Max-Planck-Institut
für Mathematik
in den Naturwissenschaften
Leipzig

Low-Rank wavelet solver for the Ornstein-Zernike integral equation

by

Maxim V. Fedorov, Heinz-Jürgen Flad, Lars Grasedyck, and Boris N. Khoromskij

Preprint no.: 59 2005

Low-Rank wavelet solver for the Ornstein-Zernike integral equation

Maxim V. Fedorov [*] Heinz-Jürgen Flad [†] Lars Grasedyck [†]
Boris N. Khoromskij [†]

Abstract

A structured wavelet algorithm is developed to solve the Ornstein-Zernike integral equation for simple liquids. The algorithm is based on the discrete wavelet transform of radial distribution functions and different low-rank matrix approximations. The fundamental properties of wavelet bases such as interpolation properties and orthogonality are employed to improve the convergence and speed of the algorithm. In order to solve the integral equation we have applied a combined scheme in which the coarse part of the solution is calculated by the use of wavelets in a multilevel method, while the fine part is solved by the direct iteration. Tests have indicated that the proposed procedure is more effective than the conventional method based on hybrid algorithms.

AMS Subject Classification: 65F50, 65F30, 46B28, 47A80
PACS numbers: 02.60.Nm, 61.20.Ne, 61.20.Gy
Key words. Wavelets, Ornstein-Zernike equation, simple fluids, data-sparse matrix approximations

1 Introduction

Integral equation (IE) theory proved to be a successful tool for treating classical many-body systems. With this radial distribution functions (RDF) are obtained by solving a system of equations formed by the Ornstein-Zernike (OZ) integral equation and a closure relation. The OZ equation is a nonlinear Fredholm integral equation of second-kind which is frequently employed in the theories of various kinds of disordered matter, e.g. classical liquids [22, 23, 32], electrolytes [7, 24, 34], polymers [29], electrons in liquids [36] and plasmas [33, 37]. For most systems the OZ equation has no analytic solution and, therefore, has to be solved numerically. In the case of multicomponent systems (e.g. solvated molecular

[*] Theory and Computation Group, Centre for Synthesis and Chemical Biology, Conway Institute of Biomolecular and Biomedical Research, Department of Chemistry, University College Dublin, Belfield, Dublin 4, Ireland (`Maxim.Fedorov@ucd.ie`).

[†] Max-Planck-Institut für Mathematik in den Naturwissenschaften, Inselstr. 22-26, D-04103 Leipzig, Germany ({`flad,lgr,bokh`}`@mis.mpg.de`).

complexes) this would lead to an essential computational cost. The aim of our paper is to present a new approach for the numerical solution of the OZ equation which is based on the wavelet transform [30] and on certain matrix compression techniques by means of a low-rank approximation. We note that the so-called hierarchical ($\mathcal{H}$) matrix format provides a powerful tool for the data-sparse representation of a wide class of discrete nonlocal (integral) operators [19, 20, 17]. In the multi-dimensional perspective, the $\mathcal{H}$-matrix techniques can be effectively combined with the Kronecker-product approximation [21].

For the purpose of clarity we will describe the procedure only for the simplest case of a mono-atomic isotropic liquid with spherically symmetric Lennard-Jones interaction potential between the particles,

$$U_{LJ}(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right], \tag{1.1}$$

where σ and ϵ are the size and energy parameters, respectively. Our method can be in principle extended to the case of molecular systems [4], with more complicated potentials, without essential changes of the algorithm.

In the theory of liquids the principal structural quantity of interest is the pair correlation function, $g(r, \Omega)$, which is proportional to the probability of observing a pair of particles at a given distance r and mutual orientation Ω. For the case of a mono-atomic liquid with spherical potential between particles we can omit the Ω–dependence of the pair correlation function and consider it as a function $g(r)$ only.

Let us denote the total correlation function $h(r) := g(r) - 1$. The OZ equation relates this function with the direct correlation function $c(r)$ for an isotropic liquid with density ρ by

$$h(r) = c(r) + \rho \int_{\mathbb{R}^3} c(|\mathbf{r} - \mathbf{r}'|)h(|\mathbf{r}'|)d\mathbf{r}'. \tag{1.2}$$

There are two unknowns in Eq. (1.2) and therefore it is still incomplete. A second equation, usually named "closure" relation, is required which couples these quantities with the interaction potential $U(r)$. Formally the closure relation is written as

$$h(r) = \exp[-\beta U(r) + h(r) - c(r) + b(r)] - 1, \tag{1.3}$$

where $\beta = (k_B T)^{-1}$ is the inverse temperature, while k_B is the Boltzmann constant. The closure relation introduces the bridge function $b(r)$ [32]. Given $U(r)$, T, ρ, and $b(r)$, the IE method consists in finding a solution to Eqs. (1.2) and (1.3). By solving this system of equations all the required correlation functions and, hence, all the thermodynamic and structural properties of the fluid may be obtained. Since there is no exact expression for $b(r)$, the approximation of the bridge function is the key to contemporary IE theories. The list of such approximating closures is still expanding and includes, for example, $b(r) = 0$ for the hypernetted chain (HNC) closure, or $b(r) = \ln(1 + h(r) - c(r)) - h(r) - c(r)$ in the Percus-Yevick approximation, etc. These approximations have been studied extensively for simple liquids, and their pros and cons are well documented in the literature [23, 32]. Correlation functions for a mono-atomic liquid with Lennard-Jones interaction potential are given in Fig. 1.1.

There are only a few special cases where Eqs. (1.2) and (1.3) can be solved analytically and, therefore, numerical solutions are necessary. For numerical calculations, the Fourier

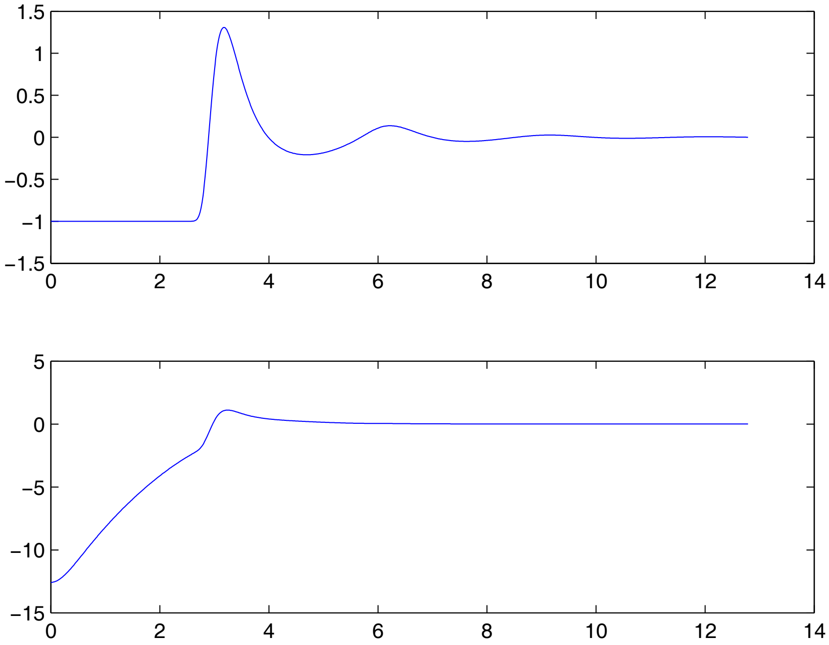

Figure 1.1: Radial parts of correlation functions $h(r)$ (top) and $c(r)$ (bottom) of a simple mono-atomic liquid with density $\rho = 0.7$ and Lennard–Jones potential parameter $\varepsilon = 0.7$.

representation of the OZ equation,

$$\hat{h}(\mathbf{k}) - \hat{c}(\mathbf{k}) = \frac{\rho\hat{c}^2(\mathbf{k})}{1 - \rho\hat{c}(\mathbf{k})}, \tag{1.4}$$

is usually applied, where the hat means the three-dimensional (3D) Fourier transform (FT), which reduces for a spherically symmetric function $f(\mathbf{r})$ to the spherical Fourier-Bessel transform of its radial part $f(r)$,

$$\hat{f}(k) = \frac{4\pi}{k}\int_0^\infty r\sin(kr)f(r)dr, \tag{1.5}$$

where $\hat{f}(k)$ is the radial part of $\hat{f}(\mathbf{k})$. The inverse Fourier transform (FT) $^{-1}$ can be obtained in a similar manner:

$$f(r) = \frac{4\pi}{r}\int_0^\infty k\sin(kr)\hat{f}(k)dk. \tag{1.6}$$

In numerical calculations, the function $f(r)$ can be assumed to be of finite support, i.e., $f(r) = 0$ for $r \geq R$. For simplicity, we approximate $f(r)$ on a regular grid with $N_J = 2^J$ points, where J is an integer which determines the step size d_J of the grid as $d_J = 2^{-J}R$. Therefore, we can regard J as the *resolution level*. On a regular grid, the discrete versions of the transformations (1.5) and (1.6) can be calculated with $O(N_J \log N_J)$ operations using the Fast Fourier Transform (FFT).

Recently, an alternative approach was proposed by several authors [5, 4, 12] based on the use of wavelet basis functions. The multi-level wavelet basis is able to treat local peculiarities of the radial distribution functions and therefore one obtains a more flexible and robust scheme for solving the arising system of integral equations. The main idea of this algorithm is to use the wavelet decomposition of the correlation functions h and c on level J, and then find the initial approximation on the "coarse resolution level" $J_0 < J$. In this case we use only N_{J_0} of the wavelet coefficients which is about 2^{J-J_0} times less than the original number of grid points N_J.

The computational scheme described in Refs. [5, 4, 12] includes numerous iteration loops which require extensive matrix operations. These operations have been performed in the full matrix format resulting in the overall complexity $O(N_{J_0}^3 \log N_{J_0})$ in each internal iteration, where N_{J_0} is the number of wavelet basis functions in usage. This complexity bound limits the number of basis functions we can handle on a standard computer, hence it does not allow an accurate representation of the radial distribution functions in terms of these bases. On the other hand, a more detailed analysis of the full matrices involved in the different steps of the computational scheme shows a high redundancy in the data. In fact, it is possible to employ special structural properties of these matrices which lead to a dramatical reduction of the computational cost while preserving the accuracy of the approximations. The main goal of this paper is to demonstrate the potential efficiency of such an approach, first for isotropic simple liquids. The compressed scheme requires an overall complexity $O(N_{J_0}^2 \log N_{J_0})$ and at average hundred times less operations than the full matrix approximation with several thousands of degrees of freedom N_J. The treatment of the general three-dimensional model includes completely similar matrix operations which potentially allows to extend our method to the case of three-dimensional models, where the full-matrix approximation seems to be non-tractable.

The simplest method is based on the Picard scheme and proposes direct iterations. Unfortunately several hundred iterations may be needed even for a rather accurate initial approximation of $\gamma(r)$. The Newton method [14, 27, 28] appears to be much more efficient, but requires the calculation of the Jacobian matrix. Thus, the conventional numerical schemes for solving IE suggest the solution to be divided into the "coarse" and "fine" parts and the algorithm to be a hybrid of the Newton-Raphson (NR) and Picard schemes for the coarse and fine parts, respectively. The former can be obtained as an expansion in the basis of roof functions [14] or plane waves [27, 28]. Because this method does not guarantee the storage problems to be completely solved, special efforts are required to approximate and store the full Jacobian matrix. Another way in this direction is to use a combination of multi-grid methods and matrix-free iterations [18]. Some matrix-free iterative procedures for solving the OZ equation were demonstrated in [25], [26], however a discussion of this topic is beyond the scope of our paper.

The current status of the implementation requires a routine for the additional Newton-Raphson acceleration, hence, in this paper we will consider both the Picard algorithm and the Newton-Raphson method and then discuss the choice of a basis set for the coarse solution, namely wavelets.

2 Wavelet based numerical scheme

2.1 Wavelet representation of the OZ equation

For simplicity we restrict our considerations only to orthonormal and compactly supported wavelets. Details of wavelet applications to IE are described in [5, 4]. Some introduction to the wavelet theory is given in the Appendix of this article.

We employ the fact that any square-integrable function $f(r)$ can be expanded as a sum of linear combinations of scaling functions $\{\varphi_s^{J_0}(r)\}$ at the chosen resolution level J_0 and wavelet functions $\{\psi_s^j(r)\}$ at all finer resolutions $J \geq j \geq J_0$. For a discretized function given on a finite interval the finest resolution is restricted by the resolution level J, which corresponds to the step size $d_j = 2^{-j}R$ of the mesh. The sequence of different translations on different scales $\{j\}$ is also limited by the number $N_j \approxeq 2^j$. The coefficient of proportionality depends on the length of the wavelet support and the boundary conditions used in the expansion

$$f(r) = \sum_{s=0}^{N_{J_0}} a_s \varphi_s^{J_0}(r) + \sum_{j=J_0}^{J} \sum_{s=0}^{N_j} d_s^j \psi_s^j(r), \tag{2.1}$$

where the coefficients $\{a_s\}$ and $\{d_s^j\}$ are obtained by the scalar product with the corresponding dual basis functions:

$$a_s = \langle f, \widetilde{\varphi}_s^{J_0} \rangle; \quad d_s^j = \langle f, \widetilde{\psi}_s^j \rangle. \tag{2.2}$$

The calculation of the coefficients in (2.1) and (2.2) is known as the discrete wavelet transform (DWT). In expansion (2.1) the first term gives a "coarse" approximation of $f(r)$ at the lowest resolution level J_0 and the second term gives a sequence of refinements (details) until the finest resolution level J. In practice the details are cut off at an appropriate resolution level J resulting in a finite number of decomposition levels $J_\Delta \equiv J - J_0$.

Applying the fast wavelet transform (FWT) to calculate the coefficients [30], we can avoid the direct calculations of (2.2). The elegant pyramidal procedure has a linear cost. To apply the wavelet technique to IE, we employ the wavelet representation of the OZ equation, which is given in [5]. Here, we only briefly outline the key steps of the method.

For numerical purposes we use a different form of the OZ equation, namely,

$$\gamma(r) = \rho \int c(|\mathbf{r} - \mathbf{r}'|)\gamma(|\mathbf{r}|')d\mathbf{r}' + \rho \int c(|\mathbf{r} - \mathbf{r}'|)c(|\mathbf{r}|')d\mathbf{r}', \tag{2.3}$$

where $\gamma(r)$ is the so-called indirect correlation function $\gamma(r) = h(r) - c(r)$. We then write the closure relations for $\gamma(r)$ in similar manner as (1.3):

$$c(r) = \exp[-\beta U(r) + \gamma(r) + b_\gamma(r)] - \gamma(r) - 1. \tag{2.4}$$

As in the conventional IE method [14, 27, 28] we decompose the radial parts of the correlation functions into the coarse and fine components:

$$c(r) = c_c(r) + c_f(r), \quad \gamma(r) = \gamma_c(r) + \gamma_f(r). \tag{2.5}$$

Then, we express the radial parts of the coarse components of direct and indirect correlation functions in terms of the chosen wavelet scaling functions on the appropriate resolution level J_0:

$$\gamma_c(r) = \sum_{s=0}^{N_{J_0}} \Gamma_s \varphi_s^{J_0}(r), \qquad c_c(r) = \sum_{s'=0}^{N_{J_0}} C_{s'} \varphi_{s'}^{J_0}(r). \tag{2.6}$$

In the following we omit the index J for N_{J_0} (i.e., $N_0 = N_{J_0}$) and J_0 in the notation φ^{J_0} because they do not change during the calculations. We denote the set of approximating coefficients $\{\Gamma_s\}$, and $\{C_{s'}\}$, as the vector–columns $\mathbf{\Gamma}$, and $\mathbf{C}$ respectively. In general the relevant coefficients are found by the inner product of $c(r)$ and $\gamma(r)$ with the scaling functions $\{\varphi\}$ similarly to (4.8). Here, we use the FWT to compute them.

We define a 3D function $\varphi_s(\mathbf{r})$ as a spherically symmetric function with the 1D scaling function $\varphi_s(r)$ as the radial part of it. Then, we introduce a 3D function $\Phi_{ss'}(\mathbf{r})$ which is the convolution product of two functions $\varphi_s(\mathbf{r})$ and $\varphi_{s'}(\mathbf{r})$:

$$\Phi_{ss'}(\mathbf{r}) = \int_{\mathbb{R}^3} \varphi_s(|\mathbf{r} - \mathbf{r}'|)\varphi_{s'}(|\mathbf{r}'|) d\mathbf{r}'. \tag{2.7}$$

As the $\varphi_s(\mathbf{r})$ and $\varphi_{s'}(\mathbf{r})$ are spherically symmetric with the same symmetry centre which we assume to be zero for simplicity, the functions $\Phi_{ss'}(\mathbf{r})$ are spherically symmetric as well. We denote the radial parts of these functions as $\Phi_{ss'}(r)$. This function can be calculated with the help of the spherical Fourier-Bessel transform (1.5) of the corresponding wavelet functions.

If we substitute (2.6) and (2.7) in the OZ equation (2.3) we obtain the following equation for the wavelet representation of the radial parts of the correlation functions:

$$\sum_{s=0}^{N_0} \Gamma_s \varphi_s(r) = \rho \sum_{s,s'=0}^{N_0} \Gamma_s C_{s'} \Phi_{ss'}(r) + \rho \sum_{s,s'=0}^{N_0} C_s C_{s'} \Phi_{ss'}(r). \tag{2.8}$$

Taking the 1D scalar product of both parts of this equation with the scaling function $\varphi_m(r)$ we obtain (due to the orthogonality of the scaling functions)

$$\Gamma_m = \rho \sum_{s,s'=0}^{N_0} \Gamma_s C_{s'} \int \varphi_m(r)\Phi_{ss'}(r) dr + \rho \sum_{s,s'=0}^{N_0} C_s C_{s'} \int \varphi_m(r)\Phi_{ss'}(r) dr, \tag{2.9}$$

for $m = 0, 1, ..., N_0$. We introduce the scalar product convolution matrix $\mathbf{W} \in \mathbb{R}^{(N_0+1)\times(N_0+1)\times(N_0+1)}$ whose elements are given by

$$W(s, s', m) = \int \varphi_m(r)\Phi_{ss'}(r) dr. \tag{2.10}$$

For the sake of clarity in the following we introduce a new variable like $N_0' = N_0 + 1$ which determines the number of wavelet coefficients used.

We can see that for each fixed s, s' a vector $W(s, s', :)$ is the set of scaling function coefficients of the wavelet transform (4.8) of the convolution product $\Phi_{ss'}$, which is, in fact, a symmetric $N_0' \times N_0'$ matrix.

We introduce matrices $\mathbf{A_C}, \mathbf{A_\Gamma} \in \mathbb{R}^{N_0' \times N_0'}$ and a column-vector $\mathbf{B} \in \mathbb{R}^{N_0'}$ whose elements are defined as

$$A_C(m,:) = \mathbf{C}^T \cdot W(:,:,m), \quad A_\Gamma(m,:) = \mathbf{\Gamma}^T \cdot W(:,:,m), \quad \mathbf{B} = \mathbf{A_C} \cdot \mathbf{C}, \tag{2.11}$$

where the upper index T denotes the transposition, the symbol " $\cdot$ " denotes the matrix product and " : " means a sub-matrix defined on an one-dimensional index set. In equation (2.10) the convolution matrix $\mathbf{W}$ depends only on the choice of the wavelet basis and the number of decomposition levels, hence it has to be evaluated only once before the calculation of the correlation functions and can then be stored. By substitution of (2.11) to (2.9), we obtain the wavelet representation of the OZ equation for the coarse parts of the chosen number J_0 of resolution levels:

$$\mathbf{\Gamma} = \rho \left[\mathbf{I} - \rho \cdot \mathbf{A_C}\right]^{-1} \cdot \mathbf{B}, \tag{2.12}$$

where $\mathbf{I}$ is the unity matrix, while $\left[\mathbf{I} - \rho\mathbf{A_C}\right]^{-1}$ is the matrix inverse to $[\mathbf{I} - \rho\mathbf{A_C}]$. Using the inverse FWT we can reconstruct $\gamma_c(r)$ from the set of coefficients $\mathbf{\Gamma}$. As the fine parts of the correlation functions are assumed to be zero at the initial step we approximate (2.4) as:

$$c_c(r) = \exp[-\beta U(r) + \gamma_c(r) + b(\gamma_c)] - \gamma_c - 1. \tag{2.13}$$

Using the FWT we perform the wavelet transform of $c_c(r)$ to obtain $\mathbf{C}$ and use it as a next input for (2.12). Thus, our numerical scheme proposes the following cycle for obtaining the coarse solution:

$$\mathbf{\Gamma}^{in} \overset{(FWT)^{-1}}{\longrightarrow} \gamma_c(r) \overset{(2.13)}{\longrightarrow} c_c(r) \overset{(FWT)}{\longrightarrow} \mathbf{C} \overset{(2.12)}{\longrightarrow} \mathbf{\Gamma}^{out}. \tag{2.14}$$

If the number of wavelet coefficients is not too large, the solution of (2.12) with the interpolation process by direct and back FWT requires less computational cost than the direct and the back FFT at each iteration.

We improve the convergence of the scheme by using the Newton algorithm. We can regard the system of equations (2.9) as the vector equation $\mathbf{F}(\mathbf{\Gamma}) = 0$ defined by a multivariate function $\mathbf{F}(\mathbf{\Gamma}) = \{F_m(\Gamma)\}$, $m = 0, 1, ..., N_0$. The components of this function are given by

$$F_m(\Gamma) = \rho \sum_{s,s'=0}^{N_0} \Gamma_s C_{s'} \int \varphi_m(r) \Phi_{ss'}(r) dr + \rho \sum_{s,s'=0}^{N_0} C_s C_{s'} \int \varphi_m(r) \Phi_{ss'}(r) - \Gamma_m. \tag{2.15}$$

In the matrix form we have to solve the nonlinear equation (since $\mathbf{B} = \mathbf{A_C} \cdot \mathbf{C}$)

$$\mathbf{F}(\mathbf{\Gamma}) = \rho \mathbf{A_\Gamma} \cdot \mathbf{C} + \rho \mathbf{B} - \mathbf{\Gamma} = 0, \tag{2.16}$$

where $\mathbf{C} = \mathbf{C}(\mathbf{\Gamma})$ depends on $\mathbf{\Gamma}$ according to (2.13). In order to perform the Newton algorithm we need to determine the elements of the Jacobian matrix $\mathbf{G} = \{G_m^k\}_{k,m=0}^{N_0}$ by differentiating this equation with respect to Γ_k,

$$G_m^k = \frac{\partial F_m}{\partial \Gamma_k} = \rho \sum_{s'=0}^{N_0} C_{s'}\, W(k,s',m) + \rho \sum_{s,s'=0}^{N_0} C_s^k\, W(k,s',m)\,(\Gamma_{s'} + 2C_{s'}) - \delta_{m,k}, \tag{2.17}$$

where $C_s^k = \frac{dC_s}{d\Gamma_k}$. We can rewrite this equation in the matrix form as:

$$\mathbf{G} = \rho \mathbf{A_C} + \rho(\mathbf{A_\Gamma} + 2\mathbf{A_C}) \cdot \mathbf{C}^* - \mathbf{I}, \tag{2.18}$$

where $\mathbf{C}^* = \{C_s^k\}_{k,m=0}^{N_0} \in \mathbb{R}^{N_0' \times N_0'}$.

2.2 Global iteration

When the required accuracy for the coarse part is achieved we start the second loop for calculating the fine part $\gamma_f(r)$. For this purpose we consider the coarse part $\gamma_c(r)$ as an initial approximation for $\gamma(r)$ at the second iteration loop. Starting from this approximation we perform the calculations by the fixed-point (Picard) direct iterations:

$$\gamma^{in}(r) \xrightarrow{(2.4)} c(r) \xrightarrow{(FFT)} \widehat{c}(k) \xrightarrow{(2.3)} \widehat{\gamma}(k) \xrightarrow{(FFT)^{-1}} \gamma^{out}(r). \tag{2.19}$$

If the coarse solution is close enough to the true one the number of fixed-point iterations for the fine discretisation is not so large (it does not exceed several dozens).

The efficiency of the method depends on the number of coefficients. Due to the special choice of the wavelet basis this number can be rather small, providing a rapid convergence of iterations.

As it was shown in [5, 4] this method allows us to solve integral equations for simple and molecular liquids with less computational effort than the conventional methods of integral equation theory. But as we will show in the next chapter the speed and the convergence of this method can be dramatically improved using the fundamental properties of wavelets and radial correlation functions.

2.3 Fast solver for the coarse solution

Although the scheme described above is significantly faster than conventional methods, we still have to pay for the direct and back FWT which costs about $O(N)$ for each cycle, where N is the size of the finest grid. Fortunately, we have a possibility to avoid these operations. Note that in most cases of practical interest (i.e., non hard sphere potentials) the functions $\gamma(r)$ and $c(r)$ are quite smooth. Thus, these functions are at least twice differentiable in the segment $(0, \infty]$. In this case we can employ the fact that for several wavelet families such as Coifman wavelets [8] (see also the hyperbolic wavelets [10] for the multi-dimensional case) the coefficients of the wavelet expansion of a function $f(x)$ can be approximately evaluated directly from the values of the function itself [9]. The approximation error depends mainly on the basis set and number of derivatives [2, 8, 31]. In fact, we can also obtain the values of the reconstructed function at the nodes directly from the wavelet coefficients. In our case this means that we can approximate the values of $\mathbf{C}$ using the relationship (2.13) as

$$C_s \approx \exp[-\beta U(sd_{J_0}) + K\Gamma_s + b(K\Gamma_s)]/K - \Gamma_s - 1/K, \tag{2.20}$$

where d_{J_0} denotes the distance between the nodes at level J_0 and K is a normalizing constant. These values depend only on the particular implementation of the FWT algorithm. In our case $d_{J_0} = d_J 2^{J_\Delta}$, where d_J is the size of the grid on the finest level J. The constant K is equal to $2^{-J_\Delta/2}$ with $J_\Delta = J - J_0$. Thereby we can avoid the direct and back FWT in the cycle (2.14) and simplify our scheme to

$$\mathbf{\Gamma}^{in} \xrightarrow{(2.20)} \mathbf{C} \xrightarrow{(2.12)} \mathbf{\Gamma}^{out}. \tag{2.21}$$

We also employ the relation (2.20) for the analytical calculation of the Jacobian matrix. Taking into account the approximate relation (2.20) we can the find elements of $\mathbf{C}^*$ in (2.18) as

$$C_s^k \approx \left(\exp[-\beta U(sd_{J_0}) + K\Gamma_s + b(K\Gamma_s)](1 + \frac{\partial b(sd_{J_0})}{\partial \Gamma_s}) - 1 \right) \delta_{s,k}, \tag{2.22}$$

where $\delta_{s,k}$ is the Kronecker symbol.

Therefore, in this approximation the matrix $\mathbf{C}^*$ is strongly diagonal dominant and we can avoid the expensive numerical calculation of the Jacobian $\mathbf{G}$ using this analytical relationship. We also use this approach for the calculation of the matrix $\mathbf{W}$.

2.4 ACA algorithm applied to $\mathbf{A_C}$ and $\mathbf{A_\Gamma}$

In this section we explain in detail how one can compress matrices like $\mathbf{A} = \mathbf{A_C}$ or $\mathbf{A} = \mathbf{A_\Gamma}$ efficiently *without* assembling the whole matrix. This method is then applied to $\mathbf{A_C}$ and $\mathbf{A_\Gamma}$, respectively.

In order to assemble the matrix $\mathbf{A}$ efficiently, we exploit the fact that $\mathbf{A}$ can be approximated up to some small error by a matrix $\tilde{\mathbf{A}}$ of considerably lower rank, i.e.,

$$K := \operatorname{rank}(\tilde{\mathbf{A}}) \ll N_0'.$$

We can exploit this theoretical feature by writing the $N_0' \times N_0'$ matrix $\tilde{\mathbf{A}} \approx \mathbf{A}$ in the factorized form

$$\tilde{\mathbf{A}} = UV^T, \qquad U, V \in \mathbb{R}^{N_0' \times K}. \tag{2.23}$$

Obviously, we have to store and compute only the $2N_0'K$ entries for U and V instead of the $(N_0')^2$ entries for $\mathbf{A}$. The matrix $\tilde{\mathbf{A}}$ can be used in the computation of the inverse $[\mathbf{I} - \rho\mathbf{A}]^{-1}$ by use of the Sherman-Morrison-Woodbury formula that we will explain in the next Section 2.5. The same holds for the inversion of the Jacobian $\mathbf{G}$.

The factorized form (2.23) of the matrix $\tilde{\mathbf{A}} \approx \mathbf{A}$ can be obtained *without* computation of the whole matrix $\mathbf{A}$. Instead, we need to compute only a few rows and columns of $\mathbf{A}$. Consequently, we need to compute only a few layers of the three-dimensional tensor $\mathbf{W}$. The setup of the factorized form (2.23) is done by the Adaptive Cross Approximation (ACA) algorithm [1] that is given in Algorithm 1.

The total complexity of Algorithm 1 is $\mathcal{O}(N_0'K^2)$, but only $2N_0'K$ entries of $\mathbf{A}$ have to be computed, namely the rows $I_{\mathrm{row}} := \{i_1, \ldots, i_K\}$ and the columns $J_{\mathrm{col}} := \{j_1, \ldots, j_K\}$ of $\mathbf{A}$. This means, we need only the entries $W(:, i, :)$ and $W(:, :, j)$ of the tensor $\mathbf{W}$ for $i \in I_{\mathrm{row}}$ and $j \in J_{\mathrm{col}}$. Instead of the $N_0'^3$ entries of $\mathbf{W}$ we have to setup only $2N_0'^2K$ entries. This is further reduced to $N_0'K$ if $\mathbf{W}$ is sparse. We will comment on the choice of the parameter K in Section 3.

Remark 2.1 (On the choice of the pivots i_k^*, j_k^*) *The general choice of the pivots in Algorithm 1 can be changed in the following three ways.*

1. *The initial pivot index i_1^* can be chosen based on knowledge about the underlying physical or chemical application. Basically, we are only interested in a good initial guess for a row i_1^* with a large norm.*

2. *If all pivot indices i_k^*, j_k^* are given, e.g., by some a priori knowledge or due to the fact that we have already computed a low rank "cross approximation" for a "similar" matrix, then the whole choice of pivot indices can be skipped.*

Algorithm 1 ACA with partial pivoting

```
procedure ACA($\mathbf{A}$, var $U$, $V$, $K$)
Choose an initial pivot index $i_1^* \in \{1, \dots, N_0'\}$
$k := 0$
repeat
    $k := k + 1$
    Compute the entries of the vector $V(:, k) \in \mathbb{R}^{N_0'}$ by
        $V(j, k) := A(i_k^*, j) - \sum_{\mu=1}^{k-1} U(i_k^*, \mu) V(j, \mu).$
    Determine an index $j_k^*$ that maximises $\delta := |V(j_k^*, k)|$
    Compute the entries of the vector $U(:, k) \in \mathbb{R}^{N_0'}$ by
        $U(i, k) := \frac{1}{\delta} \left( A(i, j_k^*) - \sum_{\mu=1}^{k-1} U(i, \mu) V(j_k^*, \mu) \right)$
    Determine the next pivot index $i_{k+1}^* \neq i_k^*$ that maximises $|U(i_{k+1}^*, k)|$
until $k = K$
```

3. *If we want to compute a cross approximation not only for a single matrix but a set of quite "similar" matrices, then we can determine the pivot indices just once and use these for all matrices. Since the pivot indices are determined only once, we can enhance the search for the pivots in the following way: given an initial guess (i_k^*, j_k^*) and corresponding vectors*

$$V(:, k) = A(i_k^*, :) - \sum_{\mu=1}^{k-1} U(i_k^*, \mu) V(:, \mu), \quad U(:, k) = A(:, j_k^*) - \sum_{\mu=1}^{k-1} U(:, \mu) V(j_k^*, \mu)$$

we alternately update one of the two by exchanging the pivot index i_k^* (or j_k^*) by the maximiser of $|U(:, k)|$ (or $|V(:, k)|$, respectively). We call this the p times enhanced *pivot search if we update both i_k^* and j_k^* in this way p times.*

2.5 Sherman-Morrison-Woodbury formula for the matrix inverse

The computation of the inverse $[\mathbf{I} - \rho \mathbf{A_C}]^{-1}$ and the vector Γ in formula (2.12) can be simplified if the matrix $\mathbf{A_C}$ is replaced by the low-rank approximation $\tilde{\mathbf{A}}_\mathbf{C} = UV^T \approx \mathbf{A_C}$ due to the so-called Sherman-Morrison-Woodbury formula [16]:

$$\begin{aligned}
\Gamma &= \rho[\mathbf{I} - \rho \mathbf{A_C}]^{-1} \mathbf{B} \\
&\approx \rho[\mathbf{I} - \rho U V^T]^{-1} \mathbf{B} \\
&= \rho[\mathbf{I} + \rho U [I - \rho V^T U]^{-1} V^T] \mathbf{B} \\
&= \rho \mathbf{B} + \rho U [I - \rho V^T U]^{-1} \left(V^T \rho \mathbf{B} \right).
\end{aligned}$$

By this means, we first compute the matrix-vector product $Y := V^T\rho\mathbf{B}$ for the $K \times N_0'$ matrix V^T, solve the $K \times K$ system $[I - \rho V^T U]Z = Y$ and multiply Z to the $N_0' \times K$ matrix ρU. The total complexity for the computation of Γ reduces from $\mathcal{O}(N_0'^3)$ to $\mathcal{O}(N_0'K^2)$.

The computation of the inverse to the Jacobian $\mathbf{G}$ works in the same way: let both $\mathbf{A_C}$ and $\mathbf{A_\Gamma}$ be given in the factorized form

$$\mathbf{A_C} \approx UV^T, \quad \mathbf{A_\Gamma} \approx U_\Gamma V_\Gamma^T, \qquad U, V, U_\Gamma, V_\Gamma \in \mathbb{R}^{N_0' \times K}.$$

We define the $N_0' \times 2K$ matrices

$$U_G := [U \mid U_\Gamma], \quad V_G := \left[(\mathbf{I} + 2(\mathbf{C}^*)^T)V \mid (\mathbf{C}^*)^T V_\Gamma\right] \quad \in \mathbb{R}^{N_0' \times 2K} \tag{2.24}$$

and observe

$$\begin{aligned} \mathbf{G} &= \rho\mathbf{A_C} + \rho\left(\mathbf{A_\Gamma} + 2\mathbf{A_C}\right)\mathbf{C}^* - \mathbf{I} \\ &= \rho\left(\mathbf{A_\Gamma}\mathbf{C}^* + \mathbf{A_C}(\mathbf{I} + 2\mathbf{C}^*)\right) - \mathbf{I} \\ &\approx \rho\left(UV^T(\mathbf{I} + 2\mathbf{C}^*) + U_\Gamma V_\Gamma^T \mathbf{C}^*\right) - \mathbf{I} \\ &= \rho U_G V_G^T - \mathbf{I}. \end{aligned}$$

As above we can derive a formula for the evaluation $\Gamma_{i+1} := \Gamma_i - \mathbf{G}^{-1}F(\Gamma_i)$ of the inverse in order to obtain

$$\begin{aligned} \Gamma_{i+1} &\approx \Gamma_i + \left[\mathbf{I} - \rho U_G V_G^T\right]^{-1} F(\Gamma_i) \\ &= \Gamma_i + F(\Gamma_i) + \rho U_G (I - \rho V_G^T U_G)^{-1}\left(V_G^T F(\Gamma_i)\right). \end{aligned}$$

The computation of Γ_{i+1} involves the inversion of the $2K \times 2K$ matrix $I - \rho V_G^T U_G$ and two matrix-vector products for the $N_0' \times 2K$ matrices V_G, U_G. The total complexity to approximate Γ_{i+1} is $\mathcal{O}(N_0'K^2)$. One can even omit setting up U_G, V_G explicitly and instead insert formula (2.24) in the above approximation for Γ_{i+1}. An efficient procedure for this is given in Algorithm 2.

2.6 Details of calculations

To illustrate our scheme we have investigated a uniform Lennard-Jones fluid with interacting potential $U(r)$ given by $U(r) = 4\epsilon[(\sigma/r)^{12} - (\sigma/r)^6]$, where σ and ϵ are the size and energy parameters, respectively. The precision parameter η for the numerical solution was equal to 10^{-4}.

We solved the OZ equation (see Introduction) of the isotropic Lennard-Jones liquid with $\rho = 0.7$, $\sigma = 3$, and $\epsilon = 0.7$. For the finest grid we used $N_J = 4096$ and a stepwidth $d_J = 1/80$.

For the wavelet algorithm we have used $N_0' = 256$ of the scaling function coefficients. The Picard iteration loops were running until the mean square norm of the difference between the solutions of the adjacent loops was less than the threshold parameter 10^{-4}.

First of all we have solved the coarse approximation to the OZ equation by use of the full-format Picard algorithm to obtain the matrix $\mathbf{A_C} \in \mathbb{R}^{256\times256}$. Then, starting from the obtained coarse solution, we performed several dozens of direct Picard iterations on the finest grid (using the FFT-based algorithm) to refine the final solution of the OZ equation.

Algorithm 2 Efficient computation of Γ_{i+1}

```
procedure NewtonStep($U$, $V$, $U_\Gamma$, $V_\Gamma$, $C^*$, $\rho$, $\Gamma_i$, $F(\Gamma_i)$, var $\Gamma_{i+1}$)
Initialize $\Gamma_i + 1 := \Gamma_i + F(\Gamma_i)$
Compute the three vectors
    $Y_0 := C^* F(\Gamma_i) \ \in \mathbb{R}^{N_0'}, \quad Y_1 := V^T(F(\Gamma_i) + 2Y_0) \ \in \mathbb{R}^K, \quad Y_2 := V_\Gamma^T Y_0 \ \in \mathbb{R}^K.$
Setup the $2K \times 2K$ matrix
    $M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}, \quad M_{11}, M_{12}, M_{21}, M_{22} \in \mathbb{R}^{K \times K}$
by using a temporary matrix $U_C \in \mathbb{R}^{N_0' \times K}$:
$U_C := C^* U$, $M_{21} := -\rho V_\Gamma^T U_C$, $U_C := 2U_C + U$, $M_{11} := I - \rho V^T U_C$
$U_C := C^* U_\Gamma$, $M_{22} := I - \rho V_\Gamma^T U_C$, $U_C := 2U_C + U_\Gamma$, $M_{12} := -\rho V^T U_C$
Solve the linear system
    $\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} := \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}^{-1} \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$
Compute the vector
    $\Gamma_{i+1} := \Gamma_i + F(\Gamma_i) + \rho U Z_1 + \rho U_\Gamma Z_2$
```

The same procedure was repeated for the Newton scheme.

The evaluation of the convolution product (2.7) can be easily performed in the case of spherically symmetric functions. In this case it reduces to a one-dimensional integration of the Fourier transform (1.4) of the relative scaling functions. Moreover, most of the coefficients (2.10) are nullified due to the symmetry of the convolution and compression properties of the FWT for integral operators. According to [2], we can ignore all the coefficients which are less than a constant μ. We choose $\mu = 10^{-5}$, which provides an accurate calculation of correlation functions. For all our calculations we start from the zero initial vector Γ_1^{in}.

In addition we should mention that **W** depends only on the basis set and the resolution level. Therefore, we do not need to calculate **W** in each cycle (and in the each run of the program as well) but evaluate the matrix **W** only once and then save it as a file in an operating memory. In general, it is possible to create a date-base of these matrices. That is why we suppose to regard them as table data and do not take into account its calculation expenses.

All calculations were performed on a single CPU of a SUN UltraSPARC running on 900 MHz clock rate. We used the Matlab 7.1 program [6] for all our calculations.

3 Numerical results and discussion

3.1 Comparative analysis of the compressed matrix representation

In Figure 3.1 we have depicted the structure of the matrices $\mathbf{A_C}$ and $\mathbf{A_\Gamma}$ for the final solution of the coarse parts of the correlation functions, and $\mathbf{G}+I$ in the first and 10th NR iteration.

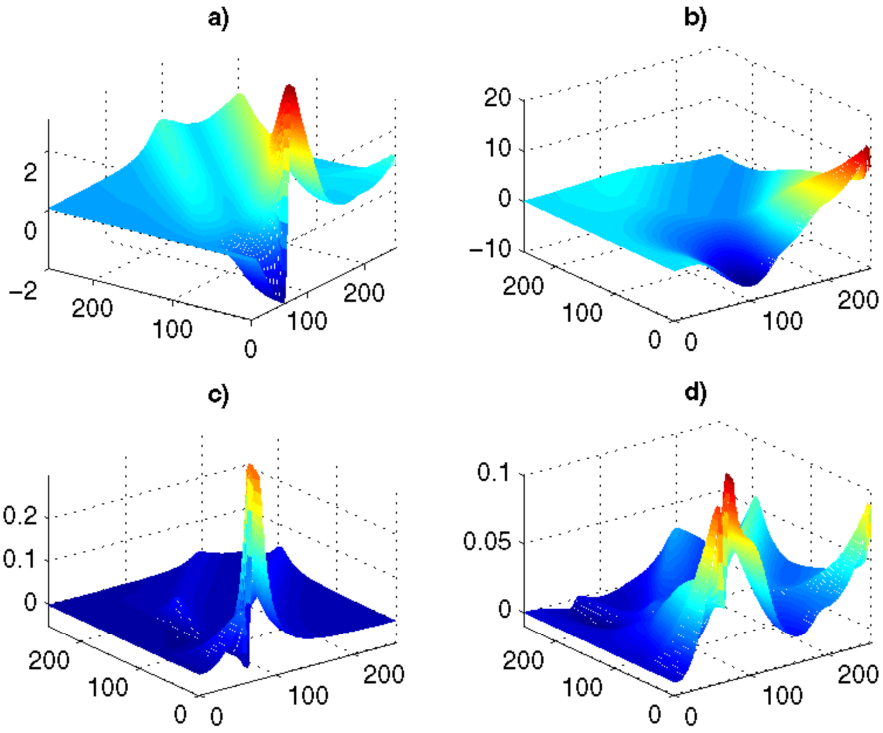

Figure 3.1: Structure of the matrices $\mathbf{A_C}$ - **a)**, $\mathbf{A_\Gamma}$ - **b)** for the final solution of the coarse parts of the correlation functions and $\mathbf{G}+I$ in the first - **c)** and 10th NR iteration - **d)**.

We have compared the efficiency of the ACA-compression of the matrices $\mathbf{A_C}$ and $\mathbf{A_\Gamma}$ with the compression rate of a Singular Value Decomposition (SVD). As a criterion we have chosen the parameter $\Delta := \frac{\|\mathbf{A_C}-\mathbf{A_C^{comp}}\|_2}{\|\mathbf{A_C}\|_2}$, where $\|\ldots\|_2$ denotes the spectral norm of a matrix and $\mathbf{A_C^{comp}}$ denotes the compressed version of the matrix $\mathbf{A_C}$. The same parameter was calculated for the matrix $\mathbf{A_\Gamma}$, respectively. The results are presented in Table 3.1, where we have shown the dependence of Δ from the rank of the compressed matrices.

We can see from Table 3.1 that we can roughly approximate the full matrices $\mathbf{A_C}, \mathbf{A_\Gamma} \in \mathbb{R}^{256\times256}$ up to an error of 10 percents by the low-rank ACA approximation with a rank in the range $10-15$. Since in this part of the algorithm we are seeking only the coarse solution, we consider this accuracy sufficient for our purposes.

Δ	SVD, $\mathbf{A_C}$	ACA, $\mathbf{A_C}$	SVD, $\mathbf{A_\Gamma}$	ACA, $\mathbf{A_\Gamma}$
10%	8	10	12	15
5%	10	20	18	20
1%	18	35	22	32
0.1%	36	60	35	58

Table 3.1: Comparison of the square norm discrepancy Δ between the full-format matrices $\mathbf{A_C}$, $\mathbf{A_\Gamma}$ and the compressed counterparts $\mathbf{A_C^{comp}}$, $\mathbf{A_\Gamma^{comp}}$ with a suitable rank for both the SVD and ACA compression scheme.

3.2 Comparative analysis of the speed of different algorithms

In Figure 3.2 we have depicted the matrices $\mathbf{A_C}$, $\mathbf{A_\Gamma}$ together with their compressed counterparts $\mathbf{A_C^{comp}}$ (rank 10) and $\mathbf{A_\Gamma^{comp}}$ (rank 15). Obviously, the compressed versions of the matrices are quite similar to the original ones.

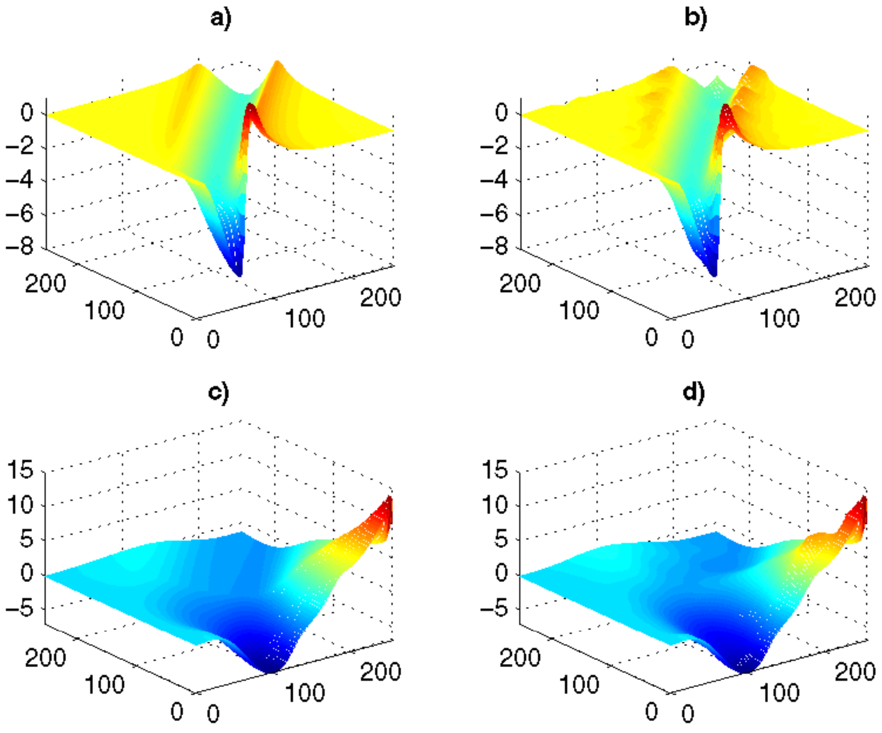

Figure 3.2: Structure of the original matrices $\mathbf{A_C}$ - **a)**, $\mathbf{A_\Gamma}$ - **c)** for the final solution of the coarse parts of the correlation functions and theirs compressed counterparts: $\mathbf{A_C^{comp}}$ (rank 10)- **b)** and $\mathbf{A_\Gamma^{comp}}$ (rank 15) - **d)**.
The OZ parameters are $\rho = 0.8$, $\sigma = 3$, and $\epsilon = 0.7$.

In order to illustrate the advantage of our scheme for the matrix multiplication and inversion compared to the standard full-format operations, we have measured the CPU time needed to find a solution of the OZ equation for the full-format algorithm and the ACA-based one with $\mathrm{rank}(\mathbf{A}_{\mathbf{C}}^{\mathbf{comp}}) = 10$ and $\mathrm{rank}(\mathbf{A}_{\mathbf{\Gamma}}^{\mathbf{comp}}) = 15$.

In the ACA Picard algorithm we have used the full ACA decomposition only for the first 10 iterations. Then we fixed the ACA cross for all following iterations. It means that after 10th iteration we do not recalculate the pivots i_k^*, j_k^* of ACA decomposition for any following iterations (see Remark 2.1).

The results are presented in Table 3.2. From that table we can see that the operation time needed for the ACA-based algorithms is one order of magnitude less than the time needed for the full format operations.

The results for the Newton scheme are shown in the Table 3.2. We can see that the Newton algorithm is up to ten times faster than the Picard iteration.

	FF, P	ACA, P	FF, N	ACA, N
Number of coarse iterations	198	179	5	5
Coarse iteration's CPU time [sec.]	68.0	10.5	6.4	3.4
Δ_c	$6 \cdot 10^{-3}$	$5 \cdot 10^{-2}$	$6 \cdot 10^{-3}$	$5 \cdot 10^{-2}$
Number of refining iterations	21	50	21	50
Refining iteration's CPU time [sec.]	1.5	2.0	1.5	2.0
Complete CPU time [sec.]	61.5	9.5	7.6	5.1

Table 3.2: Comparison of the CPU time needed to perform the full-format (FF) Picard (P) and Newton (N) method and the ACA-based counterparts. The symbol Δ_c denotes the mean square norm of the difference between the obtained coarse solution and the final solution after the refining iterations on the finest grid.

We use the Matlab program without caring about the detailed efficient management of the allocated memory. Therefore, we pay some extra cost for the loading operations which can be avoided, in principle.

In order to illustrute the real calculational cost we plotted in Figures 3.3 and 3.4 the loading time (dotted line) and calculation time for the different algorithms. We can see from these pictures that the ACA-based Newton scheme is much more efficient than the full-format Newton scheme.

4 Appendix A: Wavelet Theory

The fundamental theory behind wavelets is known as the Multi–Resolution Analysis (MRA). Most of the rigorous results and definitions from MRA are not usually required for practical applications. The only equations which are needed for the work described herein will be introduced in this section. We shall introduce the wavelet bases in a general way as biorthogonal wavelets. Moreover, we shall use the Discrete Wavelet Transform (DWT) technique [8, 30] to parameterize the RDFs. There is a good introduction to the wavelet techniques in Ref. [15]. We will also follow the style of that book henceforth. The multiresolution approach is based on the idea that the wavelet functions generate a hierarchical sequence of subspaces in the space of square–integrable functions over the real axis $L^2(\mathbb{R})$, which forms the MRA.

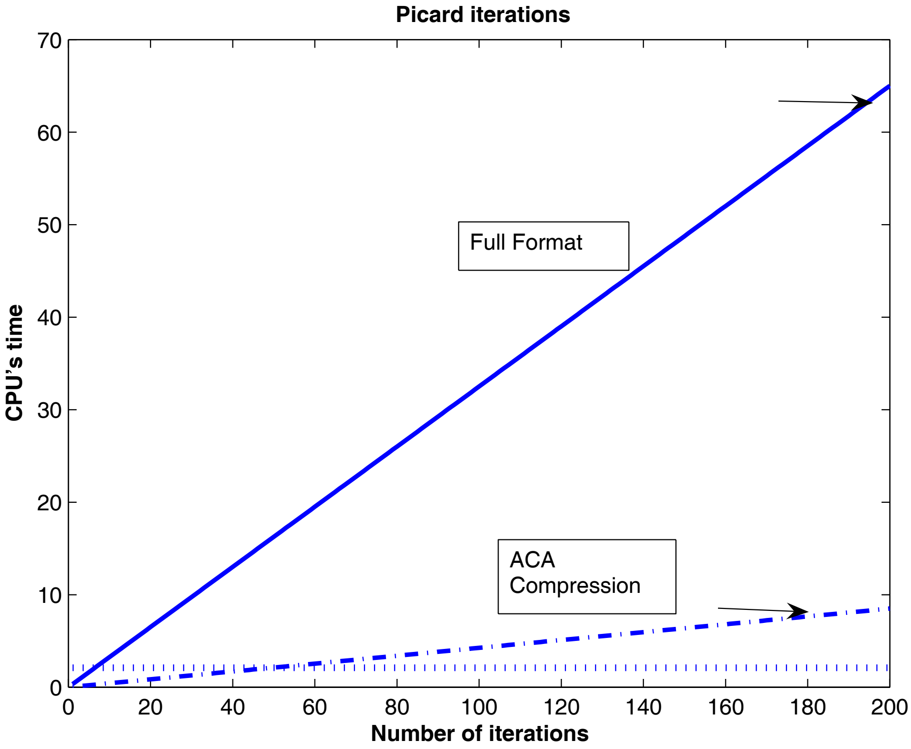

Figure 3.3: Comparison of loading CPU time (dotted line), calculational CPU time for the full-format Picard scheme (solid line) and ACA-based Picard scheme (dash-dotted line) with respect to the number of iterations. The arrows depict the time when the demanded accuracy is achieved.

The scaling functions $\varphi(r)$ and $\tilde{\varphi}(r)$ produce a biorthogonal MRA if they satisfy the following conditions.

(i) Translates of these functions with integers $\varphi_s = \varphi(r-s)$, $\tilde{\varphi}_s = \tilde{\varphi}(r-s)$, $s \in \mathbb{Z}$, are linearly independent and produce bases of the subspace $V_0 \subset L^2(\mathbb{R})$ and their dual counterpart $\tilde{V}_0 \subset L^2(\mathbb{R})$ correspondingly. Dyadic dilates of these functions $\varphi_{js} = \varphi(2^j r - s)$, $\tilde{\varphi}_{js} = \tilde{\varphi}(2^j r - s)$, $j \in \mathbb{Z}$, generate hierarchical sets of subspaces $\{V_j\}$ and $\{\tilde{V}_j\}$. Specifically, V_j contains all scaling functions on level j. This means that if a function $f(r)$ is contained in the space V_j, its integer translates have to be contained in the same space,

$$f(r) \in V_j \Leftrightarrow f(r-s) \in V_j, \qquad f(r) \in \tilde{V}_j \Leftrightarrow f(r-s) \in \tilde{V}_j, \quad s \in \mathbb{Z}.$$

Moreover, if a function $f(r)$ is contained in the space V_j, the dilated function $f(2r)$ has to be contained in the higher resolution space V_{j+1}

$$f(r) \in V_j \Leftrightarrow f(2r) \in V_{j+1}, \quad f(r) \in \tilde{V}_j \Leftrightarrow f(2r) \in \tilde{V}_{j+1}, \quad j \in \mathbb{Z}.$$

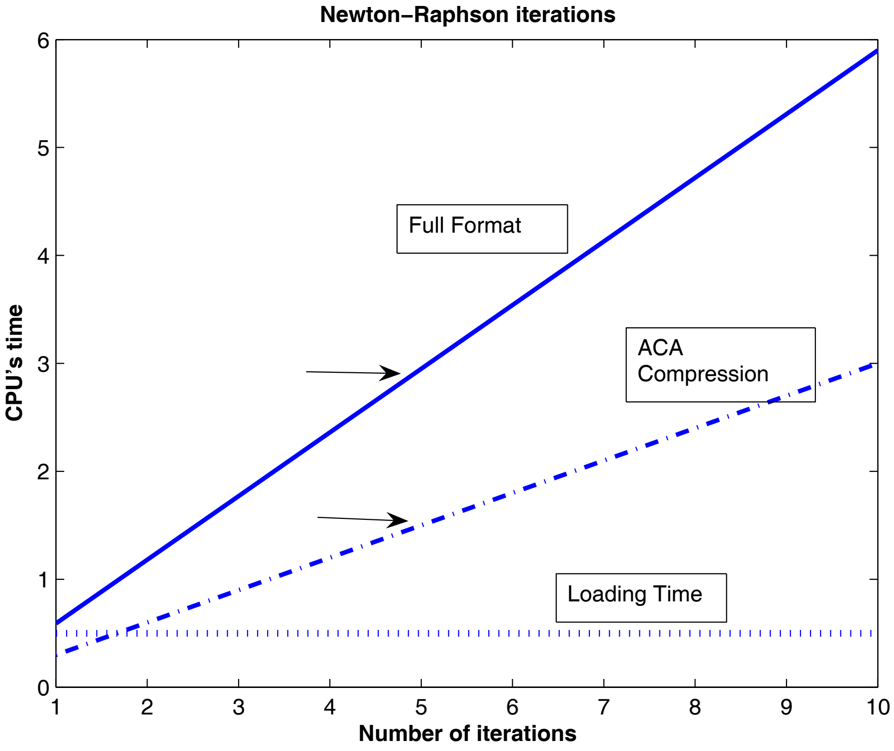

Figure 3.4: Comparison of loading CPU time (dotted line), calculational CPU time for the full-format N-scheme (solid line) and ACA-based N-scheme (dash-dotted line) with respect to the number of iterations. The arrows depict the time when the demanded accuracy is achieved.

(ii) The scaling function spaces and their dual counterpart satisfy

$$V_j \subset V_{j+1}, \qquad \bigcup_{j=-\infty}^{\infty} V_j \text{ is dense in } L^2(\mathbb{R}), \qquad \bigcap_{j=-\infty}^{\infty} V_j = 0, \tag{4.3}$$
$$\tilde{V}_j \subset \tilde{V}_{j+1}, \qquad \bigcup_{j=-\infty}^{\infty} \tilde{V}_j \text{ is dense in } L^2(\mathbb{R}), \qquad \bigcap_{j=-\infty}^{\infty} \tilde{V}_j = 0.$$

(iii) The sets of functions $\varphi_{js}(r)$ and $\tilde{\varphi}_{js}(r)$ are biorthogonal to each other. It means that for any $s, s' \in \mathbb{Z}$:

$$\int \varphi_{js}(r)\tilde{\varphi}_{js'}(r)\,dr = \delta_{ss'}.$$

(iv) There is a wavelet function $\psi(r)$ and its dual wavelet function $\tilde{\psi}(r)$ such that their integer translates $\psi_s(r) = \psi(r-s)$, $\tilde{\psi}_s(r) = \tilde{\psi}(r-s)$, and dyadic dilates $\psi_{js} = \psi(2^j r - s)$, $\tilde{\psi}_{js} = \tilde{\psi}(2^j r - s)$, form subspaces W_j and $\tilde{W}_j$ which are complementary to V_j and $\tilde{V}_j$ so that:

$$V_{j+1} = V_j \oplus W_j, \qquad \tilde{V}_{j+1} = \tilde{V}_j \oplus \tilde{W}_j, \qquad \tilde{W}_j \perp V_j, \qquad \tilde{V}_j \perp W_j. \tag{4.5}$$

(v) From the above relations it follows that $L^2(\mathbb{R})$ can be decomposed into the approximation space V_{j_0} and the sum of the detail spaces W_j of higher resolutions $j \geqslant j_0$:

$$L^2(\mathbb{R}) = V_{j_0} \bigoplus_{j \geq j_0}^{\infty} W_j, \tag{4.6}$$

where $j_0 \in \mathbb{Z}$ is a chosen level of resolution. This means that any square–integrable function $f(r)$ can be represented as a sum of linear combinations of the scaling functions $\{\varphi_{j_0}\}$ at a chosen resolution $j = j_0$ and the wavelet functions $\{\psi_j\}$ at all finer resolutions $j \geq j_0$. This can be written as

$$f(r) = \sum_{s=-\infty}^{\infty} a_{j_0 s}\varphi_{j_0 s}(r) + \sum_{j \geq j_0}^{\infty} \sum_{s=-\infty}^{\infty} d_{js}\psi_{js}(r), \tag{4.7}$$

where the coefficients $\{a_{j_0 s}\}$ and $\{d_{js}\}$ are obtained as the scalar products with the appropriate dual basis functions,

$$a_{js} = \int f(r)\tilde{\varphi}_{js}(r)\, dr, \quad d_{js} = \int f(r)\tilde{\psi}_{js}(r)\, dr. \tag{4.8}$$

The later equation defines the Discrete Wavelet Transform (DWT).

As $\varphi(r) \in V_0$ and $V_0 \subset V_1$, $\tilde{\varphi}(r) \in \tilde{V}_0$ and $\tilde{V}_0 \subset \tilde{V}_1$, we can express $\varphi(r)$ (as well as $\tilde{\varphi}(r)$) as a linear combination of the basis functions in V_1 ($\tilde{V}_1$):

$$\varphi(r) = \sum_s h_s \varphi(2r - s), \qquad \tilde{\varphi}(r) = \sum_s \tilde{h}_s \tilde{\varphi}(2r - s). \tag{4.9}$$

This equation is called the *dilation equation*. Similarly, $\psi(r)$ and $\tilde{\psi}(r)$ must satisfy a *wavelet dilation equation*:

$$\psi(r) = \sum_s w_s \varphi(2r - s), \qquad \tilde{\psi}(r) = \sum_s \tilde{w}_s \tilde{\varphi}(2r - s). \tag{4.10}$$

The above sets of coefficients are usually called "filters" and they are completely sufficient in order to describe a chosen wavelet basis because there are several procedures on how to build up numerical values of the wavelet functions from the set of filters [30, 8, 15]. We should emphasize here that there are no analytic expressions for biorthogonal (orthogonal) wavelets with a finite support[1]. However, these basis functions are determined in terms of their filter coefficients only.

The scaling functions and the wavelets have a finite support only in the case of a finite number of the coefficients h_s and w_s. Due to their biorthogonal nature these functions satisfy

[1] This is true except of the simplest basis, Haar basis, which is constructed from piecewise constant functions [8].

the relations:

$$
\begin{aligned}
\int \varphi_{ja}(r)\tilde{\varphi}_{jb}(r)\,dr &= \delta_{ab}, \\
\int \varphi_{ja}(r)\tilde{\psi}_{lb}(r)\,dr &= 0, \quad (l \geq j), \\
\int \tilde{\varphi}_{ja}(r)\psi_{lb}(r)\,dr &= 0, \quad (l \geq j), \\
\int \psi_{ja}(r)\tilde{\psi}_{lb}(r)\,dr &= \delta_{jl}\delta_{ab},
\end{aligned} \tag{4.11}
$$

for any integer j, l, a, b such that $l \geq j$. If the pairs of the decomposition functions $\{\tilde{\varphi}, \tilde{\psi}\}$ and the reconstruction functions $\{\varphi, \psi\}$ are identical, the transform is called 'orthogonal wavelet transform'. Otherwise we shall talk about a more general 'biorthogonal wavelet transform'.

In the expansion (4.7) the first term gives a 'coarse' approximation for $f(r)$ at the resolution j_0 and the second term gives a sequence of successive 'details' (see also (2.1)). In practice, we actually do not need to use the infinite number of resolutions. Therefore, the sequence of details is cut–off at an appropriate resolution $j_{\max}$. Since all functions used in numerical work are given in a finite interval, the sequence of translates $\{s\}$ has also a finite number of terms S_j, which is reduced dyadically from level to level.

Importantly, the explicit form of the basis functions is not required if we are using (bi)orthogonal wavelets with a finite support and a dyadic set of scales j. Then the coefficients in Eq. (4.8) (see also (2.2)) can be calculated by the Fast Wavelet Transform (FWT) algorithm [30, 31, 15]. The main idea of this algorithm is that a set of (bi)orthogonal discrete filters at consequently dilated scales is used for the multi–resolution analysis of a signal. As a result, to calculate the approximating coefficients, the convolution of the signal and the relevant filter is only required for each scale, and the latter can be easily obtained.

By choosing relevant basis functions and scales we can nullify most of the coefficients $\{a\}$ and $\{d\}$, however controlling the L^2-error since DWT satisfies the Parseval's identity [8]. Therefore, the function of interest can be reconstructed with the use of only a few nonzero coefficients without any significant loss of accuracy, making the total number of the approximating coefficients rather small. This feature of the of wavelet approximation is widely used in processing of signals and images, the data for which should be compressed with minimal losses [11]. In statistical mechanics of disordered media the wavelets are used for the approximations of different kinds of the correlation functions [3, 13, 35].

Acknowledgement. The authors would like to thank Prof. W. Hackbusch for useful discussions and for the support provided. Discussions with Prof. G. Chuev are gratefully acknowledged.

References

[1] M. Bebendorf and S. Rjasanov. Adaptive Low-Rank Approximation of Collocation Matrices. *Computing*, 70(1):1–24, 2003.

[2] G. Beylkin, R. Coifman, and V. Rokhlin. Fast wavelet transforms and numerical algorithms. *Communications On Pure And Applied Mathematics*, 44(2):141–183, 1991.

[3] G. N. Chuev and M. V. Fedorov. Wavelet treatment of radial distribution functions of solutes. *Physical Review E*, 68(2):027702, 2003.

[4] G. N Chuev and M. V. Fedorov. Wavelet algorithm for solving integral equations of molecular liquids. a test for the reference interaction site model. *Journal of Computational Chemistry*, 5(11):1369–1377, 2004.

[5] G. N. Chuev and M. V. Fedorov. Wavelet treatment of structure and thermodynamics of simple liquids. *Journal of Chemical Physics*, 120(3):1191–1196, 2004.

[6] Matlab ®. www.mathworks.com.

[7] B. Daguanno and R. Klein. Structural effects of polydispersity in charged colloidal dispersions. *Journal of the Chemical Society - Faraday Transactions*, 87(3):379–390, 1991.

[8] I. Daubechies. *Ten Lectures on Wavelets*, volume 61 of *CBMS/NSF Series in Applied Mathematics*. SIAM, Philadelphia, 1992.

[9] G. Deslauriers and S. Dubuc. Symmetric iterative interpolation processes. *Constructive Approximation*, 5(1):49–68, 1989.

[10] R. A. DeVore, S. V. Konyagin, and V. N. Temlyakov. Hyperbolic wavelet approximation. *Constructive Approximation*, 14(1):1–26, 1998.

[11] D. L. Donoho. De-noising by soft-thresholding. *IEEE Transactions On Information Theory*, 41(3):613–627, 1995.

[12] M. V. Fedorov, G. N. Chuev, C. T. Kelley, and B. M. Pettitt. Multilevel wavelet solver for the ornstein-zernike equation. *Journal of Computational Physics*, 2005. Submitted.

[13] M. V. Fedorov, G. N. Chuev, Yu. A. Kuznetsov, and E. G. Timoshenko. Wavelet treatment of the intrachain correlation functions of homopolymers in dilute solutions wavelet treatment of the intrachain correlation functions of homopolymers in dilute solutions. *Physical Review E*, 70(5):051803, 2004.

[14] M. J. Gillan. New method of solving the liquid structure integral-equations. *Molecular Physics*, 38(6):1781–1794, 1979.

[15] S. Goedecker. *Wavelets and Their Application for the Solution of Partial Differential Equations in Physics.* Presses Polytechniques et Universitaires Romandes, Lausanne, 1998.

[16] G. H. Golub and C. F. van Loan. *Matrix Computations.* Johns Hopkins University Press, 1996.

[17] L. Grasedyck and W. Hackbusch. Construction and arithmetics of $\mathcal{H}$-matrices. *Computing*, 70:295–334, 2003.

[18] W. Hackbusch. *Multi-grid methods and Applications.* Springer-Verlag, Berlin, 1985.

[19] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$ -matrices. *Computing*, 62:89–108, 1999.

[20] W. Hackbusch and B. N. Khoromskij. A sparse $\mathcal{H}$-matrix arithmetic. Part II: Application to multi-dimensional problems. *Computing*, 64:21–47, 2000.

[21] W. Hackbusch and B. N. Khoromskij. Hierarchical kronecker tensor-product approximation to a class of nonlocal operators in high dimensions. Preprint 16, Max-Planck-Institut für Mathematik in den Naturwissenschaften, Leipzig, 2004. Submitted.

[22] J. P. Hansen and G. Zerah. How reliable are integral-equations for the pair structure of binary fluid mixture. *Physics Letters A*, 108(5–6):277–280, 1985.

[23] Jean Pierre Hansen and Ian R. McDonal. *Theory of Simple Liquids.* Elsevier Science & Technology Books, first edition, Jan 1990.

[24] T. Ichiye and A. D. J. Haymet. Structural effects of polydispersity in charged colloidal dispersions. *Journal of Chemical Physics*, 93(12):8954–8962, 1990.

[25] C. T. Kelley. *Iterative methods for linear and nonlinear equations*, volume 16 of *Frontiers in Applied Mathematics.* SIAM, Philadelphia, 1995.

[26] C. T. Kelley and B. M. Pettitt. A fast solver for the ornstein-zernike equations. *Journal of Computational Physics*, 197(2):491–501, 2004.

[27] S. Labik, A. Malijevsky, and P. Vonka. A rapidly convergent method of solving the oz equation. *Molecular Physics*, 56(3):709–715, 1985.

[28] S. Labik, R. Pospiil, A. Malijevsky, and W. R. Smith. An efficient gauss-newton-like method for the numerical solution of the ornstein-zernike integral equation for a class of fluid models. *Journal of Computational Physics*, 115(1):12–21, 1994.

[29] L. G. MacDowell, M. Muller, C. Vega, and K. Binder. Equation of state and critical behavior of polymer models: A quantitative comparison between wertheim's thermodynamic perturbation theory and computer simulations. *Journal of Chemical Physics*, 113(1):419–433, 2000.

[30] S. G. Mallat. *A Wavelet Tour of Signal Processing.* Academic press, San Diego, 1999.

[31] Y. Meyer. *Wavelets: Algorithms and Applications.* SIAM, Philadelphia, 1993.

[32] P. A. Monson and G. P. Morriss. Recent progress in the statistical-mechanics of interaction site fluids. *Advances In Chemical Physics*, 77:451–550, 1990.

[33] R. K. Pandey and D. N. Tripathi. Analytical solution of the ornstein-zernike equation for the structure factor of ordered plasmas using sogami-ise potential. *Journal of the Physical Society of Japan*, 73(7):1748–1753, 2004.

[34] B. M. Pettitt and P. J. Rossky. Alkali-halides in water - ion solvent correlations and ion ion potentials of mean force at infinite dilution. *Journal of ChemicaL Physics*, 84(10):5836–5834, 1986.

[35] A. B. Romeo, C. Horellou, and J. Bergh. N-body simulations with two-orders-of-magnitude higher performance using wavelets. *Monthly Notices of the Royal Astronomical Society*, 342(2):337–344, 2003.

[36] C. F. Strnadl and G. Kahl. The electronic density of states in liquids: computer simulation versus integral-equation approach. *Journal of Physics: Condensed Matter*, 5(37):6801–6818, 1993.

[37] H. Xu and J. P. Hansen. Density-functional theory of pair correlations in metallic hydrogen. *Physical Review E*, 57(1):211–223, 1998.