Introduction to Numeric Simulation of Magnetic Fluid Flows

Ataias Pereira Reis, Yuri Dumaresq Sobral, Francisco Ricardo da Cunha
Universidade de Brasília
Departamento de Matemática
Brasília, Brasil
ataiasreis@gmail.com

***Abstract*—This work aims at studying how magnetic fluids behave in a 2D lid-driven cavity. In order to achieve this goal, a computer program was developed to solve the Navier Stokes equation using finite differences. In addition to that, models of the magnetostatic force were introduced. Solving the Poisson equation is done by means of an implicit method with sparse matrices and Cholesky factorization. Once the hydrodynamic and magnetostatic parts are solved, vector fields and streamlines are plotted for analysis of the flow. A few distinct Reynolds numbers are tested, as well as some different magnetic parameters. The governing equations in continuous form and also in discrete form are presented, as well as the vectors fields resulted from computations.**

***Index Terms*—Magnetic fluid, Driven Cavity, Finite differences, Magnetohydrodynamics**

I. Introduction

This project aims at studying magnetic fluids in a 2D lid-driven cavity. The ideal cavity used here is a square of side equal to unity. All the boundaries but the top are with null velocities. The lid has a stationary velocity pattern that will be presented. Even though this may seem a simplistic method, papers have been presented not only in the last century but also in this century to study real problems. Garandet [2] uses a similar approach to study solute segregations. On the field of biomagnetic fluids, Tzirtzilakis [6] has also considered a cavity and analysed the stationary solutions under different conditions of the Reynolds numbers and magnetic parameters. Differently from this last researcher, our project has not considered a method to find directly the steady state, but has evolved in discrete time steps from time 0 to an end time defined. The advantage is that more data can be analysed, so that it is possible to know when vortices start to appear and if there is any that disappear. Animations were made in order to investigate this.

A colloidal magnetic fluid, or ferrofluid, consists typically of a suspension of monodomain ferromagnetic particles such as magnetite in a nonmagnetic carrier fluid. Particle-to-particle agglomeration is avoided by surfactants covering the particles, and Brownian motion prevents particle sedimentation in gravitational or magnetic fields. [5]. A magnetic fluid has usually around 10^{23} particles per cubic meter and is opaque at visible light.

The method chosen to solve the differential equations is finite differences [4]. This method approximates each derivative by discrete terms involving differences. The domain in which the problem is to be solved must be turned into a mesh. The smaller the cells of the mesh, the better the finite differences solution will be. All the finite differences used here are of second order, which means that the error decays quadratically as the mesh size is lowered. The mesh that is being used is not the trivial one, but a staggered grid. In such a grid, not every information is stored in the same point. For cell ij, the pressure will be in position $i+\frac{1}{2}j+\frac{1}{2}$, while velocity in the x direction will be $u_{ij+\frac{1}{2}}$ and the y direction will be $v_{i+\frac{1}{2}j}$. This avoids spurious pressure modes [3]. A cell of this grid is presented in Figure 1. An example of the mesh is presented in Figure 2. The simulation will have to use all the round points presented, and almost all the velocities marked, except the extreme left and bottom ones. After the solution is obtained, an interpolation can be made to obtain the values at a specified point. It was decided to save the results in the middle point of each cell. The pressure is already in that position, but the velocities are not. Therefore, averages were computed to obtain those values in the middle point.

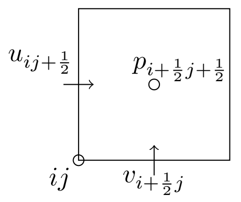

Fig. 1. Cell of the staggered grid

The code for the simulator was made with aid of programming language Julia[1]. This language offers several linear algebra packages that were easy to use. Graphics were made using Python's graphical packages. Codes and articles are hosted on GitHub[2]. In the Julia language, matrices are indexed from 1 to n. This is different from C, because its indexing starts from 0.

[1]http://julialang.org
[2]https://github.com/ataias/ferrofluidos

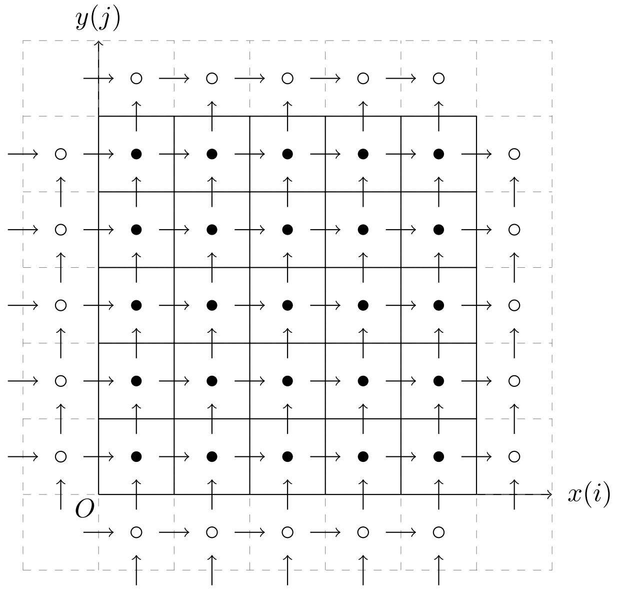

Fig. 2. Staggered-grid. Dark circles are internal points, while empty ones are part of the extra layer

II. Governing Equations

A. Hidrodynamics

Equations 1 and 2 govern the flow of an incompressible fluid. The first one is the Navier Stokes equation, while the second one is the condition for incompressibility.

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t}+\mathbf{v}\cdot\nabla\mathbf{v}\right)=-\nabla p+\mu\nabla^2\mathbf{v}+\mathbf{f} \tag{1}$$

$$\nabla\cdot\mathbf{v}=0 \tag{2}$$

Computer solutions require certain conditions on the time step and spacing of grid points. The requirements are: stable diffusion, Equation 3; stable advection, Equation 4 and hydrodynamic boundary layer, Equation 5. U is the maximum speed in the cavity. If those equations are not satisfied, the solution will be wrong as the program may be trying to solve faster than the laws of physics allow. The program may not even converge and stay on an infinite loop.

$$\Delta t < \frac{1}{4}Re\Delta x^2 \tag{3}$$

$$\Delta t < \frac{\Delta x}{U} \tag{4}$$

$$\Delta x < \frac{1}{Re} \tag{5}$$

The Navier Stokes equation is thorough, encompassing several types of flow. A specific solution is found by specifying its boundary conditions. For the current work, the conditions are

$$u(x,1) = \sin^2(\pi x), \tag{6}$$

$$u(x,0) = u(0,y)=u(1,y)=0 \tag{7}$$

$$v(x,0) = v(x,1)=v(0,y)=v(1,y)=0 \tag{8}$$

B. Magnetism

Besides the hydrodynamics equations and conditions, the magnetic force **- f -** must be computed. From Maxwell's theory, Equations 9 and 10 are obtained.

$$\mathbf{B} = \mu_0(\mathbf{M}+\mathbf{H}) \tag{9}$$

$$\nabla\cdot\mathbf{B} = 0 \tag{10}$$

From the latter equations, it is possible to conclude that

$$\nabla\cdot\mathbf{M}=-\nabla\cdot\mathbf{H}. \tag{11}$$

The magnetostatic regime implies $\nabla\times\mathbf{H}=\mathbf{0}$ which indicates there is a potential field ϕ that satisfies $\mathbf{H}=-\nabla\phi$. With these last equations, it is possible to obtain Equation 12. This is a Poisson equation that will give ϕ according the the applied field $\mathbf{H}$. The superparamagnetic case is being considered, what implies $\mathbf{M}=\chi\mathbf{H}$.

$$\nabla^2\phi = \nabla\cdot\mathbf{M} \tag{12}$$

Nevertheless, the potential ϕ cannot be introduced directly in the Navier Stokes equation. What is needed is a relation that gives the magnetic force, which could then be incorporated in the fluid equations. Equation 13 stands for the magnetic force, while Equation 14 is the magnetic reynolds number.

$$\mathbf{f} = \mathrm{Cpm}\mathbf{M}\cdot\nabla\mathbf{H} \tag{13}$$

$$\mathrm{Cpm} = \frac{\mu_0 H_0^2}{\rho u^2} \tag{14}$$

One thing that is missing is the boundary conditions for the potential field ϕ. To understand how they are derived, first consider an applied magnetic field **H**. A condition that is imposed is continuity of the normal component of **B - B** can be obtained by Equation 9. This continuity can be mathematically represented by Equation 15. From this, Equation 16 can be obtained.

$$(\mathbf{B}_{\mathrm{out}}-\mathbf{B}_{\mathrm{in}})\cdot\mathbf{n} = 0 \tag{15}$$

$$\mu_0(H^{\mathrm{n}}_{\mathrm{out}}+M^{\mathrm{n}}_{\mathrm{out}}) = \mu_0(H^{\mathrm{n}}_{\mathrm{in}}+M^{\mathrm{n}}_{\mathrm{in}}) \tag{16}$$

The medium outside is air and therefore its magnetization, $M^{\mathrm{n}}_{\mathrm{out}}$ is zero. It is possible to use $\mathbf{H}=-\nabla\phi$ in the remaining outer term to obtain Equation 17.

$$\nabla\phi^{\mathrm{n}}_{\mathrm{d}} = -H^{\mathrm{n}}_{\mathrm{f}}+M^{\mathrm{n}}_{\mathrm{d}} \tag{17}$$

From that, Neumann boundary conditions can be obtained. For the sake of clarity, they are presented in Equations 18 to 21. The subscripts are indicating matrix indices.

$$\left.\frac{\partial \phi}{\partial x}\right|_{\text{left}} \approx -H^x_{2,j} + M^x_{2,j} \tag{18}$$
$$\left.\frac{\partial \phi}{\partial x}\right|_{\text{right}} \approx -H^x_{n,j} + M^x_{n,j} \tag{19}$$
$$\left.\frac{\partial \phi}{\partial y}\right|_{\text{lower}} \approx -H^y_{i,2} + M^y_{i,2} \tag{20}$$
$$\left.\frac{\partial \phi}{\partial y}\right|_{\text{upper}} \approx -H^y_{i,n} + M^y_{i,n} \tag{21}$$

III. Solution Method

In order to solve the flow with a computer, the Navier Stokes equation must be expanded in equations for directions x and y and then discretized. Nevertheless, a naive application of the discretized formulas will not yield a proper solution, as the fluid flow equation is highly non-linear due to its convective term - $\mathbf{v}\cdot\nabla\mathbf{v}$. Another problem is that the pressure needed at each time step is also an unknown that must be solved: the pressure is needed to know the velocity and the velocity is needed to obtain the pressure. To solve this dilemma, the Chorin method [1] was used. This method is referred to as time splitting. In order to come to grips with this method, one can start by using finite differences for the velocity derivative with respect to time and then doing a smart manipulation with two new terms. This is presented in Equation 22.

$$\frac{\partial \mathbf{v}}{\partial t} \approx \frac{\mathbf{v}^{n+1}-\mathbf{v}^n}{\Delta t} = \frac{\mathbf{v}^{n+1}-\mathbf{v}^*}{\Delta t} + \frac{\mathbf{v}^*-\mathbf{v}^n}{\Delta t} \tag{22}$$

The idea of time-splitting is to impose each parcel of Equation 22 to a different part of the Navier Stokes equation. The last term in the right hand side is used to solve Equation 23, in which pressure is not considered, while the term just before it is related to the pressure field. Equation 24 shows the relation between $\mathbf{v}^{n+1}$ and $\mathbf{v}^*$.

$$\frac{\mathbf{v}^*-\mathbf{v}^n}{\Delta t} = \nu\nabla^2\mathbf{v} + \frac{\mathbf{f}}{\rho} - \mathbf{v}\cdot\nabla\mathbf{v} \tag{23}$$
$$\frac{\mathbf{v}^{n+1}-\mathbf{v}^*}{\Delta t} = -\frac{1}{\rho}\nabla p \tag{24}$$

Even though there is a straightforward formula to obtain $\mathbf{v}^*$, how to use it to obtain p? It is easy to notice that one can apply the divergence operator to both sides of Equation 24. As $\mathbf{v}^{n+1}$ obeys the incompressibility condition, it is zero, with the remaining equation being feasible of solution by a Poisson routine.

$$-\frac{1}{\rho}\nabla\cdot\nabla p = \nabla\cdot\left(\frac{\mathbf{v}^{n+1}-\mathbf{v}^*}{\Delta t}\right)$$
$$\nabla^2 p = \frac{\rho}{\Delta t}\nabla\cdot\mathbf{v}^*$$

Now that Equation 25 is available, it is easier to devise a solution to the problem, as only $\mathbf{v}^*$ is needed and it is already a known value because of Equation 23. However, nothing was stated about the boundary conditions this problem involves. For that, it is better to expand the Navier Stokes equation, as in Equations 25 and 26.

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial^2 x} + \frac{\partial^2 u}{\partial^2 y}\right) + f_x \tag{25}$$
$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \nu\left(\frac{\partial^2 v}{\partial^2 x} + \frac{\partial^2 v}{\partial^2 y}\right) + f_y \tag{26}$$

Given the two latter equations, boundary conditions on the walls can be applied. Many terms are zero on the wall, and the result is presented in Equations 27 and 28. Notice that those are Neumann boundary conditions.

$$\left.\frac{\partial p}{\partial x}\right|_{\text{wall}} = \rho\nu\left.\frac{\partial^2 u}{\partial x^2}\right|_{\text{wall}} + \rho f_x \tag{27}$$
$$\left.\frac{\partial p}{\partial y}\right|_{\text{wall}} = \rho\nu\left.\frac{\partial^2 v}{\partial y^2}\right|_{\text{wall}} + \rho f_y \tag{28}$$

As the four wall have flux conditions, there are many solutions that differ only by a constant. A point in the mesh can be specified arbitrarily so that the program converges to a specific solution. This is allowed in our problem, as pressure itself is not directly needed, but it is gradient.

There is one last step to obtain $\mathbf{v}^{n+1}$, but this is quite straightforward: Equation 29 must be computed. It is an explicit equation with known terms at this stage.

$$\mathbf{v}^{n+1} = \mathbf{v}^* - \frac{\Delta t}{\rho}\nabla p \tag{29}$$

The three time-splitting steps were presented, but not their discretizations. The discretization for step 1 is on Equations 30 to 33. For each inner point of the mesh, the equations there presented are applied in an explicit form.

$$\begin{aligned}\mathbf{v}^s_{ij} &= (u^n_{i+1j} + u^n_{i-1j} + u^n_{ij+1} + u^n_{ij-1})\mathbf{i}\\ &+ (v^n_{i+1j} + v^n_{i-1j} + v^n_{ij+1} + v^n_{ij-1})\mathbf{j}\end{aligned} \tag{30}$$
$$\begin{aligned}\mathbf{v}^t_{ij} &= 0.25(u^n_{ij} + u^n_{i+1j} + u^n_{i+1j-1} + u^n_{ij-1})\mathbf{i}\\ &+ 0.25(v^n_{ij} + v^n_{i-1j} + v^n_{i-1j+1} + v^n_{ij+1})\mathbf{j}\end{aligned} \tag{31}$$

$$\begin{aligned}u^*_{ij} &= u^n_{ij} + \frac{\Delta t}{\rho}\left[\mu\left(\frac{u^s_{ij}-4u^n_{ij}}{\Delta x^2}\right) + Fx^n_{ij}\right] +\\ &- \left[u^n_{ij}(u^n_{i+1j}-u^n_{i-1j}) + \dots\right.\\ &+ \left. v^t_{ij}(u^n_{ij+1}-u^n_{ij-1})\right]\frac{\Delta t}{2\Delta x}\end{aligned} \tag{32}$$

$$\begin{aligned}v^*_{ij} &= v_{ij} + \frac{\Delta t}{\rho}\left[\mu\left(\frac{v^s_{ij}-4v^n_{ij}}{\Delta x^2}\right) + Fy^n_{ij}\right]\\ &- \left[u^t_{ij}(v^n_{i+1j}-v^n_{i-1j}) + \dots\right.\\ &+ \left. v^n_{ij}(v^n_{ij+1}-v^n_{ij-1})\right]\frac{\Delta t}{2\Delta x}\end{aligned} \tag{33}$$

The second step, solving for the pressure, can have an explicit or implicit solution method. Both methods were used, and the implicit one is much faster, so it is presented here. The hardest part of this method is that one can not apply the discretized equations directly, but it is necessary to form a linear system from it. For the implicit method, matrix A and vector b are needed to solve system $Ax = b$. The pressure elements are in the unknown x. This system gets very large, but most of the elements in matrix A are 0. This allows use of sparse matrices. Firstly, let's present the steps to obtain A and b.

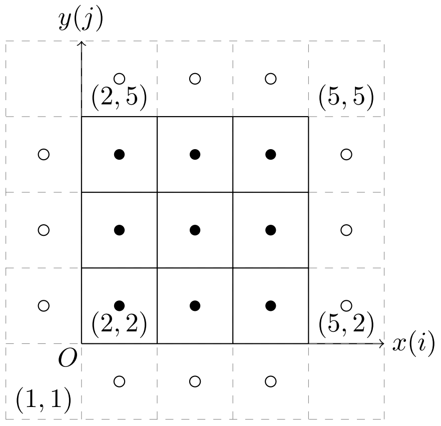

Fig. 3. Example grid to solve for pressure

Consider a 5×5 grid as in Figure 3. The equation to be solved is Equation 34, a generic poisson equation.

$$\nabla^2 p = \frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = g(x,y) \tag{34}$$

Using second order central differences, Equation 35 is obtained. This equation is valid for internal points (black points in Figura 3).

$$\frac{p_{i-1j} - 2p_{ij} + p_{i+1j}}{\Delta x^2} + \frac{p_{ij-1} - 2p_{ij} + p_{ij+1}}{\Delta x^2} = g_{ij} \tag{35}$$

If term p_{ij} is isolated in Equation 35, an explicit equation is obtained to all the internal points. The external points will be obtained from boundary conditions. Nevertheless, what is needed is an implicit representation of the system. For that, all the equations need to be considered and then organized in matrix form. The first equations for the internal points are presented in Equations 36 to 38. Equations for the other internal points (more 6 equations) are easily obtainable.

$$\begin{aligned}
-4p_{22} + p_{32} + \boldsymbol{p_{12}} + p_{23} + \boldsymbol{p_{21}} &= \Delta x^2 g_{22} \quad (36)\\
-4p_{23} + p_{33} + \boldsymbol{p_{13}} + p_{24} + p_{22} &= \Delta x^2 g_{23} \quad (37)\\
-4p_{24} + p_{34} + \boldsymbol{p_{14}} + \boldsymbol{p_{25}} + p_{23} &= \Delta x^2 g_{24} \quad (38)
\end{aligned}$$

The bold pressure points are boundary points. This means there exists an equation that can account for them. There are twelve external points and from the boundary conditions it is easy to get the twelve equations. Equations 39 to 42 can have its indices substituted in order to get 12 equations.

$$\begin{aligned}
\frac{\partial p}{\partial x}(0,y) &= p_x(0,y) \approx (p_{2j} - p_{1j})/\Delta x \quad (39)\\
\frac{\partial p}{\partial x}(1,y) &= p_x(1,y) \approx (u_{5j} - u_{4j})/\Delta x \quad (40)\\
\frac{\partial p}{\partial y}(x,0) &= p_y(x,0) \approx (p_{i2} - p_{i1})/\Delta x \quad (41)\\
\frac{\partial p}{\partial y}(x,0) &= u_y(x,0) \approx (p_{i5} - p_{i4})/\Delta x \quad (42)
\end{aligned}$$

From the 21 equations obtained, a rearrangement can be done to obtain matrix A. Equation 43 has this matrix. The b vector is in Equation 44. Nevertheless, one of its Eigenvalues is zero, so the matrix is not invertible. This problem has to do with the fact that all boundary conditions are Neumann conditions and many solution are possible, only differing by a constant. To solve this problem, one value of the mesh must be imposed and this is arbitrary with the exception of the corner points not influencing anything, as they don't take any part on the equations.

$$\begin{bmatrix}
-2 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
1 & -3 & 1 & 0 & 1 & 0 & 0 & 0 & 0\\
0 & 1 & -2 & 0 & 0 & 1 & 0 & 0 & 0\\
1 & 0 & 0 & -3 & 1 & 0 & 1 & 0 & 0\\
0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0\\
0 & 0 & 1 & 0 & 1 & -3 & 0 & 0 & 1\\
0 & 0 & 0 & 1 & 0 & 0 & -2 & 1 & 0\\
0 & 0 & 0 & 0 & 1 & 0 & 1 & -3 & 1\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -2
\end{bmatrix}
\begin{bmatrix} p_{22}\\ p_{23}\\ p_{24}\\ p_{32}\\ p_{33}\\ p_{34}\\ p_{42}\\ p_{43}\\ u_{44} \end{bmatrix} = b \tag{43}$$

$$\begin{aligned}
b &= \Delta x \begin{bmatrix}
p_x\left(0,\frac{\Delta x}{2}\right) + p_y\left(\frac{\Delta x}{2},0\right)\\
u_x\left(0,\frac{3\Delta x}{2}\right)\\
p_x\left(0,\frac{5\Delta x}{2}\right) - p_y\left(\frac{\Delta x}{2},1\right)\\
p_y\left(\frac{3\Delta x}{2},0\right)\\
0\\
-p_y\left(\frac{3\Delta x}{2},1\right)\\
-p_x\left(1,\frac{\Delta x}{2}\right) + p_y\left(\frac{5\Delta x}{2},0\right)\\
-u_x\left(1,\frac{3\Delta x}{2}\right)\\
-p_x\left(1,\frac{5\Delta x}{2}\right) - p_y\left(\frac{5\Delta x}{2},1\right)
\end{bmatrix}\\
&+ \Delta x^2 \left[g_{22}\; g_{23}\; g_{24}\; g_{32}\; g_{33}\; g_{34}\; g_{42}\; g_{43}\; g_{44}\right]^T \quad (44)
\end{aligned}$$

For our problem, $p_{12} = 0$ was chosen. This updates one element of the matrix and one element of the vector. The updated values are shown in Equations 45 and 46.

$$\begin{aligned}
A[1,1] &= -3 \quad (45)\\
b[1] &= p_y\left(\frac{\Delta x}{2},0\right)\Delta x + \Delta x^2 g_{22} \quad (46)
\end{aligned}$$

The equations just presented can be coded to define matrix A and vector b, but to actually obtain the vector x

of pressure points the system must be solved. A sparse matrix solver available in Julialang[3] is used. It uses a Cholesky factorization in order to solve the system. As the matrix is fixed, the factorization can be done once and for every other time-step the fatorized matrix is used. This is a performance-wise decision.

For the magnetism parts, it was only necessary to use finite differences on the governing equations. The Poisson implicit elements obtained in this section are also used for the magnetic part. The rest are explicit applications of the finite differences equations obtained.

IV. Results

V. Conclusion

Acknowledgments

References

[1] Alexandre Joel Chorin. A numerical method for solving incompressible viscous flow problems. *Journal of Computational Physics*, 135(2):118 – 125, 1997.

[2] J.P. Garandet, N. Kaupp, D. Pelletier, and Y. Delannoy. Solute segregation in a lid driven cavity: Effect of the flow on the boundary layer thickness and solute segregation. *Journal of Crystal Growth*, 340(1):149 – 155, 2012.

[3] E.J. Hinch. Lecture notes on computational methods in fluid dynamics: Part i - a first problem., 2006.

[4] L.M. Milne-Thomson. The calculus of finite differences. London: Macmillan & Co., Ltd. XIX, 558 S., 23 Fig. (1933)., 1933.

[5] R E Rosensweig. Magnetic fluids. *Annual Review of Fluid Mechanics*, 19(1):437–461, 1987.

[6] E.E. Tzirtzilakis and M.A. Xenos. Biomagnetic fluid flow in a driven cavity. *Meccanica*, 48(1):187–200, 2013.

[3] http://julialang.org