

# CONAN – the cruncher of local exchange coefficients for strongly interacting confined systems in one dimension

N. J. S. Loft,<sup>1</sup> L. B. Kristensen,<sup>1</sup> A. E. Thomsen,<sup>1,2</sup> A. G. Volosniev,<sup>3,1</sup> and N. T. Zinner<sup>1</sup>

<sup>1</sup>*Department of Physics and Astronomy, Aarhus University, DK-8000 Aarhus C, Denmark*

<sup>2</sup>*CP<sup>3</sup>-Origins & the Danish Institute for Advanced Study Danish IAS,  
University of Southern Denmark, DK-5230 Odense M, Denmark.*

<sup>3</sup>*Institut für Kernphysik, Technische Universität Darmstadt, 64289 Darmstadt, Germany*

(Dated: March 8, 2016)

We consider a one-dimensional system of particles with strong zero-range interactions. This system can be mapped onto a spin chain of the Heisenberg type with exchange coefficients that depend on the external trap. In this paper, we present an algorithm that can be used to compute these exchange coefficients. We introduce an open source code CONAN (Coefficients of One-dimensional N-Atom Networks) which is based on this algorithm. CONAN works with arbitrary external potentials and we have tested its reliability for system sizes up to around 35 particles. The computation time typically scales with the number of particles as  $O(N^{3.5\pm 0.4})$ . Computation times are around 10 seconds for  $N = 10$  particles and less than 10 minutes for  $N = 20$  particles.

## I. INTRODUCTION

The study of strongly interacting systems might be motivated by the realization that coherent quantum phenomena, such as high-temperature superconductivity, Helium superfluidity, and quantum magnetism, seems to occur in systems where particle interactions are intrinsically strong. In one-dimensional (1D) systems, examples are strong Coulomb interactions in quantum nanowires and in nanotubes, as well as linear compounds with strong exchange interactions that provide realizations of various spin chains. To treat these systems in a 1D context, some popular approaches include bosonization and Luttinger liquid theory [1], the Bethe ansatz [2], and, most recently, the density matrix renormalization group [3, 4] invented by Steven White.

Also, it has now become possible to build setups with cold atomic gases that can provide quantum simulation of these low-dimensional quantum few- and many-body systems [5–9]. In particular, the Tonks-Girardeau gas [10, 11] consisting of strongly interacting bosons in 1D was realized in experiments [12–16]. Luttinger-liquid behavior in strongly interacting bosons was also reported [17]. More recently, strongly interacting fermionic systems were realized in 1D [18], also in the limit of just a few particles [19–22]. In the latter setup it was shown that one can experimentally access strongly interacting few-body systems in the regime where they realize a Heisenberg spin chain [23]. These developments were a motivation for our work during the last few years which resulted in the method presented below.

Before we proceed with our presentation we overview some previous studies related to our work. In 1960 Girardeau [11] demonstrated the close connection between strongly interacting bosons and spin-polarized/spinless fermions. In particular, the ground state of the bosons can be obtained by taking the absolute value of the corresponding fermionic wave function. This connection was subsequently generalized in various ways. In 2004, within

the context of atomic systems, Girardeau and Olshanii provided the so-called Fermi-Bose mapping strategy to describe two-component Fermi and Bose systems [24], and in 2006 Girardeau combined these observations with exchange interactions that related the mapping to spin models [25]. In the strongly interacting regime a formal mapping that produces eigenstates of the total spin and the spin projection along one direction was then proposed not long after [26]. Note that this connection was also established within condensed-matter physics where a study from 1990 by Ogata and Shiba [27] showed how the 1D Hubbard model becomes a spin model for strong interactions by using the Bethe ansatz solution. The spin models that were derived have exchange coefficients that are site-independent because the starting point was usually either homogeneous 1D space or a lattice model with some discrete translation invariance.

It was not until around 2013, when scientists realized that once you deviate from the homogeneous case, the spin models and particularly their exchange coefficients should be site-dependent as they need to reflect the geometric landscape of the inhomogeneity of the confinement potential that is typical of many experimental setups. This was initially identified as a short-coming of the Fermi-Bose mapping in the case of harmonic confinement [28, 29]. It was then realized that this does not spell the end of the Fermi-Bose mapping and effective spin model Hamiltonians [30–32], but it does mean that a more careful treatment that takes the external confinement into account is necessary. What this implies is that a confined  $N$ -body system will have a set of distinct 'local exchange coefficients' that are completely specified by the external potential. The first reference to provide an explicit formula for these coefficients is Volosniev *et al.* [30], and it was also realized in this paper that the exchange coefficients do not depend on the system composition (bosons/fermions/mixtures, two- or more internal degrees of freedom) as long as the interactions between all species are strong and the particle masses are the same. This

paper was soon followed by several other papers exploring various approximations to these coefficients [33] and their applications [34–41].

Due to the presence of confinement in the experimental setups, it is crucial to be able to compute these coefficients to describe the strongly interacting regime. However, the explicit formula for the local exchange coefficients is quite complicated because it involves an  $N - 1$  dimensional integrals over an antisymmetrized  $N$ -body wave function (a Slater determinant) [30]. In the present paper, we describe a procedure for computing these coefficients that bypasses the complexity of high-dimensional integrations, and introduce the associated software called CONAN that computes the coefficients for up to 35 particles in arbitrary external potentials.

The source code to the CONAN program is freely available at [42] including pre-compiled versions that run out of the box. A description of how to use CONAN can be found in Section VI and VII and in the documentation accompanying the program. We ask that in any scientific publication based wholly or in part on the CONAN software, the use of CONAN must be acknowledged and the present paper must be cited.

## II. THE SYSTEM

In this section we discuss the mapping of a strongly interacting one-dimensional gas onto a spin chain, see Figure 1. A more detailed discussion of the mapping can be found in Ref. [32].

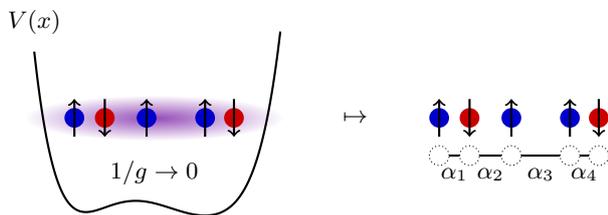


FIG. 1: Illustration of the mapping for a system with the interaction strength,  $g$ . In the strong interaction regime,  $1/g \rightarrow 0$ , this system can be related (see the text) to a spin chain with trap-dependent exchange coefficients,  $\alpha_k$ .

### A. Confined strongly interacting gas

We consider a system of particles in a one-dimensional trapping potential  $V(x)$ . All particles are of the same mass,  $M$ , and divided into two types which we call spin up and spin down. We denote by  $N_\uparrow$  ( $N_\downarrow$ ) the number of spin up (spin down) particles and by  $N = N_\uparrow + N_\downarrow$  the total number of particles. The spin up particles have coordinates  $x_1, x_2, \dots, x_{N_\uparrow}$ , whereas  $x_{N_\uparrow+1}, \dots, x_N$  are the positions of the spin down particles. We postulate that a spin up particle interacts with a spin down particle

via a contact interaction of strength  $g > 0$ . Furthermore, we assume that other interactions have strength  $\kappa g$  with  $\kappa > 0$ . The dynamics of the system is governed by the many-body Hamiltonian

$$H = \sum_{i=1}^N H_0(x_i) + g \sum_{i=1}^{N_\uparrow} \sum_{j=N_\uparrow+1}^N \delta(x_i - x_j) + \frac{\kappa g}{2} \sum_{i,j=1}^{N_\uparrow} \delta(x_i - x_j) + \frac{\kappa g}{2} \sum_{i,j=N_\uparrow+1}^N \delta(x_i - x_j), \quad (1)$$

where the single-particle Hamiltonian is given by

$$H_0(x) = -\frac{1}{2} \frac{\partial^2}{\partial x^2} + V(x), \quad (2)$$

where for simplicity we put  $\hbar = M = 1$ . We assume that particles of the same type are bosons. However, from the limit  $\kappa \rightarrow \infty$  one can learn about fermions, see, e.g., Ref. [32].

### B. Effective spin chain model

Before we consider the case of strong interaction, i.e.,  $1/g \rightarrow 0$ , we focus on the Tonks-Girardeau (TG) limit ( $1/g = 0$ ). In the TG limit, the wavefunctions vanish whenever any two particles meet,  $x_i = x_j$ . Therefore, in this limit the particles cannot exchange their positions, and for each ordering of particles (e.g.  $x_1 < x_2 < \dots < x_N$ ) the system is described with a wave function of spinless fermions  $\Phi_0(x_1, \dots, x_N)$ . For clarity from now on we assume that  $\Phi_0$  corresponds to a ground state of spinless fermions. The discussion, however, can be easily extended to the excited manifolds. To construct  $\Phi_0$  we should find  $\psi_i(x)$  – the real eigenstates of the single-particle Hamiltonian

$$H_0(x) \psi_i(x) = E_i \psi_i(x), \quad i = 1, 2, \dots, \quad (3)$$

where  $E_i$  are the corresponding eigenvalues. From the  $N$  lowest eigenstates, we construct the Slater determinant wavefunction,  $\Phi_0$ . The corresponding energy  $E_0$  equals to the sum of the  $N$  lowest single-particle energies,  $E_i$ . Note that the system has  $N!/(N_\uparrow! \cdot N_\downarrow!)$  distinct orderings, and therefore the ground state manifold of  $H$  is  $N!/(N_\uparrow! \cdot N_\downarrow!)$ -fold degenerate.

When the interaction strength is moved slightly away from the TG limit, the wavefunctions become non-vanishing at  $x_i = x_j$ , thus allowing the particles to hop. To linear order in  $1/g$ , we can describe this hopping using a spin chain Hamiltonian with nearest neighbor interaction:

$$H_s = E_0 - \sum_{k=1}^{N-1} \frac{\alpha_k}{g} \left[ \frac{1}{2} (1 - \sigma^k \cdot \sigma^{k+1}) + \frac{1}{\kappa} (1 + \sigma_z^k \sigma_z^{k+1}) \right], \quad (4)$$

where  $\sigma^k = (\sigma_x^k, \sigma_y^k, \sigma_z^k)$  are the Pauli matrices acting on the spin of the particle at the  $k$ th position and  $\alpha_k$  are the exchange coefficients. In Ref. [32] it was shown that

$$\alpha_k = \frac{\int_{x_1 < x_2 < \dots < x_{N-1}} dx_1 \dots dx_{N-1} \left| \frac{\partial \Phi_0}{\partial x_N} \right|_{x_N=x_k}^2}{\int_{x_1 < x_2 < \dots < x_N} dx_1 dx_2 \dots dx_N \Phi_0^2}, \quad (5)$$

for convenience we assume that  $\Phi_0$  is normalized such that  $\int_{x_1 < x_2 < \dots < x_N} dx_1 dx_2 \dots dx_N \Phi_0^2 = 1$ . The exchange coefficients  $\alpha_k$  are also called geometric coefficients, derived from the fact that they only depend on the geometry of the trap potential.

### III. ALGORITHM FOR CALCULATING $\alpha_k$

Access to the coefficients  $\alpha_k$  would effectively solve the strongly interacting  $N$ -body trapped system, and, thus, we seek a way to compute them. In principle, one could find the  $N$  lowest energy single-particle wavefunctions  $\psi_i$ , construct the Slater determinant wavefunction,  $\Phi_0$ , and evaluate the  $(N-1)$ -dimensional integral in Eq. (5). However, multidimensional integrals are hard to compute numerically. Therefore, for more than just a few particles, we need to cast  $\alpha_k$  in a form better suited for numerical calculations. In Appendix A we show that  $\alpha_k$  can be expressed as a sum of one-dimensional integrals

$$\begin{aligned} \alpha_k &= 2 \sum_{i=1}^N \sum_{j=1}^N \sum_{l=0}^{N-1-k} \frac{(-1)^{i+j+N-k}}{l!} \binom{N-l-2}{k-1} \\ &\times \int_a^b dx \frac{2m}{\hbar^2} (V(x) - E_i) \psi_i(x) \frac{d\psi_j}{dx} \\ &\times \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0} \\ &+ \sum_{i=1}^N \left[ \frac{d\psi_i}{dx} \right]_{x=b}^2, \end{aligned} \quad (6)$$

where  $\mathbf{I}$  denotes the identity matrix,  $(\ )^{(ij)}$  denotes the  $ij$ 'th minor obtained by removing the  $i$ 'th column and the  $j$ 'th row,  $B(x)$  is a  $N \times N$  symmetric matrix with the  $mn$ 'th entry defined as the partial wavefunction overlap

$$[B(x)]_{m,n} := \int_a^x dy \psi_m(y) \psi_n(y). \quad (7)$$

The interval  $[a, b]$  denotes the support of  $V(x)$ . For example, for a harmonic oscillator potential the support is the entire  $x$ -axis, i.e.,  $[a, b] = (-\infty, \infty)$ ; for a hard box, defined from  $x = 0$  to  $x = L$ , we have  $[a, b] = [0, L]$ .

In Eq. (6) the  $(N-1)$ -dimensional integral has been rewritten as a sum of terms which scale more advantageously with  $N$ . Unfortunately, the new expression contains the  $l$ 'th order derivative in the square brackets which complicates numerical calculations as  $N$  increases. Therefore, we would like to simplify the expression further.

### A. Simplifying the determinant

From the standpoint of an effective numerical implementation of Eq. (6) the problem lies in evaluating of the derivatives of the determinant,

$$\left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0}, \quad (8)$$

in an efficient manner. Our method for evaluating this expression is due to the fact that  $B$  is symmetric (and real because we have chosen real wavefunctions  $\psi_i$  without loss of generality), and, hence, diagonalizable using an orthogonal matrix  $U = (\mathbf{u}_1 \dots \mathbf{u}_N)$  such that  $B = U^T D U$ , where  $D$  is a diagonal matrix composed of the eigenvalues of  $B$ . We note that taking the  $ij$ 'th minor of  $B$  is equivalent to removing a row and a column from  $U^T$  and  $U$  respectively. This observation allows us to show (see Appendix B for details) that the expression in Eq. (8) can be written as

$$(-1)^{i+j} l! \mathbf{u}_j^T \left( \sum_{n=0}^l p_{l-n} D^{-(n+1)} \right) \mathbf{u}_i, \quad (9)$$

where  $p_k$  are the coefficients of the polynomial

$$\det(D - \lambda \mathbf{I}) = p_N \lambda^N + \dots + p_1 \lambda + p_0. \quad (10)$$

Several comments are in order here. First of all, as  $D$  is diagonal, it can be easily inverted as long as its entries are nonzero. In Appendix B we prove that this is in fact the case. Secondly, the coefficients  $p_k$  are easily computable because  $\det(D - \lambda \mathbf{I})$  is a determinant of a diagonal matrix. Thirdly, a further reduction in computation requirements can be achieved by doing the sum over  $l$  inside the integral. Then we diagonalize  $B$  only once, rather than once for each  $l$ . Therefore, to evaluate the integrand we need to compute the expression

$$\begin{aligned} &(-1)^{i+j} \sum_{l=0}^{N-1-k} \binom{N-2-l}{k-1} \frac{1}{l!} \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0} \\ &= \mathbf{u}_j^T \left[ \sum_{l=0}^{N-1-k} \binom{N-2-l}{k-1} \sum_{n=0}^l p_{l-n} D^{-(n+1)} \right] \mathbf{u}_i. \end{aligned} \quad (11)$$

Finally, and perhaps most interestingly, the expression in Eq. (11) depends on  $i$  and  $j$  only through  $\mathbf{u}_i$  and  $\mathbf{u}_j$ . Hence the majority of the computations are independent on  $i$  and  $j$  which prompts us to take the sum over  $i$  and  $j$  inside the integral in Eq. (6) as well and reuse the result for the derivatives of the determinant. The significance of this procedure is a great reduction in computation time as the derivatives have to be computed only once for each  $x$  rather than doing it  $\sim N^2$  times.

### B. Procedure

Here we outline our method for computing the coefficients  $\alpha_k$  from Eq. (6), where we first take all the sums

for a given  $x$  and then perform the integration. To take the sums the following is done:

1. The entries of  $B(x)$  are evaluated.
2. The matrix  $B(x)$  is diagonalized.
3. The coefficients  $p_k$  are computed.
4. The matrices  $D^{-(n+1)}$  are computed for all  $0 \leq n \leq N - 1 - k$  utilizing that  $D$  is diagonal.
5. The matrix inside the square brackets of Eq. (11) is evaluated.
6. The sum over  $i$  and  $j$  is taken. For this the product in Eq. (11) is multiplied by the appropriate factors of  $\psi_i$ ,  $d\psi_j/dx$ , and  $(V - E_i)$ . Note that the matrix in the brackets is diagonal.

From the procedure sketched above we can estimate how the computation time scales with the number of particles,  $N$ . It seems that the steps 2, 5, and 6 are the most demanding for large  $N$ , because the computation time of all these steps scales roughly as  $O(N^3)$ . Still, this scaling is surprisingly good.

To further reduce the computation time we have exploited the fact that modern computers have multiple computing cores. The easiest way to go about parallelizing the program is to let each core compute a separate geometric coefficient. As there is not a great overlap in computation from coefficient to coefficient this way of going about it will use the computing power much more effectively. Furthermore, the geometric coefficients only have a use when all of them are known, so there is not much drawback to doing this.

### C. Arbitrary precision matrix computations

There is one caveat associated with the outlined procedure which we should like to bring attention to. The success of our algorithm relies on an accurate diagonalization of  $B(x)$ . Recall that the matrix  $B$  is composed of partial overlap integrals of the wave functions. Some of these overlaps may be much smaller than the others depending on which overlap and to what value of  $x$  it is being evaluated. Therefore the entries of  $B$  might span many orders of magnitude, and the use of too small precision in the diagonalization of  $B$  might yield a wrong outcome.

It turned out that the usual machine precision is insufficient for  $N \sim 7$ . This is likely due to an enhanced error because the eigenvalues are taken to a high negative power in Eq. (11). We have found ourselves forced to implement a diagonalization routine using numbers with until several thousands bits precision (for large  $N$ ) in order to determine the eigenvalues and eigenvectors accurately. The requirement for the numerical precision to this diagonalization scales with the number of particles

so this inevitably influences the original estimate of an  $O(N^3)$  computation time. In Section VII we will see that the computation time typically scales with  $O(N^{3.5 \pm 0.4})$ .

### D. Expanding the one-body problem in a box

The computation of  $B(x)$  from Eq. (7) is straightforward. It requires the eigenstates of the one-particle Hamiltonian  $H_0(x)$  from Eq. (2). Furthermore, it requires a numerical procedure to perform the integration in Eq. (7). Both of these requirements can be fulfilled by picking a suitable basis of states for the one-body problem. As such a basis, we choose the eigenstates of a hard box potential on the interval  $[0, L]$ :

$$\phi_n(x) = \sqrt{\frac{2}{L}} \sin\left(\frac{n\pi x}{L}\right), \quad n = 1, \dots, N_b, \quad (12)$$

where  $N_b$  defined the basis size. The basis of  $N_b$  box wavefunctions in Eq. (12) form an orthogonal basis on  $[0, L]$ , which is complete in the limit  $N_b \rightarrow \infty$ .

To solve the one-body problem accurately we need to pick the values for  $N_b$  and  $L$  carefully. The number of basis states should be chosen much larger than the number of particles,  $N_b \gg N$ , such the higher momentum contributions are not important. For our calculations it is sufficient to have the basis size in the order of a few hundred.

The basis states are defined on a finite interval, therefore  $L$  should be chosen sufficiently large such that the wavefunctions,  $\psi_i$ , are effectively confined to the region  $[0, L]$ . In other words, putting  $a = 0$  and  $b = L$  in Eq. (6) should not change the value of  $\alpha_k$  within a given precision.

After expressing the single-particle Hamiltonian,  $H_0(x)$ , as a  $N_b \times N_b$  matrix in the basis of box eigenstates, we find the expansion coefficients for the wavefunctions,  $\psi_i$ , through the usual diagonalization procedure. Let us denote with

$$[C]_{i,m} = \int_0^L dx \phi_m(x) \psi_i(x), \quad (13)$$

the expansion coefficients, and with  $C$  the  $N \times N_b$  matrix of these. Then we derive the simple expression (see Appendix C) for the matrix  $B$

$$B(x) = C f(x) C^T, \quad (14)$$

where  $f(x)$  is the matrix:

$$[f(x)]_{m,n} = \begin{cases} \frac{1}{\pi} \left[ \frac{1}{m-n} \sin\left(\frac{(m-n)\pi x}{L}\right) - \frac{1}{m+n} \sin\left(\frac{(m+n)\pi x}{L}\right) \right] & \text{for } m \neq n \\ \frac{1}{\pi} \left[ \frac{\pi x}{L} - \frac{1}{2m} \sin\left(\frac{2m\pi x}{L}\right) \right] & \text{for } m = n \end{cases} \quad (15)$$

The elements of this matrix are less computationally expensive to calculate than the original integrals in the

$B(x)$ -matrix, and the coefficient matrix  $C$  only needs to be calculated once. Calculating  $B(x)$  by using Eq. (14) is therefore significantly faster than performing the  $N^2$  integrals in Eq. (7) numerically. It should be noted that this increase in speed does not alter the way the computation time scales with the number of particles. Simply doing the  $N^2$  integrals from Eq. (7) would ideally scale as  $O(N^2)$ , but in practice increases a bit faster due to the difficulty of integrating the rapidly oscillating excited states that come into play when  $N$  becomes large. Similarly, calculating  $B(x)$  using Eq. (14) requires 2 matrix products involving the  $C$ -matrix. This matrix has  $N$  rows and  $N_b$  columns, and so the calculation scales as  $O(N^2 N_b^2)$ . Because the basis size does not need to be altered significantly as  $N$  is increased, the scaling of this calculation is also close to, but slightly larger than,  $O(N^2)$ .

#### IV. UNITS AND CHANGE OF UNITS

Using the algorithm presented in the previous section we write a code CONAN, which produces  $\alpha_k$  for given  $V(x)$ . We start the presentation of our code by describing the conversion of the relevant quantities into unitless numbers that CONAN works with.

When looking at the expressions above, only two types of dimensions appear: dimensions of length and dimensions of energy. Therefore, we need to specify what units are used for lengths and energies. We let  $\ell$  to be some unit of length, and define from this the unit of energy as

$$\varepsilon = \frac{1}{2\ell^2}. \quad (16)$$

Let the unitless variable that corresponds to a quantity  $q$  be denoted as  $\tilde{q}$ , i.e.,  $\tilde{x} = x/\ell$  and  $\tilde{V}(x) = V(x)/\varepsilon$ . It is easy to show (see Appendix D) that

$$\tilde{\alpha}_k = \ell^3 \alpha_k. \quad (17)$$

The procedure for translating from a physical system to the language of the program and back works as follows:

1. Pick a unit of length  $\ell$ .
2. Pick a sufficiently large box of length  $\tilde{L}$ .
3. Find the dimensionless potential as  $\tilde{V}(\tilde{x}) = V(x/\ell)/\varepsilon$ .
4. Enter  $\tilde{L}$  and  $\tilde{V}(\tilde{x})$  into the program and run it.
5. Convert  $\tilde{\alpha}_k$  to  $\alpha_k$  using Eq. (17).

As an example, let us consider a simple harmonic oscillator,

$$V(x) = \frac{\omega^2 x^2}{2}. \quad (18)$$

For the concrete implementation in the program, we should shift it  $x \rightarrow x - L/2$  such that the minimum

of  $V(x)$  is in the center of the box, but this matters not for the present discussion.

A reasonable choice for the unit of length and the corresponding unit of energy is

$$\ell = \sqrt{\frac{1}{\omega}}, \quad (19)$$

$$\varepsilon = \frac{1}{2}\omega. \quad (20)$$

With these definitions, the numerical potential to feed into the program becomes

$$\begin{aligned} \tilde{V}(\tilde{x}) &= \frac{1}{\varepsilon} V(\tilde{x}\ell) \\ &= \frac{1}{\omega} \omega^2 \tilde{x}^2 \ell^2 \\ &= \omega \tilde{x}^2 \ell^2 \\ &= \tilde{x}^2, \end{aligned} \quad (21)$$

and when the program finishes calculating the coefficients  $\tilde{\alpha}_k$ , these should be converted back to dimensional quantities as

$$\alpha_k = \omega^{\frac{3}{2}} \tilde{\alpha}_k. \quad (22)$$

With this example in mind, we can now approach the subject of how the program behaves when we scale the units. Assume we scale the unit of length by a factor  $\delta$ . Because the unit of energy is connected to  $\ell$  by Eq. (16), this would also rescale the unit of energy. The scaling then is

$$\ell \rightarrow \delta\ell, \quad (23)$$

$$\varepsilon \rightarrow \delta^{-2}\varepsilon, \quad (24)$$

$$\tilde{V}(\tilde{x}) \rightarrow \delta^2 \tilde{V}(\delta\tilde{x}). \quad (25)$$

Similarly, a scaling of the energy by a factor  $\lambda$  implies a scaling of the unit of length,

$$\varepsilon \rightarrow \lambda\varepsilon, \quad (26)$$

$$\ell \rightarrow \lambda^{-\frac{1}{2}}\ell, \quad (27)$$

$$\tilde{V}(\tilde{x}) \rightarrow \lambda^{-1} \tilde{V}(\lambda^{-\frac{1}{2}}\tilde{x}). \quad (28)$$

Note that these two types of scaling are identical up to a change of notation  $\delta = \lambda^{-2}$ .

For the harmonic oscillator in the example above, we see that changing the unit of energy to  $\gamma\omega/2$  yields the following change of the unitless potential

$$\tilde{V}(\tilde{x}) \rightarrow \gamma^{-1} \tilde{V}(\gamma^{-\frac{1}{2}}\tilde{x}) = \gamma^{-2} \tilde{x}^2.$$

The change of the units can be seen as the change of the size of the potential. For some potentials, this observation might be used to yield scaling laws for  $\alpha_k$ , as explored in the following section.

## V. SCALING OF THE COEFFICIENTS WITH POTENTIAL STRENGTH

As mentioned above, the understanding of how the units enter in the problem allows for the derivation of scaling laws for  $\alpha_k$ . To see this, note first that the only information about the system entered into the calculations is the potential  $\tilde{V}(\tilde{x})$ . In other words, if we pick a sufficiently large  $\tilde{L}$  (as well as suitable other precision-parameters, as described in Section VI), the coefficients should depend only on the potential, i.e.,

$$\tilde{\alpha}_k = \tilde{\alpha}_k[\tilde{V}(\tilde{x})]. \quad (29)$$

Assume now that we are dealing with a potential which is homogeneous function, that is, a potential where there is a point  $a \in (0, L)$  and a real number  $s$  such that for any  $k \in \mathbb{R}$  the following holds:

$$V(k(x - a)) = k^s V(x - a). \quad (30)$$

One example of such a potential, with  $s = 2$ , is the harmonic oscillator introduced in the previous section.

For potentials of this type, a scaling of the size of the potential may be countered by a rescaling of  $\ell$ , so that the combinations of the two scalings keep  $\tilde{V}$  and, therefore,  $\tilde{\alpha}$  unchanged. As shown in Appendix D, this leads to the following scaling-law when the potential is scaled by a factor  $\gamma$ :

$$\alpha_k(\gamma V) = \gamma^{\frac{3}{s+2}} \alpha_k(V). \quad (31)$$

For the harmonic case with  $s = 2$  the data from the program, run with varying potential-sizes, confirms that the coefficients scale as  $\gamma^{\frac{3}{4}}$ .

## VI. GUIDE TO THE USE OF CONAN

Our program, CONAN, is a powerful theoretical tool, but in order to receive a useful output that can be used to solve the physical problem at hand, one has to think carefully about the input given to the program. In this section we go through the steps one should consider before submitting an input to CONAN, and we discuss the precision of the results. Then in Section VII we consider two specific examples: a harmonic potential and an asymmetric tilted potential in a box.

1. Choose the number of particles,  $N$ , between 2 and up to around 35. It is sometimes possible to perform calculations for up to 40 particles, but since the error on some of the coefficients might become unpredictable and large, it is generally not advised to use  $N > 35$ . Submit the number of particles with the option `-N`.
2. Choose a smooth dimensionless potential,  $\tilde{V}(\tilde{x})$ , defined on the interval  $[0, \tilde{L}]$ . When choosing the

potential, it is important to think of the physics of problem. Since the coefficients are local exchange coefficients, one should think of how the particles respond to the chosen potential and what the wavefunctions look like. Unless the expansion box is considered a part of the potential, allowing for non-analyticity of the potential at  $x = 0$  and  $x = L$ , the potential at the boundaries of the box should be much larger than the typical energy scale of the system such that the presence of the box does not affect the  $N$  lowest-energy single-particle wavefunctions. Choosing  $\tilde{V}(0) \sim \tilde{V}(\tilde{L})$  of the order of a few hundreds is sufficient. Submit the dimensionless potential as a mathematical function with the option `-V`. The length of the box may be changed from the default value,  $\tilde{L} = 100$ , by submitting another value using the option `-L`.

For some potentials, especially potentials that are antisymmetric around some point, it is often a good idea to translate the potential such that  $\tilde{V}(\tilde{L}/2) = 0$ . Also, a constant potential should be chosen to be zero,  $\tilde{V}(\tilde{x}) = 0$ . The technical reason for this is due to the numerical integration routines when the matrix elements for the potential is calculated.

3. Choose the number of box basis states,  $N_b$ , large enough such that the  $N$  lowest-energy single-particle wavefunctions are well represented in the basis of box states. The default value of the basis size in CONAN is  $N_b = 300$ , but it is sometimes necessary to increase it with the option `-b` depending on the potential and the number of particles. Typically it is not a problem to choose  $N_b$  larger than necessary, but it may happen that picking  $N_b$  too large results in instabilities in the numerical integral routines due to quickly oscillating integrands in the matrix elements that represent the potential in the basis of box states,  $V_{nm} = \int_0^L dx \phi_n(x) \phi_m(x) V(x)$ . One can check whether  $N_b$  is chosen sufficiently by comparing the results for different values of  $N_b$ , for instance  $N_b = 200, 400, 600, 800$ ;  $N_b$  should be chosen in the range where the coefficients do not change with changing  $N_b$  (within some desired precision).

Increasing the basis size increases the computation time by a few minutes through the diagonalization of the  $N_b \times N_b$  Hamiltonian matrix, but the increased computation time is mostly independent of  $N$ . So if one is performing a few calculations for large  $N \sim 30$ , where the computation time is an hour or so, it is a good idea to pick  $N_b$  on the safe side of what is necessary. On the other hand, if one is performing many calculations for a small  $N \sim 10$  with almost identical potentials (say, for noise studies), then a lot of computation time can be saved by optimizing  $N_b$  to fulfill ones desired precision.

4. Choose the bit precision on the arbitrary precision calculations,  $p$ , with the option `-p`. The default

value is 256, and it should be increased when  $N$  increases. As a rule of thumb, always choose the bit precision as a power of 2, i.e. it should be increased in steps 256, 512, 1024, 2048. If the bit precision is not sufficiently large, the computation will run into an error or give coefficients that clearly wrong by being several orders of magnitude different from each other, but if a reasonable result can be obtained with the chosen bit precision, it does typically not improve the precision of the result to increase  $p$ . Since higher bit precision increases the computation time drastically, one do not want to set  $p$  higher than necessary. For just a few particles,  $N \lesssim 5$ , it suffices to use a bit precision smaller than the default value, say  $p = 64$  or  $p = 128$ .

5. Consider whether it is necessary to increase the absolute or relative precision on the calculation of the single-particle wavefunctions and energies with the options `--abs-solver` and `--rel-solver` or the integral in Eq. (6) with `--abs-final` and `--rel-final`. Unless one is working with complicated-looking potentials or many particles,  $N > 30$ , it is usually not necessary to change these precision parameters from their default values.

The above list serve as a checklist one should review before submitting an input to CONAN. There are other setting that can be changed in order to achieve a specific output format or parameter dependent potentials, please consult the documentation for further details.

It is important to evaluate the result from the program. Even though CONAN does not return an error, the resulting coefficients may be wrong or not the physically desired results. One should ask oneself whether it makes sense to interpret the coefficients as local exchange coefficients, given the studied potential. To estimate the precision of a result, one can perform the calculation again with increased basis size and/or precision and compare the two results. One should keep increasing the basis size and/or precision as described above until the coefficients stabilizes to the desired degree of precision. However this method only works as long as the precision of the results can be increased by tuning the basis size and the precision parameters, which is up to around 30 particles.

For very large particle numbers over 30, we advise the user to be extra cautious because the complexity of the calculations introduce errors in the results that become harder and harder to avoid, even with increased basis size and precision parameters. Since the calculation time is also large, it becomes very tedious to estimate the precision of the result by performing multiple calculations with different precision settings. There is, however, a way to estimate the error on the coefficients that works also for large values of  $N$ : We know that reflecting the potential around the middle of the box would also reflect the coefficients, i.e.  $V(x) \mapsto V(L-x)$  yields  $\alpha_k \mapsto \alpha_{N-k}$ . Thus comparing  $\alpha_k$  calculated for  $V(x)$  and  $\alpha_{N-k}$  for  $V(L-x)$  digit by digit, we can estimate the error on the coefficients

as their deviation. This method only works because the coefficients we compare are calculated differently, i.e. we do not compare the results of identical numerical routines, and thus it makes sense to estimate the error in this way. In the special case of a symmetric potential,  $V(x) = V(L-x)$ , we can compare the coefficients directly by checking to which degree of accuracy the coefficients are symmetric,  $\alpha_{N-k} = \alpha_k$ .

We are hesitant to state anything general about the precision of the results because every case should be considered separately, but we do note a few trends. For  $N \leq 25$  high precision is only a matter of picking  $N_b$  and  $p$  sufficiently, and the error can be reduced to less than 0.0001% or 0.00001%. Increasing the number of particles to  $N \approx 30$  introduces a small error on a few coefficients of the order 0.001% that cannot be reduced by changing  $N_b$  or the precision parameters. We also note that the error is not evenly distributed among the  $N-1$  coefficients, but a few coefficients gain larger errors while the remaining coefficients are still very precise. This trend of non-uniformly distributed error continues as  $N$  increases. For  $N = 35$  the largest errors are of the order 0.1% to 1%, depending on the system. Increasing  $N$  to around 40, the largest errors become unacceptably large, but since the computation time is several hours for such large  $N$ , we have not done any systematic studies in this range of  $N$ . We recommend that CONAN is run for at most  $N \approx 35$ .

## VII. EXAMPLES

Let us in the following turn to two specific examples to demonstrate typical values of the various settings. We wish to calculate the geometric coefficients for various values of  $N$ . As  $N$  increases we gradually increase the basis size  $N_b$  and bit precision  $p$  in an effort to get high precision results and keeping the computation time low. We run CONAN on a computer with an Intel Xenon processor (CPU E5-2630 v3 @ 2.40 GHz  $\times$  8) and note the computation time,  $T$ . To get an estimate of how  $T$  scales with  $N$ , we fit our data to the model  $T = a + b \cdot N^c$ , where  $a, b, c$  are fit parameters. We find that the computation time, depending on the potential and range of  $N$  values, typically scales as  $O(N^{3.5 \pm 0.4})$ , and that it is generally very low: Calculating for  $N = 10$  particles takes approximately 10 seconds, and computations for  $N = 20$  can be done in less than 10 minutes.

### A. Harmonic potential

Suppose we want to know the geometric coefficients for a harmonic oscillator potential

$$V(x) = \frac{1}{2}m\omega^2x^2 \quad (32)$$

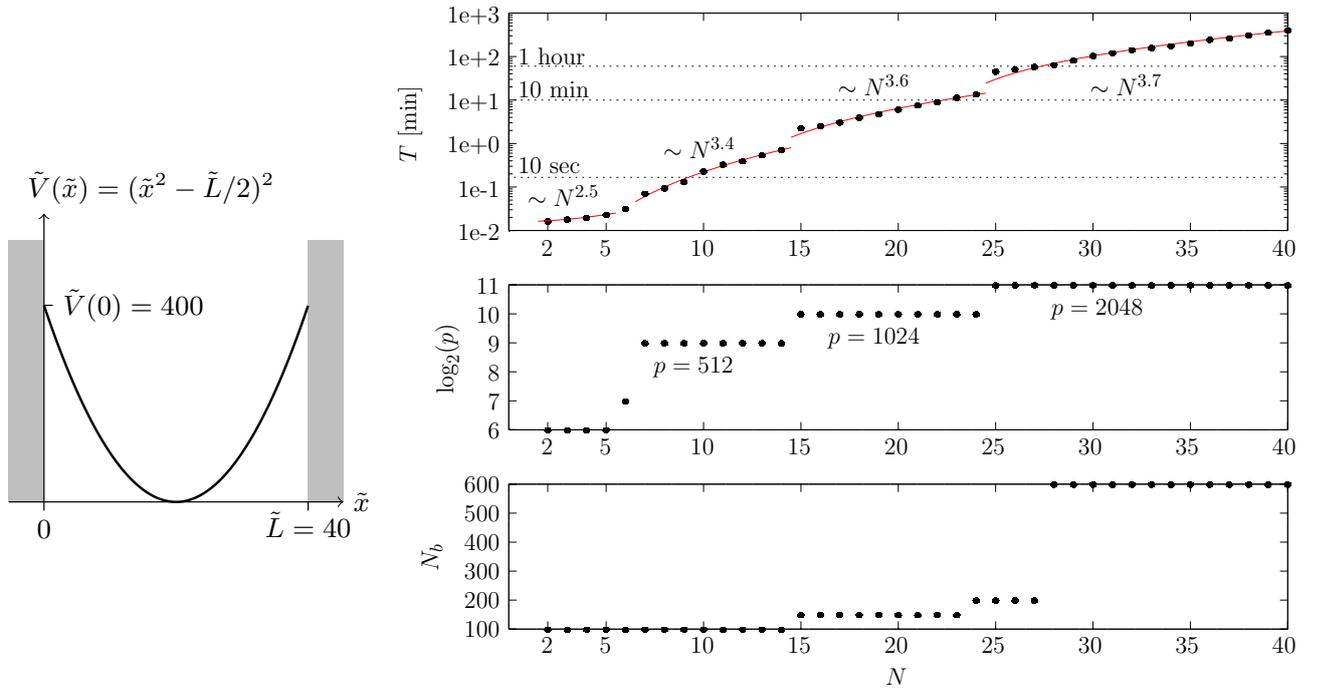


FIG. 2: **Harmonic oscillator.** To the left we show a sketch of the harmonic potential submitted to CONAN. To the right, on the top panel, we plot the computation time  $T$  versus the number of particles  $N$  including the scaling on the four plateaux indicated. Below we show the chosen bit precision  $p$  (middle) and basis size  $N_b$  (bottom). The errors on the calculated coefficients are below 0.0001% for  $N < 29$  which increases to  $\sim 0.1\%$  for  $N = 35$  and becomes even larger for larger  $N$ , see the main text for further discussion.

for various values of  $N$  using the typical oscillator units given in Eqs. (19) and (20). These were calculated for  $N \leq 30$  in Ref. [39]. In the oscillator units, as shown in Section IV, the dimensionless potential becomes:

$$\tilde{V}(\tilde{x}) = \tilde{x}^2. \quad (33)$$

However, we cannot submit the above potential to CONAN because the potential has its minimum at the boundary of the box, so we shift the minimum of the potential to the center of the box.

$$\tilde{V}(\tilde{x}) = (\tilde{x} - \tilde{L}/2)^2. \quad (34)$$

A shift in the  $x$ -direction does not change the geometric coefficients. At the boundaries of the box, the above potential yields  $\tilde{V}(0) = \tilde{V}(\tilde{L}) = \tilde{L}^2/4$ , so picking  $\tilde{L} = 40$  leads to boundary value of the potential being 400, which is what we want. As an alternative to changing  $\tilde{L}$  from the default value of 100, we could have scaled the potential and rescaled the coefficients accordingly as described in Section V.

We calculate the geometric coefficients for  $N = 2, \dots, 40$  with gradually increasing basis size  $N_b$  and bit precision  $p$ . An input line to the program may look like `./Conan -V '(x-L/2)^2 -L 40 -N 8 -p 512 -b 100'`.

The basis size, bit precision and computation time for  $N = 2, \dots, 40$  are shown in Figure 2. In the top

panel of the figure we show the computation time  $T$  versus  $N$ . We see that the computation time jumps drastically when the bit precision is increased. For the four plateaux characterized with  $p = 64, 512, 1024$  and  $2048$ , we calculate how the computation time scales with  $N$  and see that it typically scales with  $O(N^{3.5 \pm 0.2})$ .

The precision on the resulting coefficients can be estimated from calculations using a larger number of basis states and higher precision parameters up to around 30 particles. Since the potential is symmetric around the middle of the box, we may also estimate of the precision by checking to which degree the coefficients are symmetric,  $\alpha_k = \alpha_{N-k}$ .

For  $N < 29$  the error on the coefficients are all less than 0.0001%. For larger  $N$  the error on a few coefficients increases to 0.001% for  $N = 30$ , while most of the coefficients are still symmetric to a high degree of precision. For  $N = 33$  the error on the most imprecise coefficients is  $\sim 0.01\%$ , growing to  $\sim 0.1\%$  for  $N = 35$ . It is worth noticing that the precision on the  $N = 35$  result is not increased by increasing the basis size to  $N_b = 800$  and the absolute and relative integral precision to  $1e-08$  using the options `--abs-final` and `--rel-final`, indicating that the precision cannot become better for these large  $N$  results. Pushing the limits of CONAN's capabilities by increasing  $N$  further, we note that the error increases to a few percent for  $N = 37$  and up to worst case devi-

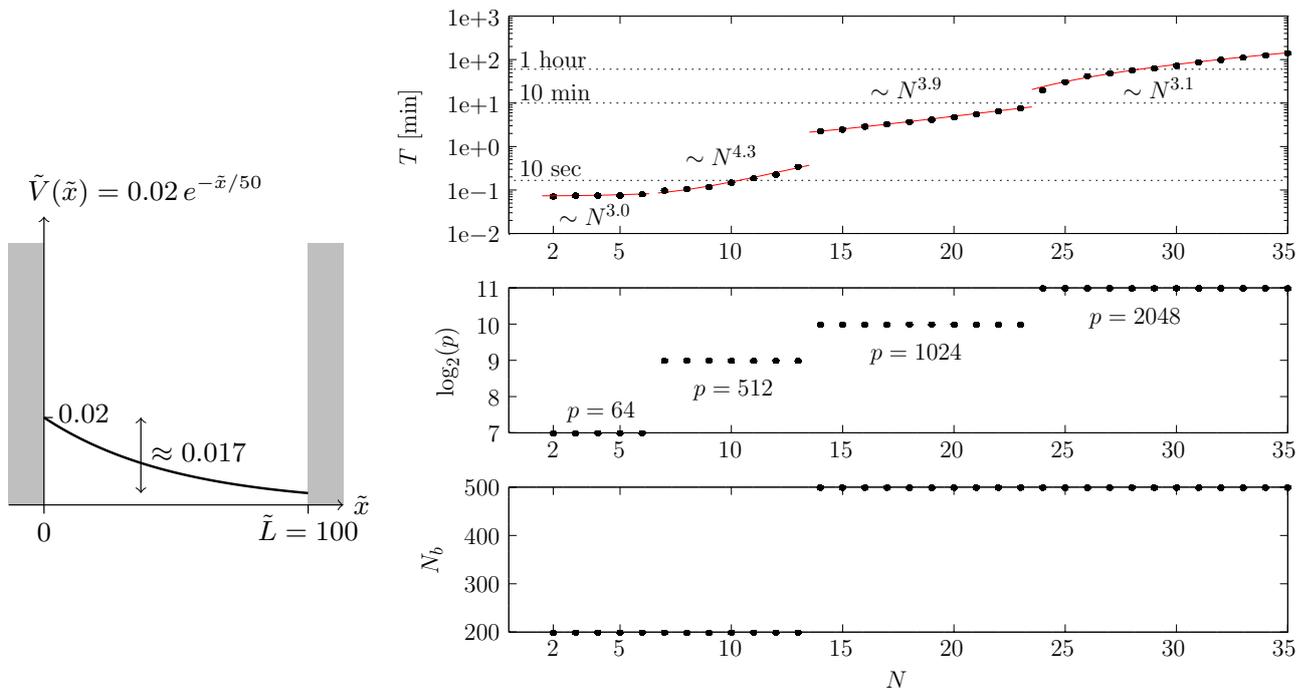


FIG. 3: **Asymmetric tilted potential in a box.** To the left we show a sketch of the potential submitted to CONAN. To the right, on the top panel, we plot the computation time  $T$  versus the number of particles  $N$  including the scaling on the four plateaux indicated. Below we show the chosen bit precision  $p$  (middle) and basis size  $N_b$  (bottom). The error is 0.0001% or less for  $N \lesssim 28$ , but increases to at most 3% for  $N = 35$ .

ations  $\sim 100\%$  for  $N = 40$ . Clearly, one cannot blindly trust results for these large values of  $N$ , and we advocate that CONAN is run for no more than approximately 35 particles.

### B. Asymmetric tilted potential in a box

In this example we exploit that the system is placed in a box of length  $L = 100\ell$ , where  $\ell$  is some length unit. We modify the simple box potential by adding an exponentially decaying potential, thus the resulting potential is not symmetric around the middle of the box, but tilted to one side,

$$V(x) = \begin{cases} 0.02 e^{-x/50} & \text{if } 0 < x < L \\ \infty & \text{otherwise} \end{cases} \quad (35)$$

where the numerical factors have been chosen such that the difference  $V(0) - V(L) \approx 0.017\varepsilon$  is comparable with the energy scale of the system, and the energy unit  $\varepsilon$  used by CONAN is given by Eq. (16). Thus we submit to CONAN the following dimensionless potential:

$$\tilde{V}(\tilde{x}) = 0.02 e^{-\tilde{x}/50}. \quad (36)$$

We calculate the geometric coefficients for  $N = 2, \dots, 35$  with the basis size  $N_b$  and bit precision  $p$  chosen as to

retain a high degree of precision on the results, but without spending too much computation time. See Figure 3 where we in the top panel show how the computation time scales with  $N$  at the four plateaux characterized by  $p = 64, 512, 1024$  and 2048.

We found that  $N_b = 200$  was sufficient to ensure an error below 0.0001% for  $N < 14$ , but that the basis size had to be increased for more particles to retain a high degree of precision. By comparing  $\alpha_k$  with  $\alpha_{N-k}$  calculated for the mirror reflected potential  $V(L - x)$ , we estimate an error of 0.0001% for  $N = 28$  for the most inaccurate coefficients. This increases to 0.001% for  $N = 30$  and 3% for  $N = 35$ .

## VIII. CONCLUSIONS

We have presented an algorithm for computing the local exchange coefficients for a strongly interacting  $N$ -particle system confined to one dimension by an external potential. We discussed the numerical implementation of the method into a piece of open source software, CONAN. Then we discussed the use of the program, including examples and estimates of the precision of the results. We found that CONAN could produce reliable results for up to around  $N = 35$ , and that the computation time typically scaled as  $O(N^{3.5 \pm 0.4})$ . Computation times where around 10 seconds for  $N = 10$  and less than 10

minutes for  $N = 20$ . The approach described here may be extended in a straightforward manner to compute for instance densities in strongly interacting systems or to correlation functions.

*Remark:* While preparing this paper we became aware of a recent paper by Deuretzbacher *et al.* [43] which presents a method for computing the coefficients which has some similarities to the one used here. We note that while Deuretzbacher *et al.* use a fitting method with Chebyshev polynomials to get high-order derivatives, we use a recursive formula as described in the appendix. This could have influence on the numerical stability of either algorithm. It would be very interesting to explore whether one may combine these two different approach to achieve even larger stability than currently available in either approach.

### Acknowledgments

We would like to thank M. Valiente, A. S. Jensen, D. V. Fedorov, O. V. Marchukov, and D. Petrosyan for collab-

oration on the spin model approach. We thank E. M. Eriksen and E. J. Lindgren for discussions on early developments on a formula to compute the local exchange coefficients, as well as J. M. Midtgaard and A. A. S. Kalae for help in testing these developments in spin systems. A. G. V. and N. T. Z. acknowledges support from the Danish Council for Independent Research DFF Natural Sciences Sapere Aude program, and A. G. V. acknowledges partial support from the Helmholtz Association under contract HA216/EMMI. N. J. S. L., L. B. K., A. E. T., and N. T. Z. acknowledges support by grants from the Carlsberg Foundation. The development of the CONAN software was assisted by J. Termansen and J. H. Jensen with support from the Carlsberg Foundation.

### APPENDIX A: DERIVATION OF EQ. (6)

In this appendix we derive Eq. (6) from Eq. (5). We start by rewriting the wave function for the system of  $N$  spinless fermions (a Slater determinant) using the Leibniz formula for determinants

$$\Phi_0(x_1, \dots, x_N) = \sum_{\pi} \text{sign}(\pi) \prod_{i=1}^N \psi_{\pi(i)}(x_i), \quad (\text{A1})$$

where  $\pi$  denotes the permutation operator that acts on the set of  $N$  numbers  $\{1, 2, \dots, N\}$ , and  $\psi_j$  are the normalised one-body wave functions that are defined in the main text. Next, we write  $\alpha_k$  as

$$\alpha_k = \sum_{i=1, j=1}^{i=N, j=N} (-1)^{i+j} \int_{-\infty}^{\infty} dx_k \frac{\partial \psi_i(x_k)}{\partial x_k} \frac{\partial \psi_j(x_k)}{\partial x_k} \int_{x_1 < x_2 < \dots < x_{N-1}} dx_1 \dots dx_{k-1} dx_{k+1} \dots dx_{N-1} (f_i f_j)(x_1, \dots, x_{N-1}). \quad (\text{A2})$$

To write this expression we use the Laplace expansion for the Slater determinant, i.e.,

$$\Phi_0 = \sum_{i=1}^N (-1)^{i+N} \psi_i(x_N) f^i(x_1, \dots, x_{N-1}), \quad (\text{A3})$$

where  $f^i(x_1, \dots, x_{N-1}) = \sum_{\pi_i} \text{sign}(\pi_i) \prod_{j=1}^{N-1} \psi_{\pi_i(j)}(x_j)$ ,  $\pi_i$  is a permutation in the set of  $N-1$  elements:  $\{1, \dots, N\} \setminus i$ . To produce Eq. (6) we simplify the inner integral

$$I_{k,ij} = \int_{x_1 < x_2 < \dots < x_{N-1}} dx_1 \dots dx_{k-1} dx_{k+1} \dots dx_{N-1} (f^i f^j)(x_1, \dots, x_{N-1}). \quad (\text{A4})$$

Note that  $f^i f^j$  is a symmetric function, i.e.,  $(f^i f^j)(x_k, x_l) = (f^i f^j)(x_l, x_k)$ , this symmetry allows us to change the integration limits

$$I_{k,ij} = \frac{1}{(k-1)!(N-1-k)!} \int_a^{x_k} dx_1 \int_a^{x_k} dx_2 \dots \int_a^{x_k} dx_{k-1} \int_{x_k}^b dx_{k+1} \dots \int_{x_k}^b dx_{N-1} f^i f^j. \quad (\text{A5})$$

Throughout our investigation we noticed that similar integrals appear also for observables. Therefore, we find it useful to consider the following integral

$$P_l(c)[F\Phi] \equiv \int_a^c dx_1 \dots \int_a^c dx_{l-1} \int_c^b dx_l \dots \int_c^b dx_n F(x_1, \dots, x_n) \Phi(x_1, \dots, x_n), \quad (\text{A6})$$

where the subscript  $l$  defines how many integrals should be taken from  $a$  to  $c$ , other integrals are taken from  $c$  to  $b$ . Functions  $F$  and  $\Phi$  are determinants build with the orthonormalized sets of functions  $\{f_i\}$  and  $\{\phi_i\}$  correspondingly. That is

$$\Phi(x_1, x_2, \dots, x_n) = \begin{vmatrix} \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_n(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x_n) & \phi_2(x_n) & \cdots & \phi_n(x_n) \end{vmatrix}, \text{ and } F(x_1, x_2, \dots, x_n) = \begin{vmatrix} f_1(x_1) & f_2(x_1) & \cdots & f_n(x_1) \\ f_1(x_2) & f_2(x_2) & \cdots & f_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(x_n) & f_2(x_n) & \cdots & f_n(x_n) \end{vmatrix}.$$

Let us first consider  $P_{n+1}(c)[F\Phi]$ . Note that both  $F$  and  $\Phi$  are fully antisymmetric in their variables. Therefore, we can write  $P_{n+1}$  as

$$\begin{aligned} P_{n+1}(c)[F\Phi] &\equiv n! \int_a^c dx_1 \dots \int_a^c dx_n f_1(x_1) f_2(x_2) \dots f_n(x_n) \begin{vmatrix} \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_n(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x_n) & \phi_2(x_n) & \cdots & \phi_n(x_n) \end{vmatrix} = \\ &n! \int_a^c dx_1 \dots \int_a^c dx_n \begin{vmatrix} f_1(x_1)\phi_1(x_1) & f_1(x_1)\phi_2(x_1) & \cdots & f_1(x_1)\phi_n(x_1) \\ f_2(x_2)\phi_1(x_2) & f_2(x_2)\phi_2(x_2) & \cdots & f_2(x_2)\phi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_n(x_n)\phi_1(x_n) & f_n(x_n)\phi_2(x_n) & \cdots & f_n(x_n)\phi_n(x_n) \end{vmatrix}, \end{aligned} \quad (\text{A7})$$

the integration can be performed inside of the determinant, which leads to  $P_{n+1}(c)[F\Phi] = n! \det A(c)$  where the matrix  $A(c)$  is defined as  $A_{ij}(c) = \int_a^c dx f_i(x) \phi_j(x)$ .

Next we consider  $P_n$

$$\begin{aligned} P_n(c)[F\Phi] &= \int_a^c dx_1 \dots \int_a^c dx_{n-1} \int_c^b dx_n F(x_1, \dots, x_n) \Phi(x_1, \dots, x_n) = \\ &\sum_{i,j=1}^n (-1)^{i+j} \int_c^b dx_n f_i(x_n) \phi_j(x_n) P_n(c)[F^i \Phi^j] = (n-1)! \sum_{i,j=1}^n (-1)^{i+j} \int_c^b dx_n f_i(x_n) \phi_j(x_n) \det A^{ij}(c), \end{aligned} \quad (\text{A8})$$

where the functions  $F^i$  and  $\Phi^j$  are obtained from the Laplace expansions of  $F$  and  $\Phi$  respectively, the matrix  $A^{ij}(c)$  is obtained from the matrix  $A(c)$  by crossing out the  $i$ th row and  $j$ th column (i.e.,  $A^{ij}$  is the  $(i, j)$ -submatrix). From here we have two possible paths to proceed: First we can notice that the expression in Eq. (A8) can be rewritten using Jacobi's formula for the derivative of the determinant as  $\text{tr}(T \text{adj}(A)) = \left. \frac{\partial \det(A + \lambda \mathbf{T})}{\partial \lambda} \right|_{\lambda=0}$ , where the matrix  $T$  is defined

as  $T_{ij} = \int_c^b dx f_i(x) \phi_j(x)$ . This observation allows us to derive the identity  $P_n(c)[F\Phi] = (n-1)! \left. \frac{\partial \det(A + \lambda \mathbf{T})}{\partial \lambda} \right|_{\lambda=0}$ . By repeating the same steps (i.e., first is to write  $P_j$  using  $P_{j-1}$ , second is to use Jacobi's formula) for  $P_3, P_4, \dots$  we obtain  $P_l(c)[F, \Phi] = (l-1)! \left. \frac{\partial^{n-l+1} \det(A + \lambda \mathbf{T})}{\partial \lambda^{n-l+1}} \right|_{\lambda=0}$ .

Another path, which was used to derive Eq. (6), rests on the assumption that  $f_i$  and  $\phi_j$  are orthonogonal to each other, i.e.,  $\int_a^b dx f_i(x) \phi_j(x) = \delta_{ij}$ , where  $\delta_{ij}$  is Kronecker's delta. This assumption allows us to rewrite Eq. (A8)

$$P_n(c)[F\Phi] = (n-1)! \sum_{i,j=1}^n (-1)^{i+j} \det A^{ij}(c) (\delta_{ij} - A_{ij}) = -(n-1)! \left[ \left. \frac{\partial \det(A - \lambda \mathbf{I})}{\partial \lambda} \right|_{\lambda=0} + n \det A \right]. \quad (\text{A9})$$

Let us now consider  $P_{n-1}(c)$

$$\begin{aligned} P_{n-1}(c)[F\Phi] &= \sum_{i,j=1}^n (-1)^{i+j} P_n(c)[F^i \Phi^j] \delta_{ij} - P_n(c)[F\Phi] = \\ &(n-2)! \left[ \left. \frac{\partial^2 \det(A - \lambda \mathbf{I})}{\partial \lambda^2} \right|_{\lambda=0} + (n-1) \left. \frac{\partial \det(A - \lambda \mathbf{I})}{\partial \lambda} \right|_{\lambda=0} \right] + (n-1)! \left[ \left. \frac{\partial \det(A - \lambda \mathbf{I})}{\partial \lambda} \right|_{\lambda=0} + n \det A \right]. \end{aligned} \quad (\text{A10})$$

The pattern for  $P_l(c)$  is now clear and we write  $P_l$  as

$$P_l(c)[F\Phi] = (-1)^{n+1-l} \sum_{i=0}^{n+1-l} \frac{(n-i)!(n+1-l)!}{(n+1-l-i)! i!} \left. \frac{\partial^i \det(A - \lambda \mathbf{I})}{\partial \lambda^i} \right|_{\lambda=0}. \quad (\text{A11})$$

The meaning of the terms in the sum can be understood from the expression for  $P_{n-1}$ :  $(n-i)!$  factor always comes with the  $i$ th derivative, whereas the  $\frac{(n-i)!}{(n+1-l-i)!i!}$  shows how many terms with the  $i$ th derivative are in the expression.

Now let us establish the final ingredient needed to write down  $\alpha_k$ . For this we consider the following integral

$$I(x_1) = \int_a^c dx_2 \dots \int_a^c dx_n F(x_1, x_2, \dots, x_n) \Phi(x_1, \dots, x_n) = (n-1)! \sum_{i,j=1}^n (-1)^{i+j} \det A^{ij}(x_1) f_i(x_1) \phi_j(x_1). \quad (\text{A12})$$

Note that  $\phi_j(x_1) f_i(x_1) = \frac{\partial A_{ij}(x_1)}{\partial x_1}$ . Therefore, Jacobi's formula allows us to conclude that  $I(x) = (n-1)! \frac{\partial \det A(x)}{\partial x}$ .

Combining the expressions for  $I$  and  $P_l$  we derive

$$\alpha_k = \sum_{i=1}^N \sum_{j=1}^N \sum_{l=0}^{N-1-k} \frac{(-1)^{i+j+N-1-k}}{l!} \binom{N-l-2}{k-1} \int_a^b dx \frac{d\psi_i}{dx} \frac{d\psi_j}{dx} \frac{d}{dx} \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0}. \quad (\text{A13})$$

Partial integration allows us to eliminate the derivative of the expression in square brackets, turning the integral into

$$- \int_a^b dx \left( \frac{d^2 \psi_i}{dx^2} \frac{d\psi_j}{dx} + \frac{d\psi_i}{dx} \frac{d^2 \psi_j}{dx^2} \right) \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0} + \frac{d\psi_i}{dx} \frac{d\psi_j}{dx} \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0} \Big|_a^b.$$

Using the Schrödinger equation (3) and utilizing symmetry in  $i \leftrightarrow j$ , Eq. (A13) becomes

$$\alpha_k = 2 \sum_{i=1}^N \sum_{j=1}^N \sum_{l=0}^{N-1-k} \frac{(-1)^{i+j+N-1-k}}{l!} \binom{N-l-2}{k-1} \int_a^b dx \frac{2m}{\hbar^2} (V(x) - E_i) \psi_i(x) \frac{d\psi_j}{dx} \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0} + \mathcal{B} \quad (\text{A14})$$

where  $\mathcal{B}$  denotes the boundary term arising from the partial integration

$$\mathcal{B} = \sum_{i=1}^N \sum_{j=1}^N \sum_{l=0}^{N-1-k} \frac{(-1)^{i+j+N-1-k}}{l!} \binom{N-l-2}{k-1} \frac{d\psi_i}{dx} \frac{d\psi_j}{dx} \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0} \Big|_a^b. \quad (\text{A15})$$

Comparing Eq. (A14) with Eq. (6), we see that only need to simplify the boundary term (A15) in order to complete the proof of Eq. (6).

We start by considering

$$K_{ij}(x) = \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0} \quad (\text{A16})$$

evaluated in the boundary points  $x = a$  and  $x = b$ . Here the matrix  $B(x)$  reduces to  $B(a) = \mathbf{0}$  or  $B(b) = \mathbf{I}$ , so we consider the determinant

$$\det \left[ ((\mu - \lambda) \mathbf{I})^{(ij)} \right],$$

where  $\mu = 0$  in the lower limit  $x = a$  and  $\mu = 1$  in the upper limit  $x = b$ . If  $i = j$  then  $\mathbf{I}^{(ij)}$  is the  $(N-1) \times (N-1)$  identity matrix, but if  $i \neq j$  it has a zero-row (and zero-column), so its determinant is a Kronecker delta,  $\delta_{ij}$ . Thus we may simplify

$$\begin{aligned} \det \left[ ((\mu - \lambda) \mathbf{I})^{(ij)} \right] &= \det \left[ (\mu - \lambda) (\mathbf{I})^{(ij)} \right] \\ &= (\mu - \lambda)^{N-1} \det \left[ \mathbf{I}^{(ij)} \right] \\ &= (\mu - \lambda)^{N-1} \delta_{ij}. \end{aligned}$$

Using this expression for the determinant, one may give a simple induction argument for the relation

$$\frac{\partial^l}{\partial \lambda^l} \det \left[ ((\mu - \lambda) \mathbf{I})^{(ij)} \right] = (-1)^l \frac{(N-1)!}{(N-l-1)!} (\mu - \lambda)^{N-l-1} \delta_{ij} \quad (\text{A17})$$

for  $l = 0, 1, 2, \dots$ . Now let us evaluate Eq. (A16) in the boundary points. In the lower limit it follows from Eq. (A17) that

$$K_{ij}(a) = (-1)^l \frac{(N-1)!}{(N-l-1)!} (-\lambda)^{N-l-1} \delta_{ij} \Big|_{\lambda=0} = (-1)^l \frac{(N-1)!}{(N-l-1)!} \delta_{N-l-1,0} \delta_{ij}. \quad (\text{A18})$$

Notice that  $N-l-1 \geq N-(N-1-k)-1 \geq N-(N-2)-1 = 1 > 0$  for all the terms in the sum over  $l$  in Eq. (A15), so  $K_{ij}(a) = 0$  for all the terms in the sum. In the upper limit we have

$$K_{ij}(b) = (-1)^l \frac{(N-1)!}{(N-l-1)!} (1-\lambda)^{N-l-1} \delta_{ij} \Big|_{\lambda=0} = (-1)^l \frac{(N-1)!}{(N-l-1)!} \delta_{ij}.$$

Inserting this result into Eq. (A15), the boundary term becomes

$$\mathcal{B} = \sum_{i=1}^N \left[ \frac{d\psi_i}{dx} \right]_{x=b}^2 \sum_{l=0}^{N-1-k} (-1)^{N-1-k+l} \binom{N-l-2}{k-1} \frac{(N-1)!}{(N-l-1)! l!},$$

which simplifies as the sum over  $l$  equals unity. This can be proven using the identity

$$\sum_{l=0}^n (-1)^l \binom{j}{l} \binom{j-l-1}{n-l} = (-1)^n$$

with  $n = N-1-k$  and  $j = N-1$ . Then finally we have derived Eq. (6).

## APPENDIX B: DERIVATION OF EQ. (11)

Here we prove the equality

$$\frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \Big|_{\lambda=0} = (-1)^{i+j} l! \mathbf{u}_j^T \left( \sum_{n=0}^l p_{l-n} D^{-(n+1)} \right) \mathbf{u}_i, \quad (\text{B1})$$

presented in Sec. III.

By choosing real valued wave functions one can ensure that  $B$  is a real and symmetric matrix. By the spectral theorem it is then possible to diagonalize it using an orthogonal matrix  $U = (\mathbf{u}_1 \dots \mathbf{u}_N)$  such that  $B = U^T D U$  with  $D$  being diagonal containing the eigenvalues of  $B$ . Denote by  $U_n = (\mathbf{u}_1 \dots \mathbf{u}_{n-1} \mathbf{u}_{n+1} \dots \mathbf{u}_N)$  the matrix  $U$  with the  $n$ 'th column removed. In this notation the  $ij$ 'th minor, inside the determinant, is simply

$$(B - \lambda \mathbf{I})^{(ij)} = (U^T (D - \lambda \mathbf{I}) U)^{(ij)} = U_i^T (D - \lambda \mathbf{I}) U_j. \quad (\text{B2})$$

The trick is that it is possible to insert an extra row and column in  $U_i^T$  and  $U_j$  respectively thus making them into square matrices, but without changing the value of the determinant. This is possible when  $D - \lambda \mathbf{I}$  is invertible in a small region around  $\lambda = 0$  which is the case as long as all eigenvalues of  $B$  are non-zero. In this case

$$\begin{pmatrix} \mathbf{u}_i^T \\ U_i^T \end{pmatrix} (D - \lambda \mathbf{I}) \left( [(D - \lambda \mathbf{I})^{-1} \mathbf{u}_i] \ U_j \right) = \begin{pmatrix} 1 & \mathbf{u}_i^T (D - \lambda \mathbf{I}) U_j \\ \mathbf{0} & (B - \lambda \mathbf{I})^{(ij)} \end{pmatrix}, \quad (\text{B3})$$

where most of the first column is zero due to  $U$  being an orthogonal matrix. Expanding the determinant in the first column of the RHS of the above expression shows that the determinant is indeed unchanged. As the product matrices are now all square the determinant can be evaluated using the product rule for determinants, hence

$$\det \left[ (B - \lambda \mathbf{I})^{(ij)} \right] = \det \begin{pmatrix} \mathbf{u}_i^T \\ U_i^T \end{pmatrix} \det(D - \lambda \mathbf{I}) \det \left( (D - \lambda \mathbf{I})^{-1} \mathbf{u}_i \ U_j \right), \quad (\text{B4})$$

Changing the order of two rows in a matrix simply changes the sign, thus rearranging the first matrix on the RHS above gives

$$\begin{aligned} \det \left[ (B - \lambda \mathbf{I})^{(ij)} \right] &= (-1)^{i+j} \det(D - \lambda \mathbf{I}) \det \begin{pmatrix} \mathbf{u}_j^T \\ U_j^T \end{pmatrix} \det \left( (D - \lambda \mathbf{I})^{-1} \mathbf{u}_i \ U_j \right) \\ &= (-1)^{i+j} \det(D - \lambda \mathbf{I}) \det \begin{pmatrix} \mathbf{u}_j^T (D - \lambda \mathbf{I})^{-1} \mathbf{u}_i & 0 \\ U_j^T (D - \lambda \mathbf{I})^{-1} \mathbf{u}_i & \mathbf{I} \end{pmatrix} \\ &= (-1)^{i+j} \det(D - \lambda \mathbf{I}) \left[ \mathbf{u}_j^T (D - \lambda \mathbf{I})^{-1} \mathbf{u}_i \right], \end{aligned} \quad (\text{B5})$$

where in the last line the determinant has been expanded in the first row. This is a much more agreeable expression than we began with, as  $D - \lambda \mathbf{I}$  is a diagonal matrix.

From here on it is straightforward. The derivatives of Eq. (B5) is found by repeated use of the product rule for derivatives. First note that

$$\left. \frac{d^n}{d\lambda^n} (D - \lambda \mathbf{I})^{-1} \right|_{\lambda=0} = n! D^{-(n+1)}. \quad (\text{B6})$$

The determinant is simply a polynomial in  $\lambda$  and can be written in the form

$$\det(D - \lambda \mathbf{I}) = p_N \lambda^N + \dots + p_1 \lambda + p_0, \quad (\text{B7})$$

from which we find that

$$\left. \frac{d^n}{d\lambda^n} \det(D - \lambda \mathbf{I}) \right|_{\lambda=0} = n! p_n. \quad (\text{B8})$$

The computation of the coefficients  $p_k$  are computed most efficiently doing simply what a person would do when faced with polynomial multiplication. One after the other simply multiplying in the next factor of  $(d_i - \lambda)$ . Define by  $p_k^{(r)}$  the coefficients of the polynomial obtained through the multiplication of the first  $r$  factors. Thus  $p_0^{(1)} = d_1$  and  $p_1^{(1)} = -1$ . The iterative is then performed by letting

$$p_{k+1}^{(r+1)} = d_{r+1} p_{k+1}^{(r)} - p_k^{(r)}. \quad (\text{B9})$$

Repeating this until  $p_k^{(N)} = p_k$  gives the coefficients of the polynomial arising from the determinant. This computation is done in  $\mathcal{O}(N^2)$  steps.

With this all parts of the final expression for the derivatives of the determinants are ready. As advertised it is

$$\begin{aligned} \left[ \frac{\partial^l}{\partial \lambda^l} \det \left[ (B(x) - \lambda \mathbf{I})^{(ij)} \right] \right]_{\lambda=0} &= (-1)^{i+j} \sum_{n=0}^l \binom{l}{n} (l-n)! p_{l-n} n! \mathbf{u}_j^T D^{-(n+1)} \mathbf{u}_i \\ &= (-1)^{i+j} l! \mathbf{u}_j^T \left( \sum_{n=0}^l p_{l-n} D^{-(n+1)} \right) \mathbf{u}_i. \end{aligned} \quad (\text{B10})$$

Finally let us justify the assumption that none of the eigenvalues of  $B(x)$  are zero, since the entire procedure hinges on this fact. Assume for contradiction that there exist a non-zero vector  $\mathbf{v}$  so that  $B\mathbf{v} = \mathbf{0}$ . Due to the definition of  $B(x)$  this is the same as

$$0 = \sum_{j=1}^N [B(x)]_{ij} v_j = \int_a^x dy \psi_i(y) \sum_{j=1}^N v_j \psi_j(y), \quad \forall i \leq N. \quad (\text{B11})$$

Seeing as this holds for all  $i$  it must hold for linear combinations too. Hence

$$0 = \sum_{i=1}^N v_i \int_a^x dy \psi_i(y) \sum_{j=1}^N v_j \psi_j(y) = \int_a^x dy \left( \sum_{i=1}^N v_i \psi_i(y) \right)^2 \Rightarrow \sum_{i=1}^N v_i \psi_i(y) = 0, \quad y \in [a, x]. \quad (\text{B12})$$

Since the wave functions individually solves the Schrodinger, applying the Hamilton operator to the above equation yields

$$0 = H \sum_{i=1}^N v_i \psi_i(y) = \sum_{i=1}^N E_i v_i \psi_i(y), \quad y \in [a, x]. \quad (\text{B13})$$

If the wave functions of the linear combination are not all degenerate then Eq. (B13) represents a new linear combination compared to (B12). The two of those could then be combined to produce a new linear combination which also vanishes on  $[a, x]$ , but which has zero coefficients for all wave functions of one energy. By successive application of this argument it would be possible to make a vanishing function on  $[a, x]$  using a linear combination of wave functions of the same

energy. Therefore we may Wlog assume that the wave functions with nonvanishing coefficients in  $\sum_{i=1}^N v_i \psi_i$  all has energy  $E$ . In this case the linear combination solves the differential equation

$$(H - E) \sum_{i=1}^N v_i \psi_i(y) = 0, \quad y \in [a, b]. \quad (\text{B14})$$

The same differential equation is also solved by the 0-function. Now as the solutions  $\sum_{i=1}^N v_i \psi_i$  and 0 coincide on the subinterval  $[a, x]$  it follows from the theory of ordinary differential equations that given sufficiently smooth  $V$ , the two functions must be identical on the whole interval  $[a, b]$ . Thus we arrive at

$$\sum_{i=1}^N v_i \psi_i(y) = 0, \quad y \in [a, b]. \quad (\text{B15})$$

Now since the wave functions themselves are orthogonal, their linear combinations are known to be unique. Hence the only solution to Eq. (B15) is  $\mathbf{v} = \mathbf{0}$ , contradicting the assumption of a nonzero eigenvalue of  $B(x)$ . This concludes the proof that completely vanishing eigenvalues of the  $B(x)$ -matrix will not appear to cause problems with our algorithm.

### APPENDIX C: PROOF OF EQ. (14)

From Eqs. (7) and (13) we have that the  $B(x)$ -matrix elements can be calculated in the box expansion as

$$\begin{aligned} [B(x)]_{i,j} &= \sum_{n=1}^{N_b} \sum_{m=1}^{N_b} C_{i,m} C_{j,n} \frac{2}{L} \int_0^x dz \sin\left(\frac{m\pi z}{L}\right) \sin\left(\frac{n\pi z}{L}\right) \\ &= \sum_{n=1}^{N_b} \sum_{m=1}^{N_b} C_{i,m} C_{j,n} \frac{1}{L} \int_0^x dz \left[ \cos\left(\left(m-n\right)\frac{\pi z}{L}\right) - \cos\left(\left(m+n\right)\frac{\pi z}{L}\right) \right]. \end{aligned}$$

From the definition in Eq. (15), the integral yields exactly  $L[f(x)]_{m,n}$ , and so:

$$\begin{aligned} [B(x)]_{i,j} &= \sum_{n=1}^{N_b} \sum_{m=1}^{N_b} C_{i,m} [f(x)]_{m,n} C_{j,n} \\ &= (Cf(x)C^T)_{i,j}, \end{aligned}$$

which is what we wanted to show.

### APPENDIX D: UNITS AND SCALING OF THE COEFFICIENTS

In this section we wish to prove (17) and (31) from section IV. Note first that  $B(x)$  is unit-less. This means that we may write the terms contributing to  $\alpha_k$  on the form:

$$\int_0^L dx \frac{2m}{\hbar} (V(x) - E_i) \psi_i \left( \frac{d\psi_j}{dx} \right) K_{i,j,l}(x),$$

where  $K_{i,j,l}(x)$  is unit-less. Rewriting this in terms of unit-less objects, remembering that one-dimensional wave functions have units of length to the power  $-1/2$ , this is equal to

$$\frac{1}{\ell^3} \int_0^{\tilde{L}} d\tilde{x} \left( \tilde{V}(\tilde{x}) - \tilde{E}_i \right) \tilde{\psi}_i \left( \frac{d\tilde{\psi}_j}{d\tilde{x}} \right) K_{i,j,l}(x)$$

The integral in this expression is exactly what the program will calculate, and so by summing all of the contributions we arrive at exactly the expression from (17):

$$\alpha_k = \frac{1}{\ell^3} \tilde{\alpha}_k.$$

The scaling of the coefficients in the case of homogeneous potentials may be derived as follows:

Consider a potential where there exist a point  $a \in (0, L)$  and a real number  $s$  such that for any  $k \in \mathbb{R}$  the following holds:

$$V(k(x - a)) = k^s V(x - a).$$

From this follows that:

$$V(x - a) = V(\ell(\tilde{x} - \tilde{a})) = \ell^s V(\tilde{x} - \tilde{a}).$$

After dividing through with the energy unit this becomes:

$$\begin{aligned} \tilde{V}(\tilde{x} - \tilde{a}) &= \frac{2m\ell^2}{\hbar^2} \ell^s V(\tilde{x} - \tilde{a}) \\ &= \frac{2m}{\hbar^2} \ell^{s+2} V(\tilde{x} - \tilde{a}) \end{aligned}$$

From this we see that if we scale the potential and  $\ell$  at the same time as follows:

$$\begin{aligned} V(x) &\rightarrow \gamma V(x) \\ \ell &\rightarrow \ell' = \ell \gamma^{-\frac{1}{s+2}}, \end{aligned}$$

The only change to  $\tilde{V}$  will be a translation  $\tilde{a} = a/\ell \rightarrow a/\ell' = \tilde{a}'$ :

$$\begin{aligned} \tilde{V}'(\tilde{x} - \tilde{a}') &= \frac{2m}{\hbar^2} (\ell')^{(s+2)} \gamma V(\tilde{x} - \tilde{a}') \\ &= \frac{2m}{\hbar^2} \ell^{2+s} V(\tilde{x} - \tilde{a}') \\ &= \tilde{V}(\tilde{x} - \tilde{a}') \\ &= \tilde{V}([\tilde{x} - \tilde{a}' + \tilde{a}] - \tilde{a}). \end{aligned}$$

As long as  $\tilde{a}'$  remains far from the edges of the interval we are considering, this translation is irrelevant to the coefficients. We can therefore conclude:

$$\tilde{\alpha}_k \left( \frac{V}{\ell} \right) = \tilde{\alpha}_k \left( \frac{\gamma V}{\ell'} \right),$$

and so we predict the scaling:

$$\begin{aligned} \alpha_k(\gamma V) &= \frac{1}{(\ell')^3} \tilde{\alpha}_k \left( \frac{\gamma V}{\ell'} \right) \\ &= \gamma^{\frac{3}{s+2}} \frac{1}{\ell^3} \tilde{\alpha}_k \left( \frac{V}{\ell} \right) \\ &= \gamma^{\frac{3}{s+2}} \alpha_k(V), \end{aligned}$$

which is exactly the result in (31).

- 
- [1] T. Giamarchi: *Quantum Physics in One Dimension* (Oxford University Press, Oxford, 2003).  
[2] M. Takahashi: *Thermodynamics of One-Dimensional Solvable Models* (Cambridge University Press, Cambridge, 1999)  
[3] U. Schollwöck, Rev. Mod. Phys. **77**, 259 (2005).  
[4] U. Schollwöck, Ann. Phys. **326**, 96 (2011).  
[5] M. Lewenstein *et al.*, Adv. Phys. **56**, 243 (2007).  
[6] I. Bloch, J. Dalibard, and W. Zwerger, Rev. Mod. Phys. **80**, 885 (2008).  
[7] T. Esslinger, Ann. Rev. Cond. Matter Phys. **1**, 129 (2010).  
[8] M. A. Baranov, M. Dalmonte, G. Pupillo, and P. Zoller, Chem. Rev. **112**, 5012 (2012).  
[9] N. T. Zinner and A. S. Jensen, J. Phys. G:Nucl. Part. Phys. **40**, 053101 (2013).  
[10] L. W. Tonks, Phys. Rev. **50**, 955 (1936).  
[11] M. D. Girardeau, J. Math. Phys. **1**, 516 (1960).  
[12] M. Olshanii, Phys. Rev. Lett. **81**, 938 (1998).

- [13] B. Paredes *et al.*, *Nature* **429**, 277 (2004).
- [14] T. Kinoshita, T. Wenger, and D. S. Weiss, *Science* **305**, 1125 (2004).
- [15] T. Kinoshita, T. Wenger, and D. S. Weiss, *Nature* **440**, 900 (2006).
- [16] E. Haller *et al.*, *Science* **325**, 1224 (2009).
- [17] E. Haller *et al.*, *Nature* **466**, 597 (2010).
- [18] G. Pagano *et al.*, *Nature Phys.* **10**, 198 (2014).
- [19] F. Serwane *et al.*, *Science* **332**, 336 (2011).
- [20] G. Zürn *et al.*, *Phys. Rev. Lett.* **108**, 075303 (2012).
- [21] A. Wenz *et al.*, *Science* **342**, 457 (2013).
- [22] S. Murmann *et al.*, *Phys. Rev. Lett.* **114**, 080402 (2015).
- [23] S. Murmann *et al.*, *Phys. Rev. Lett.* **115**, 215301 (2015).
- [24] M. D. Girardeau and M. Olshanii, *Phys. Rev. A* **70**, 023608 (2004).
- [25] M. D. Girardeau, *Phys. Rev. Lett.* **97**, 210401 (2006).
- [26] L. Guan, S. Chen, Y. Wang, and Z.-Q. Ma, *Phys. Rev. Lett.* **102**, 160402 (2009).
- [27] M. Ogata and H. Shiba, *Phys. Rev. B* **41**, 2326 (1990).
- [28] S. E. Gharashi and D. Blume, *Phys. Rev. Lett.* **111**, 045302 (2013).
- [29] E. J. Lindgren *et al.*, *New J. Phys.* **16** 063003 (2014).
- [30] A. G. Volosniev *et al.*, *Nature Commun.* **5**, 5300 (2014).
- [31] F. Deuretzbacher *et al.*, *Phys. Rev. A* **90**, 013611 (2014).
- [32] A. G. Volosniev *et al.*, *Phys. Rev. A* **91**, 023620 (2015).
- [33] J. Levinsen *et al.* *Science Advances*, **1**, e1500197 (2015).
- [34] X. Cui and T.-L. Ho, *Phys. Rev. A* **89**, 023611 (2014).
- [35] X. Cui and T.-L. Ho, *Phys. Rev. A* **89**, 013629 (2014).
- [36] L. Yang, L. Guan, and H. Pu, *Phys. Rev. A* **91**, 043634 (2015).
- [37] L. Yang and X. Cui, *Phys. Rev. A* **93**, 013617 (2016).
- [38] O. V. Marchukov *et al.*, arXiv:1508.07164 (2015).
- [39] N. J. S. Loft *et al.*, arXiv:1508.05917 (2015).
- [40] P. Massignan, J. Levinsen, and M. M. Parish, *Phys. Rev. Lett.* **115**, 247202 (2015).
- [41] L. Yang and H. Pu, arXiv:1601.02556 (2016).
- [42] The CONAN source code and pre-compiled versions can be found at <http://phys.au.dk/forskning/forskningsomraader/amo/few-body-physics-in-a-many-body-world/conan/>.
- [43] F. Deuretzbacher *et al.*, arXiv:1602.06816 (2016).