baspl++

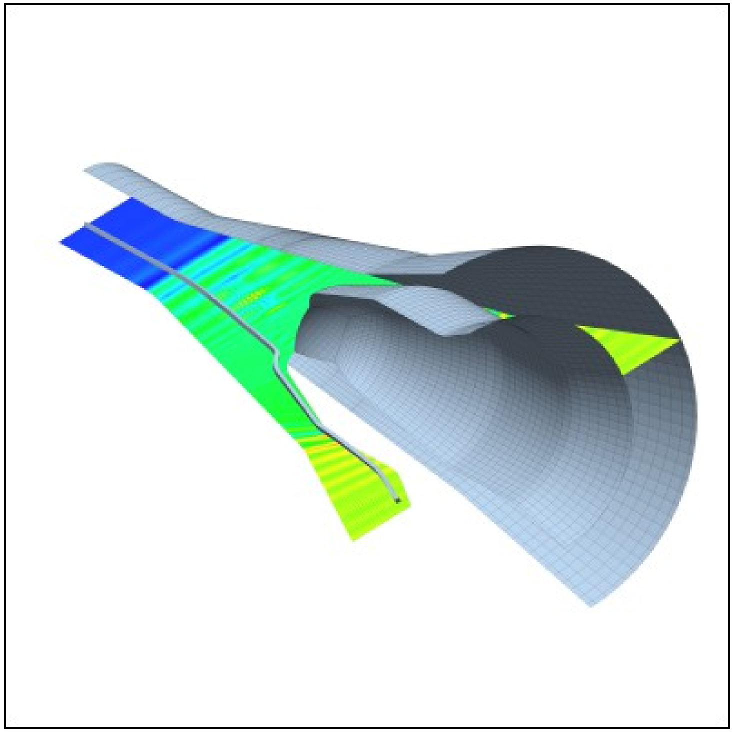
A Viewing and Post-Processing Tool

SMR Engineering & Development

CH-2500 Bienne

http://www.smr.ch

Contents



- Why another post-processor?
- Concepts
- Features
- Importers
- Examples
- Technology

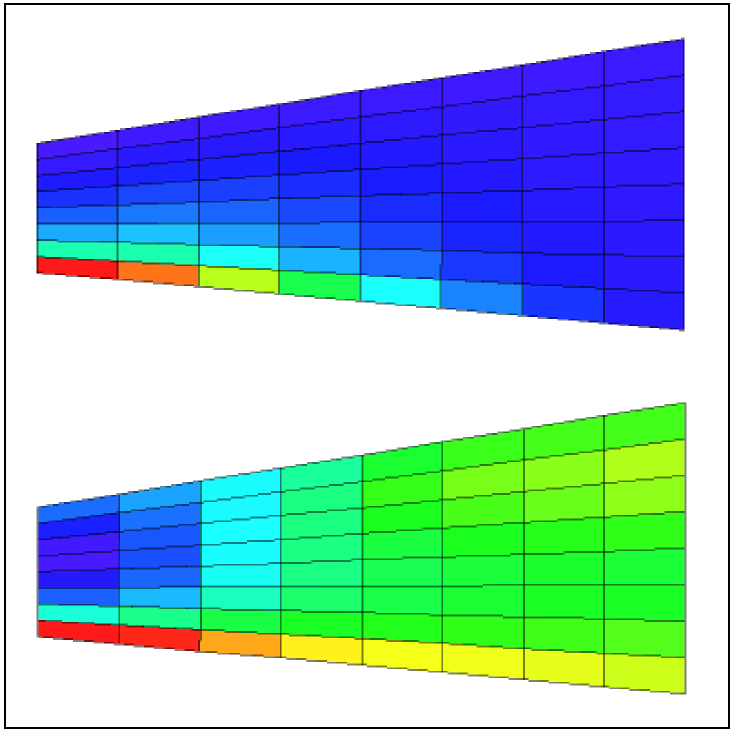
- See also
  - *baspl*++ documentation
  - *B2000*++ documentation

Why another post-processor ?

- baspl++ is the logical successor of baspl (1986) a viewer optimized for SGI platforms.
- baspl++ is efficient for large models.
- baspl++ is steerable with a scripting language (for configuration and customization, and for batch processing).
- baspl++ fits in the B2000++ concept and supports the B2000++ database.
- baspl++ runs on open source-based systems.

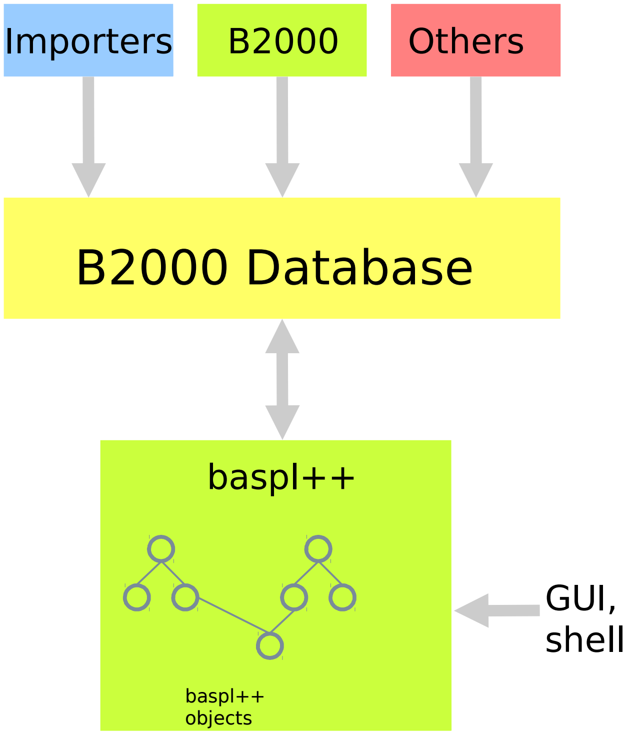
Motto: Keep it as simple as possible, while preserving B2000++ specific features like multi-block hybrid meshes and multi-model, multi-physics processing and rendering.

Concepts



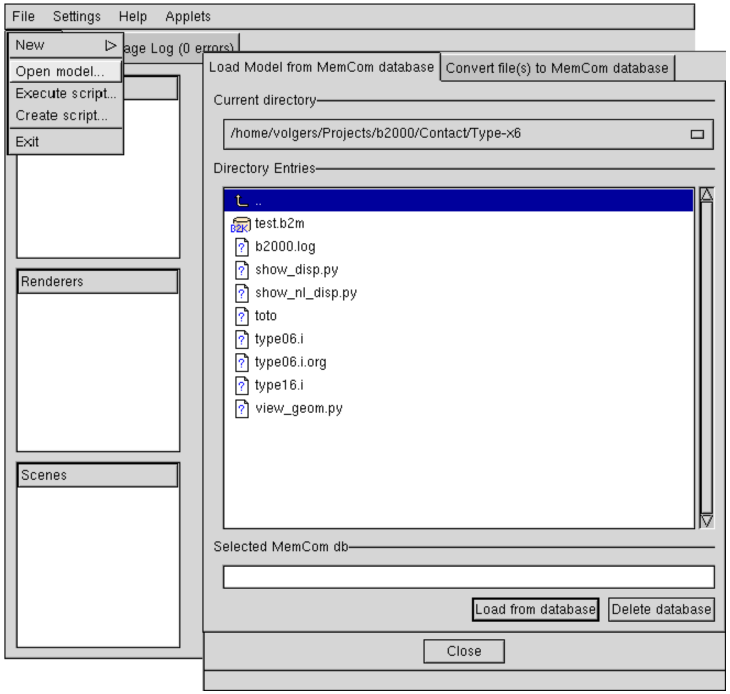
- Modular approach allows for:
  - Multiple models
  - Various scenes
  - Separate parts
  - Different solution fields
- Interface through:
  - Graphical user interface (GUI)
  - Command line interface (CLI)
  - Scripts for automation of tasks
- Integration of numerical data
  - Extraction
  - Manipulation
  - Visualisation

Concepts



- baspl++ works with a database
  - Consistent data representation
  - Very fast access to disk
  - Can condense or compressed data
  - Virtually unlimited processing capabilities with pymemcom, numpy
- baspl++ works with a small number of objects, i.e. a small number of visible classes
  - Easy to learn, being it with the GUI or with scripting.
  - Same concept with GUI and with scripting.

Concepts: Model

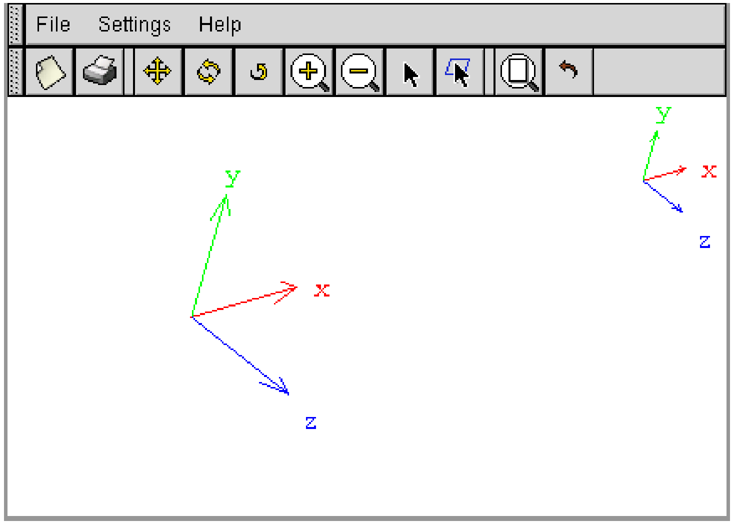


/home/volgers/Projects/b2000/Contact/Type-x6

```
..
test.b2m
b2000.log
show_disp.py
show_nl_disp.py
toto
type06.i
type06.i.org
type16.i
view_geom.py
```

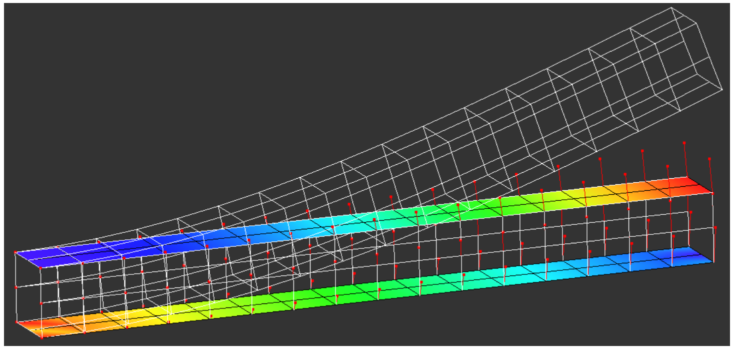
- A baspl++ Model object is the geometrical and numerical data stored on a data base.
- One or more models can be opened simultaneously
- Various formats can be imported directly or through filters:
  - B2000/NSMB (MemCom)
  - Tau - NetCDF
  - TecPlot ASCII
  - Nastran (*)
  - Medina (*)
  - Custom filters
- (*) Subsets

Concepts: Scene



- A baspl++ Scene object is a window for displaying numerical data.
- One or more scenes can be created at any given moment
- Scene functionality:
  - Rotate, translate, zoom
  - Picking
  - Customisable background
  - Print
  - Undo

Concepts: Part



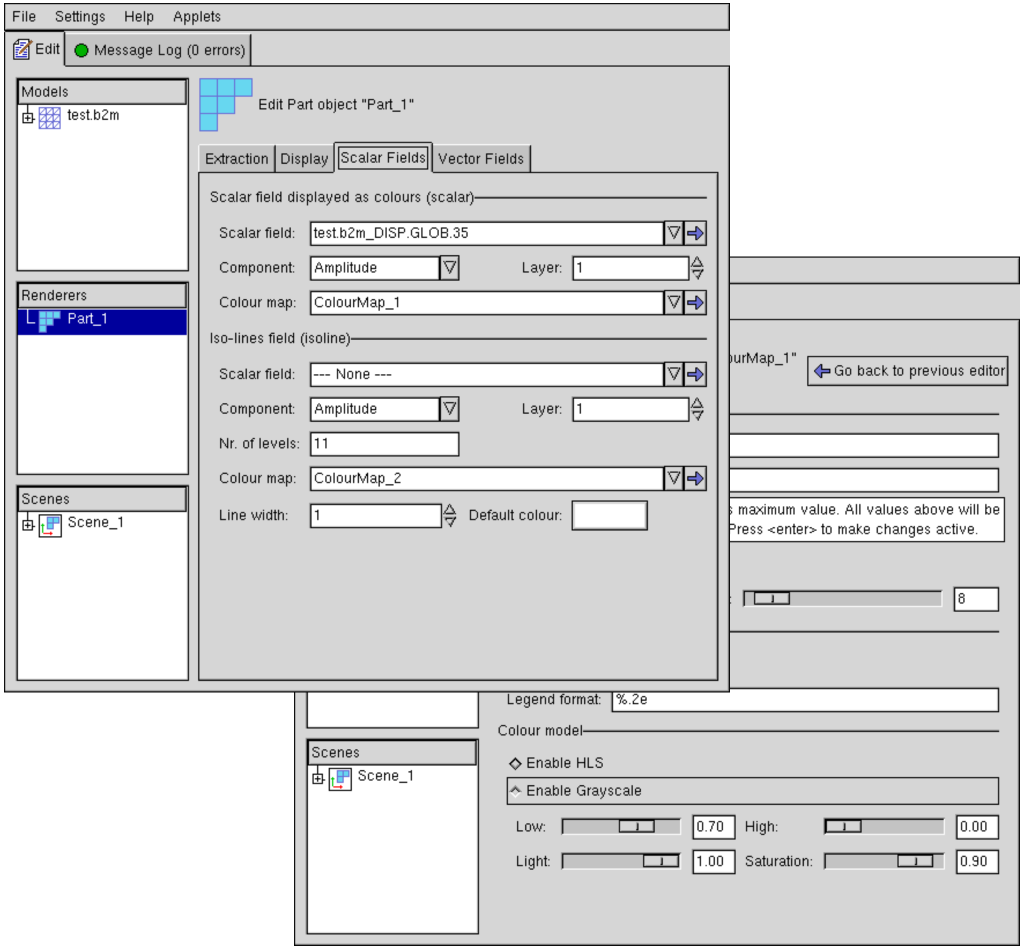
- A baspl++ *Part* object is a selection of elements or nodes from the Model to be displayed.
- One or more Parts can be created per Model.
- Different Parts can be displayed in the same or in different Scenes.
- Solutions and display methods are applied to a Part.

Features

- Element selection by
  - Number
  - Type
  - Group
  - Subdomain
  - Cuts
  - Outline or surface
  - Points
- Geometry
  - Outline
  - Wireframe
  - Solid mesh

- Solution field display
  - Deformed geometry
  - Nodal vectors
  - Colour field mapped onto geometry
  - Isolines
  - All of the above simultaneously
- Colour map options
  - Smooth interpolation
  - Texture or step-texture mapping
  - Fixed or automatic min/max settings

Features: GUI



Edit Part object "Part_1"

Scalar field displayed as colours (scalar)

Scalar field: test.b2m_DISP.GLOB.35

Component: Amplitude   Layer: 1

Colour map: ColourMap_1

Iso-lines field (isoline)

Scalar field: --- None ---

Component: Amplitude   Layer: 1

Nr. of levels: 11

Colour map: ColourMap_2

Line width: 1   Default colour:

urMap_1"

s maximum value. All values above will be
Press <enter> to make changes active.

8

Legend format: %.2e

Colour model

◇ Enable HLS

Enable Grayscale

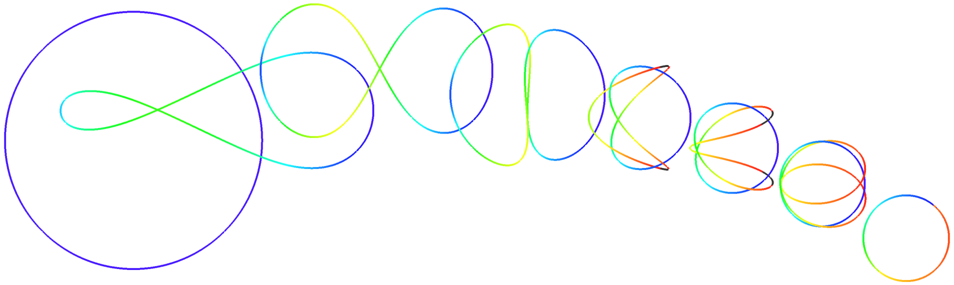
Low: 0.70   High: 0.00

Light: 1.00   Saturation: 0.90

- All data extraction and display functionality available in an integrated GUI.
- Back and forward buttons for easy switching between menus.
- Tree structured overview of object relationships.
- Automatic pop-up help windows available

SMR Engineering & Development

Features: CLI and Scripts

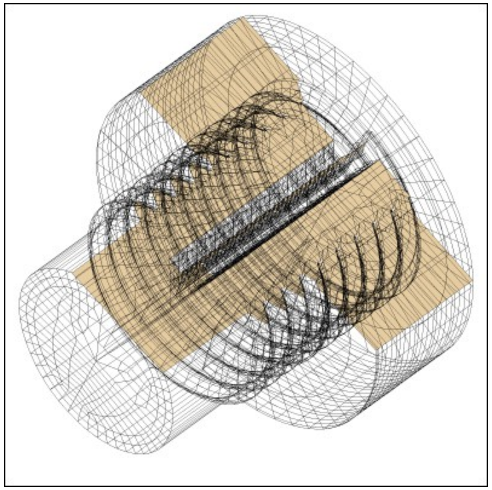


```
# View folding process of thin ring

for i in range (1,ncycles):
    f=Field(m,'DISP', cycle=i, case=1)
    p.set_displacement_field(f)
    p.set_scalar_field(f, component='Amplitude')
    time.sleep(0.1)
```

- Python scripting language.
- CLI (Command String Interpreter) allows for data manipulation.
- Python control loops for animation.
- GUI and CLI commands can be freely mixed.
- create_script() command allows for automatic generation of script reproducing CLI and GUI input.
- Scripts greatly facilitate repetitive tasks.

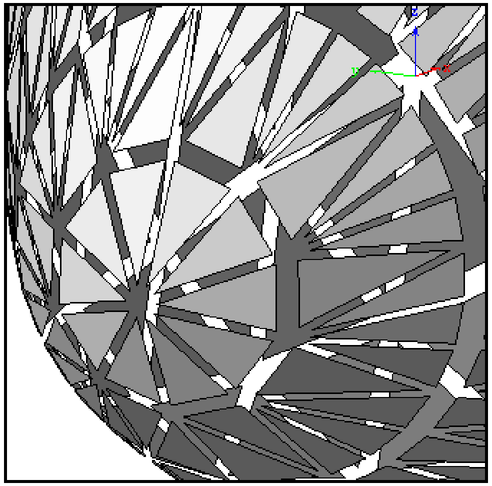
Features: Display Geometry



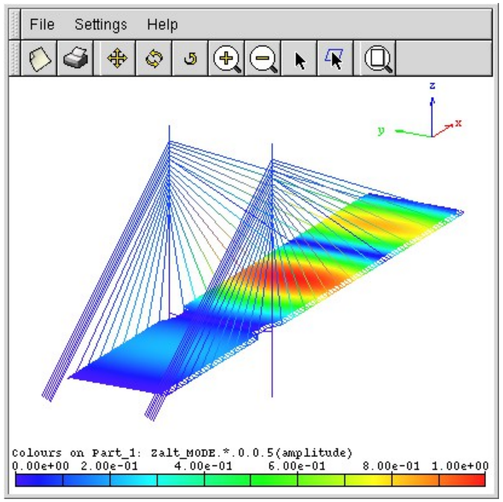
Wireframe (outline) and solid mesh (cut)

Solid with light source

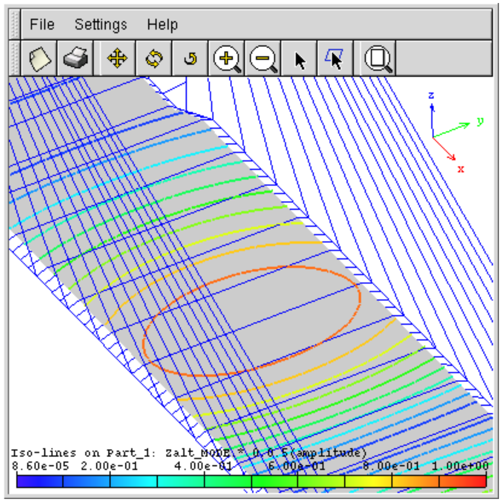
Solid mesh elements in imploded view



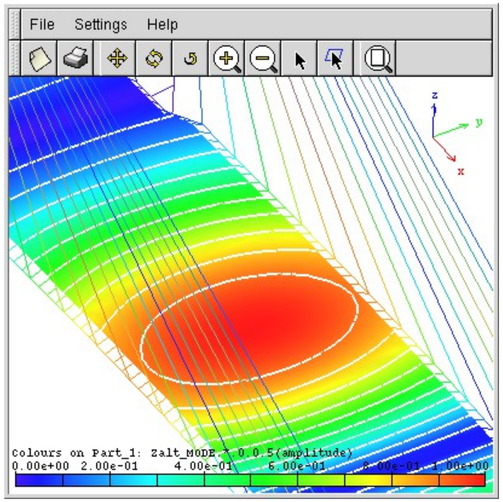
Features: Display Scalar Fields


File Settings Help
z
y
x
Colours on Part_1: Zalt_MODE.*.0.0.5(amplitude)
0.00e+00 2.00e-01 4.00e-01 6.00e-01 8.00e-01 1.00e+00

Scalar field colours

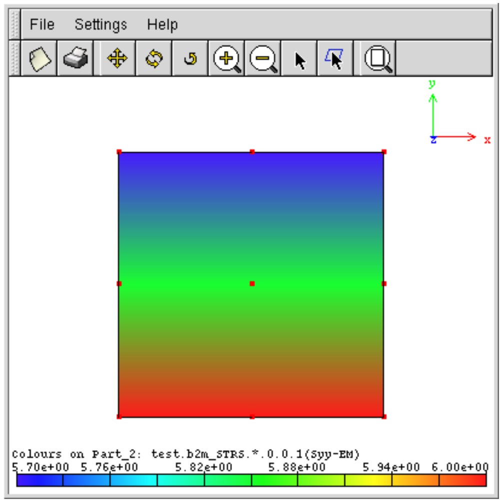

File Settings Help
z
y
x
Iso-lines on Part_1: Zalt_MODE.*.0.0.5(amplitude)
8.60e-05 2.00e-01 4.00e-01 6.00e-01 8.00e-01 1.00e+00

Isolines


File Settings Help
z
y
x
Colours on Part_1: Zalt_MODE.*.0.0.5(amplitude)
0.00e+00 2.00e-01 4.00e-01 6.00e-01 8.00e-01 1.00e+00

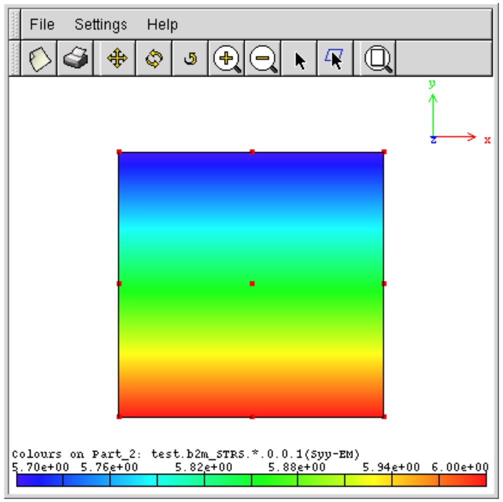
Isolines and colours superimposed

Features: Colour Maps



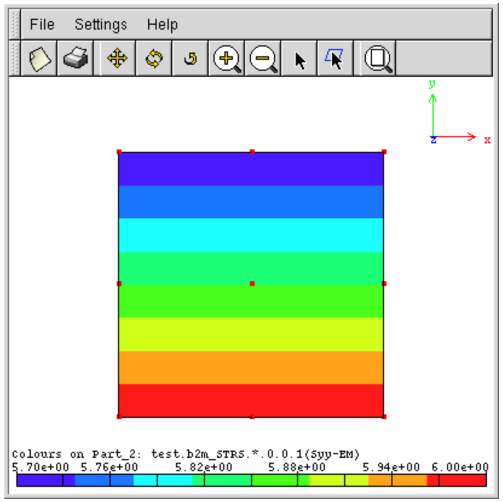
```
Colours on Part_2: test.b2m_STRS.*.0.0.1(Syy-EM)
5.70e+00  5.76e+00      5.82e+00      5.88e+00      5.94e+00  6.00e+00
```

Gouraud shading (default)



```
Colours on Part_2: test.b2m_STRS.*.0.0.1(Syy-EM)
5.70e+00  5.76e+00      5.82e+00      5.88e+00      5.94e+00  6.00e+00
```

Texture mapping

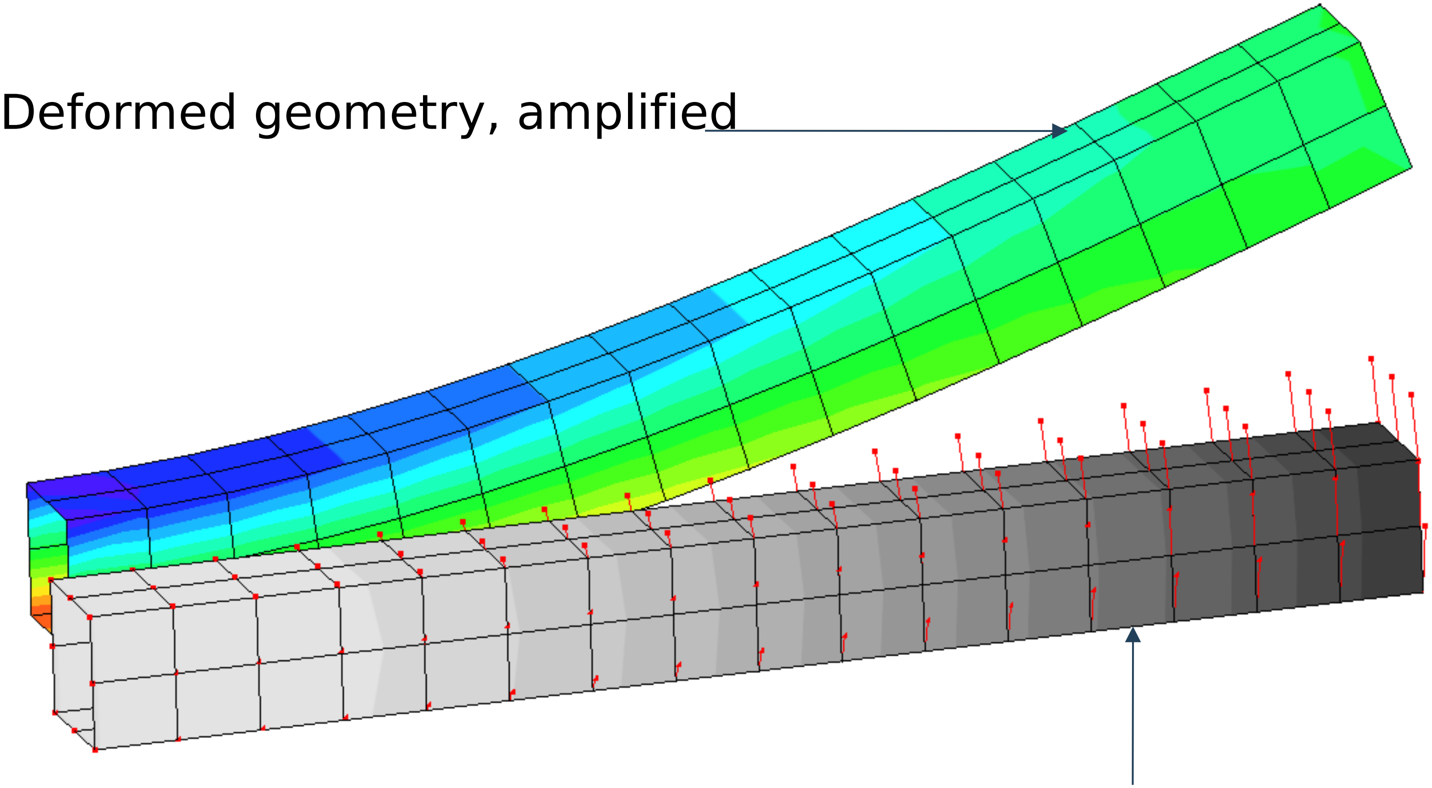


```
Colours on Part_2: test.b2m_STRS.*.0.0.1(Syy-EM)
5.70e+00  5.76e+00      5.82e+00      5.88e+00      5.94e+00  6.00e+00
```

Step-texture mapping

SMR Engineering & Development

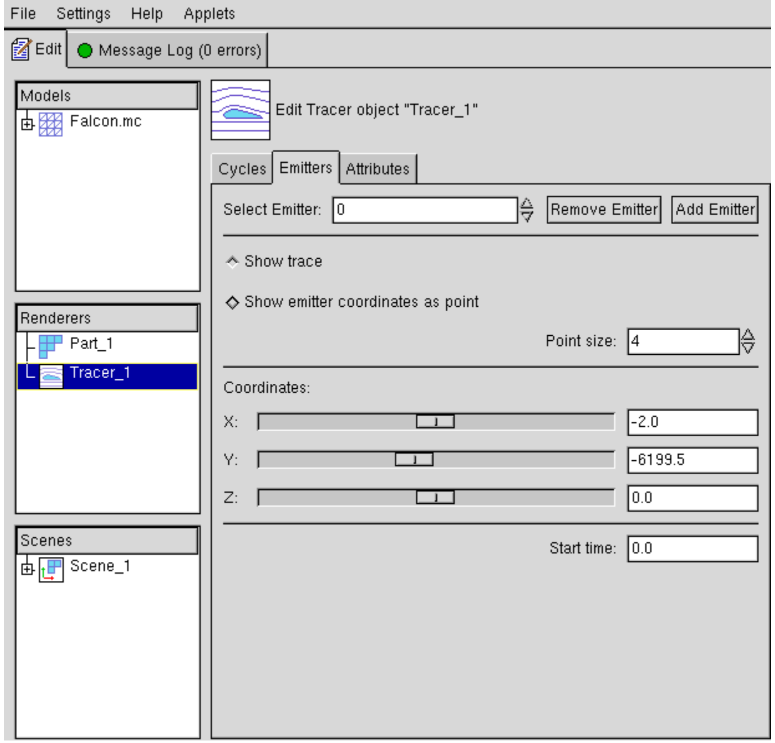
Features: Display Deformed Geometry and Vectors



Deformed geometry, amplified

Deformation as vectors
and in greyscale,
structure translated

Features: Display Streamlines



Edit Tracer object "Tracer_1"

Select Emitter: 0

◇ Show trace

◇ Show emitter coordinates as point

Point size: 4

Coordinates:

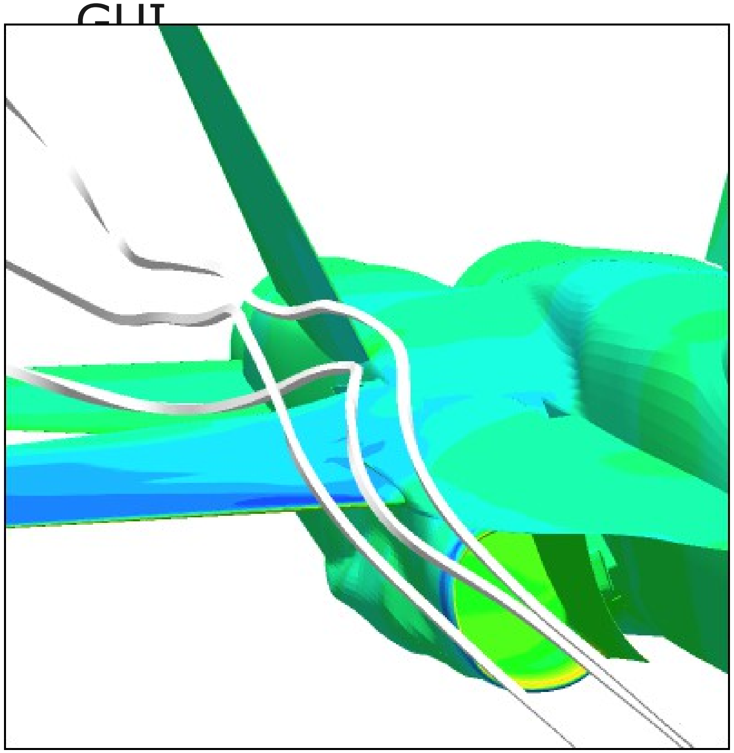
X: -2.0

Y: -6199.5

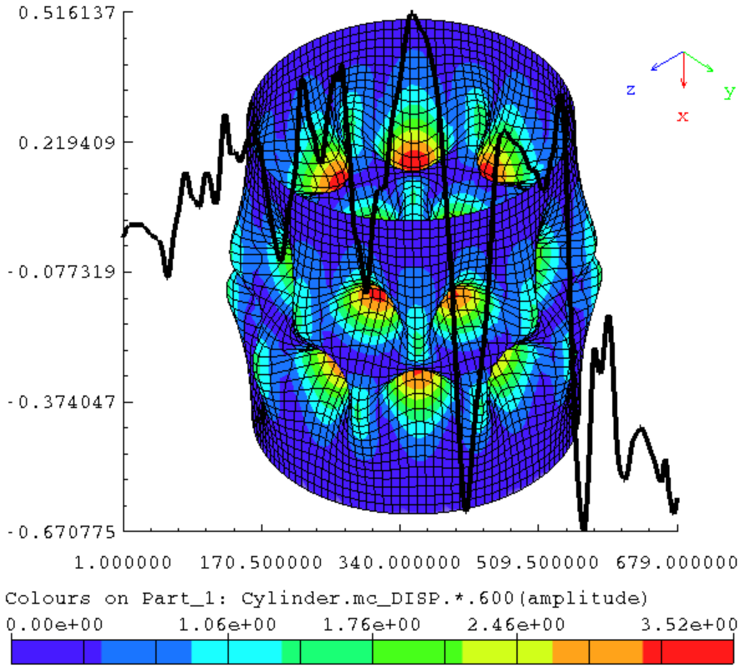
Z: 0.0

Start time: 0.0

- Streamlines or particle traces important for study of flows.
- 4th order integration for high accuracy.
- Control of particle tracer through GUI



Features: XY-graphs



Dynamic buckling of a cylinder

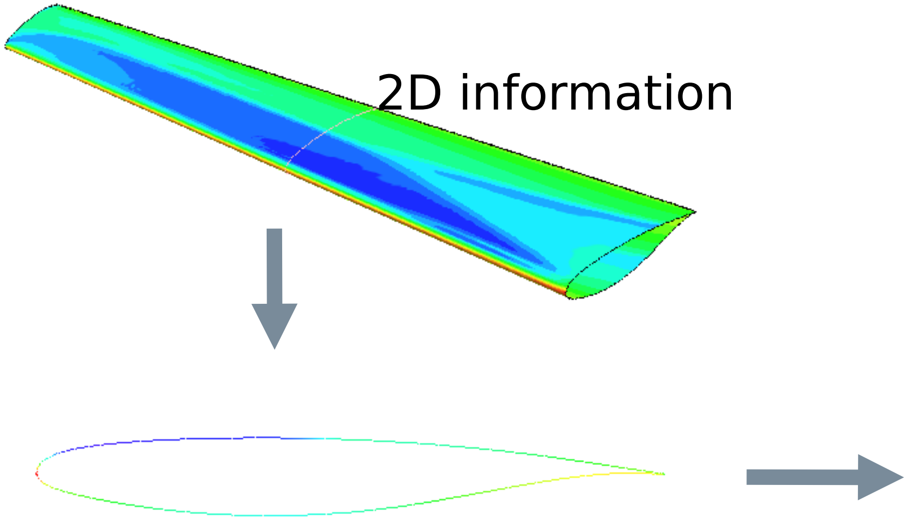
- Data input through CLI only.
- Graph layout through GUI or CLI
- Additional functionality:
  - Data manipulation with python
  - Input of external data
  - Mixing of history plot and geometry display

Features: Data manipulation with Python

- The Python programming language allows all Python code to be executed within baspl++.
- The MemCom Python API (pyMemCom) allows for the extraction of data from the MemCom data base.
- Example applications:
  - Manipulation of field data (results) for graphical display.
  - Post-processing of results to compute derived data.

Features: Data manipulation with Python

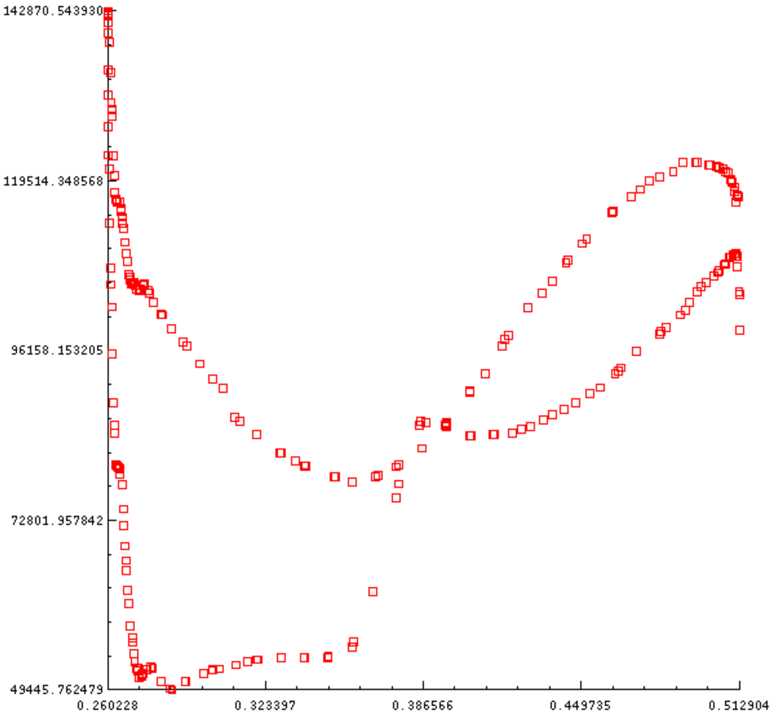
- Example: From volume grid extract surface grid, from surface grid extract 1D information, like coordinates, connectivites, field values.



2D information

1D information

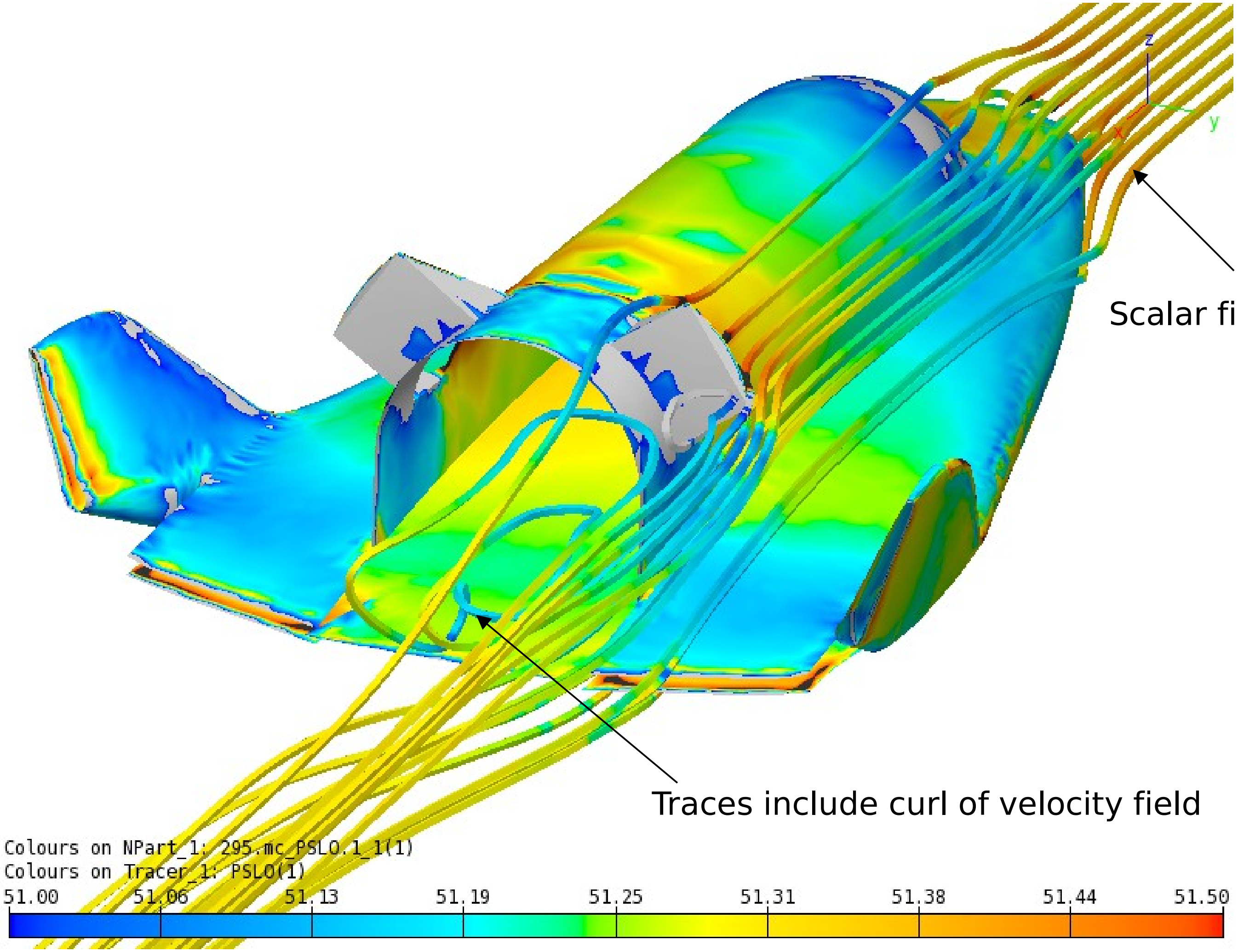
x-y plot of processed data



Importers

- baspl++ works with a database: Importers convert to baspl++ database.
- Currently available converters:
  - TAU: Read NetCDF variables and converts them to MemCom datasets. Integrated in the baspl++ GUI.
  - EDGE/FOI (Swedish Defense Agency) unstructured CFD code converter.
  - NASTRAN: Can convert BDF files (with B2000++). Can handle some solution files.
  - Tecplot: ASCII (text) files. FEPOINT, FEBLOCK, etc. zone-types supported.
  - ICEM-CFD: Read ICEM-CFD domain files.
  - Medina.
  - Ad hoc converters.

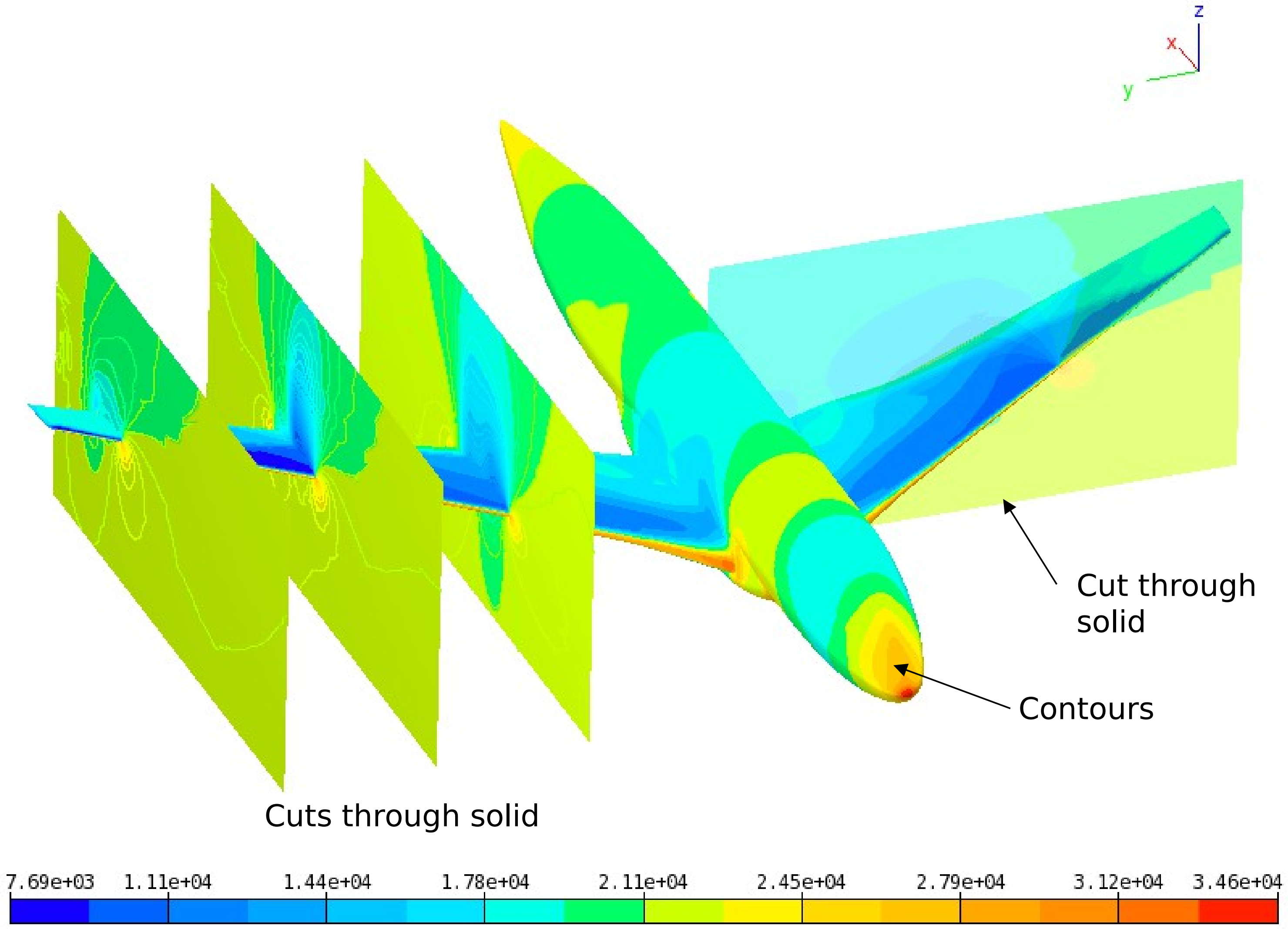
Examples: Streamlines


z
y
x
Colours on NPart_1: 295.mc_PSLO.1_1(1)
Colours on Tracer_1: PSLO(1)
51.00
51.06
51.13
51.19
51.25
51.31
51.38
51.44
51.50

Scalar field plotted along traces

Traces include curl of velocity field

Examples: Combine Viewing Methods

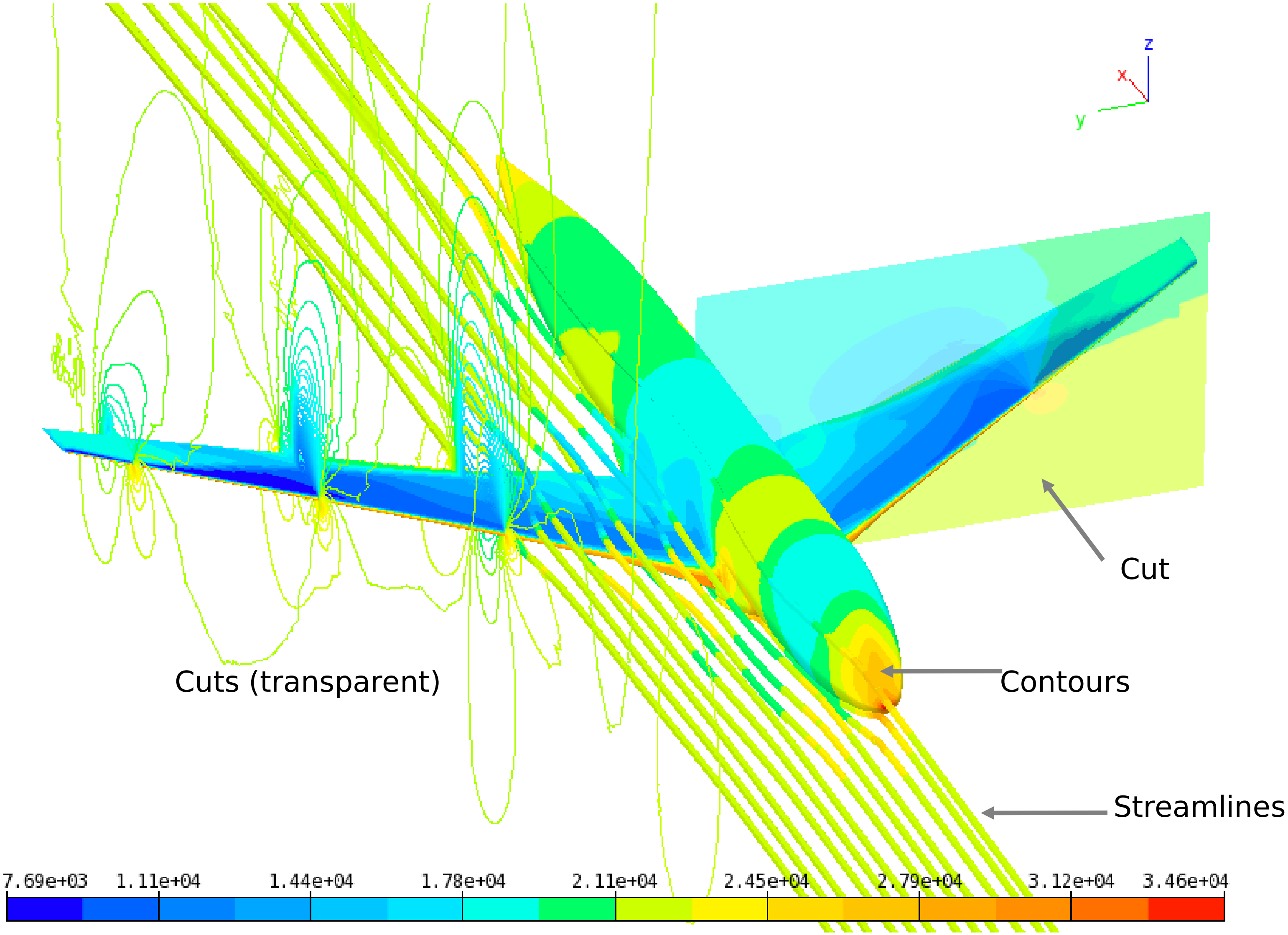


Cut through solid

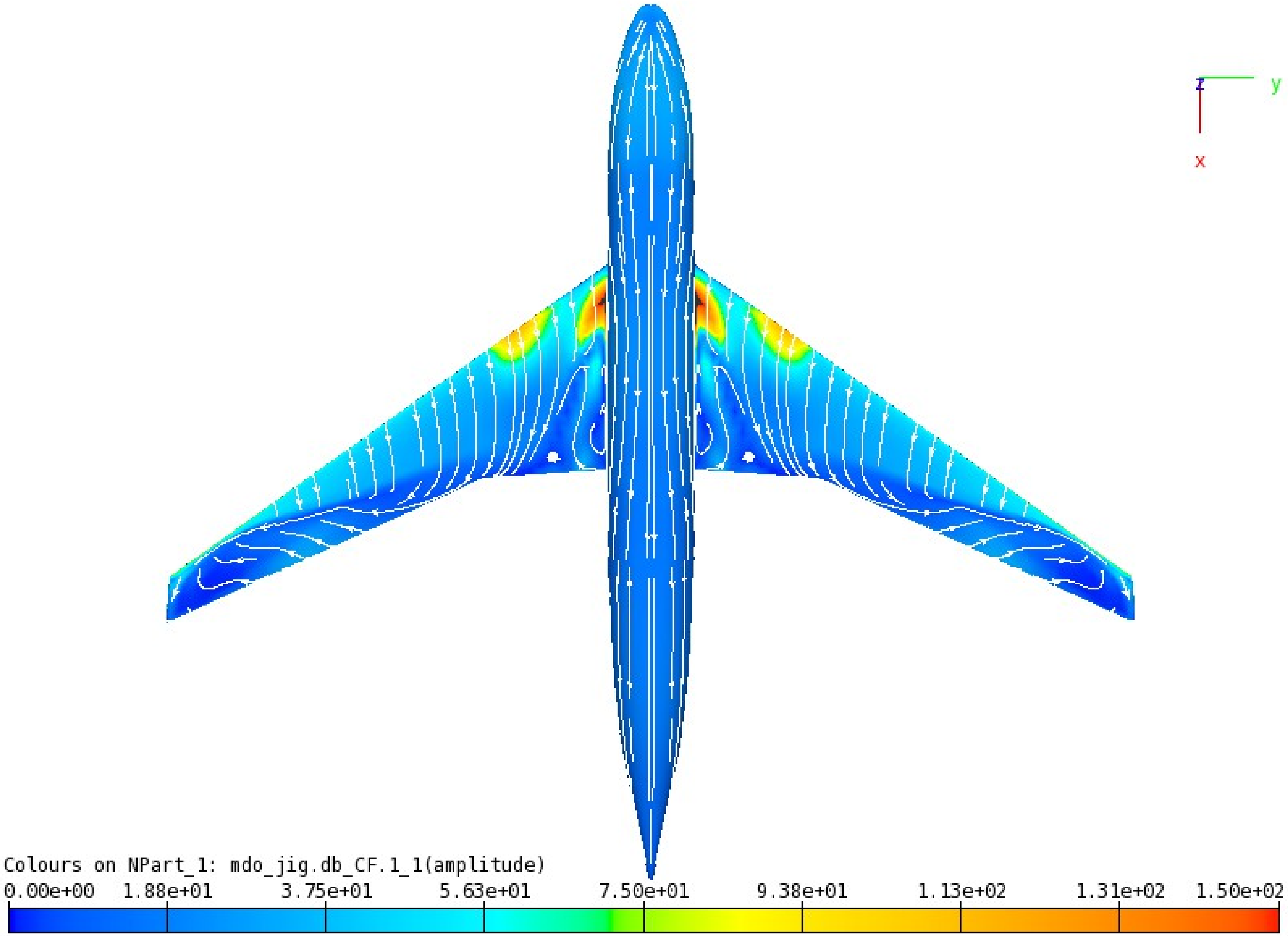
Contours

Cuts through solid

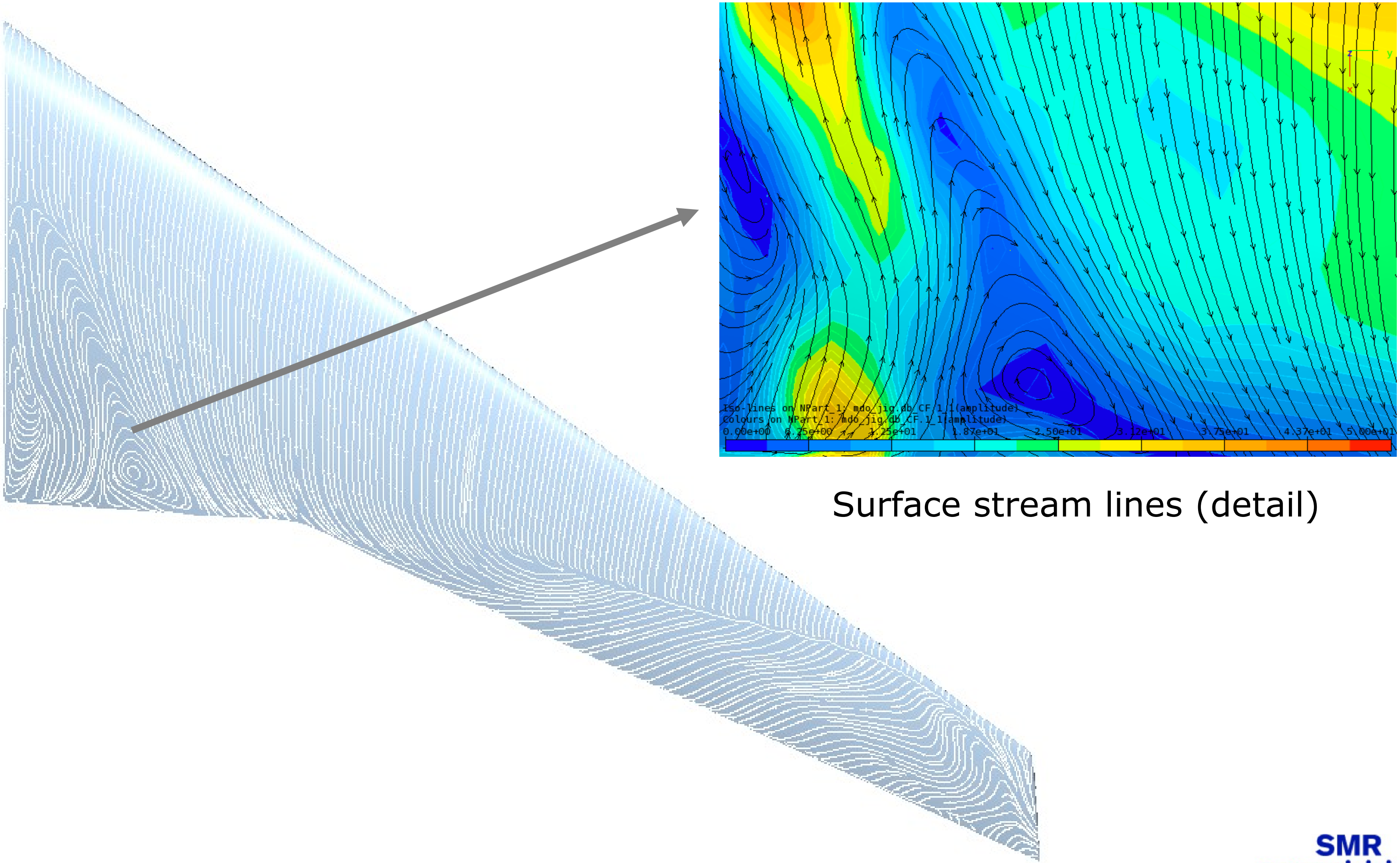
Example: Combined Viewing Techniques



Examples: Surface lines

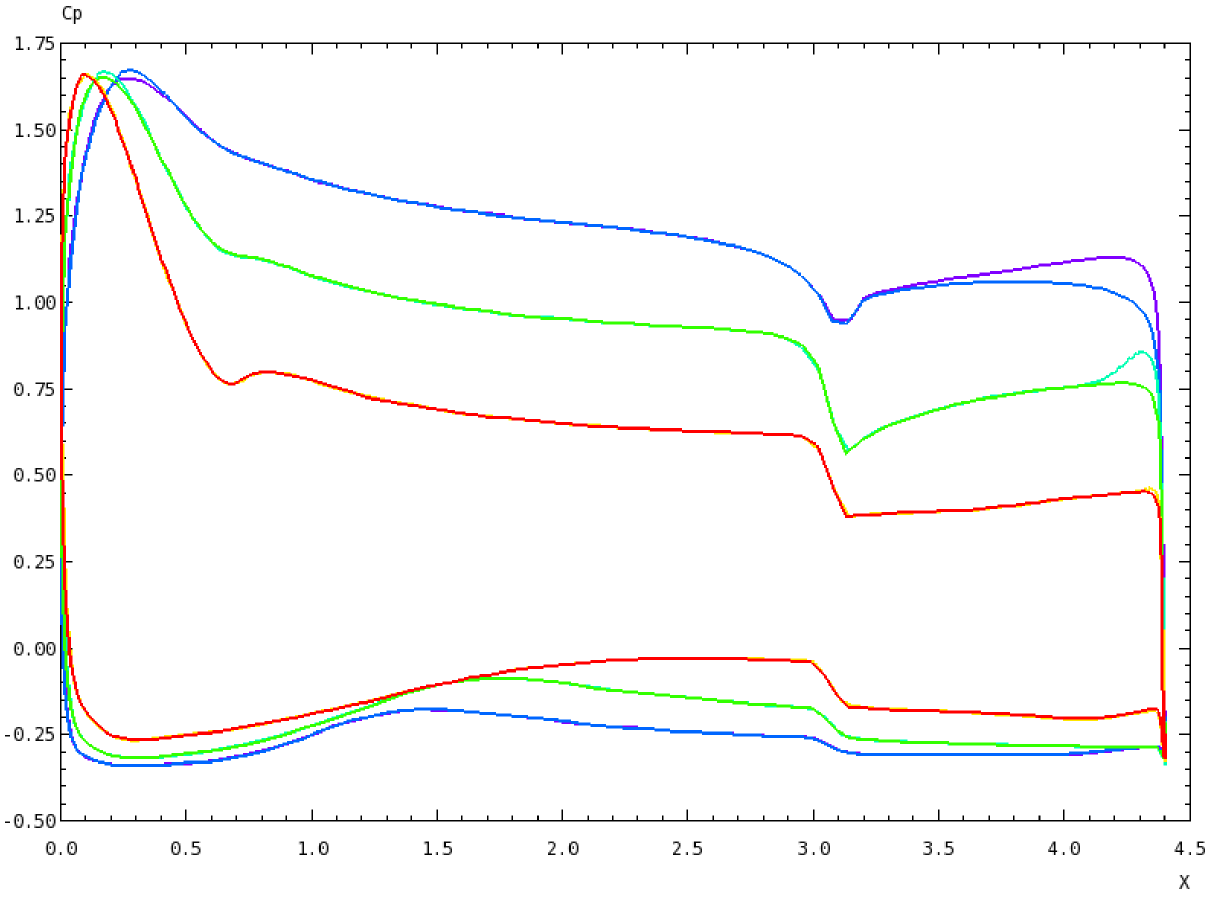


Examples: Surface Streamlines


Iso-lines on NPart_1: mdo_jig.db_CF.1_1(amplitude)
Colours on NPart_1: mdo_jig.db_CF.1_1(amplitude)
0.00e+00
6.25e+00
1.25e+01
1.87e+01
2.50e+01
3.12e+01
3.75e+01
4.37e+01
5.00e+01
z
y
x

Surface stream lines (detail)

Examples: Derived Data



- Cp synthesis plots along a cut
- Import several different models.
- Extract and integrate data.
- Execute with simple script in 'batch' mode.
- Results can be exported to CSV formats.

Technology

C++

- OpenGL graphics engine
  - Industry standard
  - Hardware acceleration on many graphics cards
- Python interface
  - Object-oriented
  - Fast development cycle
  - Powerful scripting capabilities
- C++ for underlying engine
  - Object-oriented
  - Improved maintainability
  - Fast execution of computationally intensive tasks