8th Advanced School on SCIENTIFIC VISUALIZATION

Visualising the interplanetary medium: Turbulent heating of the solar wind plasma

Luigi Calori – l.calori@cineca.it
Giusy Muscianisi – g.muscianisi@cineca.it

SuperComputing Applications and Innovation Department

CINECA

SoHPC Program

- Summer of HPC is a PRACE programme that offers summer placements at HPC centres across Europe. Up to twenty top applicants from across Europe will be selected to participate. Participants will spend two months working on projects related to PRACE technical or industrial work to produce a visualisation or video. The programme will run from July 1st, to August 30th 2013 and will include a kick-off training week.

https://summerofhpc.prace-ri.eu/

CINECA SoHPC project ---

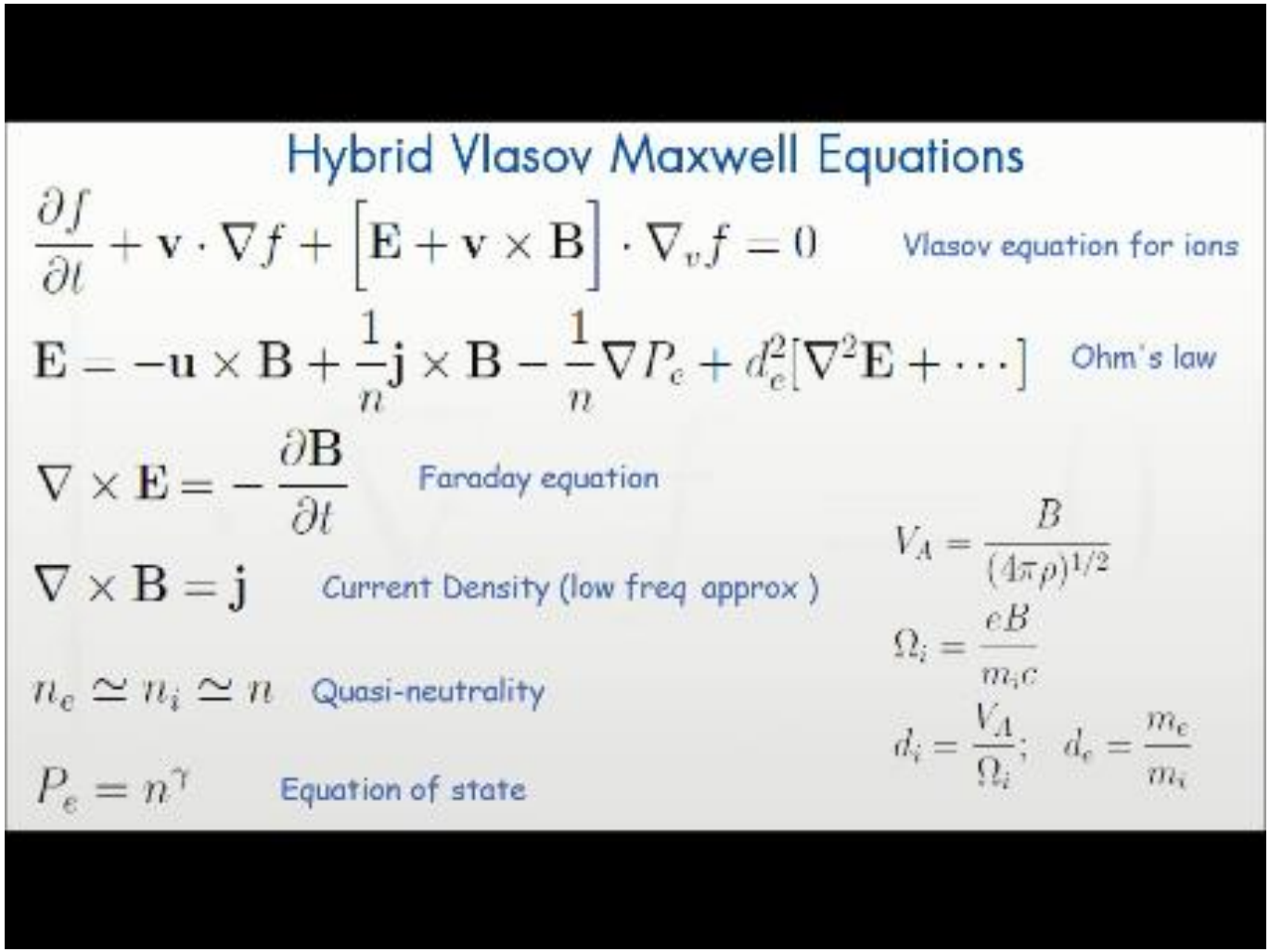
Visualisation of 3D-3V simulations of plasma turbulence

Numerical simulations are focused in particular on the analysis of the role of kinetic processes at play in the **turbulent cascade in the solar wind**, its relationship with explosive magnetic reconnection events and the evolution of the distribution of particle velocities.

Such kind of 3D-3V Vlasov simulations represent the first and unique (to date) attempt to provide a realistic interpretation of experimental measurements in space, using an Eulerian low-noise algorithm. This feature is crucial when analysing the turbulent spectral region at high frequencies where the energy level of the fluctuations is typically very low. Indeed, the artificial inclusion of numerical noise in this region could drastically affect the genuineness of the numerical results.

The analysis of the turbulent cascade at very short lengths (or high frequencies) in the solar wind is a subject of top priority within the scientific community of space plasma physics. In fact, one of the most puzzling aspects of the dynamics of the interplanetary medium consists in the empirical evidence that the solar wind is hotter than expected as an expanding gas: the cooling of the expanding solar wind is less efficient than it should be, then, how does the solar-wind energy turn into heat and keep it hot? **Understanding how energy from the Sun can be dissipated into heat in such a collision-free system represents a top priority in space physics.** The identification of the physical mechanism that replaces “energy dissipation” in a collisionless magnetized plasma and establishes the link between macroscopic and microscopic scales would open a new scenario of broad importance in the field of turbulence and space plasma heating. Scientists pointed out that the explanation of the empirical evidences mentioned above is related to the turbulent character of the solar wind plasma.

Hybrid Vlasov Maxwell Equations



3D-3V simulations

- Pure MPI code, 1000 MPI tasks
- Simulations run on FERMI (BG/Q system)

- 3D grid domain: 128 x 128 x 128
  - Physically, each side of the cube depends on the proton inertial lenght, so in solar wind in such simulations is

    19000km < L < 25000km.

- Time steps: 0 to 288, 0.5s < dt < 5s (0.5 omega_C)
  - The max time of the simulation depends on the proton cyclotron frequency, so in solar wind in such simulation, the max time is 144 s < tmax < 1440s

3D-3V simulations

- Output files (for each simulation on FERMI):

1) ASCII files with the time evolution of main fields:
   - Electric field, Magnetic field, Density field, etc. in x, y and z direction
   - Each field is written in a single file
   - Amount of data: 176 GB ASCII data

2) Proton velocity distribution at the last time step:
   - 1000 binary files, one per MPI task
   - Amount of data: 2.3 TB binary data

Visualization activity 1/2

Visualization tool: ParaView on PLX cluster

--- Time evolution of main fields

- Conversion from original ASCII files to binary VTK files readable from ParaView
- Visualization operation

--- 3D proton velocity distribution at the last time step

- Due to the dimension of the output files, look for a way to visualize on PLX, leaving the original data on FERMI
- Interactive visualization

Visualization activity 2/2

1) Creation of ParaView macros for:
   - Time evolution of main fields
   - 3D proton velocity distribution at the last time step

2) ffmpeg to create pieces of the video from png images

3) Blender tool to create the final video and audio (post-production)

Part I
Evolution of main fields

Post Processing 1/3

Original data file format (ASCII) for electric (E) and magnetic (B) fields

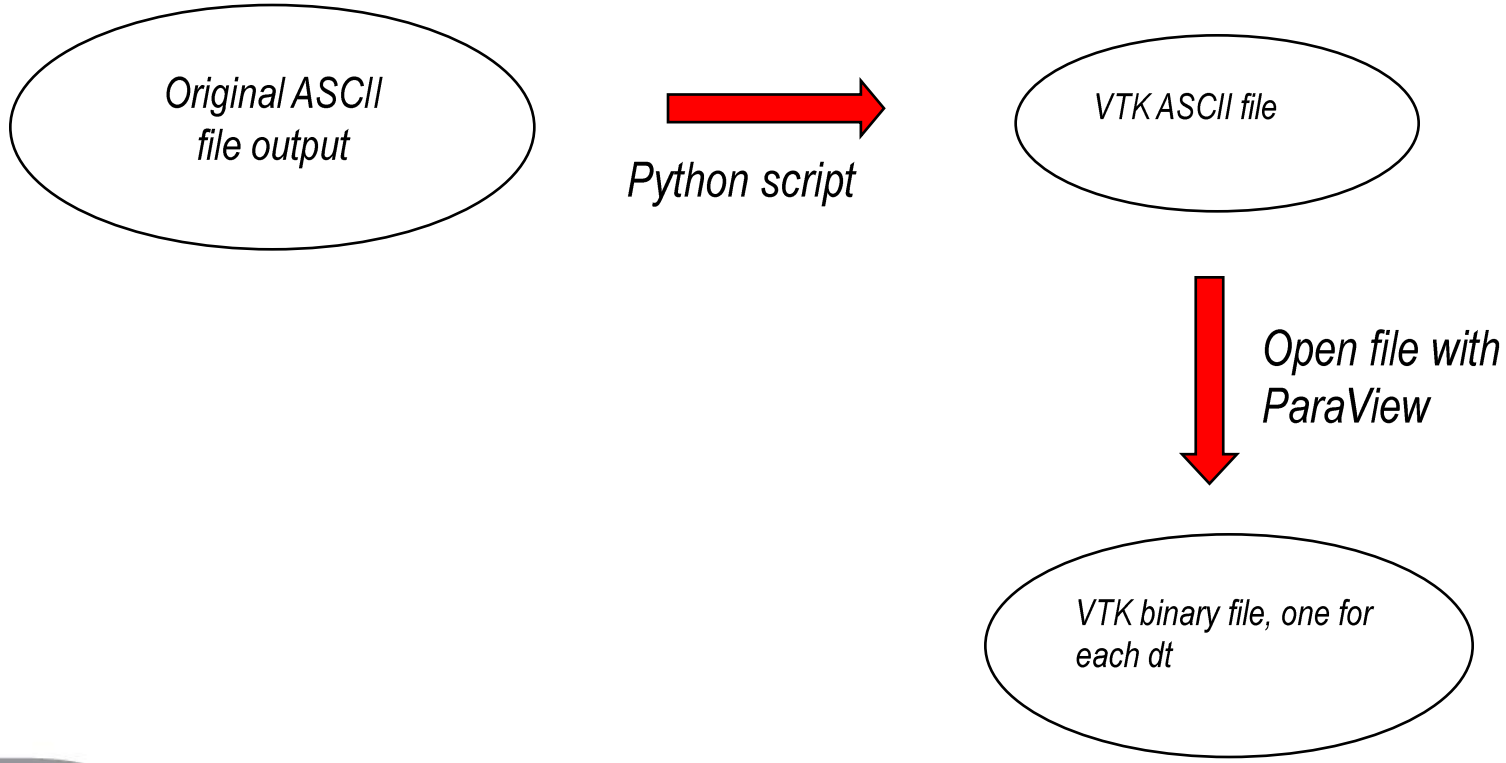
```
Time step        xdim     ydim    zdim
      Ex                    Ey                    Ez                Bx                By                Bz

0.0000E+00        128      128      128
  -0.2820825E-01 -0.3042702E+00       -0.826394E-02       0.4107692E-01     -0.3731446E-01      0.1233669E+01
  -0.5511381E-01 -0.2842225E+00       -0.129469E-01    0.4960669E-01 -0.6460796E-01       0.1208303E+01
  -0.8589098E-01          -0.2645525E+00   -0.168381E-01    0.6134776E-01 -0.9517050E-01          0.1182340E+01
  -0.1197602E+00     -0.2460799E+00   -0.1933972E-01  0.7621467E-01 -0.1280307E+00  0.1157116E+01
 .........

0.5000E+00        128      128      128
  -0.2983522E+00 -0.8451477E-01       0.7109056E-01   0.3672895E+00   -0.3274197E+00   0.1152190E+01
  -0.2755118E+00 -0.5734010E-01    0.7781404E-01   0.3960944E+00   -0.3116604E+00   0.1172771E+01
  -0.2489196E+00 -0.2734729E-01       0.8181442E-01   0.4237977E+00   -0.2901020E+00   0.1192431E+01
  -0.2204867E+00 0.4484720E-02        0.8301631E-01   0.4501191E+00   -0.2635049E+00   0.1209727E+01
.....……
```

Post Processing 2/3

Batch conversion in binary VTK file format (on PLX)



Post Processing 3/3

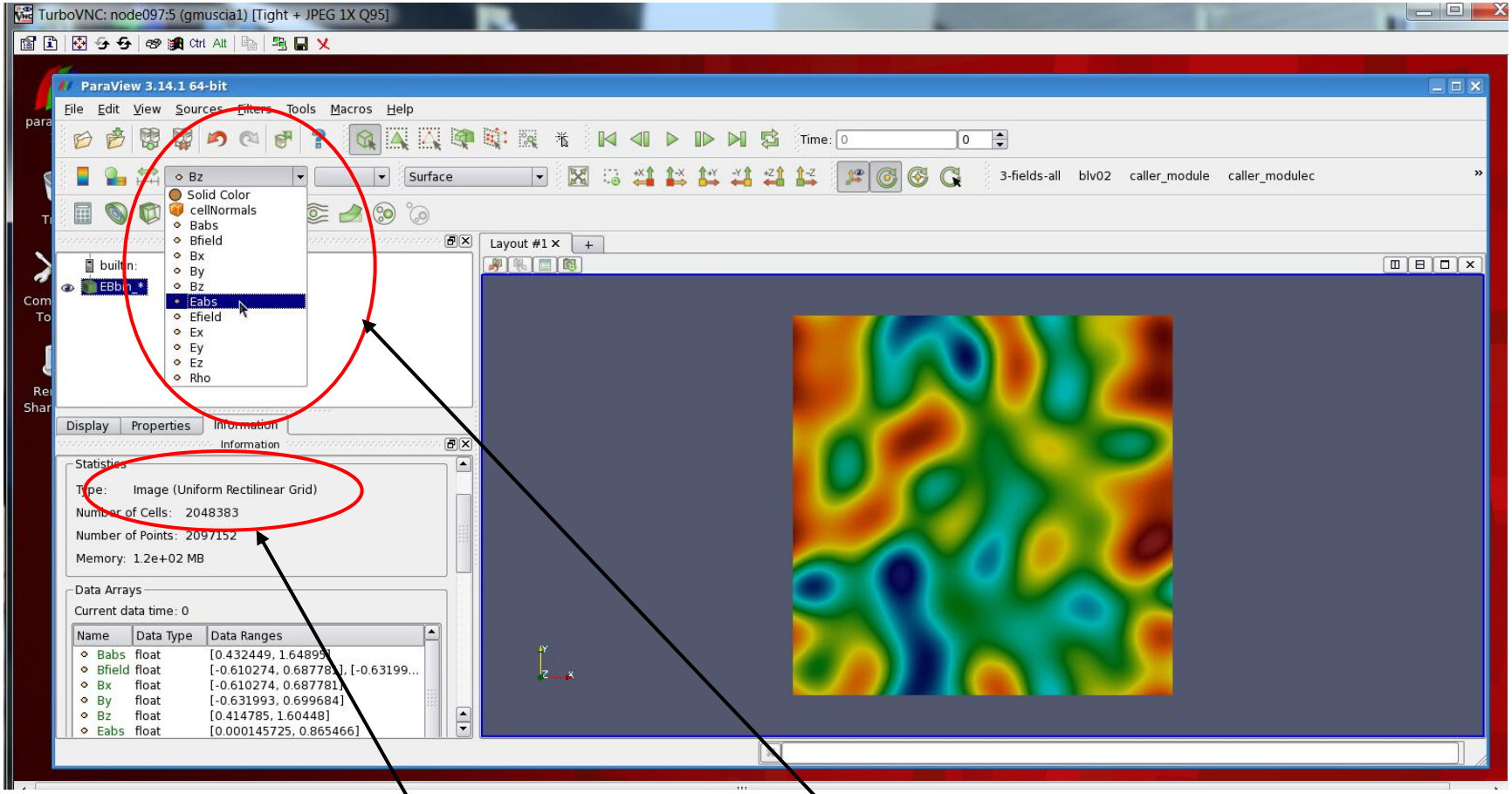
```
# vtk DataFile Version 3.0
vtk output
BINARY
DATASET STRUCTURED_POINTS
DIMENSIONS 128 128 128
SPACING 1 1 1
ORIGIN 1 1 1
POINT_DATA 2097152

SCALARS Eabs float
LOOKUP_TABLE default
… binary data

VECTORS Efield float
… binary data
```

```
FIELD FieldData 9
Babs 1 2097152 float
… binary data
Ex 1 2097152 float
… binary data
Ey 1 2097152 float
… binary data
Ez 1 2097152 float
… binary data
Bx 1 2097152 float
… binary data
By 1 2097152 float
… binary data
Bz 1 2097152 float
… binary data
Rho 1 2097152 float
… binary data
Bfield 1 2097152 float
… binary data
```

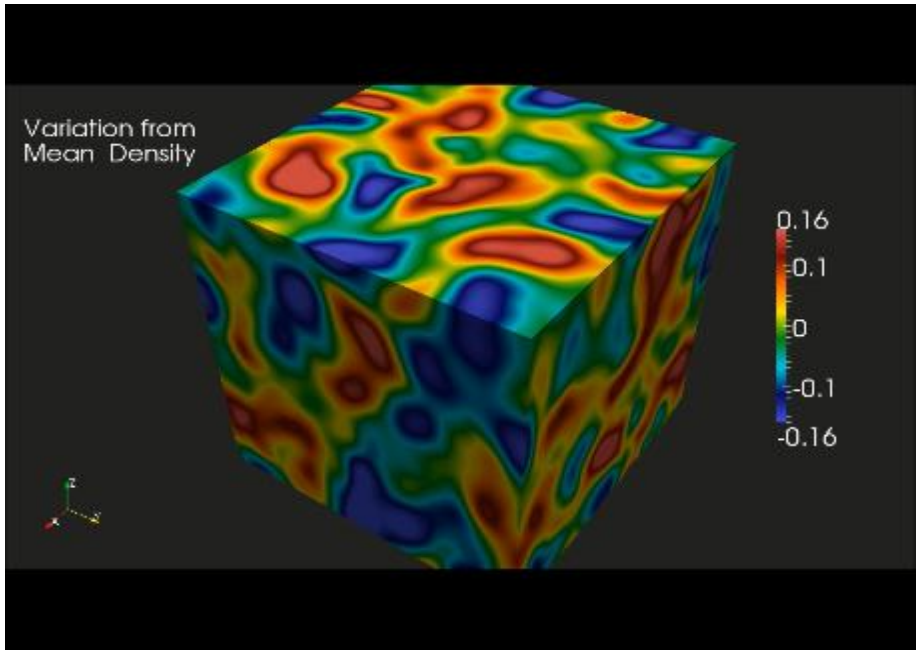
ParaView screenshot



Type: Image (Uniform Rectilinear Grid)

Available fields

Time evolution of density field



rhosurf.avi

Evolution of density field 1/3

- Load files

```
EBbin_ = LegacyVTKReader ( FileNames=['/EBbin_0.vtk', '/EBbin_1.vtk', …,
  '/EBbin_288.vtk'] )
```

- Select of “Rho” variable

```
a1_Rho_PVLookupTable = GetLookupTableForArray ( “Rho”, 1, RGBPoints=[ ],
  … )
a1_Rho_PiecewiseFunction = CreatePiecewiseFuncion ( )
```

- Select of variable representation

```
DataRepresentation1 = Show()
DataRepresentation1.ScalarOpacityFunction = a1_Rho_PiecewiseFunction
DataRepresentation1.LookupTable = a1_Rho_PVLookupTable
DataRepresentation1.Representation = 'Surface'
```

Evolution of density field 2/3

- Set of:
  - camera position and background color

```
RenderView1=GetRenderView()
RenderView1.{CameraViewUp; CameraPosition; CameraClippingRange;
    CameraFocalPoint; Background; … }
```

  - color map

```
ScalarBarWidgetRepresentation1 = CreateScalarBar( Title='Rho',
    LabelFontSize=12, Enabled=1, ... )
GetRenderView().Representations.append(ScalarBarWidgetRepresentation
    1)
```

  - Title

```
Text1 = Text()
Text1.Text = 'Variation from \nMean Density'
SetActiveSource(Text1)
DataRepresentation1 = Show()
```

Evolution of density field 3/3

- Render and save images for each time step

```
AnimationScene1.AnimationTime=0.0
for i in range(288):
    AnimationScene1.AnimationTime = i
    RenderView1.ViewTime = i
    RenderView1.UseCache = 1
    Render()
    WriteImage ('path_where_store_image')
```

All these operations are inserted in a **ParaView macro.**

(mean_density.py)

Time evolution of density field
Final video

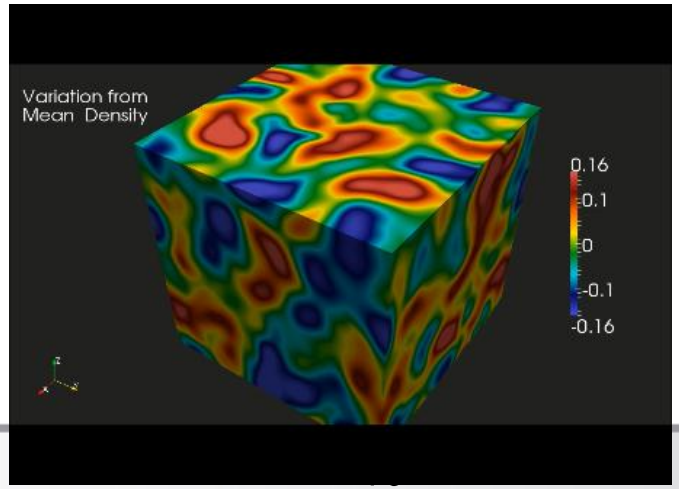
This animation shows the time evolution of the density fluctuations as the turbulent cascade develops and transfers energy from large to short wavelengths.

In the simulation, the cascade is initially triggered by injecting energy into the system in the form of large scale fluctuations (to be specific the typical scale of the initial perturbations is a little larger than the proton inertial length).

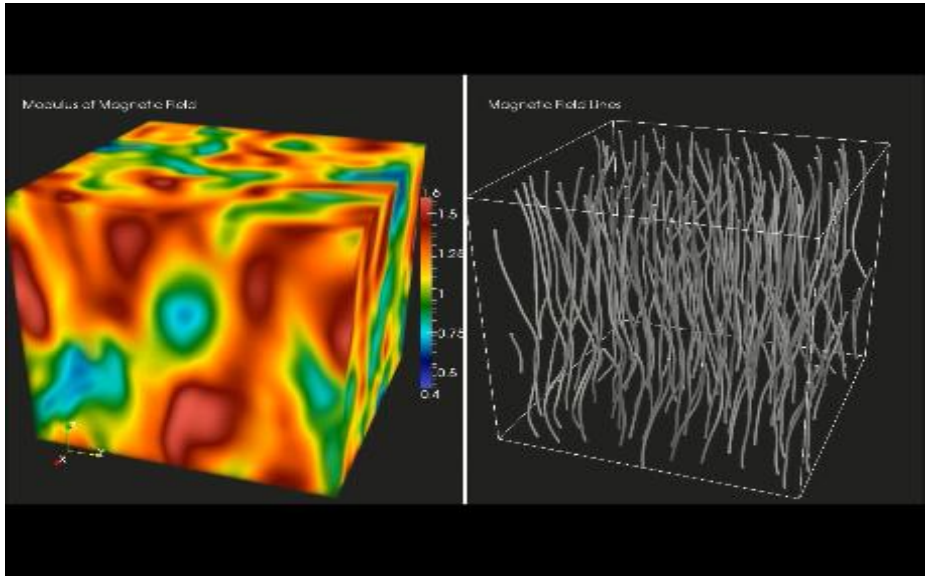
The initial condition of the simulation looks like a sea of large vortices and swirls. As is shown in this video, over time, these structures interact nonlinearly as they attract, repel and merge. As a result of these nonlinear interactions, the vortical structures fragment giving rise to new vortices of smaller and smaller size.

This is the so-called turbulent cascade.

rhosurf.avi



Time evolution of - modulus of magnetic field - magnetic field lines



Bvolandstreams.avi

Modulus of magnetic field

1) Load files
2) Select of “Babs” variable
3) Set of:
   - color map
   - background color
   - camera position
   - title
4) Save images for each time step

These operations are inserted in a **ParaView macro.**

Magnetic Field lines

1) Load files
2) Select of “Babs” variable
3) Select of filter to create the field lines
   StreamTracerWithCustomSource1
4) Set of:
   - color map
   - background color
   - camera position
   - title
5) Save of the images related to each time step

These operations are inserted in a **ParaView macro.**

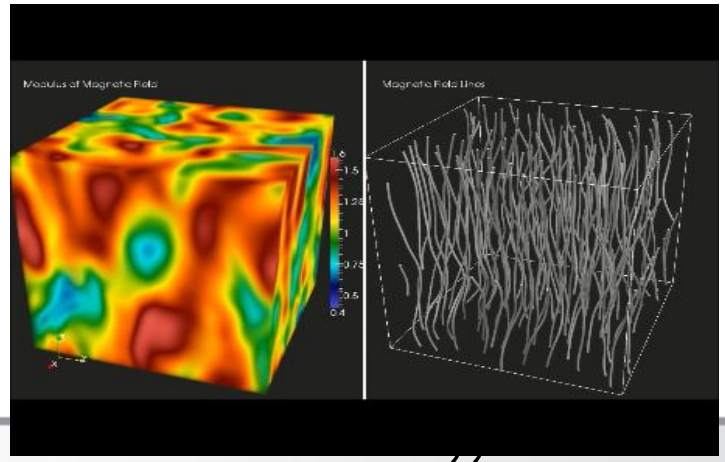
Time evolution of magnetic field Final video

This animation shows the time evolution of the magnetic field during the development of the turbulent cascade.

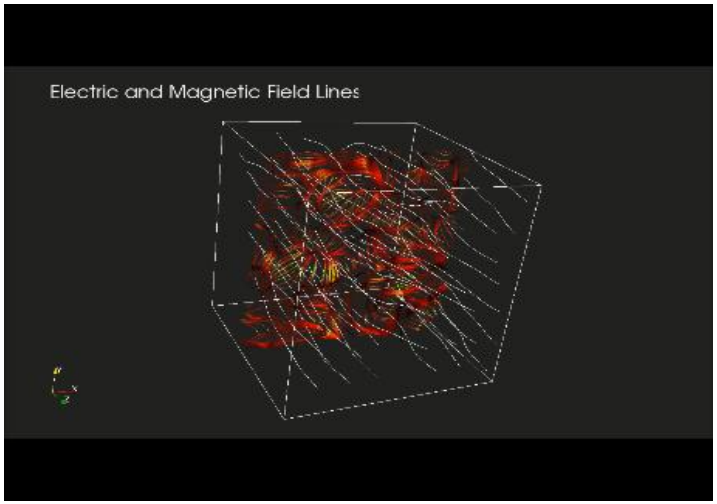
This turbulence leads to the generation of coherent structures and magnetic islands which are not static, but evolve in time. As they interact nonlinearly the magnetic field between these islands becomes very intense and local magnetic reconnection events can occur, breaking the topology of the field.

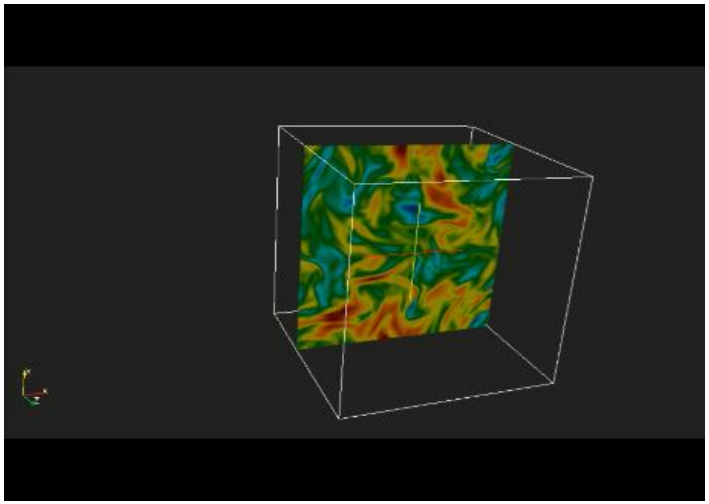
The presence of these high magnetic stress regions is a signature of the intermittent nature of the magnetic field.

Bvolandstreams.avi



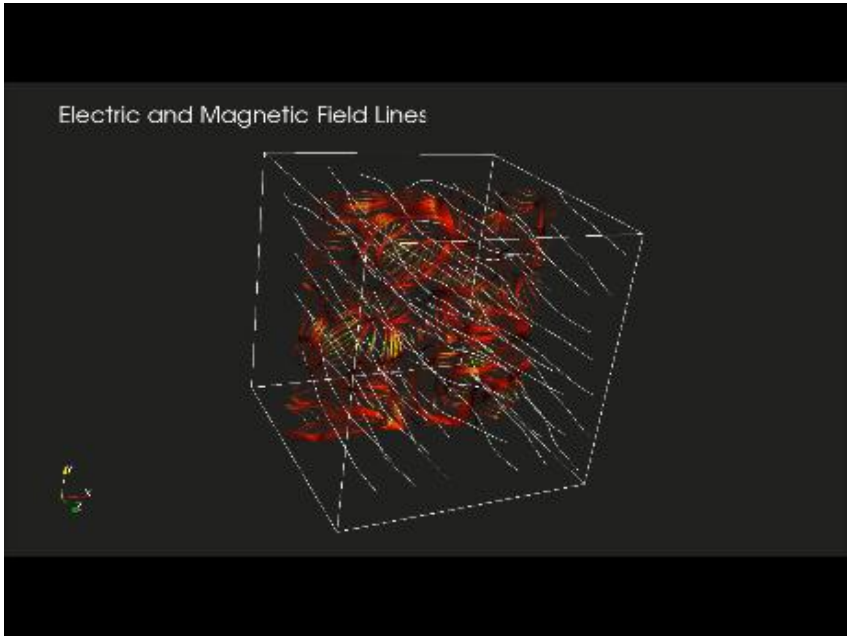
Stationary field representations





Stationary field representations

streamsweep.py



fieldsweep.avi

Time step: 144

A plane go on the 128 point grid

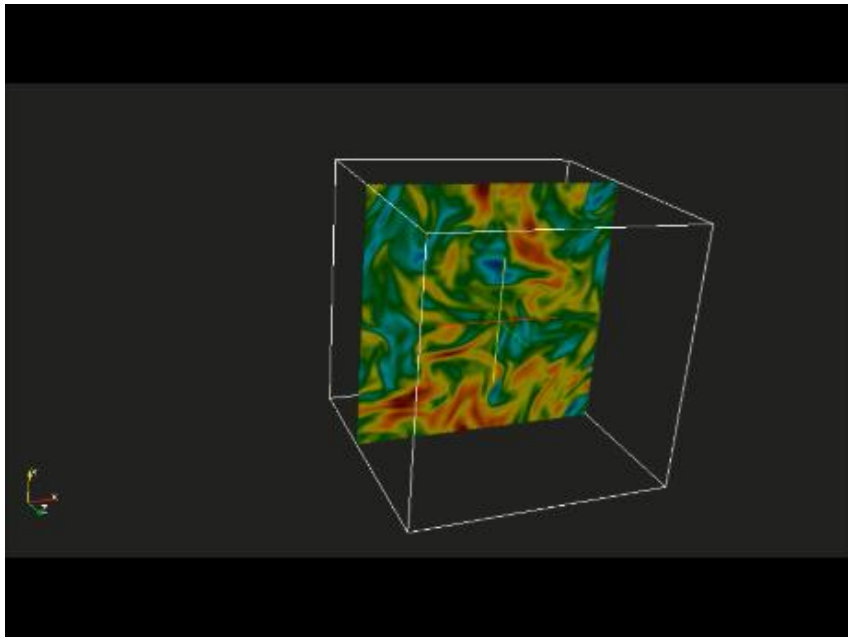
Stationary field representations

rhoslice.py

Time step: 144

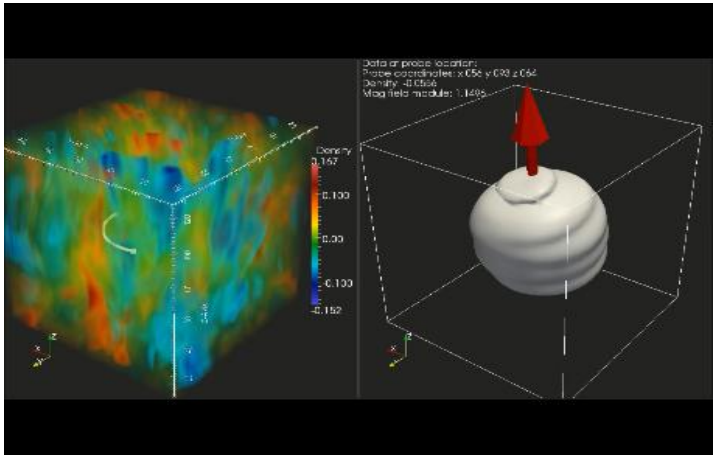
Select “slice” filter

The slice go on the 128 point grid



rhoslice.avi

Part II
3D proton velocity distribution



proton-vel-distrib.avi

Distribution function 1/2

- Distribution function is defined in the phase space, 6 dims :

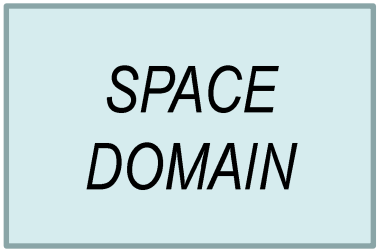
  f(x,y,z, vx,vy,vz)

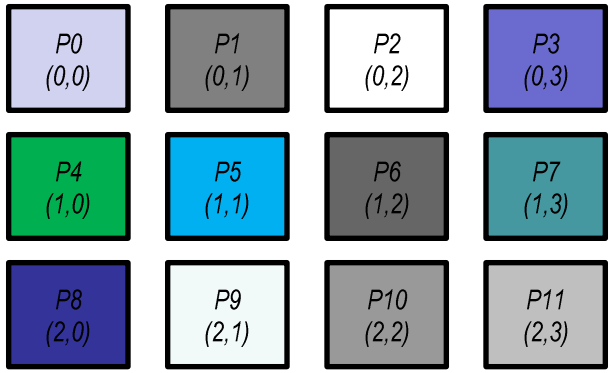
- It is possible to visualize it both in space domain “x,y,z” or in velocity “vx, vy, vz”
  - Total space domain, cartesian grid: 128 x 128 x 128
  - Total velocity domain, cartesian grid: 51 x 51 x 51

- Dumpend on files only in the last time step

Distribution function 2/2

- Written in parallel from each MPI task, in binary format.
- Each MPI task has
  - a part of the total space domain, a little cube of about 12 x 12 x 12 points
  - all the velocity grid, 51^3 points
  - dimension of each binary file: **2.4 GB**





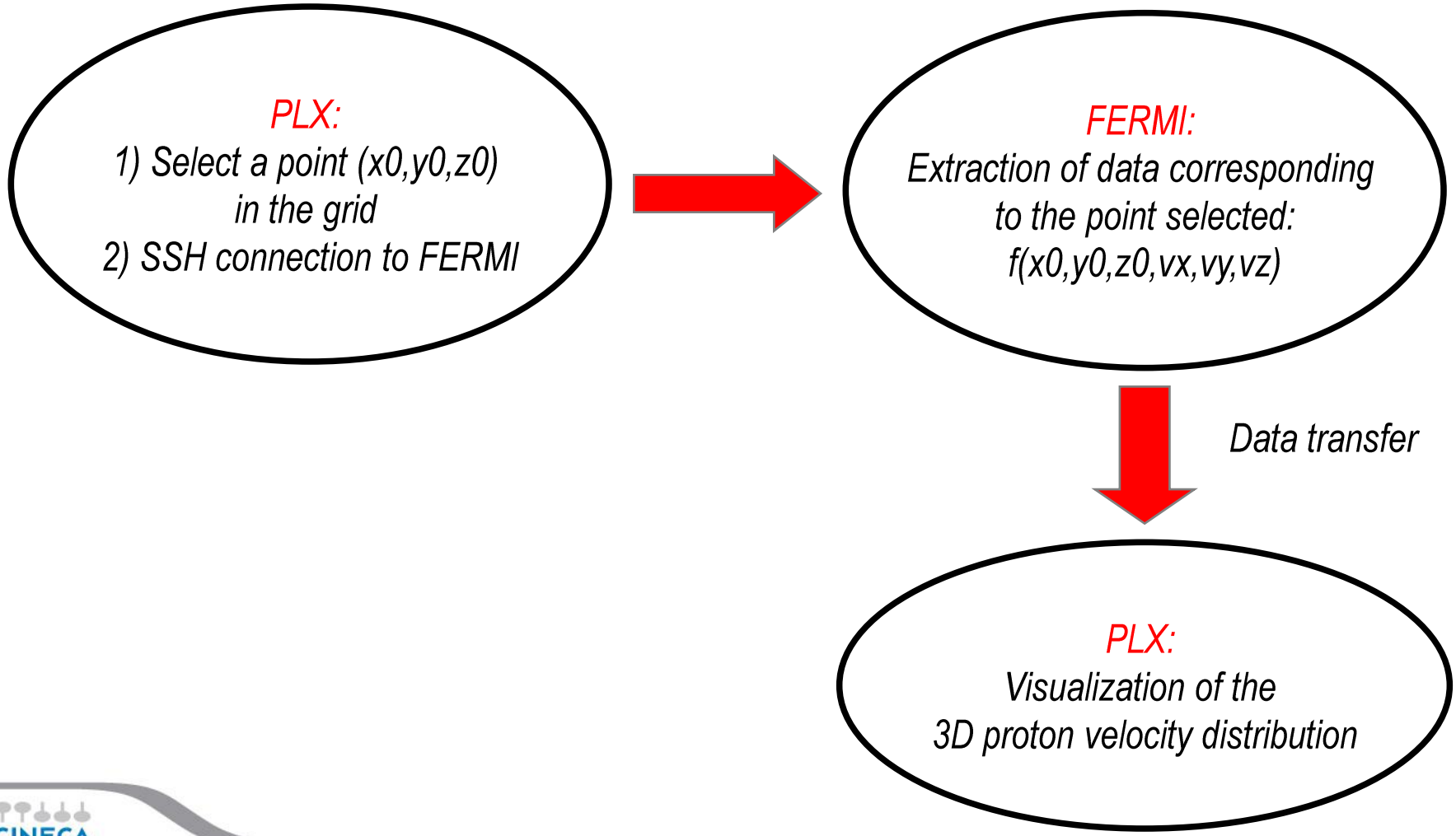
Distribution function : how to visualize …

- Select a point in the total space domain (x0,y0,z0)
- Locate the MPI task owner of such point

- Estract values of distribution function in such point:
  f(x0,y0,z0, vx,vy,vz) -25 < {vx, vy, vz} < +25

- Visualize the 3D proton velocity distribution
  - The file dimension is **2.5 MB**

Transfer data live from FERMI to PLX



Transfer data live from FERMI to PLX --- file needed ---

FERMI:

- output binary files
- domain decomposition file
- python code to extract data from bynary files (write_file2.py)

PLX:

- Python macros:
  - probe_macro.py
  - load_module.py
  - caller_module.py

Visualization of 3D proton velocity distribution

Open ParaView on PLX and ….

- Start the macro probe_macro
- when the pop up window asking user and pass open, enter your Fermi credentials and log in
- after around 10 sec, in the right window it would show up an iso-surface of the distribution function velocity sample at a default spatial point (4,7,5)
- select ProbeLocation1 from the pipeline
- change position of the probe location on the left window by dragging the white square -cross
- press Apply in the property tab
- run probe_macro again and wait around 10 sec to see the iso-surface plot update

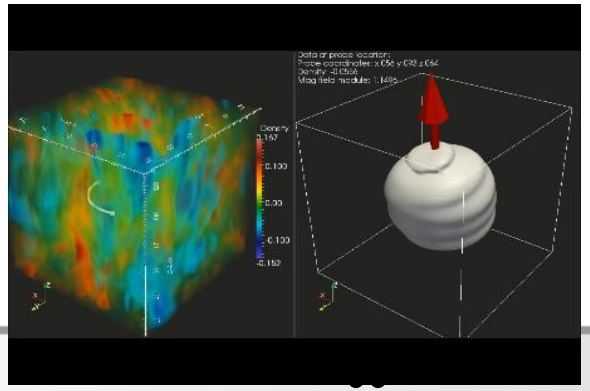
Change the point and repeat the operations as long as you want

3D proton velocity distribution

This sequence shows how the proton velocity distribution (or PVD) looks in a turbulent medium. In the box on the left, the density fluctuations are represented at the end of the simulation when the maximum level of turbulence has been reached. Many islands and swirls can be recognized in this surface plot. A cursor moves within the three-dimensional turbulent pattern.

For each spatial position explored by the cursor, the corresponding three-dimensional PVD is displayed in the box on the right. It is clearly visible that the shape of the PVD appears distorted with respect to the typical Maxwellian shape of thermodynamic equilibrium (ie. a sphere in the three-dimensional velocity space), and more resembles a deformed potato. Moreover, the shape of the PVD changes according to the spatial position of the cursor. This evidence supports the idea that kinetic effects, responsible for driving the system far from thermodynamic equilibrium, are not homogeneously distributed in space, but act differently depending on the local spatial features of the turbulent pattern.

proton-vel-distrib.avi



Final video

- Audio creation
- Movie collection

- **Tool used: BLENDER**

http://www.blender.org/

Final video

Movie_HD.avi

Thanks to …

SoHPC students:

- Jasper Kursk – University of Tart
- Peter Thompson – University of Manchester

Scientific supervisor

- Francesco Valentini – University of Calabria

Post production

- Daniele De Luca – CINECA

Thanks for the attention!!!

QUESTIONS ???