

# Enabling Tools for Extreme Scale Computation of Nanoscale Fluids

**David Day, Amalie Frischknecht,  
Michael Heroux, Michael Parks  
Sandia National Laboratories**

**Laura Frink  
Colder Insights Corp.**

**Deaglan Halligan  
Purdue University**

**Kirk Soodhalter  
Temple University**

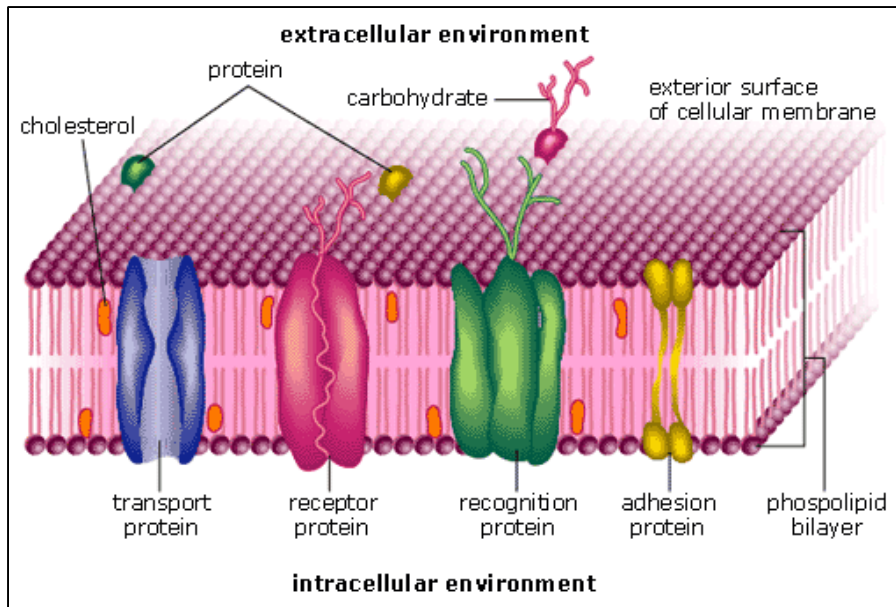


# Overview

- ❑ To realize promised performance of modern high-end multicore systems, we must develop new algorithmic capabilities to efficiently utilize multicore nodes
  - ❑ Increase performance by reducing node-level memory bandwidth and size usage
- ❑ Test several new algorithmic capabilities in Sandia's Tramoto Fluid DFT code
  - ❑ Ability to solve fluid-DFT governing equations in 3D and at large scales crucial to continued scientific progress
- ❑ Mixed-precision and precision-neutral algorithms
  - ❑ Leverage Trilinos/Tpetra (templated C++) solver stack
  - ❑ Performance and storage advantage of float over double
  - ❑ Utilize high-precision arithmetic if double inadequate
- ❑ Least-squares methods (LSQR)
  - ❑ Achieve robustness by dynamically adapting precision
  - ❑ Shield user from details of mixed-precision computation
- ❑ Block Krylov recycling methods
  - ❑ Recycling subspace information from previous solves to reduce iteration count
  - ❑ Block methods have superior convergence properties and computation to bandwidth requirements, improving processor utilization

# Nanostructured Fluids

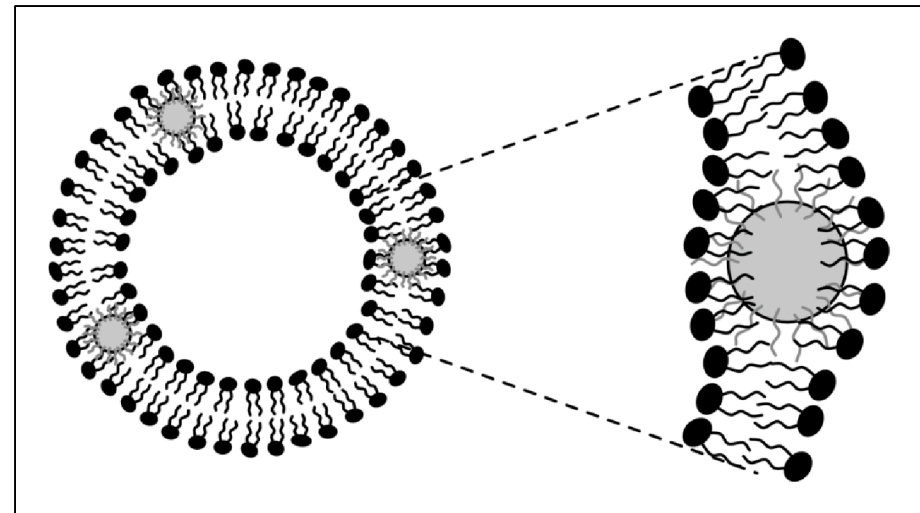
- ❑ Structure arises from surfaces, fields, self-assembly
- ❑ Density, diffusion, and viscosity different from bulk fluid properties
- ❑ Rich phase behavior: wetting, capillary condensation, layering



## Biological Membranes

Self-assembled fluid bilayer packed with proteins, peptides, etc.

**Engineered Systems**  
Lipid vesicle/nanoparticle assemblies for drug delivery)



# Density Functional Theory for Fluids

- Enable modeling and simulation of a wide range of applications, including fluids at interfaces, colloidal fluids, wetting, porous media, and biological mechanisms at the cellular level
- Given external field  $\mathbf{V}(\mathbf{r})$ , determine structure of inhomogeneous fluid as captured by density distribution  $\rho(\mathbf{r})$  via minimization of free energy functional  $\Omega(\rho(\mathbf{r}))$

$$\Omega[\rho(\mathbf{r})] = F_{\text{id}} + F_{\text{hs}} + F_{\text{vdW}} + F_{\text{c}} + F_{\text{assoc}} + \int \rho(\mathbf{r})[\mathbf{V}(\mathbf{r}) - \mu]$$

Coulomb interactions
Associations (H-bonding)
[Applied field]

Ideal gas
Hard sphere
Dispersion attractions

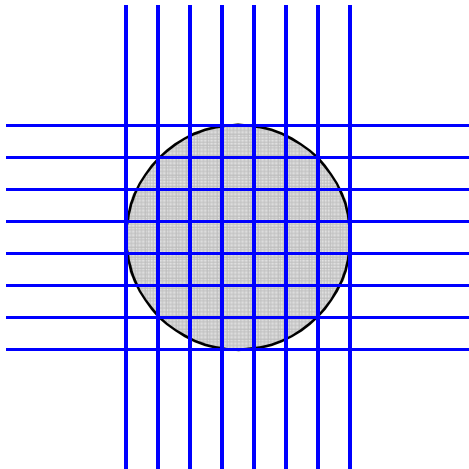
Legendre transform from Canonical to Grand Canonical ensemble

- Solve  $\left( \frac{\delta \Omega}{\delta \rho(\mathbf{r})} \right)_{T, \mu} = \mathbf{0}$  with Newton-Krylov.
- Use Sandia's *Tramonto* package for complex fluid systems
  - Built upon *Trilinos* software components: [trilinos.sandia.gov](http://trilinos.sandia.gov)
  - Open source: [software.sandia.gov/tramonto/](http://software.sandia.gov/tramonto/)

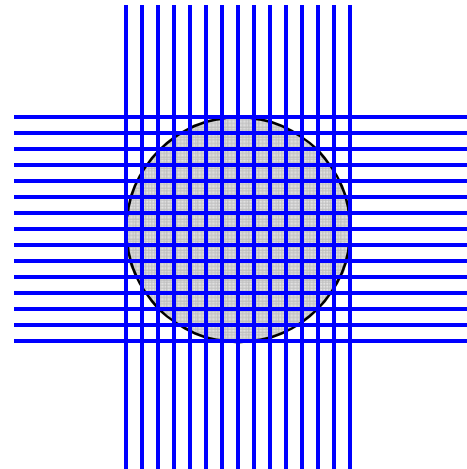


# Numerical Methods for Fluid-DFTs

- ❑ Discrete formulation
  - ❑ Uniform structured grid
  - ❑ Discretize using collocation at mesh points
  - ❑ Linear interpolation between mesh points
- ❑ Newton: Convergence in  $O(10)$  iterations
- ❑ Linear system properties (different than discrete PDEs)
  - ❑ Strong interphysics coupling
  - ❑ Large number of DOF/node
  - ❑ Nonlocal integral equations  
(matrix sparsity dependent upon mesh)



Coarse Mesh  
(Few nonzero per row)



Fine Mesh  
(Many nonzero per row)

# Segregated Schur Complement Solvers\*

- Resulting linear systems take the form

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

Each  $\mathbf{A}_{ij}$  has own physics-based block structure


- Careful ordering of unknowns makes it advantageous to solve Schur complement

$$\mathbf{S}\mathbf{x}_2 = \mathbf{f}$$

where

$$\mathbf{S} = \mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12} \quad \mathbf{f} = \mathbf{b}_2 - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{b}_1$$

- Schur system may have up to 80% fewer dofs
- Big win for hard sphere systems:  $\mathbf{A}_{11}$  is diagonal!
- Similar favorable structure to  $\mathbf{A}_{11}$  for polymer problems using Chandler-McCoy-Singer (CMS) DFT
- More complex structure for WJDC (Werthim, Jain, Dominik, and Chapman) DFT




# Mixed-Precision and Precision Neutral Computation

- ❑ Rewrite Tramoto solver managers to utilize Tpetra solver stack
  - ❑ Template scalar, local ordinal, and global ordinal types
  - ❑ Avoid 4GB limit of int - allow arbitrarily large problems (exascale necessity)
  - ❑ Arbitrary scalar types: float, complex, qd\_real
- ❑ Enhance performance while maintaining solution accuracy
- ❑ Speedup by replacing double with float
- ❑ Example polymer problem from Tramoto (8 linear solves inside Newton loop)

NProc	Float	Double	Speedup
1	3.753	10.970	2.923
2	1.766	4.195	2.375
3	1.203	2.086	1.734
4	1.380	2.643	1.915
5	1.211	2.460	2.031
6	1.056	2.313	2.190
7	1.036	2.057	1.986
8	1.524	2.387	1.566



# Block Recycling Linear Solvers

- ❑ Leverage two important algorithmic techniques: Krylov recycling + block methods
- ❑ Krylov subspace recycling 
  - ❑ In Krylov subspace methods, building search space is dominant cost
  - ❑ For sequences of systems, get fast convergence rate and good initial guess immediately by recycling selected search spaces from previous systems
  - ❑ Family of recycling methods: Recycling GMRES (GCRODR), recycling CG (RCG), recycling MINRES (RMINRES), recycling BiCG (RBiCG).
- ❑ Block methods
  - ❑ Performance advantages over single-vector methods (BLAS 1 → BLAS3, SpMV → SpMM)
  - ❑ Reduce per-core bandwidth usage
  - ❑ Introduce fictitious right-hand-sides to enhance search space

# Block Recycling GMRES (BGCRODR)

## Block Recycling GMRES New!

Implemented in Trilinos/Belos package (C++, templated)

(1) Solve  $\mathbf{A}_1 \mathbf{X}_1 = \mathbf{B}_1$

(2) Compute  $k$  recycle vectors  $\mathbf{U}_k$  (for example, harmonic Ritz vectors)

(3) Solve next linear system  $\mathbf{A}_2 \mathbf{X}_2 = \mathbf{B}_2$  by iterating orthogonally to image of  $\mathbf{U}_k$ :

$$\mathbf{A}_2 \begin{bmatrix} \mathbf{U}_k & \mathbf{W}_m \end{bmatrix} = \begin{bmatrix} \mathbf{C}_k & \mathbf{W}_{m+1} \end{bmatrix} \begin{bmatrix} \mathbf{I}_k & \mathbf{B}_k \\ \mathbf{0} & \mathbf{H}_m \end{bmatrix}$$

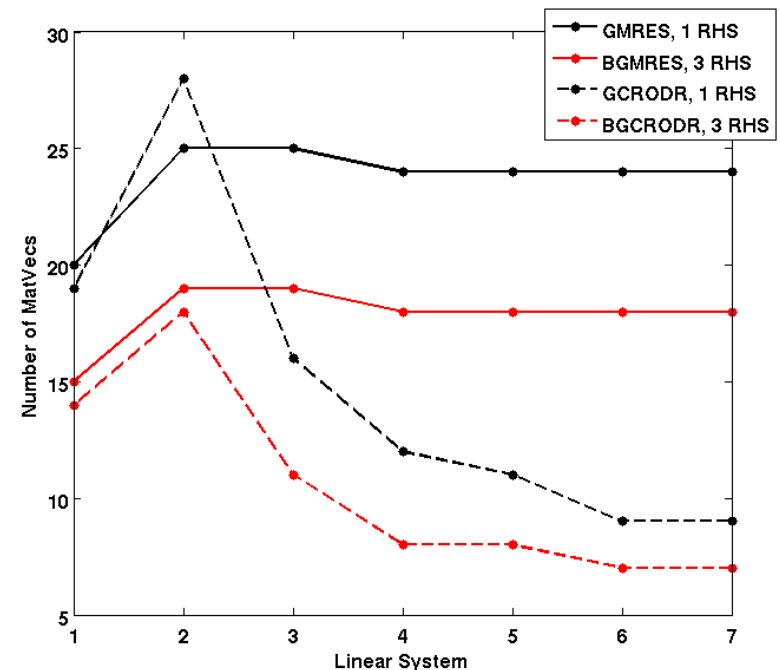
$$\mathbf{B}_k = \mathbf{C}_k^H \mathbf{A} \mathbf{W}_m \quad \mathbf{C}_k = \mathbf{A}_2 \mathbf{U}_k \quad \mathbf{C}_k^H \mathbf{C}_k = \mathbf{I}_k$$

(4) Repeat

Example hard sphere problem from Tramonto (electrostatics + attractions)

7 linear solves in from Newton loop

Savings: 60 matvecs / 36% (1 RHS),  
50 matvecs / 40%, (3 RHS)



BGCRODR on  
Tramonto Example

# LSQR

- ❑ LSQR (ref)
  - ❑ Implemented in Trilinos/Belos package (C++, templated) **New!**
- ❑ Krylov method for  $\mathbf{Ax}=\mathbf{b}$  based upon Golub-Kahan bidiagonalization process
- ❑ Algebraically equivalent to MINRES applied to normal equations  $\mathbf{A}^H\mathbf{Ax}=\mathbf{b}$ , but with better numerical properties (especially if  $\mathbf{A}$  ill-conditioned)
- ❑ Governing equations

$$\mathbf{A}^H\mathbf{U}_k = \mathbf{V}_k\mathbf{B}_k^H \quad \text{span}(\mathbf{U}_k) = \mathcal{K}(\mathbf{AA}^H, \mathbf{b})$$

$$\mathbf{AV}_k = \mathbf{U}_{k+1}\bar{\mathbf{B}}_k \quad \text{span}(\mathbf{V}_k) = \mathcal{K}(\mathbf{A}^H\mathbf{A}, \mathbf{b})$$

$$\|\mathbf{b} - \mathbf{Ax}_k\| = \min_{\mathbf{y}} \|\mathbf{b} - \mathbf{AV}_k\mathbf{y}\| = \min_{\mathbf{y}} \|\mathbf{e}_1\beta - \bar{\mathbf{B}}_k\mathbf{y}\|$$

- ❑ Short-term recurrence; Fixed memory-footprint
- ❑ Sharp estimates of  $\|\mathbf{A}\|$ ,  $\|\mathbf{A}^{-1}\|$  -> estimate of  $\text{cond}(\mathbf{A})$



# LSQR

- ❑ Robustness under reduced precision
  - ❑ Return least-squares solution to  $Ax=b$  even if  $A$  is numerically singular due to use of lower precision
  
- ❑ Adaptive precision solver
  - (1) Solve  $Ax=b$  in float
  - (2) If  $\text{cond}(A) < \text{machEpsSingle}$  return
  - (3) Else solve  $Ax=b$  in double
  - (4) If  $\text{cond}(A) < \text{machEpsDouble}$  return
  - (5) Else solve  $Ax=b$  in double-double
  - (6) If  $\text{cond}(A) < \text{machEpsDouble-Double}$  return
  - (7) ...



## Summary

- ❑ Tested several new algorithmic capabilities in Sandia's Tramonto Fluid DFT code
- ❑ Improved performance via reduction of node-level memory bandwidth and size usage
  
- ❑ Mixed-precision and precision-neutral algorithms
  - ❑ Leverage Trilinos/Tpetra (templated C++) solver stack
  - ❑ 2x or more speedup with float instead of double
  - ❑ High-precision arithmetic if double inadequate
  
- ❑ Least-squares methods (LSQR)
  - ❑ Couple with mixed-precision computation to shield user from mixed-precision computation. Achieve robustness by dynamically adapting precision.
  
- ❑ Block Krylov recycling methods
  - ❑ Recycling subspace information from previous solves to reduce iteration count
  - ❑ Block methods have superior convergence properties and computation to bandwidth requirements, improving processor utilization