ABSTRACT

The Landscape of Free Fermionic Gauge Models

Douglas G. Moore, Ph.D.

Advisor: Gerald B. Cleaver, Ph.D.

A software framework is developed to systematically construct a particular class of weakly coupled free fermionic heterotic string models, dubbed gauge models. In their purest form, these models are maximally supersymmetric (N = 4), and thus only contain superpartners in their matter sector. This feature makes their systematic construction particularly efficient, and they are thus useful in their simplicity. We first provide a brisk introduction to heterotic strings and the spin-structure construction of free fermionic models. Three systematic surveys are then presented, and we conjecture that these surveys are exhaustive modulo redundancies. Finally we present a collection of metaheuristic algorithms for searching the landscape for models with a user-specified spectrum of phenomenological properties, e.g. gauge group and number of spacetime supersymmetries. Such algorithms provide the groundwork for extended generic free fermionic surveys.

The Landscape of Free Fermionic Gauge Models

by

Douglas G. Moore, B.S.

A Dissertation

Approved by the Department of Physics

_____

Gregory A. Benesh, Ph.D., Chairperson

Submitted to the Graduate Faculty of
Baylor University in Partial Fulfillment of the
Requirements for the Degree
of

Doctor of Philosophy

Approved by the Dissertation Committee

_____

Gerald B. Cleaver, Ph.D., Chairperson

_____

Jay R. Dittmann, Ph.D.

_____

Lorin S. Matthews, Ph.D.

_____

Brian Raines, Ph.D.

_____

Anzhong Wang, Ph.D.

Accepted by the Graduate School

May 2014

_____

J. Larry Lyon, Ph.D., Dean

Page bearing signatures is kept on file in the Graduate School.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

PREFACE

I've never really understood the point of a preface, so when I decided to write one I had to guess as to what the point should be. After looking the word up in the dictionary (a real, hardcopy version, made of paper and everything), I decided to deliver this preface in a more relaxed prose. Given that I don't remember having ever read a preface, I am not even sure that this will be taken in by too many people. Subsequent chapters are written more formally, so if you do not like this style, just skip ahead.

As an undergraduate student, I did a bit of trivial work in the field of string model building, specifically for intersecting D-brane models in Type II-A strings. I began my work at Baylor University toward the end of my second semester, building off of that of Matthew Robinson and a collection of REU[1] students that preceded me. I was tasked with rewriting a small piece of code to construct what Dr. Cleaver, my esteemed advisor, was calling *gauge models*. The hodgepodge of source code, all crammed into a single file, was impenetrable. This is not a shot at the work of Dr. Robinson or the REU students; even Texas summers are not long enough, and I don't think any of them planned for the project to live for long. Well, after about a week of swimming in a sea of painfully written if-statements and for-loops, I decided to rewrite the entire thing from scratch. After all, it couldn't take more than a month or so.

Just over a year later I finally finished version v0.5 of what had become the Gauge Framework. It consisted of roughly fifteen classes spread across just as many files. It was relatively slow ($\sim$100 models/s), bug-laden, and it only built models in four dimensions. While it did have some parallel capabilities, facilitated by

[1] Research Experiences for Undergraduates

the POSIX pthread library, it was just barely able to complete the layer one survey discussed in Section 3.1. In fact, it couldn't build order 24 in any reasonable amount of time! Oh, and there was an infuriating bug that caused it to die intermittently after several hours of running; I've never identified what the source was.

We needed a new framework if we were going to pursue higher layer surveys. After eighteen months or so, the first full version of the Gauge Framework, v1.0, was functional. It ran significantly faster, $\sim$1,000 models/s, and had few known bugs. We'd switched to MPI from POSIX threads, and most of the strange bugs had disappeared. We could now do any survey we wanted, within reason.

All along, as I've worked on the framework, I've been inescapably intrigued by the massive amounts of redundancy in the free fermionic formalism. Understanding the nature of that redundancy has been the most difficult problem I've worked on, and one I have yet to crack. I try to discuss the embarrassingly little that I know about that redundancy herein. Largely, this work is an exposition of the surveys and searches that I've worked on with the Gauge Framework. It is my hope that more will be done with it, but it seems unlikely.

Around 2011 I became very interested in machine learning, neural networks and evolutionary algorithms; I have a problem in that I am easily distracted and like to distract those around me with my diversion. Dr. Tim Renner, though he hadn't defended yet, was my unsuspecting victim. We'd discuss the little projects on which I'd work, and he even started doing the same. One day at lunch he and I were discussing genetic algorithms when he proposed an idea that I found irresistible. Could we apply an evolutionary algorithm to string landscape surveys? To my knowledge he never really followed up, but Chapter 4 has grown out of that discussion as well as several others that he and I have had. It turns out that, if we can apply those algorithms, it isn't a trivial task in the slightest.

It seems appropriate here to mention a few of the projects that didn't make it into this work due to time and my reluctance to write a dissertation that exceeds 100 pages of real work.

First, I've looked into the way models change with compactification. In particular, given a model in D dimension, into what models can it transform in $D-1$ dimensions? It is a very different approach to the exploration of the string landscape, and to my knowledge is the only one of its kind. It turns out that certain progressions are significantly more likely, and the most prevalent suggests that standard model-like groups are more likely in lower dimensions. That said, I don't believe that the theory has been developed well enough yet to present here.

Second, there are two frameworks that I've tinkered with in my spare time; each of which is designed to construct free fermionic models, but the algorithms are different from those used in the Gauge Framework. The first project is designed to make the most of C++'s compile time capabilities. Why calculate the length of a basis vector at run time when you can do it at compile time? We can also take advantage of many of the new C++11 features like the `std::array` type (like a C array, but C++'y). This makes writing end-user programs more difficult, but they seem to run marginally faster. The second project discards the integer encoded arrays and simply uses floats. The idea is that there are several fantastic libraries, OpenBLAS and Lapack in particular, that handle linear algebra very efficiently. Why not use them to do all of the heavy lifting? There are a few difficulties, but I won't get into them here. Suffice it to say that, while it is faster, it is less reliable at higher order. I have a few ideas to improve the situation, but I find it advisable to invest my time in writing my dissertation instead.

With that I want to thank you for reading and hope that you enjoy the topic as much as I have over the years.

ACKNOWLEDGEMENTS

I would like to start by thanking my advisor, Dr. Gerald Cleaver, for all of his support and guidance throughout my time at Baylor. His knowledge of the field and ability to balance my exceedingly ambitious ideas against reality are nothing short of expert. How he has managed to listen to all of my technological rambling is beyond me, but I thank him for it.

My thanks go out to the members of my defense committee, Drs. Gerald Cleaver, Jay Dittmann, Lorin Matthews, Anzhong Wang and Brian Raines, for investing so much time reading and editing this work, a task that I have no doubt was grueling and painful. I thank the faculty of the Departments of Physics at both Baylor and Sam Houston State University for the fine education that I've received over the years and ensuring that I can actually pay my rent. I've enjoyed every minute teaching at these institutions.

I'd like to thank Dr. Joel Walker, my undergraduate advisor. Without Joel I probably would never have made it to graduate school and owe him a great debt. I am thankful for my time working with him and for his friendship.

To all of the members of the EUCOS group, past and present, thank you for pretending I was being helpful, even when I wasn't. Thanks to Dr. Tim Renner for the countless conversations and endless aid he rendered as I got started. Tim is a great person and a pleasure to work with, a state I hope to find myself in again in the future. To Dr. Jared Greenwald, I thank you for the many discussions we've had and for not being too upset after having been led to a dead end by yours truly. Many thanks to the rest of the members of EUCOS, Dr. Gerald Cleaver, Yanbin Deng, Brandon Mattingly and Drake Gates. It has been a pleasure working with each of you.

I would like to acknowledge all of the REU students with whom I've had the pleasure of working: William Hicks, Lesley Vestal, Rachel Elliott, Brandon Mattingly, Kara Merfeld and Caleb Smith. It was a great experience, one that shaped the course of my work.

While here I've had the pleasure of working with V.H. Satheeshkumar on a few projects. He has been a great conversationalist, and our discussions have driven my work. I am sincerely grateful for his friendship and hope that we will be able to work together in years to come.

In the Mathematics Department, I would like to thank Drs. Brian Raines and Markus Hunziker for keeping my mathematics skills fresh. I thoroughly enjoyed my time in your classes. The coursework had surprising influences on my research, and I am grateful for it.

Regarding family, I would like to thank my parents; without their support I may have never made it here. In fact I know that without them I wouldn't even be here. My mother, Kathy, has been nothing but supportive and has never forgotten to remind me of her confidence in me. My dad, Dee, has always pushed me to do better and ensured that I knew he was proud. Thank you both for everything you have done. I wish this could be a more substantial repayment. Of course, I have to thank by brother, Andrew, for never letting life get dull.

I cannot go on without thanking two of the greatest people I know, Nicole Lozano and Chet Gassett. So many thanks for all of the ethanol driven, late night conversations and years of friendship. I am certain that with enough distilled ethanol we could change the world. You are more family than friends, and I look forward to a lifetime of ballyhoo. I'd like to also thank Jerry and Carol Lozano, Reid and Sissi Gassett, and Mark and Liz Speights for the years of support; it is as though I have four sets of parents.

Most importantly, I'd like to thank my fiancée, Lindsey. I could never have done this without you, nor would I ever want to. Your love and support have been invaluable, and I look forward to spending my life with you by my side.

It is my hope that by the time the sun has expired my readership will have exceeded what one can count to on two hands. You are helping to make that possible, and so I thank you.

DEDICATION

For my friends, family and

my love of Physics.

CHAPTER ONE

Introduction to Superstrings

In this chapter I discuss the basic construction of closed superstrings. I begin in Section 1.1 with a review of the superstring action, equations of motion and the Virasoro mode decomposition of the stress-energy tensor. In Section 1.2 we look at how the Virasoro algebra, and thus the stress-energy tensor, generates the conformal symmetry on the worldsheet. I also briefly touch upon operator product expansions. I move on then to gauge fix the action in Section 1.3. As the gauge-fixing is local, we turn to the question of global properties of the string. In particular, we introduce the modular group in Section 1.4. Finally, in Section 1.5 and Section 1.6 we cover the mechanisms with which we prune our state space down to only the physical states, namely BRST quantization and the GSO projection.

Remember that this topic has been explored and expounded on extensively by people much more adept than I. For this reason, I only outline the details necessary for subsequent chapters and direct the reader to any of the standard texts [1, 2, 3, 4, 5] for a more complete treatment.

1.1 Superstring Action

The problem of expressing an action for the supersymmetric string can be approached in several ways. Historically it began with the Nambu-Goto action and subsequently evolved into the Polyakov action, each of which describe bosonic strings.[1] The Ramond-Neveu-Schwarz (RNS) and Green-Schwarz actions were then constructed to incorporate supersymmetry. As our interest is in the RNS action,

[1] Strings with bosonic worldsheet fields only.

since it readily adapts to the heterosis to follow, we will not deal with the Green-Schwarz action herein; see [1, 5] for details.

The RNS action amounts to adding $\mathcal{N} = 1$ worldsheet supersymmetry to the Polyakov action. In what follows X^μ will represent worldsheet scalars while ψ^μ are Majorana spinors. Here μ represents some internal index, and we require that X^μ and ψ^μ transform as vectors under $SO(D-1,1)$ for some natural D. The value and interpretation of D will be addressed in Subsection 1.3.3. In general the action takes the form

$$\begin{aligned} S = -\frac{1}{2\pi}\int d^2\sigma\sqrt{-h}\Big[&h^{\alpha\beta}\partial_\alpha X^\mu\partial_\beta X_\mu + \bar{\psi}^\mu\rho^\alpha D_\alpha\psi_\mu \\ &-2e\bar{\chi}_\alpha\rho^\beta\rho^\alpha\psi^\mu\partial_\beta X_\mu - \frac{1}{2}e\bar{\psi}_\mu\psi^\mu\bar{\chi}_\alpha\rho^\beta\rho^\alpha\chi_\beta\Big] \end{aligned} \tag{1.1.1}$$

where $\bar{\psi} = i\psi^\dagger\rho^0$, $h_{\alpha\beta}$ is the two-dimensional worldsheet metric, ρ^α are the two-dimensional Dirac matrices,

$$\rho^0 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \quad \text{and} \quad \rho^1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \tag{1.1.2}$$

e is the "zweibein" (two-dimensional analog of the familiar vierbein of general relativity) and D_α is the worldsheet covariant spin derivative,

$$D_\alpha\lambda^A_\beta = \partial_\alpha\lambda^A_\beta + \omega^A_{\alpha B}\lambda^B_\beta + \Gamma^\gamma_{\alpha\beta}\lambda^A_\gamma. \tag{1.1.3}$$

Throughout this work $\alpha, \beta, \ldots = 0, 1$ are worldsheet indices, $A, B, \ldots = 0, 1$ are spinor indices and $\mu, \nu, \ldots = 0, 1, \ldots, D-1$ represent some internal indexing which will ultimately be related to the number of spacetime dimensions.

This action is overwhelmed by local symmetries that allow us to make particular gauge choices. Reparameterization invariance allows us to gauge fix our worldsheet metric to

$$h_{\alpha\beta} \to e^\phi\eta_{\alpha\beta}, \tag{1.1.4}$$

local supersymmetry transformations allow us to set

$$\chi_\alpha = 0 \tag{1.1.5}$$

and Weyl symmetry lets us drop the conformal scaling, e^{ϕ}. In this *conformal gauge* we have

$$S = -\frac{1}{2\pi}\int d^2\sigma \left(\eta^{\alpha\beta}\partial_\alpha X^\mu \partial_\beta X_\mu + \bar{\psi}^\mu \rho^\alpha \partial_\alpha \psi_\mu\right). \tag{1.1.6}$$

The process of gauge fixing the worldsheet metric and the gravitino will be approached more systematically in Section 1.3. Until then, we will assume the conformal gauge.

Now, our next steps are to carry out two changes of coordinates. The first is a Wick rotation, $\sigma^0 \to -i\sigma^0$. This euclideanizes the worldsheet so that

$$\eta_{\alpha\beta} \to \delta_{\alpha\beta}. \tag{1.1.7}$$

By convention, the action also undergoes a sign change. This Wick rotation will make our path integrals better behaved because the exponential of our action will be damped as opposed to oscillatory. After Wick rotating, we move to complex coordinates

$$z = e^{(\sigma^0 + i\sigma^1)} \quad \text{and} \quad \bar{z} = e^{(\sigma^0 - i\sigma^1)}. \tag{1.1.8}$$

This is a judicious choice as we could have taken $z = \sigma^0 + i\sigma^1$, the choice in many standard texts [1, 3]; however, since they are clearly related by a conformal transformation, and the former handles the closed string boundary conditions more intuitively, we prefer it. The action now takes the form

$$S = \frac{1}{4\pi}\int dz d\bar{z} \left(4\partial_z X^\mu \partial_{\bar{z}} X_\mu + \tilde{\psi}^\mu \partial_z \tilde{\psi}_\mu + \breve{\psi}^\mu \partial_{\bar{z}} \breve{\psi}_\mu\right) \tag{1.1.9}$$

where

$$\psi = \begin{pmatrix} \tilde{\psi} \\ \breve{\psi} \end{pmatrix}. \tag{1.1.10}$$

At this point we can vary our action to produce the equations of motion:

$$\partial_z \partial_{\bar{z}} X^\mu = 0, \tag{1.1.11}$$

for our bosonic fields, and

$$\partial_z \tilde{\psi}^\mu = 0 \quad \text{and} \quad \partial_{\bar{z}} \breve{\psi}^\mu = 0 \tag{1.1.12}$$

for our fermionic fields. From (1.1.12) we see that the spinor components of ψ, namely $\breve{\psi}$ and $\tilde{\psi}$, are holomorphic and anti-holomorphic, respectively. Taking note of the form of (1.1.11), we see that $\partial_z X^\mu$ and $\partial_{\bar{z}} X^\mu$ are also holomorphic/anti-holomorphic. Consequently we can split our bosonic fields into holomorphic and anti-holomorphic pieces as well,

$$X^\mu(z,\bar{z}) = \breve{X}^\mu(z) + \tilde{X}^\mu(\bar{z}). \tag{1.1.13}$$

Hence forth, we will refer to our holomorphic components, $\breve{X}$ and $\breve{\psi}$, as our *left-movers*, and our anti-holomorphic components, $\tilde{X}$ and $\tilde{\psi}$, as *right-movers*.

This brings us now to considerations of the boundary conditions on our worldsheet fields. There are, in fact, many admissible sets of boundary conditions. Because we are interested in closed strings, we need to consider how $X^\mu(z,\bar{z})$ and $\psi^\mu(z,\bar{z})$ will change as $(z,\bar{z}) \to (e^{2\pi i}z, e^{-2\pi i}\bar{z})$. Upon inspection of the boundary terms of the variation of our action (1.1.9), we note that all boundary terms vanish if

$$\breve{X}^\mu(z) = \breve{X}^\mu(e^{2\pi i}z), \qquad \tilde{X}^\mu(\bar{z}) = \tilde{X}^\mu(e^{-2\pi i}\bar{z}) \tag{1.1.14}$$

and

$$\breve{\psi}^\mu(z) = \pm\breve{\psi}^\mu(e^{2\pi i}z), \qquad \tilde{\psi}^\mu(\bar{z}) = \pm\tilde{\psi}^\mu(e^{-2\pi i}\bar{z}). \tag{1.1.15}$$

In this complex coordinate system we see that $z \to e^{i\alpha}z$ amounts to traversing the closed string a distance α in the σ^1 direction. Given that σ^1 has a periodicity of 2π, we have that the conditions above amount to either periodic or anti-periodic boundary conditions on the worldsheet fields. One thing to note is that the periodicity of the bosonic fields is not strictly required; anti-periodicity will also cause our surface terms to vanish. This possibility is important for the orbifold and spin construction formalism, and thus the anti-periodicity will be allowed in later analysis.

The most important detail is that the closed string boundary conditions do not mix left- and right-moving modes: a left-moving mode will never reflect and become a right-moving mode (nor vice versa). This prompts the question as to whether or

not the supersymmetry transformations do the same. They do, in fact, preserve this distinction which can be seen by inspection of the supersymmetry transformations

$$\delta X^\mu = \bar{\varepsilon}\psi^\mu, \tag{1.1.16a}$$

$$\delta\psi^\mu = \rho^\alpha \partial_\alpha X^\mu \varepsilon. \tag{1.1.16b}$$

Here ε is a global infinitesimal Majorana spinor of the form

$$\varepsilon = \begin{pmatrix} \tilde{\varepsilon} \\ \breve{\varepsilon} \end{pmatrix}. \tag{1.1.17}$$

Working in this conformal gauge with complex coordinates we can observe the transformations for the holomorphic and anti-holomorphic components of X^μ and ψ^μ:

$$\delta\tilde{X}^\mu = \breve{\varepsilon}\tilde{\psi}^\mu, \qquad \delta\breve{X}^\mu = -\tilde{\varepsilon}\breve{\psi}^\mu, \tag{1.1.18a}$$

$$\delta\tilde{\psi}^\mu = -2\partial_{\bar{z}}\tilde{X}^\mu\breve{\varepsilon}, \qquad \delta\breve{\psi}^\mu = 2\partial_z\breve{X}^\mu\tilde{\varepsilon}. \tag{1.1.18b}$$

As we see, the transformations leave left- and right-moving modes invariant. That is, left-moving (right-moving) degrees of freedom of one flavor (bosonic or fermionic) are transformed into left-moving (right-moving) degrees of freedom of the other flavor.

Altogether this means that we have the freedom to introduce left-moving modes that have an internal worldsheet supersymmetry, but right-moving modes that do not. This is the spike of the heterotic string which we will detail in Chapter 2.

Our next topic is solving the equations of motion (1.1.11) and (1.1.12). It is straightforward to see that (1.1.11) is satisfied by

$$X^\mu(z,\bar{z}) = x^\mu - \frac{i}{2}p^\mu \ln(z\bar{z}) + \frac{i}{2}\sum_{n\neq 0}\frac{1}{n}\left(\breve{\alpha}_n^\mu z^{-n} + \tilde{\alpha}_n^\mu \bar{z}^{-n}\right), \tag{1.1.19}$$

so that X^μ splits into the holomorphic and anti-holomorphic components

$$\breve{X}^\mu(z) = \frac{1}{2}x^\mu - \frac{i}{4}p^\mu \ln(z) + \frac{i}{2}\sum_{n\neq 0}\frac{1}{n}\breve{\alpha}_n^\mu z^{-n}, \tag{1.1.20a}$$

$$\tilde{X}^\mu(z) = \frac{1}{2}x^\mu - \frac{i}{4}p^\mu \ln(\bar{z}) + \frac{i}{2}\sum_{n\neq 0}\frac{1}{n}\tilde{\alpha}_n^\mu \bar{z}^{-n}. \tag{1.1.20b}$$

We can do the same for ψ^μ, but this time we have a few choices regarding our boundary conditions. Because we admit the anti-periodic boundary conditions we have two possible sectors: the periodic Ramond sector (R) and the anti-periodic Naveu-Schwarz sector (NS). The Ramond sector expansions are

$$\breve{\psi}^\mu = \frac{\sqrt{2}}{2} \sum_{n=-\infty}^{\infty} \breve{d}_n^\mu z^{-n} \tag{1.1.21a}$$

$$\tilde{\psi}^\mu = \frac{\sqrt{2}}{2} \sum_{n=-\infty}^{\infty} \tilde{d}_n^\mu \bar{z}^{-n} \tag{1.1.21b}$$

while the Naveu-Schwarz sector expansions go as

$$\breve{\psi}^\mu = \frac{\sqrt{2}}{2} \sum_{n=-\infty}^{\infty} \breve{b}^\mu_{(n+\frac{1}{2})} z^{-(n+\frac{1}{2})} \tag{1.1.22a}$$

$$\tilde{\psi}^\mu = \frac{\sqrt{2}}{2} \sum_{n=-\infty}^{\infty} \tilde{b}^\mu_{(n+\frac{1}{2})} \bar{z}^{-(n+\frac{1}{2})}. \tag{1.1.22b}$$

Of course a closed string has two directions that are independent, each of which can be either the R or NS sector. Thus the spectrum of the closed string has four contributing sectors: R-R, NS-NS, R-NS, and NS-R.

In a manner analogous to the construction of the states of the quantum harmonic oscillator, we can construct the states of our theory using the quantum operators arising from the modes of the field expansions above. To do this, we impose the canonical (anti-)commutation relations on the oscillator fields:

$$\begin{aligned} [\breve{\alpha}_m^\mu, \breve{\alpha}_n^\nu] = [\tilde{\alpha}_m^\mu, \tilde{\alpha}_n^\nu] &= m\delta_{m+n}\eta^{\mu\nu} \\ \{\breve{b}_r^\mu, \breve{b}_s^\nu\} = \{\tilde{b}_r^\mu, \tilde{b}_s^\nu\} &= \delta_{r+s}\eta^{\mu\nu} \\ [\breve{d}_m^\mu, \breve{d}_n^\nu] = [\tilde{d}_m^\mu, \tilde{d}_n^\nu] &= \delta_{m+n}\eta^{\mu\nu}. \end{aligned} \tag{1.1.23}$$

These are applied, in all admissible combinations, to the ground state of the respective sector. In this way the spectrum of the theory is constructed up to the constraints imposed by superconformal invariance. One last thing that will be of use in subsequent sections is the mass operators. These operators are defined per

sector for the holomorphic and anti-holomorphic modes separately.

$$\frac{1}{2}\breve{M}^2 = \sum_{m=1}^{\infty} \breve{\alpha}_{-m} \cdot \breve{\alpha}_m + \begin{cases} \sum_{r=1/2}^{\infty} r b_{-r} \cdot \breve{b}_r - \frac{1}{2}, & \text{(NS)} \\ \sum_{n=1}^{\infty} n d_{-n} \cdot \breve{d}_n. & \text{(R)} \end{cases} \tag{1.1.24}$$

The additional $\frac{1}{2}$ contribution to the NS sector mass operator arises as a result of normal ordering. Consistency requires that this normal ordering constant be $\frac{1}{2}$.

Note that above we only presented the holomorphic operator $\breve{M}^2$. The anti-holomorphic version can be expressed simply by replacing the holomorphic modes, $\breve{\alpha}$, etc..., with the corresponding anti-holomorphic modes, $\tilde{\alpha}$, etc.... For brevity, we will omit the anti-holomorphic modes for the remainder of this section as the expressions will become rather redundant if we do not.

Recall that the left- and right-moving modes of the closed string are independent so they require the application of different mass operators. It makes little sense for the mass of the left-moveing modes to be different from the mass of the right-movers. If that were the case, what would be the mass of the string? So, we require that their masses be equal so the mass of the string is then obvious. This results in the loosening of the level-matching condition that the left- and right-movers must have the same number of mode excitations. In the R-R and NS-NS sectors, the level-matching requirement survives, while in the R-NS and NS-R sectors we can have differing number of mode excitations for holomorphic and anti-holomorphic modes due to the normal ordering constant in the NS sector.

Using Noether's theorem we can calculate the stress-energy tensor which is of course conserved, as it should be. In the context of the superstring we get two contributions. The first is the standard bosonic contribution

$$\breve{T}_B = -2 : \partial_z \breve{X}^\mu \partial_z \breve{X}_\mu : -\frac{1}{2} : \breve{\psi}^\mu \partial_z \breve{\psi}_\mu : \tag{1.1.25}$$

while the second is the fermionic contribution

$$\breve{T}_F = 2i\breve{\psi}^\mu \partial_z \breve{X}_\mu \tag{1.1.26}$$

It is a standard step to mode expand the stress-energy tensor. In doing so we have

$$\breve{T}_B = \sum_{n=-\infty}^{\infty} \breve{L}_n z^{-(n+2)} \tag{1.1.27}$$

for the bosonic component and

$$\breve{T}_F = \begin{cases} \sum\limits_{n=-\infty}^{\infty} \breve{G}_n z^{-(n+3/2)}, & \text{(NS)} \\ \sum\limits_{n=-\infty}^{\infty} \breve{F}_n z^{-(n+2)}, & \text{(R)} \end{cases} \tag{1.1.28}$$

for the fermionic components. From the oscillator expansions of X^μ and ψ^μ we find

$$L_m = L_m^{(b)} + L_m^{(f)}, \quad m \in \mathbb{Z} \tag{1.1.29a}$$

$$G_r = \frac{1}{2} \sum_{n \in \mathbb{Z}} : \alpha_{-n} \cdot b_{r+n} :, \quad m \in \mathbb{Z} + \frac{1}{2} \tag{1.1.29b}$$

$$F_m = \frac{1}{2} \sum_{n \in \mathbb{Z}} : \alpha_{-n} \cdot d_{m+n} :, \quad m \in \mathbb{Z} \tag{1.1.29c}$$

with

$$L_m^{(b)} = \frac{1}{2} \sum_{n \in \mathbb{Z}} : \alpha_{-n} \cdot \alpha_{m+n} :, \tag{1.1.30a}$$

$$L_m^{(f)} = \frac{1}{2} \sum_{r \in \mathbb{Z}+1/2} \left(r + \frac{m}{2} \right) : b_{-r} \cdot b_{m+r} :, \qquad (NS) \tag{1.1.30b}$$

$$L_m^{(f)} = \frac{1}{2} \sum_{r \in \mathbb{Z}} \left(n + \frac{m}{2} \right) : d_{-n} \cdot d_{m+n} :, \qquad (R) \tag{1.1.30c}$$

These modes form an algebra in each sector, dubbed the super-Virasoro algebra. In the Naveu-Schwarz sector we have

$$[L_m, L_n] = (m-n) L_{m+n} + \frac{D}{8} m(m^2 - 1) \delta_{m+n}, \tag{1.1.31a}$$

$$[L_m, G_r] = \left(\frac{m}{2} - r \right) G_{m+r}, \tag{1.1.31b}$$

$$\{G_r, G_s\} = 2 L_{r+s} \frac{D}{2} \left(r^2 - \frac{1}{4} \right) \delta_{r+s}, \tag{1.1.31c}$$

while in the Ramond sector we have

$$[L_m, L_n] = (m-n) L_{m+n} + \frac{D}{8} m^3 \delta_{m+n}, \tag{1.1.32a}$$

$$[L_m, F_n] = \left(\frac{m}{2} - n \right) F_{m+n}, \tag{1.1.32b}$$

$$\{F_m, F_n\} = 2 L_{m+n} \frac{D}{2} m^2 \delta_{m+n}. \tag{1.1.32c}$$

These algebras are the symmetry algebras of superconformal invariance. The details of this symmetry are a bit long-winded for this work, so in the next section we will focus solely on conformal invariance, the algebra of which is simply the Virasoro subalgebra,

$$[L_m^{(b)}, L_n^{(b)}] = (m-n)L_{m+n}^{(b)} + \frac{D}{12}m(m^2-1)\delta_{m+n}, \tag{1.1.33}$$

where $L_m^{(b)}$ is the contribution to the bosonic stress-energy tensor due solely to the bosonic worldsheet fields.

This symmetry has far-reaching implications on the state space of the theory. In particular, we require that physical states be superconformally invariant, i.e. a state must be annihilated by the super-Virasoro generators, L_m, G_r and F_m.

1.2 Conformal Invariance and Operator Product Expansions

Here we turn our attention to conformal field theory, in particular conformal invariance. As we have chosen the conformal gauge and the classical theory is conformally invariant, we hope that this invariance will survive in the quantum theory. However, as is often the case when quantizing a theory with an infinite-dimensional symmetry algebra, the (super-)Virasoro algebra picks up an anomaly term. This anomaly will be dealt with in Section 1.3. Before we get to that, however, it is important that we relate the conformal symmetry of the theory back to the Virasoro algebra. In this way, it is the stress-energy tensor that generates the conformal transformations. We will then round out this section with a brief discussion of the conformal dimension and operator product expansions.

1.2.1 Conformal Group in $D = 2$

Let us start with a generic conformal transformation

$$z \to f(z) \quad \text{and} \quad \bar{z} \to \bar{f}(\bar{z}). \tag{1.2.1}$$

Taking an infinitesimal transformation

$$z \to z - \epsilon_n z^{n+1} \quad \text{and} \quad \bar{z} \to \bar{z} - \bar{\epsilon}_n \bar{z}^{n+1} \tag{1.2.2}$$

we can read off the infinitesimal generators in the standard way

$$l_n = -z^{n+1}\partial_z \quad \text{and} \quad \bar{l}_n = -\bar{z}^{n+1}\partial_{\bar{z}}. \tag{1.2.3}$$

It is simple to check that these generators satisfy the algebra

$$[l_m, l_n] = (m-n)l_{m+n}, \quad [\bar{l}_m, \bar{l}_n] = (m-n)\bar{l}_{m+n}, \quad \text{and} \quad [l_m, \bar{l}_n] = 0. \tag{1.2.4}$$

This should look familiar as it is the algebra (1.1.33) without the central charge term, i.e. it is the *classical* Virasoro algebra. This can readily be verified using the results from Section 1.1, but skipping the quantization and thus omitting the normal ordering on the stress-energy tensor.

1.2.2 Conformal Dimension

It is convenient in conformal field theory to distinguish fields in terms of their *conformal dimension*, $(h, \bar{h})$. The components of this pair are referred to as the holomorphic and anti-holomorphic conformal dimension, respectively. Effectively this is a generalization of the tensor type of the field in the conformal field theory:

$$\Phi(z, \bar{z}) = \left(\frac{\partial w}{\partial z}\right)^h \left(\frac{\partial w}{\partial \bar{z}}\right)^{\bar{h}} \Phi(w, \bar{w}). \tag{1.2.5}$$

Just as for an action to be a Lorentz scalar it must have tensor type $(0, 0)$, the action must have conformal dimension $(0, 0)$ to be conformally invariant. Thus, to form conformally invariant actions we must keep track of our conformal dimensions. Note that when a field is (anti-)holomorphic we simply drop the "(anti-)holomorphic" qualification. Thus, to restate the above, the action must have conformal dimension 0.

1.2.3 Operator Product Expansions

In the sections to follow we will be dealing with path integral expectation values

$$\langle A[X] \rangle = \int DX \exp(-S)\, A[X] \tag{1.2.6}$$

where $A[X]$ is a functional of the field variable X. We could, of course, ask for the expectation value of a product of local operators, i.e.

$$\langle A_1(z_1,\bar{z}_1)\dots A_n(z_n,\bar{z}_n)\rangle = \int DX \exp(-S)\, A_1(z_1,\bar{z}_1)\dots A_n(z_n,\bar{z}_n). \tag{1.2.7}$$

What would happen as any two operators A_1 and A_2 approach coincidence? The tool with which we explore this question is the *operator product expansion* (OPE): a product of two local operators can be expressed as the sum of local operators

$$A_i(x)A_j(y) = \sum_k c_{ij}^k(x-y)A_k(y). \tag{1.2.8}$$

Taking the expectation value, as c_{ij}^k is independent of the field variable X,

$$\langle A_i(x)A_j(y)\rangle = \sum_k c_{ij}^k(x-y)\langle A_k(y)\rangle \tag{1.2.9}$$

with c_{ij}^k depending only on the separation $x-y$ and operators A_i, A_j, and A_k. Typically the OPE is expressed with the most singular terms in the expansion first and the nonsingular terms denoted with "...".

One OPE that is of particular interest to us is the TT OPE:

$$T(z)T(0) = \frac{c}{2z^4} + \frac{2}{z^2}T(0) + \frac{1}{z}\partial_z T(0) \tag{1.2.10}$$

From this expression we can see that T is not a tensor since the anomaly term interferes with the proper transformation laws. In order to recover T as a tensor we must ensure, in some way, that the conformal anomaly c is zero. It is this venture that the next section addresses.

1.3 Gauge Fixing and the Conformal Anomaly

In this section we will outline the process of gauge fixing in the path integral context. In particular we will introduce Faddeev-Papov ghosts to handle the excessive gauge redundancies. We start with the path integral

$$Z = \int Dh\, D\chi\, DX\, D\psi\, e^{-S(h,\chi,X,\psi)} \tag{1.3.1}$$

where $S(h, X, \psi)$ is the euclideanized action with h the worldsheet metric, X the bosonic degrees of freedom and ψ the fermionic degrees of freedom. The process of gauge-fixing amounts to separating the path integral into the product of an integral over the gauge group with an integral over the dynamic fields along a gauge slice, and dividing out by the volume of the gauge group. Now, in the case of the supersymmetric string, it is somewhat complicated by the internal supersymmetry. However, one possible interpretation of the superstring is that the worldsheet is actually the two-dimensional commuting subspace of a superspace in which the additional directions are anti-commuting Grassmann numbers. From this point of view we see that the supersymmetry is on the same footing as the (Weyl $\times$ diff) invariance of the metric. It is clear that supersymmetry must also be considered in our gauge-fixing.

The full details of this process are elaborate, and have prompted entire dissertations on their own. Because of this, we only provide what we need, and eschew expounding on all but the most trivial details in favor of brevity. Further we only address the local properties of the theory; global aspects are not discussed until Section 1.4. The interested reader can see any of the standard string theory textbooks for a much more thorough treatment.

We will start first with the reparameterization ghosts in Subsection 1.3.1, move on to the supersymmetric ghosts in Subsection 1.3.2, and finally use the result to determine the number of worldsheet bosonic and fermionic fields that our theory may admit in Subsection 1.3.3.

1.3.1 Reparameterization Ghosts

We begin by fixing our metric, $\mathbf{h}$, which in turn specifies a gauge orbit: the set of all metrics that can be obtained from our particular metric of choice under action of the gauge group. Let g be a gauge transformation with $g : h \to h^g$. We can then consider what happens when we integrate over the volume of the gauge group in

question, in our case (Weyl $\times$ diff). This will ultimately result in the introduction of several fields that encode these motions in the gauge space.

We need a measure, $\Delta(h)$, along one of these orbits in order to truly perform the integration. This measure is referred to as the Faddeev-Popov measure. Postponing the precise specification of $\Delta(h)$, we will use the identity

$$1 = \Delta(h) \int Dg\, \delta(h - \mathbf{h}^g) \tag{1.3.2}$$

Inserting this into our path integral (1.3.1) gives us

$$Z[\mathbf{h}] = \int Dg\, Dh\, D\chi\, DX\, D\psi\, \Delta(h)\delta(h - \mathbf{h}^g)\, e^{-S[h,\chi,X,\psi]} \tag{1.3.3}$$

Of course we can now carry out the integration over h leaving us with

$$Z[\mathbf{h}] = \int Dg\, D\chi\, DX\, D\psi\, \Delta(\mathbf{h}^g)\, e^{-S[\mathbf{h}^g,\chi,X,\psi]}. \tag{1.3.4}$$

Based on the form of (1.3.2) we can see $\Delta(h)$ is gauge invariant and since $S[h, \chi, X, \psi]$ is as well

$$Z[\mathbf{h}] = \int Dg\, D\chi\, DX\, D\psi\, \Delta(\mathbf{h})\, e^{-S[\mathbf{h},\chi,X,\psi]}. \tag{1.3.5}$$

Since no part of the integrand depends on an element of the gauge group we can separate out the integral over the group. This contributes an infinite multiplicative factor equivalent to the volume of the gauge group (Weyl $\times$ diff), and so we simply drop it (or divide by it if you prefer).

$$Z[\mathbf{h}] = \int D\chi\, DX\, D\psi\, \Delta(\mathbf{h})\, e^{-S[\mathbf{h},\chi,X,\psi]}. \tag{1.3.6}$$

To proceed we need to work out the form of $\Delta(\mathbf{h})$. To do this we consider how $\mathbf{h}$ transforms under infinitesimal coordinate transformations. Generally, as $\sigma^\alpha \to \sigma^\alpha + \xi^\alpha$, the metric varies as

$$\delta h_{\alpha\beta} = \nabla_\alpha \xi_\beta + \nabla_\beta \xi_\alpha \tag{1.3.5c}$$

with ξ^α infinitesimal and ∇_α the worldsheet covariant derivative. Using (1.3.2) and $\delta h = h - \mathbf{h}^g$ with g near the identity,

$$\Delta(h)^{-1} = \int D\xi\, \delta\left(\nabla_\alpha \xi_\beta + \nabla_\beta \xi_\alpha\right). \tag{1.3.7}$$

We can then use the the identity

$$\delta(x) = \int Dp\, \exp\left[2\pi i \int d^2\sigma \sqrt{-\mathbf{h}}\, (p \cdot x)\right], \tag{1.3.8}$$

with $\cdot$ representing the contraction of any indices in p and x required to make the integrand a worldsheet scalar, to remove the delta function:

$$\Delta(h)^{-1} = \int D\omega D\xi\, \exp\left[2\pi i \int d^2\sigma \sqrt{-\mathbf{h}}\, \omega^{\alpha\beta}\left(\nabla_\alpha \xi_\beta + \nabla_\beta \xi_\alpha\right)\right]. \tag{1.3.9}$$

Here ω is symmetric and traceless. The inverse determinant then takes the final form

$$\Delta(h)^{-1} = \int D\omega D\xi\, \exp\left[4\pi i \int d^2\sigma \sqrt{-\mathbf{h}}\, \omega^{\alpha\beta} \nabla_\alpha \xi_\beta\right]. \tag{1.3.10}$$

Following the standard prescription for inverting the path integral we introduce two anti-commuting "ghost" fields, $\xi \to c$ and $\omega \to b$. Making the substitution, twiddling a few indices, and selecting a nice normalization, we have our Faddeev-Popov determinant

$$\begin{aligned} \Delta(h) &= \int Db\, Dc\, \exp\left(-\frac{1}{2\pi} \int d^2\sigma \sqrt{-\mathbf{h}}\, b_{\alpha\beta} \nabla^\alpha c^\beta\right) \\ &= \int Db\, Dc\, e^{-S[\mathbf{h}, b, c]}. \end{aligned} \tag{1.3.11}$$

Our next stop along the road to gauge fixing is to substitute our determinant into (1.3.6). We are left with

$$Z[\mathbf{h}] = \int D\chi\, DX\, D\psi\, Db\, Dc\, e^{(-S - S_{bc})} \tag{1.3.12}$$

where

$$\begin{aligned} S + S_{bc} = -\frac{1}{2\pi} \int d^2\sigma \sqrt{-h} \Big[& h^{\alpha\beta} \partial_\alpha X^\mu \partial_\beta X_\mu + \bar{\psi}^\mu \rho^\alpha D_\alpha \psi_\mu \\ & -2\bar{\chi}_\alpha \rho^\beta \rho^\alpha \psi^\mu \partial_\beta X_\mu - \frac{1}{2} \bar{\psi}_\mu \psi^\mu \bar{\chi}_\alpha \rho^\beta \rho^\alpha \chi_\beta + b_{\alpha\beta} \nabla^\alpha c^\beta \Big] \end{aligned} \tag{1.3.13}$$

We are finally in the position to actually fix our gauge. We choose the complex conformal gauge presented in Section 1.1. Unfortunately, by doing so our action becomes rather unruly and horrendous in appearance. Because we have already presented the bosonic and leading fermionic terms of the action in (1.1.9), and since the remaining ghost-free terms will simply be discarded in the next section by a judicious gauge choice, we only express the ghost components here:

$$S_{bc} = \frac{1}{2\pi}\int dz d\bar{z}\left(b_{zz}\partial_{\bar{z}}c^{z} + b_{\bar{z}\bar{z}}\partial_{z}c^{\bar{z}}\right). \tag{1.3.14}$$

Varying this action with respect to b and c we are left with the classical equations of motion

$$\partial_{\bar{z}}c^{z} = \partial_{\bar{z}}b_{zz} = 0 \tag{1.3.15a}$$

$$\partial_{z}c^{\bar{z}} = \partial_{z}b_{\bar{z}\bar{z}} = 0. \tag{1.3.15b}$$

We come to the same conclusion we did for the fields X^{μ} and ψ^{μ}: our ghosts break up into holomorphic and anti-holomorphic components. Adopting our previous notation for left- and right-moving fields the ghost action takes the form

$$S_{bc} = \frac{1}{2\pi}\int dz d\bar{z}\left(\breve{b}\,\partial_{\bar{z}}\breve{c} + \tilde{b}\,\partial_{z}\tilde{c}\right). \tag{1.3.16}$$

The conformal dimensions of b and c are easily read to be $\lambda = 2$ and $(1-\lambda) = -1$, respectively. That is, the conformal dimension of $\breve{c}$ is $(-1, 0)$ while that for $\tilde{c}$ is $(0, -1)$, etc. In this way, we know that they contribute to the stress energy tensor as

$$\breve{T} = -2 : \breve{b}\,\partial_{z}\breve{c} : + : \breve{c}\,\partial_{z}\breve{b} : \tag{1.3.17a}$$

$$\tilde{T} = -2 : \tilde{b}\,\partial_{z}\tilde{c} : + : \tilde{c}\,\partial_{z}\tilde{b} : . \tag{1.3.17b}$$

Using the OPEs

$$\breve{b}(z)\breve{c}(w) = \frac{1}{z-w} + \ldots \qquad \text{and} \qquad \tilde{b}(\bar{z})\tilde{c}(\bar{w}) = \frac{1}{\bar{z}-\bar{w}} + \ldots$$

and the generic form of the TT OPEs presented in (1.2.10), we can see that the contribution of the b and c ghosts to the left- and right-moving conformal anomalies are

$$\breve{c} = \tilde{c} = -3(2\lambda - 1)^2 + 1 = -26. \tag{1.3.18}$$

This magical number will be used in Subsection 1.3.3 when we finally discuss the conformal anomaly cancellation. Until then we are done with our reparameterization ghosts and will now turn to the supersymmetric ghost fields, β and γ.

1.3.2 Superghosts

Having already outlined the process of gauge fixing in Subsection 1.3.1, we can gauge fix the worldsheet gravitino χ quite rapidly. We first note that the local supersymmetry transformations go as

$$\delta X^\mu = \bar{\varepsilon}\psi^\mu, \tag{1.3.19a}$$

$$\delta\psi^\mu = \rho^\alpha\varepsilon(\partial_\alpha X^\mu - \bar{\psi}^\mu\chi_\alpha), \tag{1.3.19b}$$

$$\delta e_\alpha^A = i\bar{\varepsilon}\rho^A\chi_\alpha, \tag{1.3.19c}$$

$$\delta\chi_\alpha = \nabla_\alpha\varepsilon. \tag{1.3.19d}$$

This is not, in fact, all of the symmetry under which the action is invariant. There is an additional bosonic symmetry arising from Weyl invariance (after all e is directly associated with the metric), and a fermionic symmetry that transforms χ as

$$\delta\chi_\alpha = \rho_\alpha\zeta \tag{1.3.20}$$

where ζ is an arbitrary Majorana spinor. These additional symmetries, while important in general, are not strong enough to allow us to gauge fix any more than our one gauge field, χ.

Restricting to our variation (1.3.19d) we expect that our partition function should look like

$$Z[\mathbf{h},\chi] = \int DX\, D\psi\, Db\, Dc\, \Delta(\chi)\, e^{(-S-S_{bc})}. \tag{1.3.21}$$

Following the procedure outlined in Subsection 1.3.1, we have

$$\begin{aligned}\Delta(\chi)^{-1} &= \int D\varepsilon\, \delta\left(\nabla_\alpha \varepsilon\right) \\ &= \int D\varepsilon\, D\varsigma\, \exp\left[2\pi i \int d^\sigma \sqrt{-\mathbf{h}}\, \varsigma^\alpha \nabla_\alpha \varepsilon\right]\end{aligned} \tag{1.3.22}$$

Inverting this by replacing $\varepsilon \to \gamma$ and $\varsigma \to \beta$, and substituting into our path integral we have

$$Z[\mathbf{h}, \chi] = \int DX\, D\psi\, Db\, Dc\, D\beta\, D\gamma\, e^{\left(-S - S_{bc} - S_{\beta\gamma}\right)}. \tag{1.3.23}$$

with

$$\begin{aligned}S + S_{bc} + S_{\beta\gamma} = -\frac{1}{2\pi} \int d^2\sigma \sqrt{-h} \Big[& h^{\alpha\beta} \partial_\alpha X^\mu \partial_\beta X_\mu + \bar{\psi}^\mu \rho^\alpha D_\alpha \psi_\mu \\ & - 2\bar{\chi}_\alpha \rho^\beta \rho^\alpha \psi^\mu \partial_\beta X_\mu - \frac{1}{2} \bar{\psi}_\mu \psi^\mu \bar{\chi}_\alpha \rho^\beta \rho^\alpha \chi_\beta \\ & + b_{\alpha\beta} \nabla^\alpha c^\beta + \beta_\alpha \nabla^\alpha \gamma \Big]\end{aligned} \tag{1.3.24}$$

As before, β and γ are ghost fields; however, now they are *commuting* fields. Additionally, because they are replacing a fermi field they carry an internal spinor index and thus have half-integer conformal dimensions. Specifically β and γ have conformal dimensions $\lambda = \frac{3}{2}$ and $(1 - \lambda) = -\frac{1}{2}$, respectively.

Gauge fixing our action so that we are in the complex conformal gauge and exploiting the residual gauge symmetry just described to force the worldsheet gravitino to vanish gives us our fully gauge-fixed action:

$$\begin{aligned}S = \frac{1}{4\pi} \int dz d\bar{z} \Big(& 4\partial_z X^\mu \partial_{\bar{z}} X_\mu + \tilde{\psi}^\mu \partial_z \tilde{\psi}_\mu + \breve{\psi}^\mu \partial_{\bar{z}} \breve{\psi}_\mu \\ & + \breve{b}\, \partial_{\bar{z}} \breve{c} + \tilde{b}\, \partial_z \tilde{c} + \breve{\beta}\, \partial_{\bar{z}} \breve{\gamma} + \tilde{\beta}\, \partial_z \tilde{\gamma} \Big).\end{aligned} \tag{1.3.25}$$

We have the equations of motion giving us left- and right-moving modes

$$\partial_{\bar{z}} \breve{\gamma} = \partial_{\bar{z}} \breve{\beta} = 0 \tag{1.3.26a}$$

$$\partial_z \tilde{\gamma} = \partial_z \tilde{\beta} = 0, \tag{1.3.26b}$$

and the contribution of β and γ to the stress-energy tensor goes as

$$\breve{T}_B = -\frac{3}{2} : \breve{\beta}\, \partial_z \breve{\gamma} : -\frac{1}{2} : \breve{\gamma}\, \partial_z \breve{\beta} : \tag{1.3.27a}$$

$$\breve{T}_F = -2 : b\gamma : + : c\partial_z \beta : + \frac{3}{2} : \beta \partial_z c : \tag{1.3.27b}$$

with similar contributions to the anti-holomorphic components. Note that there is both a bosonic and a fermionic contribution once we have introduced the supersymmetry ghosts. Of course we can now use our OPEs for the β and γ ghosts,

$$\breve{\beta}(z)\breve{\gamma}(w) = \frac{1}{z-w} + \dots \qquad \text{and} \qquad \tilde{\beta}(\bar{z})\tilde{\gamma}(\bar{w}) = \frac{-1}{\bar{z}-\bar{w}} + \dots,$$

to determine their contribution to the conformal anomaly. In particular we have

$$\breve{c} = \tilde{c} = 3(2\lambda - 1)^2 - 1 = 11. \tag{1.3.28}$$

This concludes our somewhat superficial treatment of the Faddeev-Popov ghosts. We now move on to considerations of the conformal anomaly which culminates in the conclusion that the number of worldsheet fields is strongly constrained.

1.3.3 The Conformal Anomaly

One of the most astonishing features of the quantum theory of strings is that it specifies the number of spacetime dimensions in which it exists. That is, it sets strong constraints on the number of bosonic and fermionic worldsheet fields the theory can consistently handle through conformal anomaly cancellation. As we associate the number of bosonic fields with the number of spacetime dimensions, we are left with the conclusion that bosonic strings require 26 and superstrings require 10 spacetime dimensions.

To see this, consider the free bosonic string

$$S_B = -\frac{1}{2\pi}\int d^\sigma \sqrt{-h}\, h^{\alpha\beta}\partial_\alpha X^\mu \partial_\beta X_\mu. \tag{1.3.29}$$

In this case the only ghosts that enter the action via gauge-fixing are the reparameterization ghosts, b and c. The gauge-fixed action is then

$$S_B + S_{bc} = -\frac{1}{2\pi}\int d^\sigma \sqrt{-\mathbf{h}}\, \mathbf{h}^{\alpha\beta}\left(\partial_\alpha X^\mu \partial_\beta X_\mu + b_{\alpha\gamma}\nabla_\beta\, c^\gamma\right) \tag{1.3.30}$$

with $\mathbf{h}$ the fixed metric. Through the processes outlined in the previous sections we know that the bosonic fields X^μ each contribute $+1$ to each of the left- and

right-moving central charges, and the b and c ghosts contribute -26 to each. In this way, we see that the central charge only vanishes if

$$\breve{c} = \tilde{c} = D - 26 = 0 \quad \Rightarrow \quad D = 26 \tag{1.3.31}$$

We can perform the same process with the gauge-fixed supersymmetric action, (1.3.24). Recall that each spinorial component of a Majorana fermion contributes $\frac{1}{2}$ to the conformal anomaly, and there are D such Majorana fermions as a result of $N = 1$ worldsheet supersymmetry. Now, the β and γ supersymmetry ghosts contribute $+11$ to the central charge, so we are left with

$$\breve{c} = \tilde{c} = D + \frac{D}{2} + 11 - 26 = 0 \quad \Rightarrow \quad D = 10. \tag{1.3.32}$$

Interpreting each of bosonic fields traditionally as a unique direction in space-time, we see that bosonic strings require $D = 26$ and supersymmetric strings require $D = 10$ in order to be free of the conformal anomaly! A string theory in which the conformal anomaly is canceled entirely by the presence of worldsheet fields is referred to as *critical*. Most research has been devoted to these critical string theories as they appear to be the most natural. This type of result is what has driven interest in string theory for years and proves to be one of the most fantastic results in modern theoretical physics.

Another option, one that has proven to be phenomenologically fruitful, is the possibility that the left- and right-moving conformal anomaly may be canceled by differing numbers of ghost fields. This possibility is detailed in Chapter 2.

1.4 Modular Invariance and the Worldsheet

We would be remiss to move on to the physical state selection process, Section 1.5, without pointing out a subtlety in the analysis of Section 1.3. In particular we have only *locally* gauge fixed; the properties of the topology of the worldsheet have yet to be addressed.

Since we are only dealing with closed strings, the euclidean worldsheet cannot have a boundary; since we are dealing with orientable strings, the worldsheet cannot have any "twists". This leaves only the number of "handles" the surface has to distinguish between topologies. In this way we see that our worldsheets must be homeomorphic to a sphere or the connected sum of n tori. The worldsheet is a sphere, or a torus, or a double torus, etc. The number of handles the surface has, n, is referred to as the genus.

Due to the Atiyah-Singer index theorem, we know that the metric of the genus 0 worldsheet, i.e. the sphere, can be gauge fixed globally. In a sense, even beyond just the topological structure, we can consider all genus 0 worldsheets to be equivalent! This is not generally the case for genus $n > 0$. It is, in fact, the Weyl parameter ω,

$$ds^2 = e^{-2\omega} dx \cdot dx, \tag{1.4.1}$$

that presents the obstruction. Fortunately, if our worldsheet admits a flat metric we can always *globally* gauge choose ω. The choice of ω is referred to as the *complex structure* of the manifold in question. Two Riemannian manifolds (of the same genus, etc...) with the same complex structure are conformally equivalent. It is clear that the conformally equivalent manifolds are grouped into conformal equivalence classes. In calculating our partition functions we must integrate over all of the conformally inequivalent manifolds, i.e. over the conformal equivalence classes. How do we perform such an integration? Since the space of equivalence classes is parameterized by a *moduli space*, we can "change parameters" and integrate over the moduli space instead. It is then important to know all of the inequivalent moduli of a particular genus manifold.

Let us look at the (genus 1) torus, T^2. Choosing two distinct complex numbers λ_1 and λ_2 we can identify points in the complex plane by

$$x \sim x + m\lambda_1 + n\lambda_2, \qquad m, n \in \mathbb{Z}. \tag{1.4.2}$$

This specifies the torus

$$T^2 = \mathbb{C}/(\sim) = \mathbb{C}/\langle \lambda_1, \lambda_2 \rangle_{\mathbb{Z}} \tag{1.4.3}$$

where

$$\langle \lambda_1, \lambda_2 \rangle_{\mathbb{Z}} \equiv \{ m\lambda_1 + n\lambda_2 \mid m, n \in \mathbb{Z} \}. \tag{1.4.4}$$

Defining the modular parameter

$$\tau = \frac{\lambda_1}{\lambda_2} \tag{1.4.5}$$

we can always ensure that $\mathrm{Im}(\tau) > 0$, though that may require us to swap λ_1 and λ_2. The complex structure of the torus is specified by τ. Two tori with the same τ (modulo some isometry of the complex plane) are conformally equivalent, so in our partition functions we must integrate over all inequivalent values of τ. We now set out to find the region of the upper half-plane, $\mathcal{H}$, of inequivalent complex structures, dubbed the *fundamental domain* $\mathcal{F}$.

A lattice, as described in (1.4.4), is invariant under the action of the group $SL(2, \mathbb{Z})$. To see this, note that a general element of $SL(2, \mathbb{Z})$ is of the form

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \tag{1.4.6}$$

with $a, b, c, d \in \mathbb{Z}$ and $\det(A) = 1$, and define

$$\begin{pmatrix} \lambda_1' \\ \lambda_2' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix}. \tag{1.4.7}$$

The lattice $\langle \lambda_1', \lambda_2' \rangle_{\mathbb{Z}}$ is isomorphic to $\langle \lambda_1, \lambda_2 \rangle_{\mathbb{Z}}$ since any given point $(m, n) \in \langle \lambda_1, \lambda_2 \rangle_{\mathbb{Z}}$ defines a point $(m', n') \in \langle \lambda_1', \lambda_2' \rangle_{\mathbb{Z}}$ by

$$(m, n) \mapsto (md - nc, na - mb) \tag{1.4.8}$$

and vice versa. Thus two tori are equivalent if there exists an $SL(2, \mathbb{Z})$ transformation that relates their complex structures:

$$\tau \to \tau' = \frac{a\tau + b}{c\tau + d}. \tag{1.4.9}$$

One final thing to note is that, under this action of $SL(2,\mathbb{Z})$, A and $-A$ are equivalent transformations, i.e. they map τ to the same τ'. Removing this redundancy gives us the *modular group*, $PSL(2,\mathbb{Z}) = SL(2,\mathbb{Z})/\mathbb{Z}_2$.

Identifying elements that are equivalent under $PSL(2,\mathbb{Z})$ gives us our moduli space for the 2-tori

$$\mathcal{M}_{T^2} = \mathcal{H}/PSL(2,\mathbb{Z}). \tag{1.4.10}$$

While any choice of fundamental domain, the set of representatives from elements of $\mathcal{M}_{T^2}$, will work, the standard choice is

$$|\mathrm{Re}(\tau)| \leq \frac{1}{2}, \qquad \mathrm{Im}(\tau) > 0, \qquad |\tau| \geq 1.$$

This region is depicted below below.

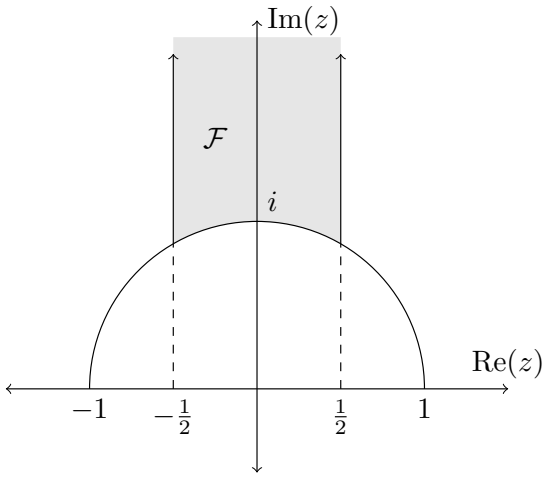

Figure 1.4.1: *The Fundamental Domain* – The shaded region represents the standard choice for the fundamental domain of the modular group.

When computing the one-loop partition functions in our theory we must ensure that they are invariant under the modular transformations $PSL(2,\mathbb{Z})$. To understand why, consider a generic one-loop amplitude

$$\int_{\mathcal{F}} \frac{d^2\tau}{\mathrm{Im}(\tau)^2} \int_{T^2} d^2z_2 \ldots d^2z_n \; \mu(\tau, z) \langle V_1(0), V_2(z_2), \ldots, V_n(z_n) \rangle$$

where $\langle \ldots \rangle$ represents the path integration over the field variables of the theory. If our integrand were not $PSL(2,\mathbb{Z})$ invariant it would result in a miscounting of distinct tori. We refer to this $PSL(2,\mathbb{Z})$ invariance as *modular invariance* and it has profound and far-reaching effects on our model building in subsequent chapters.

Though to show this is far beyond the scope of this work, we only need to ensure modular invariance up to two-loop order in the context of free fermionic heterotic strings. This is because, provided one-loop and two-loop modular invariance, one can prove the modular invariance is held to all orders in perturbation theory.

This concludes our current treatment of modular invariance, however it is important to note that the above discussion is wholly inadequate as we have made no mention of any fermionic moduli nor have we even acknowledged the existence of choices of spin structure. These will be addressed in Chapter 2 where we can deal with them more fully. We now move on to selecting out the physical states from our excessively large state space.

1.5 BRST Quantization

At this point there are two problems[2] with the state space of the theory, both relating to the fact that is much too large. First, we have not ensured that the states are in fact conformally invariant. A state that is not conformally invariant is thus unphysical and should be discarded. Within the canonical quantization formalism this is done by requiring that the states be annihilated by the Virasoro operators. In principle we could do that here, but this would do nothing for our second problem: our states may have unphysical spin-statistics. When we expanded our Fock space of states by including the ghost field excitations, we introduced the possibility that a state could have the unphysical spin-statistics associated with the ghost fields. It

[2] Well, there are really three problems with the state space, but we will deal with the issue of the GSO projection in the next section.

turns out that there is an elegant mechanism for dealing with both of these problems simultaneously: the Bacchic-Route-Store-Teuton (BRST) quantization.

To carry out this procedure we must begin by introducing an operator that counts the number of ghost excitations of a state. This is analogous to the standard number operator of quantum harmonic oscillator fame:

$$\breve{U} = \frac{1}{2\pi i} \oint dz \left(: \breve{c}\breve{b} : + : \breve{\gamma}\breve{\beta} : \right) \tag{1.5.1a}$$

$$\tilde{U} = \frac{1}{2\pi i} \oint d\bar{z} \left(: \tilde{c}\tilde{b} : + : \tilde{\gamma}\tilde{\beta} : \right) \tag{1.5.1b}$$

which count the net ghost contributions to left- and right-movers, respectively. Recalling from Section 1.3, c and γ each have ghost number $+1$ while b and β have ghost number -1, i.e. c and γ are ghosts and b and β are anti-ghosts. In this way our space of states can be separated into disjoint components, $C^{(k,l)}$, in which each state in $C^{(k,l)}$ has left-moving ghost number k and right-moving ghost number l:

$$\breve{U}\varphi = k\varphi \quad \text{and} \quad \tilde{U}\varphi = l\varphi, \qquad \Leftrightarrow \qquad \varphi \in C^{(k,l)}. \tag{1.5.2}$$

Using this mechanism for counting our ghost excitations, we can easily pick out which states have no net ghost contribution!

So, how do we go about handling the conformal invariance? Well, recall that the stress-energy tensor generates the conformal symmetries (Section 1.2). Could we construct an operator from T that allows us to pick out the states that are conformally invariant, but also interacts well with the *cochain* structure just defined? As it happens, we can: the BRST charge operator.

1.5.1 BRST Charge Operator and Cohomology Classes

In this subsection we outline the basic BRST charge construction and explain how it relates to choosing admissibly physical states. To do this we will work with a generic lie algebra G with generators T_i so that

$$[T_i, T_j] = f_{ij}{}^k T_k. \tag{1.5.3}$$

We then introduce ghost field c and anti-ghost field b which canonically anti-commute. An important property of these fields is that the anti-ghost must transform under the adjoint representation of G while the ghost transforms under the dual adjoint representation.

Using the ghost fields, c and b, and the symmetry generators, T_i, we can construct an operator

$$Q = c^i \left(T_i - \frac{1}{2} f_{ij}{}^k c^j b_k \right) \tag{1.5.4}$$

which is dubbed the BRST charge operator. Notice first that it has ghost charge +1. Thus, the application of Q to a state in C^k will produce a state in C^{k+1}. Further still, with a good bit of work, we can show that

$$Q^2 = 0. \tag{1.5.5}$$

In this way Q forms a cohomology operator. The interesting question is what do the cohomology classes under Q represent? The answer: physical states!

Suppose that φ is a state of C^k. We say that the state is BRST-invariant if

$$Q\varphi = 0 \tag{1.5.6}$$

Unapologetically stealing terminology from the de Rahm cohomology of differential forms on a manifold, we could draw a parallel and say that φ is "closed". Going further we could consider states of the form $\varphi = Q\phi$ which might be termed "exact". Because Q is nilpotent, i.e. (1.5.5), all exact forms are trivially closed. The interesting states are the states that are closed but not exact. We can then form the cohomology classes using the equivalence relation on states

$$\varphi \sim \chi \qquad \Leftrightarrow \qquad \varphi - \chi = Q\phi \tag{1.5.7}$$

with $\varphi, \chi \in C^k$ and some $\phi \in C^{k-1}$. These equivalence classes of C^k under Q form the k-th cohomology class of G.

What happens when we act on a state of zero ghost charge with the BRST operator? Since the b modes will annihilate the state, the structure constant term does not contribute to the result:

$$Q\varphi = c^i \left(T_i - \frac{1}{2} f_{ij}{}^k c^j b_k \right) \varphi = c^i T_i \varphi. \tag{1.5.8}$$

Thus the ghost-zero states that are BRST invariant are precisely those states that are ghost-zero and G-invariant. That is, a state is admissible if and only if it is BRST invariant and has ghost number zero.[3] In this way we see that the physical states are the $k = 0$ cohomology classes!

We can generalize this a bit to the closed string by introducing two BRST operators, $\breve{Q}$ and $\tilde{Q}$ which act independently on the left- and right-moving modes of the string. Since each above condition must be applied to the left- and right-movers separately, this construction amounts to a tensor product of two independent systems. We will make this explicit throughout for completeness.

We can now carry out this prescription on the super-Virasoro algebra, though there are subtleties as the above discussion is purely classical. When we quantize, and thus impose normal ordering, Q may no longer be nilpotent and U will pick up a normal ordering constant. The first point is not a problem as long as we are working in the critical dimension with the proper normal ordering constants on our Virasoro generators. The latter point simply means that we will no longer be interested in the $k = 0$ cohomology classes, but rather the $k = -\frac{1}{2}$.

We now introduce our left- and right-moving BRST charge operators in terms of the stress-energy tensor. Recall that the super-Virasoro generators are just the Laurent modes of the stress tensor, so we are not doing anything particularly interesting in making this change. Also, we will adopt a "current" representation of Q as it is pleasing to the eye and has a better physical interpretation (we are interested

[3] This final point relies on the fact that there are no ghost number -1 states.

in symmetries, after all):

$$\breve{Q} = \frac{1}{2\pi i}\oint \left(c\breve{T}_B + \gamma\breve{T}_F + \breve{b}\breve{c}\partial_{\bar{z}}\breve{c} - \frac{1}{2}\breve{c}\breve{\gamma}\partial_{\bar{z}}\breve{\beta} - \frac{3}{2}\breve{c}\breve{\beta}\partial_{\bar{z}}\breve{\gamma} - \breve{b}\breve{\gamma}^2 \right), \quad (1.5.8a)$$

$$\tilde{Q} = \frac{1}{2\pi i}\oint \left(c\tilde{T}_B + \gamma\tilde{T}_F + \tilde{b}\tilde{c}\partial_{\bar{z}}\tilde{c} - \frac{1}{2}\tilde{c}\tilde{\gamma}\partial_{\bar{z}}\tilde{\beta} - \frac{3}{2}\tilde{c}\tilde{\beta}\partial_{\bar{z}}\tilde{\gamma} - \tilde{b}\tilde{\gamma}^2 \right) \quad (1.5.8b)$$

with $\breve{T}_B$, $\breve{T}_F$, $\tilde{T}_B$ and $\tilde{T}_F$ as in Section 1.1. The less-than-obvious coefficients and mixed ghost terms arise within this current representation due the to form of the ghost stress-energy tensors (1.3.17) and (1.3.27).

You can check, after an exhausting computation, that each $\breve{Q}$ and $\tilde{Q}$ is nilpotent by using the identity $Q^2 = \frac{1}{2}\{Q, Q\}$ and the super-Virasoro algebra. For brevity, we leave that to the adventurous reader. Note that it is through the super-Virasoro algebra that the potential anomaly arises. It is straightforward to see that if there is no such anomaly in the super-Virasoro algebra, then there will be no anomaly here and hence: Q^2 is nilpotent in critical string theories (with proper normal ordering constants).

Unfortunately the ghost number operators (1.5.1) introduce non-trivialities. Let us focus now only on the left-moving ghost number operator as the analysis is entirely symmetric. Consider the operator mode expansion form

$$\begin{aligned}\breve{U} = \frac{1}{2}(\breve{c}_0\breve{b}_0 - \breve{b}_0\breve{c}_0) + \frac{1}{2}(\breve{\gamma}_0\breve{\beta}_0 - \breve{\beta}_0\breve{\gamma}_0) \\ + \sum_{n=1}^{\infty}(\breve{c}_{-n}\breve{b}_n - \breve{b}_{-n}\breve{c}_n + \breve{\gamma}_{-n}\breve{\beta}_n - \breve{\beta}_{-n}\breve{\gamma}_n).\end{aligned} \quad (1.5.9)$$

The difficulty arises from the zero-modes. The reparameterization ghosts commute with the Hamiltonian. As a result, the ground state must be doubly degenerate. Let us denote the degenerate states as $|\uparrow\rangle$ and $|\downarrow\rangle$ and take

$$c_0|\downarrow\rangle = |\uparrow\rangle, \;\; b_0|\downarrow\rangle = |\downarrow\rangle, \quad \text{and} \quad c_0|\uparrow\rangle = b_0|\downarrow\rangle = 0. \quad (1.5.10)$$

Now, since c_0 "raises" the state and adds $+1$ to the ghost number we know that

$$U|\uparrow\rangle = U|\downarrow\rangle + 1. \quad (1.5.11)$$

This of course is not enough to specify the ghost number of both states; in the end, we have some freedom to choose a nice convention in the form of a normal ordering constant. Choosing ghost numbers that are symmetric about 0 makes for cleaner results down the line. So, we take

$$U|\uparrow\rangle = +\frac{1}{2} \quad \text{and} \quad U|\downarrow\rangle = -\frac{1}{2}. \tag{1.5.12}$$

We should then be able to force the ghost wave function of a given physical state into one of the two ground states $|\downarrow\rangle$ or $|\uparrow\rangle$ by applying the similarity relation (1.5.7) which defined the cohomology classes, i.e. some state in the cohomology class will have a ghost wave function in one of these ground states. This then means that our physical states must carry a ghost charge of either $\pm\frac{1}{2}$. To see which, we simply need to apply Q to two states, one with each possible ghost charge. When we do that we see that in the case of $+\frac{1}{2}$, the constraint that the zero-mode of the stress-energy tensor vanish for physical states is not required; however, it is for $-\frac{1}{2}$.

Thus we see that the physical states of our theory should have reparameterization ghost number of $-\frac{1}{2}$. What about the supersymmetry ghosts? The answer is that in the NS sector, the ground state is doubly degenerate just as in the preceding discussion. However, the complication comes from the Ramond sector: it contributes an infinite degeneracy. This is not really a problem, it just introduces much more freedom into the construction of states. The supersymmetric ghost zero-mode Fock space introduces infinitely many different *pictures* of the same physical state. We can move around in this Fock space using *picture changing* operators which are a bit beyond the topic here. This process of picture changing is important in the construction of superpotential terms and as such is necessary for considering the detailed phenomenology of the theory.

1.6 The GSO Projection

In this final section we address the issue of a tachyonic ground state in the NS sector of our strings. To see that this is in fact the case requires simply that we "measure" the mass of our NS ground state using the mass operator defined in (1.1.24).

The ground state in the NS sector satisfies

$$\alpha_m^i|0;k\rangle = b_r^i|0;k\rangle = 0, \quad m,r>0 \tag{1.6.1}$$

and

$$\alpha_0^\mu|0;k\rangle = k^\mu|0;k\rangle \tag{1.6.2}$$

Applying our NS mass operator we see that the mass-squared of our ground state is

$$\begin{aligned} M^2|0;k\rangle &= \left(2\sum_{n=-1}^{\infty}\alpha_{-n}\cdot\alpha_n + 2\sum_{n=1/2}^{\infty} r b_{-r}\cdot b_r - 1\right)|0;k\rangle \\ &= -|0;k\rangle, \end{aligned} \tag{1.6.3}$$

i.e. it is tachyonic! Ironically, this is exactly one of the pathologies of the bosonic string that made it inviable. However, upon adding worldsheet fermions we have introduced a mechanism that will allow us to discard "half" of our states, one of which will be the tachyon. This has the further advantage of allowing us to form a gauge lattice. Without this mechanism the state space would be too large and there would not, in general, be a Weyl symmetry between the roots of our gauge groups [6]; that is, there would be no gauge group!.

This mechanism, first introduced by Gliozzi, Scherk and Olive [7, 8], is called the GSO projection. It amounts to the specification of an operator to truncate the spectrum. This operator, called the G-parity operator, is defined differently in each of the two sectors. In the NS sector we have

$$G = (-1)^{F+1}, \qquad F = \sum_{r=1/2}^{\infty} b_{-r}^i b_r^i \tag{1.6.4}$$

while in the R sector

$$G = \Gamma_{11}(-1)^{F+1}, \qquad F = \sum_{n=1}^{\infty} d^i_{-n} d^i_n \tag{1.6.5}$$

where Γ_i, $i = 1, \ldots, 10$ are the typical ten-dimensional Dirac matrices with the definition $\Gamma_{11} \equiv \Gamma_1 \Gamma_2 \ldots \Gamma_{10}$.

We can now use our G-parity operator to project out our states. Since the tachyon in the NS sector has negative G-parity, we will always project the negative G-parity states out of the NS sector. However, we have no particular reason to choose positive or negative G-parity for the R sector. In the closed string theory we have distinct left- and right-movers, so we could choose one of two possibilities for the R-R sectors of the closed string: same parity or opposite parity for the left- and right-movers. The choice is more than arbitrary as in the former case we are left with the Type IIB theory while the second leads to Type IIA. Neither of these theories are in our scope, but they are interesting in their own right. In particular they admit a more non-perturbative approach with the introduction of D-branes.

The GSO projection plays a leading role in the model building to come and has a resounding impact, as does the modular invariance of Section 1.4, on the spectrum of the model.

1.7 Summary

By now we have discussed many of the grand points of superstring theory from the generic action, conformal gauge, and classical equations of motions (Section 1.1), through conformal invariance (Section 1.2), gauge fixing (Section 1.3) and modular invariance (Section 1.4), and finally to BRST quantization (Section 1.5) and the GSO projection (Section 1.6). As has been pointed out several times, this is far from a complete treatment and should be taken with a grain of salt; many of the proofs and conclusions drawn here are hand-wavy at best simply for the sake of brevity.

CHAPTER TWO

Heterotic Strings and The Free Fermionic Construction

The heterotic string, first introduced in [9] and developed in [10, 11], has proven to be a fertile arena for the construction of realistic and semi-realistic low-energy field theories, particularly via the free fermionic (FF) construction scheme [12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47]. My hope for this chapter is to loosely outline the theory of heterotic strings, and the construction of models within the FF formalism.

This chapter is deliberately different from most of the resources I have had the pleasure of finding. Specifically, I do not delve too deeply into the ten-dimensional heterotic string, nor do I treat the four-dimensional construction as the compactification of the ten-dimensional case. The reason for this is that the term "compactification" has a naturally geometric connotation while the generic models we will be considering may or may not. That is, within the FF formalism the six extra dimensions may be viewed as compactified in the topological/geometric sense, but often this interpretation is not well justified, e.g in the case of left-right paired complex fermions. In this case, the "extra" worldsheet fields are seen simply as internal degrees of freedom. The geometric viewpoint is preferred in many contexts within string model building, e.g. in orbifold, Calabi-Yau and flux compactification models, and typically in the FF regime as well. In fact, many or most models constructed via free fermions can also be realized in the setting of asymmetric orbifolds [48, 49]. It is the author's earnest belief that models that accommodate a geometric interpretation are more natural.

In this chapter, we begin by working out the form of the FF heterotic action (Section 2.1), and outline the procedure of heterotic model building (Section 2.4). We work out the modular invariance constraints on the worldsheet fermion boundary conditions (Subsection 2.2.1), and on the GSO projection coefficients (Subsection 2.2.2). Finally, we touch upon the introduction of spacetime supersymmetry (SUSY) into our models in Section 2.5.

2.1 The Free Fermionic Heterotic Action

As we recall from Chapter 1, conformal invariance requires that the central charge of the worldsheet fields vanish. In the case of closed strings, the left- and right-moving fields are independent, and, what's more, so are the left- and right-moving central charges. This alludes to the possibility of introducing left- and right-movers of different types, e.g. supersymmetric left-movers and bosonic right-movers. This is referred to as *heterosis*. However, the differing number of left- and right-moving bosonic fields introduces an interesting question in that the number of such fields is usually associated with the number of spacetime dimensions. Recalling the conformal anomalies for all of the worldsheet fields[1], we note that there must be 10 bosonic left-moving fields and 26 bosonic right-moving fields. We can consider the extra 16 right-moving fields as internal degrees of freedom, and they are ultimately responsible for providing the gauge content of our theory. We can distinguish these internal degrees of freedom as just that, internal, by fermionizing them using the Mandelstam operators

$$\lambda =: e^{-iaX} : \qquad \text{and} \qquad \bar{\lambda} =: e^{ia\bar{X}} : . \tag{2.1.1}$$

Then λ is a Majorana-Weyl worldsheet fermion. In this way our action (1.1.9) can be reexpressed as

[1] Such fields are the bosonic, fermionic, reparameterization ghost and supersymmetry ghost fields.

$$S = \frac{1}{4\pi}\int dzd\bar{z}\left(4\partial_z X^\mu \partial_{\bar{z}} X_\mu + \breve{\psi}^\mu \partial_{\bar{z}}\breve{\psi}_\mu + \tilde{\lambda}^I \partial_z \tilde{\lambda}_I\right) \tag{2.1.2}$$

where $\mu = 0, \ldots, 9$ and $I = 1, \ldots, 32$. As we have replaced each of the 16 right-moving bosons with two right-moving fermions the central charges still cancel entirely.

We can now make another generalization; namely, we could perform this same process on both our left- and right-movers. This decreases the number of large spacetime dimensions down to something more physically realistic, e.g. $D = 4$, and as long as we exchange a boson for two fermions, the left- and right-moving central charges will still vanish. In particular, if D is the number of spacetime dimensions, and $\breve{d}$ and $\tilde{d}$ are the number of left- and right-moving internal fermionic degrees of freedom, respectively, we have the central charges

$$\begin{aligned} \breve{c} &= D + \frac{D}{2} + \frac{\breve{d}}{2} + 11 - 26 = 0, \\ \tilde{c} &= D + \frac{\tilde{d}}{2} - 26 = 0. \end{aligned} \tag{2.1.3}$$

Solving for $\breve{d}$ and $\tilde{d}$, we have

$$\begin{aligned} \breve{d} &= 3(10 - D), \\ \tilde{d} &= 2(26 - D). \end{aligned} \tag{2.1.4}$$

From this we see that we have $D + \breve{d} + \tilde{d}$ worldsheet fermions and $2D$ worldsheet bosons, for a total of $3D + \breve{d} + \tilde{d}$ worldsheet fields. Moving to spacetime light-cone gauge, which mixes the X^0 and X^{D-1} coordinates so that $X^\pm = X^0 \pm X^{D-1}$, allows us to eliminate two bosonic fields ($X^\pm$) and two fermionic fields ($\breve{\psi}^\pm$) so that the final count is $D - 2 + \breve{d} + \tilde{d}$ worldsheet fermions. In particular, with $D = 4$ we must consider 64 worldsheet fermions and 4 worldsheet bosons. The bosonic fields will give rise to the graviton and gravitino modes, but nothing else. Consequently, we neglect them and only worry about counting the gravitino states as is necessary; see Section 2.5

Generically, if we have D spacetime dimensions then we require $\breve{d}$ internal left-movers and $\tilde{d}$ internal right-movers, as determined by (2.1.4), so that the action

becomes

$$S = \frac{1}{4\pi} \int dz d\bar{z} \left(4\partial_z X^\mu \partial_{\bar{z}} X_\mu + \breve{\psi}^\mu \partial_{\bar{z}} \breve{\psi}_\mu + \breve{\lambda}^I \partial_{\bar{z}} \breve{\lambda}_I + \tilde{\lambda}^J \partial_z \tilde{\lambda}_J \right). \tag{2.1.5}$$

with $\mu = 0, \ldots, D-1$, $I = 0, \ldots, \breve{d} - 1$ and $J = 0, \ldots, \tilde{d} - 1$. This is the heterotic string action in the complex conformal gauge with fermionic internal degrees of freedom.

2.2 Modular Invariance of the Partition Function

We could continue our discussion now by solving for equations of motion, etc, but a much more useful approach is to go directly to the partition function. It can be proven, though for the sake of brevity we will not do so, that the one-loop partition function is sufficient to describe all of the physical states of our theory. What's more, one- and two-loop modular invariance is sufficient to ensure modular invariance to all orders [50, 51, 52]. For this reason it is actually easier to handle the partition function than to worry too much about the action, (2.1.5).

Recall from Section 1.4 that the one-loop partition function is an integration over all of the inequivalent tori that our closed string worldsheet could trace out. We then have not just one direction, σ^1, for which boundary conditions must be specified, but rather two. The σ^0 direction is also compact, so we must also consider boundary conditions in this "time" direction as well. To be explicit, the j-th worldsheet fermion, ψ_j, will pick up a phase, V_i^j, so that

$$\psi_j \to -e^{-i\pi V_i^j} \psi_j \tag{2.2.1}$$

for the i-th boundary condition around σ^0 or σ^1.[2] We only need to consider what happens to our $D - 2 + \breve{d} + \tilde{d}$ free fermions since our spacetime bosonic fields must

[2] In general, we can admit boundary conditions that allow the j-th fermion to transform into the k-th fermion, but this is beyond the scope of this work. In the cases we consider, we assume a similarity transformation has rotated the boundary conditions into the eigenphases of a set of complex and/or real eigenstate fermions.

have periodic boundary conditions. In this way we can express the phases as a $(D-2+\breve{d}+\tilde{d})$-vector with components V^j representing the phase of the j-th fermion. These phase vectors are referred to as the *sectors* of the theory, and form a group under addition (modulo 2). This group is typically denoted as Ξ, and as a finite Abelian group can be decomposed into a direct sum of finite cyclic groups

$$\Xi = \mathbb{Z}_{N_1} \oplus \ldots \oplus \mathbb{Z}_{N_L}. \tag{2.2.2}$$

We can thus choose a linearly independent basis, $\{\alpha_1, \ldots, \alpha_L\}$, for Ξ, i.e.

$$V_i = \sum_{j=1}^{L} m_{ij}\,\alpha_j = 0, \quad 0 \leq m_{ij} \leq N_j \tag{2.2.3}$$

if and only if $m_{ij} = 0$ for all $1 \leq j \leq L$. These *basis vectors* α_i are sufficient for reconstructing our sectors, so to specify a model we need only describe the basis.

One important observation is that, as our worldsheet fermions arise from either the fermionization of our bosonic "compact" directions or the left-moving worldsheet SUSY, we require that all of the fermions be "pairable."[3] This means that every phase occurs at least twice, and we can pair two real fermions with the same phase into a complex fermion. This pairing process is restricted to left-left and right-right pairings whenever possible; however, some models actually require left-right pairings. When this occurs it results in a "rank-cut", the removal of a U_1 charge from the gauge lattice, thus reducing the rank of the resulting gauge group by one.

Returning to the one-loop partition function, we must consider all combinations of phases on our fermions; thus, the one-loop partition function can be expressed as

$$Z_1 = \sum_{V,W} C\begin{bmatrix} V \\ W \end{bmatrix} Z\begin{bmatrix} V \\ W \end{bmatrix} \tag{2.2.4}$$

[3] This is not, in fact, a requirement. Relaxing it gives rise to *chiral Ising* models, but these present additional hurdles that we do not address herein.

with V and W the "space" and "time" boundary conditions. We could, in general, consider the n-loop partition function

$$Z_n = \sum_{\substack{\text{spin} \\ \text{structures}}} C\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix} Z\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix} \tag{2.2.5}$$

where a *spin structure* is a choice of $2n$ phase vectors

$$\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix}. \tag{2.2.6}$$

Requiring modular invariance on scattering amplitudes is equivalent to requiring modular invariance on the partition function. Thus, there must be constraints on the sets of admissible spin structures, read sectors, so as to ensure that $Z\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix}$ is modular invariant. Note that modular invariance of $Z\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix}$, is not enough to ensure modular invariance of the Z_n; there are also constraints on the coefficients $C\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix}$. These constraints were originally worked out by two groups in parallel, [50] and [52], and are presented below as they play a crucial role in the construction of models to come.

2.2.1 Modular Invariance of Basis Vectors

It turns out that modular invariance of the spin structures can be reduced to modular invariance on the basis. This is not particularly surprising, so we will not prove it here. In fact, for brevity all of the following results on modular invariance will be presented, but not proven. We refer the reader to either of the original works for a more complete and enlightening treatment.

To begin, let $A = \{\alpha_i \mid 1 \leq i \leq L\}$ be the basis for our model. As each element of the basis vector α_i must represent the phase of a fermion under parallel transport, we expect that it must be either 0 or 1. However, since we require that all of the fermions of our theory be combined into complex pairs, we can admit arbitrary rational phases. We can, without loss of generality, restrict the elements of α_i to the range $(-1, 1]$. What's more, since all of the values are rational, we can find a

common denominator N_i for each α_i such that

$$N_i \alpha_i^j \in 2\mathbb{Z} \tag{2.2.7}$$

with α_i^j the j-th element of α_i and $1 \leq i \leq L$. We refer to this common denominator as the *order* of the basis vector.

As it was proven in [50, 52], modular invariance of the theory requires that the following be met

$$N_i \, \alpha_i \cdot \alpha_i = 0 \pmod 8 \qquad (N_i \text{ even}) \tag{2.2.8a}$$

and

$$N_{ij} \, \alpha_i \cdot \alpha_j = 0 \pmod 4 \tag{2.2.8b}$$

with $N_{ij} \equiv \mathrm{LCM}(N_i, N_j)$. Here "$\cdot$" represents the *Lorentzian dot product*, defined as $\alpha \cdot \beta = \alpha_R \cdot \beta_R - \alpha_L \cdot \beta_L$, with traditional dot products on the right hand side. The above constraints ensure one-loop modular invariance. Additionally, we require that each model include the all-periodic sector, $\mathbf{1}$, i.e.

$$\mathbf{1} \equiv (\overbrace{1 \ \ldots \ 1}^{D-2} \overbrace{1 \ \ldots \ 1}^{\breve{d}} \,||\, \overbrace{1 \ \ldots \ 1}^{\tilde{d}}) = (\, 1^{(D-2+\breve{d})} \,||\, 1^{\tilde{d}} \,) \in A.$$

This is necessary to ensure that we have a well-defined spin structure. Finally, as a necessary condition for modular invariance to arbitrary order in perturbation theory, we require that *for any three basis vectors, the number of simultaneous, real periodic modes is even.* In general this requirement is still too weak; in the case of chiral Ising models, models with unpaired real fermions, any set of four basis vectors must have an even number of simultaneous, real periodic modes. However, because our fermions are all complex, if any three have an even number of simultaneous, real periodic modes, then any four will as well.

Together these four requirements ensure that the individual terms in the partition function, $Z \left[\begin{smallmatrix} V_1 & \cdots & V_n \\ W_1 & \cdots & W_n \end{smallmatrix} \right]$, are modular invariant. This, of course, does not ensure modular invariance of the complete partition function. For that, we must

ensure that the coefficients $C\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix}$ are modular invariant as well. This is the topic of the next subsection.

2.2.2 Modular Invariance of Partition Coefficients

One of the most important requirements for all-order modular invariance is that the n-th order coefficient $C\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix}$ can be decomposable into a product of first-order coefficients:

$$C\begin{bmatrix} V_1 & \dots & V_n \\ W_1 & \dots & W_n \end{bmatrix} = C\begin{bmatrix} V_1 \\ W_1 \end{bmatrix} C\begin{bmatrix} V_2 \\ W_2 \end{bmatrix} \dots C\begin{bmatrix} V_n \\ W_n \end{bmatrix}. \tag{2.2.9}$$

We need only enforce modular invariance on the one-loop coefficients because of this.

Following the convention set in [52], we reexpress our partition coefficients as

$$C\begin{bmatrix} V_i \\ V_j \end{bmatrix} = (-1)^{\left(V_i^1+V_j^1\right)} \exp\left(i\pi\ m_{jk}\ m_{il}\ \ (k_{kl} - \alpha_k \cdot \alpha_l)\right)$$

where m_{ij} are the coefficients in the basis expansion of the sector V_i. Provided the newly introduced coefficients, k_{ij}, satisfy the constraints

$$N_j\, k_{ij} \in 2\mathbb{Z} \tag{2.2.10a}$$

with N_j the order of the j-th basis vector, and

$$k_{ii} + k_{i1} = \frac{1}{4}\alpha_i \cdot \alpha_i + \alpha_i^1 \pmod 2, \tag{2.2.10b}$$

$$k_{ij} + k_{ji} = \frac{1}{2}\alpha_i \cdot \alpha_j \pmod 2, \tag{2.2.10c}$$

our one-loop coefficients $C\begin{bmatrix} V_i \\ V_j \end{bmatrix}$ will be modular invariant. We refer to the coefficients, k_{ij}, as the *GSO projection coefficients* of the model as they arise in the application of the GSO projection as discussed in Section 1.6, and detailed in Section 2.4. These constraints on k_{ij}, together with those discussed in Subsection 2.2.1, ensure that our partition functions, and hence our models, are modular invariant to all order in perturbation theory.

2.3 Worldsheet Supersymmetry

Before we continue on to construct full models, we must recall that our left-moving modes have worldsheet SUSY. This requirement reduced the critical dimension of the theory down to at most ten, so we must ensure that we haven't lost this manifest symmetry after choosing a set of basis vectors. The supercharge for the worldsheet SUSY is given by

$$J = \psi^\mu \partial_z X^\mu + f_{IJK} x^I x^J x^K \tag{2.3.1}$$

where f_{IJK} are the structure constants of some semi-simple Lie group and x^I are real, internal, left-moving fermion modes [53]. The dimension of this Lie group must then be $\breve{d}$.

The traditional choice for this group, and the one assumed herein, is $(SU_2)^{\breve{d}/3}$.[4] The major consequence of this is that the phases of our left-moving fermions must be either periodic or anti-periodic, i.e. the order of our left-movers is 2 and thus our basis vectors have even order.

2.4 Free Fermionic Model Building

We now turn our attentions to the construction of FF models. The process begins with the selection of a modular invariant, linearly independent basis vector set

$$A = \{\alpha_i \mid 1 \leq i \leq L\}. \tag{2.4.1}$$

We refer to $|A| = L$ as the *layer* of the model. Upon selection of a set A, we can proceed to choose some modular invariant GSO projection coefficients, k_{ij}. These coefficients form a square matrix

$$\mathbf{k} = \begin{pmatrix} k_{11} & \dots & k_{1L} \\ \vdots & \ddots & \vdots \\ k_{L1} & \dots & k_{LL} \end{pmatrix} \tag{2.4.2}$$

[4] All other choices have been shown to prohibit $\mathcal{N} = 1$ spacetime supersymmetry [53, 54]. Since $\mathcal{N} = 1$ is the most "natural" form of SUSY we'd like to admit it.

of which the lower triangle is free to be specified. Upon choosing this lower triangle, modular invariance will uniquely determine the upper-triangle and diagonal, with the exception of k_{11}. Fortunately, this element in no way affects the final model, and is thus arbitrary within the requirement (2.2.10a). Together, the choice $\{A, \mathbf{k}\}$ specifies the model, up to vacuum expectation values.

Given our selection of A, we can construct the sectors of the model,

$$\Xi = \Big\{ V_i = \sum_j m_{ij}\alpha_j \mid 0 \leq m_{ij} < N_j \Big\}. \tag{2.4.3}$$

Provided that our choice of A is modular invariant, the elements of Ξ will be as well. Each element, $V_i \in \Xi$, generates a Hilbert space $\mathcal{H}_{V_i}$. Abusing notation, the phases of a state $Q \in \mathcal{H}_V$, are expressed as

$$Q = \frac{1}{2}V_i + F \tag{2.4.4}$$

where $F \in \{-1, 0, 1\}^{D-2+\check{d}+\tilde{d}}$. Applying every possible F to $\frac{1}{2}V_i$ constructs $\mathcal{H}_{V_i}$. Upon doing so for each sector, we build the full Hilbert spaces $\mathcal{H}$ as

$$\mathcal{H} = \bigoplus_{V \in \Xi} \mathcal{H}_V. \tag{2.4.5}$$

This Hilbert space is, of course, much too large. As it stands, most of the states are massive. Since we are only interested in the low-energy effective field theory (LEEFT), we select only the massless states, i.e. the states that satisfy

$$M_L^2 = \frac{(Q_L)^2}{2} - \frac{1}{2} = 0 \quad \Longrightarrow \quad (Q_L)^2 = 1 \tag{2.4.6a}$$

$$M_R^2 = \frac{(Q_R)^2}{2} - 1 = 0 \quad \Longrightarrow \quad (Q_R)^2 = 2 \tag{2.4.6b}$$

where Q_L, and Q_R are the left- and right-moving phases of the state Q, and $a^2 = a \cdot a$. Referring to this reduced Hilbert space as, $\mathcal{H}_{\text{massless}}$, we still have the problem described in Section 1.6. In fact, until we apply the GSO projection the gauge states have no Weyl symmetry [6], and hence do not form a gauge lattice. The matter states will not have a gauge symmetry, and so the model is ill-defined.

We now apply our GSO projection to bring our Hilbert space down to the massless, physical states only. In the FF construction the GSO projection is represented as follows

$$\alpha_i \cdot Q_j = \alpha_i^1 + \sum_{k=1}^{L} k_{ik} m_{jk} \text{ (mod 2)}. \tag{2.4.7}$$

where m_{jk} are the coefficients defining the sector V_k from which Q is constructed, (2.4.4). If the state Q satisfies the above expression, it survives the projection, and is thus an element of the physical Hilbert space $\mathcal{H}_{\text{physical}}$.

Upon applying our GSO projection, we can then begin classifying the states of $\mathcal{H}_{\text{physical}}$ according to their spacetime spin statistics (i.e. whether they fit into the matter or gauge sectors), collecting them into supermultiplets, and determine the number of spacetime supersymmetries, etc. Further consideration of this is left to the next chapter, in which we can restrict our attention to a very particular type of model.

2.5 Spacetime Supersymmetry

A main feature of most modern theories of grand unification is the presence of spacetime SUSY. Consequently, we need to be able to introduce the symmetry into the spacetime theory. To do this we first require that our basis vector set, A, include the SUSY sector

$$\mathbf{S} = \left(\, 1^{D-2} \;\; (1\;0\;0)^{\breve{d}/3} \;\; || \;\; 0^{\tilde{d}} \, \right). \tag{2.5.1}$$

This particular choice is compatible with our choice of worldsheet SUSY, as discussed in Section 2.3. Explicit inclusion of $\mathbf{S}$ is not always required. In particular, when we can express any of the basis vectors as $\mathbf{S} + \alpha$ with α of the form

$$\alpha = \left(\, 0^{D-2+\breve{d}} \;||\; \alpha_R \, \right) \tag{2.5.2}$$

with N_R odd, $\mathbf{S}$ is a generated sector, i.e. $\mathbf{S} = N_R(\mathbf{S} + \alpha)$.

The existence of $\mathbf{S}$ in Ξ is not sufficient to ensure that the model has spacetime SUSY: the GSO projection can break the symmetry. Particular choices of

GSO coefficients will either keep or project out the gravitino states, and hence all superpartners, from the spectrum. More consideration of this is given in '`surveys`'.

To determine the number of spactime supersymmetries, we simply count the number of gravitinos in the spectrum. In FF models, gravitinos are of the form

$$\chi = \left(\tfrac{1}{2}^{D-2} \ \left(\pm\tfrac{1}{2}\ 0\ 0\right)^{\breve{d}/3} \ ||\ 0^{\tilde{d}} \right). \tag{2.5.3}$$

How "strongly" the elements of A overlap with $\mathbf{S}$ determines the number of gravitinos that can survive. Tailoring the basis vector set can break the number of supersymmtries down to zero or a power of two bounded above by $2^{\breve{d}/6}$. For example, in $D = 4$ we can have $\mathcal{N} = 0, 1, 2$ or 4, while in $D = 10$ only $\mathcal{N} = 0$ or 1 are admissible. We will denote the maximum number of spacetime supersymmetries allowed for a given dimension by $\mathcal{N}_{\max}$.

At present, little work has been done to understand what happens to the number of supersymmetries when $\breve{d}/6$ is non-integral, i.e. when D is odd. It is because of this that we do not detail the scenario too deeply. However, we do touch upon this possibility again in the next chapter.

2.6 Summary

We have outlined how to construct a free fermionic heterotic string model up to vacuum expectation values. In Section 2.1 we heuristically constructed the heterotic string action using free worldsheet fermions. Section 2.2 dealt with modular invariance of the theory in terms of modular invariance of the partition function. We dealt with the additional constraints imposed by worldsheet supersymmetry in Section 2.3 and finally outlined the free fermionic construction process, Section 2.4. We then concluded the chapter with a brief discussion of spacetime supersymmetry.

This concludes our general introductory material. Subsequent chapters will present the original work done by the author, preceded by a brief introduction to the topic of gauge models.

CHAPTER THREE

Surveys of Gauge Models

Recent work puts the number of possible string derived models on the order of 10^{500} [55, 56]. Consequently, any efforts to explore this landscape of string vacua require the use of high-performance computing and a choice of construction method. Each construction method has access to different, overlapping regimes of the landscape; here we will focus on the free fermionic heterotic string (FFHS) construction formalism [50, 51, 52]. The FFHS formalism has produced some of the most phenomenologically viable models to date [12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47] and is ideal for computer construction. Random examinations of the landscape, using this formalism, have been performed in the past [57, 58]; however, due to the many-to-one nature of this construction a random survey of the input parameters has many endemic problems that are non-trivial to address [59]. One way to deal with these problems is to *systematically* survey the valid input parameters.

Two software frameworks, currently under development at Baylor University, are being designed and used specifically for the purpose of performing such systematic surveys of the FFHS landscape. One such framework, the Gauge Framework, focuses on systematically building gauge models in D spacetime dimensions. A detailed explanation of what is meant by a "gauge model" is provided in Section 3.1. These surveys serve multiple purposes including aiding in attempts at understanding and reducing the redundancies inherent to the construction method. Furthermore, we can use the results of these searches to guide slower, more detailed surveys.

While originally intended for long-term service, it has grown increasingly apparent that the Gauge Framework has exhaustively surveyed its region of the FFHS landscape. In its present iteration it can construct just over 1,000 models per second on a single processing core, and this performance scales linearly with the number of parallel cores. To date, the Gauge Framework has run on up to 120 processing cores at a rate of roughly 120,000 models per second. It can construct gauge models with any number of layers of any order. To the author's knowledge, this makes it the fastest software solution of its kind.

This chapter will focus on the surveys and results generated by the Gauge Framework to date. In Section 3.1 we define and describe the characteristics of gauge models. Subsection 3.1.2 focuses on the well-known redundancies of the gauge landscape and addresses methods for removing such redundancies. Section 3.2 presents the results of the first layer 1 survey performed by the Gauge Framework in $D = 4$ spacetime dimensions [60]. We then present in Section 3.3 an extended survey of layer 1 models in D spacetime dimensions [61]. In Section 3.4 we present the results of several higher layer[1] surveys and conjecture that the landscape of higher layer models is wholly redundant with layer one. Finally, we conclude this chapter with a brief discussion of a possible generalization of gauge models that may prove to extend the landscape of interest significantly, Section 3.5.

3.1 Gauge Model Building

The Gauge Framework focuses on the construction of gauge models. Further discussion requires a more concrete definition of what a "gauge model" is.

Definition (Gauge Model). A model is a *gauge model* if it can be built from a set of basis vectors in which every basis vector beyond the all-periodic and

[1] By "higher layer" we mean models with more than one additional bosonic sector. For more information see Section 3.1.

SUSY basis vectors is bosonic, that is of the form $(\vec{0}^{10} \mid\mid \vec{\alpha})$, within the free fermionic construction [50, 51, 52, 62].

These models are in many ways some of the most simple models that one can build, and can be thought of as the basis from which more complex models can be built. This makes them interesting as a starting point for systematic surveys. Using information gleaned from these surveys, we can guide, improve and devise further searches of both the gauge and the generic FFHS landscape. An example of this is provided in Chapter 4.

Recall that within the free fermionic framework two inputs are required, the set of basis vectors, $\mathbf{A}$, and the GSO projection coefficient matrix, $\mathbf{k}$. In order to systematically build these models we need to systematically build the input set $\{\mathbf{A}, \mathbf{k}\}$ ensuring that all of the modular invariance constraints are met. This process is unchanged by restricting ourselves to gauge models. There are, however, a few useful simplifications that can be made.

First of all, because our additional basis vectors have all anti-periodic[2] left-movers, the left-moving component of our Lorentz dot products must be zero. This allows us to dispense with that part of the computation; what is the point of explicitly computing something when you already know the result? This, together with the factor of two reduction due to moving from a real to a complex basis,[3] significantly reduces the number of floating-point operations required to perform basic vector operations.

Secondly, by having all anti-periodic left-movers, we ensure that the basis vectors have a zero dot product with our gravitinos; recall the form of the gravitino states (2.5.3). If we then consider the GSO projection operations, (2.4.7), of $\mathbf{1}$, $\mathbf{S}$

[2] Recall that periodic and anti-periodic modes are represented by 1 and 0, respectively. This can be deduced from (2.2.1)

[3] Since all fermions must be pairable we can collect the real fermions into complex pairs. In this way the phase vectors are shortened by a factor of two.

and α_i on the gravitino χ,

$$\begin{aligned} \mathbf{1} \cdot \chi &= k_{12} + 1 \pmod 2, \\ \mathbf{S} \cdot \chi &= k_{22} + 1 \pmod 2, \\ \alpha_i \cdot \chi &= k_{i2} \pmod 2 \end{aligned} \tag{3.1.1}$$

we see that, given the form of $\mathbf{1}$ and $\mathbf{S}$, half of the gravitino states are immediately projected out. The α_i projection will then project out either all of the remaining gravitinos or none of them depending on the choice of k_{i2}. We thus only admit two types of spacetime supersymmetry in our models: $\mathcal{N} = 0$ and $\mathcal{N} = \mathcal{N}_{\max}$, with $\mathcal{N}_{\max}$ defined in Section 2.5.

Finally, because the left-movers have order one,[4] the order of the basis vector as a whole has the same parity as the order of the right-movers, i.e. is even or odd. This results in no odd-order models with $\mathcal{N} = 0$, [60]. A proof of this is rather brief:

Proof: Consider a basis vector set with an odd-order bosonic sector α_i, $i > 2$. Since the order of $\mathbf{S}$ is $N_{\mathbf{S}} = 2$, $k_{i2} \in \{0, 1\}$. The modular invariance constraint, presented in (2.4.6b), gives us

$$\begin{aligned} k_{ij} + k_{ji} &= \frac{1}{2}\, \alpha_i \cdot \alpha_j \pmod 2, \\ k_{i2} + k_{2i} &= \frac{1}{2}\, \mathbf{S} \cdot \alpha_i \pmod 2 \\ &= 0 \pmod 2, \end{aligned} \tag{2.4.6b}$$

so that $k_{2i} = k_{i2} \pmod 2$. However, together with the constraint

$$N_i \alpha_i^j = 0 \pmod 2, \tag{2.4.6a}$$

the only admissible choice for k_{i2} is $k_{i2} = 0$. Thus, we cannot realize $\mathcal{N} = 0$ in a modular invariant way.

[4] Recall that the order of the i-th basis vector is the smallest integer, N_i, such that $N_i \alpha_i^j = 0 \pmod 2$

In these landscape surveys, because the redundancy arises from our representation of the input space, we should not expect the recurrence of models to be meaningful. We are only interested in the "unique" models.

3.1.1 Uniqueness

When considering uniqueness of models, both gauge and matter content should be considered. The nice thing about supersymmetric gauge models is that models with the same gauge group will always have the same matter spectrum and are thus identical. However, this is not true for non-SUSY models,[5] so to consider uniqueness in this case we must investigate the matter content of these models. Fortunately, the exact particle spectrum of these models is not of interest here. We are only concerned with the gauge content and whether the model is supersymmetric. From that, we can use additional software to prepend left-movers to our basis vectors and build models using these gauge models as a starting point. When there is no left-right pairing, the new models will either keep or break the gauge group of their base gauge models. So, for our purposes we will define uniqueness as follows:

> *Definition* (Uniqueness). A model is considered unique if no other model has been previously generated with both the same gauge group and number of space-time supersymmetries.

As we generate models, any model that has a combination of gauge groups and number of spacetime supersymmetries that has not yet been created is retained it. However, any model after that with the same combination of gauge states and SUSY is discarded.[6] This has an impact on the statistics of the non-SUSY models which will be discussed in more detail in Section 3.2.

---

[5] While not generally true, recent tests of the FF Framework on gauge models suggest that even the matter of two non-SUSY gauge models are identical. A future work may expound upon this.

[6] This is not strictly true. For some investigations, in particular into redundancies, it is useful to keep track of models that are not unique as well.

Before we go on, it is interesting to consider the question of exactly how many gauge models there could possibly be regardless of modular invariance, etc. We can easily determine the maximum number of unique models that can be built by considering that only simply-laced gauge groups can produced and that there are no rank cuts [6], thus the total rank must be 22 in $D = 4$. Determining all of the combinations of simply-laced gauge groups whose rank sums to 22 and doubling that, for SUSY and non-SUSY, gives us at most 48,952 unique gauge models. These calculations have been performed for $D = 4$ through $D = 10$ and are provided in Table 3.1.1. Of course, it is unlikely that all of these combinations can exist. In fact, we find that 5,714 models have the proper form, but it is well known that only 9 models are realized by the $D = 10$ heterotic landscape [63], even when considering full matter content. However, because the $D = 4$ landscape is much more complex,[7] we should expect a higher occurrence of unique models than at $D = 10$.

Table 3.1.1: Maximum Number of Unique Simply-Laced Gauge Models in D Spacetime Dimensions

D	$\vartheta of Models$
10	5,714
9	4,140
8	11,988
7	8,576
6	24,508
5	17,341
4	48,952

Note in Table 3.1.1 that there are fewer possible models in $D+1$ than there are in D spacetime dimensions, with D even. This is a manifestation of the fact that there are no $\mathcal{N} = 0$ models in odd dimensions, Section 3.1 (p. 64).

[7] There are 40 real worldsheet fermions in $D = 10$ versus 64 in $D = 4$. This increased number of fermionic degrees of freedom results in a richer landscape.

3.1.2 Redundancies

The free fermionic construction formalism has the inherent problem of redundancy; the mapping from input space to output space is many-to-one. This property is what condemns random surveys and remains a problem for systematic searches. Reducing these redundancies will bring systematic surveys within current technological limits. Both the basis vectors and the GSO projection coefficients present redundancies that can be accounted for and removed in many cases.

The systematic generation of basis vectors admits redundancy in at least two ways, permutations and charge conjugacy. Permutations of the elements of a basis vector leave the mapping invariant as long as the same permutation is applied to each of the basis vectors in the set, i.e.

$$\left\{ \begin{array}{l} (\ \vec{0}^{10} \ || \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \ \vec{0}^{18} \ ) \\ (\ \vec{0}^{10} \ || \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ \vec{0}^{18} \ ) \end{array} \right\} \cong \left\{ \begin{array}{l} (\ \vec{0}^{10} \ || \ 1 \ 1 \ 0 \ 1 \ 0 \ 1 \ \vec{0}^{18} \ ) \\ (\ \vec{0}^{10} \ || \ 1 \ 1 \ 1 \ 0 \ 1 \ 0 \ \vec{0}^{18} \ ) \end{array} \right\}.$$

Here the third column of the right-movers was switched with the sixth. These two sets will generate the same output, given that the same GSO projection matrix is chosen. A scheme for removing these permutation redundancies was developed in [6]. Additionally, we can always flip the signs of the charges as long as that change does not remove modular invariance, i.e.

$$\left\{ \begin{array}{l} (\ \vec{0}^{10} \ || \ 1 \ 1 \ \ 1 \ \ 1 \ 0 \ 0 \ \ 0 \ \ \ 0 \ \vec{0}^{18} \ ) \\ (\ \vec{0}^{10} \ || \ \frac{2}{3} \ \frac{2}{3} \ -\frac{2}{3} \ 0 \ 0 \ 0 \ \ 0 \ \ \ 0 \ \vec{0}^{18} \ ) \\ (\ \vec{0}^{10} \ || \ \frac{1}{2} \ \frac{1}{2} \ \ \frac{1}{2} \ \ \frac{1}{2} \ \frac{1}{2} \ \frac{1}{2} \ -\frac{1}{2} \ -\frac{1}{2} \ \vec{0}^{18} \ ) \end{array} \right\} \cong \left\{ \begin{array}{l} (\ \vec{0}^{10} \ || \ 1 \ 1 \ \ 1 \ \ \ 1 \ \ 0 \ 0 \ 0 \ 0 \ \vec{0}^{18} \ ) \\ (\ \vec{0}^{10} \ || \ \frac{2}{3} \ \frac{2}{3} \ \ \frac{2}{3} \ \ \ 0 \ \ 0 \ 0 \ 0 \ 0 \ \vec{0}^{18} \ ) \\ (\ \vec{0}^{10} \ || \ \frac{1}{2} \ \frac{1}{2} \ -\frac{1}{2} \ -\frac{1}{2} \ \frac{1}{2} \ \frac{1}{2} \ \frac{1}{2} \ \frac{1}{2} \ \vec{0}^{18} \ ) \end{array} \right\}.$$

This is referred to as charge conjugacy and does not change the gauge group. Modding out these two redundancies is not sufficient to completely remove the many-to-one nature of the mapping, and studies are currently under way to find more sources of basis vector redundancy.

There is also a significant source of redundancy in our selection of GSO projection matrices, especially in the case of odd-order basis vectors. An account of how this redundancy presents itself is detailed in [60]. For our purposes it is sufficient to

point out that at layer 1 there are only two unique choices of GSO projection matrix for even-order and only one for odd-order. In the case of even-order the choices of **k** are distinguished by whether or not they admit spacetime supersymmetry. As for the odd-order sets, modular invariance requires that the GSO projection is compatible with SUSY (Section 3.1). Because the redundancies are multiplicative, the overall redundancy grows exponentially with the layer.

Table 3.1.2: *Number of $L = 1$ Models* – For each order we list the most models possible and number of models after the permutation, charge conjugacy and GSO projection redundancies are accounted for. These are not necessarily distinct models, in fact the majority are still redundant.

N	*InitialModels*	*Permutation*	*ChargeConjugacy*	*GSOProjection*
2	8.39×10^6	20	20	10
3	3.14×10^{10}	47	7	7
4	3.76×10^{13}	640	152	76
5	2.38×10^{15}	873	55	55
6	2.63×10^{17}	8,292	772	386
7	3.91×10^{18}	9,352	328	328
8	1.48×10^{20}	71,724	3,748	1,874
9	9.85×10^{20}	70,759	1,679	1,679
10	2.00×10^{22}	463,948	16,172	8,086
11	8.14×10^{22}	413,948	7,339	7,339
12	1.10×10^{24}	2,434,404	62,704	31,352
13	3.21×10^{24}	2,007,773	28,979	28,979
14	3.28×10^{25}	10,756,336	223,020	111,510
15	7.49×10^{25}	8,378,335	104,453	104,453
16	6.19×10^{26}	41,719,604	730,020	365,010

The result of accounting for these redundancies is a significant improvement in the volume of models that must be built, which is depicted in Table 3.1.2. One thing to note is that each of these affects even- and odd-orders to differing extents. However, if we account for all of them the result is that the number of models that must be built (not the number of unique models) at orders $2N$ and $2N + 1$ are of the same magnitude.

3.2 Layer One Survey

Traditionally, the collection of string derived, low energy effective field theories (LEEFTs) is referred to as the landscape. However, because we are interested less in the field theories and more in the mapping from the FFHS input space to this landscape, we can consider only those LEEFTs that are mapped to by a particular input sub-space; namely the layer 1, order 2 through 22 gauge input space. We will refer to this set of inputs as the "layer 1 landscape." We can then look at the relationships between the input and output spaces as well as the mapping between them.

Using the FFHS formalism, we constructed all unique, layer 1 gauge models from order 2 through 22. This amounted to 68 SUSY and 502 non-SUSY models and required a total of 31,863,121 models to be built. Of all of the group combinations found, 50 had both SUSY and non-SUSY realizations. In this section we review the statistics for these 570 models as well as how we may use these results to improve further surveys and what LEEFTs are accessible from these types of inputs.

3.2.1 Model Generation and Redundancy

Here we look at relationships between model generation, the basis vector order and redundancy. Strictly speaking, these relationships have no physical meaning; however, they are important when creating algorithms for the systematic generation of FFHS models, particularly for studies into how redundancies manifest themselves in the gauge input space.

In our model building process, all models of a particular order are generated before progressing to the next. This allows us to ask how the number of unique models generated is dependent on the order. There is a subtlety to these questions in that any model can be, in general, generated at other orders. However, because we have imposed an ordering on the build process this inherently gives preference to lower orders. This has the advantage of improving the efficiency of the build

process and does not affect statistics beyond the physically meaningless question of "at what order was this model generated?"

We know from Section 3.1 that there are no odd-order $\mathcal{N} = 0$ models. This immediately suggests that there is a difference in the way SUSY and non-SUSY models are generated at each order. This difference can be seen in Figure 3.2.1 where the number of unique models generated is plotted with respect to order for both SUSY and non-SUSY data sets. We see that a statistical majority of SUSY models are generated at low order, from order 2 through 12, while a statistically significant number of non-SUSY models are generated through 22.

One may also be interested in how higher orders subsume lower orders. That is, because higher orders admit a significantly higher number of potential models, one might suspect that higher orders may well contain all of the models generated at lower orders. To verify this we look at the number of unique models generated at each order as well as the number of models that were generated at lower orders but are absent from higher orders, Figure 3.2.2.

This information can be used to more efficiently generate models. For example, even-order SUSY models completely subsume lower orders. This means that we could simply build SUSY order 22 and we would get everything below it. That reduces the number of SUSY models that must be built from approximately 1.82×10^7 to 8.4×10^6, roughly half. This is not quite as nice for non-SUSY models in that we would have to build orders 16 through 22. This amounts to 98.89% of the total number of models. Only SUSY would benefit from this approach. Unfortunately, there is no known way to predict which orders subsume lower orders.

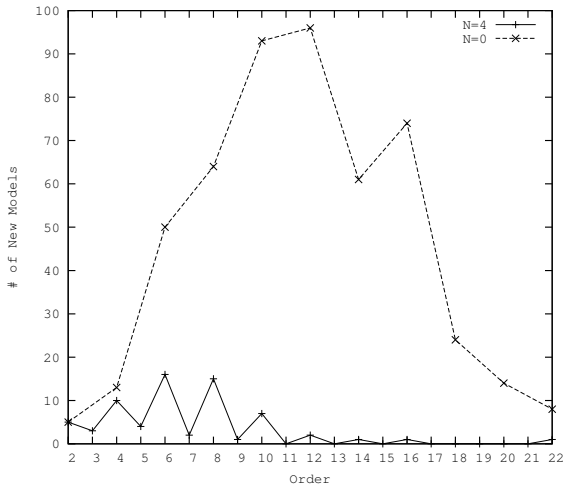

Figure 3.2.1: *Number of New Models at Each Order* – The generation of unique $\mathcal{N} = 4$ models peaks at order 6 with 18 unique models generated. For $\mathcal{N} = 0$ this occurs at order 12 with 96 unique models. Note that the $\mathcal{N} = 0$ curve only has data for even orders because no odd order $\mathcal{N} = 0$ models exist.

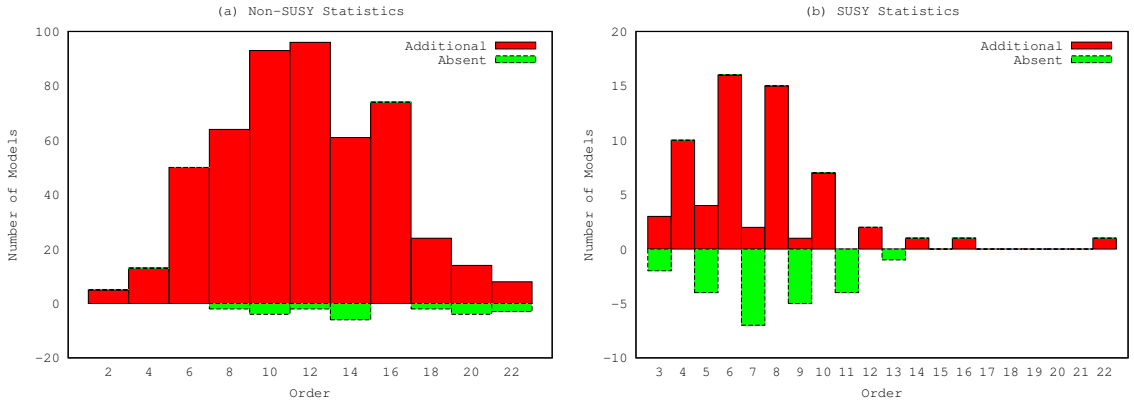

Figure 3.2.2: *Number of Additional and Absent Models at Each Order* – At each order we look at the number of models generated in addition to the models previously created as well as the number of models that are absent at that order. Note that no non-SUSY models are generated at odd-orders so, for brevity, those orders are not plotted.

3.2.2 Group Distribution Statistics

We now focus on specific properties of the LEEFTs, in particular how group factors of each rank manifest themselves across the layer 1 landscape. We begin by considering the number of models with a group factor of a particular rank, M_n, for each, SUSY and non-SUSY, data set, Figure 3.2.3. SU_2 is highly prevalent in both

datasets because it is relatively simple to generate. It amounts to a single, disjoint charge in our gauge states and consequently occurs often when groups are broken.

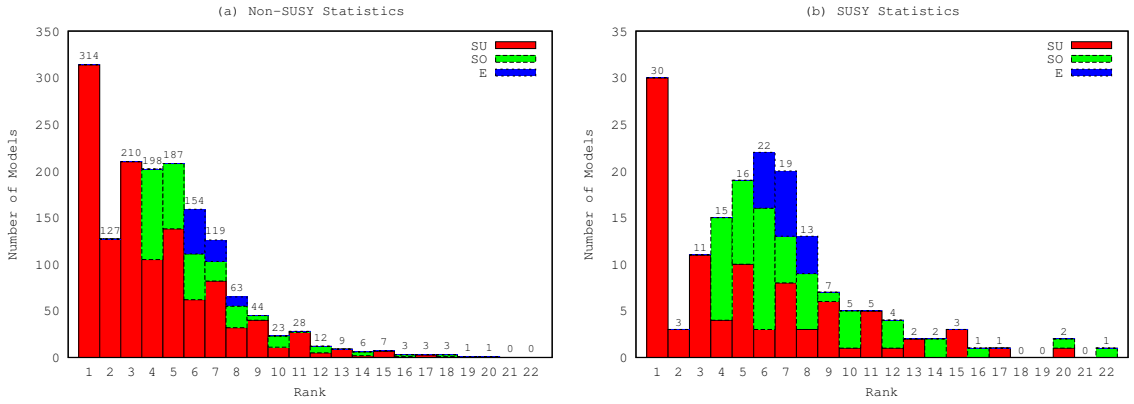

Figure 3.2.3: *Number of Models with Factors of Each Rank* – For each rank and class of gauge group, the number of models with at least one factor of that type is plotted. The label on each bar is the total number of models with at least one group of that rank. The plots for the SUSY and non-SUSY models are provided for comparison. Here the red, green and blue bars represent the number of A, D and E algebras groups, respectively.

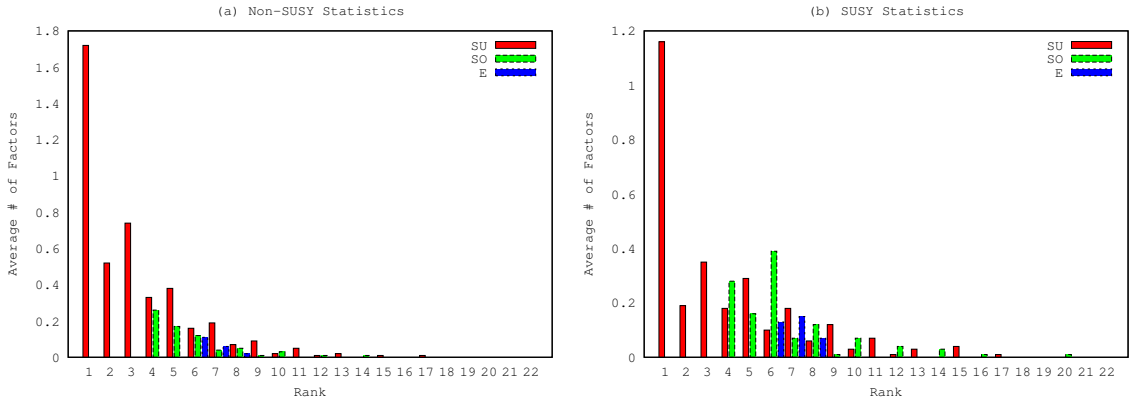

Figure 3.2.4: *Average Number of Factors of Non-Abelian Groups* – For each rank, the average number of factors for each class of groups is plotted for each set of statistics, (a) Non-SUSY Models and (b) SUSY Models. Here the red, green and blue bars represent the number of A, D and E algebras groups, respectively.

For the $\mathcal{N} = 0$ models we see that $M_n > M_{n+2}$ for all classes of group factor, SU_N, SO_N and E_N. However, this trend only occurs for SU_N of odd rank up to $n = 11$. Additionally, we can see $M_{2n-1} > M_{2n}$ up to $n = 10$ for non-SUSY models.

This does not speak to how the factors are distributed amongst the models. Of the non-SUSY models, 314 have at least one factor of SU_2, but generally we can expect more than one for a particular model, approximately 1.74 on average. The average number of factors of each rank, $\bar{M}_n$, is plotted in Figure 3.2.4.

3.2.3 Group Combinations

In this subsection we present statistics for the occurrence of specific group factors in various combinations across the layer 1 landscape. As well as combinations of two group factors, we look at combinations of specific compound factors in conjunction with single and other compound factors. Such compound factors include $E_6 \otimes E_6$, $G_{PS} \equiv SU_4 \otimes SU_2 \otimes SU_2$ (Pati-Salam), $G_{LRS} \equiv SU_3 \otimes SU_2 \otimes SU_2$ (Left-Right Symmetric), $G_{SM} \equiv SU_3 \otimes SU_2 \otimes U_1$ (Standard Model), and $G_{RSM} \equiv SU_3 \otimes SU_2$ (Reduced Standard Model). We also include $\mathcal{F}SU_5 \equiv SU_5 \otimes U_1$, though, because we are not considering matter content, we can only say that the model has the $\mathcal{F}SU_5$ gauge group; it may not actually be $\mathcal{F}SU_5$.

Recall that the modular invariance constraints and redundancies lead to two GSO projection matrices for even-order models and only one for odd-order models. In either case, the GSO projection that admits $\mathcal{N} = 4$ SUSY is consistent, as discussed in Section 3.1. The SUSY landscape exhibits 68 unique models. From these, the percentage of models exhibiting each combination of group factors at least once is calculated as a straight percentage of the 68 models. The full statistics are provided in Table B.0.3 with special GUT[8] group statistics provided in Table 3.2.3 below.

[8] Remember that throughout this work we are only interested in the gauge group, so when we say "GUT group" we are literally referring to the group and not the model as a whole.

While the groups G_{LRS}, G_{RMS} and G_{SM} do not occur in the SUSY landscape, this is not true for $\mathcal{N} = 0$ models; we include their respective columns here for consistency.

As can be seen from either Table B.0.3 or Table 3.2.3, SU_3 never occurs in tandem with SU_2. This means there is no Standard Model gauge group in the SUSY layer 1 landscape, as defined here. Pati-Salam and $\mathcal{F}SU_5$ occur in an equal number across the SUSY landscape but never in the same model. The only compound factor that occurs more than once in any model is $\mathcal{F}SU_5$ and it does so 75% of the time, though this only amounts to 3 models in total.

Table 3.2.3: $\mathcal{N} = 4$ *GUT Group Statistics* – The percentage of all unique $\mathcal{N} = 4$ models with each combination of gauge groups is tabulated. For example, 4.41% of the 68 unique SUSY models have the combination $\mathcal{F}SU_5 \otimes SU_5$ at least once.

$N = 68$	SO_{10}	$E_6 \otimes E_6$	$\mathcal{F}SU_5$	G_{PS}	G_{LRS}	G_{RSM}	G_{SM}
U_1	7.35	1.47	5.88	2.94	0	0	0
SU_2	7.35	0	0	1.47	0	0	0
SU_3	0	0	1.47	0	0	0	0
SU_4	4.41	1.47	0	2.94	0	0	0
SU_5	0	0	4.41	0	0	0	0
$SU_{N>5}$	12.24	1.47	2.94	4.41	0	0	0
SO_8	0	0	0	0	0	0	0
SO_{10}	2.94	0	0	0	0	0	0
$SU_{N>10}$	–	0	0	0	0	0	0
E_N	–	1.47	0	1.47	0	0	0
$E_6 \otimes E_6$	–	0	0	0	0	0	0
$\mathcal{F}SU_5$	–	–	4.41	0	0	0	0
G_{PS}	–	–	–	0	0	0	0
G_{LRS}	–	–	–	–	0	0	0
G_{RSM}	–	–	–	–	–	0	0
G_{SM}	–	–	–	–	–	–	0
Total	13.24	2.94	5.88	5.88	0	0	0

Turning our attention to $\mathcal{N} = 0$ models, we can perform the same statistical analysis we did above. This time, however, we note that there are 502 unique non-SUSY models; see Tables B.1.4 and 3.2.4.

It is interesting to note that the occurrence of $\mathcal{N} = 0$ group combinations is not simply an extension of the $\mathcal{N} = 4$. That is, groups that are uncommon in $\mathcal{N} = 4$ models are not necessarily less common in $\mathcal{N} = 0$. We also find that $SU_3 \otimes SU_2$ combinations never occur with SO_N nor E_N.

Table 3.2.4: $\mathcal{N} = 0$ *GUT Group Statistics* – The percentage of all unique $\mathcal{N} = 0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 502 $\mathcal{N} = 0$ models, i.e. $SO_{10} \otimes SU_5$ occurs in 1.00% of these 502 models.

$N = 502$	SO_{10}	$E_6 \otimes E_6$	$\mathcal{F}SU_5$	G_{PS}	G_{LRS}	G_{RSM}	G_{SM}
U_1	13.35	1.39	20.92	15.54	8.76	14.74	14.74
SU_2	8.17	1.00	12.35	9.36	4.78	8.76	8.76
SU_3	1.20	0	10.76	3.78	6.57	8.96	8.96
SU_4	5.98	0.60	9.56	8.37	3.78	7.17	7.17
SU_5	1.00	0	8.76	2.59	4.38	7.17	7.17
$SU_{N>5}$	10.16	0.40	12.35	9.36	3.00	7.17	7.17
SO_8	1.79	0	0.80	2.39	0	0	0
SO_{10}	2.59	0.40	1.00	2.39	0	0	0
$SU_{N>10}$	–	0.20	0	0.40	0	0	0
E_N	–	0.20	0.60	1.59	0	0	0
$E_6 \otimes E_6$	–	0	0	0.20	0	0	0
$\mathcal{F}SU_5$	–	–	8.76	2.59	4.38	7.17	7.17
G_{PS}	–	–	–	2.99	0.80	1.99	1.99
G_{LRS}	–	–	–	–	1.99	3.78	3.78
G_{RSM}	–	–	–	–	–	6.57	6.57
G_{SM}	–	–	–	–	–	–	6.57
Total	13.94	1.39	20.92	16.33	8.76	14.74	14.74

3.2.4 Extended Layer One Survey

The results above were published in 2011 and have since been extended up through order 32. It was found that the statistics above changed only negligibly as no new SUSY models were generated and only seven unique non-SUSY models were

produced. What's more, all seven additional models were constructed at order 24. Nothing new was found beyond order 24. For the sake of completeness, we provide the additional models built at order 24 in Table 3.2.5, and the $\mathcal{N} = 0$ layer one statistics in four dimensions in Table B.1.5. Note that models 2 and 6 contribute to the Pati-Salam content while models 4–6 contribute potentially standard-model like content.

Table 3.2.5: *Unique Order 24 $\mathcal{N} = 0$ Models* – The unique non-supersymmetric models generated at order 24.

ϑ	*UniqueModels*
1	$(SU_2)^{14} \otimes (U_1)^8$
2	$(SU_2)^{12} \otimes SU_4 \otimes (U_1)^7$
3	$(SU_2)^{12} \otimes SO_8 \otimes SO_{12}$
4	$(SU_2)^8 \otimes (SU_3)^2 \otimes (U_1)^{10}$
5	$(SU_2)^7 \otimes (SU_3)^3 \otimes (U_1)^9$
6	$(SU_2)^6 \otimes (SU_3)^2 \otimes SU_4 \otimes (U_1)^9$
7	$(SU_3)^6 \otimes (U_1)^{10}$

3.3 Survey of Layer One in D Dimensions

In 2013 the results of a survey of layer one models in $D = 4$ through $D = 10$ spacetime dimensions were published in [61]. The survey constructed all layer one models from order 2 through 24. This section presents those results with full statistics in Section B.2.

In conformation with [64], we find two $\mathcal{N}_{\max}$ and six $\mathcal{N} = 0$[9] models in $D = 10$ with one gauge group, SO_{32}, occurring with both possible SUSYs. The corresponding results for zero to six compactifications are presented in Table 3.3.6.

The general trend, an increase in unique models with compactification, is expected; each compactification adds an addition U_1 gauge degree of freedom. In the

[9] An additional model can be produced utilizing chiral Isings, however such models are not considered here.

FFHS formalism, several factors influence how the U_1 alters the initial gauge group: it may enhance the initial gauge group, manifest as an additional group factor, or in more rare cases, result in a splitting of group factors.

Table 3.3.6: *Number of Unique Models* – Number of unique $\mathcal{N} = \mathcal{N}_{\max}$ and $\mathcal{N} = 0$ models for each value of D. Also included is the number of models that have both $\mathcal{N} = \mathcal{N}_{\max}$ and $\mathcal{N} = 0$ realizations.

SUSY	$D = 10$	$D = 9$	$D = 8$	$D = 7$	$D = 6$	$D = 5$	$D = 4$
$\mathcal{N} = \mathcal{N}_{\max}$	2	9	13	16	18	40	68
$\mathcal{N} = 0$	6	32	50	85	73	292	509
Both	1	3	6	8	18	26	50

Upon construction of all unique models, it is a straightforward matter to consider the rate of occurrence of various combinations of group factors, which is presented in the tables in the appendix. Of particular interest is the emergence of GUT groups with compactification. As most common GUTs require multiple low-rank special unitary group factors, a notable exception being SUSY SO_{10}, they arise significantly more often in low dimensions. This can be attributed to the enhancement of the additional U_1 factors produced with compactification. Additionally, the application of the GSO projection to reduce the number of spacetime supersymmetries has the tendency of increasing the production of special unitary groups. This leads to $\mathcal{N} = 0$ models favoring the occurrence of the "unitary GUTs" while the $\mathcal{N}_{\max}$ models favor those with special orthogonal groups.

Beyond GUT groups we can examine how individual group factors arise. As an example, consider SU_3 whose first occurrence is at $D = 7$, $\mathcal{N} = 0$ and first manifests with $\mathcal{N}_{\max}$ at $D = 5$. In all cases, SU_3 occurs in conjunction with a U_1 factor which gives rise to the MSSM group in any situation where SU_2 and SU_3 arise together. Of particular interest is the manner in which SU_3 is produced. Specifically, each compactification produces a U_1 gauge charge. If this charge does not enhance any of the present factors it may remain external, in which case a

subsequent compactification is likely to provided an enhancement to $SU_2 \otimes SU_2$. A further compactification often yields another enhancement to SU_4, suggesting that the SU_3 does not result from the typical enhancement pattern. In fact, it is believed that the most probable method of producing an SU_3 is via breaking of the SU_4 previously described. This is due to the combination of the additional U_1 charge and the application of the GSO projection to reduce the model to $\mathcal{N} = 0$.

3.4 Higher Layer Surveys

It was the hope of this work to explore a large region of the string landscape, namely the gauge landscape. However, the results of several long-running higher layer[10] surveys present evidence that there are far fewer gauge models than expected. The table below describes the present state of these higher layer survey projects.

Table 3.4.7: *Higher Layer Survey Status*

D	*Orders*	*Status*
4	$2 \times 2 - 2 \times 11$	*Completed*
	$3 \times 3 - 5 \times 5$	*Completed*
	$2 \times 2 \times 2 - 2 \times 7 \times 7$	*InProgress*
	$3 \times 3 \times 3 - 3 \times 5 \times 5$	*Planned*
	$4 \times 4 \times 4 - 2 \times 5 \times 5$	*Planned*
5–10	$2 \times 2 - 2 \times 11$	*Completed*
	$3 \times 3 - 10 \times 10$	*InProgress*
	$2 \times 2 \times 2 - 5 \times 5 \times 5$	*Planned*

As of this writing, the surveys have found all of the layer one models described in Sections 3.2 and 3.3 but nothing new. This is surprising in that the number of input sets grows exponentially with the layer of the model. Most surveys to date have

[10] By "higher layer" we mean $l > 1$.

focused on increasing the layer rather than the order because of this expectation.[11] This curious observation prompts the following conjecture:

> *Conjecture* (Completeness). All gauge models can be produced by a layer 1 basis vector set with an appropriate choice of GSO coefficient matrix.

Presently the evidence to support this is "empirical", however it is the expectation of the author that a proof exists and will be found in the near future. The remainder of this section discusses one possible explanation for this result. For the sake of the author's imminent carpal tunnel, we will use the short hand $N_1 \times N_2 \times \ldots \times N_l$ to represent the layer l model with order $N_1 \ldots N_l$, e.g. the layer 3 model with orders 2, 2, and 5 can be represented as $2 \times 2 \times 5$ throughout the remainder of this chapter.

3.4.1 Coprime Orders

Recall that in Section 2.2 partition function modular invariance was discussed. One requirement of modular invariance is that the sectors of the theory form an abelian group, Ξ, which can thus be decomposed into a direct sum of additive cyclic groups

$$\Xi = \mathbb{Z}_{N_1} \oplus \ldots \oplus \mathbb{Z}_{N_L}. \tag{2.2.2}$$

Moving from the generic models discussed in Chapter 2 to gauge models, the cyclic decomposition of Ξ becomes

$$\Xi = \mathbb{Z}_2 \oplus \mathbb{Z}_2 \oplus \mathbb{Z}_{N_1} \oplus \ldots \oplus \mathbb{Z}_{N_l}. \tag{3.4.1}$$

where the first two $\mathbb{Z}_2$ factors are generated by the all-periodic and SUSY sectors, **1** and **S**, while the remaining factors are generated by the additional bosonic generators $\alpha_1,\ldots,\alpha_l$.

---

[11] Another reason for this preference toward increasing layer is that order 2 basis sets have several computational advantages, so only adding order 2 basis vectors is a simpler process than increasing the order.

Suppose for simplicity that we have a layer two model such that

$$\Xi = \mathbb{Z}_2 \oplus \mathbb{Z}_2 \oplus \mathbb{Z}_{N_1} \oplus \mathbb{Z}_{N_2}. \tag{3.4.2}$$

We know that if N_1 and N_2 are coprime, i.e. $\mathrm{GCD}(N_1, N_2) = 1$, then

$$\mathbb{Z}_{N_1} \oplus \mathbb{Z}_{N_2} = \mathbb{Z}_{N_1 N_2}. \tag{3.4.3}$$

This implies that we can replace any pair of basis vectors that have coprime orders with a single basis vector with an order equal to the product of the orders and still generate precisely the same model. This redundancy was, to the author's knowledge, first noted in [64], and poses a rather advantageous simplification: rather than building all order 2×3 we prefer to build all order 6 models instead. This is a vast improvement over building everything. The table below shows a few equivalent layer one and layer two runs and the number of models built in each. Note that in all cases precisely the same models were generated. The only difference between the layer one and layer two surveys was the volume and distribution of models.

Table 3.4.8: *Redundancy of Layer 2 Models* – The total number of SUSY and non-SUSY models at each layer and order is tabulated. For each row, the unique models generated in the $l = 2$ survey are precisely the same as those generated in the $l = 1$ survey. We see the significant redundancy of $l = 2$ with $l = 1$, e.g building the 7394 order 2×3 models is equivalent to building the 362 order 6 models. All models were built in $D = 4$.

$l = 2$			$l = 1$		
N	*#ofSUSY*	*#ofNon-SUSY*	N	*#ofSUSY*	*#ofNon-SUSY*
2×3	3,697	3,697	6	181	181
2×5	342,699	342,699	10	3,983	3,983
2×7	12,032,612	12,032,612	14	55,422	55,422
3×4	1,018,921	1,026,378	12	15,262	15,262
3×5	8,377,415	0	15	104,391	0

It is simple to see why the equivalence occurs. Consider a layer two model with bosonic basis vectors α_1 and α_2. These sectors generate Ξ, but the sector $\alpha_1 + \alpha_2$ also generates Ξ if N_1 and N_2 are coprime. As a concrete example the following

basis vector sets produce the same models

$$\left\{ \begin{array}{l} (\ 0^{10}\ ||\ 1\ 1\ 1\ 1\ \ 0^{7}\ \ 0^{18}\ ) \\ (\ 0^{10}\ ||\ 0\ 0\ \frac{2}{3}\ \frac{2}{3}\ (\frac{2}{3})^{7}\ \ 0^{18}\ ) \end{array} \right\} \cong \left\{ (\ 0^{10}\ ||\ 1\ 1\ -\frac{1}{3}\ -\frac{1}{3}\ (\frac{2}{3})^{7}\ \ 0^{18}\ ) \right\},$$

namely $SO_{28} \otimes E_8$ and $(SU_2)^3 \otimes SO_{24} \otimes E_7$, with and without SUSY respectively.

3.4.2 Generating Sectors

Based on the preceding discussion we can ask whether every redundant, higher layer model is equivalent to a model that is generated by one of its sectors. This is not the case, though it is remarkably common. Consider the basis vector set

$$A = \left\{ \begin{array}{l} (\ 0^{20}\ ||\ 1\ 1\ 1\ 1\ 0\ \ 0\ \ 0\ \ 0\ \ 0^{14}\ ) \\ (\ 0^{20}\ ||\ 0\ 0\ 0\ 1\ \frac{1}{2}\ \frac{1}{2}\ \frac{1}{2}\ \frac{1}{2}\ \ 0^{14}\ ) \end{array} \right\}.$$

This set produces the gauge groups $SO_8 \otimes SO_{36}$ and $SO_{16} \otimes SO_{28}$ with $\mathcal{N} = 0$, and the groups SO_{44} and $SO_{28} \otimes E_8$ with both $\mathcal{N} = 0$ and $\mathcal{N} = 4$. In this case, each gauge group can be built from a basis set consisting of one sector from Ξ_A, the sectors generated by A.

A counterexample, however is the basis set

$$B = \left\{ \begin{array}{l} (\ 0^{20}\ ||\ 1\ 1\ 0\ 0\ 0\ 1\ \ 1\ \ 0^{10}\ \ 0^{5}\ ) \\ (\ 0^{20}\ ||\ 0\ 0\ 1\ 1\ 1\ \frac{1}{2}\ \frac{1}{2}\ \frac{1}{2}^{10}\ \ 0^{5}\ ) \end{array} \right\}$$

which generates the models listed in Table 3.4.9. However, in this case one model, $(SU_2)^2 \otimes SU_6 \otimes SU_{10} \otimes SO_{10} \otimes U_1$ with $\mathcal{N} = 0$, cannot be realized in the manner previously described. All others are produced by one of the individual sectors.

Table 3.4.9: *Models Generated by B* – Two groups, namely $SU_{12} \otimes SO_{10} \otimes E_6$ and $SU_{16} \otimes SO_{14}$, have $\mathcal{N} = 4$ and $\mathcal{N} = 0$ realizations from B while the remaining four groups are produced with $\mathcal{N} = 0$.

ϑ	*GaugeGroup*	$\mathcal{N}$
1	$SU_{12} \otimes SO_{10} \otimes E_6$	4 and 0
2	$SU_{16} \otimes SO_{14}$	4 and 0
3	$SU_{12} \otimes (SO_{10})^2 \otimes U_1$	0
4	$(SU_2)^3 \otimes SU_{14} \otimes SO_{10} \otimes U_1$	0
5	$(SU_2)^2 \otimes SU_6 \otimes SU_{10} \otimes SO_{10} \otimes U_1$	0
6	$SU_4 \otimes SU_{12} \otimes SO_{14} \otimes U_1$	0

A survey to ascertain the typicality of this redundancy is presently underway and results are expected to submitted for publication within the year.

3.5 Generalized Gauge Models

We finish off this chapter with one possible generalization of gauge models that is presently of interest. Consider the following gauge basis set

$$A = \left\{ \begin{array}{l} (\ 1\ \ (1\ \ 1\ \ 1)^3\ \ ||\ \ 1\ \ 1^{12}\ \ 1^9\ ) \\ (\ 1\ \ (1\ \ 0\ \ 0)^3\ \ ||\ \ 0\ \ 0^{12}\ \ 0^9\ ) \\ (\ 0\ \ (0\ \ 0\ \ 0)^3\ \ ||\ \ 1\ \ \frac{1}{2}^{12}\ \ 0^9\ ) \end{array} \right\}.$$

This is easily verified to be modularly invariant and the basis vectors are linearly independent. Provided an admissible choice of $\mathbf{k}$, it will produce either $SO_{20} \otimes SO_{24}$ with $\mathcal{N} = 4$ or $SU_{12} \otimes SO_{18} \otimes U_1 \otimes U_1$ with $\mathcal{N} = 0$. One method of modifying this basis while ensuring modular invariance is to "move" the periodic right-mover to the left. For example

$$A' = \left\{ \begin{array}{l} (\ 1\ \ (1\ \ 1\ \ 1)\ \ (1\ \ 1\ \ 1)^2\ \ ||\ \ 1^{12}\ \ 1^{10}\ ) \\ (\ 1\ \ (1\ \ 0\ \ 0)\ \ (1\ \ 0\ \ 0)^2\ \ ||\ \ 0^{12}\ \ 0^{10}\ ) \\ (\ 0\ \ (0\ \ 1\ \ 0)\ \ (0\ \ 0\ \ 0)^2\ \ ||\ \ \frac{1}{2}^{12}\ \ 0^{10}\ ) \end{array} \right\}.$$

Let us refer to the third basis vector as α. This basis is still modularly invariant and linearlly independent; however, now α's non-trivial left-mover allows a right-mover that is not modularly invariant on its own and some of the sectors generated by α will be unable to generate massless states. In particular, only sectors constructed from an even multiple of α will yield massless states. Because the α and $\mathbf{S}$ share no simultaneously periodic fermions, we can still only generate $\mathcal{N} = 4$ and $\mathcal{N} = 0$. Ultimately, A' produces $SU_{12} \otimes SO_{20} \otimes U_1$ with both $\mathcal{N} = 4$ and $\mathcal{N} = 0$.

Fortuitously, this particular example provides a prototype of the situation in which we are interested: models that cannot be generated by gauge models as presently defined. Adding left-movers that share no periodic modes with $\mathbf{S}$ still results in models of the form in which we are interested, models dominated by gauge content.

Unfortunately, the Gauge Framework is not capable of constructing such generalized models in its present incarnation. Two options for further work on these models exist: either extend the Gauge Framework to build these models as well or use the FF Framework [65]. It would likely be the case that extending the Gauge Framework would be both easier and result in more efficient surveys. In its present form the FF Framework is neither optimized for the construction of gauge models, nor does it have the capacity for parallelism.

It is the hope of the author that future work can be done to extend the gauge surveys discussed herein to include the generalized gauge models. One question in particular is whether the higher layer redundancy will persist upon expanding our definition.

3.6 Summary

This section has covered the author's work on systematic investigations of the string gauge landscape. Gauge models were defined and described in Section 3.1, at which point we discussed several redundancies that plague the surveys as well as what we mean by "uniqueness". Sections 3.2 and 3.3 outline the results of several systematic surveys of layer one gauge models from $D = 4$ up to $D = 10$. We then reviewed the significant redundancy of the multi-layer gauge models and conjectured that they are all redundant with some layer one model (Section 3.4). Finally, we described one possible approach to extending the gauge landscape to include models that may not have such redundancies.

This concludes our discussion of large scale systematic surveys. This remainder of this work will overview a new approach to looking for models in the landscape with specific properties without building everything along the way.

CHAPTER FOUR

Landscape Surveys Through Metaheuristic Algorithms

This final chapter is intended to describe a novel approach to landscape surveys that focuses on finding models with particular properties. Something of a holy grail of string model building is the ability to pick out vacua from the landscape that exhibit specified properties, e.g. gauge group, matter representations, number of supersymmetries, cosmological constant, etc. At present we are forced to build large swaths of the landscape and hope that a model with the desired properties is represented. This is somewhat distasteful in that it is computationally intensive and relatively brute-force. The process can be improved, as was discussed in Chapter 3, by whittling down the input space into equivalence classes of redundant models, but the current state of the art is still much too weak to bring larger scale surveys[1] within reach. We still have redundancy that is far beyond tractable in general; a new approach is necessary.

Given that we have at our disposal frameworks for constructing FFHS models, namely the Gauge and FF Frameworks, we simply need a smarter way of generating the input sets, $\{\mathbf{A}, \mathbf{k}\}$, so that models with particular properties are more likely to appear. If we have a way of assigning some value to a given input, e.g. how well it generates a model with the desired phenomenology, then we can express the problem as an optimization problem:

> *Optimization* – Let $\mathcal{P}$ be the input space and $\texttt{energy} : \mathcal{P} \to \mathbb{R}$ an "energy" function on the inputs. Find an input $\{\mathbf{A}, \mathbf{k}\} \in \mathcal{P}$ that optimizes `energy`.

[1] Even an exhaustive survey of layer three is approaching the limits of the Gauge Framework.

Whether we are minimizing or maximizing `energy` will depend on the context. Throughout this chapter we will liberally mix pseudocode with mathematical language as is demonstrated above.

In this chapter we discuss a *metaheuristic algorithm*, an algorithm that uses lower-level heuristics to find a *near* optimal solution to an optimization problem, as an approach to targeted landscape searches. It is important to realize that while the string landscape is exceedingly vast, it is finite. This means that there will be models that the starry eyed theoretician may dream up that will not exist in the landscape. For this reason we need algorithms that search for solutions that are close to those requested; a truly optimal solution may not exist at all! This is the strength of metaheuristic algorithms.

The remainder of this chapter will describe the *simulated annealing* algorithm (Section 4.1). We will describe the algorithmic process, discuss the heuristic character of each, and finally some of the obstacles of applying these to the problem domain at hand. One thing to note before proceeding is that this algorithm is by no means the only metaheuristic that may be applied to these problems; it is simply the one in which the author grew interested. We will conclude the chapter with a short discussion of genetic algorithms that may be applicable to this problem domain (Section 4.2).

4.1 Simulated Annealing

Many or most metaheuristic algorithms derive from natural processes. Simulated annealing is analogous to metallurgical annealing in which a metal is heated to a high temperature and allowed to slowly cool making the metal softer and more ductile. As the system cools slowly, phase transformations can occur resulting in lower energy structure. A similar process can be applied to an optimization problem in which the energy of a potential solution is minimized as the temperature of the

"system" falls. This metaheuristic algorithm was independently developed in [66, 67] and can be applied to a very broad class of optimization problems.

4.1.1 Description and Pseudocode

The first step in the process of defining an optimization problem is to choose a function, `energy`, to optimize over the input set, $\mathcal{P}$. In the case of simulated annealing this function is referred to as the *energy function*, hence the name, and the objective is to minimize it. To do this we must heuristically define three constructs: the cooling schedule, a neighbor function and a transition function.

The cooling schedule, $\texttt{temp} : [0,1] \to \mathbb{R}^+$, controls the temperature of the system. This can be thought of as a function that specifies the temperature based on how long the system has been running. Throughout this discussion we will represent the "time" as $t \in [0,1]$ and the temperature as $T = \texttt{temp}(t)$. Since we are interested in cooling, we require that `temp` be non-increasing with $\texttt{temp}(0) = T_{\max}$ and $\texttt{temp}(1) = 0$. $T_{\max}$ is a user-specified maximum value for the temperature and should typically be large in comparison to the scale of the energy. The choice of cooling schedule is largely problem specific and is highly complex. An effective cooling schedule must be slow enough for the system to come to equilibrium, and the time that it takes for the system to come to equilibrium depends on both the energy function and the temperature.

The next object of interest is the neighbor function, $\texttt{neighbor} : \mathcal{P} \to \mathcal{P}$ which, given the current solution, selects a neighboring solution, i.e. new trial solutions. Your choice of `neighbor` can be either deterministic or probabilistic and is entirely problem-specific. A standard heuristic for choosing a neighbor function is to make small changes whenever possible. By "small" we mean that for the current solution s and the neighbor $s' = \texttt{neighbor}(s)$, the difference in energies $|\texttt{energy}(s) - \texttt{energy}(s')| << T_{\max}$.

Finally we address the transition probability $\texttt{P}(e, e', T)$ which is a function of the energies of the current and neighboring solutions, e and e', and the temperature of the system T. The only strict requirements on `P` are that $\texttt{P}(e, e', T) > 0$ when $e' > e$ and $T > 0$, and that it should tend to zero with T. The first of these requirements ensures that the system does not get stuck in a local minimum and the latter ensures that we make smaller transitions as time progresses. Most implementations use the standard transition function

$$\texttt{P}(e, e', T) = \begin{cases} 1, & \text{if } e' < e \\ \exp\left(\frac{e-e'}{T}\right), & \text{otherwise,} \end{cases} \tag{4.1.1}$$

first chosen in [66].

We are now in a position to provide a `Julia`[2] implementation of the simulated annealing algorithm.

```
function anneal(energy, neighbor, temp, P, randsol, numsteps)
  time, timestep = 0, 1/numsteps
  s  = randsol()
  es = energy(s)
  while time < 1
    t  = neigh(s)
    et = energy(t)
    if P(es,et,temp(time)) > rand()
      s  = t
      es = et
    end
    time = time + timestep
  end
  return s
end
```

Figure 4.1.1: *Simulated Annealing Algorithm* – Written in `Julia`, it traverses the input space in search of a global minimum energy solution.

The process is pretty straightforward. First we generate an initial solution. If time remains, we generate a neighbor and probabilistically transition to it based on

[2] `Julia` is a high-level, high-performance dynamic programming language for technical computing, with syntax that is familiar to users of other technical computing environments, `http://julialang.org`.

the temperature and energies of the new and initial solutions. We then step time forward and continue generating new potential solutions until time runs out.

There is absolutely no guarantee that this process will converge on an optimal solution; however, if the heuristic components `temp`, `neighbor` and `P` are well selected, and a solution actually exists, the algorithm will often converge.

One variation of this algorithm is to keep track of the best solution presently found. From time to time doing so would be beneficial, but typically the system arrives at the optimal solution last. Because of this and the added computational step, we don't worry about keeping the predecessors of the current solution.

4.1.2 Simulated Annealing as Applied to Gauge Models

We now turn our attention back to our problem: finding models in the landscape with specified properties. As shamefully little is known about the landscape, the problem domain is relatively amorphous. This makes explicit algorithms extraordinarily difficult to devise; however, if we can reexpress our problem in terms of some optimization, simulated annealing may lend itself nicely to finding a solution, i.e. a vacuum with the desired properties.

Since gauge models are so simple, we will try to develop such an approach for these models. Gauge models have only two distinguishing characteristics: the gauge group and number of spacetime supersymmetries. As a simplifying assumption for this treatment, let us only deal with maximally supersymmetric models, $\mathcal{N} = \mathcal{N}_{\max}$, with only one bosonic layer. Recalling from Chapter 3, layer one gauge models admit at most two distinct GSO projection coefficient matrices; the two choices select between $\mathcal{N} = 0$ and $\mathcal{N} = \mathcal{N}_{\max}$. In this way we can neglect $\mathbf{k}$ as well and always choose so it to force $\mathcal{N} = \mathcal{N}_{\max}$, ensuring modular invariance. Additionally, let us restrict our discussion herein to models in four dimensions. This serves no purpose other than to make our subsequent discussion simpler and ensure that $\mathcal{N}_{\max} = 4$. We now formally state our problem.

Simulated Annealing – Develop a simulated annealing algorithm, i.e. choose an energy function, `energy`, a neighbor function, `neighbor`, and cooling schedule, `temp`, that will survey the $D = 4$ gauge landscape in search of $\mathcal{N} = 4$ models with a specified target gauge group, G. Assume the standard acceptance probability, (4.1.1).

We now move on to a choice of neighbor function, `neighbor`. The chief difficulty in defining `neighbor` arises from modular invariance. How does one make a small change to a basis vector without mucking up modular invariance? This is a question to which the author has no wholly satisfying answer. However, for the purposes herein, we do have an example algorithm.

```
function rand(x::Type{BasisVector}, dim)
  # randomly generate a BasisVector of length dim
  ...
end

function neighbor(s)
  t = rand(BasisVector, 22)
  while !modinv(s,t)
    t = rand(BasisVector, 22)
  end
  s + t
end
```

Figure 4.1.2: *Example of* `neighbor` – Randomly generate a basis vector of length 22 and repeat until it is modularly invariant with the original. Return the sum.

We first note that if a layer two basis set $\{\alpha, \beta\}$ is modular invariant then each of its sectors is modular invariant on its own; hence $\alpha + \beta$ is modular invariant. So, as long as we can randomly generate basis vectors that are modular invariant as pairs, then we can randomly generate neighbors. The definition of `rand`(`x` :: `Type`{`BasisVector`}, `dim`), while simple, is a bit technical and unenlightening, thus we eschew a description here. Of course this algorithm is by no means efficient or elegant, but it works as a starting point.

Next we define our cooling schedule, `temp`. We will actually define several for testing purposes later.

```
function mktemp(tmax,exponent)
  @assert(exponent > 0)    # temp must decrease with time
  t -> tmax*(1-t^exponent) # return the temp function
end

lineartemp(t) = mktemp(1000, 1)   # temp(t) = 1000 * (1 - t)
quadtemp(t)   = mktemp(1000, 2)   # temp(t) = 1000 * (1 - t^2)
roottemp(t)   = mktemp(1000, 0.5) # temp(t) = 1000 * (1 - t^0.5)
```

Figure 4.1.3: *Cooling Schedules* – We define a closure `mktemp` for creating some basic temp functions and create linear, quadratic and square-root cooling schedules, for example, each with a maximum temperature of 1000.

Our choices of cooling schedule are somewhat unsophisticated. It will turn out, that because of our trouble defining `energy`, the nature of `temp` is insignificant.

We turn now to the energy function, `energy`, which is the most difficult. The functions `energy` and `neighbor` are intimately connected. In order for the algorithm to effectively evolve a solution toward optimal, we need `neighbor` to favor states of lower `energy`; that is, the probability of a neighbor to be generated should be higher for neighbors that have a lower energy than the present solution. Our choice of `neighbor` is not particularly good at this because the requirements of modular invariance. The energy function is actually the greater trouble. What we find in defining `energy` is that it tends to favor features of the target gauge group rather than the precise group. The implications of this will be seen in Subsection 4.1.4. Here we define two possible energy functions, neither of which will prove to be sufficient for finding particular gauge groups.

The first trial energy function is based on the "entropy" of a gauge group. Effectively, we are interested in the number of gauge groups with both the same number of group factors and non-abelian rank. To this end, we simply count the number of such groups, referring to each as a microstate, and take the natural logarithm in line with Boltzmann's formula

$$S = \log \Omega. \tag{4.1.2}$$

We then define the energy of a solution as the absolute difference between its entropy and the entropy of the target solution. An implementation is provided in Figure 4.1.4.

```
# Number of group factors with each rank
states(r::Int) = (r < 4) ? 2 : (r < 6) ? 2 : (r < 9) ? 3 : 2;

# Number of states based on the number of group factors per rank.
states(prt::Array{Int,1}) = mapreduce(states, *, prt)

# partitions is a built in Julia function
# partition(n,m) = ⟨ partitionings of n into m partitions ⟩
# Number of microstate = sum of numstates per partitioning
function microstates(s)
  mapreduce(states,+,partitions(narank(s),length(s)))
end

# The entropy of a Group
entropy(g::Group) = log(microstates(g))

# The entropy of a BasisVector = entropy of its Group
entropy(s::BasisVector) = entropy(buildgroup(s))

# Closure to return an energy function for target group t
mkentropic(t::Group) = s -> norm(entropy(s) - entropy(t))

# An example energy function for the group SO44
entropicenergy = mkentropic(parsegroup("D22"))
```

Figure 4.1.4: *Entropic Energy Function* – Create a closure that takes the group in which the user is interested and returns an energy function that takes the norm of the difference between the entropies of the target and the newly generated solution.

The first thing to note about this implementation is that it miscounts the number of microstates of a given group. Consider the two groups

$$SU_2 \otimes SU_6 \otimes SO_8 \otimes SO_{24} \quad \text{and} \quad SU_2 \otimes SU_5 \otimes SO_{10} \otimes SO_{24}. \tag{4.1.3}$$

These would each have the same entropy since they have the same number of microstates as defined. This, of course, could be improved by considering the entropy of each of the classes of groups; e.g. the A-class and D-class entropies would be $(1.099, 2.079)$ and $(0.069, 2.079)$, respectively. In this way a better distinction be-

tween the classes is provided and it may improve the effectiveness of the algorithm. We will not expound upon this course of action herein, but it is presently under examination.

The second point is that one might expect this algorithm to drive the groups toward a given non-abelian rank rather than toward a particular group. In particular, note that the number of microstates is determined by the number of ways the non-abelian rank can be partitioned into a particular number of factors. In this way the entropy increases with the non-abelian rank and number of group factors. This will drive the groups toward a particular set of rank distributions rather than toward a particular group.

Our second energy algorithm is actually just the absolute difference in the sample variance of the non-abelian rank of the solution with that of the target.

```
# var is Julia's built in sample variance. It requires the sample
# size to be greater than one; it gives NaN if that is not the
# case. We ensure that it gives 0 in that situation.
variance(rs::Array{Int,N}) = (length(rs) <= 1) ? 0 : var(rs)

variance(g::Group) = variance(map(naranks,g))

mkvarenergy(t::Group) = s -> norm(variance(s) - variance(t))

varenergy = mkvarenergy(parsegroup("D22"))
```

Figure 4.1.5: *Variance Energy Function* – Create a closure that takes the group in which the user is interested and returns an energy function that takes the norm of the difference between the variance of the non-abelian ranks of the target and newly generated solution.

As one might expect this is actually going to drive the ranks toward homogeneity: all group factors with the same rank! One might be able to incorporate the mean rank of the non-abelian factors of the group as well, providing a push in the right direction. Unfortunately, tests are suggesting that this does not work particularly well. We won't discuss the details of that here.

4.1.3 Random Sampling and Random Search

We are now in a position to run our simulated annealing algorithm, but before we do we have to have a method of determining how well the algorithm works. To that end, we choose two algorithms, random sampling and random search, to which we will compare the simulated annealing algorithm. This subsection will discuss those algorithms.

The random sampling algorithm is pretty straightforward. Generate a random solution, determine how will that solutions solves the problem. If the energy of the solution is zero we terminate as it is a minimum; otherwise we generate a new random solution and continue. We repeat this process at most some specified number of times and return the best after the process has terminated. The Julia implementation is below.

```
function randomsample(energy, numloops, randsol)
  best = randsol()
  bestenergy = energy(best)
  if bestenergy == 0
    return best, bestenergy
  end
  for i in 1:numloops
    s = randsol()
    e = energy(s)
    if e < bestenergy
      best = s
      bestenergy = e
    end
    if bestenergy == 0
      break
    end
  end
  best, bestenergy
end
```

Figure 4.1.6: *Random Sampling* – Generate random solutions until you find an optimal one or you run out of loops. Return the best.

The random search algorithm is somewhere between random sampling and simulated annealing. Like random sampling it always keeps a solution if it is at least as good as the best yet found. However, it generates is random solutions as neighbors

of the present best solution, like simulated annealing. Random search is *almost* simulated annealing with a step function for a transition probability

$$\mathtt{P}(e, e', T) = \begin{cases} 1 & e - e' >= 0 \\ 0 & e - e' < 0. \end{cases} \tag{4.1.4}$$

The major distinction is in the simulated annealing algorithm the neighbor is taken from the *current* solution rather than the global *best* solution. An implementation of random search follows.

```
function randomsearch(energy, numloops, randsol, neighbor)
  best = randsol()
  bestenergy = energy(best)
  if bestenergy == 0
    return best, bestenergy
  end
  for i in 1:numloops
    s = neighbor(best)
    e = energy(s)
    if e < bestenergy
      best = s
      bestenergy = e
    end
    if bestenergy == 0
      break
    end
  end
  best, bestenergy
end
```

Figure 4.1.7: *Random Search* – Generate random neighbors of the best solution found until you find an optimal one or you run out of loops. Return the best.

4.1.4 Comparison of Algorithms

We can now determine how effective simulated annealing (SA) actually is. To do this we will search specifically for groups that we know are in the landscape; after all, we have already built all of the gauge models! This will give us an idea of how well the algorithm finds groups that are there. We will compare the results to those for random sampling (RSa) and random search (RSe).

Each algorithm was run on the same target group with each completing a maximum of 1000 loops. Additionally we ran each algorithm on both the entropic and variance energy functions discussed in Figures 4.1.4 and 4.1.5, respectively.

We will focus on SO_{44}, $SO_{20} \otimes SO_{24}$, $(SU_2)^{22}$ and $E_7 \otimes E_7 \otimes E_8$, as we know they exist in the gauge landscape. These were chosen to illustrate the way in which the energy function favors certain features. In each case the four algorithms are run 100 times and the results collected. Three metrics are collected, the energy of the generated solution, whether the energy was minimized and whether the gauge group in question was actually generated. The results are presented in Tables 4.1.1 and Tables 4.1.2 for the entropic and variance energies, respectively. In both tables, the energy represents the average energy of the best solution produced by each algorithm in 1,000 trials. The *% Minimized* and *% Successful* columns depict the percentage of best models produced with the minimal energy and the target group, respectively.

Table 4.1.1: *Comparison of Algorithms* (Entropic Energy) – Results for searches for four gauge groups utilizing three search algorithms are presented. The algorithms, simulated annealing, random sampling and random search, are denoted by SA, RSa and RSe, respectively. Note that RSa outperformed both SA and RSe for group and metric.

Target	*Algorithm*	*Energy*	*%Minimized*	*%Successful*
SO_{44}	SA	3.774 ± 1.566	8	8
	RSa	2.162 ± 2.133	29	29
	RSe	3.79 ± 1.666	11	11
$SO_{20} \otimes SO_{24}$	SA	0.11 ± 0.363	81	15
	RSa	0.0 ± 0.0	100	42
	RSe	0.196 ± 0.492	77	13
$(SU_2)^{22}$	SA	7.501 ± 1.921	2	2
	RSa	5.771 ± 2.174	0	0
	RSe	7.768 ± 1.431	0	0
$E_7 \otimes E_7 \otimes E_8$	SA	0.12 ± 0.591	96	0
	RSa	0.0 ± 0.0	100	0
	RSe	0.27 ± 0.863	91	0

Considering the results of the entropic energy search, we see that simulated annealing is actually less effective than random sampling in every way. This is likely a consequence of two causes. First of all, the neighbor function inhibits exploration of the landscape, evidenced by the fact that both SA and RSe are comparable; the only thing they have in common is the use of `neighbor`. Second, we see that our energy function is not particularly effective at representing an optimal solution. In the case of SO_{44}, whenever `energy` is minimized the target is produced. However, in the case of $SO_{20} \otimes SO_{24}$ this is only true roughly 50 of the time and is never the case for the other two groups. We can conclude that, in combination, `neighbor` and `energy` are incompatible.

Table 4.1.2: *Comparison of Algorithms* (Variance Energy) – Results for searches for four gauge groups utilizing three search algorithms are presented. The algorithms, simulated annealing, random sampling and random search, are denoted by SA, RSa and RSe, respectively. Note that RSa outperformed both SA and RSe for group and metric.

Target	*Algorithm*	*Energy*	*ϱMinimized*	*ϱSuccessful*
SO_{44}	SA	0.023 ± 0.085	93	6
	RSa	0.0 ± 0.0	100	1
	RSe	0.053 ± 0.287	94	3
$SO_{20} \otimes SO_{24}$	SA	0.065 ± 0.075	56	36
	RSa	0.0 ± 0.0	100	98
	RSe	0.08 ± 0.073	45	35
$(SU_2)^{22}$	SA	0.017 ± 0.073	95	0
	RSa	0.0 ± 0.0	100	0
	RSe	0.03 ± 0.096	91	0
$E_7 \otimes E_7 \otimes E_8$	SA	0.037 ± 0.105	89	0
	RSa	0.0 ± 0.0	100	0
	RSe	0.077 ± 0.254	85	0

The results of the variance energy searches are fairly straightforward: the variance energy is not effective. For $SO_{20} \otimes SO_{24}$ the probability that the optimal solution has been found given that the energy has been minimized is quite high. This is not the case for the other three groups. In fact, for those three groups the chance that a minimal energy solution will be found is 85 or higher, but for

$SO_{20} \otimes SO_{24}$ the probability is on the order of 50. In all cases, the SA algorithm is outperformed by RSa.

These results give insight into the design of optimization algorithms in this domain. In particular, the choice of energy function is coupled to the gauge group. For example, if you are searching for $SO_{20} \otimes SO_{24}$ the variance energy is a decent option. The difficulty with this is that one cannot know how to design the energy function based solely on the group itself. We are left to conclude that this approach is largely ineffectual unless better `neighbor` and `energy` functions are designed.

We do not need to consider the results of searches for models that we know are absent from the landscape because the searches for models that we know are present fail. For this reason, and brevity, we do not consider the results here. For more information, see subsequent publications.

4.2 Genetic Algorithms

Genetic algorithms are some of the more interesting metaheuristic algorithms. A genetic algorithm (GA) is an algorithm in which the solutions are encoded in some nice manner, a population of random solutions are generated, and a series of *genetic operators* are applied to push the population toward high and higher *fitness*. This requires the definition of a fitness function, much like the energy function of simulated annealing.

GAs are modeled after the evolutionary process, so one might expect genetic operators representing mutation, selection and crossover. Mutation is a process that randomly selects an individual and modifies it slightly. The probability of this occurring is typically chosen to be very small, otherwise the fitness of the population fluctuates rapidly. Selection is a process of selecting the most fit individuals of the population for crossover, and crossover is the act of mixing the genetic representation of the solutions. Selection rates, the percentage of the population that is select

during each cycle, vary based on the form of the crossover. There are many types of crossover, but it will not be of interest to go into them.

Much as in the case of simulated annealing, modular invariance makes devising a genetic algorithm for landscape surveys difficult. While in SA modular invariance only appears in the random generation of neighbors, it interferes in three places: mutation, crossover, and population generation. Population generation is no more difficult than that dealt with in SA. Mutation and crossover are not so simple. How does one randomly mutate a basis vector without breaking modular invariance? What's more, how might one mix two basis vectors together in crossover to produce a single modular invariant basis vector?

The author has no answer to these questions as of this writing, but research into possibilities is underway.

4.3 Summary

We have found that simulated annealing is not likely to prove a viable landscape search technique in the immediate future. Obstacles of this approach include the definition of an effective energy function, `energy`, and the selection of a neighbor function, `neighbor`, that can produce nearby basis vectors without discarding modular invariance. In fact, the proposed functions, `energy` and `neighbor`, actually inhibit the computation of a optimal solution.

Genetic algorithms are more severely afflicted as the choice of the mutation and crossover genetic operators are likely to fail to produce modularly invariant solutions.

The author's hope is that this line of inquiry will be further explored in subsequent publications.

APPENDICES

APPENDIX A

The Gauge Framework

This appendix discusses the structure, form and function of the Gauge Framework. This is largely a discussion of the basics of the Gauge Framework; documentation will be made available with the repository at the EUCOS github page, `https://github.com/EUCOS`.

A.1 Philosophy and Influences

When the Gauge Framework was first written it was entirely object-oriented. This was the style that I, the author, first learned and what was preferred by the group at the time. As development progressed I grew more disenchanted by the object-oriented paradigm. In particular, as the second version was being written I noticed that by making some of the data members public and removing the associated accessors we gained a roughly 40% speed improvement. That amounted to approximately 1,000 models/second as opposed to 700 models/second. This, together with my learning `Haskell` drove me to use more functional and procedural techniques when it was appropriate. While all programming paradigms are technically equivalent[1], they are not all equal. Object oriented design is very good at representing data and data structures. We have used these features extensively in creating classes found in the `include/Datatypes` directory. However, processes and transformations on that data are often better represented as a functional and thus are more naturally represented by a procedural or functional paradigm. These also appear from time to time throughout the project. All of this said, it is important to realize that the project is still evolving, and there are several places were a class could be replaced by a collection of functions or even the other way around.

The most important detail for this project is that it is *correct*. We need to be sure that the computed statistics are true. Another aspect of this is reliability; we need to be able to start a program running and return two weeks later to find that

[1] There is no program that can be written procedurally that cannot be written in an object-oriented fashion, and vice versa.

it is still running[2]. The are both facets of good design. To facilitate the vetting process I've implemented the test-driven design principles: write a test, write the code until the test passes, and repeat. This has a limit; some aspects of model building are long-running. We cannot reasonably devise tests for the systematic surveys on the whole, but we can write tests for the smaller parts. This is where functional and procedural programming shine.

After reliability is speed. This project needs to be as fast as it reasonably can be. To make this work, we've used MPI for parallelism and implemented many of the performance critical algorithms in low-level C. This seems to have done the trick as the Gauge Framework can build anywhere from 1,000 to 120,000 models per second on Baylor's 128-node beowulf cluster.

Finally we want the Gauge Framework to be *used*. What is the good in writing software that no one uses? To that end I have tried to make everything as user friendly as possible while still allowing a developer low-level access to the framework. You be the judge as to whether I've succeeded here.

A.2 Structure

We now discuss the structure of the Gauge Framework. There are three aspects to the structure. First is the obvious directory and file structure. Where do the various files belong and can you tell the purpose of a file based on its location? Next is the object structure. How does inheritance work for the framework; how interconnected are the components? Finally there is the aspect that is most important to the user, the survey structure. How are surveys structured and run?

A.2.1 Directory and File Structure

[2] Or has terminated successfully, of course.

The Gauge Framework has, by default, four directories in its root: `cmd`, `tests`, `include` and `src`. Additional, there are two relevant files `Doxyfile` and `Makefile`. We will use this subsection to discuss each of these and their substructures.

We will start with the `cmd` directory. This directory contains all of the main implementation files, i.e. files with main methods that can be compiled into executable programs. It has any number of subdirectories, each of which contains at least one file: `main.cpp`. The name of the subdirectory determines the name of the executable upon building. For example, the file `cmd/layer-one/main.cpp` would compile to a binary file `bin/layer-one`. Executable binaries are built and placed in `bin` which is found in the project root directory.

Next consider `tests`. Within `tests` one finds three subdirectories, `gtest`, `include` and `src`. The `gtest` subdirectory contains all of the source for a version of Google's testing framework, Google Test. Details regarding Google Test can be found at `http://code.google.com/p/googletest/`. This framework is used to provided testing facilities to the Gauge Framework. The other directories `include` and `src` contain the actual tests for the framework. When a file of the form `<Name>Test.cpp` is created in the `tests/src` directory, it can be compiled into an executable. This executable will also be found in `tests/src`. Header files for the tests can be placed in `tests/include`.

The directories `include` and `src` go hand in hand; `include` is for API headers and `src` is for implementations. Within `include` one finds four directories. The first is `Datatypes`. This subdirectory contains the declarations of the various datatypes of the framework, e.g. `Gauge::Vector, Gauge::BasisVector, Gauge::GSOMatrix, Gauge::Inputs`, etc. The next subdirectory of interest is the `Interfaces` directory. Most objects in the framework have the capacity to be printed and serialized. Rather than reimplement all of the features necessary for each for every class, we abstract the details into interfaces. These interfaces are

found in `include/Interfaces`; there is no analogous directory in `src` because interfaces are abstract classes. Now, as models are constructed we must perform various processing activities on them. Since there is no way for the developers to plan for every such processing pattern, the Gauge Framework has a *processor system*. The ideas is that the end user can create a processor and add it to a chain of processors through which models can be passed as they are build. The public APIs for these processors are found in `include/Processor` and the implementation in `src/Processor`. Finally, the `Utility` subdirectory contains various utilities used through the framework. Presently we only have a single collection of utility functions for creating filesystem directories. This may be extended in the future.

As mentioned there are two files found in the project root, `Doxyfile` and `Makefile`. The former is a configuration file for `Doxygen`[3], a program that reads commented source code and generates API documentation in HTML, LaTeX, etc. The `Makefile` is the recipe used by the build tool `make` to compile, link and archive the Gauge Framework library and its executables. When run, `Doxygen` creates a single directory structure, rooted at `share`, filled will all of the generated documentation. When the project is built, three directories are created: `bin` contains binary executables, `obj` contains binary object files[4], and `lib` contains a static library, `libgauge.a`, against which projects may be linked.

[3] `http://www.stack.nl/ dimitri/doxygen/`

[4] For those unfamiliar with the C/C++ build process, object files are the result of compiling a source file, but not linking it. This means it is not executable.

A.2.2 Object Structure

The object structure of the Gauge Framework is pretty simple. There is very little inheritance in general; rather than going through and listing all of the different lines of inheritance, we will only discuss a few of the major classes and examples.

Let us start with the `Vector`-family. The `Vector` datatype, declared in `include/Datatypes/Vector.h`, represents the structure of each of the three main types of phase vectors in the project: `BasisVector`, `Sector` and `State`. These are each an array of rational number, so they are all represented by an array of numerators and a common denominator. An distilled version of the `Gauge::Vector` struct is provided in Figure A.2.1.

```
namespace Gauge {
  struct Vector : public Printable, public Serializable {
    int *numerators; // The numerators of the vector
    int denominator; // The common denominator
    size_t size;     // The size of the vector (26 - D)

    ⟨other data members⟩

    ⟨constructors⟩
    ⟨operators⟩

    // Printable Interface
    virtual void PrintTo(std::ostream *out) const;

    // Serializable Interface
    virtual void SerializeWith(Serializer *serializer) const;
    virtual void DeserializeWith(Serializer *serializer);
  };

  ⟨template functions on vectors, e.g. dot products, summation, etc.⟩
}
```

Figure A.2.1: `Gauge::Vector` – A simplified implementation for the Vector struct. Note that the basic data members, `numerators`, `denominator` and `size` are defining characteristics of all phase vectors. For this reason `Gauge::BasisVector`, `Gauge::Sector` and `Gauge::State` each derive from `Gauge::Vector`.

The `Gauge::Vector` class typifies a typical inheritance of the interfaces, `Gauge::Printable` and `Gauge::Serializable`. These classes are inherited by all of the Gauge Framework's datatypes. The former facilitates printing to screen and

the latter provides a mechanism for efficiently converting an object to a binary encoded string[5]. `Gauge::Serializable` is also inherited by several of the working classes, e.g. `Gauge::ModelFactory`.

Our final example of inheritance is of the `Processor` class. To ensure that end-user processors have the correct machinery to run, at least in principle, and to ensure that we can them all in a single list, all processors inherit from `Gauge::Processor`. The difficulty with the processor class is the `Merge` method. As we construct models in parallel, discussed in Subsection A.2.3, each model builder will have its own set of processors. Once the building is completed, we must merge those processors into one for consolidated handling, hence the `Merge` method.

A.2.3 Survey Structure

Gauge surveys come in two varieties serial and parallel. In both cases a `Gauge::GeometryFactory` object is constructed and used to facilitate the generation of `Gauge::Geometry`s, $\{\mathbf{A}, \mathbf{k}\}$. The geometries are then passed on to a `Gauge::ModelFactory` object which constructs the associated `Gauge::Model`. After creating the model it is pushed through a series of `Gauge::Processor`s. After all processing has finished the program terminates. The distinction between serial and parallel amounts to the topology of the processes involved. In fact, the serial surveys have only one process.

Serial surveys follow the above description. Throughout its life cycle, only a single computing process exists. Control is passed from object to object as the algorithm progresses. Figure A.2.3 depicts the a serial process.

[5] This makes such luxuries as check-pointing and message passing possible.

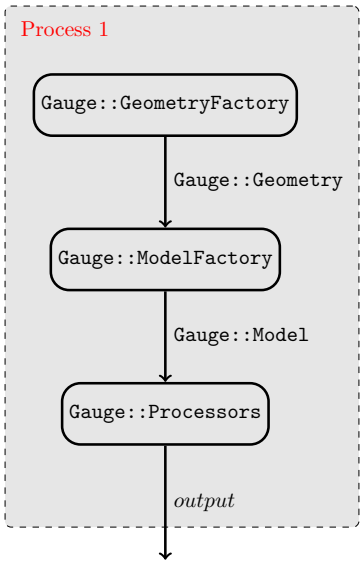

Figure A.2.2: *Serial Topology* – Each gray rectangle represents an operating system process. Within each we find at least one algorithmic objects, in rounded rectangles, and lines of data flow marked by the type of the data. For example, within Process 1, `Gauge::GeometryFactory` and `Gauge::ModelFactory` exchange a `Gauge::Geometry`.

On the other hand, a parallel survey will have many concurrently running algorithms. Each of which typically has at least one algorithmic object and the processes themselves exchange data via the standard Message Passing Interface (MPI). Because the `Gauge::GeometryFactory` can construct the `Gauge::Geometry` objects much more quickly, approximately 10^5 times faster, we need only run one and distribute the outputs to each of the *slave* processes. The slaves then construct the model and process it. Once the `GeometryFactory` has terminated, the root process requests that each of the slaves send their `Gauge::Processor`s to back to the root. Once received, the processors are merged together and finalized. Output is then issued from the root process, and the job terminates.

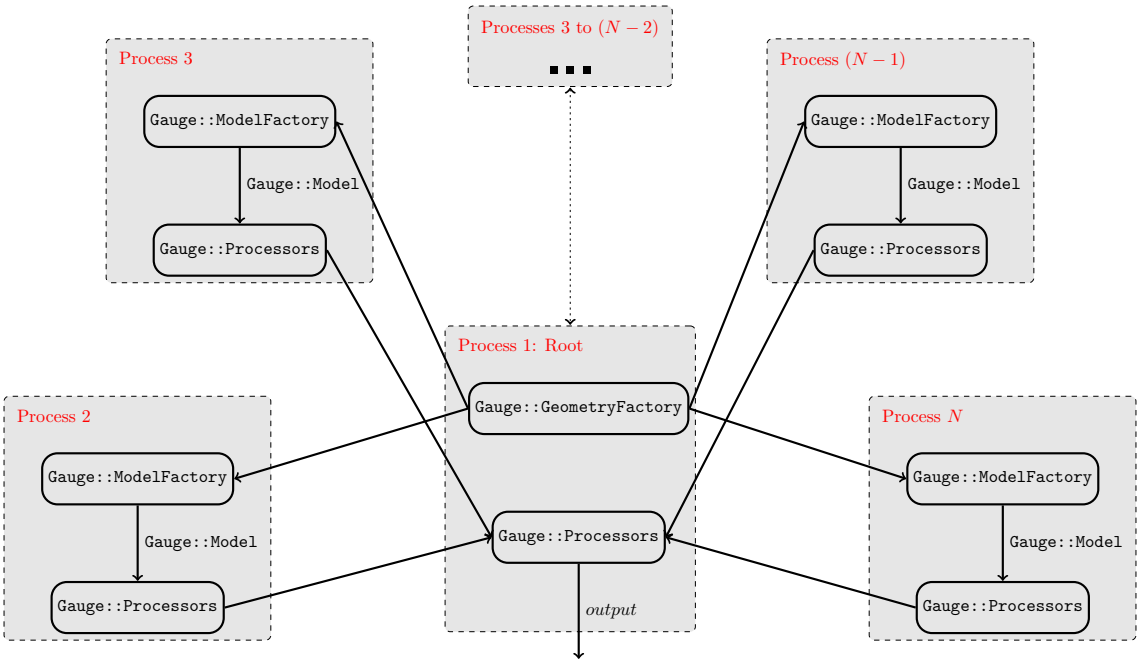

Figure A.2.3: *Parallel Topology* – Each gray rectangle represents an operating system process. Within each we find at least one algorithmic objects, in rounded rectangles, and lines of data flow marked by the type of the data.

Now that we know the basic structure of the framework, let us give a brief presentation of how to run a survey.

A.3 Usage

The Gauge Framework has abstracted much away from the user. In order to run a standard survey, say "construct all models from orders 2×2 to 5×5", we need only make a function call! It does not matter whether one is running in series or parallel, the process is basically the same; the main difference is the function that is executed. Below are two example implementations, Figure A.3.4 and Figure A.3.5, which run a survey in serial and parallel, respectively.

```
#include <Processor/ByGroup.h>
#include <Survey.h>
#include <Utility.h>

using namespace std;

int main(int argc, char **argv) {
  const int D = 4, L = 2;
  const int lower[] = {2,2}, upper[] = {5,5};
  auto susytype = Gauge::Input::kSUSY;

  const string root = "results/L=" + to_string(L) + "/";
  const string output = root + "D=" + to_string(D) + "/";
  const string log_file = root + "D=" + to_string(D) + ".log";
  Utility::Dir::Create(root_dir);

  Gauge::Survey::Serial(
    // Processors
    { new Gauge::Process::ByGroup(output, false) },

    // Geometry Factory
    Gauge::GeometryFactory::SystematicFactory(),

    // Input Factory
    new Gauge::InputFactory::Range(lower, upper, L, D, susytype),

    // Log File
    log_file
  );

  return 0;
}
```

Figure A.3.4: *Serial Survey* – An example demonstrating how to run a gauge survey in serial.

```
#include <Processor/ByGroup.h>
#include <Survey.h>
#include <Utility.h>

using namespace std;

int main(int argc, char **argv) {
  const int D = 4, L = 2;
  const int lower[] = {2,2}, upper[] = {5,5};
  auto susytype = Gauge::Input::kSUSY;

  const string root = "results/L=" + to_string(L) + "/";
  const string output = root + "D=" + to_string(D) + "/";
  const string log_file = root + "D=" + to_string(D) + ".log";
  Utility::Dir::Create(root_dir);

  Gauge::Survey::Parallel(
    // Required for MPI
    argc, argv,

    // Processors
    { new Gauge::Process::ByGroup(output, false) },

    // Geometry Factory,
    Gauge::GeometryFactory::SystematicFactory(),

    // Input Factory
    new Gauge::InputFactory::Range(lower, upper, L, D, susytype),

    // Log File
    log_file
  );

  return 0;
}
```

Figure A.3.5: *Parallel Survey* – An example demonstrating how to run a gauge survey in parallel.

APPENDIX B

Layer One Statistics

Table B.0.3: $\mathcal{N} = 4$ *Gauge Group Combinations* – The percentage of all unique $\mathcal{N} = 4$ models with each combination of gauge groups is tabulated. For example, 11.76% of the 68 unique SUSY models have the combination $SU_4 \otimes U_1$ at least once.

$N = 68$	U_1	SU_2	SU_3	SU_4	SU_5	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$	$\mathcal{F}SU_5$	G_{PS}	G_{LRS}	G_{RSM}	G_{SM}
U_1	29.41	11.76	4.41	11.76	5.88	36.76	4.41	7.35	11.76	5.88	1.47	5.88	2.94	0	0	0
SU_2	–	29.41	0	7.35	0	32.35	11.76	7.35	25.00	10.29	0	0	1.47	0	0	0
SU_3	–	–	2.94	0	1.47	2.94	0	0	0	0	0	1.47	0	0	0	0
SU_4	–	–	–	5.88	0	11.76	1.47	4.41	1.47	2.94	1.47	0	2.94	0	0	0
SU_5	–	–	–	–	4.41	2.94	0	0	0	0	0	4.41	0	0	0	0
$SU_{N>5}$	–	–	–	–	–	33.82	7.35	12.24	16.18	8.82	1.47	2.94	4.41	0	0	0
SO_8	–	–	–	–	–	–	4.41	0	5.88	1.47	0	0	0	0	0	0
SO_{10}	–	–	–	–	–	–	–	2.94	0	2.94	0	0	0	0	0	0
$SU_{N>10}$	–	–	–	–	–	–	–	–	17.65	13.24	0	0	0	0	0	0
E_N	–	–	–	–	–	–	–	–	–	11.76	1.47	0	1.47	0	0	0
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	–	–	0	0	0	0	0	0
$\mathcal{F}SU_5$	–	–	–	–	–	–	–	–	–	–	–	4.41	0	0	0	0
G_{PS}	–	–	–	–	–	–	–	–	–	–	–	–	0	0	0	0
G_{LRS}	–	–	–	–	–	–	–	–	–	–	–	–	–	0	0	0
G_{RSM}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0	0
G_{SM}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0
Total	45.58	44.12	4.41	16.18	5.88	54.41	16.18	13.24	44.12	23.53	2.94	5.88	5.88	0	0	0

Table B.1.4: $\mathcal{N}=0$ *Gauge Group Combinations* – The percentage of all unique $\mathcal{N}=0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 502 $\mathcal{N}=0$ models, i.e. $SO_{10}\otimes SU_5$ occurs in 1.00% of these 502 models.

$N=68$	U_1	SU_2	SU_3	SU_4	SU_5	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6\otimes E_6$	$\mathcal{F}SU_5$	G_{PS}	G_{LRS}	G_{RSM}	G_{SM}
U_1	75.70	49.80	25.30	40.64	20.92	63.75	11.75	13.35	10.36	10.36	1.39	20.92	15.54	8.76	14.74	14.74
SU_2	–	42.63	14.74	24.50	12.35	39.24	12.55	8.17	12.75	9.96	1.00	12.35	9.36	4.78	8.76	8.76
SU_3	–	–	13.94	11.95	10.76	15.74	1.00	1.20	0	0.60	0	10.76	3.78	6.57	8.96	8.96
SU_4	–	–	–	19.12	9.56	28.09	6.37	5.98	4.18	3.98	0.60	9.56	8.37	3.78	7.17	7.17
SU_5	–	–	–	–	8.76	12.35	0.80	1.00	0	0.60	0	8.76	2.59	4.38	7.17	7.17
$SU_{N>5}$	–	–	–	–	–	31.27	9.76	10.16	9.56	7.97	0.40	12.35	9.36	3.00	7.17	7.17
SO_8	–	–	–	–	–	–	4.58	1.79	4.98	3.78	0	0.80	2.39	0	0	0
SO_{10}	–	–	–	–	–	–	–	2.59	1.20	2.19	0.40	1.00	2.39	0	0	0
$SU_{N>10}$	–	–	–	–	–	–	–	–	5.98	5.38	0.20	0	0.40	0	0	0
E_N	–	–	–	–	–	–	–	–	–	3.78	0.20	0.60	1.59	0	0	0
$E_6\otimes E_6$	–	–	–	–	–	–	–	–	–	–	0	0	0.20	0	0	0
$\mathcal{F}SU_5$	–	–	–	–	–	–	–	–	–	–	–	8.76	2.59	4.38	7.17	7.17
G_{PS}	–	–	–	–	–	–	–	–	–	–	–	–	2.99	0.80	1.99	1.99
G_{LRS}	–	–	–	–	–	–	–	–	–	–	–	–	–	1.99	3.78	3.78
G_{RSM}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6.57	6.57
G_{SM}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6.57
Total	83.67	62.55	25.30	41.83	20.92	66.53	19.32	13.94	21.51	15.54	1.39	20.92	16.33	8.76	14.74	14.74

Table B.1.5: *Statistics of* $D = 4$, $\mathcal{N} = 0$ *Models* – The percentage of all unique $\mathcal{N} = 0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 509 $\mathcal{N} = 0$ models, i.e. $SO_{10} \otimes SU_5$ occurs in 0.98% of these 509 models.

$N = 509$	U_1	SU_2	SU_3	SU_4	SU_5	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$	$\mathcal{F}SU_5$	G_{PS}	G_{LRS}	G_{SM}
U_1	75.83	50.10	25.74	40.47	20.63	62.87	11.59	13.16	10.22	10.02	1.38	20.63	15.72	9.23	15.13
SU_2	–	43.22	15.13	24.56	12.18	38.70	12.57	8.06	12.77	9.43	0.98	12.18	9.63	5.30	9.23
SU_3	–	–	14.54	11.98	10.61	15.52	0.98	1.18	0	0.59	0	10.61	3.93	7.07	9.43
SU_4	–	–	–	18.86	9.43	27.70	6.29	5.89	4.13	3.93	0.59	9.43	8.25	3.93	7.27
SU_5	–	–	–	–	8.64	12.18	0.79	0.98	0	0.59	0	8.64	2.55	4.32	7.07
$SU_{N>5}$	–	–	–	–	–	30.84	9.63	10.02	8.64	7.86	0.39	12.18	9.23	2.95	7.07
SO_8	–	–	–	–	–	–	4.52	1.77	5.11	3.54	0	0.79	2.36	0	0
SO_{10}	–	–	–	–	–	–	–	2.55	1.18	2.16	0.39	0.98	2.36	0	0
$SU_{N>10}$	–	–	–	–	–	–	–	–	5.89	5.30	0.20	0	0.39	0	0
E_N	–	–	–	–	–	–	–	–	–	3.14	0.20	0.59	1.57	0	0
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	–	–	0	0	0.20	0	0
$\mathcal{F}SU_5$	–	–	–	–	–	–	–	–	–	–	–	8.64	2.55	4.32	7.07
G_{PS}	–	–	–	–	–	–	–	–	–	–	–	–	2.95	0.98	2.16
G_{LRS}	–	–	–	–	–	–	–	–	–	–	–	–	–	2.55	4.32
G_{RSM}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	7.07
G_{SM}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	7.07
Total	83.69	62.87	25.74	41.65	20.63	65.62	19.25	13.75	21.41	15.32	1.38	20.63	16.50	9.23	15.13

B.2 Layer One Statistics in D Dimensions

Herein we present statistics for occurrence of specific group factors in various combinations across the layer 1 landscape. As well as combinations of two group factors, we look at combinations of specific compound factors in conjunction with single and other compound factors. Following [60] we include such compound factors as $E_6 \otimes E_6$, $G_{PS} \equiv SU_4 \otimes SU_2 \otimes SU_2$ (Pati-Salam), $G_{LRS} \equiv SU_3 \otimes SU_2 \otimes SU_2$ (Left-Right Symmetric), and $G_{SM} \equiv SU_3 \otimes SU_2 \otimes U_1$ (Standard Model). We also include $\mathcal{F}SU_5 \equiv SU_5 \otimes U_1$, though, because we are not considering matter content, we can only say that the model has the $\mathcal{F}SU_5$ gauge group; it may not actually be $\mathcal{F}SU_5$.

The percentage of all unique $\mathcal{N}_{\max}$ and $\mathcal{N} = 0$ models in $D = 10$ through $D = 4$ with each combination of gauge groups is tabulated. As an example, 11.76% of the 68 unique $\mathcal{N} = 4$ models, Table B.1.5, have the combination $SU_4 \otimes U_1$ at least once. Note that since the $D = 4$ results have already been presented, Tables B.0.3 and B.1.5.

B.2.1 Maximally Supersymmetric Models

Table B.2.6: *Statistics of* $D = 10$, $\mathcal{N} = 1$ *Models* – The percentage of all unique $\mathcal{N} = 1$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 2 $\mathcal{N} = 1$ models.

$N = 2$	$SU_{N>10}$	E_N
$SU_{N>10}$	0	0
E_N	–	50.00
Total	50.00	50.00

Table B.2.7: *Statistics of* $D = 9$, $\mathcal{N} = \mathcal{N}_{\max}$ *Models* – The percentage of all unique $\mathcal{N} = \mathcal{N}_{\max}$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 9 $\mathcal{N} = \mathcal{N}_{\max}$ models, i.e. $SU_2 \otimes SU_2$ occurs in 0.98% of these 9 models.

$N = 9$	U_1	SU_2	SU_4	$SU_{N>5}$	SO_{10}	$SU_{N>10}$	E_N
U_1	0	0	0	0	0	22.22	11.11
SU_2	–	11.11	0	11.11	0	0	0
SU_4	–	–	0	0	0	0	11.11
$SU_{N>5}$	–	–	–	0	0	0	11.11
SO_{10}	–	–	–	–	0	11.11	0
$SU_{N>10}$	–	–	–	–	–	11.11	11.11
E_N	–	–	–	–	–	–	22.22
Total	33.33	11.11	11.11	22.22	11.11	55.56	44.44

Table B.2.8: *Statistics of* $D = 8$, $\mathcal{N} = 1$ *Models* – The percentage of all unique $\mathcal{N} = \mathcal{N}_{\max}$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 13 $\mathcal{N} = \mathcal{N}_{\max}$ models, i.e. 38.46% of these 13 models have at least one E_N factor.

$N = 13$	U_1	SU_2	SU_4	$SU_{N>5}$	SO_8	$SU_{N>10}$	E_N
U_1	0	0	0	7.69	0	0	7.69
SU_2	–	23.08	0	7.69	0	23.08	15.38
SU_4	–	–	0	7.69	0	0	0
$SU_{N>5}$	–	–	–	7.69	0	0	7.69
SO_8	–	–	–	–	0	0	7.69
$SU_{N>10}$	–	–	–	–	–	23.08	15.38
E_N	–	–	–	–	–	–	15.38
Total	7.69	38.46	7.69	30.77	7.69	53.85	38.46

Table B.2.9: *Statistics of $D = 7$, $\mathcal{N} = \mathcal{N}_{\max}$ Models* – The percentage of all unique $\mathcal{N} = \mathcal{N}_{\max}$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 16 $\mathcal{N} = \mathcal{N}_{\max}$ models, i.e. $SU_2 \otimes U_1$ appears in 12.50% of these 16 models.

$N = 16$	U_1	SU_2	SU_4	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$
U_1	0	12.50	0	18.75	0	0	18.75	12.50	6.25
SU_2	–	6.25	0	12.50	0	0	6.25	12.50	0
SU_4	–	–	0	0	0	0	12.50	6.25	0
$SU_{N>5}$	–	–	–	12.50	6.25	6.25	6.25	6.25	0
SO_8	–	–	–	–	0	0	0	0	0
SO_{10}	–	–	–	–	–	0	0	6.25	0
$SU_{N>10}$	–	–	–	–	–	–	18.75	12.50	0
E_N	–	–	–	–	–	–	–	18.75	6.25
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	0
Total	37.50	18.75	18.75	37.50	6.25	12.50	50.00	37.50	6.25

Table B.2.10: *Statistics of $D = 6$, $\mathcal{N} = 2$ Models* – The percentage of all unique $\mathcal{N} = 2$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 18 $\mathcal{N} = 2$ models, i.e. $SO_8 \otimes SO_8$ appears in 5.56% of these 18 models.

$N = 18$	U_1	SU_2	SU_4	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$
U_1	11.11	0	0	11.11	0	5.56	5.56	5.56	5.56
SU_2	–	22.22	0	11.11	0	0	11.11	5.56	0
SU_4	–	–	0	5.56	0	0	0	5.56	0
$SU_{N>5}$	–	–	–	16.67	0	11.11	5.56	5.56	0
SO_8	–	–	–	–	5.56	0	11.11	5.56	0
SO_{10}	–	–	–	–	–	0	0	0	0
$SU_{N>10}$	–	–	–	–	–	–	16.67	16.67	0
E_N	–	–	–	–	–	–	–	16.67	5.56
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	0
Total	16.67	22.22	5.56	38.89	22.22	11.11	50.00	33.33	5.56

Table B.2.11: *Statistics of* $D = 5$, $\mathcal{N} = \mathcal{N}_{\max}$ *Models* – The percentage of all unique $\mathcal{N} = \mathcal{N}_{\max}$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 40 $\mathcal{N} = \mathcal{N}_{\max}$ models, i.e. $SO_8 \otimes SO_8$ appears in 2.50% of these 40 models.

$N = 40$	U_1	SU_2	SU_3	SU_4	SU_5	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$	$\mathcal{F}SU_5$
U_1	2.50	12.50	2.50	5.00	5.00	40.00	7.50	7.50	25.00	15.00	2.50	0
SU_2	–	12.50	0	5.00	0	12.50	2.50	2.50	7.50	5.00	2.50	0
SU_3	–	–	0	0	0	2.50	0	0	0	0	0	0
SU_4	–	–	–	2.50	0	10.00	0	2.50	5.00	5.00	0	0
SU_5	–	–	–	–	2.50	2.50	0	0	0	0	0	2.50
$SU_{N>5}$	–	–	–	–	–	25.00	5.00	10.00	12.50	10.00	0	2.50
SO_8	–	–	–	–	–	–	2.50	0	2.50	2.50	0	0
SO_{10}	–	–	–	–	–	–	–	2.50	5.00	2.50	0	0
$SU_{N>10}$	–	–	–	–	–	–	–	–	15.00	15.00	0	0
E_N	–	–	–	–	–	–	–	–	–	10.00	2.50	0
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	–	–	0	0
$\mathcal{F}SU_5$	–	–	–	–	–	–	–	–	–	–	–	0
Total	62.50	20.00	2.50	17.50	5.00	52.50	10.00	17.50	47.50	27.50	2.50	5.00

Table B.2.12: *Statistics of $D = 10$, $\mathcal{N} = 0$ Models* – The percentage of all unique $\mathcal{N} = 0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 6 $\mathcal{N} = 0$ models, i.e. $SU_2 \otimes SU_2$ appears in 16.67% of these 6 models.

$N = 6$	U_1	SU_2	$SU_{N>5}$	SO_8	$SU_{N>10}$	E_N
U_1	0	0	16.67	0	0	0
SU_2	–	16.67	0	0	0	16.67
$SU_{N>5}$	–	–	0	0	0	0
SO_8	–	–	–	0	16.67	0
SO_{10}	–	–	–	–	0	0
$SU_{N>10}$	–	–	–	–	16.67	16.67
E_N	–	–	–	–	–	16.67
Total	16.67	16.67	16.67	16.67	66.67	33.33

Table B.2.13: *Statistics of $D = 9$, $\mathcal{N} = 0$ Models* – The percentage of all unique $\mathcal{N} = 0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 32 $\mathcal{N} = 0$ models, i.e. $SO_{10} \otimes SU_4$ appears in 3.13% of these 32 models.

$N = 32$	U_1	SU_2	SU_4	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$	G_{PS}
U_1	21.88	28.13	12.50	40.63	12.50	6.25	25.00	15.63	3.13	3.13
SU_2	–	34.38	12.50	15.63	0	0	18.75	15.63	3.13	3.13
SU_4	–	–	0	12.50	0	3.13	9.38	6.25	0	0
$SU_{N>5}$	–	–	–	12.50	3.13	6.25	3.13	3.13	0	3.13
SO_8	–	–	–	–	6.25	6.25	12.50	3.13	0	0
SO_{10}	–	–	–	–	–	0	6.25	3.13	0	0
$SU_{N>10}$	–	–	–	–	–	–	12.50	12.50	0	6.25
E_N	–	–	–	–	–	–	–	6.25	0	0
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	0	0
G_{PS}	–	–	–	–	–	–	–	–	–	0
Total	71.88	40.63	21.88	40.63	21.88	15.63	50.00	25.00	3.13	9.38

Table B.2.14: *Statistics of $D = 8$, $\mathcal{N} = 0$ Models* – The percentage of all unique $\mathcal{N} = 0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 50 $\mathcal{N} = 0$ models, i.e. $SU_4 \otimes SU_4$ appears in 4.00% of these 50 models.

$N = 50$	U_1	SU_2	SU_4	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$	G_{PS}
U_1	32.00	24.00	20.00	44.00	0	8.00	2.00	8.00	2.00	2.00
SU_2	–	46.00	6.00	24.00	16.00	2.00	32.00	16.00	0	0
SU_4	–	–	4.00	18.00	0	2.00	0	4.00	2.00	0
$SU_{N>5}$	–	–	–	22.00	0	8.00	2.00	6.00	0	2.00
SO_8	–	–	–	–	8.00	0	18.00	6.00	0	0
SO_{10}	–	–	–	–	–	0	0	0	0	0
$SU_{N>10}$	–	–	–	–	–	–	18.00	12.00	0	0
E_N	–	–	–	–	–	–	–	8.00	0	0
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	0	0
G_{PS}	–	–	–	–	–	–	–	–	–	0
Total	46.00	64.00	20.00	44.00	24.00	8.00	46.00	26.00	2.00	2.00

Table B.2.15: *Statistics of $D = 7$, $\mathcal{N} = 0$ Models* – The percentage of all unique $\mathcal{N} = 0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 85 $\mathcal{N} = 0$ models, i.e. $SO_{10} \otimes SU_4$ appears in 3.53% of these 85 models.

$N = 85$	U_1	SU_2	SU_3	SU_4	SU_5	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$	$\mathcal{F}SU_5$	G_{PS}
U_1	45.88	38.82	4.71	20.00	3.53	51.76	12.94	12.94	18.82	17.65	2.35	3.53	4.71
SU_2	–	36.47	0	10.59	0	28.24	5.88	9.41	18.82	14.12	0	0	4.71
SU_3	–	–	1.18	0	2.35	3.53	0	0	0	0	0	2.35	0
SU_4	–	–	–	8.24	0	17.65	7.06	3.53	9.41	4.71	0	0	1.18
SU_5	–	–	–	–	2.35	2.35	0	0	0	0	0	2.35	0
$SU_{N>5}$	–	–	–	–	–	25.88	5.88	9.41	7.06	7.06	0	2.35	4.71
SO_8	–	–	–	–	–	–	4.71	0	9.41	4.71	1.18	0	0
SO_{10}	–	–	–	–	–	–	–	2.35	3.53	3.53	1.18	0	1.18
$SU_{N>10}$	–	–	–	–	–	–	–	–	11.76	10.59	0	0	2.35
E_N	–	–	–	–	–	–	–	–	–	5.88	0	0	3.53
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	–	–	0	0	0
$\mathcal{F}SU_5$	–	–	–	–	–	–	–	–	–	–	–	2.35	0
G_{PS}	–	–	–	–	–	–	–	–	–	–	–	–	0
Total	75.29	48.24	4.71	30.59	3.53	55.29	21.18	17.65	37.65	24.71	2.35	3.53	8.24

Table B.2.16: *Statistics of $D = 6$, $\mathcal{N} = 0$ Models* – The percentage of all unique $\mathcal{N} = 0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 73 $\mathcal{N} = 0$ models, i.e. $SU_3 \otimes SU_5$ appears in 4.11% of these 73 models.

$N = 73$	U_1	SU_2	SU_3	SU_4	SU_5	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$	$\mathcal{F}SU_5$	G_{PS}
U_1	45.21	23.29	6.85	20.55	5.48	45.21	8.22	12.33	6.85	10.96	2.74	5.48	8.22
SU_2	–	41.10	0	9.59	0	24.66	9.59	2.74	15.07	10.96	0	0	5.48
SU_3	–	–	2.74	0	4.11	4.11	0	0	0	0	0	4.11	0
SU_4	–	–	–	10.96	0	17.81	4.11	4.11	1.37	2.74	0	0	4.11
SU_5	–	–	–	–	2.74	2.74	0	0	0	0	0	2.74	0
$SU_{N>5}$	–	–	–	–	–	24.66	6.85	8.22	5.48	6.85	0	2.74	6.85
SO_8	–	–	–	–	–	–	6.85	0	10.96	5.48	0	0	0
SO_{10}	–	–	–	–	–	–	–	2.74	0	2.74	1.37	0	0
$SU_{N>10}$	–	–	–	–	–	–	–	–	9.59	10.96	0	0	0
E_N	–	–	–	–	–	–	–	–	–	8.22	1.37	0	0
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	–	–	0	0	0
$\mathcal{F}SU_5$	–	–	–	–	–	–	–	–	–	–	–	2.74	0
G_{PS}	–	–	–	–	–	–	–	–	–	–	–	–	1.37
Total	58.90	45.21	6.85	21.92	5.48	52.05	26.03	13.70	32.88	26.03	2.74	5.48	8.22

Table B.2.17: *Statistics of $D = 5$, $\mathcal{N} = 0$ Models* – The percentage of all unique $\mathcal{N} = 0$ models with each combination of gauge groups is tabulated. Here each value is calculated against the 292 $\mathcal{N} = 0$ models, i.e. $SU_3 \otimes SU_2 \otimes U_1$ appears in 7.53% of these 292 models.

$N = 292$	U_1	SU_2	SU_3	SU_4	SU_5	$SU_{N>5}$	SO_8	SO_{10}	$SU_{N>10}$	E_N	$E_6 \otimes E_6$	$\mathcal{F}SU_5$	G_{PS}	G_{LRS}	G_{SM}
U_1	68.15	47.60	15.07	35.96	15.75	63.36	16.10	13.70	18.15	14.04	1.71	15.75	13.70	4.45	7.53
SU_2	–	41.78	7.53	22.95	6.85	36.99	9.59	7.88	13.36	8.90	0.68	6.85	9.25	1.03	4.45
SU_3	–	–	9.25	4.79	5.82	10.27	1.03	1.37	0	1.03	0	5.82	2.05	2.74	4.45
SU_4	–	–	–	16.44	6.51	26.03	7.53	5.14	7.19	4.45	0	6.51	7.19	2.05	3.42
SU_5	–	–	–	–	6.51	10.62	0.68	0.68	0	0.34	0	6.51	1.71	2.05	2.74
$SU_{N>5}$	–	–	–	–	–	29.79	9.59	9.59	10.27	8.22	0.34	10.62	8.90	1.71	4.79
SO_8	–	–	–	–	–	–	4.11	2.40	5.82	3.42	0	0.68	3.77	0	0
SO_{10}	–	–	–	–	–	–	–	2.40	3.42	2.74	0.68	0.68	1.71	0	0
$SU_{N>10}$	–	–	–	–	–	–	–	–	6.85	6.85	0.34	0	2.74	0	0
E_N	–	–	–	–	–	–	–	–	–	4.11	0.34	0.34	0.68	0	0
$E_6 \otimes E_6$	–	–	–	–	–	–	–	–	–	–	0	0	0	0	0
$\mathcal{F}SU_5$	–	–	–	–	–	–	–	–	–	–	–	6.51	1.71	2.05	2.74
G_{PS}	–	–	–	–	–	–	–	–	–	–	–	–	2.74	0.34	0.34
G_{LRS}	–	–	–	–	–	–	–	–	–	–	–	–	–	1.03	1.03
G_{RSM}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	2.74
G_{SM}	–	–	–	–	–	–	–	–	–	–	–	–	–	–	2.74
Total	87.67	54.79	15.07	40.75	15.75	65.07	20.55	17.12	26.71	18.15	1.71	15.75	16.78	4.45	7.53

BIBLIOGRAPHY

[1] M. Green, J.H. Schwarz, and E. Witten. *Superstring Theory. Vol. 1: Introduction.* Cambridge University Press, 1987.

[2] M. Green, J.H. Schwarz, and E. Witten. *Superstring Theory. Vol. 2: Loop Amplitudes, Anomalies And Phenomenology.* Cambridge University Press, 1987.

[3] J. Polchinski. *String theory. Vol. 1: An introduction to the bosonic string.* Cambridge University Press, 1998.

[4] J. Polchinski. *String theory. Vol. 2: Superstring theory and beyond.* Cambridge University Press, 1998.

[5] K. Becker, M. Becker, and J.H. Schwarz. *String theory and M-theory: A modern introduction.* Cambridge University Press, 2007.

[6] M. Robinson, G. Cleaver, and M. Hunziker. Free fermionic heterotic model building and root systems. *Mod. Phys. Lett.*, A24:2703–2715, 2009.

[7] F. Gliozzi, J. Scherk, and D. Olive. Supergravity and the spinor dual model. *Phys.Lett.*, B65:282, 1976.

[8] F. Gliozzi, J. Scherk, and D. Olive. Supersymmetry, supergravity theories and the dual spinormodel. *Nucl.Phys.*, B122:253–290, 1977.

[9] D. Gross, J. Harvey, E. Martinec, and R. Rohm. The heterotic string. *Phys.Rev.Lett.*, 54:502–505, 1985.

[10] D. Gross, J. Harvey, E. Martinec, and R. Rohm. Heterotic string theory. 1. the free heterotic string. *Nucl.Phys.*, B256:253, 1985.

[11] D. Gross, J. Harvey, E. Martinec, and R. Rohm. Heterotic string theory. 2. the interacting heterotic string. *Nucl.Phys.*, B267:75, 1986.

[12] G. Cleaver, A. Faraggi, D. Nanopoulos, and J. Walker. Phenomenological study of a minimal superstring standard model. *Nucl. Phys.*, B593:471–504, 2001.

[13] J. Lopez, D. Nanopoulos, and K. Yuan. The search for a realistic flipped su(5) string model. *Nucl. Phys.*, B399:654–690, 1993.

[14] A. Faraggi, D. Nanopoulos, and K. Yuan. A standard like model in the 4d free fermionic string formulation. *Nucl. Phys.*, B335:347, 1990.

[15] A. Faraggi. Construction of realistic standard - like models in the free fermionic superstring formulation. *Nucl. Phys.*, B387:239–262, 1992.

[16] I. Antoniadis, G. Leontaris, and J. Rizos. A three generation su(4) x o(4) string model. *Phys. Lett.*, B245:161–168, 1990.

[17] G. Leontaris and J. Rizos. N=1 supersymmetric su(4)xsu(2)lxsu(2)r effective theory from the weakly coupled heterotic superstring. *Nucl. Phys.*, B554:3–49, 1999.

[18] A. Faraggi. A new standard - like model in the four-dimensional free fermionic string formulation. *Phys. Lett.*, B278:131–139, 1992.

[19] A. Faraggi. Aspects of nonrenormalizable terms in a superstring derived standard - like model. *Nucl. Phys.*, B403:101–121, 1993.

[20] A. Faraggi. Generation mass hierarchy in superstring derived models. *Nucl. Phys.*, B407:57–72, 1993.

[21] A. Faraggi. Hierarchical top - bottom mass relation in a superstring derived standard - like model. *Phys. Lett.*, B274:47–52, 1992.

[22] A. Faraggi. Yukawa couplings in superstring derived standard like models. *Phys. Rev.*, D47:5021–5028, 1993.

[23] A. Faraggi. Top quark mass prediction in superstring derived standard - like model. *Phys. Lett.*, B377:43–47, 1996.

[24] A. Faraggi. Calculating fermion masses in superstring derived standard - like models. *Nucl. Phys.*, B487:55–92, 1997.

[25] G. Cleaver. Advances in old-fashioned heterotic string model building. *Nucl. Phys. Proc. Suppl.*, 62:161–170, 1998.

[26] G. Cleaver and A. Faraggi. On the anomalous u(1) in free fermionic superstring models. *Int. J. Mod. Phys.*, A14:2335–2356, 1999.

[27] G. Cleaver, M. Cvetic, J. Espinosa, L. Everett, and P. Langacker. Classification of flat directions in perturbative heterotic superstring vacua with anomalous u(1). *Nucl. Phys.*, B525:3–26, 1998.

[28] G. Cleaver, M. Cvetic, J. Espinosa, L. Everett, and P. Langacker. Flat directions in three-generation free-fermionic string models. *Nucl. Phys.*, B545:47–97, 1999.

[29] G. Cleaver et al. Physics implications of flat directions in free fermionic superstring models. i: Mass spectrum and couplings. *Phys. Rev.*, D59:055005, 1999.

[30] G. Cleaver et al. Physics implications of flat directions in free fermionic superstring models. ii: Renormalization group analysis. *Phys. Rev.*, D59:115003, 1999.

[31] G. Cleaver. Quark masses and flat directions in string models. pages 332–340, 1998.

[32] G. Cleaver, A. Faraggi, and D. Nanopoulos. String derived mssm and m-theory unification. *Phys. Lett.*, B455:135–146, 1999.

[33] G. Cleaver, A. Faraggi, and D. Nanopoulos. A minimal superstring standard model. i: Flat directions. *Int. J. Mod. Phys.*, A16:425–482, 2001.

[34] G. Cleaver, A. Faraggi, D. Nanopoulos, and J. Walker. Phenomenological study of a minimal superstring standard model. *Nucl. Phys.*, B593:471–504, 2001.

[35] G. Cleaver. M fluences on string model building. 1999.

[36] G. Cleaver, A. Faraggi, D. Nanopoulos, and J. Walker. Non-abelian flat directions in a minimal superstring standard model. *Mod. Phys. Lett.*, A15:1191–1202, 2000.

[37] G. Cleaver, A. Faraggi, and C. Savage. Left-right symmetric heterotic-string derived models. *Phys. Rev.*, D63:066001, 2001.

[38] G. Cleaver, A. Faraggi, D. Nanopoulos, and J. Walker. Phenomenology of non-abelian flat directions in a minimal superstring standard model. *Nucl. Phys.*, B620:259–289, 2002.

[39] G. Cleaver, D. Clements, and A. Faraggi. Flat directions in left-right symmetric string derived models. *Phys. Rev.*, D65:106003, 2002.

[40] G. Cleaver, A. Faraggi, and S. Nooij. Nahe-based string models with su(4) x su(2) x u(1) so(10) subgroup. *Nucl. Phys.*, B672:64–86, 2003.

[41] G. Cleaver. Parameter space investigations of free fermionic heterotic models. 2002.

[42] G. Cleaver et al. On the possibility of optical unification in heterotic strings. *Phys. Rev.*, D67:026009, 2003.

[43] J. Perkins, B. Dundee, R. Obousy, E. Kasper, M. Robinson, et al. Heterotic string optical unification. pages 86–93, 2003.

[44] J. Perkins et al. Stringent phenomenological investigation into heterotic string optical unification. *Phys. Rev.*, D75:026007, 2007.

[45] G. Cleaver, A. Faraggi, E. Manno, and C. Timirgaziu. Quasi-realistic heterotic-string models with vanishing one-loop cosmological constant and perturbatively broken supersymmetry? *Phys. Rev.*, D78:046009, 2008.

[46] J. Greenwald, K. Pechan, T. Renner, T. Ali, and G. Cleaver. Note on a nahe variation. *Nucl.Phys.*, B850:445–462, 2011.

[47] G. Cleaver, A. Faraggi, J. Greenwald, D. Moore, K. Pechan, et al. Investigation of quasi-realistic heterotic string models with reduced higgs spectrum. *Eur.Phys.J.*, C71:1842, 2011.

[48] S. Elitzur, E. Gross, E. Rabinovici, and N. Seiberg. Aspects of bosonization in string theory. *Nucl.Phys.*, B283:413, 1987.

[49] D. Bailin, D. Dunbar, and A. Love. Bosonization of four-dimensional real fermionic string models and asymmetric orbifolds. *Nucl.Phys.*, B330:124, 1990.

[50] I. Antoniadis, C. Bachas, and C. Kounnas. Four-dimensional superstrings. *Nucl. Phys.*, B289:87, 1987.

[51] I. Antoniadis and C. Bachas. 4-d fermionic superstrings with arbitrary twists. *Nucl. Phys.*, B298:586, 1988.

[52] H. Kawai, D. Lewellen, and S.H.H. Tye. Construction of fermionic string models in four- dimensions. *Nucl. Phys.*, B288:1, 1987.

[53] H. Dreiner, J. Lopez, D. Nanopoulos, and D. Reiss. String model building in the free fermionic formulation. *Nucl. Phys.*, B320:401, 1989.

[54] G. Cleaver. Supersymmetries in free fermionic strings. *Nucl. Phys.*, B456:219–256, 1995.

[55] R. Bousso and J. Polchinski. Quantization of four-form fluxes and dynamical neutralization of the cosmological constant. *JHEP*, 06:006, 2000.

[56] S. Ashok and M. Douglas. Counting flux vacua. *JHEP*, 01:060, 2004.

[57] K. Dienes. Statistics on the heterotic landscape: Gauge groups and cosmological constants of four-dimensional heterotic strings. *Phys. Rev.*, D73:106010, 2006.

[58] K. Dienes, M. Lennek, D. Senechal, and V. Wasnik. Supersymmetry versus gauge symmetry on the heterotic landscape. *Phys. Rev.*, D75:126005, 2007.

[59] K. Dienes and M. Lennek. Fighting the floating correlations: Expectations and complications in extracting statistical correlations from the string theory landscape. *Phys. Rev.*, D75:026008, 2007.

[60] D. Moore, J. Greenwald, T. Renner, M. Robinson, C. Buescher, et al. Systematic investigations of the free fermionic heterotic string gauge group statistics: Layer 1 results. *Mod.Phys.Lett.*, A26:2411–2426, 2011.

[61] D. Moore, J. Greenwald, and G. Cleaver. Gauge models in d dimensions. *Mod.Phys.Lett.*, A28:1350055, 2013.

[62] H. Kawai, D. Lewellen, J. Schwartz, and S.H.H. Tye. The spin structure construction of string models and multiloop modular invariance. *Nucl. Phys.*, B299:431, 1988.

[63] H. Kawai, D. Lewellen, and S.H.H. Tye. Classification of closed fermionic string models. *Phys. Rev.*, D34:3794, 1986.

[64] T. Renner, J. Greenwald, D. Moore, and G. Cleaver. Redundancies in explicitly constructed ten dimensional heterotic string models. *Int.J.Mod.Phys.*, A26:4451–4473, 2011.

[65] T. Renner. Initial systematic investigations of the weakly coupled free fermionic heterotic string landscape statistics. *http://hdl.handle.net/2104/8240*, 2011.

[66] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[67] V. Cerny. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51, 1985.