FARGO3D

a new GPU tool for MHD simulations

Pablo Benítez-Llambay, Frédéric Masset, Cristián Beaugé

FOF2013 – Córdoba – Abril 2013

400 AÑOS
UNC | Universidad
Nacional de Córdoba

Qué es FARGO3D?

- Un código MHD de propósito general, optimizado para el estudio de los procesos astrofísicos en discos gaseosos.
- El nombre proviene del algoritmo de advección utilizado, llamado algoritmo FARGO (Masset, 2000)
- Es un código desarrollado desde cero. Los desarrolladores somos Frédéric y yo.

Para qué hacer un código?

- Estamos interesados en la dinámica de la formación planetaria, más precisamente, en los mecanismos de transferencia de momento angular entre un protoplaneta y el disco de gas.

- Los códigos actuales (en nuestra área) no explotan la tecnología más poderosa existente.

- Es divertido!

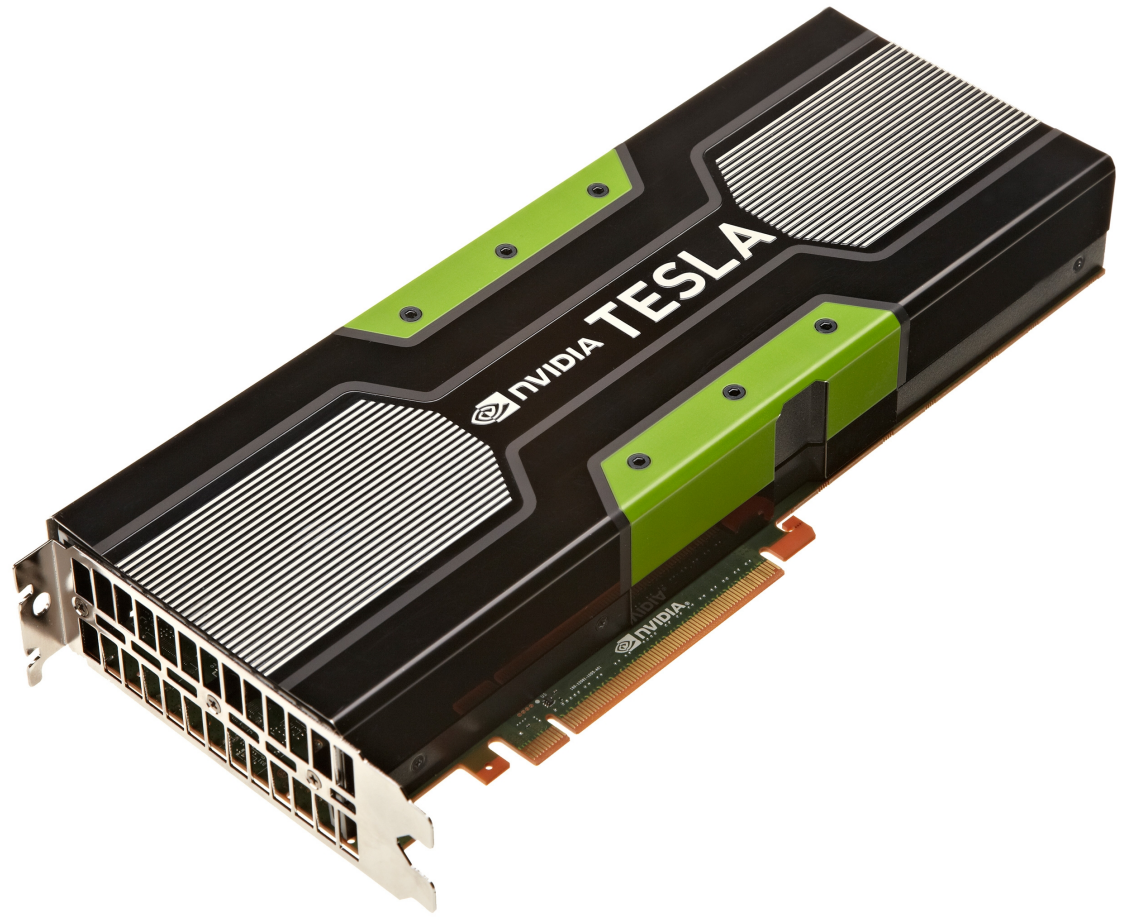
Algunas características...

- Grid. (Staggered), regular, no adaptativa.
- Multidimensional (1D, 2D, 3D).
- Multigeometría (Cartesianas, cilíndricas, esféricas).
- N-body.
- Hidrodinámico (Navier-Stokes).
- Magneto-Hidrodinámico (Resistivo).
- Isotermo, Adiabático.
- Paralelo (memoria compartida o distribuida). Se optimizan comunicaciones entre nodos (area de contacto mínima).
- Corre en una GPU!

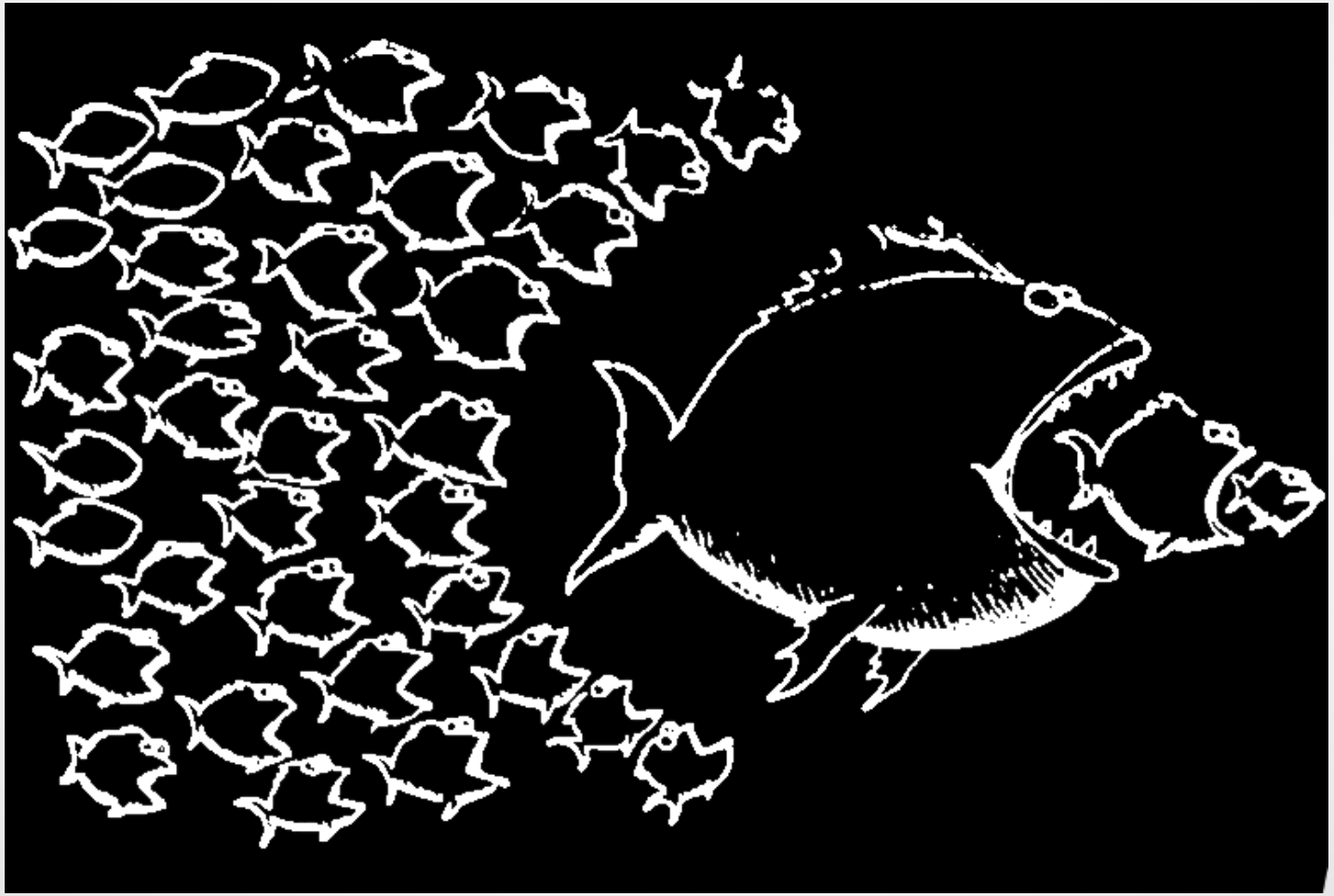
Qué es una GPU?

Crysis 3 (Cada una de estas imágenes debe ser realizada a una tasa de 60 Frames por segundo o más! (HD))

Qué es una GPU?



Qué es una GPU?



Paralelo y en GPU?

Proposición: *Si FARGO3D corre en un cluster de memoria compartida, y FARGO3D corre en una GPU, entonces FARGO3D puede correr en un cluster de GPU'S.*

Cluster de CPU's + GPU's

(Muy pronto en el IATE)

La parte mala de la GPU

La programación en GPU está relativamente lejos/cerca de ser trivial. Usar esta tecnología implica dejar de lado lo que uno esta haciendo por ~1 mes y dedicarlo a estudiar los manuales y a hacer experimentos hasta entender. (La peor parte es que la GPU no sirve para resolver cualquier problema...)

La parte buena de FARGO3D

- Si algún usuario necesita desarrollar rutinas para atacar un problema particular que no hayamos considerado, no necesita saber programar en GPU's...

- Todo el código GPU es generado automáticamente en tiempo de compilación (al hacer el “make”).

Ejemplo

CPU

```
1  //<FLAGS>
2  //#define __GPU
3  //#define __NOPROTO
4  //<\FLAGS>
5
6  //<INCLUDES>
7  #include "fargo3d.h"
8  //<\INCLUDES>
9
10 void ComputePressureFieldIso_cpu() {
11 //<USER_DEFINED>
12   INPUT(Energy);
13   INPUT(Density);
14   OUTPUT(Pressure);
15 //<\USER_DEFINED>
16
17 //<EXTERNAL>
18   real* dens = Density->field_cpu;
19   real* cs   = Energy->field_cpu;
20   real* pres = Pressure->field_cpu;
21   int pitch  = Pitch_cpu;
22   int stride = Stride_cpu;
23   int size_x = Nx;
24   int size_y = Ny+2*NGHY;
25   int size_z = Nz+2*NGHZ;
26 //<\EXTERNAL>
27
28 //<INTERNAL>
29   int i;
30   int j;
31   int k;
32 //<\INTERNAL>
33
34 //<MAIN_LOOP>
35       (k=0; k<size_z; k++) {
36         (j=0; j<size_y; j++) {
37           (i=0; i<size_x; i++ ) {
38 //<#>
39         pres[ll] = dens[ll]*cs[ll]*cs[ll];
40 //<\#>
41      }
42     }
43   }
44 //<\MAIN_LOOP>
45 }
```

GPU

```
1  #define __GPU
2  #define __NOPROTO
3  #include "fargo3d.h"
4
5             void ComputePressureFieldIso_kernel(real* dens,
6                                                 real* cs,
7                                                 real* pres,
8                                                 int pitch,
9                                                 int stride,
10                                                int size_x,
11                                                int size_y,
12                                                int size_z) {
13
14   int i;
15   int j;
16   int k;
17   int ll;
18
19 i = threadIdx.x + blockIdx.x * blockDim.x;
20 j = threadIdx.y + blockIdx.y * blockDim.y;
21 k = threadIdx.z + blockIdx.z * blockDim.z;
22
23    (k>=0 && k<size_z) {
24      (j>=0 && j<size_y) {
25        (i<size_x) {
26        ll = l;
27        pres[ll] = dens[ll]*cs[ll]*cs[ll];
28      }
29    }
30  }
31 }
32
33 extern "C" void ComputePressureFieldIso_gpu() {
34
35   INPUT(Energy);
36   INPUT(Density);
37   OUTPUT(Pressure);
38
39 dim3 block (BLOCK_X, BLOCK_Y, BLOCK_Z);
40 dim3 grid ((Nx+block.x-1)/block.x,
41 ((Ny+2*NGHY)+block.y-1)/block.y,
42 ((Nz+2*NGHZ)+block.z-1)/block.z);
43
44 #ifdef BIGMEM
45 #define xmin_d &Xmin_d
46 #define ymin_d &Ymin_d
47 #define zmin_d &Zmin_d
48 #define Sxj_d &Sxj_d
49 #define Syj_d &Syj_d
50 #define Szj_d &Szj_d
51 #define Sxk_d &Sxk_d
52 #define Syk_d &Syk_d
53 #define Szk_d &Szk_d
54 #define InvVj_d &InvVj_d
55 #endif
56
57  cudaFuncSetCacheConfig(ComputePressureFieldIso_kernel, cudaFuncCachePreferL1 );
58  ComputePressureFieldIso_kernel<<<grid,block>>>(Density->field_gpu,
59                                                 Energy->field_gpu,
60                                                 Pressure->field_gpu,
61                                                 Pitch_gpu,
62                                                 Stride_gpu,
63                                                 Nx,
64                                                 Ny+2*NGHY,
65                                                 Nz+2*NGHZ);
66  check_errors("ComputePressureFieldIso_kernel");
67 }
```

Ejemplo

CPU

```
1  //<FLAGS>
2  //#define __GPU
3  //#define __NOPROTO
4  //<\FLAGS>
5
6  //<INCLUDES>
7  #include "fargo3d.h"
8  //<\INCLUDES>
9
10 void ComputePressureFieldIso_cpu() {
11 //<USER_DEFINED>
12   INPUT(Energy);
13   INPUT(Density);
14   OUTPUT(Pressure);
15 //<\USER_DEFINED>
16
17 //<EXTERNAL>
18   real* dens = Density->field_cpu;
19   real* cs   = Energy->field_cpu;
20   real* pres = Pressure->field_cpu;
21   int pitch  = Pitch_cpu;
22   int stride = Stride_cpu;
23   int size_x = Nx;
24   int size_y = Ny+2*NGHY;
25   int size_z = Nz+2*NGHZ;
26 //<\EXTERNAL>
27
28 //<INTERNAL>
29   int i;
30   int j;
31   int k;
32 //<\INTERNAL>
33
34 //<MAIN_LOOP>
35       (k=0; k<size_z; k++) {
36         (j=0; j<size_y; j++) {
37           (i=0; i<size_x; i++ ) {
38 //<#>
39         pres[ll] = dens[ll]*cs[ll]*cs[ll];
40 //<\#>
41      }
42     }
43   }
44 //<\MAIN_LOOP>
45 }
```

GPU

```
1  #define __GPU
2  #define __NOPROTO
3  #include "fargo3d.h"
4
5             void ComputePressureFieldIso_kernel(real* dens,
6                                                 real* cs,
7                                                 real* pres,
8                                                 int pitch,
9                                                 int stride,
10                                                int size_x,
11                                                int size_y,
12                                                int size_z) {
13
14   int i;
15   int j;
16   int k;
17   int ll;
18
19 i = threadIdx.x + blockIdx.x * blockDim.x;
20 j = threadIdx.y + blockIdx.y * blockDim.y;
21 k = threadIdx.z + blockIdx.z * blockDim.z;
22
23    (k>=0 && k<size_z) {
24      (j>=0 && j<size_y) {
25        (i<size_x) {
26        ll = l;
27        pres[ll] = dens[ll]*cs[ll]*cs[ll];
28      }
29    }
30  }
31 }
32
33 extern "C" void ComputePressureFieldIso_gpu() {
34
35   INPUT(Energy);
36   INPUT(Density);
37   OUTPUT(Pressure);
38
39 dim3 block (BLOCK_X, BLOCK_Y, BLOCK_Z);
40 dim3 grid ((Nx+block.x-1)/block.x,
41 ((Ny+2*NGHY)+block.y-1)/block.y,
42 ((Nz+2*NGHZ)+block.z-1)/block.z);
43
44 #ifdef BIGMEM
45 #define xmin_d &Xmin_d
46 #define ymin_d &Ymin_d
47 #define zmin_d &Zmin_d
48 #define Sxj_d &Sxj_d
49 #define Syj_d &Syj_d
50 #define Szj_d &Szj_d
51 #define Sxk_d &Sxk_d
52 #define Syk_d &Syk_d
53 #define Szk_d &Szk_d
54 #define InvVj_d &InvVj_d
55 #endif
56
57  cudaFuncSetCacheConfig(ComputePressureFieldIso_kernel, cudaFuncCachePreferL1 );
58  ComputePressureFieldIso_kernel<<<grid,block>>>(Density->field_gpu,
59                                                 Energy->field_gpu,
60                                                 Pressure->field_gpu,
61                                                 Pitch_gpu,
62                                                 Stride_gpu,
63                                                 Nx,
64                                                 Ny+2*NGHY,
65                                                 Nz+2*NGHZ);
66  check_errors("ComputePressureFieldIso_kernel");
67 }
```

Ejemplo

CPU

```
1  //<FLAGS>
2  //#define __GPU
3  //#define __NOPROTO
4  //<\FLAGS>
5
6  //<INCLUDES>
7  #include "fargo3d.h"
8  //<\INCLUDES>
9
10 void ComputePressureFieldIso_cpu() {
11 //<USER_DEFINED>
12   INPUT(Energy);
13   INPUT(Density);
14   OUTPUT(Pressure);
15 //<\USER_DEFINED>
16
17 //<EXTERNAL>
18   real* dens = Density->field_cpu;
19   real* cs   = Energy->field_cpu;
20   real* pres = Pressure->field_cpu;
21   int pitch  = Pitch_cpu;
22   int stride = Stride_cpu;
23   int size_x = Nx;
24   int size_y = Ny+2*NGHY;
25   int size_z = Nz+2*NGHZ;
26 //<\EXTERNAL>
27
28 //<INTERNAL>
29   int i;
30   int j;
31   int k;
32 //<\INTERNAL>
33
34 //<MAIN_LOOP>
35       (k=0; k<size_z; k++) {
36         (j=0; j<size_y; j++) {
37           (i=0; i<size_x; i++ ) {
38 //<#>
39         pres[ll] = dens[ll]*cs[ll]*cs[ll];
40 //<\#>
41      }
42     }
43   }
44 //<\MAIN_LOOP>
45 }
```

cudaparser.py

GPU

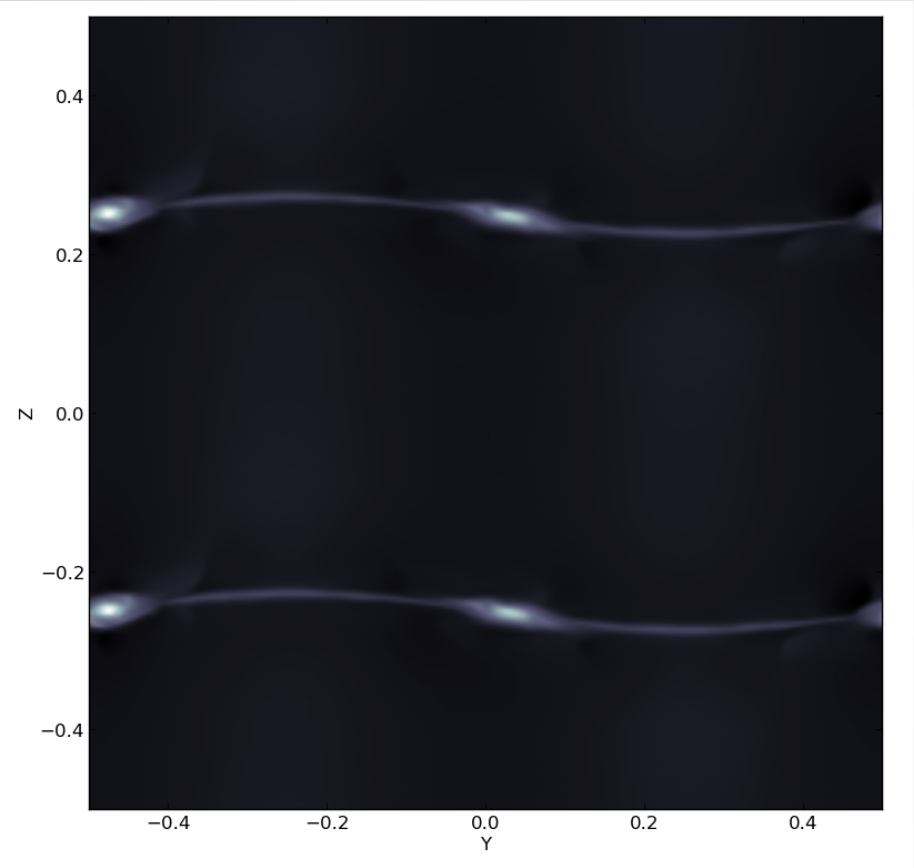
```
1  #define __GPU
2  #define __NOPROTO
3  #include "fargo3d.h"
4
5             void ComputePressureFieldIso_kernel(real* dens,
6                                                 real* cs,
7                                                 real* pres,
8                                                 int pitch,
9                                                 int stride,
10                                                int size_x,
11                                                int size_y,
12                                                int size_z) {
13
14   int i;
15   int j;
16   int k;
17   int ll;
18
19 i = threadIdx.x + blockIdx.x * blockDim.x;
20 j = threadIdx.y + blockIdx.y * blockDim.y;
21 k = threadIdx.z + blockIdx.z * blockDim.z;
22
23    (k>=0 && k<size_z) {
24      (j>=0 && j<size_y) {
25        (i<size_x) {
26        ll = l;
27        pres[ll] = dens[ll]*cs[ll]*cs[ll];
28      }
29    }
30  }
31 }
32
33 extern "C" void ComputePressureFieldIso_gpu() {
34
35   INPUT(Energy);
36   INPUT(Density);
37   OUTPUT(Pressure);
38
39 dim3 block (BLOCK_X, BLOCK_Y, BLOCK_Z);
40 dim3 grid ((Nx+block.x-1)/block.x,
41 ((Ny+2*NGHY)+block.y-1)/block.y,
42 ((Nz+2*NGHZ)+block.z-1)/block.z);
43
44 #ifdef BIGMEM
45 #define xmin_d &Xmin_d
46 #define ymin_d &Ymin_d
47 #define zmin_d &Zmin_d
48 #define Sxj_d &Sxj_d
49 #define Syj_d &Syj_d
50 #define Szj_d &Szj_d
51 #define Sxk_d &Sxk_d
52 #define Syk_d &Syk_d
53 #define Szk_d &Szk_d
54 #define InvVj_d &InvVj_d
55 #endif
56
57  cudaFuncSetCacheConfig(ComputePressureFieldIso_kernel, cudaFuncCachePreferL1 );
58  ComputePressureFieldIso_kernel<<<grid,block>>>(Density->field_gpu,
59                                                 Energy->field_gpu,
60                                                 Pressure->field_gpu,
61                                                 Pitch_gpu,
62                                                 Stride_gpu,
63                                                 Nx,
64                                                 Ny+2*NGHY,
65                                                 Nz+2*NGHZ);
66  check_errors("ComputePressureFieldIso_kernel");
67 }
```

Algunas características más

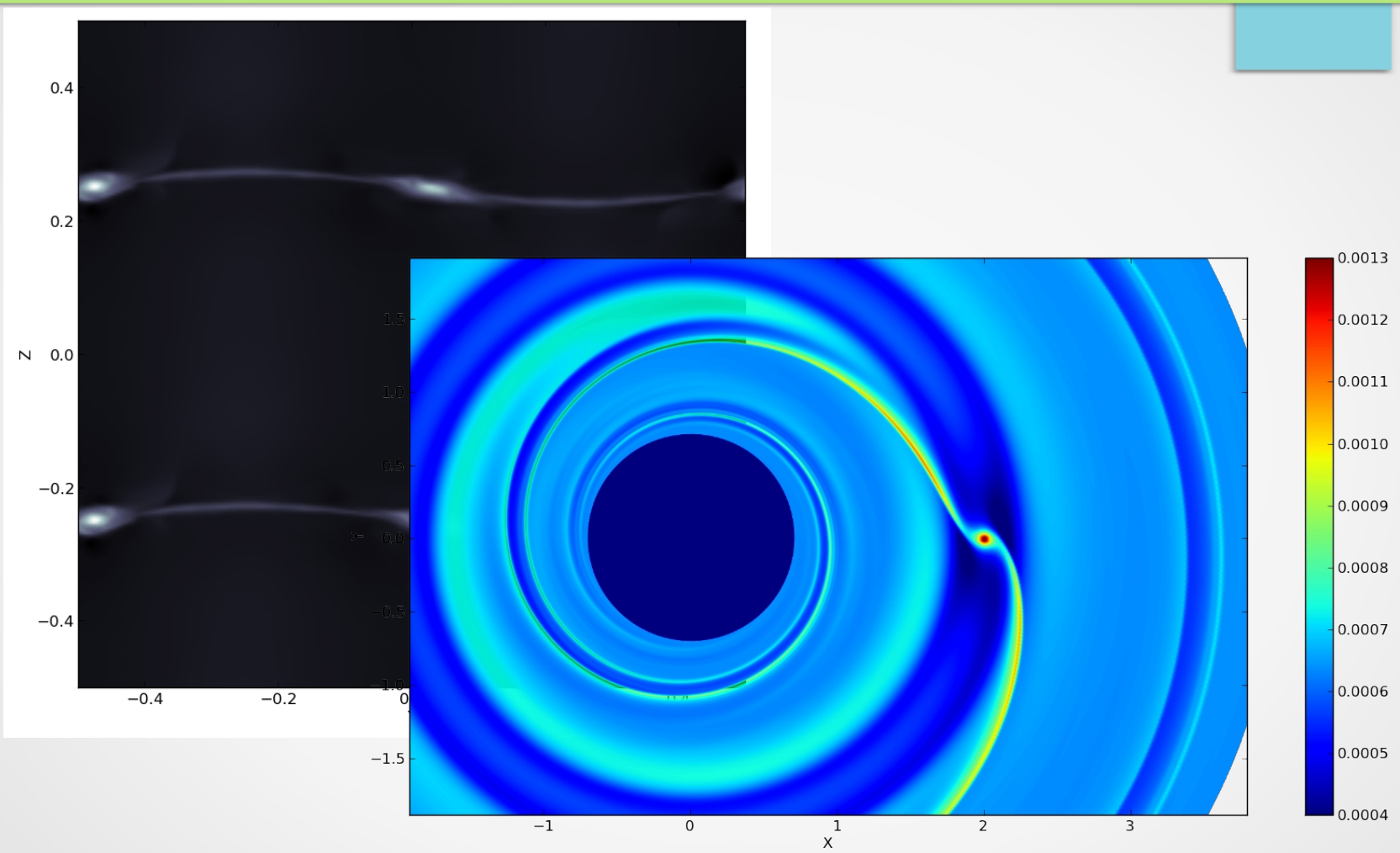
- Viene con una Test-Suite (ej: make SETUP=[otvortex, sod, sedov, mri, planet2d, disk3d, kelvin-helmoltz, rayleigh-taylor, wave2d, brio-wu, current-sheet], etc...).
- Vendrá con un paquete de análisis (python, idl, gnuplot, fortran, c).
- Compatible con formato VTK.
- Widget de visualización en tiempo real

ALGUNOS EJEMPLOS

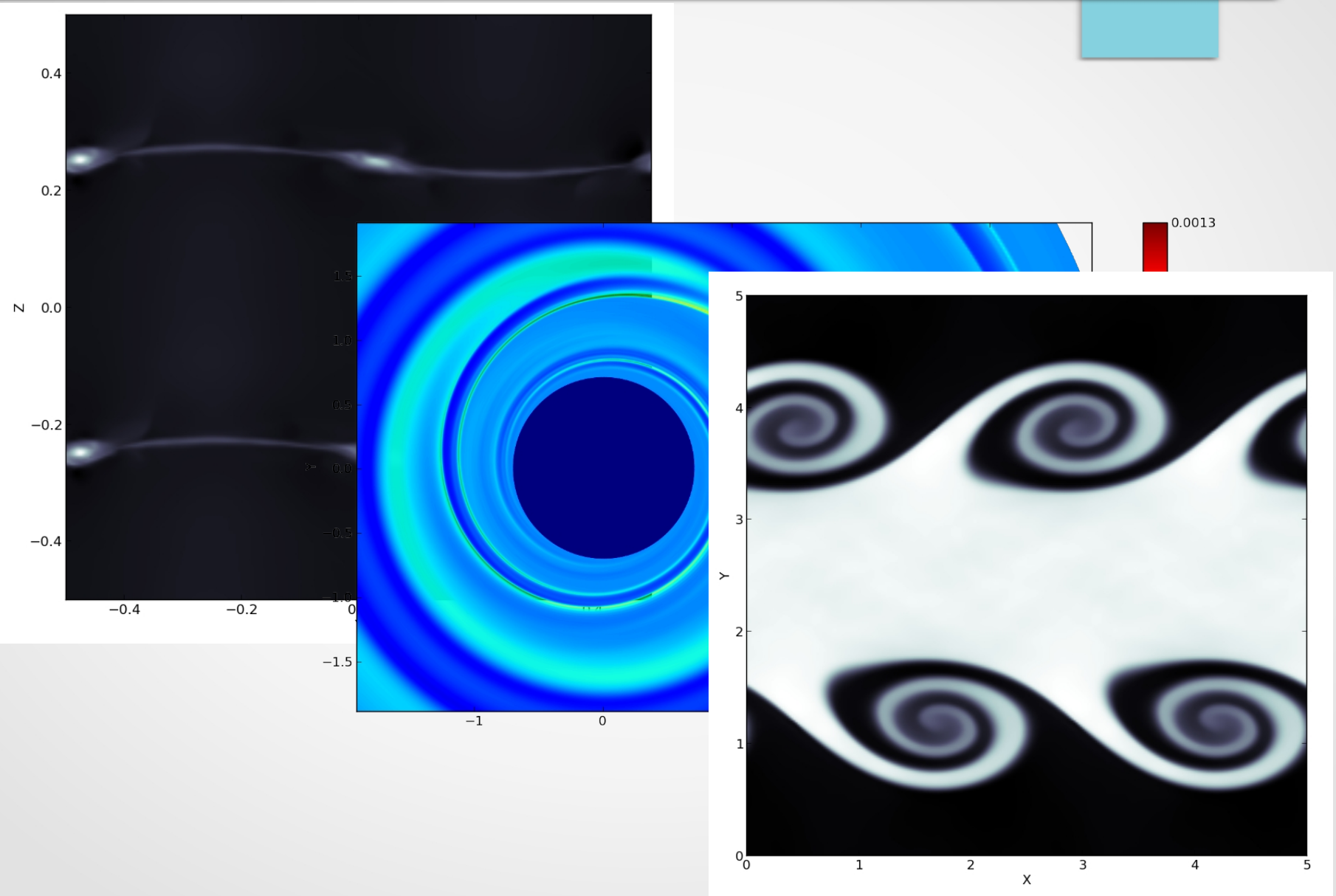
Ejemplos



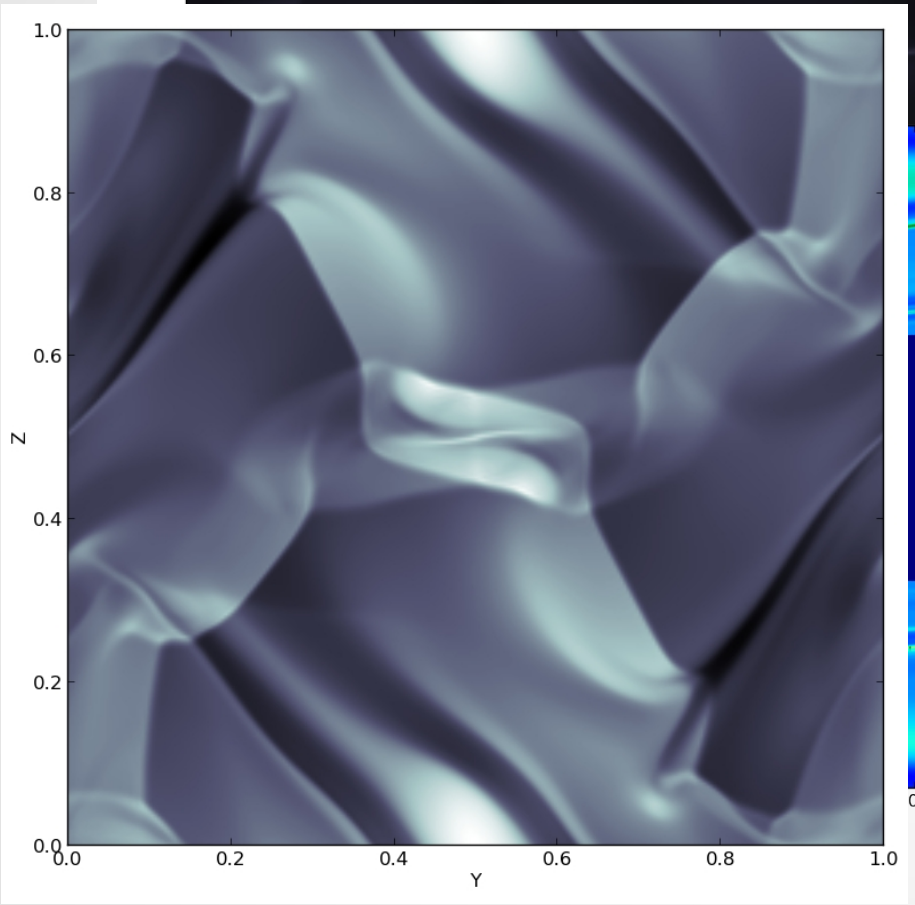
Ejemplos

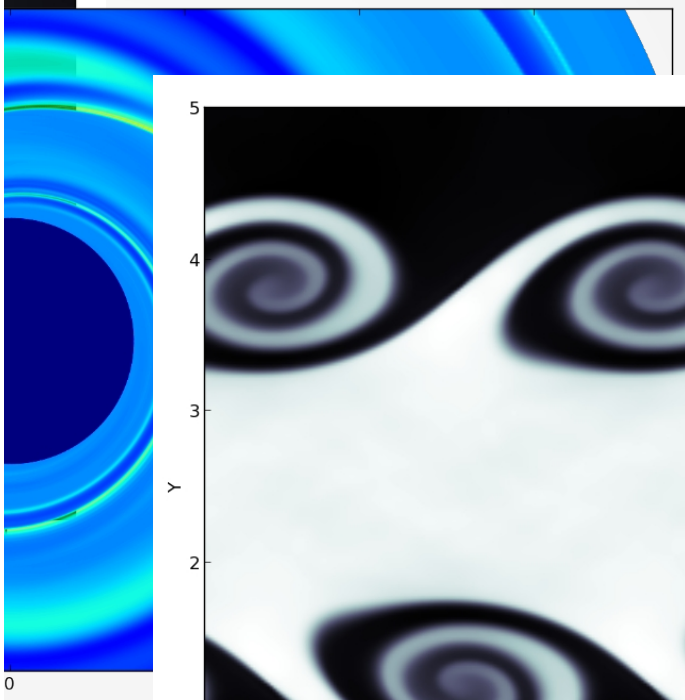

0.4
0.2
0.0
−0.2
−0.4
Z
−0.4
−0.2
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1
0
1
2
3
X
0.0013
0.0012
0.0011
0.0010
0.0009
0.0008
0.0007
0.0006
0.0005
0.0004

Ejemplos

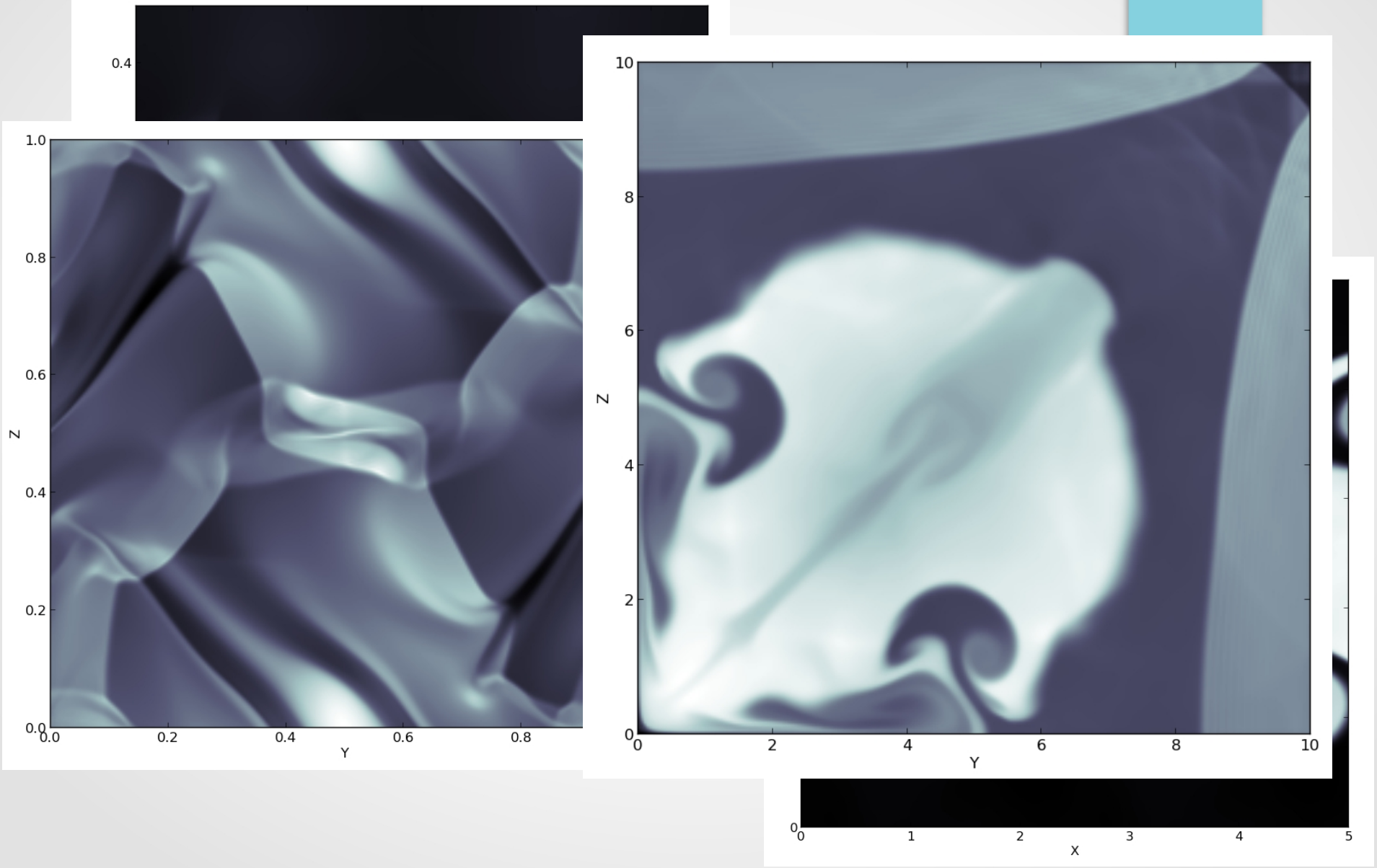

0.4
0.2
0.0
−0.2
−0.4
Z
−0.4
−0.2
0.0
0.0013
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1
0
5
4
3
2
1
0
Y
0
1
2
3
4
5
X

Ejemplos

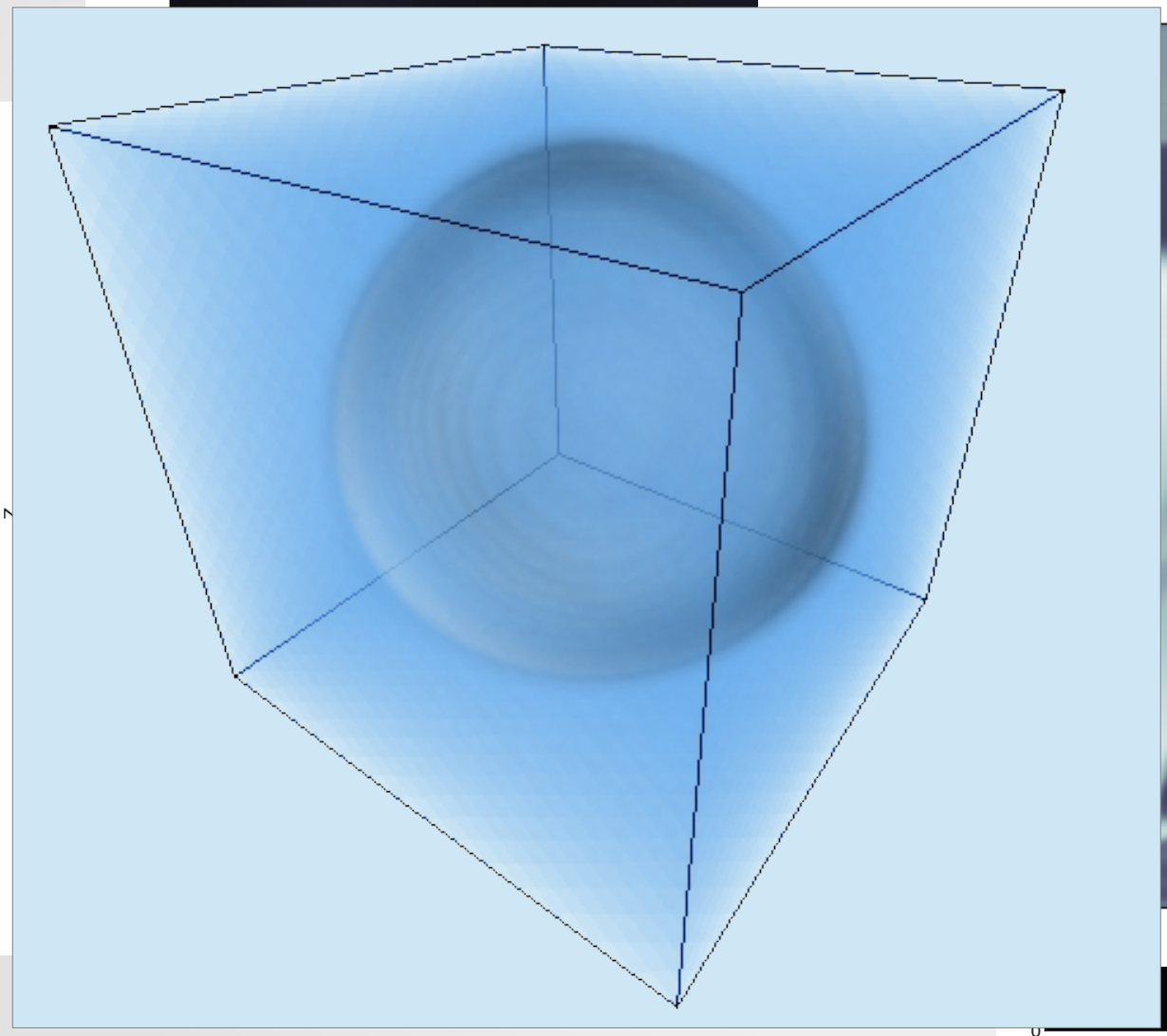




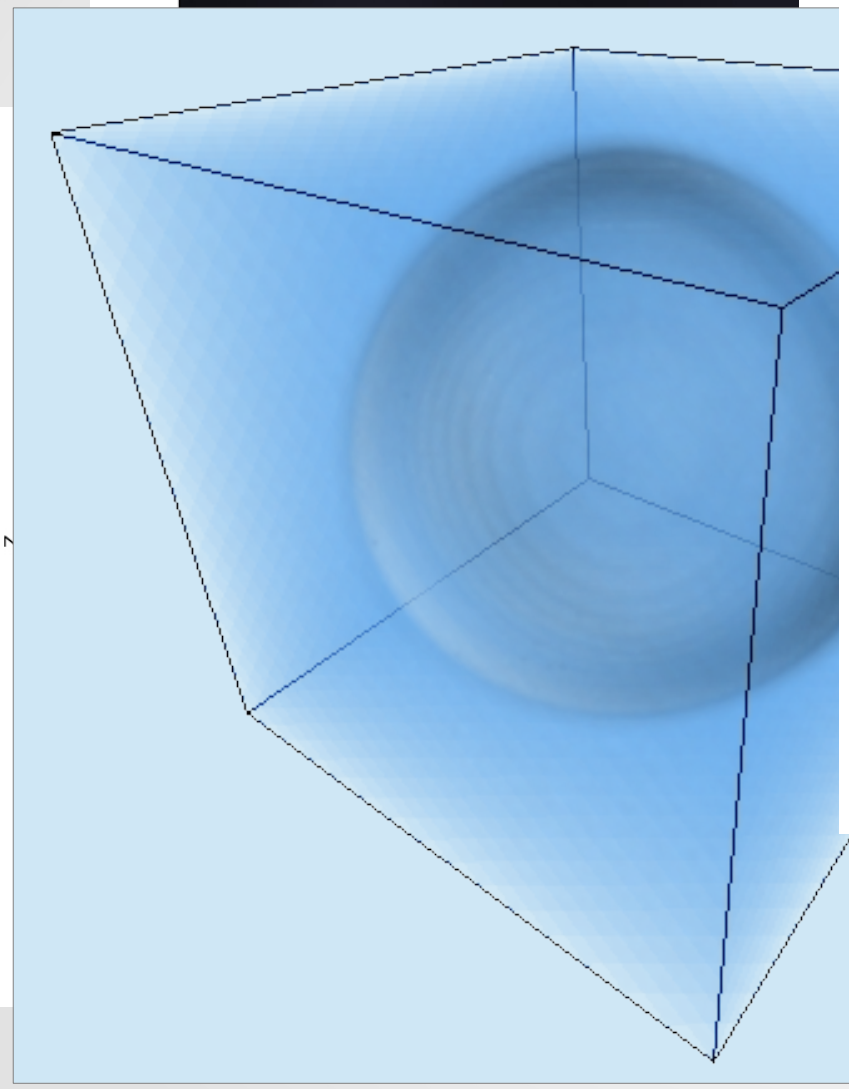
Ejemplos

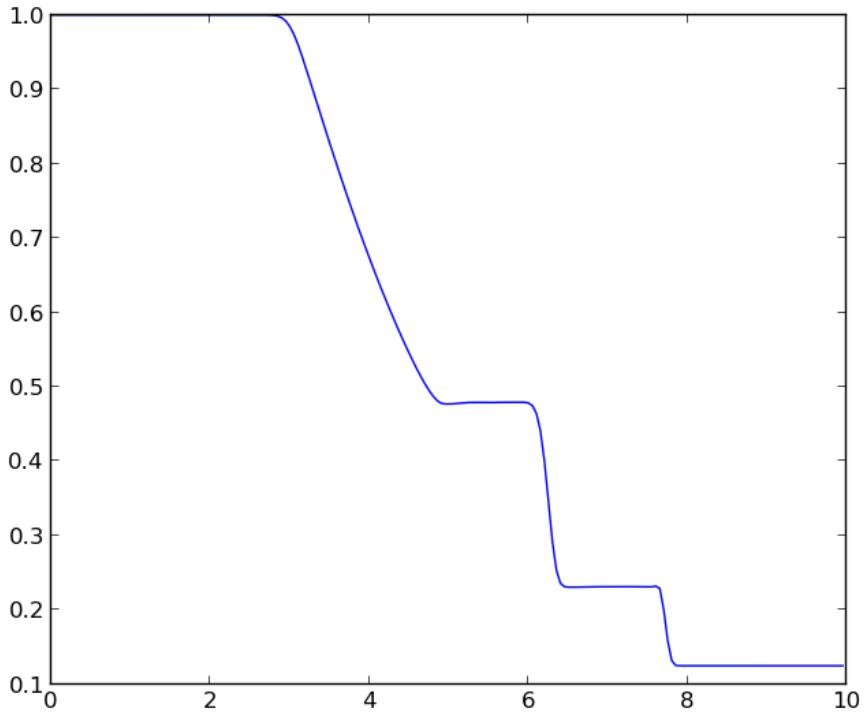

0.4
1.0
0.8
0.6
0.4
0.2
0.0
Z
0.0
0.2
0.4
0.6
0.8
Y
10
8
6
4
2
0
Z
0
2
4
6
8
10
Y
0
0
1
2
3
4
5
X

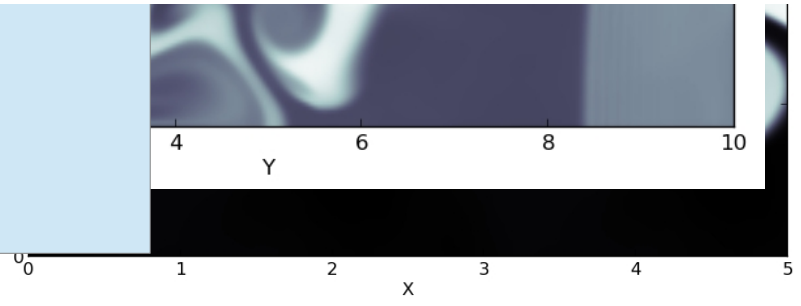
Ejemplos



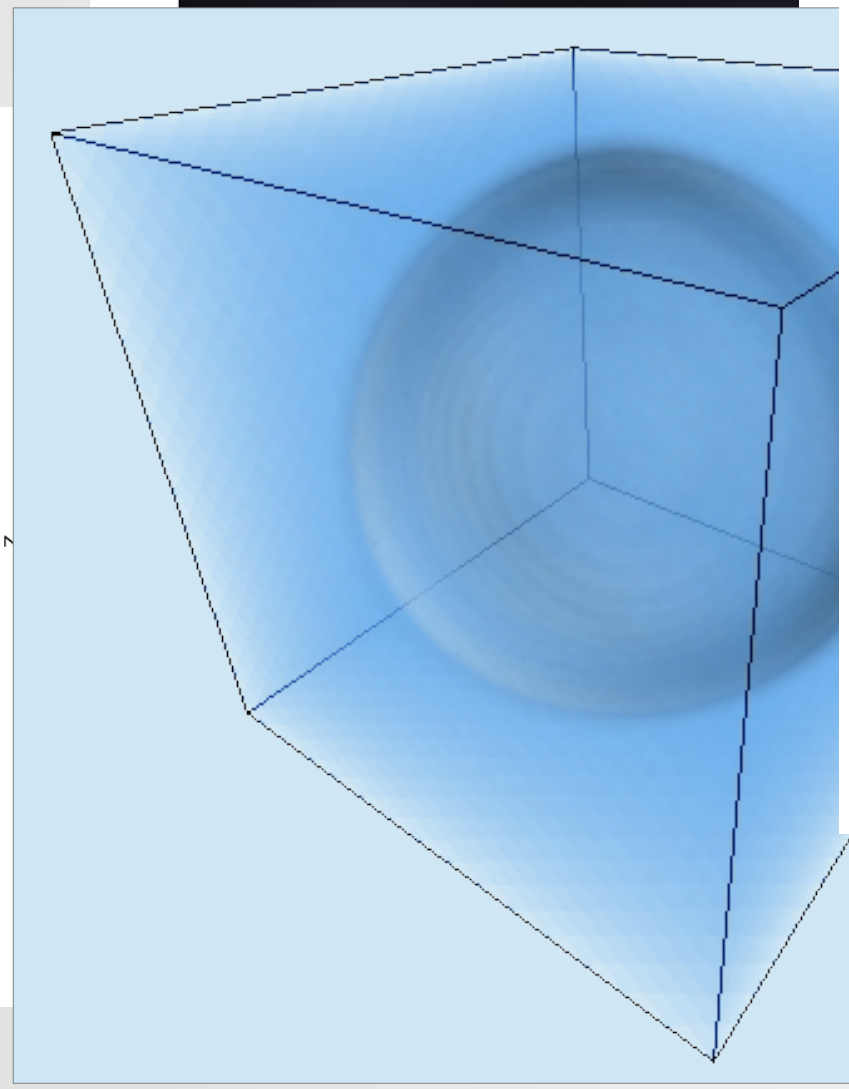
Ejemplos

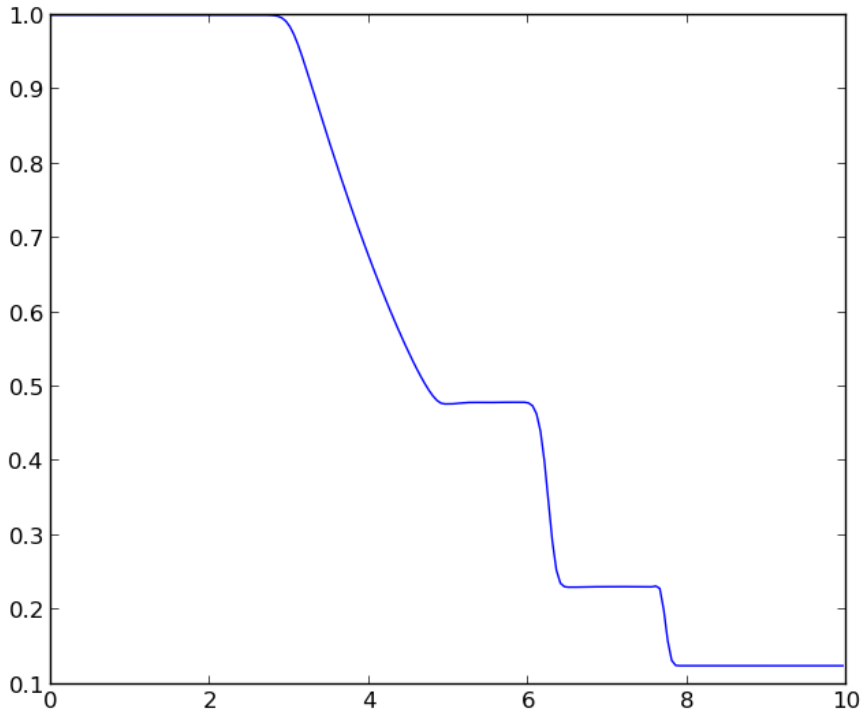


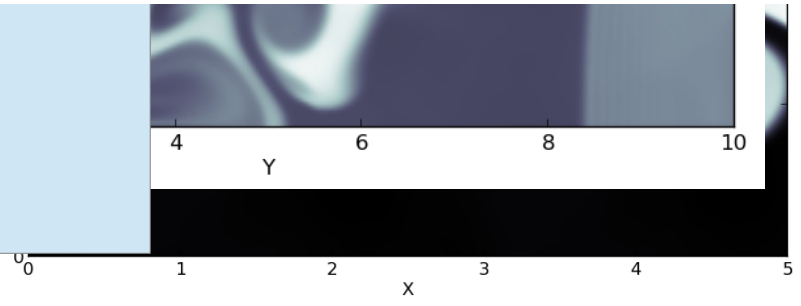




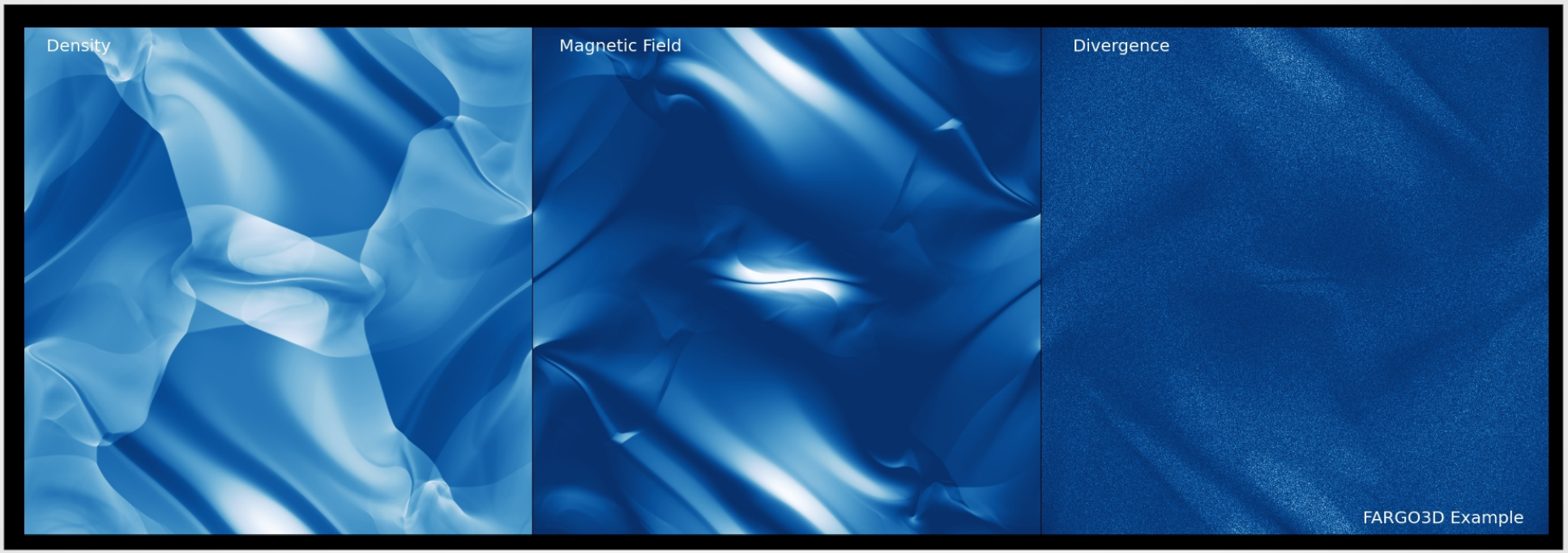
Ejemplos





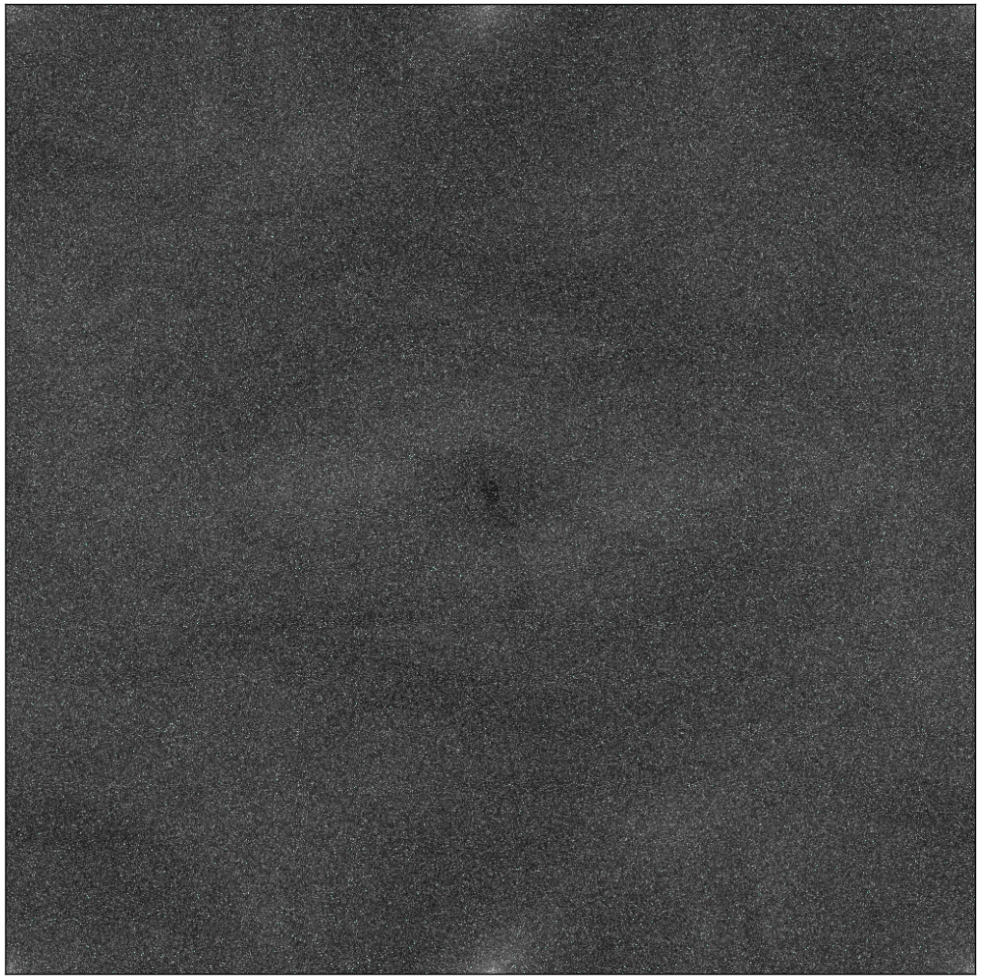


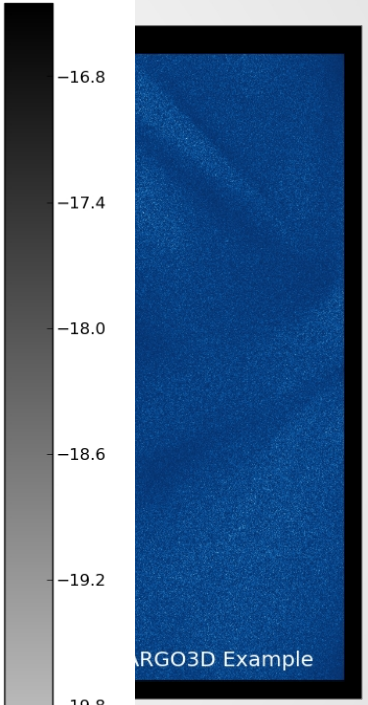
Orzag-Tang Vortex Test



**Un par de horas en una GPU modesta.
Quizás media hora o menos en una
Tesla K20. (hasta t=5.0)**

Orzag-Tang Vortex Test

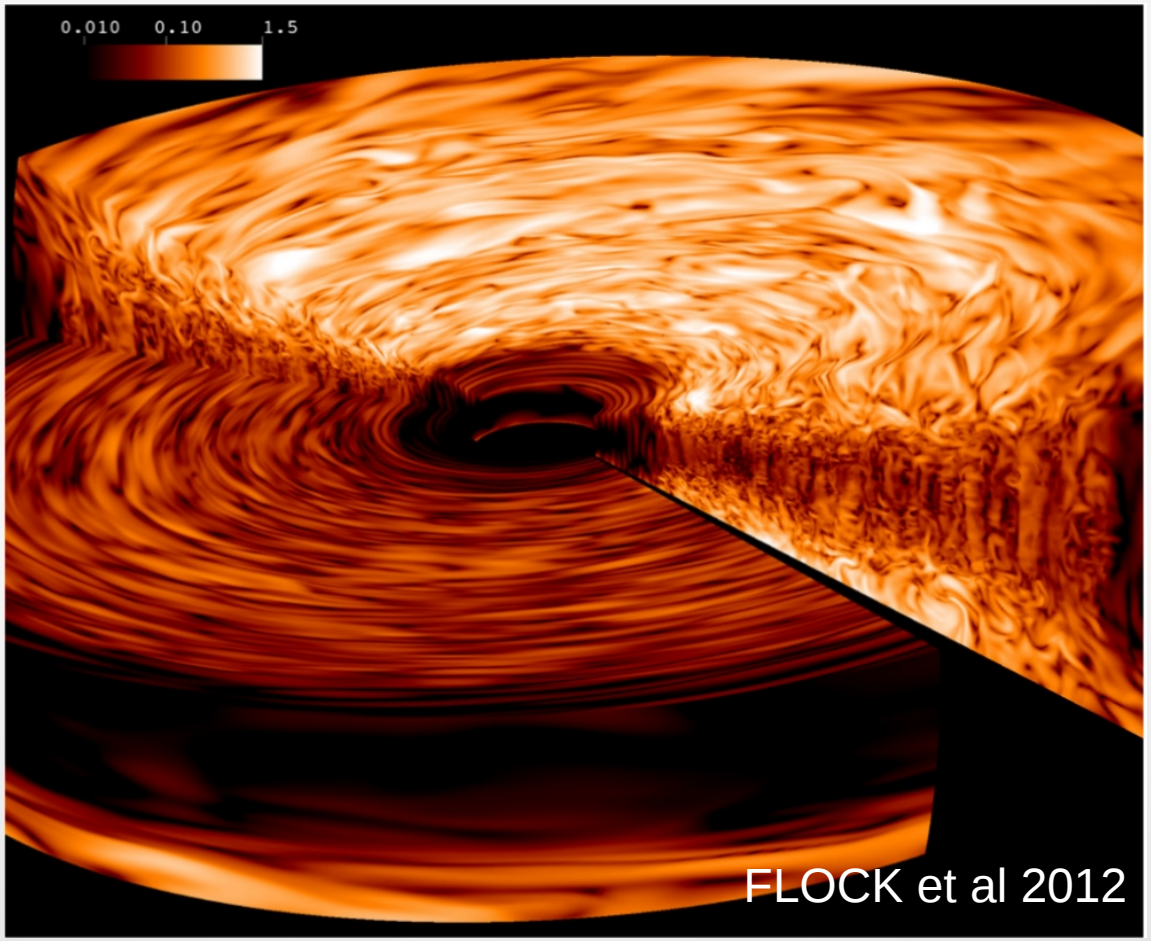




MI INVESTIGACION

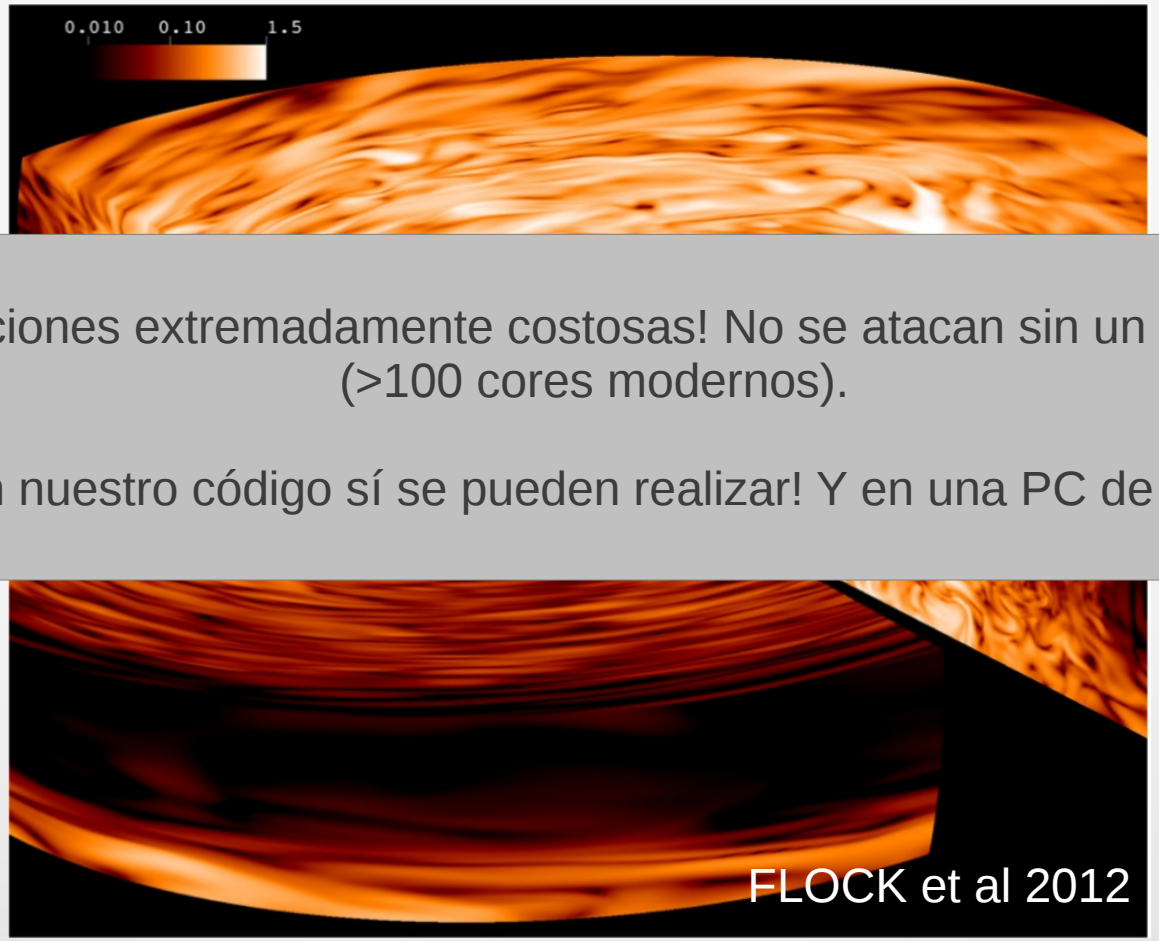
Algunas características más

Estudio el intercambio de momento angular entre un protoplaneta y un disco gaseoso.



Algunas características más

Estudio el intercambio de momento angular entre un protoplaneta y un disco gaseoso.

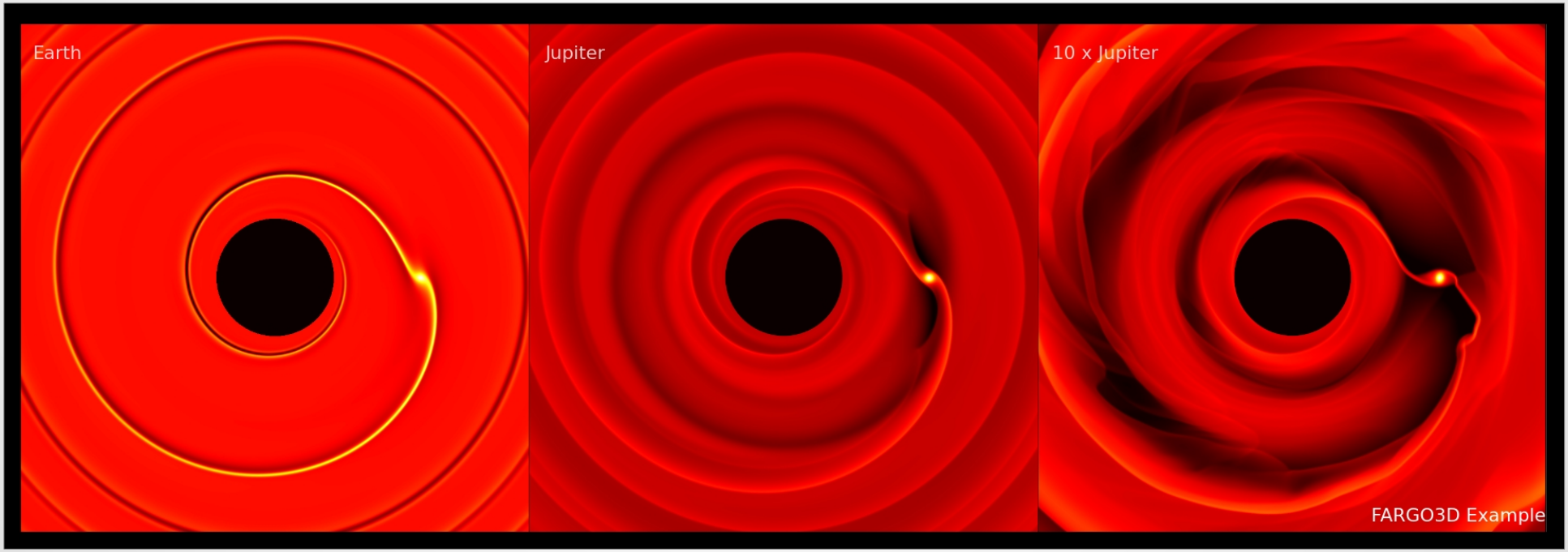


Simulaciones extremadamente costosas! No se atacan sin un cluster... (>100 cores modernos).

Pero con nuestro código sí se pueden realizar! Y en una PC de escritorio!

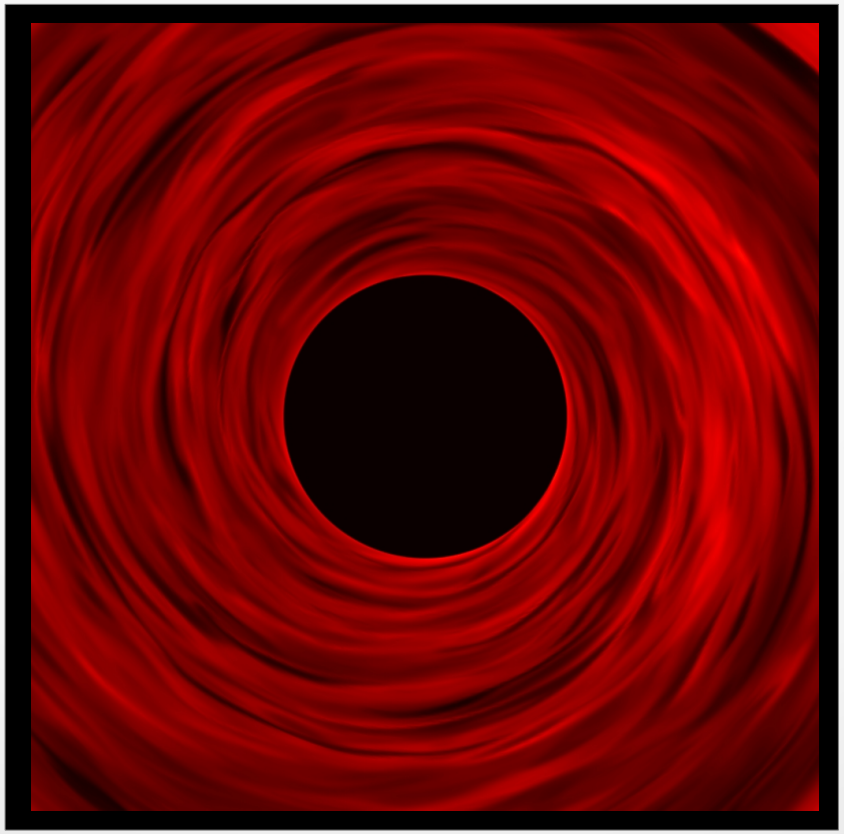
Recién comenzamos...

- Recién comenzamos a explotar el código...



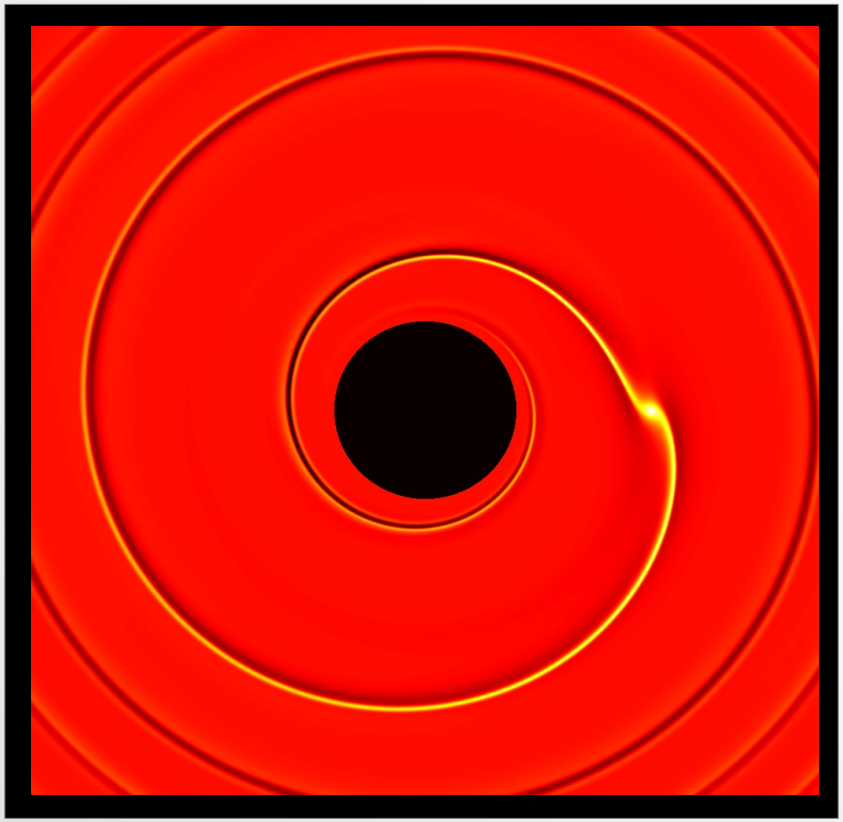
Recién comenzamos...

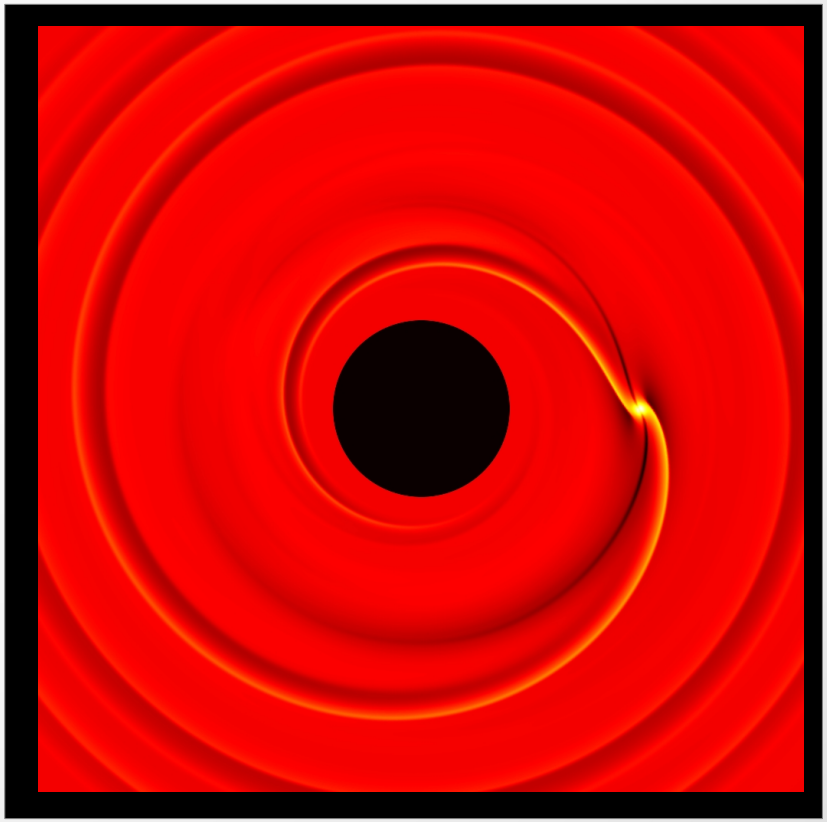
- Simulaciones con campo magnético (3D):



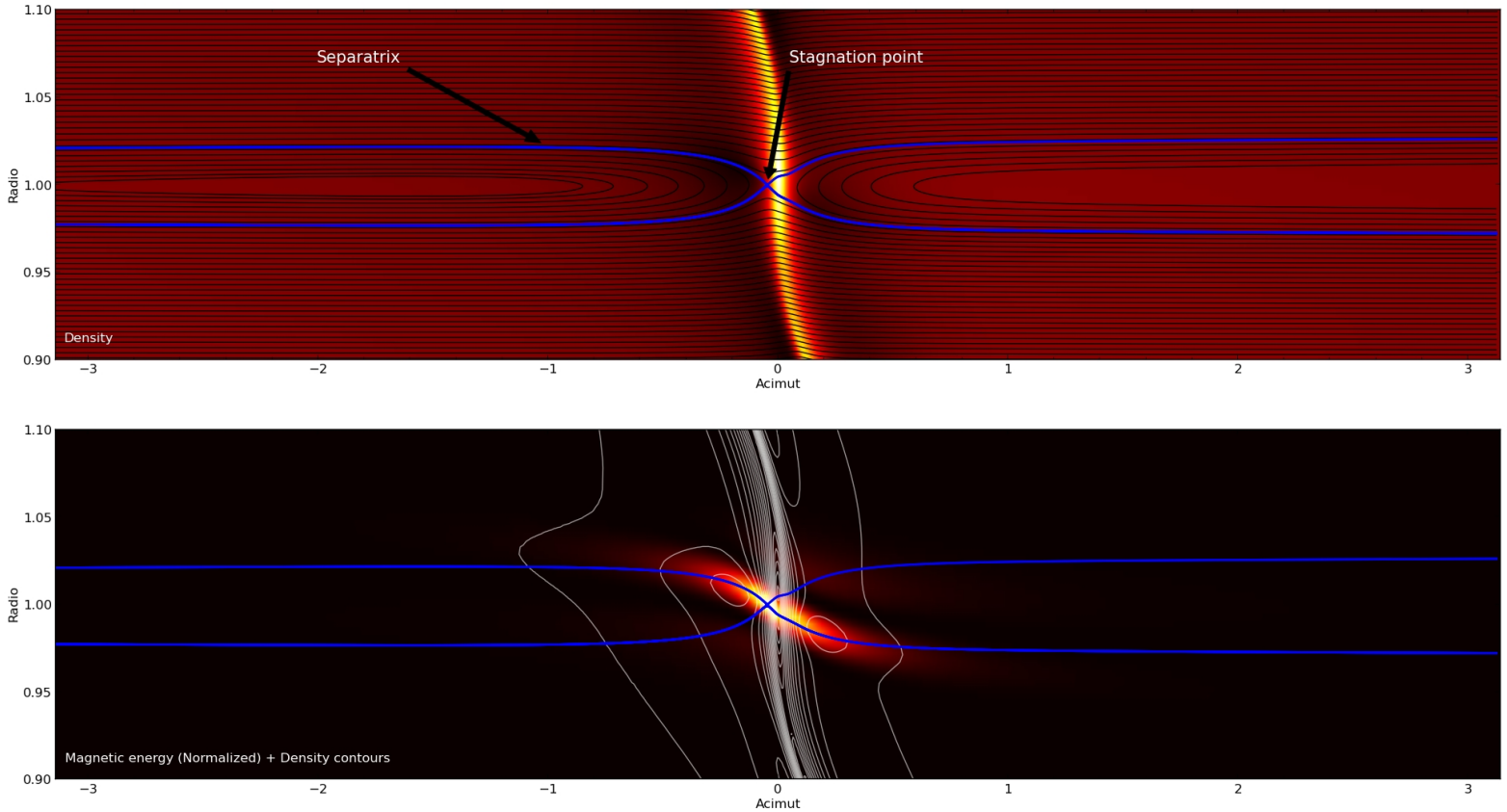
Recién comenzamos...

- Simulaciones con campo magnético (2D):





Recién comenzamos...



Gracias

Simulación de un planeta sumergido en un disco gaseoso.
Resolución: 8192x1536 – FARGO3D.
(para Kley et al. 2012)