ALMA MATER STUDIORUM · UNIVERSITY OF BOLOGNA

School of Science
Department of Physics and Astronomy
Master Degree in Physics

Enhancing Diagrammatic Monte Carlo via machine learning

Supervisor:

Prof. Cesare Franchini

Submitted by:

Luca Leoni

Academic Year 2022/2023

Abstract

Since their introduction in 1949 Feynman's diagrams have proven over time to be the most precise and intuitive way of approaching quantum field theory, quantum statistical mechanics, and many-body physics. Feynman's diagrams approach is used in many physical problems, as they are able to simplify complex formalism and provide efficient tools for numerical simulations. The Diagrammatic Monte Carlo (DMC) technique is one such computational methods, which stands tall among the most precise approximation-free Markov Chain integration methods. Still, As all Monte Carlo approaches, the main limitation of DMC is the huge computational cost. Thus, in this thesis work, we aimed to reduce the computational time by proposing new ways of constructing the diagrams Markov Chain to reduce correlation with respect to today's standard approaches. This study has led us to the creation of two new proposals: an analytical approach that grants the minimum correlation possible in the Markov Chain, and a more general neural network protocol based on the Normalizing Flow architecture. Both methods have been tested on different models showing effectiveness in reducing the correlation, and so the number of samples needed for convergence, giving a boost in performances if used in the proper context.

The thesis is structured in 4 chapters, starting in Ch. (1) with a comprehensive construction of both the model Hamiltonians used as tests and the Feynman's diagrams formalism used in DMC. DMC is the main focus of Ch. (2), where the theoretical statistical foundations are presented along with the description of the standard algorithm for the wanted models. Ch. (3) is focused on the description of the Machine learning model used in this work, introducing the most commonly used architectures to create Normalizing Flows and how they can be adapted to DMC. At last, Ch. (4) presents the two new chain construction methods and the results from their application. Results obtained from computations implemented inside a C++ framework designed and coded during the thesis work, named LLDMC, which is also presented in the beginning of the chapter.

Sommario

Dalla loro introduzione nel 1949 i diagrammi di Feynman hanno costituito il metodo più preciso ed intuitivo per approcciare lo studio di varie aree della fisica come: teoria quantistica dei campi, meccanica statistica quantistica e la fisica dei molti corpi. Il loro vasto spettro di applicazioni ha generato nel tempo un interesse verso la creazione di routine numeriche capaci di effettuare stime delle complicate equazioni matematiche che risultano dall'applicazione di tale formalismo. Il metodo Diagrammatic Monte Carlo (DMC) è una di tali tecniche computazionali, facendo parte delle più precise tra quest'ultime in quanto metodo di integrazione approximation-free basato sulle Markov Chain. DMC è quindi limitato solamente dai soliti problemi comuni ad ogni metodo Monte Carlo, e quindi relativi principalmente all'elevato tempo necessario al risultato per convergere. In questo lavoro di tesi si è quindi cercato un modo per velocizzare la convergenza di tale algoritmo proponendo due nuovi metodi per costruire la Markov Chain di diagrammi atti a ridurre la loro correlazione rispetto ai metodi standard usati oggi. Lo studio della statistica dei diagrammi ha portato a proporre due metodi in particolare: uno analitico che, anche se limitato nelle applicazioni, è in grado di ridurre al minimo la correlazione, e un secondo metodo più generale basato sulle reti neurali e in particolare sull'architettura del Normalizing Flow. Entrambi i metodi sono stati testati su modelli differenti che ne hanno mostrato l'effettiva efficacia nel ridurre la correlazione nella catena, e quindi il numero di diagrammi necessario alla convergenza, portando un miglioramento delle performance se usati nel giusto contesto.

La tesi è strutturata in quattro capitoli. Il capitolo (1) contiene un introduzione teorica relativa alle Hamiltoniane dei modelli usati per effettuare i test, e sul formalismo dei Diagrammi di Feynman usato all'interno del metodo DMC. Quest'ultimo viene quindi affrontato più nel dettaglio nel capitolo (2), dove le fondamenta teoriche relative alle proprietà statistiche dell'algoritmo sono riportate insieme alla descrizione dell'algoritmo standard che il nostro lavoro ha cercato di superare. Il capitolo (3), invece, si focalizza sulla descrizione del modello di Machine Learning usato all'interno del lavoro, introducendo la principale letteratura riguardante i Normalizing Flow e come quest'ultima possa essere modificata allo scopo di essere usata dentro DMC. Infine, il capitolo (4) è usato per presentare i due nuovi metodi nella loro interezza mostrando anche i rispettivi risultati ottenuti nelle loro applicazioni. I quali sono stati ottenuti da conti implementati attraverso un framework C++ pensato e scritto apposta per questo lavoro di tesi, e chiamato LLDMC, che è presentato all'inizio del capitolo.

Contents

1

Model Hamiltonians for condensed matter

The theoretical modelling of materials is based on the study of matter as a quantum many-body system composed by numerous electrons and nuclei interacting together. *First principles* or *ab-initio* methods are used to solve numerically the *Schröedinger equation* (SE) associated to an entangled many-body Hamiltonian. Techniques have evolved to recent years in possessing a strong predictive power that, along with the increase in computational resources, brought computational material physics to become a wide research area and a standard practice in industries. Nevertheless, their applications are still limited from a series of approximations used to simplify certain interactions to disentangle the problem, and making it computationally trackable. This creates the needs of a framework that can be used when the standard methods fail to describe the right behavior. Therefore, a brief review of the mains first principle SE methods is presented focussing on their limitations. An alternative method to solve the many-body interacting problem is based on model Hamiltonian, that can precisely account for specific interactions, and allow for corrections of the first principles results, or to access new systems'properties. In the framework of DMC methods this is achieved by employing Feynman diagrams, which will be briefly introduced.

1.1 First principles methods in Many Body

We assume that matter is composed by a series of non-relativistic electrons, N_e, and nuclei, N_n, that are interacting through electrostatic forces. In this way, is possible to write down a total Hamiltonian that describe a generic material as

$$\widehat{\mathscr{H}} = \sum_{\alpha}^{N_n} \frac{\widehat{\mathbf{P}}_\alpha}{2M_\alpha} + \sum_{i}^{N_e} \frac{\widehat{\mathbf{p}}_i}{2m} + U_{nn}\left(\widehat{\mathbf{R}}\right) + U_{en}\left(\widehat{\mathbf{R}}, \hat{\mathbf{r}}\right) + U_{ee}\left(\hat{\mathbf{r}}\right) \tag{1.1}$$

Where capital letters correspond to the nuclei properties while lower cases one to the electrons, also taking into account how the masses of the former, M_α, can be different from one another. The terms U_{nn}, U_{ne} and U_{ee}, instead, describes the interaction poten-

tial between the components

$$U_{nn} = \frac{1}{2}\sum_{\alpha\neq\beta}^{N_n} V\left(\hat{\mathbf{R}}_\alpha - \hat{\mathbf{R}}_\beta\right), \qquad U_{en} = -\frac{1}{2}\sum_{i}^{N_e}\sum_{\alpha}^{N_n} V\left(\hat{\mathbf{R}}_\alpha - \hat{\mathbf{r}}_i\right), \qquad U_{ee} = \frac{1}{2}\sum_{i\neq j}^{N_e} V\left(\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_i\right), \quad (1.2)$$

with $V(\mathbf{r})$ that is usually taken as the Coulomb potential Q/r, but also screened versions are used for certain cases. The goal of first principles ab-initio methods is to numerical solve the many-body SE associated to Eq. (1.1), and describe the state of the system. In order to accomplish this a first common approximation is used to separate the electronic and nuclei degrees of freedoms called *Born-Oppenheimer approximation*. The idea is to assume that, due to mass difference, the kinetic energy of electrons is much larger than the nuclei one which can, therefore, be neglected to describe materials properties. In this way, the nuclei are assumed to be frozen in space to certain positions becoming parameters inside the simplified SE

$$\left[\sum_{i}^{N_e}\frac{\hat{\mathbf{p}}_i}{2m} + U_{nn}(\mathbf{R}) + U_{en}(\mathbf{R},\hat{\mathbf{r}}) + U_{ee}(\hat{\mathbf{r}})\right]|\Psi(\mathbf{r},\{\mathbf{R}\})\rangle = E_e(\mathbf{R})\,|\Psi(\mathbf{r},\{\mathbf{R}\})\rangle\,. \quad (1.3)$$

The electronic energy obtained, E_e, as a function of the nuclei coordinates is the final goal of the vast majority of the ab-initio methods for material modelling. In fact, further contributions arising from the quantum treatment of the nuclei motion or *phonons* are usually discarded, even if modifications of the Coulomb interaction due to nuclei motion can be adopted to improve results [17]. At this point Eq. (1.3) still poses a challenge due to the entanglement between electrons generated by U_{ee}. Bringing to the second needed simplification that modify the electron-electron interaction in a way that allows for the $\mathbf{r}$ to be disentangled. Such a task is usually performed by rewriting the total wave function as a combination of single particle orbitals

$$\Psi(\mathbf{r}) = F\left[\psi_1(\mathbf{r_1}),\psi_2(\mathbf{r_2}),\ldots,\psi_{N_e}(\mathbf{r}_{N_e})\right], \quad (1.4)$$

which is done differently depending on the method in question. For example, in *density functional theory* (DFT) the ground state is described in terms of the electron density $n_0(\mathbf{r}) = \sum_i^{N_e}\left|\psi_i(\mathbf{r})\right|^2$ [20, 27], while in the context of *Hartree-Fock* (HF) the function F becomes the Slater determinant [47]. Then, the variational principle is used in order to minimize the energy associated to the selected form, obtaining a set of Schrödinger like equations defining the functions that better approximate the real ground state

$$\left[-\frac{\hbar^2}{2m}\nabla^2 + U_{nn}(\mathbf{R}) + U_{en}(\mathbf{R},\mathbf{r}) + U_H(\mathbf{r}) + U_{XC}(\mathbf{r})\right]\psi_i(\mathbf{r}) = \varepsilon_i(\mathbf{R})\psi_i(\mathbf{r}). \quad (1.5)$$

This is exactly the form obtained inside the DFT formalism, but can be seen how the general structure remains the same also in the HF one. In particular, one can notice how the total U_{ee} interaction has been split in two parts. The first is called Hartree potential, U_H, describing the interaction of the single electron with the cloud composed by the others, forming kind of a zeroth order approximation for the interaction that is analogous to all methods. The *exchange and correlation* potential, U_{XC}, instead accounts for higher order effects of the electron-electron contribution and posses different forms depending on the method we are using. In DFT, for example, a function of the local density

and of its gradient is often used [42, 43], while for HF the exchange integral of the electron orbitals is present. Of course, none of them results in being an exact rewriting of the interaction, but only an approximation that is able to account for: exchange, in the case of HF, or approximated forms of exchange and correlation, DFT. Still, both are able to give us good approximations and allows Eq. (1.5) to be solved using a self-consistent scheme that can estimate both $\{\psi_i\}$ and ε_i from which material's ground state properties are predicted.

From such description is possible to understand that several limitations to both the applicability and the precision of the methods are intrinsically present. The first one is that the number of equations to solve increase with the number of electrons in the system. Meaning that the effective sizes of the system under study are limited by our computational resources. Therefore, ab-initio methods can only study materials with *limited size unit cells*, for larger cells only much less precise methods are available. Making the study of defects or doped materials a challenge. Then, if the system under investigation posses a strong electron-phonon coupling or electron correlation the approximations used fails to predicting the right material behaviors. Bringing to the need of using heavier numerical approaches based on more complex theories that allows for better approximations. Methods such as *coupled cluster* [29] or *GW method* [4] in order to renormalize the energies and account for electron correlation, and *density functional perturbation theory* (DFPT) [5] to take into consideration the *electron-phonon interaction*. All powerful approaches that are able to give us precise predictions, but their cost in computational time limits their applicability to small systems. Lastly, all plain first principles methods previously described are constructed on top of the variational principle, which *can only describe the ground state* of a system. Meaning that the results are exact only at 0 K not giving information about the exited states appearing at higher temperatures that becomes important in phase transitions phenomena.

All of those three weaknesses can be overcome by approaching the study of the many body systems through the use of an effective Hamiltonian that explicitly quantize the missing interactions in which we are interested. In particular, we want to focus our attention on how we can account for the nuclei effects in the contest of electron-phonon coupling and in the interaction with a two level system. Showing how we can use this framework to evaluate correction to the energies arising from the coupling of electronic and nuclei motion, and how partition function studies allows access to thermodynamics properties at wanted temperatures.

1.2 Electron-Phonon coupling

It is known from experiments how atoms in solids positions themselves in order to generate highly symmetric configurations known as lattices. For these reason the nuclei are usually thought as still in the positions that defines the lattice, $\{\mathbf{R}^0_\alpha\}$, while the material's properties are described by the energy of the electronic motions. In reality thermal and quantum effects makes so that the nuclei are never really still but constantly vibrates respect to an equilibrium position given by $\mathbf{R}^0$. This implies that to describe the real position of every nucleus, $\mathbf{R}_\alpha$, we need to decompose them into the equilibrium, lattice, position and a displacement $\mathbf{Q}_\alpha$. Where it's important to notice how the collection of $\mathbf{R}^0$ is a constant inside our system that describes the lattice, so that the real nuclear

degrees of freedoms are now represented by $\mathbf{Q}$. Now, we can insert such decomposition inside the expression of the electron-nuclei interaction in Eq. (1.2) and expand through Taylor expansion to obtain

$$U_{en}(\hat{\mathbf{R}},\hat{\mathbf{r}}) = \sum_{i\alpha} V(\mathbf{R}^0_\alpha + \hat{\mathbf{Q}}_\alpha - \hat{\mathbf{r}}_i) = \sum_{i\alpha} V(\mathbf{R}^0_\alpha - \hat{\mathbf{r}}_i) + \sum_{i\alpha} \hat{\mathbf{Q}}_\alpha \cdot \nabla V(\mathbf{R}^0_\alpha - \hat{\mathbf{r}}_i) + \mathcal{O}(Q^2). \tag{1.6}$$

From Eq. (1.6) we can understand that neglecting the nuclear degrees of freedom in the Born-Oppenheimer approximation is like counting only the zeroth order term of the expansion. Even though such truncation can be ideal when small displacements are present there exists situations where such interactions play an important role, such as polarizable materials or superconductors. Therefore, we want to formally quantize the nuclei motions in order to account for their effects on the electron properties by introducing a second quantized effective Hamiltonian.

Quantization of nuclei motion

The motion of the nuclei can be described as a series of particles inside a binding potential given by $E_e(\mathbf{R})$, which has a minimum corresponding to the value of $\mathbf{R}^0$. That allow us to write down the Hamiltonian describing the nuclei degrees of freedom in the following way

$$\widehat{\mathscr{H}_n} = \sum_{\alpha}^{N_n} \frac{\hat{\mathbf{P}}_\alpha}{2M_\alpha} + E_e(\mathbf{R}^0 + \hat{\mathbf{Q}}). \tag{1.7}$$

We can then expand the potential under study as a Taylor series around the minimum. So that by truncating the expansion at the second order and by not counting the constant energy shift given by zeroth order term we obtain

$$\widehat{\mathscr{H}_n} \approx \sum_{\alpha}^{N_n} \frac{\hat{\mathbf{P}}_\alpha}{2M} + \frac{1}{2}\sum_{\alpha\beta}^{N_n} \hat{\mathbf{Q}}_\alpha \mathbf{\Phi}_{\alpha\beta} \hat{\mathbf{Q}}_\beta, \qquad \mathbf{\Phi}_{\alpha\beta} = \left.\frac{\partial E_e}{\partial \mathbf{Q}_\alpha \mathbf{Q}_\beta}\right|_{\mathbf{Q}=0}, \tag{1.8}$$

where we have taken all nuclei masses to be the same for convenience. This is a system of coupled Harmonic oscillators that can be recast in a decoupled one by switching the problem to Fourier space. In particular, if we let the lattice be composed by $N = N_x N_y N_z$ unit cells we can apply the *Born-Von-Karman boundary conditions* to obtain an infinite lattice with periodicity of N_i cells in the i direction. Under such mathematical assumptions we can perform a discrete Fourier transform of the operators under study as

$$\hat{\mathbf{Q}}(\mathbf{R}) = \frac{1}{\sqrt{N}}\sum_{\mathbf{k}} \tilde{\mathbf{Q}}(\mathbf{k}) e^{i\mathbf{k}\cdot\mathbf{R}}, \qquad \tilde{\mathbf{Q}}(\mathbf{k}) = \frac{1}{\sqrt{N}}\sum_{\alpha} \hat{\mathbf{Q}}(\mathbf{R}_\alpha) e^{-i\mathbf{k}\cdot\mathbf{R}_\alpha}, \tag{1.9}$$

$$\hat{\mathbf{P}}(\mathbf{R}) = \frac{1}{\sqrt{N}}\sum_{\mathbf{k}} \tilde{\mathbf{P}}(\mathbf{k}) e^{i\mathbf{k}\cdot\mathbf{R}}, \qquad \tilde{\mathbf{P}}(\mathbf{k}) = \frac{1}{\sqrt{N}}\sum_{\alpha} \hat{\mathbf{P}}(\mathbf{R}_\alpha) e^{-i\mathbf{k}\cdot\mathbf{R}_\alpha}. \tag{1.10}$$

The values of $\mathbf{k}$ represents the N wave vectors of the first Brillouin zone. Since $\hat{\mathbf{P}}$ and $\hat{\mathbf{Q}}$ must be self-adjoint than the following identity holds

$$\tilde{\mathbf{Q}}^\dagger(\mathbf{k}) = \tilde{\mathbf{Q}}(-\mathbf{k}), \qquad \tilde{\mathbf{P}}^\dagger(\mathbf{k}) = \tilde{\mathbf{P}}(-\mathbf{k}). \tag{1.11}$$

Also, using the translational invariance of the material's lattice one can notice $\mathbf{\Phi}_{ij}$ will only depend on atomic distances, allowing for it to be written as function of type $\mathbf{\Phi}_{ij} =$

$\mathbf{\Phi}(\mathbf{R}_i - \mathbf{R}_j)$. Inserting these new expressions for the operators in Eq. (1.8) bring us the following form [16]

$$\widehat{\mathscr{H}_n} = \sum_\nu \sum_{\mathbf{k}} \left[\frac{\left|\tilde{\mathbf{P}}_\nu(\mathbf{k})\right|^2}{2M} + \frac{M}{2}\omega_\nu^2(\mathbf{k})\left|\tilde{\mathbf{Q}}_\nu(\mathbf{k})\right|^2 \right]. \tag{1.12}$$

This is a sum of independent oscillators described by the mode of oscillation, given by $\mathbf{k}$, and for every polarization direction, indexed by ν. Also, every one of them has its own characteristic frequency $\omega_\nu(\mathbf{k})$ that is obtained by solving the eigenvalue equation

$$M\omega_\nu(\mathbf{k})^2 \boldsymbol{\epsilon}_\nu(\mathbf{k}) = \mathbf{D}(\mathbf{k})\boldsymbol{\epsilon}_\nu(\mathbf{k}), \qquad \mathbf{D}(\mathbf{k}) = \sum_\alpha \mathbf{\Phi}(\mathbf{R}_\alpha) e^{i\mathbf{k}\cdot\mathbf{R}_\alpha}. \tag{1.13}$$

$\mathbf{D}$ is also called *dynamical matrix* and $\boldsymbol{\epsilon}_\nu(\mathbf{k})$ are its eigenvectors that defines the polarization directions of the different modes, so that in this context ν tells which element of the three space directions we are looking at.

From Eq. (1.12) it's possible to see how the nuclei motion can be decoupled in a series of collecting motions in k-space that contributes to the energy of the system with a form that is analogous to the one of the simple Harmonic oscillator. These oscillations may be treated as particles which can be created and destructed through the use of specific operators taking the forms

$$\hat{b}^\dagger_{\nu\mathbf{k}} = \boldsymbol{\epsilon}_\nu(\mathbf{k}) \cdot \left[-\frac{i}{\sqrt{2M\hbar\omega_\nu(\mathbf{k})}} \tilde{\mathbf{P}}_\nu(\mathbf{k}) + \sqrt{\frac{\omega_\nu(\mathbf{k})M}{2\hbar}} \tilde{\mathbf{Q}}_\nu(\mathbf{k}) \right], \tag{1.14}$$

$$\hat{b}_{\nu\mathbf{k}} = \boldsymbol{\epsilon}_\nu(\mathbf{k}) \cdot \left[\frac{i}{\sqrt{2M\hbar\omega_\nu(\mathbf{k})}} \tilde{\mathbf{P}}_\nu(-\mathbf{k}) + \sqrt{\frac{\omega_\nu(\mathbf{k})M}{2\hbar}} \tilde{\mathbf{Q}}_\nu(-\mathbf{k}) \right]. \tag{1.15}$$

It's easy to see how such creation and destruction operators satisfy the commutation relation $\left[\hat{b}_{\nu\mathbf{k}}, \hat{b}^\dagger_{\nu'\mathbf{k}'}\right] = \delta_{\nu\nu'}\delta_{\mathbf{k}\mathbf{k}'}$ meaning that atomic vibrations behaves like bosons that are commonly called *phonons*. By inverting Eq. (1.14) and Eq. (1.15) we can rewrite the Fourier components of momentum and displacement that can be inserted inside Eq. (1.12) to obtain the main form used in literature for the Hamiltonian describing atomic motion

$$\widehat{\mathscr{H}_n} = \sum_\nu \sum_{\mathbf{k}} \hbar\omega_\nu(\mathbf{k}) \left(\hat{b}^\dagger_{\nu\mathbf{k}} \hat{b}_{\nu\mathbf{k}} + \frac{1}{2} \right). \tag{1.16}$$

A term like this will always appear when we are dealing with the study of lattice interaction describing its motion that will be coupled to other quantities through other terms in the Hamiltonian, as we will see.

At last, it's easy to see how the form of $\tilde{\mathbf{Q}}$ and $\tilde{\mathbf{P}}$ in terms of phononic creation and destruction operators can be transformed back to obtain a form for the real space momentum and displacement operators

$$\widehat{\mathbf{P}}(\mathbf{R}) = -\frac{i}{\sqrt{N}} \sum_\nu \sum_{\mathbf{k}} \sqrt{\frac{\hbar M\omega_\nu(\mathbf{k})}{2}} \left[\hat{\mathbf{b}}_{\nu\mathbf{k}} + \hat{\mathbf{b}}^\dagger_{\nu-\mathbf{k}} \right] e^{i\mathbf{k}\cdot\mathbf{R}}, \tag{1.17}$$

$$\widehat{\mathbf{Q}}(\mathbf{R}) = \frac{1}{\sqrt{N}} \sum_\nu \sum_{\mathbf{k}} \sqrt{\frac{\hbar}{2M\omega_\nu(\mathbf{k})}} \left[\hat{\mathbf{b}}_{\nu\mathbf{k}} + \hat{\mathbf{b}}^\dagger_{\nu-\mathbf{k}} \right] e^{i\mathbf{k}\cdot\mathbf{R}}. \tag{1.18}$$

This result will be of particular importance in the next description of the electron-phonon coupling.

Holstein Hamiltonian

To create a completely quantized model of the lattice we want to use expansion Eq. (1.6) inside Eq. (1.1) and use the second quantization formalism to rewrite it with creation and annihilation operators of a certain base within the Hilbert space. The best choice that we can make is to use basis set of Block wave functions $\{|n\mathbf{k}\rangle\}$ that diagonalize the electronic degrees of freedom

$$\widehat{\mathscr{H}}_e = \sum_i^{N_e}\left[\frac{\hat{\mathbf{p}}_i}{2m} + \sum_\alpha^{N_n} V(\mathbf{R}_\alpha^0 - \hat{\mathbf{r}}_i)\right] + U_{ee}(\hat{\mathbf{r}}) = \sum_{n\mathbf{k}} \varepsilon_n(\mathbf{k})\hat{c}^\dagger_{n\mathbf{k}}\hat{c}_{n\mathbf{k}}. \tag{1.19}$$

With $\varepsilon_n(\mathbf{k})$ the form of the n-th band dispersion that can be computed using ab-initio methods, and the operators used are the creation and destruction ones for fermions. This term can be added to Eq. (1.16) to account for both the independent motion of nuclei and electrons. Nevertheless, to couple those degrees of freedom we shall need to quantize also the first term of the electron-nuclei interaction using the same base, having

$$\widehat{\mathscr{H}}_{en} = \sum_{nn'}\sum_{\mathbf{k}\mathbf{k}'}\left\langle n\mathbf{k}\middle|\sum_\alpha \hat{\mathbf{Q}}_\alpha \cdot \nabla V(\mathbf{R}_\alpha^0 - \hat{\mathbf{r}}_i)\middle|n'\mathbf{k}'\right\rangle \hat{c}^\dagger_{n\mathbf{k}}\hat{c}_{n\mathbf{k}}. \tag{1.20}$$

To compute the matrix elements of such operator we take advantage of the periodicity of the potential $V(\mathbf{R}_\alpha^0 - \hat{\mathbf{r}})$ inside Born-Von-Karman boundary conditions so that we can write

$$V(\mathbf{R}_\alpha^0 - \hat{\mathbf{r}}) = \sum_{\mathbf{q}} \tilde{V}(\mathbf{q})e^{i\mathbf{q}\cdot(\mathbf{R}_\alpha^0 - \hat{\mathbf{r}})}, \qquad \nabla V(\mathbf{R}_\alpha^0 - \hat{\mathbf{r}}) = -i\sum_{\mathbf{q}} \mathbf{q}\tilde{V}(\mathbf{q})e^{i\mathbf{q}\cdot(\mathbf{R}_\alpha^0 - \hat{\mathbf{r}})}. \tag{1.21}$$

By inserting it inside the matrix element along with the known expression for the phonon displacement in Eq. (1.18) we can obtain, after some manipulations, the following result

$$\langle n\mathbf{k}|\cdots|n'\mathbf{k}'\rangle = \sum_{\mathbf{G}\mathbf{q}}\sum_\nu -i(\mathbf{q}+\mathbf{G})\cdot\boldsymbol{\epsilon}_\nu(\mathbf{q})\tilde{V}(\mathbf{q}+\mathbf{G})\sqrt{\frac{N\hbar}{2M\omega_\nu(\mathbf{q})}}\left(\hat{b}_{\nu\mathbf{q}} + \hat{b}^\dagger_{\nu-\mathbf{q}}\right)\times$$
$$\times \frac{1}{V_{UC}}\int_{UC} \mathrm{d}\mathbf{r} u_{n\mathbf{k}'+\mathbf{q}+\mathbf{G}}(\mathbf{r})u_{n'\mathbf{k}'}(\mathbf{r})\delta_{\mathbf{k},\mathbf{k}'+\mathbf{q}+\mathbf{G}}.$$

Where $u_{n\mathbf{k}}$ is the periodic part of the Block wave function, and $\mathbf{G}$ the reciprocal lattice vectors. If we insert the result inside Eq. (1.20) and add the electronic and phononic part the following general model Hamiltonian is obtained

$$\widehat{\mathscr{H}} = \sum_{n\mathbf{k}}\varepsilon_n(\mathbf{k})\hat{c}^\dagger_{n\mathbf{k}}\hat{c}_{n\mathbf{k}} + \sum_\nu\sum_{\mathbf{k}}\hbar\omega_\nu(\mathbf{k})\hat{b}^\dagger_{\nu\mathbf{k}}\hat{b}_{\nu\mathbf{k}} + \sum_{nn'\nu}\sum_{\mathbf{k}\mathbf{q}\mathbf{G}} M^{nn'\nu}_{\mathbf{k}\mathbf{G}}(\mathbf{q})\hat{c}^\dagger_{n\mathbf{k}+\mathbf{q}+\mathbf{G}}\hat{c}_{n'\mathbf{k}}\left(\hat{b}_{\nu\mathbf{q}} + \hat{b}^\dagger_{\nu-\mathbf{q}}\right) \tag{1.22}$$

The zero energy point given by the 1/2 was omitted in the expression and the interaction strength of the electron-phonon coupling has been collected inside the vertex $M^{nn'\nu}_{\mathbf{k}\mathbf{G}}$

$$M^{nn'\nu}_{\mathbf{k}\mathbf{G}}(\mathbf{q}) = -i\sqrt{\frac{N\hbar}{2M\omega_\nu(\mathbf{q})}}(\mathbf{q}+\mathbf{G})\cdot\boldsymbol{\epsilon}_\nu(\mathbf{q})\frac{\tilde{V}(\mathbf{q}+\mathbf{G})}{V_{UC}}\int_{UC}\mathrm{d}\mathbf{r} u_{n\mathbf{k}+\mathbf{q}+\mathbf{G}}(\mathbf{r})u_{n'\mathbf{k}}(\mathbf{r}). \tag{1.23}$$

In this way we have obtained a general model that is able to evaluate how the motion of the nuclei influences the electronic properties inside a lattice.

The final form obtained in Eq. (1.22) is too general and complicated, and to perform meaningful computations with it we shall make some assumptions on the specific system under study. In particular, following the reasoning of Holstein in its seminal papers [21, 22] we are going to make the following assumptions:

Dispersionless phonons We assume that only optical phonons give a contribution to electronic properties allowing to drop the ν subscript and use a Einstein model for the dispersion giving $\omega(\mathbf{q}) = \Omega$;

Single band The Vertex becomes small if two different bands $n \neq n'$ are involved, meaning that we can study the effects on one band at a time without counting the others and dropping the n index;

Normal processes For similar reasons to the single band we are going to count only the processes with $\mathbf{G} = 0$;

Constant coupling The interaction between electrons and phonons is taken to be independent of the phonon's momentum so that one can simplify $M_{\mathbf{k}}(\mathbf{q}) = g$.

Inserting all of these considerations inside the Hamiltonian leave us with a form that is known in literature as the *Holstein Hamiltonian*

$$\hat{\mathscr{H}} = \sum_{\mathbf{k}} \varepsilon(\mathbf{k}) \hat{c}^{\dagger}_{\mathbf{k}} \hat{c}_{\mathbf{k}} + \Omega \sum_{\mathbf{q}} \hat{b}^{\dagger}_{\mathbf{q}} \hat{b}_{\mathbf{q}} + \frac{g}{\sqrt{N}} \sum_{\mathbf{kq}} \hat{c}^{\dagger}_{\mathbf{k}+\mathbf{q}} \hat{c}_{\mathbf{k}} \left(\hat{b}_{\mathbf{q}} + \hat{b}^{\dagger}_{-\mathbf{q}} \right). \tag{1.24}$$

As was shown by its developer, this model is able to describe the behavior of *small polarons* quasiparticles. Giving information on their binding energy, $\varepsilon_p(\mathbf{k})$, and effective masses, $m_p(\mathbf{k})$. Meaning that from Eq. (1.24) we can estimate corrections $\varepsilon_p(\mathbf{k})$ that can be added to the ab-initio bands $\varepsilon(\mathbf{k})$ in order to account for the electron-phonon coupling. Such renormalization will modify the bands bending that will change the effective mass of the electrons influencing the conduction properties of the material.

Inside this thesis work we are interested in a further simplified version of the model that can be used as a test ground for new computational methods. We will so assume that a single energy level ε is present, setting the system in the *atomic limit*

$$\hat{\mathscr{H}} = \varepsilon \hat{c}^{\dagger} \hat{c} + \Omega \sum_{\mathbf{q}} \hat{b}^{\dagger}_{\mathbf{q}} \hat{b}_{\mathbf{q}} + \frac{g}{\sqrt{N}} \sum_{\mathbf{q}} \hat{c}^{\dagger} \hat{c} \left(\hat{b}_{\mathbf{q}} + \hat{b}^{\dagger}_{-\mathbf{q}} \right). \tag{1.25}$$

This form can be solved exactly by mean of the *Lang-Firsov transformation* [30] which bring to the following diagonalized form [35]

$$\hat{\mathscr{H}}' = \left(\varepsilon + \varepsilon_p \right) \hat{c}'^{\dagger} \hat{c}' + \Omega \sum_{\mathbf{q}} \hat{b}^{\dagger}_{\mathbf{q}} \hat{b}_{\mathbf{q}}, \qquad \varepsilon_p = -\frac{g^2}{\Omega}. \tag{1.26}$$

An exact correction for the energy level taken into account is obtained and can be used as a numerical check to see if the evaluation obtained from the algorithm under study is converging to the right value.

1.3 Dissipative two level systems

Dynamical properties are often the ones that plays the most important roles when it comes to applications of material physics. Nuclear magnetic resonance (NMR), inelastic electron scattering or transport properties investigations, are all examples of techniques that exploits the response over time, $\gamma(t)$, of the system to an external force, $\xi(t)$, to control its dynamic. Theoretically, we can drive such experiments using models able to describe $\gamma(t)$ in terms of the time correlation function of the response. The way in which this is usually done is by simplifying the complex many-body problem focussing the attention on the few relevant dynamical variables that interacts with the rest of the system. In NMR, for example, we can focus on the nuclei spins and how they distribute in time while they interact with each other and the phonons and electrons of the material. Knowledge on this process can be retrieved using results from statistical mechanics

$$P(t) = \langle \hat{\sigma}_z(t) \rangle = \frac{1}{Z} \mathrm{tr}\left[\hat{\sigma}_z e^{-\beta \widehat{\mathscr{H}}}\right], \qquad Z = \mathrm{tr}\left[e^{-\beta \widehat{\mathscr{H}}}\right], \tag{1.27}$$

where $\widehat{\mathscr{H}}$ is the Hamiltonian describing the system, $P(t)$ represent the probability of a nucleus to be in state up at time t, and Z is the partition function of the system. This type of dynamical studies are really challenging for first principles methods, if not impossible, since they often cope with large systems or with temperature dependencies that are complex to capture with ground state theories. Therefore, we want to introduce a category of model Hamiltonians that are able to study such behaviors by thinking at the interactions as a dissipative contribution to the energy of a reduced system. In particular, focussing on a material perspective we will see how to introduce dissipation as a phonon bath and how such environment can couple to a general two level system (TLS).

Phonic bath with linear coupling

Let's consider a system with few degrees of freedom coupled with a huge environment composed of a series of harmonic oscillators. We can decompose the Hamiltonian for the complete system in the sum of three contributions [52]

$$\widehat{\mathscr{H}} = \widehat{\mathscr{H}}_S + \widehat{\mathscr{H}}_R + \widehat{\mathscr{H}}_I. \tag{1.28}$$

Here $\widehat{\mathscr{H}}_S$ represent the contribution given by the selected degrees of freedom, that we assume being the ones of a particle with mass M in a potential $V(q)$, and $\widehat{\mathscr{H}}_R$ the Harmonic reservoir

$$\widehat{\mathscr{H}}_S = \frac{\hat{p}^2}{2M} + V(\hat{q}), \qquad \widehat{\mathscr{H}}_R = \sum_\alpha \left(\frac{\hat{p}_\alpha^2}{2m_\alpha} + \frac{1}{2} m_\alpha \omega_\alpha^2 \hat{x}_\alpha^2 \right). \tag{1.29}$$

The choice of a particle in a potential was taken only for convenience, we will see how the description of the interaction part $\widehat{\mathscr{H}}_I$ will remain applicable to other kind of systems. In fact, to write an expression for it, we only need to assume that the degrees of freedom we are looking at couple with each single oscillator with an intensity proportional to the inverse of the volume of the bath. Therefore, *the coupling to an individual bath mode*

is weak and the strength of the interaction is built up by the large number of modes that are present. In this assumption, it's reasonable to assume that the coupling can be described by a linear function of the type

$$\widehat{\mathscr{H}_I} = -\sum_\alpha F_\alpha(\hat{q})\hat{x}_\alpha + \Delta V(\hat{q}), \qquad \Delta V(q) = \sum_\alpha \frac{F_\alpha^2(q)}{2m_\alpha\omega_\alpha^2}. \tag{1.30}$$

The potential variation $\Delta V(q)$ is present in order to avoid unwanted renormalization of the potential $V(q)$ that will bring to a modification of the physics, while the aim of this interaction is only to add dissipation. Also, usually a further assumption is made to give a simple expression to $F_\alpha(q)$ as linear functions of type $F_\alpha(q) = c_\alpha q$, which is typical for *state independent dissipation*. Then, by inserting it inside Eq. (1.30) the general form for the global Hamiltonian is obtained

$$\widehat{\mathscr{H}} = \frac{\hat{p}^2}{2M} + V(\hat{q}) + \frac{1}{2}\sum_\alpha \left[\frac{\hat{p}_\alpha^2}{2m_\alpha} + m_\alpha\omega_\alpha^2\left(\hat{x}_\alpha - \frac{c_\alpha}{m_\alpha\omega_\alpha^2}\hat{q}_\alpha\right)^2\right]. \tag{1.31}$$

From this expression, is possible to see how a damped dynamic for the expectation values of the system q can be obtained in the form of a *generalized Langevin equation* [52]

$$M\ddot{q} + V'(q) + M\int_0^t \mathrm{d}t'\gamma(t-t')\dot{q}(t') = \xi(t). \tag{1.32}$$

Where the dissipative effects generated by the coupling with the environment are encapsulated inside the memory friction coefficient γ and a random force ξ

$$\gamma(t) = \frac{\Theta(t)}{M}\sum_\alpha \frac{c_\alpha^2}{m_\alpha\omega_\alpha^2}\cos(\omega_\alpha t), \qquad \xi(t) = \sum_\alpha c_\alpha\left(x_\alpha^0\cos(\omega_\alpha t) + \frac{p_\alpha^0}{m_\alpha\omega_\alpha}\sin(\omega_\alpha t)\right). \tag{1.33}$$

These two quantities encapsulate all the effects of the phononic bath on the system under study by effectively dissipate its energy over time, and inducing state decoherence.

Usually, instead on focussing on the forms of γ and ξ to describe the environment a simpler function is defined that can infer both called *spectral density*

$$J(\omega) = \frac{\pi}{2}\sum_\alpha \frac{c_\alpha^2}{m_\alpha\omega_\alpha}\delta(\omega - \omega_\alpha). \tag{1.34}$$

This form is chosen in order to be able to substitute it inside the expression of the friction coefficient directly, and can be also showed that determines the correlation of the random force $\langle\xi(t)\xi(0)\rangle = X(t)$. Assuming that ξ obeys a Gaussian statistics we can obtain all that we need to know on the dynamic simply from knowing J

$$\gamma(t) = \frac{\Theta(t)}{M}\frac{2}{\pi}\int_0^\infty \mathrm{d}\omega\frac{J(\omega)}{\omega}\cos(\omega t), \quad X(t) = \frac{\hbar}{\pi}\int_0^\infty \mathrm{d}\omega J(\omega)\left[e^{-i\omega t}(1-n(\omega)) + e^{i\omega t}n(\omega)\right], \tag{1.35}$$

with $n(\omega)$ being the Bose-Einstein occupation of frequency ω. Eq. (1.35) narrows down the information needed to describe the environment effects to the spectral density alone. Thus, we can see how a better description of the function can be done, since the δ-peak of Eq. (1.34) is a result of a discrete number of degrees of freedom inside the reservoir.

In a realistic situation, like the one of a phononic lattice, the sum can be extended to an integration and J will become a smooth function describing the coupling strength with every oscillatory mode. To describe it a separation is usually made between low and high frequency contribution using a cutoff function $f(\omega/\omega_c)$ as follows

$$J_{lf}(\omega) = J(\omega) f(\omega/\omega_c), \qquad J_{hf}(\omega) = J(\omega)[1 - f(\omega/\omega_c)], \tag{1.36}$$

so that $J(\omega)$ can be recovered by summing the two. The cutoff frequency ω_c is usually selected in the range of the Drude, Debye or Fermi frequency, so that we can focus on the low-frequency regime being the one physically relevant. In fact, taking $\omega \ll \omega_c$ we can see how J_{hf} can be accounted through a renormalization of the mass $\Delta M \propto \int \mathrm{d}\omega J_{hf}(\omega)/\omega^3$. A contribution that is assumed to still be negligible since in physical systems $J(\omega)$ usually decays quickly for large ω. The dynamic is so dictated by J_{lf} which is physically reasonable to assume as a power law of the type

$$J_{lf}(\omega) = \alpha \omega_c^{1-s} \omega^s \Theta\left(1 - \frac{\omega}{\omega_c}\right), \tag{1.37}$$

where α and s are parameters and the cutoff function f has been taken as a step one. Eq. (1.37) is the standard form mostly used to describe various dissipation phenomena in solid state and different dissipation regimes are obtained based on the value of s. In particular, the most interesting case is for $s = 1$ called *Ohmic regime* since generates a damping inside the system analogous to the one of the Drude model. From it, the regimes are divided in *super-Ohmic*, $s > 1$, and *sub-Ohmic*, $s \in (0,1)$, but the $s = 1$ case remains almost ubiquitously met in real systems at low temperature. Thus, leaving us with a really simple description of how to model the environment based on mainly two real parameters ω_c and α.

Spin-Boson model

Many physical and chemical systems can be described using a set of coordinates able to map the problem into one with a double-well potential possessing two nearly degenerate minima. The motion of defect in crystalline solids, tunneling of light particles in metals [34] and anomalous conductance in mesoscopic wires [8] are all possible examples. Such systems become especially interesting reaching low temperatures, when the thermal fluctuations are much lower than the energy spacing of the low-lying states. In that regime only the ground states of the two wells are involved in the physics of the phenomena, leaving a simple two-dimensional Hilbert space that can act as a *qubit* described by the well known *two level system* (TLS)

$$\widehat{\mathscr{H}}_{TLS} = \frac{\epsilon}{2}\hat{\sigma}_z + \frac{\Delta}{2}\hat{\sigma}_x. \tag{1.38}$$

Where the $\hat{\sigma}_i$ are the Pauli operators, ϵ gives the energy difference between the two ground state and Δ is the tunneling matrix element that can be related to the barrier height V_0 showed in Fig. (1.1). The popularity of such models has risen in the recent years thanks to the race in improving quantum computing technologies bringing a lot of effort in the search for new systems that can act as *qubits*. Still, the case described by

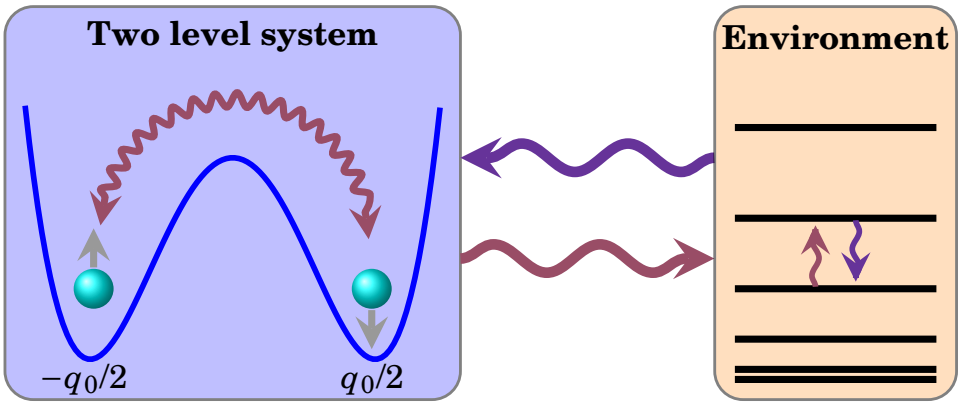


Figure 1.1: Graphical description of the Spin-Boson model as a two level system that can interact with the environment modelled as a phononic bath. Energy exchange by the two systems bring to excitations in the environment and state flips in the two level state.

Eq. (1.38) is ideal, allowing to use external fields to bring the system into precise superpositions of the ground states, $|\pm\rangle$, without decoherence. In fact, what really happens is that the system is coupled with the environment surrounding it, giving rise to random interactions that are difficult to control. In most experimental cases this coupling can still be inserted inside the model using the approach described in Sec. (1.3.1) and account for a linear interaction term in the coordinates of a phononic bath

$$\widehat{\mathcal{H}}_I = \hat{\sigma}_z \mathfrak{P}, \qquad \mathfrak{P} = \frac{q_0}{2} \sum_\alpha c_\alpha \hat{x}_\alpha. \tag{1.39}$$

One might argue that we should consider the possibility of coupling with $\hat{\sigma}_x$ and $\hat{\sigma}_y$ operators. Even if there are situations, like NMR, where such contributions are important their influence on the system's properties is still much lower compared to Eq. (1.39). This is due to both the x and y Pauli matrices possessing only non-diagonal elements in the $\hat{\sigma}_z$ representation. Thus, any interaction proportional to them must also be proportional to the overlap of $|+\rangle$ with $|-\rangle$ in coordinate space, i. e., giving a contribution of order Δ assumed to be an exponentially small energy [31]. Therefore, we can insert the environment effects inside Eq. (1.38) obtaining a form that has come to be known as the *spin-boson model*

$$\widehat{\mathcal{H}}_{SB} = \frac{\epsilon}{2}\hat{\sigma}_z + \frac{\Delta}{2}\hat{\sigma}_x + \frac{1}{2}\sum_\alpha \left(\frac{\hat{p}_\alpha}{m_\alpha} + m_\alpha \omega_\alpha^2 \hat{x}_\alpha^2 + q_0 \hat{\sigma}_z c_\alpha \hat{x}_\alpha \right). \tag{1.40}$$

In this way, by making the right choice for the spectral function $J(\omega)$, we can predict the realistic behavior of qubit-like systems.

Looking at $\widehat{\mathcal{H}}_{SB}$ from a condensed matter point of view allows us to reconduce $\hat{p}_\alpha$ and $\hat{x}_\alpha$ to the same $\tilde{\mathbf{P}}(\mathbf{k})$ and $\tilde{\mathbf{Q}}(\mathbf{k})$ described in Sec. (1.2.1). Thus, by mapping α to a one dimensional momentum variable q we can rewrite the Hamiltonian using bosonic creation and destruction operators instead of the momentum and displacement ones. Therefore, by Fourier transforming Eq. (1.17) and Eq. (1.18) in a 1D case a form for $\hat{p}_q$ and $\hat{x}_q$ is obtained that transform Eq. (1.40) into

$$\widehat{\mathcal{H}}_{SB} = \frac{\epsilon}{2}\hat{\sigma}_z + \frac{\Delta}{2}\hat{\sigma}_x + \hat{\sigma}_z \sum_q \lambda_q \left(\hat{b}_q + \hat{b}_q^\dagger \right) + \sum_q \omega_q \hat{b}_q^\dagger \hat{b}_q. \tag{1.41}$$

This is the second quantized form of the spin-boson model, where the environment interaction is encoded inside the spectral function taking the form

$$J(\omega) = \sum_q \lambda_q^2 \delta(\omega - \omega_q) = \frac{q_0^2}{2} \sum_q \frac{c_q^2}{m_q \omega_q} \delta(\omega - \omega_q). \tag{1.42}$$

The above expression being analogous to the one seen in Eq. (1.34) we can use the same considerations as before in order to describe it by using the low-frequency regime only. Phonon dispersion inside such limit can be also simply described by a linear model approximating the usual acoustic phonon band, resulting in

$$J(q) = \frac{\alpha}{2} \omega_c q^s \Theta(1-q), \qquad \omega_q = \omega_c q. \tag{1.43}$$

By assuming to be in the Ohmic regime we will have that the dynamic of the whole model is defined by simply three parameters: the tunneling matrix Δ, the coupling strength α and the cutoff frequency ω_c.

1.4 Solving many-body problems with time evolution

All models presented so far exhibit a form that can be decomposed in an exactly solvable part $\widehat{\mathscr{H}}_0$ and a perturbation $\widehat{\mathscr{V}}$. To obtain results from such Hamiltonians one usually starts from the known results from the exactly solvable part and gradually insert the troublesome perturbation. In practice this is done by rewriting the Hamiltonian in a form where the intensity of $\widehat{\mathscr{V}}$ can be controlled, like

$$\widehat{\mathscr{H}}_\eta = \widehat{\mathscr{H}}_0 + e^{-\eta|t|} \widehat{\mathscr{V}}, \qquad \eta > 0. \tag{1.44}$$

In this way for $t \to \pm\infty$ the system can be described by the states of $\widehat{\mathscr{H}}_0$, while as $t \to 0$ is approached they will be modified from the perturbation reaching the solution of the interacting system. To take advantage of that we need to describe how quantum states evolves in time under the influence of Eq. (1.44) by approaching the study of the time-dependent SE

$$i\hbar \frac{\partial \left|\psi_\eta(t)\right\rangle}{\partial t} = \widehat{\mathscr{H}}_\eta \left|\psi_\eta(t)\right\rangle. \tag{1.45}$$

Therefore, knowing how the states $\left|\psi_\eta(t)\right\rangle$ evolves we can take the states of $\widehat{\mathscr{H}}_0$, and make them evolve from $t = \infty$ to 0 to transform them in the interaction ones. For these reasons we are going to tackle the time evolution problem within quantum mechanics for a general interaction Hamiltonian. Ultimately, we are interested to see how time evolution would bring insight on the energy spectrum of the systems and on the thermodynamical properties through the formulation of the Feynman diagrams. Also, from now on all our treatment will assume to use the natural units $\hbar = 1$ since it will make our equations much lighter.

Interaction picture

Consider an Hamiltonian composed simply by $\widehat{\mathscr{H}}_0$ and a time dependent perturbation $\widehat{\mathscr{V}}(t)$, our goal is to find a way to formally describe the evolution of a state $|\psi\rangle$ described by the following equation

$$\widehat{\mathscr{H}}(t) = \widehat{\mathscr{H}}_0 + \widehat{\mathscr{V}}(t), \qquad i\frac{\partial |\psi(t)\rangle}{\partial t} = \widehat{\mathscr{H}}(t)|\psi(t)\rangle. \tag{1.46}$$

It is known that such mathematical problem admits a solution in the form of a operator $\widehat{\mathscr{U}}(t,t')$, which allows to describe the time evolution acting on the states

$$\widehat{\mathscr{U}}(t,t')|\psi(t')\rangle = |\psi(t)\rangle. \tag{1.47}$$

Using Eq. (1.46) can be shown that such operator exists and posses the following properties

$$\widehat{\mathscr{U}}(t,t) = \hat{\mathbb{1}}, \qquad \widehat{\mathscr{U}}(t,t'')\widehat{\mathscr{U}}(t'',t') = \widehat{\mathscr{U}}(t,t'), \qquad \widehat{\mathscr{U}}^{\dagger}(t,t') = \widehat{\mathscr{U}}(t',t) = \widehat{\mathscr{U}}^{-1}(t,t'). \tag{1.48}$$

Along with that, by inserting Eq. (1.47) inside Eq. (1.46), a Schrödinger-like equation defining the form of the time evolution operator can be obtained as

$$i\frac{\partial \widehat{\mathscr{U}}(t,t')}{\partial t} = \widehat{\mathscr{H}}(t)\widehat{\mathscr{U}}(t,t'). \tag{1.49}$$

This operator partial differential equation is a complex task to tackle analytically, but simplifications can be made in order to reduce its complexity using *pictures*. The idea is analogous to change the system of reference in classical dynamics, so that we can take a general unitary transformation $\hat{U}$ to redefine the state $|\psi(t)\rangle$ and operators $\hat{O}$ as if we were rotating the system of reference

$$|\psi(t)\rangle_U = \hat{U}|\psi(t)\rangle, \qquad \hat{O}_U = \hat{U}\hat{O}\hat{U}^{\dagger}. \tag{1.50}$$

The new states will be associated to a modified SE describing the time evolution that depends on the form of $\hat{U}$, so that by making the right transformation large simplifications can be achieved.

Different kind of standard pictures are used inside different problems of quantum mechanics: the trivial case given by Eq. (1.46) is called *Schrödinger picture*, the use of $\hat{U} = \exp\left(i\widehat{\mathscr{H}}t\right)$ describe instead a picture where $\widehat{\mathscr{U}}(t,t') = \hat{\mathbb{1}}$ having states frozen in time but evolving observable, called *Heisenberg picture*. In our case, the evolution will be formulated in the *interaction picture* defined by $\hat{U} = \exp\left(i\widehat{\mathscr{H}}_0 t\right)$ so that we have

$$|\psi(t)\rangle_I = e^{i\widehat{\mathscr{H}}_0 t}|\psi(t)\rangle, \qquad \hat{O}_I(t) = e^{i\widehat{\mathscr{H}}_0 t}\hat{O}e^{-i\widehat{\mathscr{H}}_0 t}. \tag{1.51}$$

To see how time evolution is affected by such choice we take the derivative of the new state

$$i\frac{\partial |\psi(t)\rangle_I}{\partial t} = -\widehat{\mathscr{H}}_0|\psi(t)\rangle_I + e^{i\widehat{\mathscr{H}}_0 t}i\frac{\partial |\psi(t)\rangle}{\partial t} = e^{i\widehat{\mathscr{H}}_0 t}\left(\widehat{\mathscr{H}} - \widehat{\mathscr{H}}_0\right)|\psi(t)\rangle = \widehat{\mathscr{V}}_I(t)|\psi(t)\rangle_I. \tag{1.52}$$

This shows how the evolution is dictated only by the interactive part of the Hamiltonian. A variation that is also seen inside the time evolution operator, whose form can be easily seen to change into

$$\widehat{\mathscr{U}}_I(t,t') = e^{i\widehat{\mathscr{H}}_0 t}\widehat{\mathscr{U}}(t,t')e^{-i\widehat{\mathscr{H}}_0 t'}, \qquad i\frac{\partial\widehat{\mathscr{U}}_I(t,t')}{\partial t} = \widehat{\mathscr{V}}_I(t)\widehat{\mathscr{U}}_I(t,t'), \tag{1.53}$$

while the properties inside Eq. (1.48) remains intact. A general solution to this equation can now be found by simply integrating directly the expression having

$$\widehat{\mathscr{U}}_I(t,t') = \widehat{\mathscr{U}}_I(t',t') - i\int_{t'}^{t} \mathrm{d}t_1 \widehat{\mathscr{V}}_I(t_1)\widehat{\mathscr{U}}_I(t_1,t') = \hat{\mathbb{1}} - i\int_{t'}^{t} \mathrm{d}t_1 \widehat{\mathscr{V}}_I(t_1)\widehat{\mathscr{U}}_I(t_1,t'). \tag{1.54}$$

This integral equation cannot be solved analytically, but can be seen how Eq. (1.54) can be inserted into itself recursively to obtain an expansion for the time evolution operator

$$\widehat{\mathscr{U}}_I(t,t') = \sum_{n=0}^{\infty}(-i)^n \int_{t'}^{t}\mathrm{d}t_1\int_{t'}^{t_1}\mathrm{d}t_2\cdots\int_{t'}^{t_{n-1}}\mathrm{d}t_n \widehat{\mathscr{V}}_I(t_1)\cdots\widehat{\mathscr{V}}_I(t_n). \tag{1.55}$$

Such form for $\widehat{\mathscr{U}}_I$ is known as *Dyson expansion* and represent a general solution that can describe time evolution in every interacting systems.

For our purposes is better to rewrite Eq. (1.55) in a more compact form. To do that, we introduce the *time-ordering operator* acting on time-dependent operator as follows

$$\widehat{\mathscr{T}}\left[\widehat{\mathscr{V}}_I(t_1)\widehat{\mathscr{V}}_I(t_2)\cdots\widehat{\mathscr{V}}_I(t_n)\right] = \widehat{\mathscr{V}}_I(t_2)\widehat{\mathscr{V}}_I(t_1)\cdots\widehat{\mathscr{V}}_I(t_k), \qquad t_2 < t_1 < \cdots < t_k. \tag{1.56}$$

Having that we can start the rewriting of the expansion taking the term of order two and seeing how we can add it to its permutation using the Heaviside step function Θ as

$$\begin{aligned}\int_{t'}^{t}\mathrm{d}t_1\int_{t'}^{t_1}\mathrm{d}t_2\widehat{\mathscr{V}}_I(t_1)\widehat{\mathscr{V}}_I(t_2) &= \frac{1}{2}\int_{t'}^{t}\mathrm{d}t_1\int_{t'}^{t}\mathrm{d}t_2\left[\Theta(t_1-t_2)\widehat{\mathscr{V}}_I(t_1)\widehat{\mathscr{V}}_I(t_2)+\Theta(t_2-t_1)\widehat{\mathscr{V}}_I(t_2)\widehat{\mathscr{V}}_I(t_1)\right]\\ &= \frac{1}{2}\int_{t'}^{t}\mathrm{d}t_1\int_{t'}^{t}\mathrm{d}t_2\widehat{\mathscr{T}}\left[\widehat{\mathscr{V}}_I(t_1)\widehat{\mathscr{V}}_I(t_2)\right].\end{aligned}$$

Where it was seen how the form with the step functions acted as the $\widehat{\mathscr{T}}$ operator. The same form can be generalized to every order inside the Dyson expansion by accounting for all $n!$ permutations of the operators in that order. Leading to the most known time-ordered form of the Dyson expansion

$$\widehat{\mathscr{U}}_I(t,t') = \sum_{n=0}^{\infty}\frac{(-i)^n}{n!}\int_{t'}^{t}\mathrm{d}t_1\int_{t'}^{t}\mathrm{d}t_2\cdots\int_{t'}^{t}\mathrm{d}t_n\widehat{\mathscr{T}}\left[\widehat{\mathscr{V}}_I(t_1)\cdots\widehat{\mathscr{V}}_I(t_n)\right]. \tag{1.57}$$

In the next sessions we will be using this form in order to study the time evolution of Eq. (1.44) and describe the effects of the interaction on the unperturbed system. Also, we will always work with time independent Hamitlonians inside the interaction picture. Therefore, we will assume that operators that shows time dependence are written in the interaction picture without the need of the subscript.

Interacting Green's functions

Taking into consideration a general interaction Hamiltonian we can always cast it in a form where the coupling is switched on in time like Eq. (1.44). Its time evolution can then be described by the use of Eq. (1.57) allowing us to write the interacting eigenstates $|\Psi_i\rangle$ from the non-interacting ones $|\psi_i\rangle$ as

$$|\Psi_i\rangle = \lim_{\eta\to 0}\widehat{\mathcal{U}}_\eta(0,-\infty)|\psi_i\rangle, \quad \widehat{\mathcal{U}}_\eta(0,-\infty) = \sum_{n=0}^{\infty}\frac{(-i)^n}{n!}\int_{-\infty}^{0} dt_1\ldots dt_n e^{-\eta\sum_i|t_i|}\widehat{\mathcal{T}}\left[\hat{\mathcal{V}}(t_1)\cdots\right]. \tag{1.58}$$

Such approach is also called *adiabatic switching on* and can be shown to be an exact approach in a rigorous way thanks to the *Gell-Mann-Low theorem* [14]. In particular, the theorem states that an eigenstate of the interacting Hamiltonian can be obtained from $|\psi_i\rangle$ in the following form

$$|\Psi_j\rangle = \frac{|\Phi_i\rangle}{\sqrt{\langle\Phi_i|\Phi_i\rangle}}, \qquad |\Phi_i\rangle = \lim_{\eta\to 0}\frac{\widehat{\mathcal{U}}_\eta(0,-\infty)|\psi_i\rangle}{\sqrt{\langle\psi_i|\widehat{\mathcal{U}}_\eta(0,-\infty)|\psi_i\rangle}}. \tag{1.59}$$

So that $|\Psi_j\rangle$ is the normalized eigenstate related to the j-th energy level inside the interaction spectra. Is important to notice that the state j is generated using state i of the unperturbed spectra. That is not an error but an important, if not the only, limitation of this approach. We cannot know in advance if the perturbation will generate an energy cross-over, as described in Fig. (1.2), changing the orders of the states in the spectrum. In our case we are going to assume to work with *simple perturbations*, but that is not always true like in the case of superconductivity.

Since, now, we have access to the eigenstates of the complete $\widehat{\mathcal{H}}$ the goal becomes to extract information about the spectra and system properties. The best way to achieve that is usually through the study of an object called *Green function* (GF) or *propagator*

$$G_{\mathbf{k}}(t'-t) = -i\left\langle\widehat{\mathcal{T}}\left[\hat{c}_{\mathbf{k}}(t')\hat{c}^\dagger_{\mathbf{k}}(t)\right]\right\rangle, \tag{1.60}$$

where $\langle\cdot\rangle$ represents the thermal average. For sake of simplicity we will pursue a description of such object in the limit of $T\to 0\,\mathrm{K}$, but all the arguments can be generalized to finite temperatures. In the zero temperature limit we can make the association $\langle\cdot\rangle = \langle\Psi_0|\cdot|\Psi_0\rangle$. Giving $G_{\mathbf{k}}(t'-t)$ to be proportional to the probability amplitude of adding a particle in a state $\mathbf{k}$ inside the ground state at time t and removing it from the same state at a time t'. By using Eq. (1.59) we can obtain a general form for the electronic Green function given by

$$G_{\mathbf{k}}(t) = -i\lim_{\eta\to 0}\frac{\left\langle\psi_0\left|\widehat{\mathcal{T}}\left[\hat{c}_{\mathbf{k}}(t)\hat{c}^\dagger_{\mathbf{k}}(0)\widehat{\mathcal{U}}_\eta(\infty,-\infty)\right]\right|\psi_0\right\rangle}{\langle\psi_0|\widehat{\mathcal{U}}_\eta(\infty,-\infty)|\psi_0\rangle}. \tag{1.61}$$

Both the numerator and the denominator, called *vacuum polarization* S, are further expandable using Eq. (1.58) leading to

$$G_{\mathbf{k}}(t) = -\frac{i}{S}\sum_{n=0}^{\infty}\frac{(-i)^n}{n!}\int_{-\infty}^{\infty}dt_1\cdots\int_{-\infty}^{\infty}dt_n\left\langle\psi_0\left|\widehat{\mathcal{T}}\left[\hat{c}_{\mathbf{k}}(t)\hat{c}^\dagger_{\mathbf{k}}(0)\hat{\mathcal{V}}(t_1)\ldots\hat{\mathcal{V}}(t_n)\right]\right|\psi_0\right\rangle \tag{1.62}$$

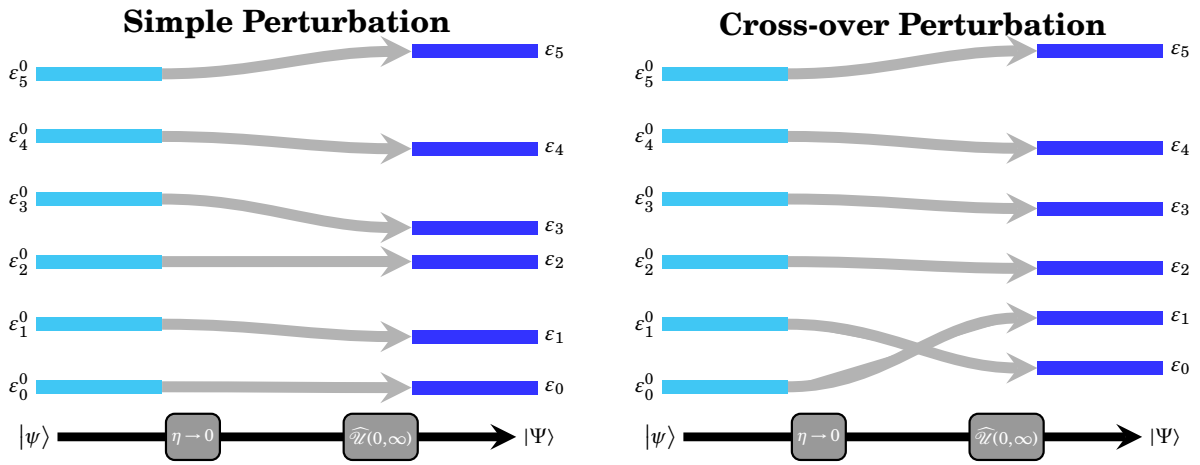


Figure 1.2: Example of adiabatic switching on applied to a general spectrum. In the left figure one can see how the levels remain in the same order with the interaction changing their energies, but on the right the ground state of the unperturbed Hamiltonian is brought to the first exited state of the perturbed one creating a cross-over.

The limit for $\eta \to 0$ has already been taken into account by not counting the exponents inside the expansion, and an analogous form is contained inside S. In this form the term inside the braket is not manageable, however a decomposition of it is always possible using the *Wick's theorem* [53]. To understand it we are going to take into consideration the electron-phonon Hamiltonian in Eq. (1.24) so that the expansion will be constructed by terms of the type

$$\left\langle \psi_0 \middle| \widehat{\mathcal{T}} \left[\hat{c}_{\mathbf{k}}(t) \hat{c}^\dagger_{\mathbf{k}}(0) \hat{c}^\dagger_{\mathbf{k}_1+\mathbf{q}_1}(t_1) \hat{c}_{\mathbf{k}_1}(t_1) \hat{B}_{\mathbf{q}_1}(t_1) \dots \right] \middle| \psi_0 \right\rangle, \qquad \hat{B}_{\mathbf{q}}(t) = \hat{b}_{\mathbf{q}}(t) + \hat{b}^\dagger_{\mathbf{q}}(t). \tag{1.63}$$

What the Whick's theorem allows us to do is to rewrite it as a sum of all the fully contracted terms. Where a contraction of two operators is defined as $\widehat{\mathcal{T}}[\hat{A}\hat{B}] - \widehat{\mathcal{N}}[\hat{A}\hat{B}]$, with $\widehat{\mathcal{N}}$ the normal-ordered operator that simply swaps positions to have the creator ones on the left. For example, the first order term gets decomposed to

$$\begin{aligned} &\left(\langle \psi_0 | \widehat{\mathcal{T}} \left[\hat{c}_{\mathbf{k}}(t) \hat{c}^\dagger_{\mathbf{k}}(0) \right] | \psi_0 \rangle \langle \psi_0 | \widehat{\mathcal{T}} \left[\hat{c}^\dagger_{\mathbf{k}_1+\mathbf{q}_1}(t_1) \hat{c}_{\mathbf{k}_1}(t_1) \right] | \psi_0 \rangle + \right. \\ &\left. \langle \psi_0 | \widehat{\mathcal{T}} \left[\hat{c}_{\mathbf{k}_1}(t_1) \hat{c}^\dagger_{\mathbf{k}}(0) \right] | \psi_0 \rangle \langle \psi_0 | \widehat{\mathcal{T}} \left[\hat{c}_{\mathbf{k}}(t) \hat{c}^\dagger_{\mathbf{k}_1+\mathbf{q}_1}(t_1) \right] | \psi_0 \rangle \right) \langle \psi_0 | \hat{B}_{\mathbf{q}_1}(t_1) | \psi_0 \rangle . \end{aligned}$$

Two main features can be seen for this result. The first is that by taking into account that the non-interacting phononic ground state is the vacuum, $|0\rangle$, follows that $\langle 0|\hat{B}_{\mathbf{q}}|0\rangle = 0$. This means that no first order terms are present in the expansion, a fact that can be generalized by seeing how the same form is obtained for every odd order. This brings to the general result that **inside the expansion of the electron-phonon green function no odd order terms are present**. The second point is that now all the expressions only contains Green functions terms related to the unperturbed part of the system, that we can compute. In fact, it's easy to see how inside the interaction picture the Heisenberg equation allows us to find

$$i \frac{\partial \hat{c}_{\mathbf{k}}(t)}{\partial t} = \left[\widehat{\mathscr{H}}_0, \hat{c}_{\mathbf{k}} \right] = \varepsilon_{\mathbf{k}} \hat{c}_{\mathbf{k}}, \qquad \hat{c}_{\mathbf{k}}(t) = \hat{c}_{\mathbf{k}} e^{-i\varepsilon_{\mathbf{k}} t}, \tag{1.64}$$

the same holds also for $\hat{b}_{\mathbf{q}}$ with $\omega_{\mathbf{q}}$ instead of $\varepsilon_{\mathbf{k}}$. Plugging them inside the definition of the propagator we will obtain

$$\langle\psi_0|\widehat{\mathcal{T}}\left[\hat{c}^{\dagger}_{\mathbf{k}'}(t)\hat{c}_{\mathbf{k}}(t)\right]|\psi_0\rangle = \delta_{\mathbf{k}'\mathbf{k}}n_{\mathbf{k}} = \delta_{\mathbf{k}'\mathbf{k}}G^0_{\mathbf{k}}(0^+), \tag{1.65}$$

$$\langle\psi_0|\widehat{\mathcal{T}}\left[\hat{c}_{\mathbf{k}'}(t')\hat{c}^{\dagger}_{\mathbf{k}}(t)\right]|\psi_0\rangle = \delta_{\mathbf{k}'\mathbf{k}}\Theta(t'-t)e^{-i\varepsilon_{\mathbf{k}}(t'-t)} = \delta_{\mathbf{k}'\mathbf{k}}G^0_{\mathbf{k}}(t), \tag{1.66}$$

$$\langle\psi_0|\widehat{\mathcal{T}}\left[\hat{B}_{\mathbf{q}'}(t')\hat{B}^{\dagger}_{\mathbf{q}}(t)\right]|\psi_0\rangle = \delta_{\mathbf{q}'\mathbf{q}}\left(\Theta(t'-t)e^{-i\omega_{\mathbf{q}}(t'-t)} + \Theta(t-t')e^{i\omega_{\mathbf{q}}(t'-t)}\right) = \delta_{\mathbf{q}'\mathbf{q}}D^0_{\mathbf{q}}(t). \tag{1.67}$$

These objects are known as non-interacting GF, and they are the building blocks on which the full interacting $G_{\mathbf{k}}$ is built, giving a complete analytical form to the expansion.

At this point one may start to wonder why going through all this trouble in order to obtain the full electronic propagator of the system. In that regard, we shall now see how the ground state properties can be inferred from $G_{\mathbf{k}}(t)$ in a really easy way. To do that the imaginary-time GF is usually studied by performing a transformation that takes the name of *Wick's rotation* and maps $it \to \tau$ to have

$$G_{\mathbf{k}}(\tau) = -\langle\Psi_0|\widehat{\mathcal{T}}\left[\hat{c}_{\mathbf{k}}(\tau)\hat{c}^{\dagger}_{\mathbf{k}}(0)\right]|\Psi_0\rangle. \tag{1.68}$$

Moving from the real to the imaginary axis highly simplifies our studies since now all the oscillating exponential will become simple decaying ones, for example

$$G^0_{\mathbf{k}}(\tau) = -\Theta(\tau)e^{-\varepsilon_{\mathbf{k}}\tau}, \qquad D^0_{\mathbf{q}}(\tau) = -\Theta(\tau)e^{-\omega_{\mathbf{q}}\tau} + \Theta(-\tau)e^{\omega_{\mathbf{q}}\tau}. \tag{1.69}$$

Now, let $\{|\Psi_{\mathbf{k}}\rangle\}$ be the set of interacting eigenstates having the properties

$$\widehat{\mathcal{H}}|\Psi_{\mathbf{k}}\rangle = \varepsilon_{\mathbf{k}}|\Psi_{\mathbf{k}}\rangle, \qquad \hat{\mathbb{1}} = \sum_{\mathbf{k}}|\Psi_{\mathbf{k}}\rangle\langle\Psi_{\mathbf{k}}|. \tag{1.70}$$

We can insert them inside the definition in Eq. (1.68) to obtain the result

$$G_{\mathbf{k}}(\tau) \propto \langle\Psi_0|\hat{c}_{\mathbf{k}}(\tau)\hat{\mathbb{1}}\hat{c}^{\dagger}_{\mathbf{k}}(0)|\Psi_0\rangle = \sum_{\mathbf{k}}\langle\Psi_0|\hat{c}_{\mathbf{k}}(\tau)|\Psi_{\mathbf{k}}\rangle\langle\Psi_{\mathbf{k}}|\hat{c}^{\dagger}_{\mathbf{k}}(t)|\Psi_0\rangle, \tag{1.71}$$

which can be easily written in a more compact form by expliciting the time dependence of the operators. In particular, by recalling Eq. (1.51) and using the full $\widehat{\mathcal{H}}$ instead of $\widehat{\mathcal{H}}_0$ to place us in the Heisenberg picture we can simply obtain

$$G_{\mathbf{k}}(\tau) = \sum_{\mathbf{k}}e^{\varepsilon_0\tau}e^{-\varepsilon_{\mathbf{k}}\tau}\langle\Psi_0|\hat{c}_{\mathbf{k}}(0)|\Psi_{\mathbf{k}}\rangle\langle\Psi_{\mathbf{k}}|\hat{c}^{\dagger}_{\mathbf{k}}(t)|\Psi_0\rangle = \sum_{\mathbf{k}}e^{-\varepsilon_{\mathbf{k}}\tau}\left|\langle\Psi_{\mathbf{k}}|\hat{c}^{\dagger}_{\mathbf{k}}|\Psi_0\rangle\right|^2. \tag{1.72}$$

Where ε_0 was set to 0 since $|\Psi_0\rangle$ is the vacuum state of the interacting system, which is true in the *low density limit* that assumes the electron inserted as the only one present. The GF has transformed into a sum of decaying terms leaded by the one possessing the smallest $\varepsilon_{\mathbf{k}}$ corresponding also to the ground state for the electron inserted. In this way the high-τ behavior of $G_{\mathbf{k}}(\tau)$ will give us information about such properties since it's form becomes

$$G_{\mathbf{k}}(\tau) \xrightarrow[\tau\to\infty]{} Z_0(\mathbf{k})e^{-\varepsilon^0_{\mathbf{k}}\tau}, \qquad Z(\mathbf{k}) = \left|\langle\Psi_{\mathbf{k}}|\hat{c}^{\dagger}_{\mathbf{k}}|\Psi_0\rangle\right|^2. \tag{1.73}$$

The quantity $Z(\mathbf{k})$ is also called *quasiparticle weight* and describes the overlap between the free particle state $\hat{c}^{\dagger}_{\mathbf{k}}|\Psi_0\rangle$ and the interacting state $|\Psi_{\mathbf{k}}\rangle$. Basically tells us how close to a quasi particle the inserted electron behaves. Therefore, we can now understand how this can tell us a lot in cases like the Holstein Hamiltonian where the interacting ground state is a polaron one. This method could retrieve both the polaronic character of the electron inside the system $Z_0(\mathbf{k})$ and the binding energy of the polaron $\varepsilon_0(\mathbf{k})$ completely describing the quasiparticle properties.

Feynman diagrams

Feynman introduced the idea of using the GF interpretation as probability amplitude of time propagation of a particle in state $\mathbf{k}$ to visualize them and give a graphical form to mathematical expressions [13]. The idea was to create a set of rules working as a vocabulary to translate from drawings to analytical form and vice versa, that in our case would look like the followings:

1. The unperturbed electronic GF, $G^0_{\mathbf{k}}(t'-t)$, becomes a solid line with an arrow that goes from t' to t and a $\mathbf{k}$ on top;

2. Phonons propagators, $D^0_{\mathbf{q}}(t'-t)$, are drawn as wiggly lines with a $\mathbf{q}$ on top, no arrow is used in this case since can be seen how

$$D^0_{\mathbf{q}}(t'-t) = D^0_{-\mathbf{q}}(t-t'),$$

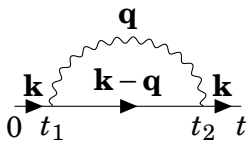
   and since $\omega_{\mathbf{q}} = \omega_{-\mathbf{q}}$ such propagator is symmetric in time, the direction in which the phonon is propagating does not matter;

3. Every intersection between an electron and phonon line generates an interaction vertex which correspond to a factor $M_{\mathbf{q}}$ inside the equation;

4. The factor $n_{\mathbf{k}}$ is thought as a Green's function where an electron propagates returning to the same point in time and so forming a solid loop.

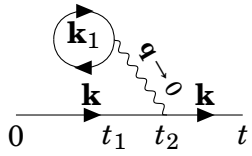
Starting from these simple concepts we want to make some examples by drawing some second order terms of the electron-phonon GF. Thus, by applying the Wick's theorem one finds out that the following term is present

$$\sum_{\mathbf{q}} |M_{\mathbf{q}}|^2 D^0_{\mathbf{q}}(t_2-t_1) G^0_{\mathbf{k}}(t_1) G^0_{\mathbf{k}-\mathbf{q}}(t_2-t_1) G^0_{\mathbf{k}}(t-t_2) \quad = \quad \tag{1.74}$$



also called *sunset diagram*. It's the diagram with the largest contribution at second order and the only one that gives a contribution at all in the low density limit. In order to see that we can have a look at the expression of another important diagram called *tadpole diagram* taking the form

$$\sum_{\mathbf{k_1}} |M_{\mathbf{q}\to 0}|^2 D^0_{\mathbf{q}\to 0}(t_2-t_1)(-n_{\mathbf{k_1}}) G^0_{\mathbf{k}}(t_2) G^0_{\mathbf{k}}(t-t_2) \quad = \quad \tag{1.75}$$



This is the second most important diagram and is proportional to the occupation of the state $\mathbf{k}_1$. Nevertheless, by working in a situation where the non-interacting ground state is the vacuum we have zero occupation in every level, so that **no contribution arise from diagrams with fermionic loops**. This condition takes the name of *low density limit* and will be our working regime. In this way, we even out a large portion of diagrams in our computations, but still an even larger simplification needs to be done to effectively make Eq. (1.74) the only contribution at $n = 2$. To understand this we can look at Fig. (1.3) where the first diagrams in the expansion of Eq. (1.62) are drawn divided

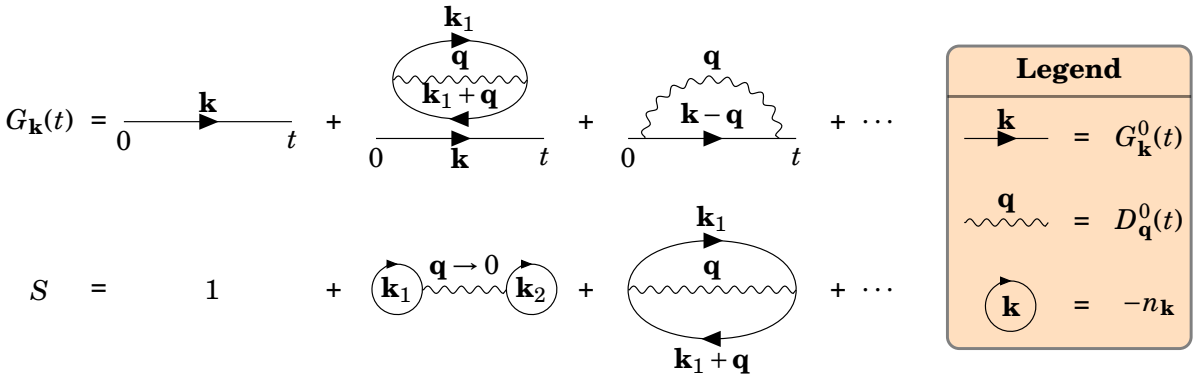
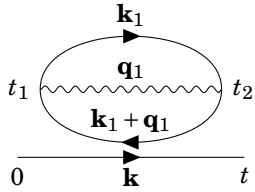


Figure 1.3: Diagrammatic expansions of the Green's function contribution at the numerator, denoted as $G_{\mathbf{k}}$, and denominator, the vacuum polarization S. A legend to the right is present to read the diagrams, remembering that every vertex bring a factor $M_{\mathbf{q}}$ to the equation.

into numerator and denominator parts. In the former a second order term composed by two separate parts can be noted, so that we want to focus a moment on it reporting here its translation into equations

$$-\sum_{\mathbf{k}_1,\mathbf{q}_1} \left|M_{\mathbf{q}}\right|^2 D^0_{\mathbf{q}_1}(t_1-t_2)G^0_{\mathbf{k}_1+\mathbf{q}_1}(t_1-t_2)G^0_{\mathbf{k}_1}(t_2-t_1)G^0_{\mathbf{k}}(t) =$$

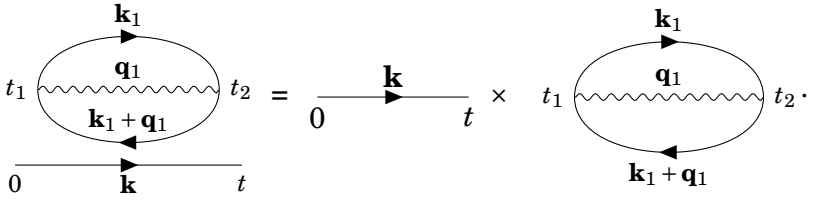


. (1.76)

By looking at the expression obtained it is possible to notice how the elements composing it can be separated in two groups differentiated by the ones depending on the loop's variables and not. A distinction that allow us to factorize Eq. (1.76) into the following product

$$\left[-\sum_{\mathbf{k}_1,\mathbf{q}_1} \left|M_{\mathbf{q}}\right|^2 D^0_{\mathbf{q}_1}(t_1-t_2)G^0_{\mathbf{k}_1+\mathbf{q}_1}(t_1-t_2)G^0_{\mathbf{k}_1}(t_2-t_1)\right] \times \left[G^0_{\mathbf{k}}(t)\right], \tag{1.77}$$

which is composed by two equations that can be easily translated back into diagram forms. In fact, by doing that one can see how the expression inside the left bracket of Eq. (1.77) describes a diagram composed by only the loop part of Eq. (1.76), as the one shown in the expansion of the vacuum polarization in Fig. (1.3). The right bracket, instead, can be seen to contain the simple unperturbed Green's function diagram, the first element of the $G_{\mathbf{k}}$ expansion. By noticing that we can transfer the factorization of the expression directly in terms of diagrams, having that is possible to rewrite the initial separated diagram as the following product of simple ones



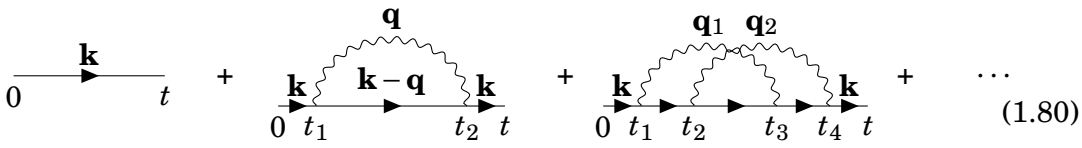
. (1.78)

Such diagrams are called *disconnected* and can be always factorized in the product of a *connected* one and another diagram present in the expansion of S, as it's happened in this case. By accounting for the factorization of all possible disconnected diagrams it's possible to demonstrate that every connected diagram has a disconnected form with

every element of the vacuum polarization. Bringing to the complete factorization of the expansion also called *linked cluster theorem*

$$G_{\mathbf{k}}(t) = -i\frac{S}{S}\sum_{n=0}^{\infty}\frac{(-i)^n}{n!}\int_{-\infty}^{\infty}\mathrm{d}t_1\cdots\int_{-\infty}^{\infty}\mathrm{d}t_n\left\langle\psi_0\middle|\widehat{\mathcal{T}}\left[\hat{c}_{\mathbf{k}}(t)\hat{c}^{\dagger}_{\mathbf{k}}(0)\widehat{\mathcal{V}}(t_1)\ldots\widehat{\mathcal{V}}(t_n)\right]\middle|\psi_0\right\rangle_C. \quad (1.79)$$

This tells us how only connected diagrams contribute in the expansion so that the sunset diagram not only becomes the only contribution at second order, but the whole expansion is composed by repetitions of it

$G_{\mathbf{k}}(t) =$ 

(1.80)

This is a much simpler form that allow us to study systems with generic electron-phonon interactions, and was obtained thanks to the mathematical insight given by the diagrammatic representation.

Non-zero temperature

The entire treatment revolved around zero-temperature Green's functions, but often we are interested in the thermodynamic properties of the system. This can be easily addressed through some simple generalizations of what was stated so far, extending the use of diagrams also to thermal averages. To see that let $\hat{O}$ be an observable quantity of the interacting system under study, we can so write its thermal average with the gran-canonical Hamiltonian $\widehat{\mathcal{K}} = \widehat{\mathcal{H}}_0 + \hat{\mathcal{V}} - \mu\widehat{\mathcal{N}}$ as

$$\langle O\rangle = \frac{1}{Z}\mathrm{Tr}\left[\hat{O}e^{-\beta\widehat{\mathcal{K}}}\right] = \frac{\mathrm{Tr}\left[\hat{O}\widehat{\mathcal{U}}_S(\beta,0)\right]}{\mathrm{Tr}\left[\widehat{\mathcal{U}}_S(\beta,0)\right]} = \frac{\mathrm{Tr}\left[e^{-\beta\widehat{\mathcal{K}}_0}\widehat{\mathcal{U}}(\beta,0)\hat{O}\right]}{\mathrm{Tr}\left[e^{-\beta\widehat{\mathcal{K}}_0}\widehat{\mathcal{U}}(\beta,0)\right]} = \frac{\left\langle\widehat{\mathcal{U}}(\beta,0)\hat{O}\right\rangle_0}{\left\langle\widehat{\mathcal{U}}(\beta,0)\right\rangle_0}. \quad (1.81)$$

Where $\exp\left(-\beta\widehat{\mathcal{K}}\right)$ was interpreted as the time evolution operator in the Schrödinger picture using imaginary time representation, as introduced in Sec. (1.4.2), and used Eq. (1.53) to obtain $\widehat{\mathcal{U}}_S(\beta,0) = \exp\left(-\beta\widehat{\mathcal{K}}_0\right)\widehat{\mathcal{U}}(\beta,0)$. Also, the cyclic property of the trace has been used. Thus, we have written the interacting average as the fraction of two non-interacting ones that, using Eq. (1.57), shows a form analogous to the GF one

$$\langle O\rangle = \frac{1}{Z}\sum_{n=0}^{\infty}\frac{(-1)^n}{n!}\int_0^{\beta}\mathrm{d}\tau_1\cdots\int_0^{\beta}\mathrm{d}\tau_n\left\langle\widehat{\mathcal{T}}\left[\hat{\mathcal{V}}(t_1)\cdots\hat{\mathcal{V}}(t_n)\right]\hat{O}\right\rangle_0. \quad (1.82)$$

At this point, the Wick's theorem can be applied to decompose the average in different contracted terms forming finite temperature unperturbed GF as defined in Eq. (1.60). Such function can be easily derived, and in the electron case we have

$$\begin{aligned} G^0_{\mathbf{k}}(\tau) = -\left\langle\widehat{\mathcal{T}}\left[\hat{c}_{\mathbf{k}}(\tau)\hat{c}^{\dagger}_{\mathbf{k}}(0)\right]\right\rangle_0 &= -\Theta(\tau)\left\langle\hat{c}_{\mathbf{k}}\hat{c}^{\dagger}_{\mathbf{k}}\right\rangle_0 e^{-(\varepsilon_{\mathbf{k}}-\mu)\tau} - -\Theta(-\tau)\left\langle\hat{c}^{\dagger}_{\mathbf{k}}\hat{c}_{\mathbf{k}}\right\rangle_0 e^{(\varepsilon_{\mathbf{k}}-\mu)\tau} \\ &= -\Theta(\tau)(1+n_{\mathbf{k}})e^{-(\varepsilon_{\mathbf{k}}-\mu)\tau} - \Theta(-\tau)n_{\mathbf{k}}e^{(\varepsilon_{\mathbf{k}}-\mu)\tau}. \end{aligned}$$

Where we used the analytic form for the time-dependent destruction operator in Eq. (1.64) with $\widehat{\mathscr{K}_0}$ instead of $\widehat{\mathscr{H}_0}$. One can understand that such result is much more complex respect the one for 0 K, but the latter can be easily restored by taking the low density limit as $n_{\mathbf{k}} \to 0$ and $\mu \to -\infty$. The phononic propagator, instead, assumes a more complex form also in such simplified regime [35]

$$D^0_{\mathbf{q}}(\tau) = -\Theta(\tau)\frac{\cosh[\omega_{\mathbf{q}}(\beta/2-\tau)]}{\sinh(\omega_{\mathbf{q}}\beta/2)} - \Theta(\tau)\frac{\cosh[\omega_{\mathbf{q}}(\beta/2+\tau)]}{\sinh(\omega_{\mathbf{q}}\beta/2)}. \tag{1.83}$$

Still, it can be seen how it remains symmetric in time and for low temperatures, $\beta \to \infty$, the form in Eq. (1.69) can be recovered. Therefore, from such simple considerations one can understand how the same results as before are still valid in this regime, meaning that the vacuum polarization can be factorized from both the numerator and Z, leaving us with

$$\langle O\rangle = \frac{\sum_{n=0}^{\infty}\frac{(-1)^n}{n!}\int_0^\beta \mathrm{d}\tau_1\cdots\int_0^\beta \mathrm{d}\tau_n \left\langle \widehat{\mathscr{T}}\left[\widehat{\mathscr{V}}(t_1)\cdots\widehat{\mathscr{V}}(t_n)\right]\widehat{O}\right\rangle_{0,C}}{\sum_{n=0}^{\infty}\frac{(-1)^n}{n!}\int_0^\beta \mathrm{d}\tau_1\cdots\int_0^\beta \mathrm{d}\tau_n \left\langle \widehat{\mathscr{T}}\left[\widehat{\mathscr{V}}(t_1)\cdots\widehat{\mathscr{V}}(t_n)\right]\right\rangle_{0,C}}. \tag{1.84}$$

Once again, only connected diagrams matters in this expansion and the rules to draw them of course depends on the system under study.

So far we have worked with the Holstein model, showing the diagrams used in such case. Now, to show how general the approach is, the Spin-Boson model will be taken under study showing how the diagrams describing its thermodynamic averages look like. We remember from Sec. (1.3.2) how the system is described by the Hamiltonian in Eq. (1.41) that can be separated as

$$\widehat{\mathscr{H}_0} = \frac{\epsilon}{2}\hat{\sigma}_z + \sum_q \omega_q \hat{b}_q^\dagger \hat{b}_q, \qquad \widehat{\mathscr{V}} = \frac{\Delta}{2}\hat{\sigma}_x + \hat{\sigma}_z\sum_q \lambda_q\left(\hat{b}_q + \hat{b}_q^\dagger\right). \tag{1.85}$$

The interaction is so composed by the sum of two parts, which can be thought of as two separate interactions: a tunneling one with strength Δ, and one with the environment controlled by λ. By inserting it in the denominator of Eq. (1.84) we can look at the expressions for the first order terms and elaborates how to treat them. Since only one interaction is present we can ignore the time ordering operator having as first order contribution

$$\left\langle\widehat{\mathscr{V}}(\tau_1)\right\rangle_0 = \frac{\Delta}{2}\langle\hat{\sigma}_x(\tau_1)\rangle_0 + \sum_q \lambda_q\left\langle\hat{\sigma}_z(\tau_1)\hat{B}_q(\tau_1)\right\rangle_0, \tag{1.86}$$

the same boson operator convention used in the GF formalism is present. Two contributions are obtained as expected, one for every of the two possible interactions. In order to see their contribution to the whole expansion we can first focus on the first one and write down explicitly its average as follows

$$\frac{\Delta}{2}\langle\hat{\sigma}_x(\tau_1)\rangle_0 = \frac{\Delta}{2}\left\langle e^{\widehat{\mathscr{H}_0}\tau_1}\hat{\sigma}_x e^{-\widehat{\mathscr{H}_0}\tau_1}\right\rangle_0 = \frac{\Delta}{2}\left\langle e^{\frac{\epsilon}{2}\hat{\sigma}_z\tau_1}\hat{\sigma}_x e^{-\frac{\epsilon}{2}\hat{\sigma}_z\tau_1}\right\rangle_0 \tag{1.87}$$

where the phononic variables simplifies since commutes with the spins ones and the exponents can cancel each other outs. In this way the average can be performed only on the spin variables, so that by choosing as a basis set the eigenvectors of $\hat{\sigma}_z$, defined by $\hat{\sigma}_z|\pm\rangle = \pm|\pm\rangle$, we can write

$$\frac{\Delta}{2}\langle\hat{\sigma}_x(\tau_1)\rangle_0 = \frac{\Delta}{2}\sum_{s=\pm}\langle s|e^{-\widehat{\mathscr{H}_0}\beta}e^{\frac{\epsilon}{2}\hat{\sigma}_z\tau_1}\hat{\sigma}_x e^{-\frac{\epsilon}{2}\hat{\sigma}_z\tau_1}|s\rangle = \frac{\Delta}{2}\sum_{s=\pm}\langle s|e^{-\frac{\epsilon}{2}\hat{\sigma}_z(\beta-\tau_1)}\hat{\sigma}_x e^{-\frac{\epsilon}{2}\hat{\sigma}_z\tau_1}|s\rangle. \tag{1.88}$$

Using the Pauli matrices representation of $\hat{\sigma}_x$ it's easy to see how its effect on the basis state is to swap them as $\hat{\sigma}_x|\pm\rangle = |\mp\rangle$, which bring us to

$$\frac{\Delta}{2}\langle\hat{\sigma}_x(\tau_1)\rangle_0 = \sum_{s=\pm}\frac{\Delta}{2}e^{-\frac{\epsilon}{2}(-s)(\beta-\tau_1)}e^{-\frac{\epsilon}{2}s\tau_1}\langle s|-s\rangle = 0. \tag{1.89}$$

As a result we obtain that no contribution is given, but a lot of insight can be still obtained from it. In fact, one can see how at all orders the spin and bosons terms inside the interactions can be factored to always obtain the multiplication of two unperturbed averages on the spin variables and on the bosonic ones. The former can always be expanded as in Eq. (1.88) having that a zero contribution is obtained if an odd number of $\hat{\sigma}_x$ are present. Thus, we obtain that **the number of tunneling interactions needs to be even** exactly in analogy to the bosonic ones as we have seen in Sec. (1.4.2). The same result is still valid also in this case as we can see in the form of the second first order term

$$\sum_q \lambda_q\left\langle\hat{\sigma}_z(\tau_1)\hat{B}_q(\tau_1)\right\rangle_0 = \sum_q\lambda_q\langle\hat{\sigma}_z(\tau_1)\rangle_0\left\langle\hat{B}_q(\tau_1)\right\rangle_0 = \sum_q\lambda_q\langle\hat{\sigma}_z(\tau_1)\rangle_0\sum_{n_{q'}}\left\langle n_{q'}\middle|\hat{B}_q\middle|n_{q'}\right\rangle. \tag{1.90}$$

Where $\left\langle n_{q'}\middle|\hat{B}_q\middle|n_{q'}\right\rangle$ will give zero as for the zero temperature case. Therefore, we arrive at the conclusion that **only terms with an even number of phononic and field interactions have a non-zero contribution**. Still, this is not the only insight that can be obtained from these equations. By looking carefully at Eq. (1.89) one can notice how the expression already resembles the ones seen in the GF expansion. In fact, can be seen how the time dependence of the spin variables inserts GF-like terms in the expression interpretable as the time spin propagator $G_s(\tau) = \Theta(\tau)\exp\left(-\frac{\epsilon}{2}s\tau\right)$. The step function is present since only forward propagation is possible in this formalism due to time-ordering. At last, since phonons degrees of freedom interacts with the state through $\hat{\sigma}_z$ and not through momentum exchange we have that the resulting terms with a phonon will always have a weight of type

$$\left\langle\widehat{\mathcal{T}}[\hat{\sigma}_x(\tau_1)\cdots\hat{\sigma}_z(\tau_i)\hat{\sigma}_z(\tau_j)]\right\rangle_0\sum_{qq'}\lambda_q\lambda_{q'}\left\langle\widehat{\mathcal{T}}[\hat{B}_q(\tau_i)\hat{B}_{q'}(\tau_j)]\right\rangle_0 = \langle\cdots\rangle_0\sum_q|\lambda_q|^2D_q^0(\tau_i-\tau_j). \tag{1.91}$$

That separation allow us to insert such summation in the definition of the propagator itself, and by taking the continuous limit we can use the spectral density in Eq. (1.42) to have

$$\mathscr{D}(\tau) = \int_{\mathbb{R}}\mathrm{d}q\,J(\omega_q)D_q^0(\tau) = \frac{\alpha\omega_c}{2}\int_0^1\mathrm{d}q\,q^s\frac{\cosh[\omega_c q(\beta/2-\tau)]}{\sinh\left(\omega_c q\beta/2\right)}. \tag{1.92}$$

Where Eq. (1.43) has been used to represent $J(\omega)$ and the phononic spectra, and once again τ is assumed positive since contribution for negative is the same thanks to symmetry. Taking that into account a set of rules analogous to the one for the electron-phonon interaction diagrams can be stated as follows

1. The spin propagators $G_s(\tau'-\tau)$ are solid lines if the state is up, $s=1$, or dashed lines if it is down, $s=-1$, going from τ' to τ assuming that $\tau'>\tau$;

2. The tunneling interactions $\hat{\sigma}_x(\tau)$ are drawn as points in time that add to the expression a $\Delta/2$ term and flips the spin state. It needs to be an even number;

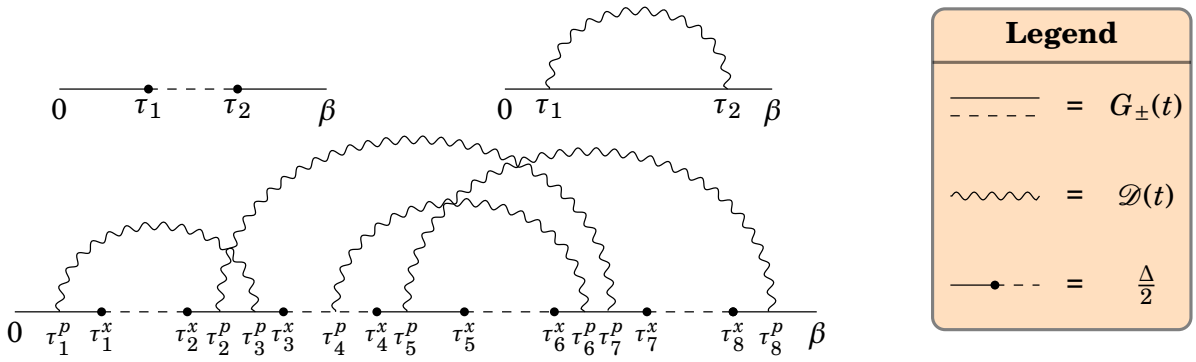


Figure 1.4: Examples of Feynman diagrams for the Spin-Boson model. On top the two second-order diagrams are represented and a more complex generic 16-th order one on the bottom, to the right the legend with the analytic expression is reported.

3 Phonons are described by wiggly lines that give a contribution $\mathscr{D}(\tau'-\tau)$ along with adding a sign given by the multiplication of the spin state at those times, $\hat{\sigma}_z(\tau')\hat{\sigma}_z(\tau)$.

Using this set of rules all the diagrams for such system can be drawn as in Fig. (1.4), allowing us to obtain the expression of all the contributions to the expansion just by constructing a specific type of diagram.

One may still argue that the description done to obtain the diagram so far was based on the expansion of the denominator, but also a numerator is present inside Eq. (1.84). Such term differs from the former by only the presence of the $\widehat{O}$ operator which is usually diagonal in the base of $\widehat{\mathscr{H}}_0$, and so a contribution coming from every diagram can be evaluated. To make an example we can study the magnetization inside the Spin-Boson model by setting $\widehat{O}=\hat{\sigma}_z(\tau)$ and obtain

$$\mathrm{Tr}\left[\hat{\sigma}_z^H(\tau)e^{-\widehat{\mathscr{K}}\tau}\right]=\mathrm{Tr}\left[e^{\widehat{\mathscr{K}}\tau}\hat{\sigma}_z e^{-\widehat{\mathscr{K}}\tau}e^{-\widehat{\mathscr{K}}\beta}\right]=\mathrm{Tr}\left[e^{-\widehat{\mathscr{K}}(\beta-\tau)}\hat{\sigma}_z e^{-\widehat{\mathscr{K}}\tau}\right], \tag{1.93}$$

where the operator is expressed in the Heisenberg picture so that the time dependence was given by the whole grand-canonical Hamiltonian. Then we can add the exponential related to the unperturbed part to obtain

$$\mathrm{Tr}\left[\hat{\sigma}_z^H(\tau)e^{-\widehat{\mathscr{K}}\tau}\right]=\mathrm{Tr}\left[\widehat{\mathscr{U}}_S(\beta,\tau)\hat{\sigma}_z\widehat{\mathscr{U}}_S(\tau,0)\right]=Z_0\left\langle\widehat{\mathscr{U}}(\beta,\tau)\hat{\sigma}_z(\tau)\widehat{\mathscr{U}}(\tau,0)\right\rangle_0, \tag{1.94}$$

where the relation $\widehat{\mathscr{U}}_S(\tau,\tau')=e^{-\tau\widehat{\mathscr{H}}_0}\widehat{\mathscr{U}}(\tau,\tau')e^{\tau'\widehat{\mathscr{H}}_0}$ was used. In this way the variation to the diagrams' contribution becomes clear, first we propagate to the wanted point in time, then $\hat{\sigma}_z$ is evaluated taking the state of the diagram at time τ to then continue the propagation. Such result is usually written in a more compact form as a sum over all the diagrams configurations $\mathscr{C}$

$$\langle\hat{\sigma}_z(\tau)\rangle=\frac{\sum_{n=0}^{\infty}\int_0^\beta \mathrm{d}\tau_1\cdots\int_0^\beta \mathrm{d}\tau_n\sum_{\mathscr{C}_n}s(\tau)W(\mathscr{C}_n)}{\sum_{n=0}^{\infty}\int_0^\beta \mathrm{d}\tau_1\cdots\int_0^\beta \mathrm{d}\tau_n\sum_{\mathscr{C}_n}W(\mathscr{C}_n)}=\frac{\sum\!\!\!\!\!\!\int_{\mathscr{C}}s(\mathscr{C})W(\mathscr{C})}{\sum\!\!\!\!\!\!\int_{\mathscr{C}}W(\mathscr{C})}. \tag{1.95}$$

Where the term $(-1)^n/n!$ vanishes since only even orders are present and combinatorial properties simplify the factorial. Also, we introduced a set of symbles to simplify the

notation: $\mathscr{C}_n$ for the connected diagrams of a certain order, $W(\mathscr{C})$ describe the weight of the diagram computed using the known rules, and $\sum\!\!\!\!\!\!\int$ to indicate that the presence of both sums and integration over the finite and real variables of the diagrams respectively. The final result in Eq. (1.95) is really general and can be generalized for whatever observable $\widehat{O}$ by placing the right function $O(\mathscr{C})$ instead of $s(\mathscr{C})$.

2

Diagrammatic Monte Carlo

Feynman's diagrams expansions can be used in order to estimate several quantities in different fields of quantum mechanics using the same mathematical framework. Which, leaves us with an abstract form that can be used to express a general quantity $Q(\mathbf{y})$, depending on a set of variables $\mathbf{y}$, by decomposing it into [44]

$$Q(\mathbf{y}) = \sum_{n=0}^{\infty} \sum_{\mathbb{T}} \int d\mathbf{x}_1 \cdots \int d\mathbf{x}_n Q(\mathbf{y}, n; \mathbf{x}_1, \ldots, \mathbf{x}_n) W(\mathbf{y}, n, \mathbb{T}; \mathbf{x}_1, \ldots, \mathbf{x}_n). \tag{2.1}$$

Q is so turned into a sum of contributions coming from the observable and diagrams weights W, both depending on *external variables*, such as $\mathbf{y}$ and order n, and *internal* ones, as $\mathbf{x}_i$, and topology $\mathbb{T}$. Performing such integration is of course a challenging problem which cannot be tackled analytically. Over the years different computational approaches have been created in order to numerically compute such quantities [48, 36], but all poses some limits to either system sizes or precision. In this chapter we will introduce a computational approach able to generally compute such expansions by an approximation-free Monte Carlo algorithm taking the name of *Diagrammatic Monte Carlo* (DMC). We first discuss the basic aspects of general Monte Carlo integration to then specialize it in the case of the Holstein GF integration and for the Spin-Boson thermal averages.

2.1 Monte Carlo integration

Monte Carlo integration is a computational technique that uses random numbers in order to numerically integrate functions in general dimensions [37]. In order to understand how this is can be done we can take a general function $f : D \to \mathbb{R}$, with $D \subset \mathbb{R}^M$, and a *probability density function* (PDF) $p : D \to \mathbb{R}^+$ and write

$$\int_D d\mathbf{x}\, f(\mathbf{x}) = \int_D d\mathbf{x}\, \frac{f(\mathbf{x})}{p(\mathbf{x})} p(\mathbf{x}) = \left\langle \frac{f}{p} \right\rangle_p, \qquad \int_D d\mathbf{x}\, p(\mathbf{x}) = 1. \tag{2.2}$$

The integral has been rewritten as the expectation value of an observable $f(\mathbf{x})/p(\mathbf{x})$ over the p distribution. This allow us to compute it by approximating the average as the

arithmetic mean over a set of samples $\{\mathbf{x}_i\}_{i=1}^N$ distributed following p

$$\left\langle \frac{f}{p} \right\rangle_p = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}_i)/p(\mathbf{x}_i) \approx \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}_i)/p(\mathbf{x}_i), \qquad \mathbf{x}_i \sim p. \tag{2.3}$$

Such approach allows for an estimation of the integral with a precision given by the variance of the estimation that, by assuming the samples to be *independent* between each other, becomes

$$\sigma^2 = \frac{1}{N} \left[\left\langle \left(\frac{f}{p} \right)^2 \right\rangle_p - \left\langle \frac{f}{p} \right\rangle_p \right] = \frac{1}{N^2} \sum_{i=0}^{N} \left[\left(\frac{f(\mathbf{x}_i)}{p(\mathbf{x}_i)} \right)^2 - \frac{f(\mathbf{x}_i)}{p(\mathbf{x}_i)} \right]. \tag{2.4}$$

Now, this quantity can be decreased in two ways: increase N generating more statistic, or by selecting a PDF that facilitate the convergence. In fact, can be demonstrated [45] that a distribution that minimize σ keeping N fixed exist and is $p(\mathbf{x}) = f(\mathbf{x})/\int_D f$, assuming f as positive. So, the best choice is to sample directly from a distribution with the same form of the function itself. In order to do that we are going to introduce the concept of *Markov Chain* (MC) that will allow us to obtain samples from a general function at the cost of correlation between them.

Markov Chain Monte Carlo

Let $\{X_1, \ldots, X_N\}$ be a sequence of random variables in $\mathbb{R}^M$, we define such a sequence to be a Markov Chain if the conditional probabilities associated to every X_i satisfy

$$P(X_i | X_{i-1}, \ldots, X_1) = P(X_i | X_{i-1}). \tag{2.5}$$

That is, the probability of obtaining a certain value for X_i depends only on the random variable immediately before it and not the others. For simplicity, we are going to assume that X_i can only possess a discrete number of values, so that $X_i \in \{\mathbf{x}_1, \ldots, \mathbf{x}_j, \ldots\}$. In this way we can describe the probability distribution and the conditional probabilities of such variables using a vector and a matrix respectively

$$P(X_i) = \left[p^{(i)}(\mathbf{x}_1), \ldots, p^{(i)}(\mathbf{x}_j), \ldots \right], \qquad P(X_i | X_j) = \begin{pmatrix} p(\mathbf{x}_1, \mathbf{x}_1) & \cdots & p(\mathbf{x}_1, \mathbf{x}_k) \\ \vdots & \ddots & \vdots \\ p(\mathbf{x}_k, \mathbf{x}_1) & \cdots & p(\mathbf{x}_k, \mathbf{x}_k) \end{pmatrix}. \tag{2.6}$$

Where one can see how inside the matrix the function that defines the entries, P_{ij}, does not depend on which variables in the sequence we are using thanks to the Markovian properties of the chain. One can also see how these associations give some interesting properties to both $\mathbf{p}^{(i)}$ and $\mathbf{P}$, having that $\forall i$ they satisfy

$$\sum_j p_j^{(i)} = 1, \qquad \sum_k P_{ij} = 1, \tag{2.7}$$

thanks to the normalization properties of PDFs. Now, let's imagine starting the sequence from a variable with a known distribution $\mathbf{p}^{(0)}$. Thanks to the knowledge of $\mathbf{P}$ it's easy to understand how the distribution for the next X_i in the chain can be obtained

simply by applying it to the vector density. A process that can be repeated giving a recursive way of describing the statistic of the chain

$$p_j^{(i+1)} = \sum_k P_{ik} p_k^{(i)}. \tag{2.8}$$

For this reason $\mathbf{P}$ is also called *transition matrix* and, under very general condition, completely defines the properties of the chain. In order to see these we need to assume that the matrix is *ergodic*, so that $P_{ij} > 0$ for every entry allowing every possible values inside the space we are studying to appear. Under such condition, is possible to demonstrate [18] that the sequence defined by Eq. (2.8) reaches a stationary point $\mathbf{p}$ independent of $\mathbf{p}^{(0)}$. In general the form of $\mathbf{p}$ is unknown, but an equation defining it can be obtained as

$$P_{ij} p_j = P_{ji} p_i \quad \text{where} \quad p_i = \sum_j P_{ij} p_j. \tag{2.9}$$

This equation is referred to as the *detailed balance condition* and can tell us at which PDF will the samples from the chain converge to.

Such mathematical framework is really flexible, powerful, and can be generalized to continuous variables by simply substituting summation with integration. In this way, $\mathbf{p}$ will become a PDF and $\mathbf{P}$ a kernel function $\pi : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^+$ that defines the chain as

$$p^{(i+1)}(\mathbf{x}) = \int_{\mathbb{R}^N} \mathrm{d}\mathbf{y} \pi(\mathbf{x},\mathbf{y}) p^{(i)}(\mathbf{y}), \qquad \pi(\mathbf{x},\mathbf{y}) p(\mathbf{y}) = \pi(\mathbf{y},\mathbf{x}) p(\mathbf{x}). \tag{2.10}$$

Knowing that what we want to achieve is a form for π that allows to converge the chain to a selected p. Such problem was studied first by Metropolis [37] and then generalized by Hastings [19] obtaining at the end a general solution given by

$$\pi(\mathbf{x},\mathbf{y}) = \Gamma(\mathbf{x} \to \mathbf{y}) \mathscr{A}(\mathbf{x},\mathbf{y}), \qquad \mathscr{A}(\mathbf{x},\mathbf{y}) = \min\left\{1, \frac{\Gamma(\mathbf{y} \to \mathbf{x}) f(\mathbf{y})}{\Gamma(\mathbf{x} \to \mathbf{y}) f(\mathbf{x})}\right\}. \tag{2.11}$$

Where $\Gamma(\mathbf{x} \to \mathbf{y})$ is an *update function* meaning how we can go from $\mathbf{x}$ to $\mathbf{y}$ and output the probability of such process, while $\mathscr{A}$ is usually called *acceptance ratio* and f a non-negative function. It's possible to demonstrate that by assuming

$$\Gamma(\mathbf{x} \to \mathbf{y}) > 0, \qquad \int \mathrm{d}\mathbf{y} \Gamma(\mathbf{x} \to \mathbf{y}) = 1, \tag{2.12}$$

the transition kernel in Eq. (2.11) is ergodic and satisfies the detailed balance condition

$$\pi(\mathbf{x},\mathbf{y}) \frac{f(\mathbf{y})}{\int \mathrm{d}\mathbf{x} f(\mathbf{x})} = \pi(\mathbf{y},\mathbf{x}) \frac{f(\mathbf{x})}{\int \mathrm{d}\mathbf{x} f(\mathbf{x})}. \tag{2.13}$$

This means that by evolving the elements in the chain through such kernel we will obtain that a sequence of samples $\{\mathbf{x}_i\}_{i=1}^N$ where after a certain t large enough, also called *thermalization time*, we will have

$$\mathbf{x}_i \sim \frac{f(\mathbf{x})}{\int \mathrm{d}\mathbf{x} f(\mathbf{x})} \quad \text{for} \quad i > t. \tag{2.14}$$

Thus, such idea constitutes a general way of sampling from every possible probability distribution by simply know it's unnormalized form f.

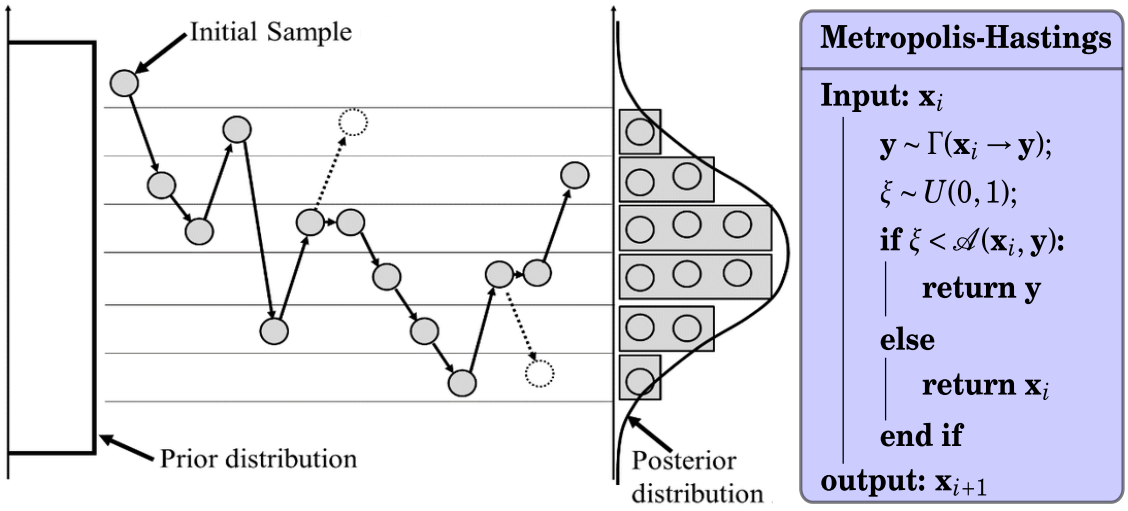


Figure 2.1: Graphical representation of the application of Metropolis-Hastings algorithm transforming a general prior distribution to a latent one, on the left (adapted from [54]). On the right a schematic representation of the general algorithm to perform a step inside the chain.

The last thing left to understand is how we can recreate the process of applying $\pi(\mathbf{x},\mathbf{y})$ on a computational level in order to obtain general sampling. Fortunately, thanks to the freedom left to the choice of Γ, it can be done in a really simple way that also becomes really cheap to perform. The idea is to start from a sample $\mathbf{x}_0$ taken from the *prior distribution* $p^{(0)}$ and modify it locally to propose a new sample $\mathbf{y}$ that can become $\mathbf{x}_1$ with a probability $\mathscr{A}(\mathbf{x}_0,\mathbf{y})$. For example, one can do the following

$$\mathbf{y} = \mathbf{x}_0 + \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \frac{1}{\sqrt{2\pi}} e^{-\boldsymbol{\epsilon}^2/2}, \tag{2.15}$$

obtaining a new sample proposal that deviates little from the previous one thanks to a Gaussian displacement. Then, a random number ξ is sampled from a uniform distribution in the unitary interval, $U(0,1)$, and used to determine the next element in the chain as

$$\mathbf{x}_1 = \begin{cases} \mathbf{y} & \xi < \mathscr{A}(\mathbf{x}_0,\mathbf{y}) \\ \mathbf{x}_0 & \xi > \mathscr{A}(\mathbf{x}_0,\mathbf{y}) \end{cases}, \qquad \mathscr{A}(\mathbf{x}_0,\mathbf{y}) = \frac{e^{-\boldsymbol{\epsilon}^2/2} f(\mathbf{y})}{e^{-(-\boldsymbol{\epsilon})^2/2} f(\mathbf{x}_0)} = \frac{f(\mathbf{y})}{f(\mathbf{x}_0)}. \tag{2.16}$$

Defining a computational routine that will mimic the effect of applying the Metropolis-Hasting kernel constructed using $\Gamma(\mathbf{x} \to \mathbf{y}) = \exp\left(-(\mathbf{y}-\mathbf{x})^2/2\right)$ to achieve a sampling from the *posterior distribution* f. Of course this is only an example, and the possible choices for Γ are infinite, but the main idea behind the process remains the same for all of them giving rise to the *Metropolis-Hastings algorithm* (MH) that we will use extensively as reported in Fig. (2.1).

Correlation and convergence

Using Markov Chains to sample from a general function always comes at a price, and that price is *correlation*. To understand what we mean by that we shall first describe in

more details how to compute errors inside a Monte Carlo integration. Therefore, we will look at a situation where we want to numerically evaluate an observable and its error in the following way

$$\langle O \rangle = \int \mathrm{d}\mathbf{x} O(\mathbf{x}) p(\mathbf{x}), \qquad \sigma_O^2 = \langle O^2 \rangle - \langle O \rangle^2, \tag{2.17}$$

with p a general PDF. Let now $\{\mathbf{x}_i\}_{i=1}^M$ be a thermalized Markov Chain that samples from p, we can obtain estimates for both the quantities from the sequence as

$$\bar{O} = \frac{1}{M} \sum_{i=1}^{M} O_i, \qquad \sigma_{\bar{O}}^2 = \langle \bar{O}^2 \rangle - \langle \bar{O} \rangle, \tag{2.18}$$

where we used $O_i = O(\mathbf{x}_i)$. It is possible to understand how $\bar{O}$ will converge to the real mean in the limit of $M \to \infty$ thanks to the *central limit theorem*. In fact, using it we can identify the arithmetic mean as a random variable distributed as a Gaussian with the following form [18]

$$f_{\bar{O}}(x) = \frac{1}{\sqrt{2\pi\sigma_{\bar{O}}^2}} e^{-(\langle O \rangle - x)^2 / 2\sigma_{\bar{O}}^2}. \tag{2.19}$$

Showing how we obtain a more reliable result as $\sigma_{\bar{O}}$ becomes lower, and to achieve that we can give a form to it as

$$\sigma_{\bar{O}}^2 = \frac{1}{M^2} \left\langle \sum_{ij} \left(O_i O_j - \langle O_i \rangle \langle O_j \rangle \right) \right\rangle = \frac{1}{M^2} \left\langle \sum_{i} \left(O_i^2 - \langle O_i \rangle^2 \right) + 2 \sum_{j>i} \left(O_i O_j - \langle O_i \rangle \langle O_j \rangle \right) \right\rangle. \tag{2.20}$$

By defining the *correlation function* as

$$\chi_{|i-j|} \equiv \langle O_i O_j \rangle - \langle O_i \rangle \langle O_j \rangle, \tag{2.21}$$

we can see how the variance related to the arithmetic mean can be rewritten from Eq. (2.20) into

$$\sigma_{\bar{O}}^2 = \frac{\sigma_O^2}{M} [1 + 2\tau_O], \qquad \tau_O = \sum_{k=1}^{M-1} \left(1 - \frac{k}{M} \right) \frac{\chi_k}{\chi_0}, \tag{2.22}$$

where τ_O is usually called *correlation time*. This shows us how in the limit $M \to \infty$ the variance tends to zero, having that Eq. (2.19) becomes a delta distribution on the real mean and so $\bar{O}$ converges to the right result. Still, the rate at which this convergence happen depends on how large the numerator is, and consequently on σ_O and χ_k. The former is a statistical property of the observable and cannot be touched, while the correlation χ_k depends on how the samples are generated. In the best case scenario all the samples are independent so that

$$\chi_k = \langle O_i O_{i+k} \rangle - \langle O_i \rangle \langle O_{i+k} \rangle = \langle O_i \rangle \langle O_{i+k} \rangle - \langle O_i \rangle \langle O_{i+k} \rangle = 0, \tag{2.23}$$

but inside a MC the sample O_i is generated by modification of the previous one so that their product mean cannot be separated. This property propagates along the chain giving rise to a non-zero correlation function that decrease as k increase, so that some

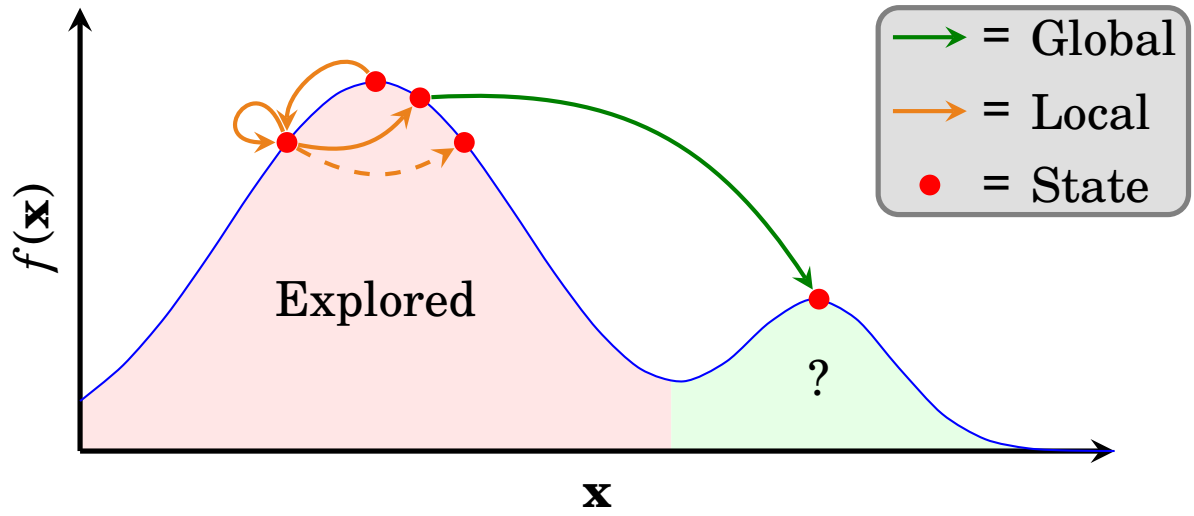


Figure 2.2: Representation of the exploration of the domain of a multidimensional function $f(\mathbf{x})$ using the MCMC algorithm. It's possible to see how the effect of the local updates are only able to move the state by little steps giving the possibility of getting stuck limiting the exploration, while global updates can move freely even for complicated functions.

statistical information is loss. In particular, a set of M samples drawn independently bring the same information and error as $M(1+2\tau_O)$ samples from a Markov Chain.

Knowing that, we may want to understand how to reduce at most the samples' correlation to improve the convergence of our algorithm. To see exactly how to do that we can study the probability of having k consecutive rejections in the Metropolis-Hastings algorithm

$$\rho(k) = \left\langle \prod_{i=1}^{k} \mathbb{1}_{\text{rej}}(i) \right\rangle \approx \frac{1}{M-k} \sum_{j=1}^{M-k} \prod_{i=1}^{k} \mathbb{1}_{\text{rej}}(j+i). \tag{2.24}$$

This quantity can be understood as a correlation function itself since the numerical estimator for χ_k/χ_0 take the form [56]

$$\frac{\chi_k}{\chi_0} = \frac{\frac{1}{M-k}\sum_{i=1}^{M-k}\left(O_i-\bar{O}\right)\left(O_{i+k}-\bar{O}\right)}{\frac{1}{M}\sum_{i=1}^{M}\left(O_i-\bar{O}\right)^2}. \tag{2.25}$$

Not only Eq. (2.24) is consistent with such definition, but can be demonstrated that $\chi_k/\chi_0 \to \rho(k)$ in the limit of $M \to \infty$ [1]. A result that tells us how to maximize the efficiency inside our Markov Chain Monte Carlo integration we need to focus on finding an update function Γ that maximize $\mathscr{A}$. And it's easy to see how the best possible scenario is given by $\Gamma(\mathbf{x} \to \mathbf{y}) = p(\mathbf{y})$, which gives

$$\mathscr{A}(\mathbf{x},\mathbf{y}) = \min\left\{1, \frac{\Gamma(\mathbf{y}\to\mathbf{x})p(\mathbf{y})}{\Gamma(\mathbf{x}\to\mathbf{y})p(\mathbf{x})}\right\} = \min\left\{1, \frac{p(\mathbf{x})p(\mathbf{y})}{p(\mathbf{y})p(\mathbf{x})}\right\} = 1. \tag{2.26}$$

In this case no correlation will be present, and the samples obtained can be thought as sampled directly from the right distribution. Nevertheless, updates of this type are difficult to construct since involves the modification of several entries of $\mathbf{x}$ if not the entire vector, and for this are called *global updates*. Instead, *local updates* are usually used by slight modifications of single entries inside $\mathbf{x}$ that allow for simpler and efficient routines that insert correlation in the chain.

On a more practical level the difference between the two approach can be seen in Fig. (2.2), where a sampling from a two mode function is depicted. Can be seen how the local updates can only move the state of the chain by little getting stuck on local maxima of the target function f. This is due to $\mathscr{A}$ becoming small when we try to move from a high point to a lower one, creating situations where only one part of the all phase space is explored efficiently, a known problem that takes the name of *critical slowing down* [55]. One can thus understand why correlations arises, and more samples are needed for convergence. On the other hand, using global updates the state can make large jumps that will always be accepted, bringing to a more complete and efficient exploration that has a faster convergence. For these reasons we shall generally prefer to work with global updates when we have the possibility.

2.2 Integrating diagrammatic expansions

Inside Sec. (1.4) we have seen how interacting Green's functions and thermal averages can be written as a sum over connected diagrams contributions as in Eq. (2.1). These type of expansions can be viewed under a Monte Carlo perspective by making the following connection

$$\langle O(\mathbf{y})\rangle = \frac{\sum\!\!\!\!\!\!\int_{\mathscr{C}} O(\mathscr{C})W(\mathscr{C})}{\sum\!\!\!\!\!\!\int_{\mathscr{C}} W(\mathscr{C})} = \sum\!\!\!\!\!\!\int_{\mathscr{C}} O(\mathscr{C})\left(\frac{W(\mathscr{C})}{\sum\!\!\!\!\!\!\int_{\mathscr{C}} W(\mathscr{C})}\right) = \langle O(\mathbf{y})\rangle_W\,, \tag{2.27}$$

where we used the same form as in Eq. (1.95). That shows how a thermal average can be seen as a mean over the distribution of the diagrams, a quantity of which we can have an exact numeric estimation by employing what we have seen in Sec. (2.1). In particular, we can create a Markov Chain of diagrams $\{\mathscr{C}_i\}_{i=1}^{M}$ sampled following their weight function in order to obtain

$$\langle O(\mathbf{y})\rangle \approx \frac{1}{M}\sum_{i=1}^{M} O(\mathscr{C}_i), \qquad\qquad \mathscr{C}_i \sim W(\mathbf{y}, n, \mathbb{T}; \mathbf{x}_1, \ldots, \mathbf{x}_n). \tag{2.28}$$

The only thing that is left to complete the workflow is finding a suitable update function $\Gamma(\mathscr{C} \to \mathscr{C}')$ that allow us to efficiently and ergodically sample the diagrams specific to the model under study.

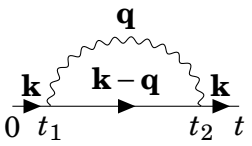
Here we will introduce the standard series of local updates that is used for the study of the single site Holstein model and the Spin-Boson model. We will also show how for the latter a problem arises from the oscillating form of the diagrams' weight giving rise to the infamous *sign problem*.

Single site Holstein

Let's focus on the simplified version of the Holstein Hamiltonian introduced in Eq. (1.25) that we rewrite here

$$\widehat{\mathscr{H}} = \varepsilon \hat{c}^\dagger \hat{c} + \Omega \sum_{\mathbf{q}} \hat{b}^\dagger_{\mathbf{q}} \hat{b}_{\mathbf{q}} + \frac{g}{\sqrt{N}} \sum_{\mathbf{q}} \hat{c}^\dagger \hat{c} \left(\hat{b}_{\mathbf{q}} + \hat{b}^\dagger_{-\mathbf{q}}\right). \tag{2.29}$$

Such form highly simplifies the diagrams in the expansion since no phononic or electronic dispersion is present. In this way the electrons can only have one value of the momentum associated to the energy ϵ, so that we should not account for $\mathbf{k}$ inside the electronic lines. Also, since no correlation between phonons and electron momenta is present we can directly integrate the former inside the weight. To understand it better we can have a look at the weight of the sunset diagram in this regime



$$\begin{aligned} &= \frac{g^2}{N}\sum_{\mathbf{q}} D_{\mathbf{q}}(\tau_2-\tau_1)G^0_{\mathbf{k}}(t_1)G^0_{\mathbf{k}-\mathbf{q}}(t_2-t_1)G^0_{\mathbf{k}}(t-t_2) \\ &= \frac{g^2}{N}\sum_{\mathbf{q}} e^{-\Omega(\tau_2-\tau_1)}e^{-\varepsilon\tau_1}e^{-\varepsilon(\tau_2-\tau_1)}e^{-\varepsilon(\tau-\tau_2)} \\ &= g^2 e^{-\Omega(\tau_2-\tau_1)}e^{-\varepsilon\tau}. \end{aligned}$$

No $\mathbf{k}$ or $\mathbf{q}$ dependence is present in the final weight, allowing us to say that the only variables that we need to represent are the creation and annihilation times of the phonons in the system. This also allows us to easily evaluate the weight of a certain diagram by looking simply at the number of phonons present n, their start b_i and end e_i, and the electron's time of flight τ

$$W(\tau,n;b_1,e_1,\dots,b_n,e_n) = g^{2n}e^{-\varepsilon\tau}\prod_{i=1}^{n} e^{-\Omega(e_i-b_i)}, \tag{2.30}$$

which will be the final form we will use in our computations.

From the form of W we can start thinking at an effective update function $\Gamma(\mathscr{C}\to\mathscr{C}')$, able to sample from the whole diagrams' space, to use inside the Metropolis-Hasting algorithm. In general a single local update cannot be ergodic, making it necessary to use a series of updates that changes one part of the diagram each $\{\Gamma_i\}_{i=1}^N$, all normalized as in Eq. (2.12). In this way an ergodic update can be constructed from them by summing them up as

$$\Gamma(\mathscr{C}\to\mathscr{C}') = \frac{1}{N}\Gamma_1(\mathscr{C}\to\mathscr{C}')+\cdots+\frac{1}{N}\Gamma_N(\mathscr{C}\to\mathscr{C}') = \frac{1}{N}\sum_{i=0}^{N}\Gamma_i(\mathscr{C}\to\mathscr{C}'). \tag{2.31}$$

One can see how a function defined in this way is still and update since will still have $\Gamma(\mathscr{C}\to\mathscr{C}')>0$ and

$$\sum\!\!\!\!\!\!\int_{\mathscr{C}'}\Gamma(\mathscr{C}\to\mathscr{C}') = \frac{1}{N}\sum_{i=0}^{N}\sum\!\!\!\!\!\!\int_{\mathscr{C}'}\Gamma_i(\mathscr{C}\to\mathscr{C}') = \frac{1}{N}\sum_{i=1}^{N}1 = 1. \tag{2.32}$$

Therefore, we are constructing an ergodic update from a series of non-ergodic ones, which translate in practice into choosing one update at random, $k\sim U(1,N)$, and sampling $\mathscr{C}'$ from the selected one, $\mathscr{C}'\sim\Gamma_k(\mathscr{C}\to\mathscr{C}')$. Thus, we can focus on the construction of simple updates that changes the diagrams variables one at a time, and in this case only three of them are needed to reach ergodicity:

Change τ. The simplest possibility is to change the time of flight of the electron, and to do that we need to choose a new value τ' so that $\tau'\in[e_n,\infty]$ in order to respect time

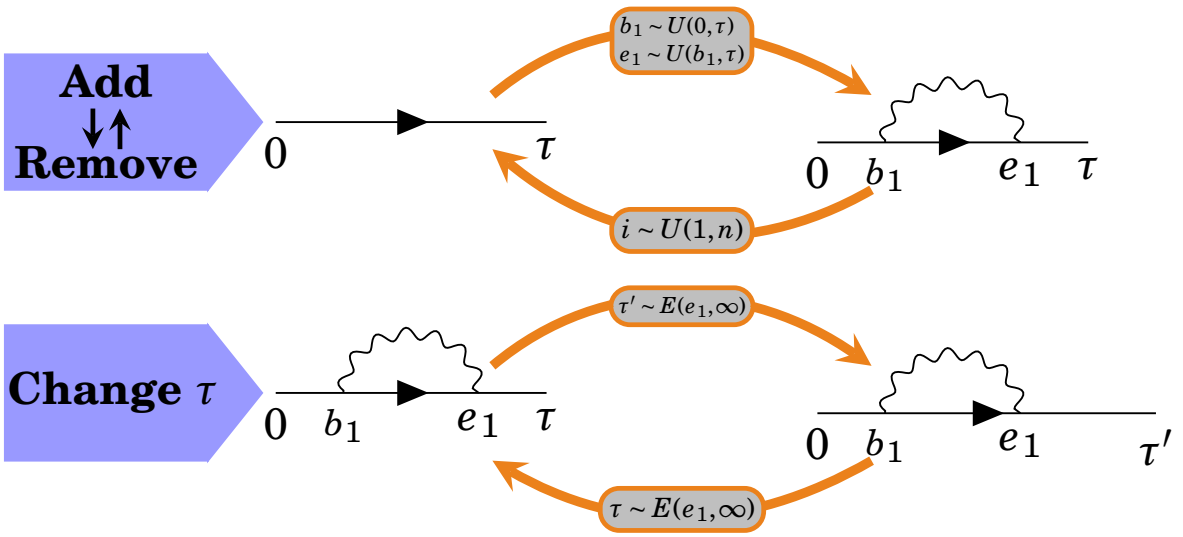


Figure 2.3: Representation of the updates inside the single site Holstein model, with the Add and Remove update working together to satisfy the detailed balance on top and the change τ on the bottom.

ordering. The best way to do that is by drawing it from an exponential distribution using the *inverse CDF* method, described in App. (A). So by using it on $p(\tau) = \exp(-\varepsilon\tau)$, after normalizing it inside the domain $[e_n, \infty]$, the inverse CDF would be

$$\tau' = e_n - \frac{1}{\varepsilon}\ln(u). \tag{2.33}$$

In this way $\tau' \sim E(e_n, \infty)$ and the update function becomes $\Gamma_{chg}(\tau \to \tau') = \exp\big(-\varepsilon\tau'\big)/C$, with C the normalizing constant. Considering that we can also easily compute the acceptance rate to see its effectiveness as

$$\mathscr{A}(\tau,\tau') = \frac{\Gamma_{chg}(\tau' \to \tau)W(\tau', n; \dots)}{\Gamma_{chg}(\tau \to \tau')W(\tau, n; \dots)} = \frac{e^{-\varepsilon\tau}e^{-\varepsilon\tau'}}{e^{-\varepsilon\tau'}e^{-\varepsilon\tau}} = 1, \tag{2.34}$$

a perfect acceptance rate is obtained, meaning that the sampling is performed directly from the exact diagrams' distribution. Also, it's important to point out how this update has the capability of returning to the same point, meaning that both $\Gamma_{chg}(\tau \to \tau')$ and $\Gamma_{chg}(\tau' \to \tau)$ are > 0. Updates of such kind are called *self balanced*, and we will soon see with the next updates that often updates works in couples since one may be able to go in one direction but not coming back.

Add and Remove n. The next step is to change the number of phonons inside the diagram, and to do that two updates are used: one for adding one phonon, and one for removing it. It's easy to understand how one update balance the other since the former will have $\Gamma_{add}(n \to n+1) > 0$ but $\Gamma_{add}(n+1 \to n) = 0$ not being able to balance itself, while the other one is the contrary. Now, the implementation of the two updates is generally really simple. For adding a phonon what can be done is uniformly drawing the beginning and the end of the phonon as

$$b_{n+1} \sim U(0,\tau), \qquad e_{n+1} \sim U(b_{n+1},\tau), \tag{2.35}$$

where it can be seen how $e_{n+1} > b_{n+1}$ respecting the time ordering. For removing it, instead, one can easily randomly select one phonon as $i \sim U(1,n)$ and eliminating it. In this way we will have the following expressions for the updates functions

$$\Gamma_{add}(n \to n+1) = \tau^{-1}(\tau - b_{n+1})^{-1}, \qquad \Gamma_{rem}(n \to n-1) = n^{-1}, \tag{2.36}$$

with the constraint $\Gamma_{rem}(0 \to -1) = 0$ for obvious reasons. Thus, we can once again obtain the acceptance rates of the two type of updates analytically as follows

$$\mathscr{A}(n \to n+1) = \frac{\Gamma_{rem}(n+1 \to n)W(\tau, n+1;\dots)}{\Gamma_{add}(n \to n+1)W(\tau, n;\dots)} = g^2 e^{-\Omega(e_{n+1}-b_{n+1})}\frac{\tau(\tau - b_{n+1})}{n+1}, \tag{2.37}$$

$$\mathscr{A}(n \to n-1) = \frac{\Gamma_{add}(n-1 \to n)W(\tau, n-1;\dots)}{\Gamma_{rem}(n \to n-1)W(\tau, n;\dots)} = g^{-2} e^{\Omega(e_i - b_i)}\frac{n}{\tau(\tau - b_i)}. \tag{2.38}$$

Where we can see how the acceptance rate is different from unity, meaning that these are the updates that will generate correlation inside our chain.

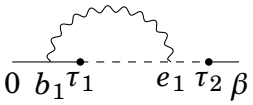
Now, it's easy to demonstrate how these updates, summarized in Fig. (2.3), satisfy the wanted normalization condition and together are able to generate an ergodic $\Gamma(\mathscr{C} \to \mathscr{C}')$. Which we will use inside our study to sample from the single site Holstein distribution.

Spin-Boson

Once again, we shall start the study of the update function by remembering the model Hamiltonian that we want to study, and for the Spin-Boson model we obtain in Sec. (1.3.2) the form

$$\widehat{\mathscr{H}} = \frac{\Delta}{2}\hat{\sigma}_x + \sum_q \omega_q \hat{b}_q^\dagger \hat{b}_q + \hat{\sigma}_z \sum_q \lambda_q \left(\hat{b}_q + \hat{b}_q^\dagger\right). \tag{2.39}$$

Where the separation between the ground state has been set to zero, $\epsilon = 0$, following the approach of [9]. Before trying to find out a good set of updates for this model we need to notice that all that has been said so far assumed $W > 0$. This is easy to understand, because if $W \in \mathbb{C}$ then we could not interpret $\mathscr{A}$ as a probability and the Metropolis-hastings algorithm cannot be performed. With that being said, one should remember how the diagrams' weight for the SB system, even though not possessing complex phases, can be negative



$$= s(b_1)s(e_1)\left(\frac{\Delta}{2}\right)^2 \mathscr{D}(e_1 - b_1) = -\left(\frac{\Delta}{2}\right)^2 \mathscr{D}(e_1 - b_1). \tag{2.40}$$

Where the spin propagators were set to unity, $G_s(\tau) = 1$, since $\epsilon = 0$. In this way we are not able to create a Markov chain that directly sample from W, but we can decompose it into it's sign and modulus in order to rewrite Eq. (2.27) as

$$\langle O(\mathbf{y})\rangle = \frac{\sum\!\!\!\!\!\!\int_{\mathscr{C}} O(\mathscr{C})W(\mathscr{C})}{\sum\!\!\!\!\!\!\int_{\mathscr{C}} W(\mathscr{C})} = \frac{\sum\!\!\!\!\!\!\int_{\mathscr{C}} O(\mathscr{C})S(\mathscr{C})|W(\mathscr{C})|}{\sum\!\!\!\!\!\!\int_{\mathscr{C}} S(\mathscr{C})|W(\mathscr{C})|} = \frac{\langle OS\rangle_{|W|}}{\langle S\rangle_{|W|}}. \tag{2.41}$$

Therefore, we can overcome the problem by performing a Monte Carlo average on the modulus of the diagrams' weight at the cost of having to evaluate the fraction of two

oscillating averages. This needed trade off takes the name of *sign problem* in literature since the convergence of Eq. (2.41) is exponentially worse respect to the normal case [33]. Still, this is the only way we have available to compute such averages, and so we shall understand how to sample from $|W|$ in this case.

Knowing all of this we can start approaching the construction of the right set of updates in order to achieve ergodicity in a way analogous to the one used in the single site Holstein case. In particular, for the SB diagrams the variables that we are interested to modify are the number of tunneling interaction n_x and the number of phonons in the system n_p. No variation on the β variable of the diagram is needed here since that would mean change the temperature of the system and one is mainly interested in the property at a selected one. Once we have taken that into account, it's easy to understand that the type of updates that we need in total are two add and remove couple to change n_p and n_x.

Add_p and $Remove_p$. The approach is similar to the one discussed for the Holstein model. So that we go from n_p to n_p+1 by sampling the beginning and end of the new phonon from uniform distributions as in Eq. (2.35), while to remove one of them is randomly selected as $i \sim U(1, n_p)$. This would lead to the same update functions as in Eq. (2.36), but a change is present inside the acceptance rates since the propagator has changed leading to

$$\mathscr{A}(n_p \to n_p+1) = \frac{\Gamma_{rem_p}(n_p+1 \to n_p)\left|W(\beta, n_p+1;\dots)\right|}{\Gamma_{add_p}(n_p \to n_p+1)\left|W(\beta, n_p;\dots)\right|} = \frac{\beta(\beta - b_{n_p+1})}{n_p+1}\mathscr{D}(e_{n_p+1} - b_{n_p+1}), \tag{2.42}$$

$$\mathscr{A}(n_p \to n_p-1) = \frac{\Gamma_{add_p}(n_p-1 \to n_p)\left|W(\beta, n_p-1;\dots)\right|}{\Gamma_{rem_p}(n_p \to n_p-1)\left|W(\beta, n_p;\dots)\right|} = \frac{n_p}{\beta(\beta - b_i)}\frac{1}{\mathscr{D}(e_i - b_i)}. \tag{2.43}$$

In this way we can sample through the different bosonic variables of the diagram following the modulus of the weight, as we described before.

Add_x and $Remove_x$. Even if the idea is similar to the one for the bosonic degrees of freedom here we need to keep in mind that the number of interaction needs to be even. This means that we will need to add two interaction at a time, so that the idea is to select one of the interaction time as $i \sim U(0, n_x)$, with $\tau_0 = 0$, and sample the two time in between $[\tau_i, \tau_{i+1}]$ as described in Fig. (2.4). For the elimination instead we can still use the same approach as before by selecting an interaction as $i \sim U(1, n_x - 1)$ and then eliminate τ_i and τ_{i+1}. In this way one can easily understand how the updates functions will become

$$\Gamma_{add_x}(n_x \to n_x+2) = [(n_x+1)(\tau_{i+1} - \tau_i)(\tau_{i+1} - b)]^{-1}, \quad \Gamma_{rem_x}(n_x \to n_x-2) = (n_x-1)^{-1}, \tag{2.44}$$

once again we put $\Gamma_{rem_x}(0 \to -2) = 0$. At last, the acceptance rates can be computed

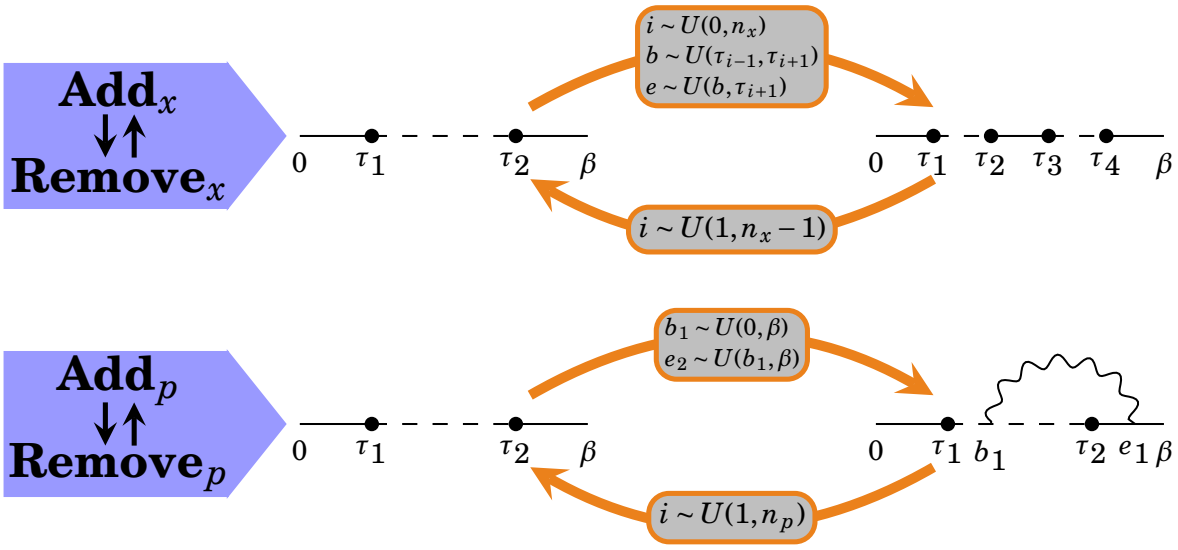


Figure 2.4: Representation of the updates used inside the Spin-Boson model, both of them are add and remove update that balance each other. On top the one that modify the number of tunneling interaction adding two at a time, and on the bottom the one that changes the number of phonons in the system.

also for such updates obtaining the simple form of

$$\mathscr{A}(n_x \to n_x + 2) = \frac{\Gamma_{rem_x}(n_x + 2 \to n_x)\left|W(\beta, n_x + 2; \ldots)\right|}{\Gamma_{add_x}(n_x \to n_x + 1)\left|W(\beta, n_x; \ldots)\right|} = \left(\frac{\Delta}{2}\right)^2 (\tau_{i+1} - \tau_i)(\tau_{i+1} - b), \tag{2.45}$$

$$\mathscr{A}(n_x \to n_x - 2) = \frac{\Gamma_{add_x}(n_x - 2 \to n_x)\left|W(\beta, n_x - 2; \ldots)\right|}{\Gamma_{rem_x}(n_x \to n_x - 2)\left|W(\beta, n_x; \ldots)\right|} = \left(\frac{\Delta}{2}\right)^{-2} \frac{1}{(\tau_{i+2} - \tau_{i-1})(\tau_{i+2} - \tau_i)}. \tag{2.46}$$

Allowing us to modify ergodically the spin variables inside the diagrams.

Thus, also here one can demonstrate how such updates are normalized so that summing them up all together we will give rise to an ergodic $\Gamma(\mathscr{C} \to \mathscr{C}')$ that will allow us to obtain a Markov Chain with samples from $|W|$.

2.3 Observable

After understanding how to sample diagrams from the wanted distribution we only need to define the observable that we are interested in and construct the functions $O(\mathscr{C})$ to use in the MC average. We have considered the following observables in the two models under study: Green function $G(\tau)$ and polaron energy ε_p for the single site Holstein, and the magnetic susceptibility $\chi(\beta)$ in the Spin-Boson. For the first two the same approach used in [38] will be used to introduce the concept of *exact estimator* for DMC. While, the susceptibility will be obtained from simple results of *linear response theory* that can be easily translated into Monte Carlo form.

Green function

The exact diagrammatic expansion of $G(\tau)$ was introduced inside Sec. (1.4.3), and using the weights obtained in Sec. (2.2.1) we can limit the integration to the form

$$G(\tau) = \sum_{n=1}^{\infty} \int_0^{\tau} \mathrm{d}b_1 \int_{b_1}^{\tau} \mathrm{d}e_1 \cdots \int_0^{\tau} \mathrm{d}b_n \int_{b_n}^{\tau} \mathrm{d}e_n W(n, \tau; b_1, e_1 \ldots b_n, e_n). \tag{2.47}$$

Now, this analytic equation it's not the best option for our DMC computational protocol. The reason for it is that the Green's function written as in Eq. (2.47) would correspond to the normalization factor of the diagram distribution at a fixed time of flight

$$p(\mathscr{C}_\tau) = W(\mathscr{C}_\tau)/G(\tau), \qquad G(\tau) = \sum\!\!\!\!\!\!\int_{\mathscr{C}_\tau} W(\mathscr{C}_\tau). \tag{2.48}$$

This implies that, to reconstruct the whole GF we would need to perform a series of simulations where τ is kept fixed, $\mathscr{C}_\tau$, by not using the *change* τ update, and obtain an estimate for the normalization constant of the distribution at different diagram lengths. We can avoid the need of multiple simulation by simply notice that the equation can be rewritten using the Dirac delta function as

$$G(\tau) = \sum_{n=1}^{\infty} \int_0^{\infty} \mathrm{d}\tau' \int_0^{\tau'} \mathrm{d}b_1 \int_{b_1}^{\tau'} \mathrm{d}e_1 \cdots \int_0^{\tau'} \mathrm{d}b_n \int_{b_n}^{\tau'} \mathrm{d}e_n \delta(\tau - \tau') W(n, \tau'; b_1, e_1 \ldots b_n, e_n). \tag{2.49}$$

Now a further integration of the external variable τ is present allowing us to simply rewrite the all expression as a DMC average as

$$G(\tau) = C \left\langle \delta(\tau - \tau') \right\rangle_W, \qquad C = \sum\!\!\!\!\!\!\int_{\mathscr{C}} W(\mathscr{C}). \tag{2.50}$$

In this way, we can reconstruct the form of the function by collecting the average number of diagram with a certain value of τ inside the complete simulation that also updates the time of flight. Nevertheless, in a numerical simulation it's not possible to exactly evaluate $\delta(\tau - \tau')$ due to finite precision of floating numbers. That bring us to perform a simple approximation for it called *histogram method*. Instead of the delta function we can use the indicator function to estimate $G(\tau)$ as follows

$$G(\tau) \approx C \left\langle \mathbb{1}_\Delta(\tau - \tau') \right\rangle_W, \qquad \mathbb{1}_\Delta(x) = \begin{cases} 1 & x \in [-\Delta/2, \Delta/2] \\ 0 & x \notin [-\Delta/2, \Delta/2] \end{cases}, \tag{2.51}$$

with the approximation becoming better as $\Delta \to 0$. Using this estimator is, therefore, equivalent to take the range in which we want to evaluate the Green's function, $[0, \tau_M]$, and divide it in a set of bins with width Δ and centers $\{\tau_i\} = [\Delta/2, 3\Delta/2, \ldots, (1+2i)\Delta/2, \ldots]$. Then, a histogram $\{h_i\}$ is constructed during the simulation by counting the numbers of diagrams possessing a τ inside the i-th bin as in Fig. (2.5). Following this procedure we will obtain that

$$G(\tau_i) \approx C \left\langle \mathbb{1}_\Delta(\tau - \tau') \right\rangle_W = C h_i / M, \tag{2.52}$$

with M the total number of diagrams sampled. We only need to find out a form for C/M to obtain an unbiased estimate of the whole $G(\tau)$. That can be done easily by using a

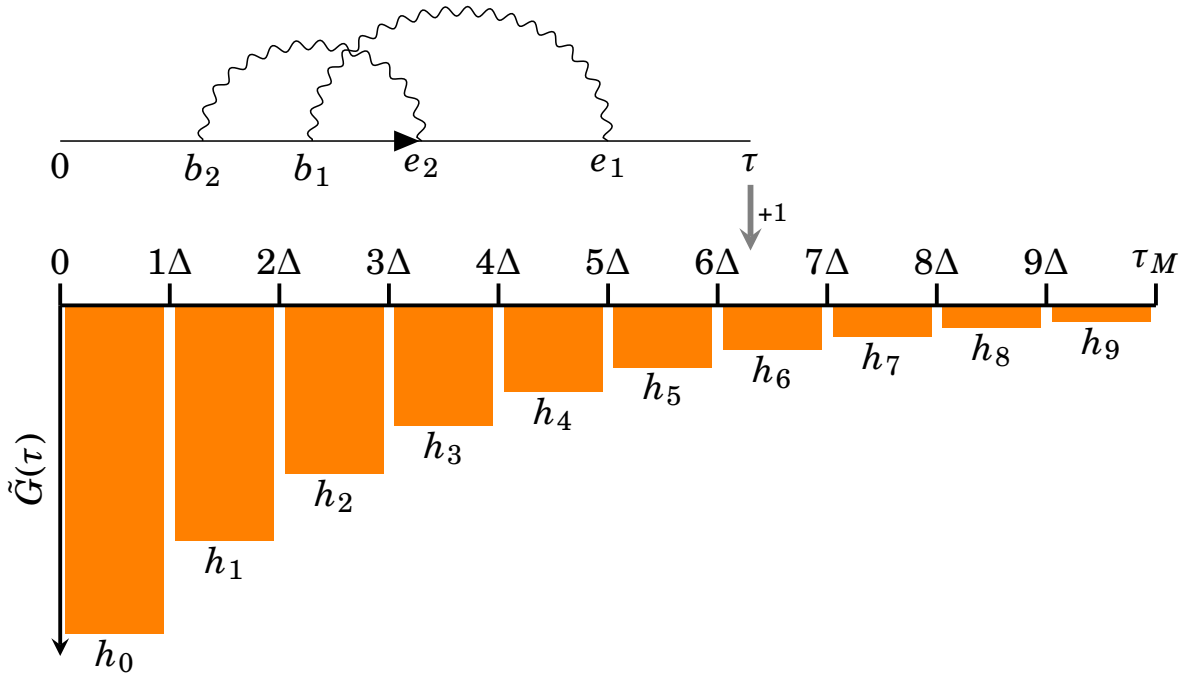


Figure 2.5: Histogram method to construct the unnormalized Green's function. Here we can see the binned axis and how one diagram is confronted with it so that the h_i corresponding to the one where τ falls in is increased by one.

known property of all the Green's functions of being unity at zero propagation time, in fact

$$G(0) = -\left\langle \mathcal{T}\left[\hat{c}(0)\hat{c}^\dagger(0)\right]\right\rangle = -\left\langle \hat{c}^\dagger\hat{c}\right\rangle + \langle \mathbb{1}\rangle = 1. \tag{2.53}$$

This is especially interesting for us because using the knowledge of the unnormalized GF at the different centers with $\{h_i\}$ we can extrapolate its value at zero using a linear form of type

$$\tilde{G}(\tau) \approx b + a\tau \quad \text{for} \quad \tau \ll 1. \tag{2.54}$$

Where $\tilde{G}$ was used to describe the unnormalized GF approximated by $\{h_i\}$. By our computation we can impose the following conditions

$$\tilde{G}(\Delta/2) \approx h_0, \qquad \tilde{G}(3\Delta/2) = h_1, \tag{2.55}$$

giving a linear system that allows us to compute b giving its value at zero to obtain

$$\tilde{G}(0) = \frac{M}{C}G(0) = \frac{M}{C} = h_0 - \frac{h_1 - h_0}{2}. \tag{2.56}$$

Therefore, by constructing the diagram during a single simulation we are able to obtain an evaluation of the complete $G(\tau)$ that becomes more accurate as we set Δ smaller, giving us arbitrary accuracy.

Polaron energy

As we have seen inside Sec. (1.4.2) the ground state information of the polaronic state are contained inside the long time behavior of the polaron propagator. In particular,

we have seen how in the general case an exponential behavior of the kind $\exp\left(-\varepsilon^0_{\mathbf{k}}\tau\right)$ is retrieved from it. Thus, we already have a choice for the estimator given by fitting $G(\tau)$, obtained as in Sec. (2.3.1), to an exponential behavior for sufficiently large times. Nevertheless, this approach brings an error in the final result that is difficult to evaluate through statistic alone. For this reason, *Mishchenko et. al.* [38] introduced a mathematical procedure that allows for the creation of unbiased observable able to extrapolate that behavior more easily.

Let us consider the log time behaviour of the ratio between the propagator with a stretched form of it where τ is expanded by a small factor $1+\lambda$

$$\lim_{\tau\to\infty}\frac{G_{\mathbf{k}}((1+\lambda)\tau)}{G_{\mathbf{k}}(\tau)}=\frac{e^{-\varepsilon^0_{\mathbf{k}}(1+\lambda)\tau}}{e^{-\varepsilon^0_{\mathbf{k}}\tau}}=e^{-\varepsilon^0_{\mathbf{k}}\lambda\tau}. \tag{2.57}$$

We can get an expression for the left side of the equation by looking at how the expansion changes as

$$G_{\mathbf{k}}((1+\lambda)\tau)=\sum_{n=1}^{\infty}\int_0^{(1+\lambda)\tau}\mathrm{d}b_1\cdots\int_{b_n}^{(1+\lambda)\tau}\mathrm{d}e_n W(n,(1+\lambda)\tau;b_1,\ldots,e_n), \tag{2.58}$$

where W is the weight of the diagrams in the general Holstein model, which can be easily expressed by the formula

$$W(n,\tau;b_1,\ldots,e_n)=g^{2n}\prod_{i=1}e^{-\varepsilon_{\mathbf{k}_i}\Delta\tau_i}\prod_{j=1}^{n}e^{-\omega_{\mathbf{q}_j}(e_j-b_j)}. \tag{2.59}$$

Where n is the number of phonons and the sum on i runs over all electronic propagators. Inside this general context we can simplify Eq. (2.58) by making a change of variable and set $\tau=(1+\lambda)\bar{\tau}$ so that one obtains

$$G_{\mathbf{k}}((1+\lambda)\tau)=\sum_{n=1}^{\infty}\int_0^{\bar{\tau}}\mathrm{d}\bar{b}_1\cdots\int_{\bar{b}_n}^{\bar{\tau}}\mathrm{d}\bar{e}_n\overline{W}(\mathscr{C}),\qquad \overline{W}(\mathscr{C})=(1+\lambda)^{2n}W(n,\bar{\tau};\bar{b}_1,\ldots,\bar{e}_n). \tag{2.60}$$

Knowing that we can substitute the expression inside the ratio in order to obtain it as a Monte Carlo average of the following type

$$\frac{G_{\mathbf{k}}((1+\lambda)\tau)}{G_{\mathbf{k}}(\tau)}=\frac{\sum\!\!\!\!\!\!\int_{\mathscr{C}}\overline{W}(\mathscr{C})}{\sum\!\!\!\!\!\!\int_{\mathscr{C}'}W(\mathscr{C}')}=\sum\!\!\!\!\!\!\int_{\mathscr{C}}\frac{\overline{W}(\mathscr{C})}{W(\mathscr{C})}\frac{W(\mathscr{C})}{\sum\!\!\!\!\!\!\int_{\mathscr{C}'}W(\mathscr{C}')}=\left\langle\frac{\overline{W}(\mathscr{C})}{W(\mathscr{C})}\right\rangle_W. \tag{2.61}$$

This allows us to focus on the ration between weights and see how can be written in a simple form as

$$\frac{\overline{W}(\mathscr{C})}{W(\mathscr{C})}=(1+\lambda)^{2n}\prod_i e^{-\lambda\varepsilon_{\mathbf{k}_i}\Delta\tau_i}\prod_{j=1}^{n}e^{-\lambda\omega_{\mathbf{q}_j}(e_j-b_j)}=(1+\lambda)^{2n}\exp\left[-\lambda\sum_i\varepsilon_{\mathbf{k}_i}\Delta\tau_i-\lambda\sum_{j=1}^{n}\omega_{\mathbf{q}_j}\nu_j\right], \tag{2.62}$$

where the j-th phonon length was indicated as ν_j. Having this form we can simply take λ as small as we want allowing us to Taylor expand the expression at first order and obtain

$$\begin{aligned}\frac{\overline{W}(\mathscr{C})}{W(\mathscr{C})}&=\left(1+2n\lambda+\mathscr{O}(\lambda^2)\right)\left[1-\lambda\left(\sum_i\varepsilon_{\mathbf{k}_i}\Delta\tau_i+\sum_{j=1}^{n}\omega_{\mathbf{q}_j}\nu_j\right)+\mathscr{O}(\lambda^2)\right]\\&=1-\lambda\left(\sum_i\varepsilon_{\mathbf{k}_i}\Delta\tau_i+\sum_{j=1}^{n}\omega_{\mathbf{q}_j}\nu_j-2n\right)+\mathscr{O}(\lambda^2).\end{aligned} \tag{2.63}$$

By doing the same with the exponential behavior at long time of flight so that $\exp\left(-\lambda\varepsilon^0_{\mathbf{k}}\tau\right) = 1-\lambda\varepsilon^0_{\mathbf{k}}\tau+\mathcal{O}(\lambda^2)$ and insert both of them inside Eq. (2.57) we will get

$$\varepsilon^0_{\mathbf{k}} = \lim_{\tau\to\infty}\left\langle \sum_i \varepsilon_{\mathbf{k}}\frac{\Delta\tau_i}{\tau} + \sum_j \omega_{\mathbf{q}}\frac{\nu_j}{\tau} - \frac{2n}{\tau}\right\rangle_W = \lim_{\tau\to\infty}\langle E(\mathscr{C})\rangle_W\,. \tag{2.64}$$

Where the limit for $\lambda\to 0$ was performed to obtain the final expression. This expression is revealing us a way to obtain an unbiased estimate for the ground state energy of our polaronic system by simply sampling from the whole distribution and compute the average so that

$$\varepsilon^0_{\mathbf{k}} \approx \frac{1}{M}\sum_{\tau>\tau_m} E(\mathscr{C}_\tau). \tag{2.65}$$

So that we let also τ change, but we evaluate the observable only for τ large enough with a threshold set using a constant τ_m.

Inside the single site Holstein model Eq. (2.64) gets simplified a lot, since $\varepsilon_{\mathbf{k}} = \varepsilon$ and $\omega_{\mathbf{q}} = \Omega$, becoming

$$\varepsilon^0 = \varepsilon + \lim_{\tau\to\infty}\left\langle \Omega\sum_j \frac{\nu_j}{\tau} - \frac{2n}{\tau}\right\rangle_W. \tag{2.66}$$

By confronting it with the known ground state energy of that model given by $\varepsilon+\varepsilon_p$ we can obtain a direct estimator for the polaron binding energy in that simple approximation as

$$\varepsilon_p = \lim_{\tau\to\infty}\left\langle \Omega\sum_j \frac{\nu_j}{\tau} - \frac{2n}{\tau}\right\rangle_W. \tag{2.67}$$

Which will be the unbiased estimator that we are going to work with.

Magnetization

In magnetic systems the physical observable carrying the majority of the information is generally the magnetic susceptibility $\chi(\tau)$. Such quantity is usually studied through the formalism on *linear response theory*, where is possible to connect it to the spin-spin time correlation function as

$$\chi(\tau) = \langle\hat{\sigma}_z(\tau)\hat{\sigma}_z(0)\rangle - \langle\hat{\sigma}_z(\tau)\rangle\langle\hat{\sigma}_z(0)\rangle\,. \tag{2.68}$$

This function has been widely used in physics mainly to detect phase transitions. In fact, its static value $\chi(0)$ has the same properties of an order parameter, and sudden changes with temperature indicate a modification of the system state generating critical behaviors. Thus, we will focus on the description of the static magnetic susceptibility to study the critical behaviors of the Spin-Boson model

$$\chi = \left\langle M^2\right\rangle - \langle M\rangle^2\,. \tag{2.69}$$

Where we simply made the association of $\langle M\rangle = \langle\hat{\sigma}_z\rangle$ and also for its squared value. Therefore, we shall focus on the discussion of their evaluation within the version of the SB model of our interest where we placed the external field ϵ to zero.

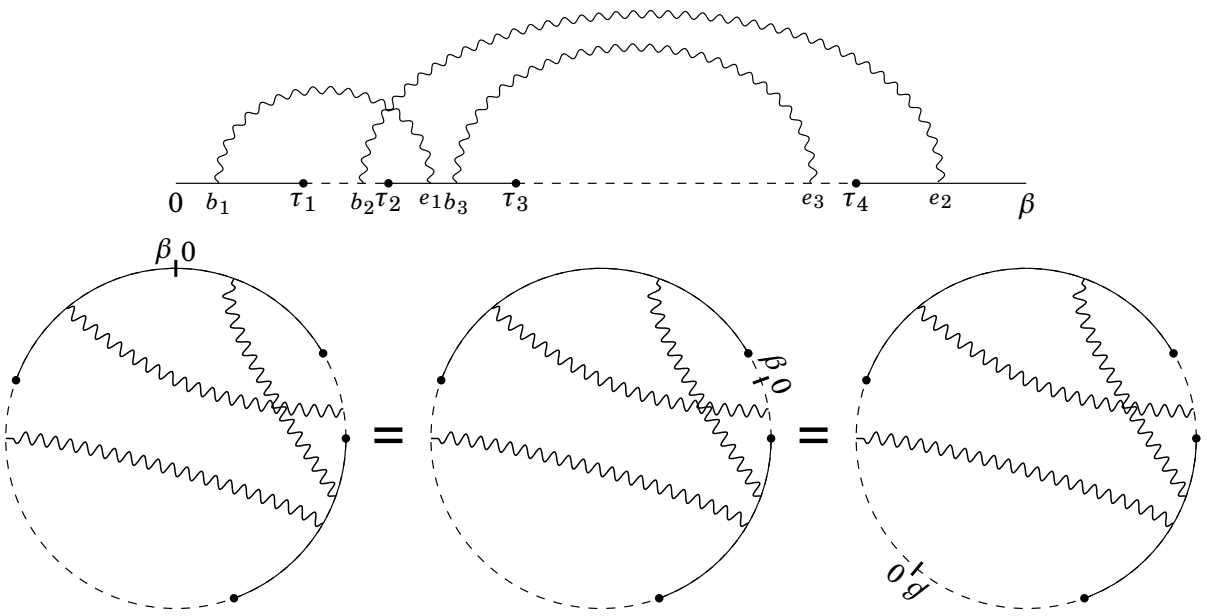


Figure 2.6: A diagram for the Spin-Boson model drawn as defined on top and presented using a circular form by connecting the end. In the circular form is easier to see how by changing the position of the bar defining the point where the circle was connected we obtain another diagram with the same weight.

To evaluate the magnetization inside our system we can use the same approach introduced inside Eq. (1.95), but a better statistical alternative for it can be obtained. The idea is to notice how the weight of the diagram is invariant under time translation. That can be seen in Fig. (2.6) where a circular representation for the diagram is used showing how shifting the starting point of the diagram, and so rotating the connection point in the circle, the weight of the configuration doesn't change. This means that for every diagram generated we can identify a set of diagrams $\{\mathscr{C}_i\}$ that have the same statistical weight as the sampled one. The contribution from such set is really easy to write for the magnetization observable $\hat{\sigma}_z(0)$ and is given by

$$M_i = \sum_i s(\tau_i) p_W(\mathscr{C}_i) = \left[s(0) + s(\mathrm{d}\tau) + \cdots + s(\beta)\right] p_W(\mathscr{C}) = \left[\int_0^\beta \mathrm{d}\tau s(\tau)\right] p_W(\mathscr{C}). \tag{2.70}$$

Basically, by using that p_W is the same for every diagram in the set we obtained that the contribution of the set is given by the integral sum of the spin state along the sampled configuration. This allows us to rewrite the sum changing the contribution of every diagram to an average that gives the same result as before

$$M_i = \sum_i \left[\frac{1}{\beta}\int_0^\beta \mathrm{d}\tau s(\tau)\right] p_W(\mathscr{C}_i) = \left[\frac{1}{\beta}\int_0^\beta \mathrm{d}\tau s(\tau)\right] p_W(\mathscr{C}_i) \sum_i = \left[\int_0^\beta \mathrm{d}\tau s(\tau)\right] p_W(\mathscr{C}), \tag{2.71}$$

where we used that the size of the set is simply given by $\int_0^\beta \mathrm{d}\tau = \beta$. In this way the contribution of every diagram contains information from the all set leading to more statistical information obtained for every sample. For this reason the best choice we can

make for the magnetization observable is the following

$$\langle M\rangle = \left\langle \frac{1}{\beta}\int_0^\beta \mathrm{d}\tau \hat{\sigma}_z(\tau)\right\rangle, \qquad \langle M^2\rangle = \left\langle \left(\frac{1}{\beta}\int_0^\beta \mathrm{d}\tau \hat{\sigma}_z(\tau)\right)^2\right\rangle. \tag{2.72}$$

Using them the static magnetic susceptibility can be computed with much better result than in the naive way.

Taking into consideration the case under study where ϵ is zero one can also notice another thing. Inside the weight, as described in Eq. (2.40), no contribution changes if we switch the initial state of the diagram. That is, if we take a diagram starting in spin down $\mathscr{C}_-$ then we have $p_{|W|}(\mathscr{C}_-) = p_{|W|}(\mathscr{C}_+)$. This result gives us the possibility of constructing a larger set of diagram with equal probability given by $\{\mathscr{C}_{i-}, \mathscr{C}_{i+}\}$. Thus, by keeping into account how the size of set now is 2β we can evaluate the total contribution to the magnetization of this group as

$$M_i = \sum_i \left[\frac{1}{\beta}\int_0^\beta \mathrm{d}\tau s(\tau)\right] p_{|W|}(\mathscr{C}_{i+}) - \sum_i \left[\frac{1}{\beta}\int_0^\beta \mathrm{d}\tau s(\tau)\right] p_{|W|}(\mathscr{C}_{i-}), \tag{2.73}$$

where the subtraction appears since the whole spins are flipped between the two subsets. By doing as before we can see how

$$M_i = \left[\int_0^\beta \mathrm{d}\tau s(\tau)\right] p_{|W|}(\mathscr{C}_+)(1-1) = 0, \tag{2.74}$$

namely, the magnetization of the system needs to go to zero for no external field. From this result we can also see how we can skip the computation of $\langle M\rangle$ in practice and focus on $\langle M^2\rangle$, for which we obtain that

$$M_i^2 = \left[\int_0^\beta \mathrm{d}\tau s(\tau)\right]^2 p_{|W|}(\mathscr{C}_+)(1+1) = \left[\int_0^\beta \mathrm{d}\tau s(\tau)\right]^2 2p_{|W|}(\mathscr{C}_+). \tag{2.75}$$

Meaning that we can simply take into account only diagrams starting with spin since using the same idea one can see how

$$2p_{|W|}(\mathscr{C}_+) = 2\frac{|W(\mathscr{C}_+)|}{\sum\!\!\!\!\!\!\int_{\mathscr{C}} |W(\mathscr{C})|} = 2\frac{|W(\mathscr{C}_+)|}{\sum\!\!\!\!\!\!\int_{\mathscr{C}_-} |W(\mathscr{C}_-)| + \sum\!\!\!\!\!\!\int_{\mathscr{C}_+} |W(\mathscr{C}_+)|} = \frac{|W(\mathscr{C}_+)|}{\sum\!\!\!\!\!\!\int_{\mathscr{C}_+} |W(\mathscr{C}_+)|}. \tag{2.76}$$

So that a statistics with only diagrams with starting spin up can be used for the evaluation, cutting the studied phase space in half. Therefore, by remembering how the SB model has sign problem we can write down the final form of the estimator of the squared magnetization, and so of the susceptibility, by using Eq. (2.41) and insert the form of O found out here having

$$\langle M^2\rangle = \frac{1}{\langle S\rangle_{|W|_+}} \left\langle \left(\frac{1}{\beta}\int_0^\beta \mathrm{d}\tau \hat{\sigma}_z(\tau)\right)^2 S\right\rangle_{|W|_+}. \tag{2.77}$$

Where $|W|_+$ was used to say that only spin up diagrams are needed in the computations and S represent the sign of the diagram.

3

Normalizing Flow

We have discussed how the fastest way to converge a Monte Carlo simulation is to compute averages from samples taken from a distribution proportional to the target function. Nevertheless, sampling from such general distribution is generally considered too complex employing analytic exact methods. Still, we can aim to sample from an approximated form of such complex distribution trading a small error in the final result with a trackable PDF. Such idea is at the core of the Machine Learning (ML) architectures called *generative models*. In recent years several types of generative architectures have been proposed, but we will focus on one of the most general and powerful between them called *flow-based generative model* or *normalizing flow* (NF). NF has been extensively studied and used mostly in the context of image processing [51, 3] with some interesting applications to quantum chemistry [39] and lattice field theory [1, 46]. Here, instead, we want to focus on the use of such architecture for describing the Feynman diagrams distribution inside the diagrammatic Monte Carlo framework. In order to do that an introduction to NF mathematical base and current architecture is presented, following the general review of *Papamakarios et. al.* [40]. Subsequently, we will introduce the main ideas that will allow the approximation of diagrams distribution.

3.1 Approximating general PDF

The normalizing flow architecture has its foundation in one of the most known result in mathematical analysis, the change of variable formula. To be more specific, in its application to probability theory, giving the possibility to map one initial distribution $p_{\mathbf{z}} : \mathbb{R}^D \to \mathbb{R}^+$ to another one $p_{\mathbf{x}} : \mathbb{R}^D \to \mathbb{R}^+$. The way in which this is done is by taking a *diffeomorphism* $T : \mathbb{R}^D \to \mathbb{R}^D$ so that the following relation is true

$$\int_{\mathbb{R}^D} \mathrm{d}\mathbf{z}\, p_{\mathbf{z}}(\mathbf{z}) = \int_{\mathbb{R}^D} \mathrm{d}\mathbf{x}\, p_{\mathbf{z}} \circ T^{-1}(\mathbf{x}) \left|\det J_{T^{-1}}(\mathbf{x})\right| = \int_{\mathbb{R}^D} \mathrm{d}\mathbf{x}\, p_{\mathbf{x}}(\mathbf{x}). \tag{3.1}$$

Where the change of variable formula in high dimensions was used, and from the result can be seen how through the use of T the PDF $p_{\mathbf{z}}$ has been mapped into

$$p_{\mathbf{x}}(\mathbf{x}) = p_{\mathbf{z}} \circ T^{-1}(\mathbf{x}) \left|\det J_{T^{-1}}(\mathbf{x})\right|, \qquad \mathbf{x} = T(\mathbf{z}). \tag{3.2}$$

This result is giving us a lot of information since, assuming that $p_{\mathbf{z}}$ is a simple distribution from which we are able to sample, we can find out both the value of $p_{\mathbf{x}}$ and a way to sample from it. The former directly given by Eq. (3.2) and the latter simply obtained by sampling $\mathbf{z} \sim p_{\mathbf{z}}$ and then take $\mathbf{x} = T(\mathbf{z})$ so that $\mathbf{x} \sim p_{\mathbf{x}}$. With that $p_{\mathbf{x}}$ is completely known and can be used to compute averages or to use as an update for a Markov Chain.

The final aim of NF is to approximate the transformation mapping one initial simple distribution to a complex target one, using a Neural Network (NN). Thus, one may argue that if such transformation exist in the first place and then how we can create a model effectively able to learn the form of a probability distribution. Here we shall first show the existence of T for every choice of $p_{\mathbf{z}}$ and $p_{\mathbf{x}}$ possible, and then introduce the basics concepts behind NF architecture.

Expressiveness of transformations

To show that a diffeomorphism $T : \mathbb{R}^D \to \mathbb{R}^D$ mapping $p_{\mathbf{z}}$ and $p_{\mathbf{x}}$ exists in general, one should start by setting as a target distribution

$$p_{\mathbf{x}}(\mathbf{x}) = 1, \qquad p_{\mathbf{x}} : (0,1)^D \to \mathbb{R}^+. \tag{3.3}$$

Basically, we want to map a general PDF to the uniform distribution on the open hypercube of dimension D. In order to do that we assume all conditional probabilities $p_{\mathbf{z}}(z_i < c|\mathbf{z}_{<i})$ of $p_{\mathbf{z}}$ to be differentiable respect to $(z_i, \mathbf{z}_{<i})$, where we used z_i to indicate the i-th component of $\mathbf{z}$. In this way, we can use the chain rule of probability in order to decompose the distribution as

$$p_{\mathbf{z}}(\mathbf{z}) = \prod_{i=1}^{N} p_{\mathbf{z}}(z_i|\mathbf{z}_{<i}), \qquad p_{\mathbf{z}}(z_i|\mathbf{z}_{<i}) = \int_{\mathbb{R}} \mathrm{d}z_N \cdots \int_{\mathbb{R}} \mathrm{d}z_{i+1}\, p_{\mathbf{z}}(\mathbf{z}). \tag{3.4}$$

By assuming $p_{\mathbf{z}}(\mathbf{z}) > 0$ inside all $\mathbb{R}^D$ we can say that this is true also for every $p_{\mathbf{z}}(z_i|\mathbf{z}_{<i})$. In this way we can define the transformation $F : \mathbf{z} \to \mathbf{x} \in (0,1)^D$ whose i-th element is defined by the cumulative distribution of the i-th conditional

$$x_i = F_i(z_i, \mathbf{z}_{<i}) = \int_{-\infty}^{z_i} \mathrm{d}t\, p_{\mathbf{z}}(t, \mathbf{z}_{<i}) = p_{\mathbf{z}}(t \leq z_i|\mathbf{z}_{<i}). \tag{3.5}$$

It's easy to understand that F_i is completely differentiable and increase monotonically since $p_{\mathbf{z}}(z_i|\mathbf{z}_{<i})$ is positive, meaning that is also invertible. Also, x_i doesn't depend on z_j with $i < j$ meaning that we can invert it element-by-element having that the following transformation can be defined

$$z_i = F^{-1}_{\mathbf{z}_{<i}}(x_i) = F^{-1}(x_i, \mathbf{z}_{<i}). \tag{3.6}$$

This expression posses a simple lower triangular Jacobian since $\partial F_i / \partial z_j = 0$ for $i < j$ giving us a simple form for the determinant as

$$\det J_F(\mathbf{z}) = \prod_{i=1}^{D} \frac{\partial F_i(\mathbf{z})}{\partial z_i} = \prod_{i=1}^{D} p_{\mathbf{z}}(z_i|\mathbf{z}_{<i}) = p_{\mathbf{z}}(\mathbf{z}). \tag{3.7}$$

In this way we have created a diffeomorphism with a trackable Jacobian and, using known results from analysis, also the inverse is completely defined with a Jacobian

given by $1/J_F$. Knowing that, we can use such result inside Eq. (3.2) and obtain that the resulting PDF obtained is

$$p_{\mathbf{x}}(\mathbf{x}) = p_{\mathbf{z}}(\mathbf{z})|\det J_F(\mathbf{z})|^{-1} = 1, \qquad \mathbf{z} = F^{-1}(\mathbf{x}), \tag{3.8}$$

implying that $\mathbf{x}$ is effectively uniformly distributed inside the open hypercube $(0,1)^D$. Therefore, we are able to define such diffeomorphism for every well-behaved PDF. Meaning that we can now let $p_{\mathbf{z}}$ and $p_{\mathbf{x}}$ be general probability functions, and we can so define $F(\mathbf{z})$ and $G(\mathbf{x})$ that maps the two to the hypercube. Concatenating the two transformation as $T = G^{-1} \circ F$ generate a diffeomorphism, since a composition of diffeomorphisms is one itself, that applied gives us

$$p_{\mathbf{z}} \circ T^{-1}(\mathbf{x})\left|J_{T^{-1}}(\mathbf{x})\right| = p_{\mathbf{z}}(\mathbf{z})|J_G(\mathbf{x})|\left|J_{F^{-1}}(G(\mathbf{x}))\right| = p_{\mathbf{z}}(\mathbf{z})|J_G(\mathbf{x})||J_F(\mathbf{z})|^{-1} = p_{\mathbf{x}}(\mathbf{x}), \tag{3.9}$$

where the notation $\mathbf{z} = T^{-1}(\mathbf{x})$ was used along with the identity $T^{-1} = F^{-1} \circ G$. In this way we can be certain that the transformation for mapping the two PDFs exists.

The result obtained not only tells us that a solution always exists, but is also telling us how to construct it using integrals in Eq. (3.5). Still, it's easy to understand how such numeric forms become unmanageable as the dimensionality increase and the storing space needed for one element increase as $\mathcal{O}(D^D)$. For this reason a Neural Network that approximate them appears to be a viable option to make it available, also because can be seen how the solution proposed is not the only one. An example of this is given by the *Box-Muller transform* [6] to map the uniform distribution in two dimension to a diagonal Gaussian as

$$x_1 = \sqrt{-2\ln z_1}\cos(2\pi z_2), \qquad x_2 = \sqrt{-2\ln z_1}\sin(2\pi z_2). \tag{3.10}$$

Where $z_1, z_2 \sim U(0,1)$, and can be seen how the transformation T defined in this way has a different form respect to the one in Eq. (3.5) since its Jacobian is not lower triangular. Nevertheless, by computing it and use the change of variable formula, one can see how arrives at the same result. Therefore, a general NN could be able to explore the space of all possible solution searching not only for the one that approximate the various integrals, but also for other unknown possible forms.

Machine learning diffeomorphisms

A Normalizing Flow model is defined by a prior, or latent, probability distribution $p_{\mathbf{z}}$ in $\mathbb{R}^D$ and a parametric transformation $T(\bullet;\boldsymbol{\theta}) : \mathbb{R}^D \to \mathbb{R}^D$ defining a diffeomorphism. Based on the parameters the transformation will map $p_{\mathbf{z}}$ to a model distribution

$$\tilde{p}_{\mathbf{x}}(\mathbf{x};\boldsymbol{\theta}) = p_{\mathbf{z}} \circ T^{-1}(\mathbf{x})\left|\det J_{T^{-1}}(\mathbf{x};\boldsymbol{\theta})\right|. \tag{3.11}$$

The final objective of the model is to find the right set of parameters $\boldsymbol{\theta}_0$ so that $\tilde{p}_{\mathbf{x}}$ approximate correctly a wanted target distribution $p_{\mathbf{x}}$. In order to do that we need mainly three things: 1) a loss function estimating the distance between $\tilde{p}_{\mathbf{x}}$ and $p_{\mathbf{x}}$, 2) a structure for T able to explore all the possible transformations, 3) make the sampling computationally cheap.

Finding an efficient loss function inside the contest of generative models is a known problem whose solution is generally given by the use of the Kullback-Liebler (KL) divergence [28]. In particular, by taking two PDFs p and f their divergence is given by

$$KL(p\|f) = \left\langle \log\left(\frac{f}{p}\right)\right\rangle_p = \int \mathrm{d}\mathbf{x}\, p(\mathbf{x}) \log\left(\frac{f(\mathbf{x})}{p(\mathbf{x})}\right), \tag{3.12}$$

such quantity is not a distance in functional space, but has the wanted properties of

$$KL(p\|f) \geq 0, \qquad KL(p\|f) = 0 \Longleftrightarrow p = f. \tag{3.13}$$

Thus, we can write down its explicit form in our case of interest, so by inserting the model distribution and the target distribution one will get as a result

$$KL(p_{\mathbf{x}}\|\tilde{p}_{\mathbf{x}}) = \langle \log p_{\mathbf{x}} - \log \tilde{p}_{\mathbf{x}} \rangle_{p_{\mathbf{x}}} = -\langle \log \tilde{p}_{\mathbf{x}} \rangle_{p_{\mathbf{x}}} + \text{const.} \tag{3.14}$$

Where the average relative to $\log p_{\mathbf{x}}$ will not depend on the parameters $\boldsymbol{\theta}$ and therefore will remain constant in the minimization. Now we can approximate the average assuming to have a database of $\{\mathbf{x}_i\}_{i=1}^N$ of samples from the target distribution $p_{\mathbf{x}}$, so that by using the known form of $\tilde{p}_{\mathbf{x}}$ one gets

$$KL(p_{\mathbf{x}}\|\tilde{p}_{\mathbf{x}}) \approx -\frac{1}{N}\sum_{i=1}^{N} \log p_{\mathbf{z}}(T^{-1}(\mathbf{x}_i;\boldsymbol{\theta})) + \log\left|\det J_{T^{-1}}(\mathbf{x}_i;\boldsymbol{\theta})\right| + \text{const.} \tag{3.15}$$

This can be defined as the loss of our model $\mathscr{L}(\boldsymbol{\theta})$ and minimized by using the *gradient descend* algorithm, directly computable as

$$\nabla_{\boldsymbol{\theta}} \mathscr{L}(\boldsymbol{\theta}) = -\frac{1}{N}\sum_{i=1}^{N} \nabla_{\boldsymbol{\theta}} \log p_{\mathbf{z}}(T^{-1}(\mathbf{x}_i;\boldsymbol{\theta})) + \nabla_{\boldsymbol{\theta}} \log\left|\det J_{T^{-1}}(\mathbf{x}_i;\boldsymbol{\theta})\right|. \tag{3.16}$$

By following such gradient we can search for the minima in parameter space $\boldsymbol{\theta}_0$ giving the best approximation for the wanted transformation. This approach is called *forward KL divergence*, and assumes that we are able to obtain a large enough set of samples from the wanted distribution in order to evaluate $\mathscr{L}$ with good precision. This is not always the case, and for this reason another approach can also be taken by using the *reverse KL divergence*

$$KL(\tilde{p}_{\mathbf{x}}\|p_{\mathbf{x}}) = \langle \log \tilde{p}_{\mathbf{x}} - \log p_{\mathbf{x}} \rangle_{\tilde{p}_{\mathbf{x}}} = \langle \log p_{\mathbf{z}} \rangle_{p_{\mathbf{z}}} - \langle \log|\det J_T(\bullet;\boldsymbol{\theta})| \rangle_{p_{\mathbf{z}}} - \langle \log p_{\mathbf{x}} \circ T \rangle_{p_{\mathbf{z}}}. \tag{3.17}$$

Here a change of variable is implicitly used in order to write the expectation value in terms of the latent probability. In this way the first average becomes a constant, while the others two gives the wanted loss function that can be approximated once again using the samples from the simple latent distribution as

$$\mathscr{L}(\boldsymbol{\theta}) = -\frac{1}{N}\sum_{i=1}^{N} \log|\det J_T(\mathbf{z}_i;\boldsymbol{\theta})| + \log p_{\mathbf{x}}(T(\mathbf{z}_i;\boldsymbol{\theta})) + \text{const.} \tag{3.18}$$

Now we have a way to search for the optimal solution by knowing the form of $p_{\mathbf{x}}$, still it's common for such function to be written as $p_{\mathbf{x}} = \bar{p}_{\mathbf{x}}/C$ with C an unknown normalizing constant. Thus, if only $\bar{p}_{\mathbf{x}}$ is available that does not pose a problem since we will have

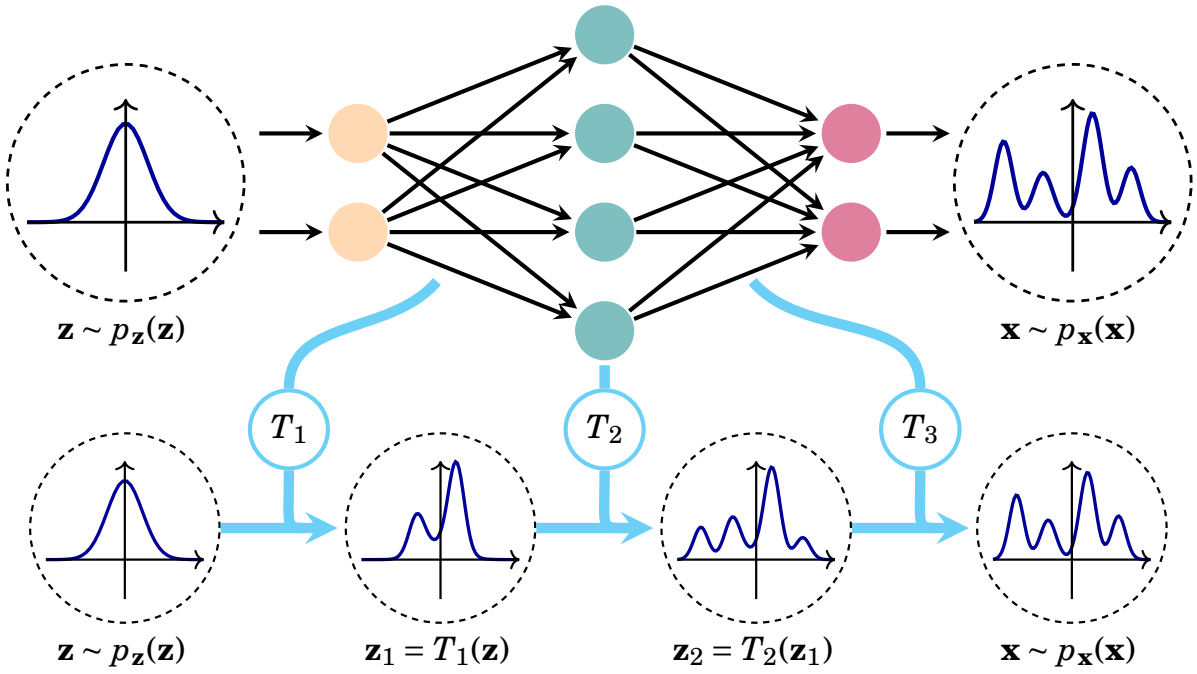


Figure 3.1: Schematic example of a Normalizing Flow generic architecture. On top, a neural network tries to approximate directly the complete transformation, while on the bottom the task is subdivided in a series of smaller transformations all given by a different NN.

that the constant can be taken out of the logarithm and incorporated inside the constant part of $\mathscr{L}$ having

$$\mathscr{L}(\boldsymbol{\theta}) = -\frac{1}{N}\sum_{i=1}^{N} \log|\det J_T(\mathbf{z}_i;\boldsymbol{\theta})| + \log \bar{p}_{\mathbf{x}}(T(\mathbf{z}_i;\boldsymbol{\theta})) + \text{const.} \tag{3.19}$$

Allowing us to find the wanted transformation by simply knowing the unnormalized form of the distribution we want to reach.

We are now able to search for the minimum of the distribution, still we need to find a way to give a neural network enough freedom to explore all possible T during training. The best way to approach this problem is not to focus on the search of a particular effective architecture, but to exploit the composition properties of diffeomorphisms. Basically, by knowing that taken T_1 and T_2 diffeomorphisms we have that $T_2 \circ T_1$ is still invertible and differentiable we can search our T by using a series of NN transformations $\{T_i\}_{i=1}^{M}$ and compose them. In this way we can avoid the creation of a big single model and focus instead on using a series of smaller ones as described in Fig. (3.1). The only complication of using this approach is given by the computation of the Jacobian determinant of the complete transformation, which is now given by using the chain rule on the total one

$$T = T_M \circ T_{M-1} \circ \cdots \circ T_1, \qquad \det J_T(\mathbf{z};\boldsymbol{\theta}) = \det J_{T_1}(\mathbf{z};\boldsymbol{\theta}_1) \det J_{T_2}(T_1(\mathbf{z};\boldsymbol{\theta}_1);\boldsymbol{\theta}_2) \cdots . \tag{3.20}$$

This can be easily inserted inside both Eq. (3.19) and Eq. (3.15) so that the log determinant will become the sum of the ones of every intermediate transformation, allowing us to compute the loss easily. Thanks to this we can increase the expressiveness of our network by simply increasing M.

The possibility of making our T more general by using an increasing number of simple intermediate transformations allow us to tackle the problem of computational efficiency by focussing on the single T_i. Three main requirements are needed for the NN model to represent T_i effectively:

Invertible network. The network needs to be an invertible function, so that the change of variable formula can be applied using it. That can also be seen inside the expression of the losses. In fact, if this property is not present then $\det J_T$ can become zero so that $\log|\det J_T| \to \infty$ not allowing us to train the model.

Trackable Jacobian. To estimate the value of $\tilde{p}_{\mathbf{x}}$ and the loss of the model itself we need to evaluate $\det J_T$. Now, the standard computation of a determinant is usually a $\mathcal{O}(D^3)$ operation, which is too expensive as the dimensionality increase. For this reason, the other requirement is that we shall be able to reduce J_T to a triangular-like form giving the possibility of compute the determinant in linear time complexity.

Expressiveness. It would be better to have a transformation with already the possibility of describing a large potion of the solution space, in this way a lower number of concatenations will be needed inside the complete form.

By constructing a T_i that respect such necessities we will be able to perform our computations on general PDFs and create general architectures for our studies that we will describe in detail in the next section.

3.2 Architectures

In Sec. (3.1.2) we have seen how constructing an effective architecture for normalizing flow is not an easy task, the network used not only needs to be invertible but have a Jacobian that can be computed in an inexpensive way. In order to obtain all of that a common strategy has developed through the years called *autoregressive flow* [24, 25]. The idea is based on the construction for the theoretic transformation done in Sec. (3.1.1) so that we define T bringing $\mathbf{z}$ to $\mathbf{z}'$ element wise as

$$z'_i = \tau(z_i; \boldsymbol{h}_i), \qquad \boldsymbol{h}_i = c_i(\mathbf{z}_{<i}; \boldsymbol{\theta}), \tag{3.21}$$

where τ is called *transformer* and c_i the i-th *conditioner*, which will depend on the model parameters $\boldsymbol{\theta}$. Creating this division allow us to obtain an invertible transformation if and only if τ is invertible, as shown in Fig. (3.2). Thus, we need to put attention on the form of the transformer, but once that is defined we have an invertible transformation that also possess an easy to compute $\det J_T$. In fact, it's easy to understand how

$$J_T^{ij} = \frac{\mathrm{d}z'_i}{\mathrm{d}z_j} = \frac{\mathrm{d}\tau(z_i; c_i(\mathbf{z}_{<\mathbf{i}}; \boldsymbol{\theta}))}{\mathrm{d}z_j} = \begin{cases} \frac{\partial\tau(z_i;c_i(\mathbf{z}_{<\mathbf{i}};\boldsymbol{\theta}))}{\partial z_j}\frac{\partial c_i(\mathbf{z}_{<\mathbf{i}};\boldsymbol{\theta})}{\partial z_j} & j < i \\ \frac{\partial\tau(z_i;c_i(\mathbf{z}_{<\mathbf{i}};\boldsymbol{\theta}))}{\partial z_j} & j = i \\ 0 & j > i \end{cases}, \tag{3.22}$$

giving rise to a lower-triangular matrix whose determinant is easily computable using the known relation

$$\det J_T(\mathbf{z};\boldsymbol{\theta}) = \prod_{i=1}^{D} \frac{\partial\tau(z_i;\boldsymbol{h}_i)}{\partial z_i}, \qquad \boldsymbol{h}_i = c_i(\mathbf{z}_{<i};\boldsymbol{\theta}). \tag{3.23}$$

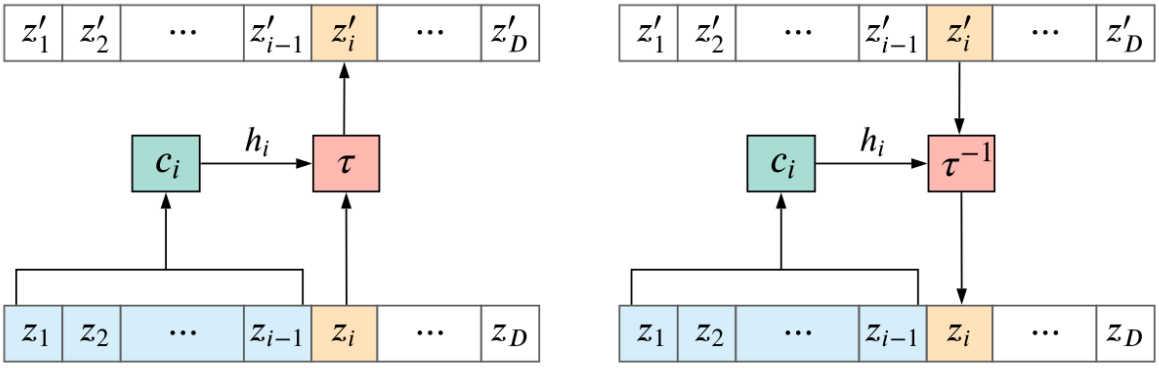


Figure 3.2: Illustration of the architecture inside a general T_i inside a flow, adapted from [40]. It's possible to see how the main properties of the function are given by τ dictating the direct transform and the inverse one, so that c_i can have the form we want giving us the freedom of movement of using traditional NN models.

Therefore, all the main requests for the creation of an effective transformation are already satisfied, the only thing remained is expressiveness. Which, can be obtained by noticing that no constraint on the function c_i were imposed during this construction. For this reason, we can use the conditioner as the real expressive part of the transformation by using a general NN in order to compute the inputs $\boldsymbol{h}_i$. In this way when we will train the model what we are going to improve is the estimate done by the conditioner on the input parameter for the transformer.

One may think that such general architecture can solve all of our problems. In fact, we can easily obtain invertibility and efficiency while construct expressibility by using larger networks for c_i or by composing more T_i together. Even if that may seem true it's not that easy when you try to apply the model in practice. The expressibility, in particular, does not only depend on the number of transformations used or by the depth of c_i, but the form of τ plays a big role on what we can represent and what not on a practical level. Also, the efficiency becomes a trick matter when the dimensionality increase often leading to the choice of simpler models specialized in the sampling process or in the inference one. For these reasons it's important to keep in mind the different possibilities for both τ and c_i, for which we are going to describe the most popular ones leaving the interested reader to the complete review [40] for other examples.

Conditioners

Conditioners c_i can be in principle constructed as any function of $\mathbf{z}_{<i}$, meaning that each conditioner can be implemented as an arbitrary model depending on a set of parameter $\boldsymbol{\theta}$. However, a naive implementation where a different model is used for every entry of the input vector $\mathbf{z}$ would scale poorly with the dimensionality D. In fact, it would require the evaluation of D separate models each with an average vector size of $D/2$. In addition, such approach would result in being memory expensive since we shall also store $\boldsymbol{\theta}$ for every model. Based on the early works on flow based models [7] define the autoregressive transformation approach as prohibitively expensive. Nonetheless, all of those problems can be addressed nowadays by sharing parameters across c_i or combin-

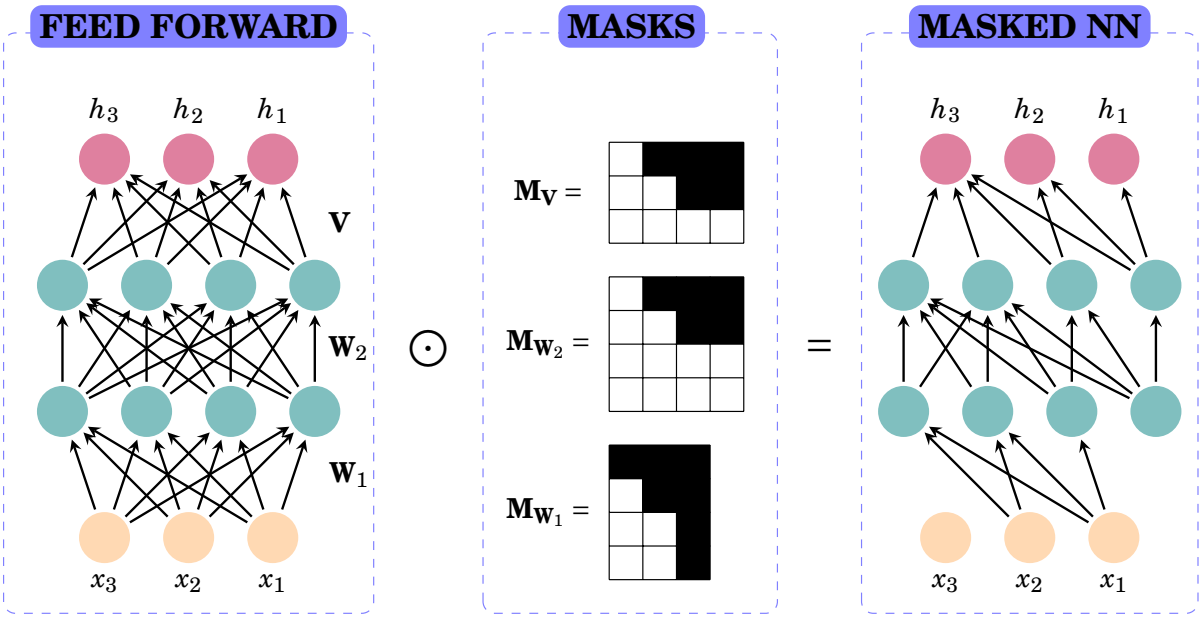


Figure 3.3: Construction of a Masked conditioner starting from a fully connected feed forward neural network to which we apply the masks in order to cut the wanted connections ending with h_i depending only on z_j with $j < i$. In the masks the black squares are the zero and the white ones the ones.

ing them into a single model, giving rise to the *masked conditioner* [15] and *coupling conditioner* [10] approaches.

Masked conditioner Instead of using set of conditioners in order to compute every $\boldsymbol{h}_i$ we can use a single feed forward NN with the property

$$c(\mathbf{z};\boldsymbol{\theta}) = [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_D], \qquad \frac{\partial \boldsymbol{h}_i}{\partial z_{j>i}} = 0. \tag{3.24}$$

To construct such a network we can use *masks* in order to cut the connections between the output $\boldsymbol{h}_i$ with the inputs $\mathbf{z}_{>i}$. Where, a mask is a binary matrix that can be applied to a network in order to cut out certain connections between layers defining a *masked network*. The application of such masks is done by simply doing the element by element multiplication of the matrices of the layer's weight with the one of the mask

$$\mathbf{W} \odot \mathbf{M}_{\mathbf{W}} = \begin{pmatrix} W^{11} & \cdots & W^{1n} \\ \vdots & \ddots & \vdots \\ W^{m1} & \cdots & W^{mn} \end{pmatrix} \odot \begin{pmatrix} M_W^{11} & \cdots & M_W^{1n} \\ \vdots & \ddots & \vdots \\ M_W^{m1} & \cdots & M_W^{mn} \end{pmatrix} = \begin{pmatrix} W^{11}M_W^{11} & \cdots & W^{1n}M_W^{1n} \\ \vdots & \ddots & \vdots \\ W^{m1}M_W^{m1} & \cdots & W^{mn}M_W^{mn} \end{pmatrix}. \tag{3.25}$$

So that, M_W^{ij} is 1 if we want the i-th output and the j-th input to be connected or 0 if we want to eliminate such weight from the final function. Thus, by selecting the right masks we can effectively obtain a network that respects the property in Eq. (3.24), like the one represented in Fig. (3.3).

Such architecture will allow us to evaluate all the parameters to perform the transformation T with only one evaluation of $c(\mathbf{z};\boldsymbol{\theta})$, an operation that can be easily parallelized using the GPU capabilities. Giving rise to a fast forward evaluation that allows for fast sampling from the approximated distribution $\mathbf{x} = T(\mathbf{z})$. However, the evaluation

of the inverse transformation, instead, becomes more troublesome using such conditioners. For example, we may want to perform statistical inference to a certain value $\mathbf{x}_0$ to know if it's likely to obtain it or not as

$$p_{\mathbf{x}}(\mathbf{x}_0) \approx \tilde{p}_{\mathbf{x}}(\mathbf{x}_0) = p_{\mathbf{z}} \circ T^{-1}(\mathbf{x}_0) \left| \det J_T(T^{-1}(\mathbf{x}_0; \boldsymbol{\theta})) \right|^{-1}. \tag{3.26}$$

In this case we need to transform $\mathbf{x}_0$ using the inverse transformation, but in order to do that we cannot make only a single evaluation of c. This is because the parameters $\boldsymbol{h}_i$ needed to obtain $z_i = \tau^{-1}(z'_i; \boldsymbol{h}_i)$ cannot be computed until all $\mathbf{z}_{<i}$ is known. Meaning that we should start by taking $\boldsymbol{h}_1$, which is a constant, and obtain $\mathbf{z}_1 = \tau^{-1}(z'_1; \boldsymbol{h}_1)$ that can be used to evaluate

$$c(z_1, z'_2, \dots, z'_D; \boldsymbol{\theta}) = [\boldsymbol{h}_1, \boldsymbol{h}_2, \boldsymbol{h}'_3, \dots, \boldsymbol{h}_D]. \tag{3.27}$$

From it, we can take $\boldsymbol{h}_2$ and evaluate z_2 to repeat the operation and obtain z_3 and so on. In this way, the evaluation of the inverse transform is an operation that requires D evaluations of the network, leading to an expensive computation that can become too costly for high dimensional data. For this reason, masked conditioner are generally used inside models that need to perform only one type of operations between sampling and inference, or with low dimensional models.

Coupling conditioner This other model for conditioners arises from the need of curing the asymmetry inside the masked conditioner architecture in order to obtain both fast sampling and inference. To achieve that one can work by choosing an index $d < D$ for the input variables and define a conditioner that work as

$$c(\mathbf{z}_{\leq d}; \boldsymbol{\theta}) = [\boldsymbol{h}_1, \dots, \boldsymbol{h}_d, \boldsymbol{h}_{d+1}(\mathbf{z}_{\leq d}), \dots, \boldsymbol{h}_D(\mathbf{z}_{\leq d})]. \tag{3.28}$$

So that $\boldsymbol{h}_{\leq d}$ are constants and the other ones are computed by simply using only the $\mathbf{z}_{\leq d}$ inputs. In this way the final transformation will of course posses the wanted property of having a triangular Jacobian and the evaluation of the direct or inverse transformation would be equally simple to compute evaluating c only once. This is explained graphically in Fig. (3.4), where a particular implementation is used where the transformation is set in order to have that

$$\mathbf{z}' = [z_1, \dots, z_d, \tau(z_{d+1}; \boldsymbol{h}_{d+1}), \dots, \tau(z_D; \boldsymbol{h}_D)], \qquad c(\mathbf{z}_{\leq d}; \boldsymbol{\theta}) = [\boldsymbol{h}_{d+1}, \dots, \boldsymbol{h}_D]. \tag{3.29}$$

In this way we can reduce the number of computations needed in our model not only to find $\mathbf{z}'$, but also for the computation of the Jacobian determinant since we have

$$\det J_T = \det \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{A} & \mathbf{D} \end{pmatrix} = \det \mathbf{D} = \prod_{i=d+1}^{D} \frac{\partial \tau(z_i; \boldsymbol{h}_i)}{\partial z_i}. \tag{3.30}$$

Both transformation and determinant gets simplified to a $\mathcal{O}(D-d)$ operation. Accounting that along to the fast computation for both forward and inverse transform make this method much more generally accessible compared to the masked conditioner.

Still, this particular architecture possess only one major downside, the expressibility. In fact, the increase in performances are paid through the fact that the entries $\mathbf{z}_{\leq d}$ are not transformed and the ones $\mathbf{z}_{>d}$ gets conditioned only by the previous entries and do

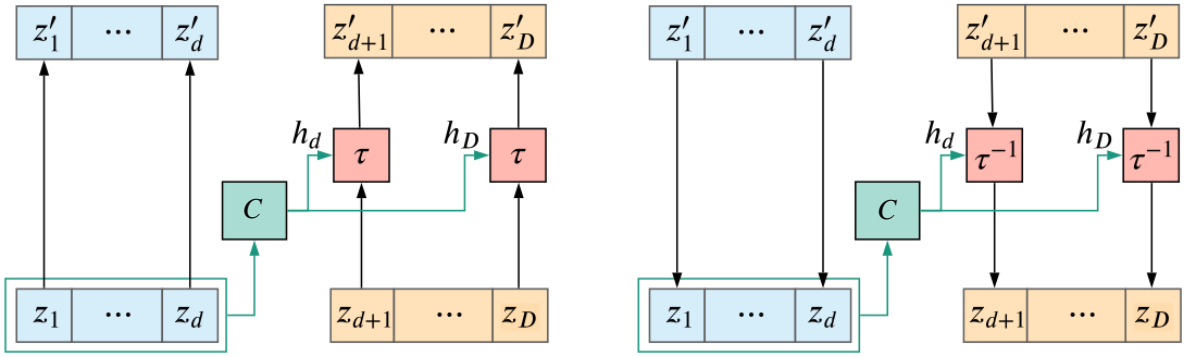


Figure 3.4: Representation of the coupling conditioner mechanism where the input vector is split in two part $\mathbf{z}_{\leq d}$ and $\mathbf{z}_{>d}$ so that only the latter is transformed with parameters computed from $\mathbf{z}_{\leq d}$. Image adapted from [40].

not account for interactions between them. This highly limit the expressibility of the final network, but a good result can be restored by inserting inside the set of all the transformations $\{T_i\}$ a series of permutations. Can be easily seen how permutations P are diffeomorphisms with the properties of

$$P(\mathbf{z}) = [z_{P(1)}, z_{P(2)}, \dots, z_{P(D)}], \qquad \det J_P = 1. \tag{3.31}$$

Basically, by changing only the order of the entries inside the vector the volume of the space is not changed, leading to a Jacobian with unit determinant. We can use that in order to have transformations inside the final flow T that changes the position of the elements so that the entries inside $\mathbf{z}_{\leq d}$ change and every entry can get conditioned by the others. Also, such solution results in being computationally really cheap since no variation to the final Jacobian determinant is obtained. Anyway, even if this solution seems to give us the possibility of making the flow expressible at will, also highly increases the number of transformations needed in order to obtain a good a result. An overhead that needs to be taken into account when selecting between masked conditioner and this one.

Transformers

The transformation τ is the most important piece inside the whole architecture since it does not only determine the invertibility of the whole T, but also the expressibility and, most importantly, the complexity. In fact, τ determines the form of $\boldsymbol{h}_i$ implicitly defining the dimensionality of the output layer of the conditioner c, which should be constructed based on that. Thus, one can imagine how the more complex τ is the larger $\boldsymbol{h}_i$ will look like, giving us a more complex T with a larger expressiveness potential. Therefore, a balance between the two aspects needs to be achieved based on the application we are interested in. Here we are going to introduce two of the most popular choices present in literature that also represent the extremes of expressiveness vs complexity: the *affine transform* [11] and the *spine-based transform* [12].

Affine transform This transformation was thought in order to have the simplest possible invertible transformation that we can think of, giving rise to the following linear function

$$\tau(x; a, b) = x \exp(a) + b. \tag{3.32}$$

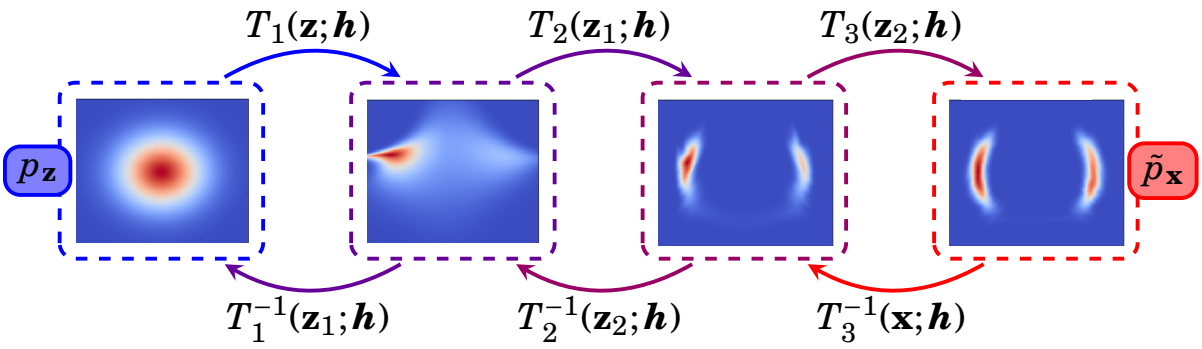


Figure 3.5: Example of use of NF to transform a bidimensional diagonal Gaussian into a correlated two moon distribution by mean of several affine transformations. In particular, this flow has been obtained using 6 transformations each using a coupling transformer and each one was followed by a permutation that swapped the two entries of the variable. Such model has been obtained using the LLDMC library [32].

Such transformation requires only two parameters to be computed, having $\boldsymbol{h} = [a, b]$, and is of course invertible thanks to $\exp(a) > 0$ so that the inverse is

$$\tau^{-1}(x;\boldsymbol{h}) = (x-b)\exp(-a), \tag{3.33}$$

where only multiplications and sums are involved in both forward and inverse, giving an incredibly optimized and stable operation to perform. Not only that, but if we look at the logarithm of the Jacobian determinant of such transformation we will find that

$$\log|\det J_T(\mathbf{z};\boldsymbol{h})| = \log\left|\prod_{i=1}^{D}\frac{\partial\tau(z_i;a_i,b_i)}{\partial z_i}\right| = \log\prod_{i=1}^{D}\exp(a_i) = \sum_{i=1}^{D}a_i, \tag{3.34}$$

showing how also this quantity, needed for the training, can be obtained in no time when using Eq. (3.32) as τ. Such properties have made affine transformers vastly popular with several applications reaching also state-of-the-art models such as Glow [26]. Nevertheless, its simplicity also undermine its expressiveness, something that can be easily seen by taking the case where $\mathbf{z}$ follows a multivariate Gaussian. Then, each z'_i conditioned by $\mathbf{z}'_{<i}$ will also follow a Gaussian distribution, having that a single affine transformation cannot ve a universal approximator. Nonetheless, expressive flows can still be obtained by stacking multiple transformations together as seen inside the results shown in Fig. (3.5), but is still unknown if affine flows composed by multiple layers are effectively universal approximators or not.

Spline-Based transform The other option is to aim at having a really high expressiveness by using a τ that is able alone to approximate every possible monotonic increasing, and so invertible, function and leave behind optimization. In order to achieve that the idea is to use a rational quadratic spline $g_{\boldsymbol{\theta}}(x)$, i.e. a picewise function consisting of K segments, where each segment is a simple function that is easy to invert. To construct such object we need a set of node $\{(x_i, y_i)\}_{i=1}^{K-1}$ and the values of the derivates on such points $\{\delta_i\}_{i=1}^{K-1}$, that together will constitute the input parameters of the transformer. Basically, in such architecture the transformer will take the form

$$\tau(z_i;\boldsymbol{h}_i) = g_{\boldsymbol{h}_i}(z_i), \qquad \boldsymbol{h}_i = [x_1, y_1, \delta_1, \ldots, x_{K-1}, y_{K-1}, \delta_{K-1}], \tag{3.35}$$

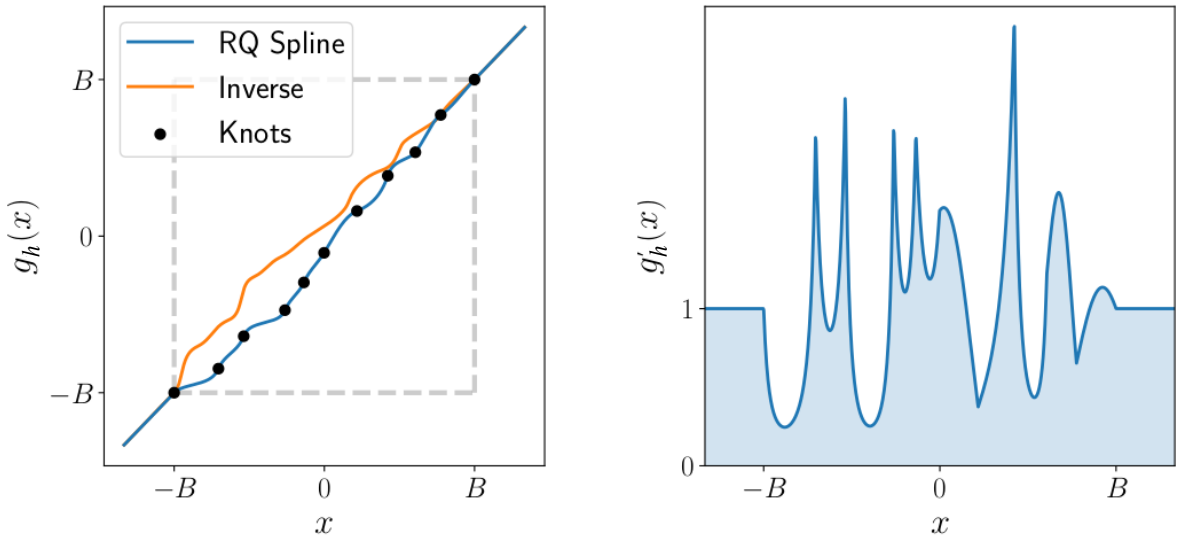


Figure 3.6: Example of a rational cubic spline transformation as described in this section. On the left the value of the transformation where all the derivative on each node were set positive and the inputs outside the given domain have been transformed through the identity. On the right the derivative of the transformation is displayed. Image adapted from [12].

showing a much larger increase in the degree of complexity coming from a high dimensionality for $\boldsymbol{h}_i$ leading to large networks for the conditioner. Still, the greatest enhancement in complexity is the evaluation of the function itself. In fact, in order to compute $g_{\boldsymbol{h}}(x)$ one should first find out in which of the K segments the input x is and then compute the value of the function present in that part of the domain. Therefore, we need first to use a quick search algorithm, like binary search having $\mathcal{O}(\log K)$ complexity, in order to find the interval $z_i \in [x_k, x_{k+1}]$ and then compute the spline as

$$g_{\boldsymbol{h}}(z_i) = y_k + \frac{(y_{k+1}-y_k)[s_k\xi^2 + \delta_k\xi(1-\xi)]}{s_k + [\delta_{k+1}+\delta_k - 2s_k]\xi(1-\xi)}. \tag{3.36}$$

Where we implicitly used the following notation

$$s_k = \frac{y_{k+1}-y_k}{x_{k+1}-x_k}, \qquad \xi = \frac{z_i - x_k}{x_{k+1}-x_k}, \tag{3.37}$$

in order to make the expression for the rational quadratic spline less heavy. Even if Eq. (3.36) may look really complex has still the handy properties of possessing an analytical form both for the derivative and for the inverse. In particular, by setting all $\delta_i > 0$ one can obtain a general monotonically increasing function with both $\det J_{g_{\boldsymbol{h}}}$ and $g_{\boldsymbol{h}}^{-1}$ available for direct computation. Inside Fig. (3.6) an example of such transformation is shown where some practical simplifications have been made. First, the starting and final nodes have been fixed to the values of $(-B,-B)$ and (B,B) possessing a unitary derivative. Then, a generalization of the method is used in order to deal with the entries that are not inside the domain $[-B,B]$ by assuming that

$$g_{\boldsymbol{h}}(x) = x, \qquad x \notin [-B,B]. \tag{3.38}$$

In this way we can deal with all $\mathbb{R}$ while also being able to approximate a vast array of invertible functions inside the $[-B,B]$ domain. Thus, the transformation has a strong

potential for approximate even complex distribution without the need of stacking a lot of transformations together, but such freedom results in a much more complex and computationally expensive structure if compared to the affine one.

3.3 Tackle DiagMC distribution

Let the distribution of interest be the one of diagrams inside a general model, which is defined by the expression

$$p_W(\mathscr{C}) = W(\mathscr{C})/Z, \qquad Z = \sum\!\!\!\!\!\!\int_{\mathscr{C}} W(\mathscr{C}). \tag{3.39}$$

Where we assume that the diagram $\mathscr{C}$ can be represented as a vector containing the main information needed to compute the weight and the observables, such as: order, time of flight and interaction times of the different type of interactions

$$\mathscr{C} = \left[n, \tau, \tau_1^1, \tau_2^1, \dots, \tau_1^2, \tau_2^2, \dots\right]. \tag{3.40}$$

In principle, one can think that we should be able to obtain information about p_W by using the methods discussed so far. In fact, it's easy to see how Eq. (3.39) give rise to a distribution of which we know the unnormalized form, leaving us the possibility of minimizing the reverse KL divergence loss in Eq. (3.19). As a matter of facts using W inside our loss is not enough to obtain the complete distribution out of a classic flow model. That is, W does not contain the complete information on the distribution, which is highly influenced by the *time ordering constraint* that can be stated as

$$\tau_{i+1}^j > \tau_i^j, \qquad \tau_i^j > 0, \qquad \tau > \tau_i^j. \tag{3.41}$$

These are domain requirements that changes the normalization of the various distributions of the single variables, giving an important contribution to the total p_W. Alongside that we also have that inside all possible p_W we will always have the order variable n, which is integer, giving rise to a distribution with both integer and real variable which so far has not been studied inside NF literature.

Therefore, here I want to address such problems by proposing possible ways of avoid them that has been studied during this thesis work. We will start by focusing on the integer distribution problem, for which a literature still exist, and then move to the novel problem of imposing time ordering.

Integer distributions

The approximation of integer distribution through Normalizing Flow is a known issue in literature due to the limitations that the change of variable formula has on discrete domains. In particular, by taking a map $T : A \to B$, with A and B subsets of $\mathbb{Z}^D$, and an initial distribution $p_{\mathbf{z}}$ we would have that the application of T lead us to [40]

$$p_{\mathbf{x}}(\mathbf{x}) = \sum_{\mathbf{z}\in\omega} p_{\mathbf{z}}(\mathbf{z}), \qquad \omega = \{\mathbf{u} | T(\mathbf{u}) = \mathbf{x}\}. \tag{3.42}$$

This relation is defined for a general map, but it's easy to understand that if one takes T to be bijective, then ω must be composed by only one value having $p_{\mathbf{x}}(T(\mathbf{z})) = p_{\mathbf{z}}(\mathbf{z})$.

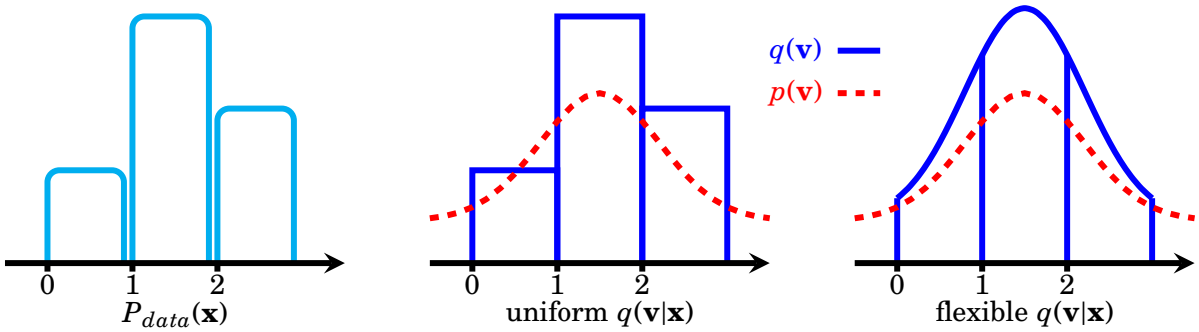


Figure 3.7: Example of dequantization transformation with a uniform q or a flexible one, taken from [23]. It's possible to see how the value of the continuous model, obtained from the flow, gets mapped into the same integer probability distribution in both cases. Still, for the flexible distribution less uncertainty is presence for the inverse transformation, helping the train when the forward KL is used.

Basically, we have that flows of integer variables can only permute the probabilities of the starting distribution between the different vectors of the space. Therefore, it is not always true that a transformation exist that bring whatever two distributions into each others, placing a theoretical limit to our expressiveness. That seems to pose a problem that cannot be solved, limiting us to try to sample only the real variables of the diagram leaving the integer ones as parameters, but that is not true. During this thesis work two main possible solutions have been thought that can help overcome this problem. The first is using a mathematical trick called *dequantization* [23, 49] and tries to treat the discrete variables as real, while the other one consists in separate the order sampling directly using the data.

Dequantization Mathematically the reason why the integer flows are so limited is the absence of the Jacobian contribution in the change of variable formula of Eq. (3.42). Thus, to give more expressibility to such construction we need to find a way to add that inside the transformations acting on such variables. Such problem is studied in literature in order to perform image processing, where colored images are described by a set of pixels with integers values describing the color $\mathbf{x} = \{0, \dots, 255\}^D$. So, the standard way to work with such type of data is to add a particular layer, called *quantization layer*, at the end of the flow that is able to transform the integer values into real one. In this way we can work with a $p_{\mathbf{z}}$ that treat them as real variables and transform them as such having expressible transformations, but also remapping them to integer at the end obtaining the real $p_{\mathbf{x}}$. This approach was first proposed by *Theis et.al.* [49] and then evolved over the years, but the core of dequantization is to use a transformation $D : \mathbb{R} \to \mathbb{N}$ inside the flow defined as

$$D(z_i) = \text{floor}\,(z_i), \qquad D^{-1}(x) = x + v. \tag{3.43}$$

Where $v \in [0,1]$ is a random value extracted by some distribution $q(v|x)$. It's easy to understand how such transformation is not bijective, having that all the values inside a $[n, n+1]$ range are mapped into n. That would mean that no inverse is present for such function, in fact D^{-1} is not the real inverse but is known in literature as *statistically exact inverse*. Can be seen that by adding a random noise to the integer entries, so that n gets mapped back inside the range $[n, n+1]$ randomly, the KL divergence loss obtained

is still $\leq$ to the real one thus leading to the same result obtained by minimization [49]. Knowing that, we can choose the distribution we want in order to implement D ranging from the simple uniform one, $U(0,1)$, or a more flexible parametrized one that can be optimized as one wants, Fig. (3.7).

This approach can be used in our study of the diagrams' PDF letting us treat the order as a real variable and then, only at the end apply a dequantization layer that only modifies the order variable inside our $\mathscr{C}$ vector. Also, the choice of $q(v|x)$ is not important in such application since we are going to use the inverse KL loss in order to train and the form of q mainly influences training using the forward one. Therefore, we can simply stick with using the naive uniform distribution as inverse without lowering our training performances.

Direct data sampling The other approach that can be taken is to treat the diagram order separately to the other variables. Basically, we have that the order statistics is usually fast to converge and collect giving us the possibility of performing a first run of our model in order to collect it with few million steps in the chain. Once one has collected $p_W(n)$ sampling from it is really easy since it's an integer distribution and so the CDF inverse algorithm, see App. (A), is really cheap to perform. In this way we can have access to the exact sampling of n that can then be used as a parameter inside the flow. Meaning that we can construct a model where we first sample n and then construct a flow that takes it as an external parameter $\tilde{p}_W(\mathscr{C}|n)$. So that every transformation present inside it changes based on the order inserted, but does not act on it

$$[n,\tau,\tau_1^1,\tau_2^1,\dots] \to [n,T(\tau;n),T\left(\tau_1^1;n\right),T\left(\tau_2^1;n\right),\dots], \tag{3.44}$$

creating an architecture called *conditional flow* [2] which allow us to switch from sampling from the distribution of diagrams with different orders using the same model. In this way we would have as a final model distribution a function composed of two parts

$$\tilde{p}_W(\mathscr{C}) = p_W(n)\tilde{p}_W(\mathscr{C}|n), \tag{3.45}$$

an exact one related to the order, and the one coming from the conditional flow to describe all the other variables. That gives us a way to obtain the final estimate of the total probability for obtaining a certain diagram, resulting in being a much simpler solution respect to quantization at the additional cost of running a relatively cheap simulation to collect the order statistic.

Time ordering

Imposing an order between the variable of a distribution is important in order to determine the final properties of the latter. That is a known fact in statistic that gives rise to a certain branch of PDFs called *order distribution*, and the simplest example of them all is given by the uniform distribution. If one takes a set of random variables $\mathbf{X} = \{X_i\}_{i=1}^D$ all uniformly distributed along with $U(0,1)$ the final PDF would simply be $p(\mathbf{x}) = \prod_{i=1}^D p_{x_i}(x_i)$. But, if we now impose that $X_{i+1} > X_i$ the probability of obtaining x_i for the i-th variable needs to take into account the probability of having the one before

sampled smaller than it, and vice versa for the one after. That would give us a final expression for the PDF that can be written as

$$p(x_i) = p_U(x_i) \prod_{j=1}^{i-1} F_j(x_i) \prod_{j=i+1}^{D} \left[1 - F_j(x_i)\right], \qquad F_j(x) = \int_0^x \mathrm{d}y \, p_U(y), \tag{3.46}$$

where F_j is the cumulative distribution function of that particular random variable, that can be easily written for a $U(0,1)$ distribution as $F_j(x) = x$. The final form of the distribution obtained is quite complicated, even if it is constructed from the simple uniform distribution, showing us how such information is essential to describe diagrams'distributions.

From the previous discussion one should be able to understand how if we train our model by simply using W only information about the distribution of the single variables are inserted. This leaves us with the need of telling the flow to also order them, generating an order distribution with every p_{τ_i} following the form given by the given propagator. In order to do that one can again choose between two main possible solutions: constraining the architecture to generate an ordered vector, or modify W in order to insert time ordering directly in it.

Architecture constrain This method mainly consist in hard coding the constraint directly inside the architecture by using an invertible transformation that maps the generated variables into the wanted domain. For example, imagine describing the creation, b, and destruction, e, of a particle with a certain propagator $D(e-b)$. In this case we would need to have $e > b$ along with $b, e > 0$, both conditions can be obtained in a model by simple known transformations. To do that, let b and e be random variable in $\mathbb{R}$ obtained from a series of transformations. We can easily transform them in order to bring them into $\mathbb{R}^+$ by mean of a softplus function, which is invertible and can describe a flow that we can add to our model defined as

$$\tau(z_i;\beta) = \frac{1}{\beta}\log\left(1 + e^{\beta z - i}\right), \qquad \log\tau'(z_i;\beta) = -\beta\tau(-z_i). \tag{3.47}$$

So that by adding a layer with such transformation at the end of the model we would satisfy the second condition of having b and e positive. From there a really simple manipulation can be performed to order them by using a cumulative sum, namely the transformation

$$\tau(z_i) = \sum_{j=1}^{i} z_j, \qquad \log\tau'(z_i) = 0. \tag{3.48}$$

By placing such transformation after the softplus layer we would have that the final vector results in being order with $z_{i+1} > z_i$. In this way, we would be able to satisfy both the conditions we imposed on b and e with the simple cost of adding two cheap final layers to the complete flow.

Such approach results in being computationally cheap and simple to perform, since both functions are well known and usually optimized version are present in all machine learning frameworks. Nevertheless, it comes at a price inside the stability and expressibility of the constructed model. In particular, the softplus is not a too flexible function

is good at approximate exponential like distribution, but having a hard time with more general shapes. That often brings instability in the parameter β, becoming too high, or in the parameters of previous layers. A drawback that should be kept in mind when choosing the ordering approach inside the model.

Weight modification Another more elegant approach to retain time ordering is to insert it inside the weight by changing W into a function that gives zero if the entries are not ordered. To explain it we can take, again, the example of creation and destruction of a particle with propagator $D(e-b)$. In that case one would set W to D giving to the model the information on the shape of the distribution, but to also tell about the ordering one should modify it as

$$W(e,b) = \begin{cases} D(e-b) & e > b \wedge e, b > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{3.49}$$

In this way, a loss constructed using such function will be able to take into account the order between the variables since a zero probability is associated to the unordered case. Nevertheless, Eq. (3.49) is not suited to perform a model training since it's not stable. In fact, since it's possible to obtain 0 from such function we would have that if during the first phases of the training the model generates a set of non-ordered variables the loss will diverge to ∞ stopping the process. In order to avoid that we should modify the form of the weight in order to allow the machine to train, and the best way to do that is by using a step function as

$$\tilde{W}(e,b) = D(e-b)\sigma_\alpha(e-b)\sigma_\alpha(b), \qquad \sigma_\alpha(x) = \frac{1}{1+e^{-\alpha x}}. \tag{3.50}$$

Now, by mean of two simple sigmoid functions, we have a way to write down the weight so that it quickly decays as the generated variables do not respect the order. In particular, the decay is controlled by the value of the hyperparameter α which will modify the form of the distribution we are approximating, tending to the original one in the limit $\alpha \to \infty$. Therefore, using Eq. (3.50) to create the loss we can train the model in a stable way allowing us to use any known architecture without any constraint and obtain an approximated form of the real distribution. Also, the expression can be easily modified in a more general form and write it for a set of ordered variable simply as

$$\tilde{W}(\mathbf{z}) = W(\mathbf{z}) \prod_{i=1}^{D-1} \sigma_\alpha(z_{i+1} - z_i). \tag{3.51}$$

A really simple form that if inserted inside the loss of the system becomes $\log \tilde{W}$, so that the added sigmoids will translate into simple softplus functions. Meaning that also in this case, the computational overhead is reduced to the minimum thanks to optimized versions of the softplus function easily accessible.

Thus, this approach is able to let the flow posses much more freedom and expressiveness respect to the other one, but the distribution we are approximating is an approximate version of W itself giving potentially more error at the end. A drawback that can be controlled by tuning the α parameter and setting it higher to obtain a more reliable result, but making also the training less stable and leaving us with the task of trying higher values until we don't reach a limit.

4

Global updates for DiagMC

In this chapter we will focus on the results obtained from the study of global updates inside the Diagrammatic Monte Carlo algorithm. Starting with an analytical study of the diagrammatic distribution aimed to obtain insight on the form of the p_W function for both the spin-boson and single site Holstein model, leading also at an analytical form of the global updates. Then, the construction of a global update for the single site Holstein model using a normalizing flow architecture is described highlighting the challenges that such task poses. In both cases the resulting update are tested performing a correlation study to show how effective the approach is at improving convergence. Also, all the results showed during the discussion were obtained through the use of the C++ LLDMC package [32], whose structure is briefly described right ahead.

4.1 LLDMC Implementation

LLDMC is a general purpose Diagrammatic Monte Carlo package created for this work thesis, and was thought to constitute a flexible platform that allows to study different models with different update types. Such goal was so obtained by the implementation of several archetypical classes that composes the building blocks of the main DMC loop structure. Which is really easy to understand provided a proper knowledge of the main theory behind its statistical properties described in Ch. (2). In particular, once the *configuration* of the diagram $\mathscr{C}$, the set of *updates* $\{\Gamma_i\}_{i=1}^{N}$, and the set of *observables* $\{O_i\}_{i=1}^{M}$, have all been defined; the working loop can be constructed as

```
auto rng         = init_random_num_gen();
auto walker      = init_configuration();
auto updates     = create_update_list();
auto observables = create_observ_list();

for(int i = 0; i != Steps; i++){
    // Randomly select an update
    int which = rng() % updates.size();

    // Obtain acceptance rate and proposed diagram
    auto [acc_rat, new_walker] = updates[which](walker);

```

```
    // Perform MH decision
    if (acc_rat < std::uniform_real_distribution()(rng))
        walker = new_walker;

    // Collect data when chain is thermalized
    if (i > Thermalization_step)
        for (auto ob : observables)
            ob.eval(walker);
}

// Study of the data collected in observables list...
```

That is all the coding one needs to do in order to perform DMC computations once the starting building blocks have been defined. What LLDMC tries to do is to automatically build such structure using the classes constructed by the user, allowing for models to be studied quickly and with high freedom. In addition to that, a submodule has been inserted inside the package in order to allow the user to create normalizing flow models with the aim of creating global updates. Such module is called LLNF and is written on top of PyTorch C++ API [41], giving some general classes in order to create the backbone of a flow.

Here we want to briefly introduce the main features of LLDMC and LLNF in a way that hopefully would allow the interested reader to try them. So that, the building blocks of both the main parts will be introduced piece by piece as if we were constructing an example together.

Making a DMC simulation

As described previously, to construct a DMC simulation we need: the configuration used as walker in the chain, the updates that randomly modify the walker, and the observables that we want to evaluate. All of these three subclasses constitute a building block of the simulation and posses a counterpart inside the LLDMC package as a pure virtual class. So that, the user is able to define a configuration inside the package by simply creating the data structure wanted for its specific case and then make it inherit the associated primitive virtual class. To make an example we can write down the simple form of the Holstein diagram configuration class, which would have a form like the following.

```
#include <LLDMC/Archetypes.h>

class Holstein : public LLDMC::Configuration {
public:
    Holstein(): Configuration("Configuration name"){}

    void set_param(std::map<std::string, double> param) override{
        g = param["g"], o = param["o"], e = param["e"];
    }

public:
    // Diagram variables
    int order{0};
    double elec_fly_time{0};
    double ph_beg[1000], ph_end[1000];

```

```
    // Hamiltonian parameters
    double g, o, e;
};
```

This data structure will be used as walker inside our algorithm and should contain all the information about the system we are working with. In our case we have inserted the data that would describe the diagram: `order` that tells the number of phonons, `elec_fly_time` describe τ, and arrays giving the start, `ph_beg`, and end, `ph_end`, of all the phonons contained. But, along that, also the values of the Hamiltonian's parameters are present, with electron-phonon coupling strength g as `g`, phonon frequency Ω as `o`, and electron energy ε as `e`. Such values are often changed in different simulations in order to study the different regimes of the model, and so the possibility of changing them without recompiling the program results in being quite useful. That is the reason for the presence of the `set_param` function implemented here, which will be called at the start of the simulation to initialize the Hamiltonian parameters as defined by the user in the LLDMC ini file that will be described afterwards. Once that such function is defined the first building block has been constructed, and now we have a data structure containing all the information needed on the system under study that is accessible to all the program.

The next step is to construct the updates needed for the modification of the walker in the chain. Such objects inside LLDMC are a little different respect to the general simple case described inside the general DMC loop above. That is because inside the package updates are interpreted as objects that can directly modify the walker diagram without the need of giving in output a new one every time, which would be computationally really expensive. To show how that work in practice the implementation of Γ_{add} for the Holstein model is reported next.

```
class add_n : public LLDMC::Update<Holstein> {
public:
  add_n(): Update("Update name", "Balancing update name"){}

  // Store the modification and give the acc_rate for such
  double atempt() override{
    new_b = std::uniform_real_distribution<double>(0, dia->t)(rng);
    new_e = std::uniform_real_distribution<double>(0, dia->t)(rng);

    if (new_e < new_b)
      std::swap(new_e, new_b);

    // Acceptance rate
    return 0.5 * std::pow(dia->g * dia->t, 2)
           * std::exp(- dia->o * (new_e - new_b)) / (dia->n + 1);
  }

  // In case it's accepted modify the diagram
  void accept() override{
    dia->ph_beg[order] = new_b;
    dia->ph_end[order++] = new_e;
  }

private:
  double new_b, new_e;
};
```

From the example one can notice how every update is composed of two main methods: the first being `atempt`, and `accept` the second. The former has the role of sampling the new modification that we try to perform to the current diagram and store it into its local variables. Then, it needs to output the acceptance rate of the sampled change so that the package is able to perform the Metropolis-Hasting step and accept or reject the update. Getting accepted means calling the `accept` function and effectively performing the sampled modification. In our case this implies adding a phonon to the diagram by inserting the new beginning and end into the arrays, and then increase the order that gives the total number of phonons. It's important to notice how inside the class all the variables `dia` and `rng` are used without never defining them. The reason is that LLDMC updates comes with an already initialized pointer to the walker diagram, called `dia`, used to modify it or access models variable and random number generator, named `rng`, that can be used for sampling. Once all of that is understood it becomes really simple to define updates for every type of situations, the only difficult part becomes getting the acceptance rate right. In particular, here a slight variation of the update introduced inside Sec. (2.2.1) is used that makes it more efficient. Instead of sampling the start from $U(0,\tau)$ and then the end using $U(b,\tau)$ they are both sampled from $U(0,\tau)$ and then ordered by swapping them if the end is lower than the start. This simplifies the form of the acceptance rate giving as final result the one showed in the code, which does not depend on the value of the sampled beginning, as was the one introduced in Sec. (2.2.1), making it more efficient.

The last step that needs to be done is the definition of the wanted observables, which will give the real object of study of the model under consideration. The way in which such object are constructed inside LLDMC is really similar to the updates, sicne they also possess the `dia` and `rng` variables and a similar syntax. Nevertheless, some differences are present and to see them through we prepared as example the class defining the observable for the ε_p value as follows.

```
class Ep : public LLDMC::Observable<Diagram> {
public:
  Ep(): Observable("Observable name"){}

  // Add the current value to the counter
  void eval() override{
    // Need high t limit
    if (dia->elec_fly_time < 7) return;

    double res = 0;
    for (int i = 0; i != dia->order; i++)
      res += dia->ph_end[i] - dia->ph_beg[i];

    val += (dia->o * res - 2 * dia->order) / dia->elec_fly_time;
    n++;
  }

  // Periodically save mean in history
  void conv() override {hist.push_back(val / n);}

  // Instruction to print the results
  void print(const std::string &path) override{
    FILE * file = fopen(path.data(), "a");
```

```
24
25      for (auto x : hist)
26        fprintf(file, "%f ", x);
27      fprintf(file, "\n");
28
29      fclose(file);
30    }
31
32  private:
33    double val{0}, n{0};
34    std::vector<double> hist;
35  };
```

The way in which the class will get used is simple. Basically, the `eval` method is called after every update so that the data can be collected and the current value for the ε_p is added to a general sum and a counter is increased. In this way the final value for the mean can be accessed in every moment by simply dividing one by the other, allowing for the creation of a history for the convergence of the value. In fact, the method `conv` is called not every step but after a fixed period that can be decided by the user, and here is used to insert values inside the convergence history without creating a too large array that would slow down the program. At last, the `print` function needs to be defined and gets used at the end of the simulation in order to print out its results. To be more precise, once the simulation comes to its end the program will call the print function of all the observables present passing as input the path `"progr_path/Obs_name.dat"`. In this way the user can decide how to print the data from the simulation as it wants inside a separate file and analyzing them as he wants.

After defining all the wanted updates and observables the simulation can be setup and performed. To do that inside LLDMC one needs to use the *manager* template class, which constitutes a simple user interface in order to quickly initialize every type of simulation in just few lines. To see that the code used to lunch every Holstein simulation has been reported in the following listing.

```
1  LLDMC::Manager<Holstein> manager;
2
3  manager.add_update<chg_t>();
4  manager.add_update<add_n>();
5  manager.add_update<rem_n>();
6
7  manager.add_observable<Ep>();
8  manager.add_observable<Green>();
9
10 manager.simulate("path/to/the/ini/file");
11
12 manager.print("path/for/printing");
```

Just a few self-explanatory commands are all that is needed in order to start the simulation using a `Holstein` object as walker and modify it using the wanted update or observables by easily add or remove one line. In this way we are able to experiment every possible combination of updates with ease, especially because the loop performed by the manager class is more sophisticated respect to the simple example described previously. In particular, it automatically collects the statistical properties of the chain such as the acceptance and reject rates of every update along with the correlation of the ac-

ceptance along the chain. So that we don't even need to construct specific observable for them since are general enough to be collected automatically for every model. Still, such feature can be turned off inside the same standard input file that is used in order to initialize the diagram and the simulation itself. To complete our example, and introduce such file, the input for a general Holstein simulation is reported next.

```
[Configuration]
g = 0.5
o = 1.0
e = 1.0

[Simulation]
THE_STEP = 5E6
MAX_STEP = 1E7
CONV_CKP = 1E4
GET_CORR = 1
```

The input file is written in a standard `.ini` format, with a first part related to the initialization of the Hamiltonian's parameters for the walker diagram, and a second one for the simulation's specifics. The former contains the names and the values that are then used inside the `set_param` function of the configuration. While, the latter is defining several specific values such as:

`THE_STEP` Number of thermalization steps before the observables starts to get collected;

`MAX_STEP` Total number of steps inside the chain before the end of the simulation;

`CONV_CKP` Number of steps after calling the `conv` method of the observables;

`GET_CORR` Boolean that tells if the correlation needs to be collected or not.

These are only a bunch of all the possible simulation options that can be used in the package, to see them all one can check the `SimOptions.h` file where all are reported.

Once the manager object and the input file are defined the simulation can be run as wanted, selecting the regime of the system by simply modifying the coupling or phonon frequency in the input file. The architecture leaves also room for parallelization, since it's easy to define a different manager object in every thread and then let every one of them run in parallel. In this way several chains can be created at the same time with results that can be printed at different locations to be analyzed separately.

Normalizing Flow module

The LLNF module was thought to give the user a set of tools to build general NF architectures focussing only on the PDFs they want to use and the transformations they want to apply. A goal achieved by the use of a `FlowManager` class that is able to construct a complete flow by the knowledge of: 1) a starting PDF, 2) the target one, 3) the list of transformations to apply. All of them can be defined using the primitives types of the module, giving rise to a structure similar to the one of the LLDMC simulations. So that, we will apply the same approach used for the previous part and construct a simple example of a flow together.

The first step in the construction of a NF is the definition of the two PDFs that are used: the starting distribution and the target one. To define them we shall use the pure

virtual classes `Distribution` and `TargetDistribution`, which are really similar to one another with the only difference that the target one does not require the definition of a `sample` method. In order to understand that better we show next the definition of the diagonal Gaussian distribution present between the standard distributions available inside the package.

```
#include <LLNF/Archetypes/distribution.h>

class GaussianImpl : public Distribution {
public:
  GaussianImpl(int n_dim, torch::TensorOptions option = torch::kFloat64
   ){
    mean = register_buffer("mean", torch::zeros(n_dim, option));
    std  = register_buffer("std", torch::ones_like(mean));
  }

  torch::Tensor sample(uint num_sample = 1) override {
    auto samples = torch::randn({num_sample, get_dim()}, mean.options()
   );
    return samples * std + mean;
  }

  torch::Tensor log_prob(const torch::Tensor &z) override {
    return -get_dim() * 0.9189385332046727 - (std.log() + 0.5 * ((z -
   mean) / std).pow(2)).sum(1);
  }

private:
  torch::Tensor mean, std;
};

TORCH_MODULE(Gaussian);
```

The influence of the Torch package should be now clear by the use of the `TORCH_MODULE` macro needed in order to give the defined class all the properties of a module. Notice that the only things that the user is required to do in this process is to define how to sample from the distribution and input the log probability of the distribution itself. Once those information are provided the distribution is complete and can be used inside the model. For the target distribution the process is even simpler since it's not required for the sampling mechanism to be inserted in the class definition, unless it is needed for the training method one intend to use. Anyway, in the definition of a base distribution there is also the possibility of defining a `cond_sample` method which takes as input a tensor of conditional parameters that can be used to create conditional NF architectures. Basically it can be used to give parameters to the distribution that can change its form or be passed to the transformations in order to condition their form. A strategy that will be better discussed, and used, in Sec. (4.3.1) during the construction of the model.

After the PDFs have been defined one can focus on the transformations by defining them through the `Flow` virtual class. The latter simply needs the definition of a `forward` method, which that transforms the input vector and compute the log determinant of the performed operation, and an `inverse` one with equal functionalities but with the inverse operation. A simple example of an implementation is given by an Affine transformer associated to a coupling conditioner, which can be written in the following way.

```
class CouplingAffineImpl : public Flow{
public:
  ToyAffine(int hidden = 50): cond(std::vector<int>{1, hidden, 2}){
    cond->to(torch::kF64);
    register_module("cond", cond);
    T_ = register_parameter("T", torch::zeros(1, torch::kF64));
    S_ = register_parameter("S", torch::zeros_like(T_));
  }

  std::tuple<torch::Tensor, torch::Tensor> forward(const torch::Tensor
   &z) override{
    auto&& h = cond->forward(z.slice(1, 1)).chunk(2, 1);

    auto S = torch::cat({h[0], S_.expand_as(h[0])}, 1);
    auto T = torch::cat({h[1], T_.expand_as(h[1])}, 1);

    return {S.exp() * z + T, S.sum(1)};
  }

  std::tuple<torch::Tensor, torch::Tensor> inverse(const torch::Tensor
   &z) override{
    auto&& h = cond->forward(z.slice(1, 1)).chunk(2, 1);

    auto S = torch::cat({h[0], S_.expand_as(h[0])}, 1);
    auto T = torch::cat({h[1], T_.expand_as(h[1])}, 1);

    return {(-S).exp() * (z - T), -S.sum(1)};
  }

private:
  LLNF::MLP cond;
  torch::Tensor S_, T_;
};

TORCH_MODULE(CouplingAffine);
```

Here the flow in question assumes that the input vector has dimension 2, so that the input vector ca be easily split in half and only the first entry is used inside the multi layer perceptron (MLP) to obtain the transformation parameters. Thus, this object constitutes a flow on its own that is able to transform half of the vector under study and can learn by training the parameters present inside the MLP object that works as conditioner. This is the approach that one should have inside the construction of every type of transformer inside the LLNF library.

Once the transformer has been defined correctly one has all the pieces needed for the construction of the complete `FlowManager`, which defines the complete model. In fact, that object posses a series of useful methods that can be used in order to perform general operations on the flow, such as: sampling, computing the log probability of a certain sample, evaluating the forward and reverse KL divergence, and so on. That is useful especially in the context of training the model, where the KL divergence is the key quantity to compute and inside our package we can do it using this simple code.

```
// Define the bits of the flow
auto base   = CreateBasePDF();
auto target = CreateTargetPDF();
```

```
std::vector<std::shared_prt<Flow>> flows = CreateFlowList();

// Construct the manager
LLNF::FlowManager flow(base, flows, target);

// Compute divergence
auto loss = flow.reverse_KL(1E5 /* = num_sample*/);
```

In this procedure the model will automatically sample the wanted number of samples from the approximated distribution and use the log probability of the target in order to compute the mean loss as in Eq. (3.19). This creates a final form that can be easily inserted into a training loop and allow us to freely experiment on the model composition by modifying only the external function `CreateFlowList`. This was exactly what was done in the work that will be presented in Sec. (4.3.1) to test the different models and search for the best hyperparameters.

4.2 Analytical global updates

Inside Sec. (2.1.2) we discussed how the main property of a global update is the one of possessing a unitary acceptance rate $\mathscr{A}$, bringing it to be always accepted inside a chain. Also, in our discussion was also shown how that condition is satisfied if the update $\Gamma(\mathbf{x} \to \mathbf{y})$ is equal to the target distribution $p(\mathbf{y})$ itself, giving no dependence on $\mathbf{x}$ and therefore no correlation. Knowing that, we can study the form of p_W inside DMC by using the condition described inside Eq. (2.26) and a vector representation of the diagrams, $\mathscr{C} = [x_1, \dots, x_D]$, to write

$$\frac{p_W(\mathscr{C})W(\mathscr{C}')}{p_W(\mathscr{C}')W(\mathscr{C})} = \frac{W(\mathscr{C}')\prod_{i=1}^{D} p_W(x_i|\mathbf{x}_{<i})}{W(\mathscr{C})\prod_{i=1}^{D} p_W(x'_i|\mathbf{x}'_{<i})} = 1. \tag{4.1}$$

Where we directly used p_W as the form of our update, and decomposed it using the chain rule of probability. Using such decomposition it's possible to focus on the modeling of the single conditional, one at a time, which results in a better approach when dealing with diagrams. In fact, the weight can be often decomposed by a series of contributions given by the different variables that allow us to have

$$\frac{W(\mathscr{C}')\prod_{i=1}^{D} p_W(x_i|\mathbf{x}_{<i})}{W(\mathscr{C})\prod_{i=1}^{D} p_W(x'_i|\mathbf{x}'_{<i})} = \frac{\prod_{i=1}^{D} f_i(x'_i) p_W(x_i|\mathbf{x}_{<i})}{\prod_{i=1}^{D} f_i(x_i) p_W(x'_i|\mathbf{x}'_{<i})}. \tag{4.2}$$

Basically, this lead us to something that can be simplified, in principle, by setting the conditional to be proportional to f_i. Nevertheless, such form is not always obtainable and also the constraint of time ordering make so that the final distribution is more complex than a simple proportionality due to domain definitions. Still, the best way to understand such concept is by application, and for this reason we will first study the test case of the single site Holstein model to show to obtain p_W in a simple case. Then, the same approach will be applied in the more interesting case of the spin-boson model showing how such approach is not restricted to toy models.

Single site Holstein

The single site Holstein model is a really simple test case characterized by a dispersionless phononic and electronic spectrum that reduces the weight function to the form discussed in Eq. (2.30). Such form gives us a really simple way of representing the diagram through a vector containing all the needed variables to compute the weight

$$\mathscr{C} = [\tau, n; b_1, e_1, \dots, b_n, e_n], \qquad W(\mathscr{C}) = g^{2n} e^{-\varepsilon\tau} \prod_{i=1}^{n} e^{-\Omega e_i} e^{\Omega b_i}. \tag{4.3}$$

Where, the absence of momentum inside the diagram make it so that W is composed by functions that only depends on one variable at a time. That allow us to use the form in Eq. (4.2) to start describing the conditionals of p_W one at a time by looking at the following form of $\mathscr{A}$

$$\frac{e^{-\varepsilon\tau} g^{2n'} \prod_{i=1}^{n'} e^{-\Omega e'_i} e^{\Omega b'_i} \prod_{i=1}^{2n+2} p_W(x_i|\mathbf{x}_{<i})}{e^{-\varepsilon\tau} g^{2n} \prod_{i=1}^{n} e^{-\Omega e_i} e^{\Omega b_i} \prod_{i=1}^{2n'+2} p_W(x'_i|\mathbf{x}'_{<i})} = 1. \tag{4.4}$$

Notice how the time of flight was taken equal in both $\mathbf{x}'$ and $\mathbf{x}$, that is because we already have discussed of an update having $\mathscr{A}(\tau \to \tau') = 1$ in Sec. (2.2.1). That allow us to think at τ as a conditional parameter inside our distribution and modify it using that update that still does not add correlation to out chain. The Eq. (4.4) is still one piece of the total information, to complete the picture we will also need the domain of definition for every variable. Such constrains can be obtained from the diagrams rules, understanding how we need to place a certain variable respect to the others in order to obtain a valid diagram. The best example is the time ordering constraint, meaning that we can't have a destruction time larger than a creation one, giving $b < e$, or that, again, we cannot have a electron-phonon interaction after the destruction of the electron, so $b, e < \tau$. Understanding that, one would easily understand the definition of the following domains

$$n \in \mathbb{N}, \qquad b_i \in [0, \tau], \qquad e_i \in [b_i, \tau], \tag{4.5}$$

where the only addition to what was already sad is that $b, e > 0$ since before the electron is not present. Eq. (4.5) and Eq. (4.3) shows how the couple $\{b_i, e_i\}$ related to one phonon has no connection with the ones related to the others. In fact W is composed by independent phononic propagators, whose value only depends on b_i and e_i, and the respective domains are not entangled so that e_i and e_j do not pose constraint on each others, same for b_i and b_j. Based on that, one can drop the complete dependence on the whole $\mathbf{x}_{<i}$ inside such variables and simply assume that

$$p_W(b_i|\mathbf{x}_{<i}) = p_W(b_i|\tau), \qquad p_W(e_i|\mathbf{x}_{<i}) = p_W(e_i|b_i, \tau), \tag{4.6}$$

since those are the variables that enters the domain or the propagator. Taking that into account we can isolate the part related to such functions inside Eq. (4.4) and focus on describing the phononic contribution to the distribution

$$\frac{e^{-\Omega e'_i} e^{\Omega b'_i}}{p_W(b'_i|\tau) p_W(e'_i|b'_i, \tau)} \frac{p_W(b_i|\tau) p_W(e_i|b_i, \tau)}{e^{-\Omega e_i} e^{\Omega b_i}}. \tag{4.7}$$

To satisfy this first requirement we can start from the distribution of e_i and simply assume that is proportional to $\exp(-\Omega e_i)$. In this way we can easily obtain a form for its PDF by normalizing it inside the defined domain, obtaining

$$p_W(e_i|b_i,\tau) = \frac{\Omega}{C} e^{-\Omega e_i}, \qquad C = \int_{b_i}^{\tau} \mathrm{d}x \Omega e^{-\Omega x} = e^{-\Omega b_i} - e^{-\Omega \tau}. \tag{4.8}$$

The final distribution is conditioned by τ and b_i through the normalizing constant, which represent the complexity added by the time ordering of the variables. By using this form inside Eq. (4.7) one can simplify the elements in the relation that depends on e_i and arrive at

$$\frac{1-e^{-\Omega(\tau-b_i')}}{p_W(b_i'|\tau)} \frac{p_W(b_i|\tau)}{1-e^{-\Omega(\tau-b_i)}}, \tag{4.9}$$

where can be noticed how the contribution of b_i has changed by mean of the normalization constant, becoming more complex. Nevertheless, the same idea as before can be used in order to eliminate b_i from the expression, so that we define as PDF for such variables the function

$$p_W(b_i|\tau) = \frac{1}{C}\left[1-e^{-\Omega(\tau-b_i)}\right], \qquad C = \tau + \frac{1}{\Omega}\left[1-e^{-\Omega\tau}\right]. \tag{4.10}$$

Once this is inserted in Eq. (4.9) it's easy to see how the fraction goes to unity, meaning that we have obtained the right expression for the conditional describing phonon's variables. After this discussion one would assume that the next step would be inserting Eq. (4.10) inside Eq. (4.4) and obtain a final form for the conditional of $p_W(n|\tau)$, but that would be an error. The error being assuming that the probability of having a diagram with phonons given by the set $\{b_i, e_i\}_{i=1}^n$ can be obtained from

$$p_W(\{b_i,e_i\}_{i=1}^n) = \prod_{i=1}^{n} p_W(b_i|\tau) p_W(e_i|b_i,\tau). \tag{4.11}$$

That is not true, because every permutation P of the indices defines a new set $\{b_{P(i)}, e_{P(i)}\}_{i=1}^n$ who leads to the same diagram. What this is telling us is that the order of the phonons inside the vector representation doesn't matter, and so the probability of having a certain set of phonons is given by the sum all possible permutations. In our case doing that is simple since all the permutation have same probability bringing so a simple $n!$ factor in front of Eq. (4.11). Knowing that we can correctly insert the distribution inside Eq. (4.4) and obtain that the two series of product simplifies into

$$\frac{\lambda^{n'}/n'!}{p_W(n'|\tau)} \frac{p_W(n|\tau)}{\lambda^n/n!}, \qquad \lambda = \frac{g^2}{\Omega^2}\left[\tau\Omega + 1 - e^{-\Omega\tau}\right]. \tag{4.12}$$

This show how by choosing the number of phonons as distributed following a Poissonian, with mean given by λ, the value of $p_W(n|\tau)$ simplifies the last term reaching a unity acceptance rate. Therefore, by studying one variable at a time, the full statistic of the phonons variables of a diagram, given a τ, was obtained. Where, we now know how number and positions of phonons are distributed as

$$p_W(n|\tau) = \frac{\lambda^n}{n!} e^{-\lambda}, \qquad p_W(b_i|\tau) = \frac{1-e^{-\Omega(\tau-b_i)}}{\tau + \frac{1}{\Omega}\left[1-e^{-\Omega\tau}\right]}, \qquad p_W(e_i|b_i,\tau) = \frac{\Omega e^{-\Omega b_i}}{e^{-\Omega b_i} - e^{-\Omega\tau}}. \tag{4.13}$$

That allows us to create an update that directly sample the correct distribution by using known numerical routine, such as the CDF inverse method in App. (A), to sample n, b_i and e_i as with the needed PDFs. Thus, the final update $\Gamma_{ph}(n \to n')$ will sample a new number of phonons, n', for the diagram from $p_W(n'|\tau)$ and then sample the values $\{b'_i\}_{i=1}^{n'}$ from $p_W(b'_i|\tau)$ that are needed to sample $\{e'_i\}_{i=1}^{n'}$ from $p_W(e'_i|b'_i,\tau)$. Once that it is constructed one can take the $\Gamma_{chg}(\tau \to \tau')$, that we discussed in Sec. (2.2.1), and add it to the newly created update in order to have a full ergodic global update that would always be accepted

$$\Gamma_g(\mathscr{C} \to \mathscr{C}') = \frac{1}{2}\left[\Gamma_{chg}(\tau \to \tau') + \Gamma_{ph}(n \to n')\right]. \tag{4.14}$$

The effectiveness of the method Γ_g was tested by directly implementing it inside the LLDMC package alongside the previously seen local set of updates in order to compare the convergence properties. Also, a third scheme of updates has been tested alongside them containing both global and local updates with a modification to Γ_{ph} to reject the update if $\left|n' - n\right| = 1$. Such requirement is present in order to simplify the detailed balance condition when an update goes from n to $n+1$, where if both local and global were present without limitations one would have

$$\Gamma_g(n \to n+1) = \frac{1}{N}\left[\Gamma_{ph}(n \to n+1) + \Gamma_{add}(n \to n+1)\right], \tag{4.15}$$

or remove in the case of n going to $n-1$, N is the number of updates in the set. The form in Eq. (4.15) needs to be taken into account inside the acceptance rate, making it much more complex and less effective. To avoid that we set $\Gamma_g(\mathscr{C} \to \mathscr{C}')$ to 0 when the two diagrams in question differs by only one phonon. That makes Γ_g simplify inside the previous expression leading to unmodified expressions for the acceptance rate of both global and local updates, bringing the possibility of their simultaneous use in a mixed global-local update. This leaves us with three possible update schemes: 1) the local one used nowadays as standard, 2) the global one that will always be accepted, 3) a mixed scheme that uses both the previous ones at the same time.

To test the convergence of the three proposed updates a series of 13 parallel simulations were performed for each update setting the Hamiltonian parameters at $\varepsilon = 1$, $\Omega = 1$ and $g = 0.5$. In this way, it was possible to look at the evolution of ε_p, defined in Eq. (2.67), at every step generating a set $\{\varepsilon_p^i(n)\}_{i=1}^{M=13}$ of 13 independent estimates of the same observable after n updates. From them one can compute statistical average and deviation of the polaron energy at a certain point in the simulation for the three updates

$$\bar{\varepsilon}_p(n) = \frac{1}{M}\sum_{i=1}^{M}\varepsilon_p^i(n), \qquad \sigma_\varepsilon^2(n) = \frac{1}{M}\sum_{i=1}^{M}\left[\varepsilon_p^i(n) - \bar{\varepsilon}_p(n)\right]^2. \tag{4.16}$$

Thus, the value of $\bar{\varepsilon}_p(n)$ gives us an average evolution of the observable and $\sigma_\varepsilon(n)$ approximate the error that one might expect for the observable in that point of the chain. By comparing how fast a certain set of updates reaches on average the exact value of $-g^2/\Omega$ and how large the $\sigma_\varepsilon(n)$ is we can give a quantitative estimation of the convergence performances. The results of such analysis for the three different update schemes are reported in Fig. (4.1) where is's also reported the value of the correlation along the chain computed as in Eq. (2.24). By looking at the results can be clearly seen how lowering the correlation inside the chain bring a great advantage to the overall convergence,

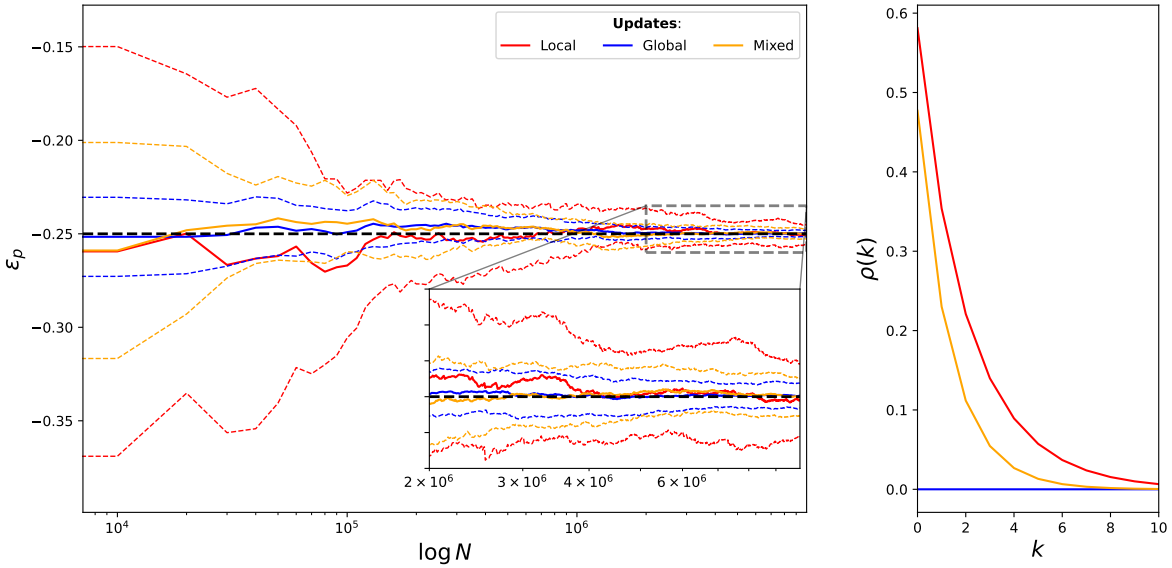


Figure 4.1: Evolution of the estimate for ε_p inside the single site Holstein polaron for the local, global and mixed update schemes, with the correct analytical result represented as the black dashed line. All the simulations were performed at $\Omega = 1$, $\varepsilon = 1$ and $g = 0.5$ with 1E6 thermalization steps. For all the schemes the results from 13 parallel chains were collected to compute the average behavior and the deviation. The former is represented as a solid line, while the colored area between dashed lines is the confidence interval obtained by adding and subtracting the standard deviation. A zoom on the last part of the convergence is present in the bottom right of the main plot, while on the right the correlation of a diagram with the one obtained after k steps is shown computed using Eq. (2.24).

as we expected from the discussion in Sec. (2.1.2). Still, here we can clearly see how the local update scheme posses the largest correlation between the three possibilities leading to the highest σ_ε and to a much larger number of steps needed before the average approaches the real value. In particular, the global scheme seems to reach the correct result in the first two million steps while the local observable is still away even after ten million. Also, the deviation associated to the former, represented as the dashed area in the figure, results in being ~ 2.9 smaller than the local updates one. This value is consistent with the expected increase of $2\tau_O$ obtained in Sec. (2.1.2) since the correlation time obtained from the chain is ~ 1.5. Therefore, the global scheme has a total advantage on the local one under a statistical point of view, but still can be understood how performing such update is more costly given that more random number needs to be generated every time. Such computational overhead can become important when the average order of the diagram increases, but still the enhancement in statistical behavior make it the better choice also for larger coupling. Still, in order to avoid even the increase in computational complexity we constructed the mixed update scheme. By using both the global update and the local ones we obtain a set of updates that perform a series of computationally cheap operations followed by global costly ones that reduces the chain correlation. The result is a final update that is on average cheap to perform, like the local one, and posses better statistics respect to simple local case. In fact, in Fig. (4.1) we can see how also for the mixed case $\bar{\varepsilon}_p$ reaches the exact result much faster and with a σ_p that is ~ 2 times smaller than the local one. The overall result is that the mixed scheme will outperform the classical one at all the possible couplings being as efficient

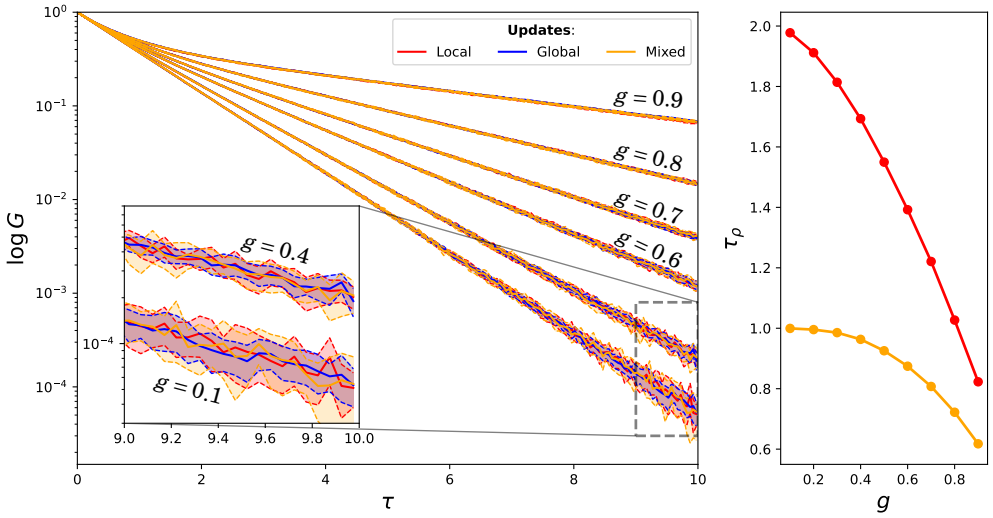


Figure 4.2: Results for the simulation done at different coupling constant g using the same setup of the simulation presented in Fig. (4.1). On the left the value of the Green function is reported in log scale with the average value between the 13 run in bold line and the confidence interval showed using the colored area. On the right, instead, the value of the correlation time as a function of g is presented showing how the mixed scheme is always better than the local one.

but with better statistical properties. In order to see how such claim holds true a series of computations for different values of g were performed using the same setup as before, but now computing the Green's function of the polaron and the total correlation time. The results are shown inside Fig. (4.2) and shows the correlation time evaluated for the ρ observable as a function of g for the mixed and local update, the global update is not reported since is identically zero. Such quantity shows how the mixed update is always less correlated then the local one at every step, showing how it's effectively the better choice at all coupling regime. Still, it's interesting to see how the correlation of both decreases rapidly as g increases. That is due to a larger average number of phonons in the system that make it so that the *add-n* and *rem-n* updates obtain a higher acceptance rate. Such behavior can be also seen in the estimation of the Green's function, where we can see how the standard deviation evaluated form the 13 runs gets lower for every τ value as g increases. Still, one last thing that can be noted is that the σ inside the G estimate is nearly the same for all the updates, as can be seen in the zoom on the left part of Fig. (4.2). That is because, even if the mixed and global updates have an overall less correlation all three schemes uses the same approach to update the external variable τ. In fact, τ_ρ is not a real estimate of the correlation of the variable τ, but a general way to look at the correlation of the chain. Thus, since the scheme used to update that variable is the same, and is a global update with unity acceptance rate, the correlation in the evaluation of the G observable is equally low for all the schemes. Therefore, it is true that the mixed and global update always leads to better converging estimates of the observable respect to the local ones. But, is also possible that the latter has already low correlation on particular variables that can lead to similar performance in the computation of averages depending only on them. Like the Green's function estimator in Eq.

(2.52) depended only on the value of τ, which was treated equally in our local and global approaches leading to the same correlation and equal performances in $G(\tau)$ estimation.

Spin-Boson

The same approached adopted so far can be used in order to describe also the PDF of the spin-boson model without further modifications. Such model was introduced in Sec. (1.3.2) and the description of their diagrams, in Sec. (2.2.2), showed how the model was affected by sign problem forcing us to work with $|W|$ instead of W. Thus, by looking at the absolute value of the weight we can see how the diagrams can be easily described by using two set of variables: the times of the tunneling interaction $\{\tau_i\}_{i=1}^{n_x}$, and the creation and destruction time of the phonons $\{b_i, e_i\}_{i=1}^{n_p}$. Using them, one can create a vector representation of the diagram similar to the one used in the Holstein one, that can be used to evaluate the weight as

$$\mathscr{C} = [\beta, n_x, n_p; \tau_1, \dots, \tau_{n_x}, b_1, e_1, \dots], \qquad |W|(\mathscr{C}) = \left(\frac{\Delta}{2}\right)^{n_x} \prod_{i=1}^{n_p} \mathscr{D}(e_i - b_i). \tag{4.17}$$

Once again the functions composing the target weight depends on one variable at a time, satisfying once again the form of Eq. (4.2) that we can use to construct $p_{|W|}$. In particular, here we can see how there is a complete separation of the spin and phonons degrees of freedom, leaving us with the possibility of separating the main function into

$$|W|(\mathscr{C}) = \left[\left(\frac{\Delta}{2}\right)^{n_x}\right]\left[\prod_{i=1}^{n_p} \mathscr{D}(e_i - b_i)\right] = \mathscr{S}(\beta, n_x; \tau_1, \dots, \tau_{n_x})\mathscr{P}(\beta, n_p; b_1, e_1, \dots, e_{n_p}). \tag{4.18}$$

This suggests that also the complete distribution can be separated in two parts that describe the statistics of spin, $p_{\mathscr{S}}$, and of phonons, $p_{\mathscr{P}}$, respectively. In the following description we will obtain expression for both of them, allowing us to create two global updates that would allow us to directly sample a spin state $\{\tau_i\}$ or a phonon state $\{b_i, e_i\}$ by using one or the other.

We will start by focussing on the spin part of the weight and work as for the Holstein model, so that we shall first describe the domain of integration for the variable, having that

$$n_x \in \mathbb{N}_2, \qquad \tau_{i+1} \in [\tau_i, \beta]. \tag{4.19}$$

Where β does not have a domain of definition since is taken as a constant describing the temperature of the simulation, also $\mathbb{N}_2$ is used to identify all even integers. Still, once again the correlation between the variable is inserted purely through the time ordering constraint, meaning that we could start the process of continuous integration of the $p(\tau_i|\beta, \tau_{i-1})$ conditionals as done in the previous subsection. Nevertheless, in this case a much cleverer way of obtaining the exact result can be used by doing the following. Let all the τ_i be sampled from the uniform distribution $U(0, \beta)$, then through the use of a sorting algorithm we can obtain a sorted set $\{\tau_j\}_{j=1}^{n_x}$ that respects the wanted boundary conditions. Mathematically speaking such approach is equivalent to sample n_x uniform random variable from the same distribution, and then select the one between the n_x!

permutations that bring the generated vector in the wanted order. As a result, the final PDF that is generated is given by

$$p_{\mathscr{S}}(\tau_i,\dots,\tau_{n_x}|\beta) = n_x! U(0,\beta)\cdots U(0,\beta) = n_x!\beta^{-n_x}, \tag{4.20}$$

where the factor $n_x!$ comes out by the same reasoning used for the Holstein's phonons distribution, that we need to add the possibility of sampling all the permutation of the same vector since they bring to the same result. Next, we can insert such form inside the decomposition condition of Eq. (4.2) so that by separating the spin variables one would obtain

$$\frac{(\Delta/2)^{n'_x} p_{\mathscr{S}}(n_x|\beta) p_{\mathscr{S}}(\tau_i,\dots,\tau_{n_x}|\beta)}{(\Delta/2)^{n_x} p_{\mathscr{S}}(n'_x|\beta) p_{\mathscr{S}}(\tau'_i,\dots,\tau'_{n_x}|\beta)} = \frac{\left(\Delta\beta/2\right)^{n'_x}/n'_x!}{p_{\mathscr{S}}(n'_x|\beta)} \frac{p_{\mathscr{S}}(n_x|\beta)}{\left(\Delta\beta/2\right)^{n_x}/n_x!} = 1. \tag{4.21}$$

Thus, the form that the distribution of the orders should take to reach unitary acceptance rate has the same form of a Poisson distribution. Nevertheless, we can't directly sample n_x from a naive Poisson since it needs to be even, a constraint that can be set inside the distribution by normalizing it as

$$\frac{1}{C}\sum_{n\in\mathbb{N}_2}\frac{\lambda^n}{n!} = \frac{1}{C}\sum_{n=0}^{\infty}\frac{\lambda^{2n}}{(2n)!} = \frac{1}{C}\cosh(\lambda). \tag{4.22}$$

Meaning that to normalize the distribution we shall set C equal to the hyperbolic cosine of the average. Knowing that is really easy to construct a numerical routine that directly sample from such distribution, so that we have obtained a form for $p_{\mathscr{S}}$ from which can easily sample from as

$$p_{\mathscr{S}}(n_x|\beta) = \frac{\left(\Delta\beta/2\right)^{n_x}}{n_x!}\cosh^{-1}\left(\frac{\Delta\beta}{2}\right), \qquad p_{\mathscr{S}}(\tau_i,\dots,\tau_{n_x}|\beta) = n_x!\beta^{-n_x}. \tag{4.23}$$

With a sampling process that starts by sampling an order from the even Poissonian and then sample n_x times uniformly distributed in between $[0,\beta]$ to pass to a sorting algorithm to obtain the final $\{\tau_i\}_{i=1}^{n_x}$ ordered set.

Next, to describe the distribution of phonons the best way is to make a slight change of variable and describe them not using the creation and destruction time, but using creation and length as

$$\mathscr{P}(\beta,n_p;b_1,\nu_1,\dots,\nu_{n_p}) = \prod_{i=1}^{n_p}\mathscr{D}(\nu_i), \qquad \nu_i = e_i - b_i. \tag{4.24}$$

From that we can now start the description of the distribution as before by recalling the variables domains, that in this case are

$$n_p\in\mathbb{N}, \qquad b_i\in[0,\beta], \qquad \nu_i\in[0,\beta-b_i], \tag{4.25}$$

which are analogous to the one of the Holstein model in Eq. (4.5). That analogy tells us that the distribution would have a similar form, but instead of depending on the exponential function we will have to deal to the integral propagator defined in Eq. (1.92). So,

by following the Holstein model case we can start by defining the $p_{\mathscr{P}}(\nu_i|b_i,\beta)$ conditional as proportional to $\mathscr{D}$ and normalize by computing the integral

$$C(b_i,\beta)=\int_0^{\beta-b_i}\mathrm{d}x\mathscr{D}(x)=\frac{\alpha\omega_c}{2}\int_0^{\beta-b_i}\mathrm{d}x\int_0^1\mathrm{d}q\,q^s\frac{\cosh[\omega_c q(\beta/2-x)]}{\sinh[\omega_c q\beta/2]}. \tag{4.26}$$

Such integral can be simplified by switching the two integration and use the hyperbolic function properties to obtain the form

$$C(b_i,\beta)=\frac{\alpha\omega_c}{2s}+\frac{\alpha\omega_c}{2}\int_0^1\mathrm{d}q\,q^{s-1}\frac{\sinh[\omega_c q(\beta/2-b_i)]}{\sinh[\omega_c q\beta/2]}. \tag{4.27}$$

Having that form the conditional can be written using $p_{\mathscr{P}}(\nu_i|b_i,\beta)=\mathscr{D}(\nu_i)/C(b_i,\beta)$. So that, by isolating the Phonons variables as in Eq. (4.7) and substituting the conditional for the length of the phonons one can see how the distribution of the creation $p_{\mathscr{P}}(b_i|\beta)$ must now be proportional to $C(b_i,\beta)$. To transform that function into a PDF we shall normalize it once again in the wanted domain by computing the integral

$$\lambda_p=\int_0^1\mathrm{d}x\,C(x,\beta)=\frac{\alpha\beta\omega_c}{2s}+\frac{\alpha\omega_c}{2}\int_0^1\mathrm{d}q\frac{q^{s-1}}{\sinh[\omega_c q\beta/2]}\int_0^\beta\mathrm{d}x\sinh[\omega_c q(\beta/2-b_i)]=\frac{\alpha\beta\omega_c}{2s}, \tag{4.28}$$

where we use the fact that sinh is an odd function giving a zero integral. Once again we can define the conditional of the phonon creation times as $p_{\mathscr{P}}(b_i|\beta)=C(b_i,\beta)/\lambda_p$ that we can plug in inside the equation for the acceptance rate and add the usual $n_p!$ contribution to have

$$\frac{\left[\prod_{i=1}^{n'_p}\mathscr{D}(\nu'_i)\right]p_{\mathscr{P}}(n_p|\beta)n_p!\prod_{i=1}^{n_p}p_{\mathscr{P}}(b_i|\beta)p_{\mathscr{P}}(\nu_i|b_i,\beta)}{\left[\prod_{i=1}^{n'_p}\mathscr{D}(\nu'_i)\right]p_{\mathscr{P}}(n'_p|\beta)n'_p!\prod_{i=1}^{n'_p}p_{\mathscr{P}}(b'_i|\beta)p_{\mathscr{P}}(\nu'_i|b'_i,\beta)}=\frac{\lambda_p^{n'_p}/n'_p!}{p_{\mathscr{P}}(n'_p|\beta)}\frac{p_{\mathscr{P}}(n_p|\beta)}{\lambda_p^{n_p}/n_p!}=1. \tag{4.29}$$

Without much surprise, we obtain that the distribution of the number of phonons is a simple Poissonian with λ_p as average, giving us the complete picture of the distribution as

$$p_{\mathscr{P}}(n_p|\beta)=\frac{\lambda_p^{n_p}}{n_p!}e^{-\lambda_p},\qquad p_{\mathscr{P}}(b_i|\beta)=C(b_i,\beta)/\lambda_p,\qquad p_{\mathscr{P}}(\nu_i|b_i,\beta)=\mathscr{D}(\nu_i)/C(b_i,\beta). \tag{4.30}$$

From which all the variables can be sampled once again using a standard Poisson sampler and the numerical CDF inverse approach.

Therefore, from this discussion we have obtained two different global updates described by $p_{\mathscr{S}}$ and $p_{\mathscr{P}}$ respectively. The former describing the statistics of the spin variables and giving a way to sample them directly, $\Gamma_{\mathscr{S}}(\mathscr{C}\to\mathscr{C}')$, while the latter $\Gamma_{\mathscr{P}}(\mathscr{C}\to\mathscr{C}')$ act in the same way for the phononic degrees of freedom. Both of them have been implemented inside the LLDMC framework along the local updates described in Sec. (2.2.2) to once again compare the performance. Also, a mixed update has been introduced in the performance study once again by using the same approach as before and simply using both spin and phonons global updates along the local ones but reject the global ones if $|n'_x-n_x|=1$ or $|n'_p-n_p|=1$ respectively. All the simulations were performed setting the tunneling strength to unity, $\Delta=1$, so that all the quantities are

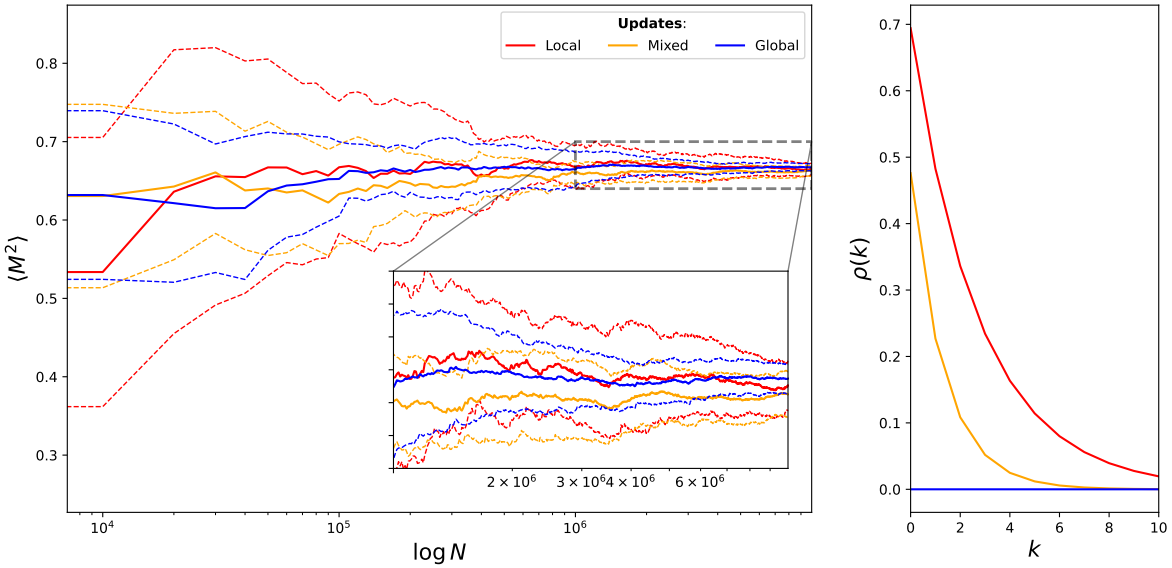


Figure 4.3: Evolution of the estimate for the squared magnetization inside the spin-boson model for the local, global and mixed update schemes. All the simulations were performed with $\omega_c = 10$, $s = 1$ and $\Delta = 1$ using 1E6 thermalization steps, while keeping β and α to the values of 20 and 0.5. Once again 13 parallel runs were performed for every update to compute the average evolution and deviation, displayed as in Fig. (4.1). Also in this case the chain correlation observable is reported on the right side of the image.

expressed in units of Δ. In particular, by following the same approach used in Ref. [9] the bath cutoff frequency was set to $\omega_c = 10\Delta$ and the regime assumed to be homnic, $s = 1$. Again, every simulation is performed using 13 parallel thread to obtain separate uncorrelated estimate of the same observable, that in the SB case is the squared magnetization $\langle M^2 \rangle$. Thus, we studied the behavior of such observable, and it's dispersion, along the chain using the idea used previously inside Eq. (4.16) to highlight the statistical behavior of the different updates. A first simulation was performed using this approach by setting the temperature at $\beta = 20$ and the coupling strength $\alpha = 0.5$ with results shown inside Fig. (4.3). Inside it, we can see how once again the global and mixed update outperform the local one in terms of precision reached at the end of the chain. Anyway, the increase in precision showed in this case results to be much lower compared to the gain expected from the decrease in correlation. In fact, by comparing the dispersion of the update schemes as was done for the previous model only a lowering of a factor ~ 2 and ~ 1.1 for global and mixed updates respectively are obtained. This is unexpected since the correlation time for the local updates in that regime is estimated to be ~ 2.8 for the local update and ~ 1 for the mixed scheme. Based on them, one should expect an increase in performance of a factor of ~ 6 if using the global update and of ~ 3 for the mixed one. Such lower gains respect to the theoretical once can be imputed to a cause analogous to the one that brought to the same performance in the evaluation of the Green's function inside Fig. (4.2). Namely, value of the correlation given by τ_ρ is only a simplistic approximation of the correlation inside the observable and in our case $\langle M^2 \rangle$ posses a complex estimator given by Eq. (2.77). Such value is composed by the rate of two separate estimates: the sign of the diagram itself, and the sign modulated squared magnetization. This generates a complex equation that posses a correlation

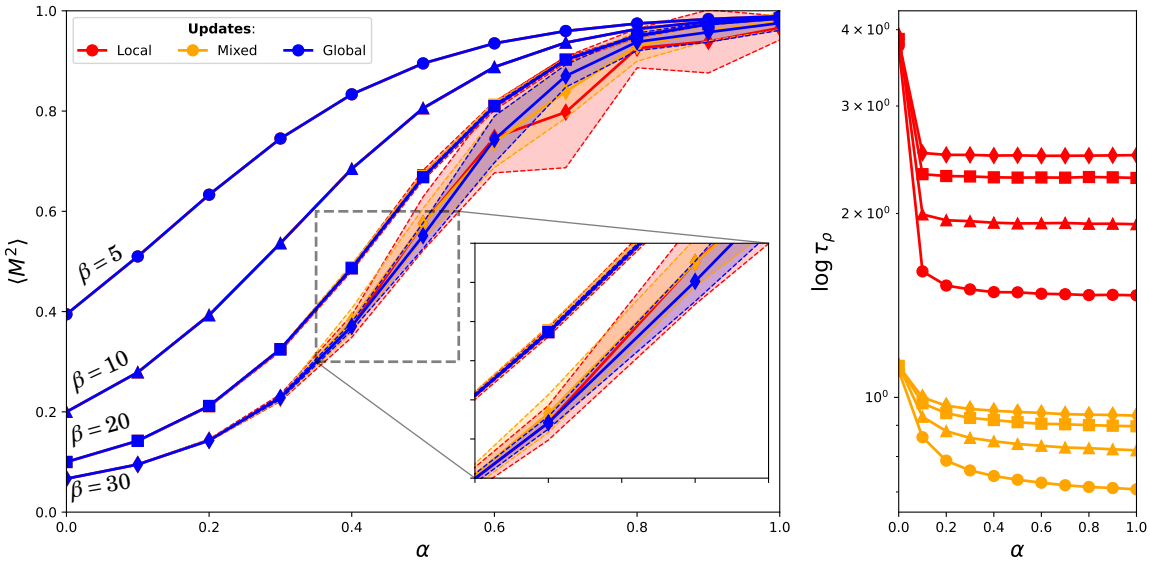


Figure 4.4: Results for the simulation done at different coupling constant α and temperature β using the same specific of the simulation presented in Fig. (4.3). On the left the value of the squared magnetization is reported with the average value between the 13 run in bold line and the confidence interval showed using the colored area. On the right, instead, the value of the correlation time as a function of α and β is presented in log scale showing how the mixed scheme is always better than the local one.

with both spin and phononic degrees of freedom that is not well approximated by our estimate. Meaning that such decrease in performance, compared to the local updates, of our global schemes is mainly due to the presence of the sign problem inside the SB model that bring to a more complex and unstable estimator for $\langle M^2 \rangle$. Still, that does not mean that the local update results in being better than our global ones. In fact, one can see how the global schemes are able to perform better as the phase space increase, being more stable in situations where the local one struggles. To see that, a series of simulations for all the schemes were performed by sweeping both the value of β and α in order to see the evolution of the correlation times and of the observable estimates. In particular the results for the $\langle M^2 \rangle$ and its σ_{M^2} between the usual 13 parallel simulations for every scheme and couple of parameters are shown in Fig. (4.4). The image shows how the results for the schemes are completely analogous for low value of both β and α, meaning high temperatures and low coupling regimes. In those cases, the phase space is smaller making it easy to converge for all of them with smaller errors depending on the correlation. When the temperature start to decrease also the integration domain expands and the estimates become less precise as the coupling increases, but can be seen how where the local update are not able to obtain the right shape the global results are more well-behaved. That can be seen in the $\beta = 30$ curve, where on the $\alpha = 0.7$ entry the blue and orange line are able to maintain the correct shape, while the red local one have a sharp jump. This shows how, even if the decrease inside the dispersion is lower than expected, the enhancement in the statistics given by the decorrelation of the sample can allow obtaining results that shows the right qualitative shape, even if the correlated case gives completely oscillating results.

Curing SB sign problem

The decrease in precision inside the estimate of $\langle M^2 \rangle$ showed by all the update schemes in Fig. (4.4) limits our possibilities in exploring the low temperature regime of the spin-boson system to a limit of $\beta \sim 30$ for high coupling. Such limit is mainly due because of the sign problem that the system has that requires the Monte Carlo average to take into account the sign of the diagrams transforming the direct estimator into the rate of two different observables. In fact, it has been shown how such form gives a much worse convergence complexity respect to the $\mathcal{O}(N^{-1/2})$ of the usual Monte Carlo integration. Having that, as the value $\langle S \rangle$ becomes smaller the deviation, σ_O, of the observable start to grows exponentially with the size of the phase space, making the all computation an *NP-hard* problem [50]. Such wall is not something that can be fought by simply reducing the sample correlation inside a simulation, having that even if we set τ_O to zero the value of σ_O inside Eq. (2.22) would still be too high. Nevertheless, the information that we obtained about the distribution in order to create the global updates can give us insight on how to cope with such a problem and eliminate it completely.

When we talked about the magnetization observable we have described how was possible to obtain better converging estimators by looking at the subsets of diagrams with same probabilities and take the subset average as estimator for them. Here we can do the same by looking at the sign of the diagram more closely and notice that taken a certain spin configuration, given by $\mathscr{S} = \{\tau_i\}_{i=1}^{n_x}$, then the sign of the diagram is defined by the positions of the phonons as

$$S(\mathscr{S}; b_1, e_1, \dots, b_{n_p}, e_{n_p}) = \sigma(b_1)\sigma(e_1)\cdots\sigma(b_{n_p})\sigma(e_{n_p}). \tag{4.31}$$

Still, one can understand that such result can be written as a product of the sign of every single phonon inside the system by rearranging the product as

$$S(\mathscr{S}; b_1, e_1, \dots, b_{n_p}, e_{n_p}) = \prod_{i=1}^{n_p} \sigma(b_i)\sigma(e_i) = \prod_{i=1}^{n_p} \tilde{S}(\mathscr{S}; b_i, e_i). \tag{4.32}$$

So, if we look at a case with a single phonon inside the system one can notice how, as long as $\mathscr{S}$ is fixed, the position of the phonon does not change the value of $|W|$ but only the sign. That can be easily seen since $\mathscr{D}$ depends only on the phonon length v_i and not on starting point. This is giving us a set of diagrams that can be used in order to obtain an average estimator by first separating spin and phononic degrees of freedom as

$$\langle M^2 \rangle = \sum\!\!\!\!\!\!\int_{\mathscr{C}} M^2(\mathscr{S}) S(\mathscr{S}; \mathscr{P}) p_{|W|}(\mathscr{C}) = \sum\!\!\!\!\!\!\int_{\mathscr{S}} M^2(\mathscr{S}) p_{\mathscr{S}}(\mathscr{S}) \sum\!\!\!\!\!\!\int_{\mathscr{P}} S(\mathscr{S}; \mathscr{P}) p_{\mathscr{P}}(\mathscr{P}), \tag{4.33}$$

where we wrote $\mathscr{P} = \{b_i, e_i\}_{i=1}^{n_p}$ and used the result obtained previously that the PDF can be separated in spin and phonon parts. Now we can focus on the phonons and for now assume that only one phonon is inside the diagram, so that by calling $\mathscr{P}_v$ the subset of diagrams having the phonon with length v we can write the sign as

$$S(\mathscr{S}) = \int_0^\beta \mathrm{d}v \sum\!\!\!\!\!\!\int_{\mathscr{P}_v} \tilde{S}(\mathscr{S}; \mathscr{P}_v) p_{\mathscr{P}}(\mathscr{P}_v) = \int_0^\beta \mathrm{d}v\, p_{\mathscr{P}}(v) \sum\!\!\!\!\!\!\int_{\mathscr{P}_v} \tilde{S}(\mathscr{S}; \mathscr{P}_v). \tag{4.34}$$

Where we used the fact that the complete $p_{\mathscr{P}}$ must depend only on the length of the phonon, so that is equal for all the diagrams inside $\mathscr{P}_v$. In this way, one can simply

understand that the diagrams inside $\mathscr{P}_\nu$ are simply the ones where the phonon creation b is shifted inside the range $[0,\beta-\nu]$, having that

$$S(\mathscr{S}) = \int_0^\beta \mathrm{d}\nu\, p_{\mathscr{D}}(\nu) \int_0^{\beta-\nu} \mathrm{d}b \tilde{S}(\mathscr{S};\mathscr{P}_\nu). \tag{4.35}$$

Now, the same trick used for the magnetization can be used in order to rewrite the estimate using an average value of the sign along all of those diagrams as

$$S(\mathscr{S}) = \int_0^\beta \mathrm{d}\nu\, p_{\mathscr{D}}(\nu) \int_0^{\beta-\nu} \langle S(\mathscr{S};\nu)\rangle\,, \qquad \langle S(\mathscr{S};\nu)\rangle = \frac{1}{\beta-\nu}\int_0^{\beta-\nu} \mathrm{d}b \tilde{S}(\mathscr{S};\mathscr{P}_\nu). \tag{4.36}$$

So that, an expression completely analogous to the one of Eq. (2.70) is obtained having so an estimator that allow us to collect a better statistics than the basic one during the simulation. In such description, the reader may ask how the phononic PDF can be cast in order to depend only on ν starting from the form that we have obtained in Eq. (4.30). That is a simple question that can be answered easily by understanding how $p_{\mathscr{D}}(\nu)$ needs to describe the probability of having a phonon with a certain length. Therefore, to obtain that we shall add up all the probabilities of having a phonon with such length created in every possible position, so that basically such probability will take the form

$$p_{\mathscr{D}}(\nu) = \int_0^{\beta-\nu} \mathrm{d}b\; p_{\mathscr{D}}(b|\beta) p_{\mathscr{D}}(\nu|\beta,b) = \int_0^{\beta-\nu} \frac{C(b,\beta)}{\lambda_p}\frac{\mathscr{D}(\nu)}{C(b,\beta)} = \frac{(\beta-\nu)}{\lambda_p}\mathscr{D}(\nu). \tag{4.37}$$

Which can be seen to be effectively normalized by integrating inside $[0,\beta]$ giving the effective probability of one phonon of having a certain length. Knowing that we can now release the condition of having only one phonon inside the diagram, a step that is easy to handle since all the phonons are independent of each others. Meaning that, we know how the total $p_{|W|}$ is composed by the product of the different conditional probabilities related to the different phonons. In this way we can write down the expectation value of the sign $S(\mathscr{S})$ in a complete way by using

$$S(\mathscr{S}) = \sum_{n_p=0}^{\infty} p_{\mathscr{D}}(n_p|\beta) \prod_{i=1}^{n_p} \int_0^\beta \mathrm{d}\nu_i\, p_{\mathscr{D}}(\nu_i) \int_0^{\beta-\nu} \mathrm{d}b\; \langle S(\mathscr{S};\nu_i)\rangle\,. \tag{4.38}$$

One can understand how the contribution inside the product can be easily computed numerically by mean of numerical integration routines. That allow us to further simplify the contribution to the sign even further by noticing how the final integral will depend only on $\mathscr{S}$ so that

$$S(\mathscr{S}) = \sum_{n_p=0}^{\infty} p_{\mathscr{D}} \langle S(\mathscr{S})\rangle^{n_p}\,, \qquad \langle S(\mathscr{S})\rangle = \int_0^\beta \mathrm{d}\nu_i\, p_{\mathscr{D}}(\nu_i) \int_0^{\beta-\nu} \mathrm{d}b\; \langle S\rangle(\mathscr{S};\nu_i). \tag{4.39}$$

From this result we can go further by explicitly using the form of the order PDF in Eq. (4.30), and see how a closed form can be obtained as

$$S(\mathscr{S}) = \sum_{n_p=0}^{\infty} \frac{(\lambda_p \langle S(\mathscr{S})\rangle)^{n_p}}{n_p!} e^{-\lambda_p} = \exp\big(-\lambda_p\,[1-\langle S(\mathscr{S})\rangle]\big). \tag{4.40}$$

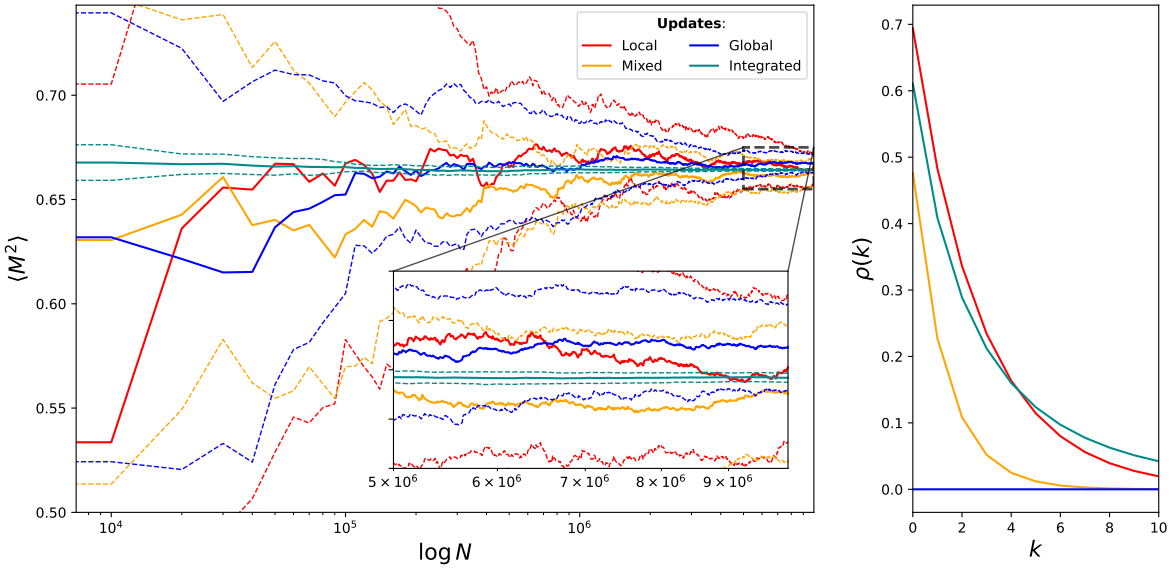


Figure 4.5: Results for the squared magnetization evaluated in the same conditions of Fig. (4.3), but now also the data obtained form the integrated sign-free scheme are shown. Is possible to see how the latter outperform all the other even if the correlation in the chain is still high, increasing above the common local scheme for large value of k.

Such result correspond to the final value of the phononic contribution to the diagram weight, generating a positive average sign that modifies the normal weight of the simple diagrams constituted by only spin degrees of freedom. To see that we can better write down how the final observable looks like once Eq. (4.40) is used inside Eq. (2.77) obtaining

$$\begin{aligned}\langle M^2 \rangle &= \frac{\sum\!\!\!\!\!\!\int_{\mathscr{C}} M^2(\mathscr{S}) S(\mathscr{S};\mathscr{P}) p_{|W|}(\mathscr{C})}{\sum\!\!\!\!\!\!\int_{\mathscr{C}} S(\mathscr{S};\mathscr{P}) p_{|W|}(\mathscr{C})} = \frac{\sum\!\!\!\!\!\!\int_{\mathscr{S}} M^2(\mathscr{S}) p_{\mathscr{S}}(\mathscr{S}) \sum\!\!\!\!\!\!\int_{\mathscr{P}} S(\mathscr{S};\mathscr{P}) p_{\mathscr{P}}(\mathscr{P})}{\sum\!\!\!\!\!\!\int_{\mathscr{S}} p_{\mathscr{S}}(\mathscr{S}) \sum\!\!\!\!\!\!\int_{\mathscr{P}} S(\mathscr{S};\mathscr{P}) p_{\mathscr{P}}(\mathscr{P})} \\ &= \frac{\sum\!\!\!\!\!\!\int_{\mathscr{S}} M^2(\mathscr{S}) e^{-\lambda_p[1-\langle S(\mathscr{S})\rangle]} p_{\mathscr{S}}(\mathscr{S})}{\sum\!\!\!\!\!\!\int_{\mathscr{S}} e^{-\lambda_p[1-\langle S(\mathscr{S})\rangle]} p_{\mathscr{S}}(\mathscr{S})} = \sum\!\!\!\!\!\!\int_{\mathscr{S}} M^2(\mathscr{S}) \frac{e^{-\lambda_p[1-\langle S(\mathscr{S})\rangle]} p_{\mathscr{S}}(\mathscr{S})}{\sum\!\!\!\!\!\!\int_{\mathscr{S}} e^{-\lambda_p[1-\langle S(\mathscr{S})\rangle]} p_{\mathscr{S}}(\mathscr{S})} \\ &= \sum\!\!\!\!\!\!\int_{\mathscr{S}} M^2(\mathscr{S}) p_W(\mathscr{S}).\end{aligned}$$

Basically, since the final form of the sign obtained is always positive it was possible to interpret the product of S and $p_{\mathscr{S}}$ as a probability distribution and use the denominator of the average as the normalization constant. In this way we have completely eliminated the need of the phononic variables inside our system curing the *sign problem* having now a simple direct form to estimate the wanted observable. One can so think at the final result as a way to rewrite the problem as a simple spin system with a diagram's weight that is modified from the simple case by the presence of an external environment through an exponential factor as

$$W(\mathscr{S}) = \left(\frac{\Delta}{2}\right)^{n_x} e^{-\lambda_p[1-\langle S(\mathscr{S})\rangle]}, \tag{4.41}$$

with the mean value obtained as in Eq. (4.39). That means, we can perform the simulation by still using the Γ_{add_x} and Γ_{rem_x} that we defined inside Sec. (2.2.2) and simply

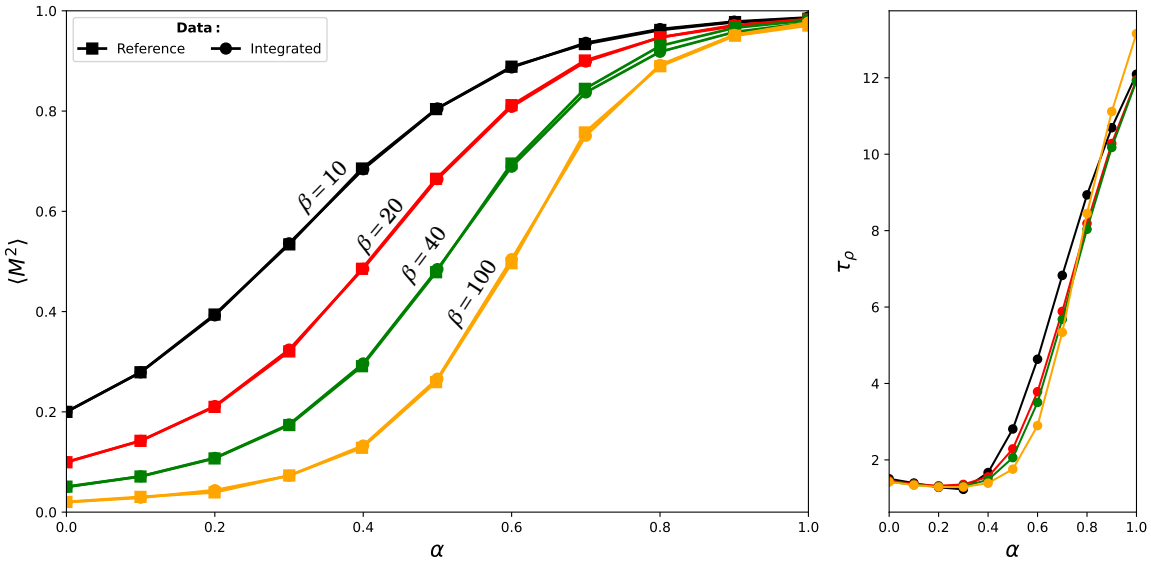


Figure 4.6: Results for the magnetization observable at different values of β and α reaching both low temperature and high coupling regime. Value of reference are taken from the work [9] while the integrated scheme is the one created inside this work and is the mean of 13 independent simulations giving rise to an error that is smaller than the size of the line. Every simulation was run for 1E7 steps after 1E6 thermalization ones, and also the correlation time was collected and reported on the right.

change the weight we are using. In this way we would have an ergodic update that samples all the spin diagram distribution arriving to the same result as our previous scheme containing also an integration over the phonons.

After all this effort to obtain such sign problem free scheme we want to test its performance against the one seen so far and see how much can improve the statistics. Therefore, the same simulation used to obtain the results inside Fig. (4.3) has been performed also for the local integrated scheme. The results obtained for the observable $\langle M^2 \rangle$ collected are shown alongside the previous ones in Fig. (4.5). From such figure, one can see how the sign-free scheme completely outperform all the other one that we have created. The error obtained from it is lower than the final one of the global scheme already in the first few thousand steps. All while retaining a high correlation in the chain, that is larger than the simple local scheme for long range sample, meaning high k. That is, eliminating the need of the sign we have not only reduced the dimension of the phase space to explore in order to converge, but we have also obtained an observable with an intrinsic deviation, σ_O from Eq. (2.20), much smaller. In fact, it's known how the more an observable $O(\mathscr{C})$ oscillate inside the space and more the uncertainty will be higher. Before, with the value of M^2 we needed also to multiply for S so that negative and positive value could appear making it oscillate rapidly also for small modification the diagram. Here such behavior is erased obtaining an incredible boost in the performances which allow us to push the study of the system at lower temperatures. In particular, a series of simulations have been performed in order to recreate the data for the single qubit case inside [9] gently given to us by Andrey Mishchenko. So, by using the same values of the Δ, s and ω_c values used so far we have performed a sweep for the α and β value. For each couple of values the usual 13 simulations have been performed

to obtain once again the average behavior and the error on it, which are reported alongside the reference in Fig. (4.6). We can see how our scheme perfectly corresponds to the reference data reported in Ref. [9] where they used a complex path-integral formulation of the integration scheme that has also brought them to a sign-free scheme. Nevertheless, the reported time needed for the results at $\beta = 100$ to converge is one day in their work, while for the scheme obtained by pure statistical consideration few minutes were enough to perform the same task. A large boost in performances that can be increased further by a more optimized update scheme. In fact, from the plot of τ_ρ it is possible to see how the correlation time highly increase as the coupling becomes larger, a behavior that is expected near phase transitions as in this case. Thus, this reformulated sign-free version of the spin-boson problem would be a good test case to see a large performance boost coming from a global update. Nevertheless, the new weight function in Eq. (4.41) is too complicated to tackle it analytically as we have done so far.

4.3 Neural global updates

So far we were able to easily describe the statistics of the models under investigation by direct inspection of their analytical form. Nevertheless, we have also seen how such approach is really limited only to simple cases where the diagram's weight is simple and, most importantly, separable. This has to be seen as a large limitation, especially considering how the use of global updates becomes more valuable the more the chain is correlated. Something that mainly happens inside more complex cases, as we have seen in the integrated version of the SB model. Therefore, in order to overcome such barrier we want to switch to a numerical approach that gives access to the complete distribution. For that we adopt the Normalizing Flows architecture discussed in Ch. (3). Such architecture has the potential to give us a way to both sample from and infer p_W, giving us the possibility of creating global updates that can be used to construct an uncorrelated Markov Chain even in complex cases.

Here we will explain the approach that we have used in order to create such model for the single site Holstein case, focussing on the challenges present in using these architectures for approximating diagrammatic distributions. Then, the results obtained for the update created form the model are shown and confronted with the exact ones obtained from the theory.

Model construction

We have already pointed out, in Sec. (3.3), how a standard Normalizing Flow architecture would not be able to catch the correct behavior needed to represent a diagram's distribution. Common network are not able to reproduce the integer variables part of the distribution, and do not take into consideration time ordering of interaction times. Still, possible solutions for both problems were introduced by simple modifications of the architecture and loss that would make the algorithm converge to the right result. Thus, our goal now is to show how such claim is indeed true by using them on a model and successfully apply it in the context of the single site Holstein polaron.

To see how we can describe the distribution of the Holstein diagrams we can recall their representation defined inside Eq. (4.3). The diagram is written as a vector whose

dimension depends on the order of the diagram itself, with an n-th order diagram transformed into an $n+2$ dimensional vector. In Machine Learning terms, that would mean a really complicated architecture depending on a hyperparameter defining the maximum possible order of the generated diagrams. Meaning the Network would work with a fixed dimensional vector, given by the maximum order, but only the variables corresponding to the order under inspection would be modified. Such a treatment would easily lead to overtraining of certain part of the network or instabilities based on how the order is treated. Luckily, in this case, that can be avoided by using the trick of *direct data sampling*, described in Sec. (3.3.1), that will ease the work by first sampling the order and then using it to tell the model what to do. In particular, if we also sample the electron's time of flight τ by using its global update Γ_{chg} then the only variables left out for a complete description of the diagram are all the phonons' creation and destruction times $\{b_i, e_i\}$. We know how such set of variables are independent of one another, meaning that we only need to approximate the distribution of the couple $p(b,e|\tau)$ and then sample the wanted amount of phonons from it. In the end, the final model we are going to consider will only depend on three variables $[b,e,\tau]$, with τ treated as a parameter that modify the transformations performed, while n would give us the number of phonons. So that, the final PDF that we will construct would have the form

$$\tilde{p}(n,\{b_i,e_i\}_{i=1}^{n}|\tau) = p_W(n|\tau)n!\prod_{i=1}^{n} p(b_i,e_i|\tau), \tag{4.42}$$

with the needed $n!$ coming from the arbitrary ordering of the phonons sampled, as we have seen also in the analytic work. Such form highly simplifies the amount of work of the NF model, so that now it only needs to focus on a parametrized two variable distribution of which we know the unnormalized form of the PDF. In fact, from the discussion in Sec. (3.3) we know how $W(\mathscr{C})$ represent such function, but since only the phonons'distribution needs to be approximated only the phononic propagator is needed. Meaning that we can perform the training using the *revers KL divergence* as in Eq. (3.19) substituting $\exp[-\Omega(e-b)]$ at $\bar{p}$. Still, that does not give a complete picture since we also need time ordering between the creation and destruction of the phonons. To insert that we selected the *weight modification* approach leading to follow Eq. (3.51) and use the following expression as our target propagator

$$\tilde{W}(b_i,e_i;\tau) = \exp(-e_i)\exp(b_i)\sigma_\alpha(b_i)\sigma_\alpha(e_i-b_i)\sigma_\alpha(\tau-e_i), \tag{4.43}$$

where Ω was set to unity to have results comparable with the ones in Sec. (4.2.1). Using such a form for the weight inserts a hyperparameter in the model, α, describing how close the modified weight is to the original one. In our study the value of 50 was selected for α and used throughout the work, that value allow for the learning algorithm to remain stable while retaining a good agreement with the real weight, as can be seen in Fig. (4.7). In this way all the main problems for the approximation of the complete distribution were solved, leaving us with the task of training different model to select the structure that reaches the best result.

In order to construct out flow model we needed to make the choice of the *conditioner* and the *transformer* selecting between the various possibilities presented in Sec. (3.2). Thus, we have started by choosing the conditioner as the standard *masked autoregressive* approach. Such choice was guided by the low dimensionality that the final

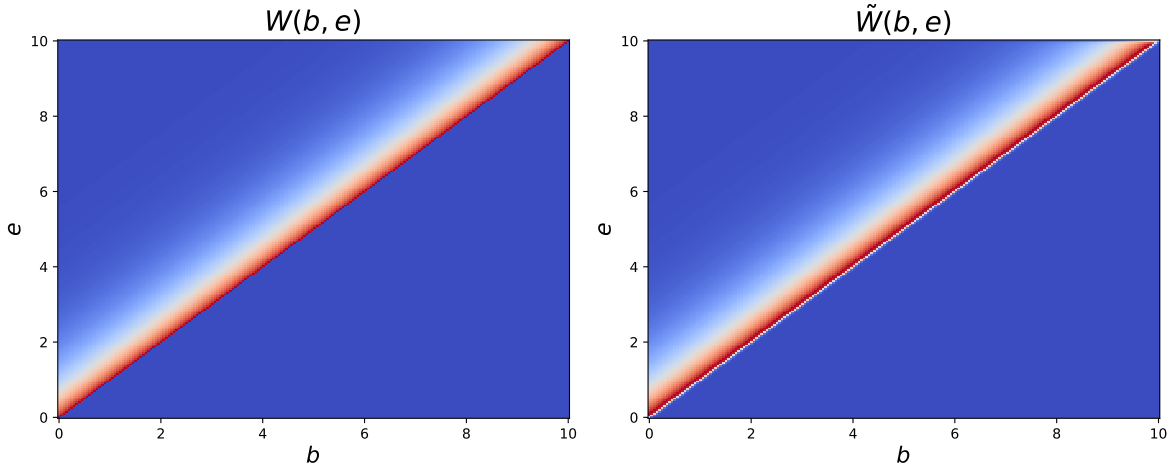


Figure 4.7: Representation of the 2D densities that are obtained by using the exact Holstein weight W or the modified $\tilde{W}$ containing also the time ordering information, which is inserted by mean of sigmoids. The effect of such functions can be seen in the presence of a white front in $\tilde{W}$ due to a more graduate decrease of the function respect to the real step one. Both images are obtained by setting the electron time of flight at 10 and the modified uses as hyperparameter $\alpha = 50$.

simplified model has, in this way using the *coupling* conditioner would give no benefit since it's less expressive and would require the use of permutation layers. The masked conditioner doesn't need additional layers and is equally fast in low dimensions. For these reasons we used a masked linear NN as conditioner in every layer of the model to compute the parameters $\boldsymbol{h}$ of the transformer. In particular, these networks are composed by: an input layer that takes τ and b as inputs, an output layer that gives $[\boldsymbol{h}_b, \boldsymbol{h}_e]$, and a hidden layer with a variable number of neurons given by $2 \times \eta$. This is a structure selected for simplicity, since allows us to easily divide the vector in separate groups depending on the variables on which they depend, so that we have an evolution of the type

$$[\tau, b] \longrightarrow \left[\{c_i(\tau)\}_{i=1}^{\eta}, \{c_i(\tau, b)\}_{i=1}^{\eta}\right] \longrightarrow [\boldsymbol{h}_b(\boldsymbol{c}(\tau)), \boldsymbol{h}_e(\boldsymbol{c}(\tau), \boldsymbol{c}(\tau, b))]. \tag{4.44}$$

In this way we are sure to have an ending set of parameters where $\boldsymbol{h}_i$ depends only on the variable $< i$, as needed in an autoregressive flow, but leaving us the freedom of increasing the expressiveness by increasing η. The selection of the transformer, instead, wasn't so straightforward since no prior assumptions on which would perform better in terms of computational cost vs approximation can be done. Thus, we used the C++ Normalizing Flow framework implemented in LLDMC to construct and train various autoregressive models using both *affine* and *spline-based* transformers, and see who performs better. So, for both cases a flow of the type seen in Fig. (4.8) was created, where a diagonal Gaussian was used as initial distribution and the final target was $\tilde{W}$. The training, instead, was performed in every model by sampling a batch of 1×10^4 samples from the model and compute the average loss out of them, from which the stochastic gradient descend algorithm with varying learning rate was used to minimize it. Specifically, we used a learning rate of 1×10^{-4} for the affine training and the first 3×10^4 steps of the spline one, after which we switched to a learning rate of 1×10^{-5}. At last, since the model needs to successfully approximate the distribution at different values of τ, the training has been performed in order to have the time of flight of the

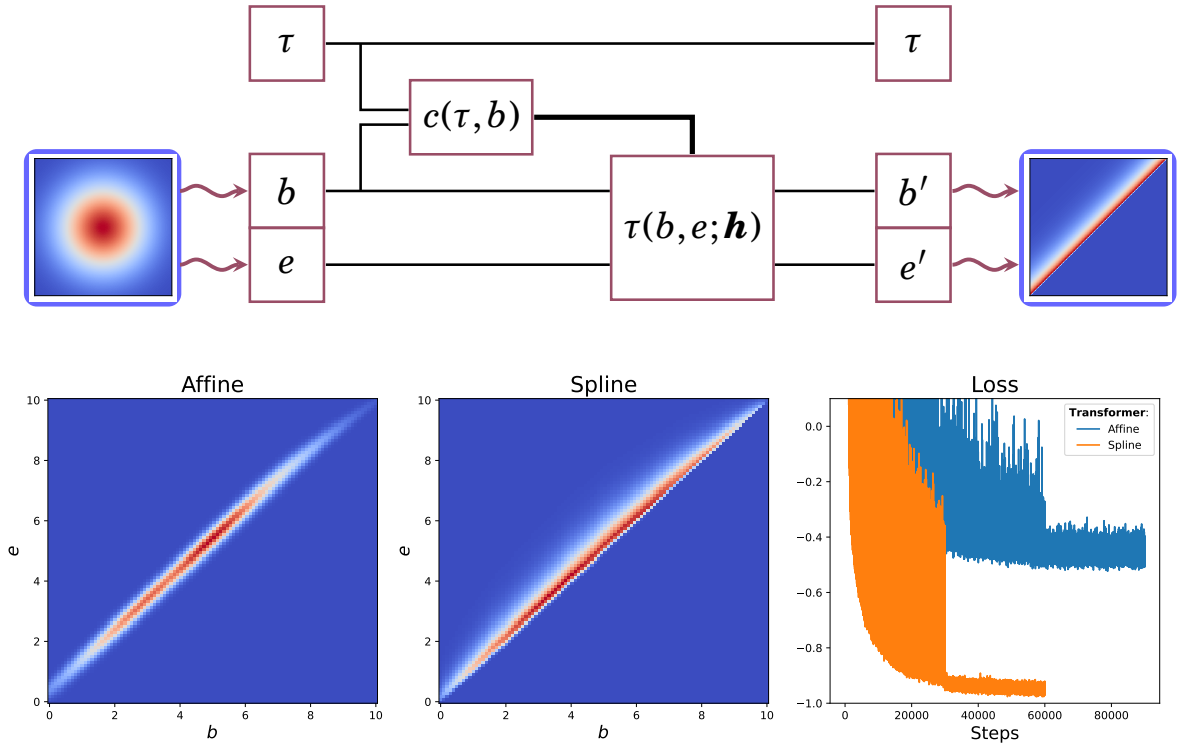


Figure 4.8: General representation of the final model used to approximate the distribution of phonons variables. It can be seen how only b and e gets transformed, while τ is kept constant being only a parameter that simply influence the form of the distribution. On the bottom one can also see the main results for the best model using affine and spline-based transformer respectively, with also the loss evolution during training compared.

1×10^4 samples homogeneously distributed in the range $[0, 20]$. To do that we divided the wanted range into 10 equally spaced points and extracted 1×10^3samples for every one of the 10 value of τ. In this way the model would be equally trained in the whole domain of interest of τ. Since the training domain is restricted to $[0, 20]$ we have set the boundary value for the spline domain to 22, so that the whole variable domain is covered and only the number of bin inside such domain needs to be selected as hyperparameter. Therefore, the main hyperparameters we needed to tune by training different models were: the value of η, the number of transformation to stack, and the number of bins used in the spline transformation.

A lot of different combinations of hyperparameters have been tried in order to see which lead to the best model for both spline and affine transformers, and the results are shown in Fig. (4.8). The affine result has been obtained stacking 4 transformations together while using an η of 50, while the spline only needed two transformations and an η of 20 with 5 bins. This leads to similar performances on the computational level, even if the spline one is still slower by a factor of ~ 1.2 but leading to much better results. In fact, by comparing the densities obtained for $\tau = 10$ to the one of the exact $\tilde{W}$, reported in Fig. (4.7), we can clearly see how the one obtained from the spline is much closer to the wanted result. Also, this can be seen by comparing the losses where the one of the spline is more than two times smaller than the affine one. Where the fact that both are negative, even if the KL divergence is positive definite, is due to the

use of the unnormalized density for our target. All of these reasons lead us to believe how the much greater expressiveness of a spline-based transformer makes it the better candidate at describing a complex non-linear distribution such as the one of this model. Thus, in the next and last part of this work we are going to use the spline-based model to construct a global update since was revealed to bring the best results in approximating the target distribution.

We now have a way to generate the phonons by approximating $p(b,e|\tau)$ using NF, but to complete the model we also need to define the $p_W(n|\tau)$ of Eq. (4.42). Such distribution can be constructed using numerical methods by collecting the statistics of the orders from a quick simulation depending on the value of τ. This allows us to directly sample from it using the CDF inverse, see App. (A), and evaluate the probability of a certain order value. Still, this approach would be completely analogous to use the exact form for such distribution that we obtained previously inside Eq. (4.13). For this reason the exact distribution has been directly used to sample the number of phonons instead of the order, so that the final PDF of the model was given by

$$\tilde{p}(\mathscr{P}|\tau) = \lambda^n \prod_{i=1}^{n} p(b_i,e_i|\tau), \qquad \lambda = g^2\left[\tau + 1 - e^{-1\tau}\right]. \tag{4.45}$$

Where, once again, Ω was set to unity for compatibility with previous results and $\mathscr{P}$ was used to represent the phonons'variable. Such form completes our model, allowing us to directly sampling the number characteristic of all phonons inside the diagram depending on the time of flight of the electron. Which can be sampled inside the chain using the already known and optimized update.

Neural Markov Chain

Once the wanted model has been created using it to generate a Markov Chain is a straightforward process. In fact, NF architecture gives us access to both sampling from, and evaluation of, the approximated version of the target PDF that will give rise to an update of the type

$$\Gamma_{neu}(\mathscr{C} \to \mathscr{C}') = \tilde{p}(\mathscr{C}') = p_{\mathbf{z}} \circ T^{-1}(\mathscr{C}')\left|\det J_{T^{-1}}(\mathscr{C}')\right|. \tag{4.46}$$

Where $p_{\mathbf{z}}$ is the starting distribution of the model, while T is the final complete bijection that maps to the wanted PDF. It's so easy to see how the diagrams proposed in this way are sampled in an independent way respect to the current one in the chain, and their acceptance rate inside it can be directly written as

$$\mathscr{A}(\mathscr{C},\mathscr{C}') = \min\left\{1, \frac{W(\mathscr{C}')\Gamma_{neu}(\mathscr{C}' \to \mathscr{C})}{W(\mathscr{C})\Gamma_{neu}(\mathscr{C} \to \mathscr{C}')}\right\} = \min\left\{1, \frac{W(\mathscr{C}')\tilde{p}(\mathscr{C})}{W(\mathscr{C})\tilde{p}(\mathscr{C}')}\right\}. \tag{4.47}$$

Thus, if the model is ergodic, the chain can be completely and efficiently constructed by simultaneously sample a set of diagrams with respective probabilities and accepting or rejecting them inside the chain one by one, In this way, if $\tilde{p}$ well approximate W, the acceptance rate should be high enough to eliminate correlation, or at least reduce it significantly respect to a simpler local update.

In our construction of the single site Holstein model we have used a final architecture which is not completely ergodic. As a matter of facts, our model is able to sample only

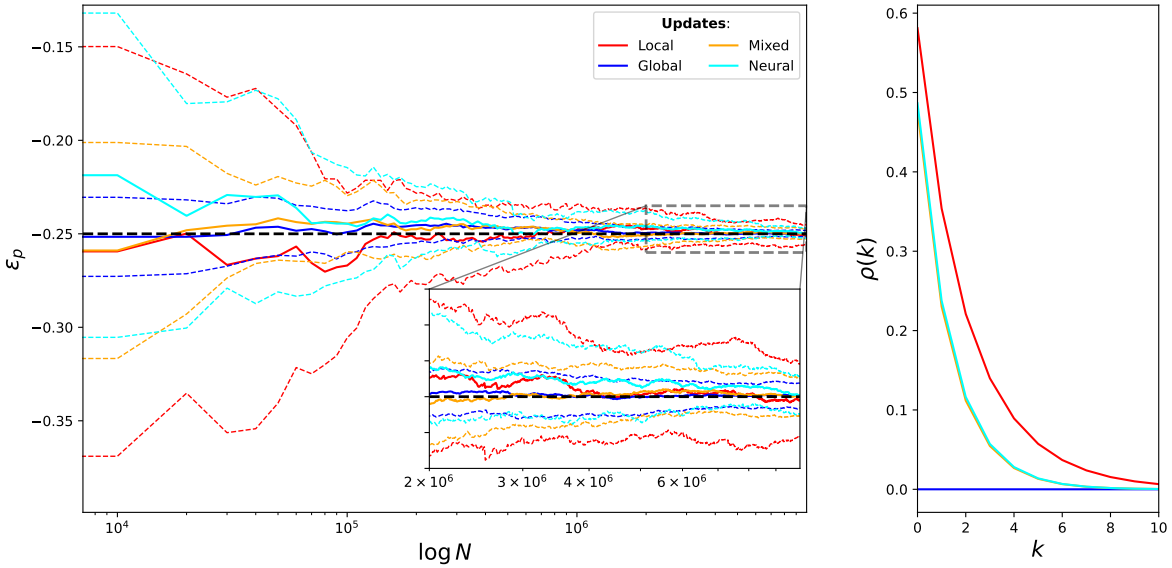


Figure 4.9: Evolution of the estimate for ε_p and correlation inside the single site Holstein polaron for the local, global, mixed and neural mixed update schemes, with the correct analytical result represented as the black dashed line. All the specific of the simulations and the results for the global, mixed and local curves are the same as in Fig. (4.1).

the phononic variables of the diagram in question while the electron's time of flight is obtained from the already known global update. Meaning that the final ergodic form of the update is given by $\Gamma(\mathscr{C} \to \mathscr{C}') = \left[\Gamma_{neu}(\mathscr{C} \to \mathscr{C}') + \Gamma_{chg}(\mathscr{C} \to \mathscr{C}')\right]/2$, analogous to the one used in the analytical study of Sec. (4.2.1). Such choice allowed for a great simplification of the model construction, but leaves behind a slower sampling for the chain entries. That being due to the form of T which depends on τ as a parameter, so that $\tilde{p}$ itself changes based on the time of flight of the previous diagram. Meaning that, it's not possible to directly sample a set of diagrams based on it since we need to know the current value of τ to correctly sample or evaluate $\tilde{p}(\mathscr{C}|\tau)$. This limits us to a sequential approach every time the Γ_{neu} update is called that is composed as follows

1. Collect the current value of τ by the last diagram in the chain;
2. Use such value to sample the phononic variables starting by the number, from $p_W(n|\tau)$, and then all the phonons from the model $p(b, e|\tau)$;
3. Compute the probability of the new and current diagram's phonons by mean of Eq. (4.45), using the current τ;
4. Evaluate the diagrams'weights and compute the acceptance rate of the new diagram.

A lot of extra steps are needed respect to an ergodic model approach, or the usual local update one where the acceptance rate was given by an analytical formula much quicker to compute than the whole weight. That is hinting us how the construction of the chain through Γ_{neu} and Γ_{chg} alone would be too much slower respect the other possibilities at our disposal. For that reason instead of focussing on the complete global update we

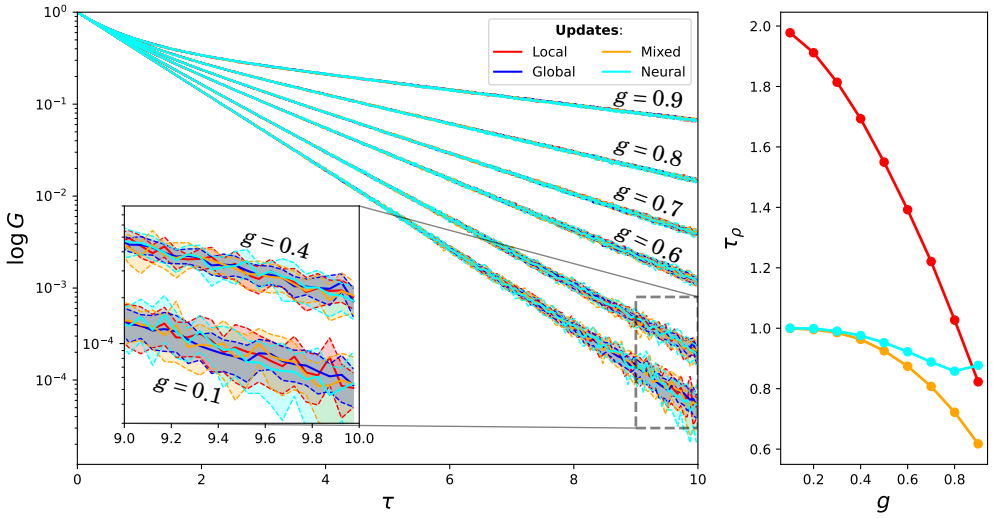


Figure 4.10: Representation of the Green function and correlation time for different coupling constant with the data for global, local and mixed are the same of Fig. (4.2) and the neural ones were computed using the same specifics.

have directly studied the mixed version of that update, constructed by both the neural and the local updates as in Sec. (4.2.1). That allowed for a much faster update that we wanted to compare on a statistical level with its analytic counterpart to see how effective our model was at approximating the real distribution. Thus, we used the final mixed neural update to construct 13 parallel Markov Chains that have been used in order to evaluate $\bar{\varepsilon}_p$ and σ_ε following the same specifics of Sec. (4.2.1). The results obtained are shown in Fig. (4.9) along with the ones previously obtained for the other analytical updates. From their comparison we can easily see how the neural update is performing in a way that is nearly identical to the analytical mixed update. That can be seen both in the evolution of the observable's standard deviation and in the estimator of the correlation along the chain. In both cases the two update schemes nearly overlap, with a slight increase in the initial value of σ_ε, where the analytical mixed seems to perform better. Such similarity in the performance of the two can be better seen in Fig. (4.10) with the results of the Green function and correlation time evaluated for different coupling strength. In particular, from the evolution of τ_ρ one can see how the correlation in the chain for the mixed and neural update is the same for low coupling and diverges as it grows larger. That is an expected result since at lower coupling the order remains smaller and so also the electron's time of flight, thus remaining in the range of τ where the model better approximates the real PDF. Higher order means that more phonons needs to be sampled at a time, so that if the value of $p(b,e|\tau)$ contains a certain error respect to the real p_W sampling n phonons from it means n-th time the error of a single sampling. Bringing us to a worst performance compared to the exact update as g increases. Still, the results obtained showed how the neural update can successfully reduce the correlation inside the chain to a significative amount, generating a mixed update that performs much better than the local one with performances close to the analytical counterpart. This shows how the model was able not only to approximate the

correct PDF, but also with a good enough precision to reach the same statistics of the exact result. On a computational point of view, instead, the neural update remains much slower respect to the exact one, as anticipated. With an overhead in complexity that increases the time needed to perform the update by a factor of more than 10 respect to its competitors. That brings a really large decrease in performances, even compared to the simple local updates, mainly due to the simplicity of the model that makes normal updates much quicker to perform. In a more complex model, with larger data structures or higher correlation, usual updates may increase in complexity while the neural approach would remain nearly the same.

5

Conclusions

The work done in this thesis aimed to investigate new approaches to the construction of updates for the Diagrammatic Monte Carlo algorithm that could improve performances respect to today's alternatives. To achieve that, a careful study of the theory behind Feynman diagrams formalism at both zero and finite temperature has been conducted in Ch. (1). The Holstein and Spin-Boson models were also introduced by deriving the associated model Hamiltonians that have been used as test cases for applying our new updates. Such updates, were constructed in order to reduce sample correlation inside the Markov Chain of diagrams used to evaluate the observables averages. That is because, as shown in Ch. (2), correlated samples result in less statistical value, requiring a larger chain to reach the same precision of uncorrelated ones. Thus, we have shown how one possible way of alleviating such problem is the use of the so-called *global updates*, a kind of updates with a unitary acceptance rate proposing new diagrams directly following the target distribution. Here two proposal for constructing such type of updates were done: an analytical based approach, and a Machine Learning driven one. The updates produced from both the approaches were successfully implemented in the newly developed LLDMC C++ package [32] and tested respect to the normal *local update* approach, also implemented using the same framework.

At first, the analytical global updates were constructed on both the introduced tests models obtaining the complete form of the diagrams'probability distribution p_W. Such updates were tested by using them to construct Markov Chains of diagrams and study their statistical properties against the one constructed with the local approach. Converged results were obtained, with a decrease in the number of samples needed in order to obtain a wanted accuracy related to the decrease in the correlation time τ_ρ of the chain. In particular, an improvement of ~ 3 and ~ 2 times in the value of the observable dispersion was obtained for the Holstein and SB models respectively. Such reduction showed how the number of sampled needed to obtain the final estimate using the global approach is reduced by the same amount respect to a chain constructed with the usual local one. All this results in a great boost in performances, showing how the approach proposed is indeed effective giving an improvement that depends on the specific model under investigation. Still, the advantages of this approach are not limited to the decorrelation of the chain, but giving us insight on p_W further simplifications to the integration can be obtained. In fact, we used such knowledge in order to directly integrate the phonons variables inside the SB model, allowing for counting only spin diagrams with a

modified weight and curing the sign problem. That brought to a tremendous increase of the convergence properties that allowed for the study of the model magnetization even at really low temperatures.

Anyway, even if the approach was deemed as fruitful it is still not applicable on a general way, but only limited to models possessing a simple and separable weight function $W(\mathscr{C})$. This brought us to the idea of implementing a second, data-driven, approach able to construct such global updates in a more general contest starting simply by the knowledge of the weight function W. In order to do that we introduced the Normalizing Flow (NF) architecture inside Ch. (3) proposing it as a way to numerically approximate p_W and sample from the final model PDF $\tilde{p}$. The general model proposed was then successfully applied in order to approximate the single site Holstein distribution and used as a global update to construct the Markov Chain. This approach showed convergence properties slightly inferior to the one of the analytical counterpart, proving how the approach can produce a $\tilde{p}$ that is effectively close to the real p_W reducing sample correlation. Still, even if the applicability of this numerical approach is more wide respect to the analytic case it is also computationally much more expensive. The time needed to sample from a NF model was estimated to be ~ 10 times higher respect to usual sampling method, making the performance increase obtained from decorrelation meaningless. Nevertheless, such a harsh increase was mainly due to our specific case where the final specific architecture used didn't allow us to exploit the parallelization potential of the NN, and the simplicity of the model made the correlation not so high. For these reasons, interesting future developments to make this approach more interesting and valuable comprehend: the creation of an ergodic model whose sampling can be parallelized, and the application to more complicated models with really high τ_ρ values. For example a good possible application would be the creation of a global update for the integrated SB model, where the form of W becomes too complex to be handled analytically and the correlation time skyrocket at low temperatures.

A

Sampling through CDF inverse

When working with Monte Carlo algorithms being able to efficiently sample from a general 1D distribution is always a matter of interest that can have huge impact on the performance of a software. Because of that, the subject has been extensively studied over the years and now the most extensively used method to perform such a task is the *Commulative Distribution Function inverse*, or CDF inverse, algorithm. To understand it let $\mathbb{A}$ be an interval of $\mathbb{R}$ and $p : \mathbb{A} \to \mathbb{R}_+$ a PDF defined in such domain. Now, we can define the CDF of such distribution as a function of the type $F : \mathbb{A} \to [0,1]$ by using the following integral expression

$$F(x) = \int_m^x \mathrm{d}t\, p(t), \qquad\qquad m = \min_x \mathbb{A}. \tag{A.1}$$

Defined in this way it's easy to see how F results in being monotonically increasing due to the positive definiteness of p that bring a derivative of type

$$F'(x) = p(x) > 0, \qquad\qquad x \in \mathbb{A}. \tag{A.2}$$

Since the function is monotonic it means that is also injective and so invertible in its domain, giving rise to a map F^{-1} that bring the entries inside the interval $[0,1]$ into $\mathbb{A}$. That is pretty interesting because standard PRNG are able to quickly sample random variable uniformly distributed inside $[0,1]$, giving rise to a uniform distribution $U(0,1)$. Having now access to a bijection that translates the unitary interval into the wanted domain we can recall the change of variable trick, discussed in Sec. (3.1), and see how

$$\int_{[0,1]} \mathrm{d}u\, p_U(u) = \int_{[0,1]} \mathrm{d}u\, 1 = \int_{\mathbb{A}} \mathrm{d}t\, F'(t) = \int_{\mathbb{A}} \mathrm{d}t\, p(t), \qquad t = F^{-1}(u). \tag{A.3}$$

Meaning that, if sample u drawn from $U(0,1)$ is transformed using F^{-1} the statistic of the new random variable $t = F^{-1}(u)$ will follow the form of p. That's a powerful tool, since by knowing the form of p one can easily construct F by numeric integration inside the domain of definition, or analytical integration where is possible. After that, the inverse can be obtained through the use of the bisection algorithm to create a lookup table that approximates the CDF inverse with a wanted precision depending on the number of bins used to represent $\mathbb{A}$. In this way w can use standard PRNG to uniformly

sample in the unitary domain and then, at the cost of a simple lookup, apply F^{-1} to obtain a random variable as was sampled from p.

To make an example we can have a look at the sampling from an exponential distribution, which is simple enough to be completely treated analytically. So, imagine we want to sample from an exponential distribution of the type $f(x) = \exp(-\alpha x)$ in the range $[a, b]$. In order to do that we shall normalize f in the wanted domain, so that we can easily obtain the normalization constant by integration

$$C = \int_a^b \mathrm{d}x\, e^{-\alpha x} = \frac{1}{\alpha}\left(e^{-\alpha b} - e^{-\alpha e}\right). \tag{A.4}$$

Thus, the PDf we are working with has the form $p(x) = f(x)/C$ and knowing it we can write down the CDF by simply using the definition as

$$F(x) = \int_b^x \mathrm{d}x\, p(x) = \frac{e^{-\alpha b} - e^{-\alpha x}}{e^{-\alpha b} - e^{-\alpha e}}, \tag{A.5}$$

where the explicit form of C was used. In this case, we can also easily invert such function analytically thanks to the simplicity of the expression and obtain the complete expression as

$$F(x) = \frac{e^{-\alpha b} - e^{-\alpha x}}{e^{-\alpha b} - e^{-\alpha e}} = u. \tag{A.6}$$

Then, by using some algebra one can obtain the following inverted expression

$$x = -\frac{1}{\alpha}\log\left[e^{-\alpha b} - u\left(e^{-\alpha b} - e^{-\alpha e}\right)\right] = F^{-1}(u), \tag{A.7}$$

so that the inverted CDF can be written analytically and used to transform a uniform random variable in the unitary interval into an exponential one in an interval of choice.

A similar approach can be used also in the case of integer distributions, which still leads to some simplifications since no inversion of the CDF is needed in that case. In particular, by letting $\mathbb{A} = \{a_0, a_1, \ldots\}$ an ordered subset of $\mathbb{N}$ and p a probability distribution on it, the CDF is given by

$$F(n) = \sum_{i=0}^{n} p(a_i), \qquad\qquad F : \mathbb{A} \to [0, 1]. \tag{A.8}$$

Once that is constructed applying the inverse it's really easy and no table needs to be constructed. The idea is simply to define F^{-1} as a step function of the type

$$F^{-1}(u) = a_i \qquad \text{if} \qquad F(i) < u < F(i+1), \tag{A.9}$$

which is really easy to apply through a single while loop, allowing us to sample a uniform rel value inside $[0, 1]$ and obtain an integer one distributed as p.

Bibliography

[1] M. S. Albergo, G. Kanwar, and P. E. Shanahan. "Flow-based generative models for Markov chain Monte Carlo in lattice field theory". In: *Phys. Rev. D* 100 (3 Aug. 2019), p. 034515. DOI: 10.1103/PhysRevD.100.034515. URL: https://link.aps.org/doi/10.1103/PhysRevD.100.034515.

[2] Malte Algren et al. *Flow Away your Differences: Conditional Normalizing Flows as an Improvement to Reweighting*. 2023. arXiv: 2304.14963 [hep-ph].

[3] Lynton Ardizzone et al. "Guided Image Generation with Conditional Invertible Neural Networks". In: (2019). DOI: 10.48550/ARXIV.1907.02392. URL: https://arxiv.org/abs/1907.02392.

[4] F Aryasetiawan and O Gunnarsson. "The GW method". In: *Reports on Progress in Physics* 61.3 (Mar. 1998), p. 237. DOI: 10.1088/0034-4885/61/3/002. URL: https://dx.doi.org/10.1088/0034-4885/61/3/002.

[5] Stefano Baroni et al. "Phonons and related crystal properties from density-functional perturbation theory". In: *Rev. Mod. Phys.* 73 (2 July 2001), pp. 515–562. DOI: 10.1103/RevModPhys.73.515. URL: https://link.aps.org/doi/10.1103/RevModPhys.73.515.

[6] G. E. P. Box and Mervin E. Muller. "A Note on the Generation of Random Normal Deviates". In: *The Annals of Mathematical Statistics* 29.2 (1958), pp. 610–611. DOI: 10.1214/aoms/1177706645. URL: https://doi.org/10.1214/aoms/1177706645.

[7] Scott Chen and Ramesh Gopinath. "Gaussianization". In: *Advances in Neural Information Processing Systems*. Ed. by T. Leen, T. Dietterich, and V. Tresp. Vol. 13. MIT Press, 2000. URL: https://proceedings.neurips.cc/paper_files/paper/2000/file/3c947bc2f7ff007b86a9428b74654de5-Paper.pdf.

[8] Kookjin Chun and Norman O. Birge. "Dissipative quantum tunneling of a single defect in Bi". In: *Phys. Rev. B* 48 (15 Oct. 1993), pp. 11500–11503. DOI: 10.1103/PhysRevB.48.11500. URL: https://link.aps.org/doi/10.1103/PhysRevB.48.11500.

[9] G. De Filippis et al. "Quantum phase transition of many interacting spins coupled to a bosonic bath: Static and dynamical properties". In: *Phys. Rev. B* 104 (6 Aug. 2021), p. L060410. DOI: 10.1103/PhysRevB.104.L060410. URL: https://link.aps.org/doi/10.1103/PhysRevB.104.L060410.

[10] Laurent Dinh, David Krueger, and Yoshua Bengio. "NICE: Non-linear Independent Components Estimation". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: http://arxiv.org/abs/1410.8516.

[11] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using Real NVP". In: (2016). DOI: 10.48550/ARXIV.1605.08803. URL: https://arxiv.org/abs/1605.08803.

[12] Conor Durkan et al. "Neural Spline Flows". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/7ac71d433f282034e088473244df8c02-Paper.pdf.

[13] R. P. Feynman. "Space-Time Approach to Quantum Electrodynamics". In: *Phys. Rev.* 76 (6 Sept. 1949), pp. 769–789. DOI: 10.1103/PhysRev.76.769. URL: https://link.aps.org/doi/10.1103/PhysRev.76.769.

[14] Murray Gell-Mann and Francis Low. "Bound States in Quantum Field Theory". In: *Phys. Rev.* 84 (2 Oct. 1951), pp. 350–354. DOI: 10.1103/PhysRev.84.350. URL: https://link.aps.org/doi/10.1103/PhysRev.84.350.

[15] Mathieu Germain et al. "MADE: Masked Autoencoder for Distribution Estimation". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, 2015, pp. 881–889. URL: http://proceedings.mlr.press/v37/germain15.pdf.

[16] Steven M. Girvin and Kun Yang. *Modern Condensed Matter Physics*. Cambridge University Press, 2019. DOI: 10.1017/9781316480649.

[17] Feliciano Giustino. "Electron-phonon interactions from first principles". In: *Rev. Mod. Phys.* 89 (1 Feb. 2017), p. 015003. DOI: 10.1103/RevModPhys.89.015003. URL: https://link.aps.org/doi/10.1103/RevModPhys.89.015003.

[18] James Gubernatis, Naoki Kawashima, and Philipp Werner. "Monte Carlo basics". In: *Quantum Monte Carlo Methods: Algorithms for Lattice Models*. Cambridge University Press, 2016, pp. 11–42. DOI: 10.1017/CBO9780511902581.003.

[19] W. K. Hastings. "Monte Carlo sampling methods using Markov chains and their applications". In: *Biometrika* 57.1 (Apr. 1970), pp. 97–109. DOI: 10.1093/biomet/57.1.97. URL: https://doi.org/10.1093/biomet/57.1.97.

[20] P. Hohenberg and W. Kohn. "Inhomogeneous Electron Gas". In: *Phys. Rev.* 136 (3B Nov. 1964), B864–B871. DOI: 10.1103/PhysRev.136.B864. URL: https://link.aps.org/doi/10.1103/PhysRev.136.B864.

[21] T Holstein. "Studies of polaron motion: Part I. The molecular-crystal model". In: *Annals of Physics* 8.3 (1959), pp. 325–342. ISSN: 0003-4916. DOI: https://doi.org/10.1016/0003-4916(59)90002-8. URL: https://www.sciencedirect.com/science/article/pii/0003491659900028.

[22] T. Holstein. "Studies of polaron motion: Part II. The "small" polaron". In: *Annals of Physics* 8.3 (1959), pp. 343–389. ISSN: 0003-4916. DOI: https://doi.org/10.1016/0003-4916(59)90003-X. URL: https://www.sciencedirect.com/science/article/pii/000349165990003X.

[23] Emiel Hoogeboom, Taco S. Cohen, and Jakub M. Tomczak. *Learning Discrete Distributions by Dequantization*. 2020. arXiv: 2001.11235 [cs.LG].

[24] Chin-Wei Huang et al. "Neural Autoregressive Flows". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 2078–2087. URL: https://proceedings.mlr.press/v80/huang18d.html.

[25] Priyank Jaini, Kira A. Selby, and Yaoliang Yu. "Sum-of-Squares Polynomial Flow". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 3009–3018. URL: https://proceedings.mlr.press/v97/jaini19a.html.

[26] Durk P Kingma and Prafulla Dhariwal. "Glow: Generative Flow with Invertible 1x1 Convolutions". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/d139db6a236200b21cc7f752979132d0Paper.pdf.

[27] W. Kohn and L. J. Sham. "Self-Consistent Equations Including Exchange and Correlation Effects". In: *Phys. Rev.* 140 (4A Nov. 1965), A1133–A1138. DOI: 10.1103/PhysRev.140.A1133. URL: https://link.aps.org/doi/10.1103/PhysRev.140.A1133.

[28] S. Kullback and R. A. Leibler. "On Information and Sufficiency". In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86. DOI: 10.1214/aoms/1177729694. URL: https://doi.org/10.1214/aoms/1177729694.

[29] HERMANN G. KÜMMEL. "A BIOGRAPHY OF THE COUPLED CLUSTER METHOD". In: *Recent Progress in Many-Body Theories*, pp. 334–348. DOI: 10.1142/9789812777843_0040. eprint: https://www.worldscientific.com/doi/pdf/10.1142/9789812777843_0040. URL: https://www.worldscientific.com/doi/abs/10.1142/9789812777843_0040.

[30] IG Lang and Yu A Firsov. "Kinetic theory of semiconductors with low mobility". In: *Sov. Phys. JETP* 16.5 (1963), p. 1301. URL: http://jetp.ras.ru/cgi-bin/dn/e_016_05_1301.pdf.

[31] A. J. Leggett et al. "Dynamics of the dissipative two-state system". In: *Rev. Mod. Phys.* 59 (1 Jan. 1987), pp. 1–85. DOI: 10.1103/RevModPhys.59.1. URL: https://link.aps.org/doi/10.1103/RevModPhys.59.1.

[32] Luca Leoni. *LLDMC: a package for simple Diagrammatic Monte Carlo simulations*. URL: https://github.com/Luca-Leoni/LLDMC.

[33] E. Y. Loh et al. "Sign problem in the numerical simulation of many-electron systems". In: *Phys. Rev. B* 41 (13 May 1990), pp. 9301–9307. DOI: 10.1103/PhysRevB.41.9301. URL: https://link.aps.org/doi/10.1103/PhysRevB.41.9301.

[34] G. M. Luke et al. “Muon diffusion and spin dynamics in copper”. In: *Phys. Rev. B* 43 (4 Feb. 1991), pp. 3284–3297. DOI: 10.1103/PhysRevB.43.3284. URL: https://link.aps.org/doi/10.1103/PhysRevB.43.3284.

[35] Gerald D. Mahan. “Exactly Solvable Models”. In: *Many-Particle Physics*. Boston, MA: Springer US, 2000, pp. 187–294. ISBN: 978-1-4757-5714-9. DOI: 10.1007/978-1-4757-5714-9_4. URL: https://doi.org/10.1007/978-1-4757-5714-9_4.

[36] Nicholas Metropolis et al. “Equation of State Calculations by Fast Computing Machines”. In: *The Journal of Chemical Physics* 21.6 (Dec. 2004), pp. 1087–1092. DOI: 10.1063/1.1699114. URL: https://doi.org/10.1063/1.1699114.

[37] Nicholas Metropolis et al. “Equation of State Calculations by Fast Computing Machines”. In: *The Journal of Chemical Physics* 21.6 (Dec. 2004), pp. 1087–1092. DOI: 10.1063/1.1699114. URL: https://doi.org/10.1063/1.1699114.

[38] A. S. Mishchenko et al. “Diagrammatic quantum Monte Carlo study of the Fröhlich polaron”. In: *Phys. Rev. B* 62 (10 Sept. 2000), pp. 6317–6336. DOI: 10.1103/PhysRevB.62.6317. URL: https://link.aps.org/doi/10.1103/PhysRevB.62.6317.

[39] Frank Noé et al. “Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning”. In: *Science* 365.6457 (2019), eaaw1147. DOI: 10.1126/science.aaw1147. URL: https://www.science.org/doi/abs/10.1126/science.aaw1147.

[40] George Papamakarios et al. “Normalizing Flows for Probabilistic Modeling and Inference”. In: *J. Mach. Learn. Res.* 22.1 (Jan. 2021). ISSN: 1532-4435. URL: https://jmlr.org/papers/volume22/19-1028/19-1028.pdf.

[41] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[42] John P. Perdew, Kieron Burke, and Matthias Ernzerhof. “Generalized Gradient Approximation Made Simple”. In: *Phys. Rev. Lett.* 77 (18 Oct. 1996), pp. 3865–3868. DOI: 10.1103/PhysRevLett.77.3865. URL: https://link.aps.org/doi/10.1103/PhysRevLett.77.3865.

[43] John P. Perdew et al. “Atoms, molecules, solids, and surfaces: Applications of the generalized gradient approximation for exchange and correlation”. In: *Phys. Rev. B* 46 (11 Sept. 1992), pp. 6671–6687. DOI: 10.1103/PhysRevB.46.6671. URL: https://link.aps.org/doi/10.1103/PhysRevB.46.6671.

[44] Nikolai V. Prokof’ev and Boris V. Svistunov. “Polaron Problem by Diagrammatic Quantum Monte Carlo”. In: *Phys. Rev. Lett.* 81 (12 Sept. 1998), pp. 2514–2517. DOI: 10.1103/PhysRevLett.81.2514. URL: https://link.aps.org/doi/10.1103/PhysRevLett.81.2514.

[45] R.Y. Rubinstein and D.P. Kroese. *Simulation and the Monte Carlo Method*. Wiley Series in Probability and Statistics. Wiley, 2016. ISBN: 9781118632161. URL: https://books.google.at/books?id=dCksDQAAQBAJ.

[46] Ankur Singha, Dipankar Chakrabarti, and Vipul Arora. "Conditional normalizing flow for Markov chain Monte Carlo sampling in the critical region of lattice field theory". In: *Phys. Rev. D* 107 (1 Jan. 2023), p. 014512. DOI: 10.1103/PhysRevD.107.014512. URL: https://link.aps.org/doi/10.1103/PhysRevD.107.014512.

[47] J. C. Slater. "A Simplification of the Hartree-Fock Method". In: *Phys. Rev.* 81 (3 Feb. 1951), pp. 385–390. DOI: 10.1103/PhysRev.81.385. URL: https://link.aps.org/doi/10.1103/PhysRev.81.385.

[48] Masuo Suzuki. "Quantum Crossover Effect in the Gas-Liquid Phase Transition". In: *Progress of Theoretical Physics* 56.3 (Sept. 1976), pp. 1007–1008. ISSN: 0033-068X. DOI: 10.1143/PTP.56.1007. URL: https://doi.org/10.1143/PTP.56.1007.

[49] L. Theis, A. van den Oord, and M. Bethge. "A note on the evaluation of generative models". In: *International Conference on Learning Representations*. Apr. 2016. URL: http://arxiv.org/abs/1511.01844.

[50] Matthias Troyer and Uwe-Jens Wiese. "Computational Complexity and Fundamental Limitations to Fermionic Quantum Monte Carlo Simulations". In: *Phys. Rev. Lett.* 94 (17 Apr. 2005), p. 170201. DOI: 10.1103/PhysRevLett.94.170201. URL: https://link.aps.org/doi/10.1103/PhysRevLett.94.170201.

[51] Yufei Wang et al. "Low-Light Image Enhancement with Normalizing Flow". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 36.3 (Jan. 2022), pp. 2604–2612. DOI: 10.1609/aaai.v36i3.20162. URL: https://ojs.aaai.org/index.php/AAAI/article/view/20162.

[52] Ulrich Weiss. *Quantum Dissipative Systems*. 4th. WORLD SCIENTIFIC, 2012. DOI: 10.1142/8334. eprint: https://www.worldscientific.com/doi/pdf/10.1142/8334. URL: https://www.worldscientific.com/doi/abs/10.1142/8334.

[53] G. C. Wick. "The Evaluation of the Collision Matrix". In: *Phys. Rev.* 80 (2 Oct. 1950), pp. 268–272. DOI: 10.1103/PhysRev.80.268. URL: https://link.aps.org/doi/10.1103/PhysRev.80.268.

[54] Wikipedia. *Metropolis-Hastings algorithm — Wikipedia, The Free Encyclopedia*. http://en.wikipedia.org/w/index.php?title=Metropolis%E2%80%93Hastings%20algorithm&oldid=1172902257. [Online; accessed 02-October-2023]. 2023.

[55] Ulli Wolff. "Critical slowing down". In: *Nuclear Physics B - Proceedings Supplements* 17 (1990), pp. 93–102. ISSN: 0920-5632. DOI: https://doi.org/10.1016/0920-5632(90)90224-I. URL: https://www.sciencedirect.com/science/article/pii/092056329090224I.

[56] Ulli Wolff. "Monte Carlo errors with less errors". In: *Computer Physics Communications* 156.2 (2004), pp. 143–153. ISSN: 0010-4655. DOI: https://doi.org/10.1016/S0010-4655(03)00467-3. URL: https://www.sciencedirect.com/science/article/pii/S0010465503004673.