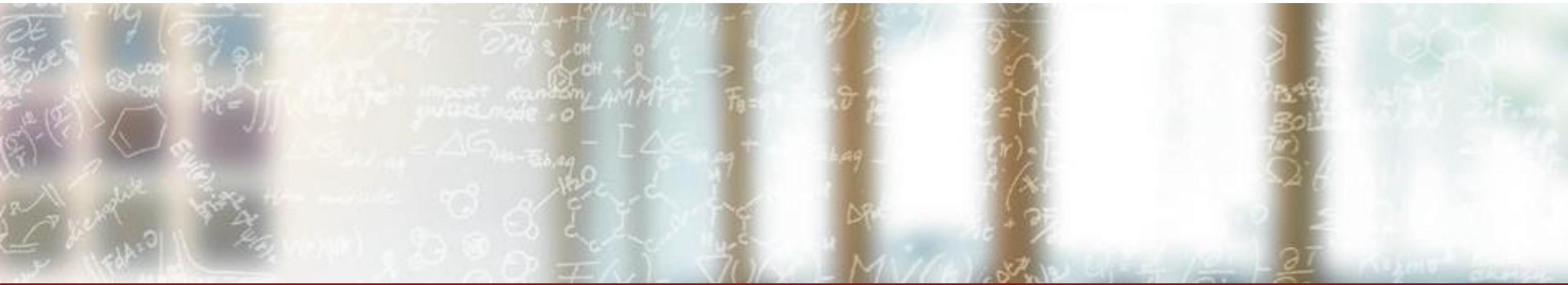




CSCS

Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



Data formats for scientific data output

Dr. Jean M. Favre, CSCS

October 12, 2020

Focus. Driving motivations

Look at data formats from the point of view of using VTK-based applications/libs

- vtkpython
- ParaView
- VisIt
- The itkwidgets lib

- Distinguish between:
 - The VTK file formats
 - The VTK data Model
- Hundreds of file format readers which produce VTK grid objects suitable for processing
- I listed 163 vtk*Reader in our current ParaView installation

Outline

- Facts about data, lessons learned
- Is your data output ready for parallel visualization?
- Time series
- Ghost cells

Facts about data, lessons learned

- The output generated by the SC19, Gordon Bell Prize winner from ETH-Zurich fits in a few KBytes.
- The output of our Scientific Visualization Finalist at SC18 has 1000x10GBytes
- Can I jump directly from timestep N to the timestep where:

$$Pressure == \max(Pressure)$$

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']
```

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

Reading hurricane Isabel data

```
>>> reader = ImageReader(FileNames=['QVAPORf48.bin'])
```

```
>>> reader.DataScalarType = 'float'
```

```
>>> reader.DataExtent = [0, 499, 0, 499, 0, 99]
```

```
>>> reader.UpdatePipeline()
```

```
>>> grid = servermanager.Fetch(reader)
```

```
>>> assert grid.IsA('vtkImageData')
```

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

parallel-aware data readers. Survey Time!

Not all data readers are capable of reading data in parallel, and splitting (load balancing) their data amongst multiple servers

How many of you think that ParaView can read:

- An OpenFoam dataset in parallel? (hint: reader was last updated in May 2020)
- A regular cartesian grid for Vol Rendering? (*.mhd)
- A regular cartesian grid using VTK XML vtkImageData format? (*.vti)
- A grid stored using VTK XML vtkUnstructuredGrid format? (*.vtu)

Details about the MetaIO data reader

<https://www.paraview.org/Wiki/ITK/MetaIO>

Create a 512x512x512 data volume. A vtkImageData!

Exercise:

pvpython

```
from paraview.simple import *  
w = Wavelet()  
w.WholeExtent=[0,511,0,511,0,511]  
w.UpdatePipeline()  
SaveData("volume.mhd")
```

Details about the MetalO data reader

The volume.mhd header file shows:

ObjectType = Image

NDims = 3

BinaryData = True

BinaryDataByteOrderMSB = False

CompressedData = True

CompressedDataSize = 479309980

CenterOfRotation = 0 0 0

ElementSpacing = 1 1 1

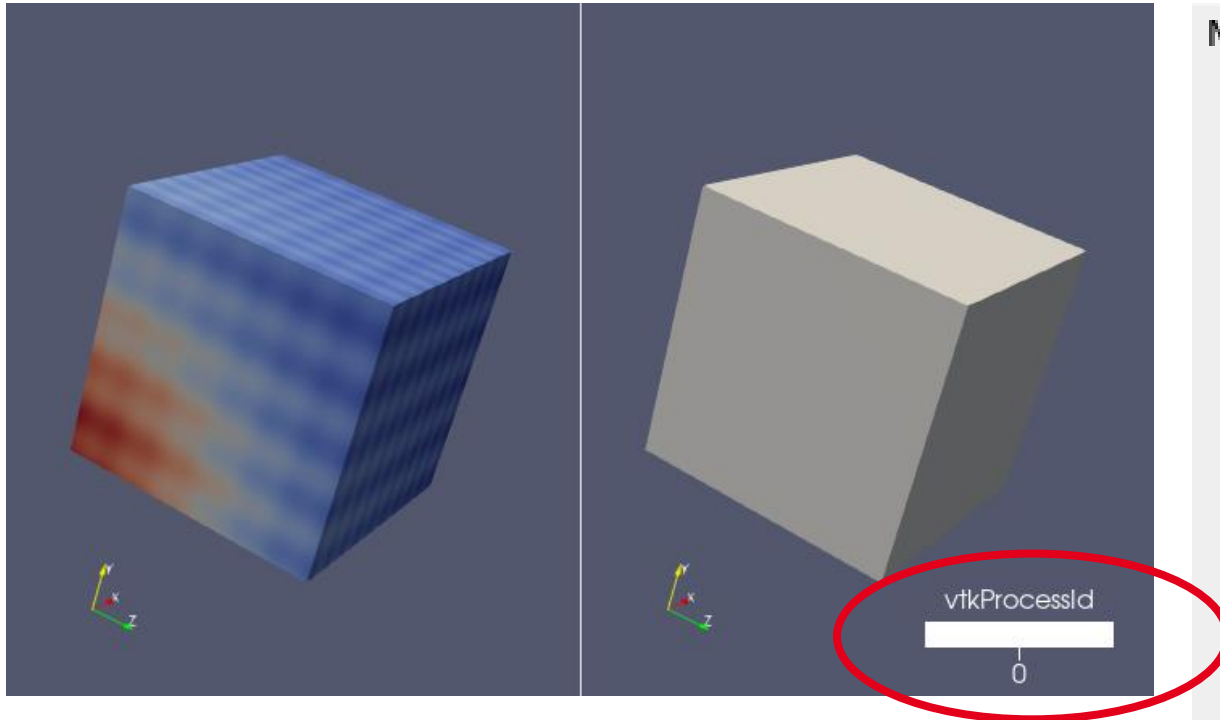
DimSize = 512 512 512

ElementType = MET_FLOAT


ElementDataFile = volume.zraw

Read volume.mhd with N tasks

- `mpiexec -n 4`
`/local/apps/ParaView/5.8/bin/pvserver`
- `paraview --server=localhost`



Memory Inspector

 client


rancate

System Total

9.07 GiB 29.18%

paraview

335.46 MiB 1.05%

 server

rancate

System Total

9.07 GiB 29.18%

pvserver

1.28 GiB 4.12%

0

47964

749.04 MiB 2.35%

1

47965

187.50 MiB 0.59%

2

47966

187.24 MiB 0.59%

3

47967

188.25 MiB 0.59%

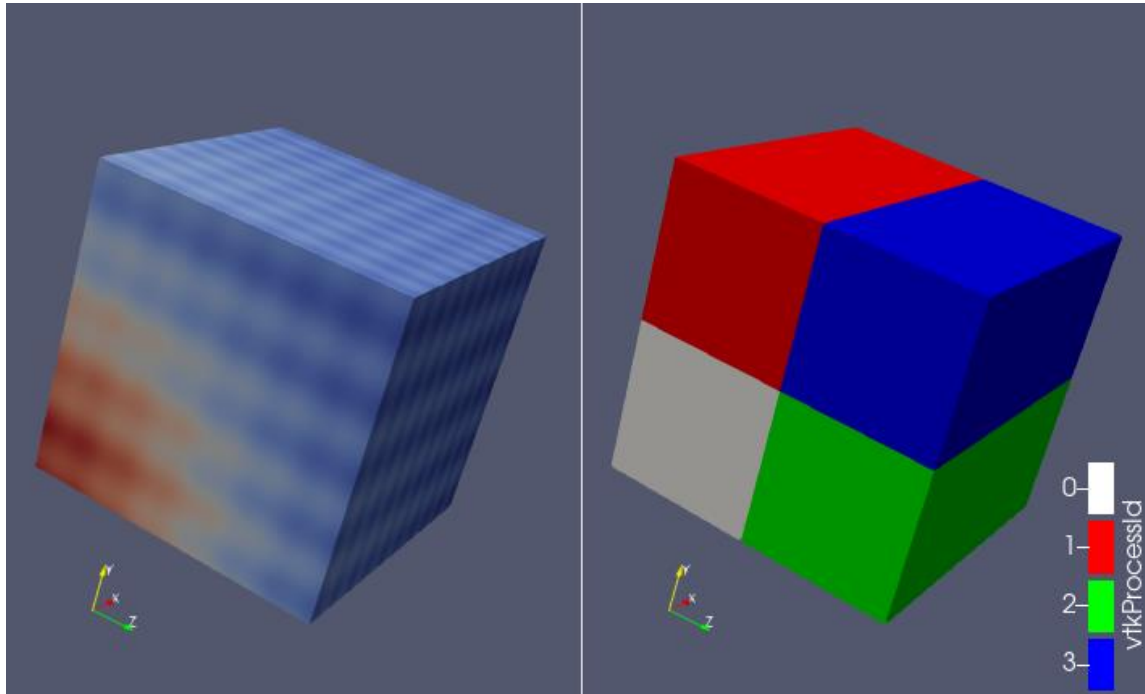
A quick fix

```
from paraview.simple import *  
f = MetaFileSeriesReader(FileNames=["volume.mhd"])  
f.UpdatePipeline()  
SaveData("volume.vti")
```

<https://kitware.github.io/paraview-docs/latest/python/paraview.simple.MetaFileSeriesReader.html>

Live Demonstration

Read volume.vti with N tasks



- `mpiexec -n 4`
`/local/apps/ParaView/5.8/bin/pvserver`
- `paraview --server=localhost`

Memory Inspector

client

runcate	System Total	<div></div>	9.93 GiB 31.95%
	paraview	<div></div>	396.14 MiB 1.24%

server

runcate	System Total	<div></div>	9.93 GiB 31.95%
	pvserver	<div></div>	2.14 GiB 6.88%
0	48205	<div></div>	553.38 MiB 1.74%
1	48206	<div></div>	545.76 MiB 1.71%
2	48207	<div></div>	545.89 MiB 1.72%
3	48208	<div></div>	545.72 MiB 1.71%

parallel-aware data readers

- An OpenFoam dataset in parallel? NO
- A regular cartesian grid for Vol Rendering? (*.mhd) NO
- A regular cartesian grid using VTK XML vtkImageData format? (*.vti) YES
- A grid stored using VTK XML vtkUnstructuredGrid format? (*.vtu) NO

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>

Generate a time series

- Try out `pvGenerateImageData.py`

`SaveData('ts.vti', Writetimestepsasfileseries=1, Filenamesuffix='_%02d')`

`pvbatch pvGenerateImageData.py`

Read a time series

paraview ts_..vti

Python code:

```
>>> reader = GetActiveSource()
```

```
>>> reader.TimestepValues
```

```
[0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8,  
1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1]
```

Ghost-cell support

- When doing visualization (or computation) in a data-parallel fashion, some analysis filters will require different layers of ghost cells, or points.
- [ParallelDataVisualization #understanding-parallel-processing](#)

A survey

- Anyone familiar with the ADIOS file format?
- Anyone being considerably slowed down by very heavy I/O?
 - Use in-situ visualization.
 - We have a nice case currently running at CSCS on 4092 compute nodes. No rendering is done, but the data extraction and filtering is done in-situ, far more reducing the size of the data that needs to be saved to disk.

New ParaView data I/O plugins developed at CSCS

- A native, parallel, Nek5000 reader
- A native, parallel, GADGET2 HDF5 reader

Summary

- There are *hundreds* of file formats available with standard visualization applications
- Test and verify that you can:
 - Group multiple timesteps in one file, or
 - Read collections of files
 - Read data in parallel
- Test and verify that you can read subsets, sub-sampled data

Questions?

- Use the chat for Q/A

**CSCS**Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre**ETH** zürich