The Projection Method

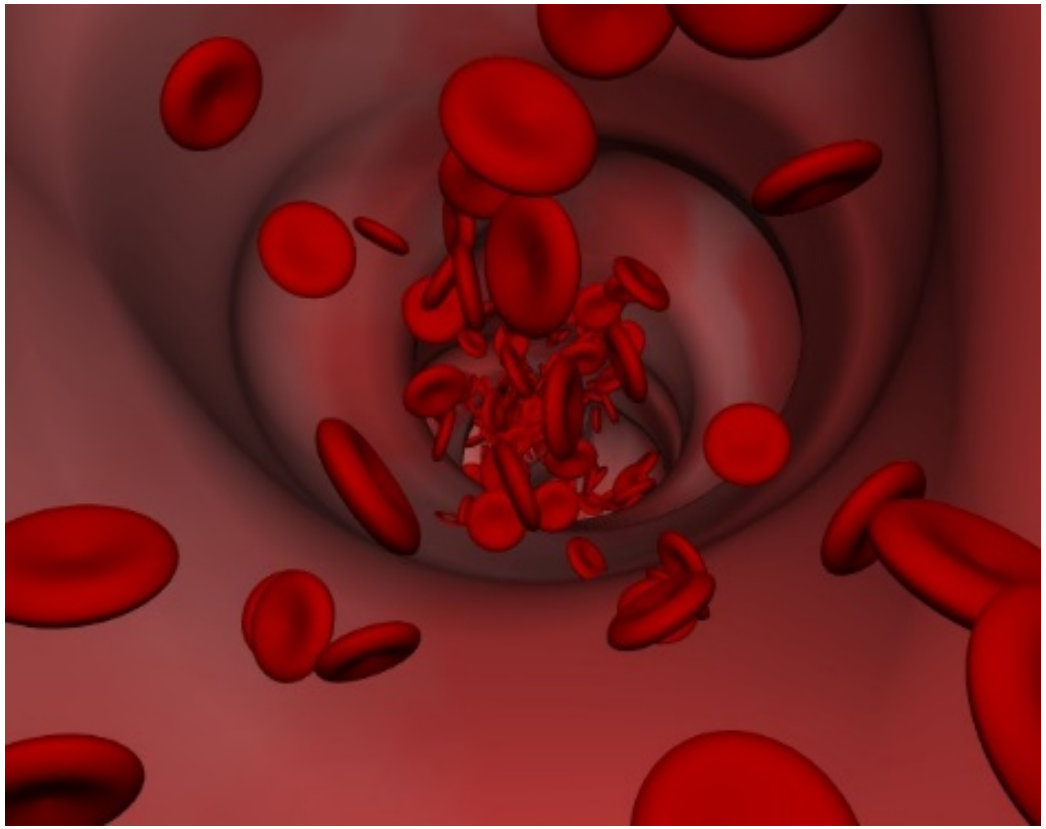
A High Performance Algorithm for Numerically Solving Stokes Flow

Contents

I. Motivation
II. Method Derivation
III. Results

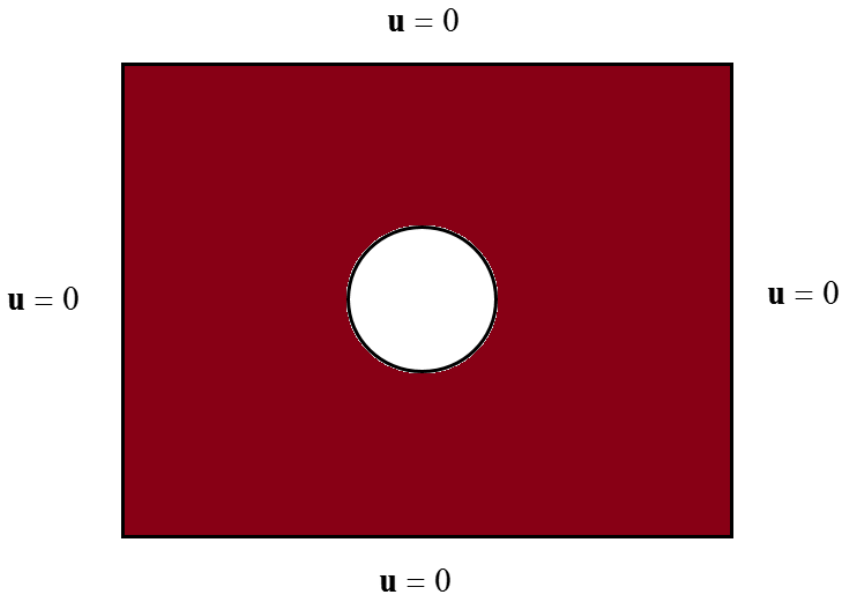
Fluid Dynamics of Red Blood Cells

- Organisms that can fill their cells with a sugar, increasing viscosity to stop cell metabolism (hibernation).
- Can this method be applied to red blood cells to replace cryopreservation?
- Biologists will greatly benefit from having a model to simulate the fluid dynamics.



Assumptions for Regime Selection

- The fluid is Newtonian and incompressible.
- Red blood cells are modeled as vesicles (cell wall only).
- Modeled in two dimensions.



Incompressible Navier Stokes Equation

- Non-dimensionalize the full Incompressible Navier Stokes Equation:

$$Re(\mathbf{u}_t + \nabla \mathbf{u} \cdot \mathbf{u}) = -\nabla p + \nabla \cdot (\mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T)) + \mathbf{f}$$

$$\nabla \cdot \mathbf{u} = 0$$

  - Re is the Reynolds Number.
  - ρ is the density, μ is the viscosity coefficient, $\mathbf{u}$ is the flow velocity, p is the pressure, and $\mathbf{f}$ is the body force.

Reynolds Number

- The Reynolds Number is the ratio of the inertial forces to the viscous forces:

$$Re = \frac{\tilde{\rho}\tilde{u}}{\frac{\tilde{\mu}}{\tilde{L}}}$$

  - $\tilde{u}$ is velocity of the blood flow
  - $\tilde{\mu}$ is the viscosity of the blood
  - $\tilde{L}$ is the radius of the blood cell
  - Thus, $\tilde{L} << 1 \Rightarrow \frac{\tilde{\rho}\tilde{u}}{\frac{\tilde{\mu}}{\tilde{L}}} = Re \to 0$

Stokes Equation

$$Re(\mathbf{u}_t + \nabla \mathbf{u} \cdot \mathbf{u}) = -\nabla p + \nabla \cdot (\mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T)) + \mathbf{f}$$

$$\nabla \cdot \mathbf{u} = 0$$

Becomes:

$$0 = -\nabla p + \nabla \cdot (\mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T)) + \mathbf{f}$$

$$\nabla \cdot \mathbf{u} = 0$$

Decoupling Method

- If μ is spatially constant, solve for p, u, and v by decoupling the variables into separate vectors.
- Take the divergence of both sides of the Stokes Equation.
- Rearrange the mixed partial derivatives and group the terms.
- Apply the divergence-free condition.

$$0 = \nabla \cdot (-\nabla p + \mu \Delta \mathbf{u} + \mathbf{f})$$

$$0 = \nabla \cdot \left(- \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \mu \begin{bmatrix} u_{xx} + u_{yy} \\ v_{xx} + v_{yy} \end{bmatrix} + \begin{bmatrix} f1 \\ f2 \end{bmatrix} \right)$$

$$0 = -(p_{xx} + p_{yy}) + \mu(u_{xxx} + u_{yyx} + v_{xxy} + v_{yyy}) + f1_x + f2_y$$

$$\Delta p = \nabla \cdot \mathbf{f}$$

Decoupling Method (Continued)

- The addition of this independent equation for p makes the other two equations for u and v fully determined, allowing them all to be solved independently using the following system:

1. $$p_{xx} + p_{yy} = f1_x + f2_y$$

2. $$\mu(u_{xx} + u_{yy}) = p_x - f1$$

3. $$\mu(v_{xx} + v_{yy}) = p_y - f2$$

Decoupling Method (Continued)

- Descretize p, u, and v each into grids of size $M \times M$.
- Stack the M^2 discretized points into a column vector for each variable.

$$\mathbf{x} = \begin{bmatrix} p_{11} \\ p_{21} \\ \dots \\ p_{M1} \\ p_{12} \\ \dots \\ p_{MM} \end{bmatrix}$$

p_{11}	p_{12}	p_{13}	p_{14}
p_{21}	p_{22}	p_{23}	p_{24}
p_{31}	p_{32}	p_{33}	p_{34}
p_{41}	p_{42}	p_{43}	p_{44}

Decoupling Method (Continued)

- Construct an $M^2 \times M^2$ matrix **A** and a right hand side vector **b** that consists of the normal second order finite difference approximations.
- Solve **Ax** = **b** once for each variable for three total solves, using the Matlab "\" operator.
- This algorithm is only possible if μ is spatially constant, otherwise the divergence operator will generate extra terms and the pressure will not be successfully decoupled.

Saddle-Point Method

- Traditional way of solving the system when μ is spatially variant.
- Discretize u, v, and p into a grid of size $M \times M$.
- Stack u, v, and p into a single column vector.

$$\mathbf{x} = \begin{bmatrix} p_{11} \\ \dots \\ p_{M1} \\ \dots \\ p_{MM} \\ u_{11} \\ \dots \\ u_{MM} \\ v_{11} \\ \dots \\ v_{MM} \end{bmatrix}$$

Saddle-Point Method (Continued)

- If μ is spatially constant, $\nabla \cdot (\mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T)) = \mu \Delta \mathbf{u}$, and the system becomes:

$$0 = -\nabla p + \mu \Delta \mathbf{u} + \mathbf{f}.$$

$$\nabla \cdot \mathbf{u} = 0$$

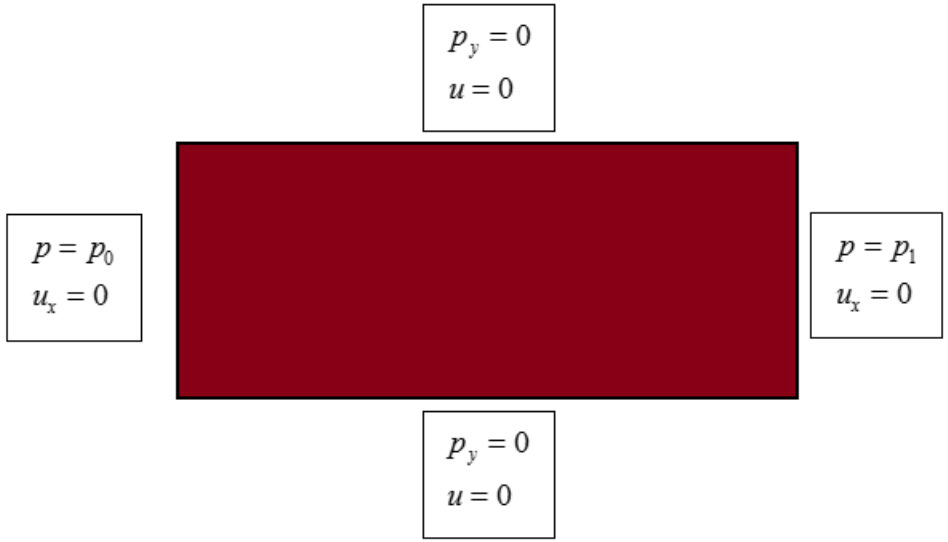
- Construct the system **Ax** = **b** and use the Matlab "\" operator to solve:

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

$$\begin{bmatrix} -\nabla & \mu\Delta \\ 0 & \nabla\cdot \end{bmatrix} \begin{bmatrix} p \\ \mathbf{u} \end{bmatrix} = \begin{bmatrix} -\mathbf{f} \\ 0 \end{bmatrix}$$

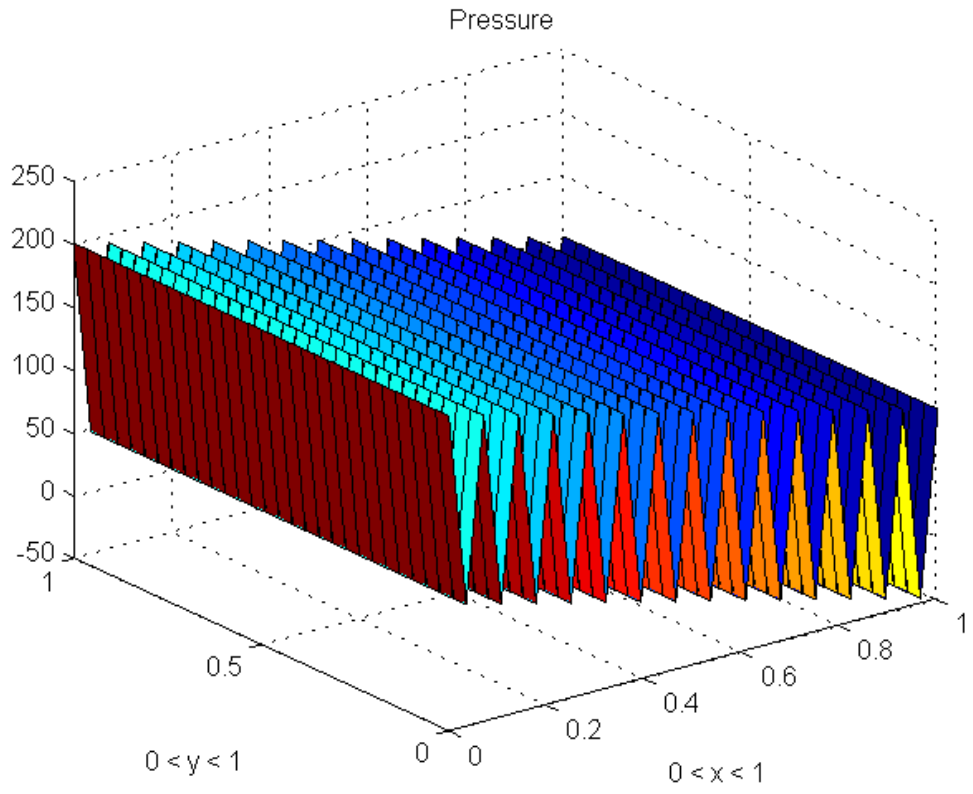
Fluid in a Pipe

- As a demonstration of concept, the Saddle-Point Method is applied to a fluid in a pipe problem.
- No-Slip top and bottom for u.
- Dirichlet conditions on the left and right to create a pressure gradient.
- Neumann for u on the left and right sides.
- Neumann for p on the top and bottom.



Fluid in a Pipe (continued)

- Incorrect solution: "Checkerboard" Pattern.
- Caused by the first derivative finite difference formula for p.
- Uses the two neighboring pressure cells, but not the actual pressure cell being described.
- Information ends up skipping every other cell.



Fluid in a Pipe (continued)

- To address this issue, create a staggered grid.
- In the staggered grid, The p nodes are a distance of $\frac{\Delta x}{2}$ and $\frac{\Delta y}{2}$ away from the u and v nodes.

p_{11}	u_{11}	p_{12}	u_{12}	p_{13}	u_{13}	p_{14}	u_{14}	p_{15}
v_{11}		v_{12}		v_{13}		v_{14}		v_{15}
p_{21}	u_{21}	p_{22}	u_{22}	p_{23}	u_{23}	p_{24}	u_{24}	p_{25}
v_{21}		v_{22}		v_{23}		v_{14}		v_{45}
p_{31}	u_{31}	p_{32}	u_{32}	p_{33}	u_{33}	p_{34}	u_{34}	p_{35}
v_{31}		v_{32}		v_{33}		v_{34}		v_{35}
p_{41}	u_{41}	p_{42}	u_{42}	p_{43}	u_{43}	p_{44}	u_{44}	p_{45}
v_{41}		v_{42}		v_{43}		v_{44}		v_{45}
p_{51}	u_{51}	p_{52}	u_{52}	p_{53}	u_{53}	p_{54}	u_{54}	p_{55}

Fluid in a Pipe (continued)

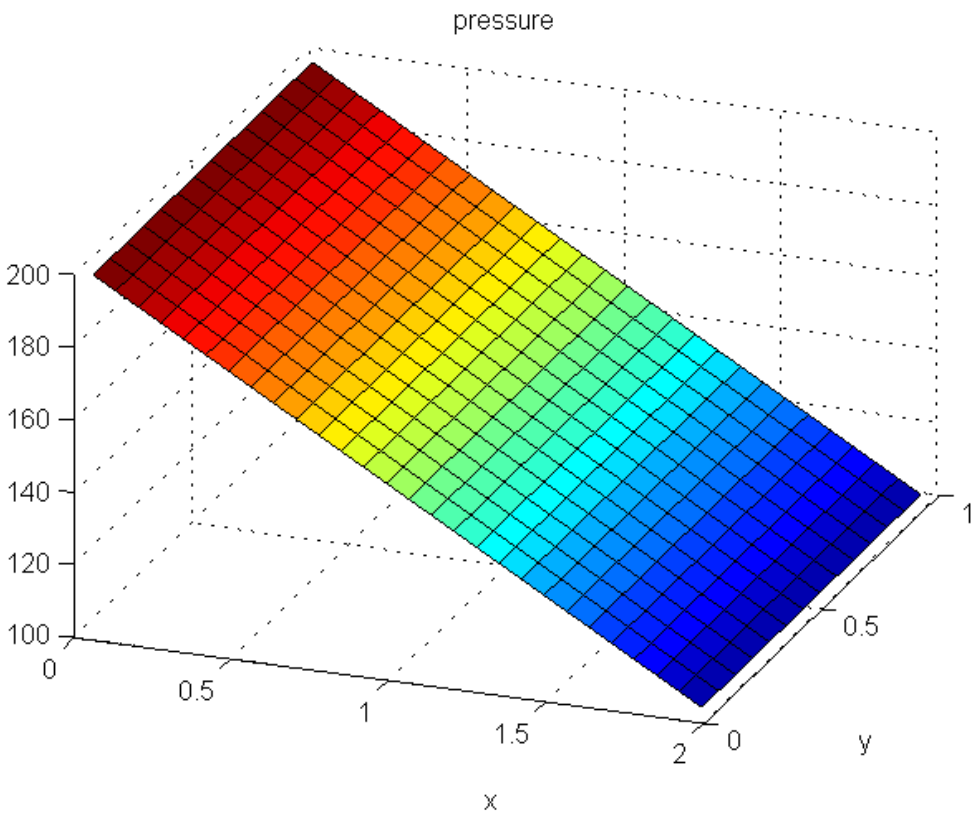
- **A** consists of the normal second order finite difference approximations, with the exception of ∇p .
- Creating a Taylor Expansion about the u and v nodes yields a modified second order difference formula for p_x and p_y :

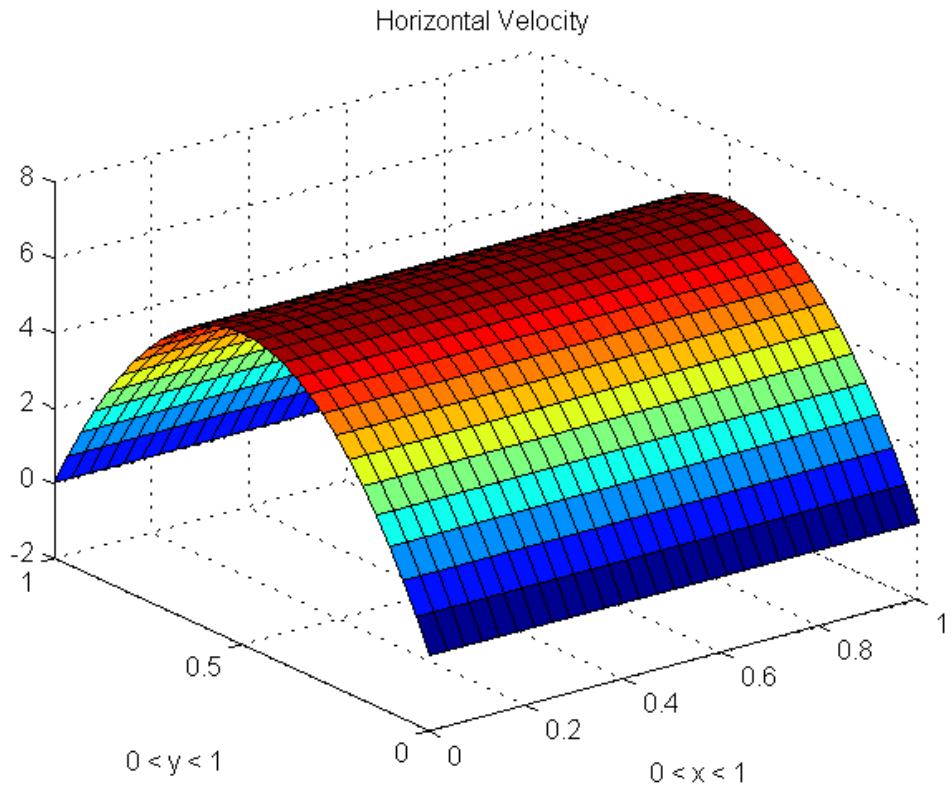
$$p_x = \frac{p_{i+1,j} - p_{i,j}}{\Delta x}$$

$$p_y = \frac{p_{i,j+1} - p_{i,j}}{\Delta y}$$

Fluid in a Pipe (continued)

- Pressure and horizontal velocity solution with the staggered grid:





Time Analysis

- As a worst-case scenario, assume the "\" operator will use Gaussian Elimination to solve **Ax = b**.
- The number of multiplications and additions to convert **A** to Reduced Row Echelon form will be a sum of squares.
- This will require $O(n^3)$ Floating Point Operations (FLOPS), where n is the number of rows in **A.**

Time Analysis (Continued)

- Decoupling Method
  - For each solve, **A** has M^2 rows, because **x** is a column vector containing each value for u, v, or p on the $M \times M$ discretized grid.
  - The number of FLOPS is $3 \cdot O(n^3) = 3 \cdot O(M^6)$, where $n = M^2$.
- Saddle-Point Method
  - **A** has $3M^2$ rows because **x** is a stacked vector containing u, v, and p for each value on the $M \times M$ discretized grid.
  - Since n is three times as large as it is in the Decoupling Method, this will require $O(n^3) = O(27M^6)$ FLOPS.

The Need for a New Algorithm

- The Saddle-Point Method is slow, scaling very badly as M increases.
- The Decoupling Method can only be used when μ is spatially constant.
- Construct a new method that decouples u, v, and p yet can still solve a system with spatially varying viscosity.

Contents

I. Motivation
II. Method Derivation
III. Results

Projection Method

- Helmholtz-Hodge Decomposition Theorem:

> A vector field $\mathbf{\Psi}$ defined on a simply connected domain can be uniquely decomposed into a divergence-free component, $\mathbf{\Gamma}$, and a curl-free component, $\nabla\Phi$:

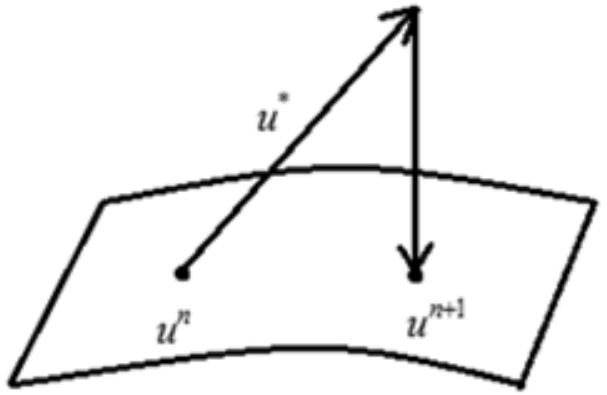
$$\mathbf{\Psi} = \mathbf{\Gamma} + \nabla\Phi$$

- Align the Navier Stokes Equation:

$$\nabla \cdot (\mu(\nabla\mathbf{u} + \nabla\mathbf{u}^T)) - Re(\nabla\mathbf{u} \cdot \mathbf{u}) + \mathbf{f} = Re(\mathbf{u}_t) + \nabla p$$

Projection Method (Continued)

- General Strategy:
  - Advance $\mathbf{u}^n$ forward in time using an iterative approximation.
  - The errors in this approximation will take $\mathbf{u}^{n+1}$ off of the divergence-free solution space where it belongs.
  - Assign this solution to a temporary vector $\mathbf{u}^*$.
  - Project $\mathbf{u}^*$ back onto the divergence-free solution space to find the correct value of $\mathbf{u}^{n+1}$.



Surface of Divergence-Free Vector Fields

$$Re\mathbf{u}_t = RHS$$

$$Re\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} \approx RHS^n$$

$$\mathbf{u}^{n+1} \approx \mathbf{u}^n + \frac{\Delta t}{Re}RHS^n$$

$$\mathbf{u}^* = \mathbf{u}^n + \frac{\Delta t}{Re}RHS^n$$

Projection Method (Continued)

- Define the projection operator, P(**a**) as the projection of vector **a** onto the divergence-free solution space:

$$P(\mathbf{a}) = \mathbf{a} - \frac{< \mathbf{a}, \nabla p >}{< \nabla p, \nabla p >} \nabla p.$$

- It can be shown that the Dirichlet boundary conditions $\mathbf{u} \cdot \hat{n} = 0$ are sufficient to cause the following properties to hold:

$$P(\mathbf{u}_t) = \mathbf{u}_t$$
$$P(\nabla p) = 0$$

Projection Method (Continued)

- Apply the P operator to both sides of the Navier Stokes Equation.
- Use the properties of the P operator to eliminate ∇p .
- For reasons that will become clear later, Add and subtract $\frac{1}{\Delta t}\mathbf{u}_t$ on the inside and the outside of the P operator.
- These steps result in:

$$Re(\mathbf{u}_t) = P\left(-Re(\nabla\mathbf{u}\cdot\mathbf{u}) + \nabla\cdot(\mu(\nabla\mathbf{u}+\nabla\mathbf{u}^T)) + \mathbf{f} + \frac{1}{\Delta t}\mathbf{u}\right) - \frac{1}{\Delta t}\mathbf{u}.$$

Projection Method (Continued)

- Integrate both sides with respect to t.
- Apply a left hand rectangular approximation to the first integral, and a right hand approximation to the second integral to create a relationship between $\mathbf{u}^n$ and $\mathbf{u}^{n+1}$:

$$Re(\mathbf{u}^n - \mathbf{u}^{n+1}) = \int_{t_n}^{t_{n+1}} P\left(-Re(\nabla \mathbf{u} \cdot \mathbf{u}) + \nabla \cdot (\mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T)) + \mathbf{f} + \frac{1}{\Delta t}\mathbf{u}\right) - \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} \mathbf{u}$$

$$Re(\mathbf{u}^n - \mathbf{u}^{n+1}) = P\left(-Re(\nabla \mathbf{u}^n \cdot \mathbf{u}) + \nabla \cdot (\mu(\nabla \mathbf{u}^n + \nabla (\mathbf{u}^n)^T)) + \mathbf{f}^n + \frac{1}{\Delta t}\mathbf{u}^n\right)\Delta t - \frac{1}{\Delta t}\mathbf{u}^{n+1}\Delta t.$$

Projection Method (Continued)

- Apply the small Reynolds Number, $Re \to 0$, and solve for $\mathbf{u}^{n+1}$:

$$\mathbf{u}^{n+1} = P(\nabla \cdot (\mu(\nabla \mathbf{u}^n + \nabla(\mathbf{u}^n)^T))\Delta t + \Delta t \mathbf{f}^n + \mathbf{u}^n)$$

- It is now apparent how to define $\mathbf{u}^*$:

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t(\nabla \cdot (\mu(\nabla \mathbf{u}^n + \nabla(\mathbf{u}^n)^T)) + \mathbf{f}^n)$$

- This simplifies the equation to:

$$\mathbf{u}^{n+1} = P(\mathbf{u}^*)$$

- Expanding the projection operator:

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{< \mathbf{u}^*, \nabla p^n >}{< \nabla p^n, \nabla p^n >} \nabla p^n$$

Projection Method (Continued)

- It can be shown that: $\dfrac{< \mathbf{u}^*, \nabla p^n >}{< \nabla p^n, \nabla p^n >} = \Delta t.$

- Thus, $\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t \nabla p^n$.
- To find an equation for p, rearrange the above equation and take the divergence of both sides:

$$\nabla p^n = \frac{\mathbf{u}^* - \mathbf{u}^{n+1}}{\Delta t}$$

$$\Delta p^n = \frac{1}{\Delta t} \nabla \cdot (\mathbf{u}^* - \mathbf{u}^{n+1})$$

$$\Delta p^n = \frac{1}{\Delta t} \nabla \cdot \mathbf{u}^*$$

Projection Method (Continued)

- The algorithm is complete:

1.

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t(\nabla \cdot (\mu(\nabla \mathbf{u}^n + \nabla(\mathbf{u}^n)^T)) + \mathbf{f}^n)$$

2.

$$\Delta p^n = \frac{1}{\Delta t} \nabla \cdot \mathbf{u}^*$$

3.

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t \nabla p^n$$

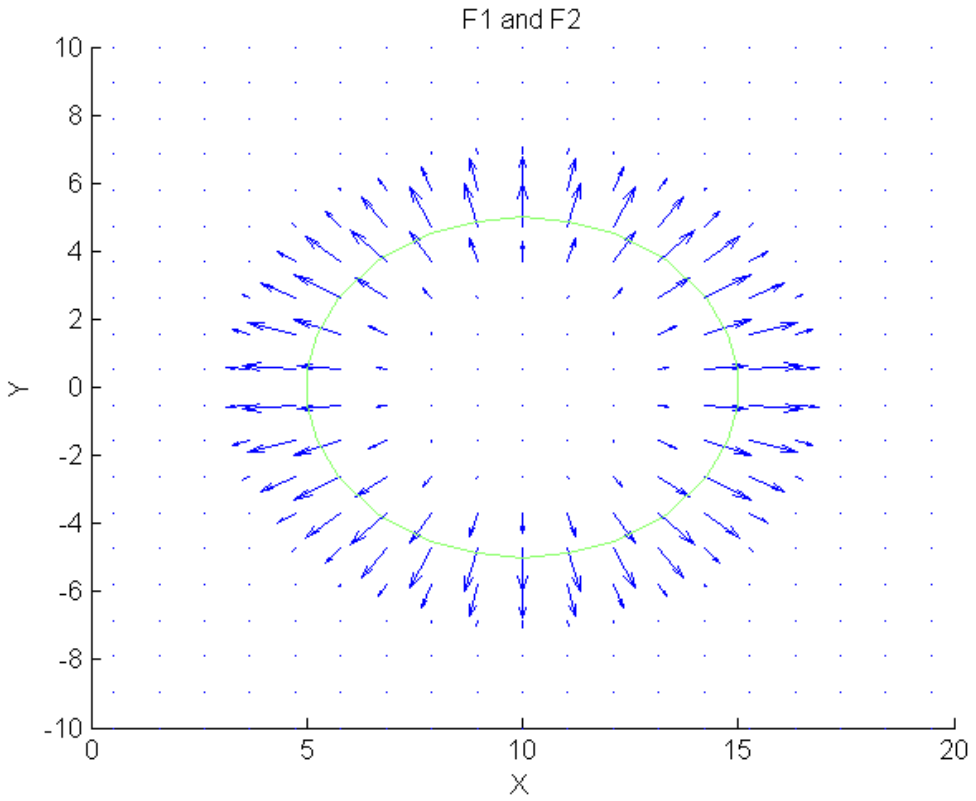
Projection Method (Continued)

- This system is decoupled and can still be solved with a spatially varying viscosity.
- The Saddle-Point Method can solve with spatially varying viscosity, but it is slow.
- The Decoupling Method is faster, but it cannot solve with spatially varying viscosity.
- The Projection Method gets the best of both worlds: It is decoupled and fast, and can solve a system with spatially varying viscosity.
- If the viscosity is spatially constant, the Projection Method can still be used, and step 1 becomes:

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t(\mu \Delta \mathbf{u}^n + \mathbf{f}^n)$$

The Vesicle Force Problem

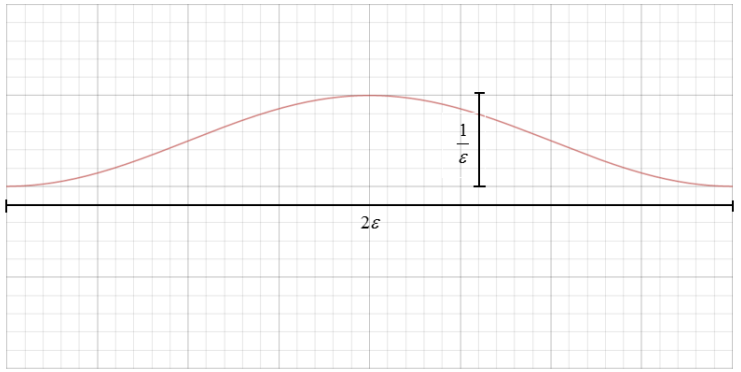
- This model will be solved by all three algorithms to verify convergence and measure execution times.
- Model the reactionary force of a vesicle membrane in non-moving incompressible fluid
- The cell wall resists bending and compression from the fluid by applying an outward force



The Vesicle Force Problem (continued)

- Define **z** as the distance from the membrane.
- Implement a smoothed Dirac Delta function at the membrane:

$$\delta(z) = \begin{cases} \frac{1+cos(\frac{\pi z}{\epsilon})}{2\epsilon} & \text{if } -\epsilon \leq z \leq \epsilon \\ 0 & \text{otherwise} \end{cases}$$



The Vesicle Force Problem (continued)

- Define the force as follows:

$$\mathbf{f} = \frac{1}{R}\delta(z)\hat{\mathbf{n}}$$

- $\frac{1}{R}$ is the curvature.
- The more compressed the cell wall is, the harder it will push back against the fluid.
- $\hat{\mathbf{n}}$ is the outward pointing normal vector.

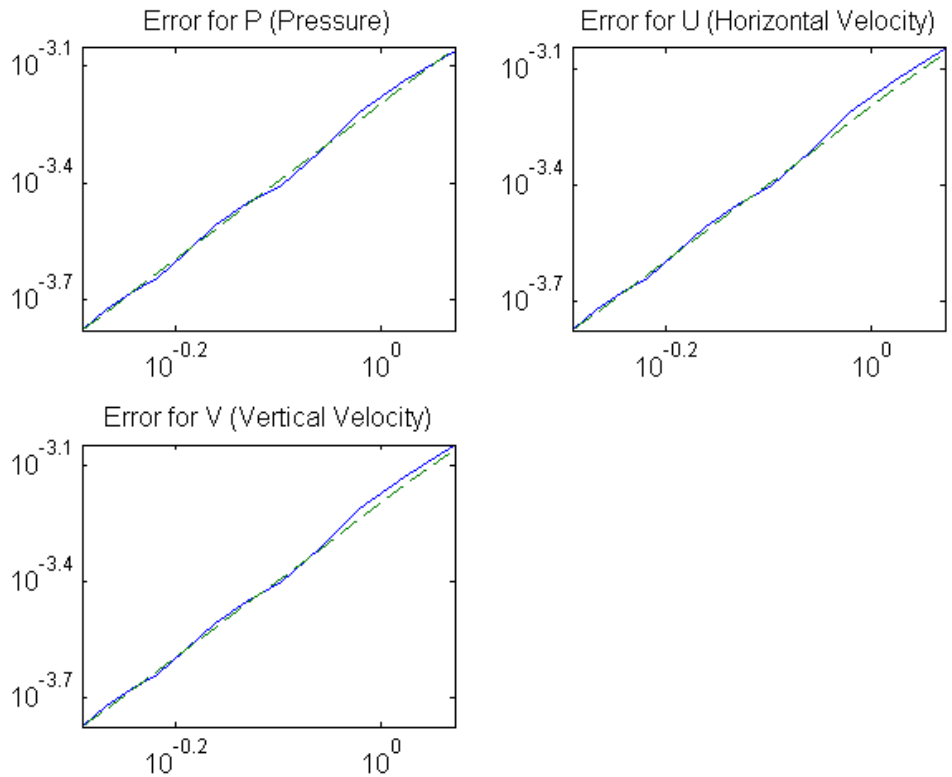
The Vesicle Force Problem (continued)

- The force is radially symmetric
- The analytic solution can be obtained by converting to polar
- Make the ansatz that $\mathbf{u} = 0$ and solve $\mu\mathbf{u} = 0 = \nabla p - f$ in polar.
- The result:

$$p = \begin{cases} \frac{-1}{2R}\left(1 - \frac{z}{\epsilon} - \frac{1}{\pi} sin\left(\pi\frac{z}{\epsilon}\right)\right) & \text{if } -\epsilon \leq z \leq \epsilon \\ -\frac{1}{R} & \text{if } z < -\epsilon \\ 0 & \text{if } z > \epsilon \end{cases}$$

Convergence Testing

- Projection Method solution vs. the analytic solution to the vesicle force problem, one time step.
- L2 Norm of the Error graphed against varying values of Δx, log-log scale.
- Reference line of slope 2
- Note: The graph is the same across indefinite time steps.

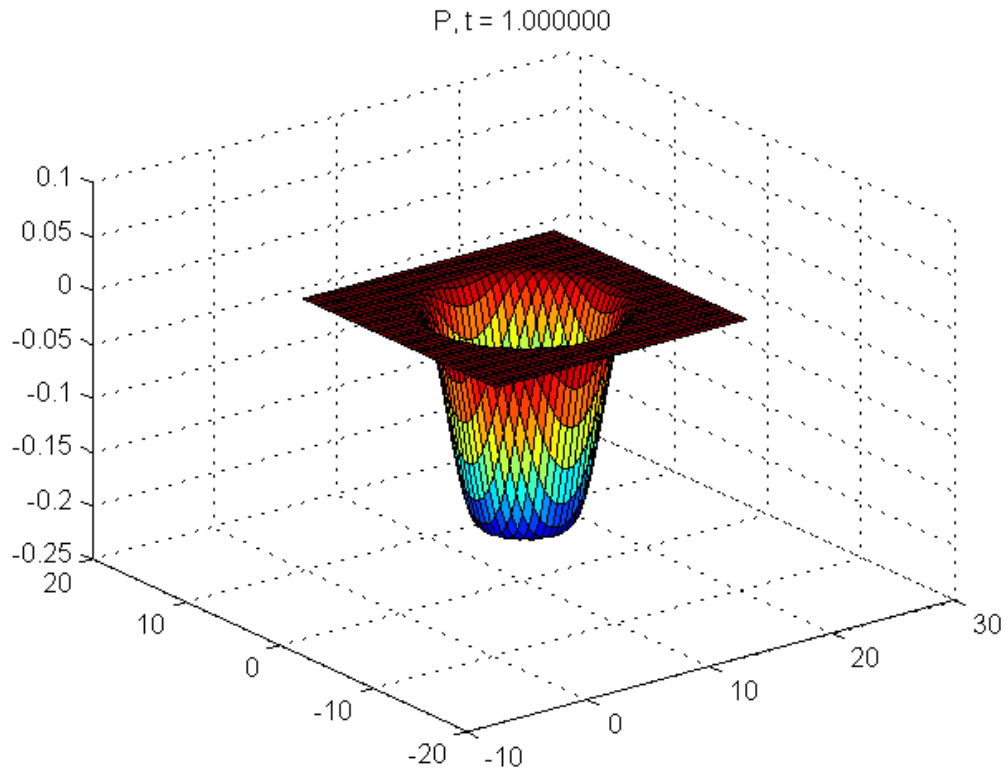


Contents

I. Motivation
II. Method Derivation
III. Results

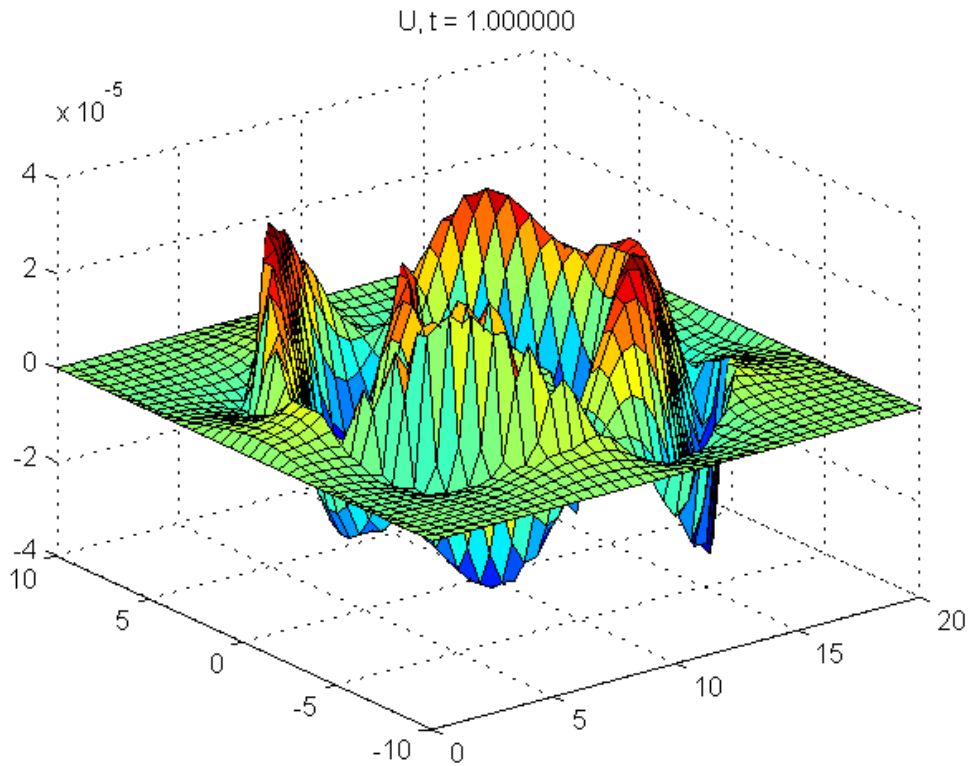
Surface Plots

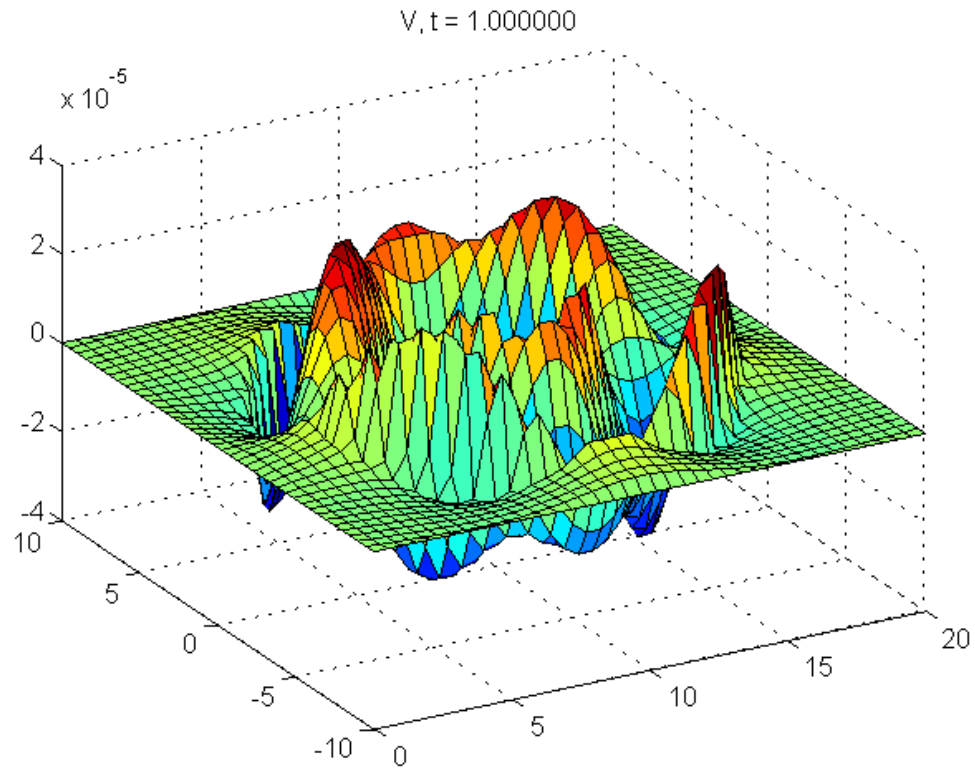
- The following plots are from the Projection Method solution:



Surface Plots (continued)

- The following plots are from the Projection Method solution:





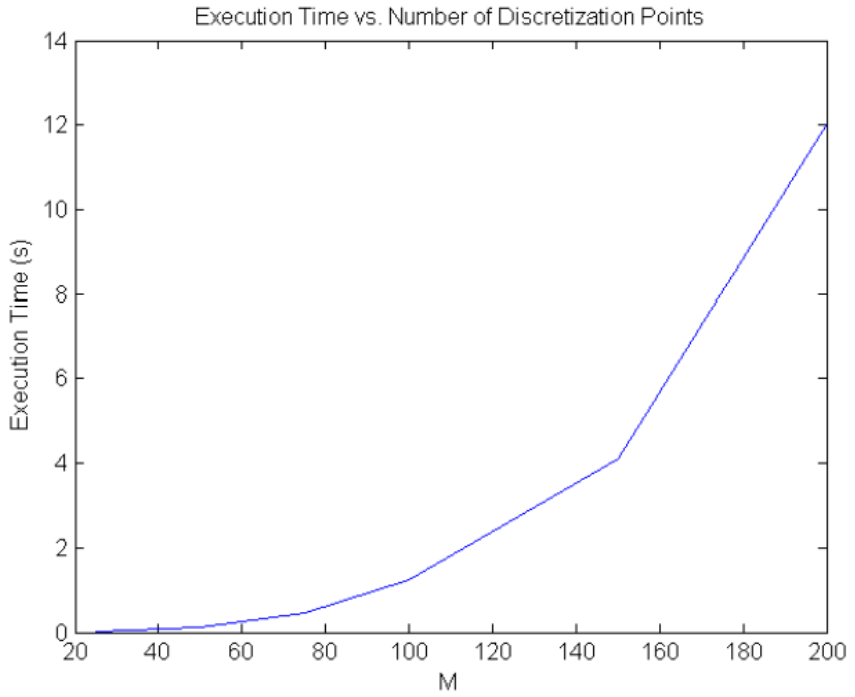
Time Analysis of the Projection Method

- Step 1: $O(n)$ assignments for each of the $n = M^2$ discretized points.
- Step 2: matrix solve for the M^2 values of p.
- The size of **A** is $M^2 \times M^2$. Thus, step 2 requires $O(n^3)$ flops, where the size of n is $n = M^2$.
- Step 3: Same as step 1.
- Neglecting the linear assignments, the total complexity is $O(M^6)$.
- This should result in an increase in performance over the Decoupling Method, which is $3 \cdot O(M^6)$, and a substantial increase over the Saddle-Point Method, which is $O(27M^6)$.

Execution Times (Decoupling Method)

Discretization Points (M)	Execution Time (s)
25	0.0184
50	0.1301
75	0.4659
100	1.2452
150	4.1129
200	12.0575

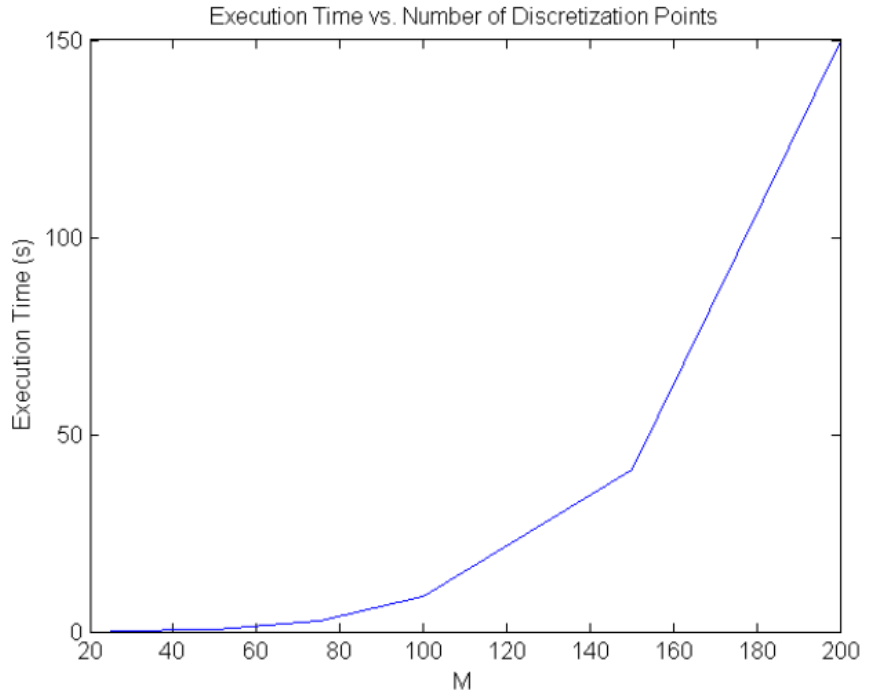
Table 1: Run Times for the Decoupling Method



Execution Times (Saddle-Point Method)

Discretization Points (M)	Execution Time (s)
25	0.0519
50	0.5195
75	2.6204
100	9.0459
150	40.9595
200	149.6981

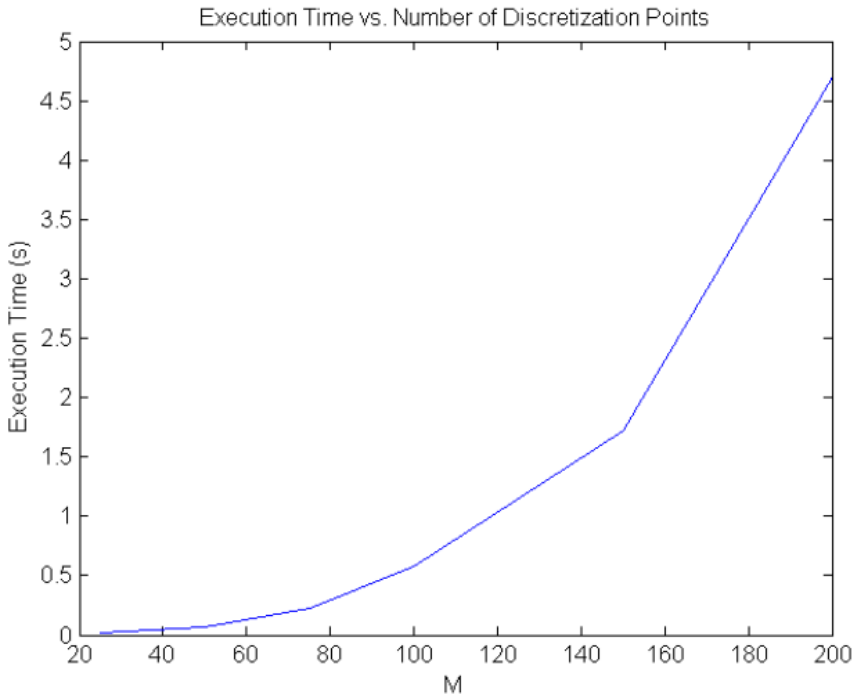
Table 2: Run Times for the Saddle-Point Method



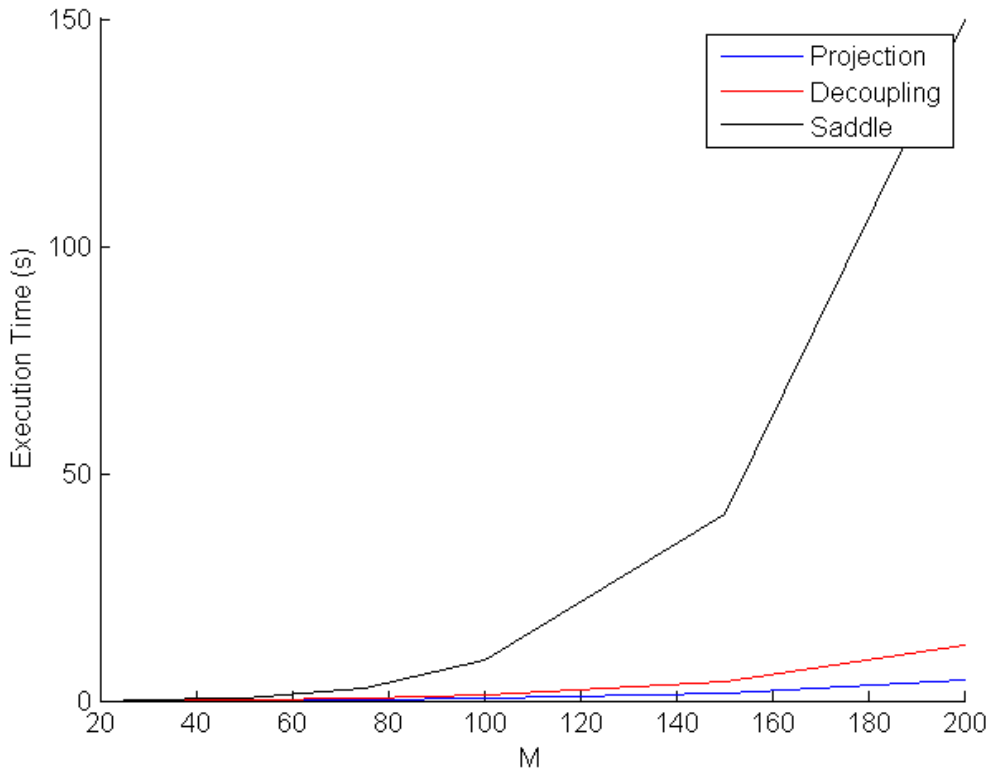
Execution Times (Projection Method)

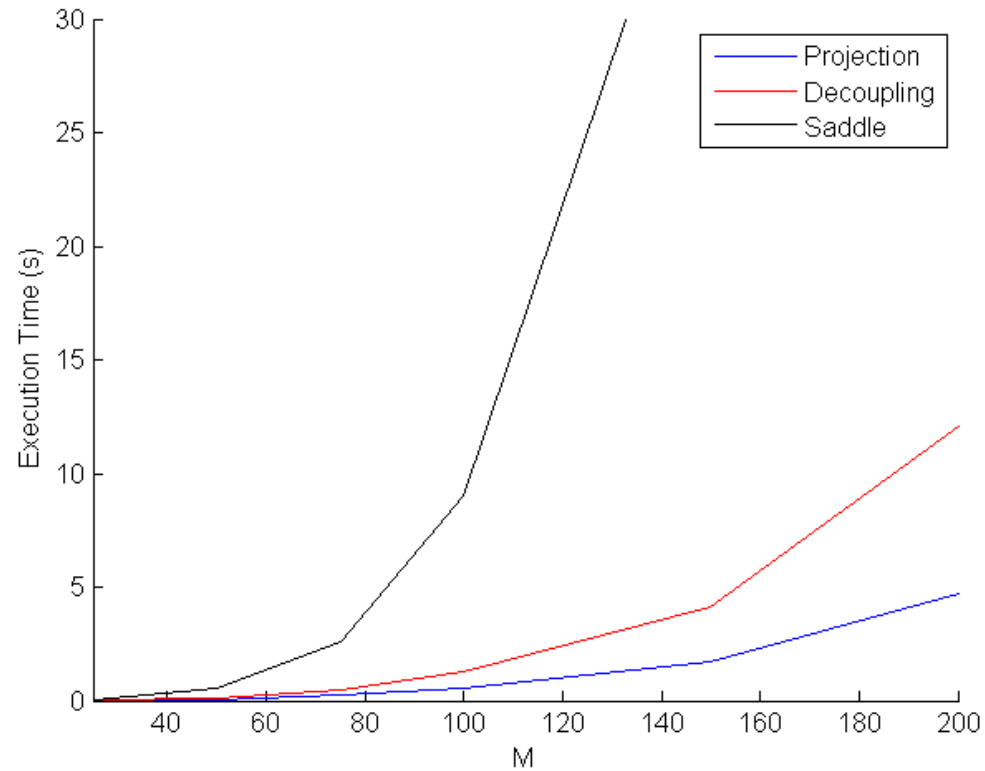
Discretization Points (M)	Execution Time (s)
25	0.0124
50	0.0715
75	0.2266
100	0.5742
150	1.7194
200	4.7079

Table 3: Run Times for the Projection Method

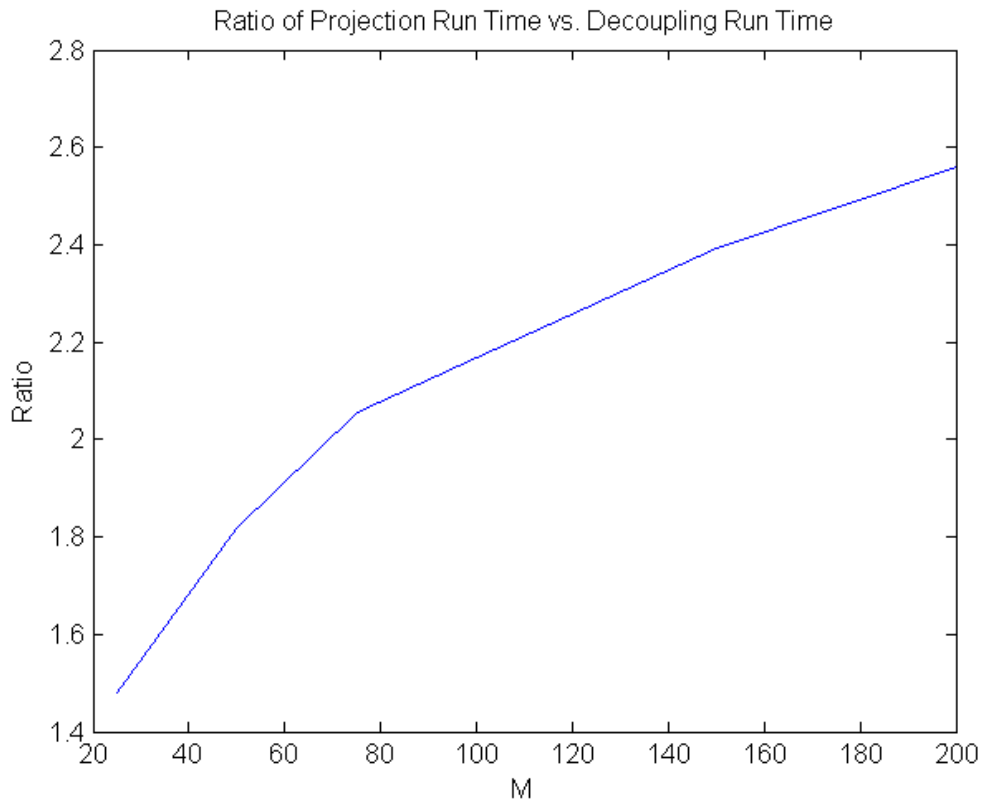


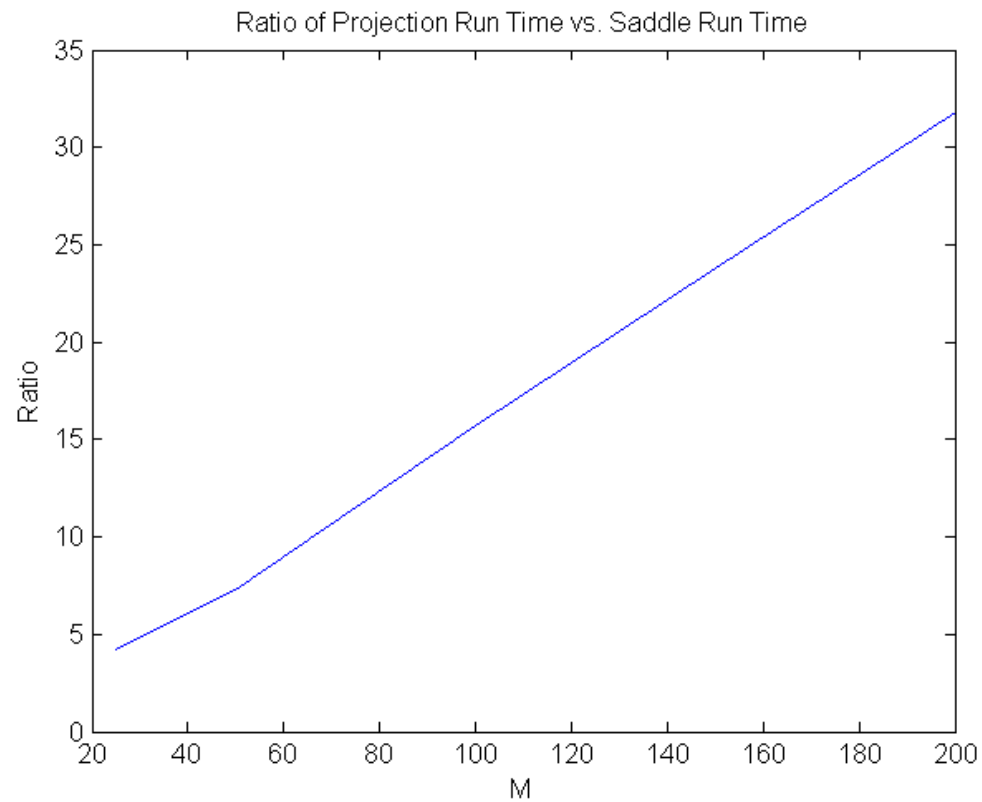
Excecution Times (All)





Ratios of Execution Times





Conclusions

- The Projection Method is much faster than the Saddle-Point Method.
- It is also faster than the Decoupling Method.
- The larger M is, the more of an advantage it gains.
- It is difficult to tell exactly how much faster
  - Overhead from other operations in the code
  - Matlab "\" operator is best case $O(n)$ and worst case $O(n^3)$
- Projection Method gains the best of both worlds: Decoupled and fast, but can solve with spatially varying viscosity.
- Modeling of red blood cells will need to simulate the changing viscosity.

Future Steps

- Add an initial fluid flow to the vesicle force problem
- Add a time dependent force
- Add a spatially varying viscosity

Bitbucket Repositories

- Saddle-Point Method Code:
- https://rhermle@bitbucket.org/rhermle/saddle-point-vesicle.git
- Decoupling Method Code:
- https://rhermle@bitbucket.org/rhermle/decouplingmethod.git
- Projection Method Code:
- https://rhermle@bitbucket.org/rhermle/2d-stokes-predictor-corrector.git

Acknowledgements

- Professor Vogl
- Awesome professors and staff at UW

Questions?