\text{Inv}_i^0)$	$\text{Acc}(\text{Inv}_i^2)$	$\text{Acc}(\text{Inv}_i^4)$	$\text{Acc}(\text{Inv}_i^6)$	$\text{Acc}(\text{Inv}_i^8)$
Inv_0	1.0000	1.0000				
Inv_1	1.0000	1.0000	0.9996			
Inv_2	0.9996	1.0000	0.9999	0.9999		
Inv_3	0.9980	1.0000	0.9973	0.9480	0.9999	
Inv_4	0.9958	1.0000	0.9999	0.9999	0.9117	0.9596
Inv_5	0.9986	1.0000	0.9995	0.9999	1.0000	
Inv_6	1.0000	1.0000	0.9948	0.9999		
Inv_7	0.9999	1.0000	1.0000			
Inv_8	1.0000	1.0000				

Table 5.7: Summary of the final test accuracy (Acc) for the full invariants and each subinvariant of the 9 invariants for A_8 simple root data.

	$\text{Acc}(\text{Inv}_i)$	$\text{Acc}(\text{Inv}_i^0)$	$\text{Acc}(\text{Inv}_i^2)$	$\text{Acc}(\text{Inv}_i^4)$	$\text{Acc}(\text{Inv}_i^6)$	$\text{Acc}(\text{Inv}_i^8)$
Inv_0	1.0000	1.0000				
Inv_1	1.0000	1.0000	0.9955			
Inv_2	1.0000	1.0000	0.9912	0.9999		
Inv_3	0.9993	1.0000	0.9995	0.9999	1.0000	
Inv_4	0.9995	1.0000	0.9988	0.9891	0.9998	1.0000
Inv_5	0.9995	1.0000	0.9986	1.0000	1.0000	
Inv_6	1.0000	1.0000	1.0000	1.0000		
Inv_7	1.0000	1.0000	0.9999			
Inv_8	1.0000	1.0000				

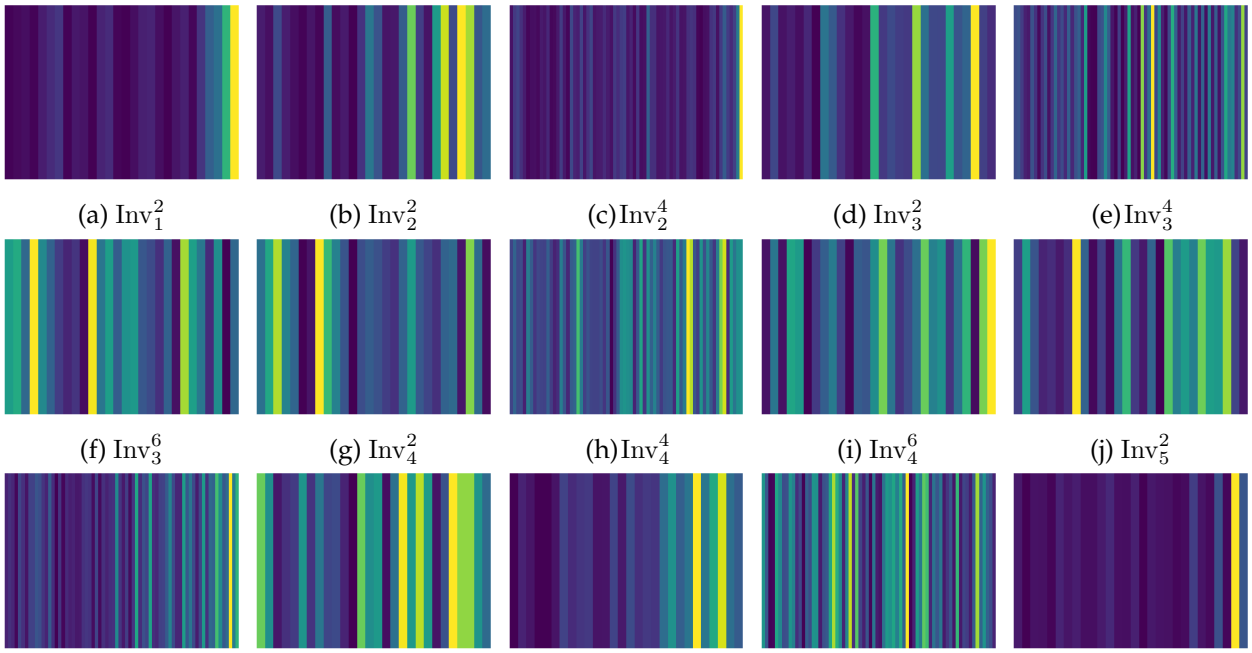
Table 5.8: Summary of the final test accuracy (Acc) for the full invariants and each subinvariant of the 9 invariants for D_8 simple root data.

	$\text{Acc}(\text{Inv}_i)$	$\text{Acc}(\text{Inv}_i^0)$	$\text{Acc}(\text{Inv}_i^2)$	$\text{Acc}(\text{Inv}_i^4)$	$\text{Acc}(\text{Inv}_i^6)$	$\text{Acc}(\text{Inv}_i^8)$
Inv_0	1.0000	1.0000				
Inv_1	1.0000	1.0000	1.0000			
Inv_2	0.9999	1.0000	1.0000	1.0000		
Inv_3	0.9994	1.0000	0.9906	0.9999	0.9891	
Inv_4	0.9969	1.0000	1.0000	0.9963	0.9793	0.9223
Inv_5	0.9994	1.0000	0.9996	0.9999	0.9990	
Inv_6	1.0000	1.0000	1.0000	1.0000		
Inv_7	1.0000	1.0000	0.9998			
Inv_8	1.0000	1.0000				

Table 5.9: Summary of the final test accuracy (Acc) for the full invariants and each subinvariant of the 9 invariants for E_8 simple root data.

cients are represented by 70 vertical lines. Note the trivial scalar as well as the pseudoscalar subinvariants are omitted, and the remaining plots approximately satisfy the observed mirror symmetry between invariant orders (i.e. $\text{Inv}_1^2 \sim \text{Inv}_7^2$, $\text{Inv}_3^4 \sim \text{Inv}_5^4$, etc.).

For Inv_1^2 and Inv_7^2, the NNs rely almost exclusively on the final components of the coefficient vector, with a similar skewed behaviour for Inv_2^2 and Inv_6^2 towards the final coefficients implying the span of these final coefficient values is most disparate between the algebras and thus can be used for classification. The $\text{Inv}_3^2, \text{Inv}_4^2, \text{Inv}_5^2$ and $\text{Inv}_3^6, \text{Inv}_4^6, \text{Inv}_5^6$ barcodes have less



(a) Inv_1^2 (b) Inv_2^2 (c) Inv_2^4 (d) Inv_3^2 (e) Inv_3^4

(f) Inv_3^6 (g) Inv_4^2 (h) Inv_4^4 (i) Inv_4^6 (j) Inv_5^2

(k) Inv_5^4 (l) Inv_5^6 (m) Inv_6^2 (n) Inv_6^4 (o) Inv_7^2

Figure 5.7: Gradient saliency maps (barcodes) for ternary classification NN model. The NN function takes as input the coefficients of the subinvariant and outputs one-hot encoded: A_8, D_8, or E_8. The saliency hence, represents the relative importance of each input coefficient (i.e. combination of simple roots) to determining the classification output. Lighter colours indicate larger gradients and greater importance.

discernible patterns, but in each case do appear to prioritise specific coefficient entries, emphasising that the entries in the subinvariant coefficients vectors are not equally important as one may naively assume. The Inv_2 (and Inv_5) quadrivector maintains the Inv_2 bivector bias towards reliance on the final coefficient entries, however, the other quadrivector barcodes have a smoother spread of importance and a more complicated (NN-approximate) function differentiable structure.

These results provide insight into the relative importance of each coefficient to uniquely identifying the respective algebra, and continues to guide our analytical study of these subinvariants in our companion paper by indicating which subinvariants have the most clear coefficient dependence (Inv_1 and Inv_7 bivector, and Inv_2 quadrivector) for us to focus on, in revealing the underlying behaviour of these invariants.

In Tables A.1, A.2, A.3, the saliency barcodes probe the NN learning of the subinvariants explicitly from the Coxeter element root permutation (i.e. the order of the 8 roots in the Coxeter element) – hence having 8 bars in each barcode. Satisfyingly, the mirror symmetry between opposite order invariants is again approximately obeyed for each of the 3 algebras considered.

The trivial order 0 and 8 full invariants (with only scalar part such that they start with a single 1 followed by 255 0s), as well as all the scalar invariants, are, as expected, perfectly learned and have random saliency behaviour since the learning of a constant function is trivial and independent of the inputs. One may expect perfectly equal barcodes, but the rounding of the outputs allows the final neuron output to vary, so the stochastic search of the optimiser has a large range of functions which give perfect results, that it will be random walking in the function space throughout the training. This makes the final function fairly

arbitrary in this class of suitable functions, and hence the barcodes are random, providing some measure of the level of noise in the learning.

Conversely, the pseudoscalar for the order 4 invariants has different saliency properties between the algebras, with focus on different parts of the permutation vector; where the A_8, D_8, E_8 order 4 invariant pseudoscalars, respectively, are computed primarily from the end, middle, and start of the permutation vector. Note that the pseudoscalar component is 0 for D_8 and therefore that barcode is essentially noise, alike the scalar barcodes.

The remaining barcodes all have similar behaviour across the algebras. This is for the bivector, quadrivector and sextivector subinvariants, which dominate the full invariant coefficient vector and thus unsurprisingly lead to similar behaviour for the full invariant. This behaviour puts focus on both ends of the permutation barcodes, indicating that the information about which roots are positioned at the ends of the Coxeter element is the most important for determining the structure of each of these respective subinvariants (as well as the full invariant).

Intuition one may extract from this is that at the ends of the permutation vectors the roots have only one direction they can be permuted, and thus the root they are adjacent to is paramount to determining whether that root can commute further into the permutation vector without changing the Coxeter element; as dictated by whether the roots are connected in the Dynkin diagram. How the 40320 permutation orders split into the 128 Coxeter elements for each algebra may then be well correlated with the end roots of the permutation, providing the NNs with important primary information in directing the first steps of their functional algorithm for information flow through their architecture, leading to correct calculations of the invariants. Further analytic analysis with a focus on permutation partitions grouped by their end roots should help to reveal the dominant factors for the distributions of these invariants.

5.6 Unsupervised: PCA

Principal component analysis (PCA) is a widely used ML technique for dimensionality reduction and exploratory data analysis [179]. In short, one computes the principal components, which are linear combinations of the initial variables, and performs a change of basis on the data. One then projects the data onto only the first few principal components to obtain a lower-dimensional data representation.

The first principal component is the normalised linear combination of initial variables that explains the largest variance in the data. The second principal component is uncorrelated with (i.e. perpendicular to) the first principal component and explains the next highest variance, and so on. The computation of the principal components can be broken down into the following steps:

1. The covariance matrix is computed. This is a symmetric matrix whose entries are the covariances associated with all possible pairs of variables.

2. The next step is to compute the eigenvectors and eigenvalues of the covariance matrix. These eigenvectors are the principal components, and the eigenvalues describe the amount of variance carried in each principal component.

Note that often in general data science, an extra step 0 is included whereby one standardises the variables so that each contributes equally, which is especially important when different

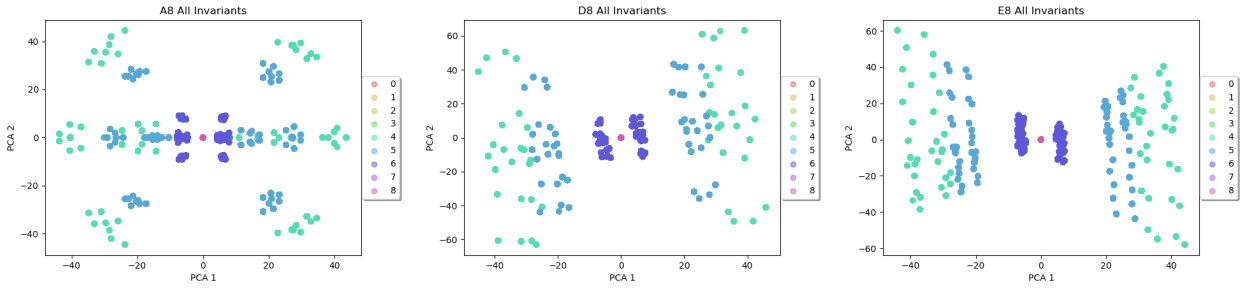
input features have different units of measurement. However, since all variables are unitless and take values in a similar range, we don't standardise here. By ranking the eigenvectors in order of their eigenvalues, highest to lowest, one gets the principal components in order of significance. Finally, to transform the data into the new representation, one performs a change of basis on the standardised data using the principal components, followed by projection.

Using the Coxeter element invariant data in the simple root basis as described in Section 5.3.1, we perform PCA on the 9 order invariants both individually and combined, for A_8, D_8 and E_8. With this, we plot the data in the first two principal components. The results for the 9 individual invariants of A_8, D_8 and E_8 are shown in Figures B.1, B.2 and B.3 respectively and the PCA results on the combined invariant data are shown in Figure 5.8. We can see clearly from Figures B.1-B.3, that Inv_0 and Inv_8 just give the trivial invariant, and furthermore Inv_1 matches Inv_7, Inv_2 matches Inv_6 and Inv_3 matches Inv_5. This aligns with the connection we made earlier in Section 5.2. Comparing the individual plots for A_8, D_8 and E_8 we see that the plots for D_8 and E_8 roughly align, while the plots for A_8 are different. In D_8, E_8, for example, the plots for $\mathrm{Inv}_3, \mathrm{Inv}_4$ and Inv_5 share a gap in the middle and all the data points are roughly scattered on either side. On the other hand, the Inv_4 plot for A_8 presents a circular pattern around the centre, and whilst separated down the middle, the data points in the Inv_3 and Inv_5 plots of A_8 are tightly clustered. For the A_8 plots, there appears to be a 2-fold reflection symmetry in the $\mathrm{Inv}_1 - \mathrm{Inv}_7$ plots, with the Inv_1 and Inv_7 plots having a second orthogonal 2-fold reflection symmetry. For D_8 and E_8, there is instead an approximate 2-fold rotational symmetry.

Figure 5.8 shows the 2-dimensional PCA projections when fitting the principal components for all the invariant orders considered together. The orders form distinct clusters, and the two 2-fold reflection symmetries of A_8 and 2-fold rotation symmetries of D_8 and E_8 are approximately preserved. Figure 5.10a also shows the elbow plot of the PCA ratios against the number of principal components for PCA performed on the combined dataset of all orders. The explained variance ratio is a measure of the proportion of the total variance in the original dataset that is explained by each principal component. This is equal to the ratio of its eigenvalue to the sum of the eigenvalues of all the principal components. The x-axis in Figure 5.10a is the order number of the first principal components, and the y-axis is the log of the explained variance ratio. For A_8 and E_8, we see a characteristic sharp drop in the ratio at around the 100th principal component and for D_8 at around the 75th principal component. This means that in all cases the 256-dimensional vectors describing the invariants (of which, of course, only 128 are not trivially zero) in fact can be reduced whilst preserving the majority of information. Furthermore, it is interesting that D_8 requires fewer principal components than the other two, which may be related to the vanishing pseudoscalar as well as some scalar parts in this case, and a quarter as many unique sextivector parts also, as shown in Table 5.3.

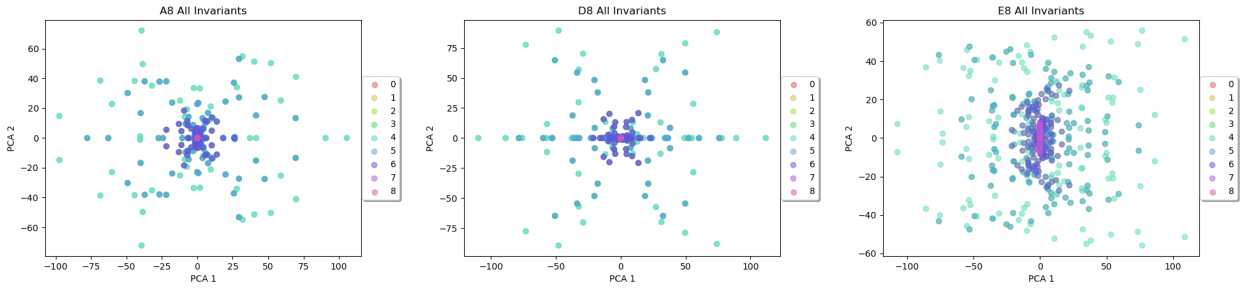
As we saw in Section 5.3.2, the 40320 permutations give rise to only 128 unique Coxeter elements for A_8, D_8 and E_8 and the frequency of these 128 vary greatly. The frequency of invariants will have a significant effect on the PCA results, and therefore, for comparison, we repeat the PCA but on the reduced dataset of 128 invariants. Again, we perform the analysis on the 9 orders of invariant individually and combined. The individual PCA plots are shown in Figures B.4-B.6 and the combined plots are given in Figure 5.9. Figure 5.10b also shows the elbow plot of the explained variance ratios for the combined PCA.

The discussed reflection and rotation symmetries of Figures B.1-B.3 become clearer to see in Figures B.4-B.6, as presumably uneven multiplicities no longer weight the projections



(a) A_8 (b) D_8 (c) E_8

Figure 5.8: PCA plots of all 9 order invariants (SOCM) simultaneously for A_8, D_8 and E_8. Note that labels 5-8 don't appear, as these invariants are mirror symmetric. This plot and analysis are a good check of this fact.



(a) A_8 (b) D_8 (c) E_8

Figure 5.9: PCA plots of reduced datasets (with duplicates deleted) of all 9 order invariants (SOCM) simultaneously for A_8, D_8 and E_8.

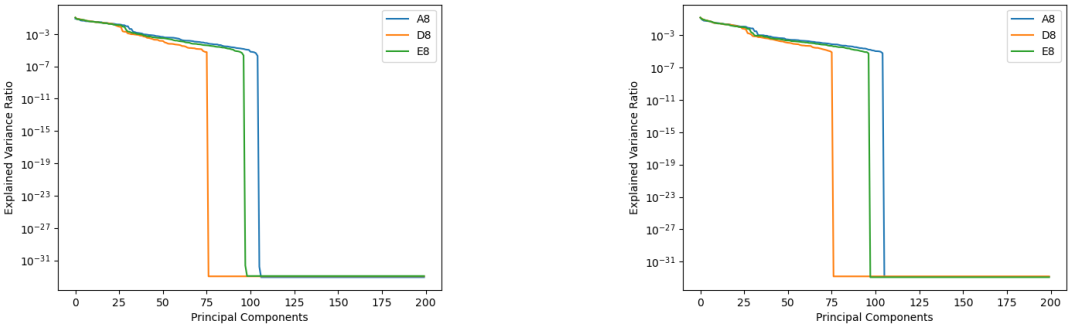
asymmetrically. Also for the A_8 and D_8 algebras, the Inv_4 invariant projections become very nearly identical to the respective Inv_3 / Inv_5 projections, emphasising a negligible impact of the inclusion of the pseudoscalar between these invariant orders on the first two principal components (which turns out to be 0 for D_8 but not A_8, see Tables 5.2 and 5.3). This, surprisingly, does not happen for E_8, indicating the pseudoscalar contribution to the principal components is more significant here.

Whereas we saw distinct clustering of the different orders in the combined PCA plots in Figure 5.8, we do not see this in the equivalent plots in Figure 5.9 from PCA on the combined datasets with duplicates deleted. Comparing the combined plots to the unique plots for the reduced datasets, we see the patterns from the unique order plots in Figures B.4-B.6 emerging in the combined plots in 5.9. It appears as if all the unique plots have simply been overlaid on top of one another. This suggests that principal components for all the 9 orders are the same and also match the principal components from the combined PCA.

The elbow plot in Figure 5.10b matches almost identically that in Figure 5.10a, and the same conclusions hold.

5.7 Summary and Outlook

The experimental mathematics paradigm explored in this work combines an HPC computational algebra approach, generating a significant amount of algebraic data with a data science analysis of the resulting dataset. Performing exhaustive calculations opens up a new



(a) (b)

Figure 5.10: Elbow plot of explained variance ratio against principal component number for (a) full A_8, D_8 and E_8 datasets and (b) the reduced A_8, D_8 and E_8 datasets with duplicates removed.

angle to conjecture formulation and theorem proving, and sheds light on the new geometric invariants for the important class of examples of Coxeter transformations. This detailed example can be used as a foundation to explore the behaviour of invariants in other dimensions or with different types of linear transformations. It was expected that there would be a large degeneracy in the mapping between input permutations, resulting in a smaller number of unique elements in the dataset. This assumption was verified through the application of data analysis techniques. Moreover, many unexpected features were discovered, such as the equality of the number of unique invariants among all three algebras. The relatively small size of the unique output invariants dataset in this example made it perfectly suited to ML tasks, which perform very impressively. One can, of course, go to an arbitrary dimension to get larger data sets for A_n and D_n, but at the expense of missing out on the E-type. The patterns observed in the reduced set of Coxeter elements, invariant bivectors, and some other approaches discussed in [19] have certainly pointed in the direction of analytical results that generalise these computational observations here, some of which we have mentioned above and some to be investigated in further work.

Chapter 6

Machine Learning Discovery of New Champion Codes

In this chapter, we present a novel method for discovering champion generalised toric codes (linear codes achieving or exceeding the best known minimum Hamming distance)[1]. The parameter search space is combinatorially large, making the brute-force search computationally intractable. To overcome this challenge, we train a transformer to predict the minimum Hamming distance of a class of linear codes and pair it with a genetic algorithm to search the parameter space. This approach successfully identified several new champion codes.

6.1 Motivation and summary

Coding theory is at the heart of modern communication. Error-correcting codes – used to detect and correct errors in data transmitted over Wi-Fi or LAN networks [180], under transatlantic waters [181], and across deep space [182], [183], [184] – are a key part of coding theory. Thus, the search for more efficient error-correcting codes is of great importance.

The minimum Hamming distance is one of the crucial parameters that determines the error-correcting capabilities of the code. Codes with the largest minimum distance among those with the same other parameters are called *champion codes*. Determining this for general toric codes is a computationally challenging task as the code parameters grow [185]. In particular, it was demonstrated that it is an NP-hard problem for binary codes [186].

A type of codes for which powerful theoretical predictions are possible are the toric codes defined by Hansen [187]. It is a class of error-correcting linear codes constructed from monomials on toric varieties. These codes offer a rich algebraic structure and are connected to algebraic geometry and commutative algebra, allowing us to set upper theoretical bounds on the minimum distance [188]. Several studies have explored linear codes in general [189] and specific families of toric over $\mathbb{F}_q$ for various q [190], [191]. New champion codes have been discovered [192], [193]. In particular, in [193] the search for new champions was performed systematically by a brute-force calculation of the minimum distance of all possible generalised toric codes over fields from $\mathbb{F}_3$ to $\mathbb{F}_7$ (and partially $\mathbb{F}_8$) [193]. For relatively small codes over fields up to $\mathbb{F}_7$, this was possible due to the current availability of computational

[1]The work presented in this chapter is based on the collaborative study [30]. The author of the thesis is the main contributor to the design, implementation, and experiments with the transformer model for minimum distance prediction, the development of the base of the core genetic algorithm code, and the data processing pipeline.

resources. However, the vast number of possible generalised toric codes over $\mathbb{F}_q$ makes a systematic search infeasible for larger values of q.

In this study, we use ML techniques to approximate the minimum Hamming distance of generalised toric codes over $\mathbb{F}_8$ and provide a methodology to discover new champion linear codes. We follow the tradition of applying data science and machine learning techniques to purely mathematical data. This tradition has origins in the study of string theory and algebraic geometry [12], and has developed to other areas of mathematics; for example [125], [194], [195], [196], [197], [198], [199]. Although machine learning does not guarantee optimal solutions to all problems, our investigation shows a successful application of machine learning to an NP-hard problem [200] and a new methodology for discovering champion codes.

We begin by creating a sequence model which performs classification using the generator or parity-check matrix of the generalised toric code as an input and outputs the Hamming minimum distance of the code. We trained these models on datasets of codes over $\mathbb{F}_7$ and measured and compared each model's performance. The best models were then paired with a genetic algorithm to achieve the largest minimum distance of a generalised toric code over $\mathbb{F}_7$ for each dimension k as in [193]. This provided a proof of concept for our exploration of $\mathbb{F}_8$, where we discovered possible champion linear codes and verified several of them. In particular, we discovered new champion linear codes over $\mathbb{F}_8$, which are listed in Table C.4 (Appendix C).

6.2 A Primer on Codes

In this section, we provide some rudiments of coding theory that will be used in this chapter.

The most common (error-correcting) codes are *linear codes* defined over a subspace C of the vector space $\mathbb{F}_q^n$. Mathematically, a linear code is a k-dimensional linear subspace C of the vector space $\mathbb{F}_q^n$ over a finite field with q elements. The vectors in C are *codewords* of block length n, and the *size* is the number of possible codewords, which is equal to q^k. The *weight* of a codeword is the number of its non-zero elements (vector components), and the *distance* between two codewords is the Hamming distance between them, that is, the number of elements in which they differ. The distance d of a linear code is the minimum weight of its non-zero codewords or, equivalently, the minimum distance between distinct codewords. Altogether, the three main parameters of a linear code C are denoted as $[n, k, d]_q$. A general reference for coding theory is [201], and one can find theoretical upper and lower bounds on the highest minimum distance for a given n and k with the https://codetables.de/ [202].

We can characterise the code $C_V(\mathbb{F}_q)$ with two matrices, the generator matrix $G_V(\mathbb{F}_q)$ and its parity-check matrix $P_V(\mathbb{F}_q)$, defined as follows:

Definition 1. *Suppose C is a linear code with block length n and dimension k.*

- *The generator matrix G of C is a $k \times n$ matrix whose rows form a basis for the code, such that any codeword c is in the row-span and thus can be expressed as $c = m \cdot G$ for some length-k vector m.*
- *The canonical form for the generator matrix G is $G = [I_k|H]$, where I_k is the $k \times k$ identity matrix and H is a $k \times (n-k)$ matrix.*
- *The dual generator matrix $G^\perp$ of G is any matrix whose rows form a basis for the dual code $C^\perp$, where*

$$C^\perp = \{y \in \mathbb{F}_q^n : y \cdot c = 0 \text{ for all } c \in C\}.$$

The parity-check matrix P of G is then the $(n-k) \times n$ matrix satisfying $GP^T = 0$. Because P satisfies $GP^T = 0$ and its rows generate $C^\perp$, P is typically taken as the dual generator matrix.

Likewise, given the canonical form of the generator matrix, the canonical form of the parity-check matrix P is $[-H^T|I_{n-k}]$. Thus, there is no ambiguity when considering the equivalent generator and parity check matrices of a linear code C. From now on, we will assume the canonical form when discussing the generator and parity check matrices. Indeed, when the context is clear, we may write the linear code, its generator matrix, its parity-check matrix, and its minimum Hamming distance by C_V, G_V, P_V, and d_V respectively, with the understanding of the underlying field $\mathbb{F}_q$. We say that V is the set of vertices of the generalised toric code C_V.

The minimum distance d_V is a measure of the potential of the code for error correction. It is equal to the minimum of a Hamming distance between any two codewords, while the Hamming distance is defined as the number of unequal elements in two vectors. An important task is to seek codes with a minimum Hamming distance d that is as large as possible for a given n and k; this is sometimes referred to as "the main coding theory problem" [201]. In particular, we refer to any code with a larger minimum distance than the currently known lower bounds as a *champion code*, as in [193].

To calculate this minimum distance, one can use an algorithm known as the Brouwer-Zimmermann (BZ) algorithm [203], which is also described in [189]. Despite several improvements by [189], the BZ algorithm is computationally expensive; it has polynomial in k and exponential in q complexity [204].

Toric codes are a class of codes introduced by J. Hansen in [187], which serves as a valuable source for constructing the linear codes. As we will see in Section 6.3, toric codes (we will refer to them as non-generalised to avoid any ambiguity) and generalised toric codes are defined in the same terms, the only difference being that the non-generalised toric codes are built from a set of vertices belonging to a certain lattice polytope, while generalised toric codes can be built from any set of vertices on a lattice[2]. Therefore, the set of non-generalised toric codes is contained in the set of generalised toric codes. As we will see later, working with generalised toric codes is easier in our case and, at the same time, allows us to infer knowledge about toric codes.

It is important to emphasise that the term *toric codes* is also used in quantum computing by Kitaev [206]. However, this is entirely different. Moreover, while Hansen defined toric codes using toric varieties and polytopes, they can also be described within the framework of evaluation codes. We need not delve into the details of defining toric codes here because we will now introduce the generalised version by J. Little [192], which has proven useful in identifying champion codes [191].

6.3 Generalised Toric Codes

In this study, we focus on generalised toric codes. This comes at the cost of losing certain geometric results specific to toric codes, such as the Minkowski length [190]. We will address this trade-off in detail later; for now, we begin by introducing the generalised toric codes.

Consider the planar lattice grid $[0, q-2]^2$ over the field $\mathbb{F}_q$ for a prime power q [3] and

[2] See [205] for more details.

[3] It is possible to work with non-prime q. In this case, there is, technically, no primitive element ξ which is an integer. However, one can introduce an object α and demand that raising it to consecutive powers generates all the non-zero elements of $\mathbb{F}_q$. We will see it used later with the $\mathbb{F}_8$ field.

primitive element $\xi \in \mathbb{F}_q$ (i.e. the primitive $(q-1)$-th root of unity in the multiplicative subgroup $\mathbb{F}_q^*$). For all $0 \leq i, j \leq q-2$, we define $P_{i,j} = (\xi^i, \xi^j) \in (\mathbb{F}_q^*)^2$, and for each $u = (u_1, u_2) \in [0, q-2]^2$, we define the mapping

$$\begin{aligned} e(u) : \mathbb{F}_q^* \times \mathbb{F}_q^* &\longrightarrow \mathbb{F}_q \\ P_{i,j} = (\xi^i, \xi^j) &\longrightarrow e(u)(P_{i,j}) = (\xi^i)^{u_1}(\xi^j)^{u_2}. \end{aligned} \tag{6.3.1}$$

Then, we have

Definition 2. *Let $V = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ be a set of vertices from $[0, q-2]^2$. We define the corresponding generalised toric code $C_V(\mathbb{F}_q)$ over the field $\mathbb{F}_q$ associated to V as the linear code of block length $n = (q-1)^2$ and dimension $k = \dim C_V(\mathbb{F}_q)$ spanned by the vectors in*

$$\left\{ (e(u)(P_{i,j}))_{0 \leq i,j \leq q-2} = (\xi^{iu_1 + ju_2})_{0 \leq i,j \leq q-2} \ : \ u \in V \right\}. \tag{6.3.2}$$

Note that the dimension k of our code may differ from the number of vertices m. As with any linear code, the minimum Hamming distance $d = d_V(\mathbb{F}_q)$ of $C_V(\mathbb{F}_q)$ is defined as the minimum distance between any two codewords of $C_V(\mathbb{F}_q)$. We can then describe $C_V(\mathbb{F}_q)$ as an $[n, k, d]_q$-linear code. Sometimes, we also use m_V to reference the number of vertices m for the code C_V.

Example: It is expedient to give an example here. If $q = 3$, then the primitive element is $\xi = 2$ (since in $\mathbb{F}_3 \simeq \{0, 1, 2\}$, the multiplicative group is $\mathbb{F}_3^* \simeq \{1, 2\}$, and both are powers of 2 modulo 3), and we have a $[0, 1]^2$ lattice grid over $\mathbb{F}_3$. Notice that:

$$\begin{aligned} e(0,0)(P_{i,j}) &= 1 \\ e(0,1)(P_{i,j}) &= 2^j \\ e(1,0)(P_{i,j}) &= 2^i \\ e(1,1)(P_{i,j}) &= 2^{i+j} \ . \end{aligned} \tag{6.3.3}$$

Consider $V = \{(0,0), (0,1), (1,0), (1,1)\}$. Then, the generalised code C_V is spanned by the following vectors:

$$(1,1,1,1), (1,1,2,2), (1,2,1,2), (1,2,2,1)$$

Then, the generator matrix G_V and parity check matrix P_V of C_V are:

$$G_V = \begin{pmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \quad P_V = \begin{pmatrix} 1 & 2 & 2 & 1 \end{pmatrix}$$

Note that the dimension of this code is not equal to the number of vertices. By computation with a brute-force algorithm (i.e. the Brouwer-Zimmermann algorithm), we calculate that the minimum Hamming distance d_V is 2. Hence, C_V is a $[4, 3, 2]$-code.

Non-generalised toric codes are constructed by ensuring that the set of vertices forms a convex polytope over $[0, q-2]^2$.

If we consider only non-generalised toric codes, we gain access to geometric methods by connecting them to toric varieties, as described in [207]. In particular, many have used the theory of toric varieties, their cohomology, and intersection theory to obtain upper and lower

bounds for the minimum Hamming distance of associated toric codes of certain shapes, as seen in [207], [190], and [188]. Although there are more tools to access by restricting to non-generalised toric codes, we decided to consider generalised toric codes instead because generating datasets for generalised toric codes is computationally cheaper than for non-generalised toric codes.

6.4 Machine Learning the Minimum Hamming Distance

In this section, we investigate whether ML can predict the minimum Hamming distance d_V of a generalised toric code $C_V(\mathbb{F}_q)$ from its generator matrix. We initially consider the problem over $\mathbb{F}_7$, as all the champion codes are known there, to develop a methodology for use in the $\mathbb{F}_8$ case.

Existing research on ML applications to toric codes primarily focuses on the study of existing quantum error-correcting codes within the framework of Kitaev's toric codes [208], [209], [210]. At the same time, there is no literature on the application of ML to Toric codes defined by Hansen or the search for new champion codes, which is the aim of our work.

A generalised toric code C, represented by a generator G or parity-check matrix P, can be viewed as a sequence of rows of length $(q-1)^2$, where the number of rows is the dimension k of the code. Thus, the problem can be formulated as the prediction of the Hamming minimum distance of that code, which can take integer values between 0 and $(q-1)^2$, from the input sequence S of length k (dimension of the code) of code rows, with each being a vector of length $(q-1)^2$.

This type of ML task is called sequence modelling. There are several ML architectures suitable for these types of tasks: Convolutional Neural Networks, Recurrent Neural Networks, and Transformers. The important property of the latter two is that they are naturally suited to work with the inputs on any dimension[4]

Furthermore, the problem of computing the Hamming minimum distance of a code is an NP-hard problem, which suggests using the most powerful available approaches currently in use in the ML community, that is, transformers [147], which are the state-of-the-art approach for sequence modelling tasks used in Natural Language Processing (NLP) tasks. Transformers were most famously used in the ChatGPT chatbot [148] by OpenAI, but also in tasks such as sentiment analysis [149]. The latter is a conceptually similar task to what we consider here: given a sequence of elements (sequence of text tokens or rows of the code), a class needs to be predicted (sentiment of a given text or a Hamming minimum distance). In the following, we sometimes refer to codes with a certain minimum distance as class.

Some experiments were conducted with Recurrent Neural Networks (RNNs); see Appendix D.1.3 for details. However, we ultimately decided to build our model based on the Transformer, as it was the architecture with the most potential. Computer code scripts and data can be found on GitHub[5].

6.4.1 Toric Codes Datasets

We aim to train an ML model to predict the minimum distance given a code, therefore, our dataset consists of tuples of code generator matrices G, generator matrices of dual codes P (parity-check matrix), and a corresponding minimum distance d.

[4]Though the transformers have a limit on the maximum length of an input.

[5]`https://github.com/QTVLe/MachineLearningGeneralisedToricCodes`

The rows of our generator matrices are calculated from the coordinates of the set vertices, as defined in Definition 2. However, the resulting matrix is not in canonical form; its row order may vary with the order of the vertices, and two non-equivalent sets of vertices may produce the same code. To resolve this, after generating the initial generator matrix, we bring them to the canonical form using the functionality of the SAGEMATH [211] or the MAGMA [212] systems. Thus, we can generate toric codes corresponding to a certain set of vertices V.

Generating a balanced dataset in which all minimum distances are equally represented is a nontrivial task. This is because, although the minimum distance d of code C generally decreases as the dimension of the code k increases, there is a considerable variation in this value. Similarly, the calculation of the minimum distance d usually takes longer when the generator matrix G and base of the field q are larger; however, it can vary significantly for individual codes. We chose to generate an equal number of codes for each number of vertices.

In the case of codes over $\mathbb{F}_7$, the generation of 100'000 codes for each number of vertices from 5 to 31 is a quick and easy task, and we performed this generation[6]. However, in the case of codes over $\mathbb{F}_8$, we were able to generate 100'000 codes only for the number of vertices between 4 and 36, while the number of vertices from 37 to 49 took too long, therefore, we decided to reduce the number of generated codes to 1'000.

The dataset generated for the codes over $\mathbb{F}_7$ contained 2'700'000 examples. In figure 6.1, a significant variation in the frequencies of different minimum distances, ranging from two to more than 600'000, can be observed. The dataset is highly unbalanced, and some classes have 300'000 times more representatives than others. Therefore, randomly splitting the data into training and test sets would be inadequate. Therefore, we split subsets with equal minimum distance into train and test sets separately and merged them into full train and test sets with oversampling to 500 or downsampling to 50'000 if necessary, and re-shuffling afterwards. The resulting dataset of size around 550'000 was used for training with 9:1 train/test split.

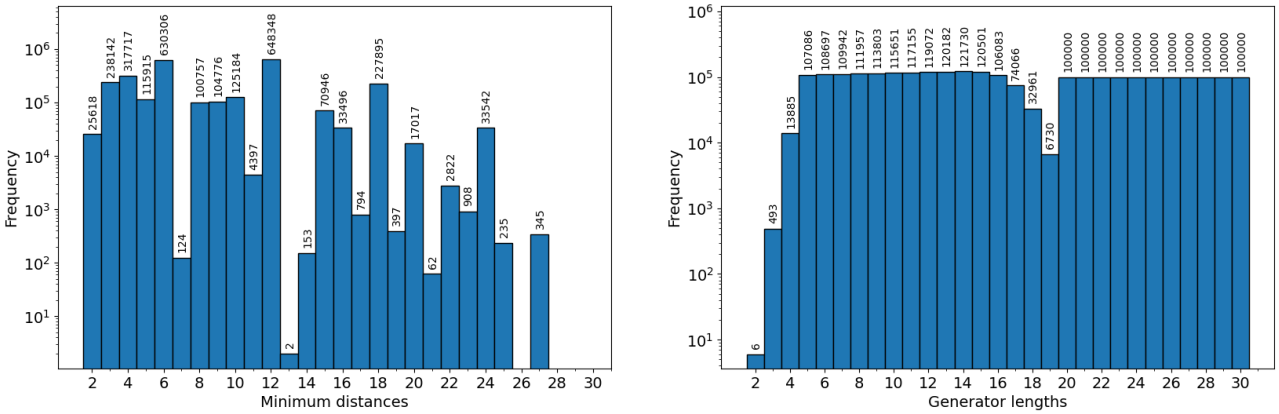


Figure 6.1: Histograms of the $\mathbb{F}_7$ codes dataset. The left histogram shows how often the codes with a specific minimum occur within the dataset. The right histogram shows how frequent generator matrices of various lengths (code dimensions) occur within the dataset. Each bar represents the frequency of codes with a specific minimum distance value (left histogram) or a particular generator dimension (right histogram).

[6]See Appendix D for more details.

For the $\mathbb{F}_8$ case, we were able to generate only a smaller dataset with 1'584'099 examples. The complexity of the prediction for the codes over $\mathbb{F}_8$ is expected to be larger than that for $\mathbb{F}_7$, but the amount of data we managed to collect was notably smaller than that for $\mathbb{F}_7$. To mitigate this, we used a two-stage training procedure with the idea of training the model to extract useful features of codes on common examples in stage I and then generalising this knowledge to all classes in stage II. Stage I training was performed on a larger dataset with approximately 300'000 examples, which included classes with more than 10'000 examples. Then, in stage II training, we used all the classes, down-sampling or over-sampling all available classes to 600, with a total dataset length of 20'000 examples. We used a 9:1 train/test split in both stages.

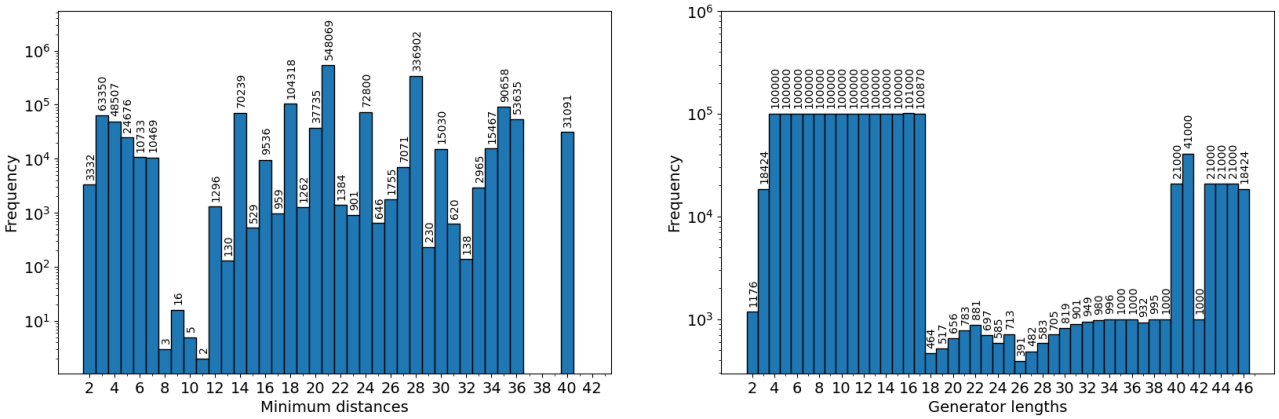


Figure 6.2: Histograms of the $\mathbb{F}_8$ codes dataset, analogous to histograms on figure 6.1. The left histogram is the distribution of minimum distances, the right one is the distribution of generators' dimensions.

Interested readers may find more details about the datasets in Appendix D.1.

6.4.2 Model Architecture

We cast the prediction of the minimum distance as an NLP-style sequence classification task: the input sequences come from interpreting the canonical generator of a code as a sequence of rows $S \in \mathbb{F}_q^{k\times(q-1)^2}$ of length $(q-1)^2$, where the number of rows is the dimension k of the code; and the class label is the minimum Hamming distance $d \in \{0,\ldots,(q-1)^2\}$. Distances of $(q-1)^2$ occur only for trivial single-row codes and are negligible, so we limit classes to $\{0,\ldots,(q-1)^2-1\}$.

Our model is based on a code available at [213], which is a simplified version of the OpenAI GPT-2 model [214]. The original model was designed for sequence-to-sequence tasks; therefore, we made some changes to enable it to work for sequence classification. The main structural elements of the model are

- embedding of the elements of the field $\mathbb{F}_q$: elements of $\mathbb{F}_7$ treated as integers, elements of $\mathbb{F}_8$ were cast into four-dimensional vectors via learnable representation and further concatenated[7],
- sinusoidal positional encoding of rows,

[7]This is done to accommodate for the noninteger primitive element α of $\mathbb{F}_8$ field.

- 2 layers of attention blocks, each with masked self-attention[8] followed by a feed-forward neural network,
- masked attention to eliminate the sequence length dimension in the output (attention pooling),
- softmax transformation to produce the probabilities of classes (minimum distances).

See tables D.2 and D.3 for hyperparameters used, figure 4.4 for general structure of the network[9], and Appendix D for a more detailed description of the model.

6.4.3 Training and performance

To measure performance, we report the proportion of estimates falling within a 3-unit range $(d_V \pm 3)$ of the actual minimum distance (which we will call accuracy for short), alongside the mean absolute error (MAE) and mean squared error (MSE) per class.

For $\mathbb{F}_7$, the model's training set MAE was 1.04, and the accuracy within a tolerance $(d_V \pm 3)$ was 91.9%. On the test set, which consisted of approximately 61,000 examples, the model's MAE was 1.05, and the accuracy within a tolerance $(d_V \pm 3)$ was 91.6%. Figure 6.3 shows the per-class metrics. Note that the MAE for all classes is less than 3, and most MSEs are roughly squares of MAEs, indicating that only a few large errors is present in the model's predictions.

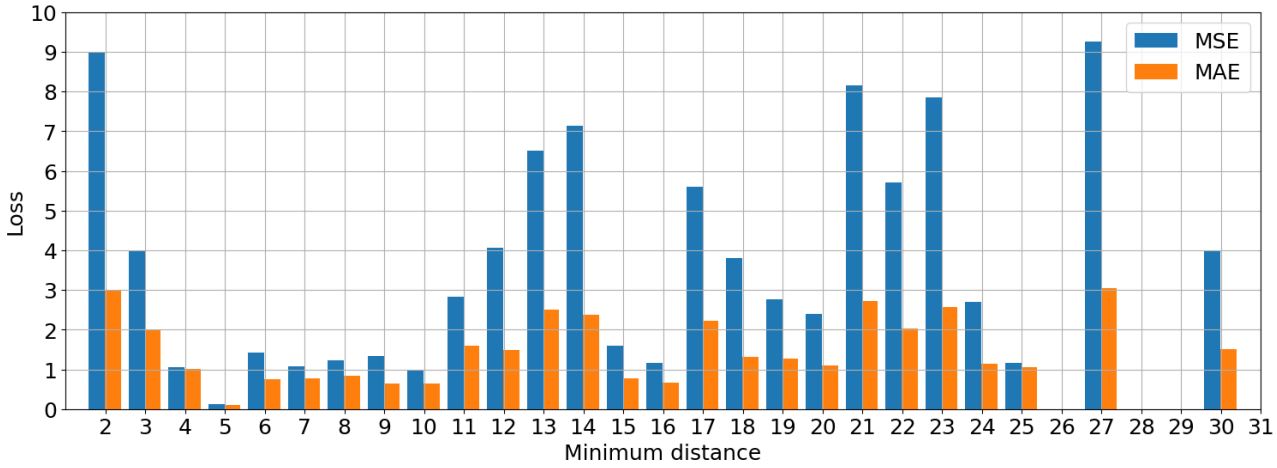


Figure 6.3: Losses on a test set for $\mathbb{F}_7$ codes with different minimum distances for the Transformer model.

For $\mathbb{F}_8$, the model's Stage II training achieved an MAE 1.09 and 93.4% accuracy within a tolerance $(d_V \pm 3)$ on the training set and an MAE 1.21 and an accuracy 92.4% on the test set. Figure 6.4 shows the per-class results. The results here are visibly worse than those for the $\mathbb{F}_7$ case. The MAEs for $d = 14$ are larger than 3, and the MSEs for $d = 7, 14, 20, 21$ and 28 are noticeably larger than the squared MAE, indicating that there are a considerable number of large prediction errors.

[8]Masking need to be used to be able to process the codes of different dimension in parallel.

[9]The only difference of the used architecture compared to the figure 4.4 is that we use Masked Multi-head attention instead of general version.

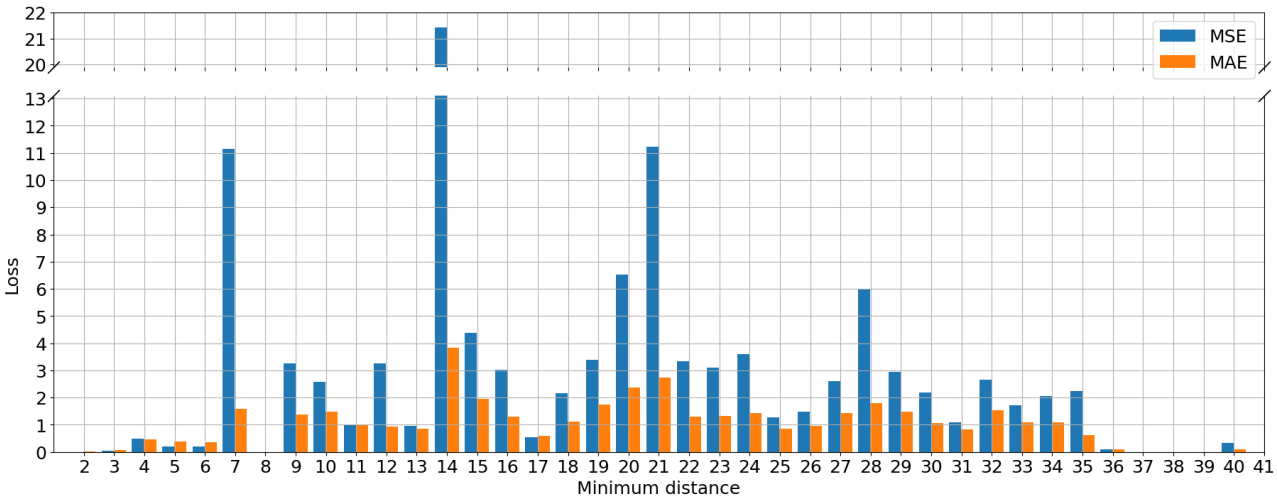


Figure 6.4: Losses on a testset for $\mathbb{F}_8$ codes with different minimum distances.

6.5 Genetic Algorithms

Once we have a model to approximate the minimum distance of a code, we search for champion codes. However, performing a full brute-force scan of the entire space of codes is computationally intractable as there are $2^{(7-1)^2} \simeq 7 \cdot 10^{10}$ possible configurations of vertices for $\mathbb{F}_7$ codes and $2^{(8-1)^2} \simeq 6 \cdot 10^{14}$ configurations for $\mathbb{F}_8$ codes. Inspired by the methodology of [198], we employ a Genetic Algorithm (GA), which enables us to perform the search more efficiently.

The GA search procedure is the following.

Fix a number n and a field $\mathbb{F}_q$, and consider a population of subsets of n vertices $\{V_i\}$, treating each n-vertex set V_i as a chromosome and each vertex (x_i, y_i) as a gene. With the fitness function of our genetic algorithm[10] using the predicted minimum distance of our model, each generation proceeds as follows:

1. Selection: Stochastic universal sampling picks 200 parents from a population of 300, prioritising candidates with a higher fitness function value.

2. Crossover: Flatten each parent's vertex set $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ into a list $[x_1, y_1, \ldots, x_n, y_n]$. A random crossover split point made between vertex pairs swaps the tails; offspring with duplicate vertices are discarded, and we repeat with other parents.

3. Mutation: With 10% probability, a vertex is replaced by a new, unused one, preserving n vertices.

4. Elitism: The best 30 solutions with the highest fitness function value carry forward unchanged, with the rest discarded.

This was repeated for 200 generations.

Note that we evolve vertex sets rather than generator matrices because there is no clear way to mutate the generator matrices and preserve the type of linear code (in this case, generalised toric code), except to mutate the parameters from which they arise.

[10]The code for the genetic algorithms used can be found here, [215].

We target champion codes of dimension k by searching only k-vertex sets and penalising candidates whose dimensions drift from k. Additional penalties[11] prevent rediscovering codes found in earlier runs. This leads to the fitness function Algorithm 1, used both over $\mathbb{F}_7$ and $\mathbb{F}_8$.

Algorithm 1 Fitness Function

Require: Target dimension k, set of k vertices V, set of sets of k vertices S_k corresponding to codes found by the genetic algorithm so far.

Ensure: Fitness function f which implements the penalties described above.

1. If $V \in S_k$, return fitness score of $f = 10$.
2. Else, calculate the dimension $k_V = \dim C_V(\mathbb{F}_q)$.
3. Predict minimum distance of code d_{approx} using model.
4. Return fitness score of $f = 300 + d_{approx} - |k_V - k|$.

During the GA search, we predicted each code's distance with our model (d_{approx}) and then confirmed promising candidates with the expensive BZ brute-force algorithm. Since distance evaluation is the bottleneck of the exploration, and verification is affordable over $\mathbb{F}_7$ but much costlier for $\mathbb{F}_8$, we apply the BZ check more selectively over $\mathbb{F}_8$.

6.5.1 Finding Best Codes in $\mathbb{F}_7$

We apply our $\mathbb{F}_7$ model to rediscover the best generalised toric codes over $\mathbb{F}_7$ from [193] using Algorithm 2.

We ran the genetic algorithm 757 times, and the number of runs required to reach the best code of a given dimension is presented in Table C.1 (Appendix C). As shown, the genetic algorithm discovered codes with the best minimum Hamming distance in a single run for most dimensions. For dimensions such as 12 and 15, many more runs appear to be required. For dimension 19, after 501 runs of the genetic algorithm, it was unable to find a code with the best minimum Hamming distance after 501 runs. This points to either a rarity of $[36, 19, 12]$ generalised toric codes or very poor accuracy of the ML model at this minimum distance.

Comparing Tables C.1 and C.2 shows a loose correlation: dimensions that contain a bigger number of the best codes in the dataset usually require fewer GA runs to rediscover them. This pattern fades at the smallest and largest dimensions, where optimal codes are naturally easier to locate than those in the 12–28 range. This partially explains why finding the best code over dimensions 12, 15, and 19 took far longer than for other dimensions: the dataset held few optimal-code examples at those sizes (this is amplified for dimensions 19 and 28 by the rarity of examples in the training set). Notably, although there were no best codes of dimension 12 in the training and testing datasets, our method still found the best generalised toric code of dimension 12. We see this more apparently over $\mathbb{F}_8$, where we discover new champion codes.

[11] The number 300 in the formula is simply a way to offset the other two terms and make the output of the formula positive as required by the PyGAD library we used [216].

Algorithm 2 Discovering Best Codes over $\mathbb{F}_7$ for Dimensions $3 \leq k \leq 34$

Require: Total number of runs so far, T; list of best known minimum Hamming distances for generalised toric codes over $\mathbb{F}_7$, L_b; list of the best found distances discovered by genetic algorithm so far for each dimension, $L_d = (d_1, d_2, \ldots, d_{36})$; set of discovered codes of dimension i, S_i.

Ensure: Recurring algorithm to discover the best generalised toric codes over $\mathbb{F}_7$.

1. If $T = 0$, then L_d and S_i ($3 \leq i \leq 34$) are empty. For $3 \leq i \leq 34$, apply a genetic algorithm to generate codes of dimension i. Place codes into S_i, calculate their Hamming distances, place the best-discovered minimum distances into L_k, and increase T by 1.
2. If $T \neq 0$, then consider $3 \leq i \leq 34$.
3. If $L_k[i] \neq L_b[i]$, then apply genetic algorithm to generate codes of dimension i. Place codes in S, calculate their minimum distances, and update L_d if required.
4. Increase T by 1.

6.5.2 Finding Champion Codes in $\mathbb{F}_8$

Compared to the exploration in $\mathbb{F}_7$, the highest possible minimum Hamming distance for generalised toric codes over $\mathbb{F}_8$ remains unknown for most dimensions. Therefore, as we are seeking champion codes in $\mathbb{F}_8$, we use the known lower bounds in [202] to provide a safe upper bound for the verification of our codes. We will denote the lower bound for the dimension k as b_k.

During the search, we use an additional criterion to discard codes with a low minimum distance found by the GA. For a code C of dimension k, we call MAGMA's `VerifyMinimumDistanceLowerBound`(C, b_k). If the routine (which uses the BZ algorithm) shows $d_C < b_k$, we discard C immediately; otherwise, we keep it for a full distance check. This filter saves considerable computation by focusing only on codes that *could* beat the current records. Thus, we use Algorithm 3.

Algorithm 3 Discovering New Champion Codes over $\mathbb{F}_8$ for Dimensions $3 \leq k \leq 46$

Require: Total number of runs so far, T; list of lower bounds for minimum Hamming distances of champion codes over $\mathbb{F}_8$, L_b; set of discovered codes of dimension i, S_i.

Ensure: Recurring algorithm to search for champion generalised toric codes over $\mathbb{F}_8$.

1. Begin with S_i ($3 \leq i \leq 46$) as empty sets.
2. For $3 \leq i \leq 46$, apply genetic algorithm to generate codes of dimension i.
3. Place codes into S_i, verify whether they are possible champion codes or not, tag the possible champion codes, and increase T by 1.
4. Repeat until you wish to finish.
5. Once finished, for tagged possible champion codes, apply the BZ algorithm to compute their minimum distances.
6. Verify possible champion codes against known bounds.

We performed one run[12][13] of the above algorithm. Despite this, we found over 700 possibly new champion codes.

As with $\mathbb{F}_7$, we compare these results with the number of champion codes found in the dataset, as shown in Table C.3 (Appendix C).

Once again, champion codes were discovered at the lowest and highest dimensions for reasons similar to those over $\mathbb{F}_7$. Surprisingly, we achieved champion codes in dimensions 18, 22, and 38, which had no champion codes in their datasets, in just a single run. Through explicit calculation compared with the best-known codes in [202], we see that all codes of dimensions 18, 22, and 38 that had been discovered are, in fact, novel. The results are presented in Table C.4 (Appendix C).

6.6 Summary and Outlook

In this study, we demonstrated a promising synergy between ML and genetic algorithms in constructing error-correcting codes with the highest Hamming distance. In particular, our investigation into generalised toric codes over finite fields was two-fold: first, we approximated the computationally expensive task of determining the minimum Hamming distance using a transformer architecture, and then paired this predictive model with a genetic algorithm to construct champion codes.

Our experiments were conducted over both $\mathbb{F}_7$ and $\mathbb{F}_8$ fields.

For $\mathbb{F}_7$, we constructed a large dataset comprising over 2.5 million code examples, with each instance consisting of a canonical generator matrix, its corresponding parity-check matrix, and the minimum Hamming distance computed using the BZ algorithm. The transformer architecture stood out as a possible choice for predicting the minimum Hamming distance. Pairing this transformer model with a genetic algorithm, we observed that for most dimensions, the best-known generalised toric codes over $\mathbb{F}_7$ as described in [193] were found in only a few runs, while other dimensions (e.g. 12, 15, and 19) required much more exploration.

We then extended our approach to $\mathbb{F}_8$, where the computational challenges regarding the BZ algorithm became even more pronounced. Despite the more limited dataset sizes (due to the increased computation times for the minimum distance calculation), our approach yielded over 700 potential champion codes in a single run. After calculating the minimum Hamming distances of these codes, we discovered new champion codes in many dimensions, including previously unknown codes in dimensions 18, 22, and 38.

Our method can be improved on several fronts. Larger and more balanced datasets would provide predictive models with richer coverage, particularly at rare distances. Broader hyperparameter searches for the transformer model including optimisers, other loss functions, alternative row/field encodings could further improve accuracy.

Beyond $\mathbb{F}_7$ and $\mathbb{F}_8$, the same framework can progress over to higher fields of prime powers ($\mathbb{F}_9$, $\mathbb{F}_{11}$, $\mathbb{F}_{13}$), expand to non-generalised toric codes that can exploit algebraic and geometric properties arising from their underlying toric varieties [185], [187], [190]. Moreover, our procedure can, in principle, be generalised to any space of linear codes such as generalised toric codes, Bose-Chaudhuri-Hocquenghem (BCH) [217], [218], Reed-Muller

[12] We ran the algorithm twice for vertices between 20 and 29. In this search, we discovered a champion code of dimension 22, as shown in Table C.4 (Appendix C). Upon discovery, we saved this code. Due to a file error, we were unable to save the corresponding dataset that included this code.

[13] This is due to time and computational constraints.

(RM) [219], [220] or Algebraic-Geometry (AG) [221], [222] codes, that can be explored by a genetic algorithm[14]. Then, one can proceed by generating a dataset consisting of the generator matrices, their minimum distances, etc. The dataset is then used to train a model. After some calibration, a genetic algorithm with minimum distance as a fitness function can be applied to perform a search and find possible new champion codes.

[14]The requirement is that the space of codes should be amenable to genetic algorithms/with an evolvable (can be subjected to a GA) parameter space or a way to explore the space of linear codes. Such properties exist for toric codes because of the description in terms of polytopes (or sets of vertices for generalised toric codes).

Chapter 7

Summary and Conclusions

This thesis explored two directions, Integrability and Machine Learning, which are two very different sets of tools applicable to a wide range of problems in theoretical physics and mathematics. Despite this difference, the two are united by the common goal of extracting structured information from complex systems, whether from the symmetries of quantum field theory or the high-dimensional data of a combinatorial task. We advanced the understanding of string theory in AdS_3 background using integrability and developed new ML techniques to approach problems of a combinatorial nature, demonstrating the complementary power of traditional and computational methods.

In Chapter 2, we considered the AdS/CFT correspondence and integrability for string theories in $AdS_5 \times S^5$ and $AdS_3 \times S^3 \times T^4$ backgrounds. Section 2.1 motivated the AdS/CFT conjecture using brane constructions for both backgrounds. Then, in Section 2.2, we consider a well-understood case of AdS/CFT in $AdS_5 \times S^5$ background. After reviewing the dual $\mathcal{N} = 4$ SYM CFT properties, we switched to the string side of the duality and discussed the symmetry algebra $\mathfrak{psu}(2|2)^2_{\text{c.e.}}$ and its representations, which allowed us to construct the S-matrix for $AdS_5 \times S^5$ superstring. We concluded this section by considering the crossing transformation and the dressing factor.

Section 2.3 was devoted to the integrability in $AdS_3 \times S^3 \times T^4$ background with pure Ramond-Ramond flux. This theory shares many features with the $AdS_5 \times S^5$ case, but also has notable differences, such as the presence of massless modes and fewer supersymmetries. Guided by the example of AdS_5, we considered the symmetry algebra $\mathfrak{psu}(1|1)^4_{\text{c.e.}}$ and its representations, and constructed an S-matrix and dressing factors which are known in the literature.

Based on the intuition built in the previous chapter, we presented new findings for the less understood mixed-flux $AdS_3 \times S^3 \times T^4$ strings in Chapter 3. We reviewed the symmetry algebra $\mathfrak{psu}(1|1)^4_{\text{c.e.}}$ and its deformed, compared to pure R-R case, representations. Despite the same symmetry algebra as in the pure RR case, the deformed representation leads to a distinct kinematical structure of the worldsheet excitations. We then analyse the strong-coupling limit along with other constraints to derive the odd part of the massive dressing phase for the exact S-matrix.

The kinematics and odd part of the dressing phase considered in Chapter 3 are the stepping stones to the construction of the full exact S-matrix and complete control over the mixed-flux $AdS_3 \times S^3 \times T^4$ background. Such control will facilitate a further understanding of the WZW and RNS descriptions of string theory, which appear in the pure NS-NS and pure R-R limits of the mixed-flux theory. At the same time, the AdS/CFT correspondence potentially allows us to map the knowledge of string theory to the symmetric-product

orbifold CFT of T^4 and understand it better as well.

We then switched our focus to the Machine Learning application to combinatorial structures, where the abundance of data makes the application of ML a viable, if not the only available option, to make progress on the problem. The work presented here includes the study of Clifford invariants of ADE Coxeter elements and the search for champion toric codes.

Chapter 4 provided an overview of the Machine Learning techniques used in the subsequent chapters.

In Chapter 5, we studied the Clifford invariants of Coxeter elements in ADE algebra root systems. After introducing the geometric algebra language, we described the dataset generation and proceeded to investigate the general patterns and symmetries present in the data, and applied neural networks for their classification, yielding a high accuracy. The purpose of this study was to explore the possibility of applying Data Science and Machine Learning to the study of Clifford algebras. As a result, we found a range of patterns that might guide further theoretical studies of ADE root systems and their Clifford invariants. There is a potential to use this to generate conjectures based on the patterns, either in an experimental mathematics approach or using symbolic ML approaches. However, the question of whether the high predictive accuracy of machine learning can be converted into theoretical insights remains.

Chapter 6 presents an application of Machine Learning to the search for new champion toric codes. Building on previous work which exhausted the brute-force computation method, we present an approach to bypass the computational complexity of the problem using deep learning. We created a model based on a transformer architecture, which was trained on a limited dataset to predict the Hamming minimum distance of a code. The resulting model is then coupled with a genetic algorithm, which searches for codes with the largest Hamming minimum distance. The obtained champion codes are then validated using conventional algorithmic computation, revealing several new champion codes. This approach can be generalised to other types of codes, such as BCH and RM codes.

More broadly, both works employing ML are among the first of their kind in their respective areas of mathematics and aim to probe new ways for such applications.

Overall, both the analytical approach and AI-assisted methods have their strengths and are best suited to different contexts. By applying one or the other, or integrating them synergistically, the frontiers of theoretical physics and mathematics can be pushed more effectively than relying on either approach in isolation.

Appendix A

NN Gradient Saliency Results

The gradient saliency results, showing the relative importance of the input parts of the Coxeter element permutation for predicting the subinvariants at each order, are presented in the subsequent Tables A.1, A.2, A.3.

	Inv_i	Inv_i^0	Inv_i^2	Inv_i^4	Inv_i^6	Inv_i^8
Inv_0						
Inv_1						
Inv_2						
Inv_3						
Inv_4						
Inv_5						
Inv_6						
Inv_7						
Inv_8						

Table A.1: Summary of gradient saliency analysis for each invariant and subinvariant for A_8 simple root data. The NN function takes as input the permutation performed on the original Coxeter element and outputs the coefficients of the respective subinvariant. The saliency barcodes, hence, represent the relative importance of each root in the permutation for computing the subinvariant coefficients. Lighter colours indicate larger gradients and greater importance.

	Inv_i	Inv_i^0	Inv_i^2	Inv_i^4	Inv_i^6	Inv_i^8
Inv_0						
Inv_1						
Inv_2						
Inv_3						
Inv_4						
Inv_5						
Inv_6						
Inv_7						
Inv_8						

Table A.2: Summary of gradient saliency analysis for each invariant and subinvariant for D_8 simple root data. The NN function takes as input the permutation performed on the original Coxeter element and outputs the coefficients of the respective subinvariant. The saliency barcodes, hence, represent the relative importance of each root in the permutation for computing the subinvariant coefficients. Lighter colours indicate larger gradients and greater importance.

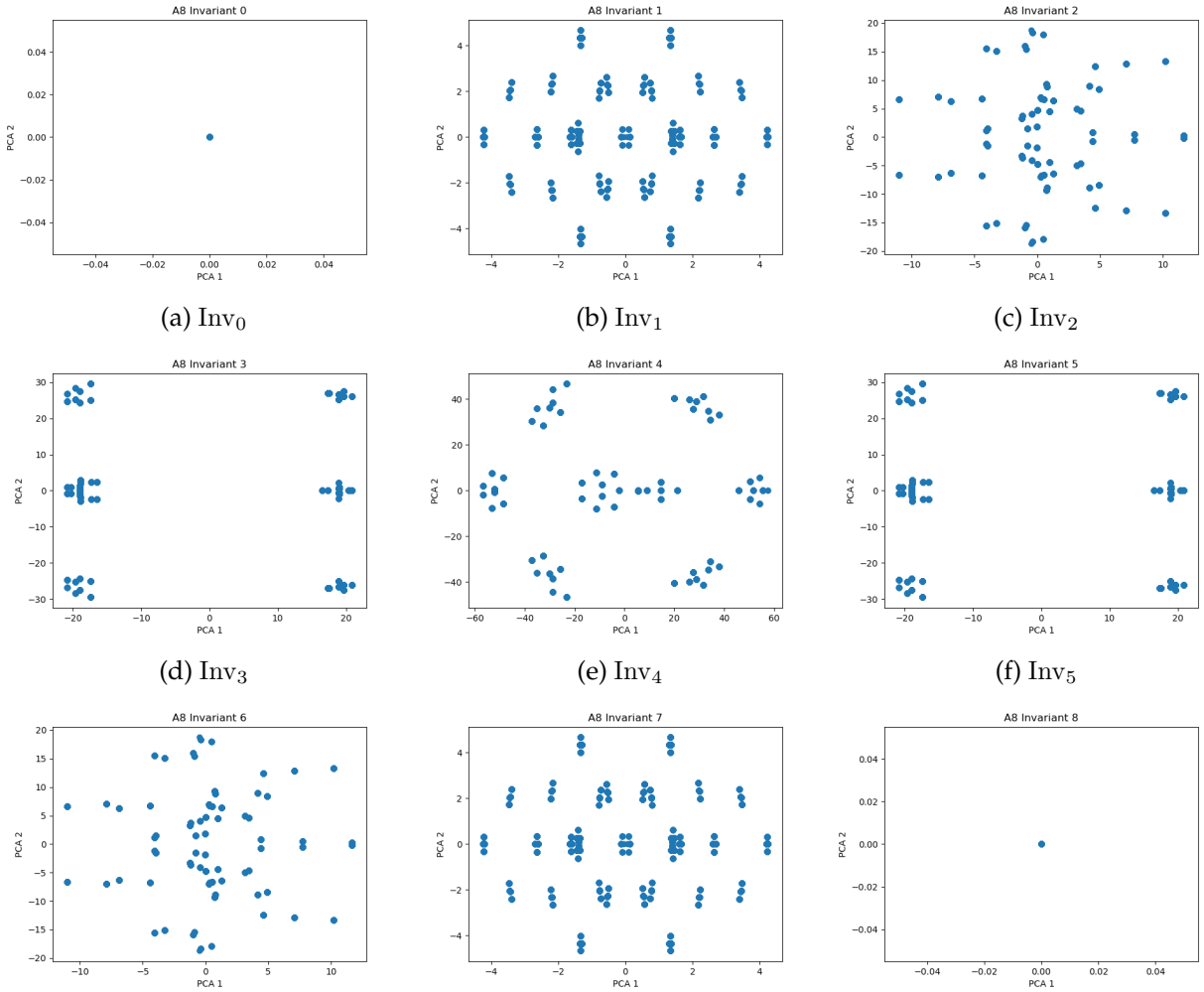
	Inv_i	Inv_i^0	Inv_i^2	Inv_i^4	Inv_i^6	Inv_i^8
Inv_0						
Inv_1						
Inv_2						
Inv_3						
Inv_4						
Inv_5						
Inv_6						
Inv_7						
Inv_8						

Table A.3: Summary of gradient saliency analysis for each invariant and subinvariant for E_8 simple root data. The NN function takes as input the permutation performed on the original Coxeter element and outputs the coefficients of the respective subinvariant. The saliency barcodes, hence, represent the relative importance of each root in the permutation for computing the subinvariant coefficients. Lighter colours indicate larger gradients and greater importance.

Appendix B

PCA Results

The 2-dimensional PCA projections for each dataset of invariants at each order for each root system A_8, D_8, E_8 are shown in Figures B.1, B.2, and B.3 respectively, whilst the 2-dimensional PCA projections for the same partitioning of invariants into orders and types – however now reducing the datasets to unique invariants – are shown in Figures B.4, B.5, and B.6 respectively.

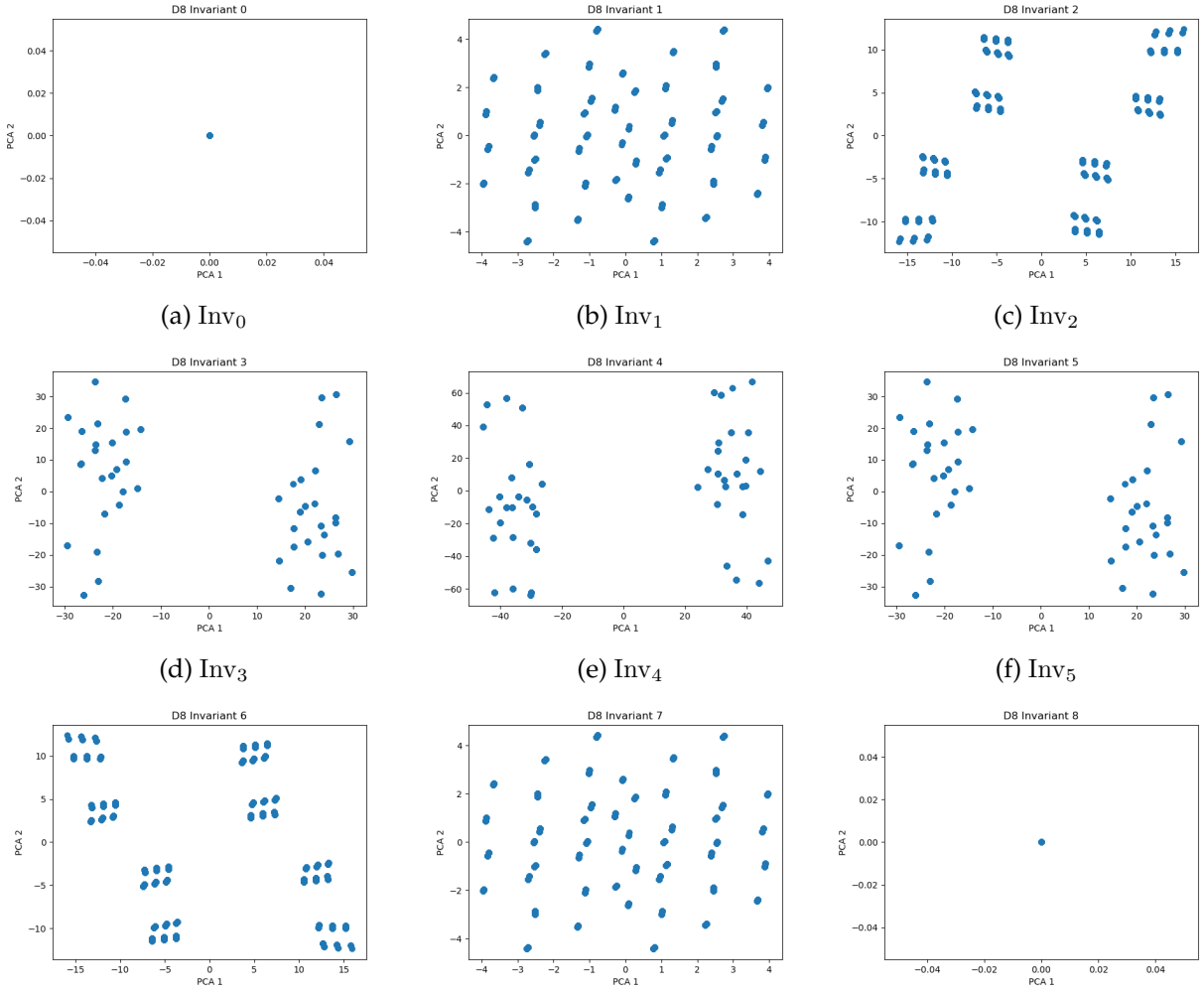


(a) Inv_0 (b) Inv_1 (c) Inv_2

(d) Inv_3 (e) Inv_4 (f) Inv_5

(g) Inv_6 (h) Inv_7 (i) Inv_8

Figure B.1: PCA plots of the 9 orders of invariant (SOCM) for A_8. The observed mirror symmetry is a good consistency check.

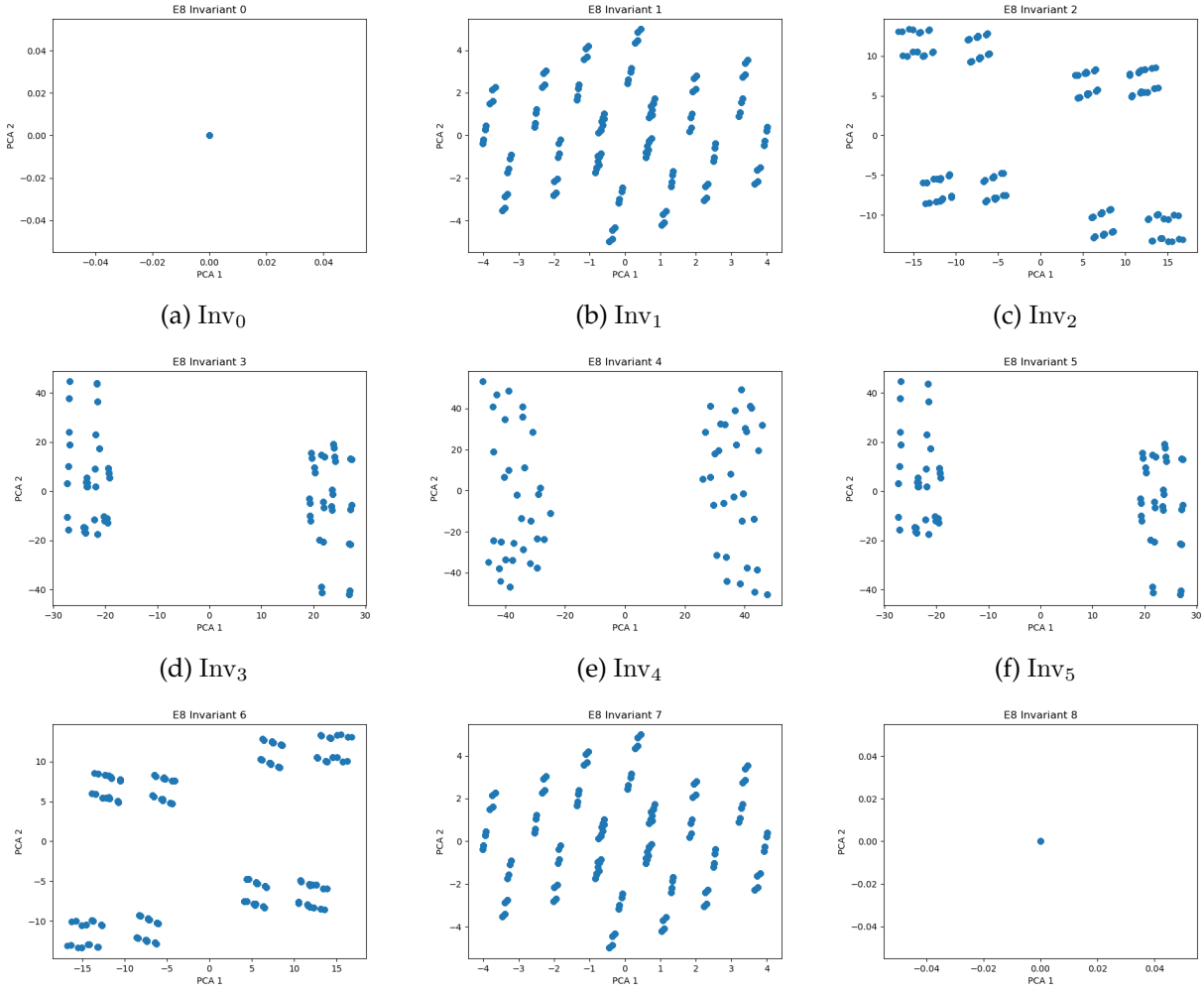


(a) Inv_0 (b) Inv_1 (c) Inv_2

(d) Inv_3 (e) Inv_4 (f) Inv_5

(g) Inv_6 (h) Inv_7 (i) Inv_8

Figure B.2: PCA plots of the 9 orders of invariant for D_8.

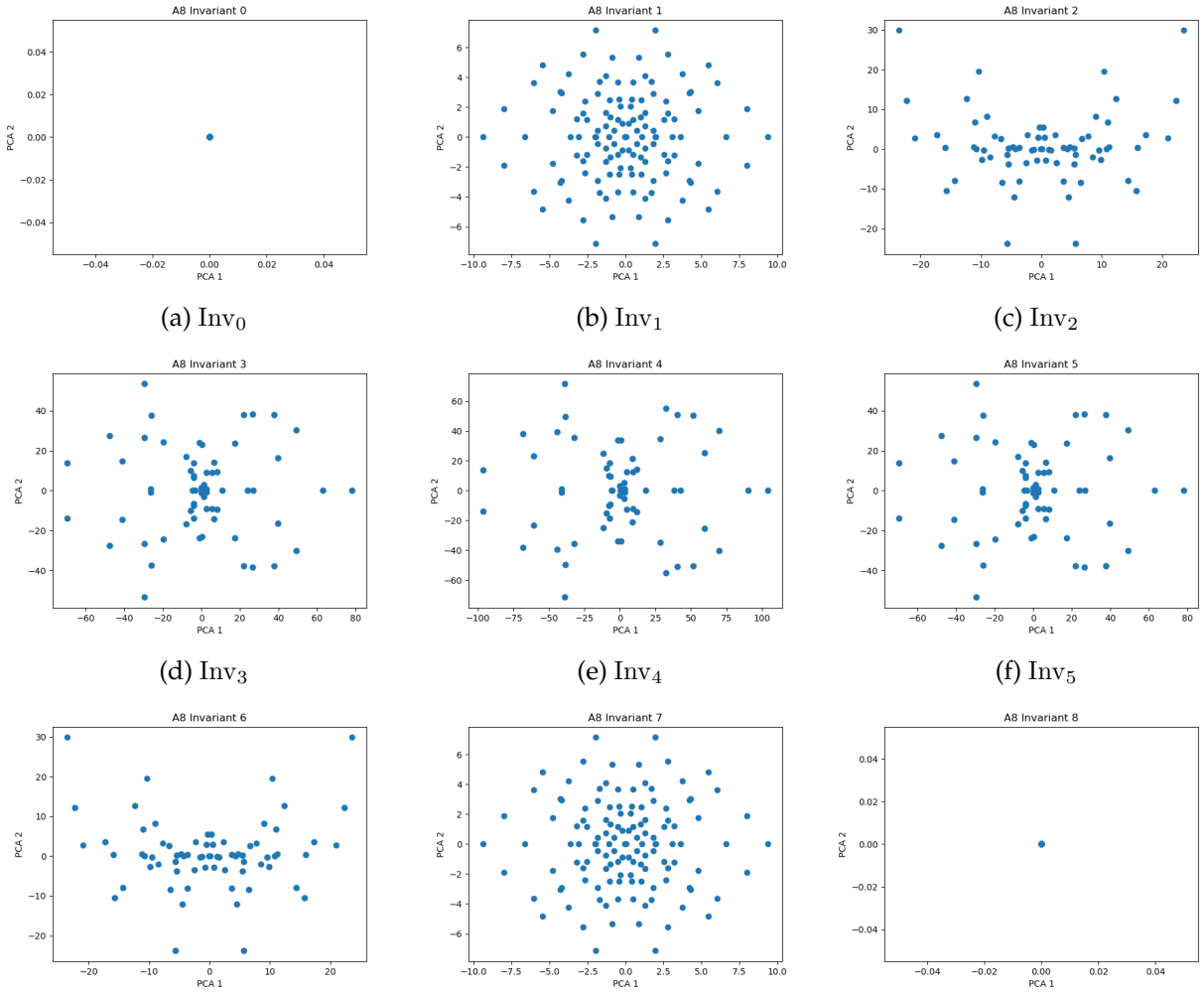


(a) Inv_0 (b) Inv_1 (c) Inv_2

(d) Inv_3 (e) Inv_4 (f) Inv_5

(g) Inv_6 (h) Inv_7 (i) Inv_8

Figure B.3: PCA plots of the 9 orders of invariant for E_8.

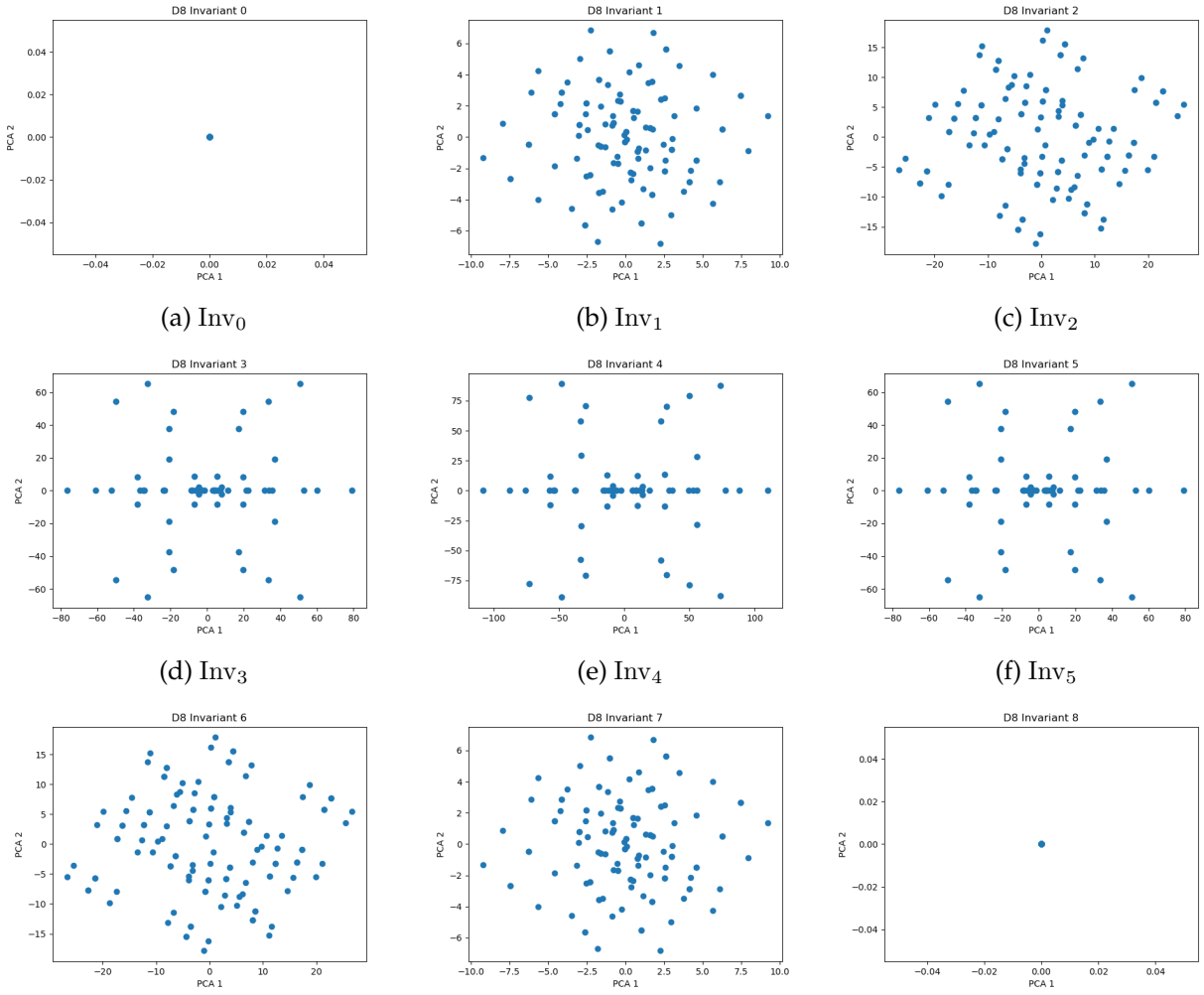


(a) Inv_0 (b) Inv_1 (c) Inv_2

(d) Inv_3 (e) Inv_4 (f) Inv_5

(g) Inv_6 (h) Inv_7 (i) Inv_8

Figure B.4: PCA plots of reduced datasets (with duplicates deleted) of the 9 orders of invariant for A_8.

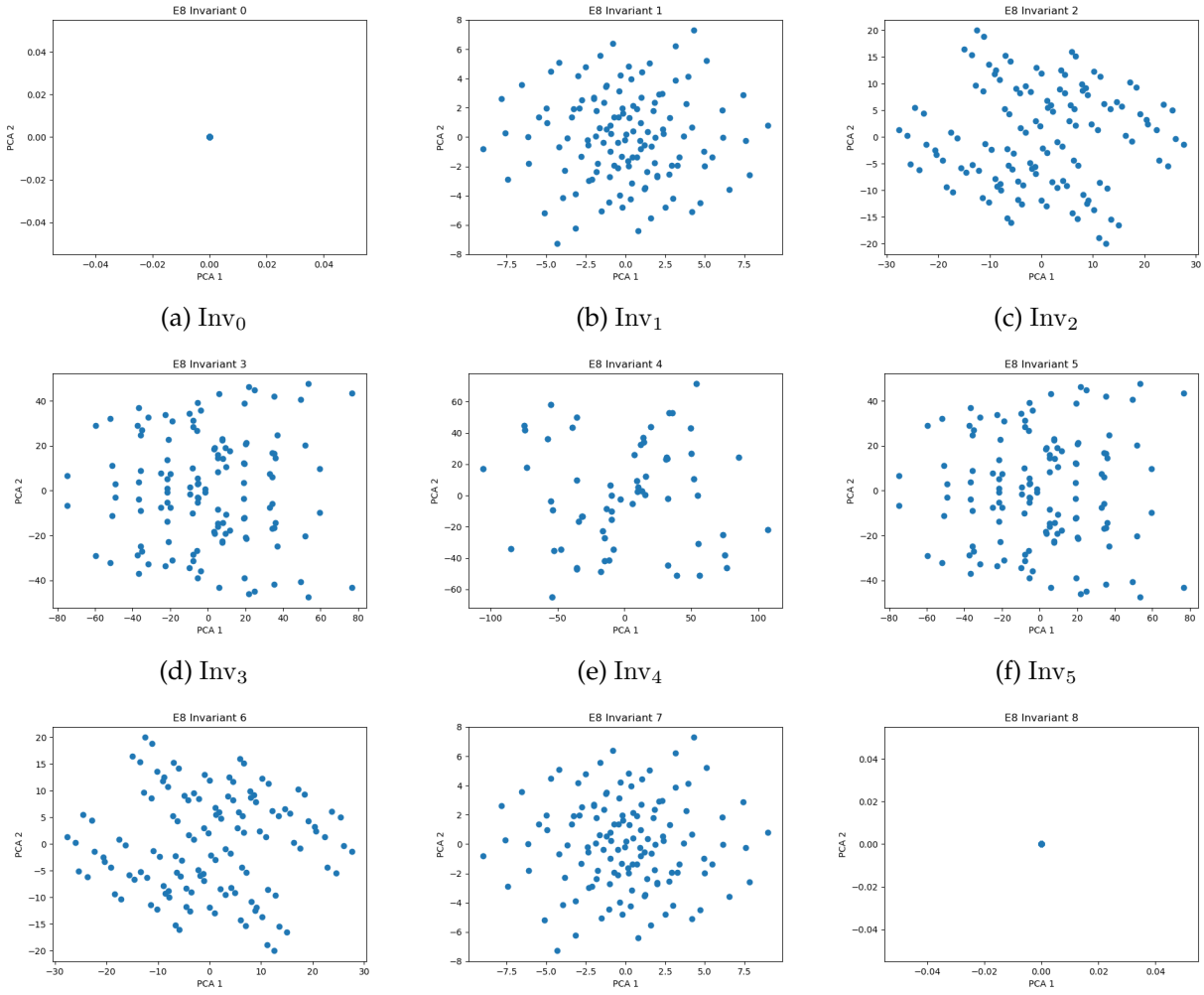


(a) Inv_0 (b) Inv_1 (c) Inv_2

(d) Inv_3 (e) Inv_4 (f) Inv_5

(g) Inv_6 (h) Inv_7 (i) Inv_8

Figure B.5: PCA plots of reduced datasets (with duplicates deleted) of the 9 orders of invariant for D_8.



(a) Inv_0 (b) Inv_1 (c) Inv_2

(d) Inv_3 (e) Inv_4 (f) Inv_5

(g) Inv_6 (h) Inv_7 (i) Inv_8

Figure B.6: PCA plots of reduced datasets (with duplicates deleted) of the 9 orders of invariant for E_8.

Appendix C

Champion codes found over $\mathbb{F}_8$

Table C.4: Champion codes found over $\mathbb{F}_8$, with block length $n = 49$, dimension k, and best minimum distance found d, and number of such codes discovered N.

k	d	N	**Example Vertices**
3	42	231	(1, 0), (2, 3), (6, 3)
4	40	66	(0, 5), (2, 6), (6, 0), (6, 4)
6	36	3	(0, 0), (0, 6), (2, 4), (2, 5), (4, 6), (6, 4)
7	35	1	(0, 3), (2, 5), (4, 6), (5, 0), (5, 1), (6, 0), (6, 3)
8	34	1	(0, 4), (0, 6), (2, 3), (3, 4), (4, 0), (5, 1), (5, 3), (6, 0)
18	21	4	(0, 1), (0, 2), (0, 3), (0, 5), (1, 0), (1, 1), (1, 5), (1, 6), (3, 2), (3, 3), (3, 5), (3, 6), (4, 3), (4, 4), (5, 1), (5, 3), (5, 5), (6, 4)
22	18	1	(0, 0), (0, 6), (1, 0), (1, 3), (1, 4), (1, 5), (2, 0), (2, 5), (2, 6), (3, 2), (3, 3), (3, 5), (3, 6), (4, 2), (5, 0), (5, 3), (5, 4), (6, 0), (6, 1), (6, 3), (6, 4), (6, 5)
38	7	1	(0, 1), (0, 3), (0, 4), (0, 6), (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (2, 0), (2, 1), (2, 3), (2, 4), (2, 6), (3, 0), (3, 1), (3, 3), (3, 4), (3, 5), (3, 6), (4, 0), (4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (5, 0), (5, 1), (5, 3), (5, 4), (5, 5), (5, 6), (6, 0), (6, 1), (6, 2), (6, 5)
40	6	94	(0, 0), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (1, 1), (1, 2), (1, 5), (1, 6), (2, 0), (2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6), (3, 0), (3, 1), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (4, 1), (4, 2), (4, 4), (4, 5), (4, 6), (5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (6, 0), (6, 1), (6, 4), (6, 5), (6, 6)

(Continued on next page)

(Continued from previous page)

k	d	N	**Example Vertices**
41	6	3	(0, 0), (0, 1), (0, 2), (0, 4), (0, 5), (0, 6), (1, 0), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (2, 0), (2, 1), (2, 2), (2, 4), (2, 6), (3, 0), (3, 1), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (4, 0), (4, 1), (4, 4), (4, 6), (5, 0), (5, 1), (5, 2), (5, 3), (5, 5), (5, 6), (6, 0), (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6)
44	4	100	(0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (1, 0), (1, 1), (1, 3), (1, 4), (1, 5), (1, 6), (2, 0), (2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6), (3, 0), (3, 1), (3, 4), (3, 5), (3, 6), (4, 0), (4, 1), (4, 2), (4, 3), (4, 4), (4, 6), (5, 0), (5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (6, 0), (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6)
46	3	235	(0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (1, 0), (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6), (3, 0), (3, 1), (3, 2), (3, 3), (3, 4), (3, 6), (4, 0), (4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (6, 0), (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6)

Dimension	No. of Runs
3	1
4	1
5	1
6	1
7	2
8	1
9	20
10	1
11	3
12	54
13	1
14	2
15	81
16	1
17	1
18	20
19	N/A
20	1
21	22
22	18
23	1
24	30
25	1
26	1
27	1
28	43
29	1
30	1
31	1
32	1
33	1
34	1

Table C.1: Dimension of code over $\mathbb{F}_7$ vs. Number of runs to find the best minimum Hamming distance

Dimension	No. of Best
3	98
4	332
5	23446
6	4023
7	115
8	564
9	27
10	2482
11	124
12	0
13	411
14	16
15	1
16	11838
17	1951
18	66
19	0
20	1094
21	39
22	57
23	336
24	5
25	45714
26	21432
27	4215
28	80
29	2420
30	33302
31	10713
32	0
33	0
34	0

Table C.2: Dimension of code vs. Number of best codes in the dataset over $\mathbb{F}_7$

Dim	No. of Champs
1	0
2	0
3	16464
4	31091
5	0
6	1034
7	216
8	28
9	3
10	11
11	1
12	144
13	1
14	0
15	146
16	1

Dim	No. of Champs
17	0
18	0
29	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0
29	0
30	0
31	0
32	0

Dim	No. of Champs
33	0
34	0
35	0
36	0
37	0
38	0
39	0
40	6138
41	357
42	0
43	0
44	5177
45	0
46	16464
47	0
48	1176

Table C.3: Dimension of code vs. Number of champion codes in the training and testing datasets over $\mathbb{F}_8$.

Appendix D

Machine Learning Experiments for Generalised Toric Codes

This appendix expands the description of the ML experiments from Section 6.4.

D.1 Datasets

In this section, we give more details on how the dataset was generated and prepared for training.

D.1.1 $\mathbb{F}_7$ dataset

To obtain a representative dataset, we generated a dataset of 100'000 tuples of the form (V, G_V, d_V) [1] for each number of vertices $m \in [5, 31]$. For a fixed m, we followed an algorithm 4. The resulting dataset is shown in Figure 6.1.

Algorithm 4 Generating datasets of size up to 100'000 over $\mathbb{F}_q$, for number of vertices m

Require: A positive integer $m \leq (q-1)^2$

Ensure: A dataset S^* of 100'000 distinct tuples of the form (V, G_V, P_V, d_V) for $|V| = m$ vertices.

1. Generate a set S of 100'00 distinct uniformly random subsets of $[0, q-2]^2$ of size m.
2. Take $V \in S$, a set of m vertices of $[0, q-2]^2$.
3. Define C_V, using the order defined in 3.1.
4. Calculate G_V in standard form, and d_V.
5. Return the tuple (V, G_V, d_V) in a dataset, S^*.
6. Repeat for each $V \in S$.
7. Return S^*.

Note that the total number of subsets in our lattice grid is $2^{(q-1)^2}$ and the number of

[1] The initial investigation showed that the generator matrices G_V and dual generator matrices P_V contain essentially the same information; therefore, we continued to generate datasets as a collection of tuples (V, G_V, d_V).

subsets of size m is $\binom{(q-1)^2}{m}$. Thus, only for $5 \leq m \leq 31$ there may exist more than 100'000 distinct codes over $\mathbb{F}_7$.

Generating the $\mathbb{F}_7$ dataset using Magma, which can perform commutations in parallel, on a computer cluster node with 96 Intel Xeon Gold 6442Y processors (48 cores, 96 threads, 2.6-4.0 GHz) took approximately 24 h.

The resulting dataset of codes over $\mathbb{F}_7$ contains 2'700'000 examples. As illustrated in Figure 6.1, there is considerable variation in the frequencies associated with different minimum distances, with counts for individual minimum distances ranging from as few as two instances to over 600'000.

Training on such an unbalanced dataset requires additional steps beyond simply partitioning the data into train and test sets randomly. Specifically, it is important to split each subset of the dataset with the same minimum distance into train and test sets separately, merging them into full train and test sets with re-shuffling afterwards. Furthermore, class imbalance can be further mitigated through the application of oversampling/downsampling of the minority/majority classes. As will be described below, the model trained for codes over $\mathbb{F}_7$ uses cross-entropy loss, which also allows for down-weighting the losses for oversampled and up-weighting downsampled classes.

The final dataset for $\mathbb{F}_7$ codes contained around 550'000 and was used for training with a 9:1 train/test split. The subsets with fewer than 500 elements were oversampled with replacement to 500 elements, and the subsets with more than 50'000 examples were downsampled to 50'000[2]. The only class with fewer than 10 elements, corresponding to a minimum distance of 13, was used in the test set only.

It is important to note that the dimension of code C_V (the number of rows of the generator) can be equal to or smaller than the number of vertices from which the code is generated. This occurs when the rows of the linear code are not linearly independent. Conversely, the dimension can be larger than the number of vertices for the dual codes.

D.1.2 $\mathbb{F}_8$ dataset

In the case of codes over $\mathbb{F}_8$, we were able to generate 100'000 codes only for the number of vertices 4-17, following an algorithm similar to 4. Due to the computation time required to calculate the minimum distances of codes of higher dimensions, we cut the number of generated codes for the number of vertices 17-39 to 1'000, with a limit on the computation time of 1 minute. An additional 20'000 codes were generated for the vertex numbers $m = 2, 3, 40 - 47$ afterwards. As can be seen in the figure 6.2[3], for $m = 2, 3$, the number of existing codes is smaller than 20'000 examples. A similar situation occurred with $m = 47$. The variation in the number of examples by minimum distance in the region $m = 17 \ldots 40$ is caused by the limit on the minimum distance computation time we set in Magma, demonstrating the complexity of the procedure in this region. The resulting dataset contained 1'584'099 examples.

Generating the $\mathbb{F}_8$ dataset using Magma on a computer node with 96 Intel Xeon Gold 6442Y processors (48 cores, 96 threads, 2.6–4.0 GHz) took around 48 hours.

An unexpected feature of the codes over $\mathbb{F}_8$, compared to $\mathbb{F}_7$ codes, is that the dimensions of codes are always equal to the number of vertices from which these codes were generated.

[2]The goal was to get a moderately imbalanced dataset with the smallest classes being no smaller than 1% of the biggest classes.

[3]The outlier at $m = 42$ is a result of a mistake when handling the data: 20'000 data points for $m = 41$ were added twice, and 20'000 for $m = 42$ were never added to the dataset. It was spotted after all the experiments were completed. The results are presented as is because it would be very time-consuming to repeat all the processes.

The minimum distance prediction task for the codes over $\mathbb{F}_8$ is expected to be more complicated than for $\mathbb{F}_7$, but the amount of data we managed to collect was marginally smaller than for $\mathbb{F}_7$. To compensate for this data limitation, we developed a two-stage training procedure. A pre-training stage was done on a larger dataset with approximately 300'000 examples, which included subsets (codes with the same minimum distance) with more than 10'000 examples; subsets with more than 20'000 examples were downsampled to 20'000. Then, in the training stage, all the subsets were used, with down-sampling or over-sampling to 600.

In goal was to train the model to extract useful features of codes on the common examples in the pre-training and then generalise this knowledge to all subsets in the training stage. In the latter had a 10 times smaller learning rate was used to change the weights on a smaller scale than in pre-training to avoid losing any learned features.

The resulting performance is described in Section 6.4.3. Here we will point out a couple of features of the found results.

Figure 6.3 reveals a notable trend: the model exhibits inferior performance for codes with minimum distances $d = 2, 14, 22,$ and 27. However, a worse performance can be explained by the scarcity of training examples only for $d = 7, 14$ and 27. In contrast, the minimum distance $d = 2$ and 22 codes are well-represented in the dataset. These performance variations may indicate a different structure of the codes with these minimum distances from others.

Figure 6.4 reveals a similar pattern for the $\mathbb{F}_8$ case. The codes with minimum distances $d = 7, 14, 21$ and 28 codes are more difficult for the model to predict accurately. This behaviour cannot be attributed to data scarcity, which may again indicate that these codes are different structurally from those with other minimum distances.

D.1.3 LSTM for Codes over $\mathbb{F}_7$

In Section 6.4, we mentioned several architectures suitable for the sequence classification problem we consider. We started our experiments with a smaller dataset and used an LSTM model to produce a baseline to compare with future experiments.

Initially, generated dataset of 100'000 tuples (V, G_V, P_V, d_V) were generated by drawing a random number $m \in [3, 28]$ of vertices and then selecting random vertex coordinates. See figure D.1.

The result is an imbalanced dataset with many minimum distances not being represented, which demonstrates the complexity of generating a representative dataset for all minimum distances. For instance, a rare minimum distance $d = 2$, which appears only once, originates from the code with dimension $k = 22$. There were 3755 codes with dimension $k = 22$, and the majority (88%) of them had a minimum distance $d = 6$. Codes with dimension $k = 23$ may seem more likely to produce a code with a minimum distance $d = 2$, but there were no such examples in the generated dataset.

This initial $100'000$ codes dataset was randomly split into training and test sets in a 9:1 ratio. Because the dataset was very unbalanced and no measures to rebalance it were taken, the test set was missing minimum distances of 2, 7, 17, and 36 compared to the whole dataset.

The model takes as input generator matrices [4] in batches. The target outputs d in the dataset were rescaled to be in region $[0, 1]$ using a formula $\tilde{d} = (d - \mu)/\sigma$, where $\mu = (30 + 3)/2 = 16$ and $\sigma = (30 - 3)/2 = 13.5$.

The model consisted of two layers of stacked LSTM. After the second layer, the hidden

[4] Without any encoding. To be discussed later.

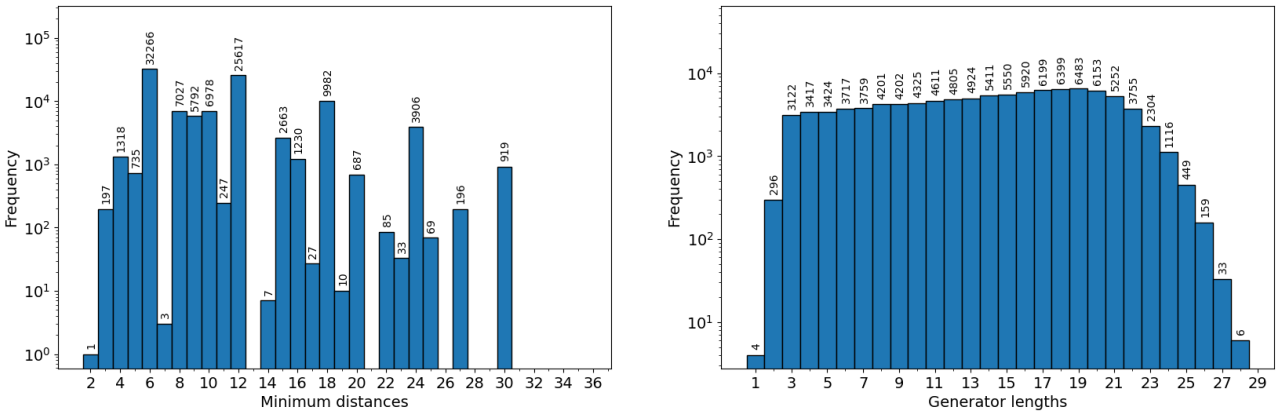


Figure D.1: Distributions of Minimum distances d and dimensions of codes k (equal to the number of rows in generator matrix G) in the initial dataset of 100'000 codes over $\mathbb{F}_7$. Generated by drawing a random number $m \in [3, 28]$ of vertices and then choosing random vertex coordinates.

and cell states were extracted, concatenated, and passed to a linear layer projecting the output onto one continuous variable. The hyperparameters are presented in Table D.1.

Hyperparameter Type	**Parameter Description**	**Value**
Network architecture	Input dimension	36
	Hidden dimension	128
	Number of layers	2
	Batch size	4
Training parameters	Optimizer	AdamW
	Learning rate	$1 \cdot 10^{-5}$
	Epochs	200

Table D.1: LSTM architecture and training hyperparameters for $\mathbb{F}_7$ codes minimum distance prediction.

On a training set, the model's MAE was 2.72 and the accuracy within a tolerance $(d_V \pm 3)$ was 93.0%. In the test set, the model's MAE was 3.02 and the accuracy within a tolerance $(d_V \pm 3)$ was 91.2%. See figure D.2 for per-minimum-distance metrics. The model performed better for the lower part of the minimum distance spectrum and worse for the upper part. This model serves as a proof of concept that it is possible to learn to predict the minimum distance of a code, and the model has a generalisation capability for unseen data.

Ultimately, the goal of the model is to be an estimator of the minimum distance, to be used in conjunction with the Genetic algorithm (GA) in Section 6.5. During our experiments with GA, we observed that the final population after the GA run was somewhat noisy – that is, the dimensions of the resulting codes' minimum distances were distributed in some regions around the maximum value. The range between the smallest and the largest minimum distance typically varies from 3 to more than 10. As a result, the model does not need to be 100% accurate to be efficient in this context; a MAE error within a reasonable range

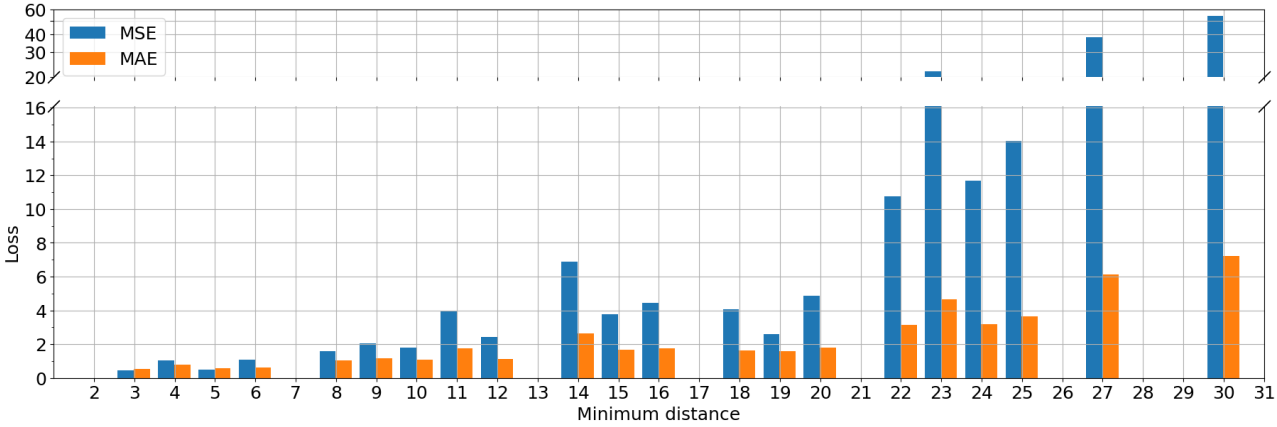


Figure D.2: Losses on a test set for $\mathbb{F}_7$ codes with different minimum distances for the LSTM model.

would still be sufficient for practical use.

D.1.4 Toric Transformer

As mentioned in Section 6.4.2, recent advances in NLP are driven by the transformer architecture, which has demonstrated high effectiveness on various sequence processing tasks. It is therefore reasonable to test it in the current setting, which can be viewed as a sequence classification task.

Compared to NLP transformers, there are several architectural changes compared we implement for this task:

- input encoding of the field elements (for $\mathbb{F}_8$),
- masking to prevent padding from affecting results,
- attention pooling to produce a sequence length independent output,

which we discuss below in the description of the architecture[5].

The general structure of the model is depicted in Figure 4.4 and is shared by both models for $\mathbb{F}_7$ and $\mathbb{F}_8$ codes.

Input

The input of the model was composed of a code tensor and a length vector. The code tensor is a PyTorch tensor containing a batch of generator matrices, padded with zeros to standardise their dimensions. This tensor had three dimensions denoted as (B, T, C), where

- B – batch dimension (i.e. the number of examples in a batch),
- T – time dimension, corresponding to the maximum number of rows in the generator matrices (i.e. $(q-1)^2$)[6]
- C – number of channels, corresponding to the length of each row in the generator matrix (i.e. $(q-1)^2$ as well).

[5]Short architecture description was given in 6.4.2 as well.

[6]For individual generator matrices, this dimension has a size equal to the code dimension k, but when padded and packed in the code tensor, it is equal to $(q-1)^2$.

The length vector stores the actual number of rows (i.e., the code dimension k) of each generator matrix in the batch. It can be used to distinguish where the true generator matrix rows and the padding within the code tensor.

Input embedding

The input sequence of the rows of the generator matrix in a batch is passed through an embedding layer. Embedding is a useful technique originating from NLP. The idea is to map the elements of the input sequence into some higher-dimensional space. In NLP, text data is first split into elementary building blocks of the sequence – tokens, which are then represented as vectors in a higher-dimensional vector space, allowing the ML model to process them effectively. In our setting, the rows of the generator and parity-check matrices are already in a vector form with entries of the vectors being elements of the $\mathbb{F}_q$ field. However,

- elements of $\mathbb{F}_7$ field are numbers in the range $[0, 6]$ and thus the rows can be used as an input "as is";
- elements of $\mathbb{F}_8$ field cannot be written as integers because $\mathbb{F}_8$ field does not have a multiplicative generator. If we introduce α as a notation for the primitive element, then the elements of the $\mathbb{F}_8$ field can be written as 0, 1, α, α^2, $\ldots$, α^6, $\alpha^7 = 1$. This representation was used in the Magma CAS. However, these elements are not linearly independent, and although one-hot encoding was found to be efficient in the $\mathbb{F}_7$ case, using it is not reasonable. The alternative we used was to include an embedding layer with learnable representation, which is less efficient[7] compared with a one-hot representation.

Attention blocks

The attention mechanism in the Transformer architecture and embedding align naturally with the structure of the problem of predicting the minimum distance. By passing the rows of the generator matrix to the model, it will first represent them as some vectors via the learned representation layer. These representations can then be compared in attention heads with each other, and useful information for the prediction of the minimum distance can be extracted.

In our model, after the addition of positional encoding, the result was passed through an encoder-only set of self-attention blocks, each containing an attention layer and a feed-forward neural network.

Pooling layer

In standard transformer architecture, the number of dimensions of the input, which is of shape (B, T, C) in our case, is preserved on all the stages. But for classification tasks, a single fixed-size vector per sequence is needed, so we must eliminate the time dimension T in the output, i.e. introduce a pooling layer. There are many variants in the literature, but the simplest are:

- BERT-like pooling – introduce the special classification token [CLS] in the input, and then use the output at [CLS] for classification,

[7] In the sense that training to the same level of loss generally takes more epochs for a learnable embedding layer.

- average pooling – simply take the average of all outputs along the T dimension and use for classification,

- attention pooling – applying an attention mechanism to compute a weighted sum of sequence elements.

However, it was shown [151] that the latter two are somewhat better than the first approach, so we used them in our model.

- We used average pooling in $\mathbb{F}_7$ case, and it demonstrated promising results despite being a very loose form of summarisation. The reason for this might be that within the attention layers, the rows of the generator matrix exchange information with each other and form more or less similar representations of the code;

- we expected the $\mathbb{F}_8$ case to be more complicated, so the attention pooling was implemented. The idea is essentially to do a transformation $v \to \text{softmax}(v^T M)v$ with a learnable square matrix M. It can reproduce the mean when M is proportional to the identity matrix and, therefore, should provide results that are not worse than the simple mean.

Output

After that, a linear classification layer, projecting from the embedding space to the output vector of dimension $(q-1)^2$ (the number of possible minimum distances treated as $(q-1)^2$ independent classes), is added on top of the pooled output. Finally, a softmax function was applied to get probabilities of classes vector $\vec{p}$, which were then used to compute the expectation of the minimum distance as $\mathbb{E}[d] = \vec{p} \cdot \vec{i}$ where $\vec{i}$ is the class labels (target minimum distances) vector.

Since we were working with padded data, we had to implement masking to prevent padding elements from influencing the results in the attention blocks and pooling layers.

Loss functions

Since we are computing the expectation of the minimum distance, we could have used just the mean square error loss between the expectation and the true minimum distance. However, this loss would discard all the information about the individual probabilities of the classes. To use this information, one can use Wasserstein loss (or distance) or a Cross-Entropy loss.

Wasserstein loss, sometimes referred to as Earth Mover's Distance, has more applications with Generative Adversarial Networks. It measures the distance defined between probability distributions using some metric (we choose just Euclidean distance). On the other hand, Cross-Entropy loss measures the difference between two probability distributions, but it is not a symmetric metric. It is more suitable for classification problems and thus has more use in NLP.

The current task naturally suggests using Wasserstein loss to bring the model's output distribution to a point-mass distribution, which we did for the $\mathbb{F}_7$ case. However, experiments demonstrated a superior performance of Cross-Entropy loss, which we continued to use for $\mathbb{F}_8$ codes.

Hyperparameters

See Table D.2 and D.3 for the hyperparameters of transformer models for codes over $\mathbb{F}_7$ and $\mathbb{F}_8$, respectively.

Hyperparameter Type	Parameter Description	Value
Network architecture	Embedding	No
	Input dimension	36
	Embedding dimension	200
	Attention heads	10
	Transformer layers	2
	Dropout	0.2
	Loss function	Wasserstein-2
	Batch size	32
Training parameters	Optimizer	AdamW
	Learning Rate	$3 \cdot 10^{-4}$
	Epochs	22

Table D.2: Transformer architecture and training hyperparameters for $\mathbb{F}_7$ codes minimum distance prediction.

Improvements

We used the models described above to search for champion codes using a genetic algorithm. We started experiments with the $\mathbb{F}_7$ codes model, iteratively improving it, at both the architecture and training procedure. These experiments showed that a number of modifications can be made to improve performance.

We found that instead of using the rows of the generator matrix as it is, it is more efficient to encode elements of rows using one-hot encoding. That is, we replaced entries in the input row by vectors of length seven containing zeros everywhere except one position unique to each of the numbers 0-6 (and flattened afterwards). As a result, the input vector of length 36 is replaced by a vector of length $36 \cdot 7 = 252$.

As described in 6.4.3, we used an oversampling technique to rebalance the dataset. To avoid biases in the model's output distribution caused by duplicated examples, we down-weighted the oversampled examples during training.

Using attention pooling instead of average pooling also demonstrated improved results.

Finally, in the case of $\mathbb{F}_7$, we used the whole dataset divided into training and test sets. But unless we oversample some classes by a factor of thousands, the datasets remain extremely unbalanced. To overcome this problem, we introduced the two-stage training procedure:

- in the pre-training stage, we used classes which has many representatives (more than

Hyperparameter Type	Parameter Description	Value
Network architecture	Embedding	Learnable embedding
	Input dimension	49
	Embedding dimension	196
	Attention heads	7
	Transformer layers	2
	Dropout	0.2
	Loss function	Cross-entopy
	Batch size	128
Training parameters	Optimizer	AdamW
	Learning Rate pre-training	$3 \cdot 10^{-6}$
	Learning Rate potty-training	$3 \cdot 10^{-7}$
	Epochs pre-train	120
	Epochs post-train	120

Table D.3: Transformer architecture and training hyperparameters for $\mathbb{F}_8$ codes minimum distance prediction.

10'000 for $\mathbb{F}_7$), and trained a model to learn general patterns useful for minimum distance prediction;

- in the training stage, we aimed to have all available classes represented, so we downsampled classes with many representatives (to 500 for $\mathbb{F}_7$), and upsampled classes with not enough representatives (to 500 for $\mathbb{F}_7$), and trained the pre-trained model on the resulting dataset. The second training stage after the pre-training has a relatively small dataset (14'000 for $\mathbb{F}_7$), so the model is prone to overfitting, so early stopping should be used.

Overall, this showed excellent results for $\mathbb{F}_7$ codes with the next iteration of the model (not used with GA), achieving Cross-Entropy loss of 0.61 on the pre-train dataset and 1.26 on the train set, accuracy within a tolerance $(d_V \pm 3)$ of 84.8% on the train set and 90.3% on test set[8].

The application of this methodology for $\mathbb{F}_8$, however, demonstrated less inspiring results. Already at the pre-training stage with plenty of data (the dataset was roughly the same in size as for the $\mathbb{F}_7$ model with two-stage training), the model was prone to overfitting after 120 epochs.

In general, there are two ways to deal with it: increase the dataset size or decrease the number of parameters in the model. Knowing that the task of predicting minimum distance for $\mathbb{F}_8$ is more complicated than for $\mathbb{F}_7$, decreasing the model size is not an option. Therefore,

[8] A counter-intuitive higher accuracy on the test than one the training set is caused by the fact that the training set has oversampled classes.

one might hope to improve the results by increasing the dataset size, but data generation is very computationally expensive for $\mathbb{F}_8$ codes.

Bibliography

[1] G. 't Hooft, *Dimensional reduction in quantum gravity*, 1993. DOI: 10.48550/ARXIV.GR-QC/9310026 arXiv: gr-qc/9310026 [gr-qc].

[2] L. Susskind, "The world as a hologram," *Journal of Mathematical Physics*, vol. 36, no. 11, pp. 6377–6396, 1995. DOI: 10.1063/1.531249

[3] J. M. Maldacena, "The large-n limit of superconformal field theories and supergravity," *Adv. Theor. Math. Phys.*, vol. 2, no. 4, pp. 231–252, 1998. DOI: 10.4310/ATMP.1998.v2.n2.a1 arXiv: hep-th/9711200.

[4] L. Rosenfeld, "Zur quantelung der wellenfelder," *Annalen der Physik*, vol. 397, no. 1, pp. 113–152, 1930.

[5] B. S. DeWitt, "Quantum theory of gravity. i. the canonical theory," *Physical Review*, vol. 160, no. 5, p. 1113, 1967. DOI: 10.1103/PhysRev.160.1113

[6] K. Papadodimas and S. Raju, "An infalling observer in ads/cft," *Journal of High Energy Physics*, vol. 2013, no. 10, pp. 1–73, 2013. DOI: 10.48550/arXiv.1211.6767

[7] D. Harlow and H. Ooguri, "Constraints on symmetries from holography," *Physical review letters*, vol. 122, no. 19, p. 191 601, 2019. DOI: 10.1103/PhysRevLett.122.191601 arXiv: 1810.05337 [hep-th].

[8] J. A. Minahan, "Review of ads/cft integrability, chapter i. 1: Spin chains in super yang-mills," *Letters in Mathematical Physics*, vol. 99, no. 1, pp. 33–58, 2012. DOI: 10.1007/s11005-011-0522-9 arXiv: 1012.3983.

[9] A. Torrielli, "Classical integrability," *Journal of Physics A: Mathematical and Theoretical*, vol. 49, no. 32, p. 323 001, 2016. DOI: 10.1088/1751-8113/49/32/323001 arXiv: 1606.02946 [hep-th].

[10] G. Arutyunov, "Student seminar: Classical and quantum integrable systems," *Lecture notes for lectures delivered at Utrecht University*, vol. 20, 2006.

[11] M. R. Douglas, "The statistics of string/m theory vacua," *Journal of High Energy Physics*, vol. 2003, no. 05, p. 046, 2003.

[12] Y.-H. He, "Machine-learning the string landscape," *Physics Letters B*, vol. 774, pp. 564–568, 2017. DOI: 10.1016/j.physletb.2017.10.024

[13] J. Carifio, J. Halverson, D. Krioukov, and B. D. Nelson, "Machine learning in the string landscape," *Journal of High Energy Physics*, vol. 2017, no. 9, pp. 1–36, 2017.

[14] D. Krefl and R.-K. Seong, "Machine learning of calabi-yau volumes," *Physical Review D*, vol. 96, no. 6, p. 066 014, 2017.

[15] F. Ruehle, "Evolving neural networks with genetic algorithms to study the string landscape," *Journal of High Energy Physics*, vol. 2017, no. 8, pp. 1–20, 2017.

[16] K. Bull, Y.-H. He, V. Jejjala, and C. Mishra, "Machine learning cicy threefolds," *Physics Letters B*, vol. 785, pp. 65–72, 2018.

[17] Z. Liu and M. Tegmark, "Machine learning hidden symmetries," *Physical Review Letters*, vol. 128, no. 18, p. 180 201, 2022.

[18] Z. Liu and M. Tegmark, "Machine learning conservation laws from trajectories," *Physical Review Letters*, vol. 126, no. 18, p. 180 604, 2021.

[19] S. Chen, P.-P. Dechant, Y.-H. He, E. Heyes, E. Hirst, and D. Riabchenko, "Machine learning clifford invariants of ade coxeter elements," *arXiv preprint arXiv:2310.00041*, vol. 34, no. 3, 2023. DOI: 10.1007/s00006-024-01325-y arXiv: 2310.00041 [cs.LG].

[20] H.-Y. Chen, Y.-H. He, S. Lal, and S. Majumder, "Machine learning lie structures & applications to physics," *Physics Letters B*, vol. 817, p. 136 297, 2021.

[21] S. Lal, S. Majumder, and E. Sobko, "The r-matrix net," *Machine Learning: Science and Technology*, vol. 5, no. 3, p. 035 003, 2024.

[22] R. Bondesan and A. Lamacraft, "Learning symmetries of classical integrable systems," *arXiv preprint arXiv:1906.04645*, 2019.

[23] S. Krippendorf, D. Lüst, and M. Syvaeri, "Integrability ex machina," *Fortschritte der Physik*, vol. 69, no. 7, p. 2 100 057, 2021.

[24] G. L. Cardoso, D. M. Peña, and S. Nampuri, "Classical integrability in the presence of a cosmological constant: Analytic and machine learning results," *arXiv preprint arXiv:2404.18247*, 2024.

[25] S. Lal, S. Majumder, and E. Sobko, "Deep learning based discovery of integrable systems," 2025. arXiv: 2503.10469 [hep-th]. [Online]. Available: https://arxiv.org/abs/2503.10469

[26] G. Carleo et al., "Machine learning and the physical sciences," *Rev. Mod. Phys.*, vol. 91, no. 4, p. 045 002, 2019. DOI: 10.1103/RevModPhys.91.045002 arXiv: 1903.10563 [physics.comp-ph].

[27] H. J. Ryser, *Combinatorial mathematics*. American Mathematical Soc., 1963, vol. 14.

[28] H. H. Hoos and T. Stutzle, "Stochastic local search," in *Handbook of approximation algorithms and metaheuristics*, Chapman and Hall/CRC, 2018, pp. 297–307.

[29] O. Ohlsson Sax, D. Riabchenko, and B. Stefański jr., "Worldsheet kinematics, dressing factors and odd crossing in mixed-flux AdS_3 backgrounds," *JHEP*, vol. 09, p. 132, 2024. DOI: 10.1007/JHEP09(2024)132 arXiv: 2312.09288 [hep-th].

[30] Y.-H. He, A. M. Kasprzyk, Q. Le, and D. Riabchenko, "Machine learning discovers new champion codes," *Research Square preprint*, 2025. [Online]. Available: https://doi.org/10.21203/rs.3.rs-7196769/v1

[31] N. Beisert et al., "Review of AdS/CFT integrability: An overview," *Lett. Math. Phys.*, vol. 99, no. 1-3, pp. 3–32, 2012. DOI: 10.1007/s11005-011-0529-2 arXiv: 1012.3982 [hep-th].

[32] T. Klose, "Review of AdS/CFT integrability, chapter IV.3: $N = 6$ chern-simons and strings on $AdS_4 \times CP^3$," *Lett. Math. Phys.*, vol. 99, no. 1-3, pp. 401–423, 2012. DOI: 10.1007/s11005-011-0520-y arXiv: 1012.3999 [hep-th].

[33] O. Aharony, S. S. Gubser, J. Maldacena, H. Ooguri, and Y. Oz, "Large n field theories, string theory and gravity," *Physics Reports*, vol. 323, no. 3-4, pp. 183–386, 2000. DOI: 10.1016/S0370-1573(99)00083-6 eprint: hep-th/9905111.

[34] M. Ammon and J. Erdmenger, *Gauge/gravity duality: Foundations and applications*. Cambridge University Press, 2015, ISBN: 9781107010345. DOI: 10.1017/CBO9780511846373

[35] R. G. Leigh, "Dirac-born-infeld action from dirichlet σ-model," *Modern Physics Letters A*, vol. 4, no. 28, pp. 2767–2772, 1989. DOI: 10.1142/S0217732389003099

[36] E. Witten, "Bound states of strings and p-branes," *Nuclear Physics B*, vol. 460, no. 2, pp. 335–350, 1996. DOI: 10.1016/0550-3213(95)00610-9

[37] G. T. Horowitz and A. Strominger, "Black strings and p-branes," *Nucl. Phys. B*, vol. 360, pp. 197–209, 1991. DOI: 10.1016/0550-3213(91)90440-9

[38] S. Demulder et al., "Exact approaches on the string worldsheet," *Journal of Physics A: Mathematical and Theoretical*, vol. 57, no. 42, p. 423 001, 2024. DOI: 10.1088/1751-8121/ad72be

[39] G. T. Horowitz, J. M. Maldacena, and A. Strominger, "Nonextremal black hole microstates and u-duality," *Physics Letters B*, vol. 383, no. 2, pp. 151–159, 1996. DOI: 10.1016/0370-2693(96)00738-1 [Online]. Available: https://www.sciencedirect.com/science/article/pii/0370269396007381

[40] T. Lloyd, O. Ohlsson Sax, A. Sfondrini, and B. Stefański Jr., "The complete worldsheet s matrix of superstrings on $AdS_3 \times S^3 \times T^4$ with mixed three-form flux," *Nucl. Phys. B*, vol. 891, pp. 570–612, 2015. DOI: 10.1016/j.nuclphysb.2014.12.019 arXiv: 1410.0866 [hep-th].

[41] J. Russo and A. A. Tseytlin, "Waves, boosted branes and bps states in m-theory," *Nuclear Physics B*, vol. 490, no. 1-2, pp. 121–144, 1997. DOI: 10.48550/arXiv.hep-th/9611047

[42] B. Hoare and A. Tseytlin, "On string theory on $AdS_3 \times S^3 \times T^4$ with mixed 3-form flux: Tree-level s-matrix," *Nuclear Physics B*, vol. 873, no. 3, pp. 682–727, 2013. DOI: 10.1016/j.nuclphysb.2013.05.005 arXiv: 1303.1037 [hep-th].

[43] J. Maldacena and A. Strominger, "Ads3 black holes and a stringy exclusion principle," *Journal of High Energy Physics*, vol. 1998, no. 12, p. 005, 1999. DOI: 10.1088/1126-6708/1998/12/005 arXiv: hep-th/9804085.

[44] J. Maldacena and H. Ooguri, "Strings in ads 3 and the sl (2, r) wzw model. i: The spectrum," *Journal of Mathematical Physics*, vol. 42, no. 7, pp. 2929–2960, 2001. DOI: 10.1063/1.1377273 arXiv: hep-th/0001053.

[45] D. Kutasov and N. Seiberg, "More comments on string theory on ads3," *Journal of High Energy Physics*, vol. 1999, no. 04, p. 008, 1999. DOI: 10.1088/1126-6708/1999/04/008 arXiv: hep-th/9903219.

[46] O. Aharony and E. Y. Urbach, "Type II string theory on $AdS_3 \times S^3 \times T^4$ and symmetric orbifolds," *Phys. Rev. D*, vol. 110, no. 4, p. 046 028, 2024. DOI: 10.1103/PhysRevD.110.046028 arXiv: 2406.14605 [hep-th].

[47] M. R. Gaberdiel and R. Gopakumar, "Tensionless string spectra on ads3," *Journal of High Energy Physics*, vol. 2018, no. 5, 2018. DOI: 10.1007/JHEP05(2018)085 arXiv: 1803.04423.

[48] L. Eberhardt, M. R. Gaberdiel, and R. Gopakumar, "The worldsheet dual of the symmetric product cft," *JHEP*, vol. 04, no. 4, p. 103, 2019. DOI: 10.1007/JHEP04(2019)103 arXiv: 1812.01007 [hep-th].

[49] F. Larsen and E. Martinec, "U (1) charges and moduli in the d1-d5 system," *Journal of High Energy Physics*, vol. 1999, no. 06, p. 019, 1999. DOI: 10.1088/1126-6708/1999/06/019 arXiv: hep-th/9905064.

[50] O. O. Sax and B. Stefański, "Closed strings and moduli in ads3/cft2," *Journal of High Energy Physics*, vol. 2018, no. 5, pp. 1–36, 2018. DOI: 10.1007/JHEP05(2018)101 arXiv: 1804.02023 [hep-th].

[51] M. R. Gaberdiel, R. Gopakumar, and B. Nairz, "Beyond the tensionless limit: Integrability in the symmetric orbifold," *arXiv e-prints*, arXiv:2312.13288, 2023, Art. no. arXiv:2312.13288. DOI: 10.48550/arXiv.2312.13288 arXiv: 2312.13288 [hep-th].

[52] G. 't Hooft, "A planar diagram theory for strong interactions," *Nucl. Phys. B*, vol. 72, no. 3, J. C. Taylor, Ed., p. 461, 1974. DOI: 10.1016/0550-3213(74)90154-0

[53] T. Bargheer, J. Caetano, T. Fleury, S. Komatsu, and P. Vieira, "Handling handles: Nonplanar integrability in n= 4 supersymmetric yang-mills theory," *Physical review letters*, vol. 121, no. 23, p. 231 602, 2018. DOI: 10.1103/PhysRevLett.121.231602 arXiv: 1711.05326 [hep-th].

[54] E. D'HOKER and D. Z. Freedman, "Supersymmetric gauge theories and the ads/cft correspondence," in *Strings, Branes and Extra Dimensions: TASI 2001*, World Scientific, 2004, pp. 3–159.

[55] N. Beisert, "Review of ads/cft integrability, chapter vi. 1: Superconformal symmetry," *Letters in Mathematical Physics*, vol. 99, no. 1, pp. 529–545, 2012. DOI: 10.1007/s11005-011-0479-8 arXiv: 1012.4004 [hep-th].

[56] P. West, *Introduction To Supersymmetry And Supergravity (Revised And Extended 2nd Edition)*. World Scientific Publishing Company, 1990.

[57] J. Wess and J. A. Bagger, *Supersymmetry and Supergravity: Revised Edition*. Princeton university press, 2020.

[58] F. Dolan and H. Osborn, "On short and semi-short representations for four-dimensional superconformal symmetry," *Annals of Physics*, vol. 307, no. 1, pp. 41–89, 2003. DOI: 10.1016/S0003-4916(03)00074-5 arXiv: hep-th/0209056.

[59] J. A. Minahan and K. Zarembo, "The bethe-ansatz for n=4 super yang-mills," *JHEP*, vol. 03, no. 03, p. 013, 2003. DOI: 10.1088/1126-6708/2003/03/013 arXiv: hep-th/0212208.

[60] N. Beisert, C. Kristjansen, and M. Staudacher, "The dilatation operator of conformal n=4 superyang-mills theory," *Nucl. Phys. B*, vol. 664, no. 1–2, pp. 131–184, 2003. DOI: 10.1016/S0550-3213(03)00406-1 arXiv: hep-th/0303060.

[61] N. Beisert and M. Staudacher, "The n=4 sym integrable super spin chain," *Nucl. Phys. B*, vol. 670, no. 3, pp. 439–463, 2003. DOI: 10.1016/j.nuclphysb.2003.08.015 arXiv: hep-th/0307042 [hep-th].

[62] N. Beisert and M. Staudacher, "Long-range psu(2,2—4) bethe ansatze for gauge theory and strings," *Nucl. Phys. B*, vol. 727, no. 1–2, pp. 1–62, 2005. DOI: 10.1016/j.nuclphysb.2005.06.038 arXiv: hep-th/0504190.

[63] H. Bethe, "On the theory of metals. 1. eigenvalues and eigenfunctions for the linear atomic chain," *Z. Phys.*, vol. 71, pp. 205–226, 1931. DOI: 10.1007/BF01341708

[64] M. Staudacher, "The factorized s-matrix of cft/ads," *JHEP*, vol. 05, no. 05, p. 054, 2005. DOI: 10.1088/1126-6708/2005/05/054 arXiv: hep-th/0412188.

[65] R. R. Metsaev and A. A. Tseytlin, "Type iib superstring action in $AdS_5 \times S^5$ background," *Nucl. Phys. B*, vol. 533, no. 1-3, pp. 109–126, 1998. DOI: 10.1016/S0550-3213(98)00570-7 arXiv: hep-th/9805028.

[66] H. Eichenherr and M. Forger, "On the dual symmetry of the non-linear sigma models," *Nuclear Physics B*, vol. 155, no. 2, pp. 381–393, 1979.

[67] I. Bena, J. Polchinski, and R. Roiban, "Hidden symmetries of the $AdS_5 \times S^5$ superstring," *Phys. Rev. D*, vol. 69, no. 4, p. 046 002, 2004. DOI: 10.1103/PhysRevD.69.046002 arXiv: hep-th/0305116.

[68] N. Berkovits, M. Bershadsky, T. Hauer, S. Zhukov, and B. Zwiebach, "Superstring theory on $AdS_2 \times S^2$ as a coset supermanifold," *Nucl. Phys. B*, vol. 567, no. 1–2, pp. 61–86, 2000. DOI: 10.1016/S0550-3213(99)00683-5 arXiv: hep-th/9907200.

[69] N. Beisert, "The su(2—2) dynamic s-matrix," *Adv. Theor. Math. Phys.*, vol. 12, no. 5, pp. 945–979, 2008. DOI: 10.4310/ATMP.2008.v12.n5.a1 arXiv: hep-th/0511082.

[70] G. Arutyunov, S. Frolov, and M. Zamaklar, "The zamolodchikov–faddeev algebra for $AdS_5 \times S^5$ superstring," *JHEP*, vol. 04, no. 04, p. 002, 2007. DOI: 10.1088/1126-6708/2007/04/002 arXiv: hep-th/0612229.

[71] A. B. Zamolodchikov and A. B. Zamolodchikov, "Factorized s-matrices in two dimensions as the exact solutions of certain relativistic quantum field theory models," *Annals Phys.*, vol. 120, no. 2, I. M. Khalatnikov and V. P. Mineev, Eds., pp. 253–291, 1979. DOI: 10.1016/0003-4916(79)90391-9

[72] G. Arutyunov, S. Frolov, J. Russo, and A. A. Tseytlin, "Spinning strings in $AdS_5 \times S^5$ and integrable systems," *Nucl. Phys. B*, vol. 671, pp. 3–50, 2003. DOI: 10.1016/j.nuclphysb.2003.08.036 arXiv: hep-th/0307191.

[73] N. Beisert, B. Eden, and M. Staudacher, "Transcendentality and crossing," *J. Stat. Mech.*, vol. 0701, no. 01, P01021, 2007. DOI: 10.1088/1742-5468/2007/01/P01021 arXiv: hep-th/0610251.

[74] B. Eden and M. Staudacher, "Integrability and transcendentality," *J. Stat. Mech.*, vol. 0611, no. 11, P11014, 2006. DOI: 10.1088/1742-5468/2006/11/P11014 arXiv: hep-th/0603157.

[75] G. Arutyunov and S. Frolov, "Integrable hamiltonian for 'classical strings on $AdS_5 \times S_5$," *JHEP*, vol. 02, no. 02, p. 059, 2005. DOI: 10.1088/1126-6708/2005/02/059 arXiv: hep-th/0411089.

[76] S. Frolov, J. Plefka, and M. Zamaklar, "The $AdS_5 \times S^5$ superstring in light-cone gauge and its bethe equations," *J. Phys. A*, vol. 39, no. 41, pp. 13 037–13 082, 2006. DOI: 10.1088/0305-4470/39/41/S15 arXiv: hep-th/0603008.

[77] G. Arutyunov, S. Frolov, J. Plefka, and M. Zamaklar, "The off-shell symmetry algebra of the light-cone $AdS_5 \times S^5$ superstring," *J. Phys. A*, vol. 40, no. 13, pp. 3583–3606, 2007. DOI: 10.1088/1751-8113/40/13/018 arXiv: hep-th/0609157.

[78] G. Arutyunov and S. Frolov, "On string s-matrix, bound states and tba," *JHEP*, vol. 12, no. 12, p. 024, 2007. DOI: 10.1088/1126-6708/2007/12/024 arXiv: 0710.1568 [hep-th].

[79] J. Ambjørn, R. A. Janik, and C. Kristjansen, "Wrapping interactions and a new source of corrections to the spin-chain/string duality," *Nuclear Physics B*, vol. 736, no. 3, pp. 288–301, 2006. DOI: 10.1016/j.nuclphysb.2005.12.007

[80] M. Luscher, "Volume dependence of the energy spectrum in massive quantum field theories. 1. stable particle states," *Commun. Math. Phys.*, vol. 104, no. 2, p. 177, 1986. DOI: 10.1007/BF01211589

[81] M. Luscher, "Volume dependence of the energy spectrum in massive quantum field theories. 2. scattering states," *Commun. Math. Phys.*, vol. 105, pp. 153–188, 1986. DOI: 10.1007/BF01211097

[82] M. C. Abbott and I. Aniceto, "Massless lüscher terms and the limitations of the AdS_3 asymptotic bethe ansatz," *Phys. Rev. D*, vol. 93, no. 10, p. 106 006, 2016. DOI: 10.1103/PhysRevD.93.106006 arXiv: 1512.08761 [hep-th].

[83] R. A. Janik, "Review of ads/cft integrability, chapter iii.5: Lüscher corrections," *Lett. Math. Phys.*, vol. 99, no. 1–3, pp. 277–297, 2012. DOI: 10.1007/s11005-011-0511-z arXiv: 1012.3994 [hep-th].

[84] A. B. Zamolodchikov, "Thermodynamic bethe ansatz in relativistic models. scaling three state potts and lee-yang models," *Nucl. Phys. B*, vol. 342, no. 3, pp. 695–720, 1990. DOI: 10.1016/0550-3213(90)90333-9

[85] N. Gromov, V. Kazakov, S. Leurent, and D. Volin, "Quantum spectral curve for planar $\mathcal{N} = 4$ super-yang-mills theory," *Phys. Rev. Lett.*, vol. 112, no. 1, p. 011 602, 2014. DOI: 10.1103/PhysRevLett.112.011602 arXiv: 1305.1939 [hep-th].

[86] A. Cavaglia, N. Gromov, B. Stefanski Jr., and A. Torrielli, "Quantum spectral curve for ads_3/cft_2: A proposal," *JHEP*, vol. 12, p. 048, 2021. DOI: 10.1007/JHEP12(2021)048 arXiv: 2109.05500 [hep-th].

[87] G. Arutyunov and S. Frolov, "Foundations of the $AdS_5 \times S^5$ superstring: I," *Journal of Physics A: Mathematical and Theoretical*, vol. 42, no. 25, p. 254 003, 2009. DOI: 10.1088/1751-8113/42/25/254003 arXiv: 0901.4937 [hep-th].

[88] N. Beisert, "The analytic bethe ansatz for a chain with centrally extended $\mathfrak{su}(2|2)$ symmetry," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2007, no. 01, P01017, 2007. DOI: 10.48550/arXiv.nlin/0610017 arXiv: nlin/0610017 [nlin.SI].

[89] G. Arutyunov, S. Frolov, and M. Staudacher, "Bethe ansatz for quantum strings," *JHEP*, vol. 10, no. 10, p. 016, 2004. DOI: 10.1088/1126-6708/2004/10/016 arXiv: hep-th/0406256.

[90] R. A. Janik, "The $AdS_5 \times S^5$ superstring worldsheet s-matrix and crossing symmetry," *Phys. Rev. D*, vol. 73, no. 8, p. 086 006, 2006. DOI: 10.1103/PhysRevD.73.086006 arXiv: hep-th/0603038.

[91] J. Plefka, F. Spill, and A. Torrielli, "On the hopf algebra structure of the AdS/CFT s-matrix," *Phys. Rev. D*, vol. 74, no. 6, p. 066 008, 2006. DOI: 10.1103/PhysRevD.74.066008 arXiv: hep-th/0608038.

[92] N. Beisert and T. Klose, "Long-range gl(n) integrable spin chains and plane-wave matrix theory," *J. Stat. Mech.*, vol. 0607, no. 07, P07006, 2006. DOI: 10.1088/1742-5468/2006/07/P07006 arXiv: hep-th/0510124.

[93] G. Arutyunov and S. Frolov, "On $AdS_5 \times S^5$ string s-matrix," *Phys. Lett. B*, vol. 639, no. 3–4, pp. 378–382, 2006. DOI: 10.1016/j.physletb.2006.06.064 arXiv: hep-th/0604043.

[94] R. Hernandez and E. Lopez, "Quantum corrections to the string bethe ansatz," *JHEP*, vol. 07, no. 07, p. 004, 2006. DOI: 10.1088/1126-6708/2006/07/004 arXiv: hep-th/0603204.

[95] N. Beisert, R. Hernandez, and E. Lopez, "A crossing-symmetric phase for $AdS_5 \times S_5$ strings," *JHEP*, vol. 11, no. 11, p. 070, 2006. DOI: 10.1088/1126-6708/2006/11/070 arXiv: hep-th/0609044.

[96] R. Roiban, A. Tirziu, and A. A. Tseytlin, "Two-loop world-sheet corrections in $AdS_5 \times S_5$ superstring," *JHEP*, vol. 07, no. 07, p. 056, 2007. DOI: 10.1088/1126-6708/2007/07/056 arXiv: 0704.3638 [hep-th].

[97] T. Klose, T. McLoughlin, J. A. Minahan, and K. Zarembo, "World-sheet scattering in $AdS_5 \times S_5$ at two loops," *JHEP*, vol. 08, no. 08, p. 051, 2007. DOI: 10.1088/1126-6708/2007/08/051 arXiv: 0704.3891 [hep-th].

[98] G. Arutyunov and S. Frolov, "The dressing factor and crossing equations," *J. Phys. A*, vol. 42, no. 42, p. 425 401, 2009. DOI: 10.1088/1751-8113/42/42/425401 arXiv: 0904.4575 [hep-th].

[99] N. Dorey, D. M. Hofman, and J. M. Maldacena, "On the singularities of the magnon s-matrix," *Phys. Rev. D*, vol. 76, p. 025 011, 2 2007. DOI: 10.1103/PhysRevD.76.025011 arXiv: hep-th/0703104.

[100] D. Volin, "Minimal solution of the ads/cft crossing equation," *J. Phys. A*, vol. 42, no. 37, p. 372 001, 2009. DOI: 10.1088/1751-8113/42/37/372001 arXiv: 0904.4929 [hep-th].

[101] A. Sfondrini, "Towards integrability for AdS_3/CFT_2," *Journal of Physics A: Mathematical and Theoretical*, vol. 48, no. 2, p. 023 001, 2014. DOI: https://doi.org/10.1088/1751-8113/48/2/023001 arXiv: 1406.2971 [hep-th].

[102] A. Babichenko, B. Stefanski Jr., and K. Zarembo, "Integrability and the AdS_3/CFT_2 correspondence," *JHEP*, vol. 03, no. 3, p. 058, 2010. DOI: 10.1007/JHEP03(2010)058 arXiv: 0912.1723 [hep-th].

[103] O. Ohlsson Sax and B. Stefanski Jr., "Integrability, spin-chains and the AdS3/CFT2 correspondence," *JHEP*, vol. 08, p. 029, 2011. DOI: 10.1007/JHEP08(2011)029 arXiv: 1106.2558 [hep-th].

[104] J. R. David and B. Sahoo, "Giant magnons in the d1-d5 system," *Journal of High Energy Physics*, vol. 2008, no. 07, p. 033, 2008. DOI: 10.1088/1126-6708/2008/07/033 arXiv: 0804.3267 [hep-th].

[105] A. Cagnazzo and K. Zarembo, "B-field in AdS_3/CFT_2 correspondence and integrability," *JHEP*, vol. 11, no. 11, p. 133, 2012, [Erratum: JHEP 04, 003 (2013)]. DOI: 10.1007/JHEP11(2012)133 arXiv: 1209.4049 [hep-th].

[106] P. Sundin and L. Wulff, "Classical integrability and quantum aspects of the $AdS_3 \times S^3 \times S^3 \times S^1$ superstring," *JHEP*, vol. 10, no. 10, p. 109, 2012. DOI: 10.1007/JHEP10(2012)109 arXiv: 1207.5531 [hep-th].

[107] M. B. Green and J. H. Schwarz, "Covariant description of superstrings," *Physics Letters B*, vol. 136, no. 5-6, pp. 367–370, 1984.

[108] J. Rahmfeld and A. Rajaraman, "Green-schwarz string action on $AdS_3 \times S^3$ with ramond-ramond charge," *Physical Review D*, vol. 60, no. 6, p. 064 014, 1999. DOI: 10.1103/PhysRevD.60.064014 arXiv: hep-th/9809164.

[109] J. Park and S.-J. Rey, "Green-schwarz superstring on $AdS_3 \times S^3$," *Journal of High Energy Physics*, vol. 1999, no. 01, p. 001, 1999. DOI: 10.1088/1126-6708/1999/01/001 arXiv: hep-th/9812062.

[110] R. Borsato, O. Ohlsson Sax, A. Sfondrini, and B. Stefanski, "The complete ads3 ×s3 × t4 worldsheet s matrix," *JHEP*, vol. 10, no. 10, p. 066, 2014. DOI: 10.1007/JHEP10(2014)066 arXiv: 1406.0453 [hep-th].

[111] R. Borsato, O. Ohlsson Sax, and A. Sfondrini, "A dynamic $su(1|1)^2$ s-matrix for ads3/cft2," *JHEP*, vol. 04, no. 4, p. 113, 2013. DOI: 10.1007/JHEP04(2013)113 arXiv: 1211.5119 [hep-th].

[112] R. Borsato, O. O. Sax, A. Sfondrini, B. Stefanski, and A. Torrielli, "The all-loop integrable spin-chain for strings on $AdS_3 \times S^3 \times T^4$: The massive sector," *Journal of High Energy Physics*, vol. 2013, no. 8, pp. 1–43, 2013. DOI: 10.1007/JHEP08(2013)043 arXiv: 1303.5995 [hep-th].

[113] R. Borsato, O. Ohlsson Sax, A. Sfondrini, and B. Stefanski, "Towards the all-loop worldsheet s-matrix for $AdS_3S^3T^4$," *Phys. Rev. Lett.*, vol. 113, no. 13, p. 131 601, 2014. DOI: 10.1103/PhysRevLett.113.131601 arXiv: 1403.4543 [hep-th].

[114] P. Dorey, "Exact s-matrices," in *Conformal Field Theories and Integrable Models: Lectures Held at the Eötvös Graduate Course, Budapest, Hungary, 13–18 August 1996*, Springer, 2007, pp. 85–125. DOI: 10.48550/arxiv.hep-th/9810026 arXiv: hep-th/9810026.

[115] L. Castillejo, R. Dalitz, and F. Dyson, "Low's scattering equation for the charged and neutral scalar theories," *Physical Review*, vol. 101, no. 1, p. 453, 1956.

[116] R. Borsato, O. Ohlsson Sax, A. Sfondrini, B. Stefanski Jr., and A. Torrielli, "Dressing phases of AdS_3/CFT_2," *Phys. Rev. D*, vol. 88, p. 066 004, 2013. DOI: 10.1103/PhysRevD.88.066004 arXiv: 1306.2512 [hep-th].

[117] S. Frolov and A. Sfondrini, "New dressing factors for ads3/cft2," *JHEP*, vol. 04, no. 4, p. 162, 2022. DOI: 10.1007/JHEP04(2022)162 arXiv: 2112.08896 [hep-th].

[118] S. Frolov, D. Polvara, and A. Sfondrini, "On mixed-flux worldsheet scattering in AdS_3/CFT_2," *JHEP*, vol. 11, p. 055, 2023. DOI: 10.1007/JHEP11(2023)055 arXiv: 2306.17553 [hep-th].

[119] M. Beccaria, F. Levkovich-Maslyuk, G. Macorini, and A. A. Tseytlin, "Quantum corrections to spinning superstrings in ads3 × s3 × m 4: Determining the dressing phase," *JHEP*, vol. 04, no. 4, p. 006, 2013. DOI: 10.1007/JHEP04(2013)006 arXiv: 1211.6090 [hep-th].

[120] S. Frolov, D. Polvara, and A. Sfondrini, "Dressing factors for mixed-flux $AdS_3 \times S^3 \times T^4$ superstrings," 2024. DOI: 10.48550/arxiv.2402.11732 arXiv: 2402.11732 [hep-th].

[121] S. Ekhammar and D. Volin, "Monodromy bootstrap for su(2—2) quantum spectral curves: From hubbard model to ads_3/cft_2," *JHEP*, vol. 03, p. 192, 2022. DOI: 10.1007/JHEP03(2022)192 arXiv: 2109.06164 [math-ph].

[122] S. Frolov, D. Polvara, and A. Sfondrini, "Massive dressing factors for mixed-flux AdS_3/CFT_2," *JHEP*, vol. 07, p. 171, 2025. DOI: 10.1007/JHEP07(2025)171 arXiv: 2501.05995 [hep-th].

[123] S. Frolov, D. Polvara, and A. Sfondrini, "Dressing Factors and Mirror Thermodynamic Bethe Ansatz for mixed-flux AdS3/CFT2," 2025. DOI: 10.48550/arxiv.2507.12191 arXiv: 2507.12191 [hep-th].

[124] F. K. Seibold and A. Sfondrini, "Ads3 integrability, tensionless limits, and deformations: A review," *arXiv preprint arXiv:2408.08414*, 2024. DOI: https://arxiv.org/abs/2408.08414

[125] Y.-H. He, "Ai-driven research in pure mathematics and theoretical physics," *Nature Reviews Physics*, vol. 6, no. 9, pp. 546–553, 2024.

[126] M. R. Douglas and K.-H. Lee, "Mathematical data science," *arXiv preprint arXiv:2502.08620*, 2025.

[127] A. Davies et al., "Advancing mathematics by guiding human intuition with ai," *Nature*, vol. 600, no. 7887, pp. 70–74, 2021.

[128] Y.-H. He, "From the string landscape to the mathematical landscape: A machine-learning outlook," in *International Workshop on Lie Theory and Its Applications in Physics*, Springer, 2021, pp. 21–31.

[129] N. A. Barricelli et al., "Symbiogenetic evolution processes realized by artificial methods," *Methodos*, vol. 9, no. 35-36, pp. 143–182, 1957.

[130] A. S. Fraser, "Simulation of genetic systems by automatic digital computers i. introduction," *Australian journal of biological sciences*, vol. 10, no. 4, pp. 484–491, 1957.

[131] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison Wesley series in artificial intelligence). Addison-Wesley, 1989, ISBN: 9780201157673.

[132] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[133] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia tools and applications*, vol. 80, pp. 8091–8126, 2021.

[134] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[135] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4.

[136] P. Mehta et al., "A high-bias, low-variance introduction to machine learning for physicists," *Physics Reports*, vol. 810, pp. 1–124, 2019. DOI: https://doi.org/10.1016/j.physrep.2019.03.001

[137] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[138] K. Hornik, M. B. Stinchcombe, and H. L. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. DOI: https://doi.org/10.1016/0893-6080(89)90020-8 [Online]. Available: https://www.sciencedirect.com/science/article/pii/0893608089900208

[139] B. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964. DOI: https://doi.org/10.1016/0041-5553(64)90137-5 [Online]. Available: https://www.sciencedirect.com/science/article/pii/0041555364901375

[140] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[141] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[142] S. Liang and R. Srikant, "Why deep neural networks for function approximation?" *arXiv preprint arXiv:1610.04161*, 2016.

[143] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735

[144] I. Sutskever, O. Vinyals, and Q. V. Le, *Sequence to sequence learning with neural networks*, 2014. arXiv: 1409.3215 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1409.3215

[145] D. Bahdanau, K. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate*, 2016. arXiv: 1409.0473 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1409.0473

[146] K. Cho et al., *Learning phrase representations using rnn encoder-decoder for statistical machine translation*, 2014. arXiv: 1406.1078 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1406.1078

[147] A. Vaswani et al., "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[148] T. Brown et al., "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[149] F. Barbieri, J. Camacho-Collados, L. Neves, and L. Espinosa-Anke, "Tweeteval: Unified benchmark and comparative evaluation for tweet classification," *arXiv preprint arXiv:2010.12421*, 2020.

[150] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1512.03385

[151] Y. Tang and Y. Yang, "Pooling and attention: What are effective designs for llm-based embedding models?" *arXiv preprint arXiv:2409.02727*, 2024.

[152] D. Hestenes and G. Sobczyk, *Clifford algebra to geometric calculus: a unified language for mathematics and physics*. Springer Science & Business Media, 1987, vol. 5.

[153] A. Lasenby, J. Lasenby, and C. Matsantonis, "Reconstructing a rotor from initial and final frames using characteristic multivectors: With applications in orthogonal transformations," *Mathematical Methods in the Applied Sciences*, 2022.

[154] A. Lasenby, "Some recent results for $SU(3)$ and octonions within the Geometric Algebra approach to the fundamental forces of nature," *Mathematical Methods in the Applied Sciences*, 2022.

[155] D. Shirokov, "On computing the determinant, other characteristic polynomial coefficients, and inverse in Clifford algebras of arbitrary dimension," *Computational and Applied Mathematics*, vol. 40, no. 5, pp. 1–29, 2021.

[156] K. Abdulkhaev and D. Shirokov, "On explicit formulas for characteristic polynomial coefficients in Geometric Algebras," in *Computer Graphics International Conference*, Springer, 2021, pp. 670–681.

[157] C. Doran and A. Lasenby, *Geometric algebra for physicists*. Cambridge University Press, 2003.

[158] P.-P. Dechant, "Clifford algebra is the natural framework for root systems and coxeter groups. group theory: Coxeter, conformal and modular groups," *Advances in Applied Clifford Algebras*, vol. 27, pp. 17–31, 2015. [Online]. Available: `http://dx.doi.org/10.1007/s00006-015-0584-3`

[159] J. E. Humphreys, *Reflection groups and Coxeter groups*. Cambridge University Press, Cambridge, 1990.

[160] P.-P. Dechant, Y.-H. He, E. Heyes, and E. Hirst, "Cluster algebras: Network science and machine learning," *accepted in Journal of Computational Algebra; arXiv preprint arXiv:2203.13847*, 2022.

[161] M.-W. Cheung, P.-P. Dechant, Y.-H. He, E. Heyes, E. Hirst, and J.-R. Li, "Clustering cluster algebras with clusters," *Accepted in Advances in Theoretical and Mathematical Physics; arXiv preprint arXiv:2212.09771*, 2022.

[162] Y.-H. He, E. Heyes, and E. Hirst, "Machine learning in physics and geometry," *arXiv preprint arXiv:2303.12626*, 2023.

[163] Y.-H. He, R.-K. Seong, and S.-T. Yau, "Calabi–Yau volumes and reflexive polytopes," *Communications in Mathematical Physics*, vol. 361, pp. 155–204, 2018.

[164] Y.-H. He, K.-H. Lee, T. Oliver, and A. Pozdnyakov, "Murmurations of elliptic curves," *arXiv preprint arXiv:2204.10140*, 2022.

[165] Y.-H. He, *The Calabi–Yau landscape: From geometry, to physics, to machine learning*. Springer Nature, 2021, vol. 2293.

[166] A. Bromborsky, U. Song, E. Wieser, H. Hadfield, and The Pygae Team, "Pygae/galgebra: V0.5.0," *Zenodo*, 2020. DOI: `10.5281/zenodo.3875882`

[167] P.-P. Dechant, "The birth of E_8 out of the spinors of the icosahedron," *Proceedings of the Royal Society A 20150504*, 2016. [Online]. Available: `http://dx.doi.org/10.1098/rspa.2015.0504`

[168] J. C. Baez, "From the icosahedron to E_8," *arXiv preprint arXiv:1712.06436*, 2017.

[169] P. Cameron, P.-P. Dechant, Y.-H. He, and J. McKay, *ADE - patterns in Mathematics*. Cambridge University Press, 2024.

[170] P.-P. Dechant, C. Bœhm, and R. Twarock, "Affine extensions of non-crystallographic Coxeter groups induced by projection," *Journal of Mathematical Physics*, vol. 54, no. 9, 2013.

[171] P.-P. Dechant, "The E_8 geometry from a Clifford perspective," *Advances in Applied Clifford Algebras*, vol. 27, no. 1, pp. 397–421, 2017.

[172] O. Perron, "Zur Theorie der Matrices," *Mathematische Annalen*, vol. 64, no. 2, pp. 248–263, 1907. DOI: 10.1007/BF01449896 [Online]. Available: https://doi.org/10.1007/BF01449896

[173] G. Frobenius, *Über Matrizen aus nicht negativen Elementen* (Preussische Akademie der Wissenschaften Berlin: Sitzungsberichte der Preußischen Akademie der Wissenschaften zu Berlin). Reichsdr., 1912, ISBN: 9783111271903. [Online]. Available: https://books.google.co.uk/books?id=fuK3PgAACAAJ

[174] J. H. Smith, "Some properties of the spectrum of a graph," *Combinatorial Structures and their applications*, pp. 403–406, 1970.

[175] N. J. A. Sloane, *The On-Line Encyclopedia of Integer Sequences, A001349*, Number of connected graphs with n nodes. [Online]. Available: https://oeis.org/A001349

[176] P. Bonacich, "Power and centrality: A family of measures," *American Journal of Sociology*, vol. 92, no. 5, pp. 1170–1182, 1987. Accessed: Jul. 24, 2023. [Online]. Available: http://www.jstor.org/stable/2780000

[177] J. A. Anderson, *An introduction to neural networks*. MIT press, 1995.

[178] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International conference on machine learning*, PMLR, 2017, pp. 3319–3328.

[179] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, 1065 2016. DOI: doi.org/10.1098/rsta.2015.0202

[180] R. Kadel, K. Paudel, D. B. Guruge, and S. J. Halder, "Opportunities and challenges for error control schemes for wireless sensor networks: A review," *Electronics*, vol. 9, no. 3, p. 504, 2020.

[181] J. Pamart et al., "Forward error correction in a 5 gbit/s 6400 km edfa based system," *Electronics letters*, vol. 30, no. 4, pp. 342–343, 1994.

[182] R. J. McEliece and L. Swanson, "Reed-solomon codes and the exploration of the solar system," 1994.

[183] J. Heller and I. Jacobs, "Viterbi decoding for satellite and space communication," *IEEE Transactions on Communication Technology*, vol. 19, no. 5, pp. 835–848, 1971.

[184] H. Fredricksen, "Error correction for deep space network teletype circuits," Tech. Rep., 1968.

[185] J. Little and H. Schenck, "Toric surface codes and minkowski sums," *SIAM Journal on Discrete Mathematics*, vol. 20, no. 4, pp. 999–1014, 2006.

[186] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Transactions on Information Theory*, vol. 43, no. 6, pp. 1757–1766, 1997.

[187] J. P. Hansen, "Toric varieties hirzebruch surfaces and error-correcting codes," *Applicable Algebra in Engineering, Communication and Computing*, vol. 13, pp. 109–117, 2013. DOI: 10.1007/s00200-002-0106-0

[188] D. Joyner, "Toric codes over finite fields," *Applicable Algebra in Engineering, Communication and Computing*, vol. 15, pp. 63–79, 2004. DOI: 10.1007/s00200-004-0152-x

[189] M. Grassl, "Searching for linear codes with large minimum distance," in *Discovering Mathematics with Magma: Reducing the Abstract to the Concrete*, Springer, 2006, pp. 287–313.

[190] I. Soprunov and J. Soprunova, "Toric surface codes and minkowski length of polygons," *SIAM Journal on Discrete Mathematics*, vol. 23, no. 1, pp. 384–400, 2009. DOI: 10.1137/080716554 [Online]. Available: http://dx.doi.org/10.1137/080716554

[191] D. Ruano, "On the structure of generalized toric codes," *Journal of Symbolic Computation*, vol. 44, pp. 499–506, 2009. DOI: 10.1016/j.jsc.2007.07.018

[192] J. B. Little, "Remarks on generalized toric codes," *Finite Fields and Their Applications*, vol. 24, no. 1, pp. 1–14, 2013. DOI: 10.1016/j.ffa.2013.05.004

[193] G. Brown and A. M. Kasprzyk, "Seven new champion linear codes," *LMS Journal of Computation and Mathematics*, vol. 16, pp. 109–117, 2013. DOI: 10.1112/s1461157013000041 [Online]. Available: http://dx.doi.org/10.1112/S1461157013000041

[194] A. Davies et al., "Advancing mathematics by guiding human intuition with AI," *Nature*, vol. 600, pp. 70–74, 2021. DOI: 10.1038/s41586-021-04086-x

[195] T. Coates, A. M. Kasprzyk, and S. Veneziale, "Machine learning the dimension of a Fano variety," *Nat. Commun.*, vol. 14, no. 5526, 2023. DOI: 10.1038/s41467-023-41157-1

[196] S. Gukov, J. Halverson, and F. Ruehle, "Rigor with machine learning from field theory to the Poincaré conjecture," *Nature Rev. Phys.*, vol. 6, no. 5, pp. 310–319, 2024. DOI: 10.1038/s42254-024-00709-0 arXiv: 2402.13321 [hep-th].

[197] M. Yau, N. Karalias, E. Lu, J. Xu, and S. Jegelka, "Are graph neural networks optimal approximation algorithms?" *Advances in Neural Information Processing Systems*, vol. 37, pp. 73 124–73 181, 2024.

[198] P. Berglund, Y.-H. He, E. Heyes, E. Hirst, V. Jejjala, and A. Lukas, "New calabi–yau manifolds from genetic algorithms," *Physics Letters B*, vol. 850, p. 138 504, 2024.

[199] J. Abramson et al., "Accurate structure prediction of biomolecular interactions with alphafold 3," *Nature*, vol. 630, no. 8016, pp. 493–500, 2024. DOI: 10.1038/s41586-024-07487-w

[200] L. Ardon, "Reinforcement learning to solve np-hard problems: An application to the cvrp," *arXiv preprint arXiv:2201.05393*, 2022.

[201] R. Hill, *A First Course In Coding Theory*. Oxford: Clarendon Press, 1988.

[202] M. Grassl, *Bounds on the minimum distance of linear codes and quantum codes*, Online available at https://www.codetables.de. Accessed on 2025-04-10.

[203] K.-H. Zimmermann, "Integral hecke modules, integral generalized reed-muller codes, and linear codes," Techn. Univ. Hamburg-Harburg, Tech. Rep., 1996.

[204] F. Hernando, F. D. Igual, and G. Quintana-Ortí, "Algorithm 994: Fast implementations of the brouwer-zimmermann algorithm for the computation of the minimum distance of a random linear code," *ACM Transactions on Mathematical Software (TOMS)*, vol. 45, no. 2, pp. 1–28, 2019.

[205] J. P. Hansen, "Toric surfaces and error-correcting codes," in *Coding Theory, Cryptography and Related Areas: Proceedings of an International Conference on Coding Theory, Cryptography and Related Areas, held in Guanajuato, Mexico, in April 1998*, Springer, 2000, pp. 132–142.

[206] A. Y. Kitaev, "Quantum communication, computing, and measurement," in *Proceedings of the 3rd International Conference of Quantum Communication and Measurement, New York: Plenum*, 1997.

[207] J. P. Hansen, "Toric varieties hirzebruch surfaces and error-correcting codes," *Applicable Algebra in Engineering, Communication and Computing*, vol. 13, pp. 289–300, 2002. DOI: 10.1007/s00200-002-0106-0

[208] P. Andreasson, J. Johansson, S. Liljestrand, and M. Granath, "Quantum error correction for the toric code using deep reinforcement learning," *Quantum*, vol. 3, p. 183, 2019. DOI: 10.22331/q-2019-09-02-183 [Online]. Available: https://quantum-journal.org/papers/q-2019-09-02-183/

[209] S. Varona and M. A. Martin-Delgado, "Determination of the semion code threshold using neural decoders," *Physical Review A*, vol. 102, no. 3, p. 032 411, 2020. DOI: 10.1103/PhysRevA.102.032411 [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.102.032411

[210] Z.-A. Jia, Y.-H. Zhang, Y.-C. Wu, L. Kong, G.-C. Guo, and G.-P. Guo, "Efficient machine learning representations of surface code with boundaries, defects, domain walls and twists," *arXiv preprint arXiv:1802.03738*, 2018. [Online]. Available: https://arxiv.org/abs/1802.03738

[211] The Sage Developers, *Sagemath, the Sage Mathematics Software System (Version 10.6)*, https://www.sagemath.org, 2025.

[212] W. Bosma, J. Cannon, and C. Playoust, "The Magma algebra system. I. The user language," *J. Symbolic Comput.*, vol. 24, no. 3-4, pp. 235–265, 1997, Computational algebra and number theory (London, 1993). DOI: 10.1006/jsco.1996.0125 [Online]. Available: http://dx.doi.org/10.1006/jsco.1996.0125

[213] A. Karpathy, *Mingpt*, https://github.com/karpathy/minGPT, 2022.

[214] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[215] Q. Le and D. Riabchenko, *Genetic algorithms*, https://github.com/QTVLe/MachineLearningGeneralisedToricCodes, 2025.

[216] A. F. Gad, "PyGAD: An intuitive genetic algorithm Python library," *Multimed. Tools Appl.*, vol. 83, pp. 58 029–58 042, 2024. DOI: 10.1007/s11042-023-17167-y

[217] A. Hocquenghem, "Codes correcteurs d'erreurs," *Chiffers*, vol. 2, pp. 147–156, 1959.

[218] R. Bose and D. Ray-Chaudhuri, "On a class of error correcting binary group codes," eng, *Information and control*, vol. 3, no. 1, pp. 68–79, 1960.

[219] D. E. Muller, "Application of boolean algebra to switching circuit design and to error detection," eng, *Transactions of the I.R.E. Professional Group on Electronic Computers*, vol. EC-3, no. 3, pp. 6–12, 1954.

[220] I. Reed, "A class of multiple-error-correcting codes and the decoding scheme," eng, *Transactions of the I.R.E. Professional Group on Information Theory*, vol. 4, no. 4, pp. 38–49, 1954.

[221] A. Couvreur and H. Randriambololona, "Algebraic geometry codes and some applications," in *Concise encyclopedia of coding theory*, CRC Press, Boca Raton, FL, 2021, pp. 307–361.

[222] V. D. Goppa, "Algebraico–geometric codes," *Mathematics of the USSR-Izvestiya*, vol. 21, no. 1, pp. 75–91, 1983. DOI: 10.1070/IM1983v021n01ABEH001641