Towards a Progressive Open Source Framework for SciVis and InfoVis

Gueunet Charles, Mazen Francois

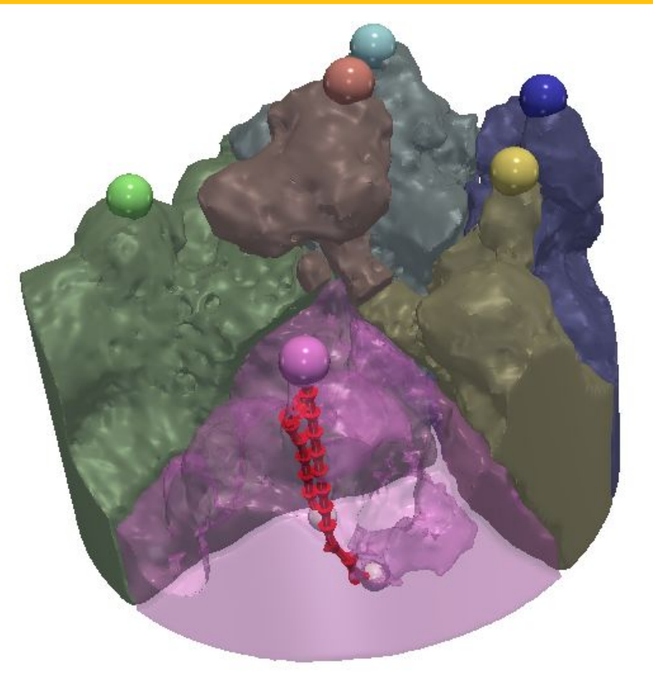
Kitware Europe

PDAV VIS2024

Scientific Visualization

- Focus on 2D/3D geometric data
- Analysis of massive simulation / acquisition

- **VTK / ParaView:** Leading **open-source** software
  - Widely used in HPC, large active community
  - Client-server architecture, with **distributed** support
  - **Demand driven**, lazy evaluated pipeline
  - **Interactive** selection

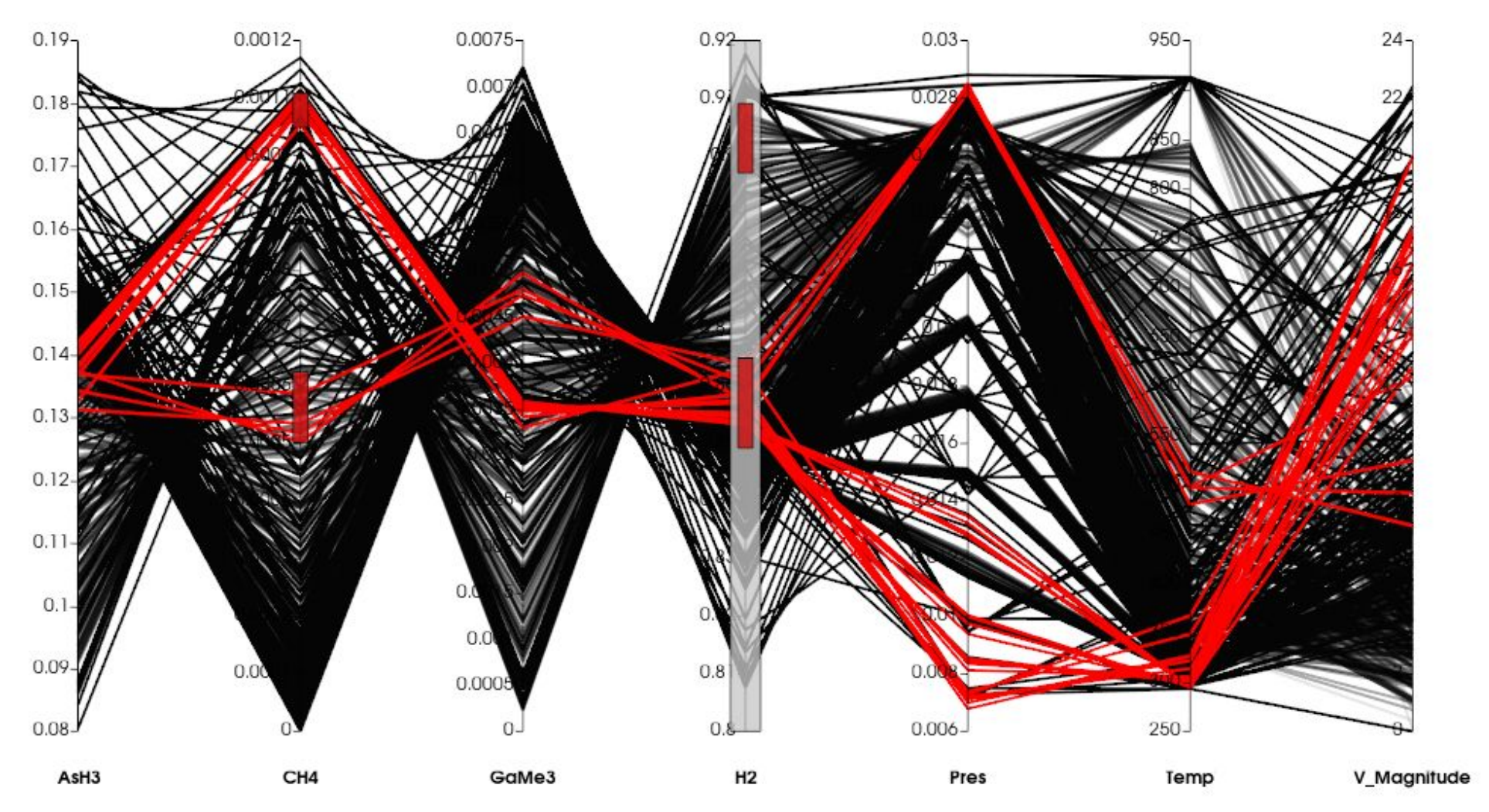


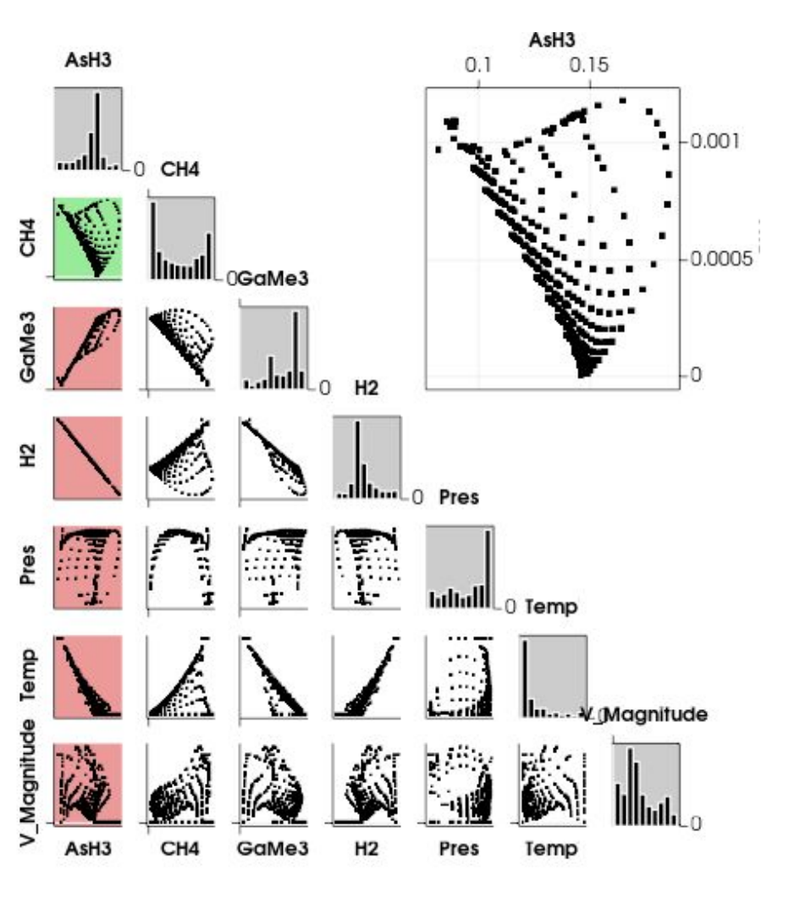
ParaView

Information Visualization

- Focus on **abstract** data analysis
- Special care for **human perception** and interaction

- Various frameworks
  - Allow for the creation of **tailored** analysis
  - Range from small scripts to rich (web) applications
  - Make use of **grammars** to build specific widgets + interactions





Progressive Visualisation

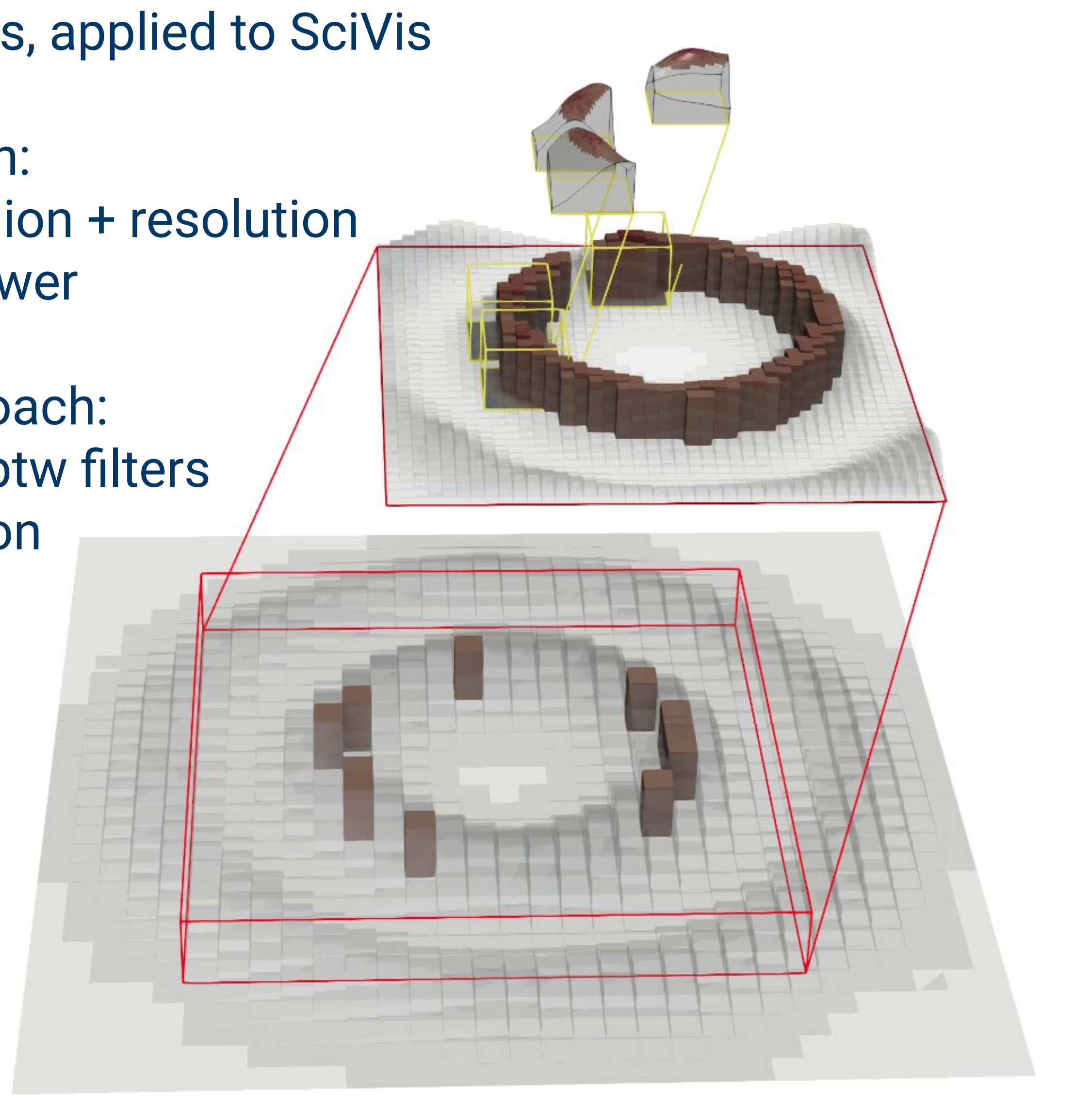
Description:

Progressive analytics, applied to SciVis

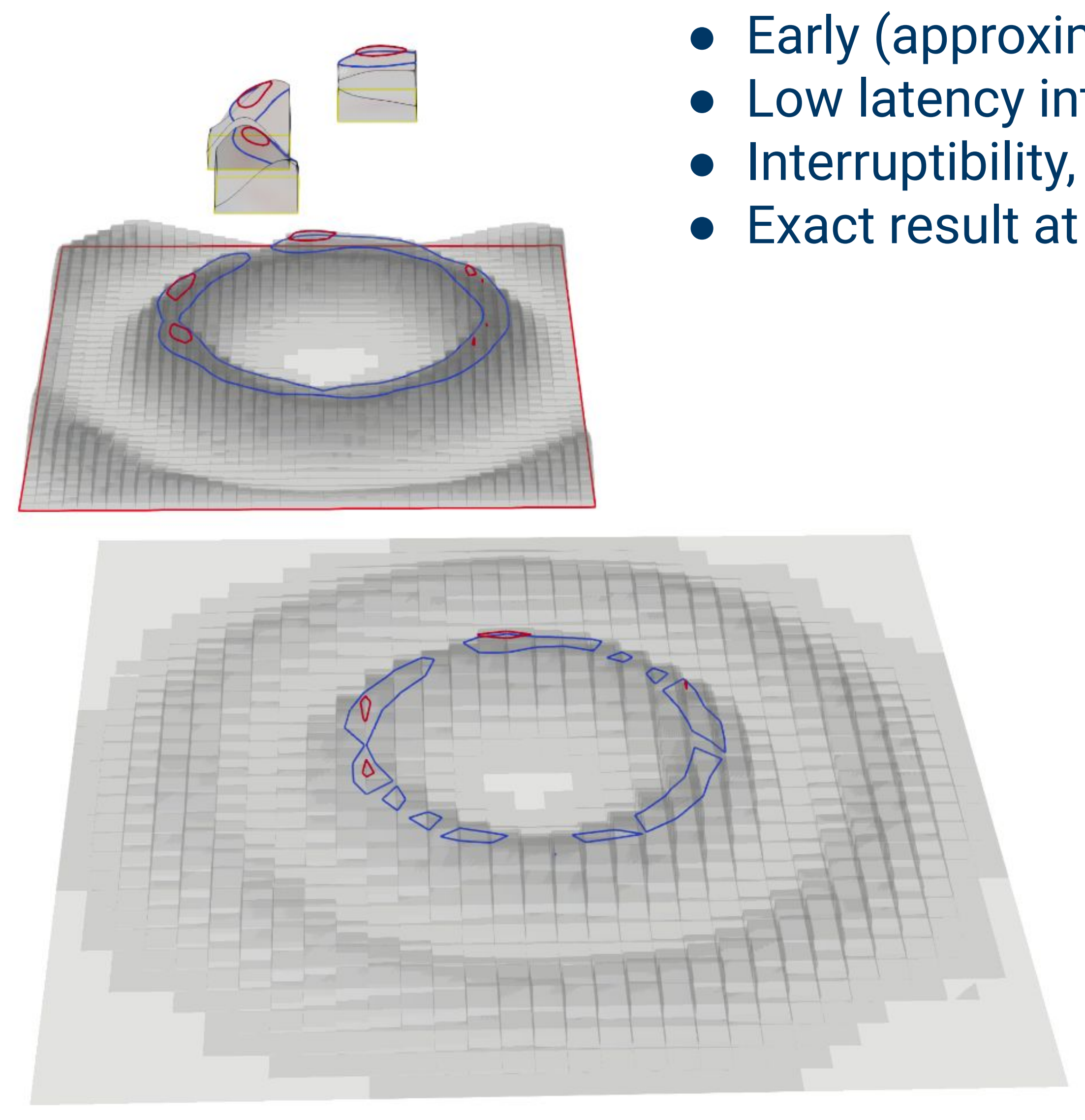
Multi-scale approach:
- Filters query: region + resolution
- Data model: answer

Asynchronous approach:
- Remove barrier btw filters
- Filters interruption
- Stitch results
- Event loop

Guarantees:
- Early (approximate) result,
- Low latency interactivity,
- Interruptibility,
- Exact result at the end



Multi-scale resolution of a Gaussian wave dataset, several chunks are shown



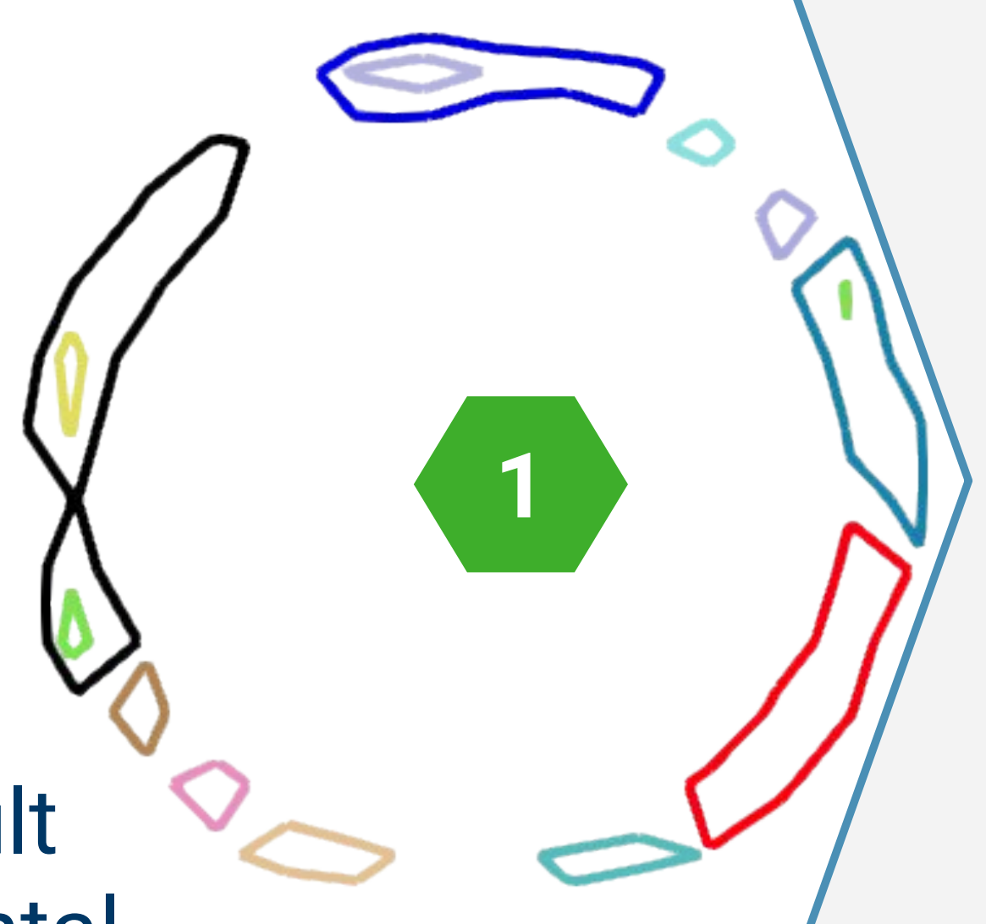
Result of a contour extraction on the given chunks, using 2 isovalues

Initialization

Global decimation

Multiscale approach,
Computation on a **decimated** input.

- Provide an insight
- Quickly available result
- Initialize the incremental computation

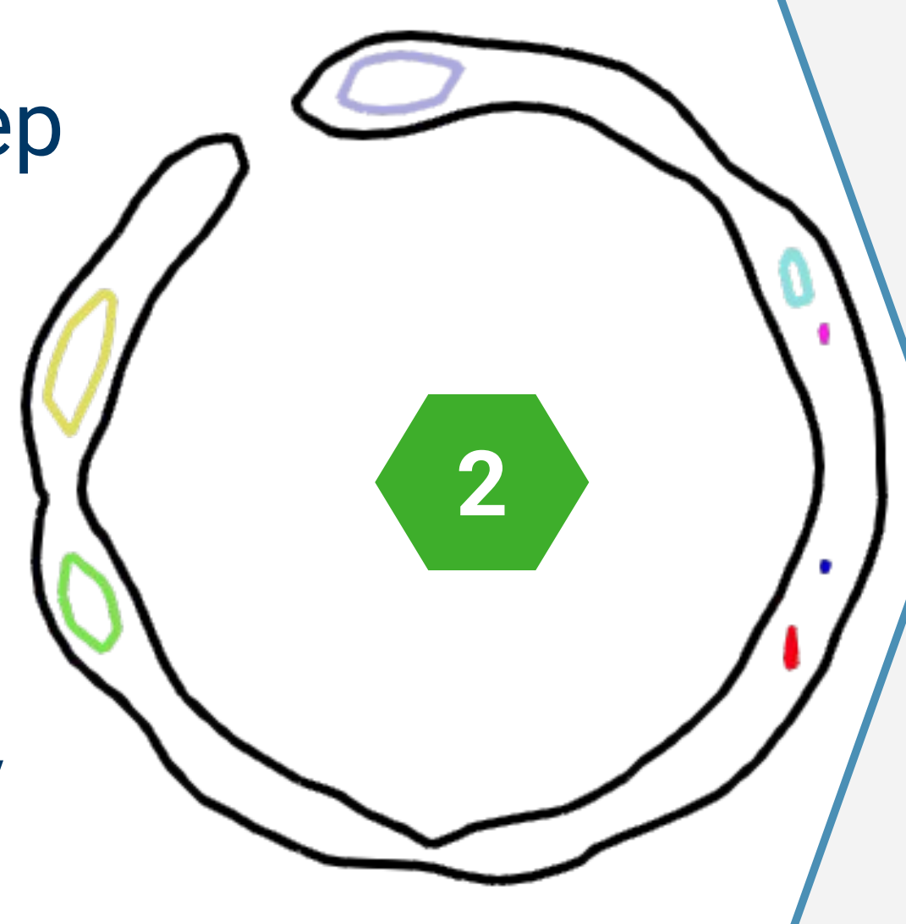


Decimation

Incremental refinement

Refine from previous step
Discard useless area
Converge

- Optimizations:
  - re-use last result
  - replace when ready
- Problem
  - input size increases over quota

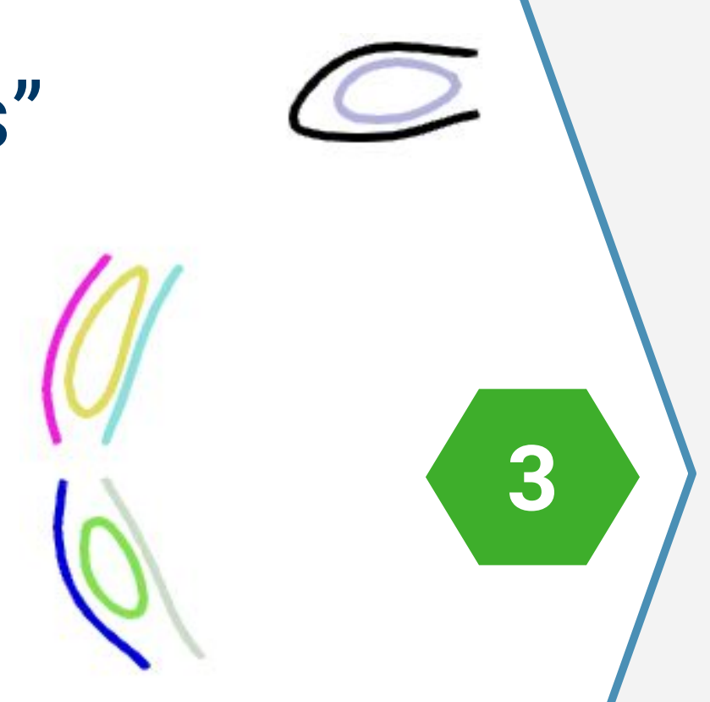


Chunk extraction

Local refinement

Exact solution on local "chunks"
Need **small** enough chunks
Re-use previous results

- How to **stitch** results ?
- How to **prioritize** chunks?
- Detect when final result is reached



Issues to address:

Data model
The VTK data model already supports most of the important features required for progressive visualization, but generalization is needed:
The ability to provide small refined chunks of data on demand is already implemented in the **vtkOverlappingAMR** class.
Generic adaptive decimation is already available using interactive rendering (relying on **vtkLODActor** and **vtkDecimateFilter**).

Executive
If respecting a **time constraint** in interactive rendering is currently supported, it only relies on a global decimation.
In the context of progressive analysis, a new pipeline executive leveraging a **query responder** architecture is needed.
It will be responsible for querying small enough data to ensure the time constraint is respected.

Rendering
Currently, the rendering is always started from scratch when the data is modified.
In the context of progressive visualisation, we will **measure** the impact of the rendering and eventually explore **partial update** techniques.
Other points are detailed in the PDAV article, for example most charts rendering is currently not distributed.