Poisson-Boltzmann Electrostatics on a GPU

Victor Ovchinnikov

Department of Chemistry and Chemical Biology

Harvard University

PB Electrostatics on a GPU

- Electrostatics play central role in many biological processes
  - Protein/protein & protein/DNA interactions
  - moderately to highly-charged macromolecules
- Effects of aqueous solvent on properties of solute
- Electrostatic potential related to Free Energy of polar solvation
  - Poisson Boltzmann Equation (PBE) for electrostatic potential
    - Extension of Debye-Huckel theory
    - Nonlinear
    - Impossible to solve analytically for complex systems
    - How fast can PBE be solved on a GPU?

Why PBE for implicit solvation?

- Why implicit solvation
  - Solvation Free Energy computed explicitly (e.g. how important are solvent forces for stability)
  - No need to integrate solvent DOF
  - Solvent viscosity absent → improved sampling !
- Arguably “best” theory of solvation
  - Approximate solutions to PBE is the basis of existing implicit solvation models
    - GBSW, FACTS (CHARMM)

PB Equation

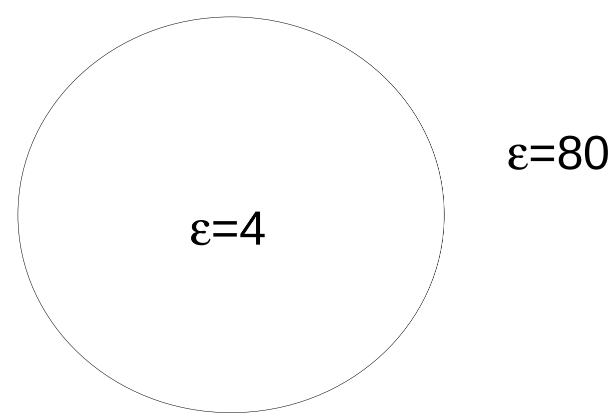
- 1-1 electrolyte

$$-\nabla \cdot (\epsilon(\boldsymbol{r})\nabla\Phi(\boldsymbol{r}) + \kappa^2 \left(\frac{1}{\beta e}\right) \sinh(\beta e \Phi(\boldsymbol{r})) = 4\pi \sum_{i=1}^{N} q_i \delta(\boldsymbol{r} - \boldsymbol{r}_i)$$

- Linearization under low ionic strength

$$-\nabla \cdot (\epsilon(\boldsymbol{r})\nabla\Phi(\boldsymbol{r}) + \kappa^2 \Phi(\boldsymbol{r}) = 4\pi \sum_{i=1}^{N} q_i \delta(\boldsymbol{r} - \boldsymbol{r}_i)$$

- Nonuniform dielectric ε
  - ~80 in water; 2-8 in protein interior
- Internal BC
  - Flux matching $\epsilon(\boldsymbol{r})\nabla\Phi(\boldsymbol{r})$



Solution methods

$$-\nabla \cdot (\epsilon(\boldsymbol{r})\nabla\Phi(\boldsymbol{r})) + \kappa^2\Phi(\boldsymbol{r}) = 4\pi\sum_{i=1}^{N} q_i\delta(\boldsymbol{r}-\boldsymbol{r}_i)$$

- FFT + cyclic reduction
  - Problem: Nonuniform coefficients + nonlinear terms
- Fast multipole methods
  - Problem: Nonlinear terms, difficult to code (L. Barba's group at BU)
- Conjugate gradient methods
- Multigrid methods (APBS, MEAD)
  - Weak nonlinearity permissible
  - Conceptually simple
- Lattice Boltzmann method (e.g. adopt a CFD GPU code)

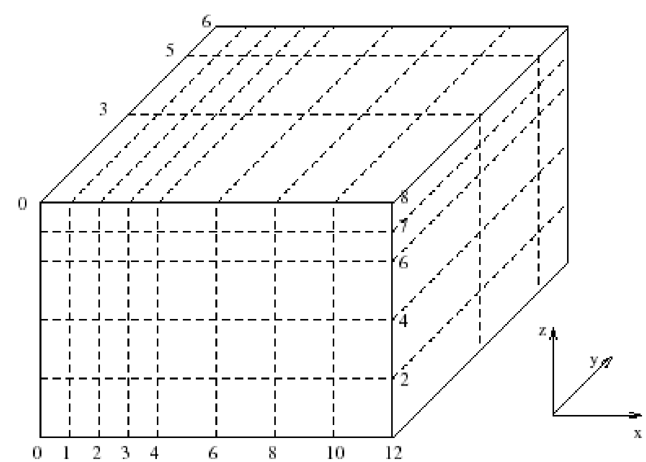
FD discretization (linear PBE)

$$-\frac{\frac{\Phi_{i+1,j,k}-\Phi_{i,j,k}}{dx_{i+1}} \times \frac{1}{2}(\epsilon_{i+1,j,k}+\epsilon_{i,j,k}) - \frac{\Phi_{i,j,k}-\Phi_{i-1,j,k}}{dx_i} \times \frac{1}{2}(\epsilon_{i-1,j,k}+\epsilon_{i,j,k})}{\frac{1}{2}(dx_{i+1}+dx_i)} + \ldots + \kappa^2_{i,j,k}\Phi_{i,j,k} = \rho_{i,j,k}$$

- Nonuniform mesh to concentrate points in region of interest
- Can be written as

$$A_h \mathbf{\Phi}_h = \rho_h$$

  - A is hepta-diagonal
  - ρ is charge density



MG Solution procedure (LPBE)

$$A_h \Phi_h = \rho_h$$

Iterate with a Jacobi or Gauss-Seidel

$$\rho_{2h} = P(\rho_h - A_h \Phi_h^*)$$

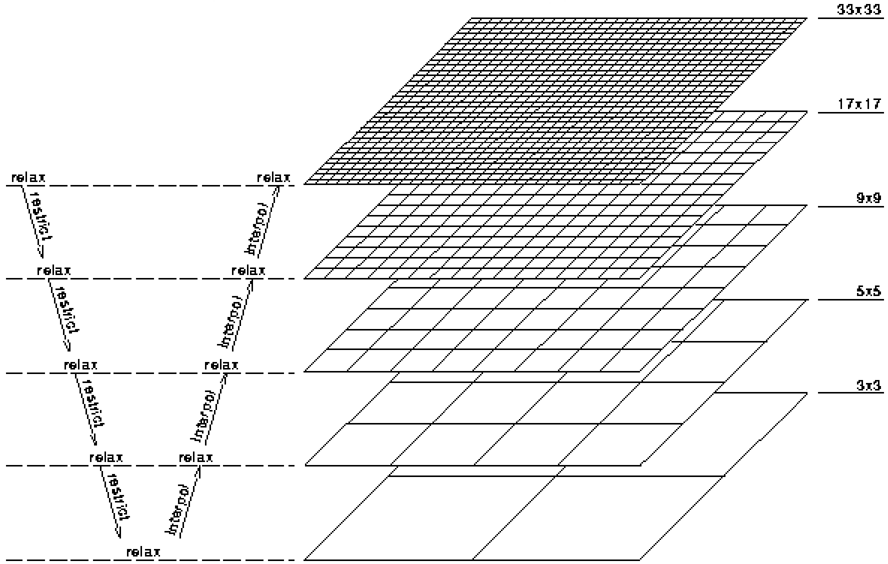
Project residual onto coarser grid

$$A_{2h} \Phi_{2h} = \rho_{2h}$$

Iterate coarser solution.
If coarsest level reached, return;
Otherwise, recurse

$$\Phi_h' = \Phi_h^* + I \Phi_{2h}^*$$

Interpolate coarse solution onto finer grid and add to approximate fine solution



Repeat cycle until fine-grid residual within tolerance

Challenges for GPUs

- Jacobi iterations require little arithmetic, but a lot of memory access
- Stencil coefficients variable
  - Nonuniform dielectric; nonuniform mesh
    - Recompute on-the-fly?
  - Store in constant memory?
    - Dielectric is constant in most regions of space
- Work load decreases with coarsening
  - Fewer grid points
  - Not all threads may be active
  - Terminate kernel & resize blocks?
- Internal boundary conditions
  - Can “cheat” by smearing boundary (for now)