Data Analysis & Visualisation

J.P. Morrissey, *University of Edinburgh*

CCC-ParaSolS Network Event 4

Queen's University Belfast, Northern Ireland

24-26th March 2026

UKRI Science and Technology Facilities Council
CoSeC Computational Science Centre for Research Communities
IMPERIAL
THE UNIVERSITY of EDINBURGH School of Engineering
Gran ED

Pre-Warning: Tutorial Datasets

- Three different datasets to work with:
  - Polydisperse spheres
  - Polydisperse multi-spheres with multiple shapes
  - Polydisperse polyhedrals with multiple shapes

- Available in both **XML** and **VTKHDF** formats

- Download from here:
  - Data_Anlaysis_Vis_VTK - pCloud

What is Data Analysis?

- A good definition:
  - *"…the systematic process of cleaning, transforming, modelling, and inspecting raw data to identify patterns, relationships, and trends."*

https://link.springer.com/rwe/10.1007/978-0-387-39940-9_324#:~:text=Definition,include%20visualization%20of%20their%20uncertainty.

What is Data Analysis?

- A good definition:
  - *"…the systematic process of cleaning, transforming, modelling, and inspecting raw data to identify patterns, relationships, and trends."*

- **Quantitative Analysis**: Uses statistical techniques (e.g., ANOVA, regression, descriptive stats) to analyse numerical data
- **Qualitative Analysis:** Analyses non-numerical data (e.g., text, interviews, audio) to identify themes and patterns

What is Data Analysis?

- A good definition:
  - *"…the systematic process of cleaning, transforming, modelling, and inspecting raw data to identify patterns, relationships, and trends."*

- **Quantitative Analysis**: Uses statistical techniques (e.g., ANOVA, regression, descriptive stats) to analyse numerical data
- **Qualitative Analysis:** Analyses non-numerical data (e.g., text, interviews, audio) to identify themes and patterns

- **Scientific Visualisation** falls into both categories. Visualisation deals with measurable data (e.g. pressure, density) but is used to identify spatial and temporal trends in the data that are difficult to see in the raw data alone

https://link.springer.com/rwe/10.1007/978-0-387-39940-9_324#:~:text=Definition,include%20visualization%20of%20their%20uncertainty.

Why Is There A Need For Visualisation?

- Statistics can tell you a lot about your data and can be essential in understanding it
  - Mean, median, std. deviation, variance, etc
- However, stats can *lie* and can only tell you so much
- You should **always** visualise your data!

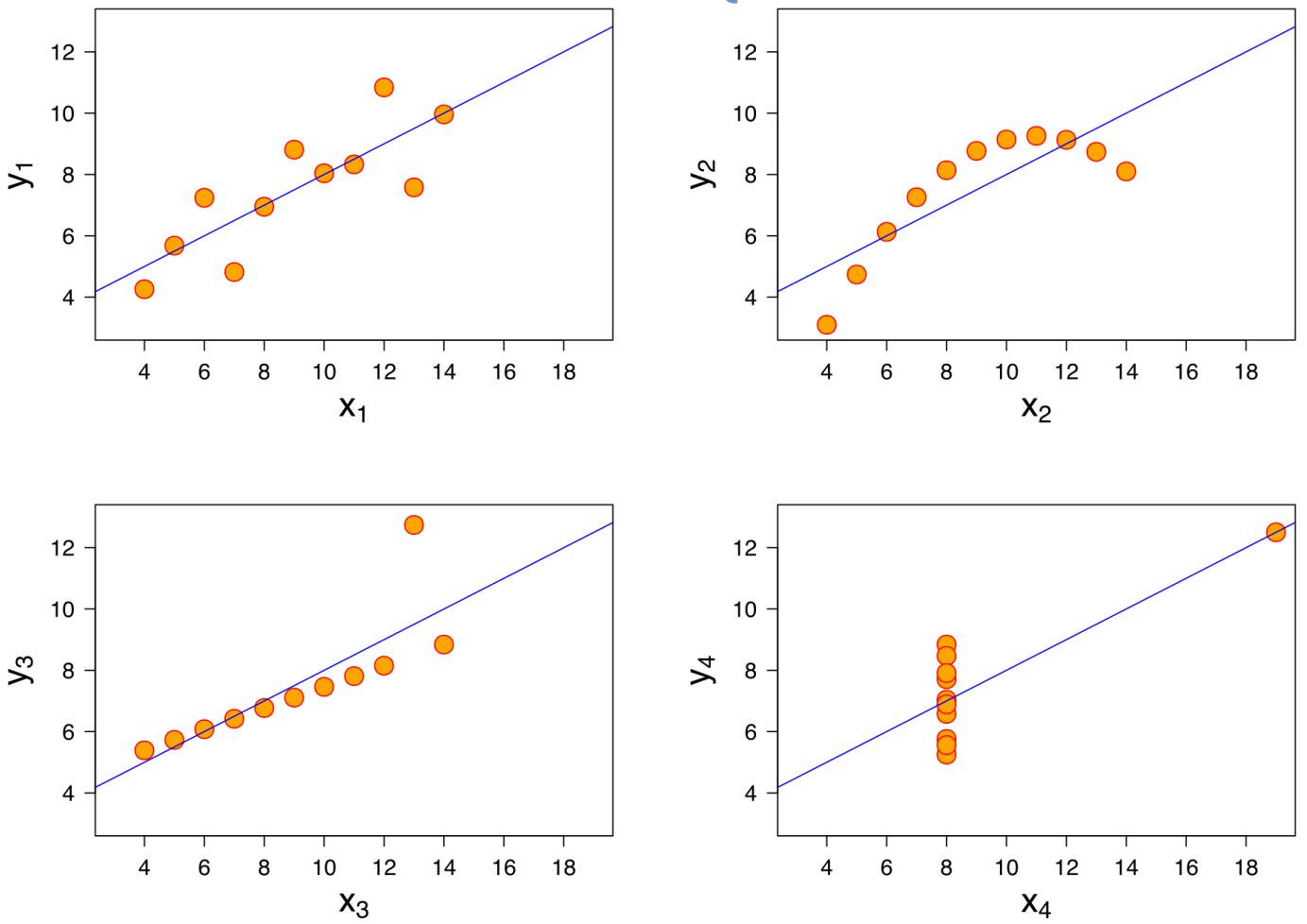
Anscombe's Quartet

Index	x1	y1	x2	y2	x3	y3	x4	y4
1	10	8.04	10	9.14	10	7.46	8	6.58
2	8	6.95	8	8.14	8	6.77	8	5.76
3	13	7.58	13	8.74	13	12.74	8	7.71
4	9	8.81	9	8.77	9	7.11	8	8.84
5	11	8.33	11	9.26	11	7.81	8	8.47
6	14	9.96	14	8.1	14	8.84	8	7.04
7	6	7.24	6	6.13	6	6.08	8	5.25
8	4	4.26	4	3.1	4	5.39	19	12.5
9	12	10.84	12	9.13	12	8.15	8	5.56
10	7	4.82	7	7.26	7	6.42	8	7.91
11	5	5.68	5	4.74	5	5.73	8	6.89
Mean	9	7.500	9	7.500	9	7.500	9	7.500
Std. Deviation	3.32	2.03	3.32	2.03	3.32	2.03	3.32	2.03
Variance	11	4.127	11	4.127	11	4.127	11	4.127
Correlation	0.816							
Linear regression Line	y = 3.00 + 0.500x							

Why Is There A Need For Visualisation?

- Statistics can tell you a lot about your data and can be essential in understanding it
  - Mean, median, std. deviation, variance, etc
- However, stats can *lie* and can only tell you so much
- You should **always** visualise your data!

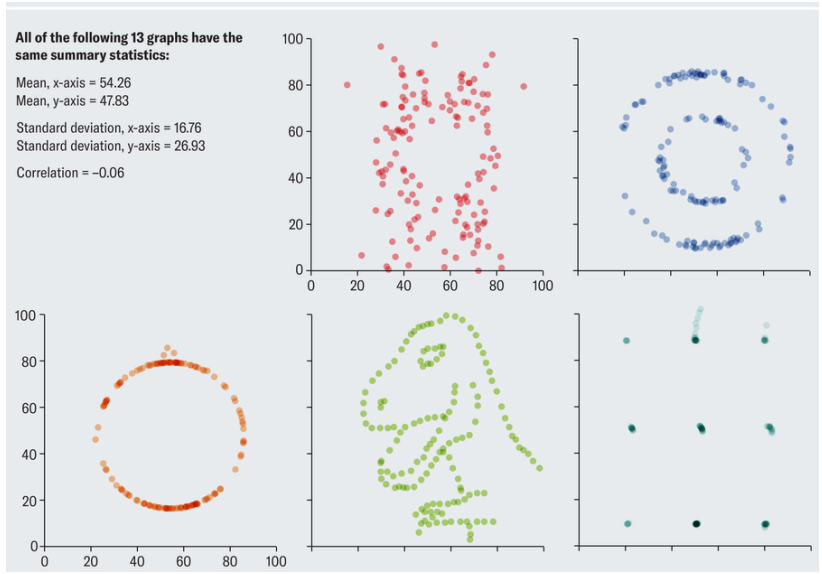
Anscombe's Quartet

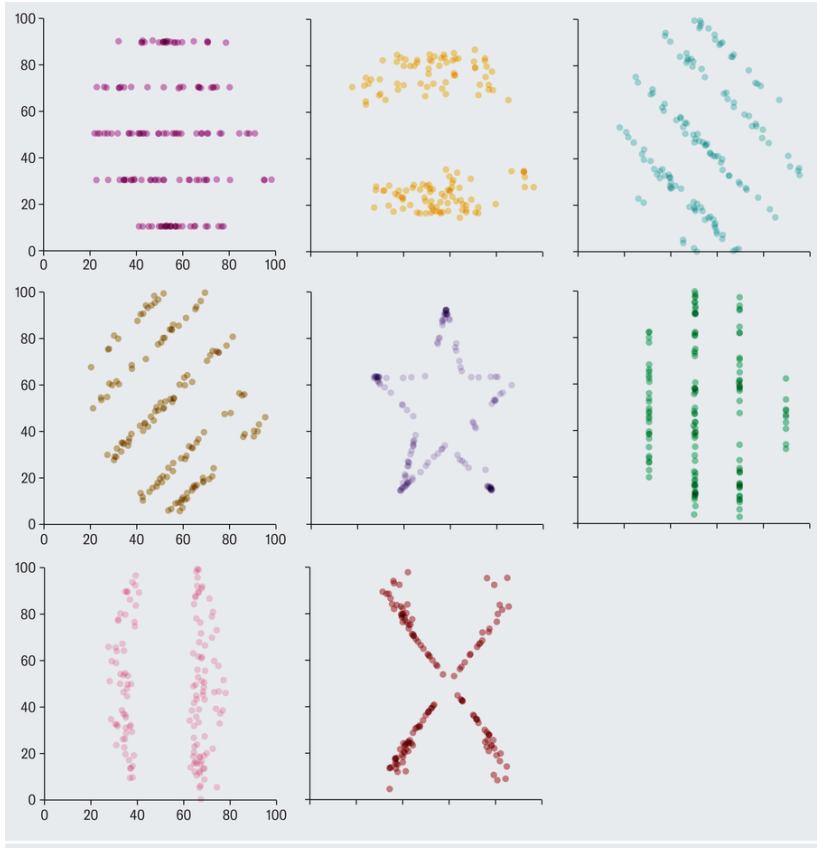


"Graphs in Statistical Analysis," by F. J. Anscombe, in *American Statistician*, Vol. 27, No. 1; February 1973

The “Datasaurus Dozen”

- A similar concept to **Anscombe’s Quartet** but without the linear regression correlation
- A single dataset can be distorted into many very different forms and still maintain all the same key summary statistics

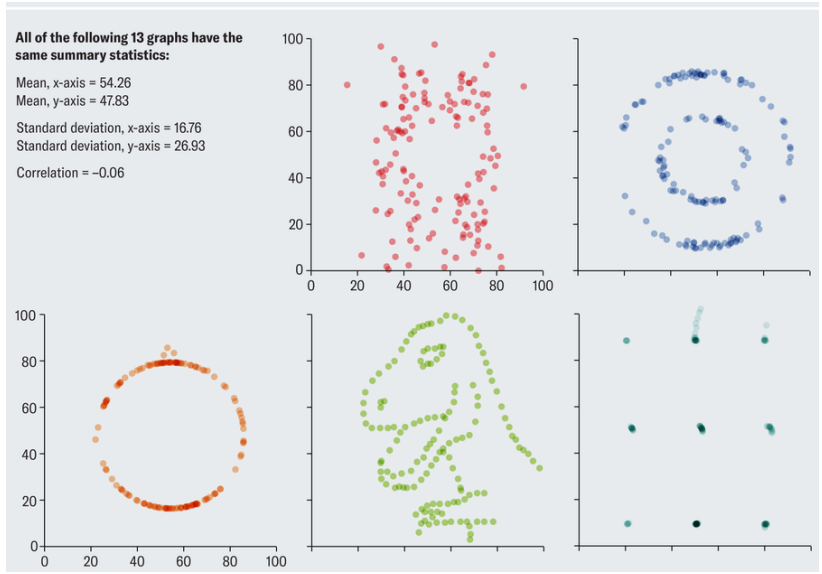


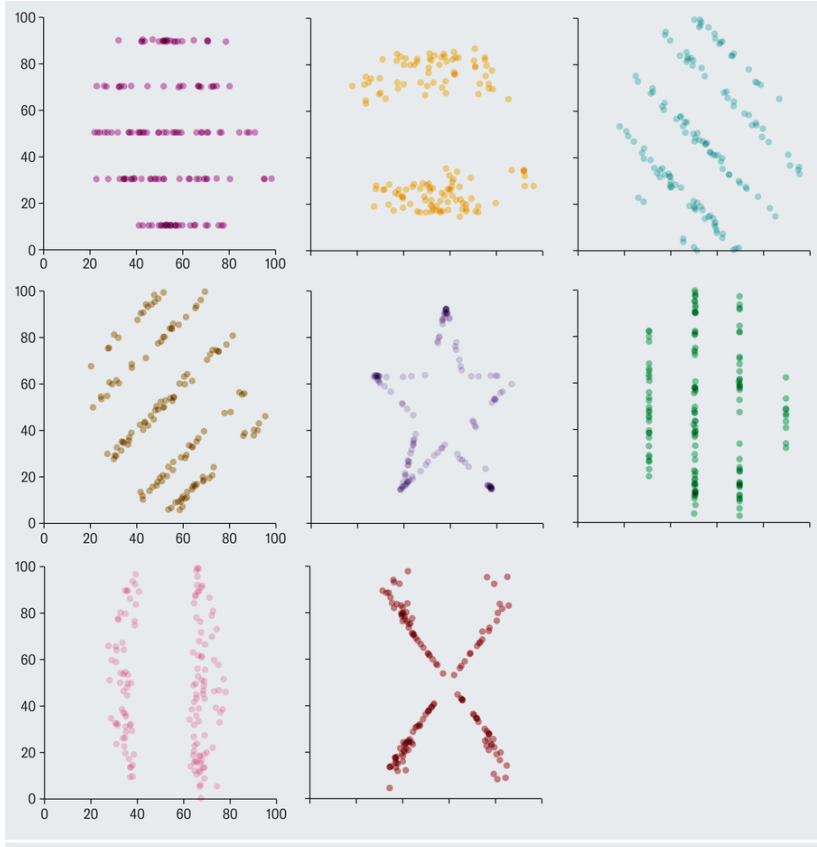


“Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing”, Justin Matejka and George Fitzmaurice, in CHI ’17: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems; May 2017. https://dl.acm.org/doi/10.1145/3025453.3025912

The "Datasaurus Dozen"

- A similar concept to **Anscombe's Quartet** but without the linear regression correlation
- A single dataset can be distorted into many very different forms and still maintain all the same key summary statistics





"Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing", Justin Matejka and George Fitzmaurice, in CHI '17: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems; May 2017. https://dl.acm.org/doi/10.1145/3025453.3025912

Moral of the story: Don't trust stats alone!

Basics of a Good Plot

- All figures/plots should have a singular clear message
  - It should be able to deliver its message even without the surrounding text
  - Clear legend
  - Descriptive caption

- All axes should be clearly labelled including units
- Use sans-serif fonts which are easier to read when small
- Use different symbols and line types to differentiate data
- Avoid colour choices that affect people with colour blindness
- Always view plot in grayscale to assess readability – don't rely on colour to deliver message

- Avoid complex 3D visualisations when possible
- Avoid pie charts

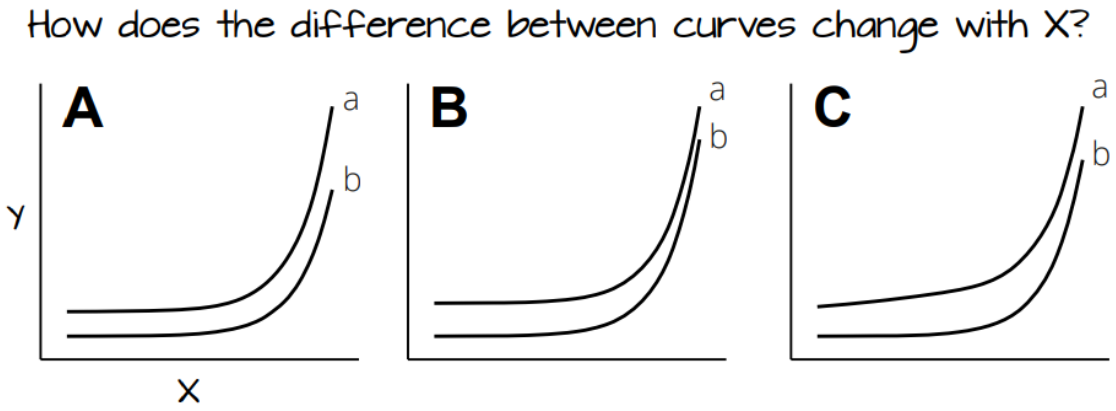
Beware of Poor Plot Selection

- Watch out for graphs that are difficult to interpret easily

Allen, E. A., & Erhardt, E. B. (2017). Visualizing scientific data. In J. T. Cacioppo, L. G. Tassinary, & G. G. Berntson (Eds.), *Handbook of psychophysiology* (4th ed., pp. 679–697). Cambridge University Press.

Beware of Poor Plot Selection

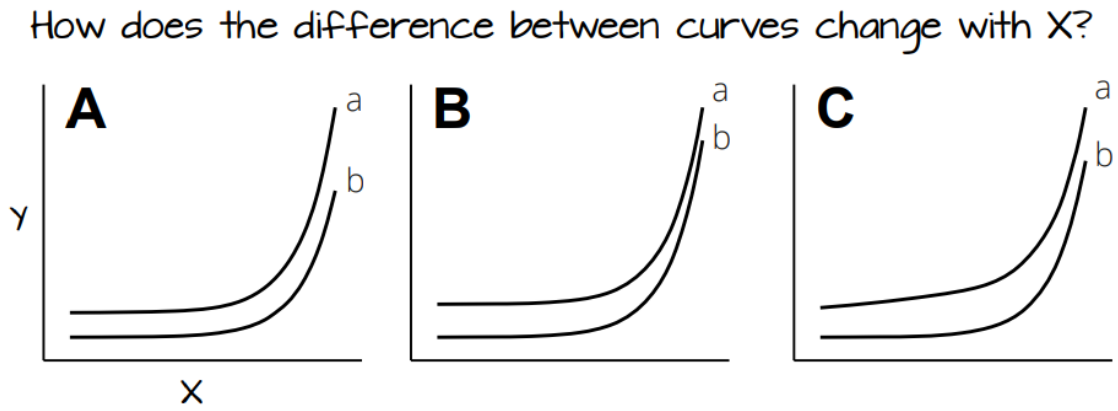
- Watch out for graphs that are difficult to interpret easily
  - "Converging Parallel Lines"

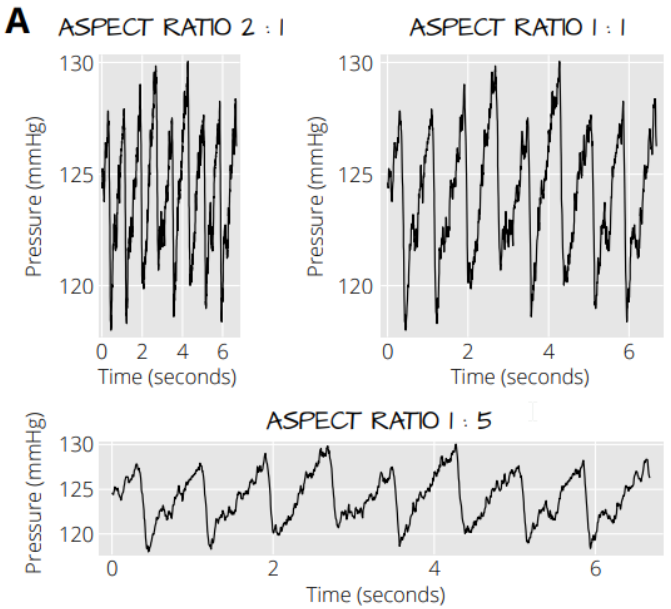


Allen, E. A., & Erhardt, E. B. (2017). Visualizing scientific data. In J. T. Cacioppo, L. G. Tassinary, & G. G. Berntson (Eds.), *Handbook of psychophysiology* (4th ed., pp. 679–697). Cambridge University Press.

Beware of Poor Plot Selection

- Watch out for graphs that are difficult to interpret easily
  - "Converging Parallel Lines"
  - Data aspect ratio

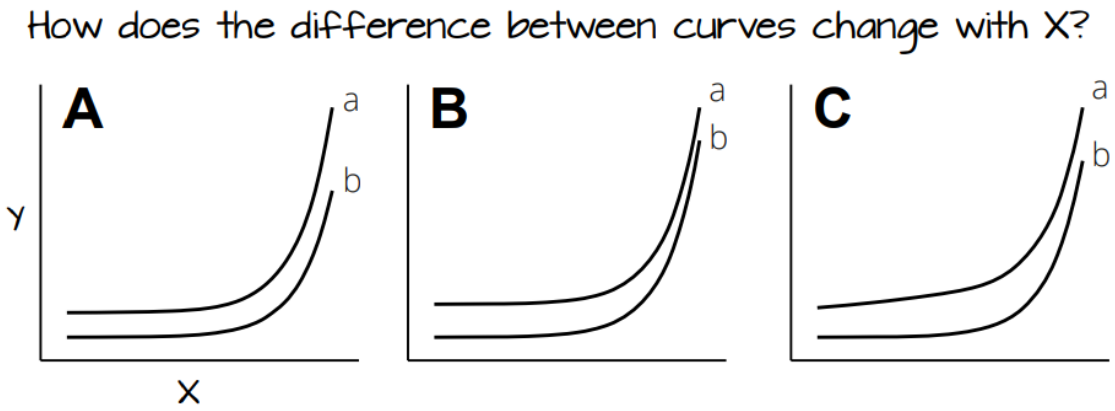


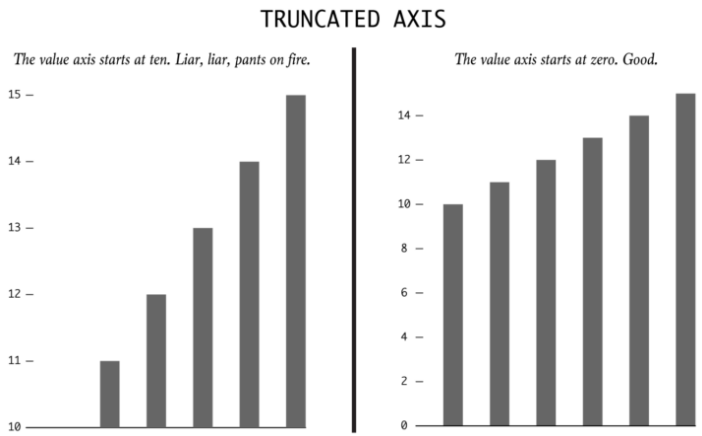


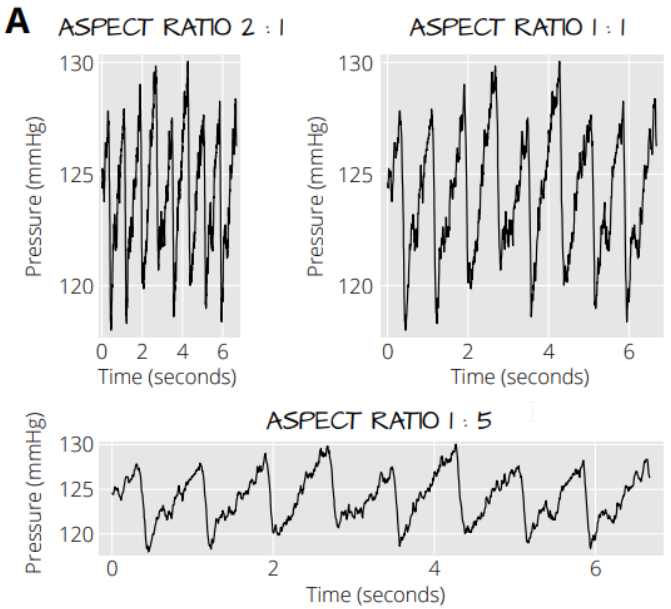
Allen, E. A., & Erhardt, E. B. (2017). Visualizing scientific data. In J. T. Cacioppo, L. G. Tassinary, & G. G. Berntson (Eds.), *Handbook of psychophysiology* (4th ed., pp. 679–697). Cambridge University Press.

Beware of Poor Plot Selection

- Watch out for graphs that are difficult to interpret easily
  - "Converging Parallel Lines"
  - Data aspect ratio
  - Truncated axes



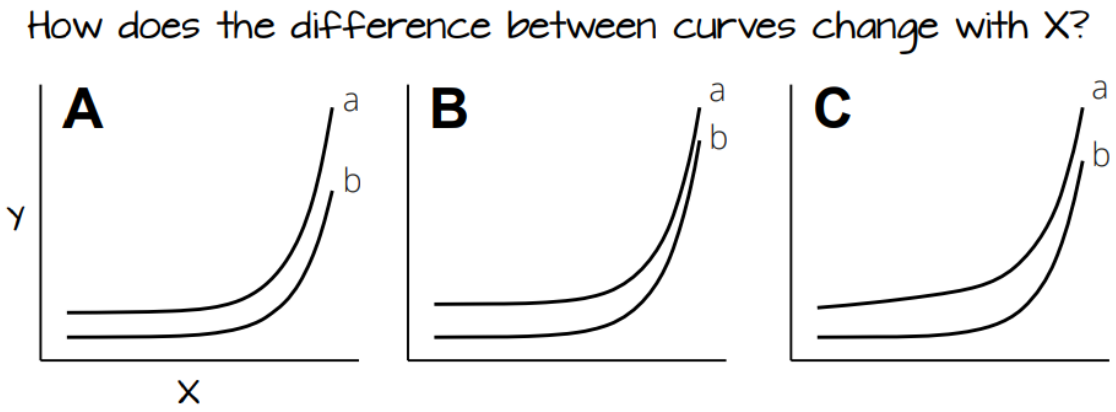


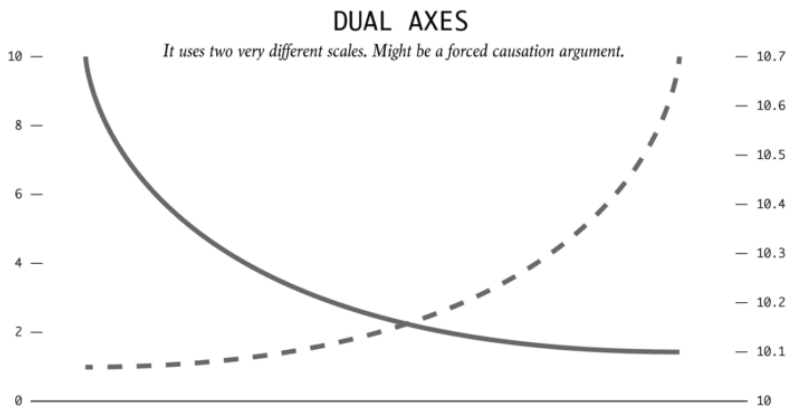


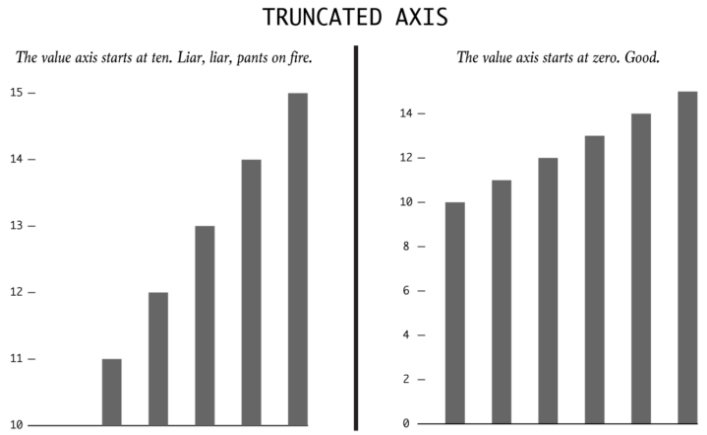
Allen, E. A., & Erhardt, E. B. (2017). Visualizing scientific data. In J. T. Cacioppo, L. G. Tassinary, & G. G. Berntson (Eds.), *Handbook of psychophysiology* (4th ed., pp. 679–697). Cambridge University Press.

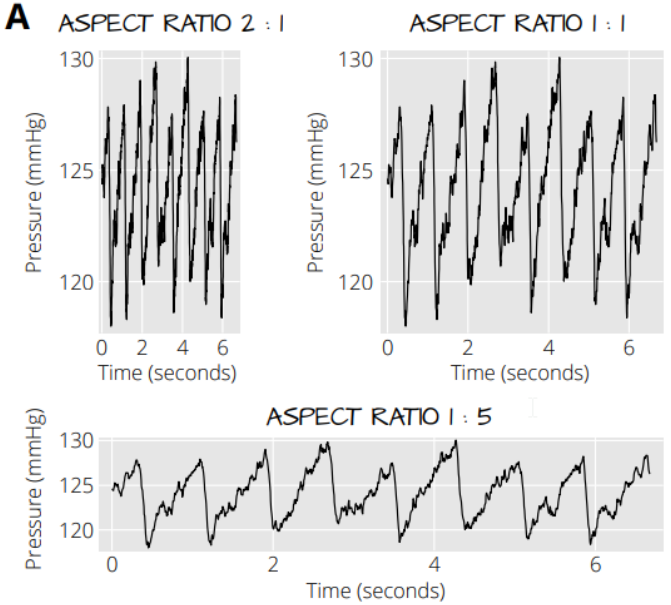
Beware of Poor Plot Selection

- Watch out for graphs that are difficult to interpret easily
  - "Converging Parallel Lines"
  - Data aspect ratio
  - Truncated axes
  - Dual axis are great ways to force a correlation





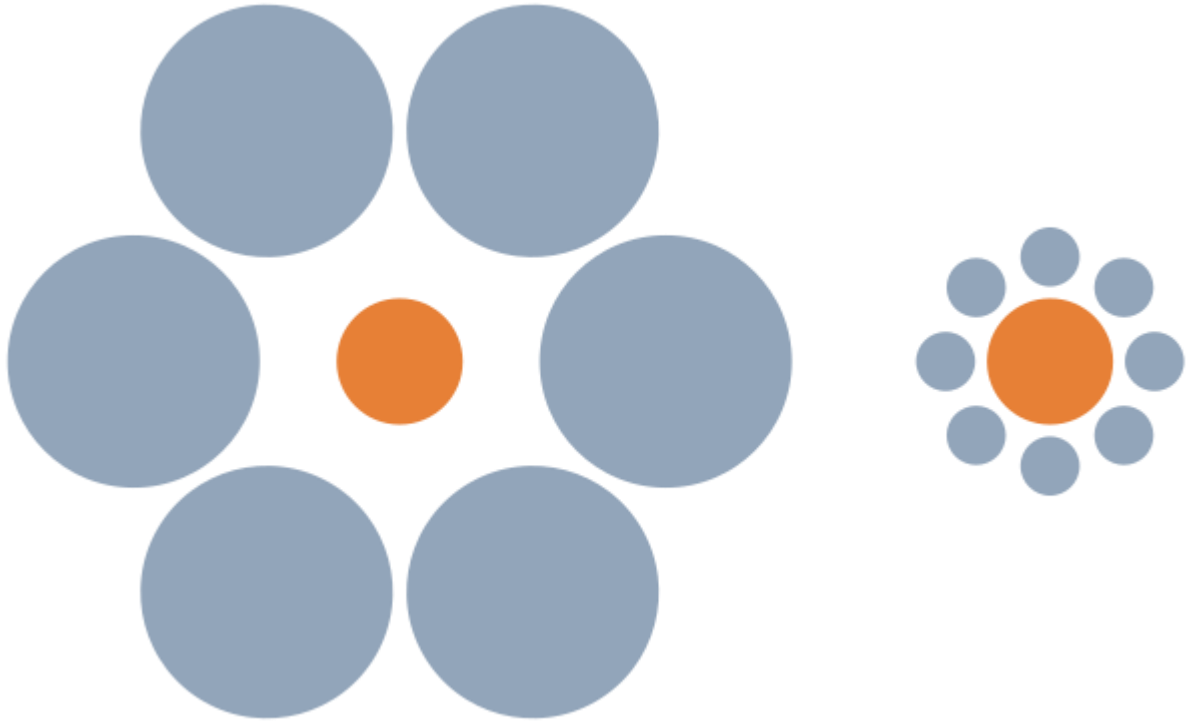




Allen, E. A., & Erhardt, E. B. (2017). Visualizing scientific data. In J. T. Cacioppo, L. G. Tassinary, & G. G. Berntson (Eds.), *Handbook of psychophysiology* (4th ed., pp. 679–697). Cambridge University Press.

Beware of Context

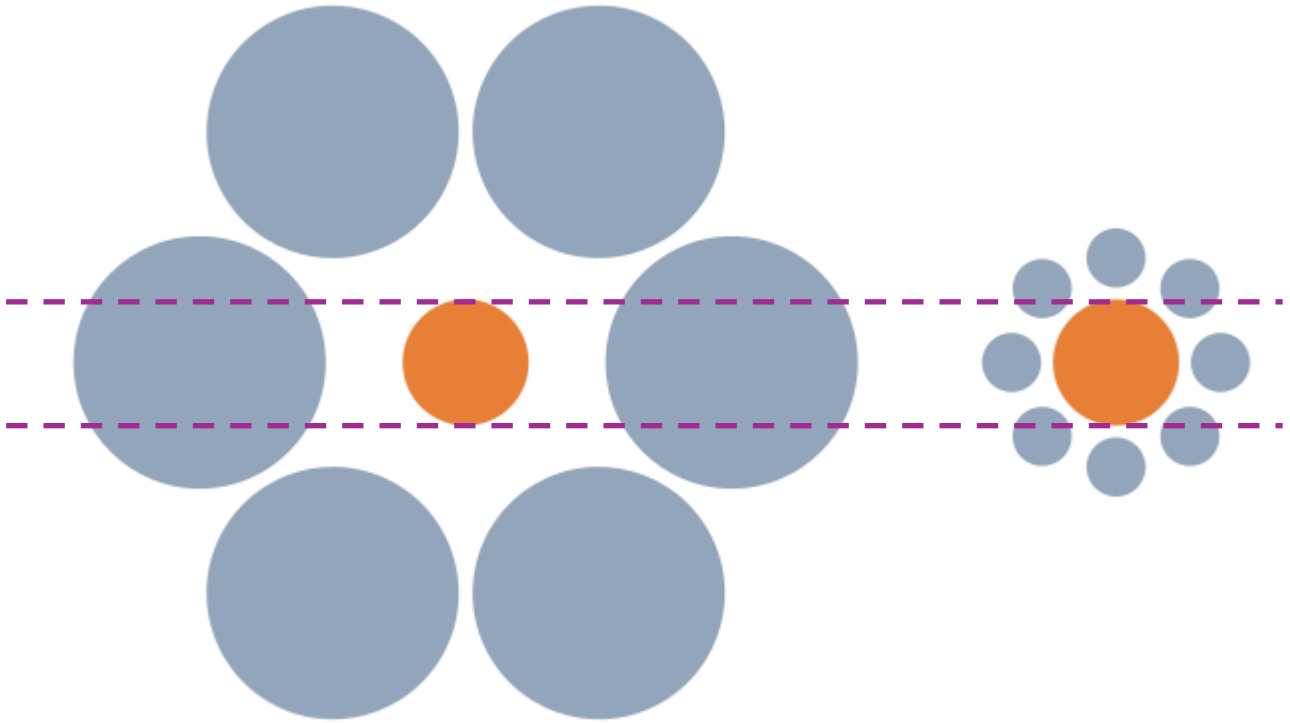
- Which orange circle is larger?



https://www.illusionsindex.org/i/ebbinghaus-illusion

Beware of Context

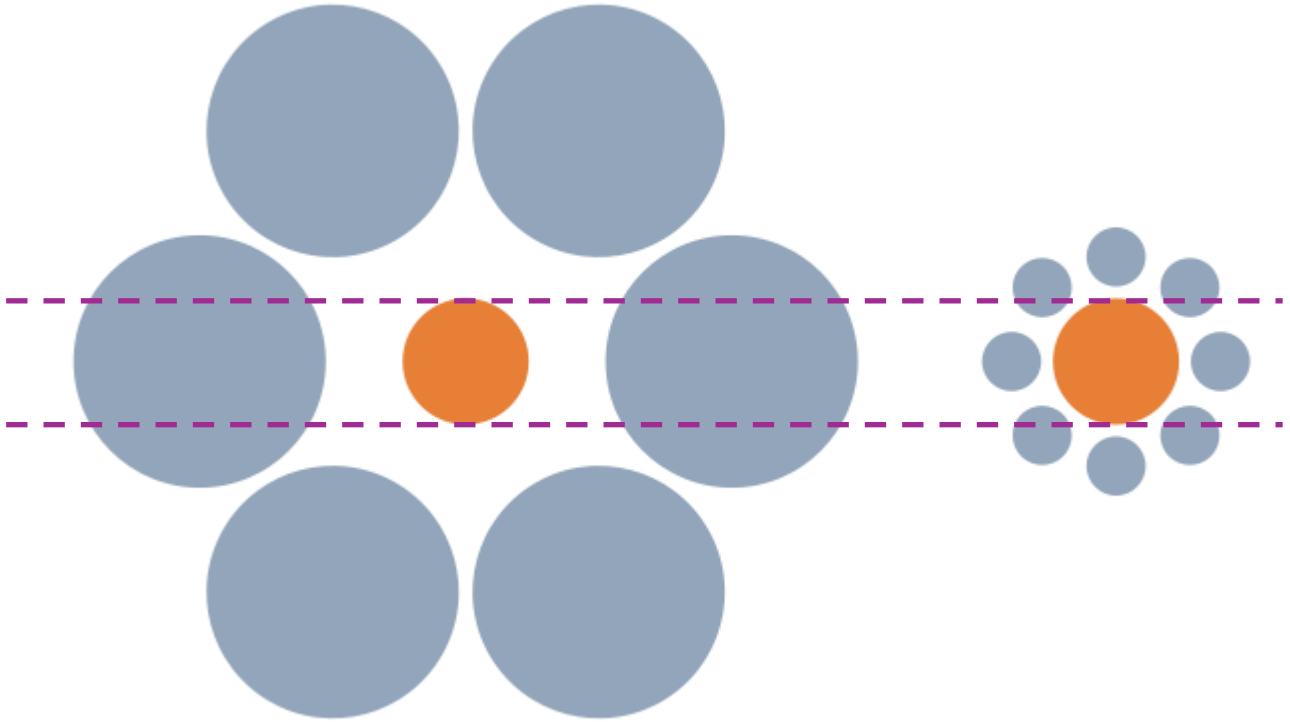
- Which orange circle is larger?



https://www.illusionsindex.org/i/ebbinghaus-illusion

Beware of Context

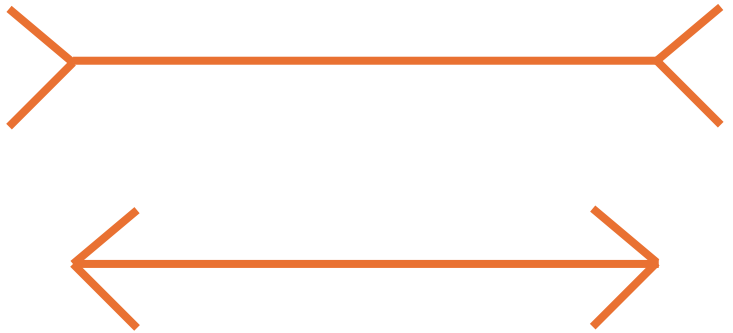
- Which orange circle is larger?
- Because of relative positioning of the larger and smaller grey circles we perceive the right one to be larger
- Can be an issue with cluster-plots



https://www.illusionsindex.org/i/ebbinghaus-illusion

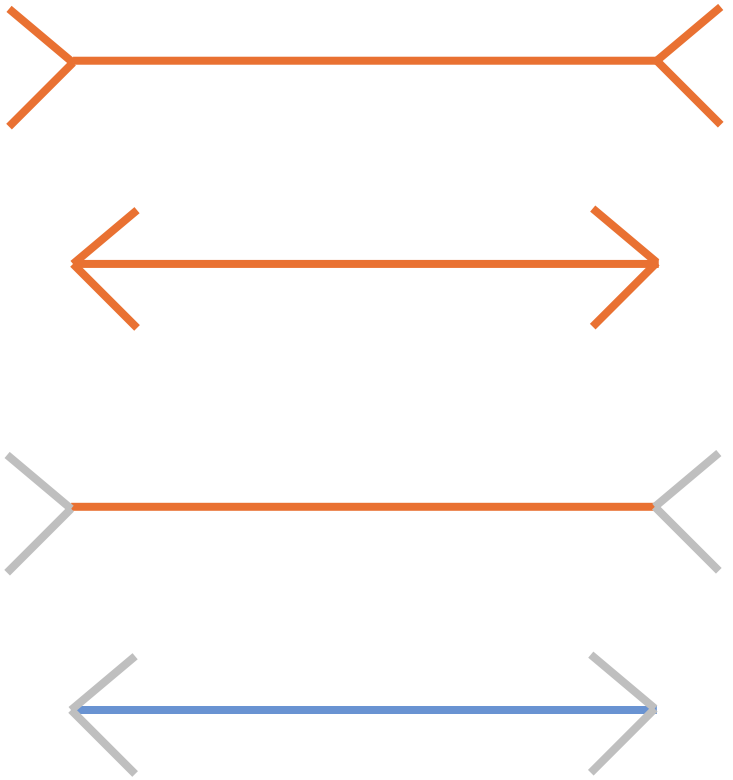
Beware of Additional Annotations

- Which line is longer?



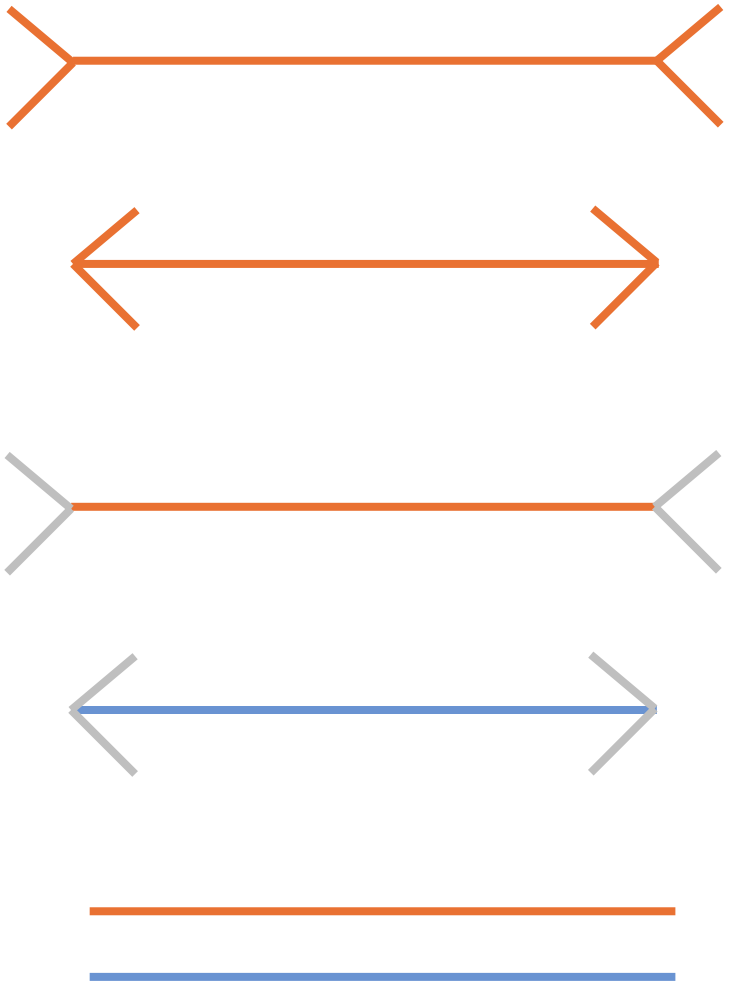
Beware of Additional Annotations

- Which line is longer?



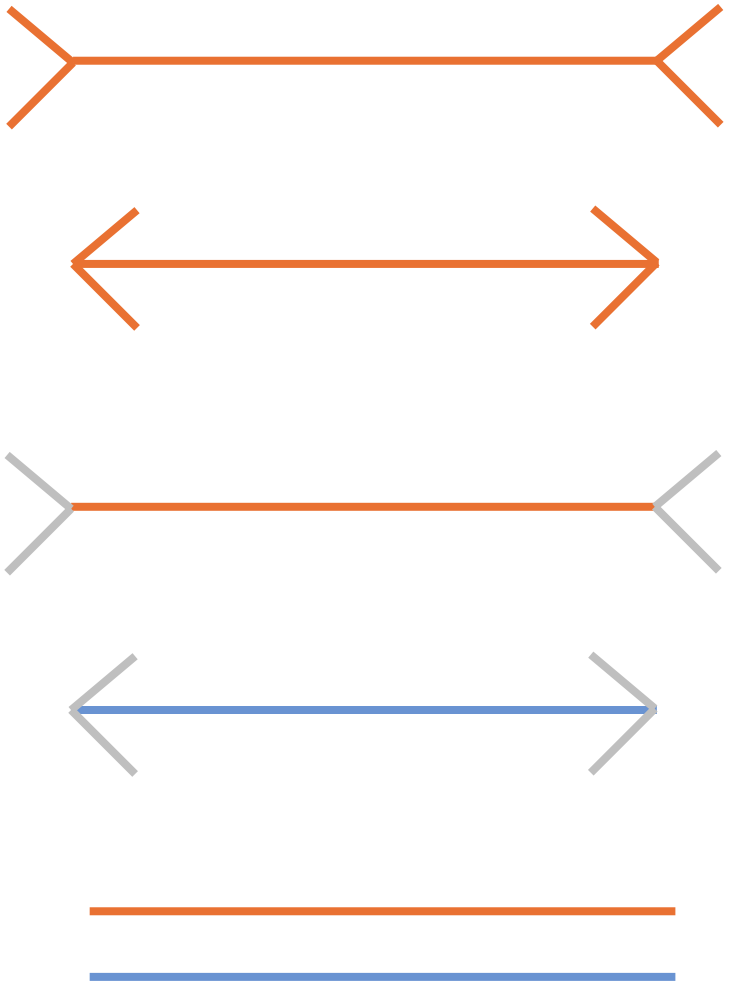
Beware of Additional Annotations

- Which line is longer?



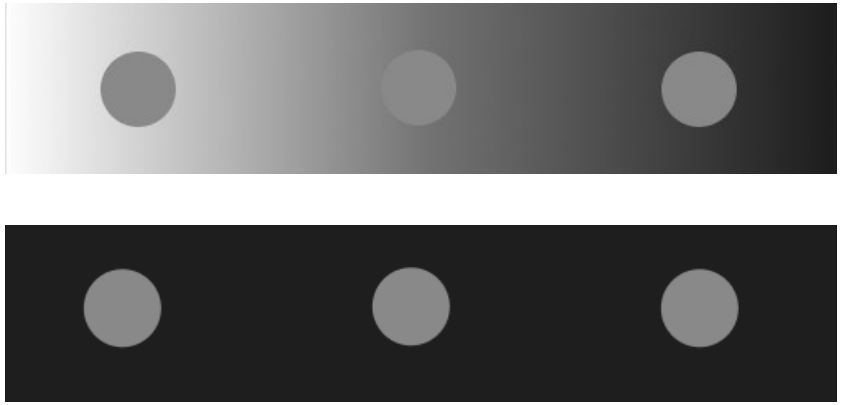
Beware of Additional Annotations

- Which line is longer?
- Additional shapes can affect our perception of length
- Consider different mark types which will not affect the interpretation or the use of a different colour



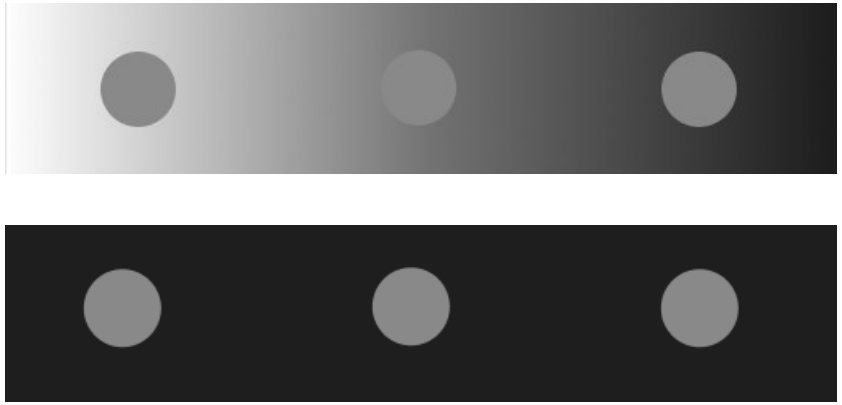
Colour & Shade

- Colour and shade can cause issues with interpretations
- For example, the following dots are the same colour but appear different because of the background gradient



Colour & Shade

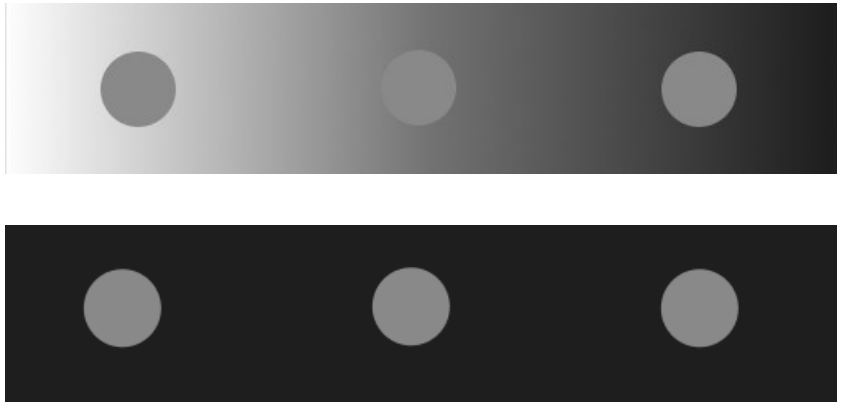
- Colour and shade can cause issues with interpretations
- For example, the following dots are the same colour but appear different because of the background gradient



- The "*Dress*" (2015) *#dressgate*
  - Blue & Black or White & Gold?

Colour & Shade

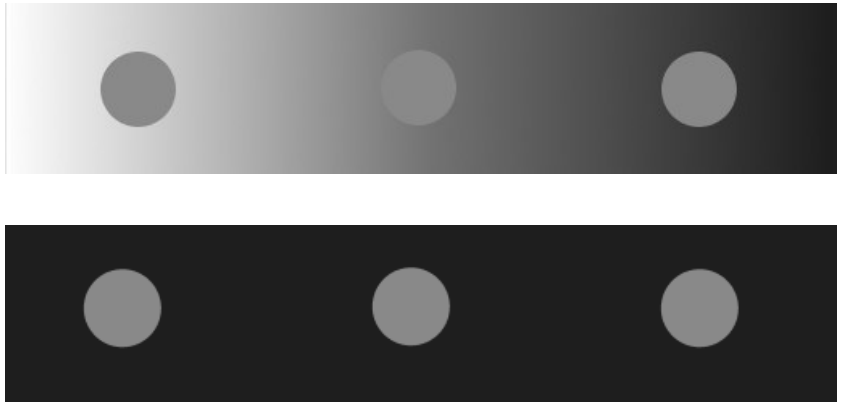
- Colour and shade can cause issues with interpretations
- For example, the following dots are the same colour but appear different because of the background gradient



- The "*Dress*" (2015) *#dressgate*
  - Blue & Black or White & Gold?

Colour & Shade

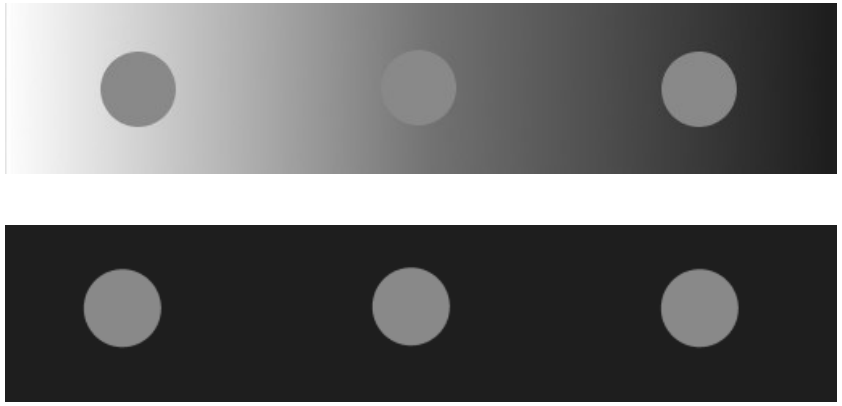
- Colour and shade can cause issues with interpretations
- For example, the following dots are the same colour but appear different because of the background gradient



- The "*Dress*" (2015) *#dressgate*
  - Blue & Black or White & Gold?

Colour & Shade

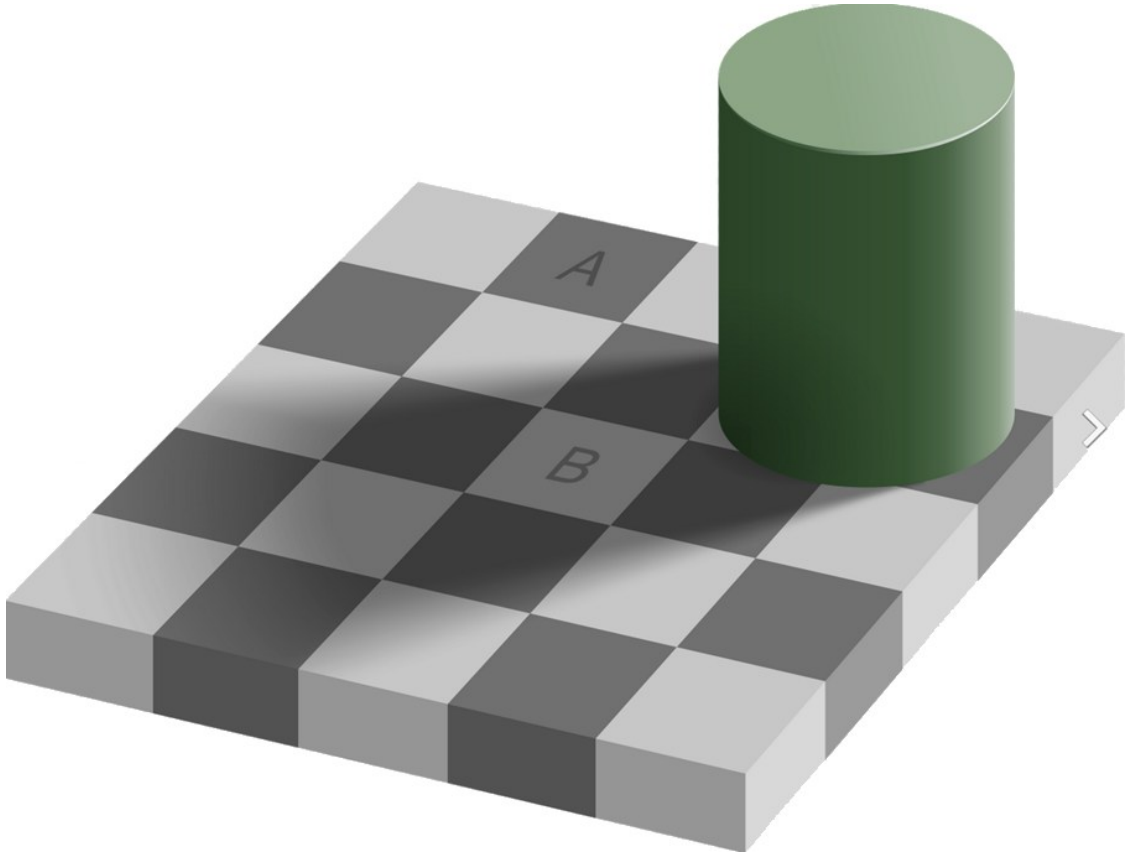
- Colour and shade can cause issues with interpretations
- For example, the following dots are the same colour but appear different because of the background gradient



- The "*Dress*" (2015) *#dressgate*
  - Blue & Black or White & Gold?

Choose colours wisely!

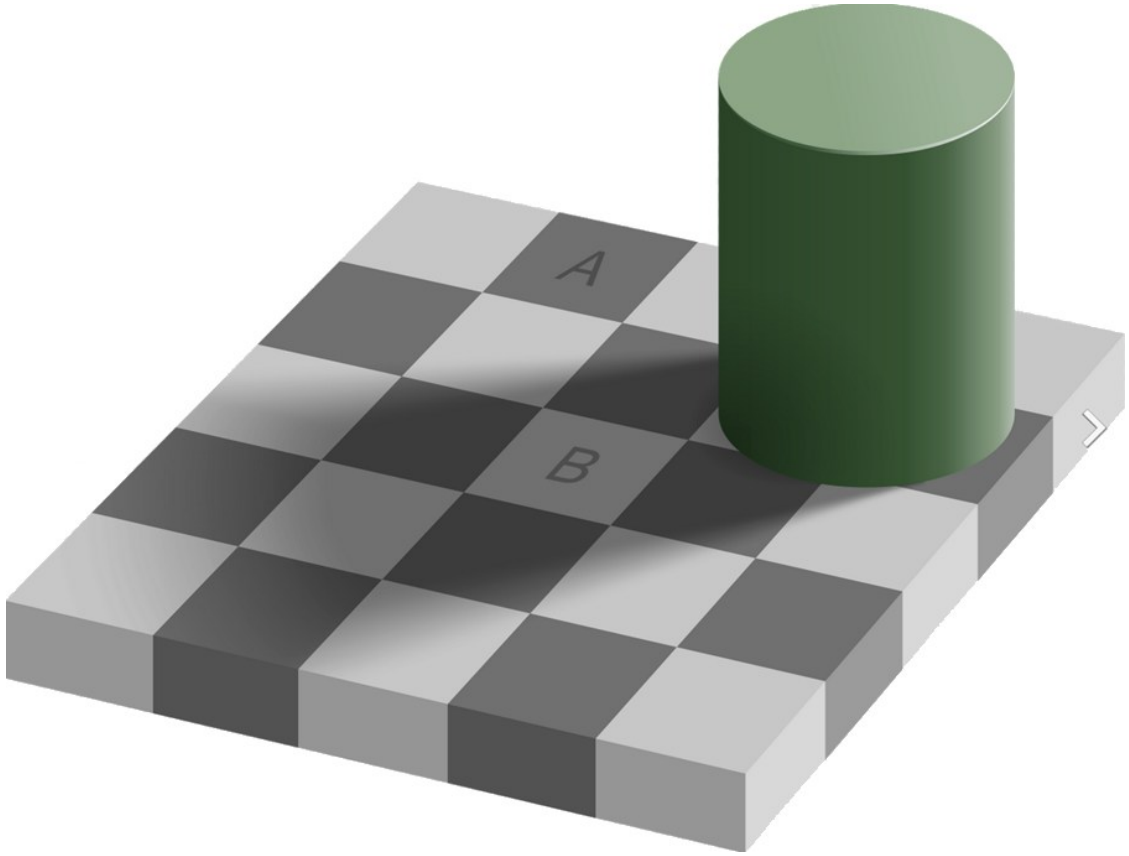
Checker-board Illusion



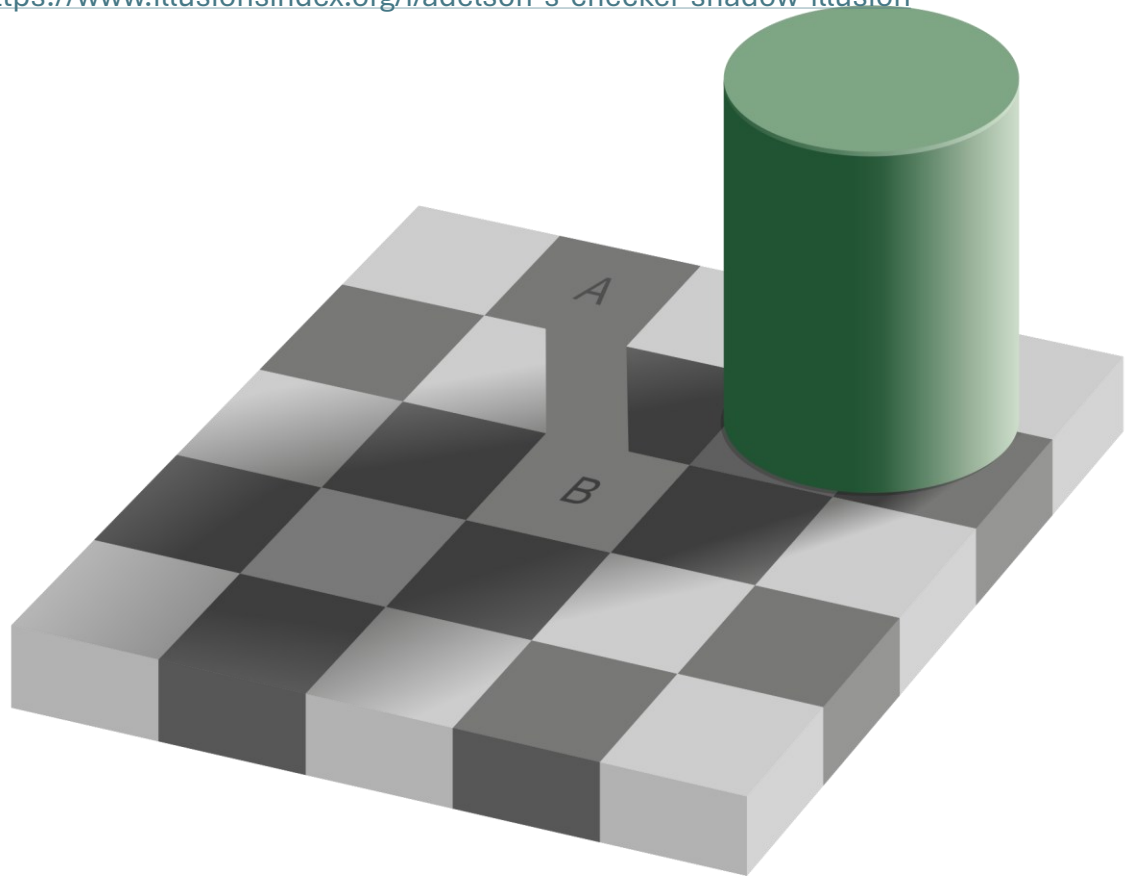
Original by *Edward H. Adelson and Pbroks13 [CC BY-SA 4.0]*.

Checker-board Illusion

https://www.illusionsindex.org/i/adelson-s-checker-shadow-illusion



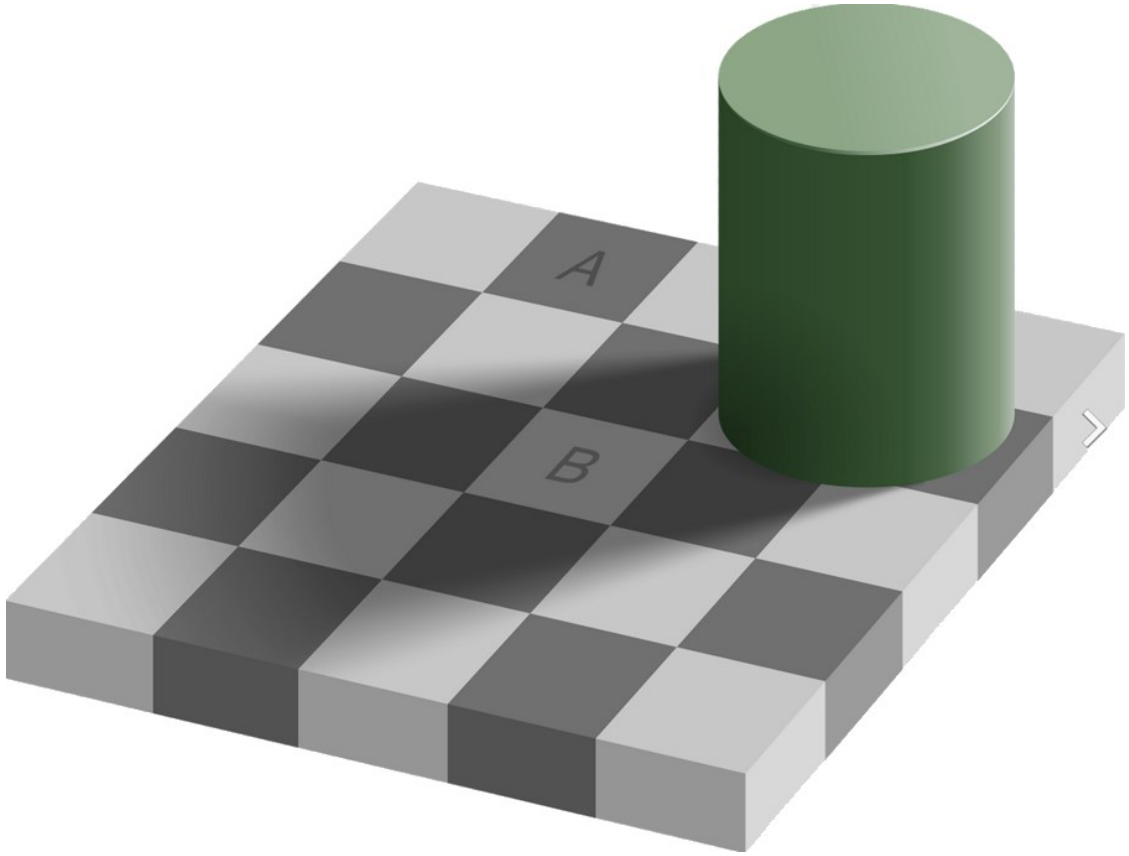
Original by *Edward H. Adelson and Pbroks13 [CC BY-SA 4.0]*.



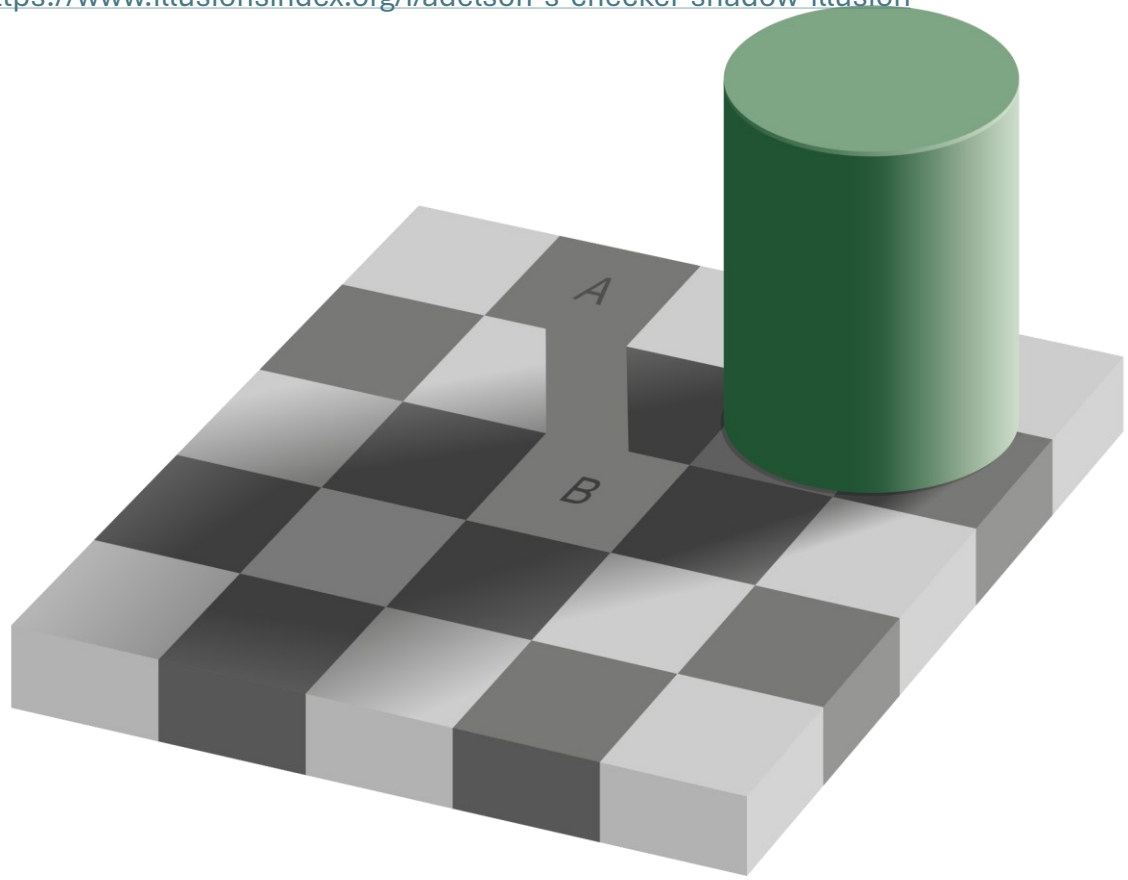
Original by Edward H. Adelson - created by Adrian Pingstone [copyrighted- free to use]

Checker-board Illusion

https://www.illusionsindex.org/i/adelson-s-checker-shadow-illusion



Original by *Edward H. Adelson and Pbroks13 [CC BY-SA 4.0]*.



Original by Edward H. Adelson - created by Adrian Pingstone [copyrighted- free to use]

Realistic rendering and shadows can affect the perception of the underlying data

Consider use carefully!

Scientific Colourmaps

Or why you should avoid rainbow....

Science and Technology Facilities Council

CoSeC
Computational Science Centre for Research Communities

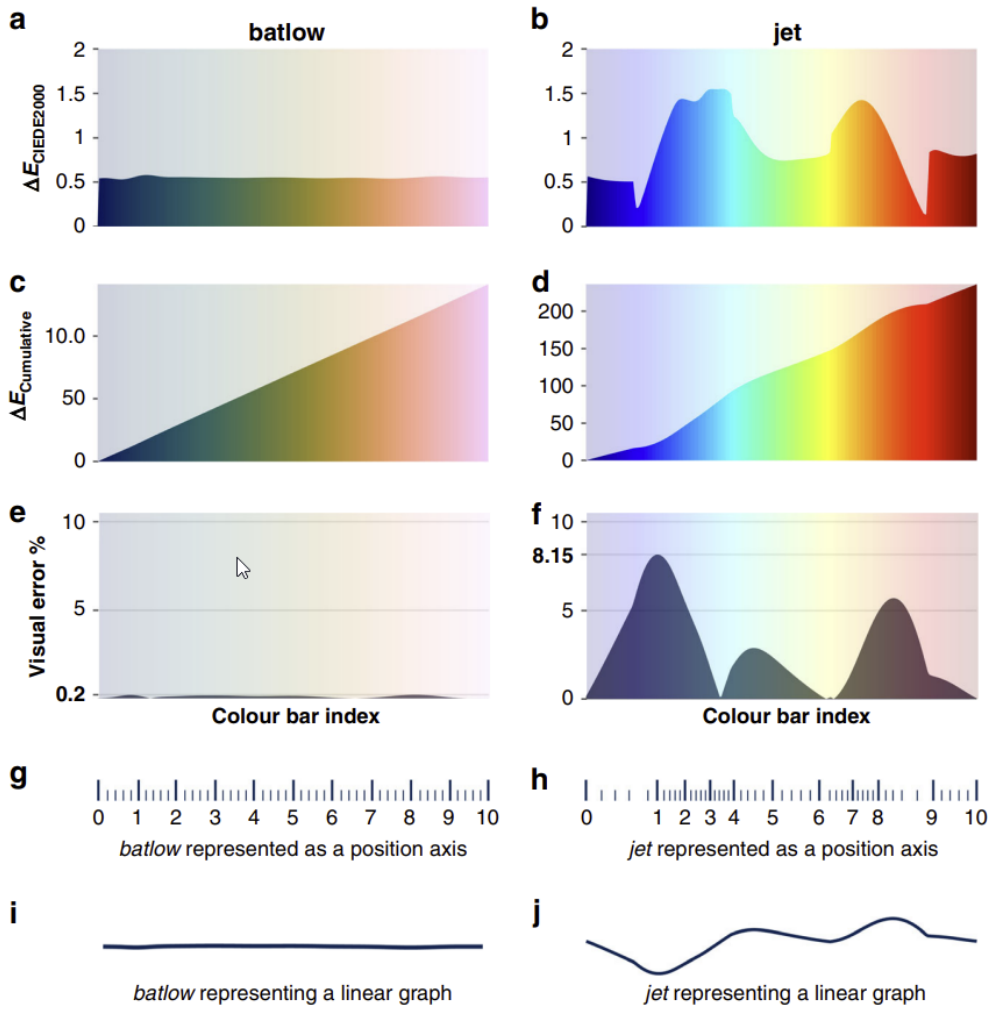
IMPERIAL

THE UNIVERSITY of EDINBURGH
School of Engineering

Gran ED

What is wrong with Jet/Rainbow?

- Comparison between a Jet/rainbow and a perceptually uniform colour map
  - **Top row:** incremental lightness difference between adjacent colours
  - **2nd row:** cumulative lightness difference
  - **3rd row:** Resulting visual error
  - **4th row:** Incremental data variation by colour gradient

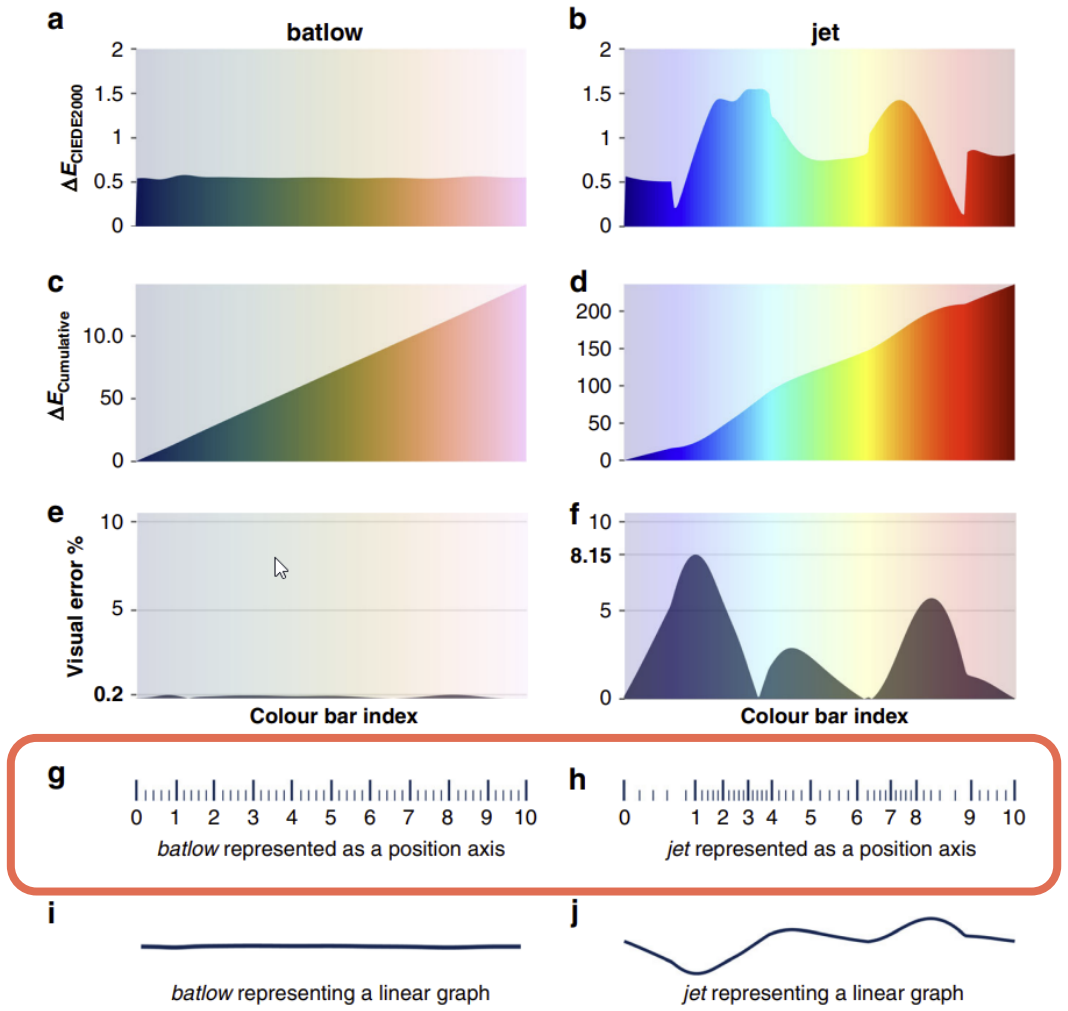


Crameri, F., Shephard, G.E. & Heron, P.J. The misuse of colour in science communication. *Nat Commun* **11**, 5444 (2020). https://doi.org/10.1038/s41467-020-19160-7

https://www.fabiocrameri.ch/colourmaps/

What is wrong with Jet/Rainbow?

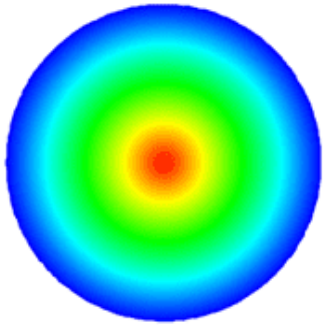
- Comparison between a Jet/rainbow and a perceptually uniform colour map
  - **Top row:** incremental lightness difference between adjacent colours
  - **2nd row:** cumulative lightness difference
  - **3rd row:** Resulting visual error
  - **4th row:** Incremental data variation by colour gradient



Crameri, F., Shephard, G.E. & Heron, P.J. The misuse of colour in science communication. *Nat Commun* **11**, 5444 (2020). https://doi.org/10.1038/s41467-020-19160-7

https://www.fabiocrameri.ch/colourmaps/

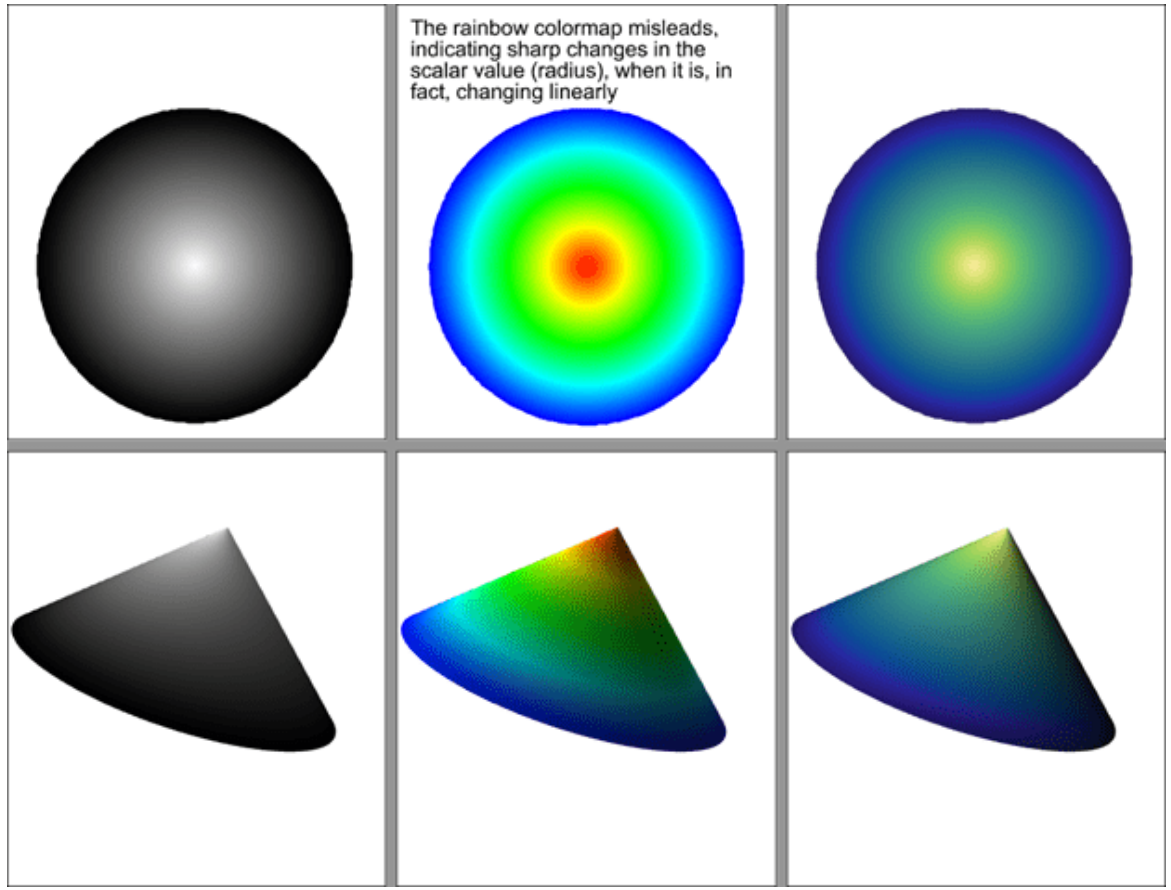
A Rainbow Example



- What does the figure tell you?
  - Maybe a series of shells or layers?

https://tecplot.com/2022/12/07/colormap-in-tecplot-360/

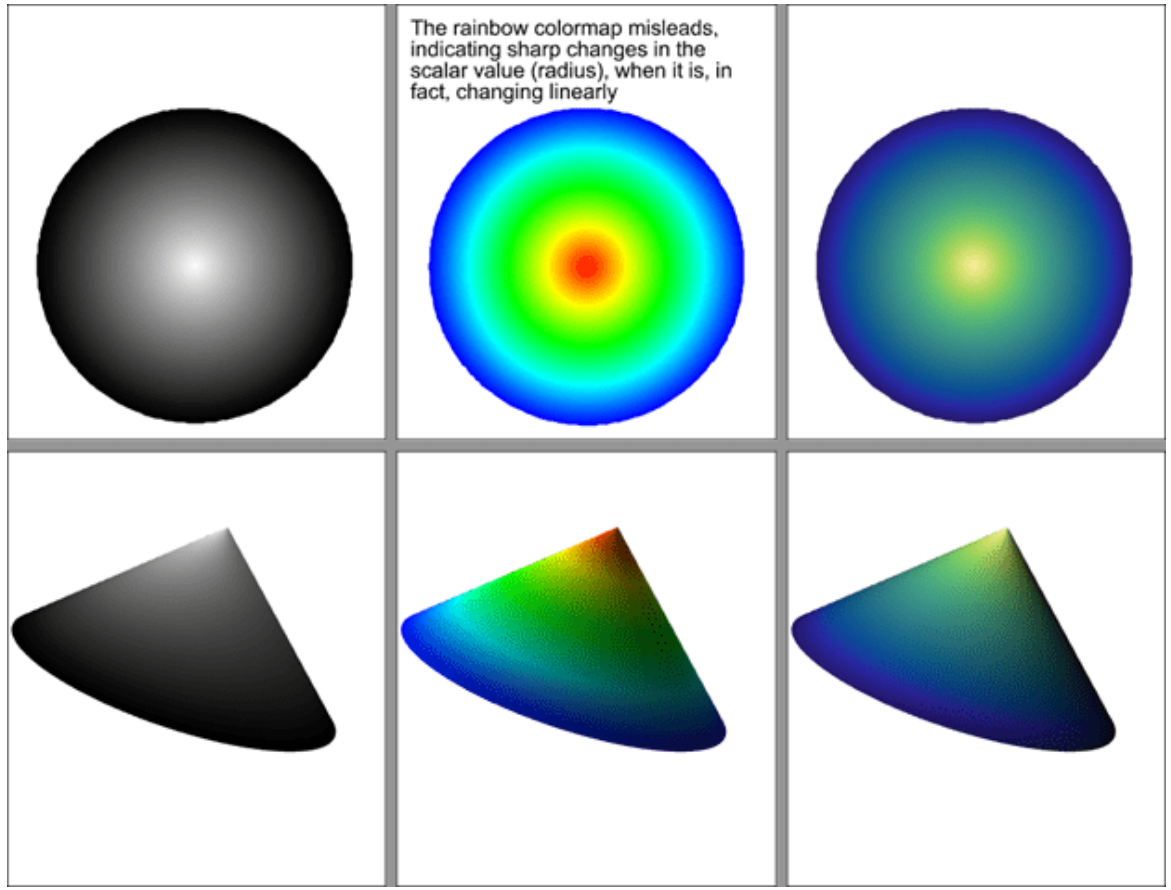
A Rainbow Example



https://tecplot.com/2022/12/07/colormap-in-tecplot-360/

- What does the figure tell you?
  - Maybe a series of shells or layers?

A Rainbow Example

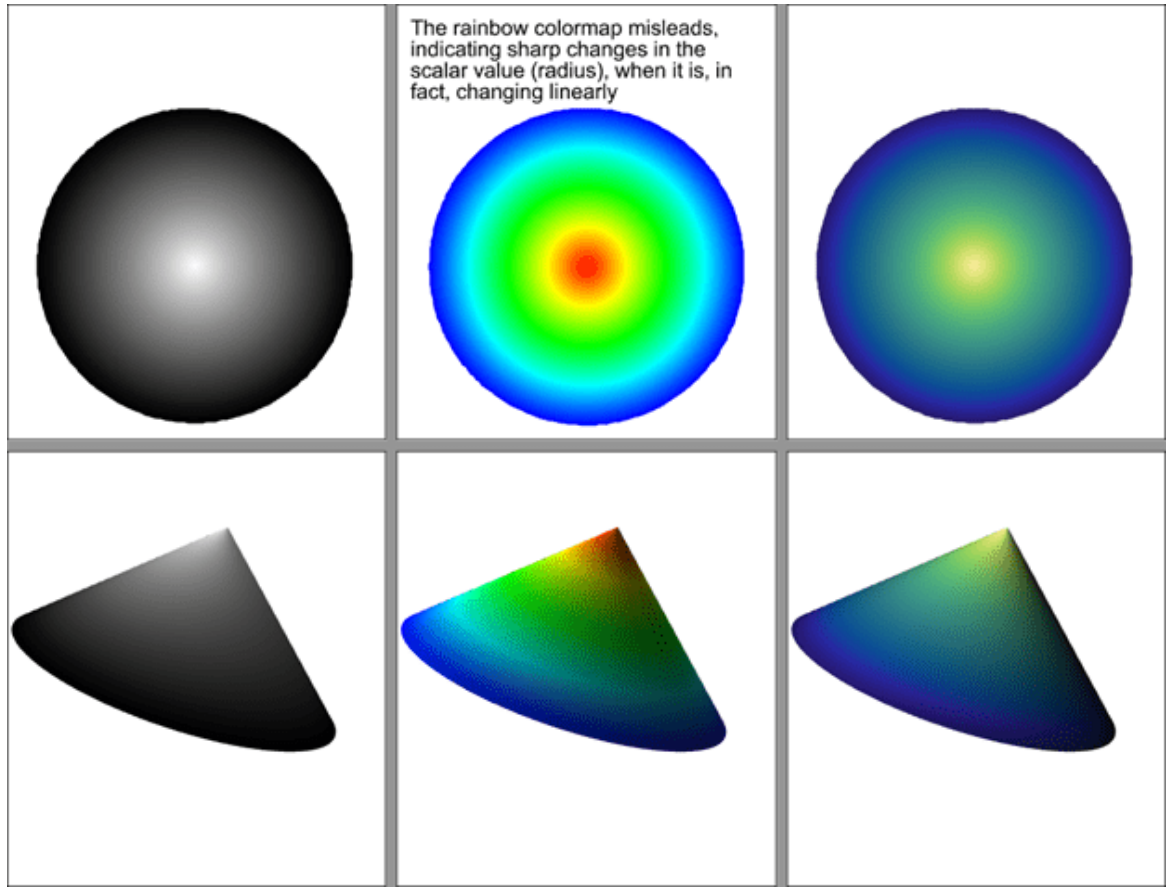


https://tecplot.com/2022/12/07/colormap-in-tecplot-360/

- What does the figure tell you?
  - Maybe a series of shells or layers?

- The colourmaps actually represent radius (distance from tip of cone)
  - Rainbow map shows several ridges in the data which is an incorrect representation of the underlying data

A Rainbow Example

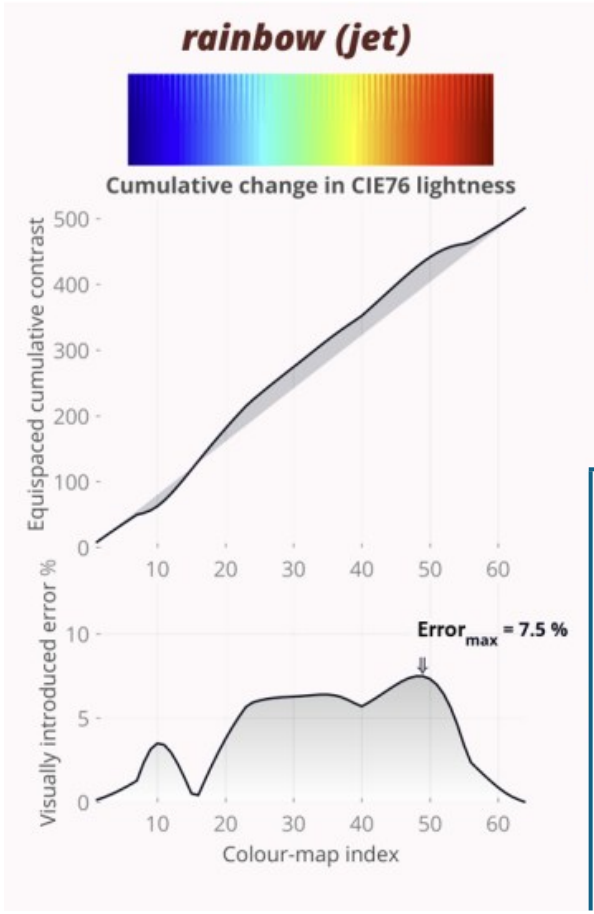


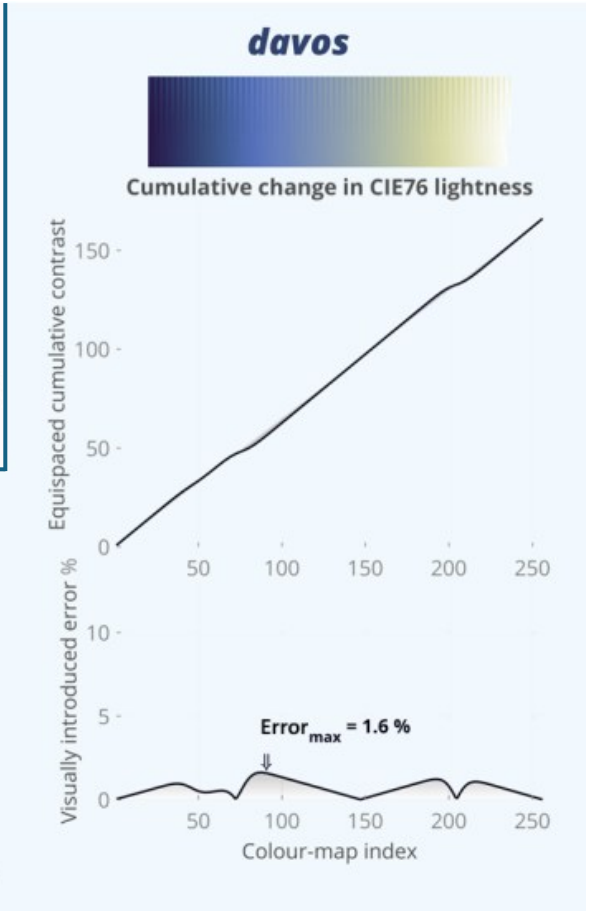
https://tecplot.com/2022/12/07/colormap-in-tecplot-360/

- What does the figure tell you?
  - Maybe a series of shells or layers?

- The colourmaps actually represent radius (distance from tip of cone)
  - Rainbow map shows several ridges in the data which is an incorrect representation of the underlying data
- Which of the representations is correct?

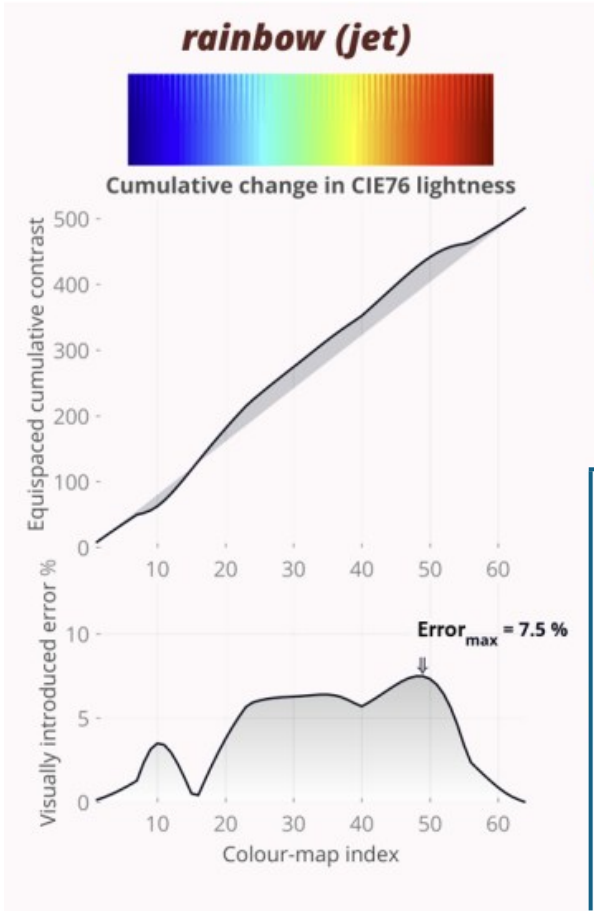
What This Means In Practice?

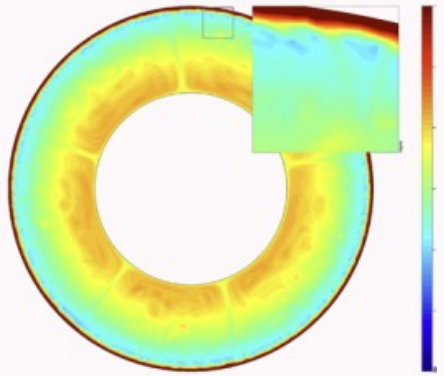


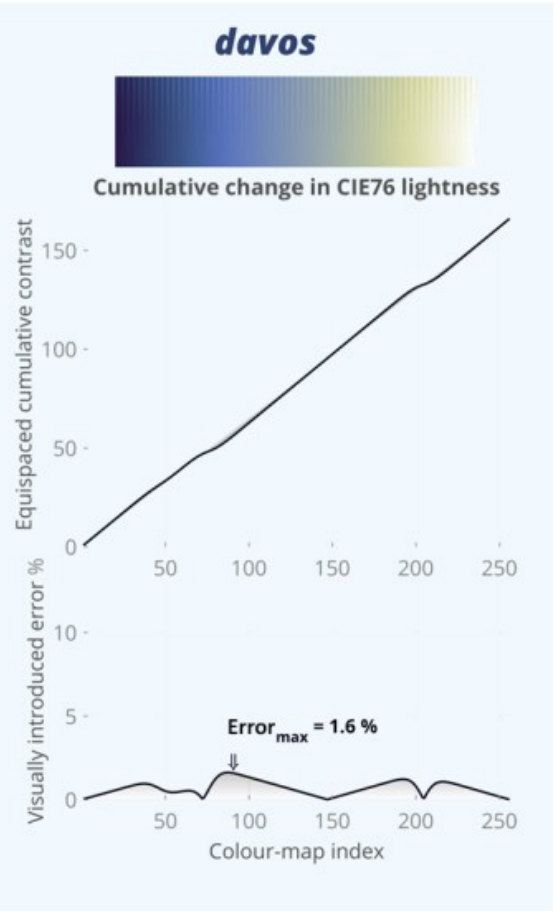


Crameri, F.: Geodynamic diagnostics, scientific visualisation and StagLab 3.0, Geosci. Model Dev., 11, 2541–2562, https://doi.org/10.5194/gmd-11-2541-2018, 2018.

What This Means In Practice?

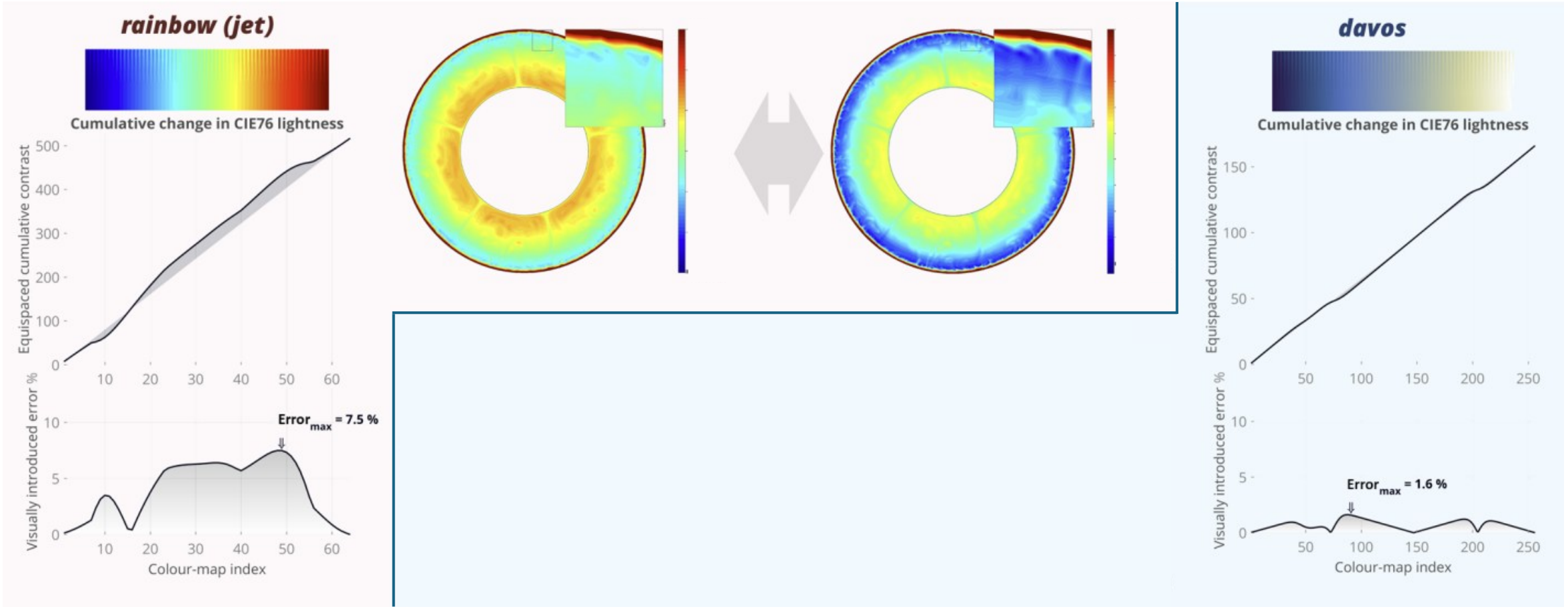






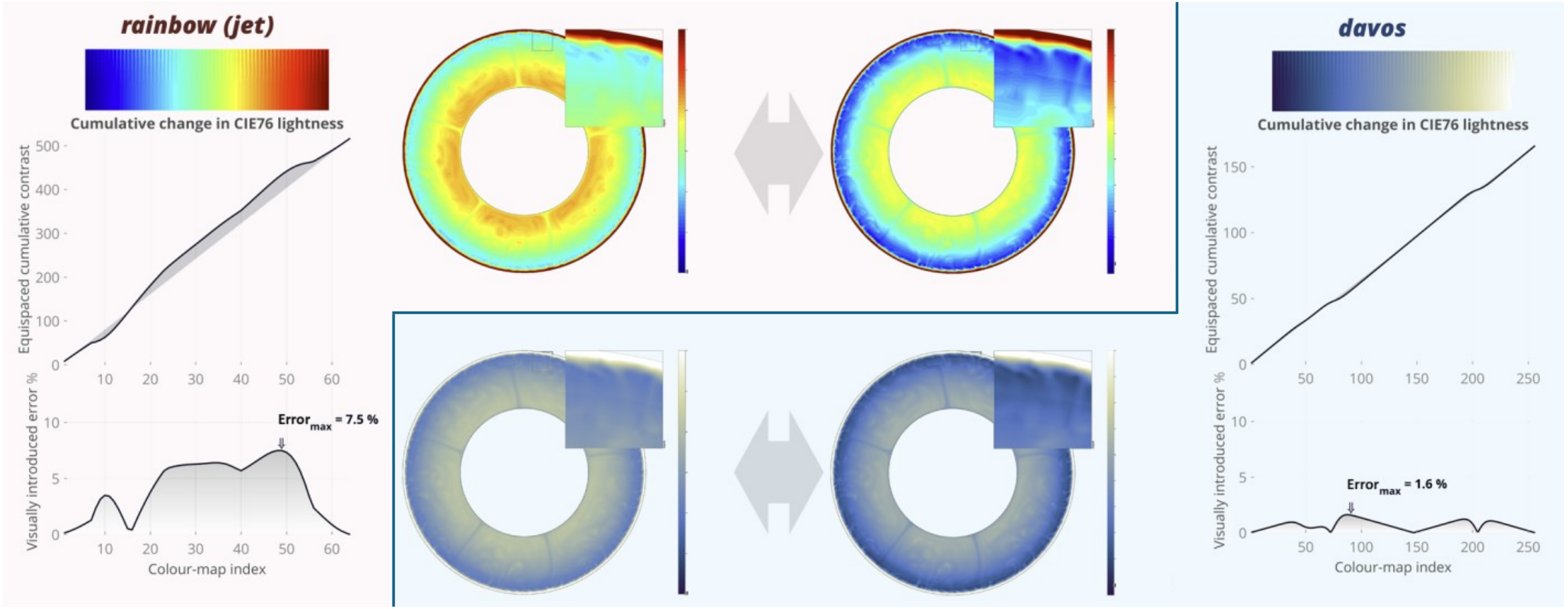
Crameri, F.: Geodynamic diagnostics, scientific visualisation and StagLab 3.0, Geosci. Model Dev., 11, 2541–2562, https://doi.org/10.5194/gmd-11-2541-2018, 2018.

What This Means In Practice?



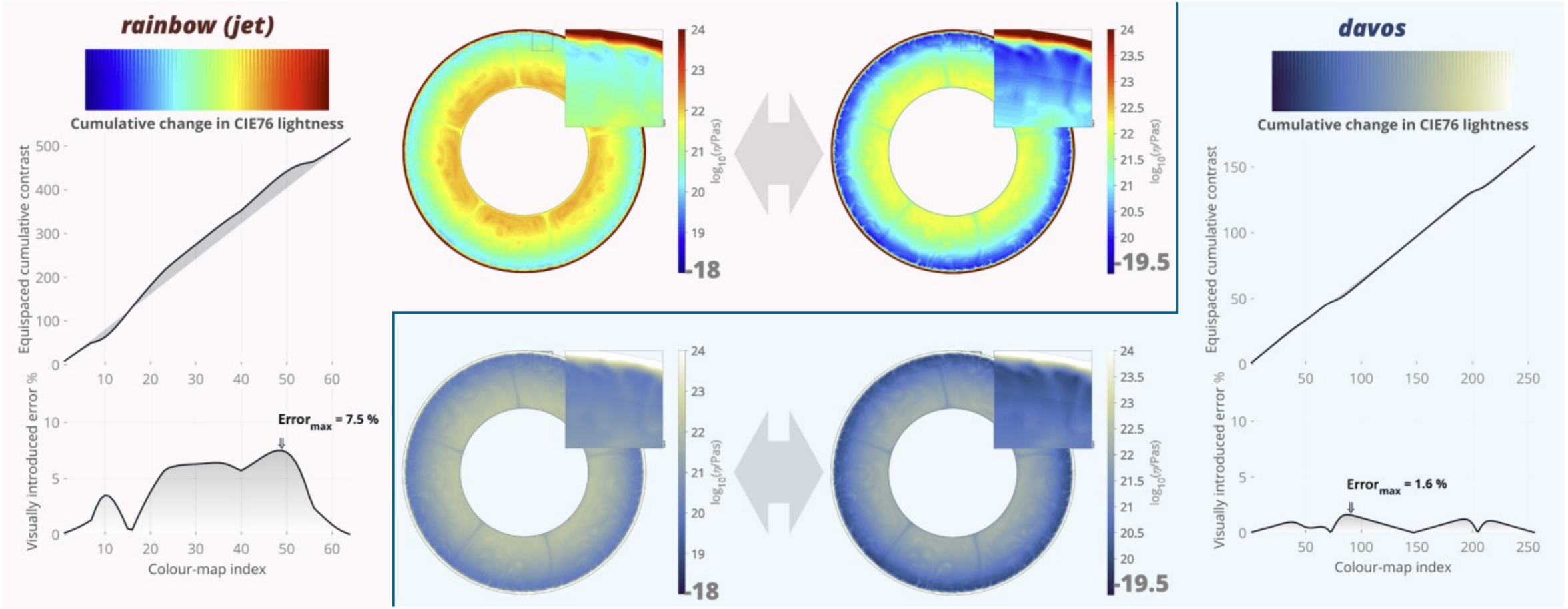
Crameri, F.: Geodynamic diagnostics, scientific visualisation and StagLab 3.0, Geosci. Model Dev., 11, 2541–2562, https://doi.org/10.5194/gmd-11-2541-2018, 2018.

What This Means In Practice?



Crameri, F.: Geodynamic diagnostics, scientific visualisation and StagLab 3.0, Geosci. Model Dev., 11, 2541–2562, https://doi.org/10.5194/gmd-11-2541-2018, 2018.

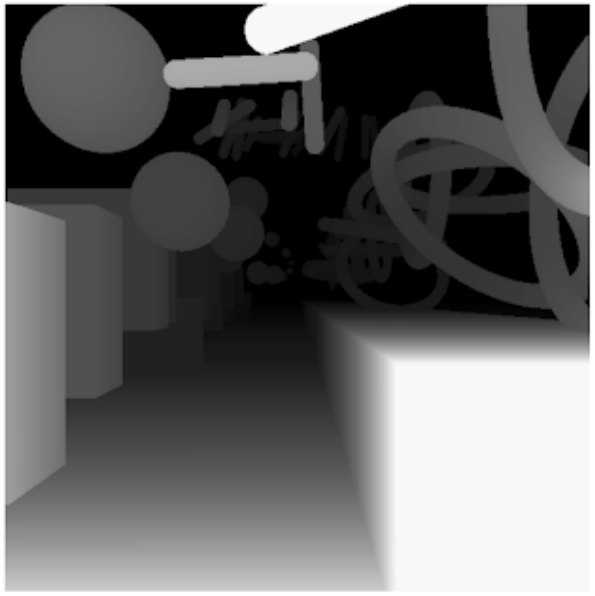
What This Means In Practice?



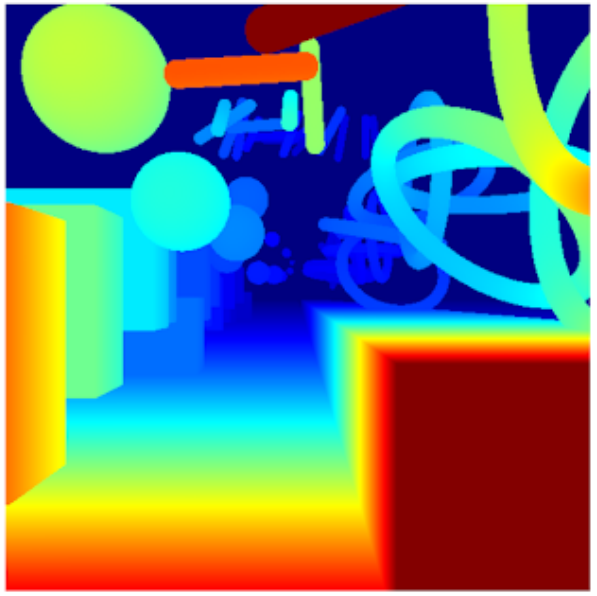
Crameri, F.: Geodynamic diagnostics, scientific visualisation and StagLab 3.0, Geosci. Model Dev., 11, 2541–2562, https://doi.org/10.5194/gmd-11-2541-2018, 2018.

A Rainbow-like Alternative - Turbo

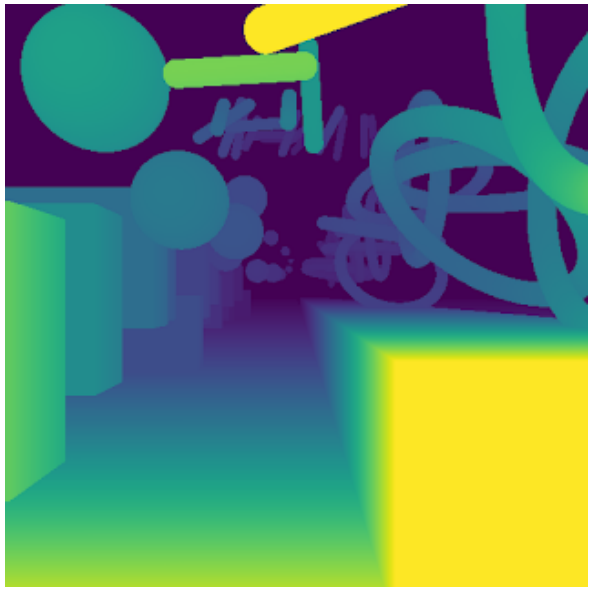
Grayscale



Rainbow



Viridis

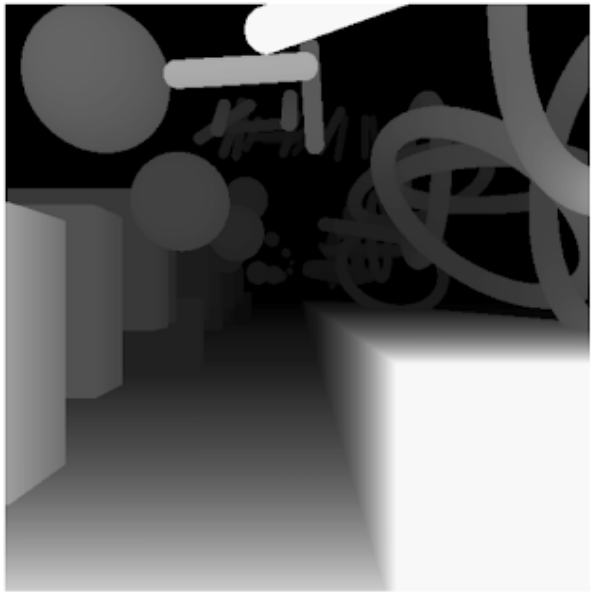


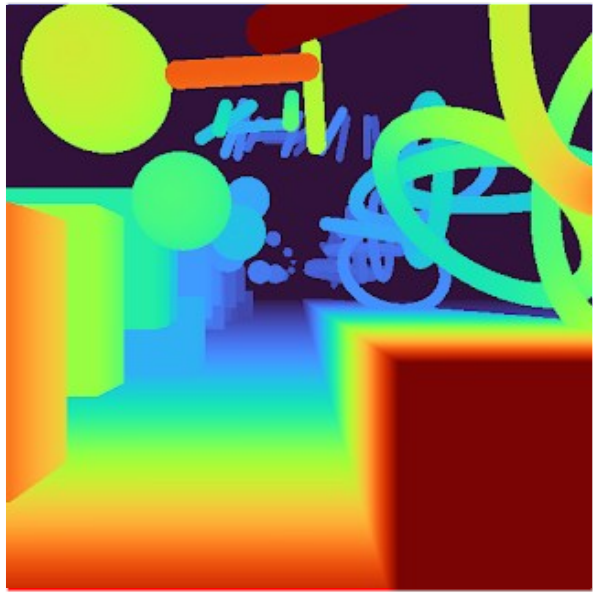
Inferno

“Turbo, An Improved Rainbow Colormap for Visualization”, Google research, https://research.google/blog/turbo-an-improved-rainbow-colormap-for-visualization/

A Rainbow-like Alternative - Turbo

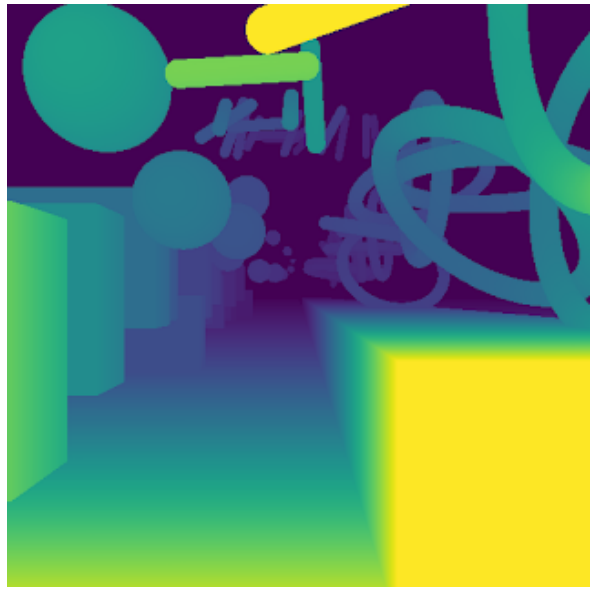
Grayscale





Turbo

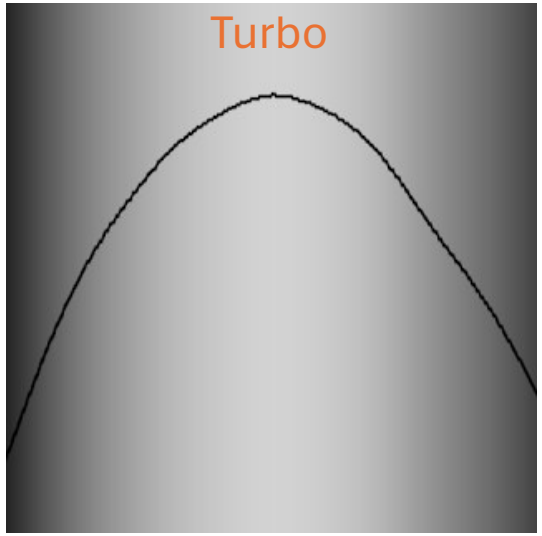
Viridis

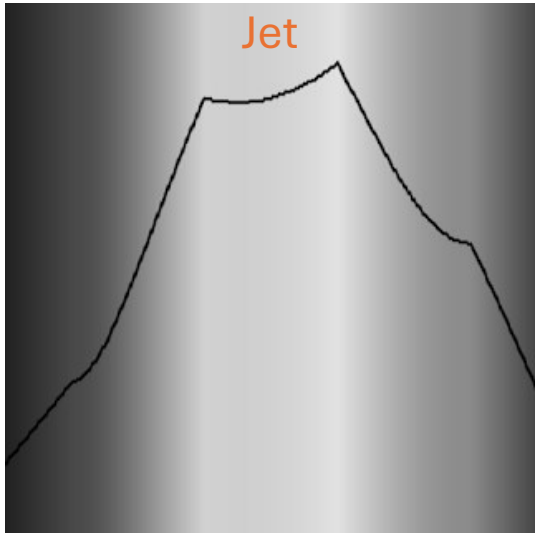


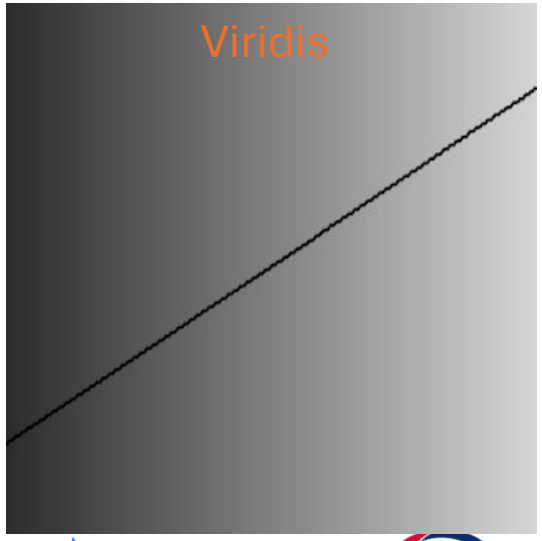
Inferno

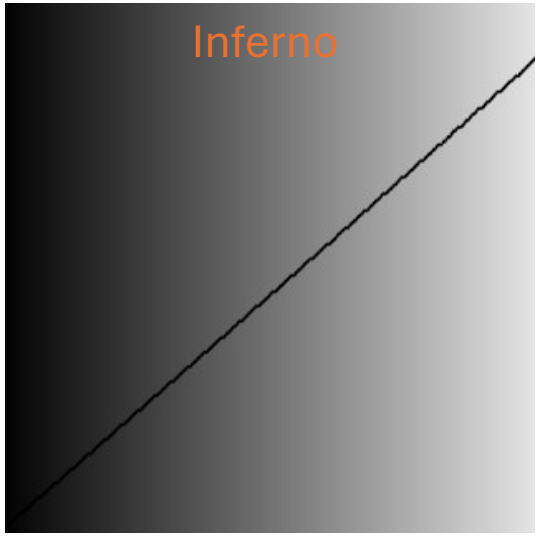
"Turbo, An Improved Rainbow Colormap for Visualization", Google research, https://research.google/blog/turbo-an-improved-rainbow-colormap-for-visualization/

Making The Case For Turbo...



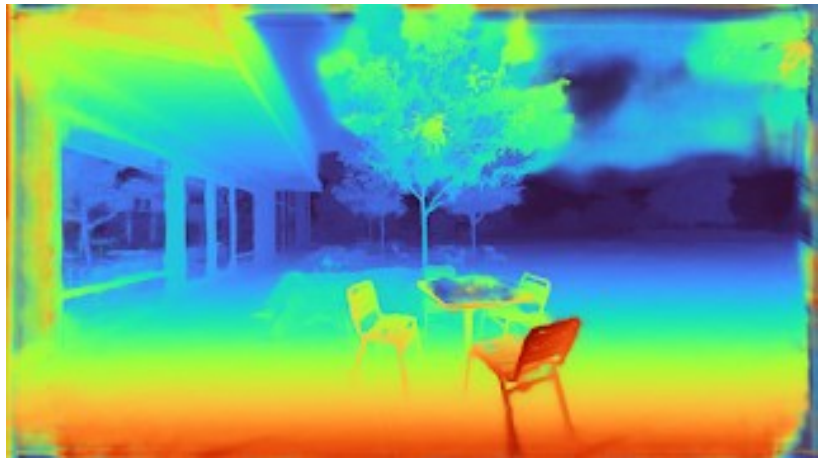






Inferno

Turbo



Asymmetry and non-linearity of turbo colourmap allow more detail in the image background to be distinguished

"Turbo, An Improved Rainbow Colormap for Visualization", Google research, https://research.google/blog/turbo-an-improved-rainbow-colormap-for-visualization/

Science and Technology Facilities Council

CoSeC Computational Science Centre for Research Communities

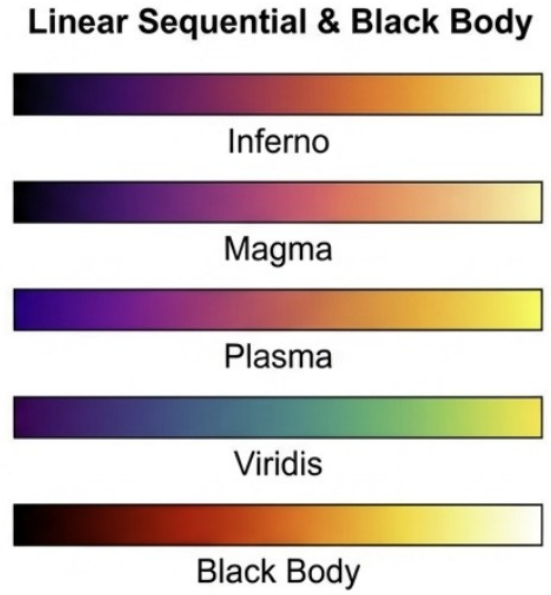
IMPERIAL

THE UNIVERSITY of EDINBURGH School of Engineering

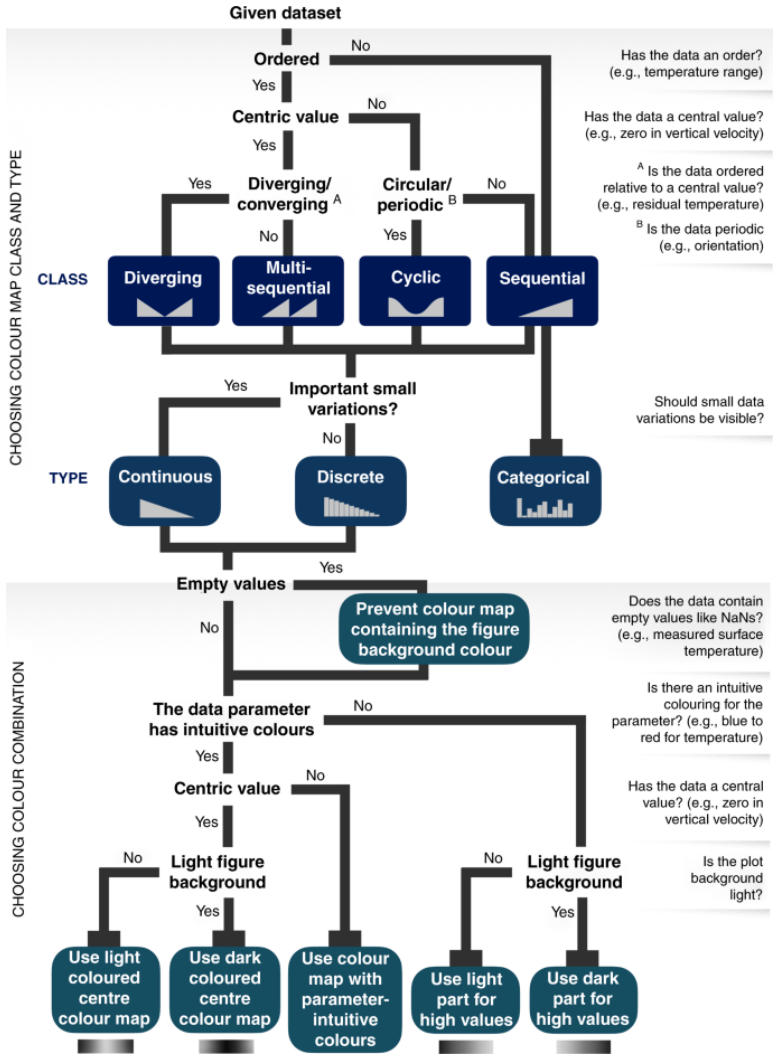
Gran ED

How To Choose A Colourmap

- Choose the right colourmap for the job
  - Continuous, discrete or categorical?
- Use diverging when you are trying to show data centred on zero or a mean value
- Use sequential when you have ordered data, with darker shades used to emphasise
- Use qualitative colourmaps for categorical data
- Avoid Rainbow/Jet as they introduce visual artefacts
- Beware of Parula in Matlab
  - Highly non-linear despite similarities to Viridis



How To Choose A Colourmap



- In case you are not good with decisions…
  - Crameri, F., Shephard, G.E. & Heron, P.J. The misuse of colour in science communication. *Nat Commun* **11**, 5444 (2020). https://doi.org/10.1038/s41467-020-19160-7

Visualisation Tools

- There is no single best option, and some relatively common ones are listed
- Open-source tools are combination of script and GUI-based visualisers
- Most support wide variety of common file formats
  - Many of the open-source tools are built upon the **VTK framework**

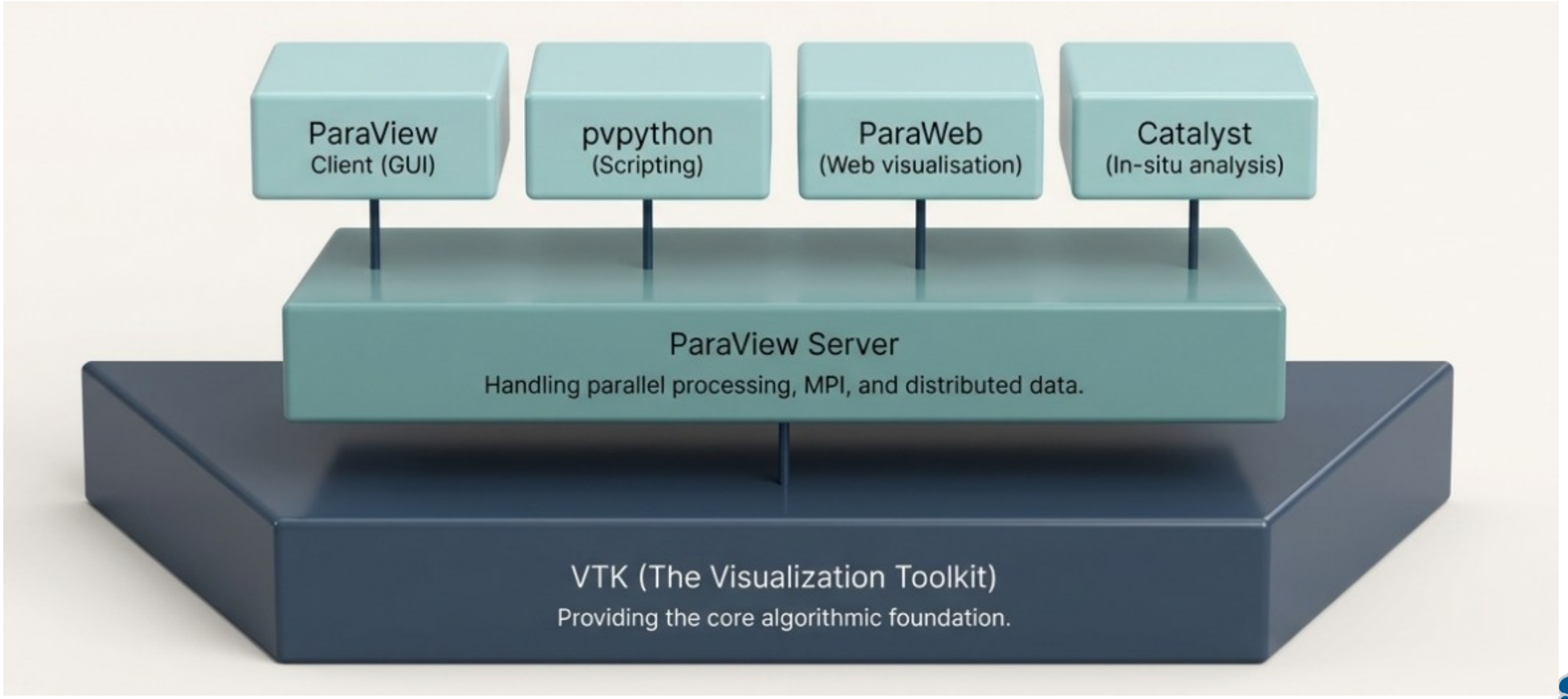
- **Commercial / Closed-source:**
  - Ensight
  - Ovito*
  - GiD
  - Tecplot
- **Open-source**
  - VisIt
  - ParaView
  - Ovito*
  - Blender
    - via BVTKNodes addon
  - Mayavi
  - PyVista
  - Vedo

  (Mayavi, PyVista, Vedo: Python libraries)
  - POV-Ray

*limited features in open-source version

The Visualisation Toolkit (VTK)

Scientific Visualisation Stack



What is the VTK Framework?

- The **Visualization Toolkit (VTK)** is an *"open-source, cross-platform system for 3D graphics, image processing, volume rendering, scientific visualization, and 2D plotting"*

- Development originally began at GE in 1993 by Schroder, Martin & Lorensen
- Founded Kitware Inc. in 1998
- Strong support from Sandia and Los Alamos National Labs
- ParaView development started in 2000

kitware VTK

Schroeder, Will; Martin, Ken; Lorensen, Bill (2006), *The Visualization Toolkit* (4th ed.), Kitware, ISBN 978-1-930934-19-1

What is the VTK Framework?

- The **Visualization Toolkit (VTK)** is an *"open-source, cross-platform system for 3D graphics, image processing, volume rendering, scientific visualization, and 2D plotting"*
- It supports a wide variety of visualization algorithms
- Takes advantage of both threaded and distributed memory parallel processing for speed and scalability

- Development originally began at GE in 1993 by Schroder, Martin & Lorensen
- Founded Kitware Inc. in 1998
- Strong support from Sandia and Los Alamos National Labs
- ParaView development started in 2000

kitware VTK

Schroeder, Will; Martin, Ken; Lorensen, Bill (2006), *The Visualization Toolkit* (4th ed.), Kitware, ISBN 978-1-930934-19-1

What is the VTK Framework?

- The **Visualization Toolkit (VTK)** is an *"open-source, cross-platform system for 3D graphics, image processing, volume rendering, scientific visualization, and 2D plotting"*
- It supports a wide variety of visualization algorithms
- Takes advantage of both threaded and distributed memory parallel processing for speed and scalability
- VTK is designed to be **platform agnostic** and has bindings in many languages such as Python and Java

- Development originally began at GE in 1993 by Schroder, Martin & Lorensen
- Founded Kitware Inc. in 1998
- Strong support from Sandia and Los Alamos National Labs
- ParaView development started in 2000

kitware VTK

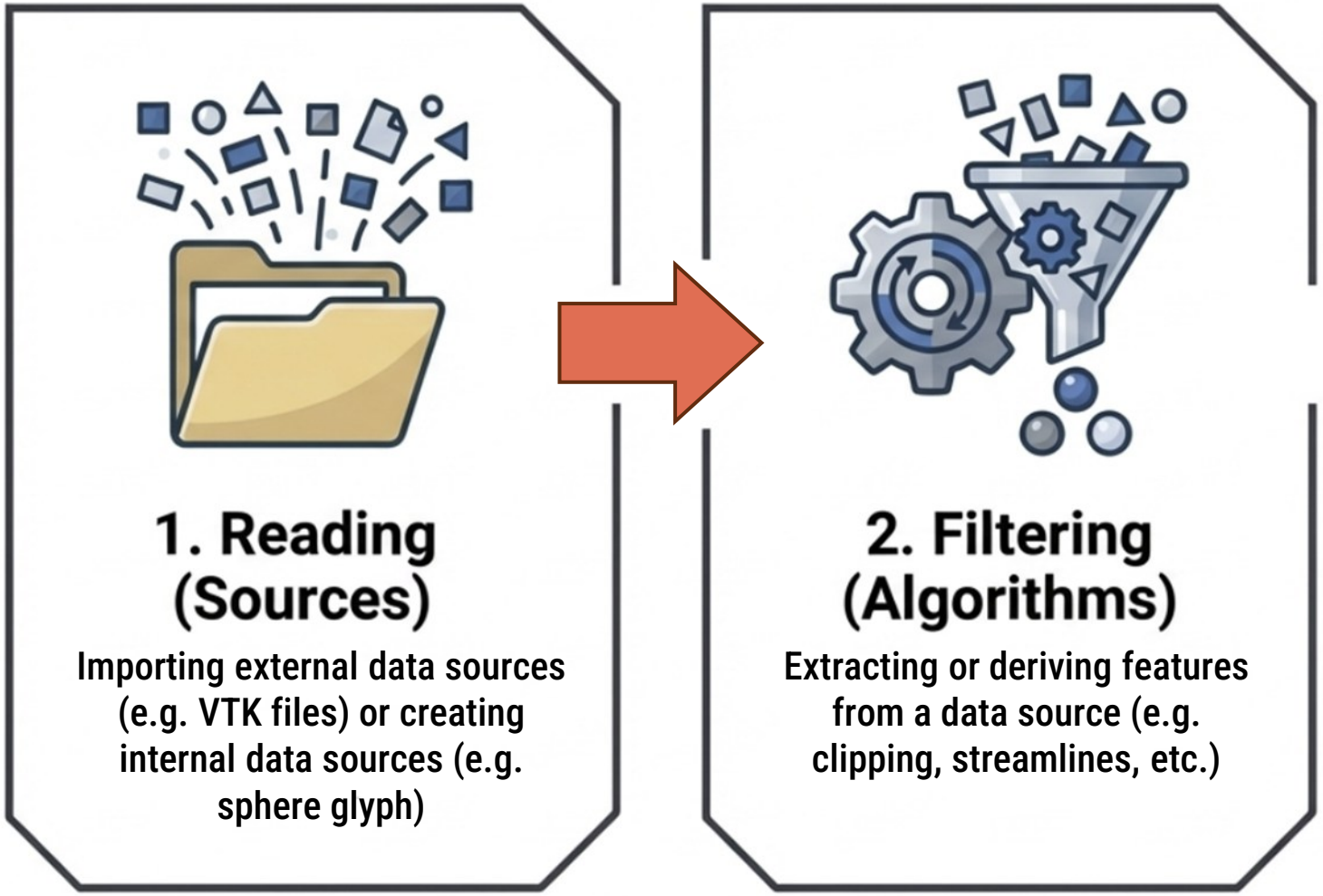
Schroeder, Will; Martin, Ken; Lorensen, Bill (2006), *The Visualization Toolkit* (4th ed.), Kitware, ISBN 978-1-930934-19-1

A Three-step Visualisation Process

1. Reading (Sources)

Importing external data sources (e.g. VTK files) or creating internal data sources (e.g. sphere glyph)

A Three-step Visualisation Process



1. Reading (Sources)

Importing external data sources (e.g. VTK files) or creating internal data sources (e.g. sphere glyph)

2. Filtering (Algorithms)

Extracting or deriving features from a data source (e.g. clipping, streamlines, etc.)

A Three-step Visualisation Process

1. Reading (Sources)

Importing external data sources (e.g. VTK files) or creating internal data sources (e.g. sphere glyph)

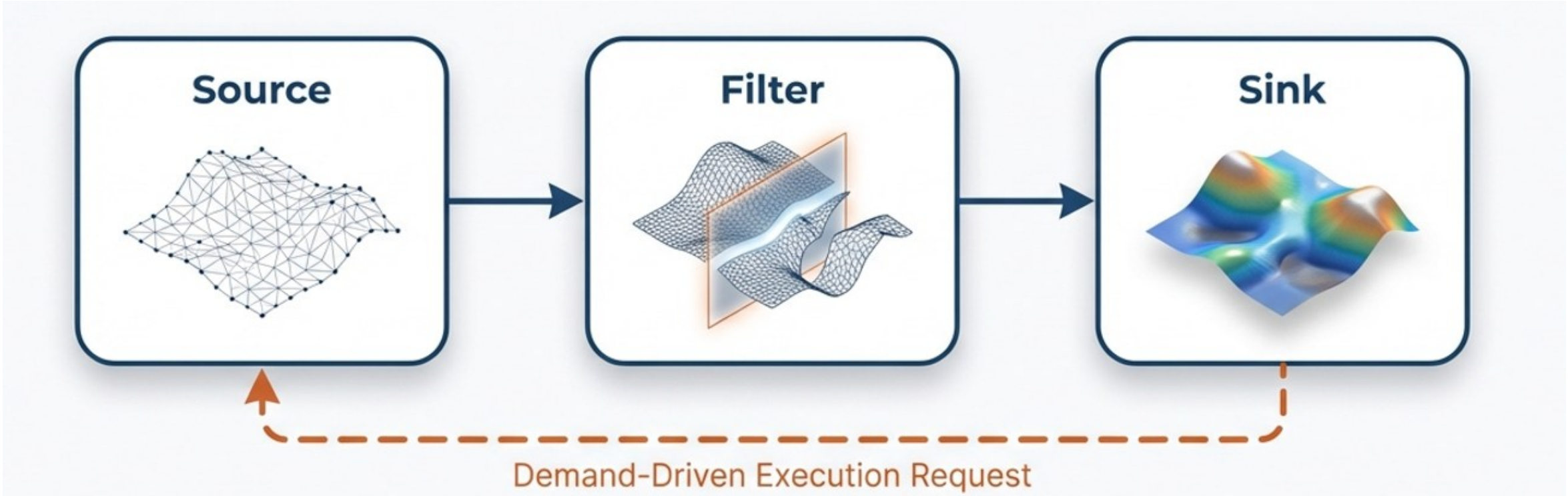
2. Filtering (Algorithms)

Extracting or deriving features from a data source (e.g. clipping, streamlines, etc.)

3. Rendering (Sinks/Views)

Converting the processed spatial data into viewable formats or images

Data Flow Architecture



Read: Sources ingest raw data

Filter: Data transformation algorithms extract key features

Render: Sinks converted the transformed data to graphics

Execute: The pipeline is executed on request

The VTK Data Format

VTK Data Format

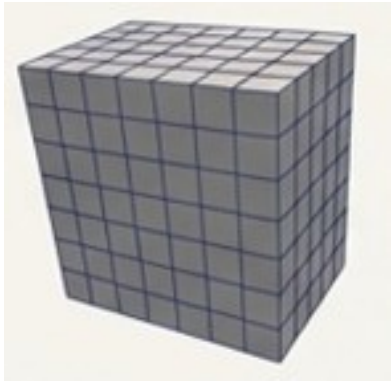
- **Structured Grids**: Regularly spaced grids where data is organised in a structured manner
  - **Image data:** A special case of structured grid where data is organized in a regular 3D array
  - **Rectilinear grid:** A structured grid where the spacing between points can vary along each axis
  - **Structured grid:** A general structured grid where points can be arranged in a regular pattern but may have varying spacing

- **Unstructured Grids**: Irregularly spaced grids that can represent complex geometries
  - **Unstructured grid:** A more general unstructured grid that can represent a wider variety of geometries, including tetrahedra, hexahedra, and other cell types
  - **Polydata:** An unstructured grid that represents geometric shapes, such as points, lines, and polygons

VTK Data Model – Structured Data

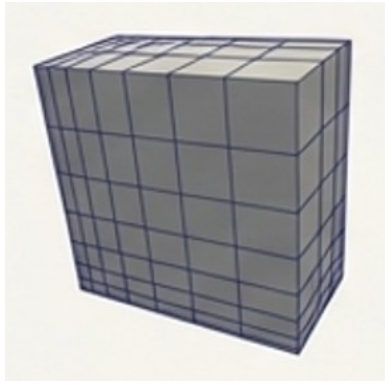
Uniform Rectilinear Grid (Image Data)

- Storage: Both topology and geometry are defined implicitly
- Performance: Because everything is implicit, it requires the least amount of storage



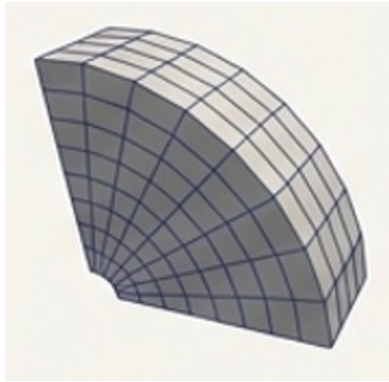
Rectilinear Grid

- Storage: Topology is defined implicitly, while point coordinates are semi-implicitly defined
- Performance: It offers significant memory savings over explicit types because the coordinate arrays are linear ($nx + ny + nz$) rather than volumetric



Curvilinear Grid (Structured Grid)

- Storage: Topology is implicit, but point coordinates are explicit
- Performance: While it preserves the compact memory footprint for topology, it requires more memory than rectilinear grids to store every point position explicitly

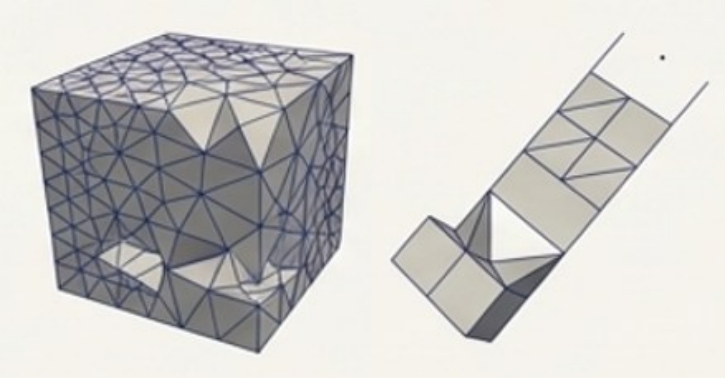


Topology refers to the connectivity, structure, and arrangement of data elements (cells and points), completely independent of their spatial location or geometry.

VTK Data Model – Unstructured Data

Unstructured Grid

- The **most general and flexible** data type
- It can represent any combination of 0D, 1D, 2D, and 3D cell types
- Storage: Both topology and point coordinates are stored explicitly
- Performance: Because all connectivity must be written out cell-by-cell, it has a much **higher memory footprint** than structured data



Polygonal Grid (Poly Data)

- It consists **strictly** of 0D cells (vertices), 1D cells (lines), and 2D cells (polygons/strips)
- It cannot represent 3D volumetric cells like tetrahedra
- Storage: A specialised version of an unstructured grid designed for efficient rendering
- Note: Polygonal data represents the basic rendering primitives for ParaView; most other data types must be converted to poly data before they can be displayed in a 3D view

VTK Data Format

- **Field Data:** Data associated with the current dataset – e.g. timestep, etc.
- **Point Data**: Data associated with individual points (each mesh node) in the dataset
- **Cell Data**: Data associated with cells (volumes) in the dataset, which can be used for visualizing interactions between particles

- **Attributes**: Additional data associated with points or cells
  - Typically, scalars, vectors, tensors, normals, texture coordiantes

Summary of File Formats

Data Type	Topology	Geometry	Memory Use	Flexibility	Extension	VTKHDF Support
Image Data	Implicit	Implicit	**Lowest**	Least (Regular spacing)	*.vti*	✓
Rectilinear	Implicit	Semi-Implicit	Low	Moderate (Variable axis spacing)	*.vtr*	✗
Curvilinear	Implicit	Explicit	Moderate	High (Arbitrary warping)	*.vts*	✗
Poly Data	Explicit	Explicit	High	High (Limited to 2D surfaces)	*.vtp*	✓
Unstructured	Explicit	Explicit	**Highest**	Greatest (Full 3D volume)	*.vtu*	✓

How Best to Store DEM Data?

- DEM data is unstructured in nature and **PolyData** offers the most efficient way of storing this data:
  - Particle information can easily be stored as point data (0D)
  - Contact information maps lines (1D - CellData)
  - Geometry and particle shape templates can easily be stored as polys (2D - CellData)

- More efficient than **UnstructuredGrid** as this requires an extra piece of topology information (cell type) in addition to the connectivity and offsets required to define each mesh
  - Although it sounds trivial, it becomes significant if you are storing a million extra integers for no good reason!

- Depending on the user/developer choice, particle information could be stored as either PointData or CellData

History of File Format

- Original file format (*.vtk*) consisted of ascii files with a specific structure
- Several limitations of this initial specification:
  - Compression not supported
  - Parallel writing not supported
  - Parsing bottleneck

- The `.vtk` format was considered deprecated from VTK 4.0 onwards which introduced the new XML-based format
  - Removed some features of older spec
  - Added compression support
  - Added parallel file format (.pvti, .pvtu, etc,)

- VTKHDF was introduced in VTK 9.2.x and is still undergoing development but will ultimately replace the XML format
  - Better read/write performance
  - Partial data loading

Storing Scientific Datasets

- Storing large datasets in ASCII is generally not a very good idea
  - Takes up more space
  - May suffer from precision errors
- Generally, binary formats should be preferred

- VTK XML files are valid XML files and can be parsed with any regular XML reader
  - **Caveat**: VTK supports one special case where raw binary data can be added to the appended section creating an **invalid XML file**
    - This means specialist readers are required making the files less portable – these should be avoided
  - Unfortunately, DEM codes like MFiX have chosen this approach
  - Fortunately, ParaView has a reader that support this

VTK XML Format – Ascii Example


```
<?xml version="1.0"?>
<VTKFile type="PolyData" version="1.0" byte_order="LittleEndian" header_type="UInt64">
  <PolyData>
  <Piece NumberOfPoints="20" NumberOfVerts="20" NumberOfLines="0" NumberOfStrips="0" NumberOfPolys="0">
    <PointData Scalars="1_temp" Vectors="3_force">
      <DataArray Name="1_temp" NumberOfComponents="1" type="Float64" format="ascii">
     0.8096282113891782 0.3047311952960293 0.6500663143518912 0.8456637563467384 0.3906199913247039 0.0493153326814327 0.0953282361963043 0.4089217432664466 0.9734786110287174 0.7813877948631190 0.0869568546620415 0.7417237568666443
0.5630292772867312 0.8719294300090183 0.8044927228429963 0.3760636376786568 0.1306022635272779 0.2908653129507012 0.3505188438702752 0.2901130193235927
      </DataArray>
      <DataArray Name="2_pressure" NumberOfComponents="1" type="Float64" format="ascii">
     0.5017704290104593 0.3785371246509415 0.0205158449338875 0.9907952905474608 0.1545654933391684 0.8782385634278128 0.6019489853133294 0.7906981230999223 0.2521929126363432 0.9716150255608206 0.0672474944937144 0.4673686197663716
0.0337000095394104 0.5347613958477050 0.1164155276297819 0.9550270647902264 0.5796338522273943 0.3412746271057875 0.8523204100536866 0.1972657806074329
      </DataArray>
      <DataArray Name="3_force" NumberOfComponents="3" type="Float64" format="ascii"> 0.0309346793427824 0.4092653096970628 0.9788367317195323 0.4223113198793231 0.0900433594418957 0.1787326345033293 0.2966289948749271
0.6954539758304935 0.2832782062952210 0.3251789785540518 0.7688481562679232 0.4766407185048503 0.9288534597190189 0.3341192036963977 0.9436476306446026 0.8789762057451738 0.0606108299327002 0.9254951873329387 0.7412746153715273
0.8109872417934331 0.4360667170346966 0.7233454095760591 0.6976210279118584 0.2955977110451682 0.0994604820813603 0.1875580422474142 0.8431228106210912 0.6213582623034001 0.6158399792301240 0.9406265592401716 0.8845641277007388
0.9350856253927768 0.2970236729050861 0.8712474915967705 0.6910999373713885 0.3187032732785566 0.0127516549008907 0.8755324934933010 0.5639212047451645 0.1967190822555136 0.9738387864540907 0.6568564085127088 0.0096719914074663
0.4238716923324118 0.3943384132270378 0.5395962035976353 0.3564427828885958 0.2408352500616302 0.7594055518045913 0.7632195886304822 0.4137798466081193 0.4414728807283862 0.5239588201936661 0.5330972627254891 0.9141526095541878
0.6308069665444429 0.5170567967508454 0.1320175579472910 0.2107269322401846 0.5006082952330471
      </DataArray>
    </PointData>
    <CellData>
    </CellData>
    <Points>
      <DataArray Name="Points" NumberOfComponents="3" type="Float64" format="ascii">
     0.4679781489940003 0.0539201795235295 0.8785521137838496 0.6662011241227854 0.6798039006229138 0.5179184112127887 0.7253673934646818 0.5500273845649549 0.9614782388606561 0.4565839967938599 0.5347282488901165 0.3175208696688832
0.1659279070777949 0.0767387959844464 0.7871958705242772 0.7152392271177217 0.6711429183806140 0.1804037800561749 0.0094476341085491 0.2176115805917300 0.6161777555253786 0.6766546248504622 0.4753594409825290 0.0787254199622341
0.2116737146235712 0.9337516804459353 0.9051117007473103 0.5781205664492428 0.1021111481025472 0.0771094947349589 0.3606986784532169 0.2044204606938260 0.9588362141721601 0.5373476404904579 0.0037939495868201 0.5440923148342509
0.0312391309332078 0.3297690748477242 0.8634945621439319 0.0935309527775745 0.9241324203495521 0.2405794370719853 0.5654258596612191 0.7982712785837751 0.4059594735292160 0.4933549421178993 0.0290714379783759 0.2925235663071808
0.4346748309322130 0.2692194427848057 0.7309894801048282 0.3220642954109557 0.9695141104130361 0.9700483721187488 0.0062263950599849 0.3090008720062422 0.8318148406832973 0.0737151178494130 0.8033901608302160 0.4849158872100066
      </DataArray>
    </Points>
    <Verts>
      <DataArray Name="connectivity" NumberOfComponents="1" type="Int32" format="ascii"> 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19   </DataArray>
      <DataArray Name="offsets" NumberOfComponents="1" type="Int32" format="ascii">  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20      </DataArray>
    </Verts>
    <Lines>
      <DataArray Name="connectivity" NumberOfComponents="1" type="Float64" format="ascii"> </DataArray>
      <DataArray Name="offsets" NumberOfComponents="1" type="Float64" format="ascii"> </DataArray>
    </Lines>
    <Strips>
      <DataArray Name="connectivity" NumberOfComponents="1" type="Float64" format="ascii"> </DataArray>
      <DataArray Name="offsets" NumberOfComponents="1" type="Float64" format="ascii"> </DataArray>
    </Strips>
    <Polys>
      <DataArray Name="connectivity" NumberOfComponents="1" type="Float64" format="ascii"> </DataArray>
      <DataArray Name="offsets" NumberOfComponents="1" type="Float64" format="ascii"> </DataArray>
    </Polys>
```

VTK XML Format – Binary Example


```
<?xml version="1.0"?>
<VTKFile type="PolyData" version="1.0" byte_order="LittleEndian" header_type="UInt64">
  <PolyData>
    <Piece NumberOfPoints="20" NumberOfVerts="20" NumberOfLines="0" NumberOfStrips="0" NumberOfPolys="0">
      <PointData Scalars="1_temp" Vectors="3_force">
        <DataArray Name="1_temp" NumberOfComponents="1" type="Float64" format="binary">
                oAAAAAAAAABgvDpseejpPw6Ed0W3gNM/kuwL31fN5D8Cfx1wrQ/rP2zR+f3q/9g/EMEd+ts/qT+oNtlobme4P/r4fBrGK9o/vym3nbwm7z/pxw36IAHpPwi+7+7NQrY/XxLNdTO85z+FYpT5VQTiP4DgSYzY5us/lGHPhWe+6T8g20I4bRH
        </DataArray>
        <DataArray Name="2_pressure" NumberOfComponents="1" type="Float64" format="binary">
                oAAAAAAAAADdZNbbgA7gP5ymrMbzOdg/oNUpDBsClT9G1D1TmLTvP7ysfVXNyMM/c0WAwoca7D8N/7iEKkPjP8EgdyZmTek/zpYDvu0j0D/tv+JkeBfvPziklr0hN7E/uK9EEl7p3T/gfLGtIEGhP9WISu7DHOE/mJ/qc2jNvT9kNELrlI/
        </DataArray>
        <DataArray Name="3_force" NumberOfComponents="3" type="Float64" format="binary">
                4AEAAAAAAAAgpVswV62fPwpRIiBnMdo/QHzbaKFS7z+4L+cOJgfbP9BgBuQUDbc/aLn1Abbgxj9aHwIu+PvSP6kVQrIoQeY/8Eft6Toh0j8oII99u8/UPxBX2HJnmug/xuV6EkiB3j/joAjkKrntP57cNYM1YtU/ZyAShFwy7j/yZzS1kiD
        </DataArray>
      </PointData>
      <CellData> </CellData>
      <Points>
        <DataArray Name="Points" NumberOfComponents="3" type="Float64" format="binary">
                4AEAAAAAAACI/0qfWvPdP7DHS/9sm6s/1RCRUhkd7D88ThUFhVHlP/nLGxz0wOU/dffEocmS4D+SfxCuNTbnP3eMkwfTmeE/Nxz3Am7E7j980oYVrDjdP21fp2p+HOE/CJ/BDUNS1D8cSTIrID3FP9htFlsnpbM/MFbuZLUw6T+
         </DataArray>
      </Points>
      <Verts>
        <DataArray Name="connectivity" NumberOfComponents="1" type="Int32" format="binary">
                UAAAAAAAAAAAAAAAAQAAAAIAAAADAAAABAAAAAUAAAAGAAAABwAAAAgAAAAJAAAACgAAAAsAAAAMAAAADQAAAA4AAAAPAAAAEAAAABEAAAASAAAAEwAAAA== </DataArray>
        <DataArray Name="offsets" NumberOfComponents="1" type="Int32" format="binary">
                UAAAAAAAAAABAAAAAgAAAAMAAAAEAAAABQAAAAYAAAAHAAAACAAAAAkAAAAKAAAACwAAAAwAAAANAAAADgAAAA8AAAAQAAAAEQAAABIAAAATAAAAFAAAAA== </DataArray>
      </Verts>
      <Lines>
        <DataArray Name="connectivity" NumberOfComponents="1" type="Float64" format="binary"> </DataArray>
        <DataArray Name="offsets" NumberOfComponents="1" type="Float64" format="binary"> </DataArray>
      </Lines>
      <Strips>
        <DataArray Name="connectivity" NumberOfComponents="1" type="Float64" format="binary"> </DataArray>
        <DataArray Name="offsets" NumberOfComponents="1" type="Float64" format="binary"> </DataArray>
      </Strips>
      <Polys>
        <DataArray Name="connectivity" NumberOfComponents="1" type="Float64" format="binary"> </DataArray>
        <DataArray Name="offsets" NumberOfComponents="1" type="Float64" format="binary"> </DataArray>
      </Polys>
    </Piece>
  </PolyData>
</VTKFile>
```

VTK XML Format – Multiblock File

- **VTM** stands for **VTK MultiBlock**
- It is an XML-based file format (`.vtm`) that describes a **static collection of datasets** (blocks), each of which can be a VTK file (e.g., `.vtp`, `.vtu`, etc.) or even another `.vtm` file
- Used to group multiple datasets into a single logical dataset (multiblock dataset)
- Does **not** encode time information

```
<?xml version="1.0"?>
<VTKFile type="vtkMultiBlockDataSet" version="1.0" byte_order="LittleEndian" header_type="UInt64">
  <vtkMultiBlockDataSet>
    <Block index="0" name="Geometries">
      <Piece index="0" name="Factory">
        <DataSet index="0" file="Factory_t_0.vtp"/>
      </Piece>
      <Piece index="1" name="Box">
        <DataSet index="0" file="Box_t_0.vtp"/>
      </Piece>
    </Block>
  </vtkMultiBlockDataSet>
</VTKFile>
```

ParaView Data File

- PVD stands for **ParaView Data**
- It is an XML-based file format (.pvd) that acts as a **meta-file** to describe **a time series** of datasets
- Lists datasets (e.g., .vtp, .vtu, .vtm, etc.) associated with specific time steps
- Used for **time-dependent simulations** to allow ParaView to load and animate data over time

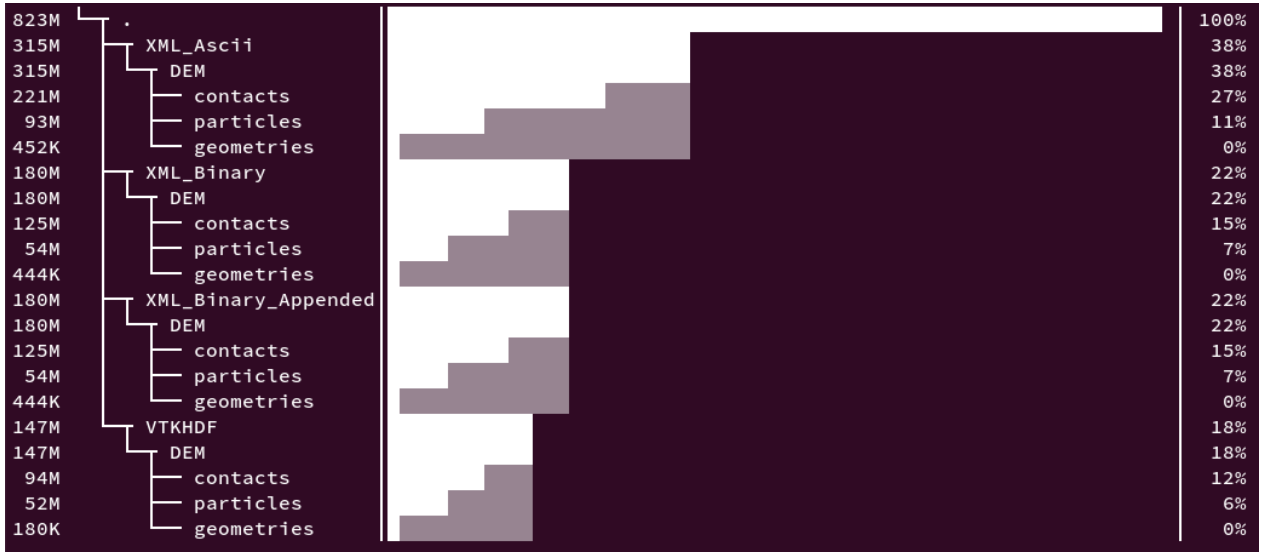
```
<?xml version="1.0"?>
<VTKFile type="Collection" version="0.1">
  <Collection>
    <!-- timestep: float value for time | file: path to the data file -->
    <DataSet timestep="0.0" group="" part="0" file="data_0.vtp"/>
    <DataSet timestep="0.5" group="" part="0" file="data_1.vtp"/>
    <DataSet timestep="1.0" group="" part="0" file="data_2.vtp"/>
  </Collection>
</VTKFile>
```

Note: In order to store the correct timestep data for a file, .pvd files are required for legacy VTK files. As of VTK 8.2, XML files support the embedding of time metadata as a `FieldData` array of `TimeValue`

Size of Various File Formats

- The amount of space used differs according to the file format
  - Ascii XML files are the largest and larger than legacy vtk files due to additional xml metadata
  - Binary XML is significantly smaller
  - VTKHDF is the most space efficient
- Read and write performance varies across files formats but VTKHDF offers significant performance increases

File Type / Method	Write Speed	Compression Ratio	Notes
Legacy VTK (.vtk)	465 MB/s	0.88	Significant overhead
VTK XML, none	256 MB/s	0.70	Significant overhead
VTK XML, zlib	105 MB/s	2.52	Default compression
VTK XML, lz4	401 MB/s	1.47	
VTK XML, lzma	9.93 MB/s	3.10	
VTK HDF (.vtkhdf), lvl0	1733 MB/s	0.93	**No compression**
VTK HDF (.vtkhdf), lvl4	137 MB/s	2.37	Default compression



ParaView Introduction

The ParaView Interface

Data Variable Controls

Common Filters

Pipeline Browser showing all sources

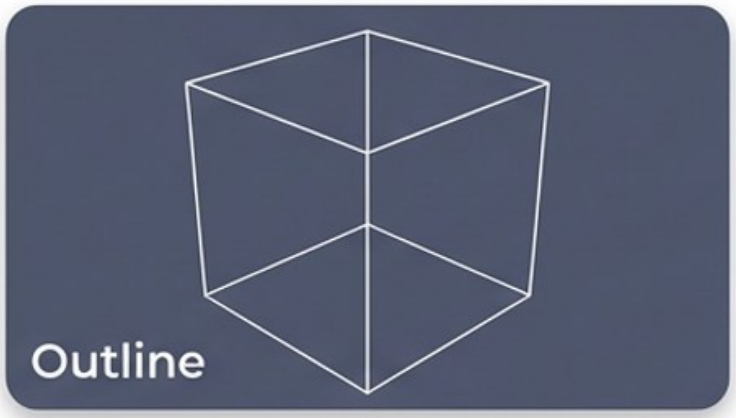
Properties and **Information** Panels

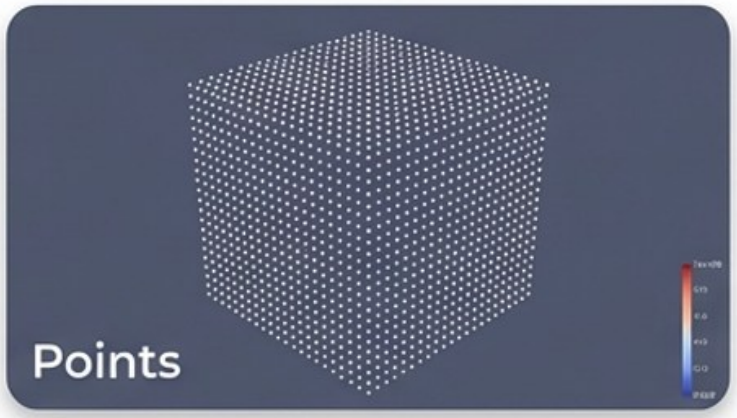
Time Controls

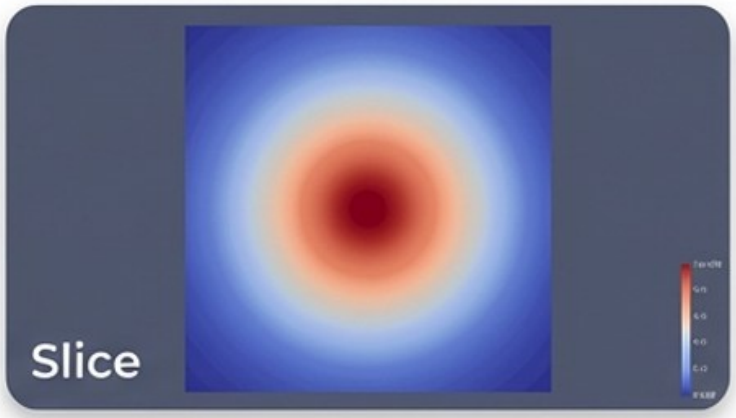
3D Viewport Controls

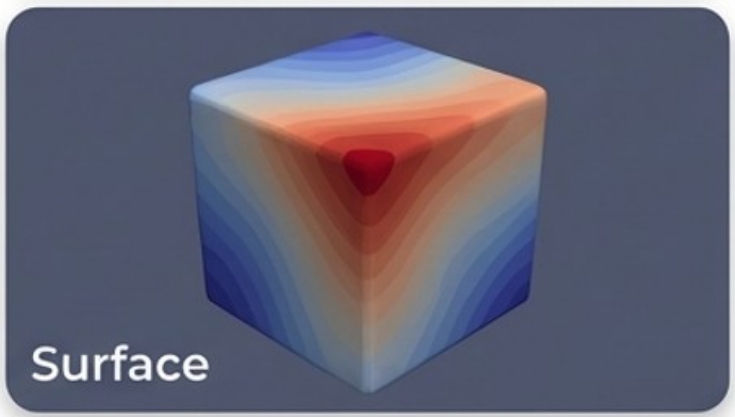
Visualisation Layouts can be renders or plots

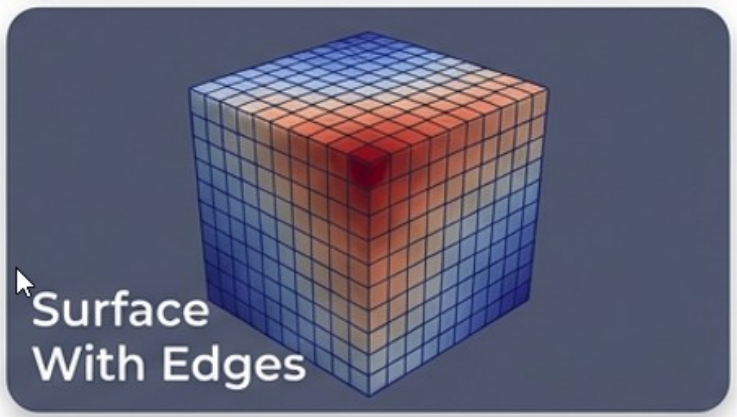
Multiple Representations of Datasets

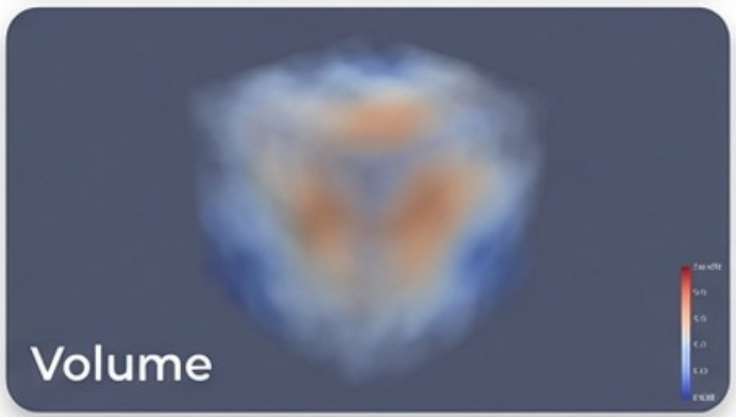






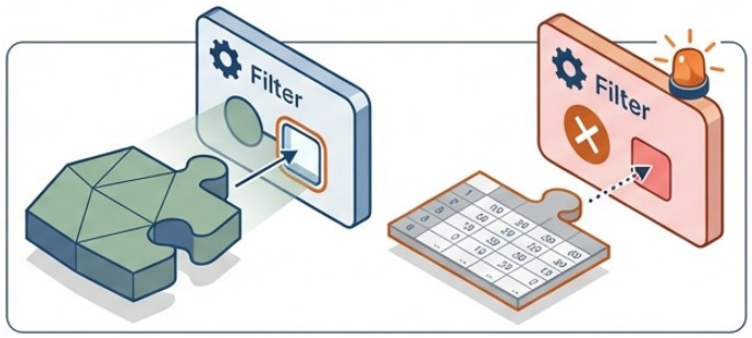






Common Filters

- VTK / ParaView has a very long list of filters
- Filters only work for specific data types
- If the filter is unavailable for a source, it will be disabled in the GUI

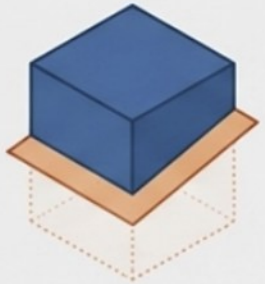


AMR Connectivity
AMR Contour
AMR Dual Clip
AMR Fragment Integration
AMR Fragments Filter
Adaptive Resample To Image
Add Field Arrays
Aggregate Dataset
Align Image Origins
Angular Periodic Filter
Animate Modes
Annotate Attribute Data
Annotate Global Data
Annotate Selection
Annotate Time Filter
Append Arc Length
Append Attributes
Append Datasets
Append Geometry
Append Location Attributes
Append Molecule
Append Reduce
Axis-Aligned Reflection Filter
Axis-Aligned Slice
Axis-Aligned Transform Filter
Block Ids
Block Scalars
Boundary Mesh Quality
Bounding Ruler
Calculator
Cell Centers
Cell Data to Point Data
Cell Grid To Unstructured Grid
Cell Grid Transform
Cell Grid Warp
Cell Quality
Cell Size
CellGrid Centers
CellGrid Elevation
CellGrid Probe
Clean
Clean Cells to Grid
Clean to Grid
Clip
Clip Closed Surface
Clip Generic Dataset
Compute Connected Surface Properties
Compute Derivatives
Compute Molecule Bonds
Compute Quartiles
Compute Sides
Connected Surface Properties
Connectivity
Contingency Statistics
Contour
Contour Generic Dataset
Convert AMR dataset to Multi-block
Convert Into Molecule
Convert Polyhedral Cells
Convert To Cell Grid
Convert To MultiBlock
Convert To PartitionedDataSetCollection
Convert To Point Cloud
Coordinates
Count Cell Faces
Count Cell Vertices
Critical Time
Curvature
D3
Date To Numeric
Decimate
Decimate Polyline
Deflect Normals
Delaunay 2D
Delaunay 3D
Descriptive Statistics
Elevation
Environment Annotation
Equalizer Filter
Evenly Spaced Streamlines 2D
Explode Data Set
Extract AMR Blocks
Extract Block
Extract CTH Parts
Extract Cells Along Line
Extract Cells Along Lines Custom Source
Extract Cells By Region
Extract Cells By Type
Extract Component
Extract Edges
Extract Enclosed Points
Extract Generic Dataset Surface
Extract Ghost Cells
Extract Location
Extract Particles Over Time
Extract Region Surface
Extract Selection
Extract Subset
Extract Subset With Seed
Extract Surface
Extract Time Steps
FFT Of Selection Over Time
Feature Edges
Feature Edges Region Ids
Field Arrays From File
Field Data to Attribute
Finite Element Field Distributor
Force Static Mesh
Force Time
Gaussian Resampling
Generate Global Ids
Generate Ids
Generate Process Ids
Generate Quadrature Points
Generate Quadrature Scheme Dictionary
Generate Spatio Temporal Harmonics
Generate Surface Normals
Generate Surface Tangents
Generate Time Steps
Ghost Cells
Global Point And Cell Ids
Glyph
Glyph With Custom Source
Gradient
Group Datasets
Group Time Steps
Histogram
Histogram 2D
HyperTreeGrid Axis Reflection
HyperTreeGrid Cell Centers
HyperTreeGrid Depth Limiter
HyperTreeGrid Evaluate Coarse
HyperTreeGrid Feature Edges Filter
HyperTreeGrid Generate Fields
HyperTreeGrid Geometry Filter
HyperTreeGrid Ghost Cells Generator
HyperTreeGrid To Dual Grid
HyperTreeGrid To UnstructuredGrid
HyperTreeGrid Visible Leaves Size
Image Data To AMR
Image Data To Uniform Grid
Image Data to Point Set
Integrate Variables
Interpolate to Quadrature Points
Intersect Fragments
Iso Volume
K Means
Level Scalars(Overlapping AMR)
Linear Cell Extrusion
Linear Extrusion
Loop Subdivision
Mask Points
Material Interface Filter
Median
Merge Blocks
Merge Time
Merge Vector Components
Mesh Quality
Molecule To Lines
Multicorrelative Statistics
Normal Glyphs
OME TIFF Channel Calculator
OT Density Map
OT Kernel Smoothing
Octree Image to PointSet
Outline
Outline Corners
Outline Curvilinear DataSet
Overlapping AMR Level Ids
Overlapping Cells Detector
PCA Normal Estimation
Particle Path
Particle Tracer
Partition Balancer
Pass Arrays
Perlin Noise
Plot Data
Plot Data Over Time
Plot Global Variables Over Time
Plot On Intersection Curves
Plot On Sorted Lines
Plot Over Line
Plot Over Lines From Custom Source
Plot Selection Over Time
Point And Cell Ids
Point Data to Cell Data
Point Dataset Interpolator
Point Line Interpolator
Point Plane Interpolator
Point Volume Interpolator
PointSet To Octree Image
Polyline Length
Principal Component Analysis
Probe Location
Process Ids
Programmable Annotation
Programmable Filter
Python Annotation
Python Calculator
Quadric Clustering
Quadric Decimation
Random Attributes
Random Vectors
Rectilinear Data to Point Set
Rectilinear Grid Connectivity
Redistribute DataSet
Reflect
Remove Ghost Information
RenameArrays
Resample AMR
Resample To Image
Resample To Line
Resample With Dataset
Reverse Sense
Ribbon
Rotational Extrusion
SPH Dataset Interpolator
SPH Line Interpolator
SPH Plane Interpolator
SPH Volume Interpolator
Scatter Plot
Shrink
Slice
Slice Along PolyLine
Slice Generic Dataset
Slice With Plane
Smooth
Spatio Temporal Harmonics
Stitch Image Data With Ghosts
Streak Line
Stream Tracer
Stream Tracer For Generic Datasets
Stream Tracer With Custom Source
Subdivide
Surface Flow
Surface Normals
Surface Tangents
Surface Vectors
Synchronize Time
Table FFT
Table To Points
Table To Structured Grid
Temporal Array Operator
Temporal Cache
Temporal Interpolator
Temporal Particles To Pathlines
Temporal Shift Scale
Temporal Smoothing
Temporal Snap-to-Time-Step
Temporal Statistics
Tensor Glyph
Tensor Principal Invariants
Tessellate
Tetrahedralize
Texture Map to Cylinder
Texture Map to Plane
Texture Map to Sphere
Threshold
Threshold Table
Time Step Progress Bar
Transform
Transpose Table
Triangle Strips
Triangulate
Tube
Validate Cells
Volume Of Revolution
Vortex Cores
Warp By Scalar
Warp By Vector
Yield Criteria
Youngs Material Interface

Common Filters – Part I

Clip

Cuts geometry with a half-space (removes one side).

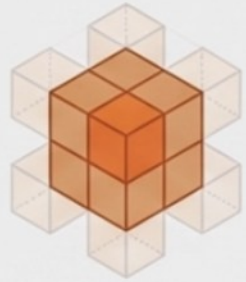


Slice

Intersects geometry with a 2D plane.

Threshold

Extracts specific cells based on a scalar value range.

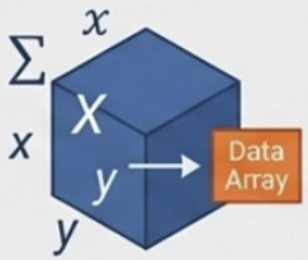


Contour

Extracts surfaces where a scalar field matches a specific value (isosurfaces).

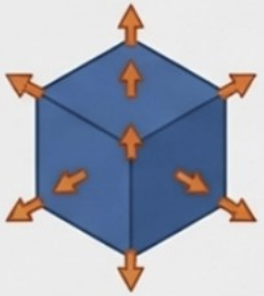
Calculator

Computes entirely new arrays via mathematical expressions.



Glyph

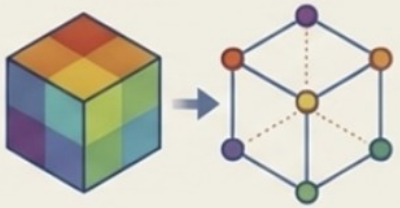
Places oriented and scaled shapes at point locations.



Common Filters – Part II

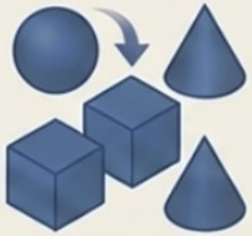
Cell Data to Point Data

Interpolates cell-centered data to point locations.



Group Datasets

Combines multiple separate datasets into a single multi-block structure.



Extract Block

Selects specific blocks from a multi-block dataset for isolation.

Tubes

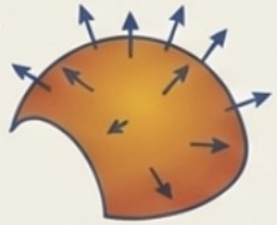
Generates cylindrical geometry around lines, useful for streamlines.

Stream Tracer

Integrates a vector field to create streamlines, pathlines, or streaklines.

Surface Normals

Computes normal vectors for points or cells on a surface.



Useful Settings / Tools / Tricks

- Properties Menu
  - Can be split into sub-panels to make navigation easier
  - I prefer the sub-panels as it slightly better organises ParaView's many, many options
- Keyboard launcher for Filter Search
  - Ctrl + space
  - I will use this almost exclusively instead of searching the dropdown menu
- Right-click options on a mesh
  - Set opacity
  - Switch off multiblock dataset
- Automatic Apply – Use with caution!! Only a good idea with very small datasets

Tutorial Datasets

- Three different datasets to work with:
  - Polydisperse spheres
  - Polydisperse multi-spheres with multiple shapes
  - Polydisperse polyhedrals with multiple shapes

- Available in both **XML** and **VTKHDF** formats

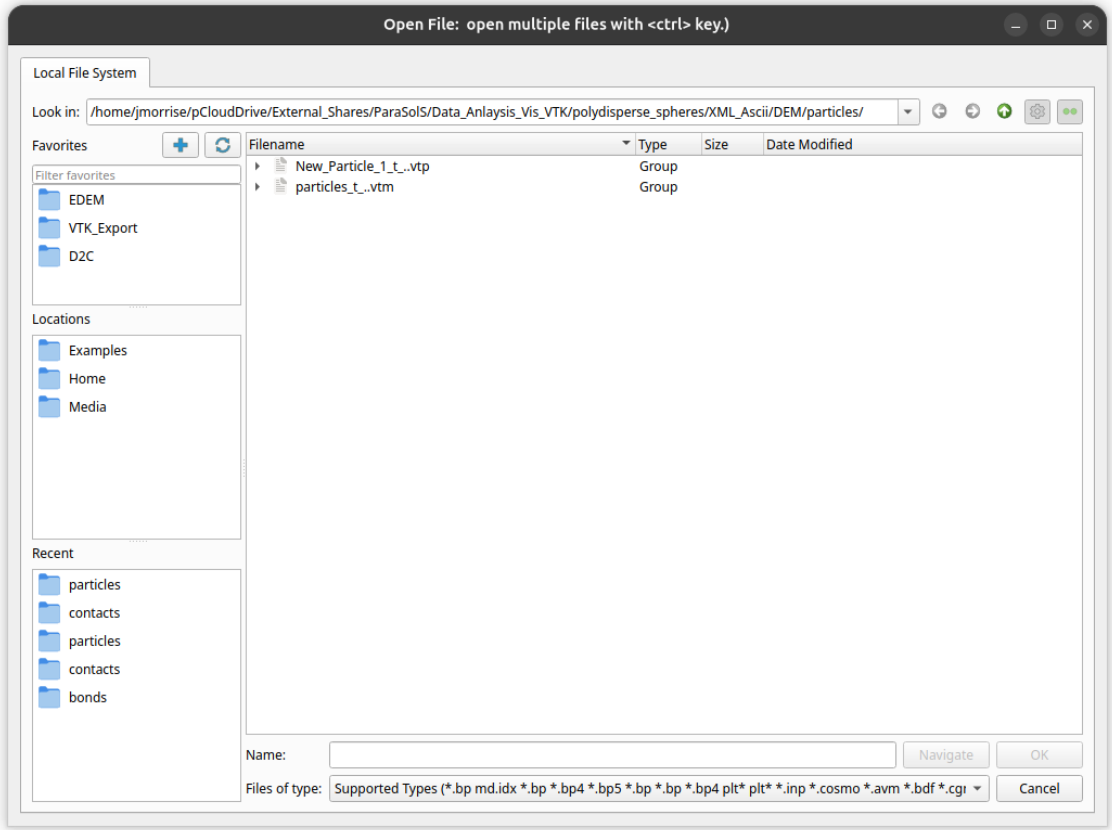
- Download from here:
  - Data_Anlaysis_Vis_VTK - pCloud

Tutorial Task-list

- Load a dataset
- Create some new variable(s)
  - Calculator filter
- Add some annotations
- Apply a glyph
- Select a variable and apply colourmap
  - Opacity Map
- Apply some filters
  - Tubes
  - Glyphs
- Save a state / pipeline
- Apply some additional filters
  - Clip
  - Slice
  - Threshold
- Create multiple viewports
- Plot some data
- Export:
  - Data
  - Images
  - Videos

Loading Dataset Source

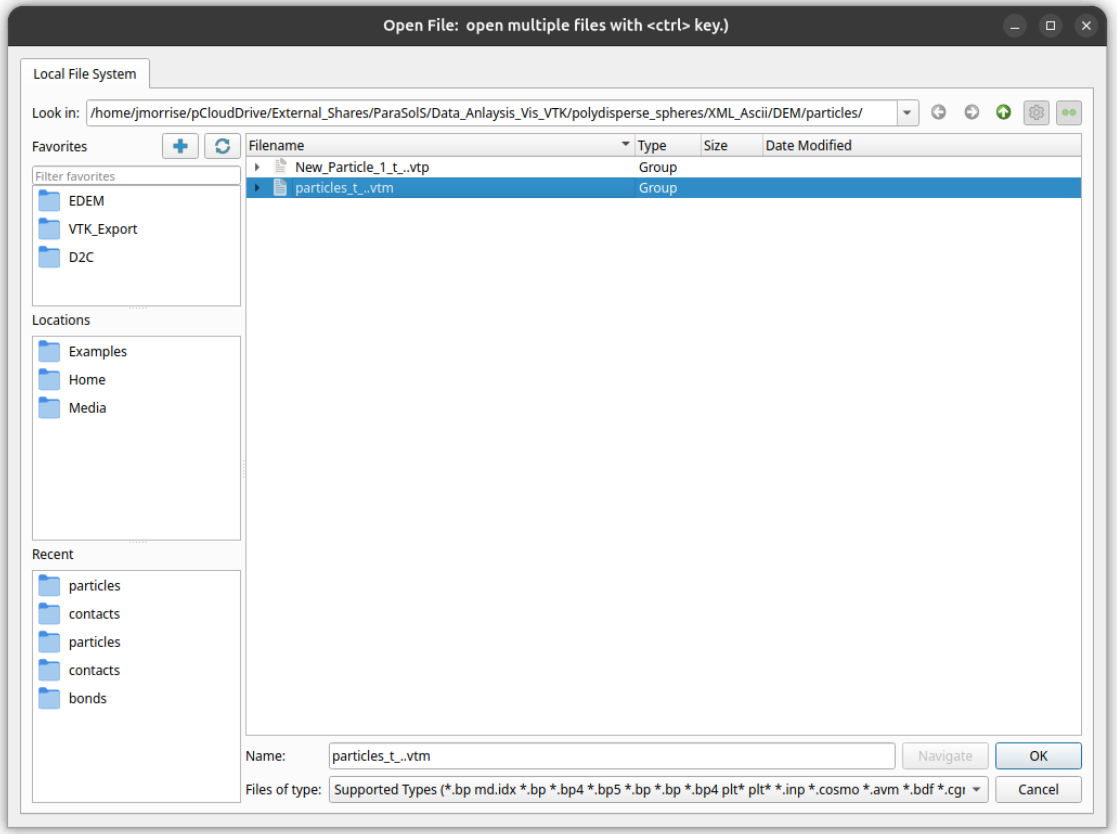
- Select "Open ..." from the file menu or using the shortcut icon 
- Navigate to the example data set (`polydisperse_spheres/XML_Ascii/DEM`) and load the particle, contact and geometry data files



- You will see a window similar to the right
- Select the *.vtm file and select "OK"

Loading Dataset Source

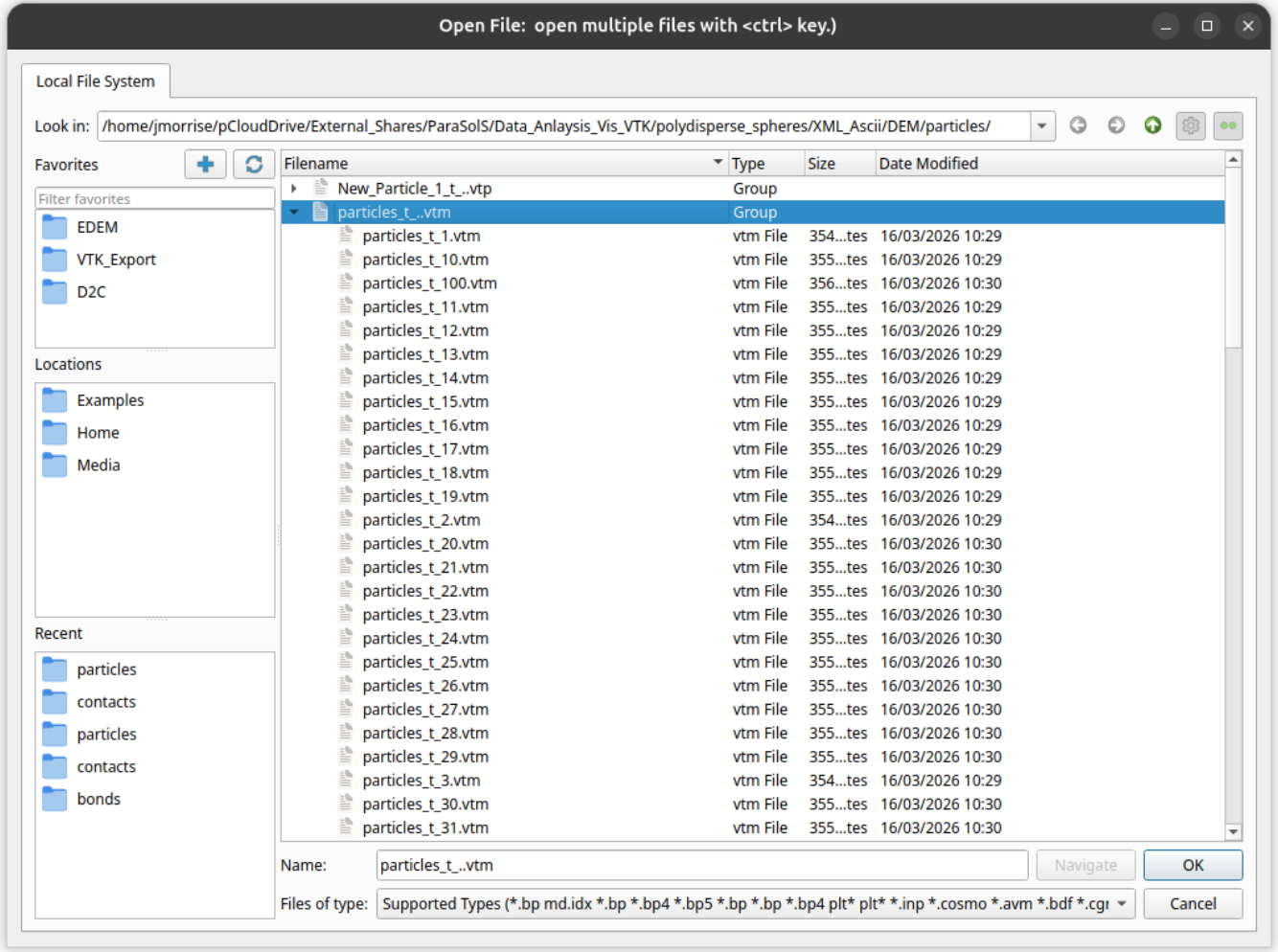
- Select "Open ..." from the file menu or using the shortcut icon 
- Navigate to the example data set (`polydisperse_spheres/XML_Ascii/DEM`) and load the particle, contact and geometry data files



- You will see a window similar to the right
- Select the *.vtm file and select "OK"

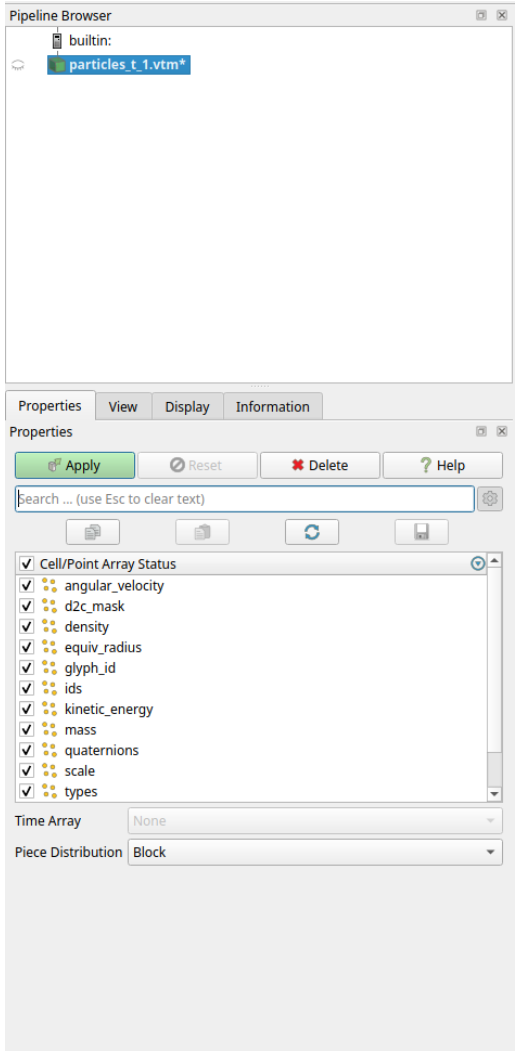
Where are all my files gone?!

- ParaView is somewhat smart and can handle sequentially numbered files as a time series
- Finds all files with the pattern and lists as one time series
  - *fooN.vtk*
  - *foo N.vtk*
  - *fooN.vtk*
  - *foo.N.vtk*
  - *Nfoo.vtk*
  - *N.foo.vtk*
  - *foo.vtk.N*
  - *foo.vtks*
- Replicates the .pvd file behaviour without needing an explicit .pvd file



Don't Forget To APPLY

- The pipeline browser will now look something similar to the screenshot
  - You may only have two panels of "properties" and "Information", depending on your settings
- Data will not be shown in ParaView until you click the green "**Apply**" button

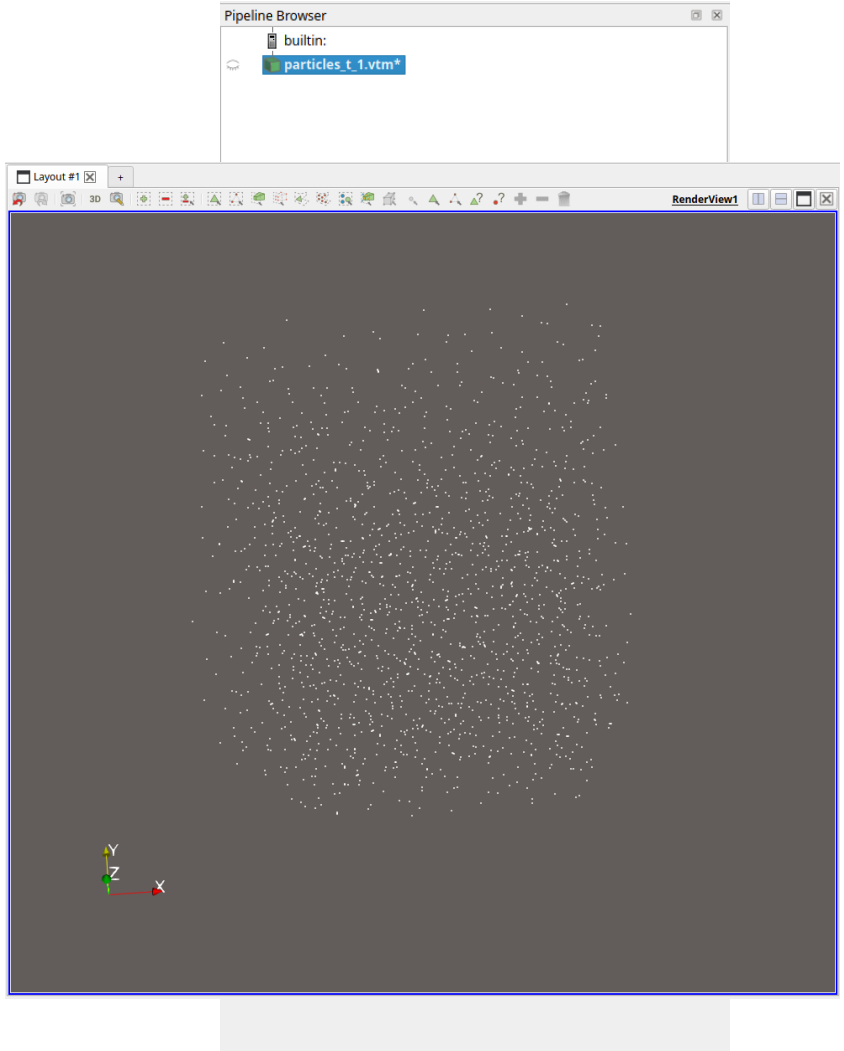


- Note that the "eye" icon changes from closed to open to indicate dataset is loaded in current layout

- You will now see data (*unimpressively*) loaded in the render layout. We will improve later.
- Repeat for contact and geometry data

Don't Forget To APPLY

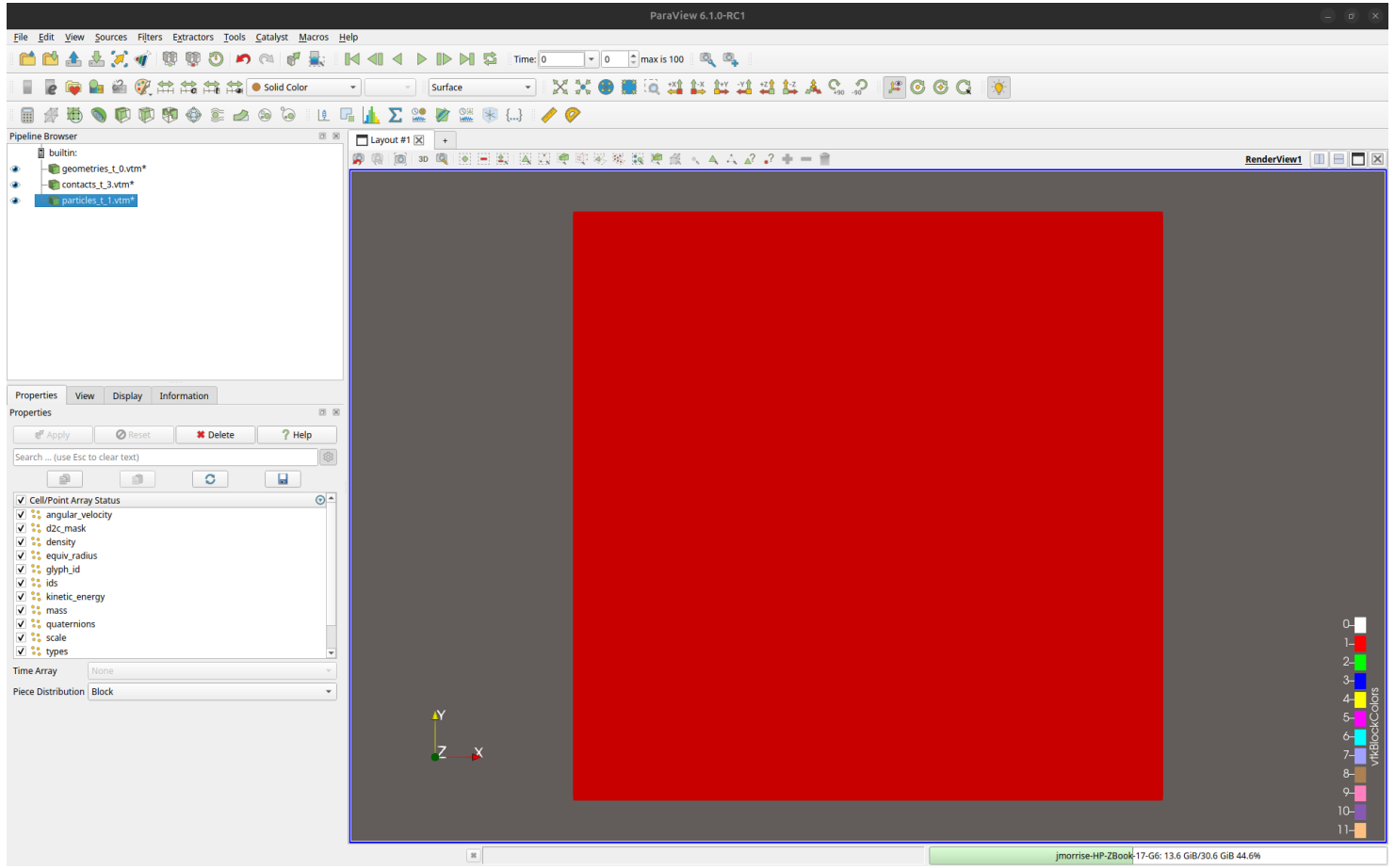
- The pipeline browser will now look something similar to the screenshot
  - You may only have two panels of "properties" and "Information", depending on your settings
- Data will not be shown in ParaView until you click the green "**Apply**" button



- Note that the "eye" icon changes from closed to open to indicate dataset is loaded in current layout

- You will now see data (*unimpressively*) loaded in the render layout. We will improve later.
- Repeat for contact and geometry data

Multiple Loaded Datasets

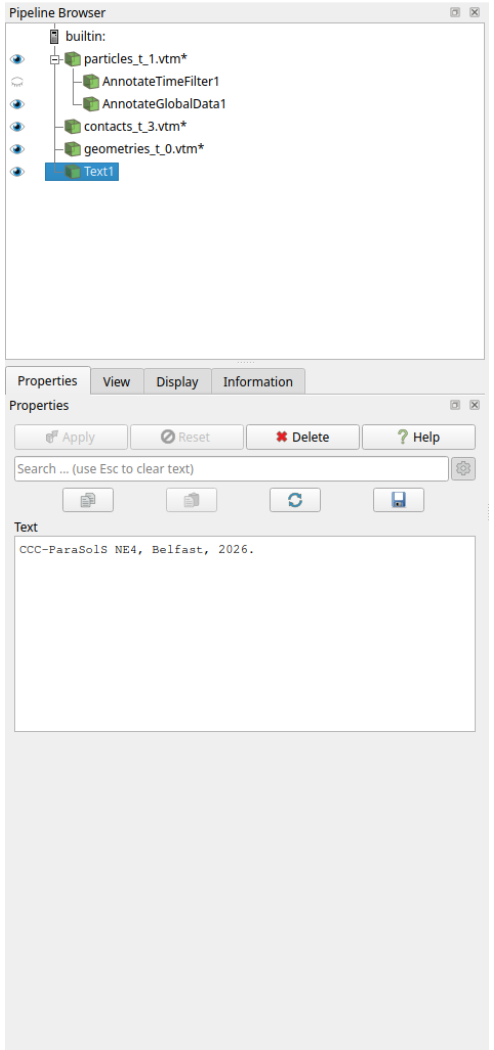


- After successfully loading all files, your screen will look something like this
- You can now pan, zoom and rotate the view
  - Use mouse or viewport controls shortcuts
- Note that the time controls have loaded timestep 0 and show max is 50

Annotations – Title

- It is often useful to include as descriptive text annotation to describe a visualisation
  - This can be especially important when using multiple views

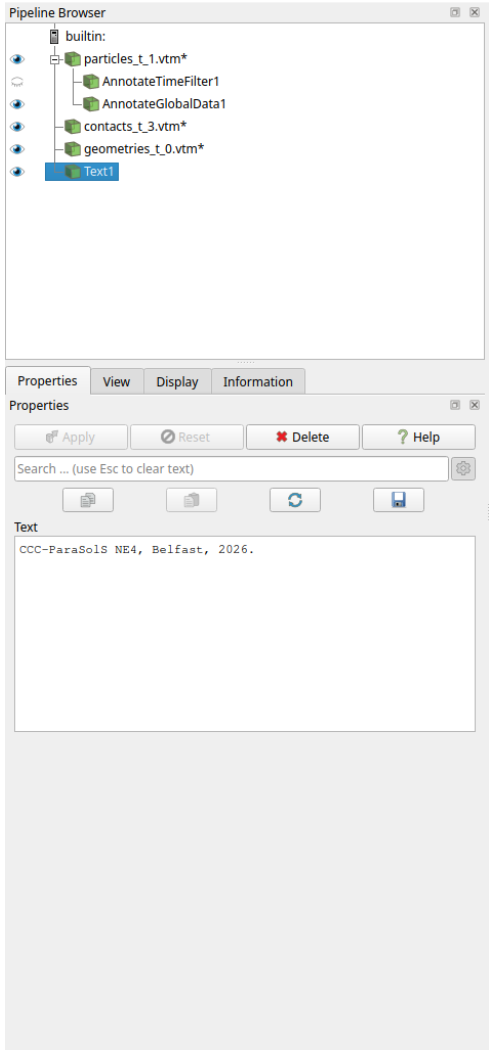
- **Exercise:** Add text annotation using Text filter

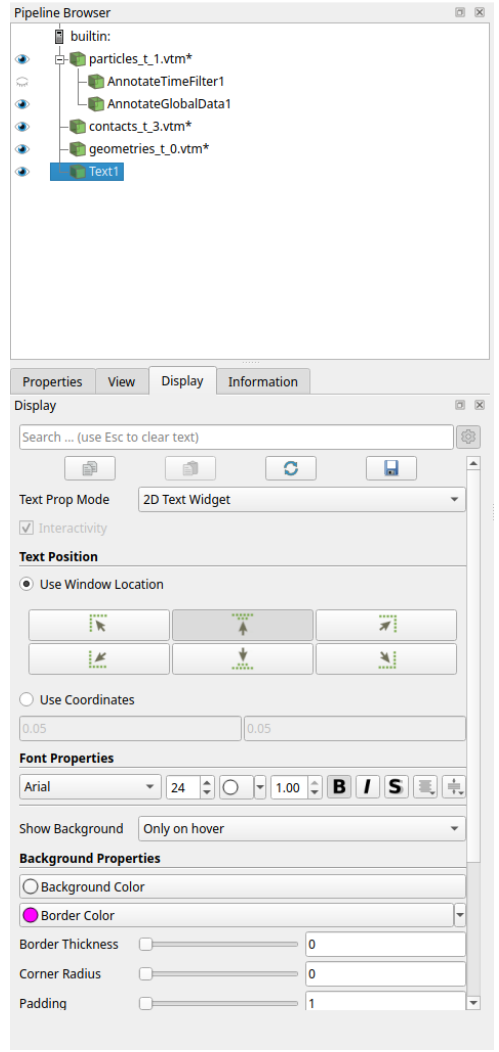


Annotations – Title

- It is often useful to include as descriptive text annotation to describe a visualisation
  - This can be especially important when using multiple views

- **Exercise:** Add text annotation using Text filter
  - Adjust display properties and position in the display panel

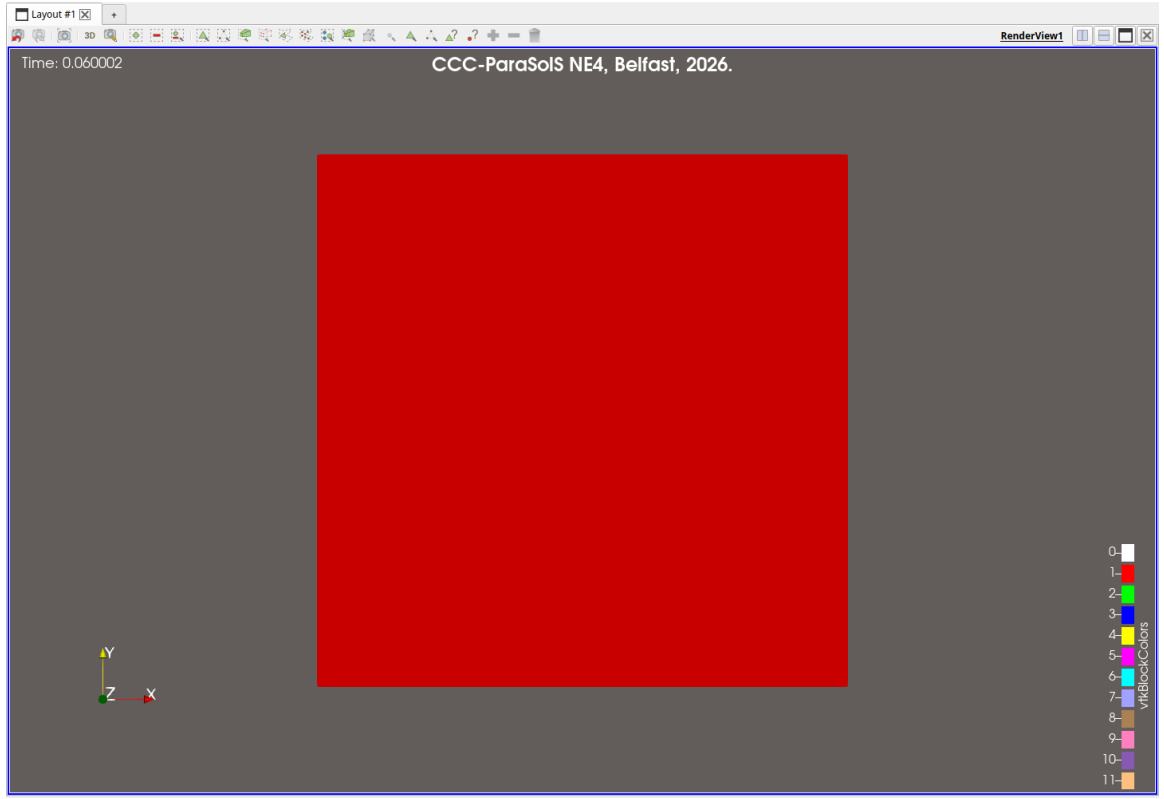




Annotations – Title

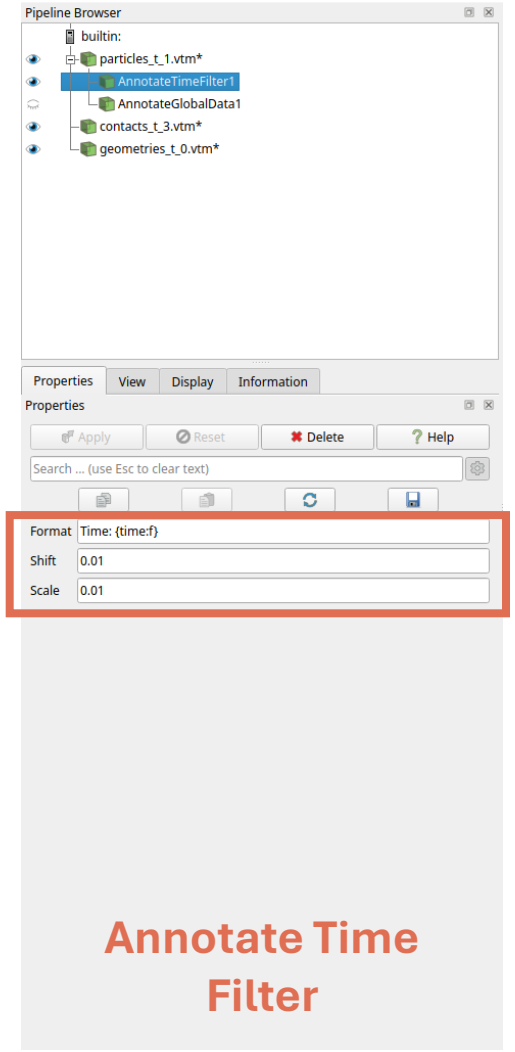
- It is often useful to include as descriptive text annotation to describe a visualisation
  - This can be especially important when using multiple views

- **Exercise:** Add text annotation using Text filter
  - Adjust display properties and position in the display panel
  - I have increased the fontsize and centred

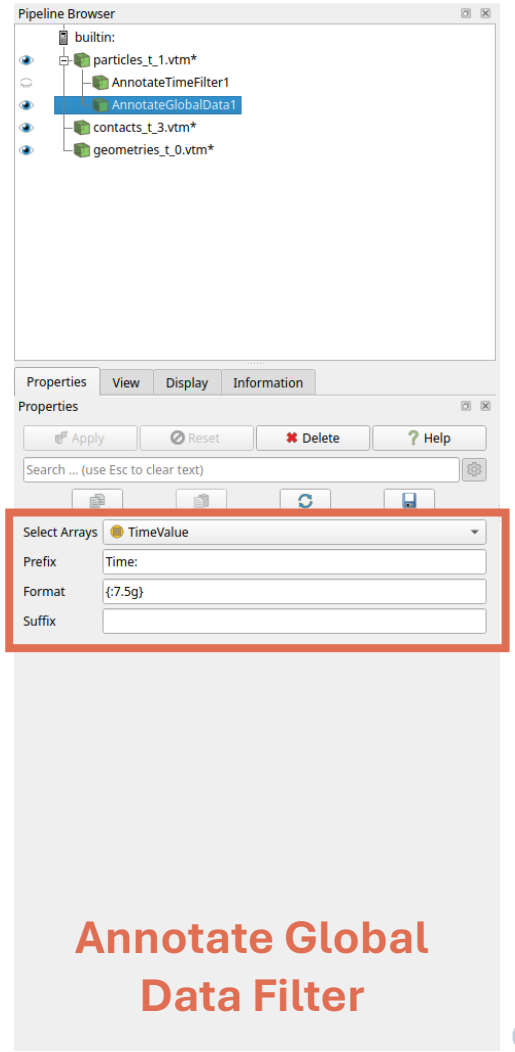


Annotations – Timestep

- It is often useful to include the current timestep as an annotation
- There are several possible filters available to add time:
  - **Annotate Time Filter**
    - Use when time is presentment in FieldData or a .pvd file
  - **Annotate Global Data**
    - Use with MultiBlocks (with or without FieldData)
- **Exercise:** Add time annotation using appropriate filter
  - Particle data contains timestep in the FieldData



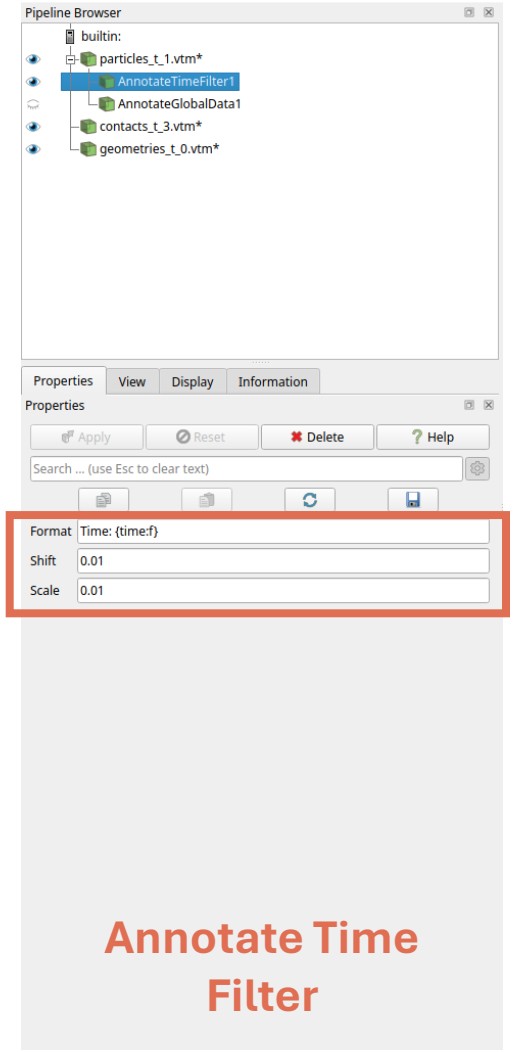
Annotate Time Filter



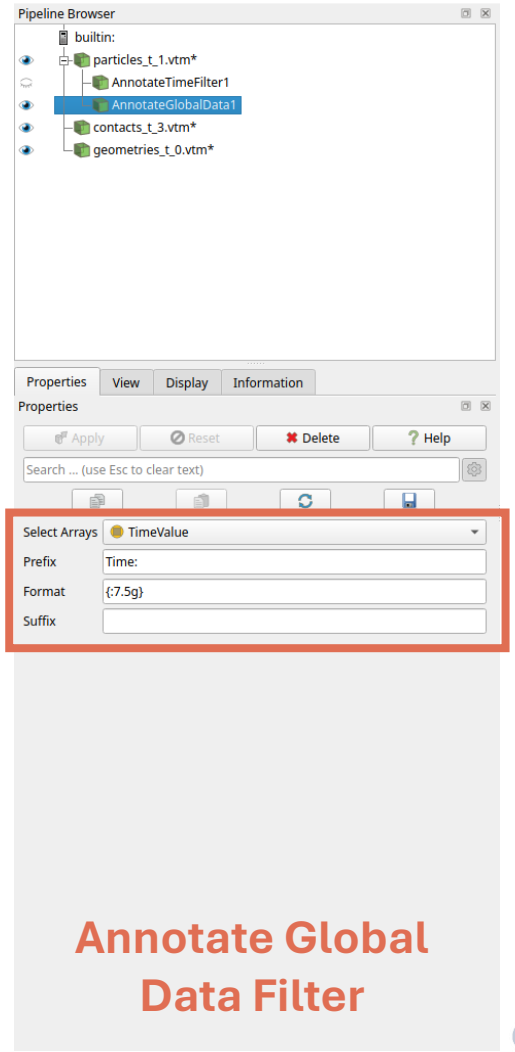
Annotate Global Data Filter

Annotations – Timestep

- It is often useful to include the current timestep as an annotation
- There are several possible filters available to add time:
  - **Annotate Time Filter**
    - Use when time is presentment in FieldData or a .pvd file
  - **Annotate Global Data**
    - Use with MultiBlocks (with or without FieldData)
- **Exercise:** Add time annotation using appropriate filter
  - Particle data contains timestep in the FieldData
  - Hint: we need to use Annotate Global Data to read *exact timestep* from file



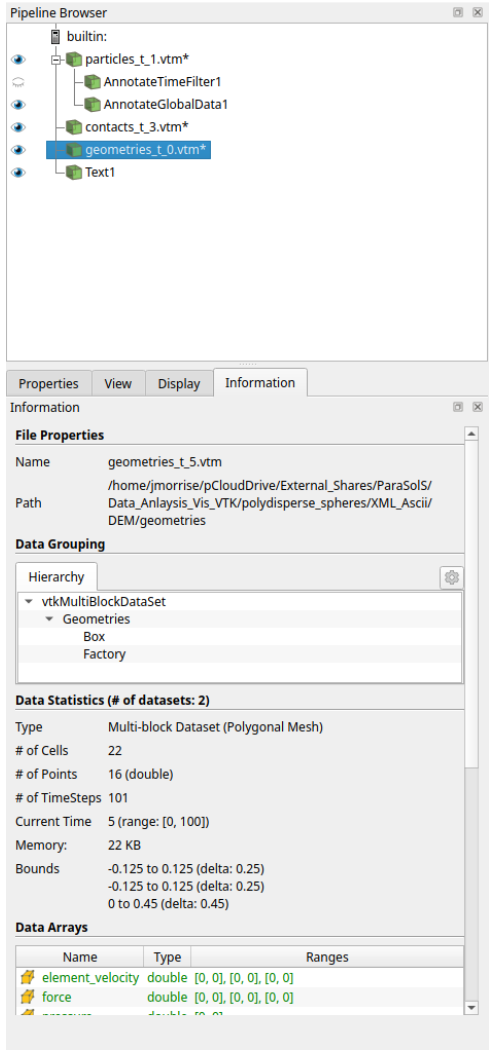
Annotate Time Filter



Annotate Global Data Filter

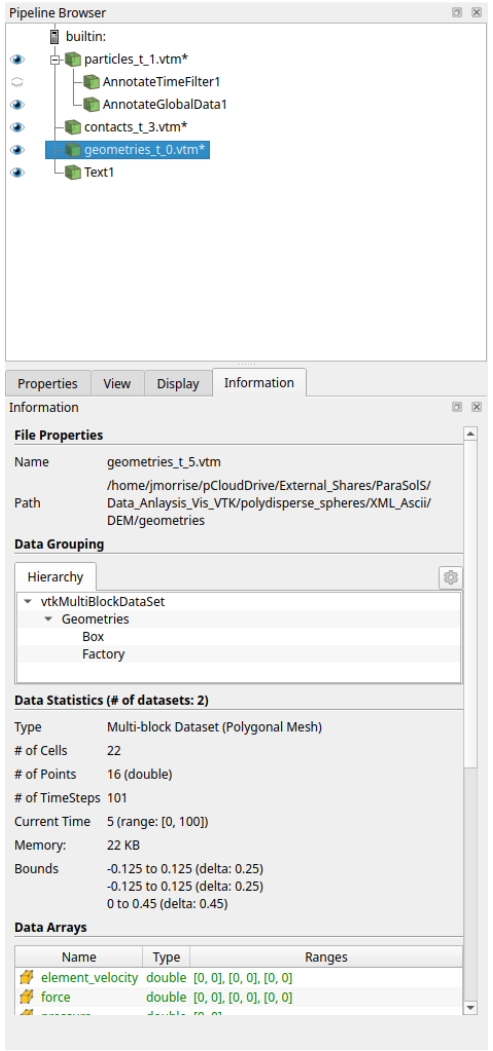
Enabling/Disabling MultiBlocks

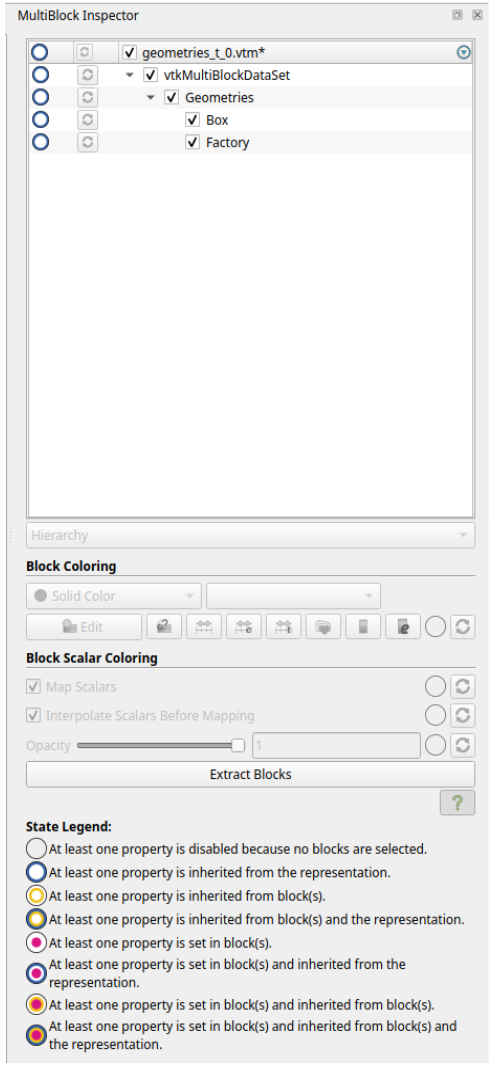
- All data sources here are loaded as MultiBlock to allow for different particle types, contact types and geometries to exist in a single file
  - See information panel to see all information about a dataset
- For visualisation, we often want to disable certain items to see things more clearly
- **Exercise:** Open the **MultiBlock Inspector** (right click in toolbar area to open) and disable Factory geometry by unticking selector



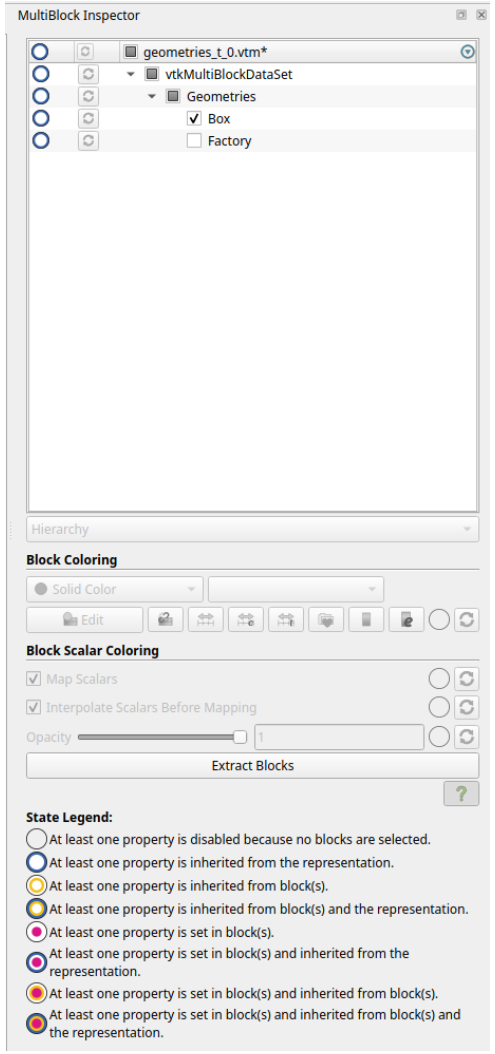
Enabling/Disabling MultiBlocks

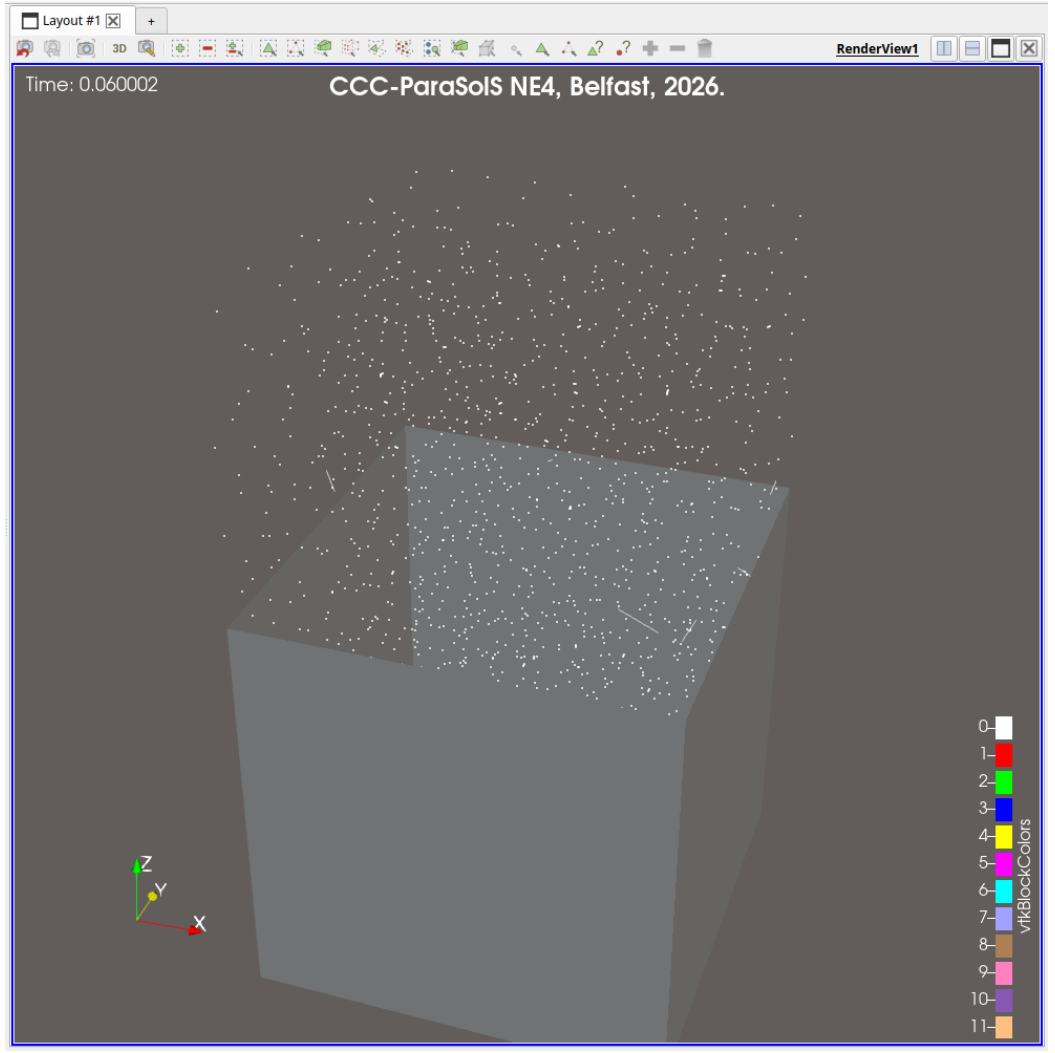
- All data sources here are loaded as MultiBlock to allow for different particle types, contact types and geometries to exist in a single file
  - See information panel to see all information about a dataset
- For visualisation, we often want to disable certain items to see things more clearly
- **Exercise:** Open the **MultiBlock Inspector** (right click in toolbar area to open) and disable Factory geometry by unticking selector





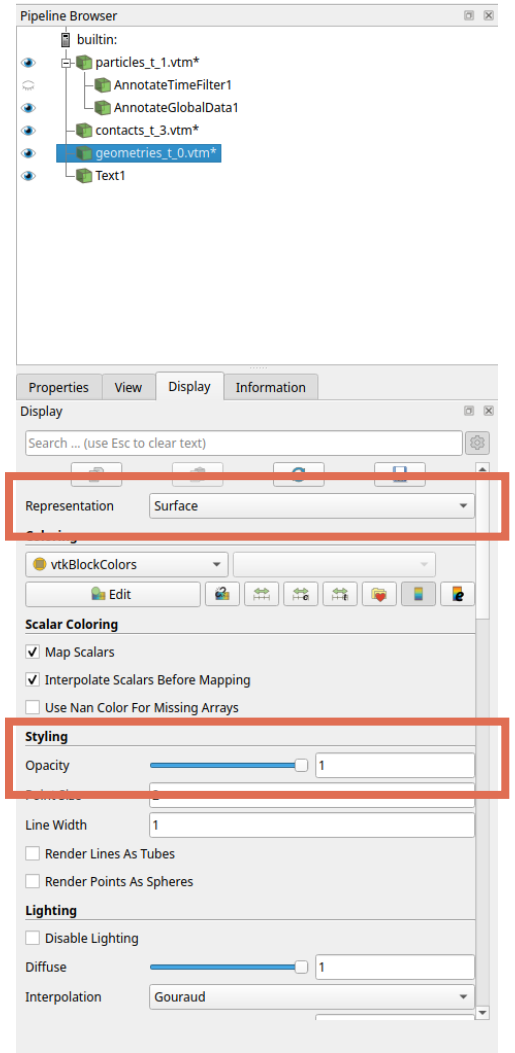
Enabling/Disabling MultiBlocks





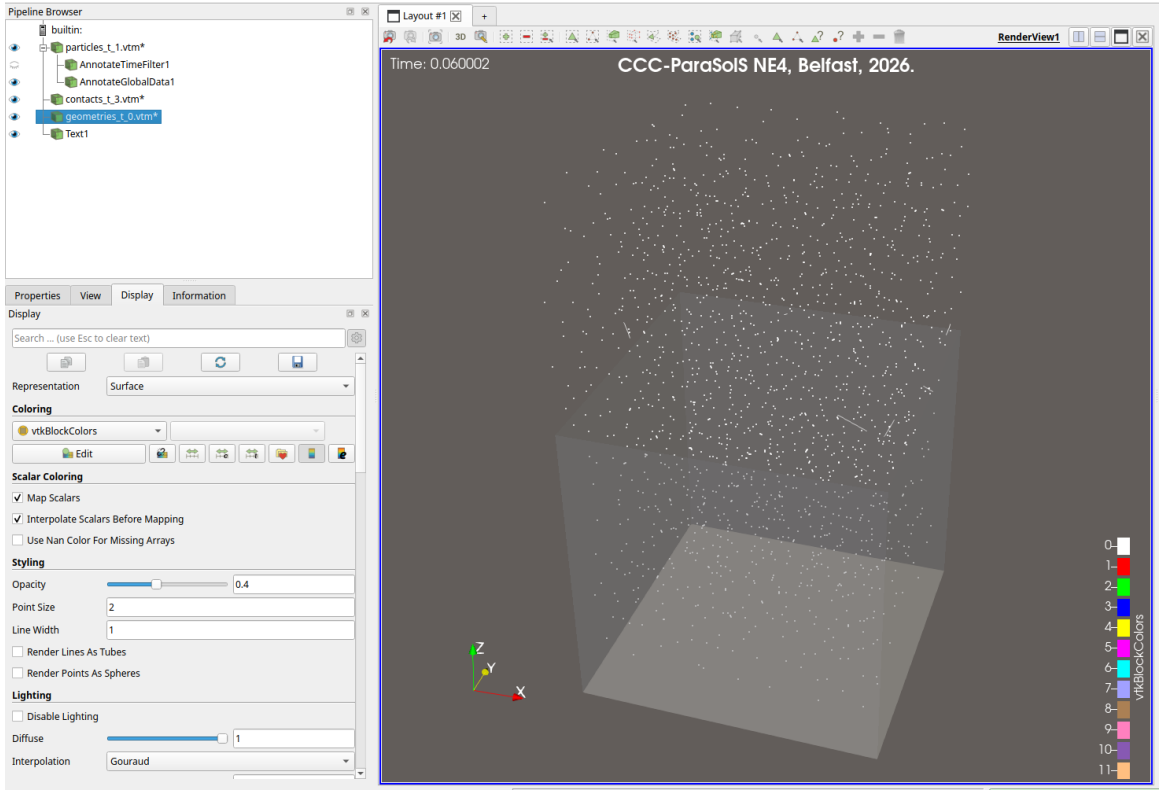
Further Display Options

- There are many different surface representations for datasets as previously shown
  - The default is "Surface"
- **Exercise:** Change the geometry representation to make your particles (currently dots!) more visible
  - Change the opacity for the surface to 0.2
  - Change the representation to "Wireframe"

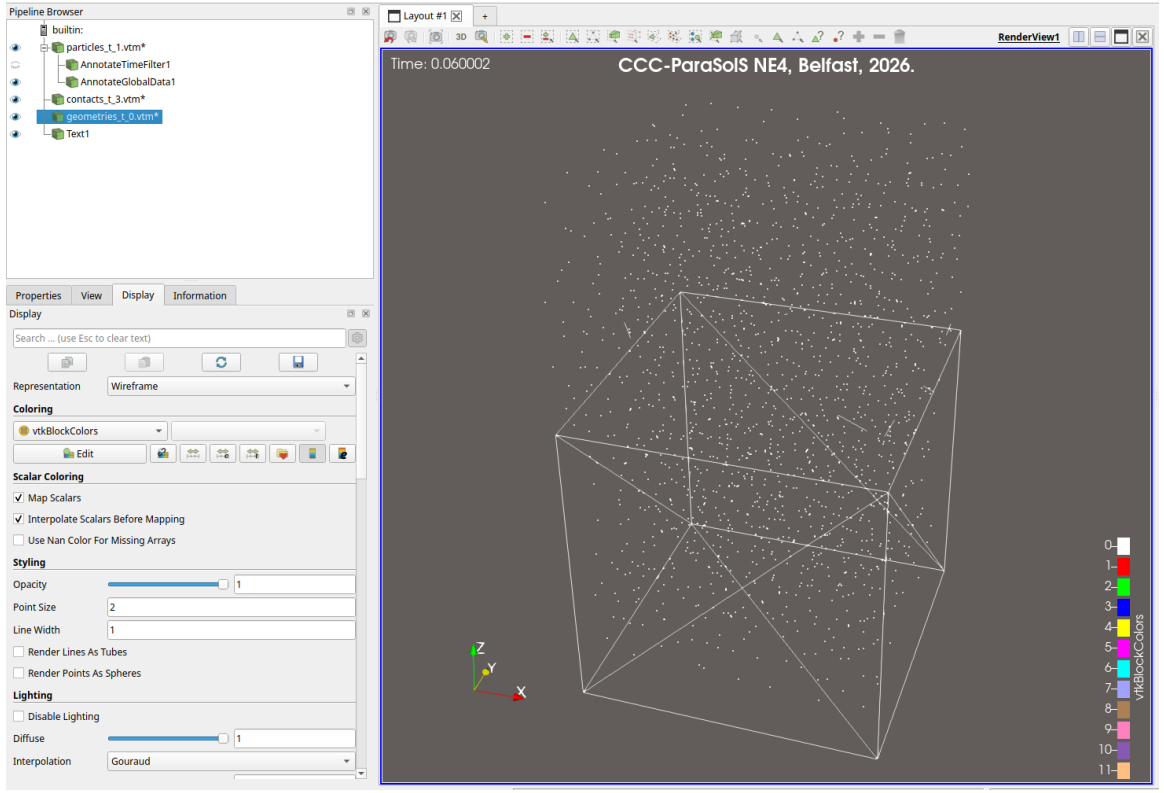


Further Display Options

Opacity



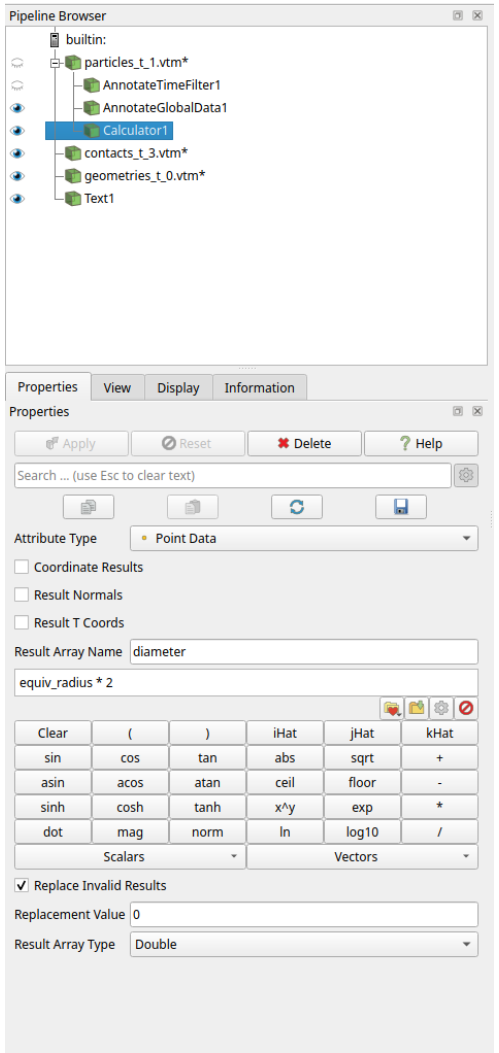
Wireframe Representation



Calculator Filter

- ParaView has a **Calculator filter** that allows you to operate on existing data to create new data variables

- **Exercise:** Create a new variable called *diameter* which is calculated from the radius (equiv_radius) for the particle dataset

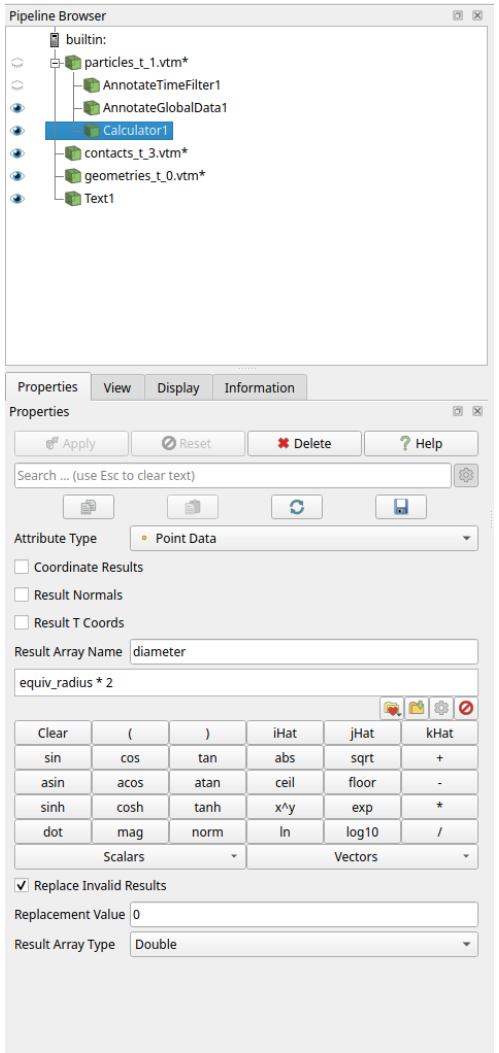


Calculator Filter

- ParaView has a **Calculator filter** that allows you to operate on existing data to create new data variables

- **Exercise:** Create a new variable called *diameter* which is calculated from the radius (equiv_radius) for the particle dataset

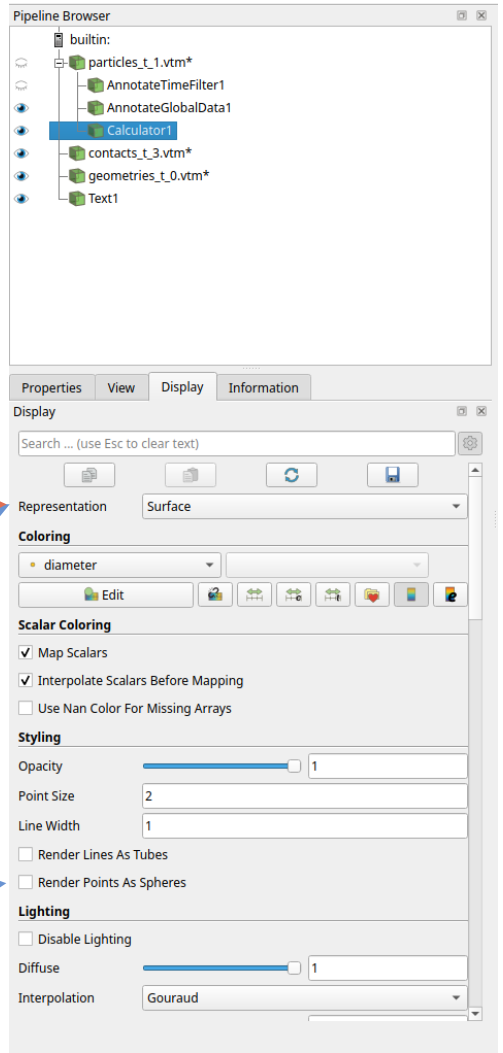
- Note that in the pipeline our active source is no longer *particles_t_1.vtm*, but *Calculator1*, which is where the new variable is added



Adding Particle Shape

- Our particles are currently horrible, useless dots. Let's fix that!

- There are often as many as 6 ways to add a shape to a particle
  - Point Gaussian representation
  - Glyph with custom Source filter
  - Glyph filter
  - 3D Glyph → Sphere
  - 3D Glyph → From pipeline
  - Styling → Select Render Points as Spheres



Important Note on Glyphing

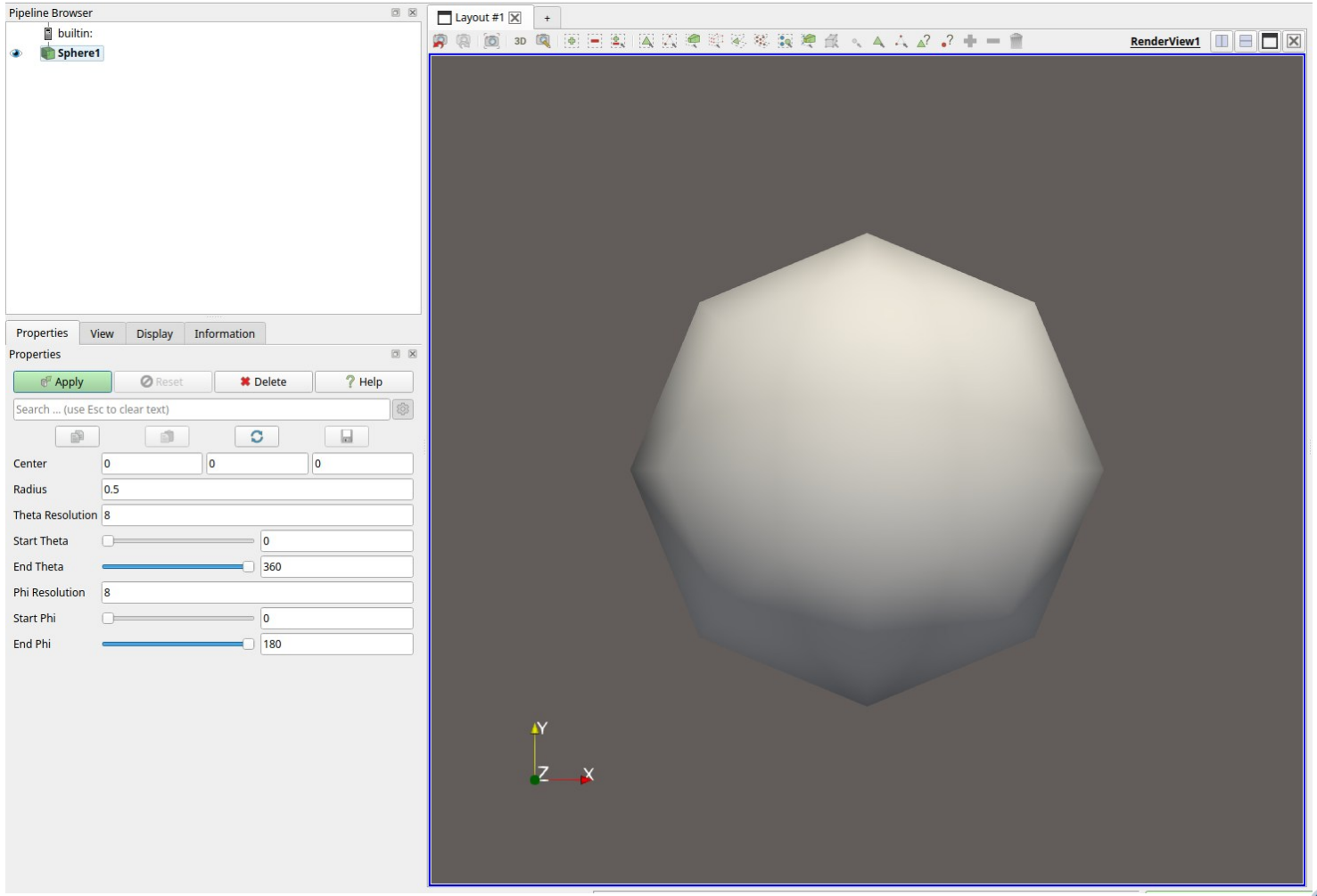
- **Glyph filters** create a complete copy of the glyph at each point in the datasets which can be memory intensive

- **3D Glyphs representation** (from the **Display** section of the Properties Panel instead of the filter) offer a significantly more memory-efficient and faster method to render
  - The representation uses **geometry instancing**, which stores only one copy of the source geometry and instructs the graphics hardware to draw multiple copies using various transformations

- Glyph filter is required if you need to export the glyph geometry to a file or if you must create actual 3D geometry for further processing in the pipeline

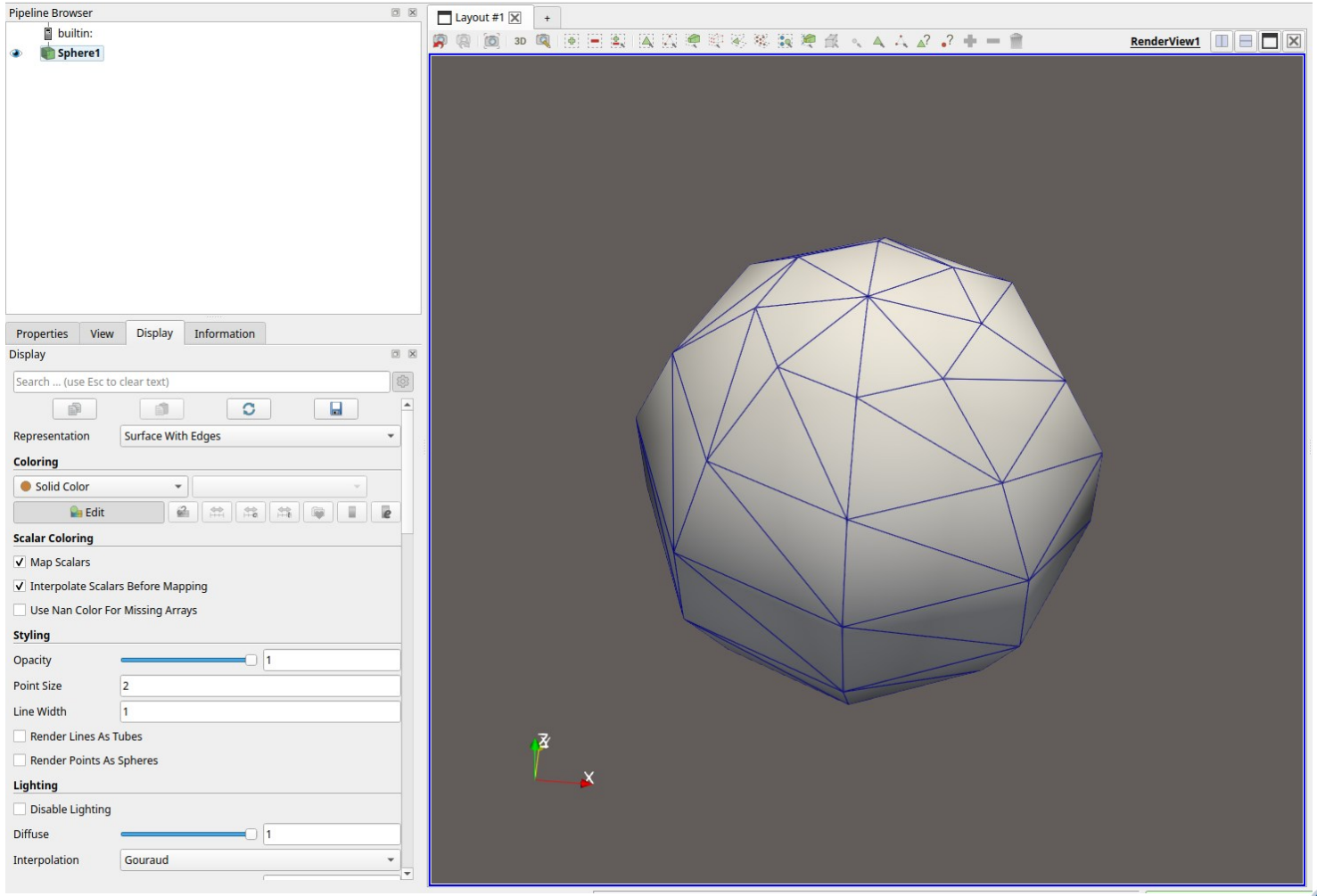
Aside: what is a Sphere?

- In ParaView, a sphere is actually a very well refined polyhedral particle



Aside: what is a Sphere?

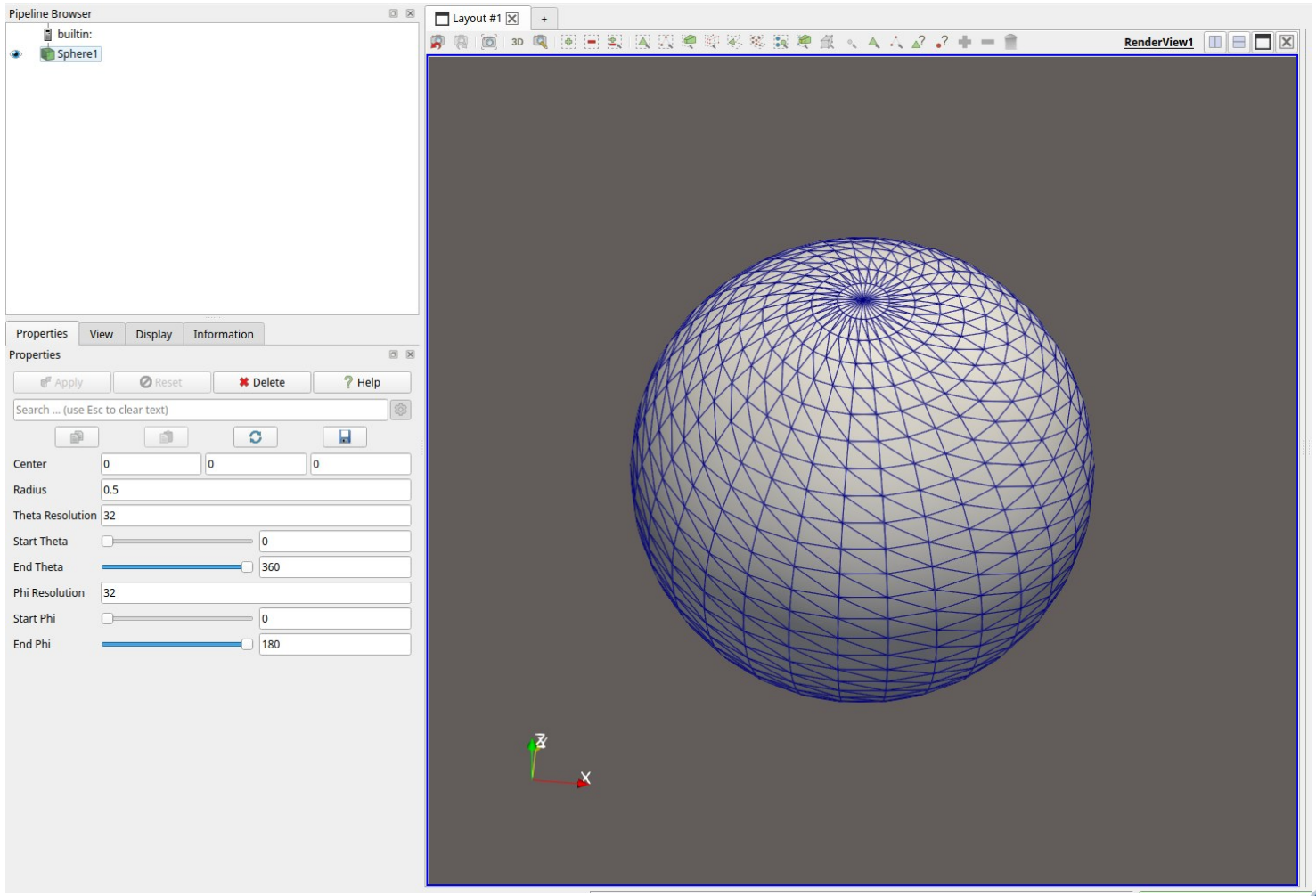
- In ParaView, a sphere is actually a very well refined polyhedral particle



Aside: what is a Sphere?

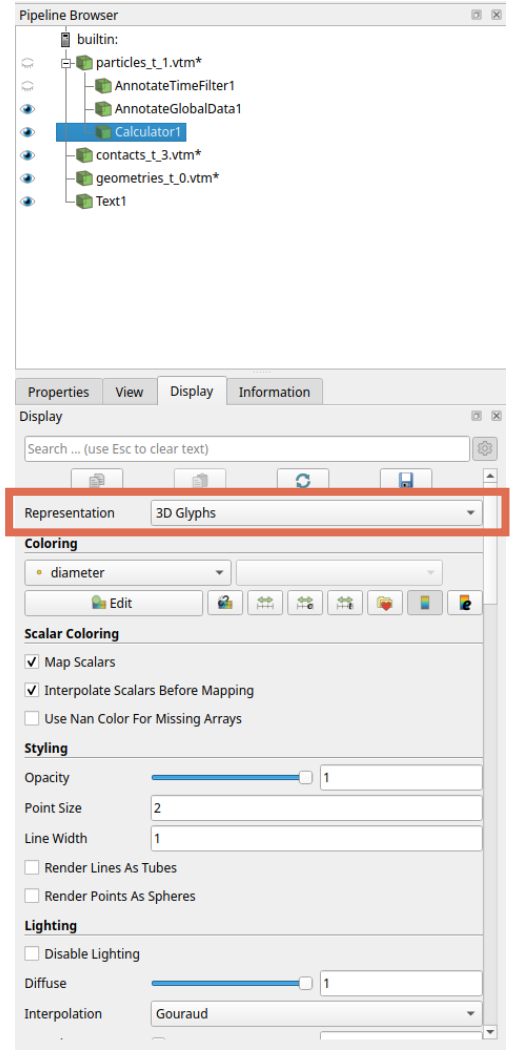
- In ParaView, a sphere is actually a very well refined polyhedral particle

- To improve the resolution, we simply modify the Theta and Phi resolution

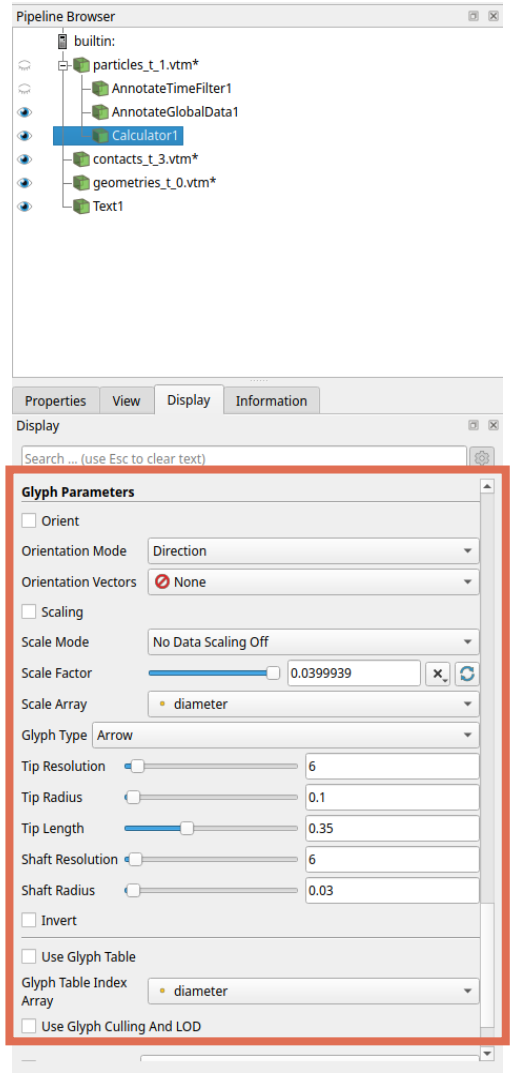


Adding Particle Shape

- **Exercise:** Add particle shape use the Sphere from 3D filters
  - Set representation to **3D Glyphs**
  - Scroll to **Glyph Parameters** section of display properties



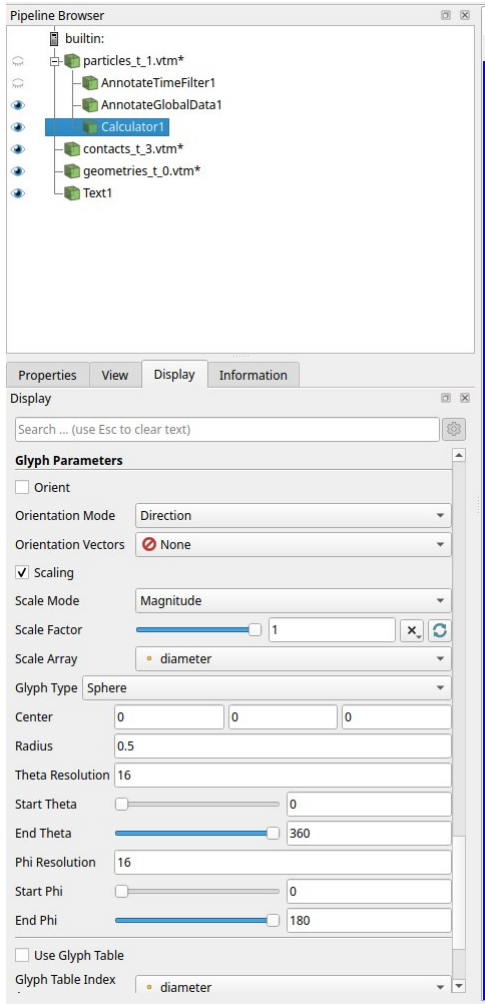
Change Representation

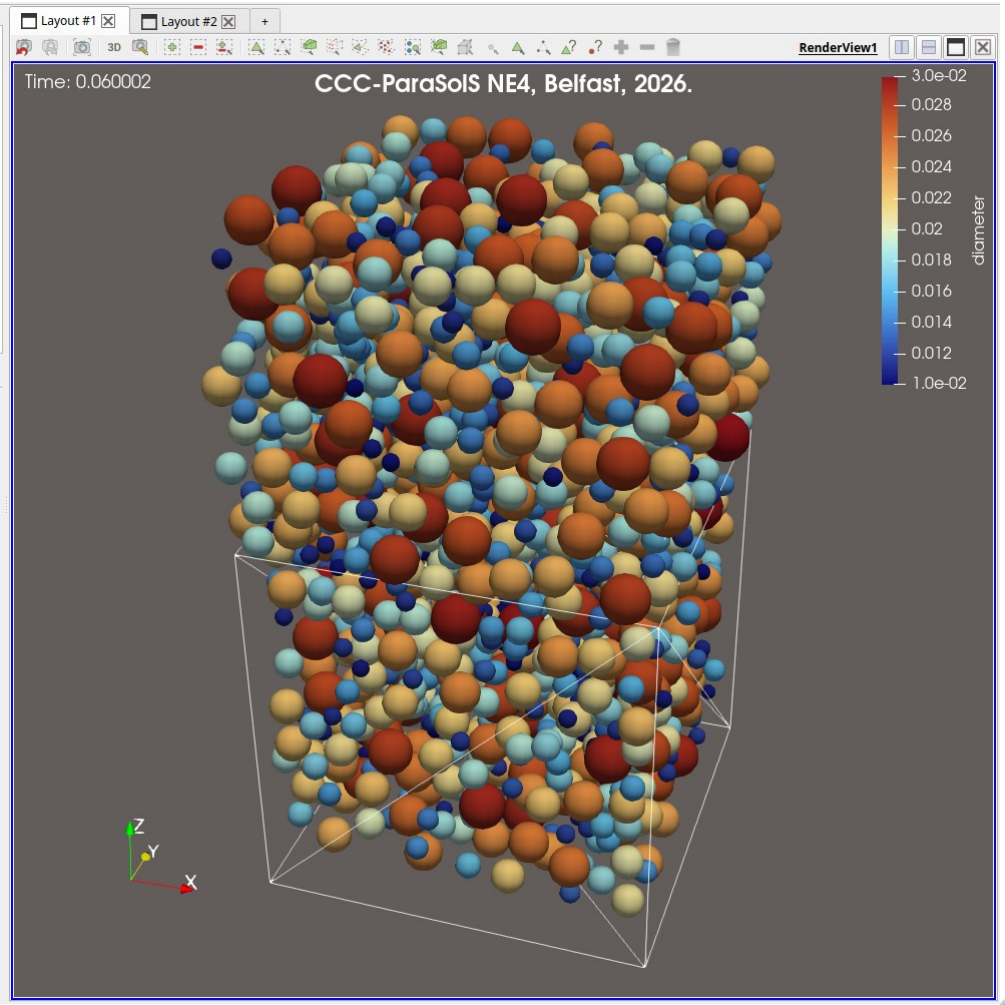


Glyph Settings

Adding Particle Shape

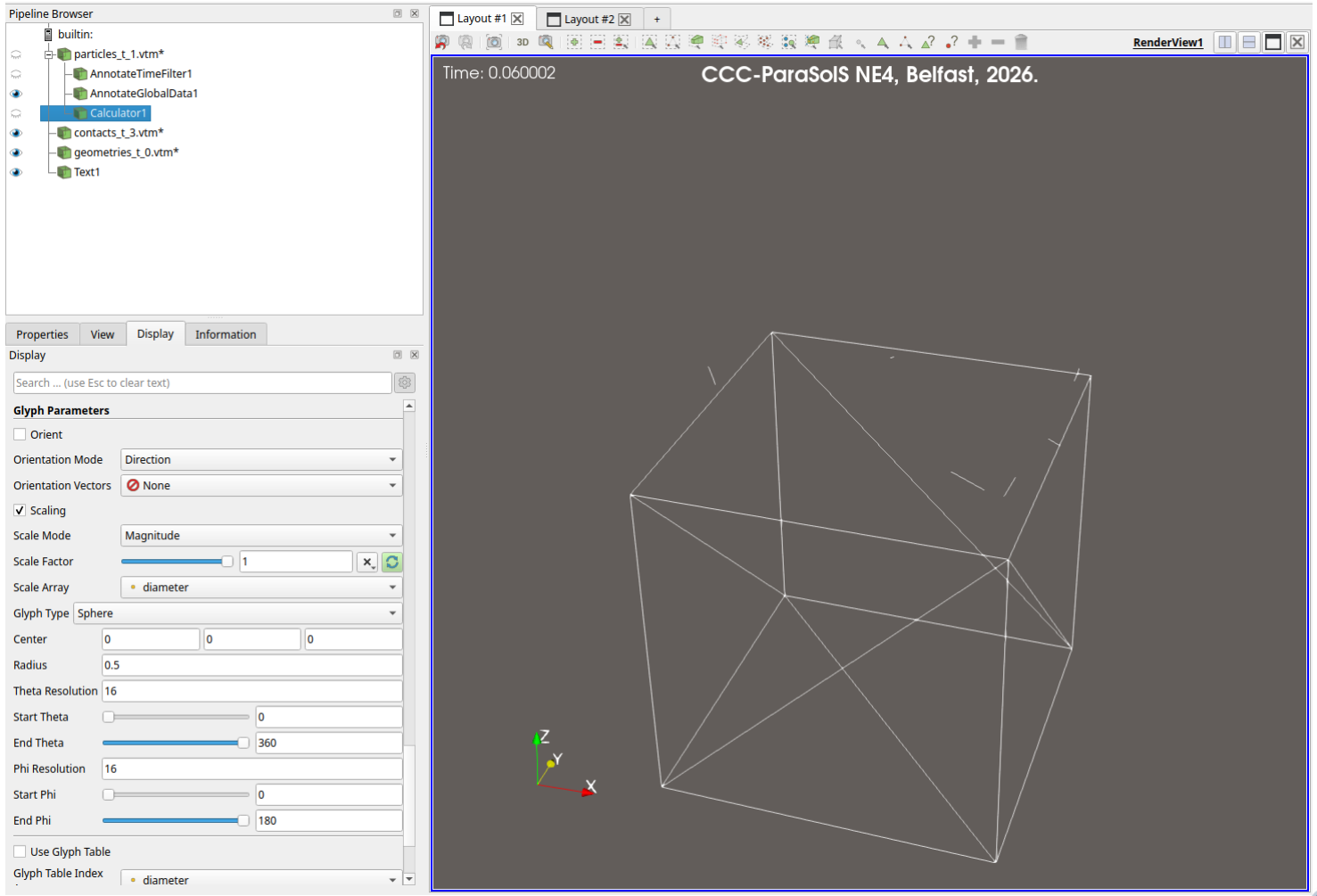
- **Exercise:** Add particle shape use the Sphere from 3D filters
  - Set representation to **3D Glyphs**
  - Scroll to **Glyph Parameters** section of display properties
  - Set Glyph Type to **Sphere**
  - Turn on **Scaling** and use particle diameter as scale array





Showing Contact Data

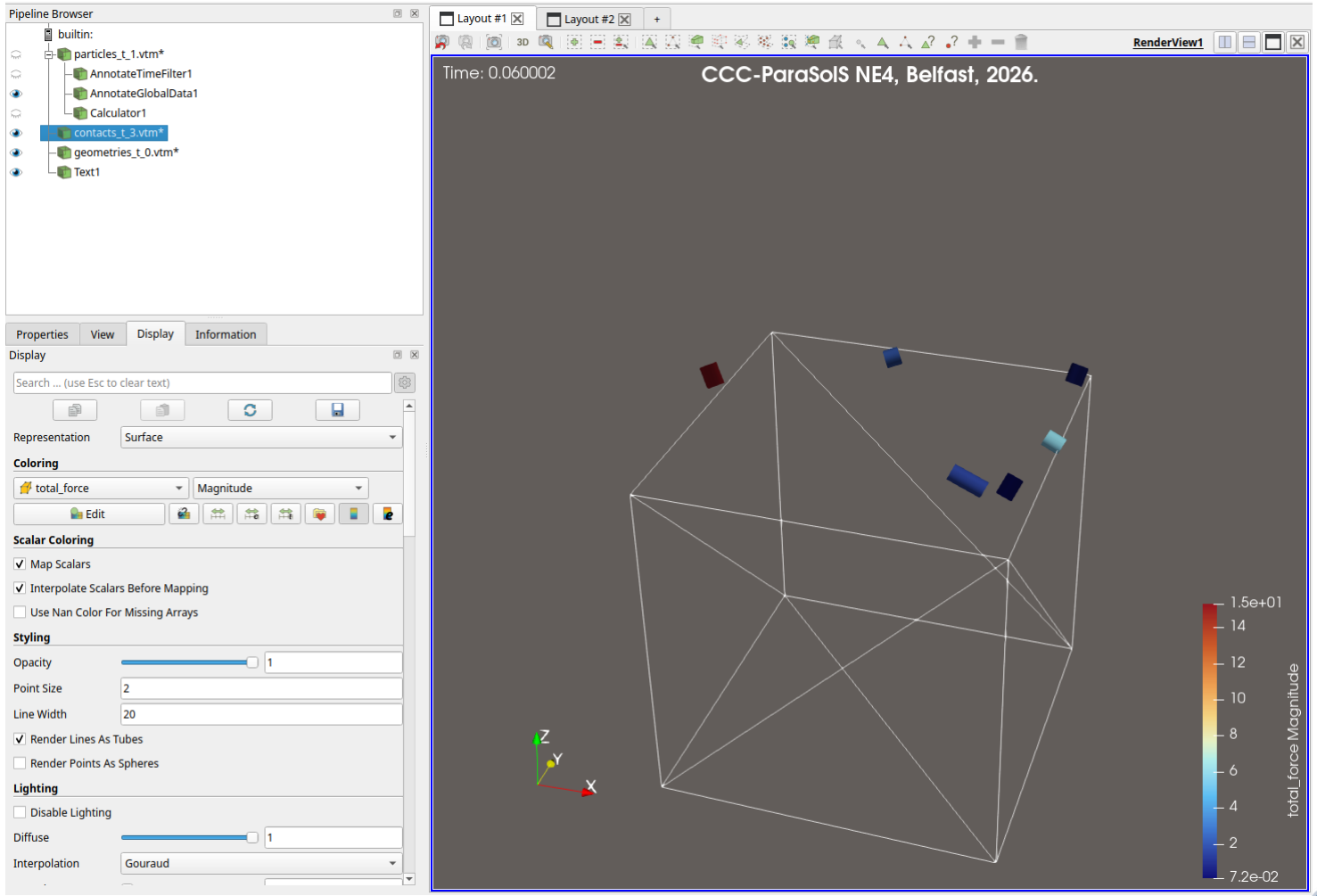
- Being able to visualise the contact network is an important technique to understand force transmission in DEM simulations
- Let's quickly visualise our contact network
- **Exercise:** Switch off particle display in the pipeline



Contacts – "Tubes"

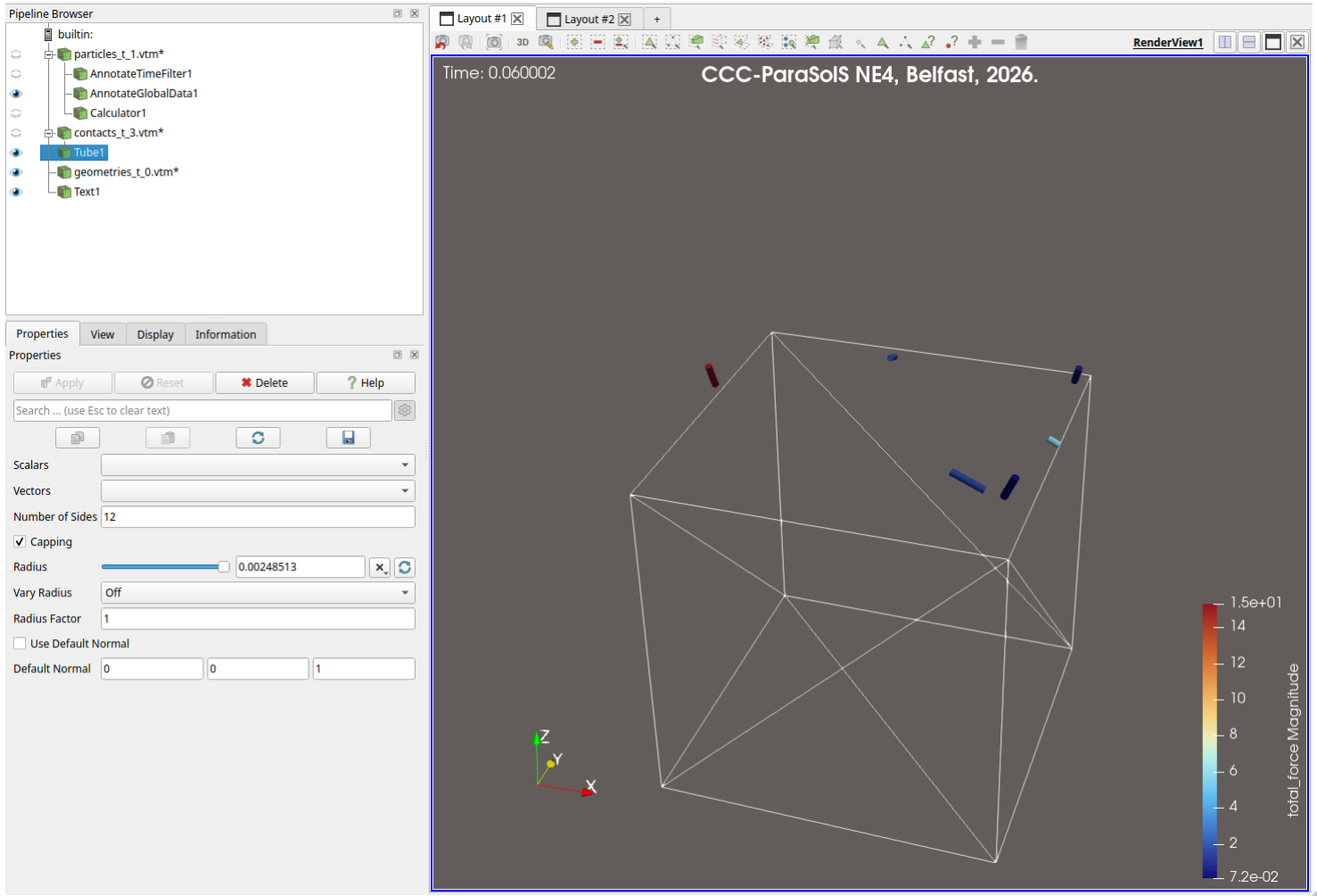
- Two possible methods to render contacts:
  - The Tube filter
  - Styling options

- The styling option is a quick way to visualise, but the Tube filter offers more control and options
- **Exercise:** Style contacts as Tubes and colour by total_force



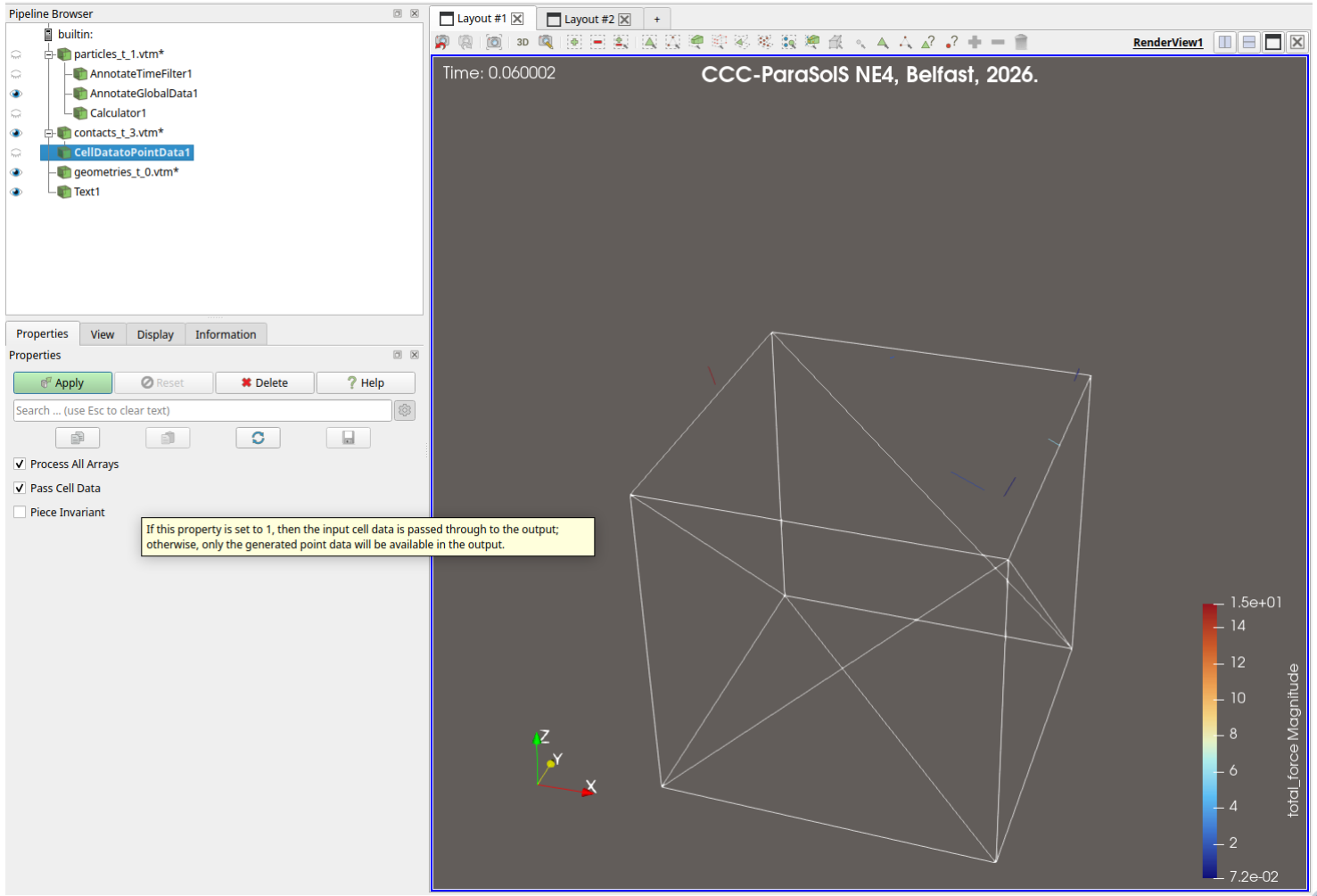
Contacts – The Tube Filter

- The Tube filter creates simple polyhedral cylinders
  - Memory intensive – use with caution
- The tube "*resolution*" is controlled by the number of sides
- It is possible to vary each tube diameter individually



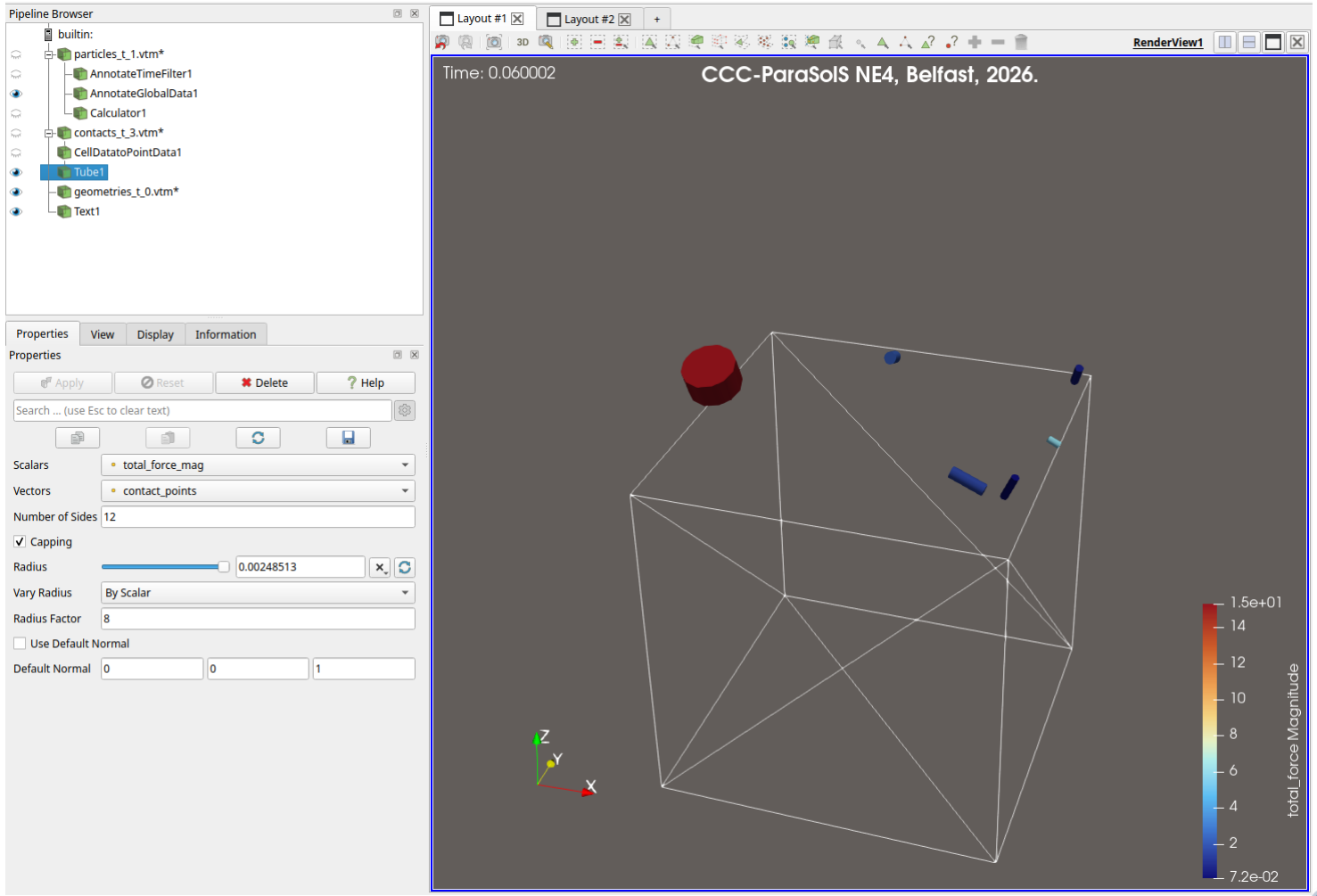
The Tube Filter – Individual Scaling

- To scale individually the contact data; which is actually stored as cell data; needs to be converted to point data at each end of the tube
- **Exercise:** create new source using **Cell Data to Point Data** filter



The Tube Filter – Individual Scaling

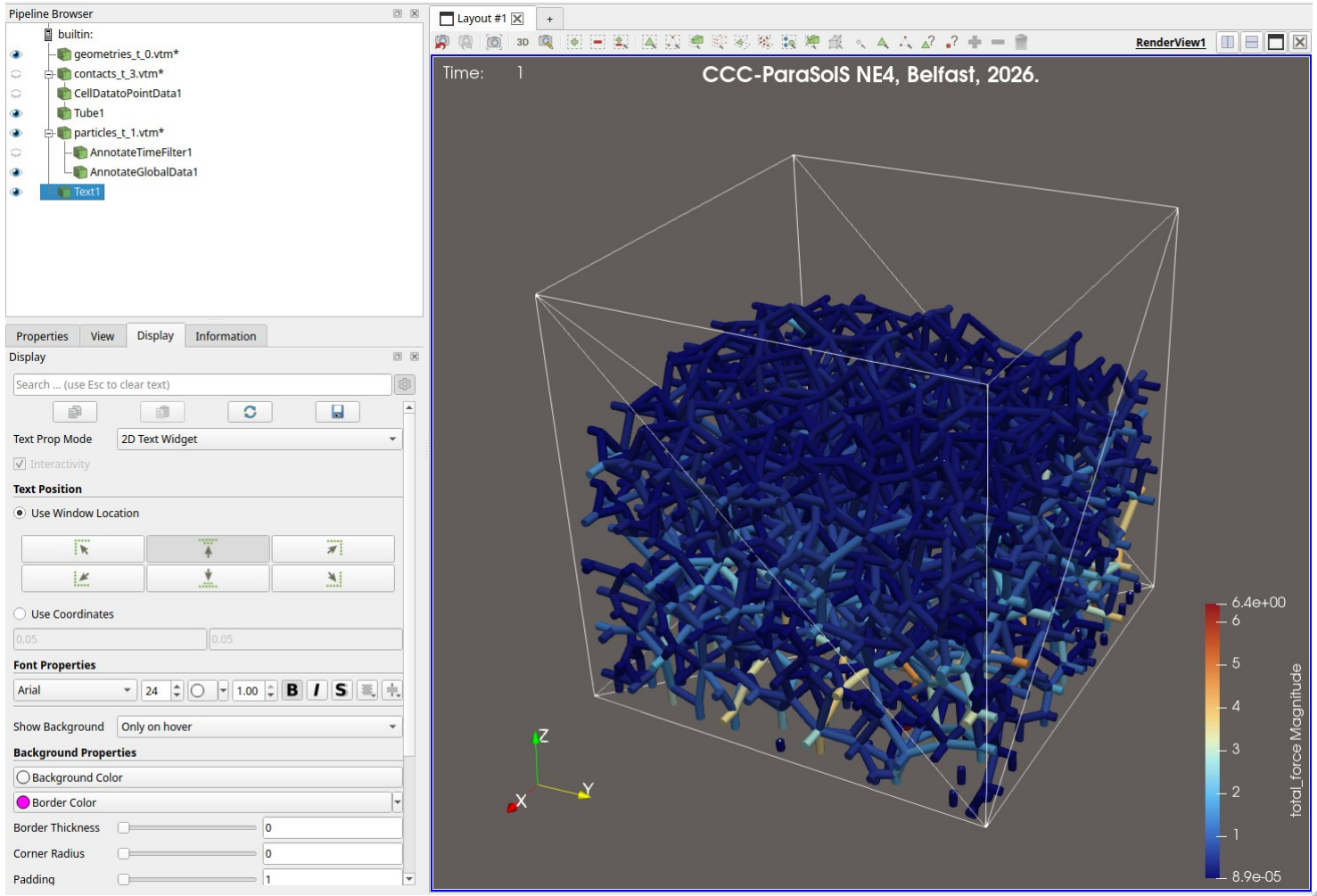
- To scale individually the contact data, which is actually stored as cell data, needs to be converted to point data at each end of the tube
- **Exercise:** Apply **Tube filter** to new CellDatatoPointData1 source and scale using total force



Note: You may get a warning about zero value here as not all contacts exist

The Tube Filter – Individual Scaling

- To scale individually the contact data, which is actually stored as cell data, needs to be converted to point data at each end of the tube
- **Exercise:** Apply **Tube filter** to new CellDatatoPointData1 source and scale using total force



Note: You may get a warning about zero value here as not all contacts exist

Save Effort – Save State

- After all that effort it would be very annoying to have to do that again the next time you wish to view the results

- Avoid the hassle by Saving State
  - From the file menu or shortcuts 
- Reload from State available also

- Can be stored as a ParaView state file (.pvsm) or a raw python file

- **Exercise:** Save state of current pipeline (We will re-use later!)

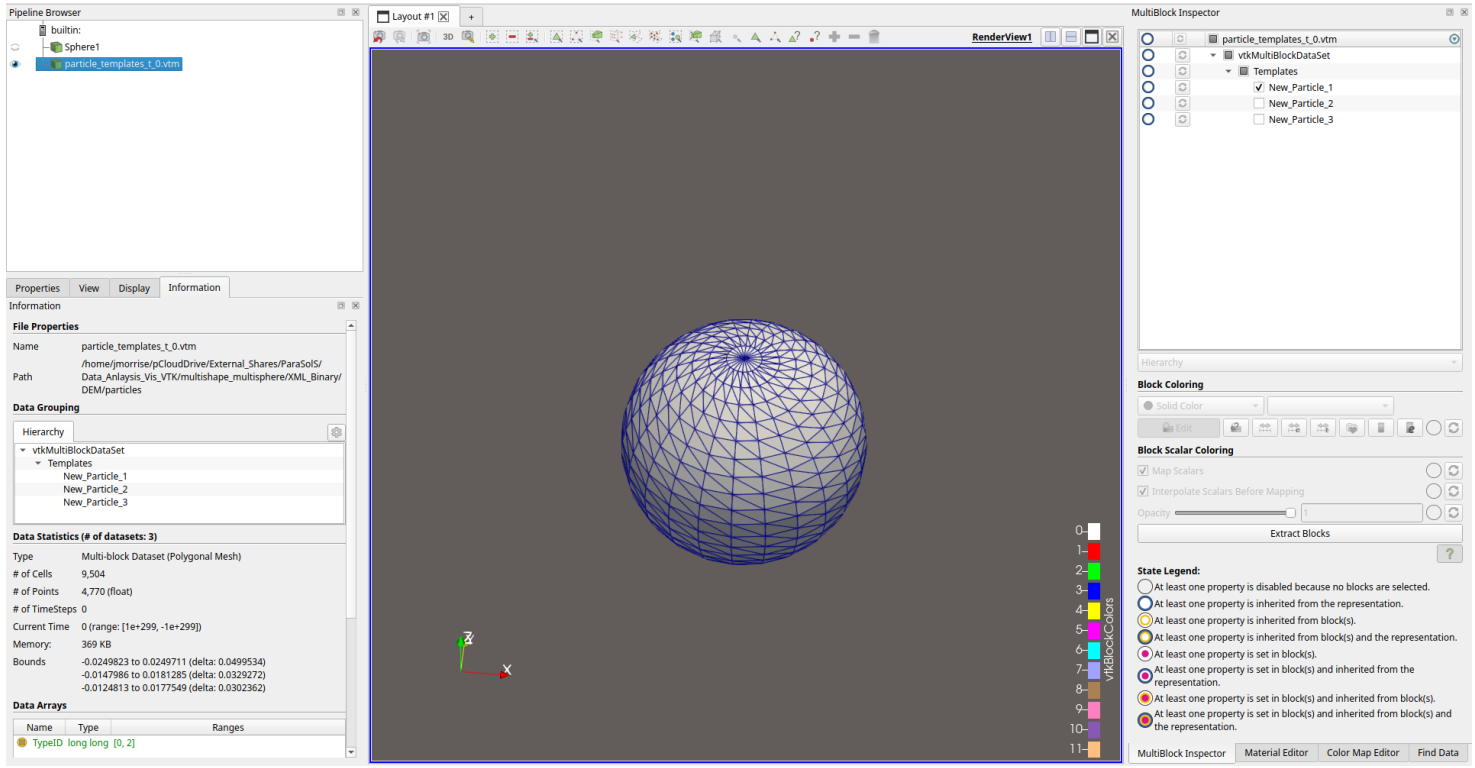
Visualising Multisphere Particles

- As we saw earlier, a sphere is simply a polyhedral particle
- A multisphere is represented as a **complex source** which can either be created within ParaView by joining multiple spheres or by importing sources glyph files created by the solver or elsehwere

- We will visualise the multishape, multisphere dataset provided (multishape_multisphere/XML_Binary/DEM)

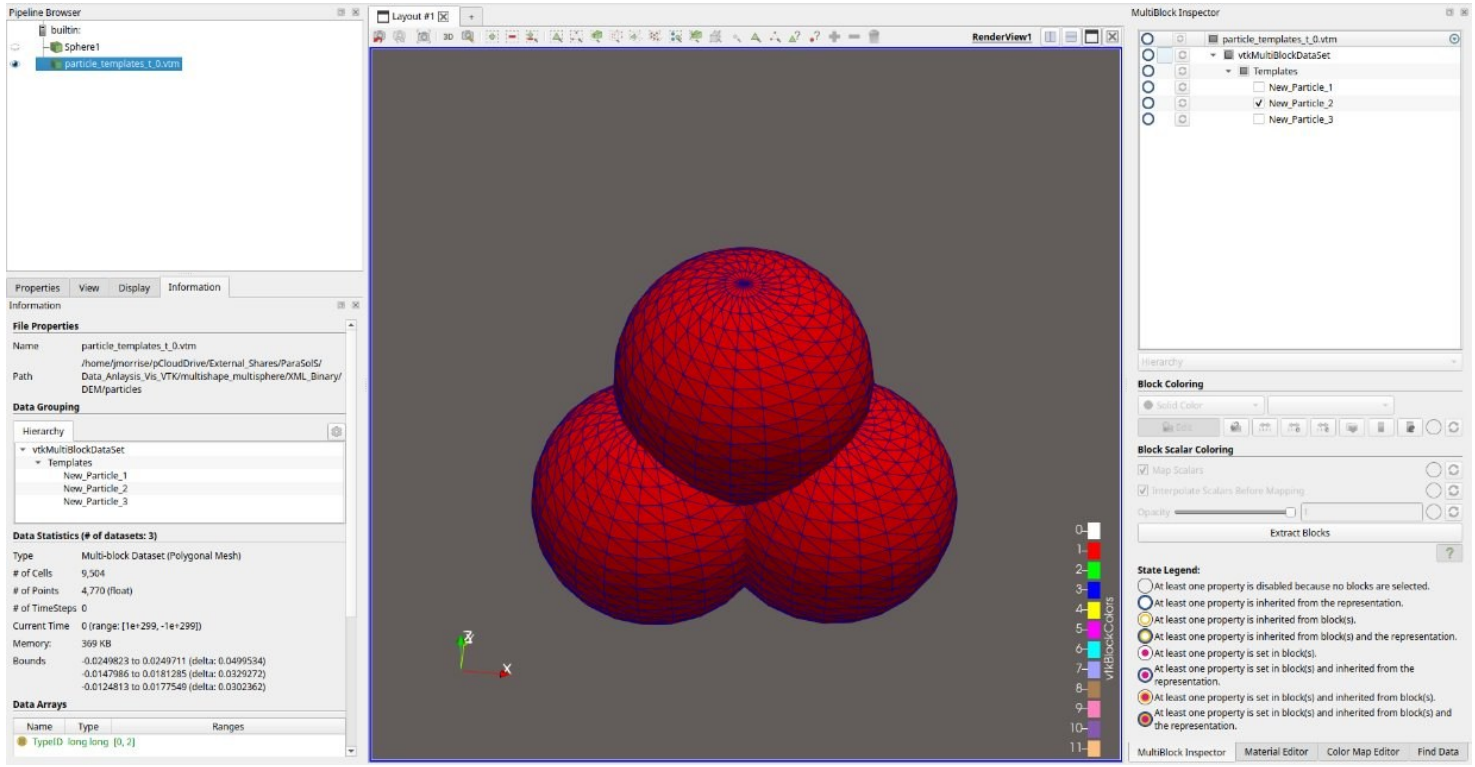
Loading and Viewing Particle Templates

- **Exercise:** Load the particle template multiblock file in a new window or new layout
  - Selectively disable blocks to view each template individually



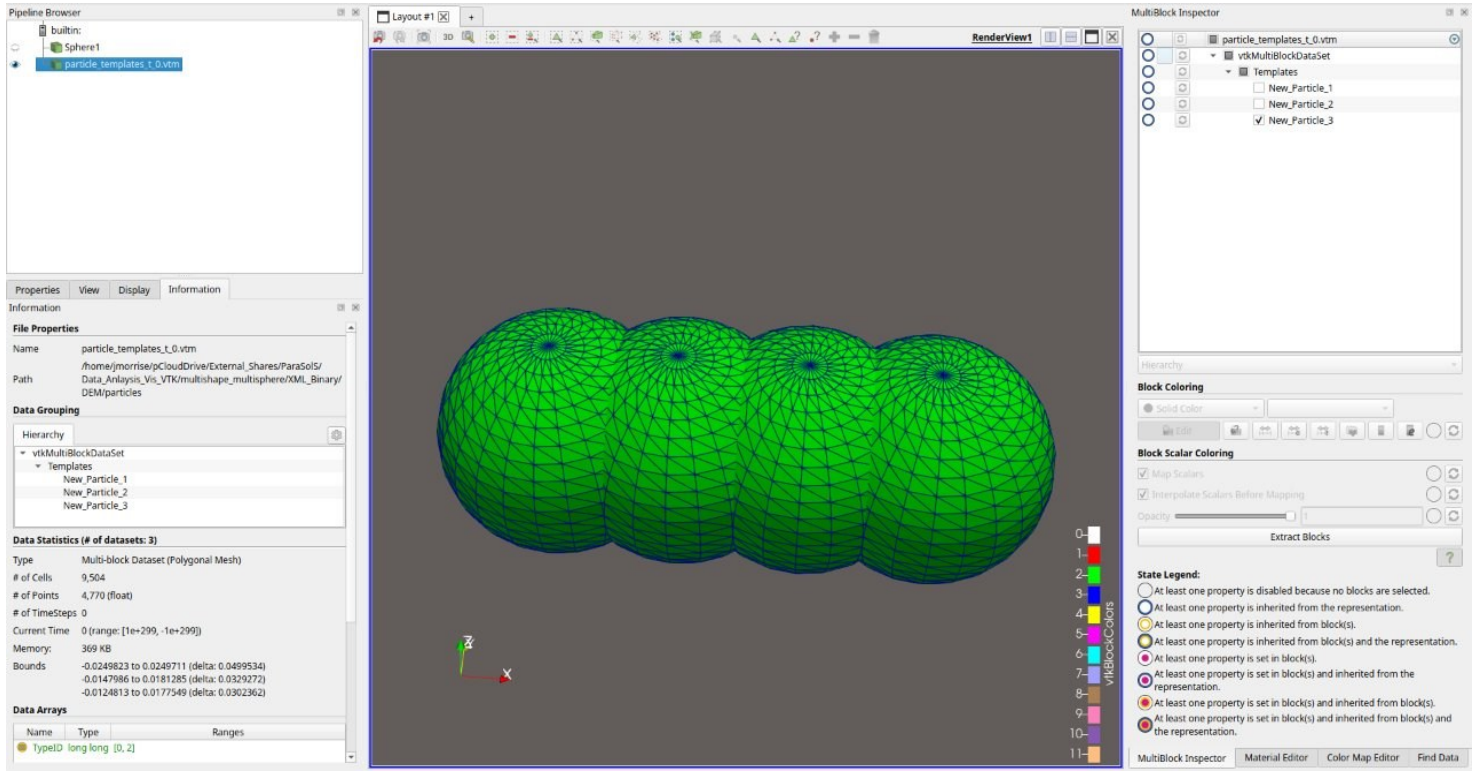
Loading and Viewing Particle Templates

- **Exercise:** Load the particle template multiblock file in a new window or new layout
  - Selectively disable blocks to view each template individually



Loading and Viewing Particle Templates

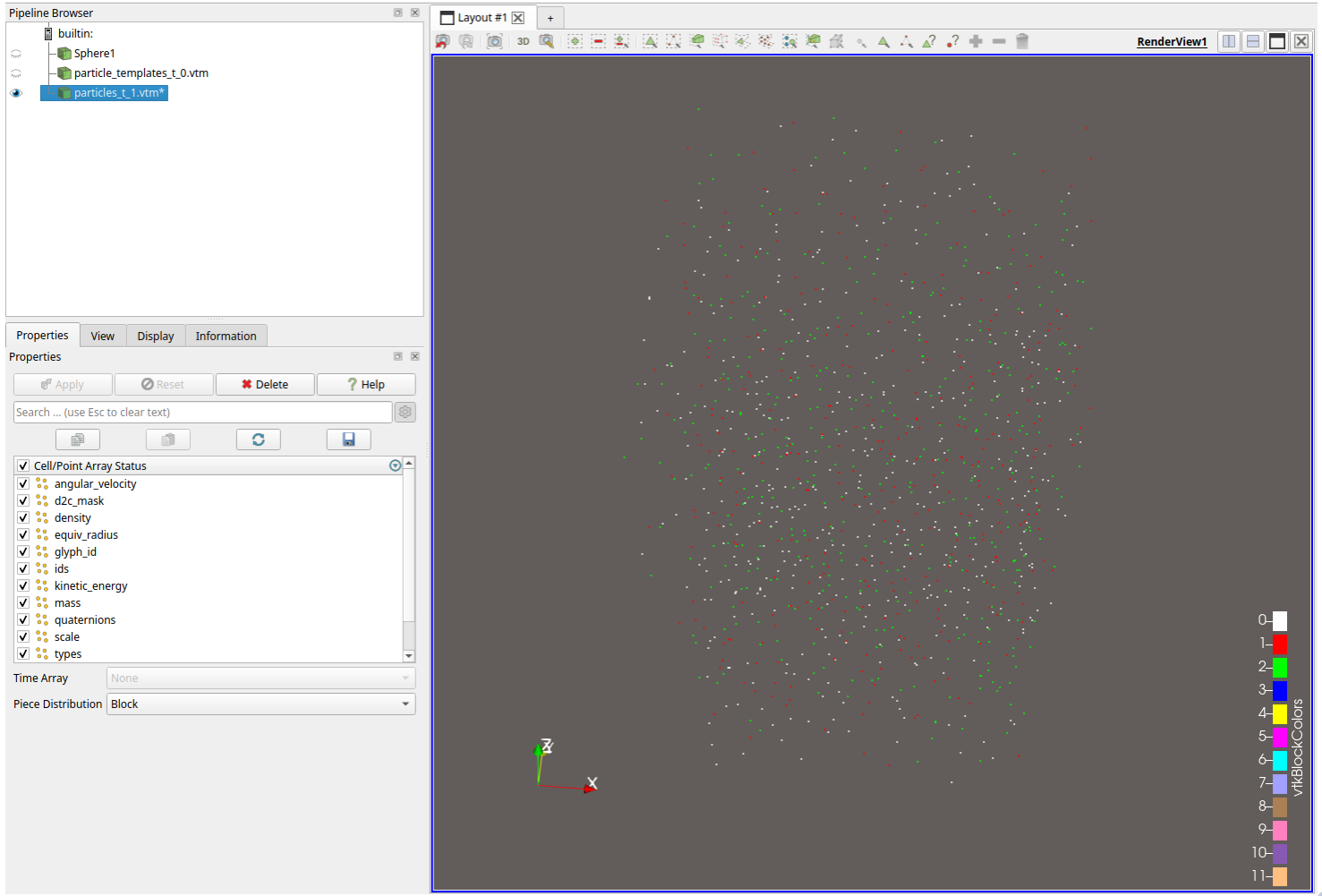
- **Exercise:** Load the particle template multiblock file in a new window or new layout
  - Selectively disable blocks to view each template individually



Loading Particle Data

- **Exercise:** Load the particle multiblock file (particle_t_1.vtm)

- Note: Ensure particle templates are now hidden and to reset the view to show only the particle data

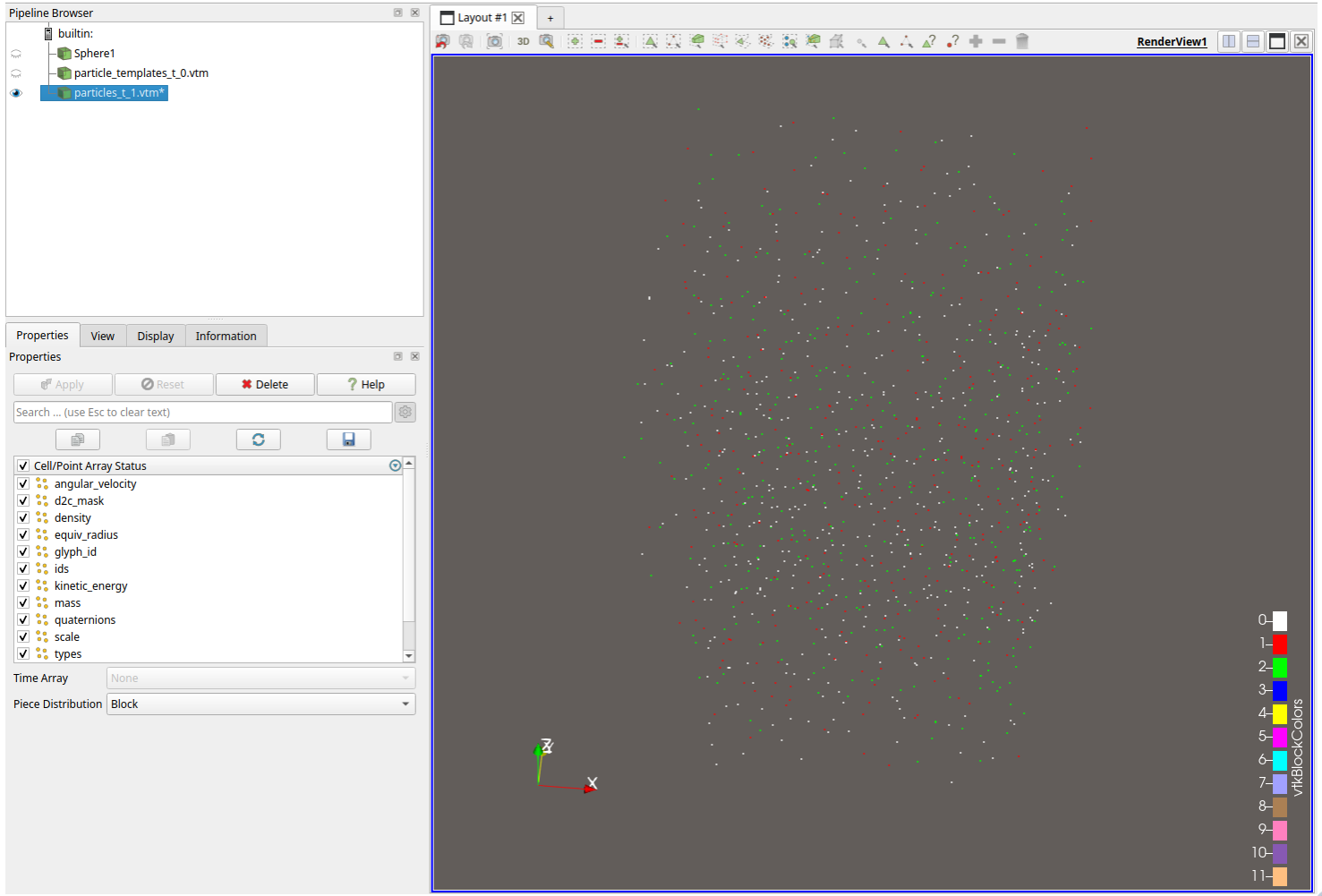


Loading Particle Data

- **Exercise:** Load the particle multiblock file (particle_t_1.vtm)

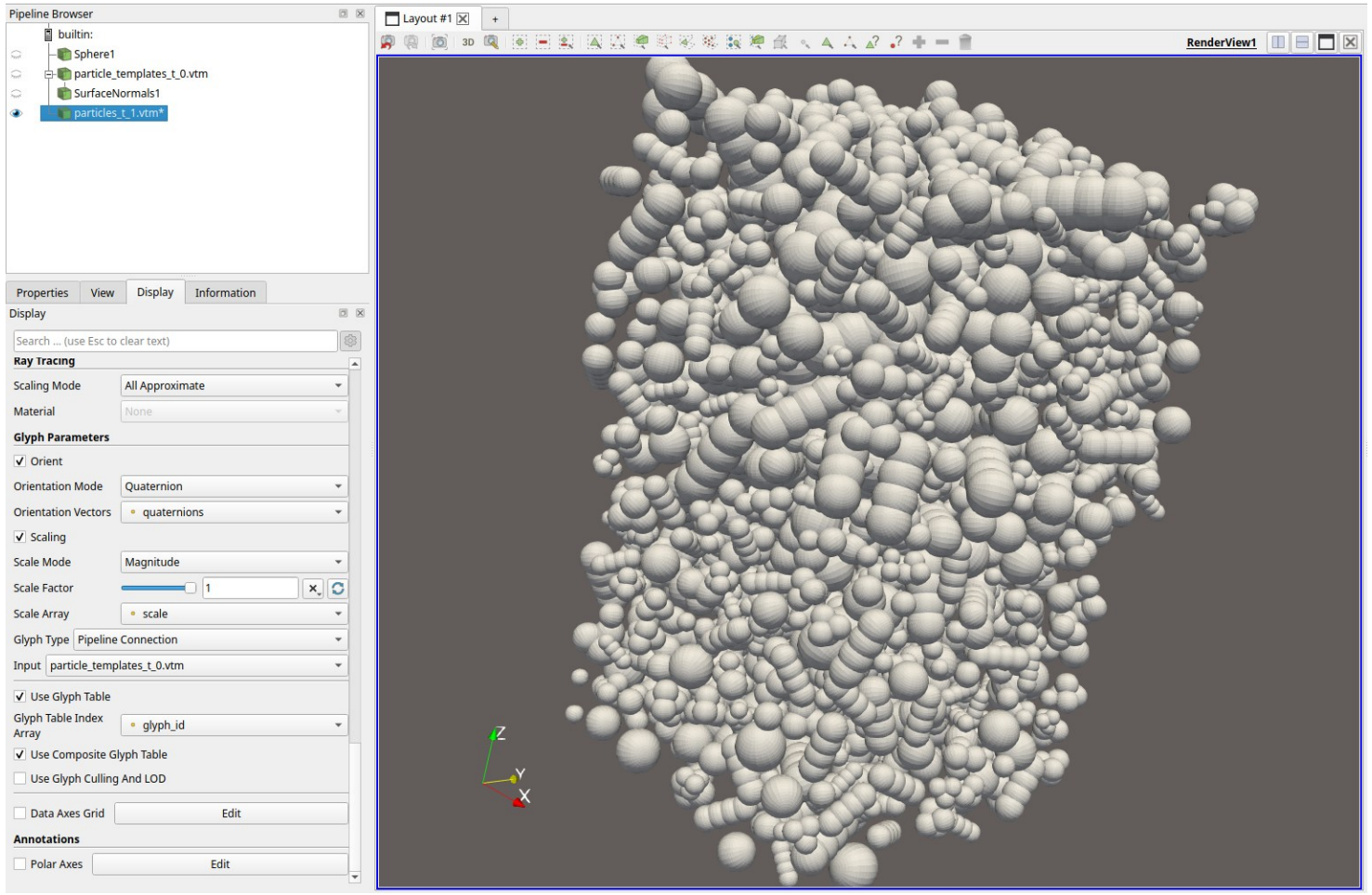
- Note: Ensure particle templates are now hidden and to reset the view to show only the particle data

- You should now have multicoloured dots...



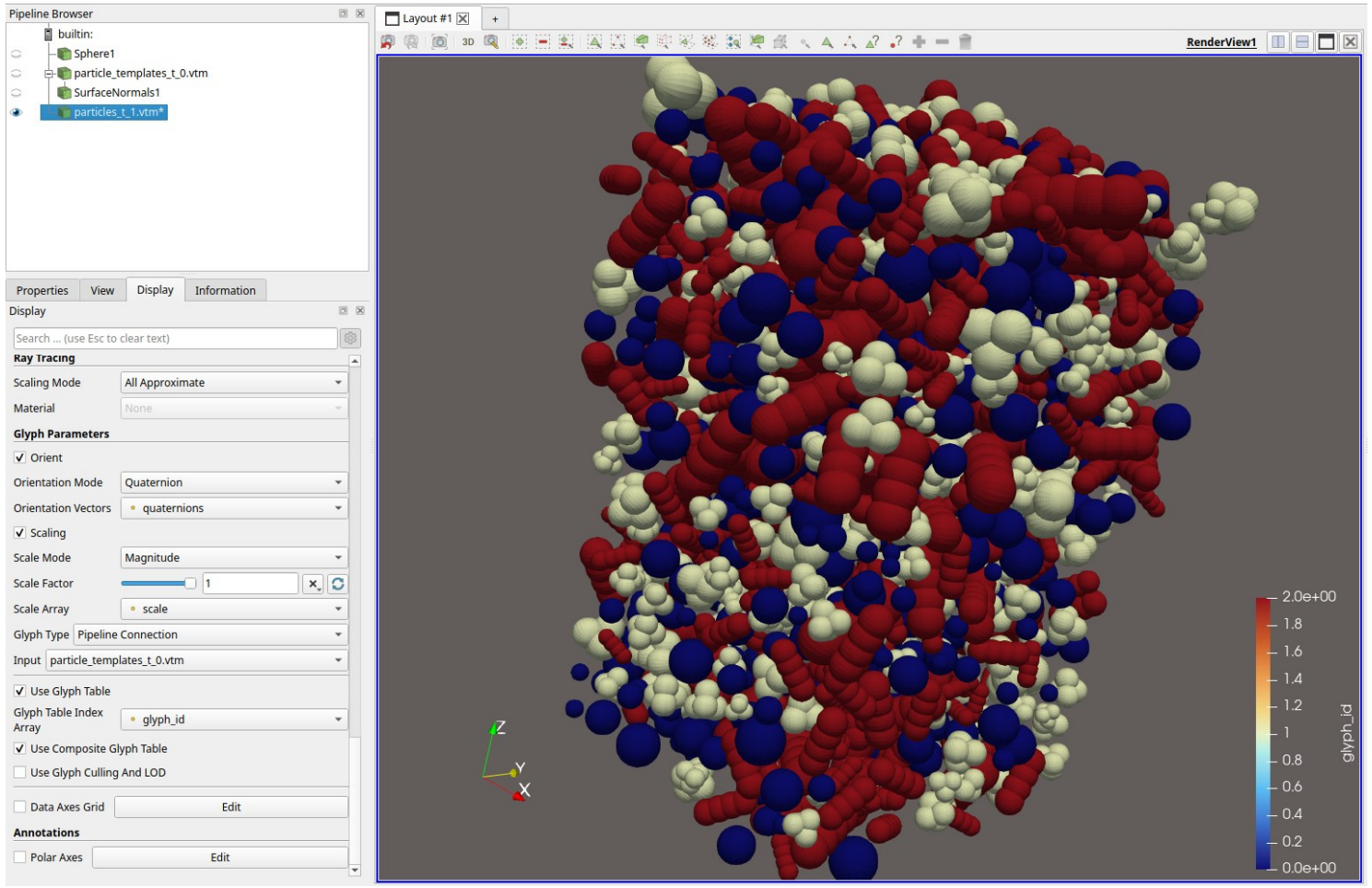
Loading Particle Glyphs

- There are multiple ways to apply glyphs as mentioned earlier
  - We will use 3D from the Pipeline Connection as we have loaded the template data
  - As these are multi-spheres we will also need to define the orientation and scales from data available in the source data set
  - To avoid having to repeat the process for each type we will use a Glyph Table



Loading Particle Glyphs

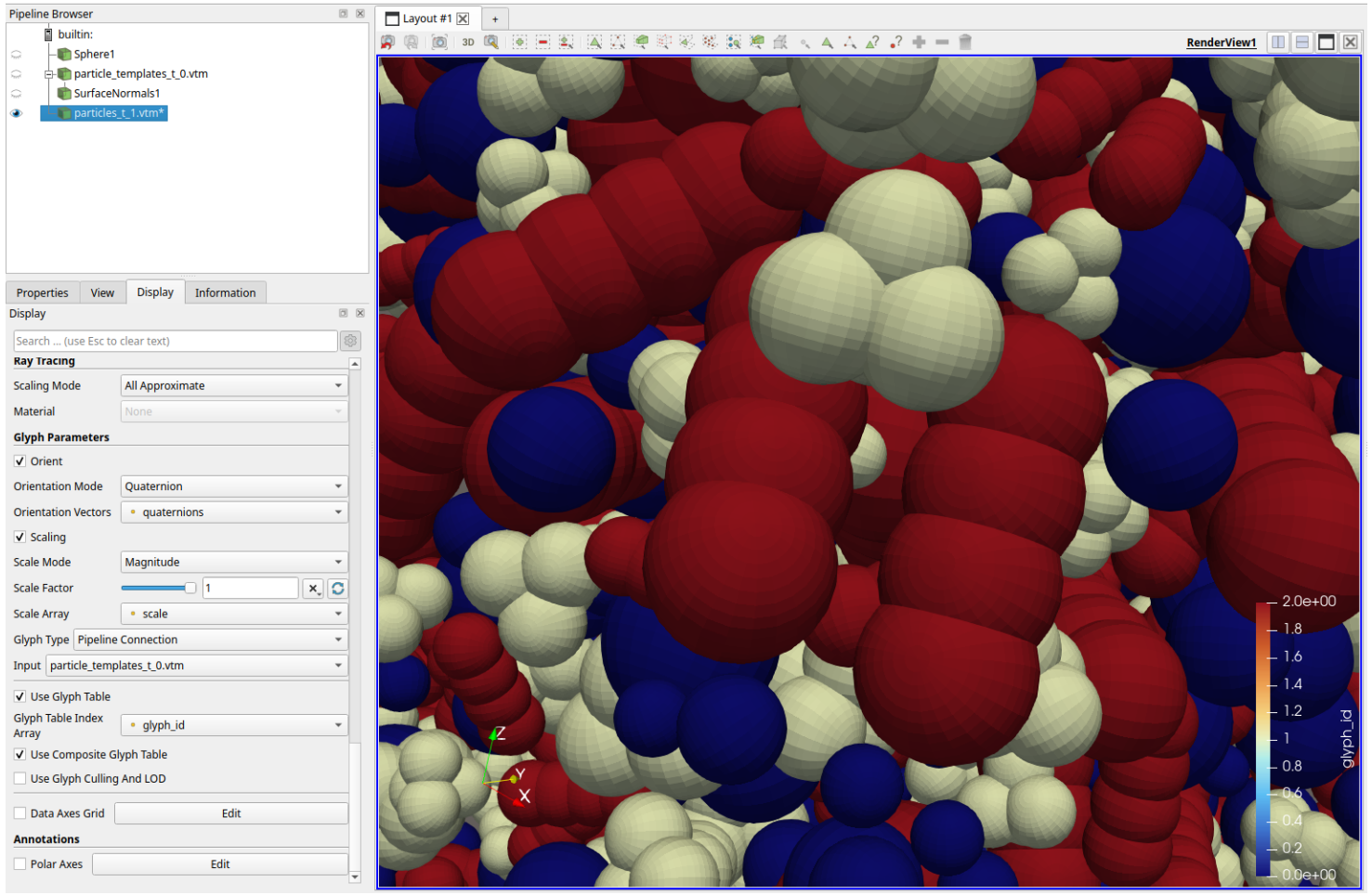
- There are multiple ways to apply glyphs as mentioned earlier
  - We will use 3D from the Pipeline Connection as we have loaded the template data
  - As these are multi-spheres we will also need to define the orientation and scales from data available in the source data set
  - To avoid having to repeat the process for each type we will use a Glyph Table



Why Does It Look A Bit "Blocky"?

- The previous spheres example looked smooth with a resolution of just 16 whereas these glyphs actually have a much higher resolution (24)

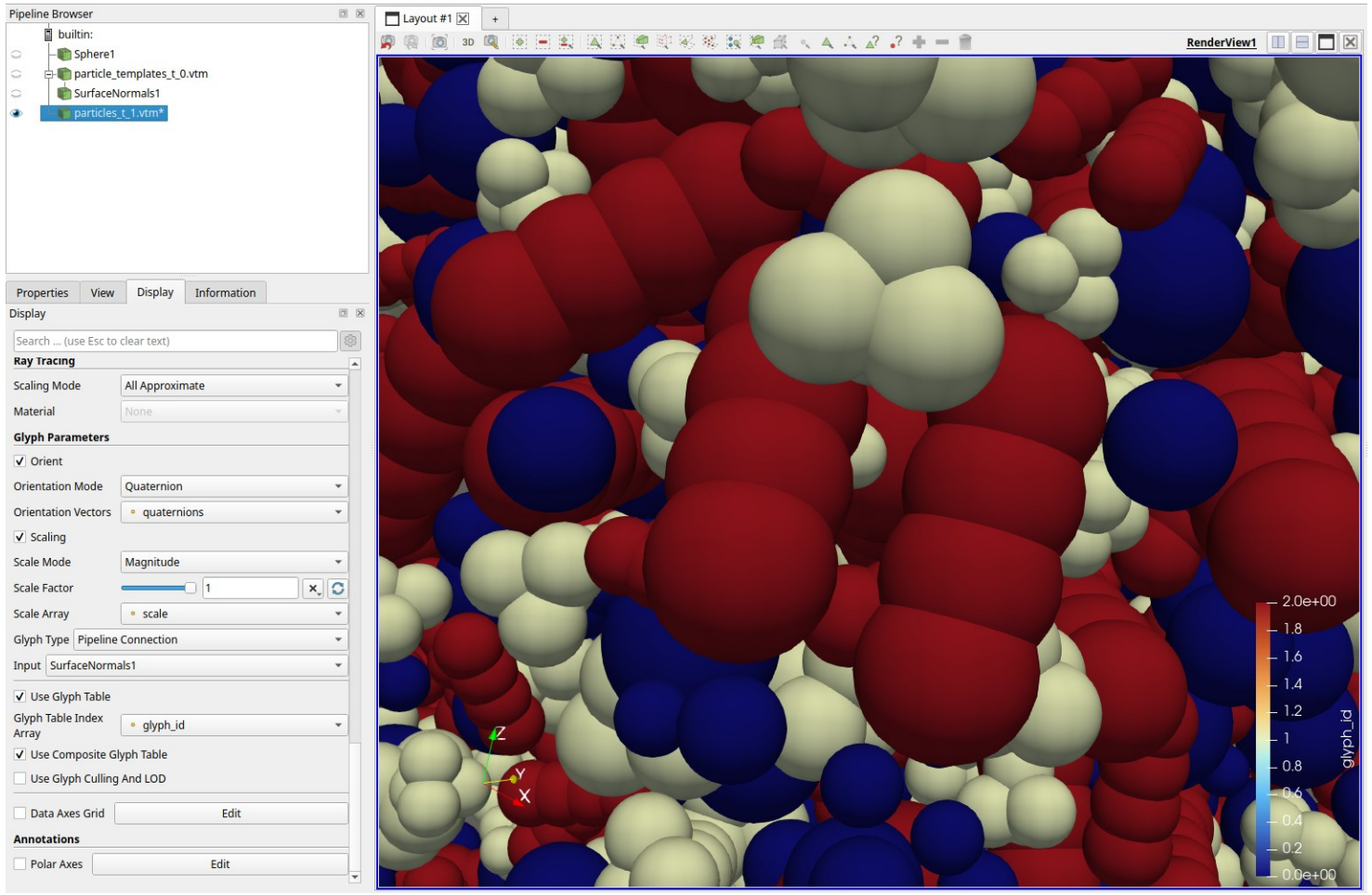
- Answer: Surface Normals
  - These are automatically calculated and applied for a sphere glyph
  - We need to do manually



Adding Surface Normals

- Before applying the template file as a glyph we will use a filter to calculate the surface normals

- **Exercise:** Apply the **Surface Normals** filter to the template data and then update glyphs using this new source

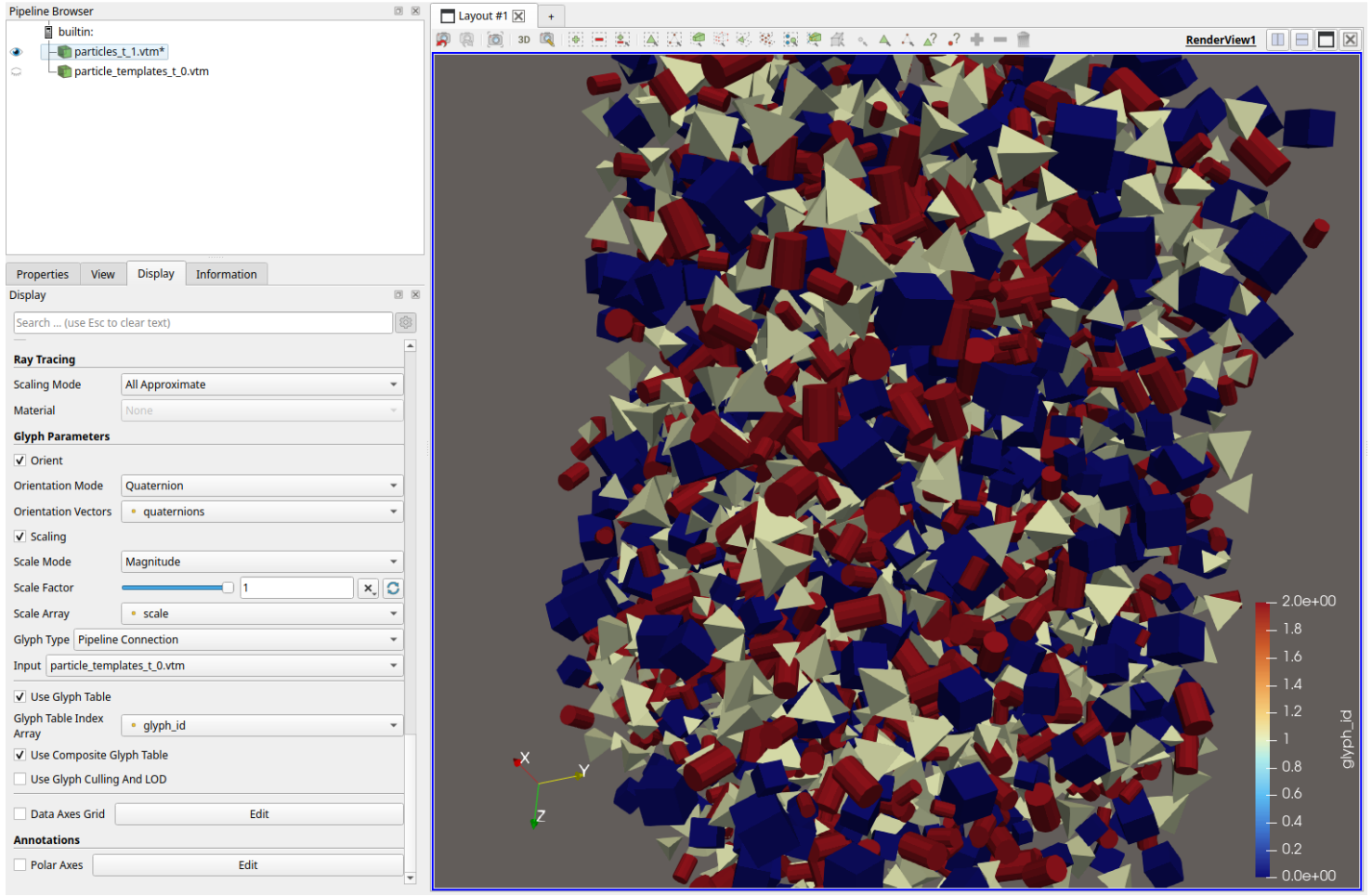


Polyhedral Particles

- We will visualise the multishape, polyhedral dataset provided (multishape_polyhedral/XML_Binary/DEM)

- **Exercise:** Load the particle data and template data and apply as 3D glyphs as before

- Note: Surface normals are not required for polyhedral particles although could be used to improve appearance of cylinders

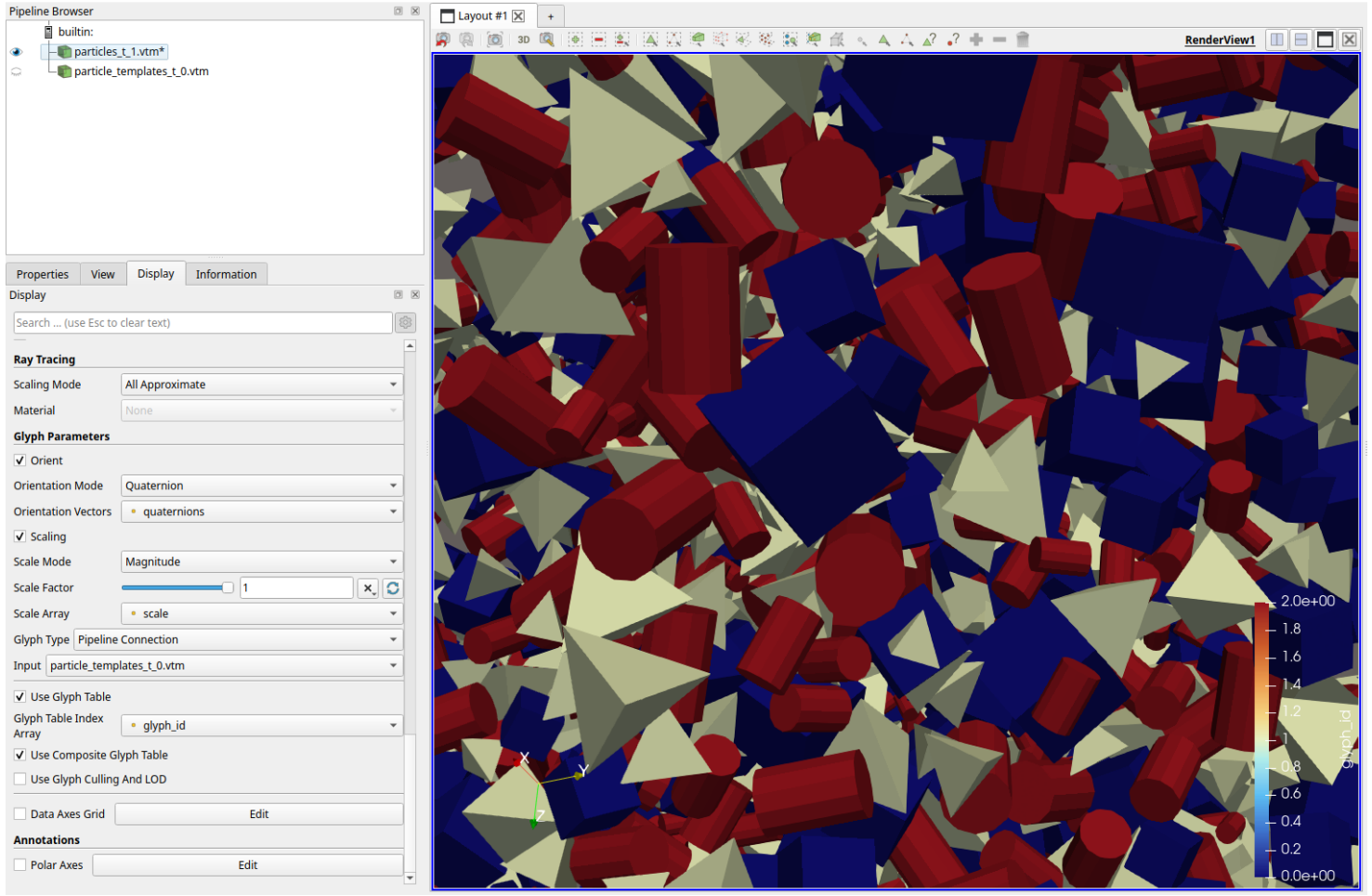


Polyhedral Particles

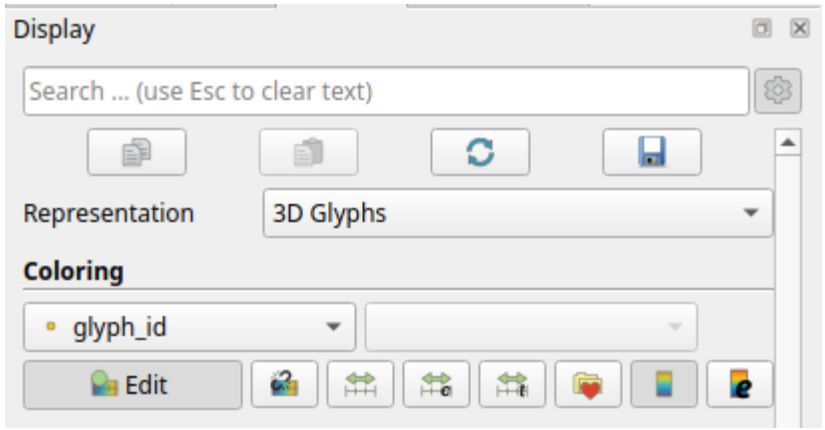
- We will visualise the multishape, polyhedral dataset provided (multishape_polyhedral/XML_Binary/DEM)

- **Exercise:** Load the particle data and template data and apply as 3D glyphs as before

- Note: Surface normals are not required for polyhedral particles although could be used to improve appearance of cylinders

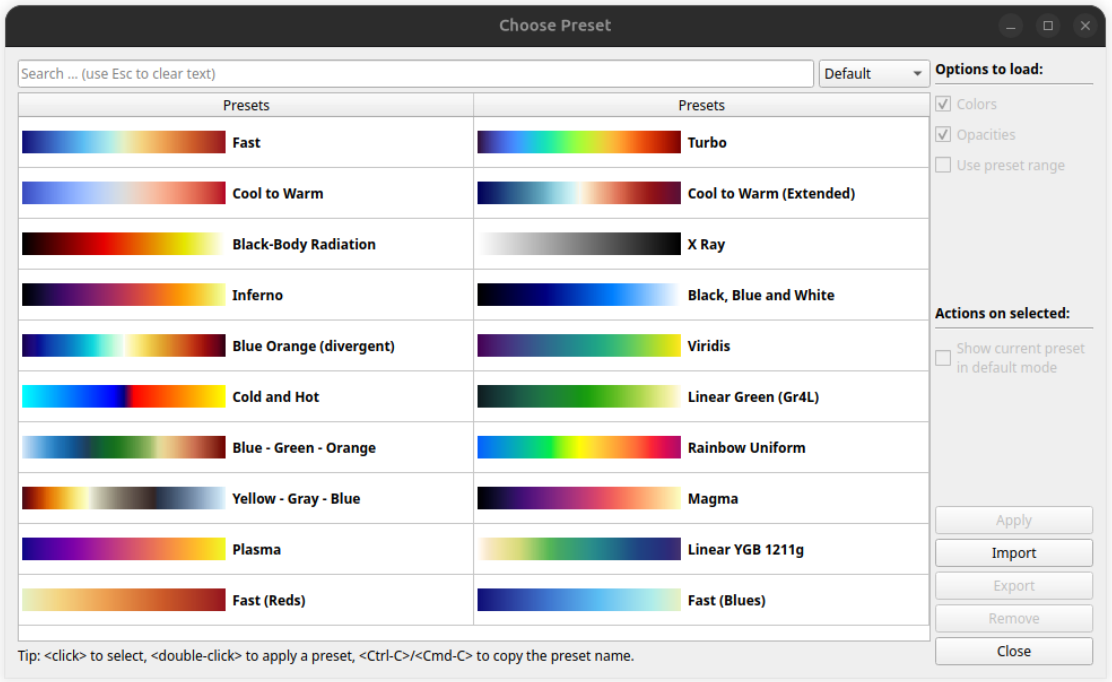


Exploring Colourmaps

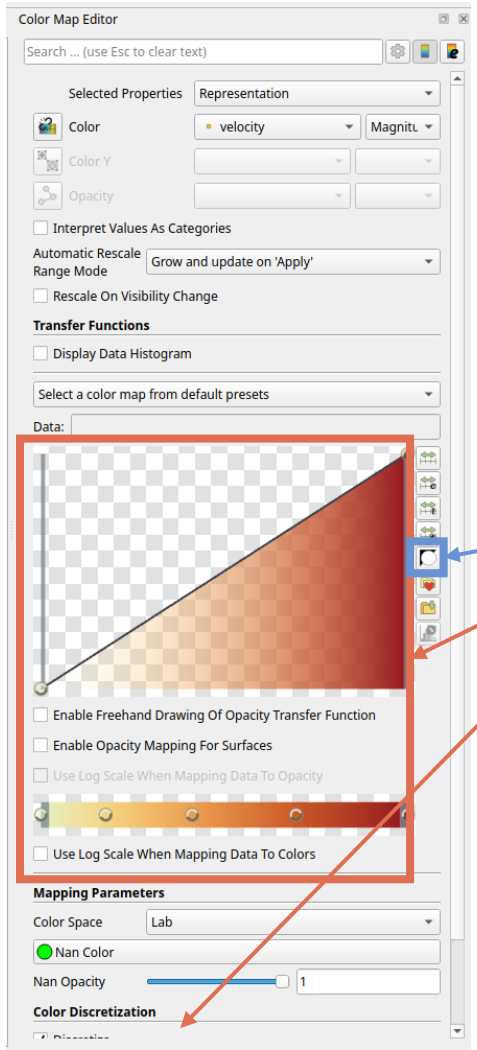


- In the **Display properties** there is a small section dedicated to colouring
  - First line is variable and component selection
  - Next line has all the colourmap related options
    - Edit Opens the Colour Map Editor Pane
    - Use separate colour map for different data sources
    - Rescale to data range (current timestep)
    - Rescale to custom range
    - Rescale to data range over all timesteps
    - Open window with list of favourite colour maps
    - Show/hide colour bar
    - Edit colour bar properties

Exploring Colourmaps



Default windows shows your favourites which you can modify

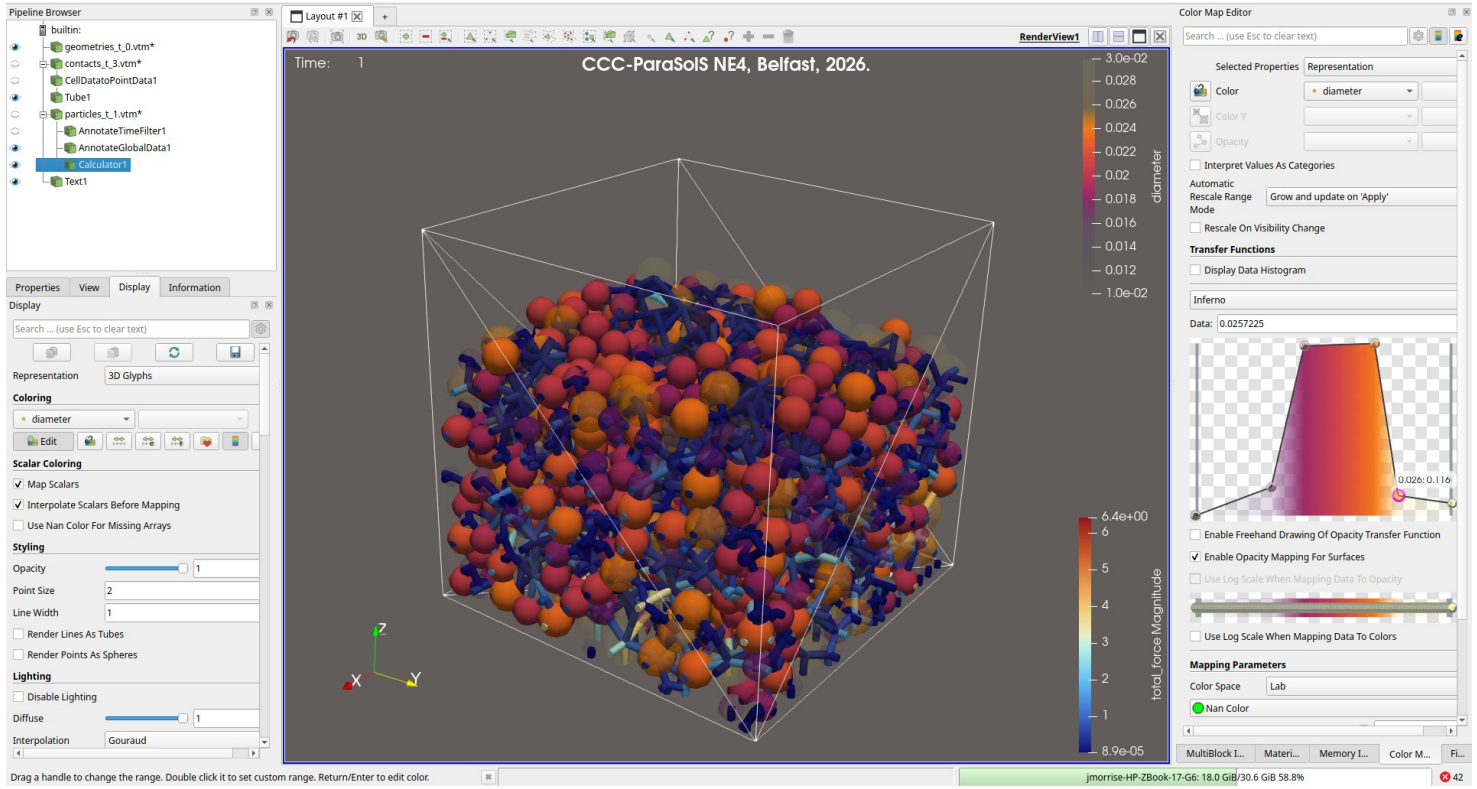


The **Colour Map Editor** allow you to access advanced settings related to the selected colour map

- All previous range options
- Reverse option
- Opacity mapping
- Colour discretisation

Opacity Mapping

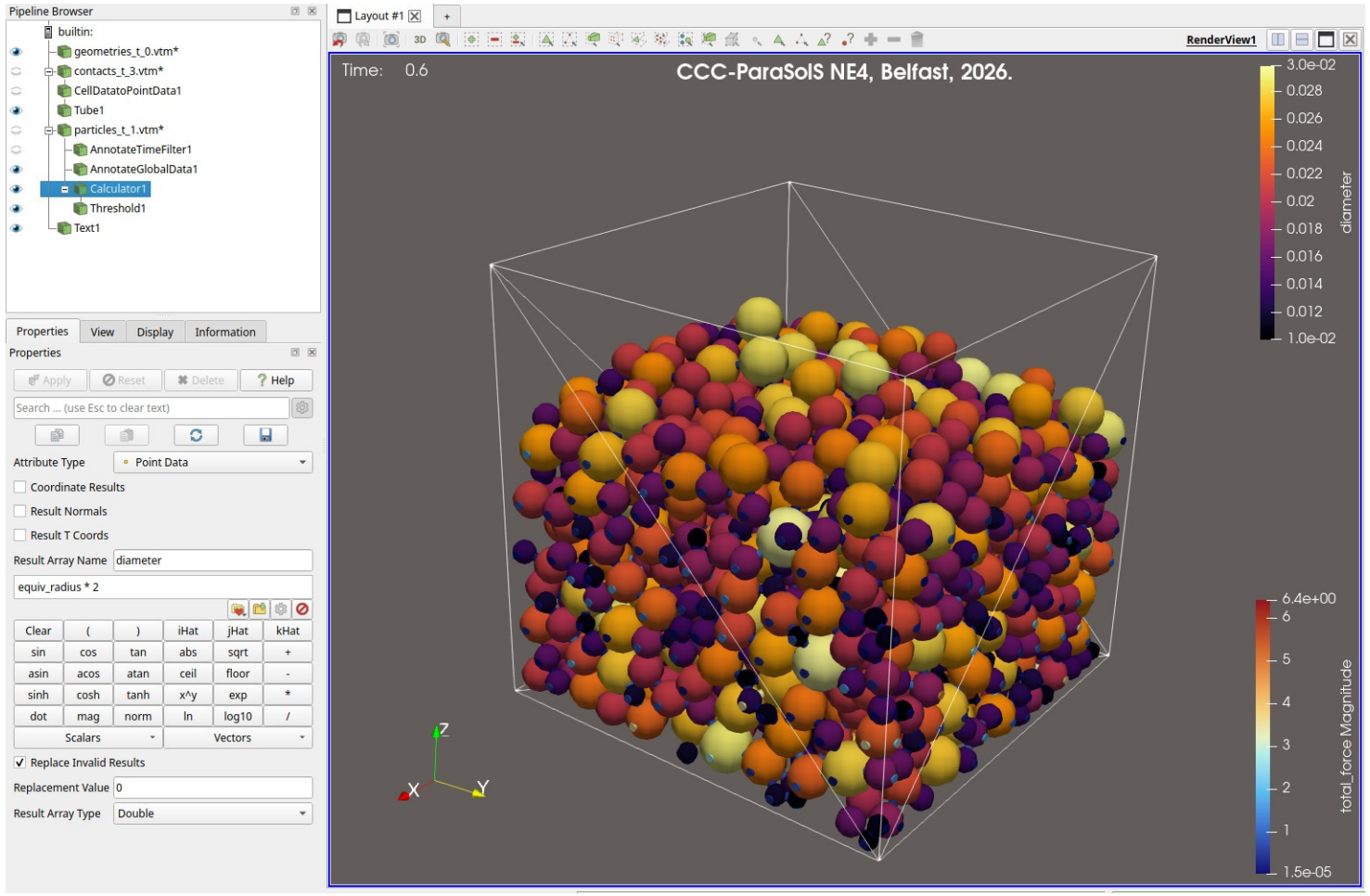
- **Exercise:** Visualise spherical particles and contacts.
  - Colour particles according to volume using a sequential colour map of your choice
  - Apply an opacity map to remove the largest and smallest particles from view



Hint: You may just want to reload our saved state from earlier

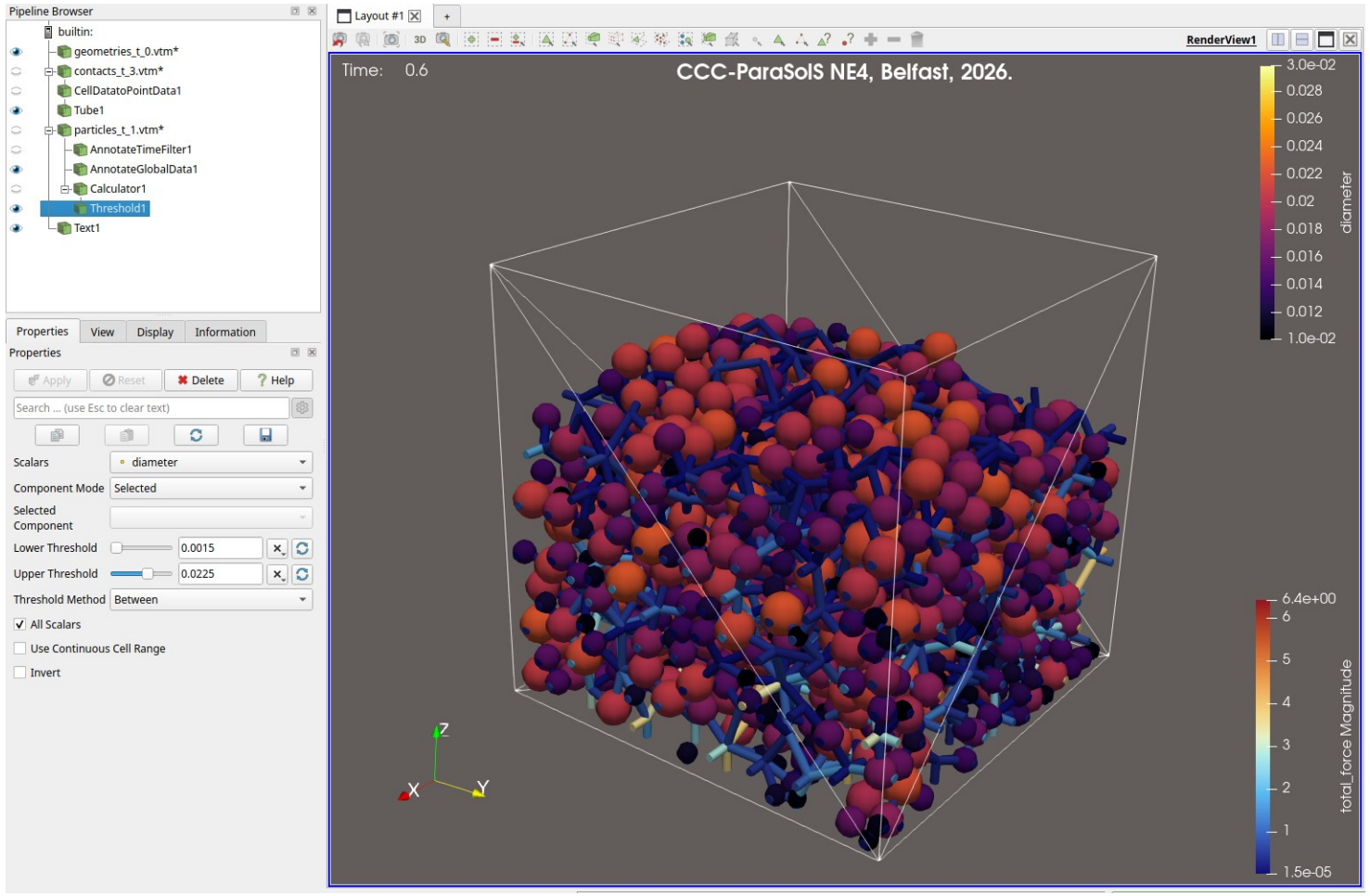
Applying A Threshold

- The **Threshold filter** allows the truncation of the data source based on a selected criteria
- **Exercise:** Apply a threshold to particle by diameter
  - Limit to particles between 0.0015 and 0.0025 m
- Note: since this creates a new source, you need to re-add any Glyphs and disable any other displayed sources



Applying A Threshold

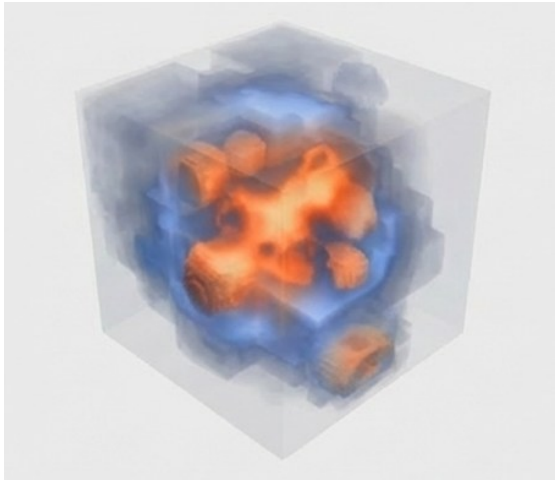
- The **Threshold filter** allows the truncation of the data source based on a selected criteria
- **Exercise:** Apply a threshold to particle by diameter
  - Limit to particles between 0.0015 and 0.0025 m
- Note: since this creates a new source, you need to re-add any Glyphs and disable any other displayed sources



Opacity and Threshold - Difference?

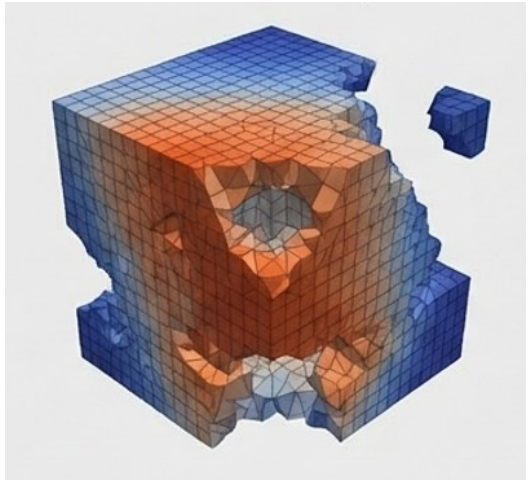
Opacity – Visual Modification

- Modifies display properties via colourmaps
- No impact on pipeline – rendered on GPU
- Use if you want to *"peek inside"*



Threshold - Data Modification

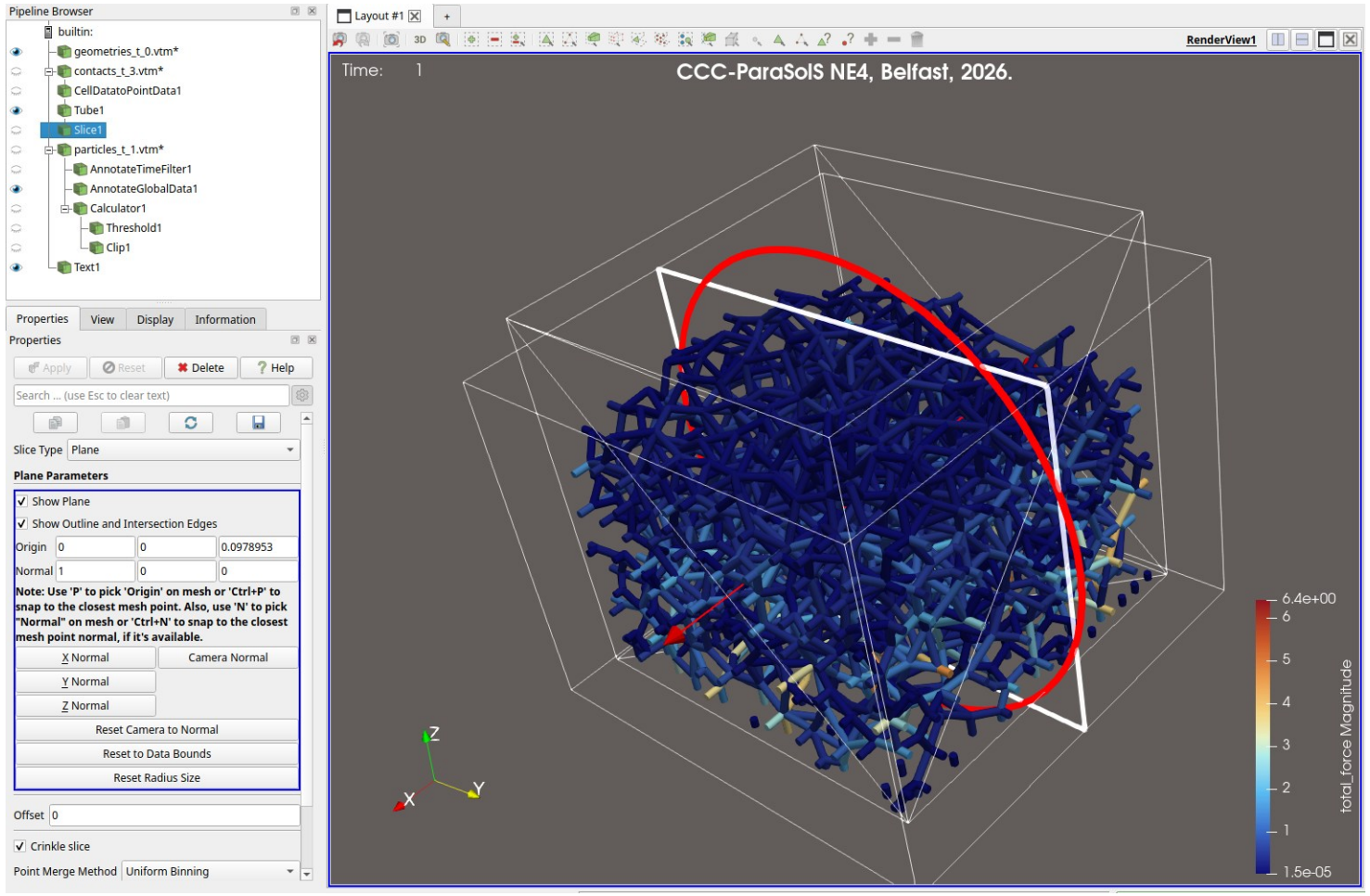
- Extracts & removes cells from the dataset
- Heavy computation and creates new source (UG/vtu) in pipeline
- Use when you need to run further complex filters on the data



Applying A Slice Filter

- Apply a **Slice filter** to get new 2D representation from the mesh

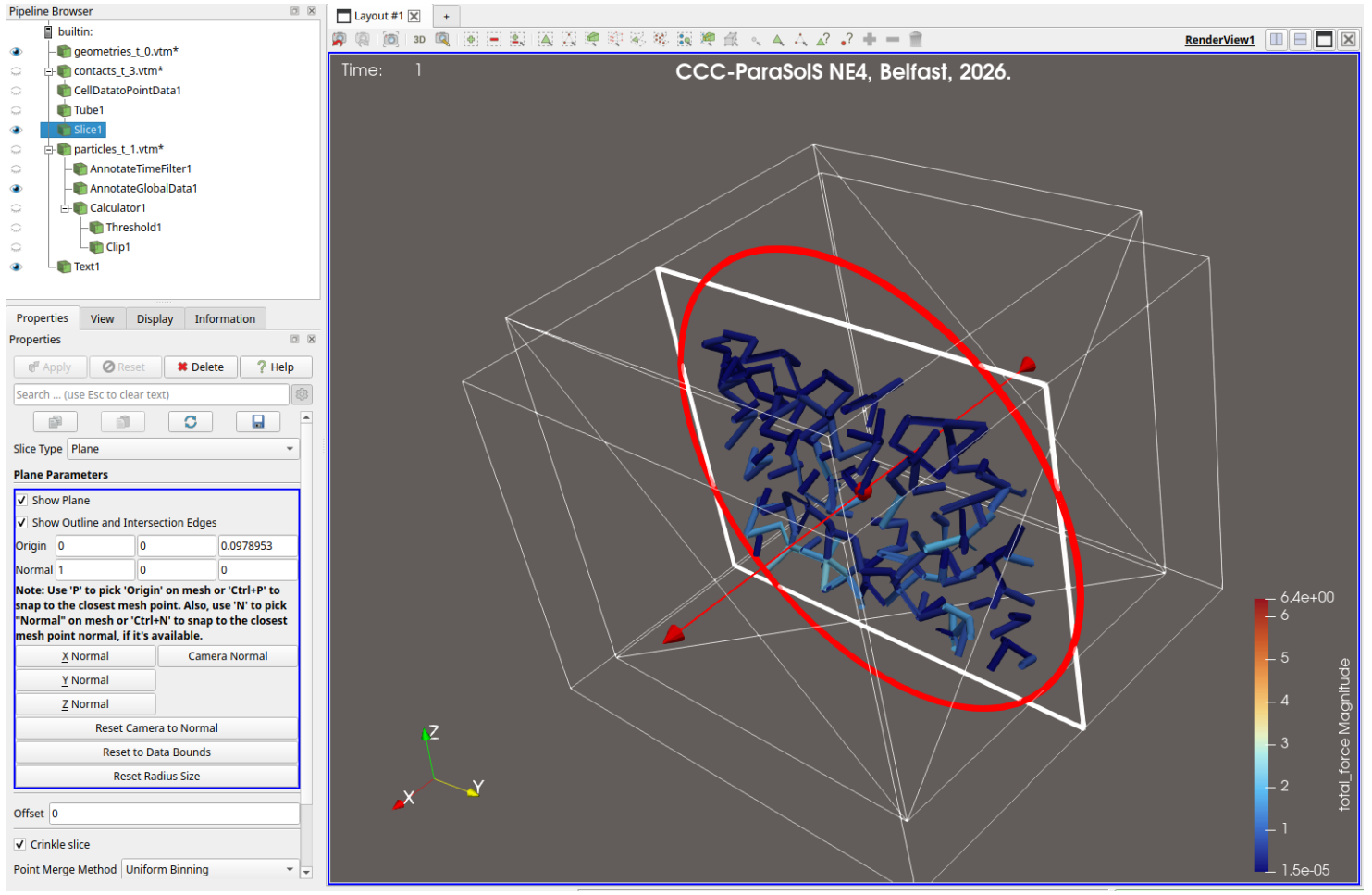
- **Exercise:** Slice the contact network ("Tube1" source) with a plane
  - Hint: enable the "Crinkle slice" option to get the full tube



Applying A Slice Filter

- Apply a **Slice filter** to get new 2D representation from the mesh

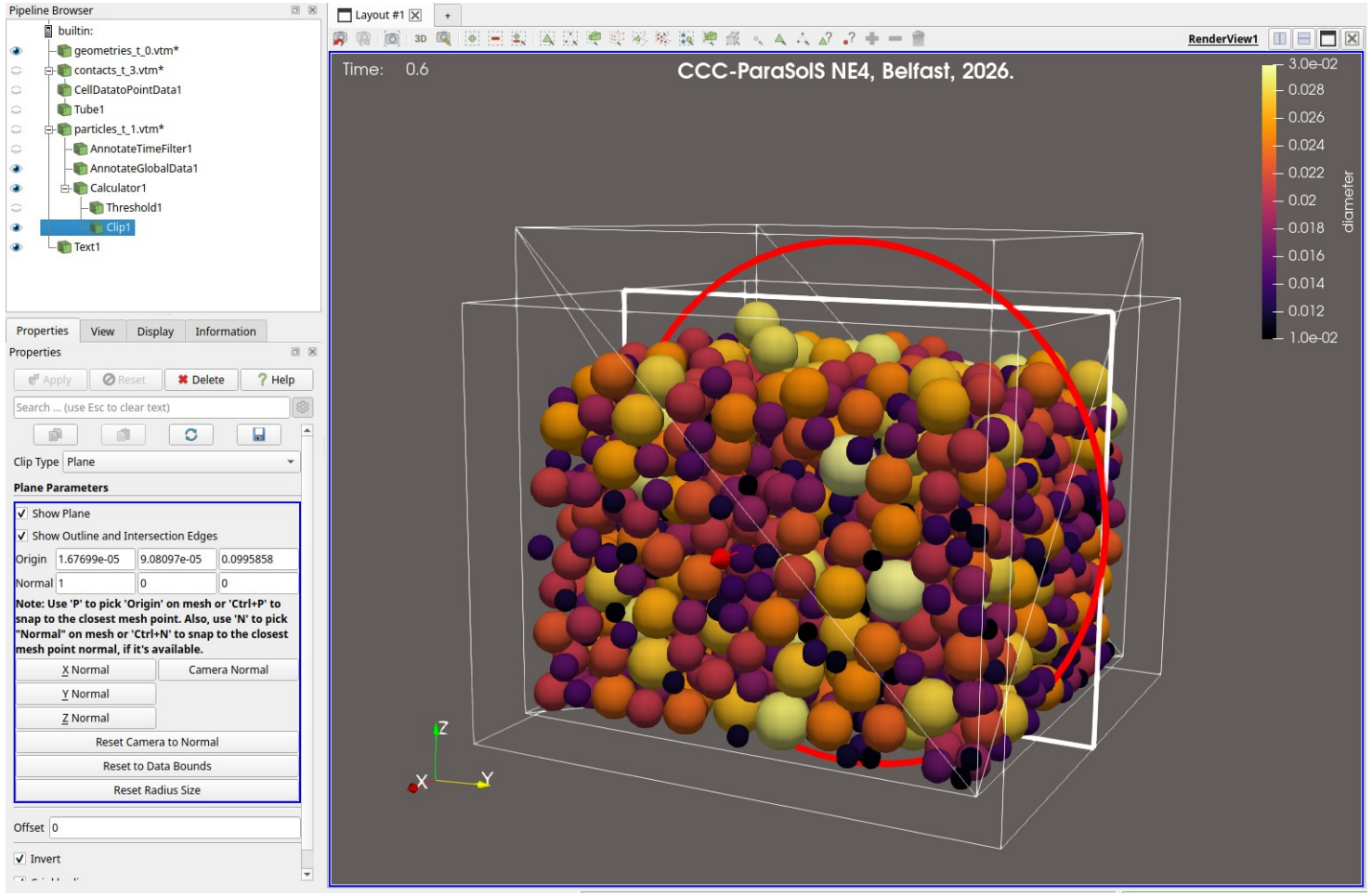
- **Exercise:** Slice the contact network ("Tube1" source) with a plane
  - Hint: enable the "Crinkle slice" option to get the full tube



Applying A Clipping Filter

- The **Clipping filter** allows the truncation of the data source based on a selected clip type
- **Exercise:** Apply a clipping plane to the particle data

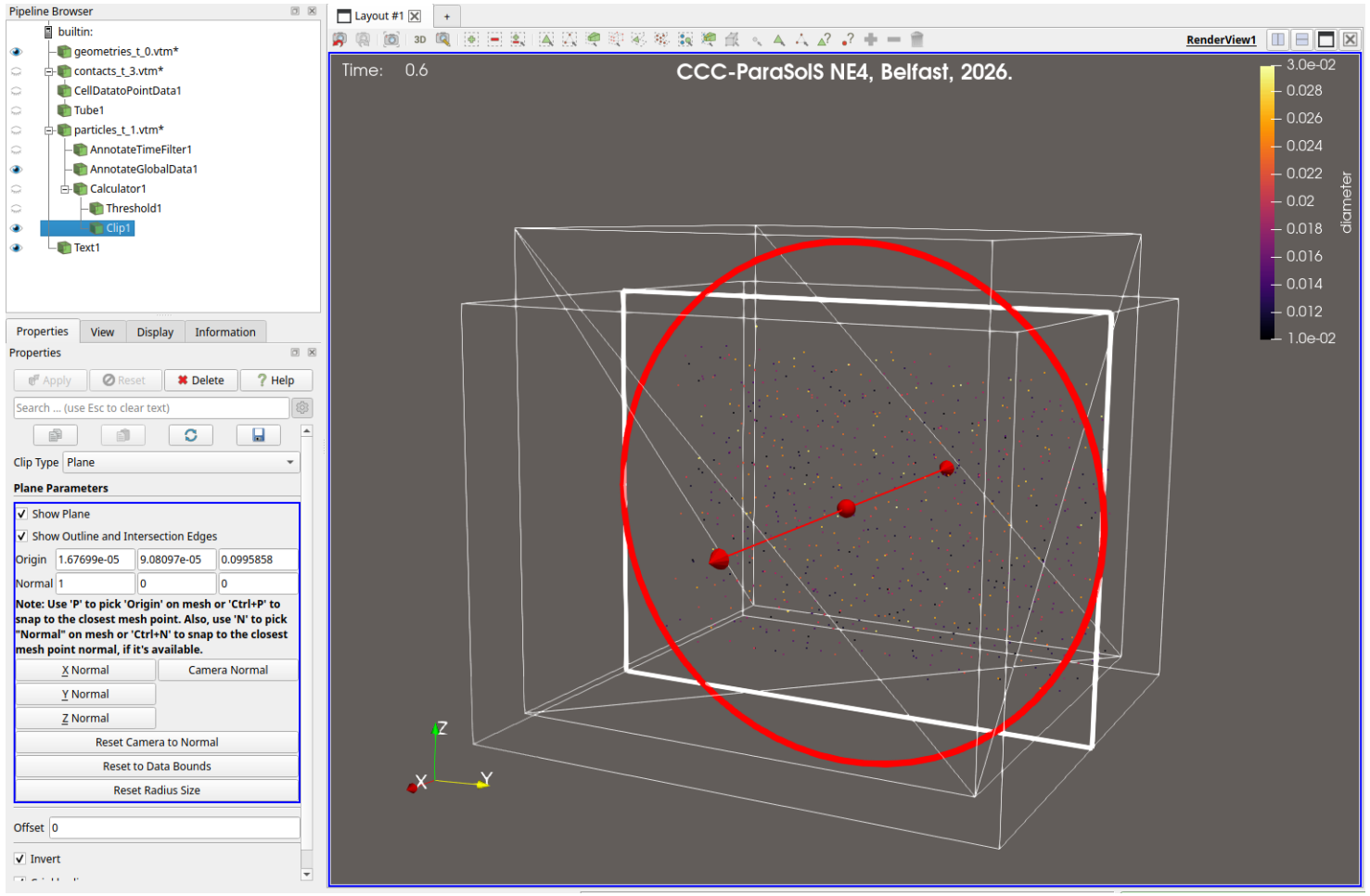
- Note: since this creates a new source, you need to re-add any Glyphs and disable any other displayed sources



Applying A Clipping Filter

- The **Clipping filter** allows the truncation of the data source based on a selected clip type
- **Exercise:** Apply a clipping plane to the particle data

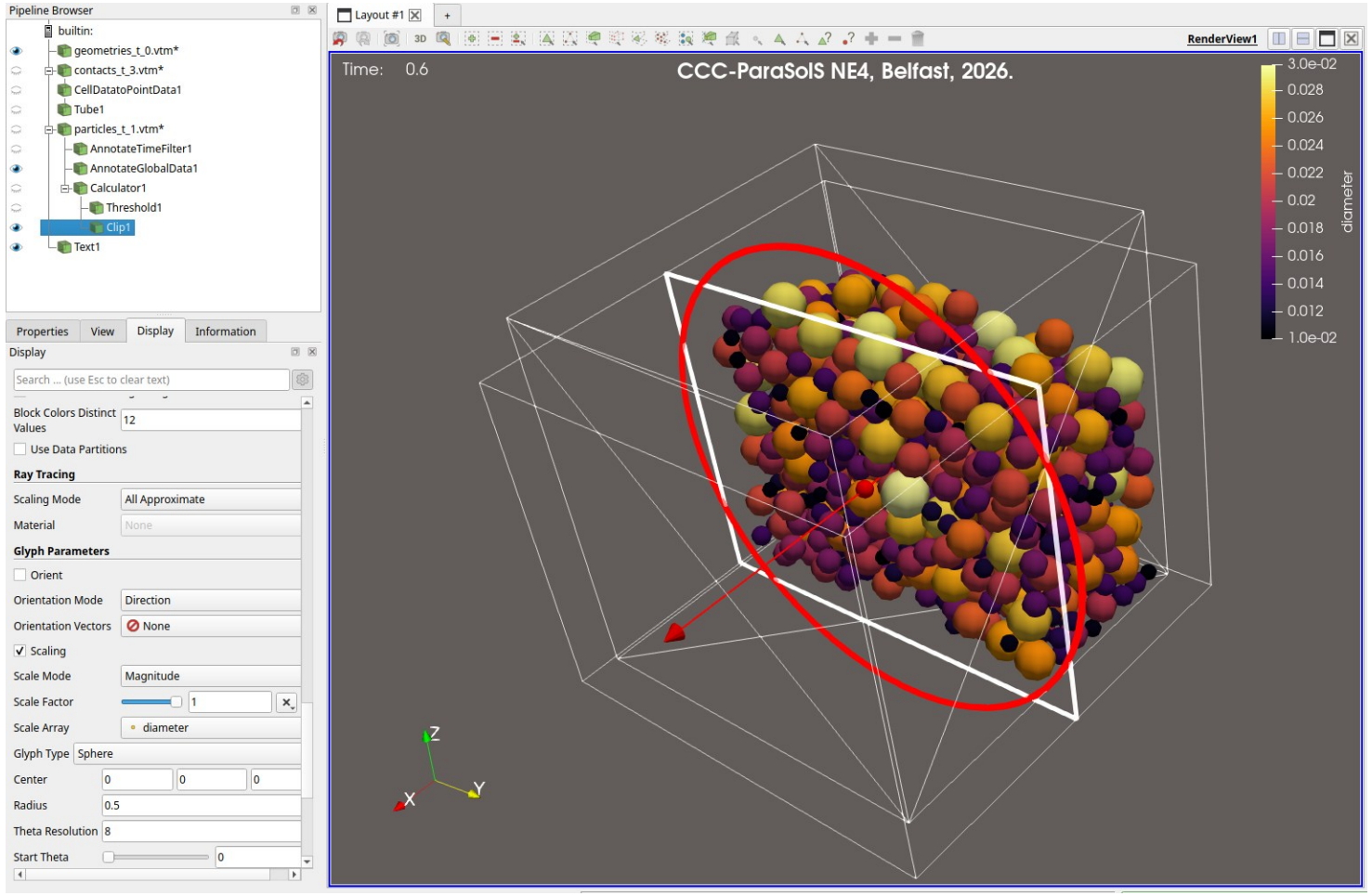
- Note: since this creates a new source, you need to re-add any Glyphs and disable any other displayed sources



Applying A Clipping Filter

- The **Clipping filter** allows the truncation of the data source based on a selected clip type
- **Exercise:** Apply a clipping plane to the particle data

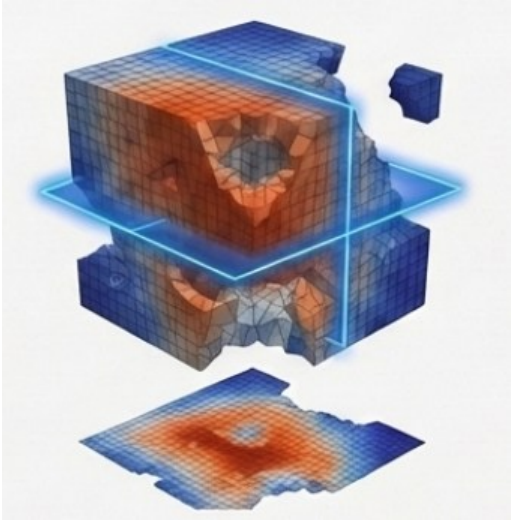
- Note: since this creates a new source, you need to re-add any Glyphs and disable any other displayed sources



Slicing and Clipping

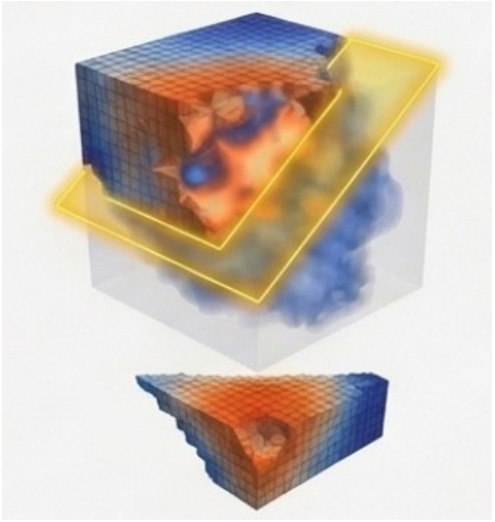
Slicing

- Reduces Dimensionality
  - 3D → 2D
  - Memory Efficient
  - New source in pipeline



Clipping

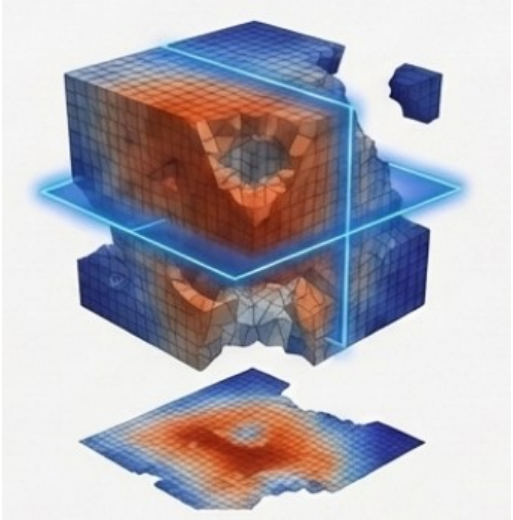
- Retains Volume
  - Removes part of dataset
  - New source in pipeline



Slicing and Clipping

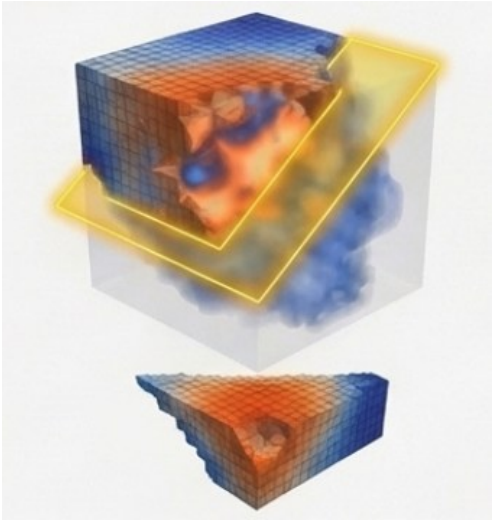
Slicing

- Reduces Dimensionality
  - 3D → 2D
  - Memory Efficient
  - New source in pipeline



Clipping

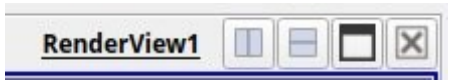
- Retains Volume
  - Removes part of dataset
  - New source in pipeline



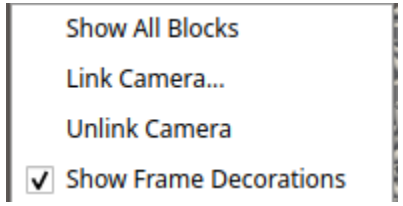
Both filters are incredibly useful but may be of limited use for DEM data. Much better suited to FEM/CFD-like data

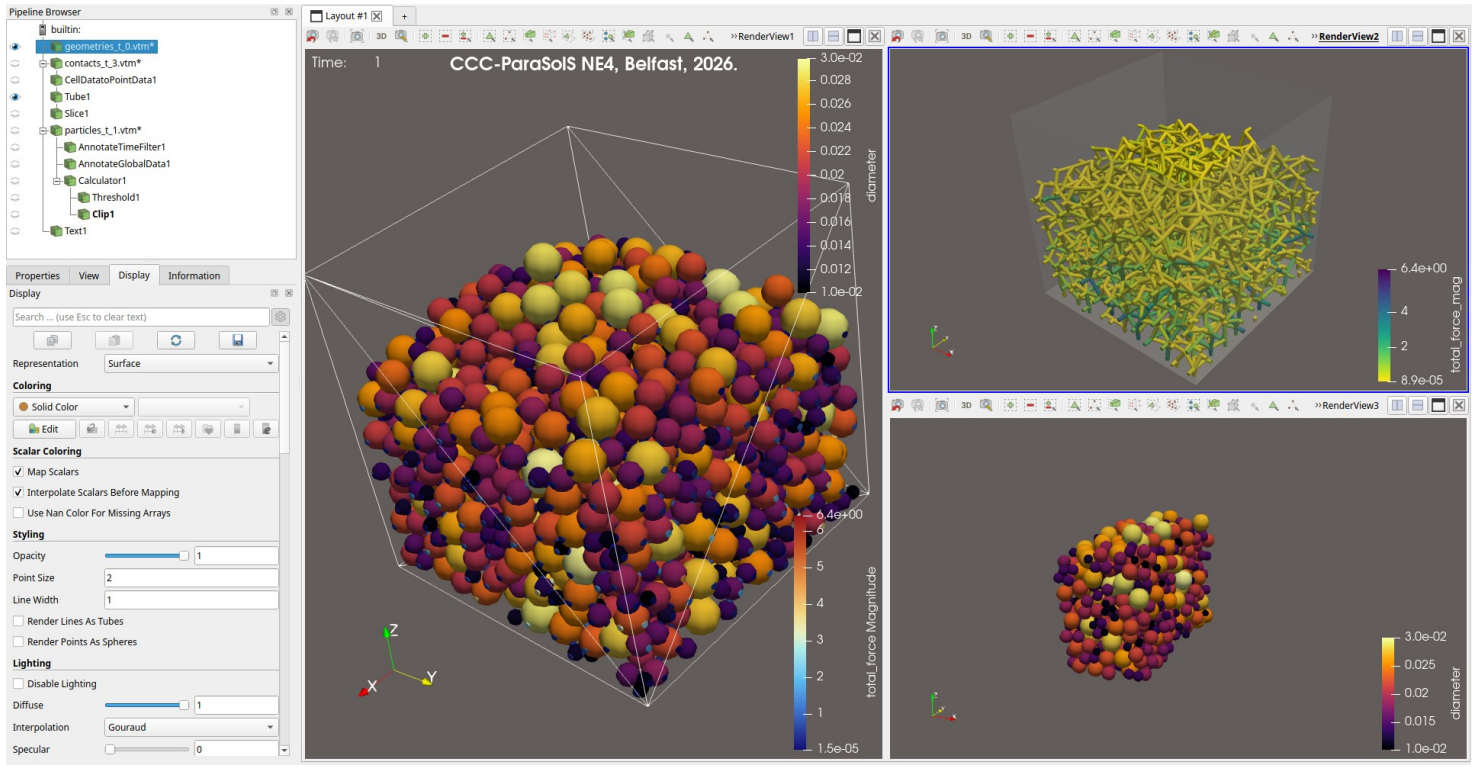
Multiple Viewports

- ParaView allow you to split your screen into many different viewports, allowing you to visualise different views or sources in each one



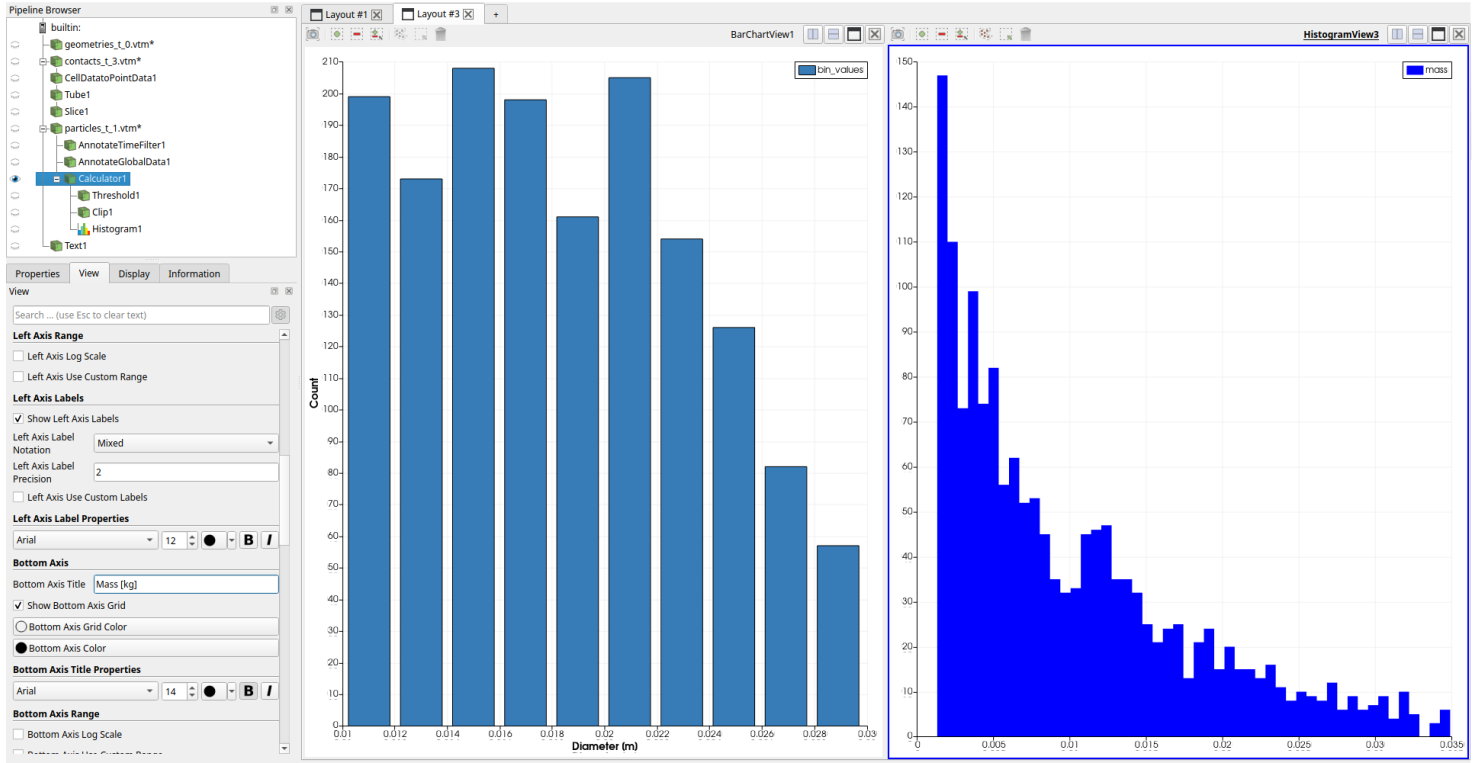
- You can link the viewports to have the same view if you wish





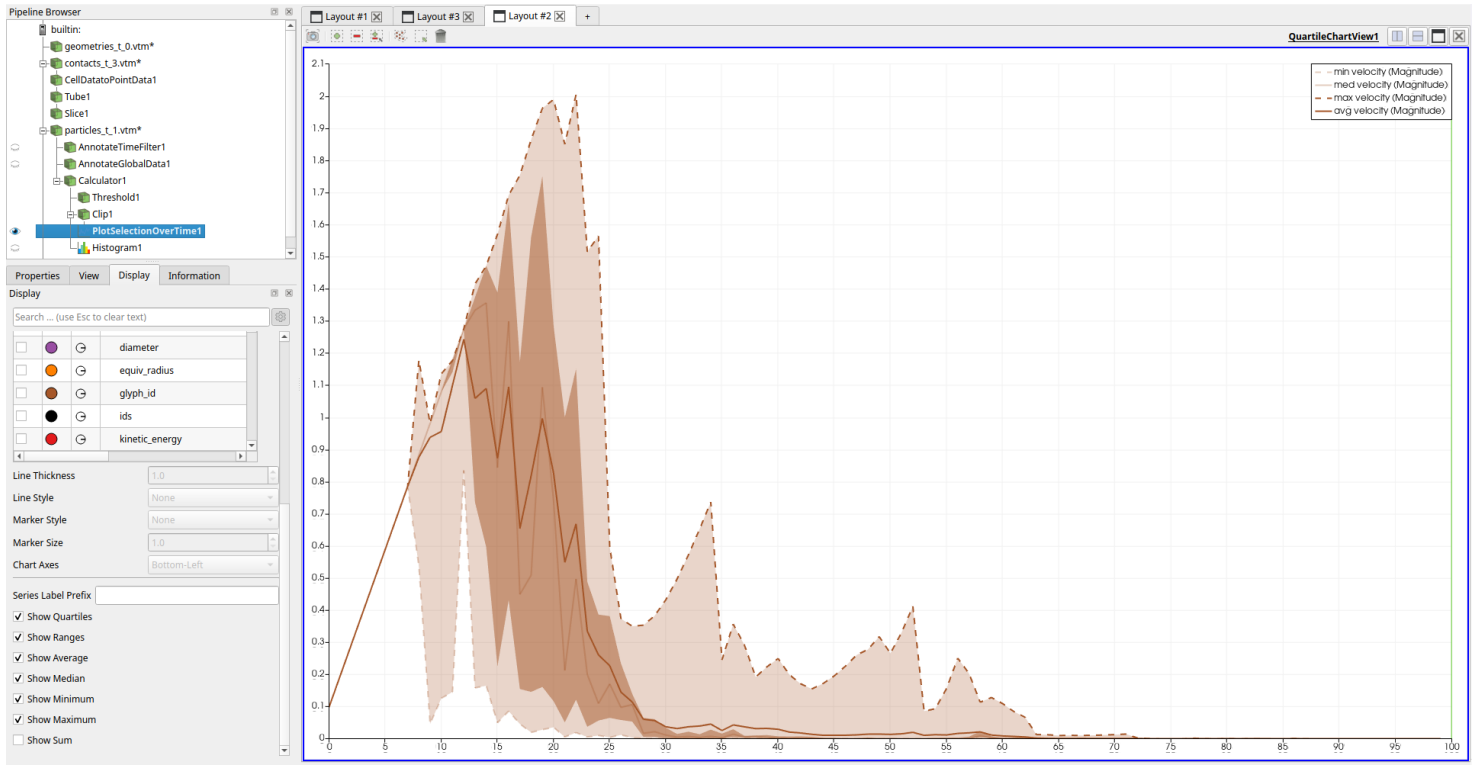
Plotting

- ParaView has support for creating plot of many different types:
  - **Histograms**
    - Filter
    - Plot View
  - Plot over line
  - Plot over time



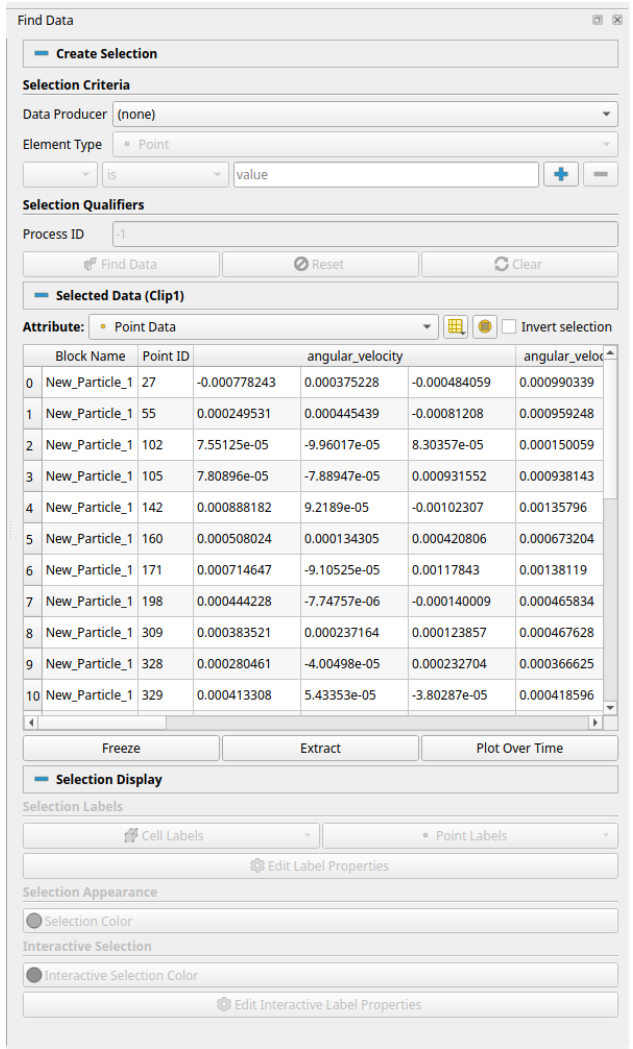
Plotting

- ParaView has support for creating plot of many different types:
  - Histograms
  - Plot over line
  - **Plot over time**



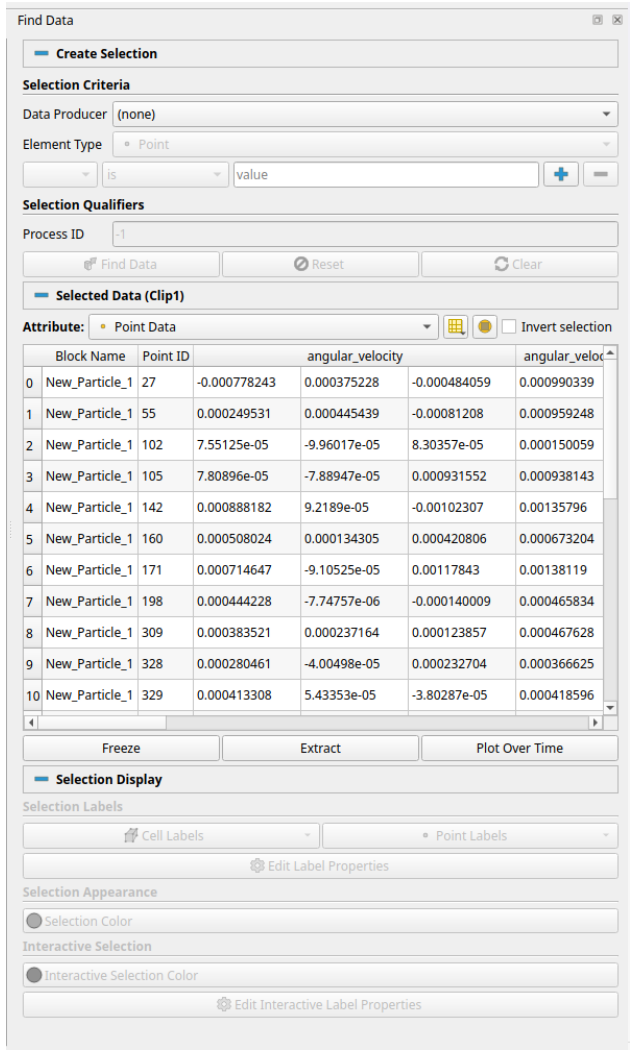
Find Data

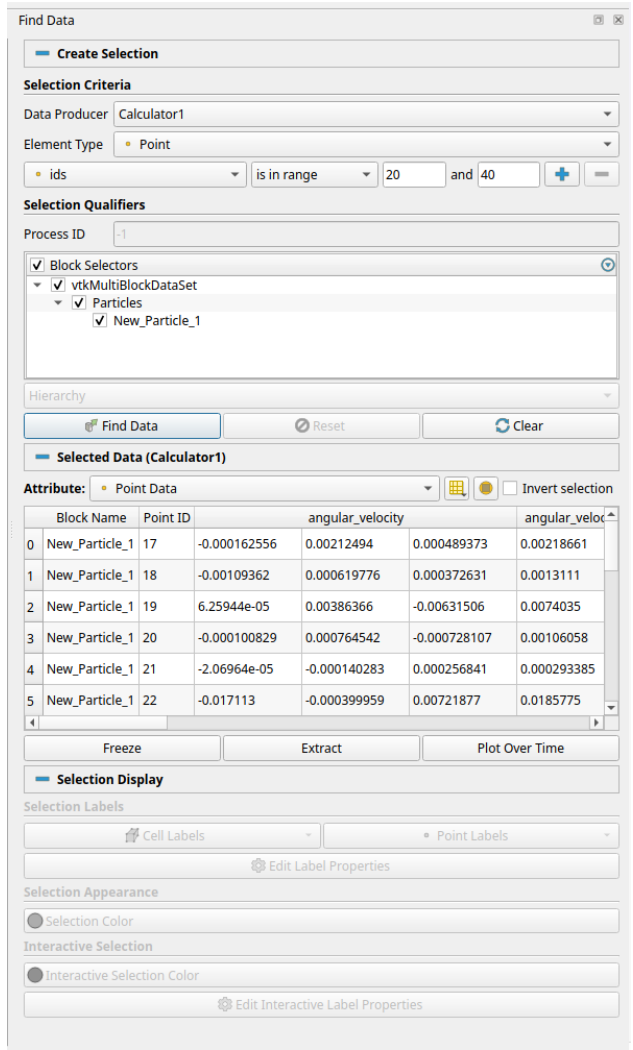
- A tool that allows you to make point/cell selections programmatically
- Access from the toolbar options or the keyboard shortcut (v)
- **Exercise:** Select particles with IDs 20-40 and extract as new block
  - Also Plot Over Time



Find Data

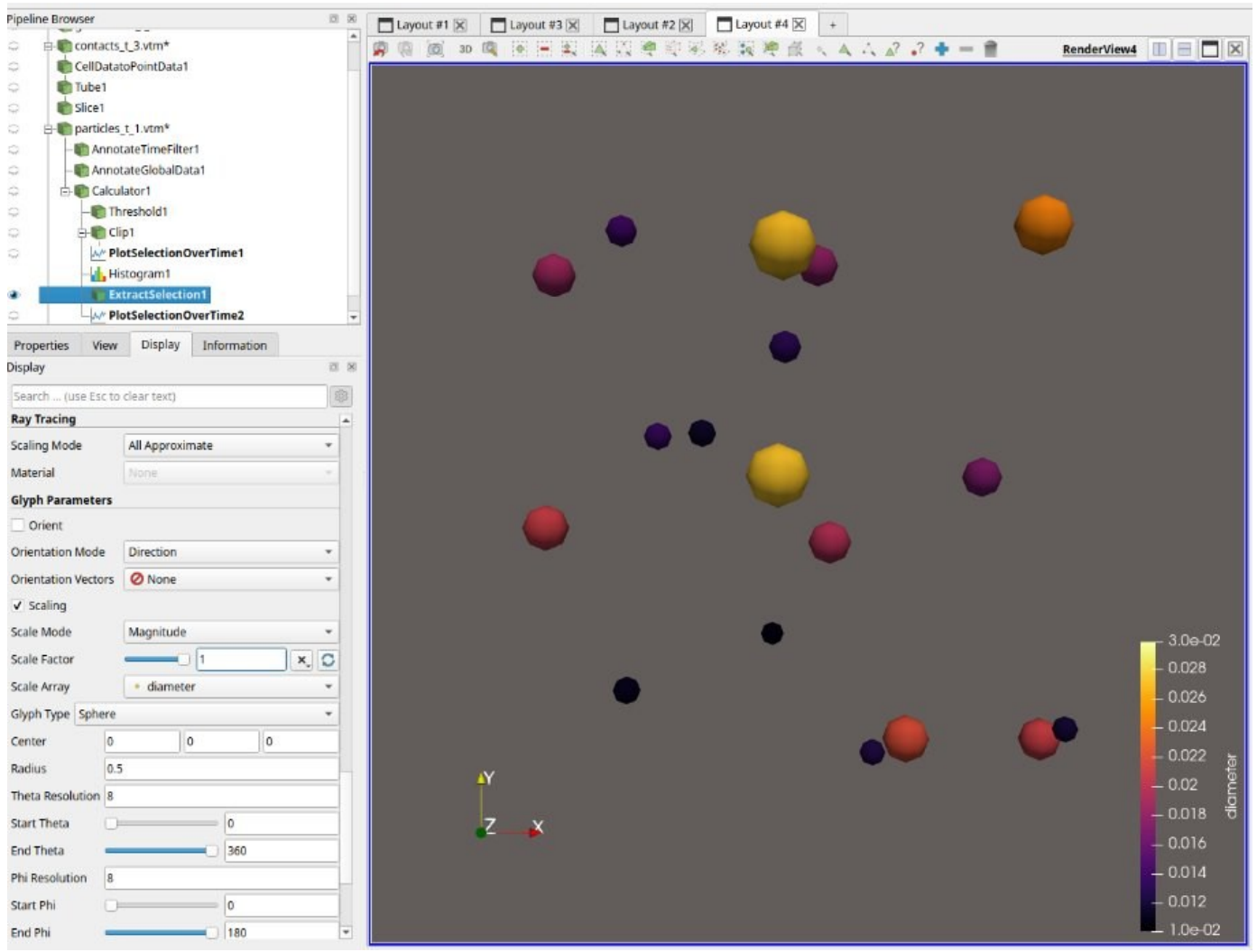
- A tool that allows you to make point/cell selections programmatically
- Access from the toolbar options or the keyboard shortcut (v)
- **Exercise:** Select particles with IDs 20-40 and extract as new block
  - Also Plot Over Time





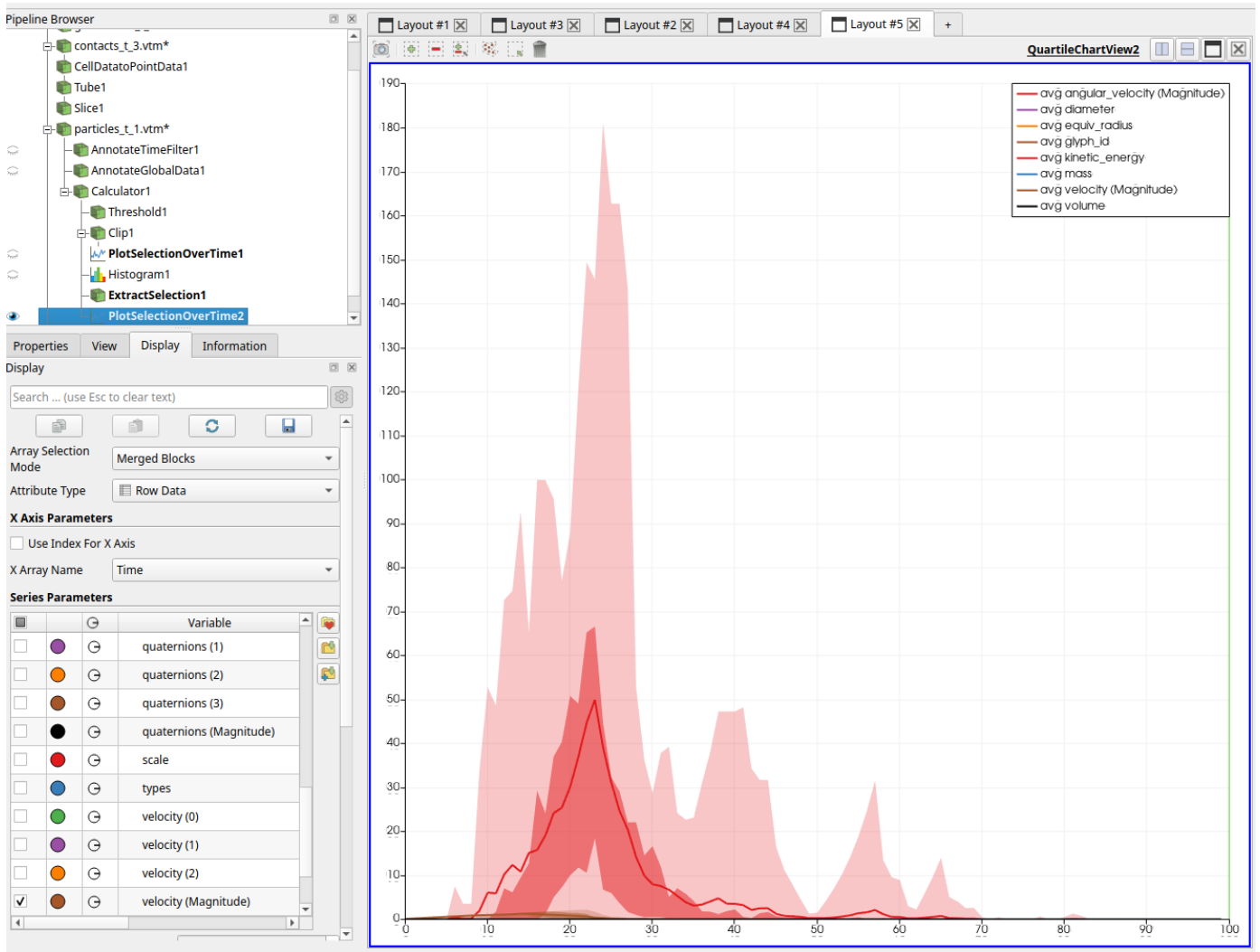
Find Data – Extract Block

- A tool that allows you to make point/cell selections programmatically
- Access from the toolbar options or the keyboard shortcut (v)
- **Exercise:** select particles with IDs 20-40 and extract as new block
  - Also Plot Over Time



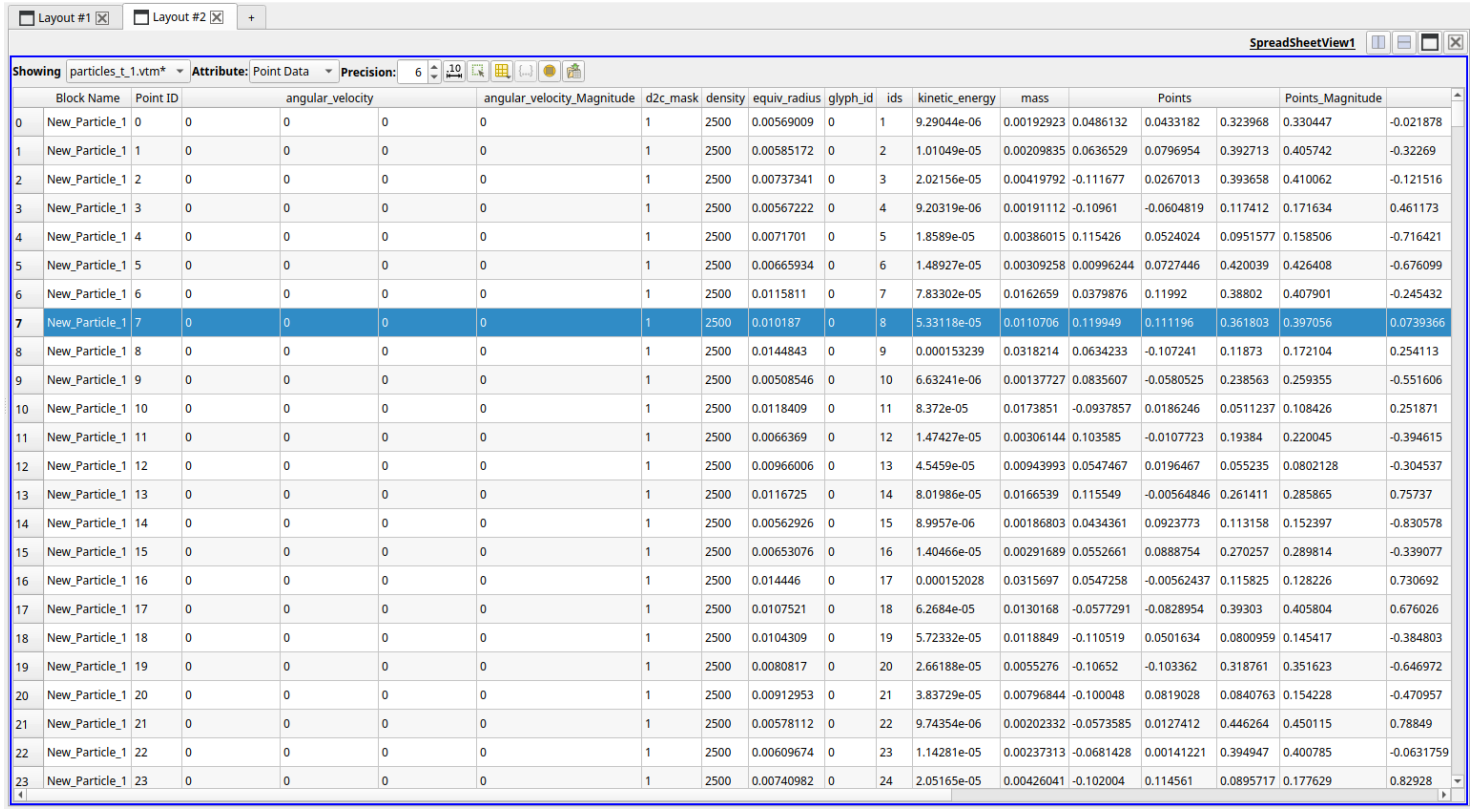
Find Data – Plot Over Time

- You can load any plot filter into a new layout
- In the display settings you can:
  - switch on/off variables
  - Add avg, min, max and error data



Spreadsheet View

- A useful tool for exploring data in your dataset
  - Allows simple filtering
  - Can be used to interrogate or isolate data
- Can be used to export data to spreadsheet



Layout #1 | Layout #2 | +

SpreadSheetView1

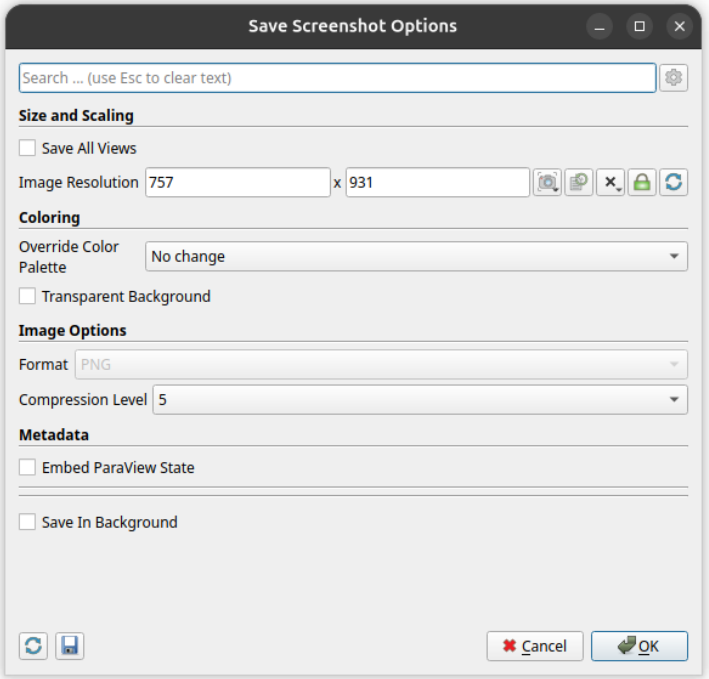
Showing particles_t_1.vtm* Attribute: Point Data Precision: 6

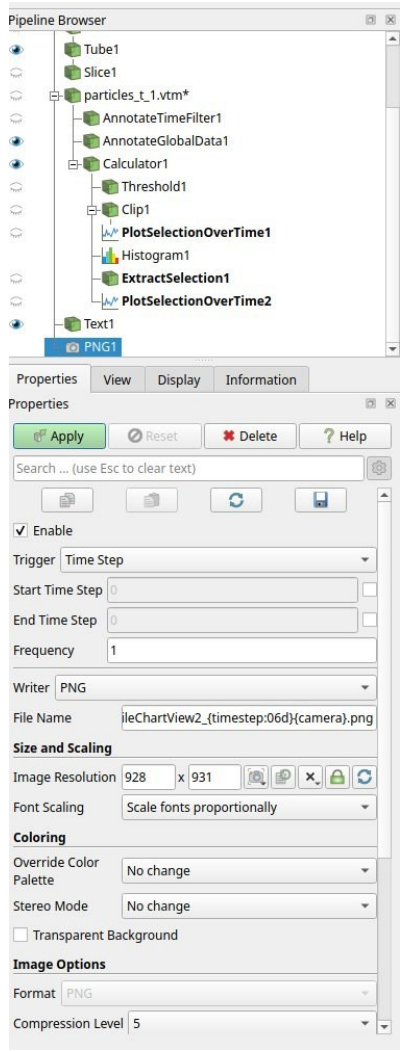
	Block Name	Point ID		angular_velocity			angular_velocity_Magnitude	d2c_mask	density	equiv_radius	glyph_id	ids	kinetic_energy	mass		Points		Points_Magnitude	
0	New_Particle_1	0	0	0	0	0	1	2500	0.00569009	0	1	9.29044e-06	0.00192923	0.0486132	0.0433182	0.323968	0.330447	-0.021878
1	New_Particle_1	1	0	0	0	0	1	2500	0.00585172	0	2	1.01049e-05	0.00209835	0.0636529	0.0796954	0.392713	0.405742	-0.32269
2	New_Particle_1	2	0	0	0	0	1	2500	0.00737341	0	3	2.02156e-05	0.00419792	-0.111677	0.0267013	0.393658	0.410062	-0.121516
3	New_Particle_1	3	0	0	0	0	1	2500	0.00567222	0	4	9.20319e-06	0.00191112	-0.10961	-0.0604819	0.117412	0.171634	0.461173
4	New_Particle_1	4	0	0	0	0	1	2500	0.0071701	0	5	1.8589e-05	0.00386015	0.115426	0.0524024	0.0951577	0.158506	-0.716421
5	New_Particle_1	5	0	0	0	0	1	2500	0.00665934	0	6	1.48927e-05	0.00309258	0.00996244	0.0727446	0.420039	0.426408	-0.676099
6	New_Particle_1	6	0	0	0	0	1	2500	0.0115811	0	7	7.83302e-05	0.0162659	0.0379876	0.11992	0.38802	0.407901	-0.245432
7	New_Particle_1	7	0	0	0	0	1	2500	0.010187	0	8	5.33118e-05	0.0110706	0.119949	0.111196	0.361803	0.397056	0.0739366
8	New_Particle_1	8	0	0	0	0	1	2500	0.0144843	0	9	0.000153239	0.0318214	0.0634233	-0.107241	0.11873	0.172104	0.254113
9	New_Particle_1	9	0	0	0	0	1	2500	0.00508546	0	10	6.63241e-06	0.00137727	0.0835607	-0.0580525	0.238563	0.259355	-0.551606
10	New_Particle_1	10	0	0	0	0	1	2500	0.0118409	0	11	8.372e-05	0.0173851	-0.0937857	0.0186246	0.0511237	0.108426	0.251871
11	New_Particle_1	11	0	0	0	0	1	2500	0.0066369	0	12	1.47427e-05	0.00306144	0.103585	-0.0107723	0.19384	0.220045	-0.394615
12	New_Particle_1	12	0	0	0	0	1	2500	0.00966006	0	13	4.5459e-05	0.00943993	0.0547467	0.0196467	0.055235	0.0802128	-0.304537
13	New_Particle_1	13	0	0	0	0	1	2500	0.0116725	0	14	8.01986e-05	0.0166539	0.115549	-0.00564846	0.261411	0.285865	0.75737
14	New_Particle_1	14	0	0	0	0	1	2500	0.00562926	0	15	8.9957e-06	0.00186803	0.0434361	0.0923773	0.113158	0.152397	-0.830578
15	New_Particle_1	15	0	0	0	0	1	2500	0.00653076	0	16	1.40466e-05	0.00291689	0.0552661	0.0888754	0.270257	0.289814	-0.339077
16	New_Particle_1	16	0	0	0	0	1	2500	0.014446	0	17	0.000152028	0.0315697	0.0547258	-0.00562437	0.115825	0.128226	0.730692
17	New_Particle_1	17	0	0	0	0	1	2500	0.0107521	0	18	6.2684e-05	0.0130168	-0.0577291	-0.0828954	0.39303	0.405804	0.676026
18	New_Particle_1	18	0	0	0	0	1	2500	0.0104309	0	19	5.72332e-05	0.0118849	-0.110519	0.0501634	0.0800959	0.145417	-0.384803
19	New_Particle_1	19	0	0	0	0	1	2500	0.0080817	0	20	2.66188e-05	0.0055276	-0.10652	-0.103362	0.318761	0.351623	-0.646972
20	New_Particle_1	20	0	0	0	0	1	2500	0.00912953	0	21	3.83729e-05	0.00796844	-0.100048	0.0819028	0.0840763	0.154228	-0.470957
21	New_Particle_1	21	0	0	0	0	1	2500	0.00578112	0	22	9.74354e-06	0.00202332	-0.0573585	0.0127412	0.446264	0.450115	0.78849
22	New_Particle_1	22	0	0	0	0	1	2500	0.00609674	0	23	1.14281e-05	0.00237313	-0.0681428	0.00141221	0.394947	0.400785	-0.0631759
23	New_Particle_1	23	0	0	0	0	1	2500	0.00740982	0	24	2.05165e-05	0.00426041	-0.102004	0.114561	0.0895717	0.177629	0.82928

Image Export

- ParaView supports video and image export a number of ways

- Images:
  - Save screenshot tool
  - Image extractors which are applied in the pipeline on trigger criterion.
  - Particularly useful for creating series of images from animation

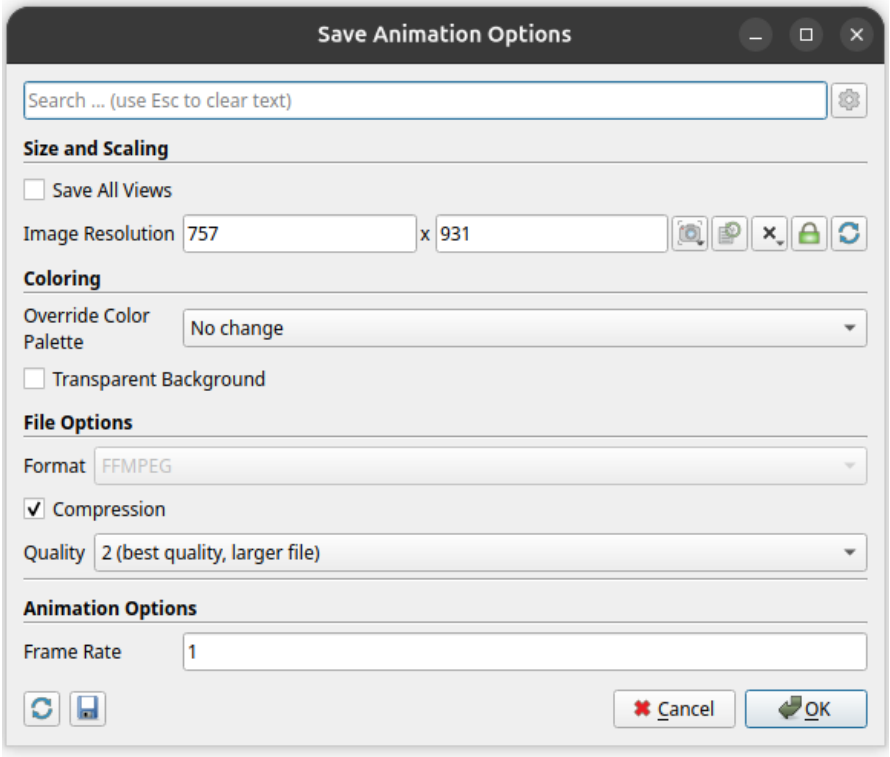


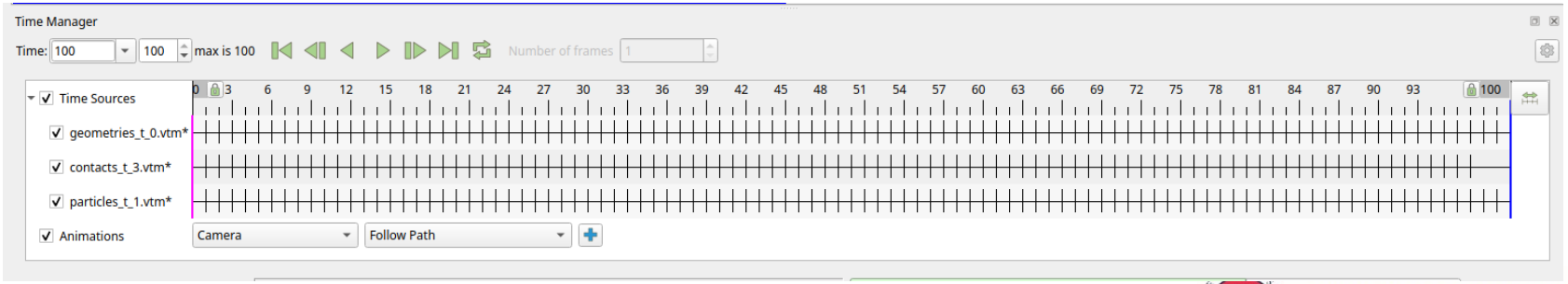


Video Export

- ParaView supports video and image export a number of ways

- Video:
  - Save Animation tool
  - Save Extracts
  - Can be used in conjunction with Time Manager and camera settings to create complex animations





Additional ParaView Features

Automating Workflows With Python

- ParaView comes with its own Python interpreter pvpython
  - This is an interactive python shell that leverages the VTK backend via interactive and reproducible scripts
  - Ideal for mass automation
  - Couple with pvbatch which is a self-contained MPI server

	pvpython	pvbatch	Catalyst
Execution Mode	Interactive serial	Parallel batch	In-situ Co-Processing
Primary Use Case	Scripting & Pipeline prototyping	Massive Post-Processing	Zero-I/O Run-time Analysis
Launch Mechanism	Pvpython script.py	Mpirun pvbatch	Compiled with Simulation
Disk I/O Dependency	High (reads raw data)	High (reads raw data)	Minimal/Zero (reads from RAM)

Automating Repetitive Tasks

- ParaView allows you to record your workflows using a trace tool and the macro tool
  - **Traces** are hardcoded to specific sources and filters
  - **Macros** are generalised using script functions such as GetActiveSource()

- Both can be helpful ways of reducing the amount of repetitive tasks

Summary

- Learned about common visualtion mistakes that make interpretation of figures difficult
- Learned some colour theory and how to choose an appropriate colour map for your data
- Learned about the Visualisation Tool Kit and the various supported file formats
- Learned about the VTK pipeline and filters
- Learned some basic ParaView usage

Questions?

ParaSolS

Particulate Solids Simulations

parasols@ed.ac.uk

https://www.ccc-parasols.ed.ac.uk/

Discourse https://parasols.discourse.group/

https://www.linkedin.com/company/ccc-parasols/