

PARALLEL SEGREGATED SCHUR COMPLEMENT METHODS FOR FLUID DENSITY FUNCTIONAL THEORIES

MICHAEL A. HEROUX , ANDREW G. SALINGER, , AND LAURA J. D. FRINK*

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 purpose 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.

Key words. inhomogeneous fluids, density functional theories, Schur complement, iterative methods

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. Furthermore, the algorithms we develop can be applied in a similar fashion to all types of problems generated by the target application, showing that our basic approach is a useful general-purpose technique in this problem domain.

Density functional theories (DFTs) have been used to treat a variety of systems at many length scales. For interfacial systems, 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 electronic structure of materials [26]. More specifically, 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 Fluid-DFTs, but are based on coarsened models or potentials (e.g. the solvent averaged Yukawa potential).

Our previous efforts to develop numerical methods for Fluid-DFT began with the development of a Newton's method real space approach that could be solved using

*Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000 (ljfrink@sandia.gov, maherou@sandia.gov, agsalin@sandia.gov).

parallel iterative solvers optimized for large distributed memory parallel computers[10, 11, 13]. This code was robust, but expensive for 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 fluid-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 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 an analysis of our new solver algorithms as compared to generic methods developed for PDEs, and demonstrates considerable improvement in speed and robustness.

2. Mathematical Framework. In Fluid-DFTs an approximate free energy functional, $\Omega[\{\rho_i(\mathbf{r})\}]$, depends on a set of density fields, $\{\rho_i(\mathbf{r})\}$. The minimization of this free energy with respect to all fields $\{\rho_i(\mathbf{r})\}$ results in a system of residual equations to be solved for the equilibrium fluid distribution. The residual equations to be solved are written,

$$(2.1) \quad \frac{\delta\Omega}{\delta\rho_i(\mathbf{r})} = 0$$

where δ is the Frechet (functional) derivative, leading to an Euler-Lagrange residual equation.

In a perturbation theory, the free energy Ω is typically written as a sum of terms starting with the contributions of a well known reference fluid (often a hard sphere fluid). Many terms in these physics based expansions can be written as

$$(2.2) \quad F_{phys} = \int f[\{\rho_i\}]g[\{n_\gamma[\{\rho_i\}]\}]dr$$

where f is a functional of the critical density fields, and g is a functional of some nonlocal variables, n . Taking the functional derivative of these terms to set up the residual equations in Eq. 2.1 one must apply the chain rule

$$(2.3) \quad \frac{\delta}{\delta\rho_i(\mathbf{r})} \left(\int f[\{\rho_i\}]g[\{n[\{\rho_i\}]\}]dr \right) = \frac{\partial f}{\partial\rho_i}g[\{n\}] + \int f[\{\rho_i\}]\frac{\partial g}{\partial n}\frac{\delta n(\mathbf{r}')}{\delta\rho_i(\mathbf{r})}d\mathbf{r}'.$$

where for simplicity Eq.2.3 reflects a problem with only one nonlocal variable, n .

For local terms (e.g. ideal gas contributions), $g[n] = 1$ and the second term drops out. Nonlocal terms can take on various forms. For strict mean field contributions, $g = n_i$ where n_i is a simple linear functional of the set $\{\rho_i\}$,

$$(2.4) \quad n_i(\mathbf{r}) = \sum_j \int w_{ij}(\mathbf{r}, \mathbf{r}', R_{ij})\rho_j(\mathbf{r}')d\mathbf{r}',$$

with a weight function,

$$(2.5) \quad w_{ij}(\mathbf{r}, \mathbf{r}', R_{ij}) = \theta(|\mathbf{r} - \mathbf{r}'| - R_{ij})\phi_{ij}(|\mathbf{r} - \mathbf{r}'|)$$

where θ is a heavyside step function, ϕ_{ij} denotes a pairwise interaction (i.e. pair potential, direct correlation function, pair correlation function, etc), and R_{ij} is the characteristic range of the pairwise interaction. On differentiation, $\partial g/\partial n_i = 1$ and $\delta n_i/\delta \rho_j = w_{ij}(\mathbf{r}, \mathbf{r}', R_{ij})$. When forming a matrix problem it is necessary to calculate second derivatives of the free energy (the Hessian). One applies the chain rule in Eq.2.3 a second time. For these strict mean field terms, $\partial^2 g/\partial n^2 = 0$ so the Jacobian contribution from these terms is simply

$$(2.6) \quad J_{ij}(\mathbf{r}, \mathbf{r}') = w_{ij}(\mathbf{r}, \mathbf{r}', R_{ij})$$

While the number of nonzeros in the matrix is determined by the characteristic length, R_{ij} , these terms do not have an integral character at the level of the Hessian.

In contrast, consider a term, F_{phys} , in the free energy expansion with the form $f[\{\rho_i\}] = 1$ and $g[\{n_\gamma\}] = \Phi[n]$ where Φ is a nonlinear functional of a set of $\{n_\gamma\}$. In this case the first term in the chain rule drops out, and the first and second derivatives of these terms will be

$$(2.7) \quad \frac{\delta F_{phys}}{\delta \rho_i(\mathbf{r})} = \int \sum_\gamma \frac{\partial \Phi}{\partial n_\gamma} \frac{\delta n_\gamma(\mathbf{r}')}{\delta \rho_i(\mathbf{r})} d\mathbf{r}'.$$

and

$$(2.8) \quad \frac{\delta^2 F_{phys}}{\delta \rho_i(\mathbf{r})\delta \rho_j(\mathbf{r}')} = \int \sum_\gamma \sum_\epsilon \frac{\partial^2 \Phi}{\partial n_\gamma \partial n_\epsilon} \frac{\delta n_\gamma(\mathbf{r}'')}{\delta \rho_i(\mathbf{r})} \frac{\delta n_\epsilon(\mathbf{r}'')}{\delta \rho_j(\mathbf{r}')} d\mathbf{r}''.$$

Often, n_γ is again a linear functional of $\{\rho_i\}$,

$$(2.9) \quad n_\gamma(\mathbf{r}) = \sum_i \int \omega_{\gamma i}(\mathbf{r}, \mathbf{r}', R_i)\rho_i(\mathbf{r}')d\mathbf{r}',$$

and in this case the Hessian contribution may be written

$$(2.10) \quad \frac{\delta^2 F_{phys}}{\delta \rho_i(\mathbf{r})\delta \rho_j(\mathbf{r}')} = \int \sum_\gamma \sum_\epsilon \frac{\partial^2 \Phi}{\partial n_\gamma \partial n_\epsilon} \omega_{\gamma i}(\mathbf{r}, \mathbf{r}'', R_i)\omega_{\epsilon j}(\mathbf{r}', \mathbf{r}'', R_j)d\mathbf{r}''.$$

Note that the weight function, ω , is usually based either on a heavyside step function, $\theta(|\mathbf{r} - \mathbf{r}'| - R_i)$ or a Dirac delta function, $\delta(|\mathbf{r} - \mathbf{r}'| - R_i)$ where R_i is often a particle radius or diameter.

Computing matrix coefficients based on Eq.2.10 requires a second order (N^2) operation in order to locate the intersection of the weight functions $\omega_{\gamma i}(\mathbf{r}, \mathbf{r}'', R_i)$ and $\omega_{\epsilon j}(\mathbf{r}', \mathbf{r}'', R_j)$. To reduce the complexity of the matrix fill, the nonlocal variables, n_γ can be included as explicit independent variables when taking the second Frechet derivatives to form the matrix problem. We will refer to this method as a first order formulation later in this paper, and note that 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. In this case, the nonzero contributions to the

Jacobian will be

$$(2.11) \quad J_{\gamma\epsilon}(\mathbf{r}, \mathbf{r}') = \delta_{\gamma\epsilon}(\mathbf{r}, \mathbf{r}')$$

$$(2.12) \quad J_{\gamma i}(\mathbf{r}, \mathbf{r}') = \omega_{\gamma}(\mathbf{r}, \mathbf{r}')$$

$$(2.13) \quad J_{i\epsilon}(\mathbf{r}, \mathbf{r}') = \sum_{\gamma} \frac{\partial^2 \Psi}{\partial n_{\gamma} \partial n_{\epsilon}}(\mathbf{r}) \omega_{\gamma i}(\mathbf{r}', \mathbf{r})$$

where the first two contribution arise from the nonlocal variable residual equation defined by Eq. 2.9, and the third contribution comes from the taking the functional derivative of the Euler-Lagrange equation in Eq. 2.7 with respect to the nonlocal variables, n_{ϵ} . Note that there is no contribution from the derivative of the Euler-Lagrange contribution in Eq. 2.7 with respect to the density fields in this case (i.e. $J_{ij}(\mathbf{r}, \mathbf{r}') = 0$).

Integral operator terms of the form discussed above are pervasive in DFTs constructed as perturbations to the hard sphere fluid. However, the general approach of introducing variables to reduce the complexity of the matrix problem is a strategy that can be applied to other classes of Fluid-DFTs as well. The polymer-DFT fluids that we discuss later in this paper are based on a second order density expansion using an ideal chain reference fluid. Although this DFT falls into a different class than hard sphere perturbation DFTs, the strategy of introducing variables to reduce complexity can also be applied there.

Including nonlocal variables, n in the systems of equations can significantly increase the number of degrees of freedom (DOF) that must be solved at each mesh point. As a result, no advantage was gained from this approach when coupled to a generic PDE optimized solver applied to the full matrix problem [10]. In that case the decrease in the complexity of the matrix fill operation was offset by the increase in problem size for the linear solver. In the current work we demonstrate that when generic PDE solvers are replaced with segregated modeling strategies, Fluid-DFTs can be solved with significantly improved efficiency.

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.

The ineffectiveness of the generic methods to solve the Fluid-DFT equations efficiently 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:

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

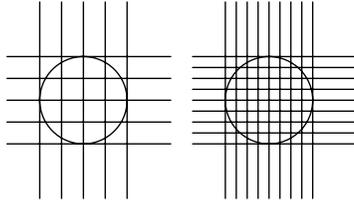


FIG. 3.1. *Stencil support for two mesh densities for integrating the area of the circle. As the mesh is refined, the stencil density increases. This corresponds to more expensive computation of the integrals and more nonzeros per row of the Jacobian matrix.*

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 3.1).
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.

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

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]

is

$$(4.1) \left(\begin{array}{ccc|ccc} A_{11}^{11} & \cdots & A_{11}^{1j} & A_{12}^{1,j+1} & \cdots & A_{12}^{1k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ A_{11}^{j1} & \cdots & A_{11}^{jj} & A_{12}^{j1} & \cdots & A_{12}^{jk} \\ \hline A_{21}^{j+1,1} & \cdots & A_{21}^{j+1,j} & A_{22}^{j+1,j+1} & \cdots & A_{22}^{j+1,k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ A_{21}^{k1} & \cdots & A_{21}^{kj} & A_{22}^{k,j+1} & \cdots & A_{22}^{kk} \end{array} \right) \begin{pmatrix} x_1^1 \\ \vdots \\ x_1^j \\ \hline x_2^{j+1} \\ \vdots \\ x_2^k \end{pmatrix} = \begin{pmatrix} b_1^1 \\ \vdots \\ b_1^j \\ \hline b_2^{j+1} \\ \vdots \\ b_2^k \end{pmatrix}$$

where k is the number of DOFs tracked per node and j is the number of DOFs associated with the first block equation. 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 Section 5 for two particular Fluid-DFTs.
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. The partitioning used to distribute grid nodes can also be effective for partitioning each DOF of the linear system. Because DOFs are segregated, providing a fairly uniform nonzero pattern within each DOF block, this approach typically gives a well-balance data and work distribution for the solver.
2. 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.
3. The dimension of S is often a small fraction of the original matrix. Specifically, only the primitive densities, density functionals and a few scalar DOFs are part of the A_{22} block. All other variables, sometimes more than 80% of the total DOFs, are part of the A_{11} block. Thus, iterative methods such as GMRES will typically converge much faster because of the reduced dimension, independent of other factors such as preconditioning.
4. Assuming nodes of the mesh are equally partitioned on the parallel machine, each matrix block A^{pq} in Equation 4.1 is distributed evenly across the parallel machine. Thus, 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 specifics of this approach for two different 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. We note that while there are other classes of Fluid-DFTs, the general strategy of extending the system size to include convenient nonlocal variables and then designing efficient segregated solvers will generalize quite easily to the broader problem space.

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. 2.9). 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).

We will consider a case where 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. If all of the nonlocal density variables are independent, the A_{11} block (which is composed of non-local density interactions only) is

$$(5.1) \quad A_{11} = I.$$

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

However, for the special case of a single component fluid, some of the non-local densities may be trivially dependent on others with

$$(5.2) \quad n_\gamma(\mathbf{r}) = C n_\epsilon(\mathbf{r}).$$

where C is a constant. We generally encode the linear dependency in Eq.5.2. For the case of Rosenfeld's functionals presented in Appendix A.1 [27], this leaves $N_{nld} = 2+D$ variables that are described by integration stencils (nonzeros in the A_{12} block).

To address these dependencies, we further decompose the A_{11} block 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

$$(5.3) \quad A_{11} = \begin{pmatrix} -I & 0 \\ X & -I \end{pmatrix}.$$

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

$$(5.4) \quad A_{11}^{-1} = \begin{pmatrix} -I & 0 \\ -X & -I \end{pmatrix}.$$

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 with 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 A.7 and A.8 in Appendix A.2).

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

$$(5.5) \quad A_{11} = \begin{pmatrix} A_{11}^{11} & 0 & 0 & 0 \\ A_{11}^{21} & A_{11}^{22} & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots \\ A_{11}^{j1} & \dots & A_{11}^{j-1,j} & A_{11}^{jj} \end{pmatrix}.$$

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:

$$(5.6) \quad A_{22} = \begin{pmatrix} D_{11} & F \\ D_{21} & D_{22} \end{pmatrix}.$$

The first block of equations in A_{22} is associated with the unknown fields, $U(\mathbf{r})$ (Eqn. A.6), and the second block with the primitive densities, $\rho(\mathbf{r})$ (Eqn. A.5). 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 density of this block is dictated by the range of the direct correlation function in Eqn. A.6. The F block is by far the most dense submatrix in the global matrix. 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

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

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:

$$(5.7) \quad A_{22} \approx \tilde{A}_{22} = \begin{pmatrix} D_{11} & F \\ 0 & D_{22} \end{pmatrix}.$$

We then define a preconditioner P for S that consists of one block Gauss-Seidel (one back-solve) using \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 numerical problem (with a third 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. Each solver uses an unscaled residual tolerance of 10^{-4} . 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 code 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 redesigned 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 *GenS* to refer to a general-purpose preconditioned Krylov solver applied to the full linear system of equations ordered as described in Equation 4.1. We use an overlapping Schwarz preconditioner with ILUT [29] as the local subdomain solver via the Trilinos package IFPACK [32]. We use non-restarted GMRES [31] from the Trilinos package AztecOO [19] as the iterative method. Note that the *GenS* solver already an improvement over previous work [10] because Trilinos algorithms (an improvement over Aztec solvers) are used, and because the ordering of equations has changed from a node-first ordering to a DOF-first ordering. The DOF-first ordering results in very favorable ILUT fill and robustness properties for the general-purpose solver.

The segregated Schur solvers presented here will be denoted *SSS* and are based on preconditioned Krylov methods that use our new 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 *SSS* solver preconditioners are composed of between several and more than 100 Epetra distributed sparse matrices. The *SSS* 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 6.1 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.

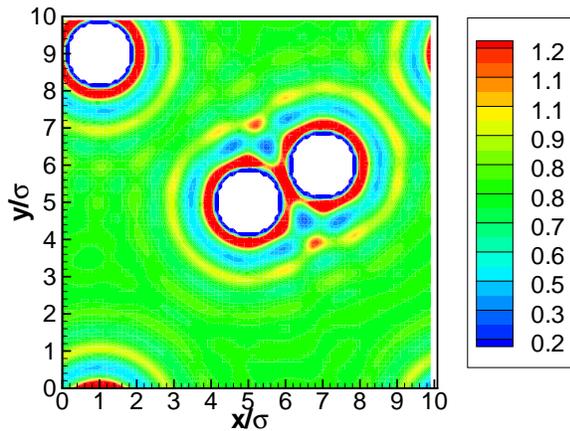


FIG. 6.1. Fluid density ($\rho\sigma^3$) distribution in the vicinity of 3 nanocylinders.

Table 6.1 compares the *SSS* and *GenS* approaches. *GenS* solutions were obtained 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 *SSS* 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 *SSS* 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

#Procs	<# Lin iters>		Time/ N_{iter}		T_{1Proc}/T (SSS)	T_{GenS}/T_{SSS}
	GenS	SSS	GenS	SSS		
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}		$T/T_{\Delta x=0.2}$ (SSS)	T_{GenS}/T_{SSS}
	GenS	SSS	GenS	SSS		
$\sigma/5$	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	-

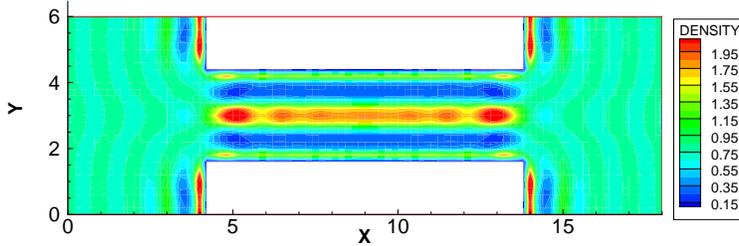
TABLE 6.1

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 GenS to the SSS 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 $\Delta x = \sigma/10$.

data at $\Delta x = 0.2\sigma$) we find that the scaling of the GenS code with mesh refinement is $T \propto N^{2.7}$, while the new code has $T \propto N^{2.0}$.

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. 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 converge at the higher density for the most refined mesh. Figure 6.2 shows the density profile for one slice (at $z = 3\sigma$) in the computational domain.

Performance results are presented in Table 6.2. Overall the results are very similar to the 2D problem of the previous section. The SSS algorithm improves on the performance of the GenS 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 SSS algorithm is $T \propto N^{1.65}$. For the GenS algorithm we find $T \propto N^{2.2}$.

FIG. 6.2. Fluid density ($\rho\sigma^3$) distribution for a hard sphere fluid in a finite length nanopore.

#Procs	N_{iter}	<# Lin iters>		Time/ N_{iter}		T_{AProc}/T (SSS)	T_{GenS}/T_{SSS}
		GenS	SSS	GenS	SSS		
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}		$T/T_{\Delta x=0.2}$ (SSS)	T_{GenS}/T_{SSS}
		GenS	SSS	GenS	SSS		
$\sigma/5$	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 6.2

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.

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 6.3 shows the solution we obtain for this problem.

Performance results for this 2D polymer test problem are shown in Table 6.3. 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 *GenS* 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

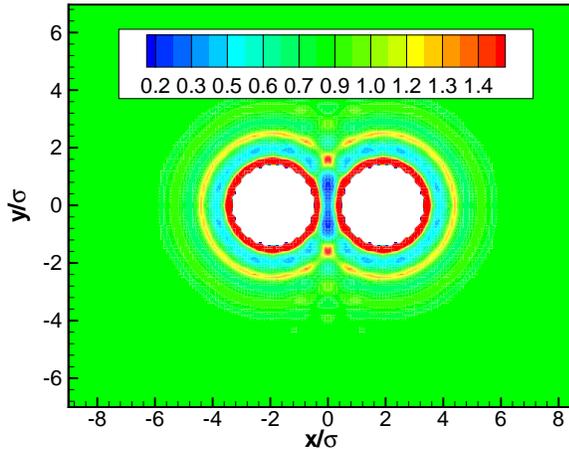


FIG. 6.3. Density ($\rho\sigma^3$) distribution for a 10-mer homopolymer near two nanocylinders. The computational domain was $1/4$ of this image with reflective boundaries at $x = 0$ and $y = 0$.

linear effect for the *SSS* algorithms while a steeper performance penalty occurs for the *GenS* method.

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 Molecular Dynamics simulation of the same models, and to map reasonably well to fluid experimental bilayers[9].

Figure 6.4 shows one slice through 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 6.4. In this case, the *SSS* algorithms provide the ability to solve this difficult problem as we were not able to solve this problem with the *GenS* approach despite using 128 processors and trying a variety of parameters for solver tolerances and ILUT fill factors. Finally, note that

#Procs	N_{iter}	<# Lin iters>		Time/ N_{iter}		T_{1Proc}/T (SSS)	T_{GenS}/T_{SSS}
		GenS	SSS	GenS	SSS		
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}		$T/T_{\Delta x=\sigma/10}$ (SSS)	T_{GenS}/T_{SSS}
		GenS	SSS	GenS	SSS		
$\sigma/10$	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}		$T/T_{N_{seg}=10}$ (SSS)	T_{GenS}/T_{SSS}
		GenS	SSS	GenS	SSS		
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 6.3

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 6.1. 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.

with the SSS approach we find a significant superlinear speed up for this case from 32 to 64 processors.

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

TABLE 6.4

Results for a 3D polymer test problem. Each case required 10 nonlinear iterations for a solution. For a description of all other columns see the table 1 caption. All data were generated with a mesh spacing of $\Delta x = \sigma/5$.

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.

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

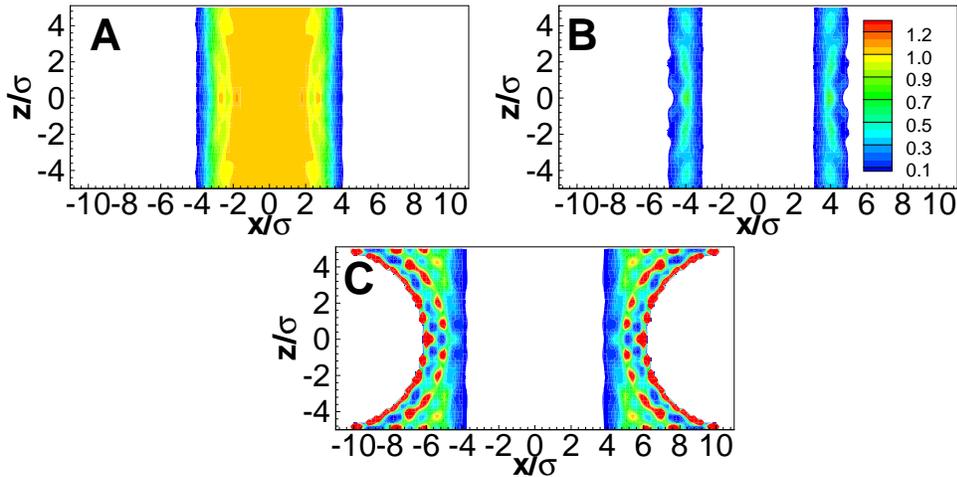


FIG. 6.4. One slice (at $y = 0$) through a 3-dimensional computational volume. The color contours show density ($\rho\sigma^3$) distributions for lipid tail beads (A), lipid head beads (B), and solvent (C) for a lipid bilayer assembly sandwiched between planar arrays of large spheres. The computational 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 for clarity.

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 *GenS* solver is comparable to the matrix: from $3k$ to $10k$ storage. The ILUT preconditioner for the *GenS* solver requires from $2k$ to $10k$ storage. Thus the *GenS* 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 *SSS* solvers over the *GenS* 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 *GenS* solver in that ILUT requires *ad hoc* parameters to determine 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, the *SSS* 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 *GenS* solver are quite remarkable. In all cases, once the number of processors is large enough for the *GenS* 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 *SSS* solvers scalability results are also superlinear, due to factor (i) above. However, since the *SSS* 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 *SSS* solvers cannot benefit as much as the *GenS* solver from large processor counts for the same global problem. However, the *SSS* 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 *GenS* solver.

6.6.4. Scalability in mesh density and chain length. The *SSS* 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 *GenS* solver iteration count also remains fixed as mesh density increases. However, the overall cost of the *GenS* solver grows much faster than the *SSS* solvers.

For polymer problems, the *SSS* 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 *GenS* 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. This fact makes the routine use of Tramonto much easier for complex applications.

Acknowledgments. The authors thank the MICS program under the Department of Energy and the referees for comments that led to an improved paper.

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. *J. Chem. Phys.*, 119:407–424, 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. *J. Chem. Phys.*, 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. *Density-Functional Theory of Atoms and Molecules*. 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.

Appendix 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 as a reference system with other physical effects (e.g. attractions, Coulomb interactions, and even bond constraints) being treated as a perturbations. The grand free energy functional for a multicomponent hard-sphere fluid is

$$(A.1) \quad \begin{aligned} \Omega[\{\rho_i(\mathbf{r})\}] &= \sum_i \int \rho_i(\mathbf{r}) [\ln \rho_i(\mathbf{r}) - 1] d\mathbf{r} + \\ &\int \Phi(\{n_\gamma[\{\rho_i\}]\}) d\mathbf{r} + \sum_i \int \rho_i(\mathbf{r}) [V_i(\mathbf{r}) - \mu_i] d\mathbf{r}. \end{aligned}$$

where V is a one body external field, and μ_i 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. 2.9. The Euler-Lagrange equation to be solved is

$$(A.2) \quad \ln \rho_i(\mathbf{r}) - V_i(\mathbf{r}) + \mu_i + \int \sum_\gamma \frac{\partial \Phi}{\partial n_\gamma}(\mathbf{r}') \frac{\delta n_\gamma(\mathbf{r}')}{\delta \rho_i(\mathbf{r})} d\mathbf{r}' = 0.$$

The energy density, Φ depends on a set of nonlocal variables, $\{n_\gamma\}$, where there are four scalar variables, n_0, n_1, n_2, n_3 and two vector variables, n_{V1}, n_{V2} . In Rosenfeld's original theory [27], the free energy density is

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

The block matrix formulation is built on both the Euler-Lagrange equation (residual denoted R_{EL}) and the definition of the nonlocal density variables (residual denoted R_{NL}), and may be written and the 2×2 block matrix is

$$(A.4) \quad \begin{bmatrix} -I & w_\gamma(\mathbf{r}, \mathbf{r}') \\ \sum_\gamma \frac{\partial^2 \Phi}{\partial n_\gamma \partial n_\epsilon}(\mathbf{r}) & D_{22}(\mathbf{r}) \end{bmatrix} \begin{bmatrix} \Delta n_\gamma \\ \Delta \rho \end{bmatrix} = - \begin{bmatrix} R_{NL}(\mathbf{r}) \\ R_{EL}(\mathbf{r}) \end{bmatrix},$$

where $D_{22}(\mathbf{r})$ is a diagonal block matrix, and each entry is $1/\rho(\mathbf{r})$.

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

$$(A.5) \quad \rho_\alpha(\mathbf{r}) = \frac{\rho_{b,\alpha}}{N_\alpha} \sum_{s \in \alpha} \frac{G_s(\mathbf{r})G_s^{inv}(\mathbf{r})}{\exp[-\beta U_\alpha(\mathbf{r})]}, \quad \alpha = 1..N_\alpha,$$

and

$$(A.6) \quad U_\alpha(\mathbf{r}) = V_\alpha(\mathbf{r}) - \sum_{\beta} \int c_{\alpha\beta}(\mathbf{r} - \mathbf{r}')(\rho_\beta(\mathbf{r}') - \rho_{b\beta})d\mathbf{r}', \quad \alpha = 1..N_\alpha.$$

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.A.5 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,

$$(A.7) \quad G_s(\mathbf{r}) = \exp[-\beta U_{\alpha(s)}(\mathbf{r})] \int \omega_{\alpha\beta}(\mathbf{r} - \mathbf{r}')G_{s-1}(\mathbf{r}')d\mathbf{r}', \quad s = 1..N_s,$$

and

$$(A.8) \quad G_s^{inv}(\mathbf{r}) = \exp[-\beta U_{\alpha(s)}(\mathbf{r})] \int \omega_{\alpha\beta}(\mathbf{r} - \mathbf{r}')G_{s+1}^{inv}(\mathbf{r}')d\mathbf{r}', \quad s = 1..N_s.$$

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.