

SANDIA REPORT

SAND2006-2099
Unlimited Release
Printed April 2006

A Schur Complement Based Approach to Solving Density Functional Theories for Inhomogeneous Fluids on Parallel Computers

Michael A. Heroux, Laura J. D. Frink, Andrew G. Salinger

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94-AL85000.

Approved for public release; further dissemination unlimited.



Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: <http://www.doe.gov/bridge>

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online ordering: <http://www.ntis.gov/ordering.htm>



SAND2006-2099
Unlimited Release
Printed April 2006

A Schur Complement Based Approach to Solving Density Functional Theories for Inhomogeneous Fluids on Parallel Computers

Michael A. Heroux
Computational Math/Algorithms Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-1110
maherou@sandia.gov

Laura J. D. Frink
Computational Biosciences Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-1413
ljfrink@sandia.gov

Andrew G. Salinger
Applied Computational Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-1111
agsalin@sandia.gov

Abstract

Numerical formulations of density functional theories for inhomogeneous fluids (Fluid-DFTs) require the solution of large systems of equations with many degrees of freedom (DOFs) per node on a computational grid. Historically solvers for these problems have used simple Picard iterations across DOFs or, more recently, fully-coupled general algebraic techniques.

In this paper we look at Fluid-DFTs from a fresh perspective, retaining a fully-coupled formulation but segregating variables for the purposed of introducing Schur complement formulations and specialized preconditioners. By viewing Fluid-DFTs from this perspective, we develop a mathematical framework and a collection of solution algorithms that have a dramatic impact on the robustness, performance and scalability of the implicit equations generated by Fluid-DFTs.

Acknowledgment

The authors thank the MICS program under the Department of Energy.

Intentionally Left Blank

Contents

1	Introduction	11
2	General Mathematical Framework	12
3	Discrete Formulations	14
4	A New Block Matrix Formulation and Solution Method	16
5	The new solver applied to two Fluid-DFTs	17
5.1	Hard-spheres	17
5.2	Polymer Problems	18
6	Computational Results	20
6.1	Solver Approaches	20
6.2	Hard sphere fluids in 2-dimensions	21
6.3	3D Hard spheres	23
6.4	2D Polymer	25
6.5	3D Polymer	26
6.6	Results Summary	26
7	Conclusions	30
	References	33

Appendix

A	Two specific Fluid-DFTs	35
----------	--------------------------------------	-----------

Figures

1	Stencil support for two mesh densities for integrating the area of the circle. As the mesh is refined, the stencil density increases. This corresponds to more expensive computation of the integrals and more nonzeros per row of the Jacobian matrix.	15
2	Fluid density ($\rho\sigma^3$) distribution in the vicinity of 3 nanocylinders.....	23
3	Fluid density ($\rho\sigma^3$) distribution for a hard sphere fluid in a finite length nanopore.	25
4	Density ($\rho\sigma^3$) distribution for a 10-mer homopolymer near two nanocylinders. The computational domain was 1/4 of this image with reflective boundaries at $x = 0$ and $y = 0$	25

- 5 One slice (at $y = 0$) through a 3-dimensional computational volume. The color contours show density ($\rho\sigma^3$) distributions for lipid tail beads (A), lipid head beads (B), and solvent (C) for a lipid bilayer assembly sandwiched between planar arrays of large spheres. The computational domain is 1/4 of the domain shown in the figure and utilized reflective boundary conditions on all edges. The legend shows the contour scale for all three figures. Densities less than $\rho\sigma^3 = 0.01$ are blanked to white. 28

Tables

- 1 Results for a 2D Hard sphere test problem. Note that every run in the table solved with 10 nonlinear iterations. In the top part of the table, the columns are: the number of processors used for the calculation, the average number of linear iterations per nonlinear iteration, the solve time per nonlinear iteration, the speedup relative to the single processor time, and the ratio of solve times from the old to the new algorithm. In the bottom part of the table we consider scaling with mesh density. The first column contains the mesh spacing, and the 6th column contain timings relative to the $\Delta x = \sigma/5$ result. All data in the lower part of the table were generated on 64 processors. All data in the upper part of the table were generated with a mesh spacing of $\Delta x = \sigma/10$ 22
- 2 Results for a 3D Hard sphere test problem. The second column now contains the number of nonlinear iterations needed to solve the problem. For a description of all other columns see the table 1 caption. All data in the upper part of the table were generated with a mesh spacing of $\Delta x = \sigma/5$, and with a bulk density of $\rho\sigma^3 = 0.75$. All data in the lower part of the table were generated on 128 processors with a bulk density of $\rho\sigma^3 = 0.6$. The data with an * indicates that complete convergence was not obtained and timings are extrapolated based on the number of completed nonlinear iterations in the parenthesis. 24
- 3 Results for a 2D polymer problem. The top part of the table shows scaling with number of processors. The middle part of the table shows scaling with mesh density. The bottom section of the table shows scaling with polymer chain length. The column descriptions can be found in the caption of Table 1. All data in the upper part of the table were generated with a mesh spacing of $\Delta x = \sigma/10$ and for polymer chains of length $N_{seg} = 10$. All data in middle part of the table were generated on 128 processors for polymer chains of length $N_{seg} = 10$. All data in the bottom part of the table were generated on 32 processors with $\Delta x = \sigma/10$. The data with an * indicates that complete convergence was not obtained and timings are extrapolated based on the number of completed nonlinear iterations in the parenthesis. 27

- 4 Results for a 3D polymer test problem. Each case require 10 nonlinear iterations for a solution. For a description of all other columns see the table 1 caption. All data in the upper part of the table were generated with a mesh spacing of $\Delta x = \sigma/5$. Note that the data flagged with a * is a case where complete convergence was not obtained. The reported data reflect timings obtained for one complete nonlinear iteration and the associated linear solution. The total time was computed assuming a consistent 10 nonlinear iterations independent of number of processors. 29

Intentionally Left Blank

A Schur Complement Based Approach to Solving Density Functional Theories for Inhomogeneous Fluids on Parallel Computers

1 Introduction

Multiple degrees of freedom (DOFs) per node properties are common to many numerical applications. Segregated solvers, which attempt to view each DOF across the grid as a sub-problem within the larger fully-coupled problem, have been successfully used in many problem domains. Similarly Schur complement methods, which formally eliminate variables by block Gaussian elimination can reduce the complexity and cost of solution. In this paper we present a combination of these two classes of algorithms applied to Density Functional Theories for inhomogeneous fluids (Fluid-DFTs). We will show that viewing Fluid-DFTs from a segregated variable perspective yields a rich structure that can be exploited in the development of robust, scalable solution methods.

Density functional theories (DFTs) have been tremendously successful in treating a variety of systems at many length scales. In all cases, the fundamental problem is to predict the structure of an inhomogeneous fluid as captured by a density distribution, $\rho(\mathbf{r})$ [39]. At the smallest length scale the most well known application of DFT is to predict the structure of quantum mechanical systems [26]. These quantum mechanical DFTs (QM-DFTs) are used to predict the structure of an electron *fluid* in an external field produced by fixed nuclei. Using a similar mathematical construct but with non-exact density functionals [16], the structure of atomic [27, 28], molecular [21, 1], and polymer fluids [3, 40, 41, 36] can be computed. Fluid inhomogeneities can result from surfaces (e.g. planar interfaces, porous materials, or large geometrically complex macromolecules [13, 38, 24, 12]) or from competing intramolecular and intermolecular interactions that can lead to self-assembly [15, 9, 22]. Mesoscale-DFTs have also been developed for colloidal fluids and biological macromolecules [5, 6]. These mesoscale-DFTs are very similar to Fluids-DFTs, but are based on coarsened models or potentials (e.g. the solvent averaged Yukawa potential).

Our previous efforts to develop numerical methods for Fluids-DFT began with the development of a Newton's method real space approach that could be solved using parallel iterative solvers optimized for large distributed memory parallel computers[10, 11, 13]. While this code was robust and in fact applied to a variety of systems of unprecedented complexity, it was still too computationally intensive to allow for routine calculation on 3-dimensional systems. Given those limitations, we then developed a matrix free method

with fast Fourier transforms (FFTs) to compute certain convolutions in the theory[33]. This approach led to a code that could be applied to 3-dimensional problems using very modest computer resources (single processor workstations). However the FFTs limit the application space of the DFTs to cases with periodic boundary conditions, and since no matrix is stored, matrix based preconditioning methods cannot be applied to converge difficult nonlinear problems.

In the remainder of this paper we present a new approach to solving Fluid-DFTs using a real space method based on segregated Schur complement techniques and demonstrate the impact of this approach on our ability to solve large, complex problems. Section 2 presents the general mathematical framework of fluids-DFTs. Section 3 discusses the complexities of the discrete formulation of DFTs with a focus on properties that can be exploited in the design of optimal algorithms. Section 4 contains a discussion of our new block equation framework, and section 5 discusses the block framework for two particular fluid-DFTs. Section 6 presents results for our new solver algorithms, comparing timing and robustness to our previous approaches. We find that the new method is a considerable improvement over generic methods developed for PDEs.

2 General Mathematical Framework

In Fluid-DFTs a free energy functional, Ω , depends on a set of critical fields, ψ , in the problem of interest, $\Omega[\{\psi_i(\mathbf{r})\}]$. The minimization of this free energy with respect to all fields results in a system of residual equations to be solved. Generally any of these residual equations can be written

$$\frac{\delta\Omega}{\delta\psi_i} = 0 = I(\psi(\mathbf{r})) + D(\psi(\mathbf{r})) + F(\psi(\mathbf{r})) \quad (1)$$

where I is a general integral operator, D is a general differential operator, and F is some function of the fields involving neither integral or differential operators. For example a problem with a charged atomistic fluid model will have two critical fields, the fluid density and the electrostatic potential. The minimization of free energy with respect to the density leads to an Euler-Lagrange equation. The minimization of the free energy with respect to the electrostatic potential leads to Poisson's equation. These two equations must be solved as a coupled set to find the distribution of a charged fluid near an interface of interest.

In many cases, some contributions to the free energy functionals are not local. Rather, the free energies are written as functionals of both the critical fields and some nonlocal variables, n with $\Omega[\psi, \{n_\gamma[\psi]\}]$. The nonlocal variables are themselves functionals of the critical fields of interest. If the free energy functional contains an integral operator term,

$$\Omega_I[\{n_\gamma[\psi]\}] = \int G(n) d\mathbf{r} \quad (2)$$

dependent on these nonlocal variables, the corresponding contribution to Eq. 1 will be

$$\frac{\delta\Omega_I}{\delta\psi_i(\mathbf{r})} = \int \sum_{\gamma} \frac{\partial G}{\partial n_{\gamma}}(\mathbf{r}') \frac{\delta n(\mathbf{r}')}{\delta\psi_i(\mathbf{r})} d\mathbf{r}'. \quad (3)$$

While the free energy minimization is always defined by the general expression in Eq.1, the formulation of the matrix problem may proceed in two ways [10].

The first approach is to consider a solution vector that contains only the critical variables. The Jacobian (or really Hessian) used in the matrix problem is then

$$J_{ij}(\mathbf{r}, \mathbf{r}') = \frac{\delta^2\Omega}{\delta\psi_i(\mathbf{r})\delta\psi_j(\mathbf{r}')} = \int \sum_{\epsilon} \sum_{\gamma} \frac{\partial^2 G}{\partial n_{\gamma}\partial n_{\epsilon}}(\mathbf{r}'') \frac{\delta n(\mathbf{r}'')}{\delta\psi_i(\mathbf{r})} \frac{\delta n(\mathbf{r}'')}{\delta\psi_j(\mathbf{r}')} d\mathbf{r}''. \quad (4)$$

Often these nonlocal variables are simple linear functionals of the critical fields defined as

$$n(\mathbf{r}) = \int w(\mathbf{r}, \mathbf{r}') \psi(\mathbf{r}') d\mathbf{r}' \quad (5)$$

with $w(\mathbf{r}, \mathbf{r}') = \delta(|\mathbf{r} - \mathbf{r}'| - R)$ or $w(\mathbf{r}, \mathbf{r}') = \theta(|\mathbf{r} - \mathbf{r}'| - R)$. where R is some characteristic dimension (a particle size or a bond length). The Jacobian can then be written

$$J_{ij}(\mathbf{r}, \mathbf{r}') = \frac{\delta^2\Omega}{\delta\psi_i(\mathbf{r})\delta\psi_j(\mathbf{r}')} = \int \sum_{\epsilon} \sum_{\gamma} \frac{\partial^2 G}{\partial n_{\gamma}\partial n_{\epsilon}}(\mathbf{r}'') w_{\gamma}(\mathbf{r}'', \mathbf{r}) w_{\epsilon}(\mathbf{r}'', \mathbf{r}') d\mathbf{r}''. \quad (6)$$

Clearly in order to compute a Jacobian entry with this structure will require a second order (N^2) operation in order to locate the intersection of the weight functions $w_{\gamma}(\mathbf{r}'', \mathbf{r})$ and $w_{\epsilon}(\mathbf{r}'', \mathbf{r}')$. In some Fluids-DFTs the definition of nonlocal variables can be even more complex resulting in multiple integrals for each Jacobian entry. In real space, this approach leads to matrix coefficient calculations that range from time consuming to completely impractical.

One solution is to pursue an FFT treatment of convolutions [33]. Another approach is to formulate the real space matrix problem in terms of not only the critical fields, ψ but also all of the nonlocal density variable, n [10]. This approach is akin to the transformation of a higher order PDE to a system of first order PDEs by introduction of additional variables in the problem. While we had implemented this approach some time ago, it is the current work presented here that demonstrates the power of this approach both for reducing complexity of many variants of DFT, and for leveraging segregated modeling strategies to solve DFTs efficiently. Taking a case where the only critical field is the density ρ and where there are

linear nonlocal density variables as defined above, the contributions of the nonlocal free energy term to the extended matrix problem written in block form is

$$\begin{bmatrix} -I & w_\gamma(\mathbf{r}, \mathbf{r}') \\ \sum_\gamma \frac{\partial^2 G}{\partial n_\gamma \partial n_\epsilon}(\mathbf{r}) & 0 \end{bmatrix} \begin{bmatrix} \Delta n_\gamma \\ \Delta \bar{\rho} \end{bmatrix} = - \begin{bmatrix} R_{NL}(\mathbf{r}) \\ R_{EL}(\mathbf{r}) \end{bmatrix}, \quad (7)$$

where \mathbf{r} is a row of the matrix, \mathbf{r}' is a column of the matrix, R_{NL} is the residual for the equation that defines n (Eq.5), and R_{EL} is the residual of the Euler-Lagrange equation (Eq.3). Note that there are no integrals in the Jacobian, and so the complexity of filling the matrix has been considerably reduced. We will refer to Eq. 7 as the first order formulation of Fluid-DFTs. In contrast Eq. 6 is an example of a higher-order formulation.

Of course there can be many more terms defining the free energy functional that must also be considered. However, we find Fluid-DFTs can very often be formulated using a 2×2 block matrix motif where all nonlocal ancillary variables (and their variations with one another and the fields) are defined in the top part of the matrix, and where the critical variables are defined in the lower part of the matrix. The result is often a very simple A_{11} block which can be exploited for fast computations as we demonstrate below.

3 Discrete Formulations

As was mentioned in the introductory section, we have previously developed a code for solving Fluid-DFTs on large distributed memory computers. That code was based on a discretization using a uniform structured grid so that the numerical integration stencils for all spatial integrals could be pre-computed on a reference grid and then applied anywhere on the grid [7, 8]. Linear interpolation was used for the critical fields between the mesh points, and the equations were discretized using collocation at the mesh points.

The discretized equations were previously solved using a fully-coupled Newton method with an analytic Jacobian and algebraic preconditioned Krylov methods. Convergence with these methods is much improved over Picard iteration reducing nonlinear iterations from $O(100 - 1000)$ to $O(10)$. However, these linear solvers were designed for PDE applications, and are far from optimal for Fluid-DFTs. They sometimes require very expensive preconditioners in order to converge. As a concrete example of the failure of the PDE solvers, we have previously documented that while the extended variable formulation of the matrix problem reduces the complexity of the fill, the overall performance of the code was not greatly impacted because the solves of these larger more extended systems was more difficult from a solver perspective [10].

The ineffectiveness of these algorithms stems primarily from the fact that the systems of equations are very dissimilar to PDEs. Key properties of the Fluid-DFTs systems of equations when compared to PDEs that lead to failure of standard algorithms are:

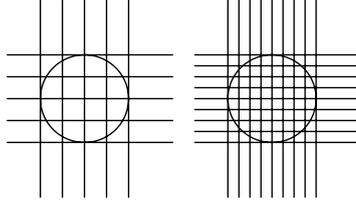


Figure 1. Stencil support for two mesh densities for integrating the area of the circle. As the mesh is refined, the stencil density increases. This corresponds to more expensive computation of the integrals and more nonzeros per row of the Jacobian matrix.

1. Inter-physics vs. Inter-nodal Coupling: While PDEs have a strong spatial coupling where the equation for a given variable at a given discrete mesh node involves interaction with the same variable at nearby nodes, Fluid-DFTs may have remarkably little of this kind of coupling¹.
2. Stencil Support as a Function of Physical Parameter: Unlike PDEs, whose stencils are typically independent of mesh node spacing (e.g., 9 point FEM stencil in 2D), the stencils in Fluid-DFTs are determined by some characteristic physical parameter in the system (e.g bead size, bond length). As the mesh is refined, more nodes fall within the range of the integral, and higher fidelity simulations result in larger problem dimensions and many more nonzeros per matrix row (see Figure 1).
3. Large number of DOFs per Node: Most PDE problems have just a handful of degrees of freedom (DOFs) in the global system (with the exception of reacting flow problems). First order formulations of Fluid-DFTs, on the other hand, typically have more than 10 DOFs per node and may have 50 or more. Furthermore, the stencils for each DOF can vary greatly in range and complexity.

Since standard algebraic preconditioners, including multi-level methods, incomplete factorizations and relaxation methods all have a bias toward inter-nodal coupling, these methods may all miss the mark. Similarly, load balancing tools largely have the same bias, and will need to be revisited for Fluid-DFTs. The scaling of stencils with mesh density in Fluid-DFTs suggests that standard preconditioners, sparse matrix computations and communication patterns become increasingly inappropriate as mesh fidelity increases. Thus it is critical that a general framework be developed that is more suitable to Fluid-DFTs. Because there are a wide variety of Fluid-DFTs in current use in the physics community (various methods for similar fluids, and a wide range of fluids from atomistic neutral particles to polarizable polymers), the framework must be quite flexible and general. We present our current efforts to develop such a framework in Section 4.

¹Exceptions to this property occur when differential operators are part of the system of Euler-Lagrange equations as with charged fluids[34, 13] or diffusing fluids[14]

4 A New Block Matrix Formulation and Solution Method

The basic framework for all of our solver algorithms reflects the importance of inter-physics coupling in the first order formulation of the Fluid-DFTs described in Section 2. This physics coupling led us to a physics-based block matrix formulation in order to partition critical and nonlocal ancillary variables. The idea is to partition the data into blocks that can be optimally managed or solved. The general 2×2 block matrix is

$$\left(\begin{array}{ccc|ccc} A_{11}^{11} & \cdots & A_{11}^{1j} & A_{12}^{1,j+1} & \cdots & A_{12}^{1k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ A_{11}^{j1} & \cdots & A_{11}^{jj} & A_{11}^{j1} & \cdots & A_{11}^{jj} \\ \hline A_{21}^{j+1,1} & \cdots & A_{21}^{j+1,j} & A_{22}^{j+1,j+1} & \cdots & A_{22}^{j+1,k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ A_{21}^{k1} & \cdots & A_{21}^{kj} & A_{22}^{k,j+1} & \cdots & A_{22}^{kk} \end{array} \right) \begin{pmatrix} x_1^1 \\ \vdots \\ x_1^j \\ \hline x_2^{j+1} \\ \vdots \\ x_2^k \end{pmatrix} = \begin{pmatrix} b_1^1 \\ \vdots \\ b_1^j \\ \hline b_2^{j+1} \\ \vdots \\ b_2^k \end{pmatrix} \quad (8)$$

where k is the number of DOFs tracked per node. The superscript (p, q) denotes the block of coefficients generated by DOF p interactions with DOF q . The subscripts and partition lines impose a coarser partitioning of the matrix into a 2-by-2 block system that will be used with a Schur complement approach. We denote by A_{11} , A_{12} , A_{21} and A_{22} the upper left, upper right, lower left and lower right submatrix of the coarse 2-by-2 block matrix, respectively. Similarly x_1 and x_2 , and b_1 and b_2 are the upper and lower parts of x and b , respectively.

Given this two-level structure, the basic strategy for solving each global linear system generated by Newton's method is as follows:

1. Identify and reorder DOFs 1 through j such that A_{11}^{-1} (the inverse of A_{11}) is easy to apply (in parallel). The details of this step are given in Sections 5.1 and 5.2 for hard-sphere and polymer problems, respectively.
2. Determine a preconditioner P for $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$, the Schur complement of A with respect to A_{22} . (See Saad [30] for an overview of Schur complement methods.)
3. Solve $Sx_2 = (b_2 - A_{21}A_{11}^{-1}b_1)$ using a preconditioned Krylov method such as GMRES, with preconditioner P . Note that S may or may not be explicitly formed, depending on other problem details.
4. Finally, solve for $x_1 = A_{11}^{-1}(b_1 - A_{12}x_2)$.

The attractive properties of this algorithm from a parallel computing perspective are as follows:

1. A_{11}^{-1} can often be applied as a matrix-vector multiplication, or sequence of such. In other cases, A_{11}^{-1} can be explicitly computed. This means that explicitly forming S , or applying it implicitly is very efficient in parallel.
2. The dimension of S is often a small fraction of the original matrix. Thus, iterative methods such as GMRES will typically converge much faster because of the reduced dimension, independent of other factors such as preconditioning.
3. Assuming nodes of the mesh are equally partitioned on the parallel machine, parallel execution of this solver is well-balanced and produces identical results, independent of number of processors, up to roundoff error.

5 The new solver applied to two Fluid-DFTs

We now describe this approach in the context of two different kinds of Fluid-DFTs. The physics description for these two models are included in Appendix A. Here we focus on the structure of the matrices, implications for formulating the Schur complement, and suitable preconditioners.

5.1 Hard-spheres

A hard-sphere fluid is the simplest type of fluid model that is of practical interest (see Appendix A.1). In a first order numerical formulation, these problems typically have one or more density unknowns (the critical fields), and a set of linear auxiliary variables referred to as non-local densities (see Eq. 5). In our block matrix formulation, the equations corresponding to the density unknowns (i.e. the Euler-Lagrange equations) are collected in the lower portion of the matrix (densities are the x_2 variables). The non-local density equations are collected in the top portion of the matrix (nonlocal densities are the x_1 variables).

In the particular case of the Rosenfeld functionals applied to a single component fluid, some of the non-local densities are trivially dependent on others with

$$n_\gamma(\mathbf{r}) = C n_\epsilon(\mathbf{r}). \tag{9}$$

where C is a constant.

If all of the nonlocal densities are all treated as independent variables, there are $N_{nld} = 4 + 2 * D$ nonlocal densities per node in the problem where D is the physical dimension (1, 2, or 3) of the problem of interest. In this case, the A_{11} block (which is composed of non-local density interactions only) is

$$A_{11} = I. \tag{10}$$

In this case, the inverse of A_{11} is obviously trivial.

However, for the special case of a single component fluid, we generally encode the linear dependency in Eq.9. This leaves $N_{nld} = 2 + D$ variables that are described by integration stencils (nonzeros in the A_{12} block). We further decompose the A_{11} block in this case into a 2-by-2 block matrix as follows. The non-local density variables that have no dependence on other non-local densities are put in the upper left block. The lower right block contains all others, which have a dependence only on variables in the first block. The A_{11} block is then

$$A_{11} = \begin{pmatrix} -I & 0 \\ X & -I \end{pmatrix}. \quad (11)$$

where X contains the various proportionality constants, C , in the problem of interest. It is immediately clear from this expression of A_{11} that the inverse is simply

$$A_{11}^{-1} = \begin{pmatrix} -I & 0 \\ -X & -I \end{pmatrix}. \quad (12)$$

Clearly the simple form of A_{11}^{-1} allows it to be explicitly applied in a matrix-vector multiply for an efficient calculation of S . Very simple preconditioners for S (e.g. Jacobi scaling based on the diagonal of A_{22}) work well to solve hard sphere systems. Using this approach we have seen scalable results for 1, 2 and 3D problems. Section 6 provides the details.

5.2 Polymer Problems

Polymer and molecular Fluids-DFTs that include a deterministic treatment of chain conformations come in several varieties [3, 35, 23, 2]². However, they can all be characterized again as a combined system of equations where the critical field variables and nonlocal ancillary variables. With the chain structure equations playing the role of the nonlocal ancillary variables. To be more specific we consider the Chandler-McCoy-Singer DFT [3] (CMS-DFT) as it is enumerated in Appendix A.2. In this case, the critical variables, $\rho(\mathbf{r})$ and $U(\mathbf{r})$ are the densities and an unknown mean field respectively. The polymer conformation information is described by the Green's function propagator equations, $G(\mathbf{r})$ and $G^{inv}(\mathbf{r})$. The structure of these equations is quite different from the nonlocal densities of the hard sphere problem as they are significantly coupled to one another through a recursive relationship (see Equations 22 and 23 in Appendix A.2).

²Note that another approach takes chain conformations from a large number of samples generated with molecular simulations [40, 25]

In our new method, the A_{11} block is composed of these propagator equations, and when properly ordered, A_{11} takes the block form

$$A_{11} = \begin{pmatrix} A_{11}^{11} & 0 & 0 & 0 \\ A_{11}^{21} & A_{11}^{22} & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots \\ A_{11}^{j1} & \dots & A_{11}^{j-1,j} & A_{11}^{jj} \end{pmatrix}. \quad (13)$$

where j is two times the length of the polymer chain in the model and each A_{11}^{ii} is diagonal. Because each A_{11}^{pq} is distributed across the parallel machine proportionally to the nodes, and because each A_{11}^{ii} is diagonal, applying A_{11}^{-1} is a sequence of j diagonal scalings and matrix multiplications. Although A_{11}^{-1} cannot be explicitly formed for polymers, we still retain sufficient parallelism to get excellent performance on distributed memory computers and application of A_{11}^{-1} is invariant under changes in processor count up to round-off error.

The preconditioner P for polymer problems is more challenging than for hard-spheres. The A_{22} block for polymers has the following form:

$$A_{22} = \begin{pmatrix} D_{11} & F \\ D_{21} & D_{22} \end{pmatrix}. \quad (14)$$

The first block of equations in A_{22} is associated with the unknown fields, $U(\mathbf{r})$ (Equation 21), and the second block with the primitive densities, $\rho(\mathbf{r})$ (Equation 20). Each of the D_{pq} blocks is diagonal, whereas the F matrix describes the dependence of the unknown field, $U(\mathbf{r})$, on the primitive densities. The F block is by far the most dense submatrix in the global matrix due to the range of attractions that apply to integrals over the direct correlation function (see Equation 21). As mentioned in Section 3, the density of F increases dramatically with mesh refinement. Also, for the purposes of communicating off-processor values in a distributed memory implementation of the solver, F will have a large overlap and will require the most attention to assure good parallel communication complexity. It is worth noting that F is the *only* submatrix block that has these unusual stencil characteristics. It is also worth noting that the values in F do not change between nonlinear iterations.

One final observation is needed to motivate our preconditioner: All values in A_{22} are $O(1)$ except the values in D_{21} , which are $O(10^{-10})$. In other words, the primitive densities have a very weak direct dependence on the unknown fields, $U(\mathbf{r})$. We take advantage of this by defining an approximation to A_{22} as:

$$A_{22} \approx \tilde{A}_{22} = \begin{pmatrix} D_{11} & F \\ 0 & D_{22} \end{pmatrix}. \quad (15)$$

We then define a preconditioner P for S that consists of a block Gauss-Seidel (one back-solve) to \tilde{A}_{22} . Given that D_{22} and D_{11} are diagonal and well-distributed and that F is also

distributed, applying P involves two diagonal scalings, using D_{22} and D_{11} , and a matrix-vector multiplication using F . All of these steps are very efficient in parallel. Thus, we once again have an effective parallel solver whose results are invariant under changes to processor count, up to round-off error. This algorithm for CMS-DFT for polymers has similar attractive features as the hard-sphere solver, except that A_{11}^{-1} cannot easily be explicitly formed.

One additional benefit is that the solver can efficiently handle very long polymer chains, since the chain length only increased the dimension of A_{11} , which has a modest impact on the performance and robustness of the solver. We note that previous approaches to solving polymer problems typically required a fairly high level of fill incomplete factorization to solve the linear system of equations. Our new solver requires almost no additional memory for the preconditioner. This reduces the memory requirement for the solver by at least a factor of two, often a factor of 4 or more.

6 Computational Results

In this section, we show results for several cases where the numerical problem is either a 2-dimensional problem (with one uniform dimension) or a full 3 dimensional problem. We compare a solution via (i) a generic global solver that attempts to solve all equations simultaneously in a single matrix via standard preconditioned Krylov methods and (ii) the new solvers described in Sections 4 and 5. Overall the improvements due to the new algorithmic framework are dramatic, resulting in several to ten times improvement, as well as making some large problems tractable. All results were generated on the Sandia system red squall, a 258-node, dual processor Opteron-based system (2.2 GHz processors with 4 GB memory per node) using a Quadrics Elan4 high-speed interconnect.

6.1 Solver Approaches

All of the results presented here are from the Fluid-DFT application Tramonto [10]. Tramonto is a parallel, distributed memory application that solves Fluid-DFT problems using real-space techniques. Early versions of Tramonto used general-purpose preconditioned Krylov solvers from the solver package Aztec [37]. Presently Tramonto has been re-designed to utilize the Trilinos solver framework, a large collection of solver packages and parallel linear algebra tools [17, 20].

In this section we use the term *old solver* to refer to a general-purpose preconditioned Krylov solver applied to the full linear system of equations ordered as described in Equation 8. We use an overlapping Schwarz preconditioner with ILUT [29] as the local sub-domain solver via the Trilinos package IFPACK [32]. We use non-restarted GMRES [31] from the Trilinos package AztecOO [19] as the iterative method. One should note that,

even though we refer to this solver as old, it is accessing solver capabilities from Trilinos and is already an improvement over the Aztec solvers previously used in Tramonto, since Trilinos performance is generally better than Aztec and also because the ordering of equations has changed from a node-first ordering to a DOF-first ordering. This is the same global ordering as we use for the new solvers and results in very favorable ILUT fill and robustness properties for the general-purpose solver.

The *new solvers* discussed here also refer to preconditioned Krylov methods. However, we now use our segregated preconditioners and apply GMRES to the Schur complement system. All preconditioners for the new solvers are constructed and applied using the matrix and vector classes in the Trilinos package Epetra [18]. Due to the number of DOFs and the sequencing requirements to apply A_{11}^{-1} , the new solver preconditioners are composed of between several and more than 100 Epetra distributed sparse matrices. The new solvers also use GMRES from AztecOO.

6.2 Hard sphere fluids in 2-dimensions

This problem considers an inhomogeneous fluid where there are three cylindrical rod surfaces in the domain. This problem is similar to systems studied extensively previously in a study on adsorption in disordered porous media[12]. Since the solution is uniform perpendicular to the rods, the numerical problem may be solved in 2-dimensions (with the third dimension treated analytically). Figure 2 shows the density profile we compute for this case. The computational domain is $10\sigma \times 10\sigma$ in size where σ is the diameter of the fluid particles. Periodic boundary conditions were applied in this calculation. The bulk fluid density was $\rho\sigma^3 = 0.75$. The three rods were all quite small with diameters of $2R = 0.5\sigma$ and were located at $(x/\sigma, y/\sigma) = (5, 5), (1, 9), (7, 6)$. Volume exclusion interactions define the rod-fluid interactions, and the corresponding discontinuity in the external field is found at a distance $R + 0.5\sigma$ from the center of the cylinders.

Table 1 compares the new solvers to a standard preconditioned Krylov method. In this case the latter was solved using an ILUT preconditioner with 2 levels of fill-in. Solves were first attempted with no preconditioning and with an ILU preconditioner. Neither were able to solve the linear problem in less than 100 iterations. As is apparent from the table the chosen preconditioner is identical to the new algorithm in nonlinear updates, and is similar in the linear solver iteration count. The table shows both parallel scaling and scaling with mesh refinement at a fixed number of processors.

These results clearly demonstrate that the new method is quite powerful particularly for smaller numbers of processors. We achieve increased performance with the new algorithms, and the decrease in solve time is on the order of one to two orders of magnitude. Using the solve time data in the second part of the table (excluding data at $\Delta x = 0.2\sigma$) we find that the scaling of the old code with mesh refinement is $T \propto N^{2.7}$. while the new code has $T \propto N^{2.0}$.

#Procs	<# Lin iters>		Time/ N_{iter}		T_{1Proc}/T (New)	T_{old}/T_{new}
	Old	New	Old	New		
1	8	22	495.1	14.7	1	33.8
2	10	22	148.7	8.5	1.7	17.5
4	11	22	47.4	4.2	3.5	11.3
8	12	22	16.1	2.1	6.9	7.6
16	13	22	6.3	1.1	13.3	5.7
32	15	22	2.1	0.6	24.8	3.6
64	18	22	1.1	0.3	43.1	3.2
Δx	<# Lin iters>		Time/ N_{iter}		$T/T_{\Delta x=0.2}$ (New)	T_{old}/T_{new}
	Old	New	Old	New		
$\sigma/5$	12	17	0.1	0.07	1	1.4
$\sigma/10$	18	22	1.1	0.3	5	3.4
$\sigma/20$	18	24	32.1	4.7	69.4	6.8
$\sigma/40$	-	24	-	82.0	1206	-

Table 1. Results for a 2D Hard sphere test problem. Note that every run in the table solved with 10 nonlinear iterations. In the top part of the table, the columns are: the number of processors used for the calculation, the average number of linear iterations per nonlinear iteration, the solve time per nonlinear iteration, the speedup relative to the single processor time, and the ratio of solve times from the old to the new algorithm. In the bottom part of the table we consider scaling with mesh density. The first column contains the mesh spacing, and the 6th column contain timings relative to the $\Delta x = \sigma/5$ result. All data in the lower part of the table were generated on 64 processors. All data in the upper part of the table were generated with a mesh spacing of $\Delta x = \sigma/10$.

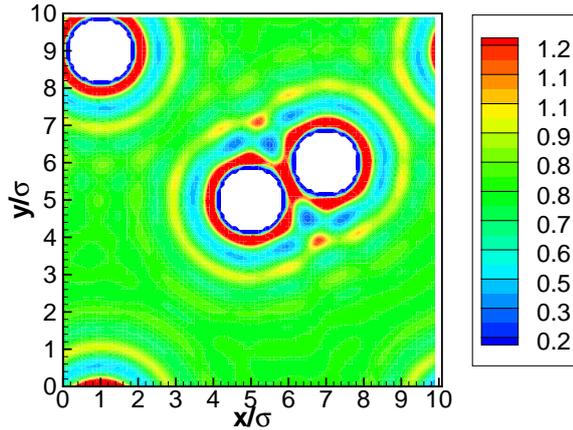


Figure 2. Fluid density ($\rho\sigma^3$) distribution in the vicinity of 3 nanocylinders.

6.3 3D Hard spheres

This problem considers a hard sphere fluid in a nanotube of finite length where the nanotube is located in a planar surface (or membrane). This particular choice is the base case for studies on ion channel proteins [13]. Specifically, the domain is of size $18\sigma \times 6\sigma \times 6\sigma$ with the long axis of the nanotube in the x dimension. The diameter of the nanotube is 2.5σ . The length of the nanotube is 10σ . The computational domain has bulk boundary conditions in the x dimension and continuation boundary conditions in the y and z . In the latter case, when performing integrals we assume that the density profile at the edge of the computational domain persists and is constant beyond the boundary. This boundary condition isolates this particular nanotube in a membrane. The bulk fluid density is again set to be $\rho_b\sigma^3 = 0.75$ for studies on parallel performance, but is reduced to $\rho_b\sigma^3 = 0.6$ for the mesh refinement studies as the system of equations becomes difficult to converge at the higher density for the most refined mesh. Figure 3 shows the density profile for one slice (at $z = 3\sigma$) in the computational domain.

Performance results are presented in Table 2. Overall the results are very similar to the 2D problem of the previous section. The new algorithm improves on the performance of the old algorithm (ILUT preconditioner with 2 levels of fill) by up to 18 times for measurable cases. The problems are also amenable to much smaller parallel platforms and a more refined mesh with the new algorithms. These improvements can be attributed to reduced memory requirements of these algorithms. The scaling with mesh density for this problem using the new algorithm is $T \propto N^{1.65}$. For the old algorithm we find $T \propto N^{2.2}$.

#Procs	N_{iter}	<# Lin iters>		Time/ N_{iter}		T_{4Proc}/T (New)	T_{old}/T_{new}
		Old	New	Old	New		
4	11	-	-	-	117.4	1	-
8	11	-	76	-	61.0	1.9	-
16	11	53*	76	544*(6)	29.8	3.9	18.*
32	11	73	76	154.5	16.7	7.0	9.3
64	11	80	76	55.4	8.3	14.1	6.7
128	11	89	76	19.9	4.7	24.8	4.2
Δx	N_{iter}	<# Lin iters>		Time/ N_{iter}		$T/T_{\Delta x=0.2}$ (New)	T_{old}/T_{new}
		Old	New	Old	New		
$\sigma/5$	7	45	44	17.2	4.1	1	4.1
$\sigma/7$	8	51	49	150.5	18.9	4.6	8.0
$\sigma/10$	9	-	51	-	121.1	29.3	-

Table 2. Results for a 3D Hard sphere test problem. The second column now contains the number of nonlinear iterations needed to solve the problem. For a description of all other columns see the table 1 caption. All data in the upper part of the table were generated with a mesh spacing of $\Delta x = \sigma/5$, and with a bulk density of $\rho\sigma^3 = 0.75$. All data in the lower part of the table were generated on 128 processors with a bulk density of $\rho\sigma^3 = 0.6$. The data with an * indicates that complete convergence was not obtained and timings are extrapolated based on the number of completed nonlinear iterations in the parenthesis.

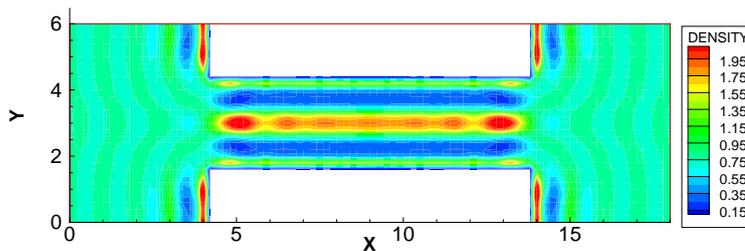


Figure 3. Fluid density ($\rho\sigma^3$) distribution for a hard sphere fluid in a finite length nanopore.

6.4 2D Polymer

Next we consider a 2-dimensional system based on the CMS polymer DFT presented in Appendix A.2. We consider a homopolymer where there are 10 identical beads on the chain and the size of each segment is σ . There are a pair of infinite cylinders in the problem (diameter 2σ), and we solve for the polymer distribution around these nanocylinders. Both polymer bead interactions and polymer-surface interactions are purely repulsive as defined by volume exclusions. A series of these calculations at different separations could be performed to compute the force between the cylinders as a function of distance. Performance studies had a computational domain of size $7\sigma \times 9\sigma$. The initial guess was a uniform solution at the bulk site type density of $\rho\sigma^3 = 0.85$. Figure 4 shows the solution we obtain for this problem.

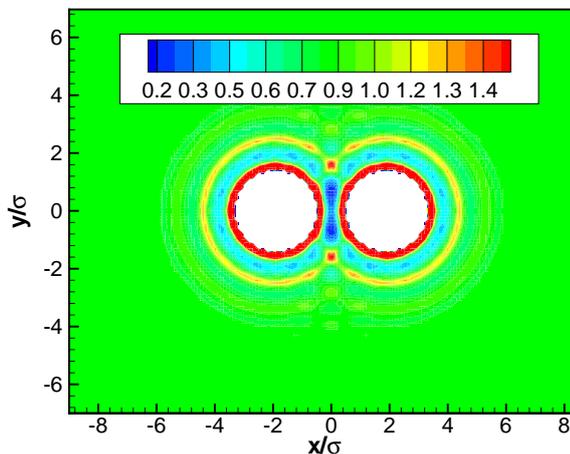


Figure 4. Density ($\rho\sigma^3$) distribution for a 10-mer homopolymer near two nanocylinders. The computational domain was 1/4 of this image with reflective boundaries at $x = 0$ and $y = 0$.

Performance results for this 2D polymer test problem are shown in Table 3. We find very similar behavior to the atomic fluids cases presented in the previous examples with up to a 25 fold improvement in performance. Again the "old" method was solved using an ILUT preconditioner with 2 levels of fill. While both methods become more expensive as the polymer chain length increases, we find this to be approximately a linear effect for the new algorithms while a steeper performance penalty occurs for the generic methods.

6.5 3D Polymer

Finally we consider a 3-dimensional system where a self-assembled lipid bilayer is sandwiched between two planar arrays of spheres of diameter 9σ separated by 10σ in a square lattice. The particular chemical model we consider has two components. The first is a model lipid molecule with head group and tail group beads. There are two head group beads on a chain, and 16 tail group beads on a linear chain. The head groups are in the middle so we have an 8-2-8 morphology on the chain. The head group beads are larger than the tail group beads with $\sigma_h/\sigma_t = 1.44$. The second component in the system is a single site solvent of the same size as the tail beads. The interactions between various species are chosen to favor self-assembly of a lipid bilayer where the head groups form an interface between the tail beads and the solvent beads. Lipid bilayers formed from this coarse grained model have been shown to be in reasonable agreement with MD simulation of the same models, and to map reasonably well to fluid experimental bilayers[9]. Figure 5 shows the solution we obtain for this problem. Computing interactions of nanoscale surfaces and colloidal particles with lipid bilayers are needed to provide a molecular theory based analysis of surface forces experiments (e.g. surface forces apparatus, atomic force microscope, optical tweezers, etc) for these systems. This type of calculation is also needed for studying the interactions of lipid bilayers with proteins.

The computational domain for this problem was $11\sigma \times 5\sigma \times 5\sigma$, and the initial guess for the density distribution is a previously converged uniform bilayer result. Note that this case had 37856 nodes in the computational domain and 44 unknowns per node for a total of 1.66×10^6 unknowns in the problem. Algorithm performance is presented as a function of number of processors only in Table 4. Once again the new algorithms improve both the performance of the code and provide the ability to solve large problems on smaller parallel platforms. We find a significant superlinear speed up for this case from 32 to 64 processors. We were not able to solve this problem using generic Krylov space methods even for the case of 128 processors where we tried a variety of parameters from solver tolerances to ILUT fill factors.

6.6 Results Summary

The results presented in this section are quite promising for the new solvers. In particular there are several observations worth noting.

#Procs	N_{iter}	<# Lin iters>		Time/ N_{iter}		T_{1Proc}/T (New)	T_{old}/T_{new}
		Old	New	Old	New		
1	9		70	-	88	1	-
2	9		70	-	45	2.0	-
4	9	25	70	290*(8)	26.2	3.4	11.0
8	9	28	70	77	12	7.3	6.4
16	10	33	70	23	6.7	13.1	3.4
32	8	40	70	7.1	3.8	23.2	1.9
64	9	46	69	2.8	2.2	40.0	1.3
128	9	58	69	1.4	1.8	49.0	0.8
Δx	N_{iter}	<# Lin iters>		Time/ N_{iter}		$T/T_{\Delta x=\sigma/10}$ (New)	T_{old}/T_{new}
		Old	New	Old	New		
$\sigma/10$	9	58	69	1.4	1.8	1	0.8
$\sigma/20$	10	55	69	28.9	13.1	7.3	2.2
$\sigma/30$	9	57	70	1580*(7)	62.3	34.6	25
N_{seg}	N_{iter}	<# Lin iters>		Time/ N_{iter}		$T/T_{N_{seg}=10}$ (New)	T_{old}/T_{new}
		Old	New	Old	New		
10	8	40	70	7.1	3.7	1	1.9
20	8	48	69	21.7	8.9	2.4	2.4
40	8	62	70	58.7	15.1	4.1	3.9
80	7	93	68	257.4	30.9	8.4	8.3

Table 3. Results for a 2D polymer problem. The top part of the table shows scaling with number of processors. The middle part of the table shows scaling with mesh density. The bottom section of the table shows scaling with polymer chain length. The column descriptions can be found in the caption of Table 1. All data in the upper part of the table were generated with a mesh spacing of $\Delta x = \sigma/10$ and for polymer chains of length $N_{seg} = 10$. All data in middle part of the table were generated on 128 processors for polymer chains of length $N_{seg} = 10$. All data in the bottom part of the table were generated on 32 processors with $\Delta x = \sigma/10$. The data with an * indicates that complete convergence was not obtained and timings are extrapolated based on the number of completed nonlinear iterations in the parenthesis.

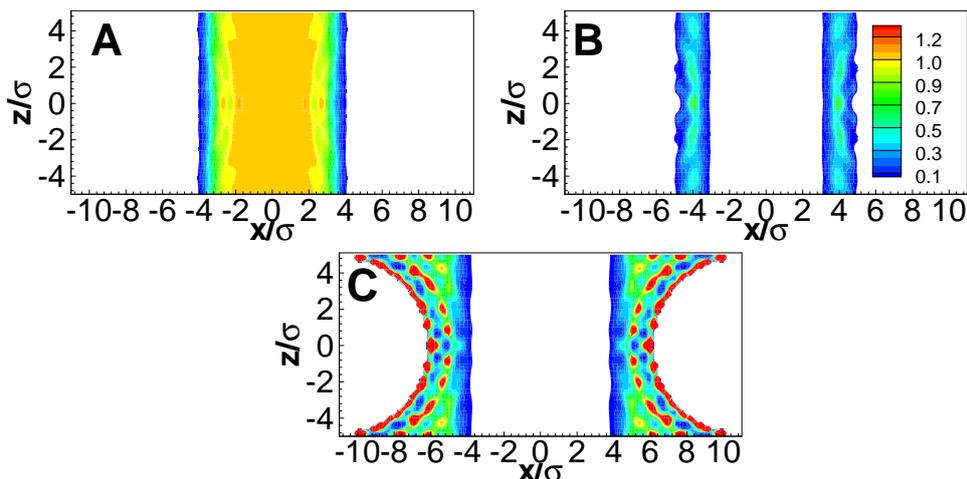


Figure 5. One slice (at $y = 0$) through a 3-dimensional computational volume. The color contours show density ($\rho\sigma^3$) distributions for lipid tail beads (A), lipid head beads (B), and solvent (C) for a lipid bilayer assembly sandwiched between planar arrays of large spheres. The computational domain is 1/4 of the domain shown in the figure and utilized reflective boundary conditions on all edges. The legend shows the contour scale for all three figures. Densities less than $\rho\sigma^3 = 0.01$ are blanked to white.

6.6.1 Memory Use

The new solvers use almost no extra memory for the preconditioners. Furthermore, the dimension of the implicit problem that GMRES must solve is ten to 100 times smaller. Since all solvers are using non-restarted GMRES and performing 100 or more iterations, the GMRES storage cost is $O(100n)$ where n is the dimension of the GMRES problem. Letting k denote the nonzero count of the global matrix, the memory use for GMRES vectors in the old solver is comparable to the matrix: from $3k$ to $10k$ storage. The ILUT preconditioner for the old solver requires from $2k$ to $10k$ storage. Thus the old solver requires minimally $6k$ storage, up to $20k$ or more. In contrast, the cost of GMRES storage for the new solvers is $0.1k$ to $0.01k$, and the overall storage is less than $2k$ with the storage cost of the matrix being dominant. The overall reduction in memory use for the new solvers over the old solver is therefore minimally a factor of 5 and sometimes more than a factor of 20.

6.6.2 Tuning parameters

One difficulty common to preconditioned iterative methods is the presence of tuning parameters. This is true for the old solver in that ILUT requires *ad hoc* parameters to determine

#Procs	<# Lin iters>	Time/ N_{iter}	T_{32Proc}/T
32	93	1510	1
64	93	414	3.6
128	93	190	7.9

Table 4. Results for a 3D polymer test problem. Each case require 10 nonlinear iterations for a solution. For a description of all other columns see the table 1 caption. All data in the upper part of the table were generated with a mesh spacing of $\Delta x = \sigma/5$. Note that the data flagged with a * is a case where complete convergence was not obtained. The reported data reflect timings obtained for one complete nonlinear iteration and the associated linear solution. The total time was computed assuming a consistent 10 nonlinear iterations independent of number of processors.

fill, and the user must prescribe these parameter values. Furthermore, as is well-known, overlapping Schwarz methods tend to lose robustness as the processor count increases, making the choice of ILUT parameters and the maximum number of GMRES iterations a function of processor count. In contrast, our new solvers have *no* tuning parameters and have identical convergence behavior independent of processor count. This behavior may be the most important feature to an application user.

6.6.3 Solver scalability in processor count

The processor scalability results for the old solver are quite remarkable. In all cases, once the number of processors is large enough for the old solver to work at all, the performance improvement as a function of processor count is superlinear. This behavior is often observed in practice and is a function of two factors: (i) increasing processor count increases the amount of cache memory and, for a fixed size problem, reduces the working data set on a processor and (ii) as processor count increases, the subdomains for overlapping Schwarz decrease in size, resulting in smaller ILUT factors and less work per iteration, even though the number of iterations increases. Some of the new solvers scalability results are also superlinear, due to factor (i) above. However, since the new solvers are already very efficient in serial cost and apply GMRES to a much smaller problem, there is much less work to distribute as processor counts increase. In effect, the new solvers cannot benefit as much as the old solver from large processor counts for the same global problem. However, the new solvers scale quite well and, because of their low memory usage, can be effective using far fewer processors and can solve much larger problems than the old solver.

6.6.4 Scalability in mesh density and chain length

Our new solvers are not invariant under mesh density changes, but nearly so. In fact, except for the coarsest mesh, the number of solver iterations remains nearly constant as the mesh density increases. The old solver iteration count also remains fixed as mesh density increases. However, the overall cost of the old solver grows much faster than the new solvers.

For polymer problems, our new solver has constant iteration count as the chain size increases. Furthermore, the cost of the solver grows approximately linearly with the length of the chain. Again, this is a marked improvement over the old solver.

7 Conclusions

In this article we have presented a general mathematical framework for describing Fluid-DFT problems and solving them using a new family of segregated Schur complement solvers. By viewing Fluid-DFTs from a segregated variable perspective we obtain a rich structure that can be exploited in the development of low-cost, robust, scalable solution methods. We have shown that this approach is very effective for two major classes of Fluid-DFTs, and is very promising for other classes as well. The improvement in solution time, robustness and memory use opens the door to easy solution of previously intractable problems and broadens the scope of applicability for real-space Fluid-DFT methods.

Although our new solvers are faster and require much less memory than the old solver, perhaps most important is the fact that no tuning parameters are required. Since most problems of interest require the use of hundreds of processors and run for many hours, this fact makes use of Tramoto much easier.

References

- [1] Dapeng Cao and Jianzhong Wu. Density functional theory for semiflexible and cyclic polyatomic fluids. *J. Chem. Phys.*, 121:4210–4220, 2004.
- [2] D.P. Cao and J.Z. Wu. Density functional theory for semi-flexible and cyclic polyatomic fluids. *J. Chem. Phys.*, 121:4210–4220, 2004.
- [3] D. Chandler, J. D. McCoy, and S. J. Singer. Density functional theory of nonuniform polyatomic systems. 1. general formulation. *J. Chem. Phys.*, 85:5971, 1986.
- [4] J.P. Donley, J.G. Curro, and J.D. McCoy. A density-functional theory for pair correlation-functions in molecular liquids. *J. Chem. Phys.*, 101(4):3205–3215, 1994.
- [5] S.A. Egorov. Effect of repulsive and attractive interactions on depletion forces in colloidal suspensions: A density functional theory treatment. *Phys. Rev. E*, 70(3):Art 031402, 2004.
- [6] F. Fang and I. Szleifer. Kinetics and thermodynamics of protein adsorption: A generalized molecular theoretical approach. *Biophys. J.*, 80(6):2568–2589, June 2001.
- [7] Laura J. Douglas Frink and Andrew G. Salinger. Two- and three-dimensional nonlocal density functional theory for inhomogeneous fluids i. algorithms and parallelization. *jcp*, 159:407–424, 2000.
- [8] Laura J. Douglas Frink and Andrew G. Salinger. Two- and three-dimensional nonlocal density functional theory for inhomogeneous fluids ii. solvated polymers as a benchmark problem. *jcp*, 159:425–439, 2000.
- [9] L.J.D. Frink and A.L. Frischknecht. Density functional theory approach for coarse-grained lipid bilayers. *Physical Review E*, 72:041923, 2005.
- [10] L.J.D. Frink and A.G. Salinger. Two and three dimensional nonlocal density functional theory for inhomogeneous fluids i. algorithms and parallelization. *J. Comp. Phys.*, 159(2):407–424, April 2000.
- [11] L.J.D. Frink and A.G. Salinger. Two and three dimensional nonlocal density functional theory for inhomogeneous fluids ii. solvated polymers as a benchmark problem. *J. Comp. Phys.*, 159(2):425–439, April 2000.
- [12] L.J.D. Frink and A.G. Salinger. Rapid analysis of phase equilibria with density functional theory ii. capillary condensation in disordered porous media. *J. Chem. Phys.*, pages 7466–7476, 2003.
- [13] L.J.D. Frink, A.G. Salinger, M.P. Sears, J.D. Weinhold, and A.L. Frischknecht. Numerical challenges in the application of density functional theory to biology and nanotechnology. *J. Phys.-Cond. Matter*, 14(46):12167–12187, November 2002.

- [14] L.J.D. Frink, A. Thompson, and A.G. Salinger. Applying molecular theory to steady-state diffusing systems. *J. Chem. Phys.*, 112(17):7564–7571, May 2000.
- [15] A.L. Frischknecht, J.G. Curro, and L.J.D. Frink. Density functional theory for inhomogeneous polymer systems ii. application to block copolymer thin films. *J. Chem. Phys.*, 117(22):10398–10411, December 2002.
- [16] D. Henderson, editor. Marcel Dekker Inc.s, New York, 1992.
- [17] M. A. Heroux. Trilinos home page, 2003. <http://software.sandia.gov/trilinos>.
- [18] M. A. Heroux. Epetra home page. <http://software.sandia.gov/Trilinos/packages/epetra>, July 2004.
- [19] Michael A. Heroux. AztecOO Users Guide. Technical Report SAND2004-3796, Sandia National Laboratories, 2004.
- [20] Michael A. Heroux, Roscoe A. Bartlett, Vicki E. Howle, Robert J. Hoekstra, Jonathan J. Hu, Tamara G. Kolda, Richard B. Lehoucq, Kevin R. Long, Roger P. Pawlowski, Eric T. Phipps, Andrew G. Salinger, Heidi K. Thornquist, Ray S. Tuminaro, James M. Willenbring, Alan Williams, and Kendall S. Stanley. An overview of the trilinos project. *ACM Trans. Math. Softw.*, 31(3):397–423, 2005.
- [21] E. Kierlik and M.L. Rosinberg. A perturbation density-functional theory for polyatomic fluids. 3. application to hard chain molecules in slitlike pores. *J. Chem. Phys.*, 100:1716–1730, 1994.
- [22] H. Löwen. Density functional theory of inhomogeneous classical fluids: recent developments and new perspectives. *J. Phys.-Cond. Matter*, 14(46):11897–11905, November 2002.
- [23] Z.D. Li, D.P. Cao, and J.Z. Wu. Density functional theory and monte carlo simulations for the surface structure and correlation functions of freely jointed lennard-jones fluids. *J. Chem. Phys.*, 122:174708, 2005.
- [24] A.P. Malanoski and F. van Swol. Lattice density functional theory investigation of pore shape effects. adsorption in collections of noninterconnected pores. *Phys. Rev. E*, 66:41603, 2002.
- [25] S.K. Nath, A.L. Frischknecht, and J.G. Curro. Density functional theory and molecular dynamics simulation of poly(dimethylsiloxane) melts near silica surfaces. *Macromolecules*, 38(20):8562–8573, 2005.
- [26] R.G. Parr and W. Yang. Oxford Science Publications, New York, 1989.
- [27] Y. Rosenfeld. Structure and effective interactions in multi-component hard-sphere liquids: the fundamental-measure density functional approach. *J. Phys. Cond. matter*, 14(40):9141–9152, 2002.

- [28] R. Roth, R. Evans, A. Lang, and G. Kahl. Fundamental measure theory for hard-sphere mixtures revisited: the white bear version. *J. Phys. Cond. Matter*, 14:12063–12078, 2002.
- [29] Y. Saad. ILUT: A dual threshold incomplete lu preconditioner. *Numer. Linear Algebra Appl.*, 1(4):387–402, 1994.
- [30] Y. Saad. *Iterative methods for sparse linear systems*. SIAM, 2nd edition, 2003.
- [31] Youcef Saad and Martin H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, July 1986.
- [32] Marzio Sala and Michael A. Heroux. Robust algebraic preconditioners using IFPACK 3.0. Technical Report SAND2005-0662, Sandia National Laboratories, 2005.
- [33] M.P. Sears and L.J.D. Frink. A new efficient method for density functional theory calculations of inhomogeneous fluids. *J. Comp. Phys.*, 190:184–200, September 2003.
- [34] Z. Tang, L. Mier y Teran, H.T. Davis, L.E. Scriven, and H.S. White. Non-local free energy density functional theory applied to the electrical double layer. i. symmetrical electrolytes. *Mol. Phys*, 10(2):369–392, October 1990.
- [35] S. Tripathi and W.G. Chapman. Microstructure of inhomogeneous polyatomic mixtures from a density functional formalism for atomic mixtures. *Phys. Rev. Lett.*, 94:087801, 2005.
- [36] Sandeep Tripathi and Walter G. Chapman. Microstructure and thermodynamics of inhomogeneous polymer blends and solutions. *Phys. Rev. Lett.*, 94:087801, 2005.
- [37] Ray S. Tuminaro, Michael A. Heroux, Scott. A. Hutchinson, and J. N. Shadid. *Official Aztec User's Guide, Version 2.1*. Sandia National Laboratories, Albuquerque, NM 87185, 1999.
- [38] H.-J. Woo, L. Sarkisov, and P.A. Monson. Mean field theory of adsorption in porous glasses. *Langmuir*, 17:7472–7475, 2001.
- [39] J. Wu. Density functional theory for chemical engineering: From capillarity to soft materials. *AIChE Journal*, 52:1169–1193, 2006.
- [40] A. Yethiraj. Density functional theory of polymers: A curtin-ashcroft type weighted density approximation. *J. Chem. Phys.*, 109:3269–3275, August 1998.
- [41] Y. Yu and J. Wu. Density functional theory of inhomogeneous mixtures of polymeric fluids. *J. Phys. Chem.*, 117:2368, 2002.

A Two specific Fluid-DFTs

In this section we present the particulars of the two distinctly different Fluids-DFTs that we consider explicitly in this paper. The first is used to treat atomic fluids, and in particular hard sphere systems. The second is a DFT used to treat polymer fluids. We note that many other kinds of DFTs have been developed as well. In fact this field contains a whole collection of disparate approaches that must be analyzed individually from the perspective of optimizing solution algorithms.

A.1 An Atomistic Fluid Model

We first consider the Fundamental Measures Theory DFT (FMT-DFT) that was first developed by Rosenfeld, and that has been modified by others [27, 28]. This theory specifically treats hard sphere fluids with other physical effects (e.g. attractions, Coulomb interactions, and even bond constraints) being treated as a perturbation to the hard sphere FM theory. The grand free energy functional for a single component fluid is

$$\Omega[\{\rho(\mathbf{r})\}] = \int \rho(\mathbf{r})[\ln(\rho(\mathbf{r})) - 1]d\mathbf{r} + \int \Phi(\{n_\gamma\})\mathbf{r} + \int \rho(\mathbf{r})[V^{ext}(\mathbf{r}) - \mu]d\mathbf{r}. \quad (16)$$

where μ is a constant chemical potential in the case of an equilibrium-DFT, and the set of nonlocal variables n_γ are linear as was defined above in Eq. 5. The complete Euler-Lagrange equation to be solved is

$$\ln(\rho(\mathbf{r})) - V(\mathbf{r})/kT + \mu + \int \sum_\gamma \frac{\partial \Phi}{\partial n_\gamma}(\mathbf{r}') \frac{\delta n_\gamma(\mathbf{r}')}{\delta \rho(\mathbf{r})} d\mathbf{r}' = 0. \quad (17)$$

and the 2×2 block matrix is

$$\begin{bmatrix} -I & w_\gamma(\mathbf{r}, \mathbf{r}') \\ \sum_\gamma \frac{\partial^2 \Phi}{\partial n_\gamma \partial n_e}(\mathbf{r}) & D_{22}(\mathbf{r}) \end{bmatrix} \begin{bmatrix} \Delta n_\gamma \\ \Delta \rho \end{bmatrix} = - \begin{bmatrix} R_{NL}(\mathbf{r}) \\ F_{EL}(\mathbf{r}) \end{bmatrix}, \quad (18)$$

where $D_{22}(\mathbf{r})$ is a diagonal block matrix, and each entry is $1/\rho(\mathbf{r})$. Typically the FMT-DFT has four scalar nonlocal variables and two vector nonlocal variables so for a single component 3D problem, the number of rows in the upper part of the 2×2 block matrix will be 10 times the number of rows in the lower part of the block matrix. Thus the size of the system of equations is dominated by the nonlocal density variables rather than the primitive densities of interest.

For completeness we note that the hard sphere free energy density in Rosenfeld's original theory is

$$\Phi(\{n\}) = -n_0 \ln(1 - n_3) + \frac{n_1 n_2}{1 - n_3} - \frac{\vec{n}_{V1} \cdot \vec{n}_{V2}}{1 - n_3} + \frac{1}{24\pi(1 - n_3)^2} \left(n_2 - \frac{\vec{n}_{V2} \cdot \vec{n}_{V2}}{n_2} \right)^3. \quad (19)$$

A.2 A Polymer Fluid Model

The particular DFT for polymers we will consider was developed by Chandler, McCoy, and Singer[3], and our formulation is similar to that of Donely and McCoy[4, 15]. This particular theory is developed by minimization of a free energy functional with respect to both the density and an effective field variable, U . This effective field variable constrains a fluid of ideal chains to have the same density profile as the interacting chains of interest in the real external field V^{ext} . The theory solves simultaneously for critical fields, $\rho(\mathbf{r})$, and $U(\mathbf{r})$ for each type of monomer segment, α in the system. The two residual equations for these critical fields are

$$\rho_\alpha(\mathbf{r}) = \frac{\rho_{b,\alpha}}{N_\alpha} \sum_{s \in \alpha} \frac{G_s(\mathbf{r}) G_s^{inv}(\mathbf{r})}{\exp[-\beta U_\alpha(\mathbf{r})]}, \quad \alpha = 1..N_\alpha, \quad (20)$$

and

$$U_\alpha(\mathbf{r}) = V_\alpha(\mathbf{r}) - \sum_{\beta} \int c_{\alpha\beta}(\mathbf{r} - \mathbf{r}') (\rho_\beta(\mathbf{r}') - \rho_{b\beta}) d\mathbf{r}', \quad \alpha = 1..N_\alpha. \quad (21)$$

where N_α is the number of segments of type α , $c_{\alpha\beta}$ is the direct correlation function taken from a liquid state theory for bulk fluids, the sum in Eq.20 is taken over all N_s monomer segments in the fluid of interest, ρ_b denotes the homogeneous bulk density far from the surface, s is a particular segment on the polymer chain of interest, and the G and G^{inv} functions are propagator functions that describe chain connectivity. Specifically,

$$G_s(\mathbf{r}) = \exp[-\beta U_{\alpha(s)}(\mathbf{r})] \int \omega_{\alpha\beta}(\mathbf{r} - \mathbf{r}') G_{s-1}(\mathbf{r}') d\mathbf{r}', \quad s = 1..N_s, \quad (22)$$

and

$$G_s^{inv}(\mathbf{r}) = \exp[-\beta U_{\alpha(s)}(\mathbf{r})] \int \omega_{\alpha\beta}(\mathbf{r} - \mathbf{r}') G_{s+1}^{inv}(\mathbf{r}') d\mathbf{r}', \quad s = 1..N_s. \quad (23)$$

where ω is a delta function with a range equal to the bond length between segments s and $s - 1$ of types α and β respectively. We note that for an end bead only the initial field term is present.

To summarize we have $2N_\alpha$ critical variables ($\{\rho_\alpha\}$ and $\{U_\alpha\}$), and $2N_s$ nonlocal ancillary variables ($\{G\}$ and $\{G^{inv}\}$). The concrete example we discussed in this paper is for an 18 bead polymer mixed with a single site solvent, where the polymer chain has two types

of beads on the chain. For this 3 component, 19 segment problem, we have 6 critical field variables and 38 nonlocal ancillary variables.

This situation is quite extreme with respect to complexity in the integral equations. If the discrete matrix problem were formed only in terms of the critical variables, the nested nonlocal variables would result in multidimensional integrals for each Jacobian entry. Specifically, for a polymer with N_s segments, an $N_s - 1$ dimensional integral would need to be performed. This is clearly out of the question. Thus we form the matrix problem with each of the nonlocal propagator functions treated as independent variables. The structure of the resulting 2×2 block matrix is described more fully in Section 4.

DISTRIBUTION:

- 1 M2497
Central Technical Files, 8945-1
- 2 2497
Central Technical Files, 8945-1

- 2 MS 0899
Technical Library, 4536