CUBIC SCALING ALGORITHMS FOR RPA CORRELATION USING INTERPOLATIVE SEPARABLE DENSITY FITTING

JIANFENG LU AND KYLE THICKE

ABSTRACT. We present a new cubic scaling algorithm for the calculation of the RPA correlation energy. Our scheme splits up the dependence between the occupied and virtual orbitals in χ^0 by use of Cauchy's integral formula. This introduces an additional integral to be carried out, for which we provide a geometrically convergent quadrature rule. Our scheme also uses the newly developed Interpolative Separable Density Fitting algorithm to further reduce the computational cost in a way analogous to that of the Resolution of Identity method.

1. INTRODUCTION

In Density Functional Theory (DFT) [9, 12], the ground state energy of a many-body quantum system is written as a functional of the density ρ. In the Kohn-Sham (KS) formalism of DFT [12], instead of considering the original interacting system of N_{occ} electrons, we consider a system of N_{occ} non-interacting electrons (the KS system) under a different external potential whose ground state density is identical to that of the interacting system. In this effective single-particle system, the ground state density is given by

$$\rho(x) = \sum_{j=1}^{N_{\text{occ}}} |\psi_j(x)|^2, \tag{1.1}$$

where $\{\psi_j\}$ are the Kohn-Sham orbitals, the eigenstates of the effective single-particle system. It is assumed throughout that the KS orbitals are ordered so that ψ_1 is the ground state of the KS system, ψ_2 is the first excited state, and so on. In KS-DFT, the ground state energy of a system with N_{occ} interacting electrons can be written as

$$E = T_{\text{s}} + U_{\text{ext}} + U_{\text{H}} + E_{\text{xc}}, \tag{1.2}$$

where

$$T_{\text{s}} = \frac{1}{2}\sum_{j=1}^{N_{\text{occ}}} \int |\nabla\psi_j(x)|^2 \,\mathrm{d}x, \tag{1.3}$$

$$U_{\text{ext}} = \int V_{\text{ext}}(x)\rho(x)\,\mathrm{d}x,$$

$$U_{\text{H}} = \frac{1}{2}\int\int \rho(x)\rho(y)v(x,y)\,\mathrm{d}x\,\mathrm{d}y, \tag{1.4}$$

are, respectively, the kinetic energy of the effective single-particle system, the potential energy due to the external potential V_{ext}, and the so-called Hartree energy, which represents the classical contribution of the energy from the Coulomb interaction between electrons. The remaining term in (1.2), E_{xc}, is known as the exchange-correlation energy. It has no known simple form in terms of the density ρ or the Kohn-Sham orbitals $\{\psi_j\}$ and therefore needs to be approximated. There are many ways [18] of approximating this functional. In this work, we consider one of the more accurate (and more computational expensive) approximations, the Random

Date: April 13, 2017.
This work is partially supported by the National Science Foundation under grants DMS-1454939.

Phase Approximation (RPA). In particular, we separate out the exchange and correlation parts: $E_{\mathrm{xc}} = E_{\mathrm{x}} + E_{\mathrm{c}}$, and we use the exact exchange $E_{\mathrm{x}}^{\mathrm{EX}}$ for the exchange energy E_{x}, and the Random Phase Approximation to approximate the correlation energy E_{c} [20].

$$E_{\mathrm{x}}^{\mathrm{EX}} = -\sum_{jk} f_j f_k \int\int \psi_j^*(x)\psi_k(x)\widehat{v}(x,y)\psi_k^*(y)\psi_j(y)\,\mathrm{d}x\,\mathrm{d}y \tag{1.5}$$

$$E_{\mathrm{c}}^{\mathrm{RPA}} = \frac{1}{2\pi}\int_0^\infty \mathrm{tr}\left[\ln(1-\widehat{\chi}^0(i\omega)\widehat{v}) + \widehat{\chi}^0(i\omega)\widehat{v}\right]\mathrm{d}\omega, \tag{1.6}$$

where

$$\widehat{\chi}^0(x,y,i\omega) = \sum_{jk} \frac{(f_j - f_k)\psi_j^*(x)\psi_k(x)\psi_k^*(y)\psi_j(y)}{\varepsilon_j - \varepsilon_k - i\omega}, \tag{1.7}$$

and $\widehat{v}$ is the Coulomb kernel (in particular, we will consider the periodic Coulomb kernel), and $\mathrm{tr}[A] = \int \langle x|A|x\rangle\,\mathrm{d}x$. We will only consider the zero temperature case. This means that, in the ground state, the first N_{occ} KS orbitals are filled while the rest are empty. So, $f_\ell = 1$ if $1 \le \ell \le N_{\mathrm{occ}}$ (the occupied orbitals), and $f_\ell = 0$ if $\ell > N_{\mathrm{occ}}$ (the virtual orbitals).

In practice, one needs a way to obtain the KS orbitals before the energy can be computed. This can be done via a self-consistent iteration. However, we do not consider this here. Instead, we only consider the calculation of the energy after the KS orbitals are known. In this sense, we are considering a perturbative, non-self-consistent calculation of the RPA correlation energy. That is, in a practical implementation, the KS orbitals could be calculated via a self-consistent iteration using a computationally less expensive, but also less accurate, approximation for the exchange-correlation energy functional (e.g., LDA, GGA). The orbitals which are output from that self-consistent iteration can then be used to compute a more accurate approximation to the true correlation energy by using them to calculate the RPA correlation energy. In this way, one obtains an approximation to the true correlation energy which is better than the less expensive method (LDA, GGA, etc.), but also does not require the self-consistent iteration to deal with the expense of RPA.

Of all the terms we have defined above, the RPA correlation energy $E_{\mathrm{c}}^{\mathrm{RPA}}$ is the most computationally expensive to calculate. The goal of this paper is to provide a cubic scaling algorithm for the computation of this term. Before we can effectively talk about scaling, we must first define some notation. In this work, we will use a spatial discretization with equally spaced grid points. We denote the total number of grid points by n. We denote the number of occupied orbitals, i.e., the number of electrons, by N_{occ}. Since there are infinitely many KS orbitals $\{\psi_j\}_{j=1}^{\infty}$ [1] and the orbitals corresponding to higher energies will tend to have smaller contributions to $\widehat{\chi}^0$, we choose to use only the N_{orb} KS orbitals of lowest energy in the RPA calculation. This gives us $N_{\mathrm{vir}} = N_{\mathrm{orb}} - N_{\mathrm{occ}}$ virtual orbitals. Since n, N_{occ}, N_{vir}, and N_{orb} all grow linearly with the system size, we will sometimes refer to a general N as a characterization of the system size.

There have very recently been a few existing cubic scaling methods for RPA correlation energy calculation presented in the literature. The general idea involved is to split up the j and k dependence in the computation of $\widehat{\chi}^0$ by introducing a new integral. The idea is easy to motivate. From (1.6), we can see that everything can be done in cubic scaling if we are able to construct the matrices $\widehat{v}$ and $\widehat{\chi}^0(i\omega)$ in cubic time. $\widehat{v}$ is not hard to construct, so we will focus

[1] When the spatial discretization is fixed with n grid points, the total number reduces to n, which is much larger than N_{occ}.

on $\widehat{\chi}^0$. $\widehat{\chi}^0$ has $\mathcal{O}(N^2)$ entries, so each entry of $\widehat{\chi}^0$ must be calculated in $\mathcal{O}(N)$ time. By inspection of (1.7), it is clear that the most natural computation will take $\mathcal{O}(N^2)$ due to the coupling of j and k in the denominator. But if we can decouple the j and k dependence, then we can sum over each index separately and calculate each entry of $\widehat{\chi}^0$ in $\mathcal{O}(N)$. Two different integrals have been utilized for this purpose. The first is

$$\int_0^\infty e^{-\varepsilon_j t} e^{\varepsilon_k t} e^{i\omega}\,\mathrm{d}t = \frac{1}{\varepsilon_j - \varepsilon_k - i\omega}. \tag{1.8}$$

Using this integral, one separates the dependence of j and k in (1.7) into a product of exponentials inside the integral. This leads to the Laplace transform cubic scaling methods. This idea was first applied to RPA calculations in [11] and [10], where a projector augmented wave (PAW) basis was utilized. The idea was later extended to Gaussian basis functions in [22]. The other integral used to break up the j and k dependency is

$$\frac{1}{2\pi i}\int_{\mathscr{C}} \frac{1}{(\lambda - \varepsilon_j + i\omega)(\lambda - \varepsilon_k)}\,\mathrm{d}\lambda = \frac{1}{\varepsilon_j - \varepsilon_k - i\omega}, \tag{1.9}$$

where $\mathscr{C}$ is a positively oriented closed contour that encloses $\varepsilon_j - i\omega$, but not ε_k. This idea was first used in the context of cubic scaling RPA in [17]. Our algorithm in this paper will also adopt this idea.

The crucial difference of our method compared with previous approaches lies in the reduction of complexity prefactor of cubic scaling algorithms. In order to motivate this second main idea of this paper, let us examine how the density fitting (also called resolution of identity) approximation lowers the computational cost in the quartic scaling method [7, 19]. The idea behind the approximation is that $\widehat{\chi}^0$ is nearly equal to a low rank matrix due to its structure. So, $\widehat{\chi}^0$ (and $\widehat{v}$) are formed into smaller sized matrices (by writing $\widehat{v}$ into a smaller auxiliary basis and $\widehat{\chi}^0$ into the dual basis) before the trace is taken. The smaller matrix sizes lower the computational cost, but it is still $\mathcal{O}(N^4)$ since the coupling of j and k is unaffected by the approximation.

After we split up j and k in the denominator of (1.7) using Cauchy's integral formula, we wish to further reduce the computational cost by taking advantage of the "low rank" nature of $\widehat{\chi}^0$ using the same idea as density fitting (DF). However, the DF approximation cannot be used in our case for two reasons. The first is that the DF itself takes $\mathcal{O}(N^4)$ operations, which destroys the cubic scaling. The second problem is that j and k are coupled in the coefficients of the density fitting method. As long as j and k remain coupled, $\widehat{\chi}^0$ cannot be constructed in $\mathcal{O}(N^3)$. Solutions to both of these problems are provided by the interpolative separable density fitting (ISDF) method [16]. The ISDF is capable of computing a decomposition of $\psi_j^* \psi_k$ similar to that of DF except that the j and k dependence in the coefficients are separated. Additionally, the decomposition can be performed in $\mathcal{O}(N^3)$ due to the use of a random projection in the method. Our use of the ISDF also reduces the memory cost of our algorithm to $\mathcal{O}(nN_{\text{aux}})$ compared to $\mathcal{O}(N_{\text{aux}}^3)$ for the traditional resolution of identity approach.

Let us also mention the recent work [14], where a related problem of phonon calculation is approached from the point of view of the Sternheimer equations to represent $\widehat{\chi}$ acting on functions. Normally, for phonon calculations, $\mathcal{O}(N^2)$ Sternheimer equations would need to be solved in order to compute $\widehat{\chi}^0\widehat{v}$. However, by use of interpolative separable density fitting and a polynomial interpolation, they reduce the number of Sternheimer equations to $\mathcal{O}(N)$, which enables a cubic scaling algorithm.

The rest of the paper is organized as follows. In Section 2, we reformulate the expression (1.6) into a new, approximate form. This new expression, characterized by (2.24), is used as the basis for our cubic scaling algorithm. Section 3 begins with a summary of our algorithm followed by a detailed description of each step. In Section 4, we run numerical tests to compare the scaling of our algorithm against the quartic scaling resolution of identity algorithm.

2. Derivation of the method

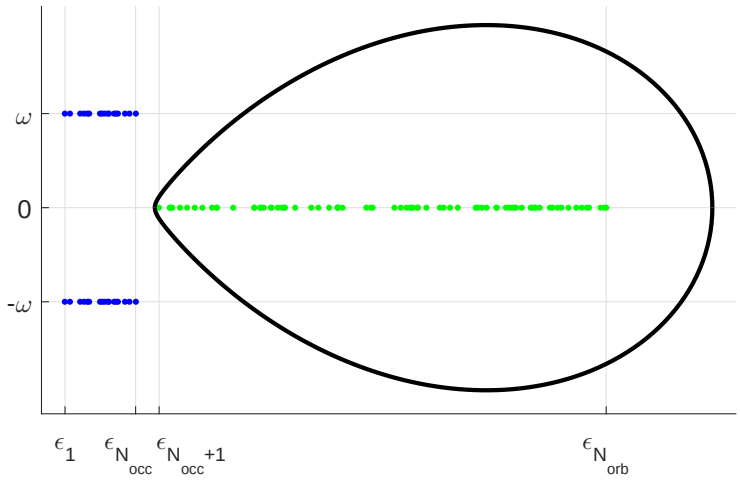


FIGURE 1. An example of the contour $\mathscr{C}$ (see derivation in Section 3.2). The blue points represent $\{\varepsilon_j \pm i\omega\}_{j=1}^{N_{\text{occ}}}$ (for a particular choice of ω), and the green points represent $\{\varepsilon_k\}_{k=N_{\text{occ}}+1}^{N_{\text{orb}}}$.

In this section, we reformulate the RPA correlation energy (1.6) which will be used to construct our cubic scaling algorithm.

2.1. Contour integral representation.

The first key idea is to split up the dependence of j and k in the denominator of (1.7). This is accomplished through the use of Cauchy's integral formula. For a given ω, let $\mathscr{C}$ be a closed contour in the complex plane oriented in the clockwise direction which encloses ε_ℓ for all ℓ which are unoccupied, and does not enclose $\varepsilon_\ell \pm i\omega$ for any ℓ that is occupied. An example of such a contour is shown in Figure 1. While in principle the contour can be chosen differently for different ω, later in Section 3.2, we will make the restriction that $\mathscr{C}$ is the same for all ω for the purpose of reducing computational costs. Using Cauchy's integral formula, we may write the coefficient in (1.7) as

$$\begin{aligned}\frac{f_j - f_k}{\varepsilon_j - \varepsilon_k - i\omega} &= \frac{1}{\varepsilon_j - \varepsilon_k - i\omega} \cdot \frac{1}{2\pi i} \int_{\mathscr{C}} \left(\frac{1}{\lambda - \varepsilon_j + i\omega} - \frac{1}{\lambda - \varepsilon_k} \right) \mathrm{d}\lambda \\ &= \frac{1}{2\pi i} \int_{\mathscr{C}} \frac{1}{(\lambda - \varepsilon_j + i\omega)(\lambda - \varepsilon_k)} \, \mathrm{d}\lambda. \end{aligned} \tag{2.1}$$

This can then be used to obtain an expression for $\widehat{\chi}^0$ which can be computed in cubic time:

$$\left\langle x \middle| \widehat{\chi}^0(i\omega) \middle| y \right\rangle = \sum_{jk} \frac{(f_j - f_k)\psi_j^*(x)\psi_k(x)\psi_k^*(y)\psi_j(y)}{\varepsilon_j - \varepsilon_k - i\omega},$$

$$
\begin{aligned}
&= \sum_{j}^{\text{occ}} \sum_{k}^{\text{vir}} \frac{\psi_j^*(x)\psi_k(x)\psi_k^*(y)\psi_j(y)}{\varepsilon_j - \varepsilon_k - i\omega} + \text{c.c.} \\
&= \frac{1}{2\pi i} \int_{\mathscr{C}} \left(\sum_{j}^{\text{occ}} \frac{\psi_j^*(x)\psi_j(y)}{\lambda - \varepsilon_j + i\omega} \right) \left(\sum_{k}^{\text{vir}} \frac{\psi_k(x)\psi_k^*(y)}{\lambda - \varepsilon_k} \right) \mathrm{d}\lambda + \text{c.c.},
\end{aligned} \tag{2.2}
$$

where c.c. is the complex conjugate. We show in Lemma 3.1 that the contour integral can be discretized with a number of quadrature points which is logarithmic in $(\varepsilon_{N_{\text{orb}}} - \varepsilon_{N_{\text{occ}}})/(\varepsilon_{N_{\text{occ}+1}} - \varepsilon_{N_{\text{occ}}})$.

Note that the formula (2.2) already provides a cubic scaling method for calculating $\chi^0(i\omega)$. In particular, ignoring logarithmic factors, $\chi^0(i\omega)$ can be calculated with cost $\mathcal{O}(N_{\text{orb}} n^2)$. However, the number of grid points n could be much larger than the number of orbitals, so to reduce the prefactor of the computational cost, we will write the problem into an auxiliary basis set instead of using the spatial grid points.

Moreover, reducing the rank of $\widehat{\chi}^0$ from $N_{\text{occ}} \cdot N_{\text{vir}}$ (its rank before spatial discretization) to the number of grid points n (its rank after spatial discretization, assuming $n < N_{\text{occ}} \cdot N_{\text{vir}}$) is really just a limitation placed on the operator by the particular discretization of the problem, rather than something inherent to the operator itself. The intuitive idea for the expected approximate low rank of $\widehat{\chi}^0$ is that $\widehat{\chi}^0$ contains $\mathcal{O}(N_{\text{orb}})$ information in its definition, and therefore its approximate rank should scale linearly with the number of *orbitals* used in the calculation, rather than the number of grid points. This motivates the use of ISDF, as recalled in the next subsection.

2.2. **Interpolative Separable Density Fitting.** We use the Interpolative Separable Density Fitting (ISDF) [15, 16] to further accelerate the computation. ISDF aims at a representation of the orbital pair functions as

$$
\psi_j^*(x)\psi_k(x) \approx \sum_{\mu=1}^{N_{\text{aux}}} \psi_j^*(x_\mu)\psi_k(x_\mu) P_\mu(x), \tag{2.3}
$$

where the $\{x_\mu\}$ and $\{P_\mu\}$ are chosen by the ISDF algorithm, which we will recall below for completeness of the presentation. Here N_{aux} denotes the number of auxiliary orbitals needed to represent the orbital pairs involved; it is empirically established that N_{aux} depends linearly on N_{orb} [15], which we will also further verify in our numerical examples. The representation (2.3) should be compared to the traditional density fitting which yields

$$
\psi_j^*(x)\psi_k(x) \approx \sum_{\mu=1}^{N_{\text{aux}}} C_{jk}^{\mu} P_\mu(x), \tag{2.4}
$$

where $\{P_\mu(x)\}$ are inputs to the DF algorithm and the coefficients C_{jk}^{μ} are determined via least square fitting (in the L^2 or Coulomb metric). In (2.3), the $\psi_j^*(x_\mu)\psi_k(x_\mu)$ factor is the coefficient for the basis function $P_\mu(x)$. The main difference is thus that the j and k dependence of the coefficients are cleanly separated in ISDF, but not in DF. This is important to achieve cubic scaling algorithm in our work. Furthermore, ISDF has some other advantages over DF: the time and memory cost of ISDF is cheaper, in particular, it requires only $\mathcal{O}(nN_{\text{aux}})$ memory and cubic scaling computational cost. In addition, the auxiliary functions are determined by the algorithm and do not have to be specified by the user.

Let us now describe the ISDF algorithm. The essential idea of the ISDF algorithm is to select important grid points $\{x_\mu\}$ via a randomized column selection algorithm. For the application to RPA correlation energy, we only need orbital pair functions of the type $\psi_j^*(x)\psi_k(x)$ where one of j or k is occupied and the other is unoccupied (see (2.2)). We can use this to our advantage by making a slight modification to the algorithm in [16], which would otherwise give an approximation for all N_{orb}^2 orbital pair functions. A version of ISDF which only calculates approximations for the orbital pair functions we are interested in is presented in Algorithm 1.

Compared with the original ISDF algorithm presented in [16], one technical difference is Step 5 in Algorithm 1. The reason for introducing $\mathbf{M}$, instead of just using M in the QRCP there, is that now we can take the basis functions $\{P_\mu\}$ to be real. This can be seen by switching j and k in (2.3) and taking the complex conjugate,

$$\psi_j^*(x)\psi_k(x) \approx \sum_{\mu=1}^{N_{\text{aux}}} \psi_j^*(x_\mu)\psi_k(x_\mu)P_\mu^*(x). \tag{2.9}$$

Note that *both* (2.3) and (2.9) are valid only because we included the conjugate of M in the QRCP. Now, we may average the two expressions to show that we may write $\psi_j^*(x)\psi_k(x)$ in terms of real basis functions,

$$\psi_j^*(x)\psi_k(x) \approx \sum_{\mu=1}^{N_{\text{aux}}} \psi_j^*(x_\mu)\psi_k(x_\mu)\,\text{Re}[P_\mu(x)]. \tag{2.10}$$

It turns out that taking the basis functions to be real considerably simplifies the expression for χ^0 that we will obtain. This leads to reduced computational effort as well as simpler code. For these reasons, we will always assume that the auxiliary basis functions from the ISDF are real.

2.3. **Representation of $\widehat{\chi}^0$ using interpolative separable density fitting.** Now we can use the ISDF to reduce the computational cost of the cubic scaling method for the RPA correlation energy. First, we approximate the $\widehat{\chi}^0$ operator by an operator $\widetilde{\chi}^0$ by using the ISDF approximation. For simplicity of notation, we define $C_j^\mu = \psi_j(x_\mu)$.

$$\begin{aligned}
&\left\langle x \middle| \widehat{\chi}^0(i\omega) \middle| y \right\rangle = \\
&\quad = \frac{1}{2\pi i}\int_{\mathscr{C}} \left(\sum_j^{\text{occ}} \frac{\psi_j^*(x)\psi_j(y)}{\lambda-\varepsilon_j+i\omega}\right)\left(\sum_k^{\text{vir}} \frac{\psi_k(x)\psi_k^*(y)}{\lambda-\varepsilon_k}\right)\mathrm{d}\lambda \\
&\qquad\qquad + \frac{1}{2\pi i}\int_{\mathscr{C}} \left(\sum_j^{\text{occ}} \frac{\psi_j(x)\psi_j^*(y)}{\lambda-\varepsilon_j-i\omega}\right)\left(\sum_k^{\text{vir}} \frac{\psi_k^*(x)\psi_k(y)}{\lambda-\varepsilon_k}\right)\mathrm{d}\lambda \\
&\quad \approx \sum_{\mu\nu}\frac{1}{2\pi i}\int_{\mathscr{C}} \left[\left(\sum_j^{\text{occ}} \frac{\overline{C}_j^\mu C_j^\nu}{\lambda-\varepsilon_j+i\omega}\right)\left(\sum_k^{\text{vir}} \frac{C_k^\mu \overline{C}_k^\nu}{\lambda-\varepsilon_k}\right)\right. \\
&\qquad\qquad \left. + \left(\sum_j^{\text{occ}} \frac{C_j^\mu \overline{C}_j^\nu}{\lambda-\varepsilon_j-i\omega}\right)\left(\sum_k^{\text{vir}} \frac{\overline{C}_k^\mu C_k^\nu}{\lambda-\varepsilon_k}\right)\right]\mathrm{d}\lambda\, P_\mu(x)P_\nu(y) \\
&\quad = \sum_{\mu\nu}\frac{1}{2\pi i}\int_{\mathscr{C}} \left[\mathbf{J}_{\mu\nu}(\lambda,\omega)\mathbf{K}_{\mu\nu}(\lambda) + \overline{\mathbf{J}_{\mu\nu}(\overline{\lambda},\omega)\mathbf{K}_{\mu\nu}(\overline{\lambda})}\right]\mathrm{d}\lambda\, P_\mu(x)P_\nu(y)
\end{aligned} \tag{2.11}$$

Algorithm 1 Interpolative Separable Density Fitting for RPA

Input: Orbitals $\{\psi_\ell\}_{\ell=1}^{N_{\text{orb}}}$, error tolerance tol.
Output: N_{aux}, $\{x_\mu\}$ and $\{P_\mu\}$ for $\mu = 1, ..., N_{\text{aux}}$.

1: Construct an $N_{\text{occ}} \times n$ matrix U^{occ} such that the jth row of U^{occ} is ψ_j. Likewise, construct an $N_{\text{vir}} \times n$ matrix U^{vir} using the virtual orbitals as the rows.

2: Multiply each of U^{occ} and U^{vir} on the left by a random diagonal matrix, and then take the discrete Fourier transform,

$$\widehat{U}^{\text{occ}}_\xi(x) = \sum_{\alpha=1}^{N_{\text{occ}}} e^{-i2\pi\alpha\xi/N_{\text{occ}}}\eta_\alpha U_\alpha(x),$$

$$\widehat{U}^{\text{vir}}_\xi(x) = \sum_{\alpha=1}^{N_{\text{vir}}} e^{-i2\pi\alpha\xi/N_{\text{vir}}}\gamma_\alpha U_\alpha(x), \tag{2.5}$$

where η_α and γ_α are random unit complex numbers.

3: Randomly choose $r_{\text{occ}} = c\sqrt{N_{\text{occ}}}$ rows of $\widehat{U}^{\text{occ}}$ and $r_{\text{vir}} = c\sqrt{N_{\text{vir}}}$ rows of $\widehat{U}^{\text{vir}}$ to create submatrices $\mathscr{U}^{\text{occ}}$ and $\mathscr{U}^{\text{vir}}$, respectively.

4: Construct an $r_{\text{occ}}r_{\text{vir}} \times n$ matrix M,

$$M_{st}(x) = \overline{\mathscr{U}}^{\text{occ}}_s(x)\mathscr{U}^{\text{vir}}_t(x), \qquad s = 1, ..., r_{\text{occ}}, \quad t = 1, ..., r_{\text{vir}}, \tag{2.6}$$

where (st) is viewed as the row index of M.

5: Find the QR factorization with column pivoting (QRCP) of the $2r_{\text{occ}}r_{\text{vir}} \times n$ matrix $\mathbf{M}$ formed by concatenating M with its complex conjugate,

$$QR = \begin{bmatrix} M \\ \text{conj}(M) \end{bmatrix} E = \mathbf{M}E, \tag{2.7}$$

where Q is unitary, R is upper triangular with diagonal entries in decreasing order, and E is a permutation matrix. In the case that M is real, we can just take $\mathbf{M} = M$.

6: Choose N_{aux} such that

$$R_{N_{\text{aux}},N_{\text{aux}}} \geq \text{tol} \cdot R_{1,1} > R_{N_{\text{aux}+1},N_{\text{aux}+1}}. \tag{2.8}$$

Then, we have $\mathbf{M} \approx (\mathbf{M}E)_{:,1:N_{\text{aux}}}P$, where we note that $\mathbf{M}E$ is a permutation of the columns of $\mathbf{M}$.

7: Calculate $P = R^{-1}_{1:N_{\text{aux}},1:N_{\text{aux}}}R_{1:N_{\text{aux}},:}E^T$, where MATLAB notation is used for the indexing. Then, the auxiliary basis functions $\{P_\mu\}_{\mu=1}^{N_{\text{aux}}}$ are the rows of P.

8: Finally, the points $\{x_\mu\}_{\mu=1}^{N_{\text{aux}}}$ are the grid points used in the first N_{aux} columns of $\mathbf{M}E$. We can be more specific as follows. First, to avoid a conflict in notation, label the grid points $\{y_\ell\}_{\ell=1}^{n}$. That is, whenever we considered ψ_j as a row vector, it was $\psi_j = [\psi_j(y_1), ..., \psi_j(y_n)]$. Next, since E is a permutation matrix, it defines a permutation σ. In particular, for a matrix A, the product AE is a column permuted version of A where the jth column of A has been sent to the $\sigma(j)$ column. Using this notation, we have $x_\mu = y_{\sigma^{-1}(\mu)}$ for $\mu = 1, ..., N_{\text{aux}}$.

$$= \sum_{\mu\nu} \chi^0_{\mu\nu}(i\omega)P_\mu(x)P_\nu(y)$$

$$= \langle x | \widetilde{\chi}^0(i\omega) | y \rangle, \tag{2.12}$$

where the last line defines notation of $\widetilde{\chi}^0$, and we have used the short hands

$$\mathbf{J}_{\mu\nu}(\lambda,\omega) = \sum_j^{\text{occ}} \frac{\overline{C}_j^\mu C_j^\nu}{\lambda - \varepsilon_j + i\omega}, \tag{2.13}$$

$$\mathbf{K}_{\mu\nu}(\lambda) = \sum_k^{\text{vir}} \frac{C_k^\mu \overline{C}_k^\nu}{\lambda - \varepsilon_k}, \tag{2.14}$$

$$\chi^0_{\mu\nu}(i\omega) = \frac{1}{2\pi i}\int_{\mathscr{C}} \left[\mathbf{J}_{\mu\nu}(\lambda,\omega)\mathbf{K}_{\mu\nu}(\lambda) + \overline{\mathbf{J}_{\mu\nu}(\overline{\lambda},\omega)\mathbf{K}_{\mu\nu}(\overline{\lambda})}\right] \mathrm{d}\lambda. \tag{2.15}$$

We note that in (2.11), the separability of the ISDF coefficients into the j and k components is crucial. Without this separability (e.g., if a conventional DF was used) we would not be able to calculate $\widetilde{\chi}^0$ in cubic time since the sums over j and k would not decouple.

Before continuing, we state explicitly our notation related to $\widehat{\chi}^0$ for the sake of clarity:

- $\widehat{\chi}^0$ – the original operator.
- $\widetilde{\widehat{\chi}}^0$ – the approximation to $\widehat{\chi}^0$ that is obtained by applying the ISDF approximation.
- χ^0 – defined by (2.15). In (2.23), we will show that it is $\widetilde{\widehat{\chi}}^0$ in the dual basis to the auxiliary basis functions.
- The argument $i\omega$ is often suppressed below for simplicity of notation.

2.4. **RPA correlation energy with auxiliary basis functions.** We now return our attention to (1.6). We want to rewrite this expression using the approximate basis $\{|P_\mu\rangle\}$. However, since we are considering a problem with periodic boundary conditions, it will be advantageous for us to instead consider the basis $\left\{\mathbb{1}, \{|\mathbf{P}_\mu\rangle\}_{\mu=1}^{N_{\text{aux}}}\right\}$, where $|\mathbf{P}_\mu\rangle$ is the shift of $|P_\mu\rangle$ so that it has zero mean, and $\mathbb{1}$ is the constant function with norm 1. The reasons for this change are explained further in Section 3.1. Since we wish to work with an orthonormal set, we introduce the orthonormalized basis functions

$$|\mathbf{B}_\mu\rangle = |\mathbf{P}_\nu\rangle\, S^{-1/2}_{\nu\mu}, \tag{2.16}$$

where $S_{\mu\nu} = \langle \mathbf{P}_\mu | \mathbf{P}_\nu \rangle$. Then we can take the trace with respect to the orthonormal set

$$\left\{\mathbb{1}, \{|\mathbf{B}_\mu\rangle\}_{\mu=1}^{N_{\text{aux}}}\right\}.$$

In the following, we consider $\widehat{\chi}^0$ and $\widehat{v}$ to be linear operators on an n-dimensional space (i.e., the discretization of the operators with respect to our spatial grid) for the purposes of justifying our steps.

$$\begin{aligned} \operatorname{tr}\left[\ln(I - \widehat{\chi}^0\widehat{v})\right] &\approx \operatorname{tr}\left[\ln(I - \widetilde{\widehat{\chi}}^0\widehat{v})\right] \\ &\approx \sum_\beta \langle \mathbf{B}_\beta | \ln(I - \widetilde{\widehat{\chi}}^0\widehat{v}) | \mathbf{B}_\beta \rangle + \langle \mathbb{1} | \ln(I - \widetilde{\widehat{\chi}}^0\widehat{v}) | \mathbb{1} \rangle. \end{aligned} \tag{2.17}$$

The first line is justified as follows. First, we note that $\widehat{v}$ is bounded on finite dimensional spaces. Therefore, for $\widetilde{\widehat{\chi}}^0$ close enough to $\widehat{\chi}^0$, we have $\|\widehat{\chi}^0\widehat{v} - \widetilde{\widehat{\chi}}^0\widehat{v}\| \ll 1$. Thus, assuming that $I - \widehat{\chi}^0\widehat{v}$ is invertible and $\ln(I - \widehat{\chi}^0\widehat{v})$ makes sense, the following linear approximation is justified

$$\operatorname{tr}\left[\ln(I - \widehat{\chi}^0\widehat{v}) - \ln(I - \widetilde{\widehat{\chi}}^0\widehat{v})\right] \approx -(I - \widehat{\chi}^0\widehat{v})^{-1} : (\widehat{\chi}^0\widehat{v} - \widetilde{\widehat{\chi}}^0\widehat{v}), \tag{2.18}$$

where $A : B$ means $\sum_i \sum_j A_{ij}B_{ij}$, i.e., the sum of the entries of the entrywise product. Before continuing our derivation, we first give a series expression for $\ln(I - \widetilde{\widehat{\chi}}^0\widehat{v})$. To justify the expansion, we assume that the eigenvalues of $\widehat{\chi}^0\widehat{v}$ are contained in the left half complex plane. Then

for $\widetilde{\chi}^0$ sufficiently close to $\widehat{\chi}^0$, there exists $c > 1$ such that the following expansion holds.

$$\begin{aligned}
\ln(I - \widetilde{\chi}^0 \widehat{v}) &= \ln\left[cI - \left((c-1)I + \widetilde{\chi}^0 \widehat{v}\right)\right] \\
&= \ln(c) I + \ln\left[I - \frac{1}{c}\left((c-1)I + \widetilde{\chi}^0 \widehat{v}\right)\right] \\
&= \ln(c) I - \sum_{\ell=1}^{\infty} \frac{[(c-1)I + \widetilde{\chi}^0 \widehat{v}]^\ell}{\ell c^\ell} \\
&= \ln(c) I - \sum_{\ell=1}^{\infty} \frac{1}{\ell c^\ell} \sum_{p=0}^{\ell} \binom{\ell}{p} (c-1)^{\ell-p} (\widetilde{\chi}^0 \widehat{v})^p. \qquad (2.19)
\end{aligned}$$

The first thing to note about (2.19) is that the nullspace of $\widehat{v}$ is contained in the nullspace of $\ln(I - \widetilde{\chi}^0 \widehat{v})$. Since $\widehat{v}$ is the periodic Coulomb operator, the constant function $|\mathbb{1}\rangle$ is in its nullspace. Therefore, we can drop the final term in (2.17), since it is zero. Next, we note that the infinite sum in (2.19) is absolutely convergent, so we can continue (2.17) by applying the above expansion and taking the trace through the sum.

$$\begin{aligned}
\operatorname{tr}\left[\ln(I - \widehat{\chi}^0 \widehat{v})\right] &\approx \sum_{\beta} \left\langle \mathbf{B}_\beta \middle| \ln(I - \widetilde{\chi}^0 \widehat{v}) \middle| \mathbf{B}_\beta \right\rangle \\
&= \sum_{\beta} \left\langle \mathbf{B}_\beta \middle| \ln(c) I \middle| \mathbf{B}_\beta \right\rangle - \sum_{\ell=1}^{\infty} \frac{1}{\ell c^\ell} \sum_{k=0}^{\ell} \binom{\ell}{p} (c-1)^{\ell-p} \sum_{\beta} \left\langle \mathbf{B}_\beta \middle| (\widetilde{\chi}^0 \widehat{v})^p \middle| \mathbf{B}_\beta \right\rangle. \qquad (2.20)
\end{aligned}$$

Next, we write the terms $\sum_\beta \left\langle \mathbf{B}_\beta \middle| (\widetilde{\chi}^0 \widehat{v})^p \middle| \mathbf{B}_\beta \right\rangle$ into a more computationally efficient, but approximate, representation. For simplicity, we only give the derivation for $p = 1$, but the others are similar.

$$\begin{aligned}
\sum_{\beta} \left\langle \mathbf{B}_\beta \middle| \widetilde{\chi}^0 \widehat{v} \middle| \mathbf{B}_\beta \right\rangle &= \sum_{\beta} \left(\sum_{\alpha} S^{-1/2}_{\beta\alpha} \langle \mathbf{P}_\alpha | \right) \widetilde{\chi}^0 \widehat{v} \left(\sum_{\mu} \left| \mathbf{P}_\mu \right\rangle S^{-1/2}_{\mu\beta} \right) \\
&\approx \sum_{\alpha\beta\gamma\mu\nu} S^{-1/2}_{\beta\alpha} \left\langle \mathbf{P}_\alpha \middle| \widetilde{\chi}^0 \middle| \mathbf{P}_\gamma \right\rangle S^{-1}_{\gamma\nu} \left\langle \mathbf{P}_\nu \middle| \widehat{v} \middle| \mathbf{P}_\mu \right\rangle S^{-1/2}_{\mu\beta} \\
&= \sum_{\alpha\gamma\mu\nu} S^{-1}_{\mu\alpha} \left\langle \mathbf{P}_\alpha \middle| \widetilde{\chi}^0 \middle| \mathbf{P}_\gamma \right\rangle S^{-1}_{\gamma\nu} \left\langle \mathbf{P}_\nu \middle| \widehat{v} \middle| \mathbf{P}_\mu \right\rangle \\
&= \sum_{\mu\nu} \left(\sum_{\alpha} S^{-1}_{\mu\alpha} \langle \mathbf{P}_\alpha | \right) \widetilde{\chi}^0 \left(\sum_{\gamma} \left| \mathbf{P}_\gamma \right\rangle S^{-1}_{\gamma\nu} \right) \left\langle \mathbf{P}_\nu \middle| \widehat{v} \middle| \mathbf{P}_\mu \right\rangle \\
&= \sum_{\mu\nu} \left\langle \widetilde{\mathbf{P}}_\mu \middle| \widetilde{\chi}^0 \middle| \widetilde{\mathbf{P}}_\nu \right\rangle \left\langle \mathbf{P}_\nu \middle| \widehat{v} \middle| \mathbf{P}_\mu \right\rangle, \qquad (2.21)
\end{aligned}$$

where

$$\left\langle \widetilde{\mathbf{P}}_\mu \right| = \sum_{\alpha} S^{-1}_{\mu\alpha} \langle \mathbf{P}_\alpha |, \qquad (2.22)$$

is the dual basis to $\{|\mathbf{P}_\mu\rangle\}$. Before commenting on the significance of this new representation, let's write $\left\langle \widetilde{\mathbf{P}}_\mu \middle| \widetilde{\chi}^0 \middle| \widetilde{\mathbf{P}}_\nu \right\rangle$ into a simpler form.

$$\begin{aligned}
\left\langle \widetilde{\mathbf{P}}_\mu \middle| \widetilde{\chi}^0 \middle| \widetilde{\mathbf{P}}_\nu \right\rangle &= \sum_{\alpha\gamma} S^{-1}_{\mu\alpha} \left\langle \mathbf{P}_\alpha \middle| \widetilde{\chi}^0 \middle| \mathbf{P}_\gamma \right\rangle S^{-1}_{\gamma\nu} \\
&= \sum_{\alpha\gamma} S^{-1}_{\mu\alpha} \int\int \langle \mathbf{P}_\alpha | x \rangle \left\langle x \middle| \widetilde{\chi}^0 \middle| y \right\rangle \langle y | \mathbf{P}_\gamma \rangle \, \mathrm{d}x \, \mathrm{d}y \, S^{-1}_{\gamma\nu}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{\alpha\gamma\mu'\nu'} S^{-1}_{\mu\alpha} \int\int \mathbf{P}_\alpha(x)\mathbf{P}_{\mu'}(x)\chi^0_{\mu'\nu'}(i\omega)\mathbf{P}_{\nu'}(y)\mathbf{P}_\gamma(y)\,\mathrm{d}x\,\mathrm{d}y\; S^{-1}_{\gamma\nu} \\
&= \sum_{\alpha\gamma\mu'\nu'} S^{-1}_{\mu\alpha} S_{\alpha\mu'}\chi^0_{\mu'\nu'}(i\omega) S_{\nu'\gamma} S^{-1}_{\gamma\nu} \\
&= \chi^0_{\mu\nu}(i\omega), \qquad (2.23)
\end{aligned}
$$

where $\chi^0_{\mu\nu}(i\omega)$ is defined in (2.15), and can therefore be computed in cubic time. Let us define $v_{\mu\nu} = \langle \mathbf{P}_\mu | \widehat{v} | \mathbf{P}_\nu \rangle$. Then the right hand side of (2.21) reads $\mathrm{tr}[\chi^0(i\omega)v]$, where the right hand side is just the standard trace of the product of the two matrices, $\mathrm{tr}[AB] = \sum_\mu \sum_\nu A_{\mu\nu}B_{\nu\mu}$. Plugging this into (2.20), we obtain our final desired approximation,

$$
\mathrm{tr}\left[\ln(1-\widehat{\chi}^0(i\omega)\widehat{v}) + \widehat{\chi}^0(i\omega)\widehat{v}\right] \approx \mathrm{tr}\left[\ln(1-\chi^0(i\omega)v) + \chi^0(i\omega)v\right]. \qquad (2.24)
$$

3. Algorithm

In this section, we present the cubic scaling algorithm for the calculation of the RPA correlation energy. We present a brief overview in Algorithm 2 before going into the details of each step. The computational effort is stated to the right of each step.

Algorithm 2 Cubic scaling calculation of the RPA correlation energy

Input: Kohn-Sham orbitals $\{\psi_k\}$ and corresponding energies $\{\varepsilon_k\}$.
Output: E_c^{RPA}

1: Use $\{\psi_k\}_{k=1}^{N_{\mathrm{orb}}}$ as the input to ISDF to obtain $\{x_\mu\}_{k=1}^{N_{\mathrm{aux}}}$ and $\{P_\mu\}_{k=1}^{N_{\mathrm{aux}}}$. $\mathcal{O}(nN_{\mathrm{orb}}^2)$
2: Compute the matrix $v_{\mu,\nu} = \langle \mathbf{P}_\mu | \widehat{v} | \mathbf{P}_\nu \rangle$. $\mathcal{O}(nN_{\mathrm{aux}}^2)$
3: For each quadrature point ω_m:
   a) Compute $\chi^0_{\mu,\nu}(i\omega_m) = \dfrac{1}{2\pi i}\displaystyle\int_{\mathscr{C}} \left[\mathbf{J}_{\mu,\nu}(\lambda,\omega_m)\mathbf{K}_{\mu,\nu}(\lambda) + \overline{\mathbf{J}_{\mu,\nu}(\overline{\lambda},\omega_m)\mathbf{K}_{\mu,\nu}(\overline{\lambda})}\right]\mathrm{d}\lambda$. $\mathcal{O}(N_{\mathrm{orb}}N_{\mathrm{aux}}^2)$
   b) Compute $\frac{1}{2\pi}\mathrm{tr}\left[\ln(1-\chi^0(i\omega_m)v) + \chi^0(i\omega_m)v\right]$. $\mathcal{O}(N_{\mathrm{aux}}^3)$
4: Calculate $E_c^{\mathrm{RPA}} = \frac{1}{2\pi}\displaystyle\int_0^\infty \mathrm{tr}\left[\ln(1-\chi^0(i\omega)v) + \chi^0(i\omega)v\right]\mathrm{d}\omega$ via numerical quadrature.

Without using the ISDF, the algorithm would be essentially exactly the same, but with Step 1 removed and Step 3a replaced by (2.2). Except, in the computational costs, each N_{aux} would be replaced by n. So clearly, if N_{aux} is much less than n, then including the ISDF can substantially speed up the algorithm.

3.1. Step 2 – Computing the Coulomb matrix.

We note that v can be efficiently computed by noticing that

$$
\begin{aligned}
\langle \mathbf{P}_\mu | \widehat{v} | \mathbf{P}_\nu \rangle &= \int\int \mathbf{P}_\mu(x)\mathbf{P}_\nu(y)\,v(x,y)\,\mathrm{d}x\,\mathrm{d}y \\
&= \int \mathbf{P}_\mu(x)\phi_\nu(x)\,\mathrm{d}x, \qquad (3.1)
\end{aligned}
$$

where

$$
\phi_\nu(x) = \int \mathbf{P}_\nu(y)\,v(x,y)\,\mathrm{d}y. \qquad (3.2)
$$

Therefore, ϕ_ν solves the Poisson equation, $-\Delta\phi_\nu = 4\pi\mathbf{P}_\nu$ with periodic boundary conditions. This equation can be efficiently solved using the fast Fourier transform. Therefore, the functions ϕ_ν can be precalculated at a total cost of $\mathcal{O}(N_{\text{aux}} n \log n)$. Then the quadrature for (3.1) is straightforward.

We have glossed over a couple details here. First, the Poisson equation with periodic boundary conditions is not solvable unless $\mathbf{P}_\nu$ has an average value of 0. This is of course true by our construction of $\mathbf{P}_\nu$, and this is the reason for the use of the $\{|\mathbf{P}_\mu\rangle\}$ basis rather than the $\{|P_\mu\rangle\}$ basis when calculating the trace in (2.17). The second detail we've skipped is that the solution ϕ_ν is not unique as we can add any constant and get another solution. However, it turns out that adding a constant to ϕ_ν does not change the integral in (3.1) since $\mathbf{P}_\mu(x)$ has mean 0. So, this problem is also avoided by the use of the $\{|\mathbf{P}_\mu\rangle\}$ basis rather than the $\{|P_\mu\rangle\}$ basis.

3.2. **Step 3 – Quadrature rule for the contour integral.** Before discussing our proposed quadrature rule, let us discuss the symmetry of (2.11). For notational purposes, define

$$I_{\mu,\nu}(\lambda,\omega) = \frac{1}{2\pi i}\left[\mathbf{J}_{\mu,\nu}(\lambda,\omega)\mathbf{K}_{\mu,\nu}(\lambda) + \overline{\mathbf{J}_{\mu,\nu}(\overline{\lambda},\omega)\mathbf{K}_{\mu,\nu}(\overline{\lambda})}\right]. \tag{3.3}$$

It is straightforward to show the following symmetry across the real line,

$$\overline{\mathbf{K}_{\mu,\nu}(\overline{\lambda})} = \mathbf{K}_{\nu,\mu}(\lambda), \tag{3.4}$$

$$\operatorname{Re}\left[I_{\mu,\nu}(\overline{\lambda},\omega)\right] = -\operatorname{Re}\left[I_{\mu,\nu}(\lambda,\omega)\right], \tag{3.5}$$

$$\operatorname{Im}\left[I_{\mu,\nu}(\overline{\lambda},\omega)\right] = \operatorname{Im}\left[I_{\mu,\nu}(\lambda,\omega)\right]. \tag{3.6}$$

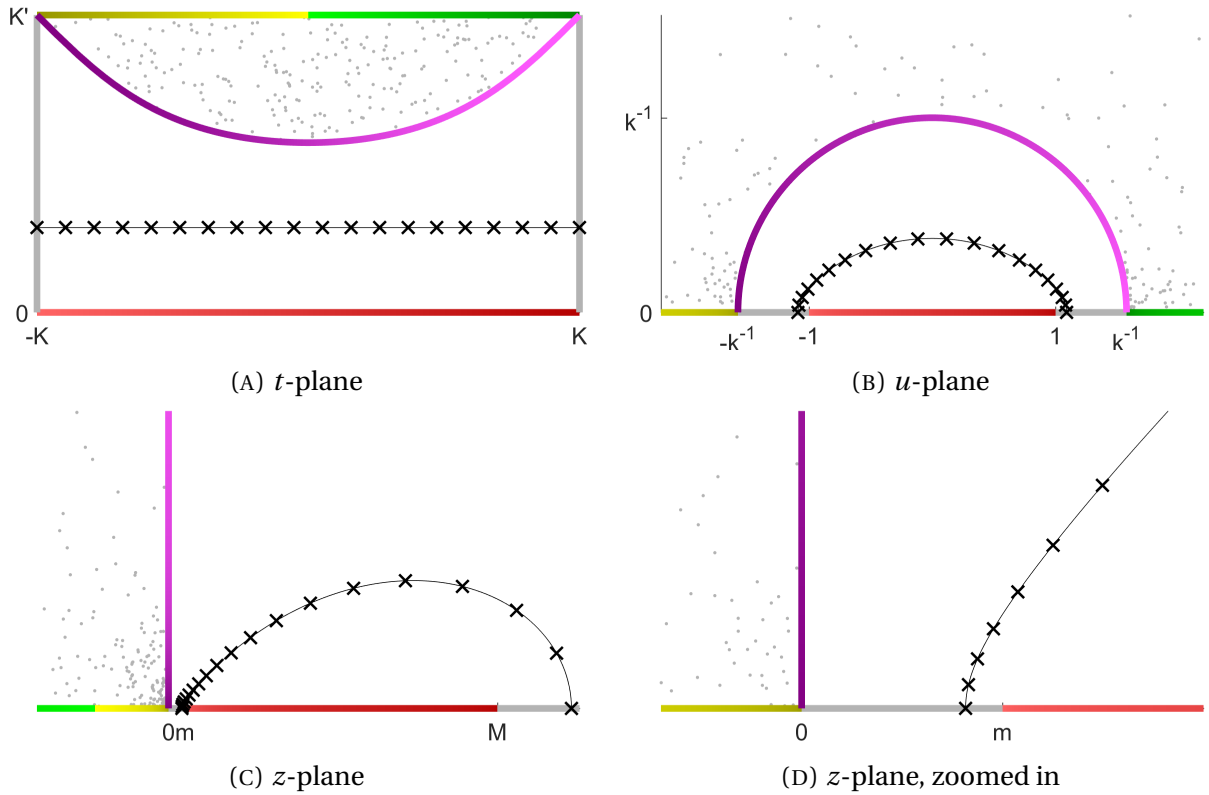
We wish to calculate

$$\chi^0_{\mu\nu}(i\omega) = \int_{\mathscr{C}} I_{\mu,\nu}(\lambda,\omega)\,\mathrm{d}\lambda, \tag{3.7}$$

where $\mathscr{C}$ is oriented clockwise and encloses $\{\varepsilon_k\}_{k=N_{\text{occ}}+1}^{N_{\text{orb}}}$ while enclosing none of $\{\varepsilon_j \pm i\omega\}_{j=1}^{N_{\text{occ}}}$. An example of such a contour is given in Figure 1. In order to use the symmetry about the real axis, we will enforce our contour to be symmetric about the real axis. Define $\mathscr{C}_{\text{upper}}$ and $\mathscr{C}_{\text{lower}}$ to be the parts of the contour in the upper and lower half plane. Then using (3.5) and (3.6), we have

$$\begin{aligned}
\chi^0_{\mu\nu}(i\omega) &= \int_{\mathscr{C}_{\text{upper}}} I_{\mu,\nu}(\lambda,\omega)\,\mathrm{d}\lambda + \int_{\mathscr{C}_{\text{lower}}} I_{\mu,\nu}(\lambda,\omega)\,\mathrm{d}\lambda \\
&= \int_{\mathscr{C}_{\text{upper}}} I_{\mu,\nu}(\lambda,\omega)\,\mathrm{d}\lambda - \int_{\mathscr{C}_{\text{upper}}} I_{\mu,\nu}(\overline{\lambda},\omega)\,\mathrm{d}\lambda \\
&= 2\operatorname{Re}\int_{\mathscr{C}_{\text{upper}}} I_{\mu,\nu}(\lambda,\omega)\,\mathrm{d}\lambda. \qquad (3.8)
\end{aligned}$$

We construct a quadrature rule for Step 3a by using similar ideas as in [8, 13]. For simplicity, let us assume that $\varepsilon_{N_{\text{occ}}} = 0$. To account for the fact that it is not, we will just have to shift the resulting quadrature points by $\varepsilon_{N_{\text{occ}}}$. For both simplicity of notation and to follow [8] more closely, let us define $m = \varepsilon_{N_{\text{occ}}+1} - \varepsilon_{N_{\text{occ}}}$ to be the energy gap and $M = \varepsilon_{N_{\text{orb}}} - \varepsilon_{N_{\text{occ}}}$. Then we map the rectangle with vertices $\pm K$ and $\pm K + iK'$, where K and K' are the complete elliptic



(A) t-plane (B) u-plane

(C) z-plane (D) z-plane, zoomed in

FIGURE 2. These figures show the transformations given in (3.11) and (3.12). Coloring is used to help show what is mapped where. The gray dots were distributed randomly in the region bounded by the purple, yellow, and green curves of the t-plane to show that this region maps to the left half z-plane. The red line contains the singularities we wish to encircle. The purple line and the regions with gray dots contain the singularities we wish to avoid.

integrals [1]

$$K(k) = \int_0^1 \frac{1}{\sqrt{(1-t^2)(1-k^2t^2)}}\,\mathrm{d}t, \tag{3.9}$$

$$K'(k) = K(1-k^2), \tag{3.10}$$

to the upper half plane via two consecutive transformations $t \mapsto u \mapsto z$ (see Figure 2).

$$u = \mathrm{sn}(t) = \mathrm{sn}(t|k^2), \qquad k = \frac{\sqrt{M/m}-1}{\sqrt{M/m}+1}, \tag{3.11}$$

$$z = \sqrt{mM}\left(\frac{k^{-1}+u}{k^{-1}-u}\right), \tag{3.12}$$

where $\mathrm{sn}(t|k^2)$ is the Jacobi elliptic function. The values of K and K' can be found, e.g., via the ellipkkp function in the Schwarz-Christoffel Toolbox for MATLAB [5]. The Jacobi elliptic functions $\mathrm{sn}(t)$, $\mathrm{cn}(t)$, and $\mathrm{dn}(t)$ are implemented in the ellipjc function in the same toolbox. One must be careful when using such functions as there are a few different common conventions for

how to parameterize the Jacobi elliptic functions. We have been using the parameter k while the Schwarz-Christoffel Toolbox uses the parameter $L = -\ln(k)/\pi$.

The idea for the quadrature rule is as follows. We ultimately wish to construct a quadrature rule in the z-plane, but since the function we are integrating is periodic and analytic, one can show via some numerical analysis [4, Section 4.6.5] that we can construct a geometrically convergent quadrature rule by using the trapezoid rule in the t-plane. Essentially, the idea is to map the z-plane to a periodic rectangle, apply the trapezoid rule in the periodic rectangle where it is known to converge geometrically, and then map the quadrature points in the t-plane back to the z-plane to obtain our desire quadrature rule. The trapezoid rule has the added bonus of having a nice nesting property of the quadrature points, so that we can create an adaptive quadrature rule.

The numerical analysis tells us that the rate of convergence will be greatest when the quadrature path in the t-plane is as far away as possible from all singularities of the function we are integrating. In order to find the singularities in the t-plane, let's first examine them in the z-plane. Due to symmetry, we will only consider the contour and singularities in the upper half plane. In our case, the function we are integrating is given by (2.11). For a given $\omega \geq 0$, its singularities (in the upper half plane) are $\{\varepsilon_j + i\omega\}_{j=1}^{N_{\text{occ}}}$ and $\{\varepsilon_k\}_{k=N_{\text{occ}}+1}^{N_{\text{orb}}}$. We first notice that the singularities we wish to avoid depend on ω. There is nothing inherently difficult about this, and we could construct a different quadrature rule for each ω. However, in order to save on computation, we want the quadrature rule to remain the same for each ω. This way, $\mathbf{K}(\lambda)$ does not need to be recalculated for each ω. Therefore, when we construct our quadrature rule, we wish to avoid all such singularities for $\omega \geq 0$. Since we have assumed that $\varepsilon_{N_{\text{occ}}} = 0$, this implies that $\{\varepsilon_j + i\omega\}_{\omega\geq 0, j=1,\ldots,N_{\text{occ}}}$ is contained in the left half z-plane. Therefore, it is sufficient for us to say that we wish to avoid the entire left half z-plane.

However, when we construct the quadrature rule, we are concerned with the singularities in the t-plane. By inverting the above mappings, we can see in Figure 2 that the left half z-plane is mapped to the region bounded by the yellow, green, and purple curves in the t-plane. So, it is sufficient for us to avoid this region. Next, we note that the rest of the singularities in the z-plane are contained in the red line. This line is mapped to the bottom edge of the rectangle in the t-plane. Finally, we wish for our contour in the z-plane to encircle the red line in the clockwise direction. This is achieved by taking a contour in the t-plane which goes from the left side of the rectangle to the right side. It is now clear how to maximize the distance between the singularities and the contour in the t-plane. We must draw our contour in the t-plane halfway between the bottom of the rectangle and the lowest point of the purple curve. This is demonstrated by a black horizontal line with X's in Figure 2a. We apply the trapezoid rule on this line. The line can be mapped back to the z-plane to create a quadrature rule as shown in Figure 2c.

Rather than now going into the rigorous details of the above argument, we will simply state the results. The details and proofs are deferred to the Appendix. First, the algorithm for Step 3a is summarized in Algorithm 3.

Algorithm 3 Step 3a – Quadrature rule for contour integral

1: Define $m = \varepsilon_{N_{\text{occ}}+1} - \varepsilon_{N_{\text{occ}}}$ and $M = \varepsilon_{N_{\text{orb}}} - \varepsilon_{N_{\text{occ}}}$.
2: Compute $k = \frac{\sqrt{M/m}-1}{\sqrt{M/m}+1}$.
3: Compute $I = \frac{1}{2}\int_0^{k^{-1}} \frac{\mathrm{d}s}{\sqrt{(1+s^2)(1+k^2s^2)}}$ via the midpoint rule with mesh size $h < 1/100$.
4: Define

$$\lambda(t) = \sqrt{mM}\left(\frac{k^{-1}+\operatorname{sn}(t)}{k^{-1}-\operatorname{sn}(t)}\right) + \varepsilon_{N_{\text{occ}}}, \tag{3.13}$$

as a shift of $z(t)$ to account for the fact that $\varepsilon_{N_{\text{occ}}}$ is typically not 0. The quadrature rule is then found by applying the trapezoid rule (in the variable t) to

$$\chi^0_{\mu\nu}(i\omega) = 2\operatorname{Re}\int_{\mathscr{C}_{\text{upper}}} I_{\mu,\nu}(\lambda,\omega)\,\mathrm{d}\lambda = 2\operatorname{Re}\int_{-K+iI}^{K+iI} I_{\mu,\nu}(\lambda(t),\omega)\frac{2k^{-1}\sqrt{mM}}{(k^{-1}-\operatorname{sn}(t))^2}\operatorname{cn}(t)\operatorname{dn}(t)\,\mathrm{d}t, \tag{3.14}$$

where the contour in the t plane is the horizontal line connecting $-K+iI$ and $K+iI$.
5: Double the number of quadrature points (via the nesting property of the trapezoid rule) until suitable convergence is achieved.

Next, we state the convergence rate of the proposed quadrature rule, whose proof may be found in the Appendix.

Lemma 3.1. *Let N_λ denote the number of quadrature points used to discretize* (3.14) *via the trapezoid rule. Then, for any $M/m > 1$, the error of the quadrature rule is*

$$\mathscr{O}\left(\exp\left(\frac{-\pi^2 N_\lambda}{2\log(M/m)+6}\right)\right). \tag{3.15}$$

Therefore, our quadrature rule for the contour integral converges geometrically in the number of quadrature points. Additionally, the number of points N_λ needed for convergence to a specific error tolerance increases only logarithmically as $(\varepsilon_{N_{\text{orb}}} - \varepsilon_{N_{\text{occ}}})/(\varepsilon_{N_{\text{occ}}+1} - \varepsilon_{N_{\text{occ}}}) \to \infty$.

Finally, we note that for Step 3b, rather than calculating the trace of the log, it is more efficient to calculate the log of the determinant,

$$\operatorname{tr}[\ln(I - \chi^0(i\omega)\,v) + \chi^0(i\omega)\,v] = \ln[\det(I - \chi^0(i\omega)\,v)] + \operatorname{tr}[\chi^0(i\omega)\,v]. \tag{3.16}$$

This expression is nice for practical computation since it avoids the necessity of calculating the matrix logarithm. Additionally, the determinant of a matrix may be calculated easily via an LU decomposition, for which there are readily available scalable codes.

3.3. **Step 4 – Quadrature rule for the frequency integral.** For the ω integral, we used the following Clenshaw–Curtis quadrature [3, Eq 3.2] on the semi-infinite interval $[0,\infty)$,

$$t_m = \frac{\pi m}{N+1}, \tag{3.17}$$

$$\omega_m = L\cot^2(t_m/2), \tag{3.18}$$

$$\int_0^\infty f(\omega)\,\mathrm{d}\omega \approx \sum_{m=1}^{N} W_m f(y_m), \tag{3.19}$$

where

$$W_m = \frac{4L\sin(t_m)}{(N+1)(1-\cos(t_m))^2}\sum_{j=1}^{N}\frac{1}{j}\sin(jt_m)[1-\cos(j\pi)], \tag{3.20}$$

where L is a parameter that must be chosen (we used $L = 10$). The value of L can affect the number of grid points needed for convergence, but this dependence is not very sensitive. A necessary condition for the geometric convergence of this method is that $f(\omega) = \mathcal{O}(\omega^{3/2})$ as $\omega \to \infty$ [3]. This is guaranteed by the following lemma.

Lemma 3.2. $\operatorname{tr}\left[\ln(I-\chi^0(i\omega)v)+\chi^0(i\omega)v\right] = \mathcal{O}(\omega^{-2})$ *as* $\omega \to \infty$.

Proof. It is straightfoward to show that $|\chi^0_{\mu\nu}(i\omega)| < c_{\mu\nu}\omega^{-1}$, where $c_{\mu\nu}$ is independent of ω. This implies $\left\|\chi^0(i\omega)\right\|_F < C\omega^{-1}$. Then we have

$$\begin{aligned}\left\|\chi^0(i\omega)v\right\|_2 &\le \left\|\chi^0(i\omega)v\right\|_F \\ &\le \left\|\chi^0(i\omega)\right\|_F \|v\|_F \\ &\le C\|v\|_F\omega^{-1}. \end{aligned} \tag{3.21}$$

Therefore, for ω large enough, the eigenvalues of $\chi^0(i\omega)v$ are all less than 1 in magnitude. This justifies the Taylor series expansion in the following,

$$\begin{aligned}\left|\operatorname{tr}\left[\ln(I-\chi^0(i\omega)v)+\chi^0(i\omega)v\right]\right| &\le C\sqrt{N_{\text{aux}}}\left\|\ln(I-\chi^0(i\omega)v)+\chi_0 v\right\|_F \\ &= C\sqrt{N_{\text{aux}}}\left\|-\frac{1}{2}(\chi^0(i\omega)v)^2+\mathcal{O}(\chi^0(i\omega)v)^3\right\|_F \\ &\le C\sqrt{N_{\text{aux}}}\|v\|_F^2\omega^{-2}+\mathcal{O}(\omega^{-3}), \end{aligned} \tag{3.22}$$

where the constant C changes from line to line. □

The use of Clenshaw–Curtis allows a simple and fast converging adaptive quadrature rule since the points of the quadrature rule have a nice nesting property as seen by (3.17).

4. Numerical results

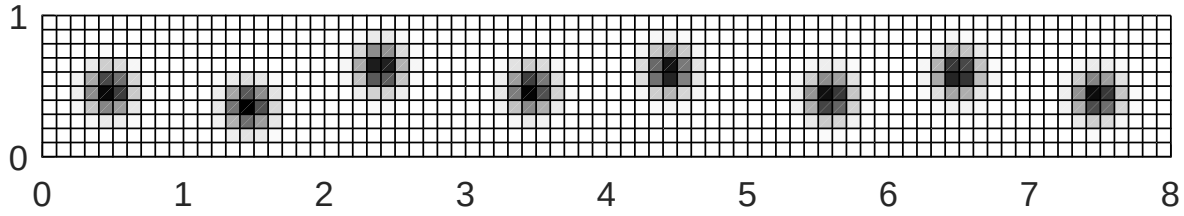


FIGURE 3. Example of our external potential with 8 wells. White is zero, darker is more negative.

Our numerical results use the following as the test problem. Our two dimensional spatial grid is $10\times 10 N_{\text{occ}}$ equally spaced points. First, we solve for the KS orbitals of the periodic system with Hamiltonian $H = T + V$, where $T = -\frac{1}{2}\Delta$ is the kinetic energy operator and V is the external potential. The external potential consists of N_{occ} Gaussian potential wells, the centers of which are randomly perturbed from the centers of their respective 10×10 box of grid points. Then the eigenvectors of H are used as the orbitals in the calculation of the RPA correlation energy.

4.1. **Convergence with respect to number of orbitals.** In these tests, we check the convergence of the RPA energy with respect to the number of orbitals used in the calculation. Figure 4a shows the results for a system with 4 electrons (and therefore a maximum of 400 orbitals). In Figure 4b, we have scaled the entire system up by a factor of 8 (maximum of 3200 orbitals) and run the same test. We can see by comparing the figures that the results are essentially identical. Both the percentage of orbitals needed in the calculation for a particular error value and the number of auxiliary basis functions (as a percentage of the number of grid points) needed for a particular error level in the ISDF step are nearly the same in the two cases.

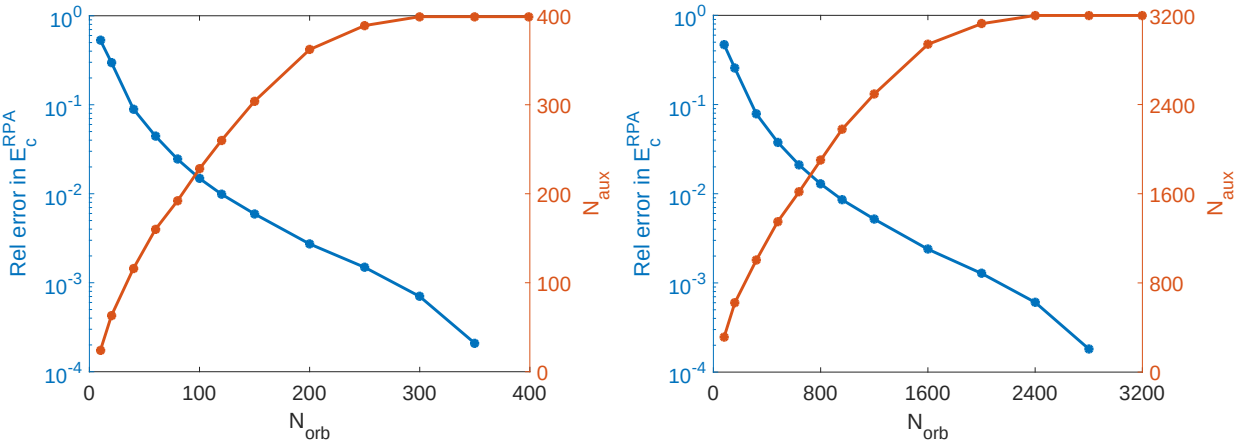
In general, one would want to work in a regime where N_{aux} is as small as possible while still achieving sufficient accuracy. Note that if $N_{\text{aux}} \approx n$, then there is no point in using ISDF and one would be better off using (2.2) instead. Figure 4 implies that ISDF is worth doing in the 10^{-2} relative error range, where we can see from Figure 4, the ISDF yields an N_{aux} significantly below n. However, this statement is highly dependent on the number of grid points. For example, in Figure 5, we see that the ISDF can be worthwhile all the way down to 10^{-4} relative error in the $n = 1600$ case. The reduction of the basis size from n to N_{aux} (as a proportion of the number of grid points) is amplified as the number of grid points is increased with all other variables fixed. This is because, as will be discussed shortly, N_{aux} depends on N_{orb}, not n.

In Figure 5, we fix an external potential and look at the behavior of the algorithm when the number of primal basis functions (grid points) is increased. Both tests are run with $N_{\text{occ}} = 4$. One is run with $n = 400$ grid points and the other with $n = 1600$. Therefore, the maximal number of auxiliary basis functions are 400 and 1600, respectively. We make two observations about the figure. First, the number of auxiliary basis functions required depends only on the number of orbitals N_{orb} used in the calculation, until saturation occurs. That is, in both the $n = 400$ and $n = 1600$ tests, N_{aux} is essentially the same for $N_{\text{orb}} \leq 200$, at which point the number of auxiliary basis functions starts to max out at 400 in the smaller system. Second, we note that the error in the two tests is mostly identical for a given number of orbitals N_{orb} used in the calculation. These two observations suggest that the ISDF procedure behaves precisely how one would hope as the number of primal basis functions (grid points) is increased. That is, (before the auxiliary basis functions max out) the number of auxiliary basis functions and the error in $E_{\text{c}}^{\text{RPA}}$ depends only on N_{orb}, and not on n. This last observation is, of course, only valid when n is large enough and tol in the IDSF is small enough that the errors due to the spatial discretization and the ISDF approximation are negligible compared to the error induced by truncating the number of orbitals.

4.2. **Cubic scaling.** In this test, we show the cubic scaling behavior of the algorithm. The quartic scaling method using traditional density fitting is also plotted for comparison. The traditional density fitting requires that we input basis functions, so we use the basis functions $\{P_\mu\}_{\mu=1}^{N_{\text{aux}}}$ from the ISDF. The results from Figure 4 suggest that as we scale up the system size, we can choose the number of orbitals N_{orb} to use in the calculation as a constant percentage of the number of grid points. So, we choose $N_{\text{orb}} = 0.2n$. We scale the system size up to a maximum of $N_{\text{occ}} = 160$. We can see in Figure 6 that the cubic scaling algorithm greatly outperforms the quartic scaling algorithm for large system sizes.

5. Conclusion

In this paper, we have presented a new cubic scaling algorithm for the computation of the RPA correlation energy. The key of the algorithm was to separate the dependence on j and k



(A) $N_{\text{occ}} = 4$ (B) $N_{\text{occ}} = 32$

FIGURE 4. Convergence of the RPA energy with respect to the total number of orbitals used in the calculation. Both the relative error in $E_{\text{c}}^{\text{RPA}}$ and the number of auxiliary basis functions used in the calculation are plotted. In the error of each plot, the numerical result with all 400 (3200) orbitals is used as the "exact" $E_{\text{c}}^{\text{RPA}}$. An error tolerance of tol $= 10^{-4}$ was used in the ISDF.

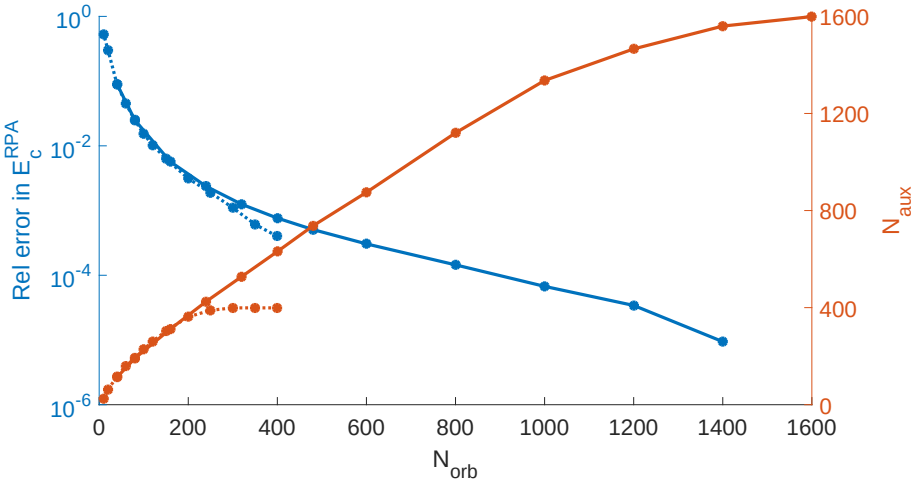


FIGURE 5. Results with $n = 400$ and $n = 1600$ for $N_{\text{occ}} = 4$ with the same external potential. Dotted lines are $n = 400$, solid lines are $n = 1600$. The numerical solution with 1600 orbitals in the $n = 1600$ case is used as the "exact" $E_{\text{c}}^{\text{RPA}}$ for purposes of plotting the error. For the determination of N_{aux}, we use tol $= 10^{-4}$ in the ISDF.

in the denominator of (1.7). This allows a natural cubic scaling method. However, in order to further reduce the computational cost, we employed the ISDF in analogy to how density fitting is used in the quartic algorithm. Another key idea to keep the computational cost down was to take advantage of the periodic and analytic nature of the function in the contour integral, which

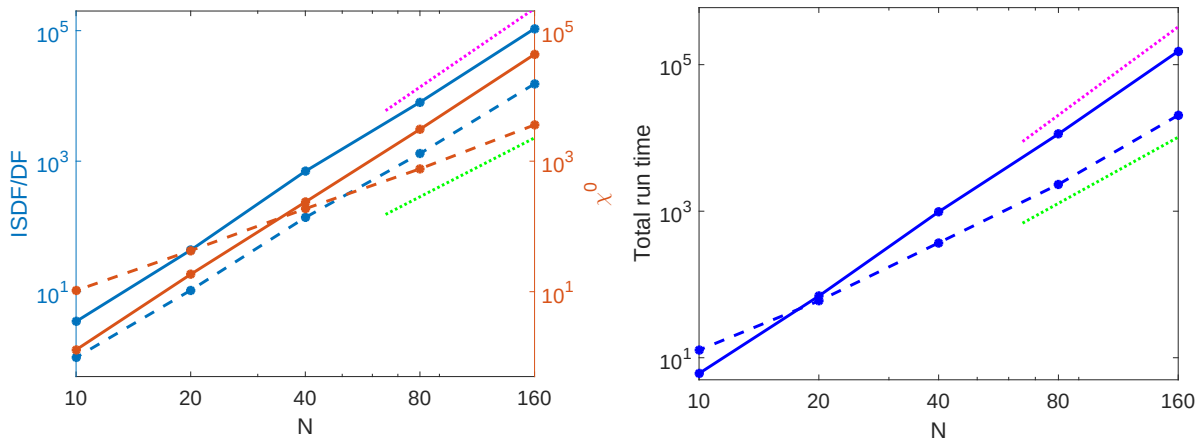


FIGURE 6. The timing results for the quartic scaling method are plotted with solid lines, and the results for the cubic scaling method are plotted with dashed lines. For reference, the purple and green dotted lines represent the slopes of N^4 and N^3 respectively. The left figure compares the time required to calculate χ^0 and the time to perform the respective density fitting schemes for each method. The right figure compares the total run time to calculate E_c^{RPA} for each method.

resulted in a geometrically convergent nested quadrature rule based on the simple trapezoid rule.

It is worth noting that the algorithm presented is highly parallelizable. Step 1, the ISDF, is composed of linear algebra routines which can be parallelized. It is clear that Steps 2 and 3a can be parallelized. Step 3b is parallelizable using the comment at the end of Section 3.2. There is no need to parallelize Step 4 as it is a simple one dimensional integral, but the Step 3 computations for each quadrature point ω_m could also be done in parallel.

Future directions include a parallel implementation of the algorithm, as well as implementation into scientific software. Another direction would be to look into the analytic properties of $\chi^0(i\omega)$. It would also be interesting to apply the ISDF to the Laplace transform method for cubic scaling RPA algorithms. We also plan to extend the algorithm presented in this paper to particle-particle RPA (ppRPA) [21].

APPENDIX A. DERIVATION OF ALGORITHM 3

In this Appendix, we first rigorously derive Algorithm 3. Then we conclude by proving Lemma 3.1.

Recall from Section 3.2 that we wish to find the lowest point of the purple curve in the t-plane. We do this now. First, we note that (3.12) is a Möbius transformation and therefore its inverse maps the imaginary line to a generalized circle in the u-plane. In particular, it maps the upper half imaginary line in the z-plane to the upper semicircle with radius k^{-1} centered at the origin (the purple curve in Figure 2b). To map this semicircle back to the t-plane, we note the formula for the inverse of $\mathrm{sn}(t)$, [2, Chapter 11.3]

$$\mathrm{sn}^{-1}(u|k^2) = \int_0^u \frac{\mathrm{d}s}{\sqrt{(1-s^2)(1-k^2s^2)}}. \tag{A.1}$$

Then we can use basic calculus to minimize $\mathrm{Im}\left[\mathrm{sn}^{-1}(k^{-1}e^{i\theta})\right]$ over $0 \le \theta \le \pi$.

$$\frac{d}{d\theta}\mathrm{Im}\left[\mathrm{sn}^{-1}(k^{-1}e^{i\theta})\right] = \mathrm{Re}\left[\frac{k^{-1}\sqrt{\cos(2\theta)-1-k^{-2}+(k^{-2}-1)e^{-2i\theta}}}{(1+k^{-4}-2k^{-2}\cos(2\theta))(2-2\cos(2\theta))}\right]. \tag{A.2}$$

This expression is 0 if and only if the expression under the radical is nonpositive. Since the imaginary part of the expression under the radical must be 0, we require $\theta \in \{0, \pi/2, \pi\}$. 0 and π correspond to the corners of the rectangle, so this means that $\theta = \pi/2$ must give us the minimum imaginary part of points along the purple curve. In conclusion, we choose our quadrature points in the t-rectangle with imaginary part given by

$$\begin{aligned}\frac{1}{2}\mathrm{Im}\left[\mathrm{sn}^{-1}(ik^{-1})\right] &= \frac{1}{2}\mathrm{Im}\int_0^{ik^{-1}} \frac{\mathrm{d}s}{\sqrt{(1-s^2)(1-k^2s^2)}} \\ &= \frac{1}{2}\int_0^{k^{-1}} \frac{\mathrm{d}s}{\sqrt{(1+s^2)(1+k^2s^2)}}. \end{aligned} \tag{A.3}$$

This integral must be carried out numerically. However, it is simple and only needs to be done once at the beginning of the algorithm, so we just use the midpoint rule. We also note that we can easily remove any guess work here by proving a practically useful bound which can be obtained via the standard error analysis for the midpoint rule. First let

$$g(s) = \frac{1}{(1+s^2)(1+k^2s^2)}. \tag{A.4}$$

Then computation shows

$$g''(s) = \frac{6k^4s^6+5k^4s^4+2k^4s^2+5k^2s^4-2k^2s^2-k^2+2s^2-1}{[(1+s^2)(1+k^2s^2)]^{5/2}}. \tag{A.5}$$

By noting $0 \le k < 1$ and $0 \le sk \le 1$, we have

$$|g''(s)| \le \frac{13s^2+11}{(1+s^2)^{5/2}}. \tag{A.6}$$

Using the fact that the right hand side of (A.6) is decreasing on $[0,\infty)$,

$$\begin{aligned}\left|\int_0^{k^{-1}} g(s)\,\mathrm{d}s - h\sum_{j=1}^{J} f(s_{j+1/2})\right| &\le \frac{1}{24}h^3\sum_{j=1}^{J}|f''(\xi_j)| \\ &\le \frac{1}{24}h^3\left(13h^2+11+\frac{1}{h}\int_0^{k^{-1}-h}|f''(s)|\,\mathrm{d}s\right) \\ &\le \frac{1}{24}h^3\left(13h^2+11+\frac{1}{h}\int_0^{k^{-1}}\frac{13s^2+11}{(1+s^2)^{5/2}}\,\mathrm{d}s\right) \\ &= \frac{1}{24}h^3\left(13h^2+11+\frac{35k^{-3}+33k^{-1}}{3h(1+k^{-2})^{3/2}}\right) \\ &\le \frac{1}{24}h^3\left(13h^2+11+\frac{35}{3h}\right), \end{aligned} \tag{A.7}$$

where the last line uses the fact that the preceding line is a strictly increasing function of k^{-1}. This estimate implies that a mesh size of $1/100$ guarantees an accuracy of 10^{-4}. Considering the fact that if the value of this integral is off by a little it will only slightly change the convergence rate, this is sufficiently accurate.

We conclude this discussion of the quadrature rule with some brief analytic results, including the proof of Lemma 3.1. First, we note that in a realistic system, $M/m \gg 1$ which implies $k \approx 1$. This guarantees that using the midpoint rule to calculate I will not require more than about 100 grid points. Next, we note that $I > K'/4$. This can be seen by showing that the circle with radius $k^{-1/2}$ centered at the origin in the u-plane maps to the horizontal line with imaginary part $K'/2$ in the t-plane. Before proving this statement, let's see why this implies $I > K'/4$. First, note that $1 < k^{-1/2} < k^{-1}$. Therefore, the circle with radius $k^{-1/2}$ in the u-plane must map between the purple and red curves in the t-plane. This means that the purple curve cannot go any lower than $K'/2$, which implies $I > K'/4$. To show that the aforementioned circle maps to a line with constant imaginary part, it is enough to show

$$\text{(A.8)}\quad \operatorname{Im} \int_{k^{-1/2}e^{i\theta_1}}^{k^{-1/2}e^{i\theta_2}} \frac{\mathrm{d}s}{\sqrt{(1-s^2)(1-k^2s^2)}} = \operatorname{Re} \int_{\theta_1}^{\theta_2} \frac{k^{-1/2}\sqrt{2\cos(2\theta) - k^{-1} - k}}{\sqrt{(1-2k^{-1}\cos(2\theta)+k^{-2})(1-2k\cos(2\theta)+k^2)}}\,\mathrm{d}\theta,$$

is equal to 0 for all $0 \le \theta_1 \le \theta_2 \le \pi$. It is easily verified that (for $0 < k < 1$) the denominator on the right is always positive and the numerator is always a pure imaginary number. Therefore, the integrand is always purely imaginary, which proves the claim. Finally, the imaginary part of the line is $K'/2$ since $\operatorname{sn}^{-1}(ik^{-1/2}) = iK'/2$ [6, Table 22.5.2].

Now following [8] and using the fact that $I > K'/4$, we have that for any $M/m > 1$, the error of the quadrature rule is

$$\text{(A.9)}\quad \mathcal{O}\left(\exp\left(\frac{-\pi^2 N_\lambda}{2\log(M/m)+6}\right)\right).$$

REFERENCES

[1] M. Abramowitz and I.-A. Segun. *Handbook of mathematical functions with formulas, graphs, and mathematical table*. Dover Publications, Inc., 1970.

[2] R. Beals and R. Wong. *Special functions: a graduate text*, volume 126. Cambridge University Press, 2010.

[3] J. P. Boyd. Exponentially convergent Fourier-Chebshev quadrature schemes on bounded and infinite intervals. *Journal of scientific computing*, 2(2):99–109, 1987.

[4] P. J. Davis and P. Rabinowitz. *Methods of numerical integration*. Academic Press, 1984.

[5] T. A. Discoll. Schwarz-Christoffel toolbox. available online at `http://www.math.udel.edu/~driscoll/SC/`.

[6] *NIST Digital Library of Mathematical Functions*. http://dlmf.nist.gov/, Release 1.0.14 of 2016-12-21. F. W. J. Olver, A. B. Olde Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller and B. V. Saunders, eds.

[7] H. Eshuis, J. Yarkony, and F. Furche. Fast computation of molecular random phase approximation correlation energies using resolution of the identity and imaginary frequency integration. *The Journal of chemical physics*, 132(23):234114, 2010.

[8] N. Hale, N. J. Higham, and L. N. Trefethen. Computing A^α, $\log(A)$, and related matrix functions by contour integrals. *SIAM Journal on Numerical Analysis*, 46(5):2505–2523, 2008.

[9] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136:B864–B871, 1964.

[10] M. Kaltak, J. Klimeš, and G. Kresse. Cubic scaling algorithm for the random phase approximation: Self-interstitials and vacancies in Si. *Physical Review B*, 90(5):054115, 2014.

[11] M. Kaltak, J. Klimeš, and G. Kresse. Low scaling algorithms for the random phase approximation: Imaginary time and laplace transformations. *Journal of chemical theory and computation*, 10(6):2498–2507, 2014.

[12] W. Kohn and L. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140:A1133–A1138, 1965.

[13] L. Lin, J. Lu, L. Ying, and W. E. Pole-based approximation of the Fermi-Dirac function. *Chin. Ann. Math. Ser. B*, 30:729–742, 2009.

[14] L. Lin, Z. Xu, and L. Ying. Adaptively compressed polarizability operator for accelerating large scale ab initio phonon calculations. *Multiscale Modeling & Simulation*, 15(1):29–55, 2017.

[15] J. Lu and L. Ying. Compression of the electron repulsion integral tensor in tensor hypercontraction format with cubic scaling cost. *Journal of Computational Physics*, 302:329–335, 2015.

[16] J. Lu and L. Ying. Fast algorithm for periodic density fitting for bloch waves. *Ann. Math. Sci. Appl.*, 1:321–339, 2016.

[17] J. E. Moussa. Cubic-scaling algorithm and self-consistent field for the random-phase approximation with second-order screened exchange. *The Journal of chemical physics*, 140(1):014107, 2014.

[18] J. P. Perdew, K. Schmidt, V. Van Doren, C. Van Alsenoy, and P. Geerlings. Jacob's ladder of density functional approximations for the exchange-correlation energy. In *AIP Conference Proceedings*, volume 577, pages 1–20. AIP, 2001.

[19] X. Ren, P. Rinke, V. Blum, J. Wieferink, A. Tkatchenko, A. Sanfilippo, K. Reuter, and M. Scheffler. Resolution-of-identity approach to Hartree-Fock, hybrid density functionals, RPA, MP2 and GW with numeric atom-centered orbital basis functions. *New J. Phys.*, 14:053020, 2012.

[20] X. Ren, P. Rinke, C. Joas, and M. Scheffler. Random-phase approximation and its applications in computational chemistry and materials science. *Journal of Materials Science*, 47(21):7447–7471, 2012.

[21] H. van Aggelen, Y. Yang, and W. Yang. Exchange-correlation energy from pairing matrix fluctuation and the particle-particle random phase approximation. *J. Chem. Phys.*, 140:18A511, 2014.

[22] J. Wilhelm, P. Seewald, M. Del Ben, and J. Hutter. Large-scale cubic-scaling random phase approximation correlation energy calculations using a Gaussian basis. *Journal of Chemical Theory and Computation*, 2016.

DEPARTMENT OF MATHEMATICS, DEPARTMENT OF PHYSICS, AND DEPARTMENT OF CHEMISTRY, DUKE UNIVERSITY, BOX 90320, DURHAM NC 27708, USA
E-mail address: jianfeng@math.duke.edu

DEPARTMENT OF MATHEMATICS, DUKE UNIVERSITY, BOX 90320, DURHAM NC 27708, USA
E-mail address: kyle.thicke@duke.edu