LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-TR-218083

Incorporating Electrokinetic Phenomena into EBNavierStokes

K. Chu, D. Trebotich

January 11, 2006

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Incorporating Electrokinetic Phenomena into `EBNavierStokes`

Kevin T. Chu[†] David Trebotich[‡]

CSGF Practicum
June 1, 2004 - August 31, 2004
ISCR/CASC at Lawrence Livermore National Laboratory

Abstract

Motivated by the recent interest in using electrokinetic effects within microfluidic devices [1], we have extended the `EBNavierStokes` code to be able to handle electrokinetic effects. With this added functionality, the code will become more useful for understanding and designing microfluidic devices that take advantage of electrokinetic effects (*e.g.* pumping and mixing).

Supporting the simulation of electrokinetic effects required three main extensions to the existing code:

1. addition of an electric field solver,
2. development of a module for accurately computing the Smulochowski slip-velocity at fluid-solid boundaries, and
3. extension of the fluid solver to handle nonuniform inhomogeneous Dirichlet boundary conditions.

The first and second extensions were needed to compute the electrokinetically generated slip-velocity at fluid-solid boundaries. The third extension made it possible for the fluid flow to be driven by a slip-velocity boundary condition (rather than by a pressure difference between inflow and outflow). In addition, several small changes were made throughout the code to make it compatible with these extensions.

This report documents the changes to the `EBNavierStokes` code required to support the simulation of electrokinetic effects. We begin with a brief overview of the problem of electrokinetically driven flow. Next, we present a detailed description of the changes to the `EBNavierStokes` code. Finally, we present some preliminary results and discuss future directions and improvements to the code.

Overview

Physics of Electrokinetic Phenomena

The origin of electrokinetic phenomena is the interaction of an applied electric field with the thin electrically charged layer that forms at fluid-solid interfaces. In this interfacial region, the electric field induces a body force in the charged fluid which drives fluid flow. However, because the electrically charged region of the fluid, often referred to as the diffuse-charge or Debye screening layer, is very thin (typically only 1 to 100 nm thick), the the coupling between the electric field and the fluid flow only shows up in the mathematical formulation as a slip-velocity[1]; Smulochowski's formula quantifies this relationship

$$\vec{u}_{slip} = b\vec{E}_{||} \tag{1}$$

where b is the electroosmotic or electrophoretic mobility and $\vec{E}_{||}$ is the tangential component of the electric field at the fluid-solid boundary.

[†]Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA 02139
[‡]Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA 94551
[1]This simple coupling between the electric field and fluid flow is only valid in the thin-double limit [2].

Mathematical Formulation

Electrokinetically driven flows are governed by the incompressible Navier-Stokes equations for the fluid flow and Laplace's equation for the electrostatics problem:

$$\begin{aligned} \vec{u}_t + (\vec{u} \bullet \nabla)\vec{u} &= -\frac{1}{\rho}\nabla p + \nu \triangle \vec{u} \qquad (2) \\ \nabla \bullet \vec{u} &= 0 \qquad (3) \\ \triangle \phi &= 0. \qquad (4) \end{aligned}$$

Here $\vec{u}$ is the fluid velocity, p is the pressure, ϕ is the electric potential; ρ and ν are the fluid density and kinematic viscosity, respectively. For fluid flows driven by electrokinetic effects, the usual "no-slip" boundary conditions at fluid-solid interfaces are replaced by the Smulochowski slip-velocity, Eq. (1), for the tangential component of the velocity. Since fluid is still not allowed to penetrate the surface, we still impose a "no-flux" condition for the normal component.

The boundary conditions for the electrostatics problem can be a bit more complicated depending on the material properties of the solid phase and the dynamics of diffuse-layer charging [1, 2]. For the purposes of this project, we have assumed that the the solid phase is nonpolarizable and that the characteristic time-scale of diffuse-layer charging is very small relative to the characteristic time-scale of the fluid flow. These approximations allow us to use a simple homogeneous Neumann boundary condition for the electrostatics problem.

It is worth noting that the electrostatics problem is completely decoupled from the fluid flow. As a result, the full problem can be solved by first solving for the electric field and subsequently solving the Navier-Stokes equations using boundary conditions calculated from the solution to the electrostatics problem.

Numerical Algorithm

To compute numerical solutions to Eqns. (4), we use the following algorithm:

1. Solve for the electrostatic potential problem using a standard multigrid algorithm.
2. Compute the cell-centered electric field througout the physical domain[2].
3. Compute the cell-centered slip-velocity at fluid-solid boundaries using Smulochowski's formula.
4. Compute the face-centered slip-velocity at fluid-solid boundaries by extrapolating the cell-centered values to faces.
5. Solve the incompressible Navier-Stokes equations using a projection method for incompressible fluid flow based on the Bell-Colella-Glaz (BCG) algorithm [3, 4].

1 Implementation

The starting point for the electrokinetic flow simulation code was `EBNavierStokes`, a the microfluidic simulation code developed by D. Trebotich in CASC [3]. This code provided the core solver for the incompressible Navier-Stokes equations. In this section, we document the modifications made to the code to support electrokinetic effects.

1.1 New Capabilities

Extension of this code to support electrokinetic effects involved adding three new features to the existing code:

- a solver for the electrostatics problem which calculates the electric field throughout the domain

[2]Technically, we only require the electric field at fluid-solid boundaries, so performance of the code could be improved by eliminating the computation of the field in the interior of the domain.

- a module that computes the slip-velocity at fluid-solid interfaces from mobility and electric field data, and
- extension of the `Operator` (manages operations involving the elliptic operator $(\alpha I - \triangle)$) to handle nonuniform inhomogeneous Dirichlet boundary conditions.

1.1.1 Class `ElectrostaticsSolver`

The `ElectrostaticsSolver` class encapsulates the solution of Laplace's equation for the electric potential. Physical and numerical solution parameters are set using the `define()` method. Computation of the solution can be explicitly initiated using the `solveElectrostaticsProblem()` method. Alternatively, the user may invoke the `getElectricPotential()` and `getElectricField()` methods; both of these methods will call `solveElectrostaticsProblem()` if the solution has not yet been computed.

In the `ElectrostaticsSolver` class, Laplace's equation is solved using a standard multigrid method that is implemented in the `MG` and `Operator` classes. The electric field is computed as a cell-centered quantity by invoking the `Operator::Gradient()` method.

1.1.2 Class `EKModule`

The `EKModule` class encapsulates the physics behind electrokinetic phenomena. As with the `ElectrostaticsSolver` class, physical and numerical parameters are set using the `define()` method. In addition, the electroosmotic mobility (currently identitcal at all fluid-solid boundaries) can be set independently using the `setMobility()` method. Once the solver is defined, the slip-velocity can be calculated directly by invoking the `calculateSlipVelocity()` method. Alternatively, the user may just request the slip-velocity through the `getSlipVelocity()` or `getMACSlipVelocity()` methods; both of these methods invoke `calculateSlipVelocity()` if the slip-velocity has not yet been computed.

The `calculateSlipVelocity()` method computes the slip-velocity by first requesting the electric field from the `ElectrostaticsSolver`. Next, it multiplies the field by the electroosmotic mobility throughout the entire domain to obtain a cell-centered "slip-velocity" throughout the region[3]. At embedded boundaries, the cell-centered slip-velocity is replaced by a first-order approximation that is calculated by correcting the cell-centered value using the following formula for each of the velocity components:

$$\phi_{EB} = \frac{\phi_{CC} - \delta\vec{x}_o \bullet \left[A^+ (\phi_1, ..., \phi_p)^T \right]}{1 - \delta\vec{x}_o \bullet \left[A^+ (1, ..., 1)^T \right]}. \tag{5}$$

Here ϕ_{CC} and ϕ_{EB} are the values of ϕ at the cell-center and embedded boundary centroid, respectively, $\delta\vec{x}_o = \vec{x}_{CC} - \vec{x}_{EB}$, and A^+ is the pseudoinverse of $A = (\delta\vec{x}_1, ..., \delta\vec{x}_p)^T$. In the matrix A, p is the number of neighboring cells used in the stencil for calculating ϕ_{EB} and $\delta\vec{x}_m = \vec{x}_m - \vec{x}_{EB}$ is the displacement of the m-th neighbor from the centroid of the embedded boundary. Refer to Appendix A for a more detailed derivation of this formula.

In addition to the cell-centered slip-velocity, a face-centered slip-velocity is computed on the outer faces of the computational domain using a third-order extrapolation from the cell-centered slip-velocity. Note that the slip-velocity on the outer faces is computed before the slip-velocities at embedded boundaries are updated.

1.1.3 Class `OperatorNonuniformBC`

The `OperatorNonuniformBC` class extends the `Operator` class to support nonuniform boundary conditions by overriding the `Action()`, `RegCells()`, `IrregCells()`, and `inhomDirichletDomainBdInIrregCell()` methods. A more detailed description of the changes to these methods follows.

- `Action()`

[3]There is no reason to calculate the product of the mobility and the electric field away from the physical boundaries. It was just easier to write the code to carry out this calculation in all cells. Streamlining this calculation will become more important for problems that have time varying electric fields because slip-velocity would have to be recalculated at each time step

  - Added arguments that pass in boundary data.
  - Pass the boundary data along to `RegCells()`, `IrregCells()`, and `inhomDirichletDomainBdInIrregCell()`.

- `RegCells()`
  - Added an argument to pass in cell-centered boundary data.
  - Modified the logic for determining which boundary condition to apply. The new logic treats all directions identically and selects which boundary condition to apply based on the domain boundary condition flags passed in when defining the `OperatorNonuniformBC` object.
  - Changed Fortran subroutine called for inhomogeneous Dirichlet boundary conditions to `FORT_NU_EDGEINHOMDIRICHLET()` which is able to handle nonuniform boundary values.

- `IrregCells()`
  - Added arguments to pass in boundary data.
  - Added a contribution from the boundary data to the computation of $(\alpha I - \triangle)$ when the user specifies an inhomogeneous Dirichlet boundary condition on embedded boundaries. The change only required making use of the inhomogeneous Dirichlet contribution weights calculated in `Operator::trebDirichletStencil()`.

- `inhomDirichletDomainBdInIrregCell()`.
  - Added arguments to pass in boundary data.
  - Modified the logic for determining which boundary condition to apply. The new logic treats all directions identically and selects which boundary condition to apply based on the domain boundary condition flags passed in when defining the `OperatorNonuniformBC` object.

The `FORT_NU_EDGEINHOMDIRICHLET()` subroutine called in `RegCells()` carries out the same calculation as the `FORT_EDGEINHOMDIRICHLET()` subroutine in `MGF.ChF` except that the inhomogeneous Dirichlet boundary condition is computed using a third-order extrapolation of the cell-centered boundary data instead of a single constant value passed into the subroutine.

1.2 Modifications to Existing Code

To support the application of nonuniform inhomogeneous Dirichlet boundary conditions, some modifications needed to be made to the previously existing code.

1.2.1 Class `Advection`

The `Advection` class was modified to replace hard-coded inflow/outflow boundary conditions with logic that sets the boundary conditions based on the domain boundary input flags used to define the `Advection` object. These changes affected the following methods: `macVelocities()`, `macGradient()`, and `macCorrection()`. In addition, an extra argument was added to `macVelocities()` and `macCorrection()` to pass in boundary data.

1.2.2 Main Program: `conTest.cpp`

The main program required a few modifications to incorporate the electrokinetics components into the simulation.

- A calculation of the slip-velocity was inserted before the main time-stepping loop for solution of the Navier-Stokes equation. This minor change involved defining an `EKModule` object and requesting the cell- and face-centered slip-velocity.

- The code that sets the boundary conditions in the MAC velocity construction was replaced with code that the boundary conditions based on the input flags for the domain boundary conditions. Previously, this was hard-coded for inflow/outflow.

- The time-step calculation was modified to:
  1. include the slip-velocities in the determination of the stable maximum convective time step size
  2. ensure that the time step size does not exceed the maximum stable viscous time step given by $C(dx)^2/\nu$ where C is a constant and ν is the kinematic viscosity.

  To support (1), we add a computation of the maximum slip-velocity on the boundaries in the `main()` program and an extra comparison in the `timestep_calculation()` function. The second issue was addressed by adding a comparison of the maximum convective time step size to the viscous time step size.

- A few additional parameters were added to the input file: electrokinetics parameters and boundary condition flags.

1.3 Bug Fixes

In the course of extending the `EBNavierStokes` code to handle electrokinetic effects, a few bugs were found. Fixes to these bugs are documented here.

1.3.1 Class `Operator`

The `Operator::Divergence()` method was an older version that carried out several unnecessary computations on regular cells. This code was replaced by the analogous code from `Advection::macDivergence()`.

1.3.2 Fortran Subroutines: `MGF.ChF`

In `EDGELAPLACIANDIRICHLET()` and `EDGEINHOMDIRICHLET()`, the calculation of the contributions to the Laplacian was inconsistent. `EDGELAPLACIANDIRICHLET()` used a zeroth-order approximation while `EDGEINHOMDIRICHLET()` used a first-order approximation. This inconsistency led to an incorrect solution of Laplace's equation for the electrostatic potential. To fix this problem, the calculation in `EDGELAPLACIANDIRICHLET()` was replaced with a first-order approximation.

It is worth noting that the error arises because the zeroth-order and first-order approximations do not converge to the same value as the grid spacing approaches zero. As a result, the value of the inhomogeneous boundary condition that is removed is inconsistent with the homogeneous form of the operator used in the multigrid solve. The important thing to realize is that the error does **not** arise solely because the zeroth-order approximation does not converge.

In addition to this bug, there was also a sign error in calculation of boundary contributions to the Laplacian: an extra factor of `lohi` multipled the contribution. This factor has been removed.

2 Preliminary Results

2.1 Verification and Validation

We are at the very early stages of verifying and validating the simulation. Currently, we have two main test problems:

1. 2D flow in an empty rectangular channel with a DC electric field applied along the direction of the channel.
2. 2D flow around a circular post (with the same electroosmotic mobility as the channel walls) in a rectangular channel with a DC electric field applied along the direction of the channel.

For the first test problem, the full analytical solution is known for both the steady-state and time-dependent problems. For the steady problem, the velocity field is simply a plug flow with velocity given by $b\vec{E}_{app}$. For the time-dependent problem, the velocity field is completely directed in the direction of the channel and only depends on the coordinate direction perpendicular to the channel:

$$u(y,t) = b\vec{E}_{app}\left[1 - \frac{4}{\pi}\sum_{n\ odd}\frac{1}{n}\exp(-\nu n^2\pi^2 t)\sin(n\pi y)\right] \tag{6}$$

where $\vec{u} \equiv (u, v)$ and y is in the direction perpendicular to the channel. Note that Eq. (6) is simply the solution to the 1D heat equation with an initial temperature of zero in the domain and fixed temperature boundary conditions. A comparison of the numerical solution to analytical results should be straightforward. Figure 1 shows that the simulation is qualitatively correct; the slip-velocity at the boundary gradually diffuses into the interior of the channel and eventually reaches a steady-state which is a plug flow with the expected value of 0.1 cm/s.

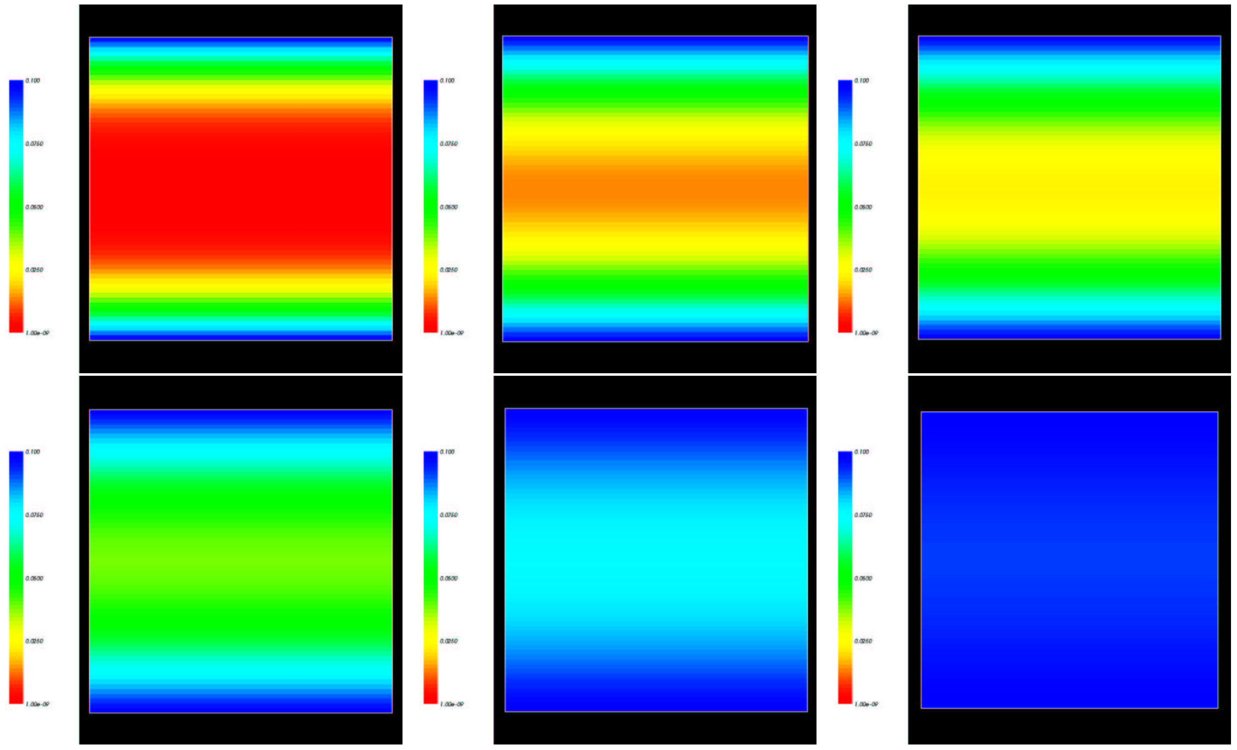

Figure 1: Fluid flow in a rectangular channel driven by an electrokinetic slip-velocity at the top and bottom boundaries at times $t = 0.00012$s, $t = 0.00037$s, $t = 0.00049$s, $t = 0.00073$s, $t = 0.00171$s, and $t = 0.00281$s (ordered left to right, top to bottom). A potential drop of 1 V in the x direction is instantaneously switched on at $t = 0$s. For reference, the channel is 100 μm wide, the fluid is an aqueous solution with a kinematic viscosity of 0.01cm^2/s, and the electroosmotic mobility is 0.001. With this set of parameters, the steady-state plug flow should have a velocity of 0.1cm/s which is what is essentially what is observed by time $t = 0.00281$s.

For the second problem, there is no analytical solution, but it is known that the steady-state solution is a potential flow that is proportional to the electric field with the constant of proportionality given by the electroosmotic mobility [5]. In this case, we can check that the electric field and steady-state flow fields are related by $\vec{u} = b\vec{E}$ everywhere in space (not just at the fluid-solid boundaries). Figure 2 shows that near steady-state, the fluid velocity field produced by the simulation is in good agreement with this expectation. While the fluid has not quite yet reached steady-state, the similarity between the electric and velocity fields is evident. The value of the mobility computed by taking a few random points in the physical domain yields $b = 7.81e-04$ which is getting close to the electroosmotic mobility value of 0.001 prescribed at the boundaries.

Figure 3 shows the time evolution of the fluid flow in response to an instantaneously applied electric field at $t = 0$s.

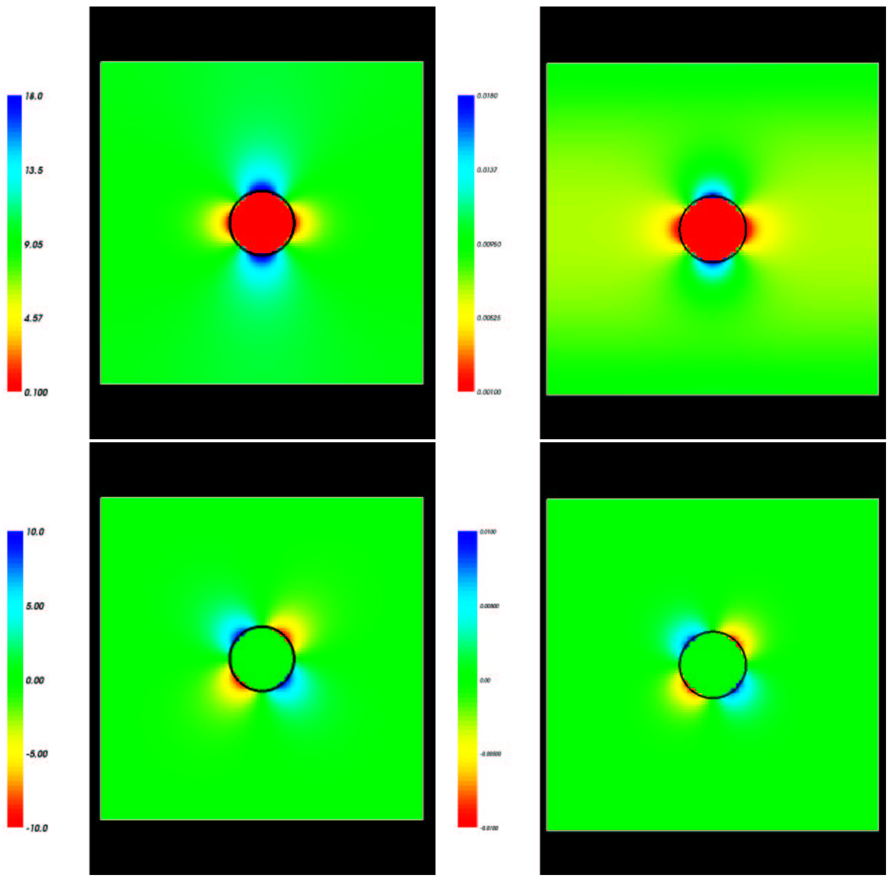

Figure 2: Comparison of electric field to steady fluid flow field in a rectangular channel containing a cylindrical post near steady-state ($t = 11.572$s). The top row compares the x- component of the electric and velocity fields (electric field on left, velocity field on right); the bottom compares the y- components. Electrokinetic slip occurs at the top and bottom boundaries as well as on the surface of the post. A potential drop of 10 V in the x direction is instantaneously switched on at $t = 0$s. For reference, the channel is 1 cm wide, the post has a radius of 1 mm, the fluid is an aqueous solution with a kinematic viscosity of 0.01cm^2/s, and the electroosmotic mobility on all physical boundaries is 0.001.

2.2 Staggered Array of Posts

Our next proposed test geometry is a small staggered array of cylindrical posts. Due to the high resolution required to simulate this geometry and the severe time step size constraints imposed to ensure stability of the time-stepping algorithm at low Reynolds numbers, we have not yet been able to carry out this computation to any meaningful time. Figure 4 shows the electric field for this geometry. Because all of the fluid-solid boundaries have the same electroosmotic mobility, the steady-state velocity field is just scaled by the mobility.

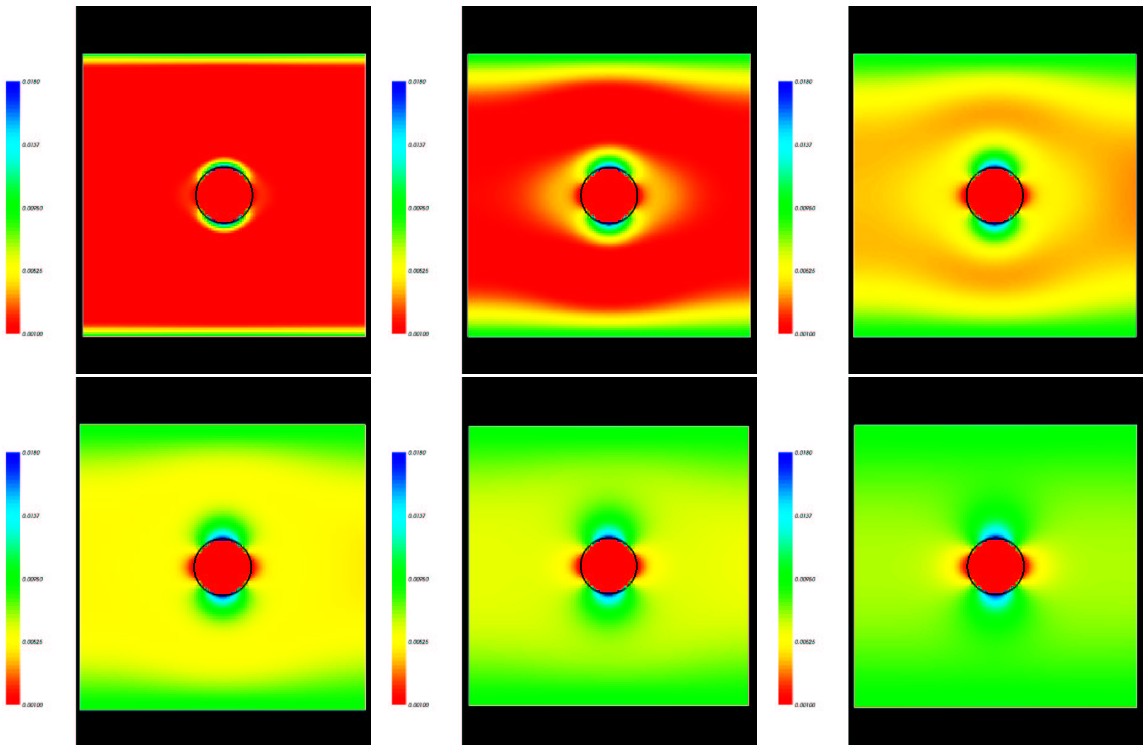

Figure 3: Two-dimensional fluid flow in a rectangular channel containing a cylindrical post at times $t = 0.0610$s, $t = 1.221$s, $t = 4.272$s, $t = 6.104$s, $t = 8.545$s, and $t = 11.572$s (ordered left to right, top to bottom). Electrokinetic slip occurs at the top and bottom boundaries as well as on the surface of the post. A potential drop of 10 V in the x direction is instantaneously switched on at $t = 0$s. For reference, the channel is 1 cm wide, the post has a radius of 1 mm, the fluid is an aqueous solution with a kinematic viscosity of 0.01cm^2/s, and the electroosmotic mobility on all physical boundaries is 0.001.

3 Future Improvements and Directions

While the current version of the electrokinetics simulation code is capable of handling simple electrokinetically driven flows, there are many areas that could be improved. What follows is a list of future improvements and directions that extend this summer's work.

- In light of recent interest in AC electrokinetic effects, it would be useful to add support for time-dependent electric fields. To support time-dependent electric fields, the primary algorithmic change would be to include the solution of the electrostatics problem within the main time-stepping loop.

- In the `EKModule`, the capabilities of the simulation code would be significantly improved by supporting the option of using different electroosmotic mobilities for different boundaries. This enhancement would allow the code to handle multi-material devices. Along these same lines, support for nonuniform zeta-potentials (possible via a nonuniform mobility) would make it possible to simulate induced-charged effects which become important if there are polarizable components of the device (*e.g.* electrodes).

- In order to accurately simulate the motion of fluid-particles systems (*e.g.* DNA flowing through a microfluidic device), we need to include electrokinetic effects of the particles themselves, especially for those particles that possess a significant diffuse charge layer (*e.g.* DNA). One possible approach to this problem would be to attribute a finite volume to each particle and use the particle positions at the

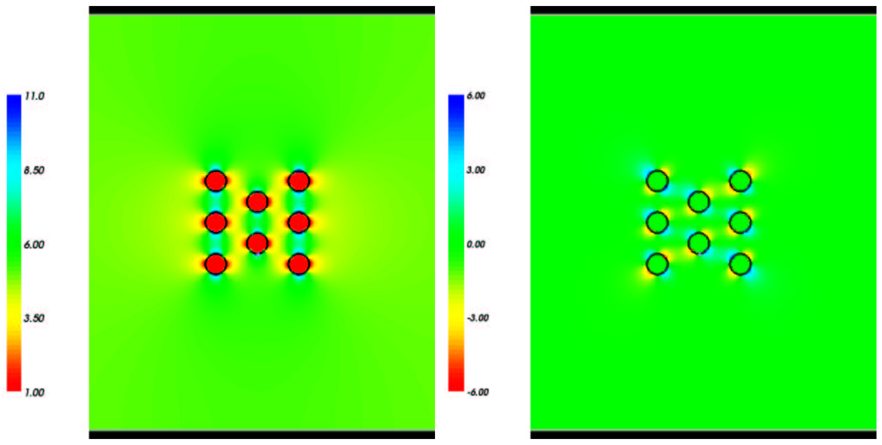

Figure 4: Electric field for a rectangular channel containing a a staggered array of cylindrical posts (E_x on left, E_y on right). Electrokinetic slip occurs at the top and bottom boundaries as well as on the surface of all posts. A potential drop of 1 V in the x direction is instantaneously switched on at $t = 0$s. For reference, the channel is 2 mm wide, the each post has a radius of 25 μm, the fluid is an aqueous solution with a kinematic viscosity of 0.01cm^2/s, and the electroosmotic mobility on all physical boundaries is 0.001.

beginning of each time step to generate a new geometry. This geometry could then be used to compute a new solution to the electrostatics problem which would give new slip-velocities for the fluid flow problem.

- Because electrokinetic effects are surface driven phenomena, high resolution of fluid-solid boundaries is important in order to obtain accurate solutions. However, it is unlikely that this high resolution will be required throughout the entire computational domain. To address these issues, it might be useful to leverage adaptive mesh refinement (AMR) technologies. Also, because electrokinetically driven flows are low Reynold's number flows, AMR might also prove helpful in dealing with the severe time step size constraints needed to maintain numerical stability.

- The low Reynold's number nature of electrokinetic flows complicates the choice of time step size by making it necessary (in some algorithms) to include considerations of the viscous term when analyzing numerical stability. At the moment, the time step size is determined by using a heuristic that includes both the convective and viscous time scales. However, this heuristic calculation seems to be incorrect because for problems with small physical dimensions ($\approx 100\mu$m), even the viscous time step size seems inadequate. It would, thus, be beneficial to further investigate the stability criteria for this problem and base the maximum time step size calculation on a more careful analysis.

- Further verification and validation of the code is required before we can be confident that the results produced by the simulation are physically accurate. To be more specific, a few possibilities include:
  - Grid convergence studies should be done to check that the numerical solution converges as the grid spacing decreases;
  - Numerical solutions in more complex geometries could be compared with analytical or experimental results;
  - In addition to comparison of steady-state flows, perhaps it would be possible to compare numerical results to experimentally observed transient flows.

One concrete problem that may be of scientific interest is simulating the 3D flow around a non-stationary, metallic cylinder in the presence of an AC electric field. Being able to simulate the flow field for this problem has the potential to be useful in understanding some experiments being conducted by Klint Rose *et. al.* at Stanford University.

4 Acknowledgements

This work was supported by the Department of Energy through the Computational Science Graduate Fellowship (CSGF) Program provided under grant number DE-FG02-97ER25308. The author would like to thank D. Trebotich for his assistance in understanding the `EBNavierStokes` code and helpful discussions on how to incorporate electrokinetic effects.

Appendix A: Derivation of Embedded Boundary Slip-Velocity Formula

To compute the slip-velocity at embedded boundaries, we add the following leading-order correction to each component of the cell-centered value of "slip-velocity":

$$\phi_{EB} - \phi_{CC} = -\nabla \phi_{EB} \bullet \delta \vec{x}_o \tag{7}$$

where $\delta \vec{x}_o = (\vec{x}_{CC} - \vec{x}_{EB})$. In this formula, $\nabla \phi_{EB}$ is computed from the values of ϕ at the centers of neighboring cells using weights determined by observing that the gradient of ϕ at $\vec{x}_{EB}$ can be estimated as the solution of a least squares fit problem [3]:

$$A \bullet \nabla \phi = \delta \phi, \tag{8}$$

where

$$\begin{aligned} A &= (\delta \vec{x}_1, ..., \delta \vec{x}_p)^T && (9) \\ \delta \phi &= (\delta \phi_1, ..., \delta \phi_p) && (10) \\ \delta \vec{x}_m &= \vec{x}_m - \vec{x}_{EB} && (11) \\ \delta \phi_m &= \phi_m - \phi_{EB} && (12) \end{aligned}$$

and p is the number of neighbor points used to estimate $\nabla \phi$. Letting $A^+ = \left(A^T A \right)^{-1} A^T$ denote the pseudoinverse of A, we see that we can write the least squares solution for $\nabla \phi_{EB}$ as

$$A^+ \delta \phi = A^+ (\phi_1, ..., \phi_p)^T - \phi_{EB} A^+ (1, ..., 1)^T \tag{13}$$

Substituting this formula into Eq. (5) and solving for ϕ_{EB}, we arrive at Eq. (5):

$$\phi_{EB} = \frac{\phi_{CC} - \delta \vec{x}_o \bullet \left[A^+ (\phi_1, ..., \phi_p)^T \right]}{1 - \delta \vec{x}_o \bullet \left[A^+ (1, ..., 1)^T \right]}. \tag{14}$$

References

[1] T. M. Squires and M. Z. Bazant, *Induced-charge electro-osmosis*, J. Fluid Mech., 509 (2004), pp. 217–252.

[2] M. Z. Bazant, K. Thornton, A. Ajdari, *Diffuse-Charge Dybamics in Electrochemical Systems*, preprint, http://arxiv.org/abs/cond-mat/0401118.

[3] D. Trebotich, *Working Notes for BCG Projection in EB Framework*, private communication.

[4] J. B. Bell, P. Colella, and H. M. Glaz *A Second-Order Projection Method for the Incompressible Navier-Stokes Equations*, J. Comp. Phys., 85 (1989), pp. 257-283.

[5] R. J. Hunter, *Foundations of Colloid Science*, Oxford University Press, Oxford, 2001.