
# Enabling Catalyst Adoption in SPARC

V. Gregory Weirs, Elaine M. Raybourn, Reed Milewicz, Killian Muollo, Jeffrey A. Mauldin, Thomas J. Otahal

Sandia National Laboratories

*Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.*

# Outline

- Introduction
- Methodology
- Technical analysis
- Content analysis
- Conclusions and future work

# Introduction

Catalyst:

- The in situ version of ParaView
- Has been accessible from several Sandia host codes for years
- Has demonstrated performance and scaling
- Has very few users at Sandia

Not the focus today:

- Performance and scaling
- Generalization to other user communities, other in situ approaches, or other in situ software

The objective of this work is to increase the adoption of Catalyst by SPARC analysts

# pvpython workflow

Current ParaView scripting workflow:

1) Develop pipeline in Paraview, postprocessing results from a “nearby” simulation
2) Export python script
3) Modify script as needed for actual simulation

Exporting a script from the GUI will always be useful:

- Basic structure, starting point
- Snippets of code
- Tool to learn pvpython

- This workflow essentially leverages the GUI to address the usability of the scripting interface.
- If you have results from a nearby simulation, it might be the preferred approach.

How easy is it to modify the script for your simulation with Catalyst?

With confidence that it will work?

# Methodology: Use Cases

This is a “bottom-up” approach

- Use Cases are concrete examples of analyst tasks that are used to identify specific issues
- Recurring specific issues indicate general issues

Some use cases will be adapted to provide Catalyst training on the SPARC Analysts

We interviewed 8 analysts across 8 walkthroughs

We very much appreciate the analysts’ participation and feedback!

Use Cases

Image Generation
- Flowfield images (1)
- LES images (4)

Data Extraction
- LES diagnostics (5)
- Lineouts (6)

Data Manipulation
- Sectional Loads (2)
- Base pressure correlations (3)

*In this presentation we only consider the Image Generation use cases*

# Use Case Walkthroughs (Think-Aloud Protocol)

## 1. Development

- SME Greg Weirs developed the Paraview steps to produce the desired output
- SME developed Analyst Instructions
- SME identified bugs and usability issues while determining the steps

## 2. Walkthrough Interview

- Analyst works through the instructions during the walkthrough interview with SME
- Analysts identify bugs and usability issues, provide feedback, and ask questions

## 3. UUA Content Analysis

- Project team reviews video and transcript of interview
- Identifies analyst statements and applies coding
- Analysis of coded data



Technical Analysis:
Specific Catalyst and pvpython bugs and usability issues

Content Analysis:
Factors enhancing or limiting Catalyst adoption

# UX Objectives and Approach

**OBJECTIVES**:

1) identify usability barriers in Catalyst and its Python scripting interface
2) develop recommendations to incentivize adoption

**APPROACH**

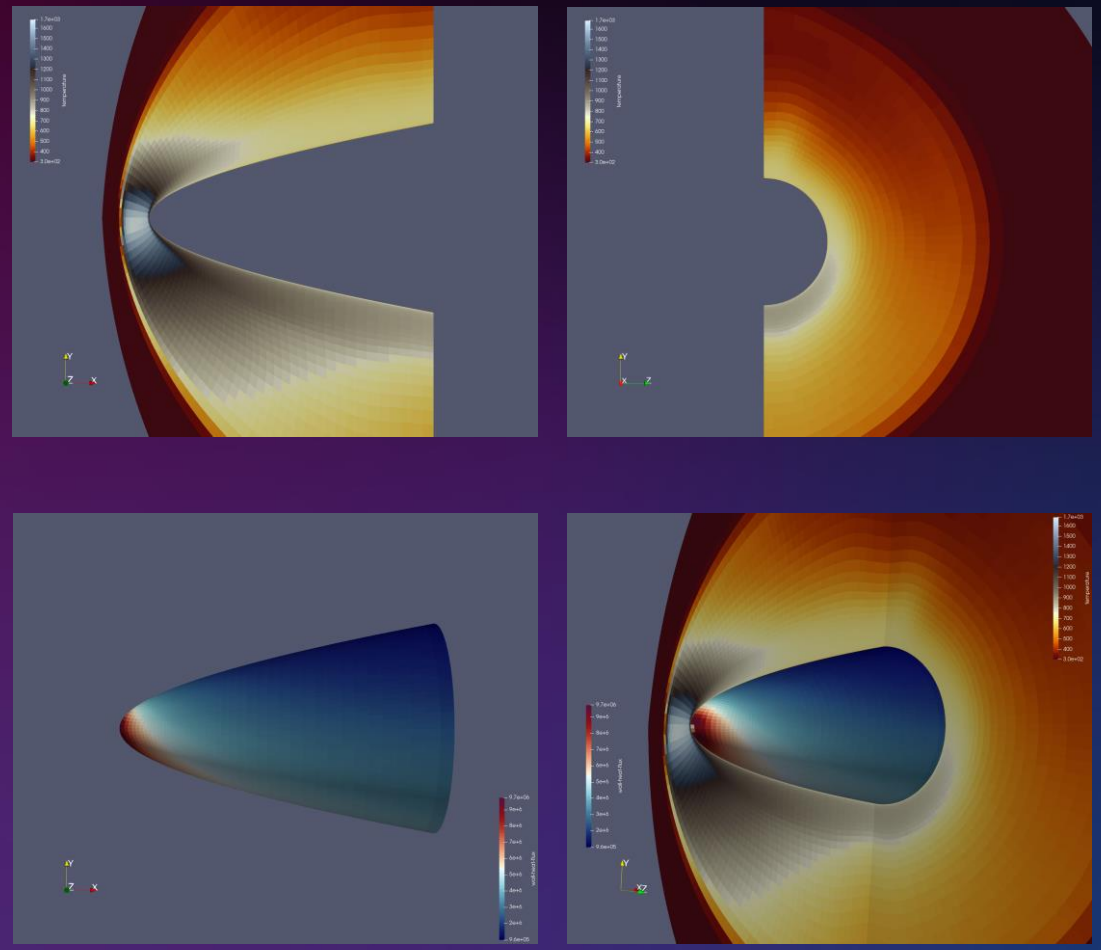
- Corroborate and/or identify UUA barriers via data manipulation requirements of 4-5 case studies (scenarios)
- Apply *discount usability engineering* (IEEE Software, 12,1,98-100; Nielsen, 1995)
- Conduct 4-5 *think aloud verbal protocol* with analysts/developers (Lewis, 1982)
- Capture audio/screen recording and transcription of Catalyst/Paraview workflow while user performs think aloud protocol, follow-up as needed with *task walkthrough* (Greenberg, 2003)
- Document observations and use case workflow
- Perform *content analysis* methodology for further data set exploration

# Example Dataset: SPARC Test Vehicle

- Flow conditions:
  - Mach 5, sea level
  - 10º angle of attack
  - Perfect gas model, RANS
- Provides:
  - Flowfield Data (Volume)
  - Wall Data (Surface)
- SPARC simulation
  - 18 cores on 1 node of CTS-1
  - About an hour (near separation on lee side)

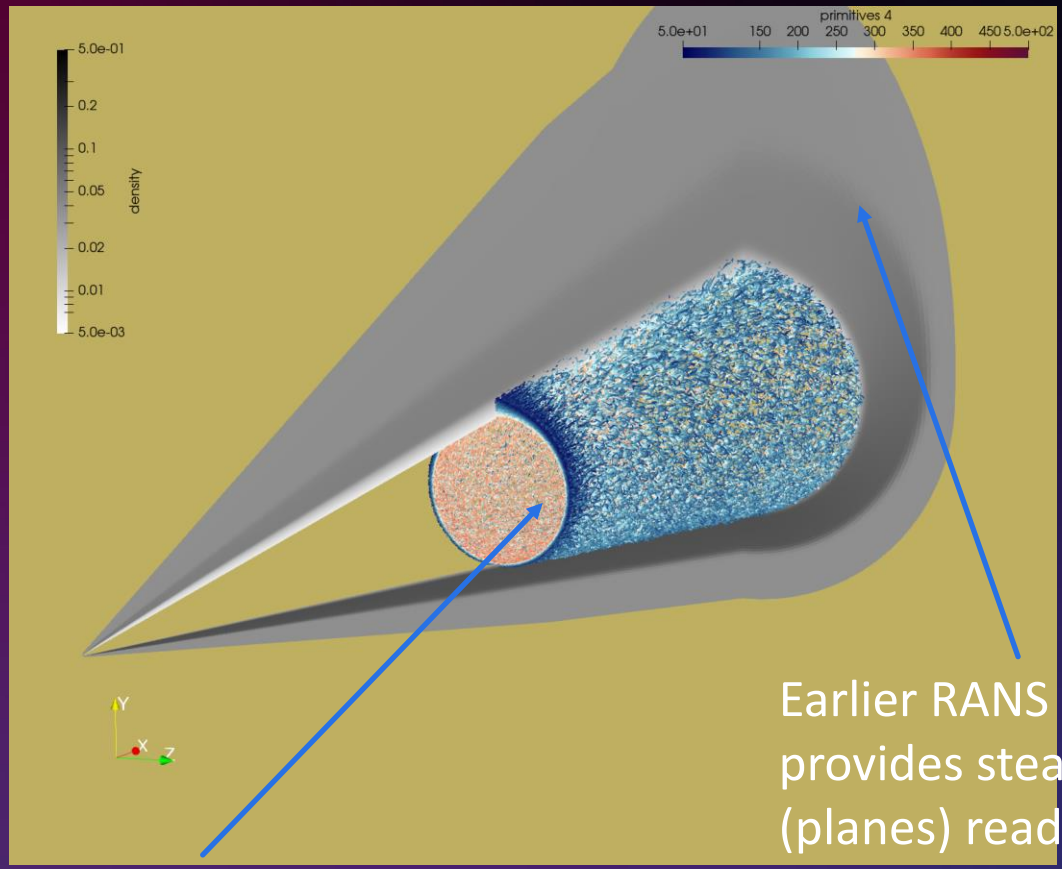
This dataset is used for all Use Cases except the LES cases

Images generated by Catalyst



# Example Dataset: LES of Flow Over a Cone

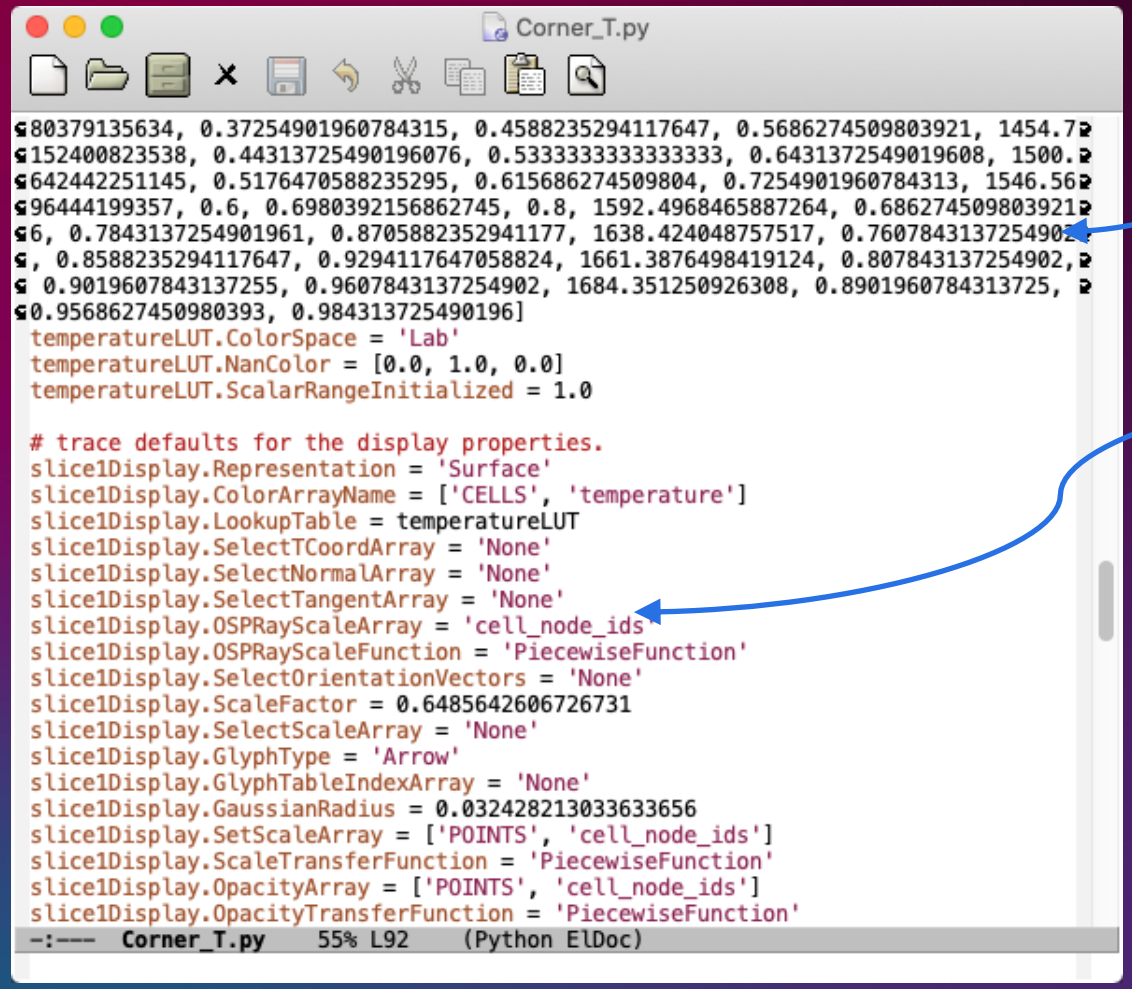
- Flow conditions:
  - Mach 8, 5000ft altitude
  - 6º angle of attack
  - Perfect gas model, LES
- LES provides, in situ:
  - Transient flowfield
  - data
- RANS provides, from files:
  - Steady state flowfield data
- SPARC simulation
  - 3600 cores on 100 nodes of CTS-1
  - About 24hrs to go through startup transient, significantly less to take turbulence statistics




Earlier RANS simulation provides steady state data (planes) read from files

LES simulation provides fine-scale data near cone surface

# Exported scripts are verbose




Last entries of the the table mapping temperature values to colors

Representation properties never set or modified by the user

Verbosity is a barrier to usability because the user cannot find the the sections of the script to focus on

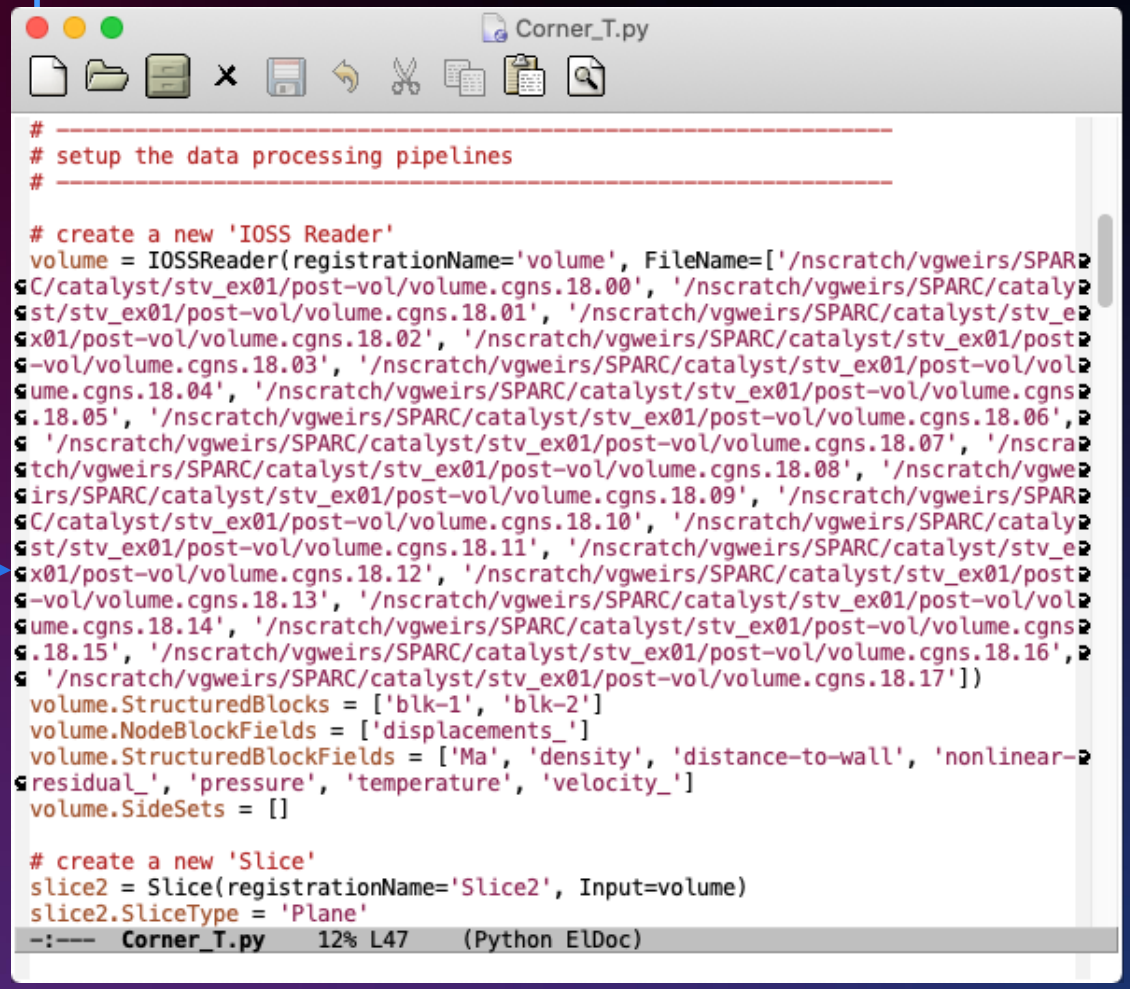
# Exported scripts are dataset specific

Information was not explicitly entered; most was determined automatically

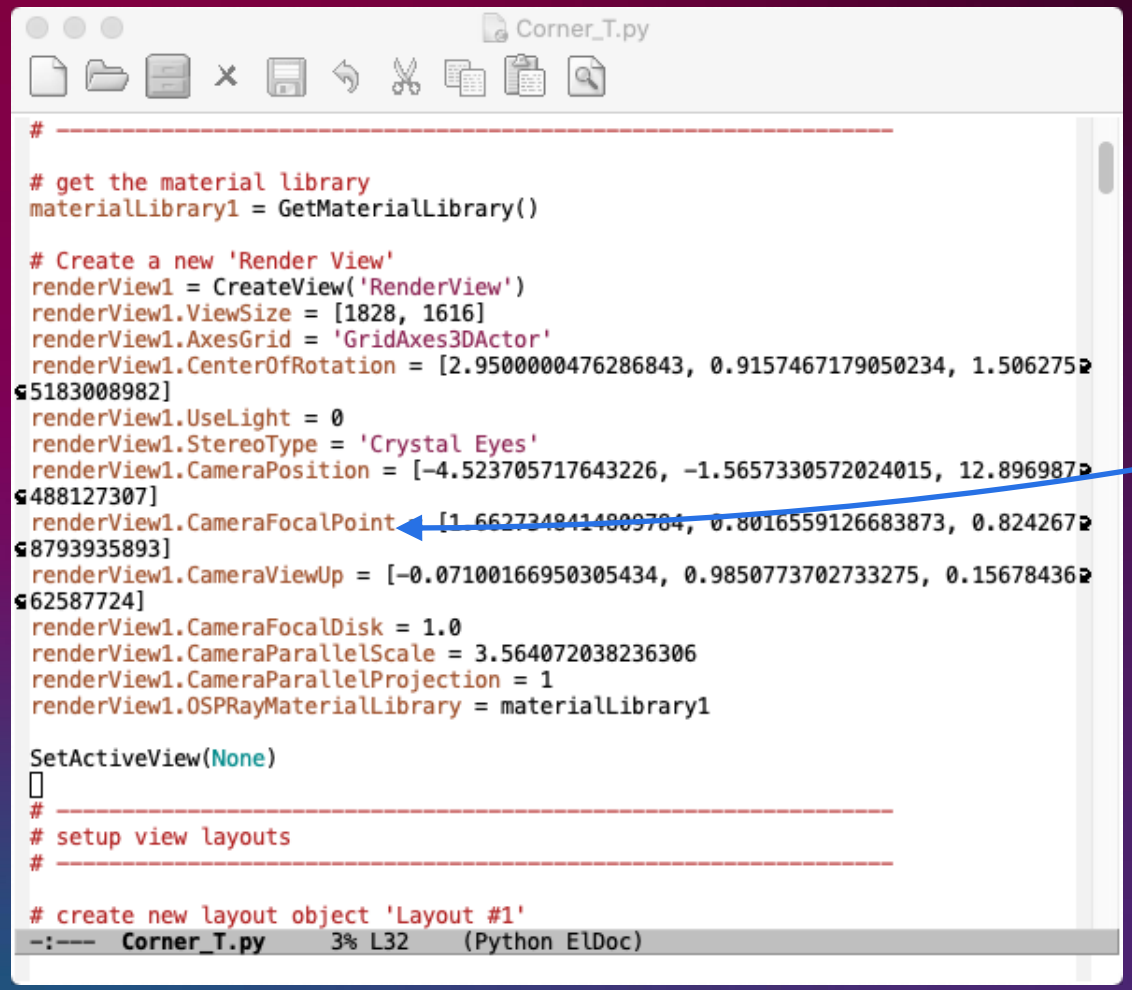
- Filenames
- Blocks
- Variables

(This is a file reader that Catalyst would ignore, but the interactive and batch python clients use file readers.)

Specificity is a barrier to usability when the intent is to modify the script for a different dataset.



# View (camera) properties are not intuitive




A user could understand what these properties mean with a schematic

But they would still have a hard time adjusting the values without the GUI or a similar tool

Such properties reduce usability for any use besides archiving.

# pvpython Filters are concisely represented

While the Views and Representations (Displays) are not satisfactory, Filters and Extracts are clear and concise

- For quantitative analyses, Views and Representations are not needed
- If the GUI can generate clear code for Filters and Extracts, maybe its output for Views and Representations can be improved

Corner_T.py

```
# create a new 'Slice'
slice2 = Slice(registrationName='Slice2', Input=volume)
slice2.SliceType = 'Plane'
slice2.HyperTreeGridSlicer = 'Plane'
slice2.SliceOffsetValues = [0.0]

# init the 'Plane' selected for 'SliceType'
slice2.SliceType.Origin = [2.95, 0.0, 0.0]

# init the 'Plane' selected for 'HyperTreeGridSlicer'
slice2.HyperTreeGridSlicer.Origin = [1.4417453155408007, 0.916163029237911, 1.51
16143433093596]

# create a new 'Slice'
slice1 = Slice(registrationName='Slice1', Input=volume)
slice1.SliceType = 'Plane'
slice1.HyperTreeGridSlicer = 'Plane'
slice1.SliceOffsetValues = [0.0]

# init the 'Plane' selected for 'SliceType'
slice1.SliceType.Origin = [0.0, 0.0, 1e-06]
slice1.SliceType.Normal = [0.0, 0.0, 1.0]

# init the 'Plane' selected for 'HyperTreeGridSlicer'
slice1.HyperTreeGridSlicer.Origin = [1.4417453155408007, 0.916163029237911, 1.51
16143433093596]

# ----------------------------------------------------------------
# setup the visualization in view 'renderView1'
# ----------------------------------------------------------------

# show data from slice1
slice1Display = Show(slice1, renderView1, 'GeometryRepresentation')
```

-:---  **Corner_T.py**  23% L84  (Python ElDoc)

# Technical Findings for Image Generation

Representations and Views
- pvpython scripts are verbose
- pvpython scripts embed data-specific information

Auxiliary objects (e.g., color lookup tables)
- Explicitly specified when they could be referenced
- Embed dataset bounds
- Impractical to modify directly

Image extracts should be rethought:
- Appearance in the pipeline browser is misleading; View state is saved, not pipeline state
- Likelihood of clobbering previous extracts is high
- No mechanism for multiple views in a single image

Image Generation Summary

- Analysts have started using Catalyst for image generation; usability issues are inconveniences
- Generally, issues identified have straightforward solutions
- Required improvements are in the details, implementation, and testing, rather than conceptual

# UUA Content Analysis

- For each walkthrough, each member of the project team reviewed the video and transcript
- Analyst statements relevant to usability and adoption were identified and coded
- Coding: interpreting analysts statements and assigning them to a single category

## Characteristics of Innovations

E. M. Rogers, 1995, *Diffusion of Innovations*, pg. 15-17.

| | |
|---|---|
| **Relative Advantage** | The degree to which an innovation is perceived as better than the idea it supersedes. The greater the perceived relative advantage of an innovation, the more rapid its rate of adoption will be. |
| Compatibility | The degree to which an innovation is perceived as being consistent with the existing values, past experiences, and needs of potential adopters. |
| Complexity | The degree to which an innovation is perceived as difficult to understand and use. New ideas that are simple to understand are adopted more rapidly than innovations that require the adopter to develop new skills and understandings. |
| Trialability | The degree to which an innovation may be experimented with on a limited basis. An innovation that is trialable represents less uncertainty to the individual who is considering it for adoption, as it is possible to learn by doing. |
| Observability | The degree to which the results of an innovation are visible to others. The easier it is for individuals to see the results of an innovation, the more likely they are to adopt it. |

# UUA Content Analysis - Results

| Use Case | Relative Advantage | Compatibility | Complexity | Trialability | Observability |
|---|---|---|---|---|---|
| Image generation | | | | | |
| (1) Flowfield | 26 | 62 | 48 | 49 | 14 |
| (4) LES images | 6 | 33 | 16 | 31 | 3 |

Data is coarse; don't try to draw fine distinctions

- 2 walkthroughs for use case (1), 1 for use case (4)
- Each analyst is unique
- Coders (analyzing transcripts) are also unique, with varied backgrounds

- Compatibility: analysts see how Catalyst fits in their existing workflow
- Trialability: analysts find it easy to experiment with Catalyst to learn how to use it
- Complexity: Following a Catalyst script is ok, but significant modification is more daunting
- Relative advantage: Catalyst offers something better than their existing solution
- Observability: Analysts have heard of or seen Catalyst being useful for others so they would pursue it

# Data for Image Generation UUA Analysis

*Raw data from Teams's automated voice-to-text transcription of collegial, informal conversations – please don't judge our grammar too harshly!

- Some components of it, I -- I think you're right, where it would be more valuable to be able to dig into the Python script. But in term of, um, you know, setting -- setting up a script, hooking it to SPARC and all that, it seems, uh, pretty well streamlined. (trialability)
- Yeah, process-wise, I think it's, yeah, I think it's, uh, very -- very useful. (compatibility)
- So, if you just had the -- a catalyst input file, we could tack on to, um, example one or two in the SPARC examples. That would just be another place for people just to pull down an example that they can just hit "go" on, and it'll -- and Catalyst will work. (trialability)
- My initial reaction is everything was relatively painless… . The syntax changes are very straightforward in the input file. (complexity)
- In a lot of those instances...we want high frequency images, say, to make movies..but we might want to shed the data that is associated...we don't want terrabyte size files. (observability)

# Conclusions

- For Visualization, we have identified a number of Catalyst and pvpython usability issues but are confident they can be addressed
- Analysts are interested in adopting Catalyst for the capabilities it provides; several walkthrough analysts are already adopting Catalyst in their production workflows

- Direct engagement with end users takes time, but provides invaluable information for increasing software adoption

# Next Steps

Grow Catalyst user base with Visualization and accessible Quantitative Analysis

- Remaining SPARC/Catalyst production readiness issues will be easier to identify with a user community
- Develop SPARC/Catalyst training – we have already discussed this with the SPARC team
- Extend Catalyst adoption to other host codes: SIERRA SM/SD (solid mechanics/solid dynamics)

With Kitware

- Catalyst API2.0 (leverages Conduit)
- Generate cleaner pvpython scripts from ParaView
- Improve Image Extractors

# Quantitative Analysis Path Forward

## Different Solutions for Different Quantitative Tasks