A New Method for Computing the Magnetic Configurations of Toroidal Fusion Devices

Ron Chris Wu

New York University, Courant Institute of Mathematical Sciences

251 Mercer Street
New York, NY 10012
January/2015

A thesis submitted in partial fulfillment
of the requirements for the degree of
master's of science
Department of Mathematics
New York University
January/2015

Antoine Cerfon
Faculty advisor's name

Mike O'Neil
Faculty thesis reader's name

Abstract

We introduce a novel substitute to the only-known conventional field line tracing method in simulation of magnetic field lines in 2D cross section in toroidal fusion devices. Using Lax-Friedrichs Hamiltonian, combined with high order derivatives, our method has 3rd order accuracy. Although our method is not as accurate as field line tracing, currently as we are applying the method to 3D, in contract to the conventional method that depicts individual magnetic field lines, our method gives entire 3D confinement magnetic surfaces, which is more fruitful in studying plasma confinement and magnetohydrodynamic equilibrium. Our method can also be potentially used as a numerical guide for testing whether magnetic field forms closed magnetic surfaces. This method can also be used for similar calculations of vortex lines in fluid dynamics and astrophysics.

Acknowledgments

My appreciation for research comes from working with my thesis adviser, and personal mentor of mine, Professor Antoine Cerfon. I am tremendously grateful for the work we have done together and all that he has taught me. I remember it was last March, I contact professor Cerfon, who I believed is one of the best numerical algorithm professors at NYU, he suggested me to work on a project that had great interest to magnetic confinement studies for nuclear fusion experiments.

The project was to find an algorithm that would produce computer images of magnetic confinement surfaces for given magnetic vector fields on the 3D grids. The project sounded very simple–just to connect the dots at the grid points along the vectors–which was essentially the idea behind the conventional field line tracing method. However experiments showed that conventional methods ran into troubles in some cases, so we thought we might find some better algorithm.

Deceived by traditional teaching, for some time I believed all phenomenology would eventually reduce to sets of simple, logically consistent mathematical equations, so I thought research would be just like that–brainstorming the ultimate equations. But once I started the research, I found nothing like that. The actual research was cluttered with messy details. At the beginning we were searching through thousands of variations to find the best algorithm and along the way new "theories" were invented and ruled out every week. Even more troublesome was that we had to constantly make up some guidelines to fit new findings.

Speaking of frustration and exhaustion, I would have given up long ago if it were not professor Cerfon that always supported me with encouragement and incisive advices. Mercifully it almost happened every time when I got so depressed and seemed that the search was going nowhere, then magically some slightly better result appeared. Since then I believe that only researchers and colleagues in the field understand the joy and pain that research entails. One cannot look only at the the number of pages in a report to asses how much work the authors have done. For instance, something as seemingly minimal as a paragraph can be the result of months of work. I remember thinking about

solutions to problems continuously day in and day out.

Because of the extraordinarily laborious and endless nature of researching, only someone who is genuinely enthusiastic as well as physically and mentally fit can engage in this type of work. I consider myself to be one of these people. Luckily after months of work, two weeks before thesis deadline, we found a good method that worked competitively well to the conventional method (at least in 2D and simple 3D magnetic surface with axial symmetry examples) in terms of both accuracy and speed. However our work is far from end. We are currently trying the method to 3D nonaxisymmetric fields using parallel computing technique to boost speed.

One thing I am fascinated about numerical analysis is that unlike other physical experiments, for us every parameter is under control. One benefit is that computers will always return exact answers if the input codes are unchanged (for randomized algorithms fixed the seed), so only one trial is convincing. And due to that convenience, we can try hundreds of new variations in a day. In this report I can only list a handful of choices we had tried to give readers a sense how we got here. Other tens of thousands of unsuccessful trials and details have been left out of the report, to be part of my experience, and will be valuable to me for my future quest in other research.

Another thing about experiments that cannot be overstated is honesty. Unlike working in theories, people are entitled to make wild guesses. For experimentalists, we should never deliberately twist data or adjust testing schemes to make data look good. No matter how close the deadline is, or how crafty the twists are. Specially in the arena of computational experiments, since the original codes used to generate the results are often not published with the paper. Many raised issues were discussed in [1] and in the opening remark of [2].

There is some historical relevance between our method and our NYU math department– the Courant institute of mathematical science. Arguably numerical analysis was born in 1928 after the paper written by Courant, Friedrichs, and Lewy. They provided the first proof [3] that under what conditions (aka CFL conditions) a partial differential equation could be converted to a finite difference equation, thus could be solved by machines that handled only additions and multiplications like computers. Courant and his doctoral student, Friedrichs were the co-founders of the NYU math department.

Following their ideas, it became natural to numerically solve analytic differential equations on grids or meshes, a plethora of new methods were invented to refine finite difference methods [6, 7]. Among them the best of the best is Lax-Friedrichs method [8], invented by Friedrichs and Lax, the doctoral student of Friedrichs. Standing on the shoulders of giants, Cerfon and his master student, we extended Lax-Friedrichs method from conservative Hamiltonian to not-so-exact-conservative Hamiltonian on convex and non convex domains for the special problems in plasma confinement and MHD equilibrium.

I am indebted to so many greatest mathematicians and physicists. I learned linear

algebra from Prof. Esteban Tabak; I learned real, complex and functional analysis from Prof. Sinan Gunturk, Prof. Eliezer Hameiri, Prof. Fengbo Hang and Prof. Nader Masmoudi; I learned differential equations from Prof. Pierre Germain and Prof. Fang-Hua Lin; I learned mathematical physics from Prof. Percy Deift and Prof. Robert Kohn.

Many thanks to Professor Mike O'Neil for reading the whole draft.

Ron Wu
1.25.2015

to heart hug sundae,
who fills me every Sunday.

Contents

1 Background

1.1 Magnetic Confinement

The magnetic field configuration inside of fusion reactors must provide confinement of the plasma. This requires magnetic fields to form closed flux surfaces. For detailed reason of this, see [9], particularly see sections around 11.5.2 of [9].

Here we give a heuristic discussion. We know that in uniform magnetic fields a free charge moves in helical paths along the magnetic lines. If the magnetic fields are like the ones in *toroidal* solenoids, i.e. magnetic fields lines are concentric circles about the polar axis of solenoids, we may think that free charges will move helically inside of solenoids. Indeed this is correct. But it's not desirable. Although charges are confined in the middle of the solenoids, there are too many overlaps of helical paths. That will create too many unpredictable collisions. We want charges to cluster and move in *layers*, like moving fluids. This sets up the stage for applying Euler's and Naiver Stokes equation from fluid dynamics.

What is the rectification? People instead design fusion reactors to have maximal flux surfaces. E.g. having magnetic field wrapping around the torus, which is a known solution, in cylindrical coordinate (ϕ, ρ, z), to the magnetohydrodynamic (MHD) equilibrium equation

$$p(\rho, z, \phi) = \frac{\rho^4}{8} + d_1 + d_2\rho^2 + d_3(\rho^4 - 4\rho^2 z^2) \tag{1.1}$$

d_1, d_2, and d_3 are geometric parameters. They determine the shape of the torus. We choose

$$d_1 \approx 0.075, \quad d_2 \approx -0.206, \quad d_3 \approx -0.0314$$

p is the equilibrium pressure profile, meaning that the interaction among charges is manifested in the macroquantity pressure gradient, just like in Euler's or Naiver Stokes equation. In equilibrium, they are balanced by the Lorentz forces acting on them, hence

$$\mathbf{F} = q\mathbf{v} \times \mathbf{B} = \mathbf{J} \times \mathbf{B} = \nabla p \tag{1.2}$$

This implies

$$\mathbf{B} \cdot \nabla p = 0 \tag{1.3}$$

We call p the pressure profile instead of just pressure, because only the gradient of p

matters, just like in Euler's or Naiver Stokes equation.

The associated **B** of (1.1) is

$$B_\rho = 8d_3\rho z \tag{1.4}$$

$$B_z = \frac{\rho^2}{2} + 2d_2 + 4d_3(\rho^2 - 2z^2) \tag{1.5}$$

$$B_\phi = \frac{\kappa}{\rho} \tag{1.6}$$

since ϕ is an independent variable in (1.1), any suitable constant κ will give equilibria, and we call such system an axisymmetric fusion reactor. Such **B** fields are illustrated on the cover of the book [9]. See its polar cross section, figure 1 on page 16.

Why do we call it axisymmetric? Consider two extreme cases:

i) $\kappa \to \infty$. This is essentially the "concentric-circles-about-the-axis-of-solenoids" magnetic fields we discussed before; we showed this is not a good confinement configuration. In the numerical experiments, the average $B_\rho \approx 0.0590$, the average $B_z \approx 0.1$. If we put $\kappa = 1$, then the average $B_\phi \approx 1.0134$, which is 100-200 times bigger than the other two components. So $k \approx 1$ is not a good confinement configuration. As we will show later, the conventional field line tracing method will return the exact answer as if $k \ll 1$. While as our method will give big irregularity, so our method is sensitive to tell confinement configurations are good or bad.

ii) $\kappa = 0$. The magnetic fields are wrapping around the torus with no ϕ components, so in any cross section we get a 2D fusion reactor. This is the simple 2D model we used to start testing our method.

For κ in between of 0 and $\ll 1$, magnetic fields are wrapping around the small circle and the big circle of the torus $S^1 \times S^1$. Since physics tells us magnetic fields are closed lines, if we *trace* the magnetic lines, we should eventually return to the starting points. But if the period of the small circle to the period of the big circle is not commensurate, it will take *infinitely* many iterations to return to the starting points. During the tracing process, we will obtain (hopefully) closed flux surfaces. This is the base of the conventional field line tracing method.

What if the period of the small circle to the period of the big circle is commensurate? This happens rare comparing to the non-commensurate cases. Because rational numbers are countable and irrational numbers are uncountable, and we can always find sequence of irrational numbers to be the limit of some rational numbers, for rationals are dense. In term of field line tracing method, if we encounter commensurate cases, we will try to move to a slightly different ρ to start with, this will change B_ϕ in (1.6), thus change the ratio of the period of the small circle to the period of the big circle. In the end, since physics tells us magnetic lines have no crossing. This ensures that we shall get a nested, continuous family (indexed by ρ) of flux surfaces. See picture 11.8 on page 262 in [9]. The

surfaces given by non-commensurate period of tracing are called irrational; the limiting surfaces given by commensurate period of tracing are called rational. As we will show this practice will not work well if we have a fractal, chaotic or Cantori system.

What does flux surfaces tell us? As we mentioned before, flux surfaces are the confinement surfaces, since particles are stick to (helically moving along) the flux surfaces. In fluid dynamics language, these are the layers of flow. They are defined to have constant pressure. In MHD, we say layers are the confinement of the thermal energy, aka they have constant temperatures. Extracting thermal energy is the holy goal of building fusion devices. Therefore analyzing temperature profiles (or synonyms pressure profiles) is the foundation, which is amount to analyze the confinement surfaces. This explains why knowing the confinement surfaces is more important than knowing the location of individual magnetic lines.

So one may think that since after solving MHD equilibrium, we will get both $p(\rho, z, \phi)$ and $\mathbf{B}$, then just find the contour surfaces of p (constant p), we will get confinement surfaces. The answer is not so fast. First of all the ideal force balance (1.2) is not exactly correct. For a multiparticle system like plasmas, one can have force out of balance and still be in (quasi-static) equilibrium, because of the involvement of diffusion. And the amount of tolerance to the out of balance is called relaxation. Therefore figuring out temperature profiles and confinement surfaces are two separated tasks. And the study of the discrepancy of the two will provide us insight of relaxation time (1/diffusion constant) and relaxation length (diffusion skin), known as transport theory.

For this project our goal is to produce images of magnetic confinement surfaces for given $\mathbf{B}$. The given magnetic fields may be directly detected by sensors inside of fusion devices or calculated numerically from solutions of MHD equilibrium.

1.2 Asymmetric Systems: Chaos

Ideal MHD, axisymmetric system (e.g. (1.4), (1.5), (1.6) with clear separations of layers), is the analog of ideal fluid. In real fluid, because of viscosity, layers will break and turbulence may be created. In MHD, a realistic fusion reactor will too have turbulence. In sum, we can have

i) formation of magnetic islands, analogous to laminar flow in fluid dynamics. This is sort of like breaking one big bubble into a few small bubble, and each moves parallelly and independently. This is a critical state, because it breaks nested layers of ideal MHD, but not yet becomes chaos.

Article [13] gave an example of such $\mathbf{B}$ in toroidal coordinates. We converted it into

cylindrical coordinate (ρ, z, ϕ) for later convenience, cf the end of subsection 2.4.3.1.

$$B_\rho = B_r\rho - B_\theta z \tag{1.7}$$
$$B_z = B_r z + B_\theta \rho \tag{1.8}$$
$$B_\phi = \rho \tag{1.9}$$

where

$$\begin{aligned} r &= \sqrt{\rho^2 + z^2} \\ \theta &= \arctan z/(\rho - 1) \\ B_r &= \iota_0 + 2\iota_1 r^2 - 2\epsilon_1 \cos(2\theta - \phi) - 3\epsilon_2 r\cos(3\theta - \phi) \qquad (1.10) \\ B_\theta &= -2\epsilon_1 \sin(2\theta - \phi) - 3\epsilon_2 r \sin(3\theta - \phi) \qquad (1.11) \end{aligned}$$

choosing

$$\iota_0 = 0.29, \quad \iota_1 = 0.38, \quad \epsilon_1 = 0.0015, \quad \epsilon_2 = 0.005$$

As we will see (figure 2 on page 16), the 2 and 3 frequency coefficients inside of sin and cosine in (1.10) and (1.11) will produce interesting polar curves with 2 and 3 islands.

ii) fractal and chaotic magnetic lines, analogous to turbulence flow in fluid dynamics. The state of chaos is defined through the idea of Poincare recurrence theorem. Because magnetic field lines are closed and no crossing, there must be some outer most magnetic confinement surface that contains the whole fields. Hence the assumption of Poincare recurrence theorem is satisfied, i.e. the space is bounded. If a distinct closed surface is not being produced, the only alternative is that field line tracing will fill the whole fractal volume thus it will return arbitrarily close to the starting point without forming a clear object.

iii) Cantori system. This is a special case of chaotic magnetic line. It is analogous to eddy in fluid dynamics. It gets its name from Cantor set: an uncountable set that has measure zero. From the field line tracing perspective, it is a black hole. It tracks field line trace and cannot get out. This is some kind of delta function in $\mathbf{B}$ (i.e. ∞ slope) of MHD solutions. How to represent them accurately numerically is an open question. But still if the field line tracing is near the Cantori, [10] showed that numerical simulation could be done, and it could take extremely long and, not practical, e.g. 10^{10} iterates to form the shape of the fractal volume.

Above shows another disadvantage of field line tracing. If two points are separated by 10^{10} iterates, field line tracing will still pretend they are on the same layer. This is clearly against causal relationship. During this time ($\gg$ coherence time) the equilibrium $\mathbf{B}$ has charged, the clustered particles around these two points have no causal relationship, so we shall not pretend them are on the same layer, nor pretend them having same temperature.

This is like saying since seas are all connected, we pretend what is happening in the Atlantic ocean (e.g. ocean currents, tsunami) will be also happening in the Pacific.

In comparison to our method, our method draws confinement surface from information from neighboring points, i.e. points on the same layers and points not on the same layer but in causal contact, and we have a mechanism that can adjust causal distance (controlled by iterations) with respect to coherence time. While as in field line tracing method, one cannot simply change number of iterations, because e.g. in the Cantori case field line tracing progresses very slow. This doesn't apply to the island example, see figure 2 on page 16. The 2-islands (or 3-islands) are in direct causal contact. Both field line tracing method and our method will connect them in short distance despite that they look separated in the 2D cross section.

1.3 Conventional Field Line Tracing

So much has been said about field line tracing. Let's give an algorithm of field line tracing method and summarize what the good and bad things about field line tracing are.

1.3.1 Ideal 2D Example

First consider $\kappa = 0$ in (1.6). So in (any ϕ) cross section, relabel $y := z$, $x := \rho$, we have

$$\frac{\Delta y}{\Delta x} = \frac{B_y}{B_x}$$

That is, the curve $y = y(x)$ is given by

$$y' = \frac{B_y}{B_x} \tag{1.12}$$

It is a first order non-linear ordinary differential equation. One can simply call Matlab function ode45. It is mainly Runge-Kutta 4-5, with 4th order accuracy, i.e. error goes down by power of 4 wrt the grid size. If one concerns that B_x becomes 0, i.e. the RHS of (1.12) blows up, one can instead of solving the following system

$$\begin{cases} x'(t) = B_x \\ y'(t) = B_y \end{cases}$$

and connect x and y by t.

Instead of calling ode45, one can write Runge-Kutta 4-5 explicitly, for it is very simple to code. Say (x_0, y_0) is the initial points at $t = 0$, then the next point (x_1, y_1) at $t = \Delta t$

is

$$x_1 = x_0 + (k_1 + 2k_2 + 2k_3 + k_4)/6$$

Similar expression for y_1,

$$y_1 = y_0 + (j_1 + 2j_2 + 2j_3 + j_4)/6$$

where

$$k_1 = \Delta t \cdot B_x(x_0, y_0) \tag{1.13}$$
$$k_2 = \Delta t \cdot B_x(x_0 + k_1/2, y_0 + j_1/2) \tag{1.14}$$
$$k_3 = \Delta t \cdot B_x(x_0 + k_2/2, y_0 + j_2/2) \tag{1.15}$$
$$k_4 = \Delta t \cdot B_x(x_0 + k_3, y_0 + j_3) \tag{1.16}$$

similar expression for the j's. Then use (x_1, y_1) as the starting point to compute (x_2, y_2), and so on.

Discussions:

To do (1.13)-(1.16), one has to know **B** in the whole interval. However **B** that is given either by sensors at the grid points inside of fusion reactors or calculated numerically from MHD equilibrium is usually not a continuous function. Hence some kind of interpolation has to be done. It can be done either before solving Runge-Kutta and obtain an analytic expression of **B** or in each step of Runge-Kutta obtain the interpolation value at the desired Runge-Kutta internal calculation points. In either way, polynomial interpolation (e.g. cubic interpolation, known as spline) are used.

If interpolation is done before Runge-Kutta, one has to invert some big matrix to ensure accuracy on the big interval. This increases computing time. There are some faster methods like Barycentric Lagrange polynomial described in [11], but they impose additional constraints on the data. If interpolation is done during Runge-Kutta, only small matrices are inverted, however, since it happens in every step of Runge-Kutta, the overall computing time is usually longer than using our method.

Contrary to our method, we use no interpolation. Everything is done on the grid. In our algorithm we first assign real numbers to each grid point. Then we update the numbers according to the **B** on the grids of neighboring points. In the end, we reexamine those numbers. We plot surfaces that have same real numbers. These are the confinement surfaces. Why is our method less accurate? Because we don't have values in between of grids. However we do use high order derivative, but it is not Runge-Kutta. Recall Runge-Kutta 4-5 divides one interval into 4 small steps, our high order derivative is taken in consideration of 2 lattice points on the right and 2 lattice points on the left, called 5-point stencil derivative, which has also 4th order accuracy.

Therefore to be fair, later when we compare our method to the conventional method, i) we will choose no interpolation for the field line tracing method, just pretend we know $\mathbf{B}$ everywhere, hence we give the field line tracing method the best scenario it can possibly have; ii) we will put grid size 5 times larger to the field line tracing method than in our method. Our numerically experiment shows that the field line tracing method has typical error (defined later cf (2.25)) one out of hundred thousand; our method has error one out of thousand. But because in field line tracing each step depends on only one previous step. If there is an error in one step then the error will accumulate to the rest of the steps. In 3D tracing (see examples below) after extremely long distance the field line tracing error is piling up. In some extreme cases field line tracing may have very bad error that we don't even know they are there.

1.3.2 Ideal 3D Example

Let $\kappa \neq 0$ in (1.6). Relabel $z := \phi$,

$$\begin{cases} x'(t) = B_x \\ y'(t) = B_y \\ z'(t) = B_z \end{cases}$$

Since $B_z = \kappa/\rho$ in (1.6) is never zero (In the next 3D island example, the $B_z = \rho \to 0$ is not achievable either, because the center of the donut is not in domain), we can simplify to solve

$$\begin{cases} x'(z) = B_x/B_z \\ y'(z) = B_y/B_z \end{cases}$$

This is why in some literature [12] calls $z := \phi$ the "time" coordinate.

As we discussed before, the confinement surfaces are now intrinsically 3D. Traditionally people draw 2D plots, called Poincare plots, see figures in [13]. Pick a cross section to start with $z_0 = \phi_0$, and a starting point (x_0, y_0) on the cross section, choosing a $n \in \mathbb{N}$, and let

$$\Delta z = 2\pi/n,$$

so that after n steps, we will be back on that cross section. Record the point, then do another iteration. After many and many iterations, we will begin to see some shape formed on the cross section. With some experience, one can then move to a slightly different cross section and repeat the process. In doing so one can sort of tell if the flux surfaces are closed or not.

Below is a Poincare plot of $Z_0 = \phi_0 = 0$ of $\mathbf{B}$ (1.4)-(1.6) of three traces with $\kappa = 1$. Starting points (x_0, y_0) are $(1.136, 0)$, $(1.242, 0)$, and $(1.320, 0)$. We make $n = 40$ steps in

one turn and 500 turns per trace, so there are 500 points on each ellipse.

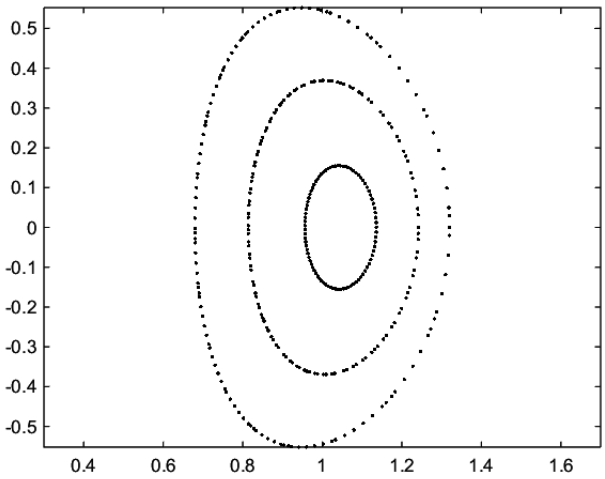

figure 1. Ideal MHD field line tracing

1.3.3 3D Islands Example

Move on to 3D example **B** (1.7)-(1.9). We reproduce figure below similar to figure 2(a) in [13] with 6 traces. Starting points are $(1.1, 0)$, $(1.2, 0)$, $(1.3, 0)$, $(1.4, 0)$, $(1.5, 0)$ and $(1.6, 0)$. We make $n = 40$ steps in one turn and 2000 turns per trace. We see the 2-3-island periodicity we mentioned before. We also see indeed that the three 3-islands are connected, so are the two 2-islands, because we start at $(1.3, 0)$, which is the edge of the most right island of the 3-islands, and field line tracing reaches to other 2 islands of the 3-islands.

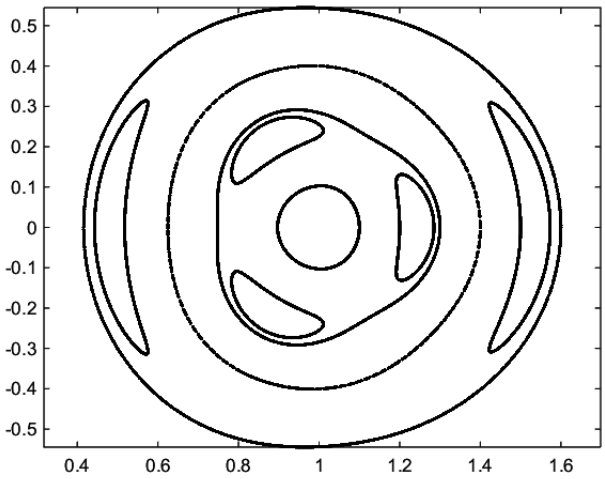

figure 2. Islands MHD field line tracing

Thus this is a vivid example of complicated twisted flux tubes. The 3-islands (or 2-islands) tubes are nested tubes, too. There are small islands inside of them. The flux

tube as a whole has periodicity. E.g. the 3-islands tube makes $3 \cdot (2\pi)$ polar angle to come back. Therefore it is not too hard to infer that since islands are nested too, if we get to the center of the island tube, there should be one trace magnetic line that makes $3 \cdot (2\pi)$ polar angle to come back. Hence this is the limiting rational surface we mentioned before, i.e. the ratio of the period of cycling around the big circle of the torus to the period of cycling the small circle of the torus is 3. Think this in reverse. Figuring out the location of the rational surface and understanding how it evolves will help us to understand how turbulence are created, because from above example we see rational surfaces are the turbulence centers.

1.3.4 Short Summary

Let us summarize what we have said about field line tracing.

1. At first sight the code is very simple. Only takes 20 lines. However when interpolation involves for interpolating $\mathbf{B}$ (because we need $\mathbf{B}$ everywhere to run Runge-Kutta), the code is not simple or not fast. Therefore choosing what kind of interpolations becomes a choice of headache. While as our method uses no interpolation.

2. It may be unreliable, when it hits turning points or chaotic fields, i.e. large slopes. Because it is too dependable on initial point, error accumulates. Just imagine there are two parallel train tracks, if at some point the two tracks get too close to each other and the train jumps from one track to the other track, we will basically go to a totally different destination. The strangest thing is that we have no idea where the error comes from and how big the error is for a generic magnetic field. While as our method depends on values on the whole domain; we don't follow one particular trace. We have developed an internal convergence mechanism that guides us whether we stay on the right track.

3. The idea of field line tracing doesn't take into causality consideration. While as our method utilizes physical properties of magnetic fields and allow information to propagate from one layer to the neighboring layers.

2 Our Method: High Order Lax-Friedrichs

We mentioned before (via analogy of fluid dynamics) that if (1.3) is true, then contour surface of p, isothermal surface, is the magnetic flux surface. Let us understand why it is true mathematically. Let us use ψ instead of p, since (1.3) is not true for real MHD, hence we want to show for given $\mathbf{B}$ if we can find ψ such that

$$\mathbf{B} \cdot \nabla \psi = 0 \tag{2.1}$$

then the contour ψ is the flux surface of $\mathbf{B}$.

Assume contour of ψ is closed. In 2D we know that (2.1) implies $\mathbf{B}$ is parallel to contour of ψ, because contour of ψ is too perpendicular to $\nabla\psi$. Therefore plotting the contour of ψ gives $\mathbf{B}$. In 3D one can have two vectors both perpendicular to a third vector, but these two vectors are not parallel to each other. It seems like (2.1) only implies $\mathbf{B}$ lives in the contour surface of ψ, so we take a cross-section then the trace of $\mathbf{B}$ on the cross-section (Poincare plots) and then the cross-section of the contour surface of ψ may not agree.

However since the boundary is smooth and closed, if $\mathbf{B}$ lives on the closed contour of ψ. Then by physics magnetic line no crossing, what is inside of the surface must content inside of it and what is outside must remain outside, therefore if $\mathbf{B}$ lives in the contour surface of ψ for the entire field inside of boundary (i.e. (2.1) is true on the whole domain), then the flux surface of $\mathbf{B}$ must be the same contour surface of ψ. QED

This is the first showcase of how our method utilizes physical properties of magnetic fields. To solve (2.1), we use the most natural choice: method of Lax-Friedrichs Hamiltonian. Before we do that, let's see the above proof in a real example.

Referring to figure 1 on page 16, let's compute contour surface of ψ. Using (1.1) to compute the p values at the starting points: $(1.136, 0)$, $(1.242, 0)$, and $(1.320, 0)$. p are -0.035, -0.0202, and 0 respectively. Then we draw the contour at those values in the following diagram: dash lines–field line tracing (same as figure 1); solid color lines–contour of p. Since p is independent of ϕ, any cross section should be the same. We see clearly the contour of p (of any ϕ) coincides perfectly with the cross section of the flux surface of $\mathbf{B}$ at $\phi = 0$.

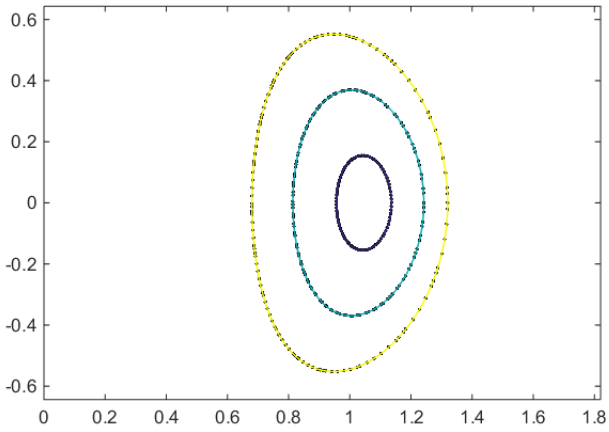

figure 3. Ideal MHD

2.1 Lax-Friedrichs

2.1.1 Lax-Friedrichs Hamiltonian

Here we are going to give a derivation of Lax-Friedrichs which is not found in the texts or paper [20]. In en route we will also derive Godunov Hamiltonian. The idea of Lax-Friedrichs and Godunov method is Taylor expansion on steroid. To save paper, let's assume our Hamiltonian (2.1) is 2D,

$$H(t, \psi, \partial_x\psi, \partial_y\psi) = B_x\partial_x\psi + B_y\partial_y\psi = R(\psi) \tag{2.2}$$

for us $R \equiv 0$ and boundary value problem

$$\psi_{bdy} = 0 \tag{2.3}$$

(later we will show $\psi_{bdy} = 0$ is not necessary. Any constant will do: $\psi_{bdy} = const$)

Borrowing idea from Hamiltonian mechanics, we treat $t, \psi, \partial_x\psi, \partial_y\psi$ as independent variables. Later we will identify $\Delta t = 1$, $d\psi/dt = \psi^{(n+1)} - \psi^{(n)}$, $q \leftrightarrow \psi$, $p_x \leftrightarrow \partial_x\psi$, $p_y \leftrightarrow \partial_y\psi$ (cf (2.18)). Since our H is 1st order in $\partial_x\psi, \partial_y\psi$, so first order expansion is exact. Then

$$H\left(\frac{\partial_x^+\psi(x_i,y_j) + \partial_x^-\psi(x_i,y_j)}{2}, \frac{\partial_y^+\psi(x_i,y_j) + \partial_y^-\psi(x_i,y_j)}{2}\right)$$

$$= H\left(\partial_x^\mp\psi(x_i,y_j), \partial_y^\mp\psi(x_i,y_j)\right) \pm \left(\frac{\partial H}{\partial(\partial_x\psi)}\right)_{\partial_x\psi=\partial_x^\mp\psi} \cdot \frac{\partial_x^+\psi(x_i,y_j) - \partial_x^-\psi(x_i,y_j)}{2}$$

$$\pm \left(\frac{\partial H}{\partial(\partial_y \psi)} \right)_{\partial_y \psi = \partial_y^{\mp} \psi} \cdot \frac{\partial_y^+ \psi(x_i, y_j) - \partial_y^- \psi(x_i, y_j)}{2} \tag{2.4}$$

where (x_i, y_j) is grid point, $\partial_{x,y}^{\pm} \psi$ are left/right derivatives. We ignore the y variable, because it works the same as x variable. Hence we get,

$$\begin{aligned} H\left(\partial_x^{\mp} \psi(x_i, y_j), \cdot\right) &= H\left(\frac{\partial_x^+ \psi(x_i, y_j) + \partial_x^- \psi(x_i, y_j)}{2} \right) \mp B_x \left(\frac{\partial_x^+ \psi(x_i, y_j) - \partial_x^- \psi(x_i, y_j)}{2} \right) \\ &= H\left(\partial_x^c \psi\right) \mp B_x \left(\frac{\psi(x_{i+1}, y_j) - 2\psi(x_i, y_j) + \psi(x_{i-1})}{2\Delta x} \right) \end{aligned}$$

where $\partial_x^c \psi = (\psi(x_{i+1}, y_j) - \psi(x_{i-1}, y_j))/2\Delta x$, central difference derivative, and

$$(\partial H / (\partial(\partial_x \psi))) = B_x \tag{2.5}$$

It will be apparent later that why we aren't hurry to define the artificial viscosity $\sigma_x = |B_x|$. This has nothing to do with the viscosity in MHD model. Here we have no other choices, but (2.5). For general H (not (2.2)), first order Taylor may not be good, so there is a choice we have to made, i.e. picking a good Taylor remainder. It turns out to pick

$$\sigma_x = \max_{[\partial_x^- \psi(x_i, y_j), \partial_x^+ \psi(x_i, y_j)]} \left| \frac{\partial H}{\partial(\partial_x \psi)} \right| \tag{2.6}$$

that are allowed by the Taylor theorem in the interval from $[\partial_x^- \psi(x_i, y_j), \partial_x^+ \psi(x_i, y_j)]$ (WLOG assume $\partial_x^- \psi(x_i, y_j) < \partial_x^+ \psi(x_i, y_j)$). Why max? because we want to get maximal speed of convergence, we will discuss this later.

Now pull out $\psi(x_i, y_j)$ from above

$$\psi(x_i, y_j) = \frac{\Delta x}{\pm B_x} \left[H\left(\partial_x^{\mp} \psi(x_i, y_j)\right) - H\left(\partial_x^c \psi\right) \right] + \frac{\psi(x_{i+1}, y_j) + \psi(x_{i-1}, y_j)}{2} \tag{2.7}$$

Till this point, everything is exact. We don't have to worry that $B_x = 0$, because the full expression has B_y as we'll see equation (2.21). The case when both $B_{x,y} = 0$ can be easily excluded by inspection, i.e. not to choose grid points lying on $\mathbf{B} = \mathbf{0}$.

(2.7) is really two equations, i.e. two different ways to Tayler expand $\psi(x_i, y_j)$. The Lax-Friedrichs method assumes, when $B_x > 0$, we should choose

$$\psi(x_i, y_j) = \frac{\Delta x}{+B_x} \left[H\left(\partial_x^- \psi(x_i, y_j)\right) - H\left(\partial_x^c \psi\right) \right] + \frac{\psi(x_{i+1}, y_j) + \psi(x_{i-1}, y_j)}{2}$$

and assume

$$H(\partial_x^- \psi(x_i, y_j)) \approx H(\partial_x \psi(x_i, y_j)); \tag{2.8}$$

when $B_x < 0$, we should choose

$$\psi(x_i, y_j) = \frac{\Delta x}{-B_x}\left[H\left(\partial_x^+\psi(x_i, y_j)\right) - H\left(\partial_x^c\psi\right)\right] + \frac{\psi(x_{i+1}, y_j) + \psi(x_{i-1}, y_j)}{2}$$

and assume

$$H(\partial_x^+\psi(x_i, y_j)) \approx H(\partial_x\psi(x_i, y_j)). \tag{2.9}$$

In above discussion we have adopted notations: $\partial_x^{\pm,c}\psi$ with superscript $\pm$ or c means the finite difference partial derivative, and without any superscript $\pm$ or c, $\partial_x\psi$ means the true derivatives. $H(\partial_x^{\pm,c}\psi)$ is the Hamiltonian evaluated at $\partial_x^{\pm,c}\psi$, so is the $H(\partial_x\psi)$, which is the true Hamiltonian in (2.2).

The physical interpretation of (2.8), (2.9) is that ψ is going with the flow. We think this is the reason why Lax-Friedrichs is the right method for us:

$$\text{Universality: Going With the Flow.} \tag{2.10}$$

As a side note one may get suspicion about above assumption and wonder that maybe $H(\partial_x^c\psi)$ is even better approximating to $H(\partial_x\psi)$, i.e.

$$H(\underbrace{\frac{\psi(x_{i+1}, y_j) - \psi(x_{i-1}, y_j)}{2\Delta x}}_{\partial_x^c\psi}) \approx H(\partial_x\psi(x_i, y_j)). \tag{2.11}$$

especially when ψ'' is nearly constant, i.e. no prefer directions, on $[x_{i-1}, x_{i+1}]$, which is the starting point of so called Godunov Hamiltonian. (2.11) has a different physical interpretation

$$\text{Another Universality: Dumping the Flow.} \tag{2.12}$$

We will investigate Godunov in the next section.

In the computational community, people call (2.10) the upwinding scheme, and (2.12) the central differences scheme in solving hyperbolic conservative pdes [1].

Now (2.7) becomes simply

$$\psi(x_i, y_j) = \frac{\Delta x}{\sigma_x}\left[R(x_i, y_j) - H\left(\partial_x^c\psi\right)\right] + \frac{\psi(x_{i+1}, y_j) + \psi(x_{i-1}, y_j)}{2} \tag{2.13}$$

where $\sigma_x = |B_x|$. We arrive Lax-Friedrichs Hamiltonian.

2.1.2 Lax-Friedrichs Sweeping

Up to this point, everything is almost exact, because we assume ψ is smooth. Since $\psi(x, y)$ appears on both sides of equation, we think we can solve boundary value pde (2.2) by *iterations*. The next big leap is to fix $\psi = 0$ on the boundary, (later we will show

for our Hamiltonian we don't even have to specify the boundary) and put initially

$$\psi^{(0)} = M \tag{2.14}$$

for other points in the domain, M is some non-zero constant value (positive or negative), then feed into (2.13) and start iteration, assuming all the time

$$H(\partial_x \psi(x_i, y_j)) = R(x_i, y_j) \tag{2.15}$$

By the way this shows R is the energy. In other words, during the iterations, especially the beginning few loops, we know that $\psi^{(n)}$ is no way close to a solution of (2.2), or in terms of physical terminology $\psi^{(n)}$ is not on the path of equation of motion. But we assume (2.15) anyway, i.e.

$$\psi^{(n+1)}(x_i, y_j) = \frac{\Delta x}{\sigma_x}\left[R - H\left(\partial_x^c \psi^{(n)}\right)\right] + \frac{\psi^{(n)}(x_{i+1}, y_j) + \psi^{(n)}(x_{i-1})}{2} \tag{2.16}$$

Here is the origin of the jargon–viscosity (σ_x). Why is called the viscosity? As time evolution with $\Delta t = 1$,

$$\ldots \to \psi^{(n-1)} \to \psi^{(n)} \to \psi^{(n+1)} \to \ldots \tag{2.17}$$

Recall one of the Hamiltonian equations, ignoring y

$$\frac{dq}{dt} = \frac{\partial H}{\partial p} \qquad \text{(identify } q \leftrightarrow \psi,\ p \leftrightarrow \partial_x \psi) \tag{2.18}$$

Hamiltonian equations don't require energy to be fixed, i.e. no need $H = R$, so in our case it becomes

$$\frac{\psi^{(n+1)} - \psi^{(n)}}{\Delta t} = \frac{\partial H}{\partial(\partial_x \psi)} = B_x \tag{2.19}$$

showing that the larger the B_x (viscosity) is, the faster ψ evolves (energy decays). LHS of (2.18) or (2.19) gives evolution of ψ and RHS gives evolution of energy.

We can also do a more mathematical derivation than (2.19). From (2.16), $\Delta t = 1$

$$\begin{aligned}
\frac{\psi^{(n+1)}(x_i) - \psi^{(n)}(x_i)}{\Delta t} &= \frac{\psi^{(n)}(x_{i+1}) + \psi^{(n)}(x_{i-1}) - \psi^{(n-1)}(x_{i+1}) - \psi^{(n-1)}(x_{i-1})}{2} \\
&+ \underbrace{\frac{\Delta x}{\sigma_x}\left[-H\left(\partial_x^c \psi^{(n)}(x_i)\right) + H\left(\partial_x^c \psi^{(n-1)}(x_i)\right)\right]}_{-\text{sign}_{(B_x)} \frac{\psi^{(n)}(x_{i+1}) - \psi^{(n)}(x_{i-1}) - \psi^{(n-1)}(x_{i+1}) + \psi^{(n-1)}(x_{i-1})}{2}} \\
&\qquad \underbrace{\frac{\Delta x}{\sigma_x}\left[-B_x\left(\partial_x^c \psi^{(n)}(x_i) - \partial_x^c \psi^{(n-1)}(x_i)\right)\right]}_{} \\
&= \begin{cases} \psi^{(n)}(x_{i-1}) - \psi^{(n-1)}(x_{i-1}) & B_x > 0 \\ \psi^{(n)}(x_{i+1}) - \psi^{(n-1)}(x_{i+1}) & B_x < 0 \end{cases}
\end{aligned} \tag{2.20}$$

This derivation approximates the actual iteration update process (or more physically we can say it shows how data update propagates), which agrees perfectly to our physical intuition and remark surrounding (2.10). (2.20) also shows the direction of data propagation is fastest if we can follow the flow. This suggests that we should think about other sweeping directions.

Equations (2.19) and (2.20) work in general. In our case, combining both $x,\ y$, the iteration rule is

$$\psi^{(n+1)}(x_i,y_j)=\frac{1}{\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}}\left[-B_x(x_i,y_j)\partial_x^c\psi^{(n)}(x_i,y_j)-B_y(x_i,y_j)\partial_y^c\psi^{(n)}(x_i,y_j)\right.$$
$$\left.+|B_x(x_i,y_j)|\frac{\psi^{(n)}(x_{i+1},y_j)+\psi^{(n)}(x_{i-1},y_j)}{2\Delta x}+|B_y(x_i,y_j)|\frac{\psi^{(n)}(x_i,y_{j+1})+\psi^{(n)}(x_i,y_{j-1})}{2\Delta y}\right] \tag{2.21}$$

thus

$$\left(\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}\right)\left(\psi^{(n+1)}(x_i,y_j)-\psi^{(n)}(x_i,y_j)\right)$$
$$=\begin{cases}\frac{|B_x(x_i,y_j)|}{\Delta x}\left(\psi^{(n)}(x_{i-1},y_j)-\psi^{(n-1)}(x_{i-1},y_j)\right)+(x\leftrightarrow y) & B_x>0,\ B_y>0\\ (x_{i-1}\leftrightarrow x_{i+1})+(x\leftrightarrow y) & B_x<0,\ B_y<0\\ (y_{i-1}\leftrightarrow y_{i+1}) & B_x>0,\ B_y<0\\ (x_{i-1}\leftrightarrow x_{i+1}) & B_x<0,\ B_y>0\end{cases} \tag{2.22}$$

where $(x\leftrightarrow y)$ means same expression with x,y switched. Therefore the idea of going with the flow is even clearer in the above 2D expression.

2.1.3 Convergence Mechanism

The initial $\psi^{(0)}$ will definitely have $H\neq R$ so we can think of the evolution process as if the system is dissipating or absorbing energy until $H=R$. When this almost occurs, because from (2.21), if $\psi^{(n)}=\text{const}$ for all $x,\ y$,

$$\begin{aligned}&\psi^{(n+1)}(x_i,y_j)\\ =\ &\frac{|B_x(x_i,y_j)|\frac{\psi^{(n)}(x_{i+1},y_j)+\psi^{(n)}(x_{i-1},y_j)}{2\Delta x}+|B_y(x_i,y_j)|\frac{\psi^{(n)}(x_i,y_{j+1})+\psi^{(n)}(x_i,y_{j-1})}{2\Delta y}}{\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}}\\ =\ &\psi^{(n)}(x_i,y_j)\end{aligned} \tag{2.23}$$

hence

$$\psi^{(\infty)}(x_i,y_j)\to\text{const}\equiv 0 \tag{2.24}$$

is the fix point solution.

Notice that (2.24) assumes the constant boundary condition $\psi_{bdy} = 0$ or $\psi_{bdy} = const$ for any constant number. $R(x, y)$ is known as the source term. Without it, (2.2) is a homogenous 1st order pde. If Lax-Friedrichs works, iterations should eventually converge to the fix solution of (2.2). If that happens, we are confident that along the way the contour of $\psi^{(n)}$ will get closer and closer to the magnetic field lines as iterations increase.

The second mathematical issue we should discuss is the uniqueness. Clearly the boundary value problem (2.2) with $\psi = 0$ at boundary doesn't have unique solution, e.g. ψ^2 is a solution too. Uniqueness is not a big issue, since as long as ψ satisfies (2.1), the contour of ψ will be flux surface of $\mathbf{B}$. What about the effect of artificial viscosity σ, which will smooth out iterations to $\psi \equiv 0$ (2.24)? So $\psi \equiv 0$ gives no contour at all? Well, as we will show later numerically (see tables on page 46), it will take infinitely many iterations to get there, and we will get a descent contour plot way before that happens. Hence it is not an issue either.

Both issues raised above are solved by the convergence mechanism we brief mentioned before. The critical step is we want ψ to approach to the fix point, cf (2.23). Therefore we will call the discrepancy between (2.23) and the real iterative formula (2.21) the error at nth iteration,

$$\text{error}^{(n)}(x_i, y_j) := \frac{\left|B_x(x_i, y_j)\partial_x^c\psi(x_i, y_j) + B_y(x_i, y_j)\partial_y^c\psi(x_i, y_j)\right|}{\frac{|B_x(x_i,y_j)|}{\Delta x} + \frac{|B_y(x_i,y_j)|}{\Delta y}} \tag{2.25}$$

We say if during the iteration, the error (2.25) goes down monotonically as n increases. We are on the path of convergence. And we are confident about the Lax-Friedrichs solutions are reasonably correct. If we want the test to be independent of the number of grid points, independent of the size of the domain, we put

$$\text{error}^{(n)} = \frac{1}{area(D)} \oiint_D (2.25) \tag{2.26}$$

As $\text{error}^{(n)} \to 0$, by (2.23)

$$\psi^{(n+1)}(x_i, y_j) \to \frac{|B_x(x_i, y_j)| \frac{\psi^{(n)}(x_{i+1},y_j)+\psi^{(n)}(x_{i-1},y_j)}{2\Delta x} + |B_y(x_i, y_j)| \frac{\psi^{(n)}(x_i,y_{j+1})+\psi^{(n)}(x_i,y_{j-1})}{2\Delta y}}{\frac{|B_x(x_i,y_j)|}{\Delta x} + \frac{|B_y(x_i,y_j)|}{\Delta y}}$$

this is a weighted average. That is because if at (x_i, y_j), $|B_x| \ll |B_y|$, we get a 1D expression

$$\psi^{(n+1)}(x_i, y_j) \to \frac{\psi^{(n)}(x_i, y_{j+1}) + \psi^{(n)}(x_i, y_{j-1})}{2}$$

the true 1D average.

The numerator of (2.25) suggests there is another possible error we may want to define

$$\frac{\mathbf{B}\cdot\nabla^c\psi_{LF}}{\|\mathbf{B}\|\,\|\nabla^c\psi_{LF}\|}=\angle(\mathbf{B},\nabla^c\psi_{LF})$$

the angle between $\mathbf{B}$, $\nabla^c\psi_{LF}$. c stands for central difference partial derivatives; LF for Lax-Friedrichs solutions. Similarly if we want the test to be independent of the number of grid points, independent of the size of the domain, and independent of the magnitudes of ψ_{LF} and $\mathbf{B}$, we put

$$\frac{1}{area(D)}\iint\!\!\!\!\!\!\diagup_D\ \frac{\left|\mathbf{B}\cdot\nabla^c\psi_{LF}^{(n)}\right|}{\|\mathbf{B}\|\left\|\nabla^c\psi_{LF}^{(n)}\right\|} \tag{2.27}$$

We call it error_deg to distinguish from (2.25).

$$\text{error\_deg}^{(n)}=\frac{\pi}{2}-\cos^{-1}(2.27) \tag{2.28}$$

is the angle between the actual magnetic vector and the ψ_{LF} contour line, averaged over the whole domain.

We think central difference partial derivatives suffices because in the Hamiltonian we use central difference too. cf (2.13). Thus the error measurement is no more or no less accurate than the Hamiltonian itself. However later when we move to high order Lax-Friedrichs, we will replace the derivatives in error by high order derivatives too.

What does convergence mean? In plain English, it means for our numerical experiment, if we i) increase number of iterations or ii) decrease the size of grid space, we will always get better results, confirmed later see tables on page 46.

2.2 Side Note: Godunov

We want to discuss a remark alluding to (2.11). In that remark we mentioned that there might be another possibility to substitute $H(\partial_x^c\psi(x_i,y_j))$ for $H(\partial_x\psi(x_i,y_j))$.

Suppose we substitute $H(\partial_x^c\psi(x_i,y_j))$ for R, so equation (2.7) becomes

$$\psi(x_i,y_j)=\frac{\Delta x}{\sigma_x}\left\{\begin{array}{ll} H(\partial_x^-\psi(x_i,y_j))-R & B_x>0\\ H(\partial_x^+\psi(x_i,y_j))-R & B_x<0\end{array}\right\}+\frac{\psi(x_{i+1},y_j)+\psi(x_{i-1},y_j)}{2} \tag{2.29}$$

we arrive Godunov Hamiltonian. Or

$$\psi(x_i,y_j)=\begin{cases}\psi(x_i,y_j)+\frac{\psi(x_{i+1},y_j)+\psi(x_{i-1},y_j)}{2} & B_x>0\\ \psi(x_i,y_j)-\frac{\psi(x_{i+1},y_j)-\psi(x_{i-1},y_j)}{2} & B_x<0\end{cases} \tag{2.30}$$

Thus Godunov sweeping

$$\frac{\psi^{(n+1)}(x_i,y_j)-\psi^{(n)}(x_i,y_j)}{\Delta t}=\begin{cases}\frac{\psi^{(n)}(x_{i+1},y_j)+\psi^{(n)}(x_{i-1},y_j)}{2} & B_x>0\\ -\frac{\psi^{(n)}(x_{i+1},y_j)-\psi^{(n)}(x_{i-1},y_j)}{2} & B_x<0\end{cases} \tag{2.31}$$

We can compare (2.31) to Lax-Friedrichs sweeping (2.20). (2.20) has character of wave and (2.30) has character of diffusion, because it transcends spatial difference into temporal difference and it has "go-with-the-flow" universality and "dumping-the-flow" universality: follow the heat flow, temperature raises; reverse the follow, temperature decreases.

It seems like by (2.29), ψ will approach to the constant-zero solution equation (2.24) even faster, because diffusion dissipates energy faster. Following the similar derivation, starting from (2.4), we obtain the complete iteration rule, combining $x,\ y$

$$\psi^{(n+1)}(x_i,y_j)=\frac{1}{\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}}\times$$

$$\left(B_x(x_i,y_j)\left\{\begin{array}{l}\partial_x^-\psi^{(n)}(x_i,y_j)\ B_x>0\\ \partial_x^+\psi^{(n)}(x_i,y_j)\ B_x>0\end{array}\right\}+B_y(x_i,y_j)\left\{\begin{array}{l}\partial_y^-\psi^{(n)}(x_i,y_j)\ B_y>0\\ \partial_y^+\psi^{(n)}(x_i,y_j)\ B_y>0\end{array}\right\}+\right.$$

$$\left.|B_x(x_i,y_j)|\,\frac{\psi^{(n)}(x_{i+1},y_j)+\psi^{(n)}(x_{i-1},y_j)}{2\Delta x}+|B_y(x_i,y_j)|\,\frac{\psi^{(n)}(x_i,y_{j+1})+\psi^{(n)}(x_i,y_{j-1})}{2\Delta y}\right) \tag{2.32}$$

slightly more involved than (2.21). We simplify above using (2.30)

$$\psi^{(n+1)}(x_i,y_j)-\psi^{(n)}(x_i,y_j)=\frac{B_x(x_i,y_j)\partial_x^c\psi^{(n)}(x_i,y_j)+B_y(x_i,y_j)\partial_y^c\psi^{(n)}(x_i,y_j)}{\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}} \tag{2.33}$$

The second term on the right is the exact error (2.25) without the absolute sign. We have tried and it won't work for our problem, seems like Godunov flux oscillates too fast, never come to rest $H=R$. Compare Godunov Hamiltonian (2.33) to Lax-Friedrichs Hamiltonian (2.22), in the next section, we will show both are hyperbolic and not conservative but Lax-Friedrichs (2.22) is almost conservative. Normally both hyperbolicity and are conservation are required to apply Godunov and Lax-Friedrichs.

We can regularize (2.32) by making substitution

$$\partial_{x,y}^{\pm}\psi^{(n)}(x_i,y_j)\ \ \rightarrow\ \ \partial_{x,y}^{c}\psi^{(n)}(x_i,y_j)$$

change L/R derivative to central difference derivative, then we will get exactly (2.21) Lax-Friedrichs with - sign. But the - sign is to flip the **B** directions. That has no effect on the contour lines.

2.3 Lax-Friedrichs Extension

We mentioned Lax-Friedrichs works for our problem, but Godunov doesn't. We also mentioned both Lax-Friedrichs and Godunov are used to solve hyperbolic conservative pdes. [1] gives examples things go wrong when the conservative condition is dropped.

[17] A general first order *hyperbolic* pde of $u(t, \vec{x})$ is

$$\partial_t u + A(t, \vec{x}) \cdot \nabla_{\vec{x}} u = F(t, \vec{x})$$

and

$$\text{matrix } A(t, \vec{x}) \text{ is diagonalizable for all } t, \vec{x}$$

$A : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d \otimes \mathbb{R}^d$ and $F : \mathbb{R} \to \mathbb{R}$ are given, and solve for $u(t, x) : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$.

A *conservative* pde can be put into the divergence form

$$\begin{cases} \partial_t u + \nabla_{\vec{x}}(G(u)) = 0 \\ u|_{t=0} = g \end{cases} \tag{2.34}$$

$G : \mathbb{R} \to \mathbb{R}$, $g : \mathbb{R}^d \to \mathbb{R}$ are given, and solve for $u(t, \vec{x}) : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$.

So our pde (2.2) is first order, non-linear, hyperbolic, homogenous, static, but not locally conservative pde. "Local" means conservative wrt fix local frame, however our pde (2.2) is conservative wrt moving frame along the $\mathbf{B}$. Just like in Euler's equation for incompressible fluid

$$\frac{d\rho}{dt} + \mathbf{v} \cdot \nabla \rho = 0$$

ρ is conservative in the moving frame along the streamline, and $\mathbf{v} \cdot \nabla$ is called convective derivative.

Thus no guarantee that either Lax-Friedrichs or Godunov will work for us. But our numerical experiment shows clearly Lax-Friedrichs works. Hence we found a way to make a not strictly conservative pde into a conservative one, we call it an extension of Lax-Friedrichs method.

We believe that is because we recognize Lax-Friedrichs method has a guiding principle (2.10), go-with-the-flow, which is consistent with the conversation law (2.34); while as the guiding principle diffusion (2.12) of Godunov doesn't relate to conservative law nor does it go well with hyperbolicity, because wave equations are hyperbolic and diffusion are parabolic.

For our problem

i) The physical natural of magnetic field is conservative. Here conservative doesn't mean that $\mathbf{B} = \nabla V$, for some scalar function V, as defined by physicists. It just means that the field line is always closed–locally what is going in must come out.

ii) Lax-Friedrichs data flow is conservative. Because the update formula applied to our

problem (2.21) is equal to,

$$\psi^{(n+1)}(x_i,y_j)=\begin{cases}\frac{\frac{|B_x(x_i,y_j)|}{\Delta x}\psi^{(n)}(x_{i-1},y_j)+\frac{|B_y(x_i,y_j)|}{\Delta y}\psi^{(n)}(x_i,y_{j-1})}{\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}} & B_x>0,\ B_y>0\\ \frac{\frac{|B_x(x_i,y_j)|}{\Delta x}\psi^{(n)}(x_{i+1},y_j)+\frac{|B_y(x_i,y_j)|}{\Delta y}\psi^{(n)}(x_i,y_{j+1})}{\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}} & B_x<0,\ B_y<0\\ \frac{\frac{|B_x(x_i,y_j)|}{\Delta x}\psi^{(n)}(x_{i-1},y_j)+\frac{|B_y(x_i,y_j)|}{\Delta y}\psi^{(n)}(x_i,y_{j+1})}{\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}} & B_x>0,\ B_y<0\\ \frac{\frac{|B_x(x_i,y_j)|}{\Delta x}\psi^{(n)}(x_{i+1},y_j)+\frac{|B_y(x_i,y_j)|}{\Delta y}\psi^{(n)}(x_i,y_{j-1})}{\frac{|B_x(x_i,y_j)|}{\Delta x}+\frac{|B_y(x_i,y_j)|}{\Delta y}} & B_x<0,\ B_y>0\end{cases}\tag{2.35}$$

typically we choose $\Delta x=\Delta y$, so $\Delta x, \Delta y$ drop out.

Above shows clearly in one iteration $\psi^{(n+1)}(x_i,y_j)$ depends only 2 neighboring points in the direction opposite to the magnetic line. Hence the flow of data propagates in the direction of the magnetic fields. Since magnetic field line is conservative, the flow of Lax-Friedrichs data is conservative.

This use of physical properties of magnetic here is the second and the last time we utilize the physical properties of magnetic field in supporting our method. The first time was in the discussion surrounding (2.1), where we used the physical natural of magnetic field to show why solving ψ in (2.1) would give the magnetic flux surfaces.

Last but no least, (2.35) is a 2D formula. In 2D magnetic lines are always closed. However as we move to 3D, that divergence free magnetic field does not imply magnetic flux (confinement) surfaces are closed, hence the 3D Lax-Friedrichs internal flow of data propagation may not be closed either, therefore if the flux surfaces are not closed, we expect Lax-Friedrichs to break down.

Reversely, if we apply Lax-Friedrichs blindly to a given magnetic, and Lax-Friedrichs results break down (this can be determined from the convergence mechanism we discussed early), then we say the magnetic confinement configuration is not closed. Hence our method is capable to detect whether a given magnetic confinement configuration is closed or not.

2.4 Customize Lax-Friedrichs Sweeping

2.4.1 Short Summary

Let's summarize what we have said about Lax-Friedrichs.

1. Hyperbolicity & Conservation: we said Lax-Friedrichs and Godunov required Hamiltonian to be hyperbolic or at least parabolic. That is because in the discussion

following (2.17) we said the artificial viscosity would force Hamiltonian to go to 0, because $R \equiv 0$. In the algorithm we can force that to happen monotonically, thus force it to converge faster. If the Hamiltonian is not hyperbolic, Lax-Friedrichs and Godunov method will fail. Because hyperbola has two branches, one is bounded below and the other is bounded above, if starting from

$$M > 0$$

(cf (2.14)) we will be on the bounded below branch and Lax-Friedrichs will take Hamiltonian down to 0. If we start with $M < 0$, we will be on the bounded above branch and Lax-Friedrichs will take Hamiltonian up to 0. cf figure 5(a) & 5(b) on page 32. If Hamiltonian is neither bounded above or below, then Lax-Friedrichs solution will go unboundedly and fail. The flux conservative condition will guarantee that the contours of Hamiltonian are well-structured, so its minimum (or maximum) is well-defined. The data flow conservation will guarantee that updating formula will follow the path that goes to the minimum (or maximum).

2. Convergence: we have shown that convergence is intrinsically related to i) hyperbolicity & conservation of pde, and ii) the conservation of the data flow is due to the physical properties of magnetic field, iii) now we want to ask how many iterations (i.e. the number n in (2.35)) are needed to reach convergence. (2.35) shows one iteration $\psi^{(n+1)}(x_i, y_j)$ depends only 2 neighboring points in the direction opposite to the magnetic line. So if we trace this back

$$(n+1) \to (n) \to (n-1) \to ... \tag{2.36}$$

to see that in the ideal 2D example $\psi^{(n+1)}(x_i, y_j)$ only depends on points from the boundary of (2.2) upto the contour line where (x_i, y_j) lies. This further implies that if

$$\text{number of iterations} \ll \text{number of grid points} \tag{2.37}$$

the sweeping may not reach to the boundary. On the another hand if

$$\text{number of iterations} \gg \text{number of grid points} \tag{2.38}$$

this actually should not cause any problem, because it is part of convergence requirement. However too big iterations (trace back too much) causes causality issue, but usually we use iteration ~ 100 so causality is not an issue. Although one may think it may cause over dependence on the boundary especially because the location of the boundary is not exactly correct since we too approximate it by the discrete grids. cf figure 4 on page 31. However one may propose that since the

location of the boundary is not precise, so why not put the precise value on these points instead of just constant 0. That turns out to cause more problem when (2.38) is met, because this will destroy the convergence solution: $\psi = const = 0$, so $\psi^{(n)}$ doesn't converge to the fix solution. Combine (2.37) & (2.38), the minimum required number of iterations:

$$\text{number of iterations} \approx \sqrt{2} \cdot \text{number of grid points for 2D square grids} \tag{2.39}$$

$$\text{number of iterations} \approx \sqrt{3} \cdot \text{number of grid points for 3D cubic grids}$$

This is a special instant of CFL number. The precise number of iteration is determined by the sweeping directions, because sweeping direction is not the same as data flow direction. Later when we study anti-wedding spiral Lax-Friedrichs, we will have more to say about this.

3. Boundary value problem & initial value problem: Both Lax-Friedrichs and Godunov require some initial data to start the iteration. cf the beginning of section 2.1.2, but the initial data may not satisfy the pde. Later we will show we will put random numbers for $\psi^{(0)}$ to start iteration, so by default it is not an initial value problem. What about boundary? Our experiment shows we don't need it either. We may question when the boundary is not specified, e.g. in 2D we choose a rectangle grids, will data outside of the original boundary of (2.2) propagate into the domain? Many of these outside data are not correct in the Lax-Friedrichs iteration, because they are not on any closed magnetic contour line covered by the rectangular grid. So Lax-Friedrichs cannot provide them with correct values. Will these wrong values confuse the updating process? We find a way to solve this problem. We put constant 0 on the edge of the grids, then these bad points will be traced back to the edge of the grid, like in (2.36), so by (2.35) they will be 0, too. Finally our numerical experiment shows without specifying the boundary Lax-Friedrichs is about the same accurate as having the boundary. cf figure 9(b) on page 34.

2.4.2 Lax-Friedrichs Algorithm

First let's program Lax-Friedrichs for the ideal 2D example in section 1.3.1. We already know what the contour line should look like, figure 1 on page 16 and figure 3 on page 19. Notice in paper [20] algorithm requires M in (2.14) to be positive and forces $\psi^{(n)}$ to go down, hence update $\psi_{LF}^{(n+1)}$ at (x_i, y_j) by (2.21) only if

$$\psi_{LF}^{(n+1)}(x_i, y_j) < \psi_{LF}^{(n)}(x_i, y_j)$$

we find that to be unnecessary, because we mentioned before our data flow direction is already conservative, the path is on the convergence path. We don't need additional convergence imposition.

For the no boundary example, option 2, paper [20] algorithm puts in a computational boundary condition. In their words "to allow data out flow". We instead put a constant 0 along the edge of grids. That will lead bad point–outside data not on any closed magnetic contour line covered by the rectangular grid–to be 0. In doing so we will see what is outside of boundary has very little influence on inside. In other words, the contour boundary shields very well just like in option 1 below.

2.4.2.1 Option 1–with boundary

This is the traditional Lax-Friedrichs. The domain and the boundary can be found from setting $\psi <$ some constant from (1.1) or using field line tracing to get one contour line. For generic field we don't know ψ nor do we want to use field line tracing, so option 1–with boundary is not really a good option. We examine that the boundary in figure below is closed, so data outside of boundary will not propagate in. Later we will do high order Lax-Friedrichs. If we choose option 1, and high order derivatives will be used, then we will put 2 layers of boundary to prevent data outside propagate in.

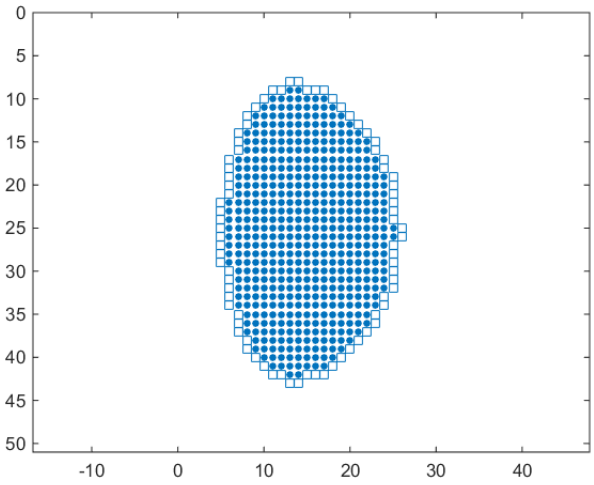

figure 4. Domain of ideal 2D example,
50 grid points in y, $\Delta x = \Delta y$

The figure 4 above shows the exact location of girds being used in the calculation. It is impressive to point out that such small number of ($\sim$ 500) grids points can produce good results.

The main Lax-Friedrichs algorithm is to fix ψ_{LF} value at boundary to be constant 0 and put $\psi_{LF}^{(0)} = M$ some (positive or negative) number inside of domain. Suppose we don't know a priori the shape of $\mathbf{B}$ lines, so we don't know the direction of data flows.

We sweep by (2.21) in all directions: from left to right and right to left, up to down and down to up, row by row and column by column. So each grid point will give 4 sweeping directions in one iteration. One can also keep track of the error measurement (2.25), and see indeed the error going down. The resulting Lax-Friedrichs ψ_{LF} is plotted below.

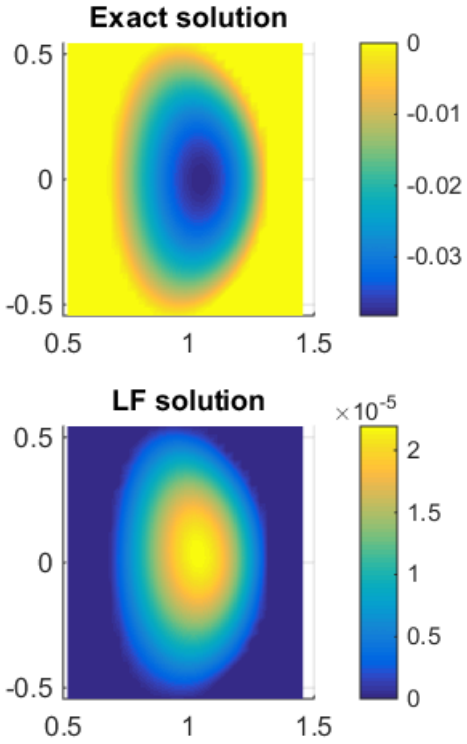

figure 5(a). 2D Surface Plots with chosen $M > 0$

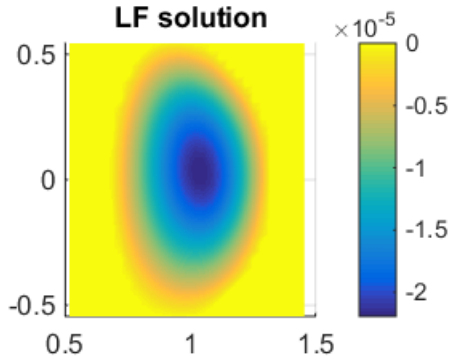

figure 5(b). 2D Surface Plots with chosen $M < 0$

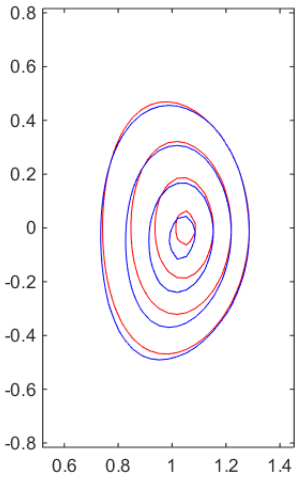

figure 6. Red: exact contours p.
Blue: contours of Lax-Friedrichs solution, same for $\pm M$

2.4.2.2 Option 2–without boundary

This is our first modification by ignoring the contour boundary. We put $\psi_{LF} = 0$ at the edge of grid then begin sweeping. The result is the following.

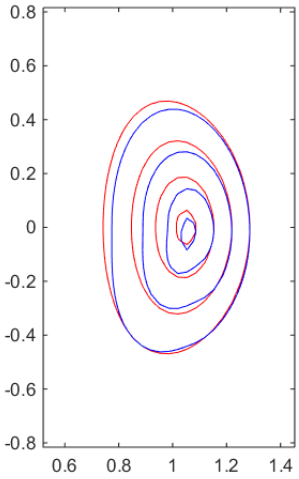

figure 7. Red: exact contours p.
Blue: contours of Lax-Friedrichs solution

In the figure above we solve Lax-Friedrichs on the whole rectangular gird, but when we graph the contour of ψ_{LF}, we only plot the contour of ψ_{LF} solution inside of boundary, because we discussed early that the solution outside of contour boundary was about 0. Indeed confirmed by the following picture

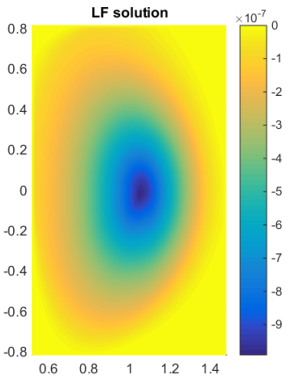

figure 8. surface plot of LF
solution in figure 7 on the whole rectangular domain

2.4.2.3 Short Summary

We can check the convergence by doubling the grid points from figure 4 on page 31, we should get better results.

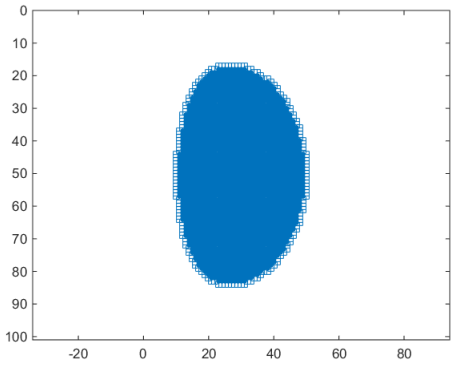

figure 9(a). grids double from figure 4,
100 grid points in y, $\Delta x = \Delta y$

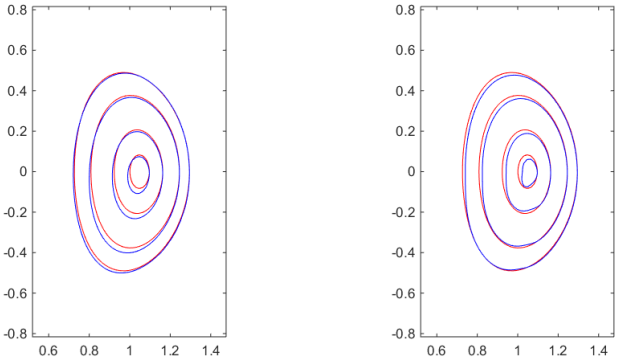

figure 9(b). Left Picture-Option 1-traditional Lax-Friedrichs
Right Picture-Option 2-without boundary

Later we'll use the same girds sizes in figure 4 on page 31, and compute them in high order Lax-Friedrichs we will see red lines and blue lines agree perfectly, cf figure 13(a) on page 43 & figure 14 on page 44.

In the beginning of this section, we figured out two important omissions from traditional Lax-Friedrichs (no need for boundary, no need for monotonicity) combining one other success in null divertor experiment (cf later section 2.4.4) will provide us the reasons why our method should work for 3D islands example, cf figure 2 on page 16, which is the future goal of this project.

That is because

1. No boundary: We mentioned before for generic field, we don't know the boundary a priori. Thanks to no boundary requirement, all we have to do is to put 0 at the edge of the square or cubic grid.

2. No monotonicity: We know from experience, if we only put 0 at the edge of the grid and a constant number uniformly inside, ψ_{LF} will be just one big dune, cf figure 5(a) on page 32, because inside uniformly constant gives 0 derivative, information can only propagate from the edge of grid to inside, not the other way around. Or look at (2.35), since $\psi^{(n+1)}(x_i, y_j)$ depends on only two neighbor points $\psi^{(n)}(x_i, y_j)$, if the two neighbor points have same value, then $\psi^{(n+1)}(x_i, y_j)$ will be that value too.

   However for the islands example we need multiple local maximums or local minimums, so not one big dune, but mountain with multiple tops. The solution to the problem is to put some initial arbitrary doping to $\psi^{(0)}_{LF}$. Thus we create initial non-zero derivatives somewhere inside, and thanks to no monotonicity requirement, these initial doping value can increase and decrease, so we are free to pick any initial values for them.

3. Null divertor: We already mentioned that in the no boundary case, points outside of contour boundary have little influence to points inside. This should be true for the islands example, where each island should act independently. Null divertor example provides a good test for that. Because near the divertor the boundary changes sharply. This is mimic of separation between islands.

2.4.3 More Modifications

So far we have emphasized a few points repeatedly:

- The Hamiltonian has to be hyperbolic.
- The data propagates in the direction of $\mathbf{B}$.

- That data propagation direction is conservative implies convergence.

How do we know they are indeed the causes? Let's think of some examples that do not have those properties. See what may happen?

2.4.3.1 Anti-Wedding Spiral

Let's check the second bullet point above. We mentioned that generic sweeping direction was not the same as the direction of data propagation, i.e. not the direction of $\mathbf{B}$. That makes Lax-Friedrichs less efficient, so let us try to change the sweeping directions so that it looks more or less like $\mathbf{B}$, nested elliptical loops.

For option 2-no boundary, let us try *labyrinth* sweeping. Keep everything else the same except the sweep directions. We start sweeping from the edge of the grid and spiral inwards.

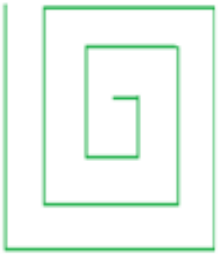

figure 10(a) Schematic picture of labyrinth sweeping

For option 1-with boundary, let us try *spiral* sweeping. We start sweeping from the edge of the potato boundary and spiral inwards.

figure 10(b) Schematic picture of spiral sweeping

Our experiments show that in such ways, we indeed obtain better results, and acquire less iterations. The number of iterations are close to (2.39).

Further improvement is possible. Since $\psi^{(n+1)}(x_i, y_j)$ depends on points from the boundary upto the contour line where (x_i, y_j) lies, we shall make the region near the boundary to be more accurate before proceeding to the center. Thus we propose "labyrinth or spiral on *anti-wedding cake*", hence sweeping starts near the boundary and stops near the boundary, then after each iteration, we move the stopping point a little bit toward

the center, so in the earlier loops points near the boundary get updated much more frequently.

One can further improve this, some kind of adoptive sweeping, hence the sweeping direction will change base on the knowledge of contour lines obtained from the previous iterations. Such method will likely be difficult to code and slow to run, so we don't try.

Because now our sweeping direction has a preferred direction: counterclockwise, which is the same as $\mathbf{B}$. We can simply change

$$\mathbf{B} \to -\mathbf{B}$$

and run labyrinth or spiral again. We see indeed sweep against the $\mathbf{B}$ direction has the worst results. So we proved our claim: sweeping in the direction of data propagation is most efficient.

This conclusion will be very important for us later when we do 3D ideal MHD example and 3D islands example. For both examples, the first task we encounter is to choose a good coordinates for Lax-Friedrichs. It turns out cylindrical coordinates is a good one, because in both examples the polar direction B_ϕ don't change along ϕ. cf (1.6) & (1.9). That means if we sweeping along the polar cylindrical direction, we only have to do one direction, not the opposite direction, so comparing to Cartesian coordinates we save half of running time and achieve better results.

Notice toroidal coordinates also has polar component, but it is bad coordinates for Lax-Friedrichs. Because grid points near $r \to 0$, r is the radius of the small circle of the torus, get far more power than points near the edge of the cross section. Cylindrical coordinates doesn't have this problem because i) on the cross section it is (ρ, z), 2D Cartesian; ii) near $\rho \to 0$, the center of the donuts, is not part of domain.

So we use cylindrical coordinates. For the 3D ideal MHD case, we will do one direction along polar and in the cross section, we can use either labyrinth or spiral sweeping, or maybe anti-wedding cake. For the 3D island case, we will do one direction along polar and in the cross section, we will use 2D traditional Cartesian sweeping.

2.4.3.2 Anti-Linearizing Ham

Let's check the 3th bullet point at the beginning of section 2.4.3. Let's change the Hamiltonian to

$$\frac{\mathbf{B} \cdot \nabla\psi}{\|\nabla\psi\|} = 0$$

Of course if ψ solves above pde, still contour of ψ is $\mathbf{B}$. But this pde is no longer 1st order in $\nabla\psi$, i.e. we *anti-linearize* it. Thus 1st order Taylor expansion is not exact, and σ_x is

not $|B_x|$, cf (2.6)

$$\begin{aligned}\sigma_x &= \max\left|\frac{\partial H}{\partial(\partial_x\psi)}\right| = \left|\frac{B_x}{\partial_y\psi}\right| \\ \sigma_y &= \max\left|\frac{\partial H}{\partial(\partial_y\psi)}\right| = \left|\frac{B_y}{\partial_x\psi}\right|\end{aligned}$$

The full iterations of (2.21) becomes

$$\psi^{(n+1)}(x_i,y_j) = \frac{1}{\frac{\sigma_x}{\Delta x}+\frac{\sigma_y}{\Delta y}}\left[\frac{-B_x(x_i,y_j)\partial_x^c\psi^{(n)}(x_i,y_j) - B_y(x_i,y_j)\partial_y^c\psi^{(n)}(x_i,y_j)}{\left\|\left\langle\partial_x^c\psi^{(n)}(x_i,y_j),\partial_y^c\psi^{(n)}(x_i,y_j)\right\rangle\right\|}\right. \tag{2.40}$$

$$\left. + \sigma_x\frac{\psi^{(n)}(x_{i+1},y_j)+\psi^{(n)}(x_{i-1},y_j)}{2\Delta x} + \sigma_y\frac{\psi^{(n)}(x_i,y_{j+1})+\psi^{(n)}(x_i,y_{j-1})}{2\Delta y}\right]$$

therefore we don't have (2.35). Indeed we have $\psi^{(n+1)}(x_i,y_j)$ depends on all 4 neighbors. Thus the elegant data propagation directions doesn't exist anymore. We shall check the convergence property.

First we find that for this Hamiltonian labyrinth sweeping or spiral sweeping do not make any improvement, as we shouldn't expect they will do much good. Second we find that now as iterations increases the error is not necessary going down.

This conclusion will be very important for us later when we do high order Lax-Friedrichs. Because in high order Lax-Friedrichs, the data flow direction is not conservative, so increasing iterations may not always be good, i.e. convergence is in question. Because of this and one other reason, (unlike in paper [21]) we don't think we should use high order Lax-Friedrichs alone. The other reason is that high order derivative change things very slow, it only make some minor changes. In paper [21] traditional Lax-Friedrichs is used to figure out the 2 layers of boundary (to prevent wrong data flow into the domain, as explained early), then use completely high order Lax-Friedrichs. In our method we don't need boundary; we use traditional Lax-Friedrichs to give a draft shape, then use high order Lax-Friedrichs to make it more refined.

2.4.4 Ideal 2D MHD with a Null Divertor

We now apply to a different domain. [18] It is one that has a null divertor. A divertor is place where particles enter or exit the fusion reactors. As we mentioned before, this is also a good test for testing Lax-Friedrichs with separatrix of islands. Also because at the divertor boundary changes sharply, we can test how good Lax-Friedrichs is in deal with sharp turns. Paper [18] gives an analytic solution of such $\mathbf{B}$ field.

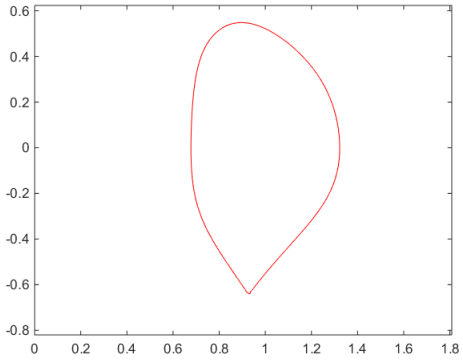

figure 11(a). actual domain of X-null divertor

We use no boundary and traditional sweeping, and choose 100 grid points in y, $\Delta x = \Delta y$. Notice in the example when we have no boundary, i.e. the rectangular boundary, ψ_{LF} should be mountain with multiple tops, so initial doping is essential.

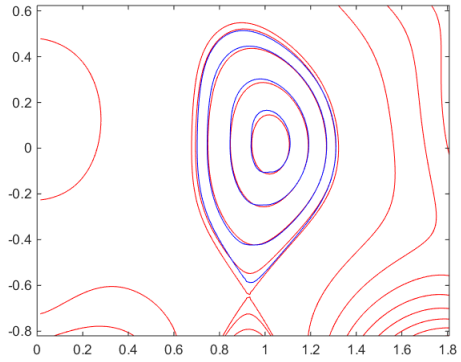

figure 11(b). Red: exact contours p.
Blue: contours of Lax-Friedrichs solution.

Again we did not plot Lax-Friedrichs solution outside of the closed contour domain, because they have incorrect values for they are not on any closed contour lines covered by the rectangular girds. However if we choose to plot them, we get

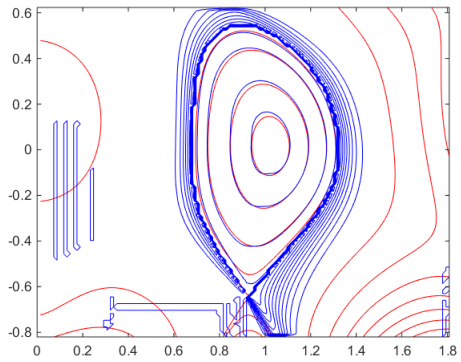

figure 11(c). Outside of closed contour domain, incorrect values

We are quite happy about the results. Not to mention that Lax-Friedrichs method is very fast too. Typically it takes about a few seconds in Matlab with 100×100 grids.

Next we add high order derivatives to our Hamiltonian, and in doing so we will see the blue lines and red lines will become perfectly coincided. The tradeoff is that in high order Lax-Friedrichs it will take much longer time, because i) $\psi^{(n+1)}(x_i, y_j)$ depends on 4 neighbors not 2; ii) high derivatives involve more operations.

Sometimes we can also get good result from traditional Lax-Friedrichs if we just simply increase grid points, which will increase running time also. Therefore which method should choose depends entirely on the running time.

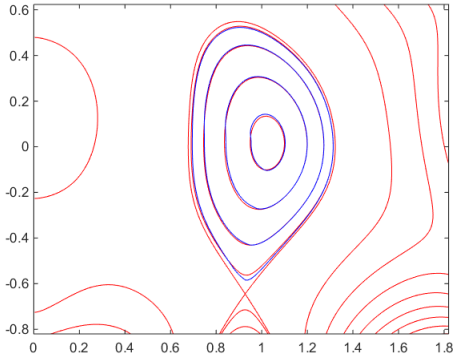

figure 12. no boundary and traditional sweeping,
400 grid points in y, $\Delta x = \Delta y$, running time 200 secs
Red: exact contours p.
Blue: contours of Lax-Friedrichs solution.

2.5 High Order Lax-Friedrichs

2.5.1 High Order Derivative: 3rd order WENO

In figure 12 apparently the left part of blue lines are not as good as the right part of blue lines, but we are not too worry about that, because we know the cause of that. It is caused by the traditional sweeping directions. We sweep all 4 directions: horizontal & vertical. In some directions the sweeping is opposite to $\mathbf{B}$ so that they do more harm than good.

What we are worry is the lowest part of blue line, where the contour line is supposed to make a sharp turn. Looks like that the simple Hamiltonian involve only central difference 1st derivatives doesn't capture the sharp turn very well. So we think WENO may help. See figure 3.1(a) in paper [14] the authors demonstrate that WENO is good for turning points.

WENO=Weighted Essentially Non-Oscillatory scheme

It is basically approximating L/R derivative at (x_i, y_j) by weighting in L/R derivatives of its neighbors $(x_{i\pm1}, y_{j\pm1})$. Here are the 3rd order WENO rules, see the bottom of page 2130 in paper [14],

$$\begin{aligned}\partial_x^+\psi(x_i,y_j) &= \frac{1}{2}\left(\partial_x^-\psi(x_{i+1},y_j)+\partial_x^-\psi(x_i,y_j)\right) \\ &\quad -\frac{w^+}{2}\left(\partial_x^-\psi(x_{i+2},y_j)-2\partial_x^-\psi(x_{i+1},y_j)+\partial_x^-\psi(x_i,y_j)\right) \qquad (2.41)\\ \partial_x^-\psi(x_i,y_j) &= \frac{1}{2}\left(\partial_x^+\psi(x_{i-1},y_j)+\partial_x^+\psi(x_i,y_j)\right) \\ &\quad -\frac{w^-}{2}\left(\partial_x^+\psi(x_{i-2},y_j)-2\partial_x^+\psi(x_{i-1},y_j)+\partial_x^+\psi(x_i,y_j)\right) \qquad (2.42)\end{aligned}$$

where $w^\pm$ are the weights. The correction (weight) terms are coming from second derivatives. $w^\pm$ are functions of L/R derivatives, but they are very close to 1/3, when the second derivative of $\psi(x_i, y_i)$ and $\psi(x_{i\pm1}, y_j)$ are not too different. Symbolically,

$$w^\pm \approx \begin{cases} 1 & |\partial_x^2\psi(x_i,y_i)| \gg |\partial_x^2\psi(x_{i\pm1},y_i)| \\ 0 & |\partial_x^2\psi(x_i,y_i)| \ll |\partial_x^2\psi(x_{i\pm1},y_i)| \\ \frac{1}{3} & \text{otherwise}\end{cases}$$

where

$$\partial_x^2\psi(x_i,y_i) := \frac{1}{\Delta x}\left(\frac{\psi(x_{i+1},y_j)-\psi(x_i,y_j)}{\Delta x}-\frac{\psi(x_i,y_j)-\psi(x_{i-1},y_j)}{\Delta x}\right)$$

One can get the feeling that (2.41), (2.42) are probably derived from Taylor expansion too, in the same way we derived Lax-Friedrichs and Godunov, and then impose go-with-flow universality.

If we choose $w^\pm$ to be these discrete values, we reduce WENO to ENO=Essentially Non-Oscillatory scheme. For the sake of speed, we prefer ENO,

Simplify (2.41), (2.42), when $w^\pm = 1/3$,

$$\partial_x^\pm\psi(x_i,y_j) = \pm\frac{1}{2\Delta x}\left(-\frac{1}{3}\psi(x_{i\pm2},y_j)+2\psi(x_{i\pm1},y_j)-\psi(x_i,y_j)-\frac{2}{3}\psi(x_{i\mp1},y_j)\right)$$

When $w^\pm = 1$

$$\partial_x^\pm\psi(x_i,y_j) = \pm\frac{1}{2\Delta x}\left(-\psi(x_{i\pm2},y_j)+4\psi(x_{i\pm1},y_j)-3(x_i,y_j)\right)$$

When $w^\pm = 0$

$$\partial_x^+\psi(x_i,y_j) = \partial_x^-\psi(x_i,y_j) = \frac{1}{2\Delta x}\left(\psi(x_{i+1},y_j)-\psi(x_{i-1},y_j)\right)$$

2.5.2 High Order Lax-Friedrichs Hamiltonian

We see that our Lax-Friedrichs Hamiltonian (2.21) and (2.35) don't have L/R derivatives, they have central difference derivatives and neighboring points $(x_{i\pm1}, y_{j\pm1})$. Thus to cooperate WENO into Lax-Friedrichs Hamiltonian, we replace neighboring points $(x_{i\pm1}, y_{j\pm1})$, using definitions of $\partial_x^{\pm}\psi(x_i, y_j)$, thus

$$\psi(x_{i\pm1}, y_j) = \psi(x_i, y_j) \pm \Delta x \cdot \partial_x^{\pm}\psi(x_i, y_j)$$

but here $\partial_x^{\pm}\psi(x_i, y_j)$ are no longer the ordinary L/R derivatives, but given by WENO.

Simplify above, we get when $w^{\pm} = 1/3$,

$$\psi(x_{i\pm1}, y_j) \;\rightarrow\; \frac{-\frac{1}{3}\psi(x_{i\pm2}, y_j) + 2\psi(x_{i\pm1}, y_j) + \psi(x_i, y_j) - \frac{2}{3}\psi(x_{i\mp1}, y_j)}{2} \tag{2.43}$$

When $w^{\pm} = 1$

$$\psi(x_{i\pm1}, y_j) \;\rightarrow\; \frac{1}{2}\left(-\psi(x_{i\pm2}, y_j) + 4\psi(x_{i\pm1}, y_j) - \psi(x_i, y_j)\right)$$

When $w^{\pm} = 0$

$$\psi(x_{i\pm1}, y_j) \;\rightarrow\; \frac{1}{2}\left(\psi(x_{i\pm1}, y_j) + 2\psi(x_i, y_j) - \psi(x_{i\mp1}, y_j)\right)$$

To further increase speed, and since our 2D $\mathbf{B}$ fields are convex loops, we will just pretend $w^{\pm} = 1/3$. Later when we move to 3D, we will see such simplification is not so good.

Hence replace $(x_{i\pm1}, y_{j\pm1})$ in traditional Lax-Friedrichs Hamiltonian (2.21), (2.35) to above values, we get high order Lax-Friedrichs sweeping.

For example from (2.35), for $B_x > 0,\ B_y > 0$, now it is

$$\begin{aligned}
\psi^{(n+1)} = {} & \frac{1}{|B_x(x_i, y_j)| + |B_y(x_i, y_j)|}\times \\
& \left\{|B_x(x_i, y_j)|\left[\frac{1}{2}\left(-\frac{1}{3}\psi^{(n)}(x_{i-2}, y_j) + 2\psi^{(n)}(x_{i-1}, y_j) + \psi^{(n)}(x_i, y_j) - \frac{2}{3}\psi^{(n)}(x_{i+1}, y_j)\right)\right]\right. \\
& \left. + |B_y(x_i, y_j)|\left[\frac{1}{2}\left(-\frac{1}{3}\psi^{(n)}(x_i, y_{j-2}) + 2\psi^{(n)}(x_i, y_{j-1}) + \psi^{(n)}(x_i, y_j) - \frac{2}{3}\psi^{(n)}(x_i, y_{j+1})\right)\right]\right\} \\
= {} & \frac{1}{|B_x(x_i, y_j)| + |B_y(x_i, y_j)|}\times \\
& \left\{|B_x(x_i, y_j)|\left[\frac{1}{2}\left(-\frac{1}{3}\psi^{(n)}(x_{i-2}, y_j) + 2\psi^{(n)}(x_{i-1}, y_j) - \frac{2}{3}\psi^{(n)}(x_{i+1}, y_j)\right)\right]\right. \\
& \left. + |B_y(x_i, y_j)|\left[\frac{1}{2}\left(-\frac{1}{3}\psi^{(n)}(x_i, y_{j-2}) + 2\psi^{(n)}(x_i, y_{j-1}) - \frac{2}{3}\psi^{(n)}(x_i, y_{j+1})\right)\right]\right\} + \frac{1}{2}\psi^{(n)}(x_i, y_j)
\end{aligned}$$

Although it seems like the elegant data flow direction is gone, we know from (2.43) the replacement is approximated. We think although the data flow is no longer conservative, but it is an approximated conservative in some weak sense. Therefore we believe high order Lax-Friedrichs doesn't have strict convergence, but it is still reasonably convergent.

2.5.3 High Order Lax-Friedrichs Algorithm

We already mentioned high order Lax-Friedrichs algorithm. First use traditional-no boundary- to get a rough shape then use high order Lax-Friedrichs to get refined shape.

We apply high order Lax-Friedrichs to figures 7 on page 33

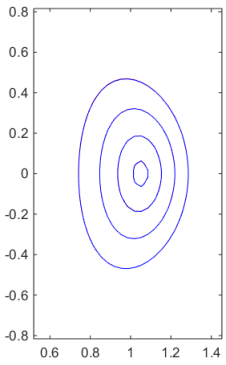

figure 13(a). Same setup in figure 7 on page 33, 29×50 grid points, $\Delta x = \Delta y$, 100 iterations (half traditional; half high order), running time 1.4 seconds, the average angular error (cf (2.28)) is 0.0637 degrees

We see blue lines-LF solution and red lines-exact solution coincided perfectly. If we blow the figure above, we see some separations.

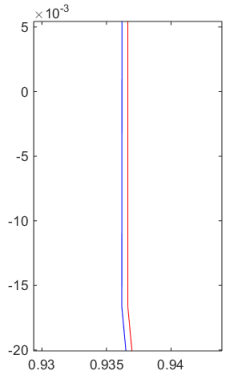

figure 13(b). Blow the figure 13(a) in the region roughly $[0.93, 0.94] \times [-0.02, +0.005]$

If we want to compare our high order Lax-Friedrichs method with field line tracing method. We can blow field line tracing picture too, e.g. figure 3 on page 19, and see

how many orders of magnitude we have to blow to see separation. However there is a smarter way to get a quantitative comparison of the two methods, we define another error measurement L^2. Mathematically pick a point in the domain, draw the contour line of exact solution (red line) and contour line of LF solution (blue line) cross that point,

$$\text{error\_L\^2} = \frac{\int |\text{red} - \text{blue}|^2}{\int |\text{red}|^2} \tag{2.44}$$

the "area" difference between blue and red lines. Our experiments show at 50 grids, field line tracing method has typical error field line tracing error $\sim 10^{-6}$; our method $\sim 10^{-3}$.

Now we apply high order Lax-Friedrichs to 11(b) on page 39.

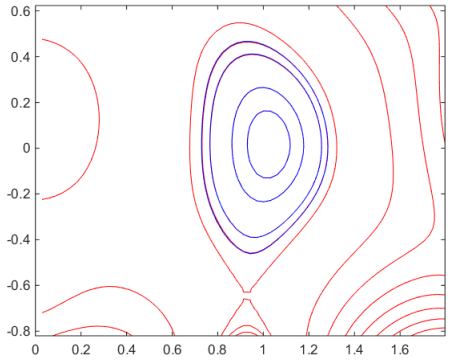

figure 14. Same setup in figure 11(b) on page 39
50 grid points in y, $\Delta x = \Delta y$, 100 iterations, running time 3.04 seconds,
the average angular error (cf (2.28)) is 0.4046 degrees

This is far better than figure 12 on page 40. Not to mention the number of grid points is very small.

2.5.4 Convergence and Stability Tests

Before we do convergence tests, i.e. error wrt grid spacing, let's summarize the properties of three different errors we have defined earlier: (2.26), (2.28), & (2.44).

- (2.26) and (2.28) have similar meaning and similar value. As explained early if (2.26) and (2.28) go down as iteration increases or grid spacing decreases, we are on the path of convergence.

- (2.44) is used to compare our method and field line tracing method. In 2D this error measurement gives the area between computational contour line and the exact solution contour line; in 3D this error measurement gives the volume between the computational flux and exact flux. Unlike the other two error definitions, it requires the knowledge of the exact solution ψ_{exact}. In the case we don't know the exact

solution, we can pretend the field line tracing solution is the exact solution and still are able to compute (2.44), the area difference between our method and field line tracing. One advantage of this error measurement is that it is exact what we want to get from Lax-Friedrichs, the shape of **B** flux; whileas the other two error measurements (2.26) and (2.28) give angles.

There is a subtlety about (2.44) that the contour lines obtained from our method ψ_{LF} and the contour lines obtained from discretized $\psi_{exact\ discretized}$ have been processed by some kind of interpolations. That will introduce additional errors not controlled by our method. However if we know $\psi_{exact\ continuous}$, we don't have to use the contour lines obtained from discretized $\psi_{exact\ discretized}$ and obtain exact contour lines, cf discussion above figure 3 on page 19.

Our experiment shows

case 1

If we compare ψ_{LF} vs. $\psi_{exact\ discretized}$ (cf figure 13(a) on page 43) and field line tracing v.s. $\psi_{exact\ continuous}$ (cf figure 3 on page 19), we get both have same ~10^{-6} error.

case 2

If we compare ψ_{LF} vs. $\psi_{exact\ continuous}$ (cf figure 15 below) and field line tracing v.s. $\psi_{exact\ continuous}$ (cf figure 3 on page 19), we get ψ_{LF} has $\sim 10^{-3}$ error.

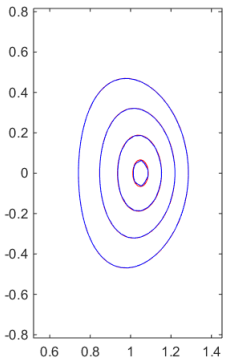

figure 15. high order Lax-Friedrichs
blue lines (contour of ψ_{LF}) same as on figure 13(a) on page 43
red lines (contour of $\psi_{exact\ continuous}$) same as solid lines on figure 3 on page 19

We see from the above figure that most error of 10^{-3} is from region near the center where the number of grid points is very small, so the contour of discrete ψ_{LF} has most error due to poor interpolation, whileas contour of $\psi_{exact\ continuous}$ has no error. Therefore whether our method is as accurate as method of field line tracing depends on what error measurement we choose: case 1 or case 2.

Because of this subtlety, for the convergence test we stay away from error_L^2 measurement (2.44). Let us vary number of iterations and vary grid points (with $\Delta x = \Delta y$), then compute angular error (2.28). Starting from the smallest grids possible, we find that 15×25 with 20 iterates are minimal to form good looking contours.

		half iter trad, half high order		no high order	
grids	iterates	ang error (deg)	run time (s)	ang error (deg)	run time (s)
15×25	20	0.8915	0.0798	14.0113	0.0325
	30	*0.5121*	0.1051	13.9774	0.0533
	40	0.5265	0.1325	13.9771	0.0458
	50	0.5511	0.1596	13.9771	0.0528
	100	0.6047	0.2957	13.9771	0.0862

		half iter trad, half high order		no high order	
grids	iterates	ang error (deg)	run time (s)	ang error (deg)	run time (s)
29×50	30	1.1305	0.4197	5.8149	0.1055
	40	0.3294	0.5500	5.8068	0.1337
	80	0.0766	1.0928	5.8113	0.2473
	200	0.0408	2.6724	5.8114	0.6071
	400	*0.0402*	5.3741	5.8114	1.1669
	600	0.0451	7.9758	5.8114	1.7323
	800	0.0479	10.6791	5.8114	2.3692

		half iter trad, half high order		no high order	
grids	iterates	ang error (deg)	run time (s)	ang error (deg)	run time (s)
44×75	70	1.1075	2.4001	4.0442	0.5001
	150	0.0414	4.9478	3.8703	1.0295
	300	0.0339	10.2654	3.8722	2.2675
	600	0.0244	19.5979	3.8722	4.0248
	1000	0.0153	32.9187	3.8722	6.6933
	2000	*0.0142*	65.7846	3.8722	13.4334

Observations:

1. We check the error v.s. grid spacing, hence our method is 3th order accurate.

$$\left(\frac{75}{25}\right)^{3.3} \approx \frac{0.5121}{0.0142} \approx 36 \qquad \left(\frac{75}{50}\right)^{2.5} \approx \frac{0.0402}{0.0142} \approx 2.83$$

2. We verify that indeed traditional Lax-Friedrichs has nice convergence property. This limiting value is independent of arbitrary doping values, e.g. we increase doping

values by 1000 to avoid that ψ_{LF} decays to too close to 0 after too many iterations so that plotting contour of ψ_{LF} may have troubles, but that doesn't change the error limit.

3. We see indeed high order Lax-Friedrichs may not converge precisely and the tail may grow, but not too much, because we explained that is because path of data flow in high order Lax-Friedrichs is almost conservative.

2.6 3D Ideal MHD Example

Armed with good knowledge of 2D ideal example, we move to 3D case. (1.4), (1.5) & (1.6). We mentioned before we should use cylindrical coordinates (ρ, z, ϕ), so that in ϕ direction, we only have to sweep in one direction that saves half of running time. In the (ρ, z) cross section we use high order Lax-Friedrichs with no boundary. In the ϕ direction we use periodic boundary, i.e. we glue $\phi = 0$ and $\phi = 2\pi$, this amounts to take care of 2 things:

i) at the beginning when we put arbitrary initial doping, we make sure $\psi^{(0)}(\rho, z, 0) = \psi^{(0)}(\rho, z, 2\pi)$;

ii) in the sweeping process, we make sure when we're getting closer to the two ends the neighboring points locations are not out of bound. For example suppose there are l grids in $\phi \in [0, 2\pi)$, and k labels the discretized location, $k = 1, ..., l$. Suppose we are at $\psi(\cdot, \cdot, \phi_k)$, high order Lax-Friedrichs requires to get to its neighbors, 2 to the right and 2 to the left. To make sure they are not out of bound, we can replace $k + 2$ by

$$\text{mod } (k + 2 - 1, l - 1) + 1$$

similarly replacing $k + 1 \to \text{mod}(k + 1 - 1, l - 1) + 1$, $k - 1 \to \text{mod}(k - 1 - 1, l - 1) + 1$, $k - 2 \to \text{mod}(k - 2 - 1, l - 1) + 1$.

There are three more points we'd like to make

1. Now ∇ in the Hamiltonian is in cylindrical, the partial derivative in ϕ direction is not same as in Cartesian. It will change a few things:

$$\nabla \to \frac{\partial}{\partial \rho}, \frac{\partial}{\partial z}, \frac{1}{\rho}\frac{\partial}{\partial \phi}$$

This additional $1/\rho$ in our original pde (2.1) can be interpreted as changing

$$B_\phi \to B_\phi/\rho \tag{2.45}$$

and what about our Lax-Friedrichs Hamiltonian (2.21), now

$$\sigma_\phi = \left| \frac{\partial H}{\partial(\partial\psi/\partial\phi)} \right| = |B_\phi/\rho|$$

Therefore if we just do substitution (2.45), everything will be exactly the same as in Cartesian, except when we compute angular error, convert $\mathbf{B}$ back, because the denominator $\|\mathbf{B}\|$ in (2.25) and (2.27) is still the old $\mathbf{B}$.

2. Running Time Issue. This is one insurmountable issue: we said in 2D magnetic loops are convex so there are little kinks, we will assume $w^\pm$ weight factor in high order derivative to be 1/3. In 3D flux tubes are not convex, there are inflection points. Our experiment shows we have to take $w^\pm$ to be 1/3, 1 or 0. This increases computational time dramatically. 20 mins to run figure 16(b) is not acceptable.

3. Stability Issue. (2.45) suggests another idea. Because our domain in cylindrical coordinate is $[0.6, 1.4] \times [-0.6, 0.6] \times [0, 2\pi]$, if we let $\Delta\rho = \Delta z = \Delta\phi$, there are 5 times more grid points in ϕ than in ρ and z. If we can partition the domain different, say

$$5\Delta\rho = 5\Delta z = \Delta\phi$$

then we can save 5 times of running time, because there are 5 times less cross sections than before, which is indeed what we did in figure 16(b).

Question: Is it true if we keep $\Delta\phi$ larger, then we can save more time? Obviously it is not possible, it is controlled by Courant stability condition. By Courant stability condition it is not possible to keep making $\Delta\phi$ small either. So what is this alluded to? Look at (2.35), we see that making $\Delta\phi$ 10 times bigger is compensated by making $B_\phi \to 10B_\phi$ in order to keep (2.35) the same. But we discussed in the beginning of the thesis large B_ϕ is an unstable configuration, because flux tubes will not be formed easily just like in uniform magnetic fields. Thus our method has the ability to link computational stability to the physical realistic configuration stability.

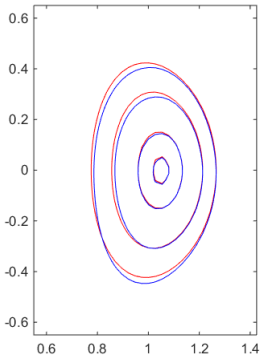

figure 16(a). traditional Lax-Friedrichs
$34 \times 50 \times 48$ grid points, 40 iterations, running time 30.9 seconds

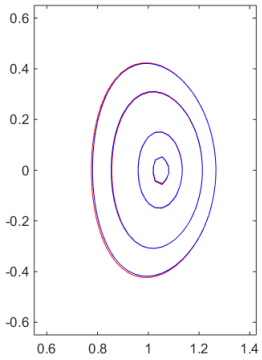

figure 16(b). high order Lax-Friedrichs
$34 \times 50 \times 48$ grid points, 80 iterations (half traditional; half high order),
running time 1194.1 seconds (20mins)!

3 Conclusion

3.1 Future Work: 3D Islands Example

At the moment there is another insurmountable issue

- Because each island has no imported boundary, wrong data outside of island or from different islands will influence it if it is not shielded very well, our experiment of no boundary Lax-Friedrichs shows to have each island act independently, i.e. shield each island and depict them clearly, require at least 20 grid points in one direction. Therefore for the 3D island example 5 times more grid points in ρ and z than in figure 16(b), thus the estimated time will be 20 min $\times 5^3 = 40$ hours. So we don't even bother to try.

Two possible solutions we have proposed: 1) use Fortran and use parallel computing, i.e. do 2D Lax-Friedrichs on each cross section parallelly, 2) look for even high order derivatives, because we see from the table in the previous pages at grids size 15×25, 3rd order ENO gives far better results than traditional Lax-Friedrichs. So we think we may want to try 5th order WENO.

3.2 Conclusion

There are some original work in this thesis:

- We gave a coherent derivation of Lax-Friedrichs formula, Godunov formula, and high order WENO formula. We recognized the usage of Taylor expansion and a universal appealing assumptions (2.10). We believe these are the reasons why Lax-Friedrichs formula, Godunov formula, and high order WENO are fell into the category of conservative method or almost conservative method (as we explained in the paper and showed numerically). Therefore this suggests in the future if we are going to find an analytic proof of the extension of Lax-Friedrichs method, we'd better define new mathematical "flow" of pde–the flow of internal data propagation, and when it's applied to our problem is the $\mathbf{B}$ field.

  The finding of higher principle of Lax-Friedrichs and Godunov is what we considered a success, besides we found a new method to simulate magnetic flux surfaces,

the original goal of the project. Because of this universal appealing assumptions (2.10), Lax-Friedrichs method may be applicable in many other physical models: Poincare flow, Ricci flow, renormalization flow...

- Our high order (3rd order accurate) Lax-Friedrichs method has maximal connection to physical systems: i) the number of iterations manifests causality; ii) the conservative data flow structure diagnoses the closeness of confinement surfaces; and iii) Courant stability links to stability of magnetic flux configurations.

- We modernized traditional Lax-Friedrichs method so that there is no fixed boundary (by the boundary value problem) nor structured initial values (by the initial value problem), and there is no imposed monotonicity. Our 2D high order Lax-Friedrichs code looks very simple too. The core Hamiltonian code consists of 7 lines and total is about 150 lines.

Bibliography

[1] LeVeque. Python tools for reproducible research on hyperbolic problems. Computing in Science and Engineering, 11:19–27, 2009. doi: 10.1109/MCSE.2009.13.

[2] Klockner. Doctoral Dissertation: High-performance high-order simulation of wave and plasma phenomena. ISBN: 978-1-124-30426-7

[3] Courant, Friedrichs, Lewy. On the Partial Difference Equations of Mathematical Physics. Translation and republish in IBM Journal of Research and Development archive Volume 11 Issue 2, March 1967 Pages 215-234 doi: 10.1147/rd.112.0215

[4] Press, Flannery, Teukolsky. Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press; 2 edition (October 30, 1992) ISBN-10: 0521431085

[5] Press, Flannery, Teukolsky. Numerical Recipes in Fortran 90: The Art of Parallel Scientific Computing. Cambridge University Press; 2 edition (October 30, 1992) ISBN-10: 052143064X

[6] LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems. SIAM (2007). ISBN-10: 0898716292

[7] Elman, Silvester, Wathen. Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics. Oxford University Press (2005) ISBN-10: 0198528671

[8] Lax. Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves. CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, Philadelphia, 11, 1973.

[9] Freidberg, Plasma Physics and Fusion Energy. Cambridge University Press; 1 edition (2008) ISBN-10: 0521733170

[10] Meiss. Transient measures in the standard map. phys D vol 74, 3–4, 1994, pp 254–267. doi:10.1016/0167-2789(94)90197-X

[11] Berrut, Trefethen. Barycentric Lagrange Interpolation. SIAM review vol. 46, no. 3, pp. 501–517.

[12] Hudson, Nakajima. Pressure, chaotic magnetic fields, and magnetohydrodynamic equilibria. Physics of Plasmas 2010; 17(5):052511-052511-10. DOI: 10.1063/1.3431090

[13] Reiman, Greenside. Calculation of three-dimensional MHD equilibria with islands and stochastic regions. Comp Phys Comm vol 43, no 1, 1986, pp 157–167. doi:10.1016/0010-4655(86)90059-7

[14] Guang-shan Jiang, Danping Peng. Weighted ENO Schemes for Hamilton-Jacobi Equations. SIAM J. Sci. Comput vol 2 No 6 pp 2126-2143. doi: 10.1137/S106482759732455X

[15] Osher. A level set formulation for the solution of the Dirichlet problem for Hamilton-Jacobi equations. SIAM J. Math. Anal., 24(5), 1145–1152. doi: 10.1137/0524066

[16] Bardi, Osher. The Nonconvex Multidimensional Riemann Problem for Hamilton–Jacobi Equations. SIAM J. Math. Anal., 22(2), 344–351. DOI:10.1137/0522022

[17] Evans, Partial Differential Equations. American Mathematical Society; 2 edition (March 3, 2010) ISBN-10: 0821849743

[18] Cerfon, Freidberg. "One size fits all" analytic solutions to the Grad–Shafranov equation. Phys. Plasmas 17, 032502 (2010)

[19] Crandall, Lions. Viscosity solutions of Hamilton-Jacobi equations. Trans. Amer. Math. Soc. 277 (1983), 1-42

[20] Kao, Osher, Qian. Lax–Friedrichs sweeping scheme for static Hamilton–Jacobi equations. J. Sci. Comput vol 196, no 1, pp 367–391. doi:10.1016/j.jcp.2003.11.007

[21] Zhang, Zhao, Qian. High Order Fast Sweeping Methods for Static Hamilton–Jacobi Equations. Journal of Scientific Computing (2006), vol 29, no 1, pp 25-56.

[22] Tsai, Cheng, Osher, Zhao. Fast Sweeping Algorithms for a Class of Hamilton–Jacobi Equations. SIAM J. Numer. Anal., 41(2), 673–694. DOI:10.1137/S0036142901396533

[23] Sethian. A fast marching level set method for monotonically advancing fronts. Proceeding of national academy of sciences vol. 93 no. 4 pp 1591–1595.