# Hashing algorithms, optimized mappings and massive parallelization of multi-configuration methods for bosons

Alex V. Andriati[a,*], Arnaldo Gammal[a]

[a] *Instituto de Física, University of São Paulo, CEP 05508-090, São Paulo-SP, Brazil*

**Abstract**

We report memory management routines to aid the manipulation of Fock states in multi-configuration methods and to use bosonic creation and annihilation operators in the spanned Hilbert space, defined by the total number combinations of a finite number of particles over few individual particle states. From a basic combinatorial problem to map integer numbers to each Fock state, we build up routines to handle creation and annihilation operators, step by step, showing the performance gain and memory consumption for different implementations, highlighting the limits of applicability and time demanded for each one. We also exploit massive parallel processors from graphics processor units with CUDA to improve a routine to act with the many-body Hamiltonian on the spanned configurational space, which expressively reduce the time demanded.

*Keywords:* many-particle physics, multi-configuration, hashing, bosons, CUDA

## 1. Introduction

Quantum and statistical mechanics often resort to second quantized formalism, specially when dealing with a system of identical particles, which automatically takes into account the symmetry of the many-particle wave function. In this formalism, all observables can be expressed in terms of creation and annihilation operators, defined by two possible algebras

$$[\hat{a}_k, \hat{a}_l^\dagger] = \delta_{kl}, \tag{1}$$

for bosons and

$$\{\hat{c}_k, \hat{c}_l^\dagger\} = \delta_{kl}, \tag{2}$$

for fermions, where $[A, B] = AB - BC$ and $\{A, B\} = AB + BA$.

It is required in the formalism a complete set of individual particle states (IPS), whose the creation/annihilation index refers to. A generic set of single particle states $\{\phi_k\}_{k\in\mathbb{N}}$, allows us to write the expression for the non-interacting many-particle operators as

$$\mathcal{T} = \sum_{k,l} \hat{a}_k^\dagger a_l \langle \phi_k | \hat{T} | \phi_l \rangle, \tag{3}$$

[*] corresponding author

*Email addresses:* andriati@if.usp.br (Alex V. Andriati), gammal@if.usp.br (Arnaldo Gammal) and the interacting many-particle operators as

$$\mathcal{V} = \frac{1}{2} \sum_{k,l,q,s} \hat{a}_k^\dagger \hat{a}_s^\dagger a_l a_q \langle \phi_k, \phi_s | \hat{V} | \phi_q, \phi_l \rangle, \tag{4}$$

or even more general operators that are a combination of interacting and non-interacting many-particle operators, as is the case of the Hamiltonian, which can be the sum of both $\mathcal{H} = \mathcal{T} + \mathcal{V}$.

Formally, the sums presented in Eq. (3) and (4) run over an infinite set of indexes that enumerate the IPS, and it is where most physical approximations starts. As the main example, we have the mean field theory for bosons, which yields the Gross-Pitaevskii equation that is widely used to describe cold atomic clouds and Bose-Einstein condensates [1, 2, 3, 4]. In this mean field approximation, it is considered a single relevant mode of the creation/annihilation operators, that holds the majority of occupation numbers, whose retain almost every particle in the system, that is $\langle \hat{a}_0^\dagger \hat{a}_0 \rangle / N \approx 1$.

Other relevant approaches admit more than a single mode as used in mean-field theory, like the Bose-Hubbard model [5], that has described cold atomic clouds loaded in optical lattices [6, 7], and explained rather well the insulator and superfluid phases [8, 9, 10] in these systems. Another important example is a two mode mean field approach, which is used for two interacting Bose-Einstein condensates, for instance to explain Josephson junction effects [11], and to add spin internal degree of freedom in multi-component Gross-Pitaevskii equations [12, 13].

The approximation techniques cited above, however, are exact in some specific limits [14, 15], or are reasonable for weak interactions where depletion from the modes considered are low [16], since terms in Eq. (4) couple all occupations in different IPS. However, in the past decade, the advances in probing correlation functions in cold atomic clouds [17, 18, 19, 20] and the lack of methods that systematically includes depletion from the condensate due to strong interactions, boosted the development of a numerical method that could include a variable number of IPS to describe the physical systems. In this scenario, the many-particle state can in principle be written as a combination of occupation vectors, also named configurations, given by

$$|\vec{n}^{(\beta)}\rangle \doteq |n_1^{(\beta)} \; ... \; n_M^{(\beta)}\rangle \;, \text{ with } \sum_{j=1}^{M} n_j^{(\beta)} = N, \; \forall \beta, \tag{5}$$

where $N$ is the total number of particles, and $M$ the number of IPS. The total number of configurations indexed by $\beta$ is obtained from the combinatorial problem of how to fit $N$ identical balls in $M$ boxes, which yields for bosons

$$N_c(N, M) = \binom{N + M - 1}{M - 1} = \frac{(N + M - 1)!}{N!(M - 1)!}. \tag{6}$$

The most important numerical methods that employed Multi-configurations described above are the Exact Diagonalization(ED) [21, 22, 23, 24], which most part of the time is used to find the low energy spectrum with Lanczos algorithm [25, 26], which consider the IPS fixed, and the Multi-configuration Time-Dependent Hartree method for Bosons (MCTDHB) [27], which determine the IPS variationally by action minimization. Both were used in many physical systems to study the ground state and dynamics as well [28, 29, 30, 31, 32, 22, 24, 33].

In this configurational basis, we can express the many-particle state of the system by a linear combination as

$$|\Psi(t)\rangle = \sum_{\beta=1}^{N_c(N,M)} C_\beta(t)|\vec{n}^{(\beta)}\rangle, \tag{7}$$

where $\vec{C}(t)$ is a complex vector of dimension $N_c(N, M)$.

Independently of how the IPS are picked up, based on ED or MCTDHB, the operators in Eq. (3) and (4) need to be represented in the configuration basis, which requires some way to act with creation/annihilation operators on configurations. This problem is studied in this article, starting from the fundamental question on how to establish the relation between $\beta$ and its configuration $|\vec{n}^{(\beta)}\rangle$. A function that does this job is called a hashing function and we have a perfect hashing if we get a one-to-one correspondence.

This problem has been studied since the last three decades at least, where some effort was done on developing hashing functions to search for configuration states [23, 34], although most part of more recent works have been directed to particles confined in sites of optical lattices [24], restricting to the Bose-Hubbard model [22], or for spin systems [23, 33]. A more recent proposal for hashing function was showed in [35], though again most part of the results are for fermionic spin systems.

The operators in the second quantized formalism also demand a conversion between configurations due to the action of creation/annihilation operators. We can define mappings to track every configuration to others, that are related by the rearrangement of 1 or 2 particles among the IPS. The number of possibilities can be very large depending on the size of the configurational space and to implement such structures for arbitrary number of IPS and particles can be challenging.

In the following, we develop perfect hashing functions to establish the connection of configurations and integer numbers for bosons, that was done already in [34], without any restriction on the Hamiltonian, like symmetries or only nearest neighbors interaction, for arbitrary number of IPS. As a novelty, we implement direct mappings from configuration to configuration, which relate all possible jumps by the action of creation/annihilation operators and show a detailed discussion on the performance gain and memory requirement by this refinement. Finally, we study the impact of using parallel processors comparing with a multi-threaded CPU with Graphics Processing Units (GPUs) with CUDA interface.

The physical operators chosen to illustrate the performance throughout this article are the Hamiltonian and density matrices of first and second order, given by the expectation values of a combination of two or four creation and annihilation operators respectively. These quantities are essential for the MCTDHB [27], which usually have to be evaluated several times for the same configurational space, thus emphasizing the relevance of our study on the problem.

The one and two-body density matrices are defined here by

$$\rho_{kl}^{(1)} = \langle \Psi | \hat{a}_k^\dagger \hat{a}_l | \Psi \rangle = \sum_{\gamma=1}^{N_c(N,M)} \sum_{\beta=1}^{N_c(N,M)} C_\gamma^* C_\beta \langle \vec{n}^{(\gamma)} | \hat{a}_k^\dagger \hat{a}_l | \vec{n}^{(\beta)} \rangle, \tag{8}$$

and

$$\rho_{klqs}^{(2)} = \langle \Psi | \hat{a}_k^\dagger \hat{a}_l^\dagger \hat{a}_q \hat{a}_s | \Psi \rangle = \sum_{\gamma=1}^{N_c(N,M)} \sum_{\beta=1}^{N_c(N,M)} C_\gamma^* C_\beta \langle \vec{n}^{(\gamma)} | \hat{a}_k^\dagger \hat{a}_l^\dagger \hat{a}_q \hat{a}_s | \vec{n}^{(\beta)} \rangle, \tag{9}$$

respectively.

For the matrix elements of the Hamiltonian, that in general is a sum of parts given in Eqs. (3) and (4), we choose not to store the matrix, which is very sparse. In this way our routines only require the matrices elements $\langle \phi_k | \hat{T} | \phi_l \rangle$ and $\langle \phi_k, \phi_s | \hat{V} | \phi_q, \phi_l \rangle$ to evaluate the sum

$$\sum_{\beta=1}^{N_c(N,M)} \langle \vec{n}^{(\gamma)} | \mathcal{H} | \vec{n}^{(\beta)} \rangle C_\beta. \tag{10}$$

This could be further used to address the time evolution in this basis with the equation for the coefficients given by

$$i\hbar \frac{\mathrm{d}C_\alpha}{\mathrm{d}t} = \sum_{\beta=1}^{N_c(N,M)} \langle \vec{n}^{(\alpha)} | \mathcal{H} | \vec{n}^{(\beta)} \rangle C_\beta(t), \tag{11}$$

where any differential integrator method needs to evaluate the right hand side of the equation.

The clear advantage appears when dealing with a method that needs to update the IPS $\{\phi_k\}$, like in the MCTDHB, to study dynamics and thus, even if the matrix in configurational space is stored, which is not trivial because its sparsity depends on number of particle and IPS, it would need to be reassembled during time evolution. There is also interest when the goal is to diagonalize using the Lanczos algorithm, where one needs to successively apply the Hamiltonian operator.

## 2. Mapping Fock states to integers

Following the theoretical framework developed in the previous section, as a first step, it is necessary to address an integer number for each configuration. The routine to perform this task assigns a cost for every IPS to be occupied, starting with all occupations zero. The problem may be depicted by a basket of balls (particles), that starts with $N$, the total number of particles, and is emptied particle by particle to fill the IPS.

Given an enumeration for the IPS from 0 to $M-1$, if we take the one with number $k$ with $0 \leq k \leq M-1$, there are others $k$ IPS before it since we are counting the number zero. In this way, the cost to put one particle in the state $k$ is defined by the total number of configurations of remaining particles in the stack over all previous IPS following the

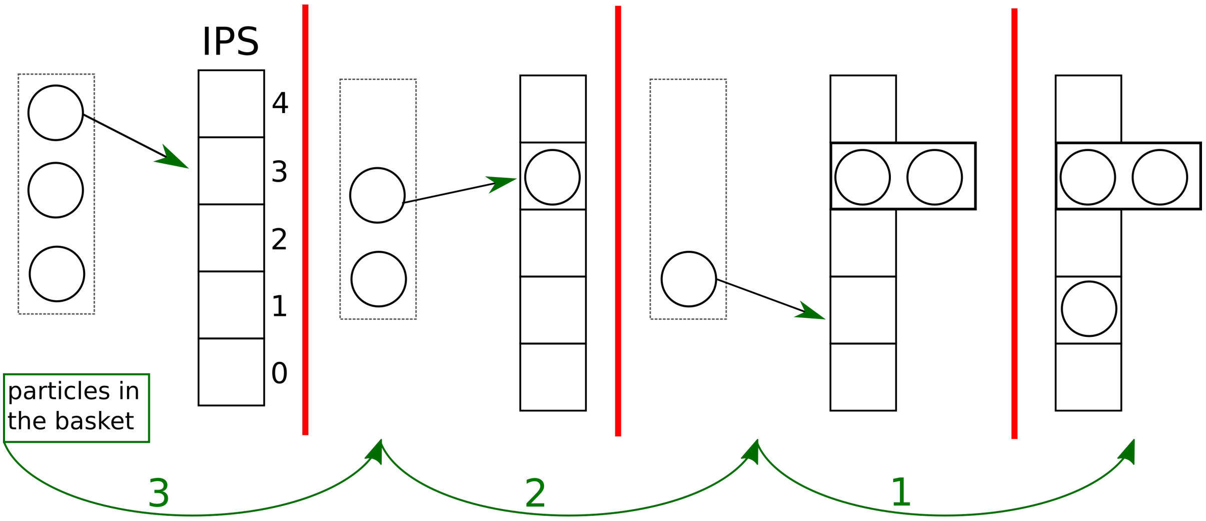

Figure 1: Example to illustrate the process of IPS occupation for $N = 3$ and $M = 5$ for a the specific configuration $|0, 1, 0, 2, 0\rangle$. The total cost of this configuration is $N_c(3, 3) + N_c(2, 3) + N_c(1, 1) = 17$ where the terms are presented following the order of arrows.

enumeration. In other words, the cost is all the combinations we could do with the lower number IPS. When the stack has none particle left, the process is finished and the total cost will be the index of the configuration.

In Fig. 1 is depicted a practical example of the description above. Moreover, the combination function defined in Eq. (6) plays a crucial role being used to compute the costs and is straightforward to check that if $p_n$ is the IPS the $n$-th particle occupies, then by construction we have $0 \leq p_1 \leq ... \leq p_N < M$ and the integer resulting from the total cost is

$$I(\vec{p}) = \sum_{n=1}^{N} N_c(n, p_n) = \sum_{n=1}^{N} \binom{n + p_n - 1}{p_n - 1}, \ \forall p_n > 0; \quad N_c(n, 0) = 0, \forall n. \tag{12}$$

This relation is identical to the results in [34], though some conventions are changed and there the derivation follows an alternative way, using a correspondence to fermions. Moreover from [34], it is already known that this relation maps uniquely integers to configurations without any left number and therefore indicates a perfect hashing function.

The prescription of the hashing can now be implemented. Given a configuration we sum up the costs, removing particle by particle using Eq. (12), procedure detailed in algorithm 1. The reverse process is quite straightforward, where given an index between 0 and $N_c(N, M) - 1$ put all the particles in a stack and check, starting from the highest cost IPS, if the index is big enough to put the particle. In positive case, it transfers a particle from the stack to the IPS, otherwise move to lower cost IPS and try again. The routine to convert index to configuration is given in algorithm 2.

The algorithm 1 makes explicit a routine to get index from a configuration, and together with the algorithm 2, set up the basis to operate with creation and annihilation given a many-body state in the configuration basis.

**Algorithm 1** Get configuration Index from a configuration

```
Require: $N > 0,\ M > 0,$ occupation vector $\vec{n}$
  $k \leftarrow 0$
  $s \leftarrow N$
  for $m = M - 1..1$ do
    $j \leftarrow n[m]$
    while $j > 0$ do
      $k \leftarrow k + N_c(s, m)$
      $s \leftarrow s - 1$
      $j \leftarrow j - 1$
    end while
  end for
  return $k$
```

**Algorithm 2** Build configuration $\vec{n}$ from index $\beta$

```
Require: $N > 0,\ M > 0,\ N_c(N, M) > \beta \geq 0$
  for $i = 0..M - 1$ do
    $n[i] \leftarrow 0$
  end for
  $k \leftarrow \beta$
  $m \leftarrow M - 1$
  $s \leftarrow N$
  while $k > 0$ do
    while $k - N_c(s, m) < 0$ do
      $m \leftarrow m - 1$
    end while
    $k \leftarrow k - N_c(s, m)$
    $n[m] \leftarrow n[m] + 1$
    $s \leftarrow s - 1$
  end while
  if $s > 0$ then
    $n[0] \leftarrow n[0] + s$
  end if
```

In the computation of the density matrices, we need to perform just the sum in $\beta$, whereas for each $\beta$ there is a unique value for $\gamma$, the one corresponding to the configuration after replacing the particles due to the action of the creation and annihilation operators. Therefore, for $\beta$ running from 0 to $N_c(N, M) - 1$, we need three steps to perform the operation required. First, obtain the configuration using algorithm 2. Second, reconfigure the occupation according to the action of creation/annihilation operators. Third, use this new occupation vector to compute the corresponding index $\gamma$ using algorithm 1 and do the multiplication of coefficients.

Nevertheless, we must use the algorithms 2 and 1 $N_c(N, M)$ times for every element of the density matrices, that results in a total of $M^2 N_c(N, M)$ calls of both functions to setup all the elements of $\rho^{(1)}$ and $M^4 N_c(N, M)$ for $\rho^{(2)}$[1]. However, we may spend more memory creating some structures to avoid the number of calls of these functions, whose will improve performance as will be shown later. Surely, the setup of any new structure would demand some equivalent time but, we again emphasize that our goal are problems that need to compute these quantities several times for the same configurational space [27].

A first improvement is to build once all occupation vectors and maintain them stored during all operations, defining a hashing table. For instance, they can be stored along rows of a matrix of integers, with the row number being the index of the respective configuration. This hashing table would require to store $M N_c(N, M)$ integers in exchange of avoiding calls of algorithm 2 when computing $\rho^{(1)}$ and $\rho^{(2)}$.

In Fig. 2 we compare a raw implementation using both algorithms 2 and 1 with an improved, that uses hashing table to setup and sort occupation vectors previously and require only algorithm 1. The basic difference between the two implementations is the call of algorithm 2 face to a memory access of the hashing table. As can be noted the performance gain is critical, highlighted by the logarithm scale, for both cases, when varying the number of particles or the number of IPS. Moreover for the left panel, the time required with respect to the number of particles has a clear linear relation in logarithmic scales, which indicates a power law.

Other routines, to build the two-body density matrices and to apply the Hamiltonian using the configuration basis shall benefit even more by a previous setup of all occupation vectors, because they require much more operations. We thus focus on these two quantities for the next improvements.

## 3. Mapping routines

The next step is to set the routines of physical quantities completely free from calls of the algorithm 1 as well. For this aim, it is necessary to define a structure where given an index it has stored all possible jumps[2] of one and two particles among the IPS. In a single particle jump, it has $M$ different states to be placed, which implies that for every configuration there

[1] Actually, this number can be halved if one use hermiticity, and reduced even more using the commutation relations in the case of $\rho^{(2)}$.

[2] Jump here means the simultaneous destruction and creation of particle in different states

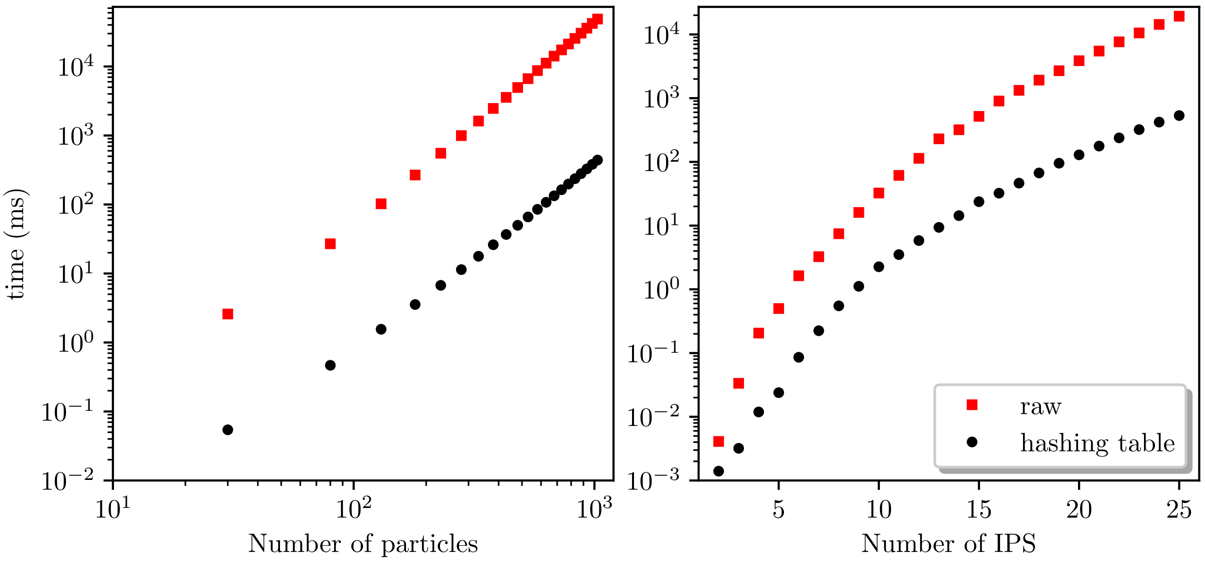

Figure 2: Time comparison between two routines to compute all elements of $\rho^{(1)}$. The red squares correspond to an implementation that uses just the conversion algorithms 2 and 1(raw) and the black circles make use of a hashing table to store and sort the configurations, which restrict to use only algorithm 1. In the left panel was fixed 3 IPS while varying the number of particles and in the right panel was varied the number of IPS with 5 particles.

are at most $M^2$ possible transitions. Thus a straightforward way to map all these transitions is to define a triple indexed structure, which stores integers, where the first index is from the configuration number, and the other two are IPS numbers, one from where the particle is being destroyed and other where it is being created. This 1-particle jump mapping would require $N_c(N, M)M^2$ new integers to store.

It is worth pointing out that the memory cost for this 1-particle jump mapping is greater than the first improvement of the hashing table, where in that case was stored all the occupation numbers and therefore had a cost of $N_c(N, M)M$ integers. This justify why we did not mind about memory cost at that stage, though it will matter in the following. Moreover, the $N_c(N, M)M^2$ integers wastes some memory because there are configurations with some empty IPS, which actually do not have $M^2$ possible transitions. Nevertheless, this wasted memory here will not matter as well, since the two-particle jump mappings will require more elements than $N_c(N, M)M^2$.

We now analyze how to implement a structure that maps double jumps, that is, two particles move to different single particle states. If we follow the same idea presented for the 1-particle jump, for each configuration there would be at most $M^2$ possibilities to take 2 particles from the occupation numbers and for each one of these possibilities there are again $M^2$ ways to replace them. Following this naive way, we would end up with the additional memory requirement of $N_c(N, M)M^4$ integers. Nevertheless, it is possible to reduce this number.

A thoroughly inspection over configurations show us that $M^2$ possibilities to take 2

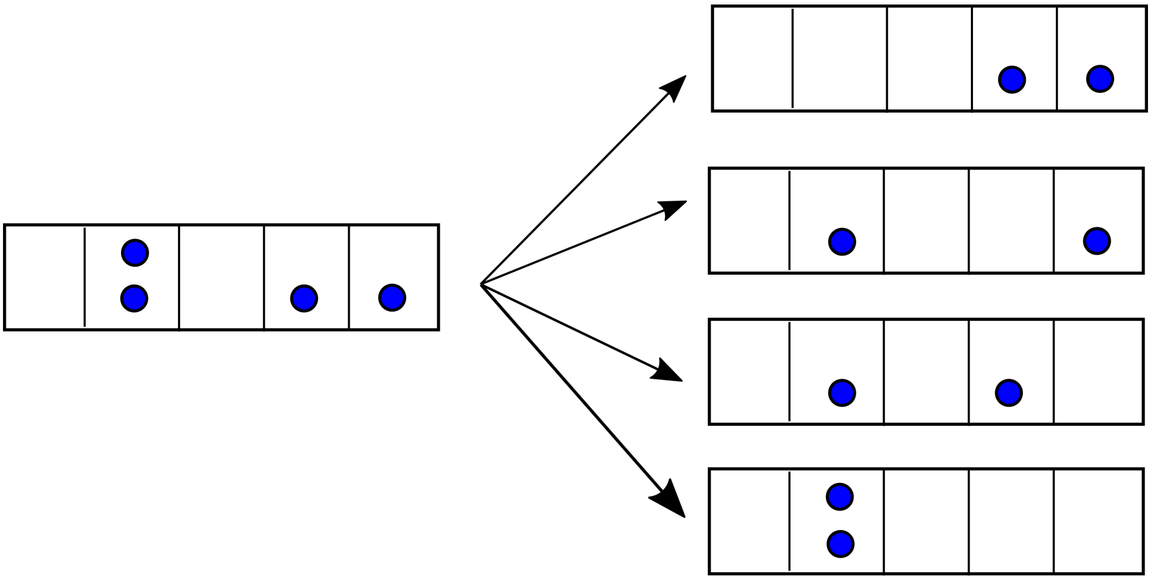

Figure 3: In the left a possible configuration for $M = 5$ and $N = 4$. In the right the arrows indicate all possible ways to remove 2 particles from the single particle states, that is 4.

particles from the IPS(equivalent to the action of 2 annihilation operators) is not true for most part of the configurations, because there are many configurations with empty IPS. The real number of possibilities can be obtained as follows: for every non-empty IPS $k$ we search for $s \geq k$ non-empty as well and, whenever we find such numbers, we will have $M^2$ another possible configurations to put these particle took from $k$ and $s$.

In Fig. 3, it is illustrated for a simple case, given a specific configuration, the possible ways to remove simultaneously two particles by the action of two annihilation operators. The arrows conducts to the possible outcomes, where for each one, we have $M^2$ possibilities to replace the particles. Originally, the naive way would store a lot of garbage since it considers a bunch of forbidden transitions, that is, removal from empty states. For instance, in the case represented in Fig. 3, it would require $5^4 = 625$ possibilities, while there are only $4 \times 5^2 = 100$ real possibilities.

In summary, to save memory for this structure that we are going to define, we cannot allocate so much garbage from those forbidden transitions. In order to overcome the problem, we define a structure like a hashing table, though each line of the table has a variable number of elements, where the line number correspond to an index of a configuration. These elements are integers, indexes of other configurations that are outcomes of all possible jumps of 2 particles.

A possible way to sort the elements for each line in the table is starting with $k = 0$ up to $k = M - 1$, we take $k \leq s < M$ and for each possible simultaneous removing of particles in $k$ and $s$, we have a stride of $M^2$ elements of new configurations obtained for every possible way to replace the particles removed. Therefore, if one wants to know, the configuration index $\gamma$, that is a result of remove particles from $i$ and $j > i$ and replace them in $q$ and $l$ from a configuration $\beta$, it is required to check out how many strides we need to ignore.

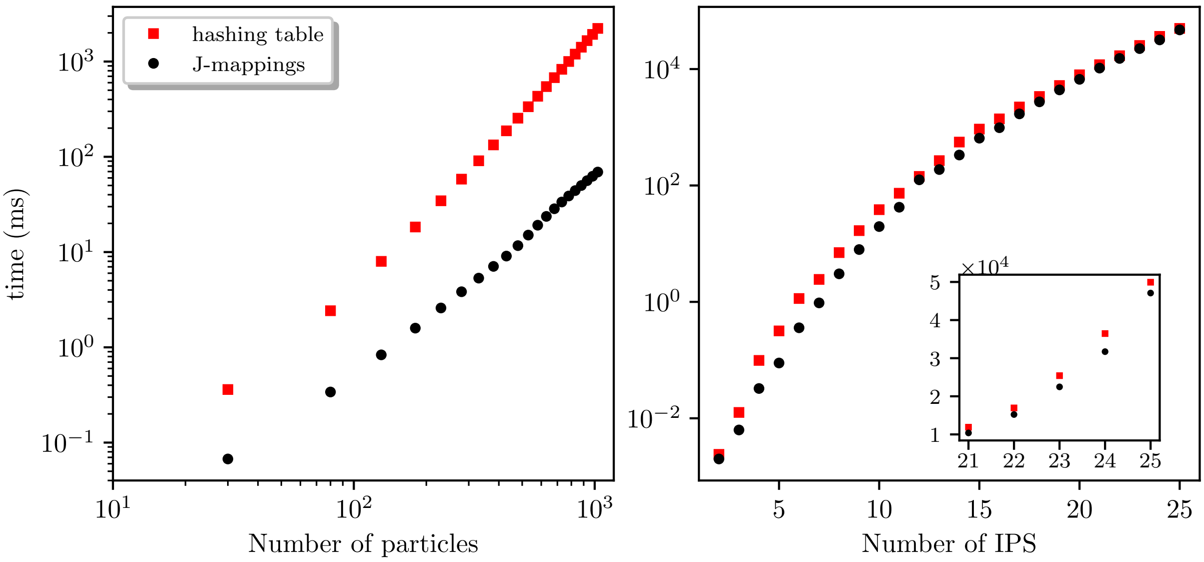

Figure 4: Time to compute all elements of $\rho^{(2)}$. The red squares correspond to an implementation that uses hashing table of configurations and calls of algorithm 1 while the black circles refers to one that uses direct jump mappings between configurations related by the action of the creation/annihilation operators and dismiss completely the use of algorithms 2 and 1. In the left while varying the number of particles it was taken 3 IPS fixed and in the right 5 particles was used throughout the curve with respect to IPS.

For example, suppose in Fig. 3 we are interested in the transitions that come from removing the last two particles, thus we need to skip 3 strides. In other words, in our table, in the line corresponding to the configuration in the figure, we need to skip the $3 \times 5^2$ elements to get the indexes of configurations we are interested.

In Fig. 4 the performance is compared between two routines to construct $\rho^{(2)}$. The first, uses only the hashing table of configurations, but still make use of algorithm 1. The second, does not use any of the algorithms of conversion between indexes and configurations, instead, use the mappings(both of 1 and 2-particle jump) explained above. For high number of particles, we see a good performance gain, while for high number of IPS, the gain is slight, indicating that the use of algorithm 1 is not the bottleneck in this case.

A careful inspection in algorithm 1 show us that it need to remove all particles from the configuration and thus demands the total number of particles as operations. Therefore, it is expected that the gain over high number of particles is greater than for high number of IPS. In other words, it is harder to empty many particles from few IPS than a few particles from many IPS.

Another very important routine to check the performance gain is the time to act with the Hamiltonian over a state expressed in the configuration basis, that is to evaluate the sum in Eq. (10). In the same way that was done for $\rho^{(1)}$ and $\rho^{(2)}$, in Fig. 5 we compare the time demanded for evaluating the Hamiltonian sum using the two routines, again one using the hashing table and algorithm 1 and other using jump mappings. Similarly there is a clear improvement varying the number of particle(left panel), but this time there is a substantial

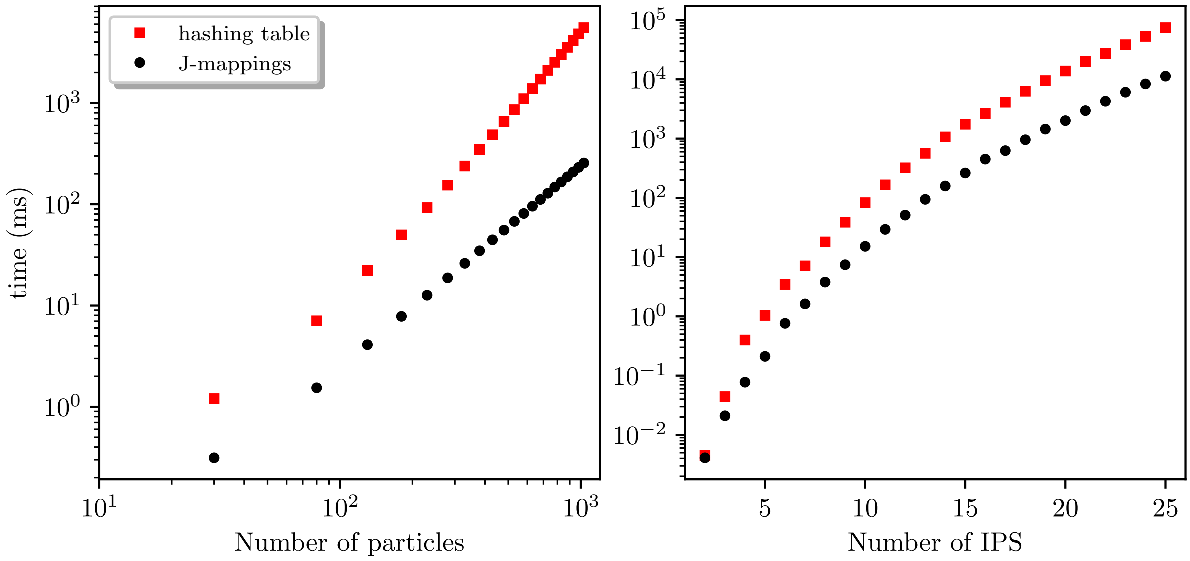

Figure 5: Time required to act with Hamiltonian operator in the configuration basis. As done for $\rho^{(2)}$ in Fig. 4 the red squares correspond to a routine that uses hashing table and algorithm 1 and the black circles to one that uses mappings instead of algorithm 1. In the left panel we vary the number of particles for 3 IPS and in the right panel the number of IPS for 5 particles.

| | $\rho^{(2)}$ | | $\mathcal{H}$ | |
|---|---|---|---|---|
| | $a$ | $b$ | $a$ | $b$ |
| hashing table | 2.835(5) | $6.33(9) \times 10^{-6}$ | 2.791(5) | $2.13(3) \times 10^{-5}$ |
| jump mappings | 2.34(3) | $6.4(6) \times 10^{-6}$ | 2.000(2) | $2.40(2) \times 10^{-4}$ |

Table 1: Fitted parameters, for implementations using hashing table and jump mappings. As the only value for the number of IPS was 3, we dropped the subscript $M$ from the parameters.

gain also varying the number of IPS.

In all comparisons between the routines that used the algorithm 1 with the hashing table and those that use mappings, when varying the number of particles, there is evident constant slope behaviour in the curve, at least, for high number of particles. This reveals that the time demanded respect to the number of particles can be written as a power law in the form

$$\tau_M(N) = b_M N^{a_M}, \tag{13}$$

with $M$ the number of IPS fixed. The parameters can be extracted from curve fitting.

For the more time demanding routines, that are for $\rho^{(2)}$ and $\mathcal{H}$, it was evaluated a linear curve fitting in the log scaled plots, which result in a power law of the form (13) for time as function of the number of particles. The values are showed in Tab. 1. The most important feature is that the mappings reduced the exponents for both cases of $\rho^{(2)}$ and $\mathcal{H}$, which show us that the improvement is more expressive as higher is the number of particles.

Nevertheless, despite we have emphasized how suitable was the introduction of the jump

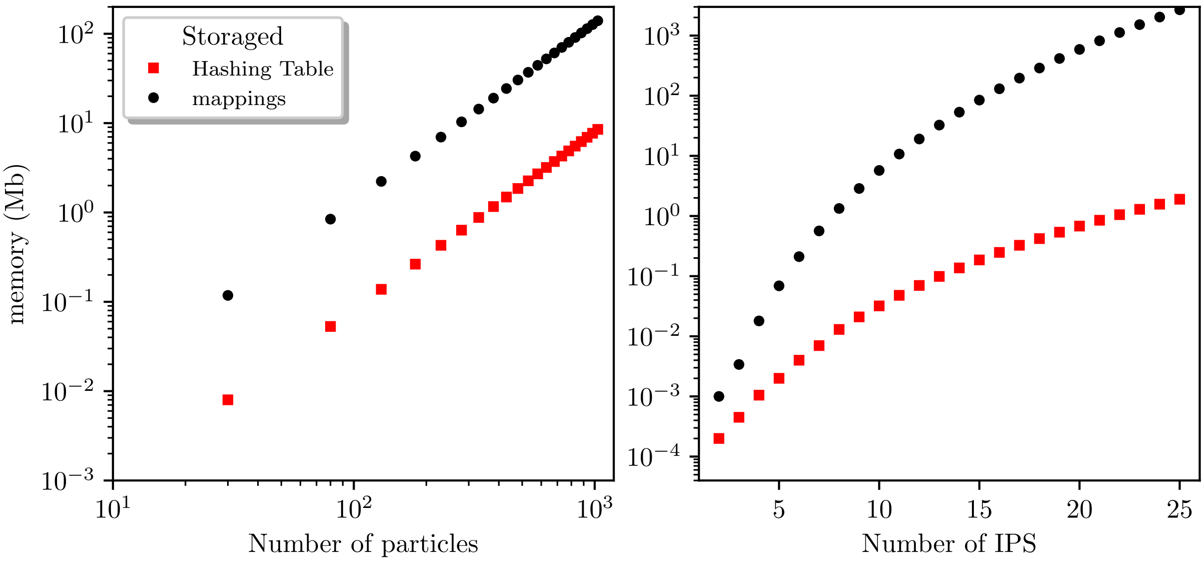

Figure 6: Memory cost of the jump mappings structure of creation/annihilation operators and hashing table for the number of particles and IPS used in previous figures measured in megabytes (MB). In the left panel again we set 3 IPS fixed and in the right panel 5 particles were used.

| | Fitting for memory | |
|---|---|---|
| | $a_M$ | $b_M$ |
| hashing table | 1.9953(2) | $8.288(5) \times 10^{-6}$ |
| jump mappings | 2.0004(1) | $1.3158(1) \times 10^{-4}$ |

Table 2: Fitted parameters of power law for the memory consumption as function of $N$ for $M = 3$ fixed.

mappings structure, we need to check the limits of application in terms of the additional memory demanded.

Indeed, all the gain in time had a cost in memory, as showed in Fig. 6. From the left panel, the case we vary the number of particles, we see that this cost is relatively cheap, some hundreds of megabytes (MB), right the case the performance gain was more expressive. The case in the right panel shows that we can not ignore the memory consumption since it demanded up to some thousands of MB, which is not a problem for present workstations, but indicates that a possible limitation may come up if one extrapolate $M = 25$ IPS with $N = 5$ particles.

It is worth to highlight the very similar behaviour between the memory cost in Fig. 6 and time demanded in Fig. 5, where again a constant slope can be identified in the log scale plot with respect to the number of particles. The results of the fitting parameters for memory consumption are shown in table 3.

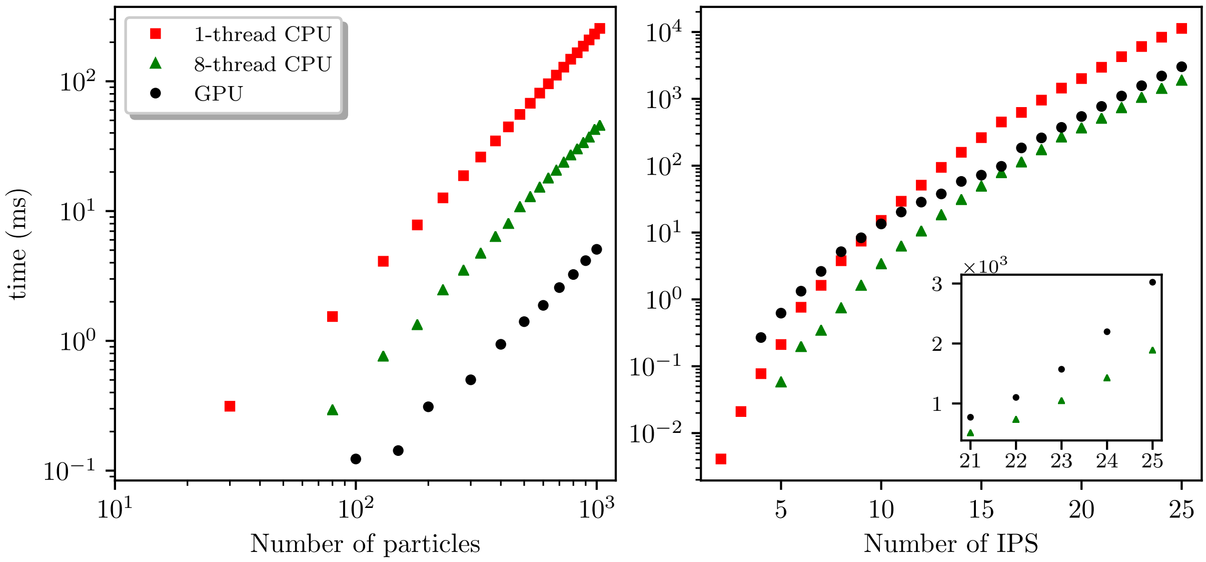

Figure 7: Performance experiment using single thread(red squares) and 8 threads(green triangles) of CPU and variable number of threads of GPU using CUDA(black circles). Just as before in the left was used 3 IPS and in the right 5 particles. The inset shows in normal scale the difference between CUDA and CPU parallelization for a region they are close in log scale plot.

## 4. Massive parallel processors application

The use of GPUs to speed up numerical evaluations is not novel. In the past decade the interest for these tools has dragged attention due to their effectiveness in improvements for workstations, in some cases as good as supercomputers. Here, in view of the Lanczos algorithm [25] for diagonalization and the time evolution Eq. (11) that is fundamental for the MCTDHB [27], we focus on an analysis based on the time required to apply the Hamiltonian in the configurational space.

Using the same parameters that were chosen throughout this article, in Fig. 7 we compare the time used for a routine to act with the Hamiltonian in configuration space, with all mappings described above, varying the number of threads in parallelization of the routine. For codes running in GPU we dynamically chose the number of blocks of threads, each one with 256 threads, for optimal usage of architecture using CUDA. Thus depending on the size of configurational space it adaptively setup the blocks.

From the left panel in Fig. 7 we see that the function is highly scalable, since the presence of more threads impact critically the performance. Nevertheless, for the right panel we see that the scalability is much smaller than the first case. An explanation for this result lies in the amount of work each thread perform, since we have only 5 particles in this case, there are a lot of configurations that have forbidden transitions and thus some threads have much less operations to do, and this benefit the CPU cores that have higher clock frequency. Thus, the scalability in out implementation, depends on a filling factor, how many particles are by IPS, where as higher this filling factor is more equally will be the work among the threads.

## 5. Conclusion and Outlook

In this article, we brought different ways to implement an effective indexing of configurations to represent a many-particle state, which is of main concern for developing numerical multi-configuration methods. We generalize the problem assuming that all particles in any individual particle state interact with each other, without restricting to a lattice with just nearest neighbor interaction or focusing in spin system. Therefore the time demanded exposed here is an upper bound for any bosonic system.

It was discussed carefully the performance and limitations of the different ways to build routines for the main physical quantities, and how direct mappings of indexes can be done to track the action of creation/annihilation operators. The limits of applicability with the mappings structure is explored in terms of memory required and our results shows to be reproducible in fair workstations.

More than the algorithm engineering done, we carried out an study of the impact from massive parallelization using GPU and compared with CPU, revealing details about the scalability of the implementation as well. For the most demanding cases, for roughly 1000 particles the best improvement was achieved by the GPU with a reduction by a factor 50 in time compared to single thread, whereas for 25 IPS the best result was with 8-threaded CPU with approximately a reduction by a factor 6 in time respect to single thread execution.

A broad audience can benefit from our results, mainly those using fair workstations for their simulations, whose can be equipped with GPU, reaching a performance that can even compete with clusters, and the detailed and general treatment develop here can popularize the numerical methods to solve numerically many-particle systems, which in the case of MCTDHB remains restricted to some groups.

All the codes used to present the result in this article are hosted in https://github.com/andriati-alex/mcpi.

## Appendix A. Technical Details of the Resources

The CPU used was an Intel® Xeon® CPU E5-2620 v4 with clock rate 2.10GHz and 8 cores. The CPU parallelization were done using the OpenMP API. The GPU used for simulation was a NVIDIA Tesla K40c, with CUDA compiler.

## Appendix B. Matrix elements for the reduced two-body density matrices

We follow Ref. [36], and present the reduced one-body and two-body density matrices explicitly, addapted to our notation. Starting with a $\beta$ configuration, $\beta_a^b$ is a resulting configuration index where one particle from the $a$-th orbital is removed and then added to the $b$-th orbital. Analogously, starting from a $\beta$ configuration, $\beta_{ab}^{cd}$ is a resulting configuration index where, two particles are removed from the $a$-th and $b$-th orbitals and then added to the $c$-th and $d$-th orbital, respectively. Sums over $\beta$ index ranges from 1 to $N_c$.

$$\rho_{kk} = \sum_{\beta}^{N_c} C_\beta^* C_\beta n_k,$$

$$\rho_{kl} = \sum_{\beta}^{N_c} C_\beta^* C_{\beta_k^l} \sqrt{(n_l+1)n_k},$$

$$\rho_{kkkk} = \sum_{\beta}^{N_c} C_\beta^* C_\beta (n_k^2 - n_k),$$

$$\rho_{kkkl} = \sum_{\beta}^{N_c} C_\beta^* C_{\beta_k^l} (n_k - 1)\sqrt{n_k(n_l+1)},$$

$$\rho_{ksss} = \sum_{\beta}^{N_c} C_\beta^* C_{\beta_k^s} n_s \sqrt{n_k(n_s+1)},$$

$$\rho_{ksks} = \sum_{\beta}^{N_c} C_\beta^* C_\beta n_k n_s,$$

$$\rho_{kkqq} = \sum_{\beta}^{N_c} C_\beta^* C_{\beta_{kk}^{qq}}$$

$$\rho_{kkql} = \sum_{\beta}^{N_c} C_\beta^* C_{\beta_{kk}^{ql}} \sqrt{(n_k-1)n_k(n_q+1)(n_l+1)},$$

$$\rho_{ksqq} = \sum_{\beta}^{N_c} C_\beta^* C_{\beta_{ks}^{qq}} \sqrt{n_k n_s (n_q+1)(n_q+2)},$$

$$\rho_{kssl} = \sum_{\beta}^{N_c} C_\beta^* C_{\beta_k^l} n_s \sqrt{n_k(n_l+1)},$$

$$\rho_{ksql} = \sum_{\beta}^{N_c} C_\beta^* C_{\beta_{ks}^{ql}} \sqrt{n_k n_s (n_q+1)(n_l+1)}.$$

## Aknowledgements

The authors thank the Brazilian agencies Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). We are also grateful for NVIDIA for providing us the Tesla K40c GPU.

[1] L. Pitaevskii, S. Stringari, Bose-Einstein Condensation and Superfluidity, 2nd ed., Oxford University Press, United Kingdom, 2016.

[2] M. H. Anderson, J. R. Ensher, M. R. Matthews, C. E. Wieman, E. A. Cornell, Observation of Bose-Einstein condensation in a dilute atomic vapor, Science 269 (5221) (1995) 198–201. doi:10.1126/science.269.5221.198.
URL http://science.sciencemag.org/content/269/5221/198

[3] O. Penrose, L. Onsager, Bose-Einstein condensation and liquid helium, Phys. Rev. 104 (1956) 576–584. doi:10.1103/PhysRev.104.576.
URL https://link.aps.org/doi/10.1103/PhysRev.104.576

[4] I. Bloch, J. Dalibard, W. Zwerger, Many-body physics with ultracold gases, Rev. Mod. Phys. 80 (2008) 885–964. doi:10.1103/RevModPhys.80.885.
URL https://link.aps.org/doi/10.1103/RevModPhys.80.885

[5] H. A. Gersch, G. C. Knollman, Quantum cell model for bosons, Phys. Rev. 129 (1963) 959–967. doi:10.1103/PhysRev.129.959.
URL https://link.aps.org/doi/10.1103/PhysRev.129.959

[6] D. Jaksch, P. Zoller, The cold atom hubbard toolbox, Annals of Physics 315 (1) (2005) 52 – 79, special Issue. doi:https://doi.org/10.1016/j.aop.2004.09.010.
URL http://www.sciencedirect.com/science/article/pii/S0003491604001782

[7] D. Jaksch, C. Bruder, J. I. Cirac, C. W. Gardiner, P. Zoller, Cold bosonic atoms in optical lattices, Phys. Rev. Lett. 81 (1998) 3108–3111. doi:10.1103/PhysRevLett.81.3108.
URL https://link.aps.org/doi/10.1103/PhysRevLett.81.3108

[8] M. P. A. Fisher, P. B. Weichman, G. Grinstein, D. S. Fisher, Boson localization and the superfluid-insulator transition, Phys. Rev. B 40 (1989) 546–570. doi:10.1103/PhysRevB.40.546.
URL https://link.aps.org/doi/10.1103/PhysRevB.40.546

[9] T. D. Kühner, H. Monien, Phases of the one-dimensional Bose-Hubbard model, Phys. Rev. B 58 (1998) R14741–R14744. doi:10.1103/PhysRevB.58.R14741.
URL https://link.aps.org/doi/10.1103/PhysRevB.58.R14741

[10] C. Bruder, R. Fazio, G. Schön, Superconductor–mott-insulator transition in Bose systems with finite-range interactions, Phys. Rev. B 47 (1993) 342–347. doi:10.1103/PhysRevB.47.342.
URL https://link.aps.org/doi/10.1103/PhysRevB.47.342

[11] M. Albiez, R. Gati, J. Fölling, S. Hunsmann, M. Cristiani, M. K. Oberthaler, Direct observation of tunneling and nonlinear self-trapping in a single bosonic josephson junction, Phys. Rev. Lett. 95 (2005) 010402. doi:10.1103/PhysRevLett.95.010402.
URL https://link.aps.org/doi/10.1103/PhysRevLett.95.010402

[12] M.-S. Chang, C. D. Hamley, M. D. Barrett, J. A. Sauer, K. M. Fortier, W. Zhang, L. You, M. S. Chapman, Observation of spinor dynamics in optically trapped $^{87}$Rb Bose-Einstein condensates, Phys. Rev. Lett. 92 (2004) 140403. doi:10.1103/PhysRevLett.92.140403.
URL https://link.aps.org/doi/10.1103/PhysRevLett.92.140403

[13] C. Wang, C. Gao, C.-M. Jian, H. Zhai, Spin-orbit coupled spinor Bose-Einstein condensates, Phys. Rev. Lett. 105 (2010) 160403. doi:10.1103/PhysRevLett.105.160403.
URL https://link.aps.org/doi/10.1103/PhysRevLett.105.160403

[14] E. H. Lieb, R. Seiringer, Proof of Bose-Einstein condensation for dilute trapped gases, Phys. Rev. Lett. 88 (2002) 170409. doi:10.1103/PhysRevLett.88.170409.
URL https://link.aps.org/doi/10.1103/PhysRevLett.88.170409

[15] E. H. Lieb, R. Seiringer, J. Yngvason, Bosons in a trap: A rigorous derivation of the Gross-Pitaevskii energy functional, Phys. Rev. A 61 (2000) 043602. doi:10.1103/PhysRevA.61.043602.
URL https://link.aps.org/doi/10.1103/PhysRevA.61.043602

[16] R. Lopes, C. Eigen, N. Navon, D. Clément, R. P. Smith, Z. Hadzibabic, Quantum depletion of a homogeneous Bose-Einstein condensate, Phys. Rev. Lett. 119 (2017) 190404. doi:10.1103/PhysRevLett.119.190404.
URL https://link.aps.org/doi/10.1103/PhysRevLett.119.190404

[17] M. Naraschewski, R. J. Glauber, Spatial coherence and density correlations of trapped Bose gases, Phys. Rev. A 59 (1999) 4595–4607. doi:10.1103/PhysRevA.59.4595.
URL https://link.aps.org/doi/10.1103/PhysRevA.59.4595

[18] R. G. Dall, A. G. Manning, S. S. Hodgman, W. RuGway, K. V. Kheruntsyan, A. G. Truscott, Ideal n-body correlations with massive particles, Nat. Phys. 9 (2013) 341–344. doi:10.1038/nphys2632.
URL https://doi.org/10.1038/nphys2632

[19] S. S. Hodgman, R. I. Khakimov, R. J. Lewis-Swan, A. G. Truscott, K. V. Kheruntsyan, Solving the quantum many-body problem via correlations measured with a momentum microscope, Phys. Rev. Lett. 118 (2017) 240402. doi:10.1103/PhysRevLett.118.240402.
URL https://link.aps.org/doi/10.1103/PhysRevLett.118.240402

[20] N. Navon, A. L. Gaunt, R. P. Smith, Z. Hadzibabic, Critical dynamics of spontaneous symmetry breaking in a homogeneous Bose gas, Science 347 (6218) (2015) 167–170. doi:10.1126/science.1258676.
URL https://science.sciencemag.org/content/347/6218/167

[21] A. Weiße, H. Fehske, Exact diagonalization techniques, in: H. Fehske, R. Schneider, A. Weiße (Eds.), Computational Many-Particle Physics, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 529–544. doi:10.1007/978-3-540-74686-7_18.
URL https://doi.org/10.1007/978-3-540-74686-7_18

[22] J. M. Zhang, R. X. Dong, Exact diagonalization: the Bose–Hubbard model as an example, European Journal of Physics 31 (3) (2010) 591–602. doi:10.1088/0143-0807/31/3/016.
URL https://doi.org/10.1088/0143-0807/31/3/016

[23] H. Q. Lin, Exact diagonalization of quantum-spin models, Phys. Rev. B 42 (1990) 6561–6567. doi:10.1103/PhysRevB.42.6561.
URL https://link.aps.org/doi/10.1103/PhysRevB.42.6561

[24] D. Raventós, T. Graß, M. Lewenstein, B. Juliá-Díaz, Cold bosons in optical lattices: a tutorial for exact diagonalization, Journal of Physics B: Atomic, Molecular and Optical Physics 50 (11) (2017) 113001. doi:10.1088/1361-6455/aa68b1.
URL https://doi.org/10.1088/1361-6455/aa68b1

[25] C. Lanczos, An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, J. Res. Natl. Bur. Stand. B 45 (1950) 255–282. doi:10.6028/jres.045.026.
URL https://doi.org/10.6028/jres.045.026

[26] C. F. V. Loan, G. H. Golub, Matrix Computations, The Johns Hopkins University Press, London, 1996.

[27] O. E. Alon, A. I. Streltsov, L. S. Cederbaum, Multiconfigurational time-dependent hartree method for bosons: Many-body dynamics of bosonic systems, Phys. Rev. A 77 (2008) 033613. doi:10.1103/PhysRevA.77.033613.
URL https://link.aps.org/doi/10.1103/PhysRevA.77.033613

[28] R. Roy, A. Gammal, M. C. Tsatsos, B. Chatterjee, B. Chakrabarti, A. U. J. Lode, Phases, many-body entropy measures, and coherence of interacting bosons in optical lattices, Phys. Rev. A 97 (2018) 043625. doi:10.1103/PhysRevA.97.043625.
URL https://link.aps.org/doi/10.1103/PhysRevA.97.043625

[29] A. U. J. Lode, B. Chakrabarti, V. K. B. Kota, Many-body entropies, correlations, and emergence of statistical relaxation in interaction quench dynamics of ultracold bosons, Phys. Rev. A 92 (2015) 033622. doi:10.1103/PhysRevA.92.033622.
URL https://link.aps.org/doi/10.1103/PhysRevA.92.033622

[30] J. H. V. Nguyen, M. C. Tsatsos, D. Luo, A. U. J. Lode, G. D. Telles, V. S. Bagnato, R. G. Hulet, Parametric excitation of a Bose-Einstein condensate: From Faraday waves to granulation, Phys. Rev. X 9 (2019) 011052. doi:10.1103/PhysRevX.9.011052.
URL https://link.aps.org/doi/10.1103/PhysRevX.9.011052

[31] K. Sakmann, A. I. Streltsov, O. E. Alon, L. S. Cederbaum, Reduced density matrices and coherence of trapped interacting bosons, Phys. Rev. A 78 (2008) 023615. doi:10.1103/PhysRevA.78.023615.
URL https://link.aps.org/doi/10.1103/PhysRevA.78.023615

[32] K. Sakmann, A. I. Streltsov, O. E. Alon, L. S. Cederbaum, Exact quantum dynamics of a bosonic josephson junction, Phys. Rev. Lett. 103 (2009) 220601. doi:10.1103/PhysRevLett.103.220601.
URL https://link.aps.org/doi/10.1103/PhysRevLett.103.220601

[33] A. W. Sandvik, Computational studies of quantum spin systems, AIP Conference Proceedings 1297 (1) (2010) 135–338. arXiv:https://aip.scitation.org/doi/pdf/10.1063/1.3518900, doi:10.1063/1.3518900.
URL https://aip.scitation.org/doi/abs/10.1063/1.3518900

[34] S. Liang, A perfect hashing function for exact diagonalization of many-body systems of identical particles, Comput. Phys. Commun. 92 (1) (1995) 11 – 15. doi:https://doi.org/10.1016/0010-4655(95)00108-R.
URL http://www.sciencedirect.com/science/article/pii/001046559500108R

[35] C. Jia, Y. Wang, C. Mendl, B. Moritz, T. Devereaux, Paradeisos: A perfect hashing algorithm for many-body eigenvalue problems, Comput. Phys. Commun. 224 (2018) 81 – 89. doi:https://doi.org/10.1016/j.cpc.2017.11.011.
URL http://www.sciencedirect.com/science/article/pii/S0010465517303946

[36] A. U. J. L. Lode, M. C. T. Tsatsos, E. Fasshauer, L. P. R. Lin, P. Molignini, C. Lévêque, S. E. Weiner, MCTDH-X: The Time-dependent multiconfigurational Hartree for indistinguishable particles software, http://ultracold.org, accessed: 2019-07-17.