Lattice Monte Carlo simulations of the discretised BFSS model

Diarmuid Dignam

Supervised by Prof. Denjoe O'Connor

In collaboration with Edžus Nakums and Liam Featherly

School of Theoretical Physics

Dublin Institute for Advanced Studies

August 2023

Abstract

This project presents an investigation of the bosonic part of the BFSS matrix model by distretisation of the model onto the lattice. A programme is written to simulate the dynamics of this discretised model by use of the Hamiltonian Monte Carlo algorithm. The results of this algorithm are tested against analytically calculated expectation values for thermodynamic systems and Wigner's semicircle distribution and found to be in excellent agreement, however further tests are required to confirm the accuracy for larger systems and for comparison with literature results of the BFSS model itself. This indicates the simulation is a useful proof of concept of the simulation method for the BFSS model and possible improvements and extensions are proposed.

Contents

Acknowledgements

I would like to express my deep gratitude to the Dublin Institute for Advanced studies for this incredible opportunity and the experience it offered, and for the support all at the institute happily gave while I explored my passion more deeply than I had ever before been able.

My particular thanks go to my supervisor, Prof. Denjoe O'Connor for not only endless, but enthusiastic support, advice, and encouragement. The immense amount I have learned during this internship all came directly from his guidance.

Thank you as well to the coordinators of the internship, Dr. Atri Dey and Dr. Saki Koizumi for their hard work, and to my fellow interns, in particular my partners in this project Liam Featherly and Edžus Nakums for helpful and always fascinating conversations.

Finally, I would like to thank all who have supported me in reaching this point in my journey into theoretical physics - most importantly all at the UCD School of Physics, and my family for their endless encouragement and drive to explore.

Chapter 1

Introduction

M-theory is an ongoing and exciting area of study in modern physics. Proposed in 1995, the theory is a unification of all consistent versions of superstring theory into one model via symmetries between the five consistent theories known as S-duality and T-duality. The proposal models these five theories (and 11-dimensional supergravity) as separate limiting cases of one broader theory. M-theory currently presents a promising area for further research and new physics, and a possible framework for the unification of the fundamental forces into a theory of everything [1].

The theory, however, has several notable open questions and topics of ongoing research. These include the nature of the compactification of the extra dimensions in the theory to the constraints of the everyday universe (from the 11 dimensions required by uncompactified M-theory to the 4-dimensional spacetime experienced in everyday life) and generating testable predictions for experimental verification of the theory[1]. Notably, the theory also lacks a complete non-perturbative formulation[1][2]. A complete formulation, or solutions to any of these questions, would greatly advance our understanding of M-theory and its related theories, and would present a significant advance in modern physics and the search for a unification of the fundamental forces[1].

Modern attempts at a formulation of M-theory beyond the limits of perturbation theory typically depend on matrix models of some kind, and are widely expected to be the infinite-size limit of such a model[2]. One such is the BFSS model proposed in 1997. This model (discussed in detail in Sec. 2 below) consists of a matrix quantum mechanical system, and is conjectured to capture the entire dynamics of M-theory[3][4].

The BFSS model consists of nine bosonic matrices and sixteen fermionic matrices[2], and forms a low energy effective description the dynamics of interacting D0-branes in type IIA superstring theory[2]. As type IIA string theory gives rise to M-theory in the strong coupling limit[3], it is

thus conjectured that the infinite size limit of the matrices (i.e. the limit of a large number of D0-branes) of the BFSS model is equivalent to uncompactified eleven-dimensional M-theory[3][1].

Thus, the BFSS model presents an excellent candidate for further study into M-theory and non-perturbative formulations[4]. Despite several limitations now known to prevent the model alone forming a complete description of the dynamics M-theory (details discussed in [5]), the BFSS model is an exciting region of study to investigate the dynamics of M-theory in a simple setting[4].

This project thus aimed to recreate the bosonic portion of the studies presented in [2], beginning with a discretisation of the BFSS model onto the lattice. A Monte Carlo algorithm was then implemented to simulate the dynamics of the discretised model and its results verified by comparison to similar studies in the literature. The details and reasons for this approach are discussed in greater detail in Sec. 2, while other possible methods that have been used are available in [6].

Chapter 2

The BFSS Model

2.1 Overview

The BFSS model is a is a one-dimensional supersymmetric Yang-Mills theory, forming a quantum mechanical system of nine bosonic and sixteen fermionic matrices[3]. Developed by Banks, Fischler, Shenker, and Susskind, the model arises naturally from type IIA superstring theory and was shown by the authors to be described in the low-energy limit by eleven-dimensional supergravity[3]. It is conjectured that in the limit of large matrix size, N, the model becomes equivalent to eleven-dimensional M-theory[3]. Thus, the BFSS model is an excellent candidate for study to investigate or search for the complete formulation of M-theory in a simple setting[7].

This project investigated the dynamics of the BFSS model in the low-energy limit, where the model becomes an effective description of of a system of D0-branes in type IIA string theory[2]. In this context, the degrees of freedom of the BFSS matrices represent degrees of interaction between the branes, and thus the dimension of the matrices, N, encodes the number of branes in the system. In the limit as N increases, the model is thus conjectured to be equivalent to full uncompactified M-theory, while for finite N it is equivalent to M-theory compactified on a light-like circle[3].

The full matrix model can be obtained by dimensional reduction of supersymmetric Yang-Mills theory in ten dimensions down to one dimension[2], yielding a ten-dimensional action:

$$S_M = \frac{1}{g^2}\int dt \mathrm{Tr}\left\{\frac{1}{2}(\mathcal{D}_0 X^i) + \frac{1}{4}[X^i, X^j]^2 - \frac{i}{2}\Psi^T C_{10}\Gamma^0 D_0 \Psi + \frac{1}{2}\Psi^T C_{10}\Gamma^i [X^i, \Psi]\right\} \tag{2.1}$$

where the latter two terms are a description of the fermionic part of the system and their various interactions. A full simulation of these components can be found in [2], however due

to time constraints this project examined only the bosonic part of the action. Wick-rotated to euclidean time for computation later, this was given by:

$$S_b = \frac{1}{g^2}\int_0^\beta dt \mathrm{Tr}\left\{\frac{1}{2}(\mathcal{D}_t X^i)^2 - \frac{1}{4}[X^i, X^j]^2\right\} \tag{2.2}$$

In this equation, X^i are nine $N \times N$ hermitian matrices (indices i and j running from 1 to 9, representing the nine spatial dimensions of the model) representing the degrees of interaction of the D0-branes in each dimension. β acts as the thermodynamic inverse temperature $\beta = \frac{1}{k_B T}$, encoding the thermodynamic temperature, and thus total energy, of the system.

This action was introduced by Hoppe as a gauge-fixed and regulated description of a system of membranes and represents the bosonic part of the BFSS action, in the form to be studied in this project. This has been studied extensively by various methods in the literature (comprehensive reviews include [6]). The approach taken in this project follows that in [2]: to discretise the action in time and numerically simulate the dynamics of the resulting system. Following verification of this method, this would allow for the calculation investigation of many observables and features of the BFSS model.

This method was chosen due to the gauge nature of the BFSS model. The gauge properties of the model and, as with the wider class of Yang-Mills theories in general, make direct computation of results extremely difficult[7], while discretisation of the continuous system allows for the problem to be treated using methods from statistical mechanics, including simulation using Monte Carlo methods for numerical computation of the system[7], an extremely powerful tool if successful to investigate the dynamics of the BFSS model, and thus to investigate broader M-theory.

2.2 Discretisation of the BFSS action

The discretisation discussed above to allow for numerical simulation of the action, Eq. **??** was performed according to the procedure described in [2].

With lattice spacing $a = \frac{\beta}{\Lambda}$, time was thus discretised onto Λ sites $t_n = an$, $(n = 0, 1, ..., \Lambda - 1)$ and periodic boundary conditions imposed such that the final lattice point was identified with the initial point: $t_\Lambda = t_0$.

Several factors must be accounted for in the discretisation due to the gauge nature of the BFSS model. First, the derivative term in Eq. **??** must be covariant to account for the local gauge field differences between lattice sites. Unitary matrices, $UU^\dagger = \mathbb{I}$, were thus used as parallel transporter

terms and combined with the pure derivative on the lattice,

$$\partial_t X_n^i \to \frac{X_{n+1}^i - X_n^i}{a} \tag{2.3}$$

to transport back the field at $t_n + 1$ to t_n, forming the discrete version of the covariant derivative:

$$\mathcal{D}_t X_n^i \to \frac{U_{n,n+1} X_{n+1}^i U_{n+1,n} - X_n^i}{a} = \frac{U_{n,n+1} X_{n+1}^i U_{n,n+1}^\dagger - X_n^i}{a} \tag{2.4}$$

Applied to Eq. 2.2 and regrouping the resulting terms, this yields the following discrete bosonic action:

$$S_b = N \sum_{n=0}^{\Lambda-1} \mathrm{Tr}\left\{ -\frac{1}{a} X_n^i U_{n,n+1} X_{n+1}^i U_{n,n+1}^\dagger + \frac{1}{a}(X_n^i)^2 - \frac{a}{4}[X_n^i, X_n^j]^2 \right\} \tag{2.5}$$

This can be further simplified for computation by taking advantage of the local U(N) symmetry at each lattice site and defining the transformation: //

$$\begin{aligned}
&X_0^i = X_0'^i \\
&U_{0,1} X_1^i U_{0,1}^\dagger = X_1'^i \\
&(U_{0,1}U_{1,2}) X_0^i (U_{0,1}U_{1,2})^\dagger = X_2'^i \\
&\dots \\
&(U_{0,1}U_{1,2}\dots U_{\Lambda-2,\Lambda-1}) X_{\Lambda-1}^i (U_{0,1}U_{1,2}\dots U_{\Lambda-2,\Lambda-1})^\dagger = X_{\Lambda-1}'^i
\end{aligned}$$

$$\Rightarrow S_b = -\frac{1}{a} N \mathrm{Tr}\left\{ \sum_{n=0}^{\Lambda-1} X_n'^i X_{\Lambda-1}'^i \mathcal{W} X_0'^i \mathcal{W}^\dagger \right\} + N \sum_{n=0}^{\Lambda-1} \mathrm{Tr}\left\{ \frac{1}{a}(X_n'^i)^2 - \frac{a}{4}[X_n'^i, X_n'^j]^2 \right\} \tag{2.6}$$

Where unitary matrix $\mathcal{W} \coloneqq (U_{0,1}U_{1,2}\dots U_{\Lambda-2,\Lambda-1}U_{\Lambda-1,0})$. This matrix can then be decomposed as $\mathcal{W} = VDV^\dagger$, with $D = \mathrm{diag}[e^{i\theta_1}, \dots, e^{i\theta_N}]$.

It can be seen from the decomposition above that the values for θ_n define the eigenvalues of the unitary transporter $\mathcal{W}$, and thus the gauge field applied across the lattice sites. The interaction of the transporter fields, $U_{n,n+1}$ with the gauge field is calculated in Sec. 2.2.1.

This decomposition can be used with the symmetries of the action Eq. 2.6 $X_n'^i \to V X_n'^i V^\dagger$ and $X_n'^i \to h_n X_n'^i h_n^\dagger$, where h_n is a diagonal unitary matrix to further simplify the action. Defining $h_n = (D_\Lambda)^n$ and D_Λ such that $(D_\Lambda)^\Lambda = D$, these symmetries can be combined to show that

$$X_n'^i = (Vh_n)\tilde{X}_n^i (Vh_n) \tag{2.7}$$

,

yielding an action of

$$S_b[\tilde{X}, D_\Lambda] = N \sum_{n=0}^{\Lambda-1} \mathrm{Tr}\left\{\frac{1}{a}\tilde{X}_n^i D_\Lambda \tilde{X}_{n+1}^i D_\Lambda^\dagger + \frac{1}{a}(\tilde{X}_n^i)^2 - \frac{a}{4}[\tilde{X}_n^i, \tilde{X}_n^j]^2\right\} \tag{2.8}$$

Finally, we choose for convenience now to work in a gauge in which the holonomy is non-trivial between only one pair of lattice sites (chosen to be between the sites t_0 and $t_{\Lambda-1}$, or equivalently by the boundary conditions $t_{\Lambda-1}$ and t_Λ), allowing the diagonal matrices h_n to be omitted at the other links. And from the definition of D_Λ and the boundary conditions at $t_0 = t_\Lambda$, this yields the final equation for the discrete bosonic action as it was simulated in this project:

$$S_b[X, D] = N\mathrm{Tr}\left\{-\frac{1}{a}\sum_{n=0}^{\Lambda-2} X_n^i X_{n+1}^i - \frac{1}{a}X_{\Lambda-1}^i D X_0^i D^\dagger + \sum_{n=0}^{\Lambda-1}\left[\frac{1}{a}(X_n^i)^2 - \frac{a}{4}[X_n^i, X_n^j]^2\right]\right\} \tag{2.9}$$

This equation is the bosonic part of the BFSS model in the low-energy limit discretised to Λ sites in time, describing the dynamics of a system of D0-branes in the low-energy limit of type IIA string theory in nine spatial dimensions, $i, j = 1, ..., 9$. The first two terms describe the dynamics in each dimension and capture the action of the portion of the system in this dimension as it progresses over time, the first term capturing the action between successive sites discretised in time and the second applying the periodic boundary conditions. The second two terms encodes the action from interactions between the dimensions at each site in time, the first from the interaction of each dimension with itself and the second accounting for the interaction of each dimension with the others.

This thus describes the total action of any sequence of Λ states the system may pass through over time. Using the method described below, this allows the dynamics of the BFSS model to be simulated by numerical methods and observables of its behaviour calculated (with extensions from this point discussed in Sec. 7).

2.2.1 Transporter Field Interactions

Another aspect of the gauge field that must be accounted for in the discretisation process is the interactions of the unitary transporters with the gauge field (details of discretisation procedure again with reference to [2]).

The transporter fields introduced by the unitary matrices in Eq. 2.6, $U_{n,n+1}$, interact with the the gauge field and its associated U(N) symmetry, and, as the transformations are local, with the transporter fields themselves. This requires the addition of a Faddeev-Popov term ([8] determined by the gauge (information about which is carried by θ_n) to the total action of the system:

$$S_{FP}[\theta] = -\sum_{l \neq m} \ln \left| \sin \frac{\theta_l - \theta_m}{2} \right| \tag{2.10}$$

This term is derived from the measure of the transporter fields $U_{n,n+1}$ and the covariant derivative of $U_{n,n+1}$, described in detail in [2].

Chapter 3

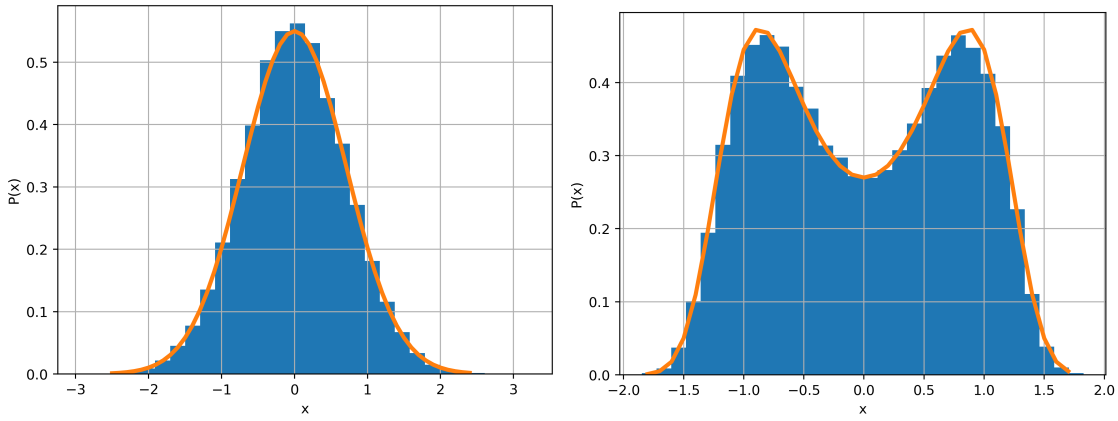
Hamiltonian Monte Carlo

3.1 Overview

The primary method used in the study of the above dynamics was the Hybrid Monte Carlo algorithm (HMC, also known as Hamiltonian Monte Carlo). Proposed in 1987 for study of lattice quantum chromodynamics, this is a Markov chain Monte Carlo method designed to generate a random sequence which converges to a given target distribution. This allows the algorithm to be used as a form of numerical integrator, allowing for the calculation of expectation values of observables of that target distribution. [9][10]

The algorithm is a form of the Metropolis-Hastings system, with iteratively proposed points accepted or rejected into the Markov chain based on relative likelihood of the new point compared to the initial point according to the target distribution. HMC, however uses an evolution step between successive proposed points derived from Hamiltonian dynamics rather than the random walk used in the base version of the Metropolis-Hastings algorithm.

To propose the next point in the Markov chain, HMC uses a symplectic integrator to solve Hamilton's equations from the current location in the phase space of the distribution. This allows for the algorithm to make significantly larger steps through the phase space than a random walk while still maintaining a high probability of acceptance due to the energy conservation of the Hamiltonian dynamics between successive states. These distant steps significantly reduce the correlation between successive states compared to the high constraint of a random walk, and allow the algorithm to explore the full target distribution with fewer Markov chain samples, thus allowing for integration with fewer samples than would be necessary using a random walk for the same degree of accuracy[9][10].



(a) $f_1(x) = A_1 e^{-x^2}$ (b) $f_2(x) = A_2 e^{-(x^4 - \frac{3}{2}x^2)}$

Figure 3.1: Two one-dimensional target distribution functions, $f_1(x)$ and $f_2(x)$, (orange) and histograms of the Markov chain generated by HMC to converge to these distributions

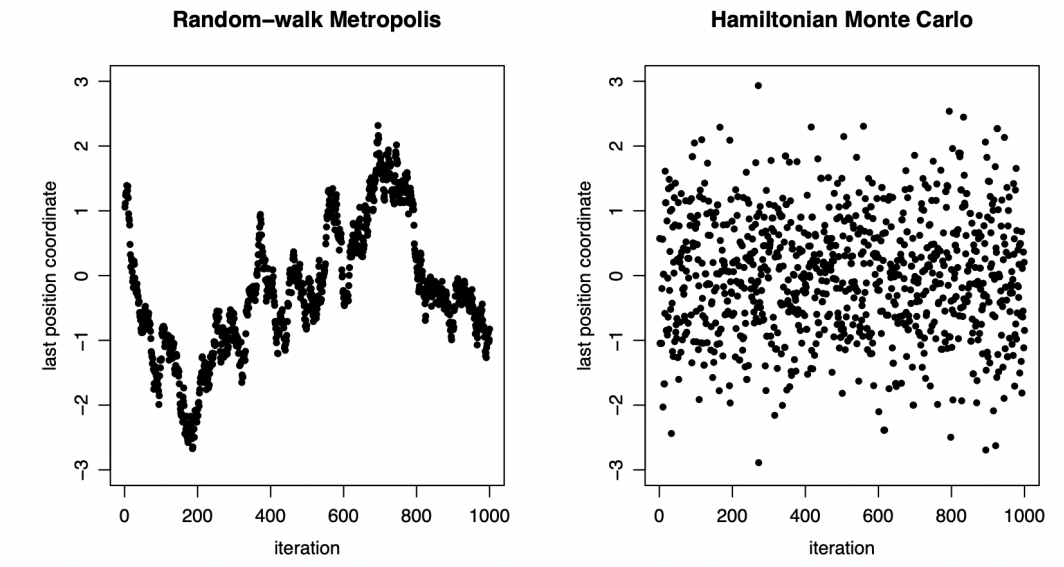


Figure 3.2: Comparison of correlation between successive points proposed by random walk and HMC illustrating the time taken to sample the full position space (y-axis). From [9]

3.2 HMC Algorithm

3.2.1 Main Iteration Procedure

The HMC algorithm generates its Markov chain by iteration from a random initial point (or from the origin, known as hot or cold starts respectively). This point is then progressed through the phase space of point position vs. momentum according to Hamilton's equations and the resulting new state is accepted or rejected by a Metropolis check, from which point the process is repeated until the resulting Markov chain converges sufficiently to the target distribution (detailed description below drawn from comprehensive reviews of the topic, [9] and [10].

Hamiltonian Flow

During each iteration in the process, a new point (new position and new momentum) is proposed by solving Hamilton's equations for a later time:

$$\frac{d\mathbf{x}}{dt} = \frac{\partial H}{\partial \mathbf{p}}, \qquad \frac{d\mathbf{p}}{dt} = -\frac{\partial H}{\partial \mathbf{x}} \tag{3.1}$$

where H is the Hamiltonian familiar from classical mechanics of the "particle" representing the point in phase space:

$$H(\mathbf{x}, \mathbf{p}) = K(\mathbf{p}) + S(\mathbf{x}) = \frac{|\mathbf{p}|^2}{2} + S(\mathbf{x}) \tag{3.2}$$

Here, momentum and the total Hamiltonian have no physical significance or effect on the simulation. They are defined simply as a computational tool to facilitate the Hamiltonian dynamics used to propose the next point in the sequence. More detail on their choice and effect on the result is given below.

The action of a given position, $S(\mathbf{x})$, has a significance beyond the Hamiltonian flow between points: it encodes the distribution to which the Markov chain should converge. For a target distribution $f(\mathbf{x})$ the action is defined in relation to $f(x)$ as:

$$S(\mathbf{x}) = -\ln\left(f(\mathbf{x})\right) \rightarrow f(\mathbf{x}) = e^{-S(\mathbf{x})} \tag{3.3}$$

The choice of this definition for the potential energy and the relations between it, the target distribution, and the Hamiltonian are motivated by studies of canonical distributions from statistical mechanics. How the target distribution arises from the Hamiltonian is discussed in Sec. 3.2.2.

At the beginning of each iteration, a random Gaussian-distributed momentum is chosen for

the particle. The particle is then evolved from the current point in the phase space, given by the random momentum and the latest position entry in the Markov chain, by solving Hamilton's equations for some later time, after interval ϵ (alternatively interpreted as a step size determining the distance traversed through the phase space in each iteration), using a symplectic integrator. Here the leapfrog integrator was used, in one dimension given by:

$$\begin{aligned} p &= p_i - \tfrac{\epsilon}{2} F(x_i) \\ x &= x_i + \epsilon p \\ p &= p - \epsilon F(x) \\ x_{i+1} &= x + \epsilon p \\ p_{i+1} &= p - \tfrac{\epsilon}{2} F(x) \end{aligned}$$

where x_i, p_i and x_{i+1}, p_{i+1} denote the initial point in phase space for the iteration and the time-evolved point, respectively, and $F(x)$ denotes the force on the particle due to the action at a given point. This was calculated from the potential as in classical mechanics,

$$F(x) = -\frac{\partial S(x)}{\partial x} \tag{3.4}$$

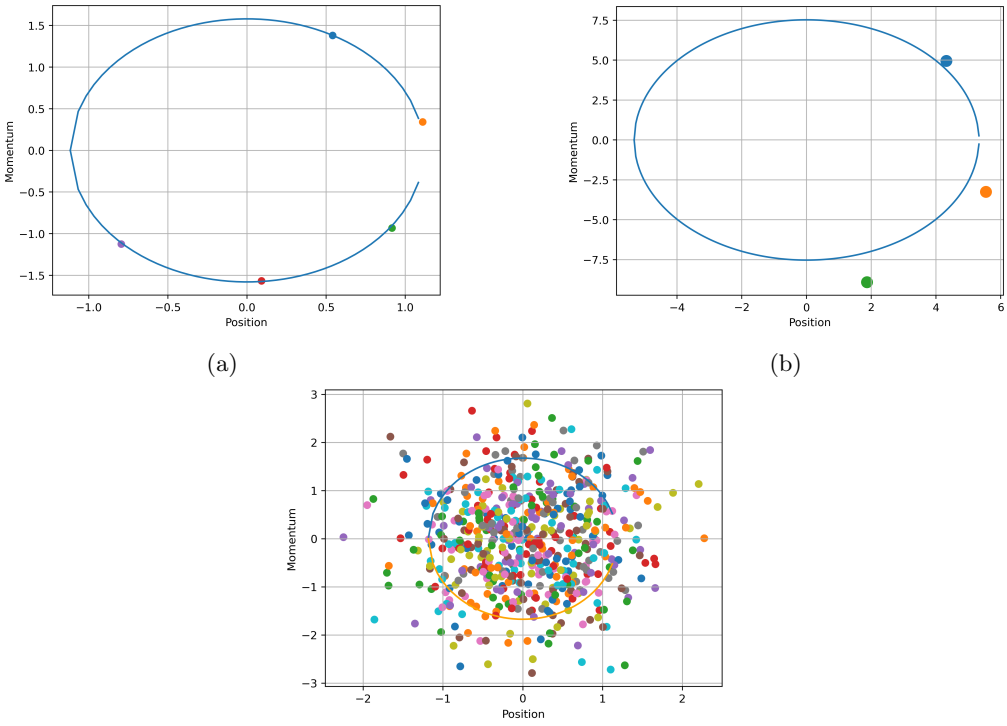
For a deterministic system such as the Hamiltonian of a real particle, the integrator would exactly solve for the path of the particle and its Hamiltonian through phase space (Fig. 3.3A). However, due to the randomness introduced in the choice of momentum for the particle at the beginning of each iteration, the algorithm will provide a proposal point slightly removed from this exact solution (Fig. 3.3B).

Metropolis Check

The Hamiltonian of the proposed point ($\mathbf{x}$ and $\mathbf{p}$ as calculated by the leapfrog integrator) is then compared to the Hamiltonian of the previous point (the previous $\mathbf{x}$ in the chain and the random momentum generated for this iteration) under a Metropolis check.

$$\begin{aligned} \mathrm{dH} &= \mathrm{H}(\mathrm{x}_{i+1}, p_{i+1}) - H(x_i, p_i) \\ \text{Accept if: } & \mathrm{e}^{-dH} > \mathrm{Random}(0,1) \end{aligned} \tag{3.5}$$

From the definition of the Hamiltonian (Eq. 3.2) and the potential energy (Eq. 3.3), this is equivalent to comparing the probability of the new point according to the target distribution to the probability of the old point. A proposed point that passes this check is accepted into the Markov chain and used as the beginning point for the next iteration. A proposed point that



(a) (b)

(c)

Figure 3.3: Comparison of standard leapfrog integration showing deterministic steps on a curve of constant Hamiltonian ($H = \frac{p^2}{2} + x^2$) (a) and deviation of the sample points from this curve once random momenta are sampled for each iteration (b). Exploration of the phase space after 200 iterations shown in (c).

is rejected is discarded and the previous point (the point used as the start for this iteration) is added to the chain again. The next iteration uses this previous point as the initial point, using a different random momentum and thus a new proposed point will be generated. In no case is the momentum stored, a new momentum is proposed in every iteration.

This check has the effect of increasing the likelihood of accepting a proposed point with a higher probability in the target distribution, increasing the overall acceptance rate of points in regions of high probability under the target distribution, while the random threshold ensures that some points of relatively low probability will still be accepted. This allows the full phase space to be explored while a greater sampling rate will be found in regions of greater target probability density, generating a Markov chain distributed according to the target distribution (in the convergence limit as the sample size increases, seen in Fig. 3.3(c) for the exploration of the phase space with sampling density converging to a normal distribution).

3.2.2 Probability Distribution from the Hamiltonian

The definition of the potential energy function used above, Eq. 3.3, arises from the canonical distribution in statistical mechanics, where for a given energy function, $E(x)$ of system state x, the probability density function over states of the system is given by

$$P(x) = \frac{1}{Z} e^{-\beta E(x)} \tag{3.6}$$

With partition function Z acting as the normalisation constant and inverse temperature, $\beta = \frac{1}{k_B T}$, acting as the temperature of the system, often conceptualised as the fixed temperature of a heat bath with which the system is in thermal equilibrium, and defining the spread of the resulting distribution.

Taking the energy function above to be the Hamiltonian, a function of the system state (position and momentum) for the total energy, the probability density function becomes

$$P(x) = \frac{1}{Z} e^{-\beta H(x,p)} = \frac{1}{Z} e^{-\beta (K(p)+S(x))} \tag{3.7}$$

As p and x are independent, each has their own canonical distributions with energy functions S and K:

$$P(x) = \frac{1}{Z} (e^{-\beta K(p)})(e^{-\beta S(x)}) \tag{3.8}$$

This allows us to choose x as the variable of interest to be studied, and to define $S(x)$ as in

Eq. 3.3 to give rise to the target distribution.

p is conversely introduced simply as a computational variable to facilitate the Hamiltonian evolution. Further, by appropriate choice of the random sampling for momentum used (the momentum used in each iteration was sampled from a normal distribution), the exact behaviour and effect of its canonical distribution can be known and discarded from the final result.

Discussion in greater detail can be found in [9], Sec. 3.1.

3.2.3 Thermalisation and Error Analysis

The choice of the initial point from which to begin the iteration process has a significant effect on the algorithm. If the starting point occurred in a region of extremely low probability under the target distribution, it may take several iterations to reach a region of high probability where sampling is more likely.

These samples taken before reaching the region to be sampled are referred to as the thermalisation period and are not distributed according to the target distribution, delaying the convergence of the chain to the target.

Several techniques are available to mitigate the affect of the thermalisation period on the final sample distribution. The most sophisticated methods rely on finding when the system is thermalised (when it reaches the region to be sampled) and either discarding samples before that point or beginning subsequent simulations from this point.

Due to time constraints, however, these methods were not implemented. Instead, the error contribution of the non-thermalised samples was minimised by increasing the total number of samples, thus increasing the portion of samples converging accurately to the target once the sampling region has been reached. This brute-force method is less effective than removing the error completely but ensured that the overall Markov chain converged well to the target distribution.

This factor falls under the larger topic of error and uncertainty in the simulation, of particular importance due to the Monte Carlo nature of the algorithm. For detailed results and conclusions to be drawn from this simulation, a detailed analysis of the statistical error needs to be performed of the algorithm and methods used. Due to time constraints this could not be performed satisfactorily and only qualitative study of the simulation's convergence to analytically expected results could be used to determine the effectiveness and accuracy. As the overall conclusions of the project drawn from the results are broadly qualitative and concern the general operation of the project methods (see Sec. 6), this does not significantly hinder the conclusions, however a detailed error analysis should be performed to confirm the conclusions drawn and to facilitate the precise measurement

of simulated results. Descriptions of Monte Carlo error and analysis of HMC can be found in [10] and [9].

3.3 Expectation Values and Numerical Integration

The Markov chain generated by the HMC algorithm then allows for the measurement of expectation values of the target distribution without the need for complex integration. As the list of samples produced by this algorithm already encode the distribution, the integral formula for expectation values:

$$\langle \Phi(\mathbf{x}) \rangle = \int_{-\infty}^{\infty} \Phi(\mathbf{x}) f(\mathbf{x}) d\mathbf{x} = \int_{-\infty}^{\infty} \Phi(\mathbf{x}) e^{-S(\mathbf{x})} d\mathbf{x} \tag{3.9}$$

can be reduced to a sum over the list of N samples:

$$\langle \Phi(\mathbf{x}) \rangle = \frac{1}{N} \sum_{i}^{N} \Phi(\mathbf{x}_i) \tag{3.10}$$

This allows observables of the action function used to define the distribution to be easily calculated. By choice of an appropriate action rather than choosing a specific target distribution, further, this algorithm may be used to simulate real systems and their observables.

For example, a simple harmonic oscillator may be simulated by defining $S(x)$ as the action of a harmonic oscillator, $S_{HO}(x) = \frac{k}{2}x^2$. With the force used in the Hamiltonian dynamics calculated as:

$$F(x) = -\frac{\partial S}{\partial x} = -kx \tag{3.11}$$

HMC then generates a distribution of the harmonic oscillator's position, a normal distribution with standard deviation determined by the parameters of the system (spring constant, total energy/temperature of the system) of the system (seen from the relation between the harmonic oscillator action and target distribution 3.3):

$$S_{HO}(x) = \frac{k}{2}x^2 \Rightarrow f(x) = e^{kx^2} \tag{3.12}$$

By this method, many real systems may be simulated and their characteristics calculated, including the action derived above for the discrete BFSS model.

3.3.1 Additional Degrees of Freedom

The HMC algorithm is also well suited to this project as systems with different degrees of freedom may be readily accommodated. Instead of a scalar x being sampled, a matrix may be used instead, with an action taking that matrix as an argument (i.e. $S(X) = e^{-\mathrm{Tr}(X^2)}$). This may be further extended to each sample consisting of multiple matrices or multiple dimensions.

Expectation values of distributions in these complicated structures would be extremely difficult to calculate analytically, possibly requiring element-wise integration in several dimensions, where this algorithm allows for them to be simulated to find distributions of configurations for the system as a whole under any appropriate action and Hamiltonian. The structure used in this project is described in Sec. 4.1.

Chapter 4

Implementing BFSS in HMC

The discretised BFSS model 2.9 was implemented in the HMC algorithm and various results of the simulation were compared to the expected values to investigate the function of the methodology and implementation.

For this project, the HMC algorithm was implemented in Python (first developed in one dimension, then expanded to the structure of the BFSS action described below) using the Jupyter development environment. The full simulation code is available at https://github.com/DiarmuidDig/DIAS_Internship.git (including previous versions for systems with various degrees of freedom implemented as progression towards the BFSS structure, and future development). The advantages and disadvantages of Python for this project and possible alternatives are discussed below (Sec. 6).

4.1 Structure of the Markov chain for BFSS simulations

The structure of the Markov chain generated by the algorithm was adapted from the basic one-dimensional case described in Sec. 3.2 to suit the BFSS model.

As described above, the BFSS action requires nine dimensions, each dimension represented by a matrix of size N. As this simulation studied the dynamics of the model, however, each dimension was represented by a sequence of Λ matrices, representing the discretised progression of that portion of the model over time as a sequence of Λ sites. Thus, the algorithm was expanded to generate this structure (nine spatial dimensions, each consisting of Λ matrices) as each entry in the Markov chain, with a distribution of entries given by the BFSS action. This represents the space of possible dynamics a nine-dimensional system of interacting D0-branes can take under the BFSS model.

$$\begin{bmatrix}\begin{bmatrix} X_0^1 \\ X_1^1 \\ \vdots \\ X_{\Lambda-1}^1 \end{bmatrix}\begin{bmatrix} X_0^2 \\ X_1^2 \\ \vdots \\ X_{\Lambda-1}^2 \end{bmatrix}\begin{bmatrix} \dots \\ \dots \\ \ddots \\ \dots \end{bmatrix}\begin{bmatrix} X_0^i \\ X_1^i \\ \vdots \\ X_{\Lambda-1}^i \end{bmatrix}\end{bmatrix} \tag{4.1}$$

Figure 4.1: Schematic view of one entry in the Markov chain generated according to the BFSS action. Each X is a matrix of size N, and each column represents one dimension and the change of its representative matrix over Λ time increments. In the BFSS model nine dimensions are required ($i = 9$). A row of matrices represents the state of the whole system at a given time site.

4.2 Hamiltonian for simulation of BFSS

The expanded HMC algorithm was then implemented to simulate the model as follows (as described in [2]). The Hamiltonian used for computation of the Hamiltonian flow and the generation of the Markov chain was defined:

$$H_b = \frac{1}{2}\sum_{n=0}^{\Lambda-1} \mathrm{Tr}\left(P_n^i . P_n^i\right) + \frac{1}{2}\sum_{l=0}^{N-1} {P_d^l}^2 + S_b[X, D(\theta)] + S_{FP}[\theta] \tag{4.2}$$

Where the first two terms are the kinetic energy used and the latter two are the action. The total action part is given by the discretised bosonic action derived above and the action associated with the gauge transformations, Sec. 2.2.1. The kinetic part is thus given by the canonical momenta P_n^i and P_d^l, corresponding to the matrices X_n^i and the angles θ_l, respectively. The momenta are sampled randomly from a normal distribution at the beginning of each iteration as described above, then subjected to Hamiltonian flow through the phase space just as the "position" values are.

As the gauge field of each proposed configuration of the system effects the probability of the system via the S_{FP} term, it must be included in the Hamiltonian and Hamiltonian flow stage and thus has its own momenta introduced for computation. These are sampled.

The forces on X and θ to perform the Hamiltonian evolution were calculated from Hamilton's equations:

$$-\frac{\partial S_b}{\partial X_{n,ml}^i} = \frac{N}{a}\left(X_{n+1}^i - 2X_n^i + X_{n-1}^i\right)_{lm} + Na\left[X_n^j, \left[X_n^i, X_n^j\right]\right]_{lm}$$

$$-\frac{\partial S_b}{\partial X_{0,ml}^i} = \frac{N}{a}\left(X_1^i - 2X_0^i + D^\dagger X_{\Lambda-1}^i D\right)_{lm} + Na\left[X_0^j, \left[X_0^i, X_0^j\right]\right]_{lm}$$

$$-\frac{\partial S_b}{\partial X_{\Lambda-1,ml}^i} = \frac{N}{a}\left(DX_0^i D^\dagger - 2X_{\Lambda-1}^i + X_{\Lambda-2}^i\right)_{lm} + Na\left[X_{\Lambda-1}^j, \left[X_{\Lambda-1}^i, X_{\Lambda-1}^j\right]\right]_{lm}$$

$$-\frac{\partial S_b}{\partial \theta_l} = \frac{2N}{a}\sum_{m=0}^{N-1}\Re\Big(iX^i_{\Lambda-1,ml}X^i_{0,lm}e^{i(\theta_l-\theta_m)}\Big) + \sum_{m,m\neq l}\cot\left(\frac{\theta_l-\theta_m}{2}\right)$$

(4.3)

These equations allowed for the dynamics of the discretised BFSS model to be numerically simulated using the HMC algorithm.

Chapter 5

Results

The simulation described above was thus implemented in Python and several checks were used to confirm the function of the algorithm and discretisation to accurately simulate the BFSS dynamics.

5.1 Thermodynamic Expectation Value Test

The first check used to investigate the accuracy of the HMC simulation was comparison to the analytically derived expectation value of the total action of each entry in the Markov chain for one dimension, $\langle S \rangle$.

The total action expectation value was calculated from the statistical mechanics of the general lattice system in one dimension, where the partition function $Z = \int [dX] = e^{-S[X]}$ encodes statistical properties of a system in thermodynamic equilibrium. From this function, various properties of a thermodynamic system can be calculated ([11]). Introducing dummy variable B (to be set = 1 during the derivation) and defining

$$Z_B = \int d^{N^2\Lambda} X e^{-S[X]B} \tag{5.1}$$

The expectation value of the action may be derived from $\frac{\partial Z_B}{\partial B}$ and found to be

$$\langle S \rangle = \frac{N^2\Lambda}{2} \tag{5.2}$$

This was checked for a range of simulation parameters, including various combinations of different parameters. Across all parameters, including matrix size N, number of lattice sites Λ, lattice spacing, and system energy, the simulation result was found to converge to the expected value as the number of Monte Carlo iterations per simulation increased:

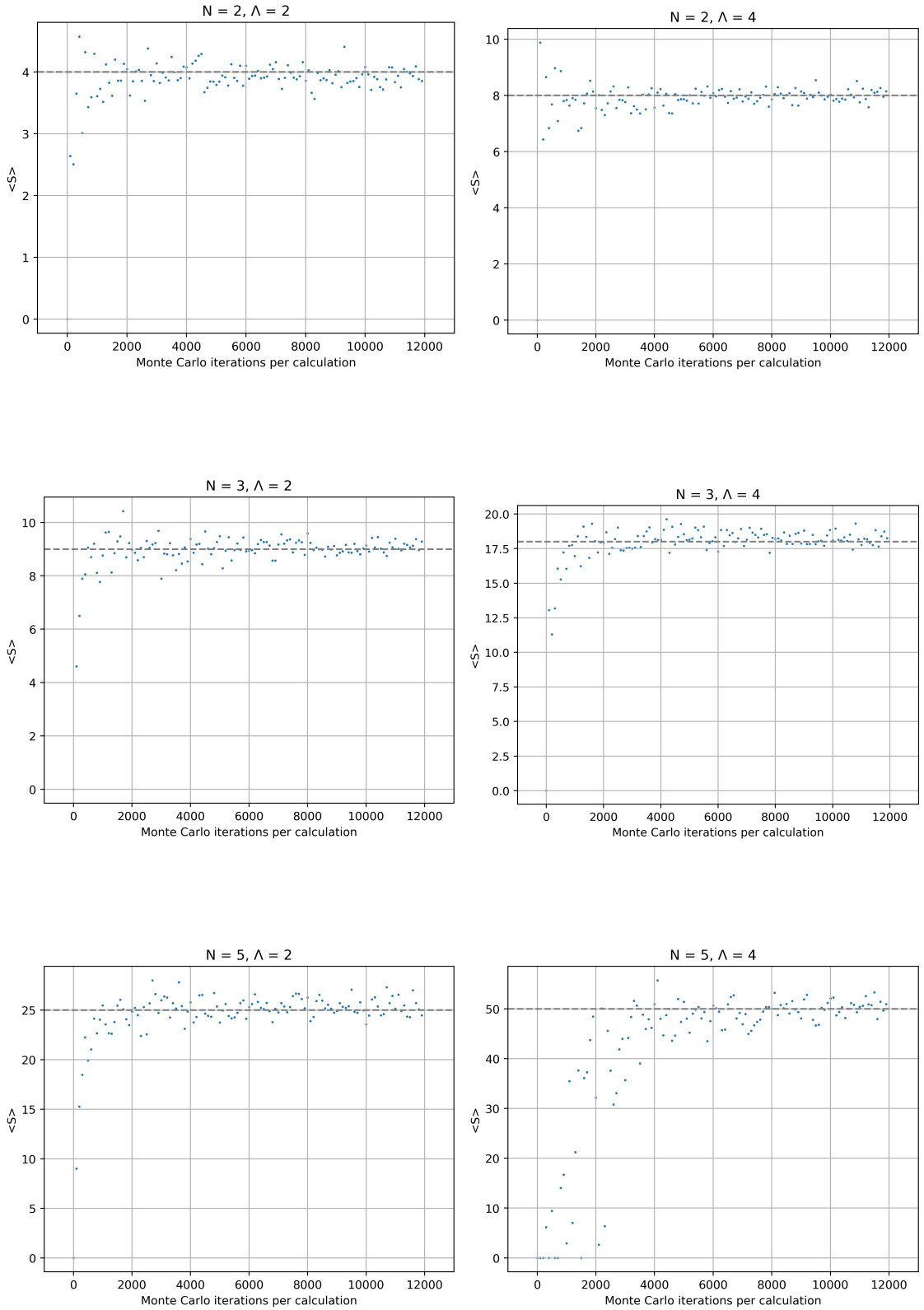


Figure 5.3: Sample plots to show convergence of simulation results to analytically predicted values as MC iterations increase across range of matrix size and number of time site. Predicted value Eq. 5.2 indicated by dashed line.

In all plots above, the simulation clearly converges to the analytically predicted value. This confirms the accurate function of the HMC algorithm for a general thermodynamic system of this structure and this implementation of it, indicating that it may be accurate for the BFSS model.

Note that due to limits in computational power and time, the results of simulations N or Λ could not be checked. The simulation could only be checked up to $N, \Lambda = 6$ and the various combinations of these values. This presents a significant limitation of is discussed in Sec. 6, however the convergence in all results up to these values is extremely clear in its agreement with the predicted values and does not indicate any deviation from this agreement as the parameters grow. Thus, while higher values should be checked as described below to confirm this conclusion, this check as it stands is considered sufficient to indicate the accurate function of the simulation for this thermodynamic system.

5.2 Wigner Distribution Test

The algorithm was also checked using Wigner's semicircle distribution. This is a result from free probability theory (the study of random non-commuting variables) that is broadly analogous to the role of the normal distribution in classical probability theory. Wigner's law states that for a collection of $N \times N$ symmetric matrices with variables randomly generated from the same distribution with bounded moments, the distribution of eigenvalues of the matrices approaches a semi-ellipse (which may be scaled to a semicircle) as N approaches infinity. This result of symmetric matrices is converted to apply to hermitian matrices by the behaviour of the Gaussian unitary ensemble[12].

The spectral distribution of the BFSS matrices in a given Markov chain (one run of the algorithm, yielding a collection of $N'timesN$ matrices) was found for a range of N values:

These distributions can be seen to converge extremely well to the Wigner semi-elliptical distribution, including in the behaviours seen during convergence as N increases such as the oscillation patterns about the semi-elliptical curve being approached. Again, due to computational limitations, the behaviour of larger matrices could not be measured. Only up to $N = 30$ could be investigated. However, from the extremely clear convergence visible in the range of N that was investigated it is clear that the simulation does generate matrices in accordance with Wigner's law.

This shows that the algorithm may be used to generate Markov chains of matrices (where the randomness of the chain would lead to the distribution). It has also been shown, however, that the matrices of simulated BFSS dynamics also converge to the Wigner distribution [2]. Thus, the

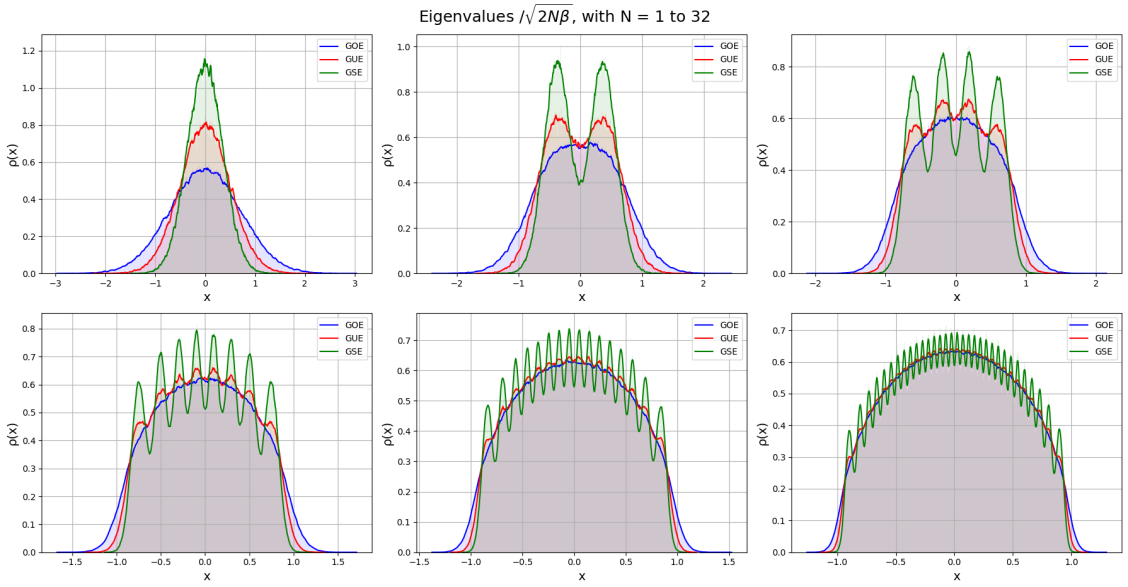
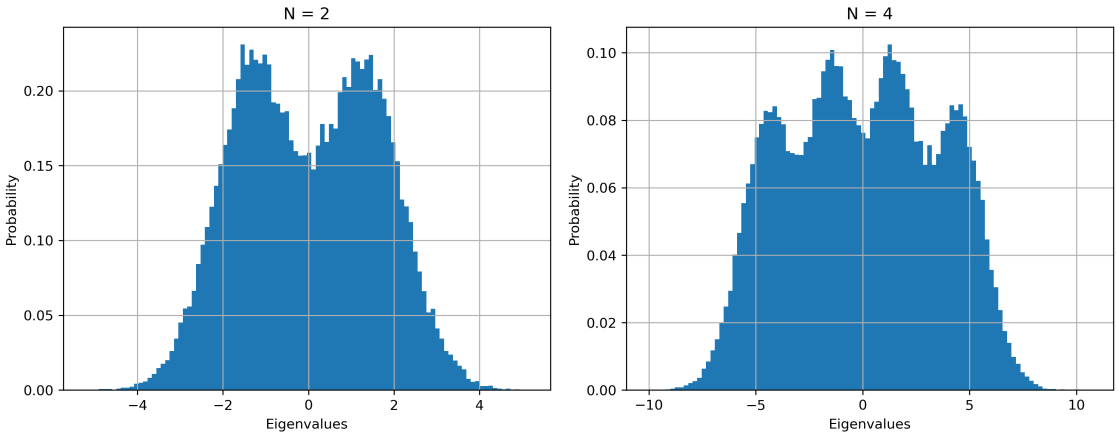


Figure 5.4: Convergence of spectra of various matrix types to the Wigner distribution with increasing N. For hermitian matrices, as were used here, the Gaussian unitary ensemble curve (red) is of interest. Image attributed to Cosmia Nebula - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=132035912, generating code accessible at this address and verified



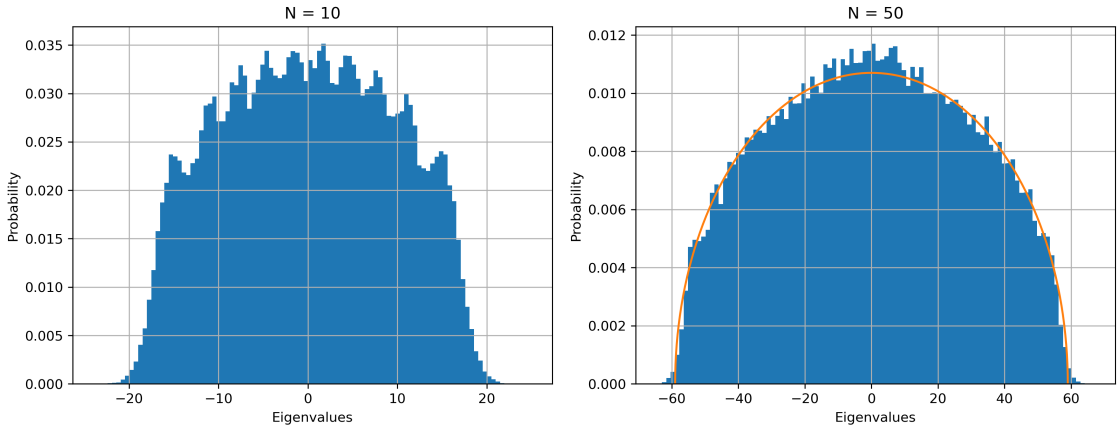


Figure 5.6: Normalised histograms of eigenvalue distributions for Markov chains of $N \times N$ matrices. Note that to account for increased computation time required at higher N, the simulation for $N = 30$ used significantly fewer iterations (3000 vs. 20000 for the other plots) and thus details of the distribution may be lost. Ellipse shown in $N = 30$ plot has $a = 0.0107, b = 59$

Wigner behaviour of this simulation of the BFSS action indicates that the simulation is accurate to the BFSS model and may be used to generate accurate measurements of the model's behaviour.

Chapter 6

Discussion

From these results it is believed that the methods used in this project present a valid technique for simulation of the BFSS model. The agreement of the algorithm's results for the expectation value of a thermodynamic system and the strong agreement of the simulation with behaviours of the BFSS model reported in the literature indicate that the system created during this project may be used to accurately simulate the bosonic BFSS model.

6.1 Check Limitations

However the limitations of the checks above should again be noted. The computational power used in this project was extremely low and due to the exponential increase in computational complexity with all of matrix size, discretisation site number, and dimension, the time required to simulate systems approaching realistic parameters was extremely high. This limited both the range of parameters that could be run during the checks and the number of checks that could be run.

The steep increase in complexity, and thus time requirements, to check higher values of any given simulation parameter restricted the checks that could be run to low values of N, Λ and dimension. These values do not approach the scale of a realistic simulation (requiring large N for the conjecture of BFSS modelling M-theory in the limit of a large number of branes, large Λ for the continuum limit of the discretisation, and nine spatial dimensions). This is not believed to present a large problem as the agreement of the smaller-scale simulations is extremely strong across a range of values for each parameter and across various combinations of the parameters. Although the maximum values checked are relatively small, there is no reason across the investigated range to suggest that the pattern of agreement does not hold as the parameters increase further.

While current equipment available prevents us from extending the simulation to systems closer to realistic parameters, it may be possible to run the checks described above at high values for one of the computationally-expensive parameters (N, Λ, and dimension) at a time (i.e. high lattice site count but maintain low dimensions and matrix size). This was necessary for the simulation of $N = 30$ for the Wigner distribution check above. These computations would take an extremely long time without improvements to the algorithm (discussed in Sec. 6.2 below) or to the hardware used, and would not fully confirm that the algorithm passes the checks for realistic parameters, but would provide further evidence that the algorithm is functioning as expected.

A greater issue that should be noted is the limitation in the number of different checks that could be run. Due to time constraints in the project overall, only the checks above could be run to investigate the function of the simulation. These were run first as they were applicable to all versions of the HMC algorithm that were implemented during development, before the inclusion of the full bosonic BFSS action while the checks below are applicable only to the final version.

The checks above, while valuable and necessary for preliminary debugging, do not confirm the results to be accurate to the BFSS model itself, only that the HMC algorithm is functioning and that the BFSS model as discretised here may be simulated with it. To confirm that the simulation provides results accurate to the model, other checks are necessary to compare the calculated results to derived predicted results or those found in the literature for the BFSS model. Possible checks of this nature are found in [2] and include the internal energy of the system and the extent of space:

$$E = N^2 \left\langle -\frac{3}{4N\beta} \int_0^\beta dt \mathrm{Tr}([X^i, X^j]^2) \right\rangle \tag{6.1}$$

$$\langle R^2 \rangle = \left\langle \frac{1}{N\beta} \int_0^\beta dt \mathrm{Tr}(X^i)^2 \right\rangle \tag{6.2}$$

(Detailed discussion and results of these simulations in [2], with further observables such as the Polyakov loop).

These calculations would require high values of the simulation parameters (as they are properties of the BFSS model and thus the simulation must approach the parameters of the real model) and would be extremely slow without the improvements discussed below or sufficient time to allow the current algorithm to run for a long period. However, these checks would directly confirm the usefulness and accuracy of this method for simulation of the BFSS model.

6.2 Possible Improvements

The above discussion illustrates the need for improvement to the system as it currently stands to be useful in the study of the BFSS model. Due to extremely low efficiency the programme cannot be tested or used for practical study at realistic scales, despite indications that it may ultimately generate accurate results. While this may be tackled by brute force methods of allowing the code to run for hours or days, it is impossible during an internship of fixed duration.

Several improvements are possible to the HMC algorithm. These were not implemented due to time constraints, but are available in [10],[9]. These include techniques such as the No U-Turn system to reduce the number of samples needed in HMC for an accurate estimate of the target distribution.

Several improvements are also possible to improve the performance of the programme itself. Most simply is naturally the transfer of the project to a computer with greater power than those used in the project. The existing code may also be refactored to improve performance. Several decisions were made in the design of the programme to improve readability or debugging during development, which may now be redesigned for pure efficiency. Immediate examples of this are to analytically solve much of the algebra currently being handled during runtime in the code, or to store computationally expensive results (such as various repeated commutators) between loops or sections of the algorithm to remove duplicate calculations.

Extremely costly to efficiency, however, is the implementation of the algorithm in Python. For its speed of development and wide ecosystem of relevant modules, the programme was written in Python as the most effective option to allow for maximum progress in the short duration of this project. However, features such as dynamic type and lack of compilation which allowed for this rapid development also greatly impede Python's efficiency. The most effective improvement to increase the efficacy of the simulation for long-term, in-depth studies is to port the programme to another language.

Possible other languages include Julia and C++, both known for high efficiency. More specialised languages such as Fortran may also allow for even greater efficiency, although would be significantly slower to develop. Python was the most feasible choice for this project, but its lack of efficiency presents a fundamental bottleneck to the simulation that these languages greatly reduce, allowing this method to be used for in-depth investigation of the BFSS model.

Chapter 7

Conclusion

This project created a programme that is believed to accurately simulate the dynamics of the discretised bosonic BFSS model via the HMC algorithm in Python. While further tests may further verify the accuracy of the simulation and yield deeper insights into the function of the programme, these are not possible given the current version of the programme and the resources available. However, evidence from the tests that were run indicate the capability of the programme to simulate the BFSS action with sufficient accuracy to yield useful results, and act as an excellent proof of concept for this method to investigate the BFSS model.

With improvements discussed above to allow for results to be generated in reasonable time, including greater computational power and the porting of the system to a language with greater efficiency, the system developed during this project may be able to generate a simulation of the dynamics of the BFSS action with sufficient accuracy and in a reasonable runtime to calculate useful results of the action.

No quantitative error analysis was performed, and the effect of the Monte Carlo error on the final results generated by the simulation is unknown. Qualitative inspection of the simulation indicates that the system converges to accurate values sufficiently well to warrant further study, however formal analysis must be completed before the simulation is used conclusively.

Following on from these improvements and verification of this version of the system, several extensions are possible using this framework. Most notably, the fermionic parts of Eq. 2.1 should be discretised and included in the simulation as in later sections of [2]. This would greatly increase the computational power and efficiency required in the simulation, but would allow for the full BFSS model to be simulated. It is our intention to continue this project to implement these improvements and extensions, and further development (both to the code and written reports)

will be published in the GitHub linked above.

As discussed above, this simulation of the full BFSS model would be immensely useful. Results for large lambda (thus approaching the continuum limit) for low energies of the system would allow for the simulation to be used in the study of several theories, including supergravity and type IIA string theory, and the conjecture that BFSS is equivalent to M-theory in the limit as N increases. This capability would be immensely powerful, allowing for investigations and insights into the theory that would be impossible with analytical methods. This project acts as an excellent proof of concept that this method may be further pursued and this full simulation may be functional as a means of study for these theories.

Bibliography

[1] Katrin Becker, Melanie Becker, and John H Schwarz. *String theory and M-theory: A modern introduction*. Cambridge university press, 2006.

[2] Veselin G Filev and Denjoe O'Connor. The bfss model on the lattice. *Journal of High Energy Physics*, 2016(5):1–25, 2016.

[3] Tom Banks, Willy Fischler, Steven H Shenker, and Leonard Susskind. M theory as a matrix model: A conjecture. In *The World in Eleven Dimensions*, pages 435–451. CRC Press, 1999.

[4] Nathan Seiberg. Why is the matrix model correct? *Physical Review Letters*, 79(19):3577, 1997.

[5] Hermann Nicolai and Robert Helling. Supermembranes and m (atrix) theory. *ICTP Spring School on Nonperturbative Aspects of String Theory and Supersymmetric Gauge Theories*, pages 29–74, 1998.

[6] Washington Taylor. M (atrix) theory: Matrix quantum mechanics as a fundamental theory. *Reviews of Modern Physics*, 73(2):419, 2001.

[7] Masanori Hanada. What lattice theorists can do for superstring/m-theory. *International Journal of Modern Physics A*, 31(22):1643006, 2016.

[8] Jean Thierry-Mieg. Geometrical reinterpretation of faddeev–popov ghost particles and brs transformations. *Journal of Mathematical Physics*, 21(12):2834–2838, 1980.

[9] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[10] Michael Betancourt. A conceptual introduction to hamiltonian monte carlo. *arXiv preprint arXiv:1701.02434*, 2017.

[11] Takeo Matsubara. A new approach to quantum-statistical mechanics. *Progress of theoretical physics*, 14(4):351–378, 1955.

[12] Alexandru Nica and Roland Speicher. Commutators of free random variables. 1998.