FuzzifiED

Julia package for numerics on the fuzzy sphere

Zheng Zhou 周正

Perimeter Institute for Theoretical Physics, Waterloo, Ontario N2L 2Y5, Canada

Department of Physics and Astronomy, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

E-mail: physics@zhengzhou.page

2 April, 2025

Abstract : The Julia package *FuzzifiED* aims at simplifying the numerical calculations on the fuzzy sphere. It supports exact diagonalisation (ED) and density matrix renormalisation group (DMRG) calculations. FuzzifiED can also apply to generic fermionic and bosonic models. This documentation provides a review of the fuzzy sphere regularisation and an instruction for using FuzzifiED for numerical calculations.

Contents

1   Purpose and outline

Since its proposal, the fuzzy sphere regularisation has made significant contributions to the study of 3d CFTs [1–20]. The Julia package FuzzifiED aims at simplifying the numerical calculations on the fuzzy sphere. It supports exact diagonalisation (ED) calculations, as well as the density matrix renormalisation group (DMRG) using the ITensor [21] library. FuzzifiED can also apply to generic fermionic and bosonic models. This package offers the following features :

1. Versatility : FuzzifiED can reproduce nearly all the ED and DMRG results from fuzzy sphere research. Its flexible design also makes it straightforward to adapt to new models.

2. Usability : The expressive Julia interface [22] simplifies coding and comprehension. To help users get started, we provide a collection of examples.

3. Efficiency : FuzzifiED produces results on reasonable system sizes within minutes.

4. Open source : The FuzzifiED codebase is freely available under the MIT License, welcoming reviews and contributions from the wider community.

This documentation provides a review of the fuzzy sphere regularisation and an instruction for numerical calculations with FuzzifiED. The rest of this documentation is organised as follows : The Part I (Sections 2–5) is devoted to a review of the fuzzy sphere regularisation and the numerical methods applied to it, mainly targeting at providing the necessary technical information for those who want to get started with the research on the fuzzy sphere. The Part II (Sections 6–9) gives a detailed instruction for numerical calculation with the package FuzzifiED.

•   In Section 2, we briefly introduce the conformal field theories in dimensions $d \geq 3$ and the fuzzy sphere regularisation.

•   In Section 3, we review the existing works related to the fuzzy sphere.

•   In Section 4, we review the setup of fuzzy sphere, the construction of interaction models and the extraction of CFT data.

•   In Section 5, we review the numerical methods used in the package FuzzifiED, *viz.* ED and DMRG.

•   In Section 6, we give an instruction for installing and using FuzzifiED.

- In Section 7, we give an instruction for performing ED calculation with FuzzifiED.

- In Section 8, we give an instruction for performing DMRG calculation with FuzzifiED.

- In Section 9, we present a collection of practical examples that reproduce many existing results.

- In Appendix A, we present the specialised data structures that we use in the ED calculation.

- In Appendix B, we present the construction of the Landau level and pseudopotentials on the torus.

- In Appendix C, we give a collection of the ED and DMRG tutorial codes.

- In Appendix D, we list a glossary for all the interfaces in FuzzifiED to facilitate searching.

Part I

Review of fuzzy sphere

2 Introduction

2.1 Conformal field theory

Conformal field theory (CFT) is one of the central topics of modern physics. It refers to a field theory that is invariant under conformal transformations that preserve the angles between vectors. In spacetime dimension $d > 2$, the global conformal symmetry transformations form a group $SO(d+1, 1)$, generated by translation, $SO(d)$ rotation[1], dilatation (scale transformation), and special conformal transformation (SCT) [23, 24]. Each CFT operator must transform under irreducible representations of rotation and dilatation. The representations are labelled by the $SO(d)$ spin l and scaling dimension Δ, respectively. A special kind of operators that are invariant under SCT, called 'primaries', deserve particular attention. By acting spatial derivatives on the primaries, their 'descendants' are obtained. Each operator in CFT is a linear combination of primaries and descendants. The conformal symmetry is the maximal spacetime symmetry (except supersymmetry) that a field theory can have. It gives powerful constraints on the property of the field theory. In particular, conformal symmetry uniquely determines the form of two-point (2-pt) and three-point (3-pt) correlation functions. The 3-pt correlator of three primary operators Φ_i, Φ_j, Φ_k contains a universal coefficient called the OPE coefficient $f_{\Phi_i \Phi_j \Phi_k}$. The collection of scaling dimensions and the OPE coefficients of primaries $\{\Delta_{\Phi_i}, f_{\Phi_i \Phi_j \Phi_k}\}$ is called the conformal data. Theoretically, with full knowledge of the CFT data, an arbitrary correlation function of a CFT can be obtained[2].

CFT has provided important insights into various aspects of theoretical physics. In condensed matter physics, it has produced useful predictions about the critical phenomena [25–27]. Many classical and quantum phase transitions are conjectured to have emergent conformal symmetry in the infrared (IR), *i.e.*, at long wavelength or low energy. The universal critical

---

[1]Here we work in Euclidean signature. In Lorentzian signature it is the Lorentz transformation $SO(1, d-1)$

[2]The full knowledge is not often possible in practice, as the number of primaries is often infinite.

exponents are directly determined by the scaling dimensions of the primary operators. *E.g.*, in the 3d Ising transition that spontaneously breaks $\mathbb{Z}_2$ symmetry, most critical exponents are given by the scaling dimensions of the lowest $\mathbb{Z}_2$-odd operator σ and $\mathbb{Z}_2$-even operator ϵ, such as

$$\eta = 2\Delta_\sigma - 1, \qquad \nu = \frac{1}{3 - \Delta_\epsilon}. \tag{2.1}$$

CFT is also closely related to string theory and quantum gravity in high-energy physics. In the string theory, CFT describes the 2d worldsheet [28]; in quantum gravity, there is a conjectured duality between the gravity theory in $(d+1)$-dimensional anti-de Sitter (AdS) space in the bulk and a d-dimensional CFT on the boundary [29]. Moreover, CFT plays an important role in our understanding of quantum field theories. It describes many fixed points in the RG flow, and many QFTs can be seen as CFTs with perturbations. It also helps us understand how physics changes under a change of scale and reveals some fundamental structures of the RG flow [30].

In 2d CFTs, besides the global conformal symmetry SO(3, 1), there also exists an infinite-dimensional local conformal symmetry [31, 32]. Altogether, they form the Virasoro algebra. The infinite-dimensional conformal algebra has made many theories exactly solvable, especially the rational theories such as the minimal models [33] and, more generally, the Wess-Zumino-Witten (WZW) theories [34, 35]. However, going to the higher dimensions, the CFTs are much less well-studied due to a much smaller conformal group. The existing methods include numerical conformal bootstrap and Monte Carlo lattice simulations. Numerical bootstrap bounds the conformal data by making use of consistency conditions such as reflection positivity and crossing symmetry, together with some information of the CFT such as the global symmetry and a certain amount of assumptions [36, 37]. It has achieved great success in 3d Ising [38, 39], O(N) Wilson-Fisher [39, 40], Gross-Neveu-Yukawa CFTs [41], *etc.* On the other hand, one can study a CFT by constructing a lattice model that goes through a phase transition in the corresponding universality class, and study the phase transition by Monte Carlo simulation. This method has achieved success in many phase transitions assuming conformal symmetry, *e.g.*, the 3d Ising model [42]. However, the extraction of universal data usually involves complicated and expensive finite-size scaling [26, 43, 44], and only the lowest few CFT operators can be accessed.

Among these higher dimensional CFTs, we mainly focus on $d = 3$, as many Lagrangians in $d \geq 4$ flow to free theories [27].

2.2 Fuzzy sphere

In addition to these existing approaches, the *'fuzzy sphere regularisation'* has recently emerged as a new powerful method to study 3d CFTs. It involves studying interacting electrons moving on a sphere under the influence of a magnetic monopole at its centre.

The idea begins with putting an interacting quantum Hamiltonian on a 2-sphere S^2. This geometry preserves the full rotation symmetry (on the contrary, lattice models often only preserve a discrete subgroup). Moreover, when the system is tuned to a critical point or critical phase, combined with the time evolution direction, the system is described by a quantum field theory living on a generalised cylinder $S^2 \times \mathbb{R}$, a manifold that is conformally equivalent to flat spacetime through the Weyl transformation

$$(\hat{\mathbf{n}}, \tau) \in S^2 \times \mathbb{R} \longmapsto r\hat{\mathbf{n}} \in \mathbb{R}^3, \qquad r = e^{\tau/R} \tag{2.2}$$

where R is the radius of the sphere. This conformal transformation maps each time slice of the cylinder to a concentric sphere in the flat spacetime.

Thanks to the conformal flatness that is not owned by other manifolds (*e.g.*, a lattice model with periodic boundary condition lives on the torus T^2, which is not conformally flat), we can make use of some nice properties of conformal field theories. The most important one is the state-operator correspondance [23, 24, 45]. Specifically, there is a one-to-one correspondence between the eigenstates of the critical Hamiltonian on the sphere and the CFT operators. One can colloquially understand the state $|\Phi\rangle$ as the insertion of the corresponding operator $\Phi(0)$ at the origin point into the vacuum $|0\rangle : |\Phi\rangle = \hat{\Phi}(0)|0\rangle$. The state and its corresponding operator have the same SO(3) spin and representation under global symmetry. More importantly, as the Weyl transformation maps the Hamiltonian H that generates the time translation on the cylinder to the dilatation D on the flat spacetime, the excitation energy of a state $|\Phi\rangle$ is proportional to the scaling dimension of the corresponding operator Δ_Φ

$$E_\Phi - E_0 = \frac{v}{R}\Delta_\Phi, \tag{2.3}$$

where E_0 is the ground state energy, R is the radius of the sphere, and v is the speed of light that is dependent on the microscopic model and is the same for every state. With this property, one can calculate the scaling dimensions simply by solving the energy spectrum of the quantum Hamiltonian without doing complicated finite-size scalings, and one can obtain the OPE coefficients simply by taking the inner product of a local operator.

Although the quantum Hamiltonians on a sphere enjoy the full rotation symmetry and the property of state-operator correspondence, it is difficult to put a lattice on the sphere due to the curvature (in particular the non-zero Euler characteristic), especially to recover an SO(3)-symmetric thermodynamic limit [46]. An alternative way is to fuzzify the sphere [47]. We consider charged free particles moving on a sphere with a magnetic monopole with a flux $4\pi s$ ($s \in \mathbb{Z}/2$) placed at its centre. The monopole exerts a uniform magnetic field on the sphere, which modifies the single-particle Hamiltonian and the single-particle eigenstates. Now, the single particle eigenstates form highly degenerate spherical Landau levels [48–51]. The lowest Landau level has a degeneracy $(2s+1)$. By setting the single-particle gap to be the leading energy scale, and projecting onto the lowest Landau level, we obtain a finite Hilbert space. For the purpose of numerical simulation, the system is analogous to a length-$(2s + 1)$ chain with long-range interaction, where different Landau level orbitals behave like the lattice sites. The difference is that the $(2s+1)$ orbital forms a spin-s representation of the SO(3) rotation group, and in this way, the continuous rotation symmetry is preserved. By putting multiple flavours on an orbital and adding interactions, various 3d CFTs can be realised. The interaction Hamiltonians are designed through matching the global symmetry and phase diagram. The word 'fuzzy' means the non-commutativity, in our case, due to the presence of magnetic field [47, 51]. The non-commutativity provides a natural length scale which serves as a UV regulator of the quantum field theory. The radius of the sphere scales as $R \sim \sqrt{s}$. The thermodynamic limit can be taken as $s \to \infty$, and we then recover a regular sphere without non-commutativity.

The power of this approach has been first demonstrated in the context of the 3d Ising transition [1], where the presence of emergent conformal symmetry has been convincingly established, and a wealth of conformal data has been accurately computed. The study has then been extended to accessing various conformal data such as the OPE coefficients [2], correlation functions [3], entropic F-function [10], conformal generators [14, 15], studying conformal

defects and boundaries such as the magnetic line defect [6, 9], various conformal boundaries in the 3d Ising CFT [12, 13], and realising various 3d CFTs such as Wilson-Fisher CFTs [8], SO(5) deconfined criticality [4], *etc*. In the following sections, we shall review the existing works, technical details and numerical methods.

3 Review of existing works

In this section, we review the existing works related to the fuzzy sphere.

The pioneering work [1] This work first proposes the idea of the fuzzy sphere and applies it to a pedagogical example of the 3d Ising CFT. Zhu *et al.* construct a model with two flavours of fermions that resemble the spin-up and spin-down in the lattice transverse-field Ising model. At half-filling, one can colloquially think that a spin degree of freedom lives on each orbital. The Hamiltonian contains a density-density interaction[3] that resembles the Ising ferromagnetic interaction and a polarising term that resembles the transverse field. By tuning the ratio between the two terms, a transition between quantum Hall ferromagnet [52, 53] (a two-fold degenerate state where either of the two flavours is completely occupied) and paramagnet (a one-fold degenerate state where the superposition of the two flavours at each orbital is occupied) occurs. This transition spontaneously breaks a $\mathbb{Z}_2$ symmetry and falls into the Ising criticality. They then make use of a unique feature of spherical models described by CFT — state-operator correspondence — at the critical point to extract the scaling dimensions of the scaling local operators. They find evidence for the conformal symmetry, including that (1) there exists a conserved stress tensor with $\Delta = 3$ (which is used as the calibrator), and (2) all the levels can be classified into conformal multiplet where the spacings between operators' scaling dimensions are very close to integer. This is one of the first numerical evidence that the 3d Ising transition has emergent conformal symmetry. More remarkably, the scaling dimensions of primaries such as $\sigma, \epsilon, \epsilon'$ are already very close to the most accurate known value by numerical bootstrap with an error within 1.2% at a small system size with the number of orbitals $N_m = 16$, for which the computational cost is comparable to a 4×4 lattice system. The structure of the Ising CFT operator spectrum already starts to show up at an even smaller system size $N_m = 4$. All these

---

[3] Here the density operator refers to a local fermion bilinear.

– 7 –

clues point towards a curious observation that the fuzzy sphere suffers from a remarkably small finite-size effect. The detail for the construction of models is presented in Sections 4.3 and 4.4, and the detail for the analysis of the spectrum is presented in Section 4.5.

This seminal work opens a new avenue for studying 3d conformal field theories. After that, most of the research on the fuzzy sphere can roughly be categorised into three directions (with several exceptions) :

1. Accessing various conformal data,
2. Realising various 3d CFTs, and
3. Studying conformal defects and boundaries.

3.1 Accessing various conformal data

The first direction is to develop methods to calculate various universal data of 3d CFTs on the fuzzy sphere. Typically, these methods are tested with the simplest example of the 3d Ising CFT. For many of those CFT data, the fuzzy sphere is the first non-perturbative method to access them ; for the others, the fuzzy sphere has achieved great consistency with previous methods such as quantum Monte Carlo and conformal bootstrap. So far, the accessible CFT data include operator spectrum, OPE coefficients [2], correlation functions [3], entropic F-function [10] and conformal generators [14, 15].

OPE coefficients [2]   Apart from the operator spectrum, a wealth of CFT data can be obtained from the local operators. This work studies the local observables on the fuzzy sphere, including the density operators and certain four-fermion operators. These observables can be expressed as the linear combination of local scaling operators in the CFT. After a finite-size scaling with the data from different system sizes, the subleading contribution can be subtracted, and only the leading contribution is left. In this way, the lowest primaries in Ising CFT in each symmetry sector, *viz.* the $\mathbb{Z}_2$-odd σ and the $\mathbb{Z}_2$-even ϵ, can be realised. The OPE coefficients are then evaluated by taking the inner product of a fuzzy sphere local observable with two CFT states $\langle\Phi_1|\Phi_2(\hat{\mathbf{n}})|\Phi_3\rangle$. Hu *et al.* compute 17 OPE coefficients of low-lying CFT primary fields with high accuracy, including four that have not been reported before. The rest are consistent with numerical bootstrap results. It is also worth noting that this work starts to apply DMRG to

the fuzzy sphere. The maximal system size is increased from $N_m = 18$ by ED to $N_m = 48$ by DMRG. The detail for calculating the OPE coefficient is presented in Section 4.6.

Correlation functions [3] In addition to the OPE coefficients, the local observables can also be used to calculate correlation functions. By taking the inner product of two local observables (density operators) at a time displacement $\langle \Phi_1 | \Phi_2(\hat{n}_0) \Phi_3(\hat{n}, \tau) | \Phi_4 \rangle$ with two CFT states, a general 4-pt function can be calculated. In practice, this piece of CFT data cannot be derived from the scaling dimensions and the OPEs due to the existence of infinitely many primaries. Han *et al.* calculate the 4-pt functions in the 3d Ising CFT with DMRG. A non-trivial check of conformality, the crossing symmetry, is verified for the correlator $\langle \sigma\sigma\sigma\sigma \rangle$. The special case — 2-pt functions by taking $\Phi_1 = \Phi_4 = \mathbb{I}$ — is also studied and compared with the expected results by conformal symmetry. The detail for calculating the correlation functions is presented in Section 4.6.

Entropic F-function [10] Beyond the correlators of local operators, a wealth of information can be learnt from the entanglement entropy and entanglement spectrum. A remarkable quantity is called the F-function, which is defined through the scaling behaviour of the entanglement entropy [54–58]. Specifically, consider a quantum system that lives on $\mathbb{R}^2$. A circle with radius R_d divides the system into inner part A and outer part B. The entanglement entropy is defined and expected to scale with R_d as

$$S_A(R_d) = -\text{tr}_A \rho \log \rho = \alpha R_d/\delta - F, \tag{3.1}$$

where δ is a UV-regulator. The constant piece is known as the F-function of a 3d CFT. The F-function is proved to be RG-monotonic, *i.e.*, along a renormalisation group flow from UV to IR, the value of F-function is non-increasing, analogous to the central charge in 2d CFTs. Despite its importance, it has never been calculated before through non-perturbative approaches in interacting 3d CFTs. This work has performed the first non-perturbative computation of F function for the 3d Ising CFT on the fuzzy sphere. The sphere is cut in the real space into two crowns along a latitude circle θ, and the entanglement entropy $S_A(\theta)$ as a function of θ is calculated [59–62]. The F-function is extracted from the $S_A(\theta)$ in the vicinity of the equator, and the result yields $F = 0.0612(5)$ after a finite-size scaling.

Conformal generators [14, 15]   Within the generators of conformal symmetry, the SO(3) rotation and the dilatation are manifest and act as rotation and time translation on the fuzzy sphere. The rest two, *viz.* translation P^μ and special conformal transformation (SCT) K^μ, need to be emergent in the IR at the conformal point. It is worthwhile to construct these IR generators by the UV operators on the fuzzy sphere. These works investigate such construction with the help of stress tensor $T^{\mu\nu}$. The time component $T^{\tau\tau}$ of stress tensor equals the Hamiltonian density $\mathcal{H}$ and it integrates into the generator $\Lambda^\mu = P^\mu + K^\mu = \int \mathrm{d}^2\hat{\mathbf{n}} \, 2n^\mu \mathcal{H}$. The action of this generator sends a scaling operator to other operators in the same multiplet, with the number of spatial derivatives increased or decreased by one. Fardelli *et al.* and Fan calculate the matrix elements of the generators Λ^μ and find good agreement with the theoretical values in the CFT, which is another non-trivial verification of conformal symmetry. Furthermore, the separate generators P^μ and K^μ can be obtained by considering the commutator $[H, \Lambda^\mu]$, which is useful in determining the primaries. The detail for constructing the conformal generators is presented in Section 4.7.

3.2   Realising various 3d CFTs

The second direction is to study various other CFTs beyond 3d Ising. The fuzzy sphere has revealed lots of new information about these theories ; the previously known results are also consistent with the fuzzy sphere. So far, the accessible CFTs include SO(5) deconfined criticality [4], O(3) Wilson-Fisher [8], a series of new theories with Sp(N) symmetry [16], and the 3-state Potts model [18].

The SO(5) **deconfined criticality [4]**   The first theory besides Ising CFT to which fuzzy sphere is applied is the SO(5) deconfined quantum critical point (DQCP). Deconfined quantum critical point (DQCP) is one of the pioneering examples of phase transitions beyond Landau paradigm [63–65]. It has led to numerous theoretical surprises, including the emergent SO(5) symmetry [66] and the duality between interacting theories [67]. Despite extensive studies over the past two decades, its nature remains controversial. Numerical simulations have shown no signal of discontinuity, but abnormal scaling behaviours have been observed [65]. A plausible proposal to reconcile the tension is that DQCP is pseudocritical, *i.e.*, a weakly first-order

phase transition that has approximate critical behaviour, and is controlled by a pair of complex fixed points very close to the pseudocritical region [67, 68].

The DQCP can be conveniently studied on the fuzzy sphere by constructing a non-linear sigma model (NLσM) on target space S^4 with a level-1 topological Wess-Zumino-Witten (WZW) term, which serves as a dual description of the DQCP with an exact SO(5) symmetry [66, 67]. The idea is to construct a 4-flavour model with symmetry $\mathrm{Sp}(2)/\mathbb{Z}_2 = \mathrm{SO}(5)$ ($\mathbb{Z}_2$ means to gauge the pseudoreal representations). At half-filling, it can be described by a NLσM on the Grassmannian $\frac{\mathrm{Sp}(2)}{\mathrm{Sp}(1)\times\mathrm{Sp}(1)} \cong S^4$ and the WZW level can be matched [69–71]. This work provides evidence that the DQCP exhibits approximate conformal symmetry. Zhou *et al.* have identified 19 conformal primaries and their 82 descendants. Furthermore, by examining the renormalisation group flow of the lowest symmetry singlet, they demonstrate that the DQCP is more likely pseudocritical, with the approximate conformal symmetry plausibly emerging from nearby complex fixed points. Several works [72, 73] appear later to follow up.

The O(3) **Wilson-Fisher [8]**   The O(N) Wilson-Fisher (WF) theories are probably one of the most studied theories for 3d criticalities with a wide range of applications [26, 27, 74]. Specifically, this work focuses on the O(3) WF CFT. The construction involves two copies of SU(2) ferromagnet with altogether four flavours. Briefly speaking, the model contains two competing terms : (1) a SU(2) ferromagnetic interaction which favours a Heisenberg ferromagnetic phase where each of the two copies is half-filled and the symmetry-breaking order parameter lives on a S^2 manifold, (2) a polarising term which favours one of the two copies being completely filled, corresponding to a Heisenberg paramagnet. The transition between these two phases falls into the O(3) Wilson-Fisher universality. Through the energy spectrum at the transition, Han *et al.* provide evidence that O(3) Wilson-Fisher fixed point exhibits conformal symmetry, as well as revealing a wealth of information about the CFT, *e.g.*, the instability to cubic anisotropy. They also calculate several OPE coefficients.

A series of new Sp(N)**-symmetric CFTs [16]**   The quest to discover new 3d CFTs has been intriguing for physicists. A virgin land on this quest is the parity-breaking CFTs. In 3d, the Chern-Simons-matter theories stand out as the most well-known and possibly the only known type of parity-breaking CFTs. The fuzzy sphere is a promising platform for studying these

theories. This work makes a concrete construction by generalising the DQCP to the WZW-NLσM on the target space of a general symplectic Grassmannian

$$\frac{\mathrm{Sp}(N)}{\mathrm{Sp}(M) \times \mathrm{Sp}(N-M)}.\tag{3.2}$$

Several candidate Chern-Simons-matter theories are known to exist on its phase diagram with N flavour of gapless bosons or fermions coupled to a non-Abelian (*viz.* Sp(1), Sp(2), *etc.*) Chern-Simons gauge field [75]. On the fuzzy sphere, this WZW-NLσM can be realised by a $2N$ layer model with $\mathrm{Sp}(N)$ flavour symmetry, and $2M$ out of the $2N$ layers are filled. Zhou *et al.* numerically verify the emergent conformal symmetry by observing the integer-spaced conformal multiplets and studying the finite-size scaling of the conformality.

The 3-state Potts model [18] The Potts models describe transitions that spontaneously break S_Q symmetries where $Q \in \mathbb{Z}$ is known as the number of states. In 2d, the transitions with $Q \leq Q_c = 4$ are continuous and captured by CFTs [76], while $Q > Q_c$ are first order. Specifically, 2d 5-state Potts transition is pseudocritical and described by a pair of complex CFTs in its vicinity [77] in a similar manner with the conjectured SO(5) DQCP. In 3d, the 3-state Potts model is found to be first-order [78, 79]. This work constructs a 3-flavour model on the fuzzy sphere with S_3 permutation symmetry among flavours. The interacting Hamiltonian resembles the Ising model. Interestingly, Yang *et al.* find out that the transition point of the 3d 3-state Potts model, despite being probably first-order, exhibits approximate conformal symmetry, indicating that there might be an underlying CFT describing it. However, it is difficult to determine the nature of the transition from the operator spectrum (specifically, from the relevance of the second singlet ϵ') due to the complicated finite-size effect.

3.3 Studying conformal defects and boundaries

Apart from the bulk CFTs, the fuzzy sphere can also be used to study their conformal defects and boundaries. Deforming a CFT with interactions living on a sub-dimensional defect may trigger a RG flow towards a non-trivial interacting IR fixed point. A defect IR theory with a smaller conformal symmetry is called a defect CFT [80, 81]. The dCFTs have rich physical structures, such as defect operators and bulk-to-defect correlation functions. Moreover, a bulk CFT can flow to several different dCFTs. Similarly, deformation on the boundary may trigger

a flow towards a boundary CFT (bCFT). So far, the accessible defects/boundaries include the magnetic line defect of the 3d Ising CFT, in particular, its defect operator spectrum, correlators [6], g-function, defect-changing operators [9], its cusp [11], and the conformal boundaries of the 3d Ising CFT [12, 13].

Conformal magnetic line defect [6]   This is the first work that studies conformal defects with the fuzzy sphere. The simplest example of conformal defect is the magnetic line defect of the 3d Ising CFT [82–85], where the defect line is completely polarised and the $\mathbb{Z}_2$ symmetry is explicitly broken. A defect line along z-direction that passes the origin point, after the radial quantisation, corresponds to the north and south poles of the sphere being polarised. Hence, to realise the magnetic line defect on the fuzzy sphere, one only needs to apply a pinning magnetic field to the north and south poles (Since only the $m = +s$ orbital has non-zero amplitude at the north pole and $m = -s$ at the south pole due to the locality, one only need to pin the $m = \pm s$ orbitals).

This work demonstrates that the defect IR fixed point has emergent conformal symmetry $\mathrm{SO}(2,1) \times \mathrm{O}(2)$: in the operator spectrum, there exists a displacement operator as the non-conservation of stress tensor at exactly $\Delta_\mathrm{D} = 2$, and the defect primaries and descendants have integer spacing ; the bulk-to-defect 1-pt and 2-pt correlation functions follow a power law. Hu *et al.* have identified six low-lying defect primary operators, extracted their scaling dimensions, and computed the 1-pt function of bulk primaries and 2-pt bulk-to-defect correlators.

The g-function and defect-changing operators [9]   This work studies the g-function of conformal defects and the defect-creation and changing operators. Similar to the central charge and the F-function in bulk CFTs, there exists a RG-monotonic quantity called the g-function for the line defects that is non-increasing along the flow [86, 87]. It is defined as the ratio between the partition functions of the defect CFT and the bulk CFT. On a different note, consider two semi-infinite magnetic line defects pinned towards opposite directions joint at one point, a defect-changing operator lives at the joining point [88, 89]. Similarly, a defect-creation operator lives at the endpoint of a semi-infinite line defect. The relevance of the defect-changing operator is related to the stability of spontaneous symmetry breaking (SSB) on the line defect.

Zhou *et al.* realise the defect-creation and changing operators for the Ising magnetic line defect by acting a pinning field at the north pole, and opposite pinning fields at the north and south poles, respectively. The scaling dimensions are calculated through state-operator correspondence $\Delta_{\text{creation}} = 0.108(5), \Delta_{\text{changing}} = 0.84(5)$, indicating the instability of SSB on the Ising magnetic line. Moreover, they show that the g-function and many other CFT data can be calculated by taking the overlaps between the eigenstates of different defect configurations. Notably, this paper has given the first non-perturbative result for the g-function $g = 0.602(2)$.

Cusp [11]   A cusp is two semi-infinite defect lines joined at one point at an angle. This can be realised on the fuzzy sphere through pinning fields at two points at an angle. Cuomo *et al.* study the cusps through various theoretical and numerical approaches. In particular, on the fuzzy sphere, they calculate the cusp anomalous dimension as a function of the angle for the Ising magnetic line defect, and verify its relation with the Zamolodchikov norm of the displacement operator.

Conformal boundaries of 3d Ising CFT [12, 13]   Apart from line defects, boundaries are also important extended objects in CFT. For the Ising CFT, there exist several conformal boundaries : normal bCFT with explicitly broken $\mathbb{Z}_2$ symmetry, ordinary bCFT that is stable and has preserved $\mathbb{Z}_2$ symmetry, extraordinary bCFT with spontaneously broken $\mathbb{Z}_2$ symmetry, and special bCFT as the transition between ordinary and extraordinary bCFTs [90−92]. These works focus on the normal and ordinary bCFTs and show that they can be realised by acting a polarising field on a hemisphere. By noting that the LLL orbitals are localised along latitude circles, the bCFTs can equivalently be realised by pinning the orbitals with $m < 0$. By studying the operator spectrum, Zhou *et al.* and Dedushenko *et al.* show numerical evidence for conformal symmetry and estimate the scaling dimensions of the conformal primaries. They also calculate the bulk-to-boundary 1-pt and 2-pt functions and extract the corresponding OPE coefficients. Interestingly, Zhou *et al.* notice a certain correspondence between the boundary energy spectrum and bulk entanglement spectrum through the orbital cut.

3.4 Other works on the fuzzy sphere

Besides the three directions of works, several other works push the frontier of our knowledge of and techniques for the fuzzy sphere.

Conformal perturbation [20]   The energy spectrum calculated numerically at a finite size does not coincide with that of the CFT. Part of the finite-size correction comes from the higher irrelevant operators that are not precisely tuned to zero (*e.g.*, in the Ising CFT, the irrelevant operators include ϵ', $C_{\mu\nu\rho\sigma}$, $T'_{\mu\nu}$, *etc.*, and the lowest singlets ϵ and ϵ' are tuned away through the two parameters). These irrelevant operators exert perturbations on the states and their energies. This paper captures this kind of correction using the conformal perturbation theory. By making use of the fact that the corrections from an irrelevant operator on the energy of the primary and its descendants are not independent, the coefficients of the irrelevant operators can be fitted and their corrections can be removed.

The conformal perturbation theory is first studied to the icosahedron [5] and then applied to the fuzzy sphere. This work opens up a new route to improving the precision of scaling dimensions on the fuzzy sphere by making better use of the existing data. The method to partly remove the finite-size correction through conformal perturbation theory is widely used by the following works.

Quantum Monte Carlo on fuzzy sphere [7]   Until this work, the numerical methods that have been applied to the fuzzy sphere include exact diagonalisation (ED) and density matrix renormalisation group (DMRG). Hofmann *et al.* further present the numerical studies of the fuzzy sphere with quantum Monte Carlo (QMC) simulation, which is known for its potential for studying criticalities in $(2 + 1)$ dimensions at larger system size. Specifically, they make use of the determinant quantum Monte Carlo (DQMC) method that converts the simulation of fermions into the simulation of bosonic auxiliary fields. To overcome the sign problem, they consider two copies of the original model and construct the Ising CFT on a 4-flavour model. They determine the lowest energy spectra within each symmetry sector by calculating the time-displaced correlation functions. They also calculate the equal-time correlation functions and compare them with the 2-pt functions of CFT.

Ising CFT from FQHE state [17]   Until this work, all the constructions of CFTs on the fuzzy sphere were based on the quantum Hall ferromagnet. Specifically, in the absence of the interaction, an integer number of the lowest Landau levels are fully occupied. This state has a finite charge gap that guarantees that the gapless spin degree of freedom does not strongly couple with the charge degree of freedom when one adds the interactions.

Voinea *et al.* further explore the possibility of constructing CFTs on other states with charge gap — in particular, the Haldane-Laughlin states that capture the fractional quantum Hall effect (FQHE) [48, 93]. Specifically, they study the fermionic LLL at fillings of $\nu = 1/3$ and $1/5$. The model Hamiltonian contains (1) a dominant projection term that puts the ground state on the Haldane-Laughlin state, and (2) an interaction term as a perturbation that drives the Ising-type phase transition. They show that the energy spectra at the critical point exhibit conformal symmetry. Notably, they also make the construction with respect to the bosonic LLL at a filling of $\nu = 1/2$.

Fuzzy circle [19]   Besides the fuzzy sphere S^2, the regularisation with the lowest Landau level can also be used for other manifolds or dimensions. This work studies the 2d CFT on a 'fuzzy circle'. Although Landau levels can only be defined on even space dimensions [94], one can reach odd space dimensions by compactifying one of the even dimensions. Specifically, Han *et al.* construct the LLL on a thin torus T^2 and sends one of the lengths to infinity while fixing the other. In this way, a circle S^1 is recovered in the thermodynamic limit. They construct the 2d Ising and 3-state Potts CFTs on the fuzzy circle and compare the operator spectrum and OPE coefficients with the Virasoro multiplet structure and the known results in the exactly solvable minimal models.

4   Model construction

In this section, we review the process of constructing a model on the fuzzy sphere and extracting the conformal data. We aim to provide the necessary technical information for those who want to get started with the research on the fuzzy sphere, especially the aspects rarely covered by other literature.

4.1 Projection onto the lowest Landau level

To build the setup of the fuzzy sphere, we consider a sphere with radius R and put a $4\pi s$-monopole at its centre. Consider free electrons moving on the sphere. The monopole modifies the single particle Hamiltonian [48–51]

$$H_0 = \frac{1}{2MR^2}(\partial^\mu + iA^\mu)^2, \tag{4.1}$$

where $\mu = \theta, \phi$ and the gauge connection is taken as

$$A_\theta = 0, \quad A_\phi = -\frac{s}{R}\,\mathrm{ctg}\,\theta. \tag{4.2}$$

The eigenstates of the Hamiltonian are the monopole spherical harmonics

$$Y_{lm}^{(s)}(\hat{\mathbf{n}}), \quad l = s, s+1, \ldots, \quad m = -l, \ldots, l-1, l, \tag{4.3}$$

where $\hat{\mathbf{n}}$ is the unit vector of the point on the sphere specified by angular coordinates θ and ϕ, and the energies are

$$E_l = \frac{1}{2MR^2}(l(l+1) - s^2). \tag{4.4}$$

Each level, known as a Landau level, has a degeneracy $(2l+1)$. Specifically, the wavefunctions on the lowest Landau level (LLL) $l = s$ are easy to write out :

$$Y_{sm}^{(s)}(\hat{\mathbf{n}}) = C_m e^{im\phi} \cos^{s+m}\frac{\theta}{2}\sin^{s-m}\frac{\theta}{2}, \quad C_m = \frac{1}{\sqrt{4\pi\mathrm{B}(s+m+1, s-m+1)}}, \tag{4.5}$$

where C_m is the normalising factor, and B is the Euler's beta function. The LLL has a degeneracy $N_m = 2s + 1$.

We now consider N_f flavours of fermions moving on the sphere, characterised by the second-quantised fermion operator $\hat{\psi}_f(\hat{\mathbf{n}})$, with a flavour index $f = 1, \ldots, N_f$. We partially fill the lowest Landau level and set the single energy gap to be much larger than the scale of interaction $H_0 \gg H_{\mathrm{int}}$, so that the quantum fluctuation can be constrained on the lowest Landau level. In practice, we often fill integer number of flavours $N_e = kN_m$ ($k \in \mathbb{Z}$) so that a quantum Hall ferromagnet (i.e., the state where integer number of LLLs are fully filled) is preferred in the absence of interaction, for which the charge degree of freedom is gapped and does not couple strongly to the gapless CFT degree of freedom when the interactions are introduced.

We then project the system onto the LLL. Technically, this can be done by writing the fermion operators in terms of the annihilation operators of the LLL orbitals

$$\hat{\psi}_f(\hat{\mathbf{n}}) = \sum_{m=-s}^{s} Y_{sm}^{(s)}(\hat{\mathbf{n}})\hat{c}_{mf}, \tag{4.6}$$

where $\hat{c}_{mf}^{(\dagger)}$ annihilates/creates an electron with L^z-quantum number m at the f-th flavour of the lowest Landau level. Hereafter, we will omit the hats on the operators.

After the projection, we obtain a finite Hilbert space on which numerical simulations can be carried out. For this purpose, the system is analogous to a length-$(2s+1)$ spin chain with long-range interaction, where different Landau level orbitals behave like the lattice sites. The difference is that the $(2s+1)$ orbital forms a spin-s representation of the SO(3) rotation group, and in this way the continuous rotation symmetry is preserved. The exact rotation symmetry shortens the RG flow from the UV to the IR and reduces the finite-size effect, so that the numerical results are considerably accurate even at a small system size.

The word '*fuzzy*' means non-commutativity [47]. Here, the magnetic field results in the non-commutativity of the coordinates. More concretely, we write the coordinate operators as a matrix on the lowest Landau level

$$X_{m_1 m_2}^{\mu} = \int \mathrm{d}^2\hat{\mathbf{n}}\, n^{\mu} \bar{Y}_{sm_1}^{(s)}(\hat{\mathbf{n}}) Y_{sm_2}^{(s)}(\hat{\mathbf{n}}). \tag{4.7}$$

These matrices $\mathbf{X}^{\mu}$ ($\mu = x, y, z$) satisfy relation [4, 51]

$$\mathbf{X}_{\mu}\mathbf{X}^{\mu} = \frac{s}{s+1}\mathbb{I}, \qquad [\mathbf{X}^{\mu}, \mathbf{X}^{\nu}] = \frac{1}{s+1}i\epsilon^{\mu\nu\rho}\mathbf{X}_{\rho}. \tag{4.8}$$

The first equation involves the radius R of the sphere, and the second equation involves the magnetic length l_B that determines the non-commutativity.

$$\mathbf{X}_{\mu}\mathbf{X}^{\mu} = R^2\mathbb{I}, \qquad [\mathbf{X}^{\mu}, \mathbf{X}^{\nu}] = l_B^2\, i\epsilon^{\mu\nu\rho}(\mathbf{X}_{\rho}/R).$$

We can take $l_B = 1$ as the unit length. In this way, the radius scales with the square root of the number of orbitals

$$R/l_B = [s(s+1)]^{1/4} \sim \sqrt{N_m}. \tag{4.9}$$

The thermodynamic limit can be taken as $N_m \to \infty$, where a regular sphere is recovered.

4.2 Density operator

Having constructed the single-particle states, we then consider the interacting many-body Hamiltonian. The simplest building block is the density operator, *i.e.*, local fermion bilinear

$$n_M(\hat{\mathbf{n}}) = \psi^\dagger_{f'}(\hat{\mathbf{n}}) M_{f'f} \psi_f(\hat{\mathbf{n}}). \tag{4.10}$$

Here, the matrix insertion M puts the density operators in a certain representation of the flavour symmetry. For example, for a 2-flavour system, M can be taken as the Pauli matrices $\mathbb{I}, \sigma^x, \sigma^y, \sigma^z$; for a system with N_f flavours in the fundamental representation of $\mathrm{SU}(N_f)$ flavour symmetry, one can put n_M in the singlet or adjoint representation

$$n_S(\hat{\mathbf{n}}) = \psi^\dagger_c(\hat{\mathbf{n}}) \psi^c(\hat{\mathbf{n}})$$
$$n_a{}^b(\hat{\mathbf{n}}) = \psi^\dagger_a(\hat{\mathbf{n}}) \psi^b(\hat{\mathbf{n}}) - \tfrac{1}{N} \delta_a{}^b \psi^\dagger_c(\hat{\mathbf{n}}) \psi^c(\hat{\mathbf{n}}). \tag{4.11}$$

Like the fermion operator, the density operator can also be expressed in the orbital space[4]

$$n_M(\hat{\mathbf{n}}) = \sum_{lm} Y_{lm}(\hat{\mathbf{n}}) n_{M,lm}. \tag{4.12}$$

Conversely,

$$
\begin{aligned}
n_{M,lm} &= \int \mathrm{d}^2\hat{\mathbf{n}} \, \bar{Y}_{lm}(\hat{\mathbf{n}}) n_M(\hat{\mathbf{n}}) \\
&= \int \mathrm{d}^2\hat{\mathbf{n}} \, \bar{Y}_{lm}(\hat{\mathbf{n}}) \left(\sum_{m_1} \bar{Y}^{(s)}_{sm_1}(\hat{\mathbf{n}}) c^\dagger_{m_1 f_1} \right) M_{f_1 f_2} \left(\sum_{m_2} Y^{(s)}_{sm_2}(\hat{\mathbf{n}}) c_{m_1 f_2} \right) \\
&= \sum_{m_1 m_2} c^\dagger_{m_1 f_1} M_{f_1 f_2} c_{m_1 f_2} \int \mathrm{d}^2\hat{\mathbf{n}} \, \bar{Y}_{lm}(\hat{\mathbf{n}}) \bar{Y}^{(s)}_{sm_1}(\hat{\mathbf{n}}) Y^{(s)}_{sm_2}(\hat{\mathbf{n}}) \\
&= \sum_{m_1} c^\dagger_{m_1 f_1} M_{f_1 f_2} c_{m+m_1, f_2} \times \\
&\qquad (-1)^{s+m+m_1} (2s+1) \sqrt{\frac{2l+1}{4\pi}} \begin{pmatrix} s & l & s \\ m_1 & m & -m_1-m \end{pmatrix} \begin{pmatrix} s & l & s \\ s & 0 & -s \end{pmatrix}. \tag{4.13}
\end{aligned}
$$

Here, we have used the properties of the monopole spherical harmonics [95, 96]

---

[4]Under this convention, the action of $n_{M,lm}$ will decrease the angular momentum L^z by m. Our convention of n_{lm} differs from Ref. [2] by a factor of R^2. We take this convension in alignment with the implementation in the code.

$$\bar{Y}_{lm}^s = (-1)^{s+m} Y_{l,-m}^{(-s)} \tag{4.14a}$$

$$\int \mathrm{d}^2\hat{\mathbf{n}}\, Y_{lm}^{(s)} \bar{Y}_{lm}^{(s)} = \delta_{ll'} \delta_{mm'} \tag{4.14b}$$

$$\int \mathrm{d}^2\hat{\mathbf{n}}\, Y_{l_1 m_1}^{(s_1)} Y_{l_2 m_2}^{(s_2)} Y_{l_3 m_3}^{(s_3)} = \sqrt{\frac{(2l_1+1)(2l_2+1)(2l_3+1)}{4\pi}} \begin{pmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \end{pmatrix} \begin{pmatrix} l_1 & l_2 & l_3 \\ -s_1 & -s_2 & -s_3 \end{pmatrix},$$
$$\tag{4.14c}$$

where $\left(\begin{smallmatrix} : & : & : \end{smallmatrix}\right)$ is the $3j$-symbol [97]. In this way, we have fully expressed the density operator in terms of the operators in the orbital space $c_{mf}^{(\dagger)}$.

4.3 Density-density interaction

The most straightforward way to construct an interaction term is to add a density-density interaction with a potential function[5]

$$H_{\text{int}} = \int \mathrm{d}^2\hat{\mathbf{n}}_1\, \mathrm{d}^2\hat{\mathbf{n}}_2\, U(|\hat{\mathbf{n}}_1 - \hat{\mathbf{n}}_2|) n_M(\hat{\mathbf{n}}_1) n_M(\hat{\mathbf{n}}_2). \tag{4.15}$$

The interacting potentials can be expanded in terms of the Legendre polynomials

$$U(|\mathbf{r}_{12}|) = \sum_l \tilde{U}_l P_l(\cos\theta_{12}) = \sum_{lm} \frac{4\pi \tilde{U}_l}{2l+1} \bar{Y}_{lm}(\hat{\mathbf{n}}_1) Y_{lm}(\hat{\mathbf{n}}_2), \tag{4.16}$$

where $\mathbf{r}_{12} = \hat{\mathbf{n}}_1 - \hat{\mathbf{n}}_2$ and $|\mathbf{r}_{12}| = 2\sin\theta_{12}/2$. Conversely

$$\tilde{U}_l = \int \sin\theta_{12} \mathrm{d}\theta_{12} \frac{2l+1}{2} U(|\mathbf{r}_{12}|) P_l(\cos\theta_{12}). \tag{4.17}$$

Specifically, for local and super-local interactions[6]

$$U(|\mathbf{r}_{12}|) = g_0 \delta(\mathbf{r}_{12}), \qquad\qquad \tilde{U}_l = g_0(2l+1)$$

$$U(|\mathbf{r}_{12}|) = g_1 \nabla^2 \delta(\mathbf{r}_{12}), \qquad\qquad \tilde{U}_l = -g_1 l(l+1)(2l+1). \tag{4.18}$$

---

[5]This is, however, not the simplest construction and we will present the simpler construction in terms of pseudopotentials in the next section.

[6]Usually, to keep the interaction Hamiltonian extensive, the factors scale with system size as $g_0 \propto R^{-2}$ and $g_1 \propto R^{-4}$

By expanding the density operators into the orbital space and completing the integrals,

$$H_{\text{int}} = \sum_{lm} \frac{4\pi \tilde{U}_l}{2l+1} n^\dagger_{M,lm} n_{M,lm}. \tag{4.19}$$

With these ingredients, we can now consider how to construct models. This comes down to matching the symmetry and phase diagram. *E.g.*, for the Ising model [1], the $\mathbb{Z}_2$ global symmetry is realised as the exchange of the two flavours $\psi_\uparrow(\mathbf{r}) \leftrightarrow \psi_\downarrow(\mathbf{r})$. We need a phase diagram with a paramagnetic (PM) phase where the $\mathbb{Z}_2$ symmetry is conserved and a ferromagnetic (FM) phase where the $\mathbb{Z}_2$ symmetry is spontaneously broken. The PM phase is favoured by a polarising term that resembles a transverse field

$$-h \int \mathrm{d}^2 \hat{\mathbf{n}} \, n_x(\hat{\mathbf{n}})$$

and the FM phase where either of the two flavours is fully filled is favoured by a repulsion between the two flavours

$$\int \mathrm{d}^2 \hat{\mathbf{n}}_1 \, \mathrm{d}^2 \hat{\mathbf{n}}_2 \, U(|\mathbf{r}_{12}|) n_\uparrow(\hat{\mathbf{n}}_1) n_\downarrow(\hat{\mathbf{n}}_2),$$

where the density operators are defined as

$$n_x(\hat{\mathbf{n}}) = \psi^\dagger_\downarrow(\hat{\mathbf{n}})\psi_\uparrow(\hat{\mathbf{n}}) + \psi^\dagger_\uparrow(\hat{\mathbf{n}})\psi_\downarrow(\hat{\mathbf{n}}), \quad n_\uparrow(\hat{\mathbf{n}}) = \psi^\dagger_\uparrow(\hat{\mathbf{n}})\psi_\uparrow(\hat{\mathbf{n}}), \quad n_\downarrow(\hat{\mathbf{n}}) = \psi^\dagger_\downarrow(\hat{\mathbf{n}})\psi_\downarrow(\hat{\mathbf{n}}),$$

and the potentials can be most conveniently taken as a combination of local and super-local interactions. Altogether, the model Hamiltonian reads

$$H_{\text{int}} = \int \mathrm{d}^2 \hat{\mathbf{n}}_1 \, \mathrm{d}^2 \hat{\mathbf{n}}_2 \, U(|\mathbf{r}_{12}|) n_\uparrow(\hat{\mathbf{n}}_1) n_\downarrow(\hat{\mathbf{n}}_2) - h \int \mathrm{d}^2 \hat{\mathbf{n}} \, n_x(\hat{\mathbf{n}}). \tag{4.20}$$

By tuning the ratio between h and $U(|\mathbf{r}_{12}|)$, a phase transition described by the Ising CFT is realised.

4.4   Interaction in terms of pseudopotentials

Another way that is much more convenient to construct the four-fermion interaction terms is through Haldane pseudopotential [48, 98]. To explain the idea, we take the 3d Ising model as an example. We first classify all the fermion bilinears $\lambda_{m_1 f_1 m_2 f_2} c_{m_1 f_1} c_{m_2 f_2}$. To simplify the discussion, we can take a specific flavour index $\lambda_{m_1 m_2} c_{m_1 \uparrow} c_{m_2 \downarrow}$. The fermion bilinears can

be classified into irreducible representations (irreps) of SO(3) rotation symmetry. Since c_{mf} carries the spin-s representation, the spin of its bilinear ranges from 0 to $2s$ and takes integer values. The spin-$(2s-l)$ combination reads

$$\Delta_{lm} = \sum_{m_1} \langle sm_1, s(m-m_1)|(2s-l)m \rangle c_{m_1\uparrow} c_{m-m_1\downarrow}, \tag{4.21}$$

where $m = -(2s-l), \ldots, (2s-l)$, and the Clebshbar-Gordan coefficients [97] is related to the $3j$-symbol by

$$\langle l_1 m_1, l_2 m_2|lm \rangle = (-1)^{-l_1+l_2-m}\sqrt{2l+1} \begin{pmatrix} l_1 & l_2 & l \\ m_1 & m_2 & -m \end{pmatrix} \tag{4.22}$$

A four-fermion interaction term is formed by contracting these paring operators with its conjugate

$$H = \sum_l U_l H_l, \quad H_l = \sum_m \Delta_{lm}^\dagger \Delta_{lm}. \tag{4.23}$$

Putting these together, the interaction Hamiltonian can be expressed as

$$H = \sum_{l,m_1m_2m_3m_4} U_l C_{m_1m_2m_3m_4}^l c_{m_1\uparrow}^\dagger c_{m_2\downarrow}^\dagger c_{m_3\downarrow} c_{m_4\uparrow} - h\sum_m (c_{m\uparrow}^\dagger c_{m\downarrow} + \text{h.c.}), \tag{4.24}$$

where the matrix elements are

$$C_{m_1m_2m_3m_4}^l = \delta_{m_1+m_2,m_3+m_4} \langle sm_1, sm_2|(2s-l)(m_1+m_2) \rangle \langle sm_3, sm_4|(2s-l)(m_3+m_4) \rangle. \tag{4.25}$$

The coupling strengths U_l of the spin-$(2s-l)$ channel are called the Haldane pseudopotentials.

We also need to consider the constraint that the two fermions must be antisymmetrised : for even l, the orbital index is symmetrised, so the spin index must be antisymmetrised, so the two fermions form a spin-singlet which is invariant under the SU(2) transformation ; for odd l, the orbital index is antisymmetrised, so the spin index is symmetrised, breaking the flavour symmetry from SU(2) to $\mathbb{Z}_2$. Hence, an odd-l pseudopotential must be added (This fact escapes the construction by density-density interaction).

The fermion bilinears with other flavour configurations $\lambda_{m_1m_2,\pm}(c_{m_1\uparrow}c_{m_2\uparrow} \pm c_{m_1\downarrow}c_{m_2\downarrow})$ can be analysed in a similar way. After that, we have enumerated all possible four-fermion interaction terms.

Each pseudopotential corresponds to a profile of interaction potential functions. The conversion between the pseudopotentials U_l and the Legendre expansion coefficients of the potential function $\tilde{U}_l$

$$U(|\mathbf{r}_{12}|) = \sum_l \tilde{U}_l P_l(\cos\theta_{12}) \tag{4.26}$$

is

$$U_l = \sum_k \tilde{U}_k (-1)^l (2s+1)^2 \begin{Bmatrix} 2s-l & s & s \\ k & s & s \end{Bmatrix} \begin{pmatrix} s & k & s \\ -s & 0 & s \end{pmatrix}^2, \tag{4.27}$$

where $\begin{Bmatrix} \vdots & \vdots \end{Bmatrix}$ is the $6j$-symbol. Specifically, a local interaction $\delta(\mathbf{r}_{12})$ contains only pseudopotential U_0 ; a superlocal interaction of form $(\nabla^2)^l \delta(\mathbf{r}_{12})$ contains $U_0, U_1, \dots, U_l$. Here we explicitly give the expressions for the lowest pseudopotentials

$$U(|\mathbf{r}_{12}|) = \delta(\mathbf{r}_{12}), \qquad U_0 = \frac{(2s+1)^2}{4s+1}$$

$$U(|\mathbf{r}_{12}|) = \nabla^2 \delta(\mathbf{r}_{12}), \qquad U_0 = -\frac{s(2s+1)^2}{4s+1}, \qquad U_1 = \frac{s(2s+1)^2}{4s-1}. \tag{4.28}$$

More details are given in Ref. [15]. We also note that the construction for interaction Hamiltonian for LLL with pseudopotentials is not restricted to sphere. *E.g.*, on a torus, the interaction can also be parametrise by pseudopotentials [99–101]. More details are presented in Appendix B.

For systems with more complicated continuous flavour symmetries, classification in terms of representation of flavour symmetry must also be considered, and the indices must be overall antisymmetrised. We explain that through the example of a $2N$-flavour system with $\mathrm{Sp}(N)$ symmetry [16]. The maximal flavour symmetry is $\mathrm{SU}(2N)$, so interactions must be added to break the symmetry from $\mathrm{SU}(2N)$ to $\mathrm{Sp}(N)$. The fermion operators c_{ma} live in the $\mathrm{Sp}(N)$ fundamental representation, where a is the $\mathrm{Sp}(N)$ index. We shall show that all the allowed terms are

$$H = \sum_{\substack{l\in\mathbb{Z} \\ m_1 m_2 m_3 m_4}} U_l C^l_{m_1 m_2 m_3 m_4} c^\dagger_{m_1 a} c^\dagger_{m_2 b} c_{m_3 b} c_{m_4 a}$$

$$-\frac{1}{2} \sum_{\substack{l\in 2\mathbb{Z} \\ m_1 m_2 m_3 m_4}} V_l C^l_{m_1 m_2 m_3 m_4} \Omega_{aa'} \Omega_{bb'} c^\dagger_{m_1 a} c^\dagger_{m_2 a'} c_{m_3 b'} c_{m_4 b}. \tag{4.29}$$

– 23 –

where $\Omega = \begin{pmatrix} 0 & \mathbb{I}_N \\ -\mathbb{I}_N & 0 \end{pmatrix}$.

To find out all the four-fermion interactions allowed by the rotation symmetry SO(3) and flavour symmetry Sp(N), we classify all the fermion bilinears $c_{m_1 a} c_{m_2 b}$ into irreps of SO(3) $\times$ Sp(N). For each irrep, by contracting the bilinear with its Hermitian conjugate, we obtain an allowed four-fermion interaction term. Each fermion carries SO(3) spin-s and Sp(N) fundamental. For the rotation symmetry SO(3), the bilinears can carry spin-$(2s - l)(l = 0, ..., 2s)$ represetation ; for even l, the orbital indices are symmetrised ; for odd l, the orbital indices are antisymmetrised. For the flavour symmetry Sp(N), the bilinears can carry singlet S, traceless antisymmetric rank-2 tensor A and symmetric rank-2 tensor T representation ; for S and A, the flavour indices are antisymmetrised ; for T, the flavour indices are symmetrised. As the two fermions altogether should be antisymmetrised, the allowed combinations are

1. Sp(N) singlet and SO(3) spin-$(2s - l)$ with even l, the bilinears are

$$\Delta_{lm} = \sum_{m_1 m_2} \langle sm_1, sm_2 | (2s - l)m \rangle \Omega_{cc'} c_{m_1 c} c_{m_2 c'} \delta_{m, m_1 + m_2}. \tag{4.30}$$

The corresponding interaction term $H_{S,l} = \sum_m \Delta_{lm}^\dagger \Delta_{lm}$ is the even-l pseudopotential for the V-term.

2. Sp(N) antisymmetric and SO(3) spin-$(2s - l)$ with even l, the bilinears are

$$\Delta_{lm,[ab]} = \sum_{m_1 m_2} \langle sm_1, sm_2 | (2s - l)m \rangle (c_{m_1 a} c_{m_2 b} - c_{m_1 b} c_{m_2 a} - \tfrac{1}{N} \Omega_{ab} \Omega_{cc'} c_{m_1 c'} c_{m_2 c}) \delta_{m, m_1 + m_2}. \tag{4.31}$$

The corresponding interaction term $H_{A,l} = \sum_m \Delta_{lm,[ab]}^\dagger \Delta_{lm,[ab]}$ is the even-l pseudopotential for the U-term.

3. Sp(N) symmetric and SO(3) spin-$(2s - l)$ with odd l, the bilinears are

$$\Delta_{lm,(ab)} = \sum_{m_1 m_2} \langle sm_1, sm_2 | (2s - l)m \rangle (c_{m_1 a} c_{m_2 b} + c_{m_1 b} c_{m_2 a}) \delta_{m, m_1 + m_2}. \tag{4.32}$$

The corresponding interaction term $H_{T,l} = \sum_m \Delta_{lm,(ab)}^\dagger \Delta_{lm,(ab)}$ is the odd-l pseudopotential for the U-term.

In summary, all allowed interactions are the U_l terms with both even and odd l, and the V_l terms with only even l.

4.5 Operator spectrum and search for conformal point

Having introduced the construction of an interacting model on the fuzzy sphere, we now turn to the verification of the conformal symmetry and the extraction of the CFT data. The most straightforward approach is to extract the scaling dimensions from the energy spectrum through the state-operator correspondence. Specifically, there is a one-to-one correspondence between the eigenstates of the Hamiltonian and the CFT operators. The state and its corresponding operator has the same SO(3) spin and representation under flavour symmetry, and the excitation energy of a state $|\Phi\rangle$ is proportional to the scaling dimension of the corresponding operator Δ_Φ

$$E_\Phi - E_0 = \frac{v}{R}\Delta_\Phi, \tag{4.33}$$

where E_0 is the ground state energy, R is the radius of the sphere (here we take $R = \sqrt{N_m}$), and v is a model-dependent speed of light. The constant v/R can be determined through a calibration process, *i.e.*, comparing the spectrum to some known properties of a CFT spectrum. The criteria to determine the conformal symmetry include

1. The existence of a conserved stress tensor $T^{\mu\nu}$. The stress tensor is the symmetry current of the translation transformation. It is known to be a singlet under the flavour symmetry, have spin-2 under SO(3) rotation and scaling dimension exactly $\Delta_{T^{\mu\nu}} = 3$.

2. The existence of a conserved flavour symmetry current J^μ if there is a continuous flavour symmetry. The symmetry current typically lives in the antisymmetric rank-2 tensor representation of the flavour symmetry. *E.g.*, if the flavour symmetry is U(1), then the symmetry current has charge-0 ; if the flavour symmetry is O(3), then the symmetry current has spin-1 and is odd under the improper $\mathbb{Z}_2$ transformation ; if the flavour symmetry is O(n) ($n \geq 4$) or SU(n) ($n \geq 3$), then the symmetry current lives in the antisymmetric rank-2 tensor representation.

3. The organisation of the operator spectrum into conformal multiplets. All the levels in the spectrum of a CFT can be organised into the conformal primaries and their descendants. The descendants live in the same representation under the flavour symmetry as the primary, and the difference between the scaling dimensions of a primary and its descendant is an integer.

Specifically, for a scalar primary Φ, its descendants have the form[7]

$$\Box^n \partial^{\mu_1} \partial^{\mu_2} \dots \partial^{\mu_l} \Phi - (\text{trace}) \qquad (n, l = 0, 1, 2, \dots)$$

with SO(3) spin-l and scaling dimension

$$\Delta = \Delta_\Phi + 2n + l,$$

where $\Box = \partial_\mu \partial^\nu$. For a spinning primary $\Phi^{\mu_1 \cdots \mu_s}$, its descendants has the two forms :

$$\Box^n \partial^{\nu_1} \dots \partial^{\nu_m} \partial_{\rho_1} \dots \partial_{\rho_k} \Phi^{\rho_1 \cdots \rho_k \mu_1 \cdots \mu_{s-k}} \qquad (k = 0, \dots, s, \ n, m = 0, 1, \dots)$$

with scaling dimension and SO(3) spin

$$\Delta = \Delta_\Phi + k + m + 2n, \qquad l = s - k + m,$$

and

$$\Box^n \partial^{\nu_1} \dots \partial^{\nu_m} \partial_{\rho_1} \dots \partial_{\rho_k} \epsilon^\sigma{}_{\tilde{\mu}\tilde{\nu}} \partial^{\tilde{\nu}} \Phi^{\rho_1 \cdots \rho_k \tilde{\mu} \mu_1 \cdots \mu_{s-k-1}} \qquad (k = 0, \dots, s-1, \ n, m = 0, 1, \dots)$$

with

$$\Delta = \Delta_\Phi + k + m + 2n + 1, \qquad l = s - k + m.$$

For the second form, the fully antisymmetric tensor ϵ alters the parity.

The most convenient way of determining the coefficient v/R is by utilising criteria 1 or 2 :

$$\frac{v}{R} = \frac{E_{T^{\mu\nu}} - E_0}{3} \quad \text{or} \quad \frac{E_{J^\mu} - E_0}{2}. \tag{4.34}$$

Alternatively, one can define a cost function that depends on the tuning parameter and the speed of light and compares the scaling dimensions obtained from the fuzzy sphere and the prediction by conformal symmetry. *E.g.*, for the Ising CFT, the tuning parameters are the pseudopotentials $\{U_i\}$ and the transverse field h. The criteria for conformal symmetry we use include the stress tensor $T^{\mu\nu}$ and the descendants $\partial^\mu \sigma$, $\partial^\mu \partial^\nu \sigma$, $\Box \sigma$, $\partial^\mu \epsilon$. The cost function is the root-mean-square of the deviations of these criteria from the expectation of the conformal symmetry

---

[7]Hereafter, we will presume the subtraction of trace and omit the terms.

$$Q^2(\{U_i\}, h, v; N_m) = \frac{1}{N_s} \Big[(\Delta_{T^{\mu\nu}}^{(\mathrm{FS})} - 3)^2 + (\Delta_{\partial^\mu\sigma}^{(\mathrm{FS})} - \Delta_\sigma^{(\mathrm{FS})} - 1)^2$$

$$+ (\Delta_{\partial^\mu\partial^\nu\sigma}^{(\mathrm{FS})} - \Delta_\sigma^{(\mathrm{FS})} - 1)^2 + (\Delta_{\Box\sigma}^{(\mathrm{FS})} - \Delta_\sigma^{(\mathrm{FS})} - 1)^2 + (\Delta_{\partial^\mu\epsilon}^{(\mathrm{FS})} - \Delta_\epsilon^{(\mathrm{FS})} - 1)^2 \Big] \quad (4.35)$$

where $N_s = 5$ is the number of criteria, the scaling dimension of an operator Φ on the fuzzy sphere is determined as

$$\Delta_\Phi^{(\mathrm{FS})}(\{U_i\}, h, v; N_m) = \frac{E_\Phi - E_0}{v/R}. \quad (4.36)$$

The optimal conformal point and calibrator are determined by minimising this cost function for each system size N_m. Note that this optimal point depends on the system size. In order to do finite-size scaling, if the CFT describes a phase transition, one could fix all but one parameters at the optimal point in the largest accessible system size and tune the last parameter to determine the critical point through a finite-size scaling.

4.6 Local observables

We have introduced how to determine the scaling dimensions from the energy spectrum. Beyond that, evaluating other CFT quantities requires realising local CFT operators on the fuzzy sphere. Any gapless local observables $\mathcal{O}(\hat{\mathbf{n}})$ on the fuzzy sphere can be written as the linear combination of CFT operators that live in the same representation of flavour symmetry and parity[8] [2]

$$\mathcal{O}(\hat{\mathbf{n}}, \tau) = \sum_\alpha \lambda_\alpha \Phi_\alpha^{(\mathrm{cyc.})}(\hat{\mathbf{n}}, \tau). \quad (4.37)$$

Here special care should be taken for the CFT operator $\Phi_\alpha^{(\mathrm{cyc.})}(\hat{\mathbf{n}}, \tau)$ on the cylinder. A conformal transformation produces a scale factor $\Lambda(\mathbf{r})^{\Delta_\Phi}$ to a primary operator Φ. The scale factor is $\Lambda(\mathbf{r}) = r/R$ for the Weyl transformation from the flat spacetime to the cylinder. Hence[9],

$$\Phi_\alpha^{(\mathrm{cyc.})}(\hat{\mathbf{n}}, \tau) = \left(\frac{e^{\tau/R}}{R} \right)^{\Delta_{\Phi_\alpha}} \Phi_\alpha^{(\mathrm{flat})}(\mathbf{r}). \quad (4.38)$$

---

[8]The realisation of CFT operators in the microscopic model has been investigated in more detail in 2d in Ref. [102].

[9]We need to clarify some of the notations : $\mathcal{O}$ represents an operator in the microscopic model, and Φ represents a CFT operator ; the arguments $\Phi(\hat{\mathbf{n}})$ or $\Phi(\hat{\mathbf{n}}, \tau)$ by default mean the operator is defined on a cylinder, and $\Phi(\mathbf{r})$ by default on flat spacetime.

For descendants, certain other factors may be produced, but the conversion factors still scale with the radius of the sphere as $R^{-\Delta}$ where Δ is the scaling dimension of the descendants. For simplicity, hereafter we focus on the equal-time correlators with $\tau = 0$, for which $\Phi_\alpha^{(\text{cyc.})}(\hat{n}) = R^{-\Delta_{\Phi_\alpha}}\Phi_\alpha^{(\text{flat})}(\mathbf{r})$. The operator with larger system size decays faster when increasing system size.

The simplest local observable is the density operator defined in Eq. (4.10), and its decomposition into angular modes is given in Eqs. (4.12) and (4.13). From the CFT perspective, the density operators are the superpositions of scaling operators with corresponding quantum numbers, *i.e.*, with the same representation under flavour symmetry and parity.

Take the Ising model as an example. Consider the density operators n^x and n^z with matrix insertion $M = \sigma^x, \sigma^z$. In the leading order, they can be used as UV realisations of CFT operators σ and ϵ.

$$n^x(\hat{n}) = \lambda_0 + \lambda_\epsilon \epsilon(\hat{n}) + \lambda_{\partial^\mu \epsilon}\partial^\mu \epsilon(\hat{n}) + \lambda_{T^{\mu\nu}}T^{\mu\nu}(\hat{n}) + ... \qquad \epsilon_{\text{FS}} = \frac{n^x - \lambda_0}{\lambda_\epsilon} + ...$$

$$n^z(\hat{n}) = \lambda_\sigma \sigma(\hat{n}) + \lambda_{\partial^\mu \sigma}\partial^\mu \epsilon(\hat{n}) + \lambda_{\partial^\mu \partial^\nu \sigma}\partial^\mu \partial^\nu \sigma(\hat{n}) + ... \qquad \sigma_{\text{FS}} = \frac{n^z}{\lambda_\sigma} + ... \qquad (4.39)$$

where the coefficients $\lambda_0, \lambda_\epsilon, \lambda_\sigma, ...$ are model-dependent and need to be determined, and all the operators on the right-hand side are defined on the cylinder.

We first consider the insertion of a single operator $\langle \Phi_1 | \Phi_2(\hat{n}) | \Phi_3 \rangle$. It helps us produce the OPE coefficients. For the simplest example of three scalars,

$$f_{\Phi_1 \Phi_2 \Phi_3} = \lim_{r_\infty \to \infty} r_\infty^{-2\Delta_{\Phi_1}}\langle \Phi_1(x_\infty)\Phi_2(x)\Phi_3(0)\rangle_{\text{flat}} = \langle \Phi_1 | \Phi_2^{(\text{flat})}(x) | \Phi_3 \rangle \qquad (4.40)$$

where x_∞ is a point on the sphere with radius r_∞, x is a point on the unit sphere, the states are obtained from acting the operator at the origin point on the vacuum state

$$|\Phi_3\rangle = \Phi_3(0)|0\rangle \qquad (4.41)$$

and its Hermitian conjugate is defined as

$$\Phi_1^\dagger(\infty) = (\Phi_1(0))^\dagger = \lim_{r_\infty \to \infty} r_\infty^{2\Delta_{\Phi_1}}\Phi_1(x_\infty), \qquad \langle \Phi_1 | = \langle 0 | \Phi_1^\dagger(\infty). \qquad (4.42)$$

After the Weyl transformation from the flat spacetime to the cylinder, we obtain the expression on the fuzzy sphere

$$f_{\Phi_1 \Phi_2 \Phi_3} = R^{\Delta_{\Phi_2}}\langle \Phi_1 | \Phi_2^{(\text{cyl.})}(\hat{n}) | \Phi_3 \rangle. \qquad (4.43)$$

The UV realisation of Φ_2 contains many other operators with different spins. By integrating the correlation function against different spherical harmonics, *i.e.*, take the angular modes of the operator inserted

$$\int d^2\hat{\mathbf{n}}\, \bar{Y}_{lm}(\hat{\mathbf{n}}) \langle \Phi_1 | \Phi_2(\hat{\mathbf{n}}) | \Phi_3 \rangle = \langle \Phi_1 | \Phi_{2,lm} | \Phi_3 \rangle, \tag{4.44}$$

we can filter out the subleading contributions with different spins. For the spinning operators, this also tells us about different OPE structures. By taking $\Phi_3 = \mathbb{I}$, we can recover the 2-pt functions

$$\langle \Phi_2 | \Phi_{2,00} | 0 \rangle = R^{-\Phi_2}$$
$$\Phi_2(\hat{\mathbf{n}})|0\rangle = R^{-\Phi_2} \Big[|\Phi_2\rangle + \lambda'_\mu(\hat{\mathbf{n}})|\partial^\mu \Phi_2\rangle + \lambda''(\hat{\mathbf{n}})|\Box \Phi_2\rangle + \lambda''_{\mu\nu}(\hat{\mathbf{n}})|\partial^\mu \partial^\nu \Phi_2\rangle \Big]. \tag{4.45}$$

It is worth noting that acting a primary $\Phi_2(\hat{\mathbf{n}})$ on the vacuum also produces various descendants in the multiplet.

In the example of Ising CFT, we first use the insertion of a single operator to determine the coefficients $\lambda_0, \lambda_\epsilon, \lambda_\sigma$

$$\lambda_0 = \frac{R^{-2}}{\sqrt{4\pi}}\langle 0|n_{00}^x|0\rangle, \quad \lambda_\epsilon = \frac{R^{\Delta_\epsilon-2}}{\sqrt{4\pi}}\langle \epsilon|n_{00}^x|0\rangle, \quad \lambda_\sigma = \frac{R^{\Delta_\sigma-2}}{\sqrt{4\pi}}\langle \sigma|n_{00}^z|0\rangle. \tag{4.46}$$

Take the OPE coefficient $f_{\sigma\sigma\epsilon}$ as an example. It can be expressed either as an inner product of σ or ϵ

$$f_{\sigma\sigma\epsilon} = R^{\Delta_\sigma}\langle \epsilon|\sigma(\hat{\mathbf{n}})|\sigma\rangle = \frac{\langle \epsilon|n_{00}^z|\sigma\rangle}{\langle 0|n_{00}^z|\sigma\rangle} + \mathcal{O}(R^{-2})$$
$$= R^{\Delta_\epsilon}\langle \sigma|\epsilon(\hat{\mathbf{n}})|\sigma\rangle = \frac{\langle \sigma|n_{00}^x|\sigma\rangle - \langle 0|n_{00}^x|0\rangle}{\langle \epsilon|n_{00}^x|0\rangle} + \mathcal{O}(R^{-(3-\Delta_\epsilon)}). \tag{4.47}$$

For the first line, the subleading contribution comes from the contribution of the descendant $\Box\sigma$ to n_{00}^z. As $\sigma(\hat{\mathbf{n}})$ scales as $R^{-\Delta_\sigma}$ and $\Box\sigma(\hat{\mathbf{n}})$ as $R^{-\Delta_\sigma-2}$,

$$\langle \epsilon|n_{00}^z|\sigma\rangle = f_{\sigma\sigma\epsilon}\lambda_\sigma R^{-\Delta_\sigma}(1 + c_1 R^{-2} + ...)$$
$$\langle \epsilon|n_{00}^z|\sigma\rangle = \lambda_\sigma R^{-\Delta_\sigma}(1 + c'_1 R^{-2} + ...)$$
$$\frac{\langle \epsilon|n_{00}^z|\sigma\rangle}{\langle 0|n_{00}^z|\sigma\rangle} = f_{\sigma\sigma\epsilon} + \mathcal{O}(R^{-2}). \tag{4.48}$$

Here c_1 and c_1' are constant factors that represent the contribution of $\Box\sigma$ and do not scale with system size. Hence, the subleading contribution scales as R^{-2}. For the second line, the subleading contribution comes from the stress tensor $T^{\mu\nu}$. Similarly, the power of the scaling is the difference of the scaling dimension $R^{-(\Delta_{T^{\mu\nu}}-\Delta_\epsilon)} = R^{-(3-\Delta_\epsilon)}$.

We then proceed to the insertion of two operators. This can help us determine up to a 4-pt function [3]. Through conformal transformation, any 4-pt function can be expressed in the form of

$$\langle\Phi_1|\Phi_2^{(\mathrm{cyl.})}(\hat{\mathbf{n}},\tau)\Phi_3^{(\mathrm{cyl.})}(\hat{\mathbf{z}})|\Phi_4\rangle = \frac{e^{\Delta_{\Phi_2}\tau/R}}{R^{\Delta_{\Phi_2}+\Delta_{\Phi_3}}}\langle\Phi_1^\dagger(\infty)\Phi_2(\mathbf{r})\Phi_3(\hat{\mathbf{z}})\Phi_4(0)\rangle, \tag{4.49}$$

where the time-displaced operator can be defined as

$$\Phi_2(\hat{\mathbf{n}},\tau) = e^{-H\tau}\Phi_2(\hat{\mathbf{n}})e^{H\tau}. \tag{4.50}$$

As a sanity check, By taking $\Phi_1 = \Phi_4 = \mathbb{I}$, $\Phi_2 = \Phi_3$ and $\tau = 0$, the 2-pt function on the unit sphere is recovered

$$\langle 0|\Phi_2^{(\mathrm{cyl.})}(\hat{\mathbf{n}})\Phi_2^{(\mathrm{cyl.})}(\hat{\mathbf{z}})|0\rangle = R^{-2\Delta_{\Phi_2}}\langle\Phi_2(\hat{\mathbf{n}})\Phi_2(\hat{\mathbf{z}})\rangle$$

$$= \frac{1}{R^{2\Delta_{\Phi_2}}|\hat{\mathbf{n}}-\hat{\mathbf{z}}|^{2\Delta_{\Phi_2}}} = \frac{1}{R^{2\Delta_{\Phi_2}}(1-\cos\theta)^{\Delta_{\Phi_2}}}. \tag{4.51}$$

4.7 Conformal generators

So far, in the conformal group, we know that the rotation and the dilatation are manifest on the fuzzy sphere. The rest, *viz.* translation and SCT, are emergent. In this section, we consider how to express the generators of these emergent symmetries in terms of the microscopic operators [14, 15].

A general Noether current and corresponding generator of the infinitesimal spacetime transformation $x^\mu \mapsto x^\mu + \epsilon^\mu(x)$ can be expressed in terms of the stress tensor

$$j_\epsilon^\mu(x) = \epsilon^\nu(x)T^\mu{}_\nu(x), \quad Q_\epsilon = \int_\Sigma \mathrm{d}^{d-1}x\,\sqrt{g}j_\epsilon^0(x), \tag{4.52}$$

where for the second equation, the integral is evaluated on a closed surface Σ. Specifically, for the generators P^μ, K^μ of translation and SCT in the embedded sphere

$$P^\mu = \int \mathrm{d}^2\hat{\mathbf{n}}\,(n^\mu T^0{}_0 + iT^{0\mu}),$$

$$K^{\mu} = \int d^2\hat{\mathbf{n}} \, (n^{\mu} T^0{}_0 - iT^{0\mu}).\tag{4.53}$$

Hence, the conformal generator $\Lambda^{\mu} = P^{\mu} + K^{\mu}$ is the $l = 1$ component of the Hamiltonian density $\mathcal{H} = T^0{}_0$[10]

$$\Lambda_m = P_m + K_m = \sqrt{\frac{16\pi}{3}} \int d^2\hat{\mathbf{n}} \, \bar{Y}_{1m}(\hat{\mathbf{n}})\mathcal{H}(\hat{\mathbf{n}}).\tag{4.54}$$

By acting it on the states, the number of derivatives is increased or decreased by 1, *e.g.*, for a primary Φ

$$\Lambda^{\mu}|\Phi\rangle = \text{const.} \times |\partial^{\mu}\Phi\rangle$$
$$\Lambda^{\mu}|\partial_{\mu}\Phi\rangle = \text{const.} \times |\Phi\rangle + \text{const.} \times |\partial^{\mu}\partial^{\nu}\Phi\rangle + \text{const.} \times |\Box\Phi\rangle.\tag{4.55}$$

The derivation of the expression and the constant factors are calculated and given in Ref. [14, 15].

We then need to find the expression for the Hamiltonian density. For example, for the Ising model, it is the local density operator and density-density interactions with some full derivatives

$$\mathcal{H}(\hat{\mathbf{n}}) = n_z \left(g_0 + g_1 \nabla^2\right) n_z - hn_x + g_{D,1}\nabla^2 n_x + g_{D,2}\nabla^2 n_z^2 + \dots,\tag{4.56}$$

where $g_{D,i}$ are undetermined constants that does not affect the Hamiltonian $H = \int d^2\hat{\mathbf{n}}\,\mathcal{H}$. We have only listed a few examples of the allowed full derivatives.

To determine those constants, we consider another strategy by considering all the possible two-fermion and four-fermion operators that are singlet under flavour symmetry and spin-1 under SO(3). We consider the example of Ising CFT. The two-fermion terms include the density operators

$$n^x_{1m} \quad \text{and} \quad n^0_{1m}.$$

Similar to what we have done for Hamiltonian, the four-fermion operators can be obtained by combining the fermion bilinears Δ_{lm} defined in Eq. (4.21)

$$\sum_{l_1 l_2 m_1 m_2} \tilde{U}_{l_1 l_2} \Delta^{\dagger}_{l_1 m_1} \Delta_{l_2 m_2} \langle (2s - l_1)(-m_1), (2s - l_2)m_2 | 1m \rangle \tag{4.57}$$

---

[10]Here the indices μ and m are two equivalent way to express the components

For $l_1 \in 2\mathbb{Z}$, the spin index in the pairing operator is antisymmetrised ; For $l_1 \in 2\mathbb{Z}+1$, the spin index in the pairing operator is symmetrised. Therefore, $l_1 - l_2 \in 2\mathbb{Z}$ for non-zero results. And since $|l_1 - l_2| \le 1$, we conclude $l_1 = l_2$. so

$$\Lambda_m = \sum_{lm_1m_2} \tilde{U}_l \Delta^\dagger_{lm_1} \Delta_{lm_2} \begin{pmatrix} 2s-l & 2s-l & 1 \\ -m_1 & m_2 & m \end{pmatrix} + \tilde{h} n^x_{1m} + \tilde{\mu} n^0_{1m} \tag{4.58}$$

Here, $\tilde{U}_l, \tilde{h}, \tilde{\mu}$ are tuning parameters.

After obtaining $\Lambda^\mu = P^\mu + K^\mu$, the separate P^μ and K^μ can be obtained by considering the commutator with the dilatation generator D, which is proportional to the Hamiltonian. As $[D, P^\mu] = P^\mu$ and $[D, K^\mu] = -K^\mu$.

$$P^\mu = \tfrac{1}{2}\Lambda^\mu + \tfrac{1}{2}[D, \Lambda^\mu]$$
$$K^\mu = \tfrac{1}{2}\Lambda^\mu - \tfrac{1}{2}[D, \Lambda^\mu]. \tag{4.59}$$

5 Numerical methods

In this section, we briefly review the numerical methods supported in FuzzifiED. The numerical methods that have been applied to the fuzzy sphere include exact diagonalisation (ED), density matrix renormalisation group (DMRG) and determinant quantum Monte Carlo (DQMC). Among these, ED and DMRG have been implemented in FuzzifiED.

5.1 Exact diagonalisation (ED)

Exact diagonalisation (ED) might be the most straightforward method for solving a quantum many-body Hamiltonian. In ED, one constructs a many-body basis and writes down all the non-zero elements of the Hamiltonian as a sparse matrix on this basis. The eigenstates of the Hamiltonian with the lowest energy can be solved without finding the full eigensystem by Arnoldi or Lanczos algorithm.

Briefly speaking, the Arnoldi algorithm [103] is an iterative method. Each iteration constructs an orthonormal basis of the Krylov subspace from an initial vector and finds an approximation to the eigenvector on that basis. This approximate eigenvector is then used as the initial

vector for the next iteration. An example of Krylov subspace is spanned by acting the matrix H repeatedly on the initial vector $|i\rangle$

$$\mathcal{K}_r(H, |i\rangle) = \text{span} \left\{ |i\rangle, H|i\rangle, H^2|i\rangle, \ldots, H^{r-1}|i\rangle \right\}. \tag{5.1}$$

The ED calculation can be optimised in several ways. The storage of the Hamiltonian matrix may be compressed by data structure tailored for sparse matrix such as compressed sparse column (CSC). The Hamiltonian matrix is usually block diagonal due to the symmetry of the Hamiltonian. The Hilbert space is divided into several sectors that carry different representations under the symmetry, and acting the Hamiltonian on a state in a sector results in a state in the same sector. *E.g.*, in the ED calculation for the Ising model on the fuzzy sphere, the symmetries we can use include two U(1) symmetries, *viz.* the conservation of particle number and the angular momentum in the z-direction, and three $\mathbb{Z}_2$ symmetries, *viz.* the Ising $\mathbb{Z}_2$ flavour symmetry, the particle-hole symmetry and the π-rotation along the y-axis[11].

The ED method enjoys several advantages, including (1) the full knowledge of the eigenstate wavefunction and (2) the ability to access relatively high excited states. However, despite these optimisations, the dimension of the Hilbert space scales exponentially with the number of orbitals. This results in exponentially growing space and time complexity. *E.g.*, for the Ising model on the fuzzy sphere, for $N_m = 14$, the dimension of Hilbert space $\dim \mathcal{H} = 1.8 \times 10^5$ and the number of elements in the Hamiltonian is $N_{\text{el}} = 1.1 \times 10^7$; for $N_m = 16$, the numbers have already grown to $\dim \mathcal{H} = 2.2 \times 10^6$ and $N_{\text{el}} = 2.1 \times 10^8$, which translates to a memory demand of 3.1 gigabytes.

In FuzzifiED, we use the Fortran library Arpack [104] to perform the Arnoldi algorithm.

5.2   Density matrix renormalisation group (DMRG)

To overcome the size limit of ED, the density matrix renormalisation group (DMRG) is a powerful method to calculate the ground state of a quasi-one-dimensional system. It was first invented by White [105] as an improvement to the numerical renormalisation group (NRG) used in the Kondo problem. Since its proposal, it has been proven potent in various problems in condensed matter physics, such as the static and dynamic properties of one-dimensional

---

[11]So far, FuzzifiED only supports U(1) and $\mathbb{Z}_p$ symmetries.

models such as the Heisenberg, $t-J$ and Hubbard models [106]. Later, Schollwöck has discovered a new point of view that implements the DMRG in the language of matrix product states (MPS) [107].

Briefly speaking, in this language, DMRG is a variational method that optimises the fidelity between the exact ground state and the variational MPS. During each 'sweep', DMRG solves a local eigenvalue problem for the active tensors to improve the approximation. To find the excited states, one needs to add projection $|0\rangle\langle 0|$ of the ground state $|0\rangle$ to the Hamiltonian by hand.

Although the fuzzy sphere deals with $(2 + 1)$-dimensional quantum systems, the basis of the lowest Landau level provides a natural way to express it as a quasi-1d problem. Therefore, DMRG has been a powerful numerical method for the fuzzy sphere. However, like other $(2+1)$d models, the DMRG on the fuzzy sphere also suffers from the divergence of the required maximal bond dimension with system size. One should thus be careful with the convergence of the results when doing DMRG.

In FuzzifiED, we use the ITensor library [21] in Julia to perform the DMRG calculations.

Part II

Numerical calculation with FuzzifiED

6   Installation and usage

The package FuzzifiED is implemented in the Julia language [22][12]. Some useful links are given in Table 1. To install the package, enter `julia` to enter the Julia REPL (read-eval-print loop), and then run the following command.

```
using Pkg ; Pkg.add("FuzzifiED")
```

To use the package, include at the head of the Julia script.

```
using FuzzifiED
```

To obtain the documentation for an interface, type '?' followed by the keyword in the Julia REPL, *e.g.*, '?Confs'.

Installation for Julia	https://julialang.org/downloads
Homepage	https://www.fuzzified.world
Documentation	https://docs.fuzzified.world
Julia source code	https://github.com/FuzzifiED/FuzzifiED.jl
JLL wrapper	https://github.com/FuzzifiED/FuzzifiED_jll.jl
Fortran source code	https://github.com/FuzzifiED/FuzzifiED_Fortran
Registry of the package	https://juliahub.com/ui/Packages/General/FuzzifiED

Table 1. Some useful links.

7   Exact diagonalisation

In this section, we briefly describe the procedure for exact diagonalisation (ED) calculation and give an instruction for using FuzzifiED for ED.

---

[12]The most intensive core functionalities are written in Fortran and wrapped by Julia interfaces, but the users are not required to have any interaction with or knowledge of the Fortran part of the source code.

Practically, the ED calculation can be divided into four steps, which will be described in detail in the following sections :

1. To construct a many-body basis that respects a given set of quantum numbers (Sections 7.2 and 7.3). Specifically, in FuzzifiED we support quantum numbers of commuting U(1) or discrete $\mathbb{Z}_p$ symmetries.

2. To construct the sparse matrix corresponding to the Hamiltonian on the basis above (Sections 7.4 and 7.5).

3. To find the lowest eigenstates of the sparse matrix and their corresponding eigenenergies (Section 7.6).

4. To make measurements on the eigenstates (Sections 7.7, 7.8 and 7.9). This includes the total angular momentum, density operators, entanglement, *etc.*

To demonstrate the usage of FuzzifiED interfaces, in this section, we use an example that calculates the eigenstates for the Ising model on the fuzzy sphere. Specifically, it

1. calculates the lowest eigenstates in the symmetry sector $L^z = 0$ and $(\mathcal{P}, \mathcal{Z}, \mathcal{R}) = (+, +, +)$,

2. measures their total angular momenta, and

3. calcultes the OPE coefficient $f_{\sigma\sigma\epsilon} = \langle\sigma|n^z_{00}|\epsilon\rangle/\langle\sigma|n^z_{00}|0\rangle$.

The full code is collected in Appendices C.1 and C.2.

7.1 Setup

Before starting the calculation, we need to input the setup for the system, including the number of flavours N_f, orbitals N_m and sites N_o. Here we specify some of our notations.

- A 'flavour' is labelled by f. The number of flavours is N_f.

- An 'orbital' is specified by the magnetic quantum number labelled by m. The number of orbitals is $N_m = 2s + 1$.

- A 'site' is specified by both the flavour and the orbital index $o = (f, m)$. The number of sites is $N_o = N_m N_f$. In practice, we label the sites with an integer from 1 to N_o. We store the sites in an ascending order of first m and then f : $o = (m + s)N_f + f$.

In the example of the Ising model with $s = 5.5$,

```
nm = 12
```

```
nf = 2
no = nm * nf
```

FuzzifiED also provides three environment parameters :

- `FuzzifiED.ElementType` — the type of the matrix elements by default, either `Float64` or `ComplexF64`.

- `FuzzifiED.NumThreads` — an integer to define how many threads OpenMP uses by default.

- `FuzzifiED.SilentStd` — a bool that determines whether logs of the FuzzifiED should be turned off by default.

7.2   Constructing the configurations

The first step for the ED calculation is to construct the basis that respects the symmetries of the Hamiltonian. This is divided into two procedures : (1) to generate the 'configurations' that carry the diagonal quantum numbers, and (2) to generate the 'basis' that also carries the off-diagonal quantum numbers (under discrete transformations). The '*configurations*' are the collection of states that can be written as direct products of occupied $|1\rangle$ or empty $|0\rangle$ on each site and carry certain diagonal quantum numbers (QNDiag).

The QNDiags supported by FuzzifiED are the charges of $U(1)$ or $\mathbb{Z}_p$ symmetry in the form of

$$Q = \sum_o q_o n_o, \qquad\qquad\qquad \text{U(1) symmetry}$$

$$Q = \sum_o q_o n_o \mod p, \qquad\qquad \mathbb{Z}_p \text{ symmetry}, \qquad (7.1)$$

where $n_o = c_o^\dagger c_o$ is the particle number on each site, and q_o is the charge that each site carries. FuzzifiED restricts q_o to be integer-valued. In FuzzifiED, the QNDiags are recorded in the mutable type `QNDiag`.

`QNDiag` — Type

The type contains the fields

- `name :: String` — the name of the diagonal quantum number[13].

---

[13] The name is only needed for conversion into quantum numbers in ITensor.

- charge :: Vector{Int64} — the symmetry charge q_o of each site.
- modul :: Vector{Int64} — the modulus p, set to 1 for U(1) QNDiags.

and can be initialised by the method

```
QNDiag([name :: String, ]charge :: Vector{Int64}[, modul :: Int64])
```

where the arguments in the brackets are facultative.

Several useful QNDiags are built-in[14]

- GetNeQNDiag(no) — the number of electrons ;
- GetLz2QNDiag(nm, nf) — twice the angular momentum $2L_z$;
- GetFlavQNDiag(nm, nf, qf[, id, modul]) — a linear combination of number of electrons in each flavour $Q = \sum_f q_f n_f$, where $\{q_f\}$ is stored in qf in the format of either an array or a dictionary[15].
- GetZnfChargeQNDiag(nm, nf) — a $\mathbb{Z}_{N_f}$-charge $Q = \sum_{f=1}^{N_f}(f-1)n_f \mod N_f$.
- GetPinOrbQNDiag(no, pin_o[, id]) — the number of electrons in the subset of sites pin_o. This QNDiag is useful for defects and boundaries.

The collection of configurations is generated from the QNDiags. It is recorded in the mutable type Confs.

Confs — Type

The mutable type contains the fields

- no :: Int64 — the number of sites.
- ncf :: Int64 — the number of configurations.
- conf :: Vector{Int64} — an array of length ncf containing all the configurations.
- nor :: Int64, lid :: Vector{Int64} and rid :: Vector{Int64} contain the information of Lin table that is used to inversely look up the index from the configuration.

and can be constructed by the method

```
Confs(no :: Int64, secd :: Vector{Int64}, qnd :: Vector{QNDiag})
```

---

[14] For a more detailed description of the interfaces, refer to the documentation at https://docs.fuzzified.world.

[15] *E.g.*, for $Q = n_{f=1} - n_{f=3}$ in a 4-flavour system, both qf = [1, 0, -1, 0] and qf = Dict(1 => 1, 3 => -1) are acceptable.

where `qnd` is the array of QNDiags, and `secd` is the array of charges Q of each QNDiag[16].

Here, each configuration is stored as a binary number with N_o bits. If the o-th site in the configuration is occupied, the $(o-1)$-th bit of the configuration is 1; if the site is empty, then the bit is 0. Besides the storage of the configuration, we also need a reverse look-up process that returns the index from the binary string. This is realised by a Lin table stored in `lid` and `rid`. The details are given in Appendix A.1.

In the example of Ising model Eq. (4.20), there are two QNDiags, *viz.* the particle number and the angular momentum

$$
\begin{aligned}
Q_1 &= N_e, & q_{1,mf} &= 1 \\
Q_2 &= 2L_z, & q_{2,mf} &= 2m.
\end{aligned}
\tag{7.2}
$$

The code for generating the configurations in the $L_z = 0$ sector is

```
qnd = [
    QNDiag(fill(1, no)),
    QNDiag([ 2 * m - nm - 1 for m = 1 : nm for f = 1 : nf ])
]
cfs = Confs(no, [nm, 0], qnd)
```

Alternatively, using the built-in models,

```
qnd = [
    GetNeQNDiag(no),
    GetLz2QNDiag(nm, nf)
]
cfs = Confs(no, [nm, 0], qnd)
```

7.3 Constructing the basis

Having constructed the configurations, we now construct the basis of the Hilbert space. The '*basis*' is the collection of states that are linear combinations of the configurations carrying certain diagonal and $\mathbb{Z}_p$ off-diagonal quantum numbers (QNOffd).

---

[16]In general, many methods in FuzzifiED admits keyword arguments `num_th :: Int64` that specifies the number of threads and `disp_std :: Bool` that specifies whether or not the log shall be displayed. Hereafter, we will omit these two arguments.

The QNOffds supported by FuzzifiED are the $\mathbb{Z}_p$ symmetries in the form of

$$\mathcal{Z} : \; c_o \to \alpha_o^* c_{\pi_o}^{(p_o)}, \quad c_o^\dagger \to \alpha_o c_{\pi_o}^{(1-p_o)}, \tag{7.3}$$

where we use a notation $c^{(1)} = c^\dagger$ and $c^{(0)} = c$ for convenience, π_o is a permutation of the sites $1, \dots N_o$, α_o is a coefficient, and p_o specified whether or not particle-hole transformation is performed for the site. Note that one must guarantee that all these transformations commute with each other and also commute with the diagonal quantum numbers. In FuzzifiED, the QNOffds are recorded in the mutable type `QNOffd`.

`QNOffd` — Type

The mutable type contains the fields

- `perm :: Vector{Int64}` — a length-N_o array that records the permutation π_o.
- `ph :: Vector{Int64}` — a length-N_o array that records p_o to determine whether or not to perform a particle-hole transformation on a site.
- `fac :: Vector{ComplexF64}` — a length-N_o array that records the factor α_o in the transformation.
- `cyc :: Int64` — the cycle p.

It can be initialised by the methods

```
QNOffd(perm :: Vector{Int64}[, ph :: Vector{Int64}][, fac :: Vector{
    ComplexF64}][, cyc :: Int64])
QNOffd(perm :: Vector{Int64}, ph_q :: Bool[, fac :: Vector{ComplexF64}])
```

By default, `ph` is set to all 0, `fac` is set to all 1 and `cyc` is set to 2. If `ph_q` is set to be true, `ph` is set to all 1.

Several useful QNOffds are built-in

- `GetParityQNOffd(nm, nf[, permf, fac])` — the particle-hole transformation $\mathcal{P}$: $c_{mf}^\dagger \mapsto \alpha_f c_{m\pi_f}$, with the permutation of flavours π_f and the factors α_f stored in `permf` and `fac` as either an array or a dictionary.
- `GetFlavPermQNOffd(nm, nf, permf[, fac][, cyc])` — the flavour permutation transformation $\mathcal{Z} : c_{mf}^\dagger \mapsto \alpha_f c_{m\pi_f}^\dagger$, with the permutation of flavours π_f and the factors α_f stored in `permf` and `fac` as either an array or a dictionary, and the cycle stored in `cyc`.

- `GetRotyQNOffd(nm, nf)` — the π-rotation with respect to the y-axis $\mathcal{R}_y : c^\dagger_{mf} \mapsto (-1)^{m+s} c^\dagger_{(-m)f}$

After implementing the QNOffds, a state on the new basis should look like

$$|I\rangle = \lambda_{i_{I1}}|i_{I1}\rangle + \lambda_{i_{I2}}|i_{I2}\rangle + \cdots + \lambda_{i_{Im_I}}|i_{Im_I}\rangle, \tag{7.4}$$

where the $|i\rangle$'s are configurations, and $|I\rangle$ is a linear combination of them. In other words, the configurations are organised into groups of size m_I. In FuzzifiED, the basis $\{|I\rangle\}$ is recorded in the mutable type `Basis`.

`Basis` — Type

The mutable type contains the fields

- `cfs :: Confs` — the configurations $\{|i\rangle\}$ that respect the QNDiags.
- `dim :: Int64` — the dimension of the basis.
- `szz :: Int64` — the maximum size max m_I of groups.
- `cfgr :: Vector{Int64}` — an array of length `cfs.ncf` and records which group $|I\rangle$ each configuration $|i\rangle$ belong to.
- `cffac :: Vector{ComplexF64}` — an array of length `cfs.ncf` and records the coefficients λ_i of each configuration.
- `grel :: Matrix{Int64}` — a `szz`×`dim` matrix that records the configurations in each group $|i_{Ik}\rangle$ $(k = 1, \dots, m_I)$.
- `grsz :: Vector{Int64}` — an array of length `dim` that records the size m_I of each group.

It can be constructed by the methods

```
Basis(cfs :: Confs, secf :: Vector{ComplexF64}, qnf :: Vector{QNOffd})
Basis(cfs :: Confs)
```

where `secf` records the eigenvalue of each transformation, typically in the form $e^{i2\pi q/p}$ where p is the cycle and q is the $\mathbb{Z}_p$ charge.

In the example of Ising model Eq. (4.20), There are three $\mathbb{Z}_2$ symmetries, *viz.* the particle-hole transformation $\mathcal{P}$, the π-rotation along the y-axis $\mathcal{R}_y$, and the flavour (Ising) symmetry

$$\mathcal{P} : \ c^\dagger_{\sigma m} \mapsto \sigma c_{-\sigma, m}$$

$$\mathcal{Z} : \ c^\dagger_{\sigma m} \mapsto c^\dagger_{-\sigma, m}$$

$$\mathcal{R}_y : \ c^\dagger_{\sigma m} \mapsto c^\dagger_{\sigma, -m} \tag{7.5}$$

The code to generate the basis in the all-positive sector is

```
qnf = [
   QNOffd([ (m - 1) * nf + [2, 1][f] for m = 1 : nm for f = 1 : nf ], true,
      ComplexF64[ [-1, 1][f] for m = 1 : nm for f = 1 : nf ]),
   QNOffd([ (m - 1) * nf + [2, 1][f] for m = 1 : nm for f = 1 : nf ]),
   QNOffd([ (nm - m) * nf + f for m = 1 : nm for f = 1 : nf],
      ComplexF64[ iseven(m) ? 1 : -1 for m = 1 : nm for f = 1 : nf ])
]
bs = Basis(cfs, [1, 1, 1], qnf)
```

Alternatively, using the built-in functions

```
qnf = [
   GetParityQNOffd(nm, 2, [2, 1], [-1, 1]),
   GetFlavPermQNOffd(nm, 2, [2, 1]),
   GetRotyQNOffd(nm, 2)
]
bs = Basis(cfs, [1, 1, 1], qnf)
```

7.4 Recording the many-body operator terms

Having constructed the basis, we now construct the many-body operators. A general many-body operator can be written as

$$\mathcal{O} = \sum_{t=1}^{N_t} U_t c^{(p_{t1})}_{o_{t1}} c^{(p_{t2})}_{o_{t2}} \dots c^{(p_{tl_t})}_{o_{tl_t}}, \tag{7.6}$$

where $c^{(0)} = c$ and $c^{(1)} = c^\dagger$. In FuzzifiED, this is recorded as an array of `Term`, and each `Term` records the building block $U c^{(p_1)}_{o_1} c^{(p_2)}_{o_2} \dots c^{(p_l)}_{o_l}$.

`Term` — **Type**

The mutable type contains the fields

- coeff :: ComplexF64 — the coefficient U.
- cstr :: Vector{Int64} — a length-$2l$ array $\{p_1, o_1, p_2, o_2, \ldots p_l, o_l\}$ recording the operator string.

It can be initialised by the method

```
Term(coeff :: ComplexF64, cstr :: Vector{Int64})
```

The addition and multiplication of terms are supported, and the terms can be simplified by the method

```
SimplifyTerms(tms :: Vector{Term})
```

After the simplification, the resulting terms satisfy

1. Each term is normal-ordered — the creation operators are in front of the annihilation operators ; the site index of the creation operators are in ascending order and the annihilation operators in descending order.

2. Like terms are combined, and terms with zero coefficients are removed.

In FuzzifiED, several useful operator terms are built-in :

- GetDenIntTerms(nm, nf[, ps_pot][, mat_a[, mat_b]]) — the normal-ordered density-density interaction term in the Hamiltonian

$$\sum_{l\{m_i f_i\}} U_l C^l_{m_1 m_2 m_3 m_4} M^A_{f_1 f_4} M^B_{f_2 f_3} c^\dagger_{m_1 f_1} c^\dagger_{m_2 f_2} c_{m_3 f_3} c_{m_4 f_4} \tag{7.7}$$

where $C^l_{m_1 m_2 m_3 m_4}$ is given in Eq. (4.25).

- GetPairIntTerms(nm, nf, ps_pot, mat_a[, mat_b]) — the normal-ordered pair-pair interaction term in the Hamiltonian

$$\sum_{l\{m_i f_i\}} U_l C^l_{m_1 m_2 m_3 m_4} M^A_{f_1 f_2} M^B_{f_3 f_4} c^\dagger_{m_1 f_1} c^\dagger_{m_2 f_2} c_{m_3 f_3} c_{m_4 f_4}. \tag{7.8}$$

- GetPolTerms(nm, nf[, mat]) — the polarisation term in the Hamiltonian

$$\sum_{m f_1 f_2} c^\dagger_{m f_1} M_{f_1 f_2} c_{m f_2}. \tag{7.9}$$

- GetL2Terms(nm, nf) — the total angular momentum.

- GetC2Terms(nm, nf, mat_gen[, mat_tr]) — the quadratic Casimir

$$C_2 = \sum_{imm'\{f_i\}} \frac{(c^\dagger_{mf_1} G^i_{f_1 f_2} c_{mf_2})(c^\dagger_{m'f_3}(G^i_{f_3 f_4})^\dagger c_{m'f_4})}{\operatorname{tr} G^\dagger_i G_i} - (\text{trace}) \qquad (7.10)$$

where G^i are the generator matrices.

In the example of the Ising model, the code that records the Hamiltonian Eq. (4.20) is

```
using WignerSymbols
ps_pot = [ 4.75, 1.0 ] * 2.
h = 3.16
tms_hmt = Term[]
m = zeros(Int64, 4)
for m[1] = 0 : nm - 1, m[2] = 0 : nm - 1, m[3] = 0 : nm - 1
    m[4] = m[1] + m[2] - m[3]
    (m[4] < 0 || m[4] >= nm) && continue
    f = [0, 1, 1, 0]
    o = m .* nf .+ f .+ 1
    mr = m .- s

    val = ComplexF64(0)
    for l in eachindex(ps_pot)
        (abs(mr[1] + mr[2]) > nm - 1 || abs(mr[3] + mr[4]) > nm - 1) && break
        val += ps_pot[l] * (2 * nm - 2 * l + 1) * wigner3j(s, s, nm - 1, mr
            [1], mr[2], -mr[1] - mr[2]) * wigner3j(s, s, nm - 1, mr[4], mr[3],
             -mr[3] - mr[4])
    end
    tms_hmt += Terms(val, [1, o[1], 1, o[2], 0, o[3], 0, o[4]])
end
for m = 0 : nm - 1
    o = m * nf .+ [1, 2]
    tms_hmt += Terms(-h, [1, o[1], 0, o[2]])
    tms_hmt += Terms(-h, [1, o[2], 0, o[1]])
end
```

Alternatively, using the built-in functions

```
sg1 = [ 1 0 ; 0 0 ]
```

```
sg2 = [ 0 0 ; 0 1 ]
sgx = [ 0 1 ; 1 0 ]
sgz = [ 1 0 ; 0 -1 ]
ps_pot = [ 4.75, 1.0 ] * 2.0
fld_h = 3.16
tms_hmt = SimplifyTerms(
    GetDenIntTerms(nm, nf, ps_pot, sg1, sg2)
    - fld_h * GetPolTerms(nm, 2, sgx)
)
```

We also need to construct the total angular momentum. It is defined as

$$L^2 = L^+ L^- + (L^z)^2 - L^z. \tag{7.11}$$

As c_m carries the SO(3) spin-s representation,

$$L^z = \sum_{mf} m c_m^\dagger c_m, \quad L^\pm = \sum_{mf} \sqrt{(s \mp m)(s \pm m + 1)} c_{m\pm 1}^\dagger c_m. \tag{7.12}$$

We can first construct its building blocks and use the addition and multiplication of the terms

```
tms_lz = [ Term(m - s - 1, [1, (m - 1) * nf + f, 0, (m - 1) * nf + f]) for m
    = 1 : nm for f = 1 : nf ]
tms_lp = [ Term(sqrt((nm - m) * m), [1, m * nf + f, 0, (m - 1) * nf + f])
    for m = 1 : nm - 1 for f = 1 : nf ]
tms_lm = tms_lp'
tms_l2 = SimplifyTerms(tms_lz * tms_lz - tms_lz + tms_lp * tms_lm)
```

Alternatively, using the built-in functions,

```
tms_l2 = GetL2Terms(nm, nf)
```

7.5 Generating sparse matrix

Having obtained the terms in the many-body operator, we need to generate the matrix elements given the initial and final basis and find their eigenstates.

In FuzzifiED, the mutable type `Operator` records the terms together with information about its symmetry, the basis of the state it acts on, and the basis of the resulting state.

`Operator` — **Type**

The type can be initialised with the method

```
Operator(bsd :: Basis[, bsf :: Basis], terms :: Vector{Term} ; red_q ::
    Int64, sym_q :: Int64)
```

where the arguments

- bsd :: Basis — the basis of the initial state.
- bsf :: Basis — the basis of the final state. Facultative, the same as bsd by default.
- terms :: Vector{Term} — the terms.
- red_q :: Int64 — a flag that records whether or not the conversion to a sparse matrix can be simplified : if bsd and bsf have exactly the same set of quantum numbers, and the operator fully respects the symmetries, then red_q = 1 ; otherwise red_q = 0 ; Facultative, if bsf is omitted, 1 by default, otherwise 0 by default.
- sym_q :: Int64 — the symmetry of the operator : if its corresponding matrix is Hermitian, then sym_q = 1 ; if it is symmetric, then sym_q = 2 ; otherwise sym_q = 0. Facultative, if bsf is omitted, 1 by default, otherwise 0 by default.

The sparse matrix is recorded in the compressed sparse column (CSC) format, which is described in detail in Appendix A.2. In FuzzifiED, the sparse matrix is stored in the mutable type OpMat{T}

OpMat{T} — Type

where T is the type of the elements, it can either be ComplexF64 or Float64. It contains the fields

- dimd :: Int64, dimf :: Int64, nel :: Int64, symq :: Int64 — Parameters of the sparse matrix, the number of columns, rows, elements and the symmetry of matrix, respectively.
- colptr :: Vector{Int64} with length dimd :: Int64 + 1.
- rowid :: Vector{Int64} with length nel.
- elval :: Vector{T} with length nel.

It can be generated from the method

```
OpMat[{T}](op :: Operator)
```

7.6   Finding eigenstates

After generating the sparse matrix, the method `GetEigensystem` uses the Fortran Arpack package to calculate its lowest eigenstates.

`GetEigensystem` — Method

```
GetEigensystem(
    mat :: OpMat{T}, nst :: Int64 ;
    tol :: Float64, ncv :: Int64, initvec :: Vector{T}
) :: Tuple{Vector{T}, Matrix{T}}
```

The arguments are

- `mat :: OpMat{T}` — the matrix.

- `nst :: Int64` — the number of eigenstates to be calculated.

- `tol :: Float64` — the tolerence for the Arpack process. The default value is 10^{-8}.

- `ncv :: Int64` — an auxiliary parameter needed in the Arpack process. The default value is `max(2 * nst, nst + 10)`.

- `initvec :: Vector{T}` — the initial vector. If empty, a random initialisation shall be used. Facultative, empty by default.

The output is a tuple that includes two elements :

- A length-`nst` array recording the eigenvalues, and

- A `dimd`×`nst` matrix where every column records an eigenstate.

In the example of the Ising model, the code to calculate the lowest $N_{\mathrm{st}} = 10$ eigenstates from the basis and the Hamiltonian terms is

```
nst = 10
hmt = Operator(bs, tms_hmt)
hmt_mat = OpMat(hmt)
enrg, st = GetEigensystem(hmt_mat, nst)
```

7.7   Inner product of states, operators and transformations

Having obtained the eigenstates, we need to make measurements on them. The simplest kind of measurement is the inner product of a many-body operator with two states $\langle j|\mathcal{O}|i\rangle$.

FuzzifiED supports the inner product and vector product of `Operator` and `OpMat{T}` with vectors that represent the state

```
(op :: Operator) * (st_d :: Vector{T}) :: Vector{T}
(mat :: OpMat{T}) * (st_d :: Vector{T}) :: Vector{T}
(st_f :: Vector{T}) * (op :: Operator) * (st_d :: Vector{T}) :: T
(st_f :: Vector{T}) * (mat :: OpMat{T}) * (st_d :: Vector{T}) :: T
```

where `T` is the type of the elements. For example, the code to measure the angular momenta of each state is

```
tms_l2 = GetL2Terms(nm, 2)
l2 = Operator(bs, tms_l2)
l2_mat = OpMat(l2)
l2_val = [ st[:, i]' * l2_mat * st[:, i] for i in eachindex(enrg)]
```

One might also need to act transformations on the state $\mathcal{Z}|i\rangle$. In FuzzifiED, the mutable type `Transf` records the transformation together with the basis of the initial and final states

---

`Transf` — Type

The type can be initialised from a `QNOffd` by

```
Transf(bsd :: Basis[, bsf :: Basis], qnf :: QNOffd)
```

where the arguments

- `bsd :: Basis` — the basis of the initial state.
- `bsf :: Basis` — the basis of the final state. Facultative, the same as `bsd` by default.
- `qnf :: QNOffd` — records the transformation $c_o \to \alpha_o^* c_{\pi_o}^{(p_o)}$.

It can act on a state by

```
(trs :: Transf) * (st_d :: Vector{T}) :: Vector{T}
```

7.8 Measuring local observables

Local observables are a particularly useful kind of operators on the fuzzy sphere. Their value at a point on the sphere can be decomposed into spherical components, and the multiplication of the components follows the triple integral formula of monopole spherical harmonics

$$\mathcal{O}(\hat{\mathbf{n}}) = \sum_{lm} Y_{lm}^{(s)}(\hat{\mathbf{n}})\mathcal{O}_{lm}$$

$$(\mathcal{O}_1\mathcal{O}_2)_{lm} = \sum_{l_1 l_2 m_1 m_2} (\mathcal{O}_1)_{l_1 m_1} (\mathcal{O}_2)_{l_2 m_2}$$

$$\times (-1)^{s+m} \sqrt{\frac{(2l_1+1)(2l_2+1)(2l_3+1)}{4\pi}} \begin{pmatrix} l_1 & l_2 & l \\ m_1 & m_2 & -m \end{pmatrix} \begin{pmatrix} l_1 & l_2 & l \\ -s_1 & -s_2 & s \end{pmatrix}$$

$$(7.13)$$

In FuzzifiED, they are stored in the type `SphereObs`.

`SphereObs` — Type

The type contains the fields

- `s2 :: Int64` and `l2m :: Int64` — is twice the spin $2s$ and twice the maximal angular momentum $2l_{\max}$ of the observable.

- `get_comp :: Function` — a function that sends the component specified by a tuple of integers $(2l, 2m)$ to a list of terms that specifies the expression of the component.

- `stored_q :: Bool` — a boolean that specifies whether or not the components of the observable are stored.

- `comps :: Dict{Tuple{Int64, Int64}, Terms}` — each component of the observable stores in the format of a dictionary whose keys are the pairs of integers $(2l, 2m)$ and values are the lists of terms that specifies the expression of the component.

and can be initialised by the methods

```
SphereObs(s2 :: Int64, l2m :: Int64, get_comp :: Function)
SphereObs(s2 :: Int64, l2m :: Int64, comps :: Dict)
```

Their adjoint, addition and multiplication are supported. The related functions are

- `StoreComps(obs)` calculates and stores each component of the observable.
- `Laplacian(obs)` takes the Laplacian of an observable.
- `GetComponent(obs, l, m)` returns a spherical component of the observable $\mathcal{O}_{lm}$.
- `GetPointValue(obs, theta, phi)` returns an observable at one point $\mathcal{O}(\hat{\mathbf{n}})$.
- `GetIntegral(obs)` returns the uniform spatial integral of an observable.
- `FilterComponent(obs, flt)` filters a certain set of modes ofan observable.

Several important types of spherical observables are built-in in FuzzifiED[17]

- GetElectronObs(nm, nf, f) — electron annihilation operator $\psi_f(\hat{\mathbf{n}})$.

- GetDensityObs(nm, nf[, mat]) — density operator $n_M(\hat{\mathbf{n}}) = \sum_{ff'} \psi_f^\dagger(\hat{\mathbf{n}}) M_{ff'} \psi_{f'}(\hat{\mathbf{n}})$.

- GetPairingObs(nm, nf, mat) — pair operator $\Delta_M(\hat{\mathbf{n}}) = \sum_{ff'} \psi_f(\hat{\mathbf{n}}) M_{ff'} \psi_{f'}$.

In the example of Ising model, to calculate the OPE coefficient $f_{\sigma\sigma\epsilon} = \langle\sigma|n_{00}^z|\epsilon\rangle/\langle\sigma|n_{00}^z|0\rangle$, one need to first calculate the eigenstates in the $\mathbb{Z}_2$-odd sector

```
bs_m = Basis(cfs, [1, -1, 1], qnf)
hmt_m = Operator(bs_m, bs_m, tms_hmt ; red_q = 1, sym_q = 1)
hmt_mat_m = OpMat(hmt_m)
enrg_m, st_m = GetEigensystem(hmt_mat_m, 10)
st0 = st[:, 1]
ste = st[:, 2]
sts = st_m[:, 1]
```

and then construct the density operator

```
obs_nz = GetDensityObs(nm, 2, sgz)
tms_nz00 = SimplifyTerms(GetComponent(obs_nz, 0.0, 0.0))
nz00 = Operator(bs, bs_m, tms_nz00 ; red_q = 1)
f_sse = abs((sts' * nz00 * ste) / (sts' * nz00 * st0))
```

Besides the spherical observable, we also provide a type `AngModes` that superposes under the rule of angular momentum superposition instead of spherical harmonics triple integral

$$(\mathcal{A}_1 \mathcal{A}_2)_{lm} = \sum_{l_1 m_1 l_2 m_2} (\mathcal{A}_1)_{l_1 m_1} (\mathcal{A}_2)_{l_2 m_2} \langle l_1 m_1, l_2 m_2 | lm \rangle. \tag{7.14}$$

The interfaces are similar.

7.9 Measuring the entanglement

A non-local quantity that bears particular significance is the entanglement. To calculate the entanglement, we divide the sphere into two parts A and B. The reduced density matrix of part A is obtained by tracing the density matrix over the part B

$$\rho_A(\Psi) = \mathrm{tr}_B |\Psi\rangle\langle\Psi| \tag{7.15}$$

---

[17]By default, the operators are normalised by angular coordinate. A parameter `ObsNormRadSq` is used to normalise the observables on a sphere with radius R.

The entanglement entropy is $S = -\text{tr}\,\rho_A \log \rho_A$ and the entanglement spectrum is the collection of eigenvalues of ρ_A taken negative logarithm.

Here we sketch the process of the calculation. More detail is given in Ref. [59]. The creation operator in each orbital is divided into the creation on A part and the creation on B part

$$c_o^\dagger = \alpha_o c_{o,A}^\dagger + \beta_m c_{o,B}^\dagger, \tag{7.16}$$

where $|\alpha_o|^2 + |\beta_o|^2 = 1$. For the cut in orbital space m_c,

$$\alpha_{mf} = \Theta(m_c - m)$$

where Θ is the Heaviside function ; for the cut in real space along latitude circle θ_c,

$$\alpha_{mf} = \sqrt{\frac{\text{B}_{\cos^2 \theta_c/2}(s - m + 1, s + m + 1)}{\text{B}(s - m + 1, s + m + 1)}}$$

where $\text{B}_x(a, b)$ is the incomplete beta function.

To calculate the reduced density matrix, we decompose the state into the direct-product basis of two subsystems

$$|\Psi\rangle = \sum_{K_0} v_{K_0} |K_0\rangle = \sum_{I_A J_B} M_{I_A J_B} |I_A\rangle |J_B\rangle, \tag{7.17}$$

where the indices $K_0 \in \mathcal{H}, I_A \in \mathcal{H}_A, J_B \in \mathcal{H}_B$ are in the overall Hilbert space and the Hilbert space of subsystem A and B. The density matrix is then

$$\rho_A = \mathbf{M}\mathbf{M}^\dagger \tag{7.18}$$

and the entanglement spectrum can be obtained from the SVD decomposition of the $\mathbf{M}$ matrix. Like the Hamiltonian, the $\mathbf{M}$ matrix is block diagonal, and each block carries different quantum numbers of the Hilbert spaces of A and B subsystem[18].

In FuzzifiED, the decomposition of state vector v_{K_0} into matrix $M_{I_A J_B}$ is done by the function `StateDecompMat`, and the calculation of entanglement spectrum is done by the function `GetEntSpec`

`GetEntSpec` — **Function**

---

[18]The α_o and M_{IJ} in our convention is equivalent to $\mathcal{F}_{m,A}$ and $R_{\mu\nu}^A$ in the convention of Ref. [59], the conversions are $\alpha_{mf} = \sqrt{\mathcal{F}_{m,A}}$ and $M_{IJ} = R_{\mu\nu}^A/\sqrt{p}$.

```
GetEntSpec(
    st :: Vector{<:Number}, bs0 :: Basis,
    secd_lst :: Vector{Vector{Vector{Int64}}},
    secf_lst :: Vector{Vector{Vector{<:Number}}} ;
    qnd_a :: Vector{QNDiag}[, qnd_b :: Vector{QNDiag}],
    qnf_a :: Vector{QNOffd}[, qnf_b :: Vector{QNOffd}],
    amp_oa :: Vector{<:Number}[, amp_ob :: Vector{<:Number}]
) :: Dict{@NamedTuple{secd_a, secf_a, secd_b, secf_b}, Vector{Float64}}
```

The arguments are

- `st :: Vector{<:Number}` — the state to be decomposed into the direct-product basis of two subsystems.

- `bs0 :: Basis` — the basis of the original state.

- `secd_lst :: Vector{Vector{Vector{Int64}}}` — the list of QNDiag sectors of subsystems to be calculated. Each of its elements is a two-element vector ; the first specifies the sector for subsystem A, and the second specifies the sector for subsystem B.

- `secf_lst :: Vector{Vector{Vector{ComplexF64}}}` — the list of QNOffd sectors of subsystems to be calculated. Each of its elements is a two-element vector ; the first specifies the sector for subsystem A, and the second specifies the sector for subsystem B.

- `qnd_a :: Vector{QNDiag}`, `qnd_b :: Vector{QNDiag}`, `qnf_a :: Vector{QNOffd}` and `qnf_b :: Vector{QNOffd}` — the diagonal and off-diagonal quantum numbers of the subsystems A and B. By default `qnd_b = qnd_a` and `qnf_b = qnf_a`.

- `amp_oa :: Vector{ComplexF64}` and `amp_ob :: Vector{ComplexF64}` — the arrays that specify the amplitudes α_o and β_o. By default $\beta_o = \sqrt{1 - \alpha_o^2}$.

and the output is a dictionary whose keys are named tuples that specify the sector containing entries `secd_a`, `secf_a`, `secd_b`, `secf_b` and values are lists of eigenvalues of the density matrix in those sectors.

In the example of the Ising model, to calculate the entanglement entropy cut from the equator, we first need to specify the quantum numbers of the subsystems : the conservation of N_e, L_z and the $\mathbb{Z}_2$ symmetry.

```
qnd_a = [ GetNeQNDiag(no), GetLz2QNDiag(nm, nf) ]
qnf_a = [ GetFlavPermQNOffd(nm, nf, [2, 1]) ]
```

we then specify the sectors to calculate : The number of electrons in subsystem A run from 0 to N_m ; the angular momenta in subsystem A can take all permitted values[19] ; for subsystem B, $N_{e,B} = N_m - N_{e,A}, L_{z,B} = -L_{z,A}$; the $\mathbb{Z}_2$ sectors of the two subsystems are the same.

```
secd_lst = Vector{Vector{Int64}}[]
for nea = 0 : nm
    neb = nm - nea
    for lza = -min(nea, neb) * (nm - 1) : 2 : min(nea, neb) * (nm - 1)
        lzb = -lza
        push!(secd_lst, [[nea, lza], [neb, lzb]])
    end
end
secf_lst = [ [[1], [1]], [[-1], [-1]] ]
```

Finally, we specify the list of amplitudes α_o.

```
amp_oa = [ sqrt(beta_inc(m, nm - m + 1, 0.5)) for m = 1 : nm for f = 1 : nf
    ]
```

To calculate the entanglement spectrum,

```
ent_spec = GetEntSpec(st_g, bs, secd_lst, secf_lst ; qnd_a, qnf_a, amp_oa)
```

The entanglement entropy can be calculated by collecting all the eigenvalues of the density matrix.

```
eig_rho = vcat(values(ent_spec)...)
ent_entropy = -sum(eig_rho .* log.(eig_rho))
```

7.10   Fuzzifino — module for boson-fermion mixture

Fuzzifino is a module for ED calculation on the fuzzy sphere for systems with both bosons and fermions. The procedure is similar to FuzzifiED. We define several new types `SQNDiag`, `SQNOffd`, `SConfs`, `SBasis`, `STerm` and `SOperator` that is parallel to the original versions `QNDiag`, `QNOffd`, `Conf`, `Basis`, `Term` and `Operator`. Here we note several points.

•   For each configuration, a boson part and a fermion part are stored. The boson configurations are indexed in ascending order from the last site to the first site. A maximal total occupation should be given. The details are given in Appendix A.3.

---

[19]The sectors with no allowed configurations are automatically ignored

- The operator string in a term is still stored in the form $\{p_1, o_1, p_2, o_2, \ldots, p_l, o_l\}$, but now positive o represents fermions, and negative o represents boson with site number $|o|$. The bosonic creation and annihilation operator acts with an additional factor, $e.g.$, $b^\dagger|n\rangle = \sqrt{n+1}|n+1\rangle$, $b|n\rangle = \sqrt{n}|n-1\rangle$ for a single site.

- Several QNs and terms for purely bosonic models are built-in.

7.11 FuzzyManifolds — module for other geometries

Apart from the sphere, FuzzifiED also supports the calculation for interacting particles on the lowest Landau level on a torus. The procedure is similar to sphere, except for two differences :

- The orbitals are now labelled by a $\mathbb{Z}_{N_m}$ number $m = 1, \ldots, N_m$, and m and $m + N_m$ are identified. Correspondingly, the conservation of angular momentum L^z is now modified into a $\mathbb{Z}_{N_m}$ translation symmetry with $\sum_{mf} m N_{mf}$ mod N_m conserved. The `GetTorusLz2QNDiag` function constructs the corresponding QNDiag.

- The interacting potentials are still parametrised by the pseudopotentials in a similar sense, but the matrix $C^l_{m_1 m_2 m_3 m_4}$ is different. The functions `GetTorusDenIntTerms` and `GetTorusPairIntTerms` generate the density-density interaction terms and the pair-pair interaction terms from the pseudopotentials.

The detail of the setup is given in Appendix B.

8 Density matrix renormalisation group

Having introduced ED, we now turn to density matrix renormalisation group (DMRG) that deals with larger systems. We briefly describe its procedure and give an instruction for using FuzzifiED for DMRG.

8.1 DMRG with ITensor

Practically, the `dmrg` function in ITensor package automatically uses DMRG to optimise a matrix product state (MPS) to be the lowest eigenstate of a Hermitian Hamiltonian represented as a matrix product operator (MPO). To generate the input of the function, one needs to

1. construct a set of sites that carries a certain set of QNDiags,

2. construct a MPO representing the Hamiltonian on the sites from a set of terms (or OpSum in ITensor), and

3. construct an initial MPS on the sites in the desired symmetry sector.

In FuzzifiED, to use this extension for ITensor, one needs to use the packages `ITensors` and `ITensorMPS`. A new SiteType `"FuzzyFermion"` is defined that behaves similarly to the built-in `"Fermion"` type, and a single site can be generated from the QNDiags

```
siteind("FuzzyFermion" ; o :: Int64, qnd :: Vector{QNDiag})
```

The set of sites can be generated from the QNDiags (*cf.* Section 7.2) by

```
GetSites(qnd :: Vector{QNDiag})
```

In the example of the Ising model, for convenience, we exchange the Pauli matrices σ^x and σ^z so that the two flavours carry $\mathbb{Z}_2$-charge 0 and 1. The sites can be constructed by

```
nm = 12
nf = 2
no = nm * nf
sites = GetSites([
    GetNeQNDiag(nm * nf),
    GetLz2QNDiag(nm, nf),
    GetZnfChargeQNDiag(nm, nf)
])
```

In ITensor, the MPO is generated from an OpSum and the sites. The OpSum can be directly converted from the terms (*cf.* Section 7.4). In the example of the Ising model,

```
sgx = [ 0 1 ; 1 0 ]
sgz = [ 1 0 ; 0 -1 ]
ps_pot = [4.75, 1.0] ./ 2
tms_hmt = SimplifyTerms(
    GetDenIntTerms(nm, nf, ps_pot) -
    GetDenIntTerms(nm, nf, ps_pot, sgx) -0F
    3.16 * GetPolTerms(nm, nf, sgz)
)
os_hmt = OpSum(tms_hmt)
hmt = MPO(os_hmt, sites)
```

To calculate the $\mathbb{Z}_2$-even $L^z = 0$ sector, the initial state can be taken as all the $\mathbb{Z}_2$-even sites being filled and all the $\mathbb{Z}_2$-odd sites being empty[20].

```
cfi_p = [ [1, 0][f] for m = 1 : nm for f = 1 : nf ]
sti_p = MPS(sites, string.(cfi_p))
```

Having these ingredients ready, we can call the `dmrg` function. To ensure performance, the maximal bond dimension should be increased gradually, and the noise decreased gradually to 0. An example that deals with maximal bond dimension 500 is

```
E0, st0 = dmrg(hmt, sti_p ;
    nsweeps = 10,
    maxdim = [10,20,50,100,200,500],
    noise = [1E-4,3E-5,1E-5,3E-6,1E-6,3E-7],
    cutoff = [1E-8])
```

To generate a $\mathbb{Z}_2$-odd initial state, we can simply flip the spin on the first orbital

```
cfi_m = [ m == 1 ? [0, 1][f] : [1, 0][f] for m = 1 : nm for f = 1 : nf ]
sti_m = MPS(sites, string.(cfi_m))
Es, sts = dmrg(hmt, sti_m ;
    nsweeps = 10,
    maxdim = [10,20,50,100,200,500],
    noise = [1E-4,3E-5,1E-5,3E-6,1E-6,3E-7],
    cutoff = [1E-8])
```

The first excited $\mathbb{Z}_2$-even state can be generated by adding a projector $w|0\rangle\langle 0|$ to the MPO

```
Ee, ste = dmrg(hmt, [st0], sti_p ;
    nsweeps = 10,
    maxdim = [10,20,50,100,200,500],
    noise = [1E-4,3E-5,1E-5,3E-6,1E-6,3E-7],
    cutoff = [1E-8],
    weight = 100)
```

The inner product can be measured by the ITensor function `inner`. For example, to measure the angular momentum L^2 of the ground state,

```
tms_l2 = GetL2Terms(nm, 2)
```

---

[20] Note that ITensor takes the string `"1"` instead of the number `1` as occupied and `"0"` instead of `0` as filled.

```
l2 = MPO(OpSum(tms_l2), sites)
val_l20 = inner(st0', l2, st0)
```

To measure the OPE coefficient $f_{\sigma\sigma\epsilon} = \langle\sigma|n^x_{00}|\epsilon\rangle/\langle\sigma|n^x_{00}|0\rangle$[21]

```
obs_nx = GetDensityObs(nm, 2, sgx)
tms_nx00 = SimplifyTerms(GetComponent(obs_nx, 0.0, 0.0))
nx00 = MPO(OpSum(tms_nx00), sites)
f_sse = abs(inner(sts', nx00, ste) / inner(sts', nx00, st0))
```

8.2 The EasySweep extension

The extension EasySweep facilitates the management of the DMRG process. It automatically records the intermediate results and recover the run from these intermediate results if a job is stopped on HPC. It also manages the gradual increase of maximal bond dimensions and the determination of convergence by the criteria of energy. This extension contains the following functions :

- `GetMPOSites` returns the MPO and sites for given operator terms and a Hilbert space with given quantum numbers. The function first checks if the MPO and sites are already stored in a specified file. If they are already stored, they are read and returned. Otherwise, the sites are generated from the quantum numbers, and the MPO is generated from the terms. The MPO and sites are then written into the file and returned[22].

- `GetMPO` returns the MPO for given operator terms and a given set of sites. The function first checks if the MPO is already stored in a specified file. If it is already stored, it is read and returned. Otherwise, MPO is generated from the terms and the sites. The MPO and sites are then written into the file and returned.

- `SweepOne` performs one round of DMRG sweeps for a given maximal bond dimensions $\chi_{\max}$ and returns the energy and the MPS. The function first checks if the calculation has already been done and the results are already stored in a specified file. If they are already stored, they are read and returned. Otherwise, the DMRG process is performed with a specified maximal

---

[21] Note that the indices x and z have already been exchanged here.

[22] We recommend the package `ITensorMPOConstruction` stored at https://github.com/ITensor/ITensorMPOConstruction.jl to construct MPOs.

bond dimension and the number of sweeps. The sweeps end if the energy difference is less than a tolerence or alternative criteria. The resulting energy and MPS are written into the file and returned.

- EasySweep automatically performs several rounds of DMRG sweeps with increasing bond dimensions and returns energy and MPS. The function first checks whether the calculation has been partly done and the intermediate results have been stored in a specified file. The calculation is then picked up from the round that was previously stopped. The entire process is stopped if the energy difference between two rounds is less than a certain tolerence or the bond dimension of the result is less than 0.9 times the maximal bond dimension. The resulting energy and MPS are written into the file and returned.

To use this extension for ITensor, one needs to use the packages ITensors, ITensorMPS and HDF5. As the setup, a path needs to be created to store the result HDF5 files, and a method that converts OpSum and sites to MPO needs to be defined.

```
function MyMPO(os, sites)
    operatorNames = [ "I", "C", "Cdag", "N" ]
    opCacheVec = [ [OpInfo(ITensors.Op(name, n), sites[n]) for name in
        operatorNames] for n in eachindex(sites) ]
    return MPO_new(os, sites ; basis_op_cache_vec = opCacheVec)
end

path = "nm_$(nm)_tmp/"
mkpath(path)
```

After specifying the terms for the Hamiltonian tms_hmt and the QNDiags qnd like in the previous section, the sites and Hamiltonian MPO can be generated with the function GetMPOSites.

```
hmt, sites = GetMPOSites("hmt", tms_hmt, qnd ; path, mpo_method = MyMPO)
```

To generate the initial MPS that respects the $\mathbb{Z}_2$ symmetry, we can use a direct product state.

```
cfi_p = [ [1, 0][f] for m = 1 : nm for f = 1 : nf ]
sti_p = MPS(sites, string.(cfi_p))
cfi_m = [ m == 1 ? [0, 1][f] : [1, 0][f] for m = 1 : nm for f = 1 : nf ]
sti_m = MPS(sites, string.(cfi_m))
```

The lowest eigenenergies and the eigenstate MPSs $|0\rangle, |\sigma\rangle, |\epsilon\rangle$ can be easily generated by the function `EasySweep`.

```
E0, st0 = EasySweep("0", hmt, sti_p ; path)
Ee, ste = EasySweep("e", hmt, sti_p ; path, proj = ["0"])
Es, sts = EasySweep("s", hmt, sti_m ; path)
```

To measure the angular momentum L^2 of the ground state, we generate the MPO for L^2.

```
tms_l2 = GetL2Terms(nm, 2)
l2 = GetMPO("l2", tms_l2, sites ; path, mpo_method = MyMPO)
val_l20 = inner(st0', l2, st0)
```

Similarly, to measure the OPE coefficient $f_{\sigma\sigma\epsilon} = \langle\sigma|n^x_{00}|\epsilon\rangle/\langle\sigma|n^x_{00}|0\rangle$

```
obs_nx = GetDensityObs(nm, 2, sgx)
tms_nx00 = SimplifyTerms(GetComponent(obs_nx, 0.0, 0.0))
nx00 = GetMPO("nx00", tms_nx00, sites ; path, mpo_method = MyMPO)
f_sse = abs(inner(sts', nx00, ste) / inner(sts', nx00, st0))
```

9   Practical examples

FuzzifiED can help reproduce almost all the ED and DMRG results in fuzzy sphere works. We offer a series of such examples to help the users get started with the package. The code can be found in the `examples` directory of the source code repository.

1. `ising_spectrum.jl` calculates the spectrum of the 3d Ising model on the fuzzy sphere at $N_m = 12$. For each $(\mathcal{P}, \mathcal{Z}, \mathcal{R})$ sector, 20 states are calculated. This example reproduces Table I and Figure 4 in Ref. [1].

2. `ising_phase_diagram.jl` calculates the phase diagram of the fuzzy sphere Ising model by calculating the order parameter $\langle M^2\rangle$. This example reproduces Figure 3 in Ref. [1].

3. `ising_ope.jl` calculates various OPE coefficients at $N_m = 12$ by taking overlaps between CFT states and density operators and composite. This example reproduces Figure 2 and Table I in Ref. [2].

4. `ising_correlator.jl` calculates the $\sigma\sigma$ 2-pt function on sphere and the $\sigma\sigma\sigma\sigma$ 4-pt function of 0 and ∞ and two points on sphere. This example reproduces Figures 1c and 2a in Ref. [3].

5. `ising_optimisation.jl` defines a cost function as the square sum of the deviations of descendants and stress tensor to evaluate the conformal symmetry for the Ising model and minimises this cost function to find the best parameter.

6. `ising_full_spectrum.jl` calculates the full spectrum of the 3d Ising model on the fuzzy sphere at $N_m = 10$ for sector $(\mathcal{P}, \mathcal{Z}, \mathcal{R}) = (+, +, +)$.

7. `ising_space_entangle.jl` calculates the entanglement entropy of the Ising ground state along the real space cut of $\theta = 0.500\pi$ and 0.499π respectively, and use these two data to extract finite-size F-function without subtracting the IQHE contribution. This example reproduces Figure 3 in Ref. [10].

8. `ising_orbital_entangle.jl` calculates the entanglement entropy of the Ising ground state along the orbital space cut at $m = 0$, and also the entanglement spectrum in the half-filled $l_z = 0, 1$ and both $\mathbb{Z}_2$ sectors.

9. `ising_generator.jl` examines the quality of conformal symmetry at $N_m = 12$ by calculating the matrix elements of conformal generators $P^z + K^z$ and compare the states $(P^z + K^z)|\Phi\rangle$ with the CFT expectations. This example reproduces Figure 7 in Ref. [14].

10. `defect_spectrum.jl` calculates the spectrum of magnetic line defect in the 3d Ising model in $l_z = 0, \mathcal{P} = \pm 1$ and $l_z = 1$ sectors, calibrated by bulk T. This example reproduces Table I in Ref. [6].

11. `defect_correlator.jl` calculates the 1-pt function σ and 2-pt function $\sigma\hat{\phi}$ of magnetic line defect in the 3d Ising model. The normalisation of the correlators requires extra bulk data. This example reproduces Figure 4 in Ref. [6].

12. `defect_changing.jl` calculates the spectrum of the defect-creation and changing operators of the magnetic line defect in the 3d Ising model. This example reproduces Table 2 and Figure 5 in Ref. [9].

13. `defect_overlap.jl` calculates the g-function of magnetic line defect in the 3d Ising model using the ovelaps between the bulk, defect ground state and the lowest defect-creation state. This example reproduces Figure 6 in Ref. [9].

14. `cusp_dim.jl` calculates the scaling dimension of the cusp of the magnetic line defect in the 3d Ising model as a function of the angle θ. This example reproduces Table 2, upper panel

in Ref. [11].

15. `surface_ordinary_spectrum.jl` calculates the spectrum of ordinary surface CFT in the 3d Ising model calibrated by surface displacement operator D in the orbital boundary scheme. This example reproduces Figures 3 and 4 in Ref. [12].

16. `surface_normal_spectrum.jl` calculates the spectrum of normal surface CFT in the 3d Ising model calibrated by surface displacement operator D in the orbital boundary scheme. This example reproduces Figure 5 in Ref. [12].

17. `o3_wf_spectrum.jl` calculates the spectrum of O(3) Wilson-Fisher CFT using the bilayer Heisenberg model. This example reproduces Table I and Figure 2 in Ref. [8].

18. `so5_spectrum.jl` calculates the spectrum of SO(5) DQCP on the fuzzy sphere. This example reproduces Table II in Ref. [4].

19. `sp3_spectrum.jl` calculates the spectrum of Sp(3) CFT on the fuzzy sphere. This example reproduces Table I in Ref. [16].

20. `ising_frac_fermion.jl` calculates the spectrum of the 3d Ising model on the fuzzy sphere for fermions at fractional filling $\nu = 1/3$. This example reproduces Figure 10 in Ref. [17].

21. `ising_frac_boson.jl` calculates the spectrum of the 3d Ising model on the fuzzy sphere for bosons at fractional filling $\nu = 1/2$ with the module Fuzzifino. This example reproduces Figure 12a, b in Ref. [17].

22. `potts_spectrum.jl` calculates the spectrum of 3d Potts model on the fuzzy sphere. This example reproduces Table I and Figure 4 in Ref. [18].

23. `circle_ising.jl` calculates the spectrum of 2d Ising CFT on a fuzzy thin torus. This example reproduces Figure 4 and Tables I–III in Ref. [19].

24. `ising_spectrum_krylov.jl` calculates the spectrum of the 3d Ising model on the fuzzy sphere by calling the `eigsolve` function in `KrylovKit.jl` instead of Arpack.

25. `ising_spectrum_cuda.jl` calculates the spectrum of the 3d Ising model on the fuzzy sphere for one sector by performing the sparse matrix multiplication on CUDA.

Acknowledgments

A Data structures in exact diagonalisation

In this appendix, we describe several data structures used in the exact diagonalisation (ED) process, *viz.* the Lin table for the reverse look-up process that returns the index from the binary string, the compressed sparse column (CSC) format to store the sparse matrices, and the indexing of boson configurations.

A.1 Construction of Lin table

In this appendix, we describe the construction of the Lin table used for a reverse look-up process that returns the index i from the binary string $conf_i$ mentioned in Section 7.2.

Each binary string $conf_i$ is divided into a left part $l(conf_i)$ from the N_{or}-th bit to the (N_o-1)-th bit and a right part $r(conf_i)$ from the 0-th bit to the $(N_{or}-1)$-th bit from the right. The cut N_{or} is typically taken as $N_o/2$. Two arrays lid and rid with length $2^{N_o-N_{or}}$ and $2^{N_{or}}$ are constructed, such that

$$i = lid_{l(conf_i)} + rid_{r(conf_i)}. \qquad (A.1)$$

c	lid_c	rid_c	i	$conf_i$	l_i	r_i	lid_{l_i}	rid_{r_i}	i	$conf_i$	l_i	r_i	lid_{l_i}	rid_{r_i}
0	1	0	1	000111	0	7	1	0	11	100011	4	3	11	0
1	2	0	2	001011	1	3	2	0	12	100101	4	5	11	1
2	5	1	3	001101	1	5	2	1	13	100110	4	6	11	2
3	8	0	4	001110	1	6	2	2	14	101001	5	1	14	0
4	11	2	5	010011	2	3	5	0	15	101010	5	2	14	1
5	14	1	6	010101	2	5	5	1	16	101100	5	4	14	2
6	17	2	7	010110	2	6	5	2	17	110001	6	1	17	0
7	20	0	8	011001	3	1	8	0	18	110010	6	2	17	1
			9	011010	3	2	8	1	19	110100	6	4	17	2
			10	011100	3	4	8	2	20	111000	7	0	20	0

Table 2. An example of explicit construction of the Lin table in the configurations on 6 sites filled with 3 particles. Here we use a shorthand notation $l_i = l(conf_i)$ and $r_i = r(conf_i)$

As a simple example, consider the configurations on 6 sites filled with 3 particles. The construction is listed in Table 2.

A.2   Compressed sparse column (CSC) sparse matrix

In this appendix, we describe the storage of sparse matrices in the format of compressed sparse column (CSC) mentioned in Section 7.5.

In the CSC format, the matrix is stored in three arrays. We first index elements from 1 to the number of elements N_{el} in the ascending order from the column to column, and in each column from row to row.

1.   The array *colptr* records the column index of each element. It is of length $N+1$ where N is the number of columns. The first N elements record the index of the first element in each column ; the last element records $N_{el} + 1$, so that the elements that belong to the i-th column have indices from $colptr_i$ to $colptr_{i+1} - 1$.

2.   The array *rowid* records the row index of each element and is of length N_{el}.

3.   The array *elval* records the value of each element and is of length N_{el}.

We give a simple example in Table 3.

$$M = \begin{pmatrix} 10 & 40 & 0 & 0 \\ 20 & 0 & 70 & 80 \\ 0 & 0 & 0 & 90 \\ 30 & 50 & 0 & 100 \\ 0 & 60 & 0 & 0 \end{pmatrix}$$

i	1	2	3	4	5	6	7	8	9	10
$colptr_i$	1	4	7	8	11					
$rowid_i$	1	2	4	1	4	5	2	2	3	4
$elval_i$	10	20	30	40	50	60	70	80	90	100

Table 3. An example of the storing a 4×5 matrix into the CSC format.

A.3 Indexing the boson configurations

In this appendix, we describe the indexing of the bosonic configurations with N_o sites and a maximal total occupation $N_{\max}$ mentioned in Section 7.10. This is equivalent to the common procedure indexing $N_o + 1$ sites with a fixed occupation $N_e = N_{\max}$. The boson configurations are indexed in ascending order from the last site to the first site. *E.g.*, for 3 sites with maximal occupation 2, the configurations for $n_3 n_2 n_1$ are 000, 001, 002, 010, 011, 020, 100, 101, 110, 200 numbered from 1 to 10.

The total number of configurations is the number of non-negative integer solutions to the Diophantine equation $n_1 + n_2 + \cdots + n_{N_o} + n_{N_o+1} = N_{\max}$, which is

$$N_{\mathrm{cf}}(N_o, N_{\max}) = \frac{(N_o + N_{\max})!}{N_o! N_{\max}!} \tag{A.2}$$

The index of the

$$
\begin{aligned}
id(n_{N_o} n_{N_o-1} \dots n_1; N_{\max}) = {}& N_{\mathrm{cf}}(0) + N_{\mathrm{cf}}(1) + \cdots + N_{\mathrm{cf}}(n_{N_o} - 1) \\
& + N_{\mathrm{cf}}(n_{N_o} 1) + N_{\mathrm{cf}}(n_{N_o} 2) + \cdots + N_{\mathrm{cf}}(n_{N_o}(n_{N_o-1} - 1)) \\
& + \dots \\
& + N_{\mathrm{cf}}(n_{N_o} \dots n_2 0) + N_{\mathrm{cf}}(n_{N_o} \dots n_2 1) + \cdots + N_{\mathrm{cf}}(n_{N_o} \dots n_2 n_1), \tag{A.3}
\end{aligned}
$$

where $N_{\mathrm{cf}}(n_{N_o} n_{N_o-1} \dots n_o)$ is the number of configurations that begin with $n_{N_o} n_{N_o-1} \dots n_o$

$$N_{\mathrm{cf}}(n_{N_o} n_{N_o-1} \dots n_o) = \frac{[(o-1) + (N_{\max} - n_{N_o} - \cdots - n_o)]!}{(o-1)!(N_{\max} - n_{N_o} - \cdots - n_o)!}. \tag{A.4}$$

Hence, the index of each configuration is

$$id(n_{N_o} n_{N_o-1} \dots n_1; N_{\max}) = \sum_{o=1}^{N_o} \sum_{i=0}^{n_o} \frac{\left[(o-1) + \left(N_{\max} - \Sigma_{o'=o}^{N_o} n_{o'}\right)\right]!}{(o-1)! \left(N_{\max} - \Sigma_{o'=o}^{N_o} n_{o'}\right)!} \tag{A.5}$$

The reverse look-up is realised by a similar Lin-table, with the left part and right part taken as

$$l(\{n_o\}) = id(n_{N_o} n_{N_o-1} \dots n_{N_{or}+1}; N_{\max}) \quad \text{and} \quad r(\{n_o\}) = id(n_{N_{or}} n_{N_{or}-1} \dots n_1; N_{\max}). \tag{A.6}$$

B Landau levels and pseudopotentials on the torus

In this appendix, we describe the construction of Landau levels and pseudopotentials on the torus [99–101], mentioned in Sections 4.4 and 7.11.

B.1 Lowest Landau level

The torus is nothing more than a $l_x \times l_y$ rectangle with periodic boundary conditions, *i.e.*, $x \sim x + l_x$ and $y \sim y + l_y$. We adopt the Landau gauge $A_x = 0, A_y = xB$ and set the magnetic length as the unit length $l_B = 1$. In this way, the number of orbitals is $N_m = l_x l_y / 2\pi$. The lowest Landau level wavefunctions are labelled by the momenta along y direction m

$$\phi_m(x, y) = \text{const.} \times \sum_{n=-\infty}^{\infty} \exp\left[-\tfrac{1}{2}(x - X_m + nl_x)^2 + iX_m(y + kl_y) \right], \qquad \text{(B.1)}$$

where $m = 1, ..., N_m$, and

$$X_m = 2\pi m / l_y.$$

Each orbital is localised in the vicinity of a stripe $x = X_m$. We can then similarly project the fermion operators onto the lowest Landau level

$$\hat{\psi}_f(\mathbf{r}) = \hat{\phi}_m(\mathbf{r})\hat{c}_{mf}, \qquad \mathbf{r} = (x, y) \qquad \text{(B.2)}$$

B.2 Many-body interaction

The simplest many-body interaction term is the density-density interaction with a potential function. Omitting the flavour index, it can be expressed as

$$H_{\text{int}} = \frac{1}{N_m^2} \int d^2\mathbf{r}_1 d^2\mathbf{r}_2 \, U(\mathbf{r}_1 - \mathbf{r}_2) n(\mathbf{r}_1) n(\mathbf{r}_2), \qquad n(\mathbf{r}) = \psi^\dagger(\mathbf{r})\psi(\mathbf{r}) \qquad \text{(B.3)}$$

The interaction potential can be expanded into Fourier modes

$$U(\mathbf{r}) = \sum_{\mathbf{q}} U_{\mathbf{q}} e^{i\mathbf{q}\cdot\mathbf{r}}, \qquad \text{(B.4)}$$

where the allowed momenta are labelled by two integers s and t

$$\mathbf{q} = (2\pi s / l_x, 2\pi t / l_y) \qquad \text{(B.5)}$$

After performing the spatial integrals, the Hamiltonian is expressed in the orbital space as[23]

$$H_{\text{int}} = \sum_{m_1 m_2 m_3 m_4} \sum_{\mathbf{q}} \frac{U_{\mathbf{q}}}{2N_m} e^{iq_x(X_{m_1} - X_{m_3})} e^{-q^2/2} \delta'_{t, m_1 - m_4} \delta'_{m_1 + m_2, m_3 + m_4} c^\dagger_{m_1} c^\dagger_{m_2} c_{m_3} c_{m_4}, \qquad \text{(B.6)}$$

---

[23] In many literatures, the summation excludes the origin point $\mathbf{q} = (0, 0)$, but here the interaction of background charge in fact plays certain role.

where the Kronecker delta is defined in the sense of mod N_m

$$\delta'_{m,n} = \begin{cases} 1, & m = n \mod N_m \\ 0, & \text{otherwise.} \end{cases}$$

The interactions can alternatively be organised into pseudopotentials U_l, describing the energy cost of a pair of electrons with relative angular momentum l.

$$U_l = \int \frac{\mathrm{d}^2\mathbf{q}}{2\pi} V_{\mathbf{q}} e^{-q^2} L_l(q^2), \tag{B.7}$$

where L_l is the Laguerre polynomial. Finally, the interaction Hamiltonian reads

$$H_{\text{int}} = \sum_{lm_1m_2m_3m_4} U_l C^l_{m_1m_2m_3m_4} c^\dagger_{m_1} c^\dagger_{m_2} c_{m_3} c_{m_4}$$

$$C^l_{m_1m_2m_3m_4} = \frac{1}{N_m} \sum_{\mathbf{q}} e^{iq_x(X_{m_1}-X_{m_3})} e^{-q^2/2} L_l(q^2) \delta'_{t,m_1-m_4} \delta'_{m_1+m_2,m_3+m_4}. \tag{B.8}$$

C   Tutorial code

In Sections 7 and 8, we demonstrate the usage of FuzzifiED interfaces with an example that

1.   calculates the lowest eigenstates in the symmetry sector $L^z = 0$ and $(\mathcal{P}, \mathcal{Z}, \mathcal{R}) = (+,+,+)$ (for DMRG, $\mathcal{Z} = +$ only),

2.   measures their total angular momenta (for DMRG, the ground state only), and

3.   calcultes the OPE coefficient $f_{\sigma\sigma\epsilon} = \langle\sigma|n^z_{00}|\epsilon\rangle/\langle\sigma|n^z_{00}|0\rangle$.

In this appendix, we collect the full version of the codes, including

1.   the ED code that uses only the core functions,

2.   the ED code that uses the built-in models,

3.   the DMRG code that converts the format into ITensor, and

4.   the DMRG code that uses the EasySweep extension.

C.1   ED using core functions

```
1 using FuzzifiED
2 using WignerSymbols
3
```

```
 4 let
 5
 6 nf = 2
 7 nm = 12
 8 no = nf * nm
 9 qnd = [
10    QNDiag(fill(1, no)),
11    QNDiag([ 2 * m - nm - 1 for m = 1 : nm for f = 1 : nf ])
12 ]
13 cfs = Confs(no, [nm, 0], qnd)
14
15 qnf = [
16    QNOffd([ (m - 1) * nf + [2, 1][f] for m = 1 : nm for f = 1 : nf ], true,
17        ComplexF64[ [-1, 1][f] for m = 1 : nm for f = 1 : nf ]),
18    QNOffd([ (m - 1) * nf + [2, 1][f] for m = 1 : nm for f = 1 : nf ]),
19    QNOffd([ (nm - m) * nf + f for m = 1 : nm for f = 1 : nf],
20        ComplexF64[ iseven(m) ? 1 : -1 for m = 1 : nm for f = 1 : nf ])
21 ]
22 bs = Basis(cfs, [1, 1, 1], qnf)
23
24 ps_pot = [ 4.75, 1.0 ] * 2.
25 h = 3.16
26 tms_hmt = Term[]
27 m = zeros(Int64, 4)
28 for m[1] = 0 : nm - 1, m[2] = 0 : nm - 1, m[3] = 0 : nm - 1
29    m[4] = m[1] + m[2] - m[3]
30    (m[4] < 0 || m[4] >= nm) && continue
31    f = [0, 1, 1, 0]
32    o = m .* nf .+ f .+ 1
33    mr = m .- s
34
35    val = ComplexF64(0)
36    for l in eachindex(ps_pot)
37        (abs(mr[1] + mr[2]) > nm - l || abs(mr[3] + mr[4]) > nm - l) && break
38        val += ps_pot[l] * (2 * nm - 2 * l + 1) * wigner3j(s, s, nm - l, mr
            [1], mr[2], -mr[1] - mr[2]) * wigner3j(s, s, nm - l, mr[4], mr[3],
```

```
                 -mr[3] - mr[4])
39      end
40      tms_hmt += Terms(val, [1, o[1], 1, o[2], 0, o[3], 0, o[4]])
41  end
42  for m = 0 : nm - 1
43      o = m * nf .+ [1, 2]
44      tms_hmt += Terms(-h, [1, o[1], 0, o[2]])
45      tms_hmt += Terms(-h, [1, o[2], 0, o[1]])
46  end
47
48  hmt = Operator(bs, tms_hmt)
49  hmt_mat = OpMat(hmt)
50  enrg, st = GetEigensystem(hmt_mat, 10)
51
52  tms_lz = [ Term(m - s - 1, [1, (m - 1) * nf + f, 0, (m - 1) * nf + f]) for m
        = 1 : nm for f = 1 : nf ]
53  tms_lp = [ Term(sqrt((nm - m) * m), [1, m * nf + f, 0, (m - 1) * nf + f])
        for m = 1 : nm - 1 for f = 1 : nf ]
54  tms_lm = tms_lp'
55  tms_l2 = SimplifyTerms(tms_lz * tms_lz - tms_lz + tms_lp * tms_lm)
56  l2 = Operator(bs, tms_l2)
57  l2_mat = OpMat(l2)
58  l2_val = [ st[:, i]' * l2_mat * st[:, i] for i in eachindex(enrg)]
59
60  bs_m = Basis(cfs, [ 1, -1, 1 ], qnf)
61  hmt = Operator(bs_m, bs_m, tms_hmt ; red_q = 1, sym_q = 1)
62  hmt_mat = OpMat(hmt)
63  enrg_m, st_m = GetEigensystem(hmt_mat, 10)
64  st0 = st[:, 1]
65  ste = st[:, 2]
66  sts = st_m[:, 1]
67
68  tms_nz00 = Term[]
69  for m = 0 : nm - 1
70      o = m * nf .+ [1, 2]
71      tms_nz00 += Terms( 1 / nm, [1, o[1], 0, o[1]])
```

```
72    tms_nz00 += Terms(-1 / nm, [1, o[2], 0, o[2]])
73  end
74  nz00 = Operator(bs, bs_m, tms_nz00 ; red_q = 1)
75  f_sse = abs((sts' * nz00 * ste) / (sts' * nz00 * st0))
76
77  end
```

C.2   ED using built-in models

```
1  using FuzzifiED
2  sg1 = [ 1 0 ; 0 0 ]
3  sg2 = [ 0 0 ; 0 1 ]
4  sgx = [ 0 1 ; 1 0 ]
5  sgz = [ 1 0 ; 0 -1 ]
6
7  nm = 12
8  qnd = [
9      GetNeQNDiag(2 * nm),
10     GetLz2QNDiag(nm, 2)
11  ]
12  cfs = Confs(2 * nm, [nm, 0], qnd)
13
14  qnf = [
15     GetParityQNOffd(nm, 2, [2, 1], [-1, 1]),
16     GetFlavPermQNOffd(nm, 2, [2, 1]),
17     GetRotyQNOffd(nm, 2)
18  ]
19  bs = Basis(cfs, [1, 1, 1], qnf)
20
21  tms_hmt = SimplifyTerms(
22     GetDenIntTerms(nm, nf, 2 .* [4.75, 1.0], sg1, sg2) -
23     3.16 * GetPolTerms(nm, 2, sgx)
24  )
25
26  hmt = Operator(bs, tms_hmt)
27  hmt_mat = OpMat(hmt)
28  enrg, st = GetEigensystem(hmt_mat, 10)
```

```
29
30  tms_l2 = GetL2Terms(nm, 2)
31  l2 = Operator(bs, tms_l2)
32  l2_mat = OpMat(l2)
33  l2_val = [ st[:, i]' * l2_mat * st[:, i] for i in eachindex(enrg)]
34
35  bs_m = Basis(cfs, [1, -1, 1], qnf)
36  hmt = Operator(bs_m, bs_m, tms_hmt ; red_q = 1, sym_q = 1)
37  hmt_mat = OpMat(hmt)
38  enrg_m, st_m = GetEigensystem(hmt_mat, 10)
39  st0 = st[:, 1]
40  ste = st[:, 2]
41  sts = st_m[:, 1]
42
43  obs_nz = GetDensityObs(nm, 2, sgz)
44  tms_nz00 = SimplifyTerms(GetComponent(obs_nz, 0.0, 0.0))
45  nz00 = Operator(bs, bs_m, tms_nz00 ; red_q = 1)
46  f_sse = abs((sts' * nz00 * ste) / (sts' * nz00 * st0))
```

C.3   DMRG using format conversion into ITensor

```
 1  using FuzzifiED
 2  using ITensors, ITensorMPS
 3  FuzzifiED.ElementType = Float64
 4  sgx = [ 0 1 ; 1 0 ]
 5  sgz = [ 1 0 ; 0 -1 ]
 6
 7  nm = 12
 8  nf = 2
 9  no = nm * nf
10
11  sites = GetSites([
12      GetNeQNDiag(nm * nf),
13      GetLz2QNDiag(nm, nf),
14      GetZnfChargeQNDiag(nm, nf)
15  ])
16  ps_pot = [4.75, 1.0] ./ 2
```

```
17 tms_hmt = SimplifyTerms(
18     GetDenIntTerms(nm, nf, ps_pot) -
19     GetDenIntTerms(nm, nf, ps_pot, sgx) -
20     3.16 * GetPolTerms(nm, nf, sgz)
21 )
22 os_hmt = OpSum(tms_hmt)
23 hmt = MPO(os_hmt, sites)
24
25 cfi_p = [ [1, 0][f] for m = 1 : nm for f = 1 : nf ]
26 sti_p = MPS(sites, string.(cfi_p))
27 cfi_m = [ m == 1 ? [0, 1][f] : [1, 0][f] for m = 1 : nm for f = 1 : nf ]
28 sti_m = MPS(sites, string.(cfi_m))
29
30 E0, st0 = dmrg(hmt, sti_p ;
31     nsweeps = 10,
32     maxdim = [10,20,50,100,200,500],
33     noise = [1E-4,3E-5,1E-5,3E-6,1E-6,3E-7],
34     cutoff = [1E-8])
35 Ee, ste = dmrg(hmt, [st0], sti_p ;
36     nsweeps = 10,
37     maxdim = [10,20,50,100,200,500],
38     noise = [1E-4,3E-5,1E-5,3E-6,1E-6,3E-7],
39     cutoff = [1E-8],
40     weight = 100)
41 Es, sts = dmrg(hmt, sti_m ;
42     nsweeps = 10,
43     maxdim = [10,20,50,100,200,500],
44     noise = [1E-4,3E-5,1E-5,3E-6,1E-6,3E-7],
45     cutoff = [1E-8])
46
47 tms_l2 = GetL2Terms(nm, 2)
48 l2 = MPO(OpSum(tms_l2), sites)
49 val_l20 = inner(st0', l2, st0)
50
51 obs_nx = GetDensityObs(nm, 2, sgx)
52 tms_nx00 = SimplifyTerms(GetComponent(obs_nx, 0.0, 0.0))
```

```
53 nx00 = MPO(OpSum(tms_nx00), sites)
54 f_sse = abs(inner(sts', nx00, ste) / inner(sts', nx00, st0))
```

C.4   DMRG with Easy Sweep

```
 1 using FuzzifiED
 2 using ITensors, ITensorMPS, HDF5
 3 using ITensorMPOConstruction
 4 sgx = [ 0 1 ; 1 0 ]
 5 sgz = [ 1 0 ; 0 -1 ]
 6
 7 function MyMPO(os, sites)
 8     operatorNames = [ "I", "C", "Cdag", "N" ]
 9     opCacheVec = [ [OpInfo(ITensors.Op(name, n), sites[n]) for name in
         operatorNames] for n in eachindex(sites) ]
10     return MPO_new(os, sites ; basis_op_cache_vec = opCacheVec)
11 end
12
13 nm = 12
14 nf = 2
15 no = nm * nf
16
17 path = "nm_$(nm)_tmp/"
18 mkpath(path)
19
20 ps_pot = [4.75, 1.0] ./ 2
21 tms_hmt = SimplifyTerms(
22     GetDenIntTerms(nm, nf, ps_pot) -
23     GetDenIntTerms(nm, nf, ps_pot, sgx) -
24     3.16 * GetPolTerms(nm, 2, sgz)
25 )
26 qnd = [
27     GetNeQNDiag(no),
28     GetLz2QNDiag(nm, nf),
29     GetZnfChargeQNDiag(nm, nf)
30 ]
31 hmt, sites = GetMPOSites("hmt", tms_hmt, qnd ; path, mpo_method = MyMPO)
```

```
32
33  cfi_p = [ [1, 0][f] for m = 1 : nm for f = 1 : nf ]
34  sti_p = MPS(sites, string.(cfi_p))
35  cfi_m = [ m == 1 ? [0, 1][f] : [1, 0][f] for m = 1 : nm for f = 1 : nf ]
36  sti_m = MPS(sites, string.(cfi_m))
37
38  E0, st0 = EasySweep("0", hmt, sti_p ; path)
39  Ee, ste = EasySweep("e", hmt, sti_p ; path, proj = ["0"])
40  Es, sts = EasySweep("s", hmt, sti_m ; path)
41
42  tms_l2 = GetL2Terms(nm, 2)
43  l2 = GetMPO("l2", tms_l2, sites ; path, mpo_method = MyMPO)
44  l2_val = st0' * l2 * stg
45
46  obs_nx = GetDensityObs(nm, 2, sgx)
47  tms_nx00 = SimplifyTerms(GetComponent(obs_nx, 0.0, 0.0))
48  nx00 = GetMPO("nx00", tms_nx00, sites ; path, mpo_method = MyMPO)
49  f_sse = abs(inner(sts', nx00, ste) / inner(sts', nx00, st0))
```

D   Glossary for interfaces in FuzzifiED

AngModes

Basis

Confs

CuSparseMatrixCSC

EasySweep

FuzzifiED.ElementType

FilterComponent

FilterL2

GetBosonC2STerms

GetBosonDenIntSTerms

GetBosonFlavSQNDiag

GetBosonFlavPermSQNOffd

GetBosonL2STerms

GetBosonLz2SQNDiag

GetBosonPairIntSTerms

GetBosonPolSTerms

GetBosonRotySQNOffd

GetC2Terms

GetComponent

GetConfId

GetConfWeight

GetDenIntTerms

GetDensityMod

GetDensityObs

GetEigensystem

GetEigensystemCuda

GetEigensystemKrylov

GetElectronMod

GetElectronObs

GetEntSpec

GetFlavPermQNOffd

GetFlavQNDiag

GetIntegral

GetIntMatrix

GetL2Terms

GetLz2QNDiag

GetMPO

GetMPOSites

GetNeSQNDiag

GetNeQNDiag

GetPairingMod

GetPairingObs

GetPairIntTerms

GetParityQNOffd

GetPinOrbQNDiag

GetPointValue

GetPolTerms

GetQNDiags

GetRotyQNOffd

GetSites

GetTorusDenIntTerms

GetTorusIntMatrix

GetTorusLz2QNDiag

GetTorusPairIntTerms

GetZnfChargeQNDiag

Laplacian

FuzzifiED.Libpath

Fuzzifino.Libpathino

Matrix

NormalOrder

FuzzifiED.NumThreads

FuzzifiED.ObsNormRadSq

FuzzifiED.OpenHelp!

Operator

OpMat

OpSum

ParticleHole

QNDiag

QNOffd

SBasis

SConfs

FuzzifiED.SilentStd

SimplifyTerms

SOperator

SparseMatrixCSC

SphereObs

SQNDiag

SQNOffd

StateDecompMat

STerm

STerms

StoreComps

```
StoreComps!                          amd * amd

STransf                              obs * obs

SweepOne                             qnf * qnf

Term                                 sqnf * sqnf

Terms                                tms * tms

Transf                               stms * stms

TruncateQNDiag                       mat * vec

                                     op * vec

                                     sop * vec

**Operations**                       trs * vec

amd + amd[24]                        strs * vec

obs + obs                            vec' * mat * vec

qnd + qnd                            vec' * op * vec

sqnd + sqnd                          vec' * sop * vec

tms + tms                            amd'

stms + stms                          obs'

# * amd                              tms'

# * obs                              stms'

# * qnd                              tms ^ #

# * sqnd                             stms ^ #

# * tms

# * stms
```

---

[24]Here each symbol represents a specific type : `amd` is a `AngModes`, `obs` is a `SphereObs`, `op` is a `Operator`, `sop` is a `SOperator`, `mat` is a `OpMat`, `qnd` is a `QNDiag`, `sqnd` is a `SQNDiag`, `qnf` is a `QNOffd`, `sqnf` is a `SQNOffd`, `trs` is a `Transf`, `strs` is a `STransf`, `vec` is a `Vector`, `#` is a number. In general, `a1 - a2` is supported where `a1 + a2` is supported ; `a * #`, `a / #` and `-a` are supported where `# * a` is supported.

– 76 –

References

[1] W. Zhu, C. Han, E. Huffman, J.S. Hofmann and Y.-C. He, *Uncovering conformal symmetry in the 3d Ising transition: State-operator correspondence from a quantum fuzzy sphere regularization*, *Phys. Rev. X* **13** (2023) 021009 [arXiv:2210.13482].

[2] L. Hu, Y.-C. He and W. Zhu, *Operator product expansion coefficients of the 3d Ising criticality via quantum fuzzy spheres*, *Phys. Rev. Lett.* **131** (2023) 031601 [arXiv:2303.08844].

[3] C. Han, L. Hu, W. Zhu and Y.-C. He, *Conformal four-point correlators of the three-dimensional Ising transition via the quantum fuzzy sphere*, *Phys. Rev. B* **108** (2023) 235123 [arXiv:2306.04681].

[4] Z. Zhou, L. Hu, W. Zhu and Y.-C. He, SO(5) *deconfined phase transition under the fuzzy-sphere microscope: Approximate conformal symmetry, pseudo-criticality, and operator spectrum*, *Phys. Rev. X* **14** (2024) 021044 [arXiv:2306.16435].

[5] B.-X. Lao and S. Rychkov, *3d Ising CFT and exact diagonalization on icosahedron: The power of conformal perturbation theory*, *SciPost Phys.* **15** (2023) 243 [arXiv:2307.02540].

[6] L. Hu, Y.-C. He and W. Zhu, *Solving conformal defects in 3d conformal field theory using fuzzy sphere regularization*, *Nature Commun.* **15** (2024) 9013 [arXiv:2308.01903].

[7] J.S. Hofmann, F. Goth, W. Zhu, Y.-C. He and E. Huffman, *Quantum Monte Carlo simulation of the 3d Ising transition on the fuzzy sphere*, *SciPost Phys. Core* **7** (2024) 028 [arXiv:2310.19880].

[8] C. Han, L. Hu and W. Zhu, *Conformal operator content of the Wilson-Fisher transition on fuzzy sphere bilayers*, *Phys. Rev. B* **110** (2024) 115113 [arXiv:2312.04047].

[9] Z. Zhou, D. Gaiotto, Y.-C. He and Y. Zou, *The g-function and defect changing operators from wavefunction overlap on a fuzzy sphere*, *SciPost Phys.* **17** (2024) 021 [arXiv:2401.00039].

[10] L. Hu, W. Zhu and Y.-C. He, *Entropic F-function of 3d Ising conformal field theory via the fuzzy sphere regularization*, arXiv:2401.17362.

[11] G. Cuomo, Y.-C. He and Z. Komargodski, *Impurities with a cusp: general theory and 3d Ising*, *JHEP* **11** (2024) 061 [arXiv:2406.10186].

[12] Z. Zhou and Y. Zou, *Studying the 3d Ising surface CFTs on the fuzzy sphere*, *SciPost Phys.* **18** (2025) 031 [arXiv:2407.15914].

[13] M. Dedushenko, *Ising BCFT from fuzzy hemisphere*, arXiv:2407.15948.

[14] G. Fardelli, A.L. Fitzpatrick and E. Katz, *Constructing the infrared conformal generators on the fuzzy sphere*, arXiv:2409.02998.

[15] R. Fan, *Note on explicit construction of conformal generators on the fuzzy sphere*, arXiv:2409.08257.

[16] Z. Zhou and Y.-C. He, *A new series of 3d CFTs with* Sp(N) *global symmetry on fuzzy sphere*, arXiv:2410.00087.

[17] C. Voinea, R. Fan, N. Regnault and Z. Papić, *Regularizing 3d conformal field theories via anyons on the fuzzy sphere*, arXiv:2411.15299.

[18] S. Yang, Y.-G. Yue, Y. Tang, C. Han, W. Zhu and Y. Chen, *Microscopic study of 3d Potts phase transition via fuzzy sphere regularization*, arXiv:2501.14320.

[19] C. Han and W. Zhu, *Quantum phase transitions on the noncommutative circle*, *Phys. Rev. B* **111** (2025) 085113.

[20] A.M. Läuchli, L. Herviou, P.H. Wilhelm and S. Rychkov, *Exact diagonalization, matrix product states and conformal perturbation theory study of a 3d Ising fuzzy sphere model*, arXiv:2504.00842.

[21] M. Fishman, S.R. White and E.M. Stoudenmire, *The ITensor software library for tensor network calculations*, *SciPost Phys. Codeb.* (2022) 4 [arXiv:2007.14822].

[22] J. Bezanson, A. Edelman, S. Karpinski and V.B. Shah, *Julia: A fresh approach to numerical computing*, *SIAM Review* **59** (2017) 65.

[23] S. Rychkov, *EPFL Lectures on Conformal Field Theory in D ≥ 3 Dimensions*, Briefs in Physics, Springer (1, 2016), 10.1007/978-3-319-43626-5, [arXiv:1601.05000].

[24] D. Simmons-Duffin, *The conformal bootstrap*, in *Theoretical Advanced Study Institute in Elementary Particle Physics: New Frontiers in Fields and Strings*, pp. 1–74, 2017, DOI [arXiv:1602.07982].

[25] A.M. Polyakov, *Conformal symmetry of critical fluctuations*, *JETP Lett.* **12** (1970) 381.

[26] J.L. Cardy, *Scaling and Renormalization in Statistical Physics*, Cambridge Lecture Notes in Physics, Cambridge University Press (1996).

[27] S. Sachdev, *Quantum Phase Transitions*, Cambridge University Press, 2 ed. (2011), 10.1017/CBO9780511973765.

[28] J. Polchinski, *String theory. Vol. 1: An introduction to the bosonic string*, Cambridge Monographs on Mathematical Physics, Cambridge University Press (12, 2007), 10.1017/CBO9780511816079.

[29] J.M. Maldacena, *The large-N limit of superconformal field theories and supergravity*, *Adv. Theor. Math. Phys.* **2** (1998) 231.

[30] A.B. Zamolodchikov, *Irreversibility of the flux of the renormalization group in a 2D field theory*, *JETP Lett.* **43** (1986) 730.

[31] P. Di Francesco, P. Mathieu and D. Sénéchal, *Conformal field theory*, Graduate texts in contemporary physics, Springer, New York, NY (1997), 10.1007/978-1-4612-2256-9.

[32] P.H. Ginsparg, *Applied conformal field theory*, in *Les Houches Summer School in Theoretical Physics: Fields, Strings, Critical Phenomena*, 9, 1988 [arXiv:hep-th/9108028].

[33] A.A. Belavin, A.M. Polyakov and A.B. Zamolodchikov, *Infinite conformal symmetry in two-dimensional quantum field theory*, *Nucl. Phys. B* **241** (1984) 333.

[34] J. Wess and B. Zumino, *Consequences of anomalous Ward identities*, *Phys. Lett. B* **37** (1971) 95.

[35] E. Witten, *Nonabelian bosonization in two-dimensions*, *Commun. Math. Phys.* **92** (1984) 455.

[36] D. Poland, S. Rychkov and A. Vichi, *The conformal bootstrap: Theory, numerical techniques, and applications*, *Rev. Mod. Phys.* **91** (2019) 015002 [arXiv:1805.04405].

[37] S. Rychkov and N. Su, *New developments in the numerical conformal bootstrap*, *Rev. Mod. Phys.* **96** (2024) 045004 [arXiv:2311.15844].

[38] S. El-Showk, M.F. Paulos, D. Poland, S. Rychkov, D. Simmons-Duffin and A. Vichi, *Solving the 3d Ising model with the conformal bootstrap*, *Phys. Rev. D* **86** (2012) 025022 [arXiv:1203.6064].

[39] F. Kos, D. Poland, D. Simmons-Duffin and A. Vichi, *Precision islands in the Ising and* O(N) *models*, *JHEP* **08** (2016) 036 [arXiv:1603.04436].

[40] F. Kos, D. Poland, D. Simmons-Duffin and A. Vichi, *Bootstrapping the* O(N) *archipelago*, *JHEP* **11** (2015) 106 [arXiv:1504.07997].

[41] L. Iliesiu, F. Kos, D. Poland, S.S. Pufu, D. Simmons-Duffin and R. Yacoby, *Bootstrapping 3d fermions*, *JHEP* **03** (2016) 120 [arXiv:1508.00012].

[42] A.M. Ferrenberg, J. Xu and D.P. Landau, *Pushing the limits of Monte Carlo simulations for the three-dimensional Ising model*, *Phys. Rev. E* **97** (2018) 043301 [arXiv:1806.03558].

[43] A.W. Sandvik, *Computational studies of quantum spin systems*, *AIP Conf. Proc.* **1297** (2010) 135 [arXiv:1101.3281].

[44] M.E. Fisher and M.N. Barber, *Scaling theory for finite-size effects in the critical region*, *Phys. Rev. Lett.* **28** (1972) 1516.

[45] D. Pappadopulo, S. Rychkov, J. Espin and R. Rattazzi, *OPE convergence in conformal field theory*, *Phys. Rev. D* **86** (2012) 105043 [arXiv:1208.6449].

[46] R.C. Brower and E.K. Owen, *The Ising model on* $\mathbb{S}^2$, arXiv:2407.00459.

[47] J. Madore, *The fuzzy sphere*, *Class. Quant. Grav.* **9** (1992) 69.

[48] F.D.M. Haldane, *Fractional quantization of the Hall effect: A hierarchy of incompressible quantum fluid states*, *Phys. Rev. Lett.* **51** (1983) 605.

[49] T.T. Wu and C.N. Yang, *Dirac monopole without strings: Monopole harmonics*, *Nucl. Phys. B* **107** (1976) 365.

[50] M. Greiter, *Landau level quantization on the sphere*, *Phys. Rev. B* **83** (2011) 115129 [arXiv:1101.3943].

[51] K. Hasebe, *Hopf maps, lowest Landau level, and fuzzy spheres*, *SIGMA* **6** (2010) 071 [arXiv:1009.1192].

[52] V. Pasquier, *Skyrmions in the quantum Hall effect and noncommutative solitons*, *Phys. Lett. B* **490** (2000) 258 [arXiv:hep-th/0007176].

[53] S.M. Girvin, *Spin and isospin: Exotic order in quantum Hall ferromagnets*, in *The multifaceted skyrmion*, G.E. Brown and M. Rho, eds., pp. 217–231 (2010), DOI.

[54] R.C. Myers and A. Sinha, *Seeing a c-theorem with holography*, *Phys. Rev. D* **82** (2010) 046006 [arXiv:1006.1263].

[55] H. Casini, M. Huerta and R.C. Myers, *Towards a derivation of holographic entanglement entropy*, *JHEP* **05** (2011) 036 [arXiv:1102.0440].

[56] D.L. Jafferis, I.R. Klebanov, S.S. Pufu and B.R. Safdi, *Towards the F-theorem:* $\mathcal{N} = 2$ *field theories on the three-sphere*, *JHEP* **06** (2011) 102 [arXiv:1103.1181].

[57] I.R. Klebanov, S.S. Pufu and B.R. Safdi, *F-theorem without supersymmetry*, *JHEP* **10** (2011) 038 [arXiv:1105.4598].

[58] H. Casini and M. Huerta, *On the RG running of the entanglement entropy of a circle*, *Phys. Rev. D* **85** (2012) 125016 [arXiv:1202.5650].

[59] A. Sterdyniak, A. Chandran, N. Regnault, B.A. Bernevig and P. Bonderson, *Real-space entanglement spectrum of quantum Hall states*, *Phys. Rev. B* **85** (2012) 125308 [arXiv:1110.2810].

[60] J. Dubail, N. Read and E.H. Rezayi, *Real-space entanglement spectrum of quantum Hall systems*, *Phys. Rev. B* **85** (2012) 115321 [arXiv:1111.2811].

[61] M.P. Zaletel and R.S.K. Mong, *Exact matrix product states for quantum Hall wave functions*, *Phys. Rev. B* **86** (2012) 245305 [arXiv:1208.4862].

[62] I.D. Rodriguez, S.H. Simon and J.K. Slingerland, *Evaluation of ranks of real space and particle entanglement spectra for large systems*, *Phys. Rev. Lett.* **108** (2012) 256806 [arXiv:1111.3634].

[63] T. Senthil, A. Vishwanath, L. Balents, S. Sachdev and M.P.A. Fisher, *Deconfined quantum critical points*, *Science* **303** (2004) 1490 [arXiv:cond-mat/0311326].

[64] T. Senthil, L. Balents, S. Sachdev, A. Vishwanath and M.P.A. Fisher, *Quantum criticality beyond the Landau-Ginzburg-Wilson paradigm*, *Phys. Rev. B* **70** (2004) 144407 [arXiv:cond-mat/0312617].

[65] T. Senthil, *Deconfined quantum critical points: a review*, arXiv:2306.12638.

[66] A. Nahum, P. Serna, J.T. Chalker, M. Ortuño and A.M. Somoza, *Emergent* SO(5) *symmetry at the Néel to valence-bond-solid transition*, *Phys. Rev. Lett.* **115** (2015) 267203 [arXiv:1508.06668].

[67] C. Wang, A. Nahum, M.A. Metlitski, C. Xu and T. Senthil, *Deconfined quantum critical points: symmetries and dualities*, *Phys. Rev. X* **7** (2017) 031051 [arXiv:1703.02426].

[68] V. Gorbenko, S. Rychkov and B. Zan, *Walking, weak first-order transitions, and complex CFTs*, *JHEP* **10** (2018) 108 [arXiv:1807.11512].

[69] J. Lee and S. Sachdev, *Wess-Zumino-Witten terms in graphene Landau levels*, *Phys. Rev. Lett.* **114** (2015) 226801 [arXiv:1411.5684].

[70] M. Ippoliti, R.S.K. Mong, F.F. Assaad and M.P. Zaletel, *Half-filled Landau levels: A*

continuum and sign-free regularization for three-dimensional quantum critical points, *Phys. Rev. B* **98** (2018) 235108 [arXiv:1810.00009].

[71] Z. Wang, M.P. Zaletel, R.S.K. Mong and F.F. Assaad, *Phases of the* $(2+1)$ *dimensional* SO(5) *nonlinear sigma model with topological term*, *Phys. Rev. Lett.* **126** (2021) 045701 [arXiv:2003.08368].

[72] B.-B. Chen, X. Zhang, Y. Wang, K. Sun and Z.Y. Meng, *Phases of* $(2+1)d$ SO(5) *nonlinear sigma model with a topological term on a sphere: Multicritical point and disorder phase*, *Phys. Rev. Lett.* **132** (2024) 246503 [arXiv:2307.05307].

[73] B.-B. Chen, X. Zhang and Z. Yang Meng, *Emergent conformal symmetry at the multicritical point of* $(2+1)d$ SO(5) *model with Wess-Zumino-Witten term on a sphere*, *Phys. Rev. B* **110** (2024) 125153 [arXiv:2405.04470].

[74] K.G. Wilson and M.E. Fisher, *Critical exponents in 3.99 dimensions*, *Phys. Rev. Lett.* **28** (1972) 240.

[75] Z. Komargodski and N. Seiberg, *A symmetry breaking scenario for QCD₃*, *JHEP* **01** (2018) 109 [arXiv:1706.08755].

[76] V.S. Dotsenko, *Critical Behavior and Associated Conformal Algebra of the Z(3) Potts Model*, *Nucl. Phys. B* **235** (1984) 54.

[77] Y. Tang, H. Ma, Q. Tang, Y.-C. He and W. Zhu, *Reclaiming the lost conformality in a non-Hermitian quantum 5-state Potts model*, *Phys. Rev. Lett.* **133** (2024) 076504 [arXiv:2403.00852].

[78] G. Barkema and J. de Boer, *Numerical study of phase transitions in potts models*, *Phys. Rev. A* **44** (1991) 8000.

[79] S.M. Chester and N. Su, *Upper critical dimension of the 3-state Potts model*, arXiv:2210.09091.

[80] M. Billó, M. Caselle, D. Gaiotto, F. Gliozzi, M. Meineri and R. Pellegrini, *Line defects in the 3d Ising model*, *JHEP* **07** (2013) 055 [arXiv:1304.4110].

[81] M. Billò, V. Gonçalves, E. Lauria and M. Meineri, *Defects in conformal field theory*, *JHEP* **04** (2016) 091 [arXiv:1601.02883].

[82] A. Hanke, *Critical adsorption on defects in ising magnets and binary alloys*, *Phys. Rev. Lett.* **84** (2000) 2180.

[83] A. Allais, *Magnetic defect line in a critical Ising bath*, arXiv:1412.3449.

[84] A. Allais and S. Sachdev, *Spectral function of a localized fermion coupled to the Wilson-Fisher conformal field theory*, *Phys. Rev. B* **90** (2014) 035131 [arXiv:1406.3022].

[85] W.H. Pannell and A. Stergiou, *Line defect RG flows in the ε expansion*, *JHEP* **06** (2023) 186 [arXiv:2302.14069].

[86] G. Cuomo, Z. Komargodski and A. Raviv-Moshe, *Renormalization group flows on line defects*, *Phys. Rev. Lett.* **128** (2022) 021603 [arXiv:2108.01117].

[87] H. Casini, I. Salazar Landea and G. Torroba, *Entropic g theorem in general spacetime dimensions*, *Phys. Rev. Lett.* **130** (2023) 111603 [arXiv:2212.10575].

[88] I. Affleck and A.W.W. Ludwig, *The fermi edge singularity and boundary condition changing operators*, *J. Phys. A* **27** (1994) 5375 [arXiv:cond-mat/9405057].

[89] I. Affleck, *Boundary condition changing operators in conformal field theory and condensed matter physics*, *Nucl. Phys. B Proc. Suppl.* **58** (1997) 35 [arXiv:hep-th/9611064].

[90] M.A. Metlitski, *Boundary criticality of the O(N) model in d = 3 critically revisited*, *SciPost Phys.* **12** (2022) 131 [arXiv:2009.05119].

[91] A. Krishnan and M.A. Metlitski, *A plane defect in the 3d O(N) model*, *SciPost Phys.* **15** (2023) 090 [arXiv:2301.05728].

[92] S. Giombi and B. Liu, *Notes on a surface defect in the O(N) model*, *JHEP* **12** (2023) 004 [arXiv:2305.11402].

[93] R.B. Laughlin, *Anomalous quantum Hall effect: An incompressible quantum fluid with fractionally charged excitations*, *Phys. Rev. Lett.* **50** (1983) 1395.

[94] K. Hasebe, SO(5) *Landau models and nested Nambu matrix geometry*, *Nucl. Phys. B* **956** (2020) 115012 [arXiv:2002.05010].

[95] T.T. Wu and C.N. Yang, *Some properties of monopole harmonics*, *Phys. Rev. D* **16** (1977) 1018.

[96] Y.M. Shnir, *Magnetic Monopoles*, Text and Monographs in Physics, Springer, Berlin/Heidelberg (2005), 10.1007/3-540-29082-6.

[97] L.C. Biedenharn and J.D. Louck, *Angular Momentum in Quantum Physics: Theory and Application*, Encyclopedia of Mathematics and its Applications, Cambridge University Press (1984).

[98] S.A. Trugman and S. Kivelson, *Exact results for the fractional quantum hall effect with general interactions*, *Phys. Rev. B* **31** (1985) 5280.

[99] D. Yoshioka, B.I. Halperin and P.A. Lee, *Ground state of two-dimensional electrons in strong magnetic fields and $\frac{1}{3}$ quantized Hall effect*, *Phys. Rev. Lett.* **50** (1983) 1219.

[100] F.D.M. Haldane, *Many-particle translational symmetries of two-dimensional electrons at rational Landau-level filling*, *Phys. Rev. Lett.* **55** (1985) 2095.

[101] F.D.M. Haldane, *The hierarchy of fractional states and numerical studies*, in *The Quantum Hall Effect*, R.E. Prange and S.M. Girvin, eds., (New York, NY), pp. 303–352, Springer US (1987), DOI.

[102] Y. Zou, A. Milsted and G. Vidal, *Conformal fields and operator product expansion in critical quantum spin chains*, *Phys. Rev. Lett.* **124** (2020) 040604 [arXiv:1901.06439].

[103] W.E. Arnoldi, *The principle of minimized iterations in the solution of the matrix eigenvalue problem*, *Quart. Appl. Math.* **9** (1951) 17.

[104] R.B. Lehoucq, D.C. Sorensen and C. Yang, *ARPACK Users' Guide*, Society for Industrial and Applied Mathematics (1998), 10.1137/1.9780898719628.

[105] S.R. White, *Density matrix formulation for quantum renormalization groups*, *Phys. Rev. Lett.* **69** (1992) 2863.

[106] U. Schollwock, *The density-matrix renormalization group*, *Rev. Mod. Phys.* **77** (2005) 259 [arXiv:cond-mat/0409292].

[107] U. Schollwoeck, *The density-matrix renormalization group in the age of matrix product states*, *Annals Phys.* **326** (2011) 96 [arXiv:1008.3477].