Hermite advection schemes

Michel Mehrenberger

I2M, Université Aix-Marseille, France

Mini-Symposium Congrès SMAI 2021

Current works with Nicolas Crouseilles (INRIA Rennes), Laurent Navoret and Ali Elarif (IRMA Strasbourg), Emily Bourne (CEA Cadarache & I2M Marseille), Mamoun Sadaka (ENSEIRB-Matmeca Bordeaux & I2M Marseille)

- We aim at solving **transport** equations

$$\partial_t f(t,z) + u(z) \cdot \nabla_z f(t,z) = 0$$

- In 1D with constant advection field u, we have the relation

$$f(t+\Delta t, z) = f(t, z - u\Delta t)$$

- This is exploited in semi-Lagrangian schemes:
  - ODE solver for characteristics
  - interpolation
- Here we focus on **Hermite interpolation**, which reads in 1D:

Proposition

- *With $f(a), f(b), f'(a), f'(b)$, $a < b$, we can define* **explicitly** *a unique polynomial P of degree ≤ 3 satisfying the interpolation conditions*

$$P(a) = f(a),\ P(b) = f(b),\ P'(a) = f'(a),\ P'(b) = f'(b).$$

- $P(a+x(b-a)) = f(a)H_0(x) + f(b)H_1(x) + (b-a)f'(a)K_0(x) + (b-a)f'(b)K_1(x)$

$$\begin{array}{ll} H_0(x) = (1-x)^2(1+2x) & K_0(x) = (1-x)^2 x \\ H_1(x) = x^2(3-2x) & K_1(x) = x^2(x-1) \end{array}$$

Outline

- Related works
- high order Hermite interpolation on uniform grid
- Hermite interpolation on cartesian grids
- Hermite interpolation on triangular mesh
- Conclusion and future works

Some known methods in the context of plasma physics

- **Propagation of gradients**

  C*ubic* **I***nterpolation* **P***ropagation* `NakamuraYabe1999`

  Propagation of $f, \partial_x f$ and $\partial_v f$

  Example for $\partial_t f + v\partial_x f(t,x,v) = 0$ step:

  1. Hermite interpolation for f and $\partial_x f$ at foot of characteristic $x_i - v_j \Delta t = x_{i_0} + \alpha h$

  $$\begin{aligned} f_{ij}^{n+1} &= f_{i_0 j}^n H_0(\alpha) + f_{i_0+1 j}^n H_1(\alpha) + h\partial_x f_{i_0 j}^n K_0(\alpha) + h\partial_x f_{i_0+1 j}^n K_1(\alpha) \\ h\partial_x f_{ij}^{n+1} &= f_{i_0 j}^n H_0'(\alpha) + f_{i_0+1 j}^n H_1'(\alpha) + h\partial_x f_{i_0 j}^n K_0'(\alpha) + h\partial_x f_{i_0+1 j}^n K_1'(\alpha) \end{aligned}$$

  2. for $\partial_v f$, use of Finite Difference on $\partial_t \partial_v f = -\partial_x(v\partial_v f(t,x,v))$

- **Reconstruction of gradients**

  Finite Difference Hermite `FilbetSonnendrücker2003`

  The gradients are reconstructed to save memory

- WENO of degree 5: `CaiQiuQiu2016` `YangFilbet2014`

  $$\text{use of } f_{i_0-1},\ f_{i_0},\ f_{i_0+1},\ f_{i_0+2} \text{ and } f'_{i_0-1},\ f'_{i_0+2}$$

- Hermite on triangular mesh `BesseSonnendrücker2003`
- Analysis of convergence `Besse2008`
- Recent work for guiding center model (⇔ 2D incompressible Euler equation) `YinMercierYadavSchneiderNave2021` following in particular `SeiboldNaveRosales2012`

- High order Lagrange interpolation is very efficient on uniform grid
  $P_{Lag}(x_{i_0} + \alpha h) = \sum_{\ell=-d}^{d+1} f(x_{i_0}) L_\ell(\alpha),\ L_\ell(\alpha) = \prod_{k=-d,\ k\neq\ell}^{d+1} \frac{\alpha-k}{\ell-k}$
  possible coupling with high order time splitting `CasasCrouseillesFaouM2017`

⇒ difficult to beat for classical Vlasov-Poisson system

$$\left\{ \begin{array}{l} \partial_t f + v \cdot \nabla_x f + E \cdot \nabla_v f = 0 \\ -\Delta_x \phi = \int f dv,\ E = -\nabla_x \Phi. \end{array} \right.$$

- However, it becomes costly for multi-D interpolation

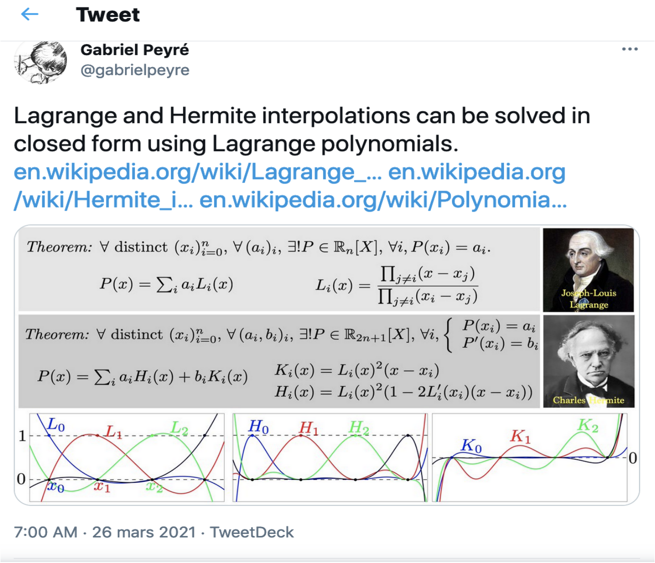
⇒ Hermite interpolation can be a solution

- We have already worked in the cubic case

  `HamiazMSellamaSonnendrücker2016` (curvilinear 2D interpolation)
  `CrouseillesGlancHirstoagaMadauleMPetri2014` (conservative 2D interpolation)
  We approach to the results of the high order Lagrange interpolation
  keeping cubic polynomials, as cubic splines, but more local
  as for cubic splines, a preparation step:
  reconstruction fo the derivatives ↔ spline coefficients computations

- We explore here the case of higher order Hermite interpolation

The beginning



Tweet

Gabriel Peyré
@gabrielpeyre

Lagrange and Hermite interpolations can be solved in closed form using Lagrange polynomials. en.wikipedia.org/wiki/Lagrange_... en.wikipedia.org/wiki/Hermite_i... en.wikipedia.org/wiki/Polynomia...

Theorem: ∀ distinct $(x_i)_{i=0}^n$, $\forall\,(a_i)_i$, $\exists! P \in \mathbb{R}_n[X]$, $\forall i, P(x_i) = a_i$.

$$P(x) = \sum_i a_i L_i(x) \qquad L_i(x) = \frac{\prod_{j\neq i}(x - x_j)}{\prod_{j\neq i}(x_i - x_j)}$$

Theorem: ∀ distinct $(x_i)_{i=0}^n$, $\forall\,(a_i, b_i)_i$, $\exists! P \in \mathbb{R}_{2n+1}[X]$, $\forall i, \begin{cases} P(x_i) = a_i \\ P'(x_i) = b_i \end{cases}$

$$P(x) = \sum_i a_i H_i(x) + b_i K_i(x) \qquad \begin{array}{l} K_i(x) = L_i(x)^2(x - x_i) \\ H_i(x) = L_i(x)^2(1 - 2L_i'(x_i)(x - x_i)) \end{array}$$

7:00 AM · 26 mars 2021 · TweetDeck

- In 1D, we choose to use the values

$$f_{i_0-d}, f'_{i_0-d}, \dots, f_{i_0+d+1}, f'_{i_0+d+1}$$

- for $d = 0$, it is the classical cubic case: $f_{i_0}, f'_{i_0}, f_{i_0+1}, f'_{i_0+1}$
- for $d = 1$, we go to degree 7 and so on...
- Formulae are completely explicit:

$$P(x_{i_0} + \alpha h) = \sum_{\ell=-d}^{d+1} f_{i_0+\ell} H_\ell(\alpha) + \sum_{\ell=-d}^{0} f'_{i_0+\ell+} K_\ell(\alpha) + \sum_{\ell=1}^{d+1} f'_{i_0+\ell-} K_\ell(\alpha)$$

$$K_\ell(\alpha) = L_\ell(\alpha)^2(\alpha - \ell) \quad H_\ell(\alpha) = L_\ell(\alpha)^2(1 - 2L'_\ell(\ell)(\alpha - \ell))$$

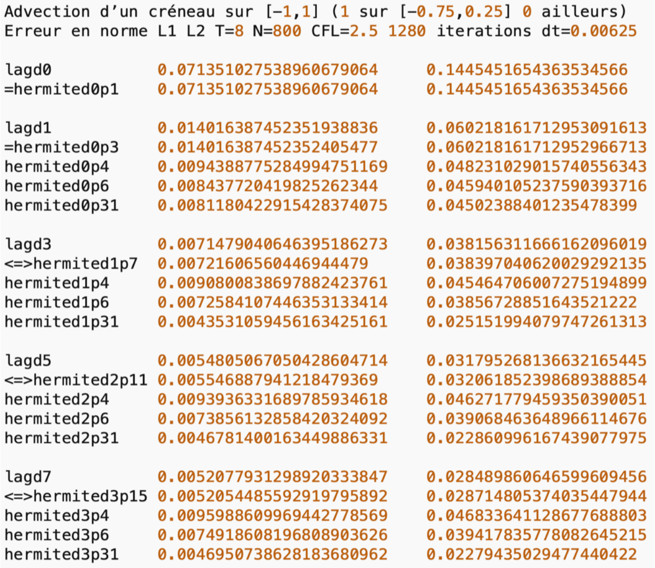
with $L'_\ell(\ell) = \sum_{j=-d,\ j\neq\ell}^{d+1} \frac{1}{\ell - j}$

- Derivatives are reconstructed in a FD fashion of order p

$p = 1$, $f'_{\ell+} = f_{\ell+1} - f_\ell,\ f'_{\ell-} = f_\ell - f_{\ell-1}$

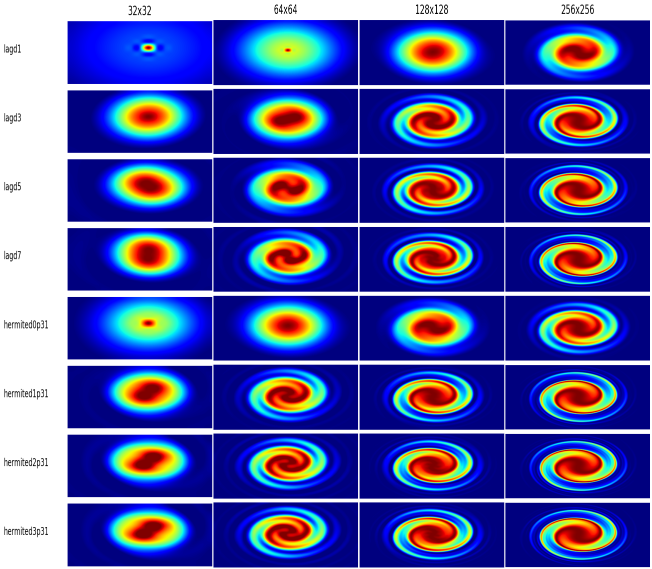
$p = 2$, $f'_{\ell+} = f'_{\ell-} = \frac{f_{\ell+1} - f_{\ell-1}}{2}$ and so on...

Numerical results in 1D



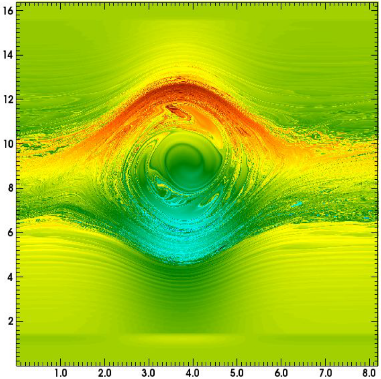
```
Advection d'un créneau sur [-1,1] (1 sur [-0.75,0.25] 0 ailleurs)
Erreur en norme L1 L2 T=8 N=800 CFL=2.5 1280 iterations dt=0.00625

lagd0           0.0713510275389606790 64   0.14454516543635345 66
=hermited0p1    0.0713510275389606790 64   0.14454516543635345 66

lagd1           0.014016387452351938836    0.060218161712953091613
=hermited0p3    0.014016387452352405477    0.060218161712952966713
hermited0p4     0.0094388775284994751169   0.048231029015740556343
hermited0p6     0.008437720419825262344    0.045940105237590393716
hermited0p31    0.0081180422915428374075   0.04502388401235478399

lagd3           0.0071479040646395186273   0.038156311666162096019
<=>hermited1p7  0.007216065604469444 79    0.038397040620029292135
hermited1p4     0.0090800838697882423761   0.045464706007275194899
hermited1p6     0.0072584107446353133414   0.03856728851643521222
hermited1p31    0.0043531059456163425161   0.025151994079747261313

lagd5           0.0054805067050428604714   0.031795268136632165445
<=>hermited2p11 0.005546887941218479369    0.032061852398689388854
hermited2p4     0.0093936331689785934618   0.046271779459350390051
hermited2p6     0.0073856132858420324092   0.039068463648966114676
hermited2p31    0.0046781400163449886331   0.022860996167439077975

lagd7           0.0052077931298920333847   0.028489860646599609456
<=>hermited3p15 0.0052054485592919795892   0.028714805374035447944
hermited3p4     0.0095988609969442778569   0.046833641128677688803
hermited3p6     0.0074918608196808903626   0.039417835778082645215
hermited3p31    0.0046950738628183680962   0.02279435029477440422
```

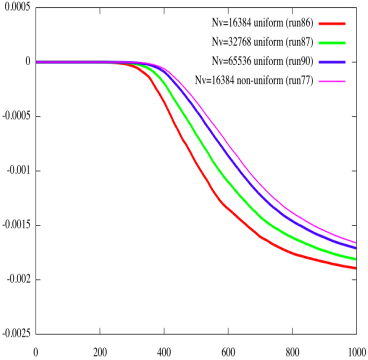
Numerical results for VP 1d1v



- We observe better accuracy than Lagrange
- ... using a larger stencil...
- derivatives can be computed once and reused for different stencils
- accuracy is improved for $d = 1 \Rightarrow$ encouraging results
- we hope to have a gain in a 2D setting (not implemented yet)

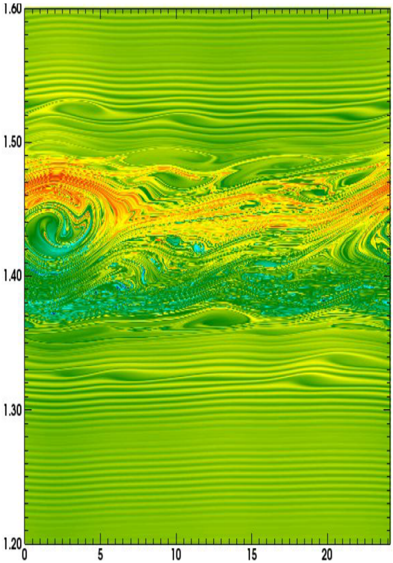
- Previous strategy can lead to stability issues on more complex problems
  - ⇒ in particular, if the grid is **non-uniform**, which can be needed for some problems localized at some regions
- One solution is to use spline interpolation `AfeyanCasasCrouseillesDodhyFaouMSonnendrucker2014`
  - ⇒ smoothing effects seem to be beneficial for stability

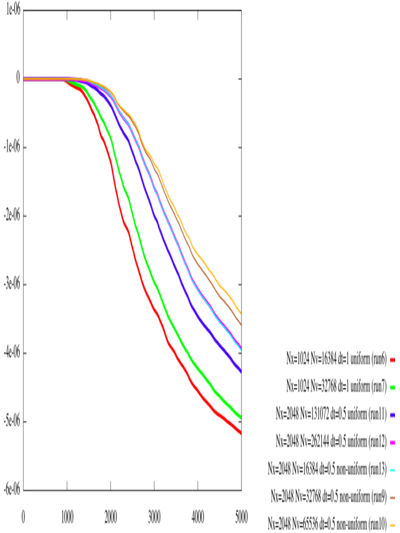




- We will explore here another more local way: the propagation of the gradients
  - ⇒ Similar to CIP method, but we also would like to remove the FD part
- Another possibility would be to use SLDG method
  - ⇒ non uniform & 2D version are however quite complex to implement

Gain of factor $\frac{262144}{16384} = 16$ with on uniform cubic splines in velociity





Looking for stability

- Some results exist for non constant advection `Ferretti2013`

  Convergence with odd degree Lagrange and splines interpolation

  SL scheme interpreted as a Lagrange-Galerkin scheme `FerrettiM2020`

  Even degree interpolation only stable for constant advection `BesseM2008` `Despres2008` `CharlesDespresM2012`
- What about non uniform grids? can we explain the good behavior of cubic splines?

  results on non constant advection can be translated to non-uniform mesh, but only for smooth mapping

A stability property in 1D

Proposition

With $f^n : [0, L]_{per} \to \mathbb{R}$ piecewise cubic solution obtained from splines or Hermite interpolation, we have

$$\int_0^L |(f'')^{n+1}(x)|^2 dx \leq \int_0^L |(f'')^n(x)|^2 dx$$

- can be generalized to higher order
  - for splines: `CarlDeBoor1963`
  - for Hermite: `GoodrichHagstromLorenz2006`
- valid on **non uniform mesh**

Proof in the Hermite case

$f^n \in H^2_{per}(0, L)$, since $f^n \in C^1_{per}([0, L])$
We consider Hermite interpolation and start from

$$\int_0^L ((f^{n+1})''(x) - (f^n)''(x - v\Delta t))(f^{n+1})''(x)dx$$
$$= \sum_{i=0}^{N-1} \int_{x_i}^{x_{i+1}} ((f^{n+1})''(x) - (f^n)''(x - v\Delta t))(f^{n+1})''(x)dx,$$

and integrate by parts

$$\int_{x_i}^{x_{i+1}} ((f^{n+1})''(x) - (f^n)''(x - v\Delta t))(f^{n+1})''(x)dx$$
$$= -\int_{x_i}^{x_{i+1}} ((f^{n+1})'(x) - (f^n)'(x - v\Delta t))(f^{n+1})'''(x)dx$$
$$= \int_{x_i}^{x_{i+1}} ((f^{n+1}(x) - (f^n)(x - v\Delta t))(f^{n+1})''''(x)dx = 0,$$

since $f^{n+1}(x_i) = (f^n)(x_i - v\Delta t)$ and $(f^{n+1})'(x_i) = (f^n)'(x_i - v\Delta t)$, by definition of the semi-Lagrangian scheme, and since $(f^{n+1})''''(x) = 0$ on (x_i, x_{i+1}), as on that interval f^{n+1} is a polynomial of degree ≤ 3 and thus the 4th derivative is zero.

Proof in the Hermite case

We then obtain from the previous orthogonality equality

$$\int_{x_i}^{x_{i+1}} |(f^{n+1})''(x) - (f^n)''(x - v\Delta t)|^2 dx + \int_{x_i}^{x_{i+1}} |(f^{n+1})''(x)|^2 dx = \int_{x_i}^{x_{i+1}} |(f^n)''(x - v\Delta t)|^2 dx,$$

and thus

$$\int_{x_i}^{x_{i+1}} |(f^{n+1})''(x)|^2 dx \leq \int_{x_i}^{x_{i+1}} |(f^n)''(x - v\Delta t)|^2 dx,$$

then

$$\int_0^L |(f^{n+1})''(x)|^2 dx \leq \int_0^L |(f^n)''(x - v\Delta t)|^2 dx = \int_0^L |(f^n)''(x)|^2 dx,$$

since f^n is L-periodic.

Proof in the splines case

$$\int_{x_i}^{x_{i+1}} ((f^{n+1})''(x) - (f^n)''(x - v\Delta t))(f^{n+1})''(x)dx$$

$$= \Big[(f^{n+1})'(x) - (f^n)'(x - v\Delta t))(f^{n+1})''(x)\Big]_{x=x_i}^{x=x_{i+1}}$$

$$- \int_{x_i}^{x_{i+1}} ((f^{n+1})'(x) - (f^n)'(x - v\Delta t))(f^{n+1})'''(x)dx$$

$$= \Big[(f^{n+1})'(x) - (f^n)'(x - v\Delta t))(f^{n+1})''(x)\Big]_{x=x_i}^{x=x_{i+1}}$$

using here only $f^{n+1}(x_i) = (f^n)(x_i - v\Delta t)$ and $(f^{n+1})''''(x) = 0$ on (x_i, x_{i+1}).
As $f^{n+1} \in C^2_{per}([0, L])$, we have

$$\sum_{i=0}^{N-1} \Big[(f^{n+1})'(x) - (f^n)'(x - v\Delta t))(f^{n+1})''(x)\Big]_{x=x_i}^{x=x_{i+1}} = 0$$

The rest of the proof is the same

A new local splines method

- Hermite advection with propagation is local but costly in memory
- Spline interpolation is not local but less costly in memory
- On uniform mesh, we could use Hermite with reconstruction of derivatives
- But on non uniform mesh, we can have to face with stability issues
- One solution is to use local splines `CrouseillesLatuSonnendrücker2006`
  - we use a domain decomposition
  - derivative information is reconstructed using neighboring points
  - on each subdomain, splines are reconstructed from Hermite boundary data
- we propose a modification:
  - Hermite boundary data are not reconstructed but propagated

  ⇒ no need to use large stencil to get stable derivative reconstruction of the initial local splines method

Proposition

With $f^n : [0, L]_{per} \to \mathbb{R}$ piecewise cubic solution obtained from new local splines method, we have

$$\int_0^L |(f'')^{n+1}(x)|^2 dx \leq \int_0^L |(f'')^n(x)|^2 dx$$

- ▶ The proof follows the proof given for Hermite and splines
- ⇒ It gives a theoretical legitimity of the scheme
- ▶ Another method is to use spline interpolation with Greville points `GucluZoni2019`, `Bourne2020` (presentation at NumKin conference)
- ⇒ However stability not available there

Advection in 2D

- We consider a $2D$ patch $[x_I, x_J] \times [y_K, y_L]$
- Unknowns are:

  point values: $f_{ij}^n \simeq f(t_n, x_i, y_j)$ for $i \in \{I, \dots, J\},\ j \in \{K, \dots, L\}$

  x-derivatives: $\partial_x f_{ij}^n \simeq \partial_x f(t_n, x_i, y_j)$ for $i \in \{I, J\},\ j \in \{K, \dots, L\}$

  y-derivatives: $\partial_y f_{ij}^n \simeq \partial_y f(t_n, x_i, y_j)$ for $i \in \{I, \dots, J\},\ j \in \{K, L\}$

  xy-derivatives: $\partial_{xy}^2 f_{ij}^n \simeq \partial_{xy} f(t_n, x_i, y_j)$ for $i \in \{I, J\},\ j \in \{K, L\}$
- It is enough to get a Hermite representation on each cell
- We use $f(t_{n+1}, x_i, y_j) = f(t_n, X = X(t_n; t_{n+1}, x_i, y_j), Y = Y(t_n; t_{n+1}, x_i, y_j))$
- $\partial_x f(t_{n+1}, x_i, y_j) = \partial_x X \partial_x f(t_n, X, Y) + \partial_x Y \partial_y f(t_n, X, Y)$
- $\partial_y f(t_{n+1}, x_i, y_j) = \partial_y X \partial_x f(t_n, X, Y) + \partial_y Y \partial_y f(t_n, X, Y)$
- mixed derivative gives more terms; for rotation case, we have not all the terms:

$$\partial_{xy}^2 f(t_{n+1}, x_i, y_j) = \partial_y X \partial_x X \partial_x^2 f(t_n, X, Y) + \partial_y X \partial_x Y \partial_{xy}^2 f(t_n, X, Y) \\ + \partial_y Y \partial_x X \partial_{xy}^2 f(t_n, X, Y) + \partial_y Y \partial_x Y \partial_y^2 f(t_n, X, Y)$$

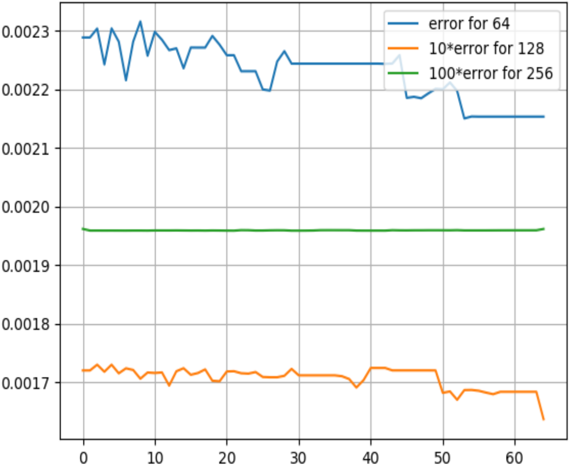
⇒ Numerically, we do the derivatives on the cell of the foot of characteristic (thus, there it is a polynomial)

First results: L^∞ error for rotation vs number of patch per direction

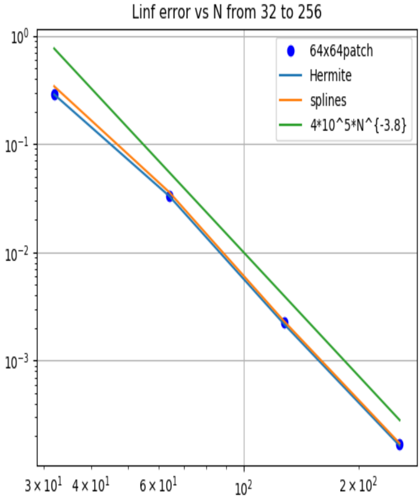
First point: splines; last point: Hermite

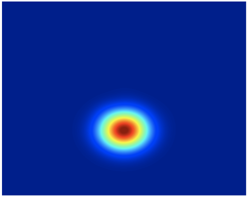
$\Delta t = 2\pi/32$ 100 iterations; grid: $64 \times 64,\ 128 \times 128,\ 256 \times 256$

$f_0(x, y) = \exp(-0.07((40x + 4.8)^2 + (40v + 4.8)^2))$ on $[-0.5, 0.5]^2$ MViolard2007



First results: L^∞ error for rotation vs N for $N \times N$ grid and solution at final time



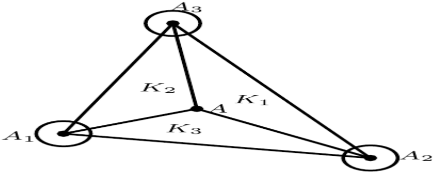


Hermite on triangles

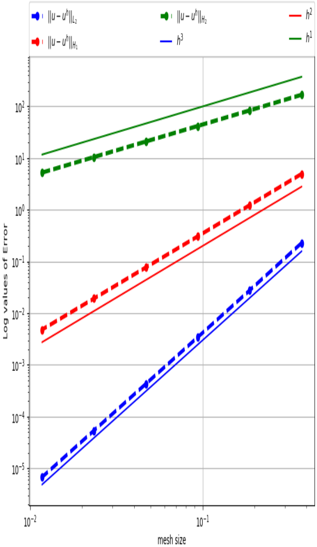
- Motivation: flexibility of the geometry
- CT: Reduced **C**lough-**T**ucher

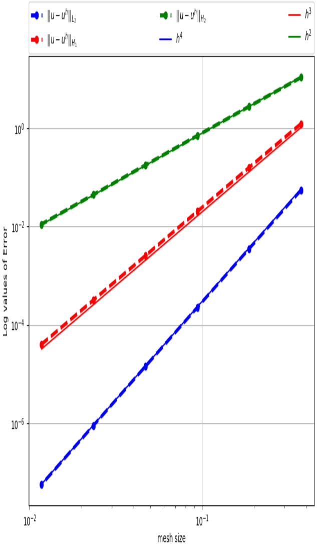
  ⇒ $f, \partial_x f, \partial_y f$ at each node
  C^1 Interpolation which reproduces polynomials of degree ≤ 2
- MT: **Mit**chell

  ⇒ $f, \partial_x f, \partial_y f$ at each node + a mixed derivative
  Reconstruction of the full Hessian matrix using the mixed derivative
  Interpolation reproduces polynomials of degree ≤ 3



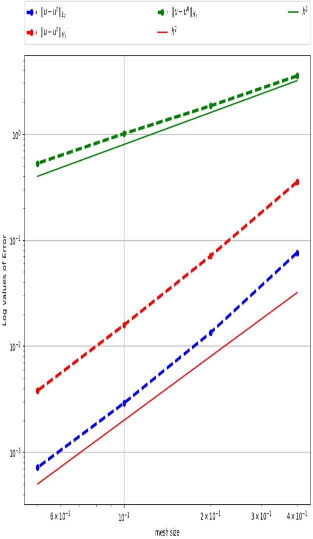
Interpolation error for CT and Mitchell

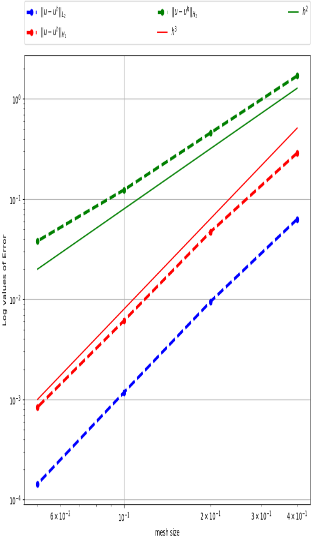




Advection (rotation) error for CT and Mitchell CFL2

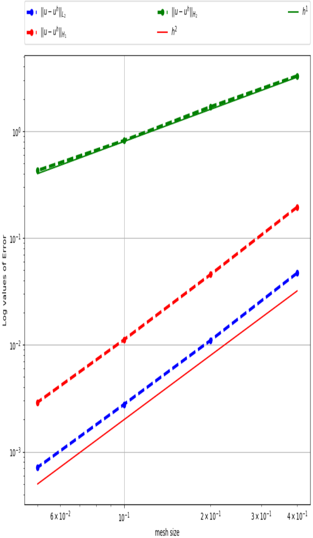
$$\Delta t = (CFL * h)/(\max(|V_1|) + \max(|V2|)),\ h = \Delta x = \Delta y$$

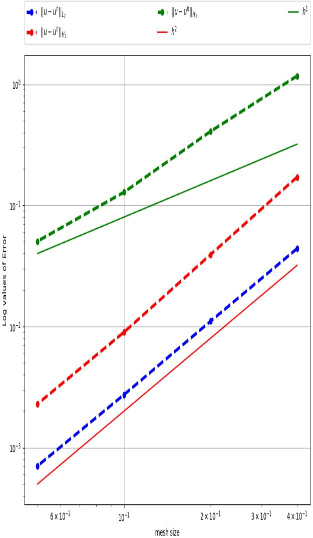




Advection (rotation) error for CT and Mitchell CFL10

$$\Delta t = (CFL * h)/(\max(|V_1|) + \max(|V2|)),\ h = \Delta x = \Delta y$$





Guiding center model

- We look for $\rho = \rho(t, x, y)$ satisfying the guiding center model

$$\begin{cases} \partial_t \rho + \partial_y \phi \partial_x \rho - \partial_x \phi \partial_y \rho = 0 \\ -\Delta \phi = \rho \end{cases}$$

- We take an annulus as domain

$$\Omega = \{(x = r\cos(\theta), y = r\sin(\theta)) \in \mathbb{R}^2, 1 \leq r \leq 10,\ \theta \in [0, 2\pi]\}$$

and consider the **diocotron instability**

$$\rho_0(x, y) = (1 + \varepsilon \cos(\ell\theta))e^{-\frac{(r-r_0)^2}{2\sigma^2}}$$

- $r_0 = 4.5,\ \sigma = 0.5$
- $\ell \in \{2, 3, 4, 5, 6\}$
- $\Delta t = 0.05$
- this testcase has been developed on polar/curvilinear grid, hexagonal (for hexagonal domain) and cartesian grid; also with Particle in Cell method...
- we study it here on unstructured triangular grid with Hermite advection scheme (other works on unstructured grids: SLDG method, Lattice Boltzmann...)

"Analytical solution" and meshes

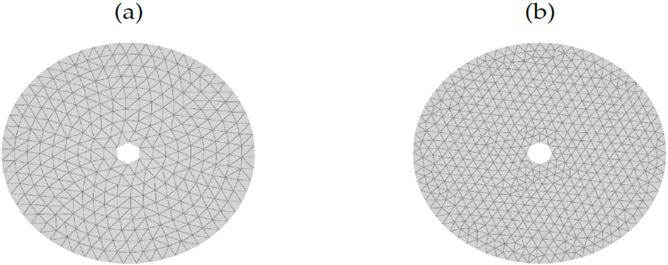
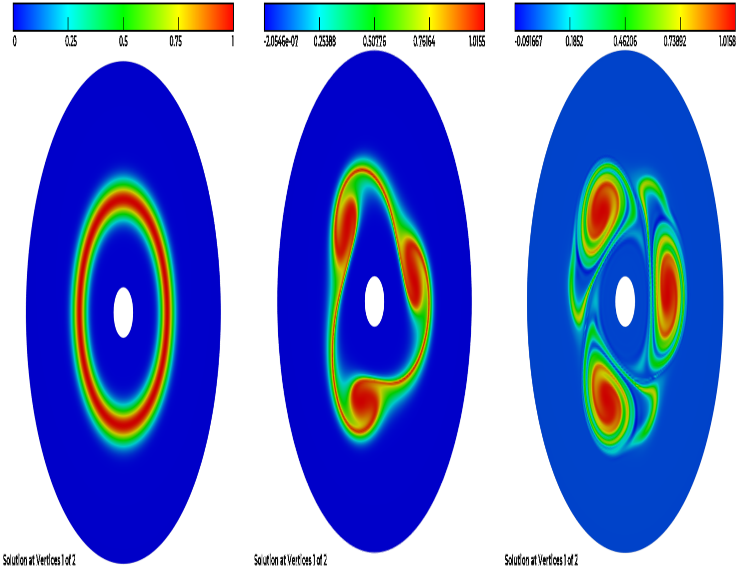


Figure 1: (a) regular mesh, (b) non regular mesh of an annulus

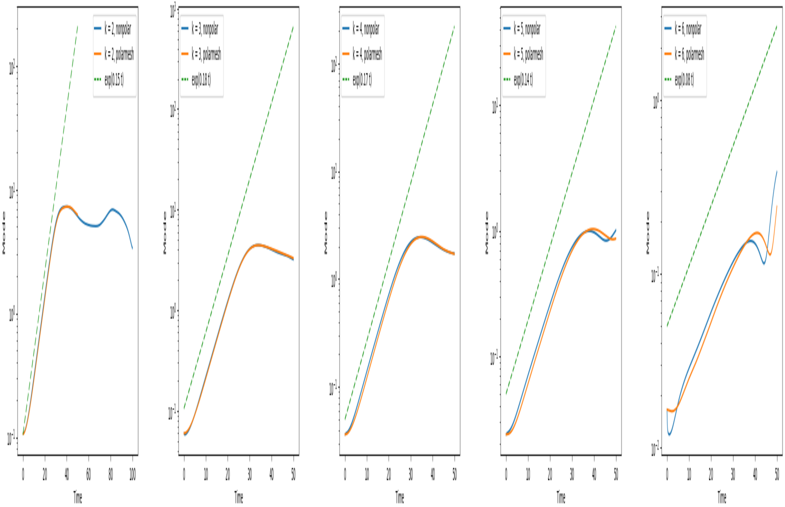
mode	growth rate of instability
2	0.1521506183167334
3	0.17522906264985497
4	0.16808429177954187
5	0.13516114350009326
6	0.07950579246451214

Table 1: Theoretical growth rate of diocotron instability

Example for the mode $\ell = 3$



Instability growth rate for $\ell \in \{2, 3, 4, 5, 6\}$



About the scheme in time

- We use a second order scheme in time based on predictor corrector method

$$\begin{aligned}
f^{n+\frac{1}{2}}(\mathbf{x}) &= f^n(\mathbf{x} - \tfrac{\Delta t}{2}\mathbf{V}_n(\mathbf{x})) \\
\partial_x f^{n+\frac{1}{2}}(\mathbf{x}) &= (1 - \tfrac{\Delta t}{2}\partial_x \mathbf{V}_1(\mathbf{x}))\partial_x f^n(\mathbf{X}_v^n) - (\tfrac{\Delta t}{2}\partial_x \mathbf{V}_2(\mathbf{x}))\partial_y f^n(\mathbf{X}_v^n) \\
\partial_y f^{n+\frac{1}{2}}(\mathbf{x}) &= (-\tfrac{\Delta t}{2}\partial_y \mathbf{V}_1(\mathbf{x}))\partial_x f^n(\mathbf{X}_v^n) + (1 - \tfrac{\Delta t}{2}\partial_y \mathbf{V}_2(\mathbf{x}))\partial_y f^n(\mathbf{X}_v^n)
\end{aligned}$$

  with $\mathbf{X}_v^n := \mathbf{x} - \frac{\Delta t}{2}\mathbf{V}_n(\mathbf{x})$
  To compute the derivatives of the velocity in this step, we use the average of the P1 gradient in all triangles around the nodes.

- Compute $\mathbf{V}_{n+\frac{1}{2}}$ by solving Poisson equation :

$$-\Delta\phi_{n+\frac{1}{2}} = f_{n+\frac{1}{2}} \quad \text{and} \quad \mathbf{V}_{n+\frac{1}{2}} = \textbf{curl}\ \phi_{n+\frac{1}{2}}$$

- Compute the characteristic's foot $\mathbf{X}(t_n)$
  The derivatives of $\mathbf{X}^n$ are computed by using a fixed point method. By denoting $\mathbf{X}^n = (X_1^n, X_2^n)$, we get:

$$\begin{cases}
X_1^n &= x - \Delta t\ \mathbf{V}_1^{n+1/2}\left(\frac{x+X_1^n}{2}, \frac{y+X_2^n}{2}\right) \\
X_2^n &= y - \Delta t\ \mathbf{V}_2^{n+1/2}\left(\frac{x+X_1^n}{2}, \frac{y+X_2^n}{2}\right)
\end{cases}$$

By deriving the equations of above system, one obtains :

$$\left\{\begin{array}{ll}
\partial_x X_1^n & = 1 - \dfrac{\Delta t}{2}\,\left[(1+\partial_x X_1^n)\partial_x \mathbf{V}_1^{n+1/2}\left(\frac{x+X_1^n}{2},\frac{y+X_2^n}{2}\right) + \partial_x X_2^n \partial_y \mathbf{V}_1^{n+1/2}\left(\frac{x+X_1^n}{2},\frac{y+X_2^n}{2}\right)\right] \\
\partial_y X_1^n & = -\dfrac{\Delta t}{2}\,\left[\partial_y X_1^n \partial_x \mathbf{V}_1^{n+1/2}\left(\frac{x+X_1^n}{2},\frac{y+X_2^n}{2}\right) + (1+\partial_y X_2^n)\partial_y \mathbf{V}_1^{n+1/2}\left(\frac{x+X_1^n}{2},\frac{y+X_2^n}{2}\right)\right] \\
\partial_x X_2^n & = -\dfrac{\Delta t}{2}\,\left[(1+\partial_x X_1^n)\partial_x \mathbf{V}_2^{n+1/2}\left(\frac{x+X_1^n}{2},\frac{y+X_2^n}{2}\right) + \partial_x X_2^n \partial_y \mathbf{V}_2^{n+1/2}\left(\frac{x+X_1^n}{2},\frac{y+X_2^n}{2}\right)\right] \\
\partial_y X_2^n & = 1 - \dfrac{\Delta t}{2}\,\left[\partial_y X_1^n \partial_x \mathbf{V}_2^{n+1/2}\left(\frac{x+X_1^n}{2},\frac{y+X_2^n}{2}\right) + (1+\partial_y X_2^n)\partial_y \mathbf{V}_2^{n+1/2}\left(\frac{x+X_1^n}{2},\frac{y+X_2^n}{2}\right)\right]
\end{array}\right.$$

where $\mathbf{V}^{n+1/2}(\mathbf{X}) = \mathbf{V}(t_{n+1/2}, \mathbf{X})$
In this step, we can use either P1 or CT interpolation to approximate the velocity on the midpoints $\frac{\mathbf{x}+\mathbf{X}}{2}$. For the CT one, we will need the gradient of $\mathbf{V}^{n+\frac{1}{2}}$ in the mesh points.

- Conclusion
  - New high order Hermite advection on uniform mesh for Vlasov-Poisson
  - $1D$ stability property for splines on patches with propagation of gradients of the boundary of the patch
  - New Hermite $2D$ advection with propagation of mixed derivative
  - Generalization in a new local splines method in $2D$ and first tests on rotation
  - Development of Hermite methods on triangular mesh:
    - CT method for guiding center model
    - first results on higher order Mitchell method
- Perspectives
  - guiding center model for new high order Hermite on uniform mesh, local splines, Mitchell
  - Influence of the mesh for other tokamak like poloidal planes (cf polar, non polar)
  - Parallelization
  - Drift kinetic simulation (4D)
  - stability and convergence; positivity? conservative version?
  - multi-resolution, multi-patch...