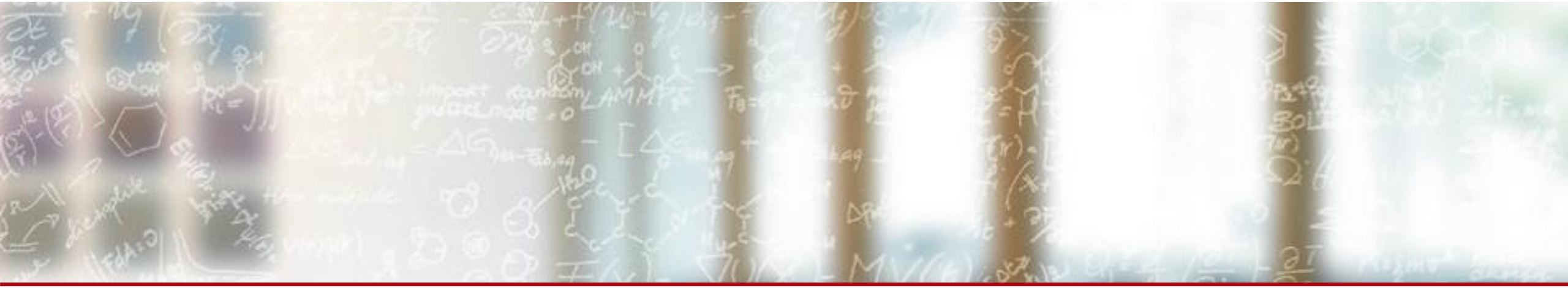
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



Scientific visualization

PDC Summer School 2023

Jean M. Favre, CSCS

August 24, 2023

Preamble 1

- We will use two different methods of using/demonstrating visualization in an HPC context
  - On your desktop with small data
  - In Dardel, with small and big data
- Paraview can be driven both by a Python script, and by interactive GUI-based actions (understand the diffs between *paraview, pvpython, pvserver, pvbatch*)

- Thus, you should clone the repo on both sides to see the exercise/demo file, e.g

  git clone https://github.com/jfavre/PDC-SummerSchool Visualization

- In my lecture notes, all examples for exercises should be written in green, i.e. pvSphere.py. You should find them in the git repo.

Preamble 2

- No training dedicated to HPC I/O in this summer school
- Difficult to know what is our common understanding of file I/O and file formats

- Most demonstration/exercises will use auto-generated data

Outline of the day

- Scientific data analysis and visualization (1hr)
  - ParaView hands-on lab session
- Parallel data visualization (1hr)
  - ParaView hands-on lab session for parallel data visualisation

- Because of the limited time, I decided to focus on a single application (ParaView), to avoid context switching
- ParaView is open-source, available everywhere on earth, and a *de-facto* leader in HPC visualization apps.
- The comprehensive ParaView guide is here:

https://docs.paraview.org/en/latest/UsersGuide/index.html

Definition, Data dimensions

Information Visualization is the study of visual representations of abstract data to reinforce human cognition. The abstract data include both numerical and non-numerical data, such as text and geographic information. However, information visualization differs from *Scientific Visualization*.

"It is *Information Visualization*, when the spatial representation is chosen, and it is *Scientific Visualization*, when the spatial representation is given“

The X,Y,Z,Time space will be our natural basis, and we will deal with grid-less or gridded data.

Grid types

Grid Types. The VTK Data Model

https://docs.paraview.org/en/latest/UsersGuide/understandingData.html#

Take ParaView's ProgrammableSource and list its possible output types:

pvpython

```
>>> from paraview.simple import *
>>> a = ProgrammableSource()
>>> a.OutputDataSetType.Available
['vtkPolyData', 'vtkStructuredGrid', 'vtkRectilinearGrid', 'vtkUnstructuredGrid',
'vtkImageData', 'vtkMultiblockDataSet', 'vtkHierarchicalBoxDataSet', 'vtkTable',
'vtkMolecule', 'vtkPartitionedDataSet', 'vtkPartitionedDataSetCollection']
```

Grid Types. The VTK Data Model

It is far more important to know the mesh type a reader will produce….

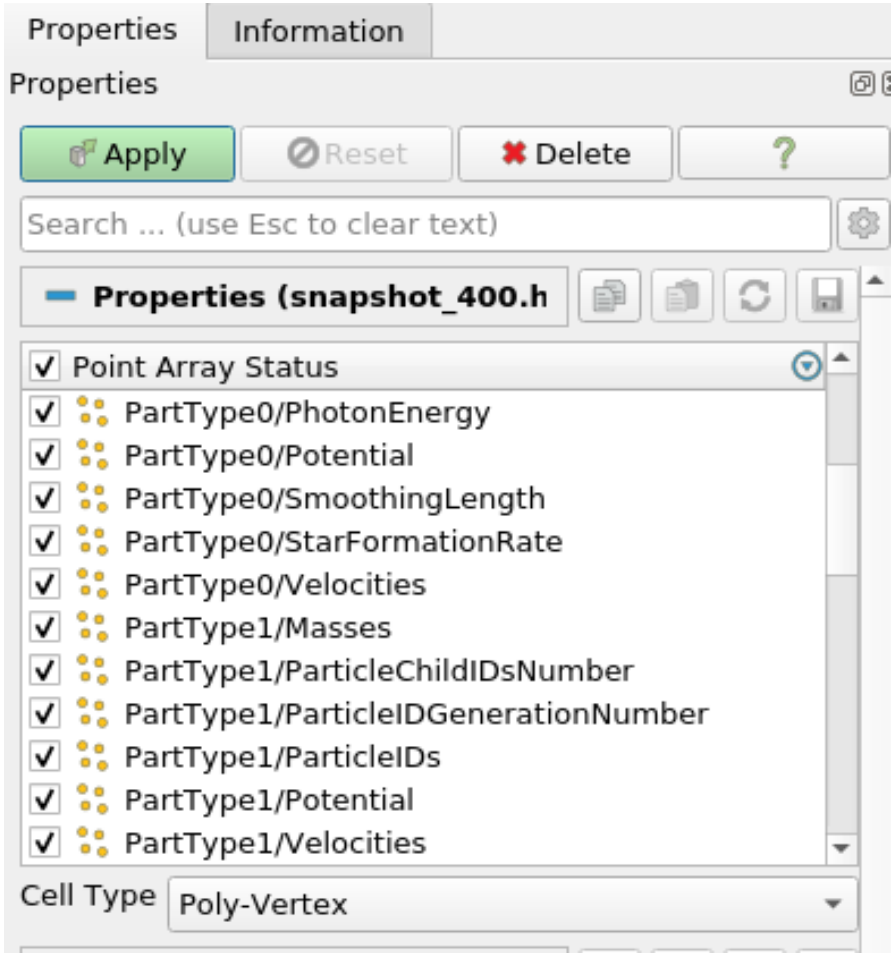
['vtkPolyData', 'vtkStructuredGrid', 'vtkRectilinearGrid', 'vtkUnstructuredGrid', 'vtkImageData', 'vtkMultiblockDataSet', 'vtkHierarchicalBoxDataSet', 'vtkTable', 'vtkMolecule']

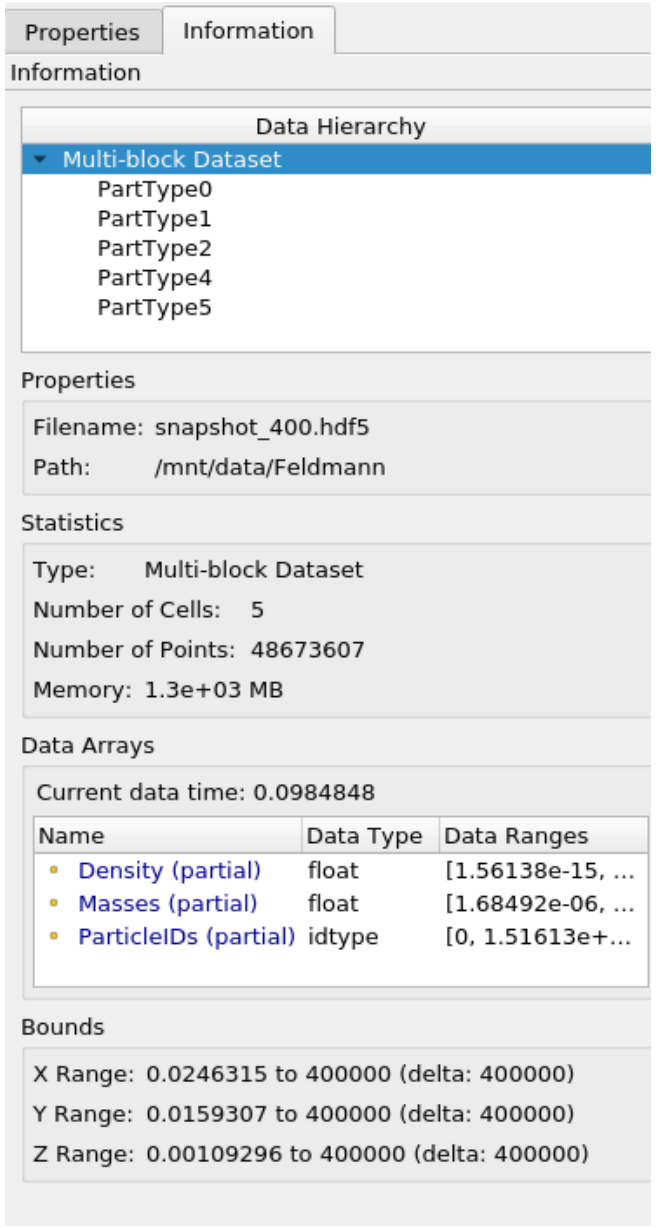
…rather than its specific name

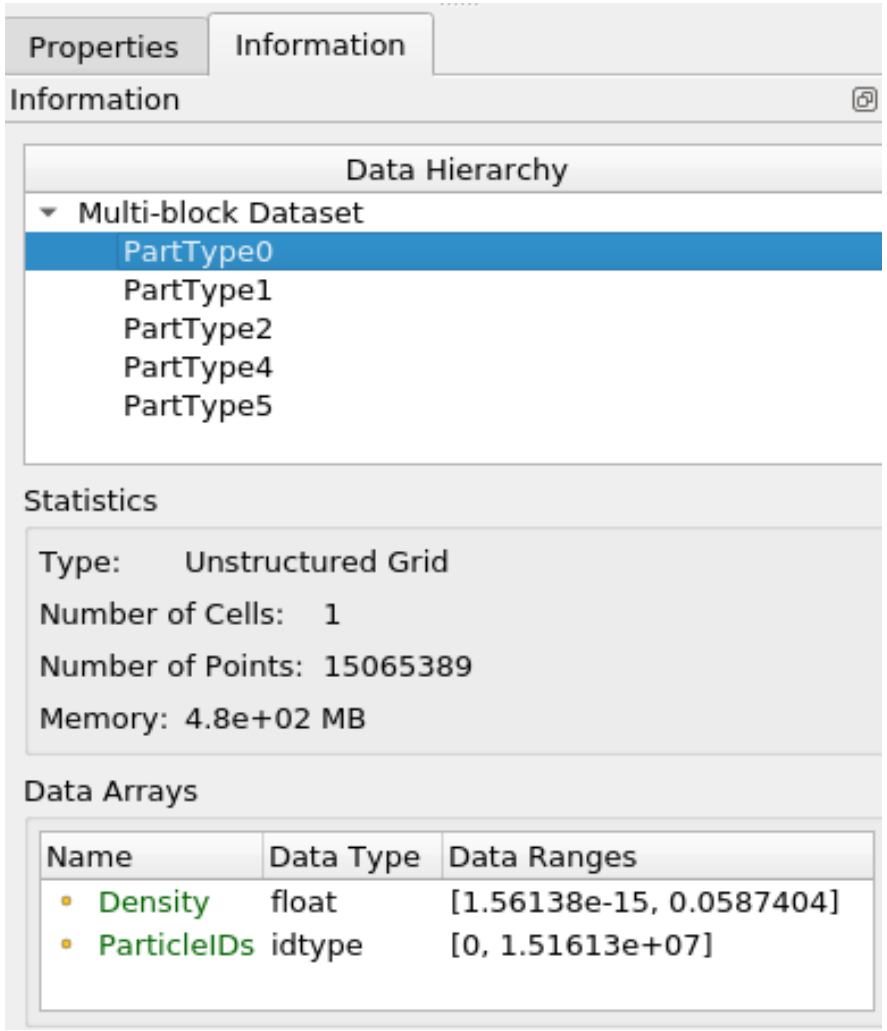
Said in a different manner “*you can encode the same data in multiple data formats*”

Implementation details will be important for time-series and parallel support

ParaView's look at data







Grid Types…and filtering

Each mesh type will have a corresponding subset of filters that can be applied to it.

Example:

Mesh Type	Can Use ExtractSubset ?
vtkPolyData	No
vtkStructuredGrid	Yes
vtkUnstructuredGrid	No
vtkImageData	Yes

Multiple view representations

- https://docs.paraview.org/en/latest/UsersGuide/displayingData.html#id3

Scalar field visualization

A quick look at the classic visualization methods

- Slicing
- Clipping
- Iso-contouring
- glyphing
- Volume rendering

Exercise

- I suggest to use two different (internal) data sources in the ParaView GUI
  - Fast Uniform Grid
  - Wavelet
- Try out all techniques introduced above
- Add a “tetrahedralize” filter, and try again

Query-ing/filtering data

ParaView’s Find Data menu

- Demonstrations

- Limitations in version <= 5.11
- The Python commands to reproduce a data selection are not saved in ParaView’s session files. Following in the next 3 slides are examples of what you would have to do…

Query-based filtering examples

https://gitlab.kitware.com/paraview/paraview/blob/master/Wrapping/Python/paraview/selection.py

```
qs1="rho > 1e-05"
sel1 = QuerySelect(QueryString=qs1, Source=mergeBlocks1)
extractSelection1 = ExtractSelection(Input=mergeBlocks1)

rep1 = Show(extractSelection1)
rep1.Representation = 'Points'
ColorBy(rep1, ('POINTS','rho'))
```

Query-based filtering examples

points is the array of coordinates. We select the 0-th coordinate < -12.15 AND rho > 1e-5

```
Qs2 = "np.logical_and(rho > 1e-05, points[:,0] < -12.15)"
sel2 = QuerySelect(QueryString=Qs2, Source=mergeBlocks1)
extractSelection2 = ExtractSelection(registrationName='ExtractSelection2', Input=mergeBlocks1)

rep2 = Show(extractSelection2)
rep2.Representation = 'Points'
ColorBy(rep2, ('POINTS','rho'))
```

Query-based filtering examples

```
# points is the array of coordinates. We select particles within a given Radius and Center


#Center = [-11.1431, 3.97869, -0.173805]; Radius = 10
Qs3 = "mag(points - [-11.1431, 3.97869, -0.173805]) < 10"
sel3 = QuerySelect(QueryString=Qs3, Source=mergeBlocks1)
extractSelection3 = ExtractSelection(registrationName='ExtractSelection3', Input=mergeBlocks1)



rep3 = Show(extractSelection3)
rep3.Representation = 'Points'
ColorBy(rep3, ('POINTS','rho'))
```

Query-based filtering

https://gitlab.kitware.com/paraview/paraview/blob/master/Wrapping/Python/paraview/selection.py

Other selections:

SelectThresholds(Thresholds=[-338000, 0], ArrayName= ='Potential', Source=GetActiveSource())

SelectIDs(IDs=[i for i in range(100000, 1100000)], , Source=GetActiveSource())

Vector field visualization

Vector field visualization

- Use a scalar field visualization technique
- Use glyphs representations (oriented arrows in the direction of the field)
- Use lines tangent to the vector field

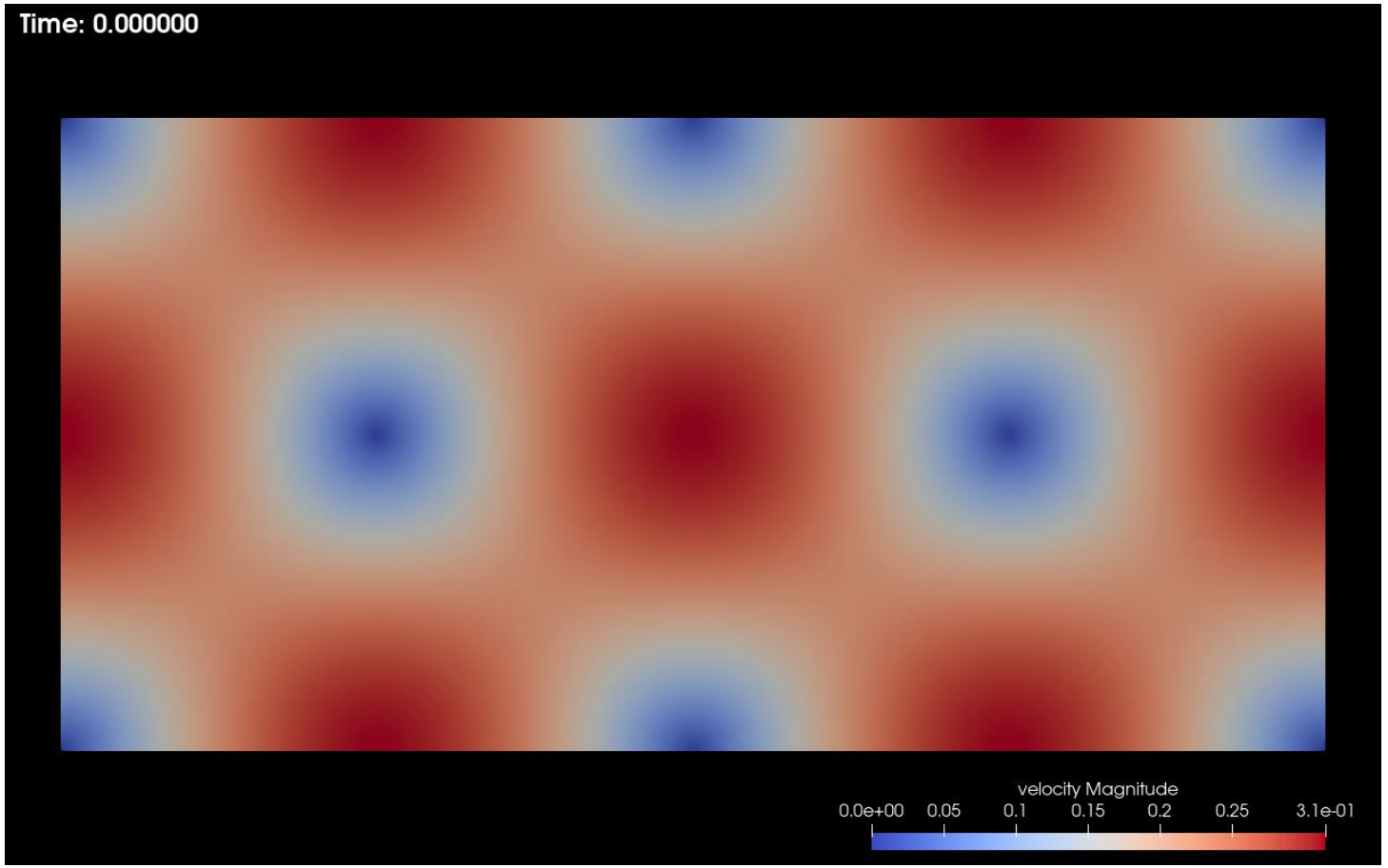
Time-dependent data:

- Use particle traces
- Use pathlines

Vector field visualization (1)

- Shade the vector field by its magnitude

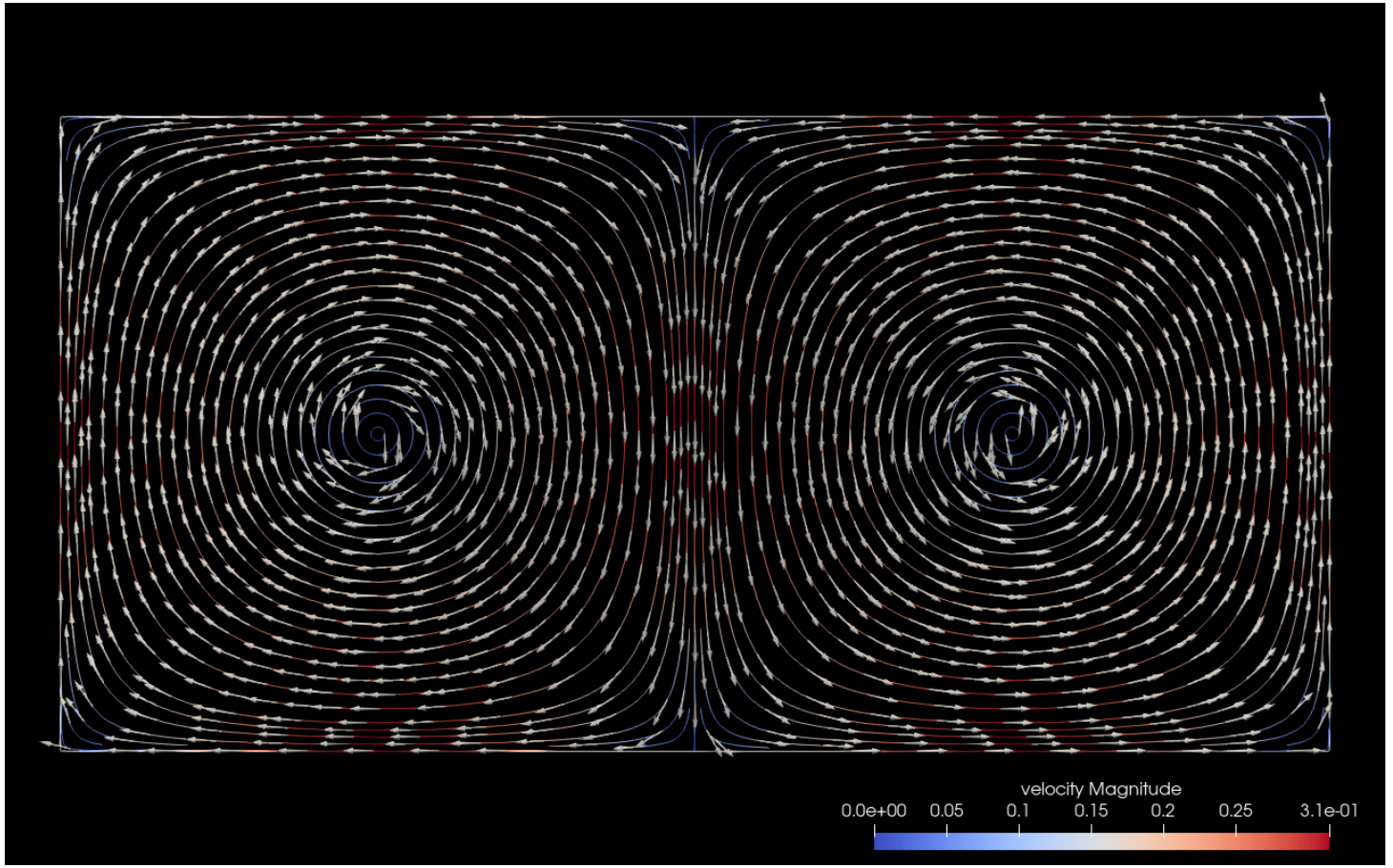
- Missing cues?
  - direction



Vector field visualization (2)

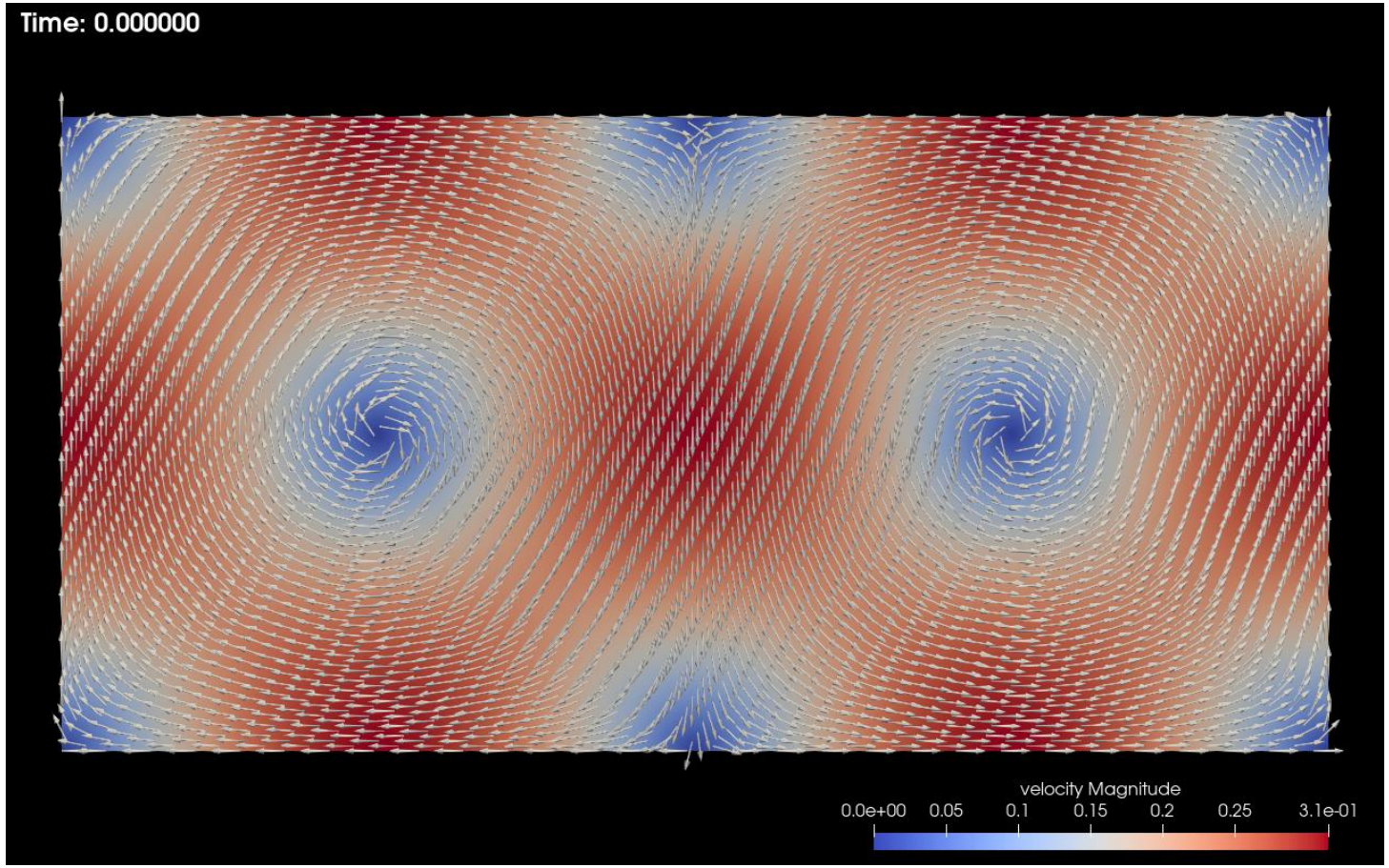
- Draw arrows oriented in the direction of the field

- Issues?
  - Make the density of arrows dependent on the zoom ration
- Difficulties?
  - Can be too complicated in 3D



Vector field visualization (2)

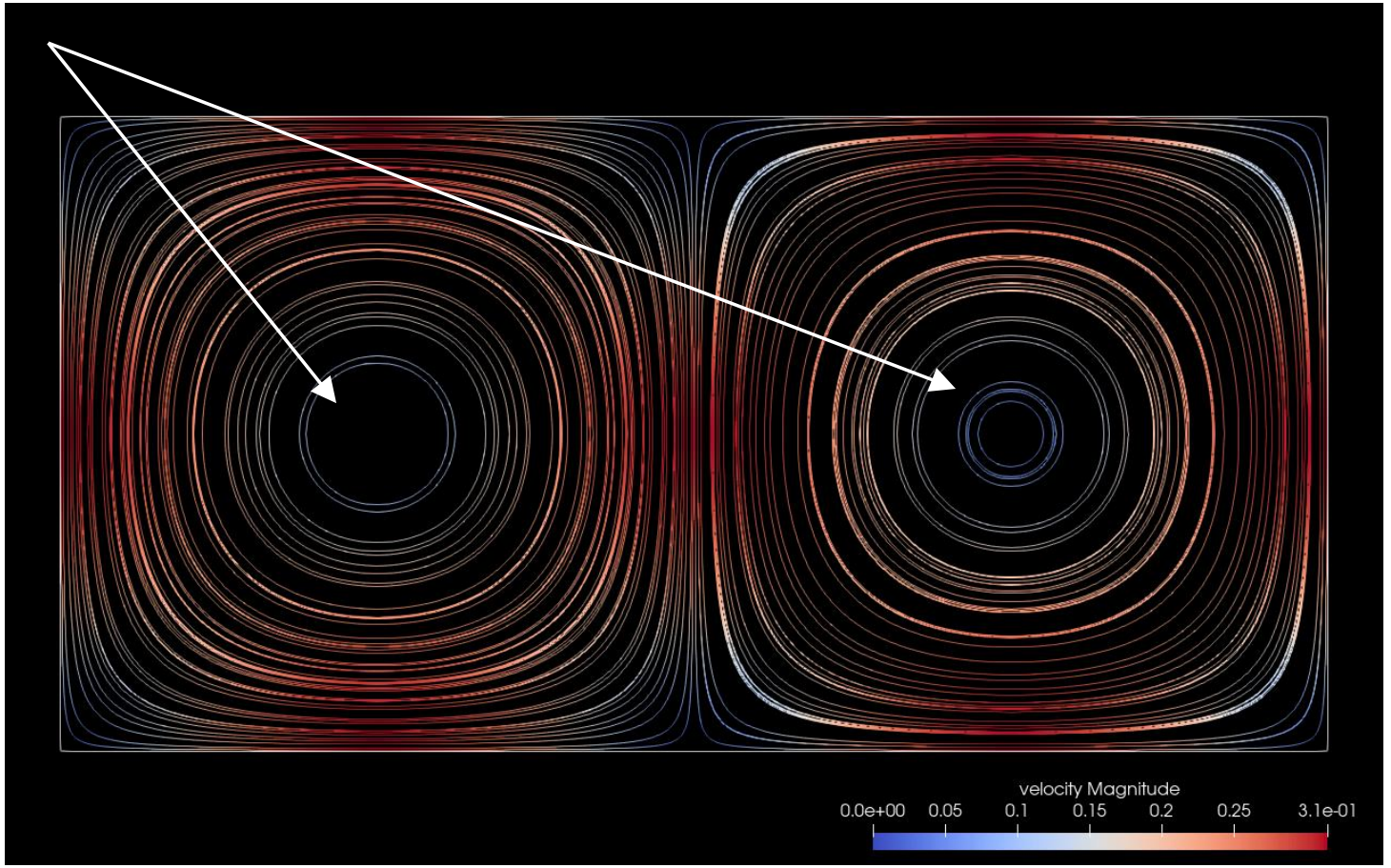
- Apply both techniques seen earlier



Vector field visualization (2)

- Draw streamlines tangent to the vector field

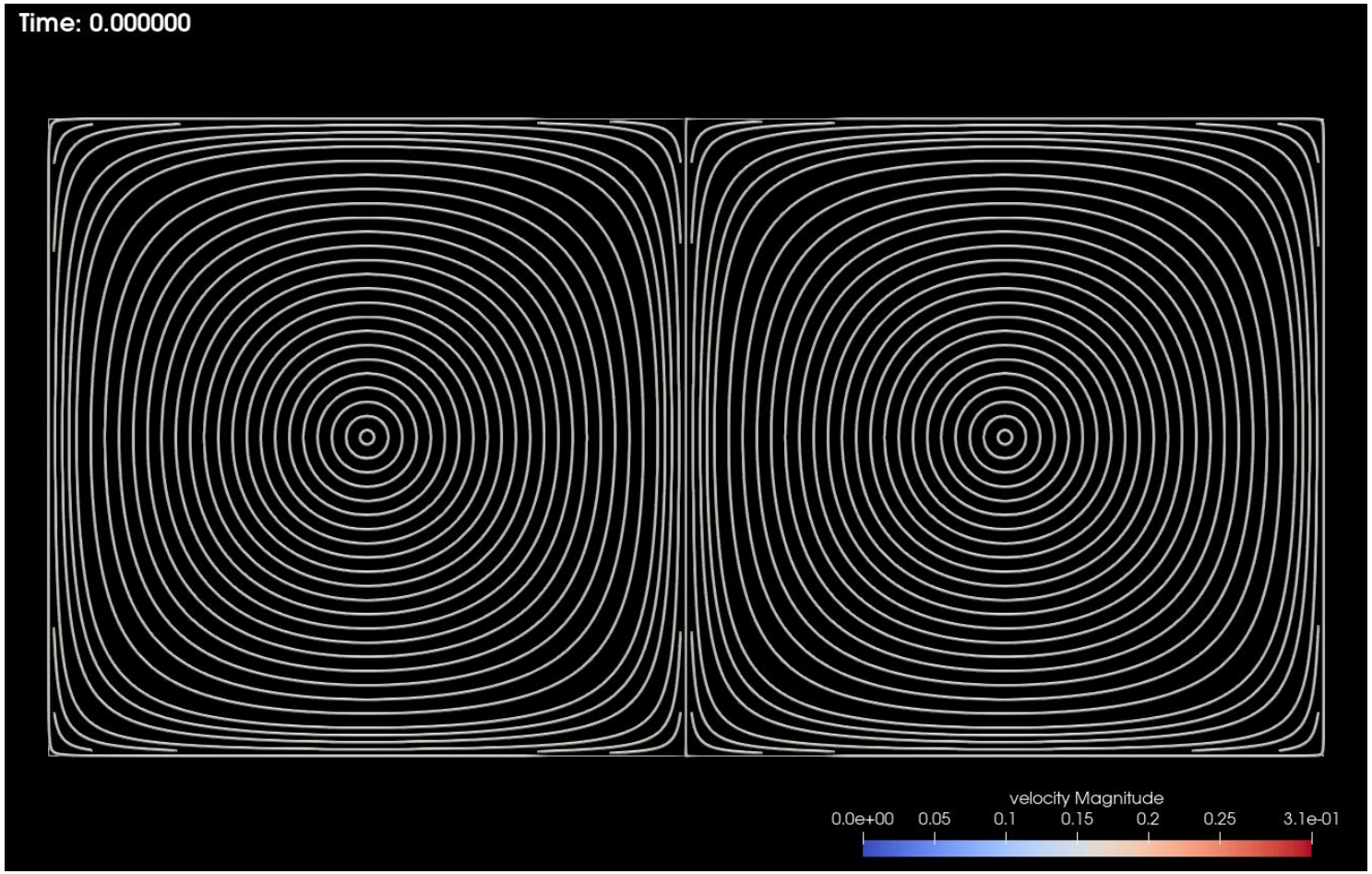
- Difficulties?
  - too dense, or too sparse



Vector field visualization (2)

- Draw **evenly-spaced** streamlines tangent to the vector field

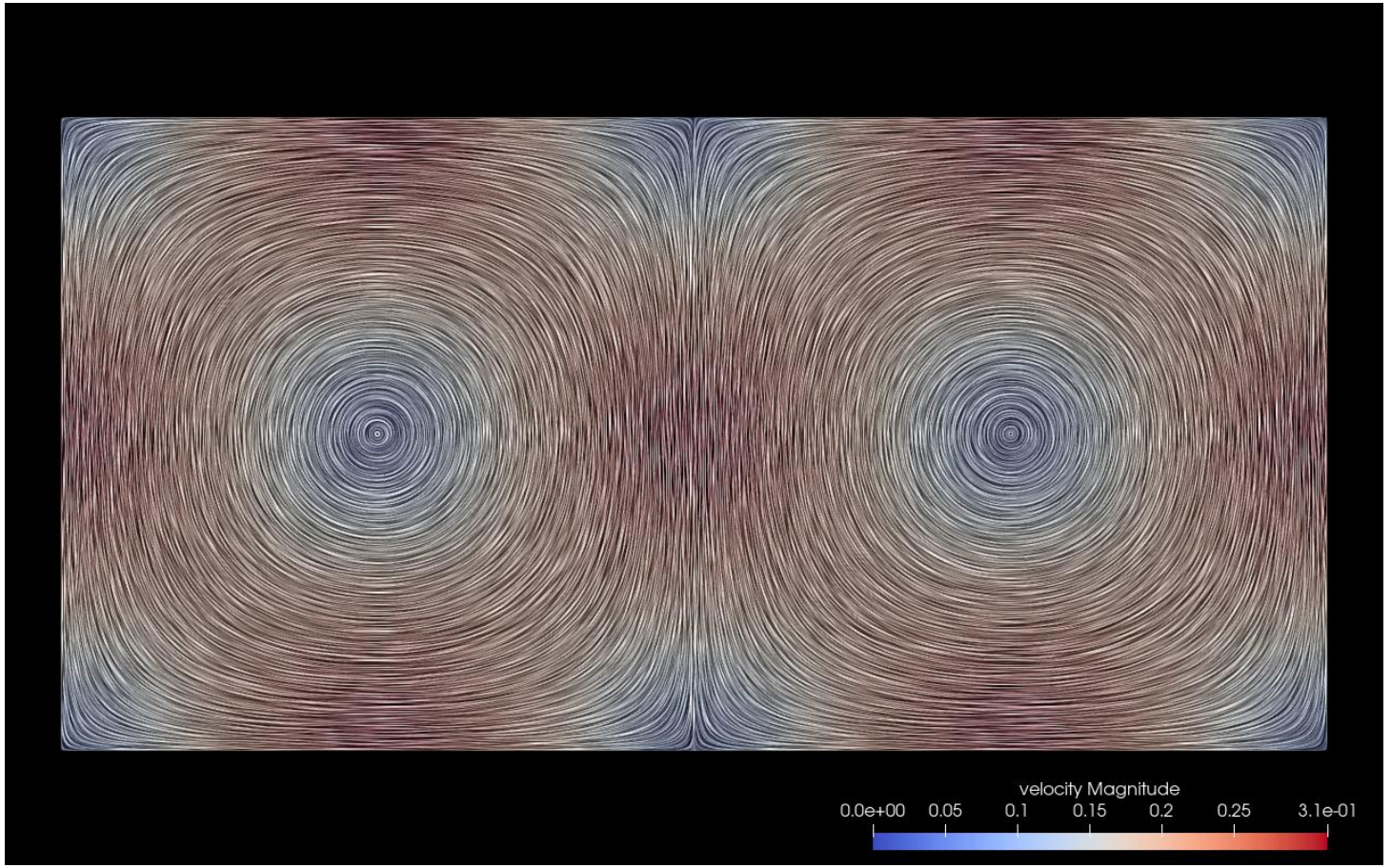
- Difficulties?
  - Missing in 3D



Vector field visualization (2)

- Use a screen-space representation to do a Linear Integral Convolution

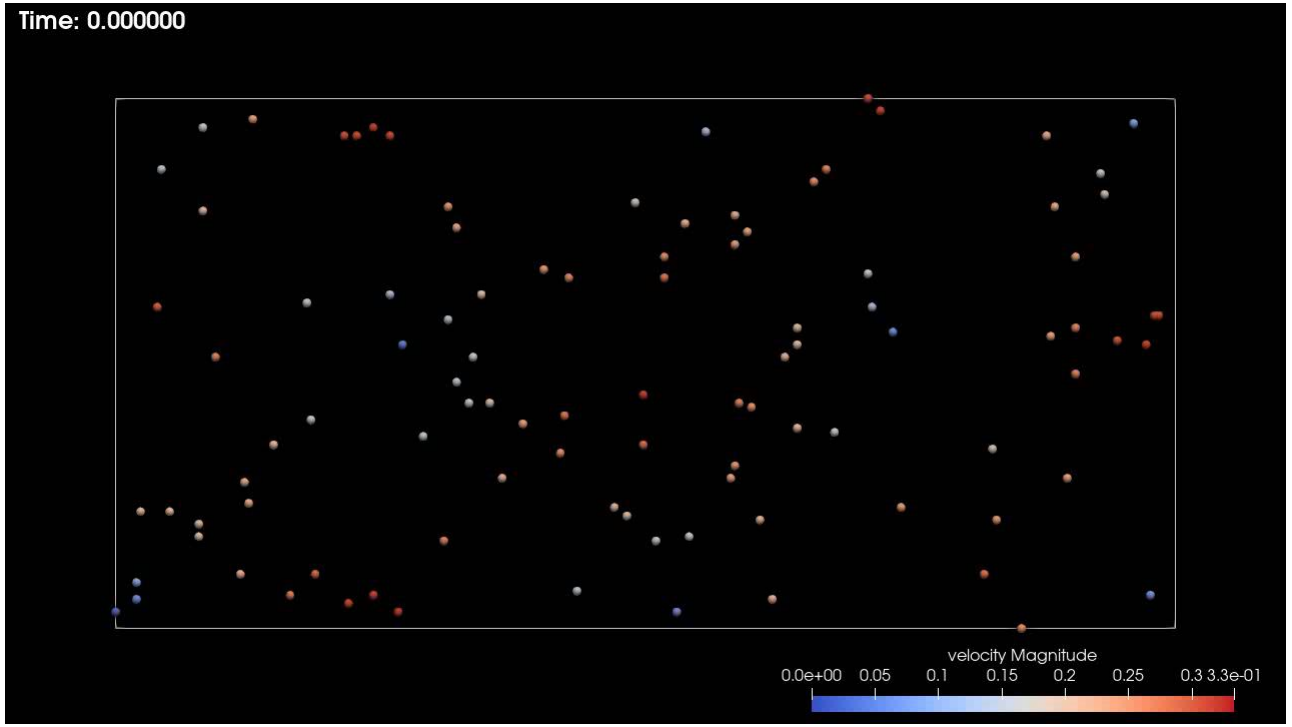
- Difficulties?
  - Missing in 3D
  - Not correct for transient data



Vector field visualization (2)

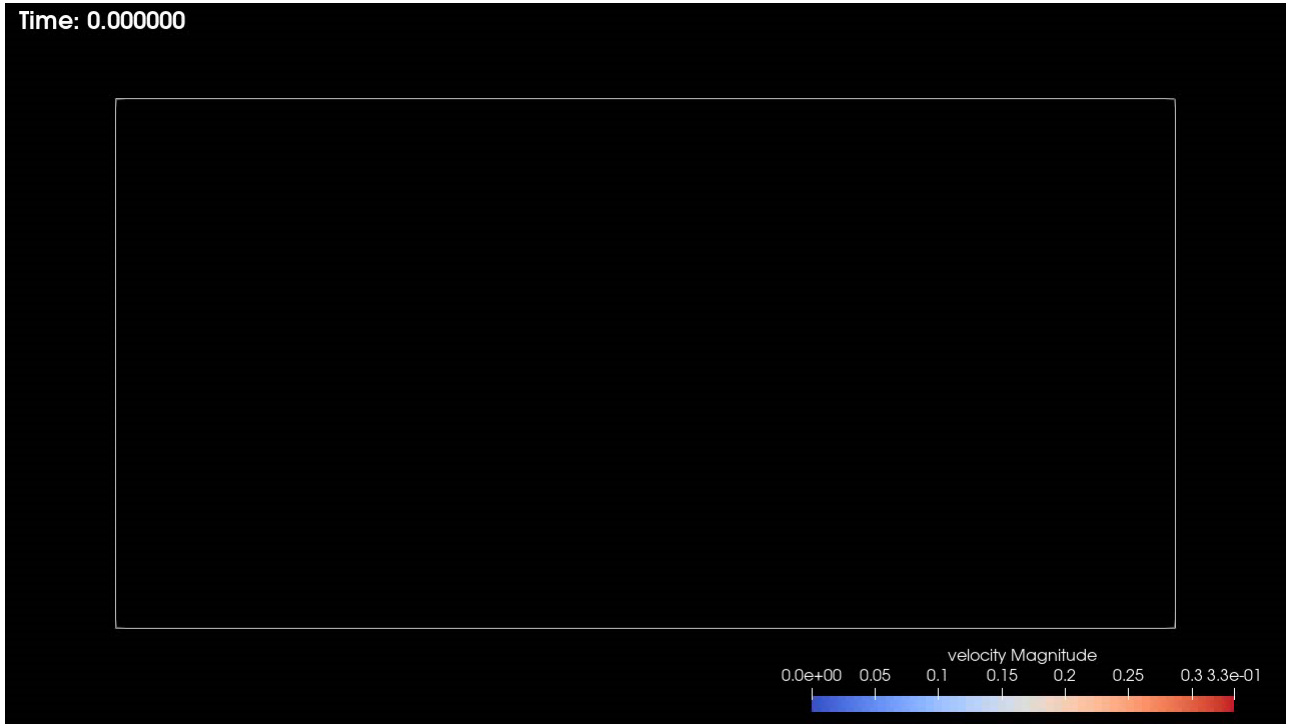
- Transient particles advected by the vector field

- Issues?
  - Particles disappearing
  - Can we re-inject particles at regular intervals?



Vector field visualization (2)

- Transient particles advected by the vector field



Exercise

- Login in to Dardel
- Execute pvTransientDoubleGyre.0*.py
- Make an animation (an mpeg file saved to disk) of temporal pathlines

Time series

- https://docs.paraview.org/en/latest/UsersGuide/dataIngestion.html#handling-temporal-file-series

- https://docs.paraview.org/en/latest/UsersGuide/dataIngestion.html#id1