

Summer 2020 Intern Presentation

Type Subtitle Here

*Sequoia Andrade, Eleni Spirakis
Santa Clara University, California and Stanford University, California
Ames Research Center, Moffett Field, California*

*Bryce Campbell
California State Polytechnic University, San Louis Obispo, California
Ames Research Center, Moffett Field, California*

*Thomas Cannon
University of California, Santa Cruz, California
Ames Research Center, Moffett Field, California*

*Sung Hyeok Cho
California State Polytechnic University, Pomona, California
Ames Research Center, Moffett Field, California*

*Alina Kim
University of California, Irvine, California
Ames Research Center, Moffett Field, California*

*Alexander Kleb
University of Michigan, Ann Arbor
Ames Research Center, Moffett Field, California*

*Albert Kutsyy
University of Cambridge
Ames Research Center, Moffett Field, California*

*Rory Lipkis
Stanford University, California
Ames Research Center, Moffett Field, California*

Chi Kei Loi
San Jose State University, California
Ames Research Center, Moffett Field, California

Andrew Moffatt, Umesh Krishnamurthy
California State Polytechnic University, Pomona, California and University of California, Merced,
California
Ames Research Center, Moffett Field, California

Nathan Scheinkman
California State Polytechnic University, San Louis Obispo, California
Ames Research Center, Moffett Field, California

Chelsea Sidrane
Stanford University, California
Ames Research Center, Moffett Field, California

Yanizbeth Yambo, Nathan Doi
University of Puerto Rico – Rio Piedras, San Jose State University, California
Ames Research Center, Moffett Field, California

NASA STI Program ... in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

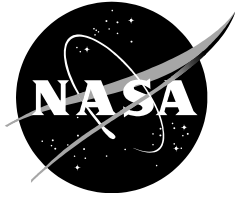
The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NTRS Registered and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- **TECHNICAL PUBLICATION.** Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.
- **TECHNICAL MEMORANDUM.** Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.
- **CONTRACTOR REPORT.** Scientific and technical findings by NASA-sponsored contractors and grantees.
- **CONFERENCE PUBLICATION.** Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.
- **SPECIAL PUBLICATION.** Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.
- **TECHNICAL TRANSLATION.** English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at <http://www.sti.nasa.gov>
- E-mail your question to help@sti.nasa.gov
- Phone the NASA STI Information Desk at 757-864-9658
- Write to:
NASA STI Information Desk
Mail Stop 148
NASA Langley Research Center
Hampton, VA 23681-2199



Summer 2020 Intern Presentation

Type Subtitle Here

*Sequoia Andrade, Eleni Spirakis
Santa Clara University, California and Stanford University, California
Ames Research Center, Moffett Field, California*

*Bryce Campbell
California State Polytechnic University, San Louis Obispo, California
Ames Research Center, Moffett Field, California*

*Thomas Cannon
University of California, Santa Cruz, California
Ames Research Center, Moffett Field, California*

*Sung Hyeok Cho
California State Polytechnic University, Pomona, California
Ames Research Center, Moffett Field, California*

*Alina Kim
University of California, Irvine, California
Ames Research Center, Moffett Field, California*

*Alexander Kleb
University of Michigan, Ann Arbor
Ames Research Center, Moffett Field, California*

*Albert Kutsyy
University of Cambridge
Ames Research Center, Moffett Field, California*

*Rory Lipkis
Stanford University, California
Ames Research Center, Moffett Field, California*

Chi Kei Loi
San Jose State University, California
Ames Research Center, Moffett Field, California

Andrew Moffatt, Umesh Krishnamurthy
California State Polytechnic University, Pomona, California and University of California, Merced,
California
Ames Research Center, Moffett Field, California

Nathan Scheinkman
California State Polytechnic University, San Louis Obispo, California
Ames Research Center, Moffett Field, California

Chelsea Sidrane
Stanford University, California
Ames Research Center, Moffett Field, California

Yanizbeth Yambo, Nathan Doi
University of Puerto Rico – Rio Piedras, San Jose State University, California
Ames Research Center, Moffett Field, California

Prepared for
2020 Summer Intern Virtual Presentations
Conference Sponsor(s)
City, State, Month Day, Year

National Aeronautics and
Space Administration

Ames Research Center
Moffet Field, California 94035-1000

August 2020

Acknowledgments

This report is available in electronic form at
<http://>



SMART-STEReO: System Modeling and Analysis of Resiliency in STEReO

Sequoia Andrade^{1,2}, Eleni Spirakis^{1,3}

NASA Internships at NASA Ames Research Center¹, Santa Clara University², Stanford University³

Background

According to the U.S. Forest Service, there are an average of 7500 wildfires every year within National Forests and Grasslands, over half of which are caused by humans [1]. The top priority of wildfire response is safety of the public and of personnel [1]. The STEReO project is about applying NASA technology to emergency response situations to improve operational effectiveness by leveraging vehicle autonomy, improved communications, and software for better coordination. At present, emergency wildfire response is running on dated, ad-hoc tools to communicate, coordinate, and commence operations. Typically, the only means of communication is through radio (with voice) and a visual line-of-sight. STEReO's suggested implementation and integration of UTM and UAS into wildfire response can simplify the coordination of aerial assets, improve the existing UAS framework, and increase the role of additional autonomous systems to reduce human risk [2].

In addition to performance metrics, resiliency metrics can be used to understand the improvements that STEReO provides wildfire response. Resiliency is the system's ability to withstand or recover from faults, and it is expected that STEReO can improve wildfire response resiliency to several potential faults including radar issues, on-board equipment failures, flight path deviation, and ground crew needs. 'fmdtools' is a Python package ideal for modeling systems and analyzing resiliency through the injection of faults [3].

Objectives

- Work alongside the STEReO project to model current aerial operations for wildfire response and the improvements that can be made to the response by leveraging advanced aviation concepts developed by NASA.
- Provide a system-level perspective on operations by developing a thorough ConOps that provides a high level understanding of the different actors in the system and how they work together.
- Analyze resiliency in current wildfire response and the improvements in resiliency that STEReO will provide.

Model Description

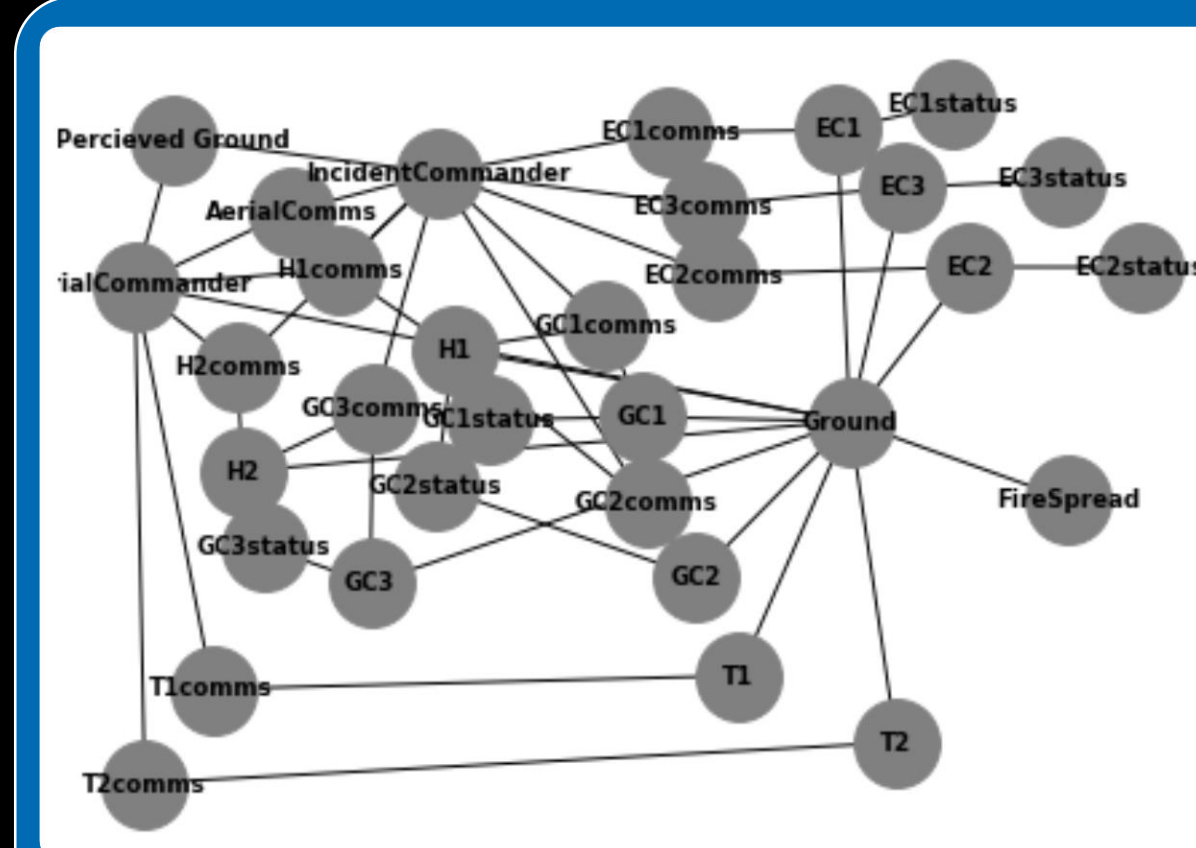


Fig 1: The bipartite graph above shows the flow of communication between different agents in the system, as described in the ConOps

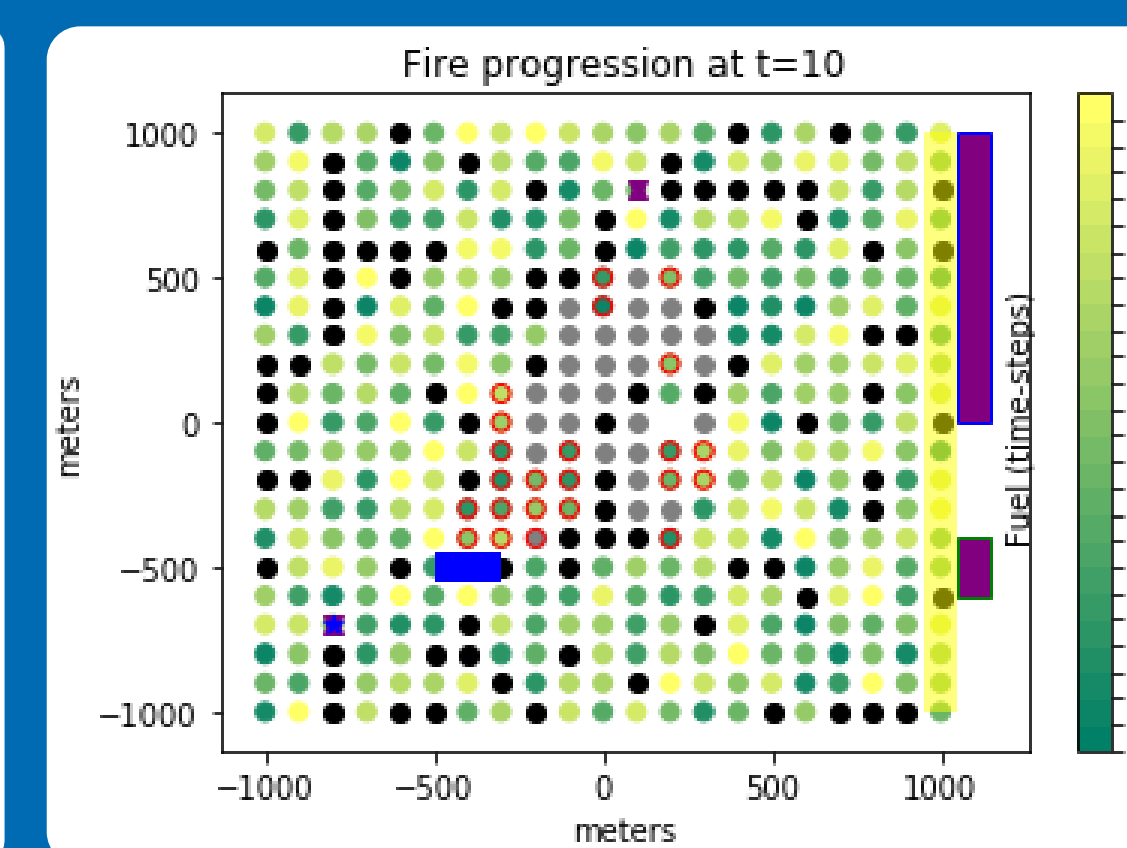


Fig 2: The grid above displays fuel type in the fire propagation component of the model

Model nodes: Fire propagation, Air Tankers, Ground Crews, Helicopters, Aerial Commander, Incident Commander, and UAS
Parameters: amount of each aircraft and crews, size and resolution of the grid, location of points of interest, wind attributes
Faults: Ground crew supplies, mechanical failures, loss of communication, and flight deviation

Model Verification and Validation

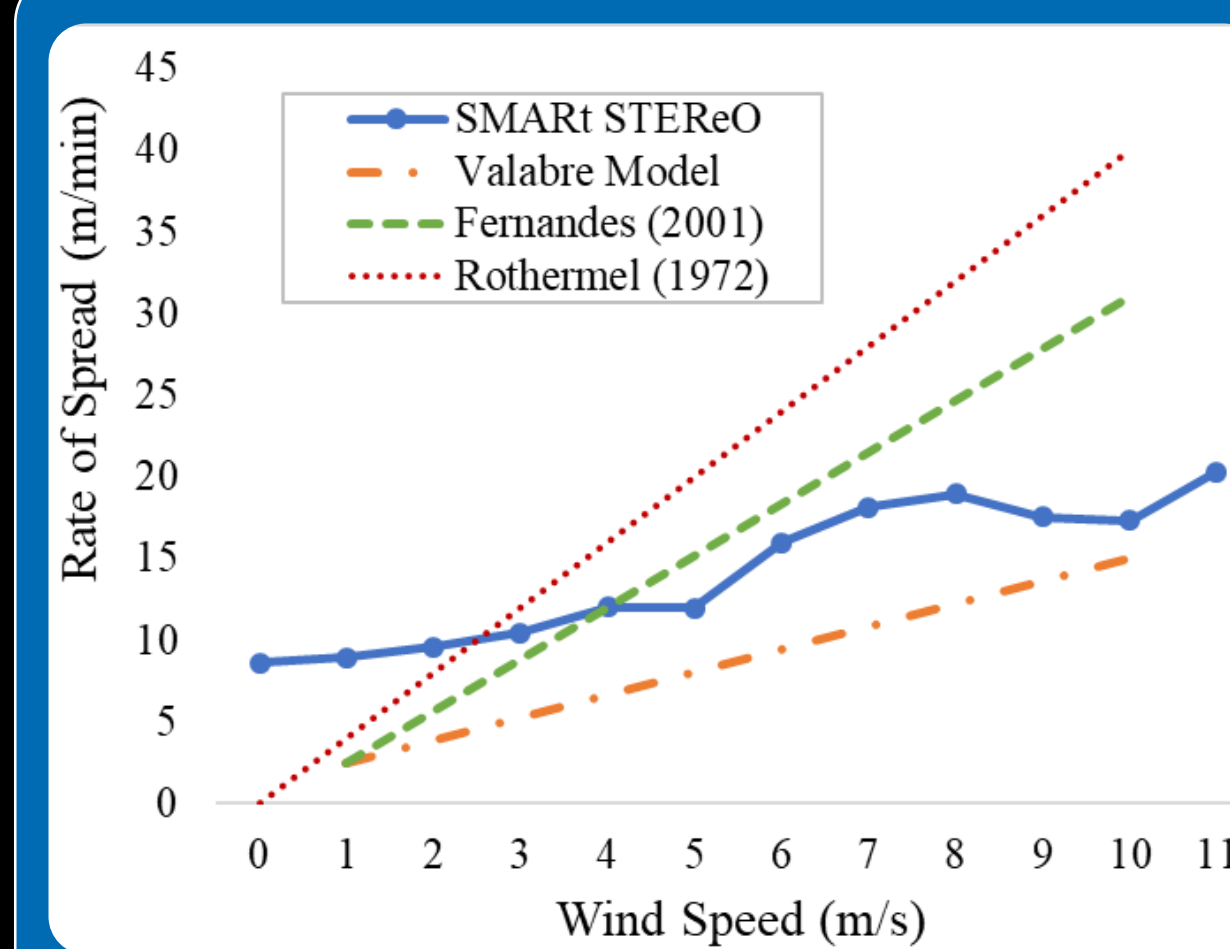


Fig 3: The inclusion of wind, terrain, fuel type, and flame length match the expected trends from existing literature [4, 5].

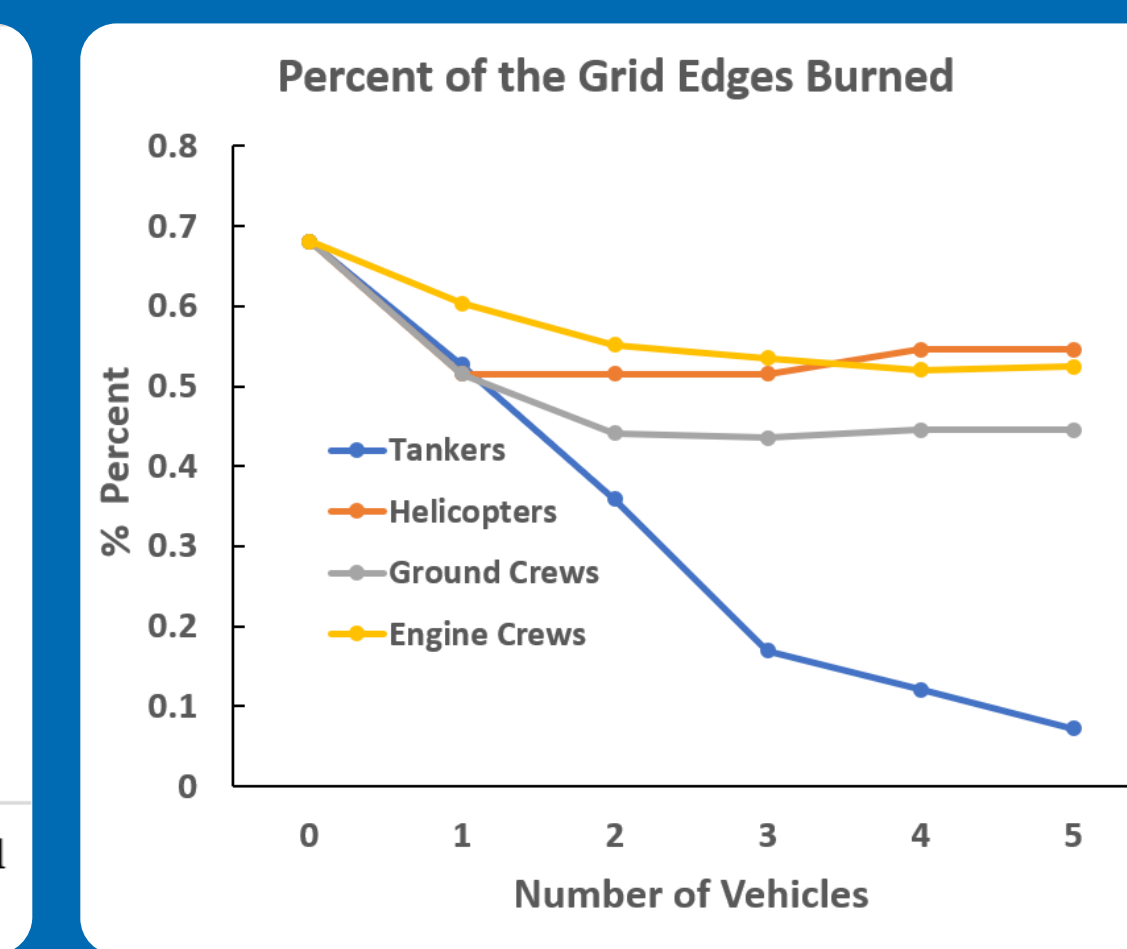


Fig 4: The roles of each crew type are accurate and have been confirmed by the PI for STEReO. Communication lines are established to be nearly identical to those used in real-life fire response.

Model Analysis and Results

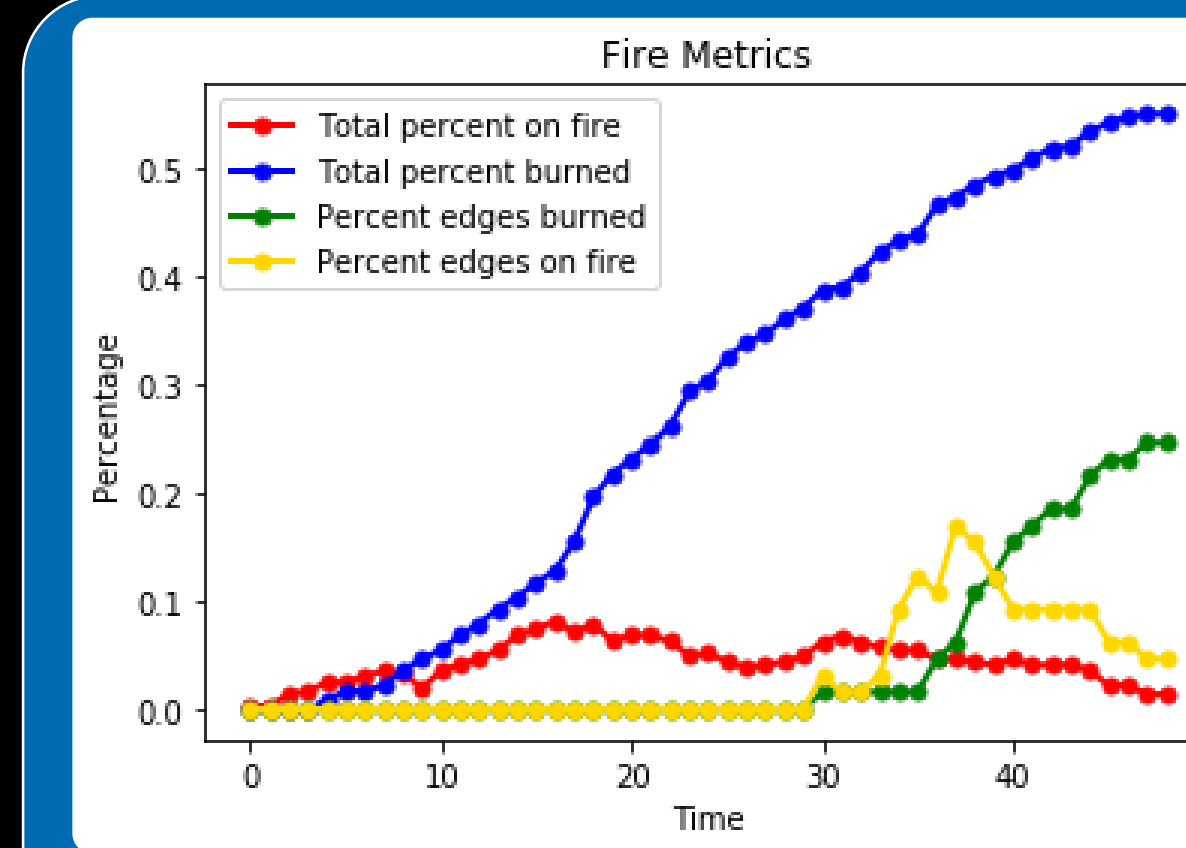


Fig 5: Above the fire metrics graph demonstrates the amount of the grid on fire over time

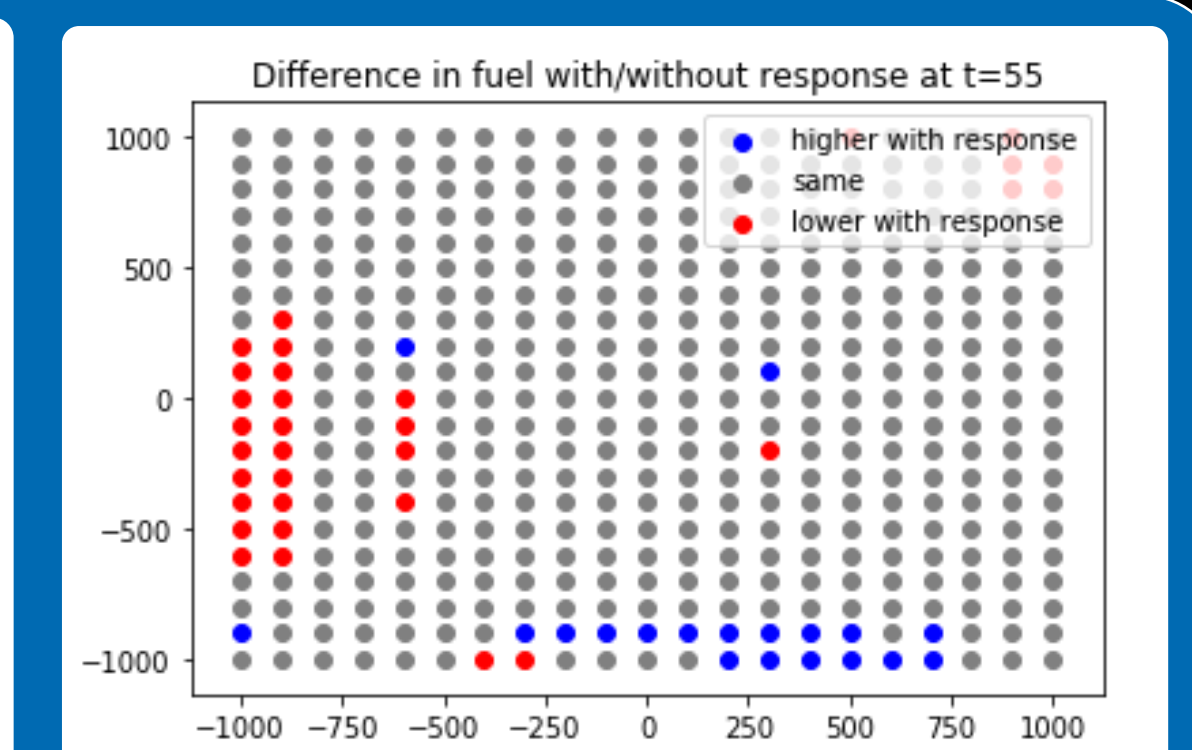
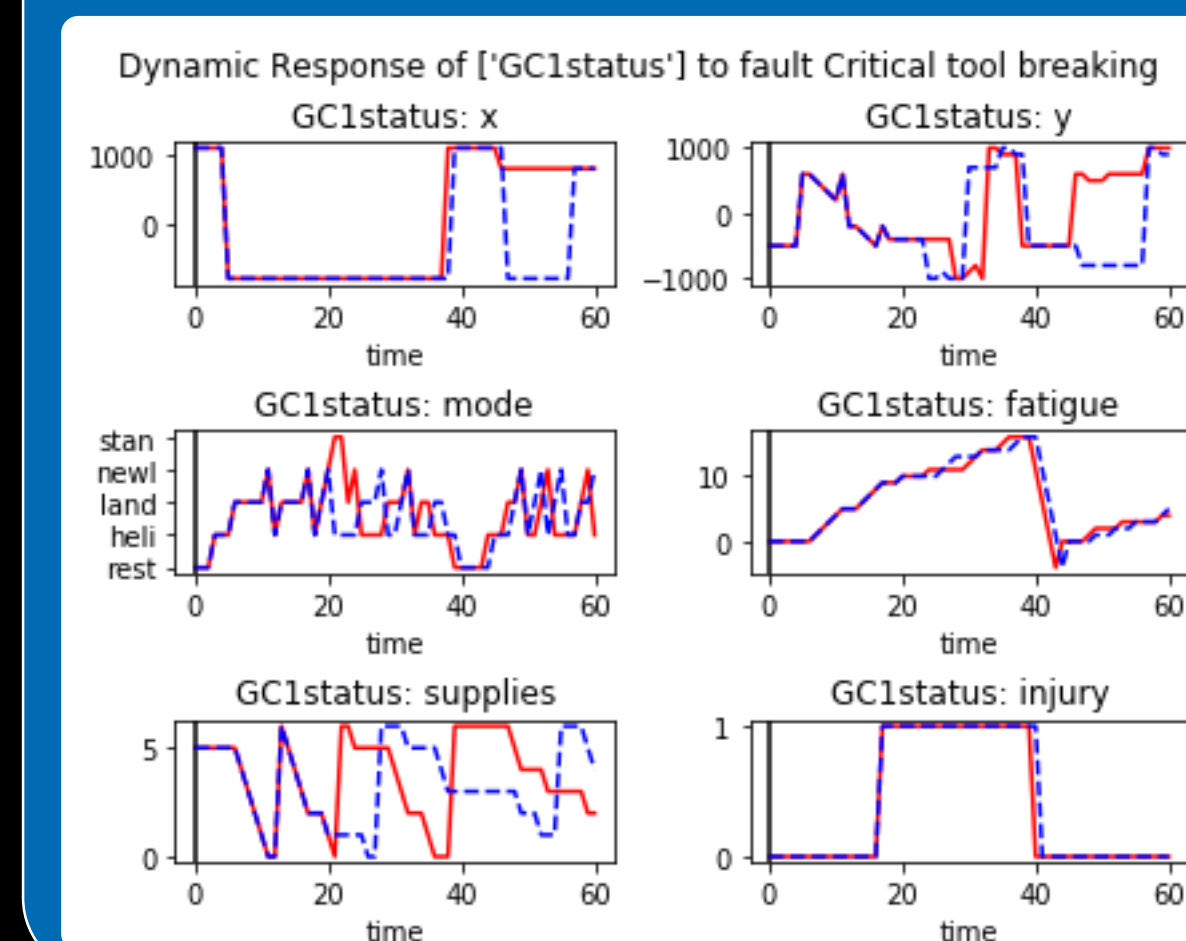


Fig 6: The delta grid above shows the difference in response when the system state awareness is off nominal
Fig 7: The system response to a fault (red) compared to the nominal (blue)



Paired samples t-test	results
0	Difference (nominal model - major mechanical fault) = -0.0047
1	Degrees of freedom = 29.0000
2	t = -2.7167
3	Two side test p value = 0.0110
4	Difference < 0 p value = 0.0055
5	Difference > 0 p value = 0.9945
6	Cohen's d = -0.4960
7	Hedge's g = -0.4896
8	Glass's delta = -0.0266
9	r = 0.4504

Fig 8: Difference between nominal and tanker major mechanical fault

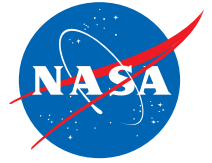
Conclusion

With the goal of demonstrating STEReO's improvement to wildfire response resiliency, the SMART-STEReO project models wildfire propagation, necessary assets used in wildfire response, and the communication lines between assets. The model was verified using existing data and by talking to STEReO representatives. As it is now, the simulations show resiliency metrics and provide the framework for future, detailed analysis of resilience.

References

[1] US Forest Service. (n.d.). Safe and Effective Wildfire Response. US Forest Service. <https://www.fs.usda.gov/managing-land/fire/response>
 [2] NASA. (n.d.). Scalable Traffic Management for Emergency Response Operations (STEReO). NASA. <https://hsi.arc.nasa.gov/groups/AOL/research/stereo.php>
 [3] Dong, A., Goebel, K., Hoyle, C., Hulse, D., Kulkarni, C., Tumer, I., Walsh H. S. (2014). fmdtools: A Fault Propagation Toolkit for Resilience Assessment in Early Design. International Journal of Prognostics and Health Management
 [4] Morvan, D., & Dupuy, J. L. (2004). Modeling the propagation of a wildfire through a Mediterranean shrub using a multiphase formulation. Combustion and flame, 138(3), 199-210.
 [5] Rothermel, R. C. (1972). A mathematical model for predicting fire spread in wildland fuels (Vol. 115). Intermountain Forest & Range Experiment Station, Forest Service, US Department of Agriculture.

Acknowledgments: We would like to thank our mentor, Dr. Misty Davies, our fellow interns who worked alongside us, Hannah Walsh, Daniel Hulse. Also thanks to Joey Mercer from the STEReO project for providing us with STEReO details.



Improving a Scripting Language for Building Autonomous Systems

Bryce Campbell, Cal Poly San Luis Obispo

Dr. Jeremy Frank, Chuck Fry, and Michael Dalal TI - ASR

Summer 2020

Introduction

Planning And Scheduling Group:

- The Planning and Scheduling Group is a part of the larger Autonomous Systems and Robotics area within the Intelligent Systems Division at NASA.
- The goal of the Planning and Scheduling Group is to solve planning and scheduling problems while working with tight temporal constraints and limited resources.

Plan Representation and Execution Language (PLEXIL):

- PLEXIL is a language used to create plans for the automation of systems with formally verifiable semantics.
- The PLEXIL Executive is an open source implementation of the PLEXIL semantics that I spent my internship working on.
- PLEXIL plans query the system they control to make deterministic decisions and command the system accordingly.

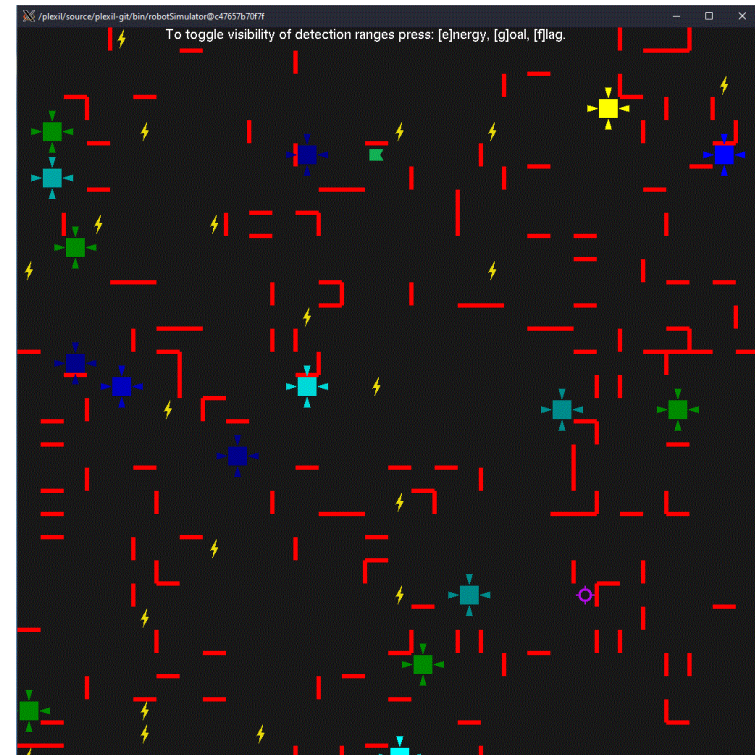
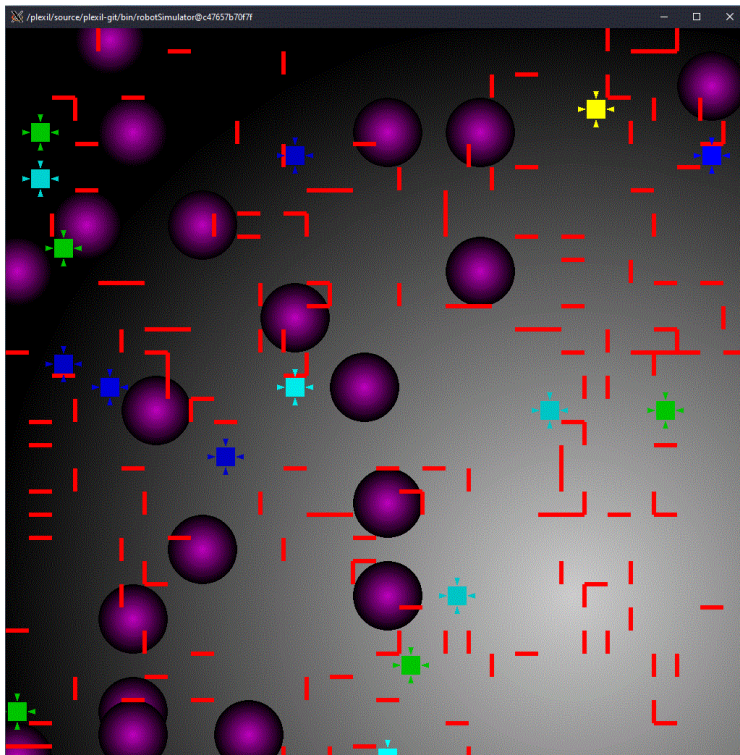
Internship Goals

- Work with mentors at NASA who use PLEXIL in their work to make improvements on the language including learning resources, ease of use, and work on command and lookup infrastructure in the language.
- Learn how to design and implement new features in a language.
- Learn a real-world project workflow that utilizes tickets and code reviews to track and debug code improvements.
- Learn how to write autonomy plans in PLEXIL

Project 1 - Robosim

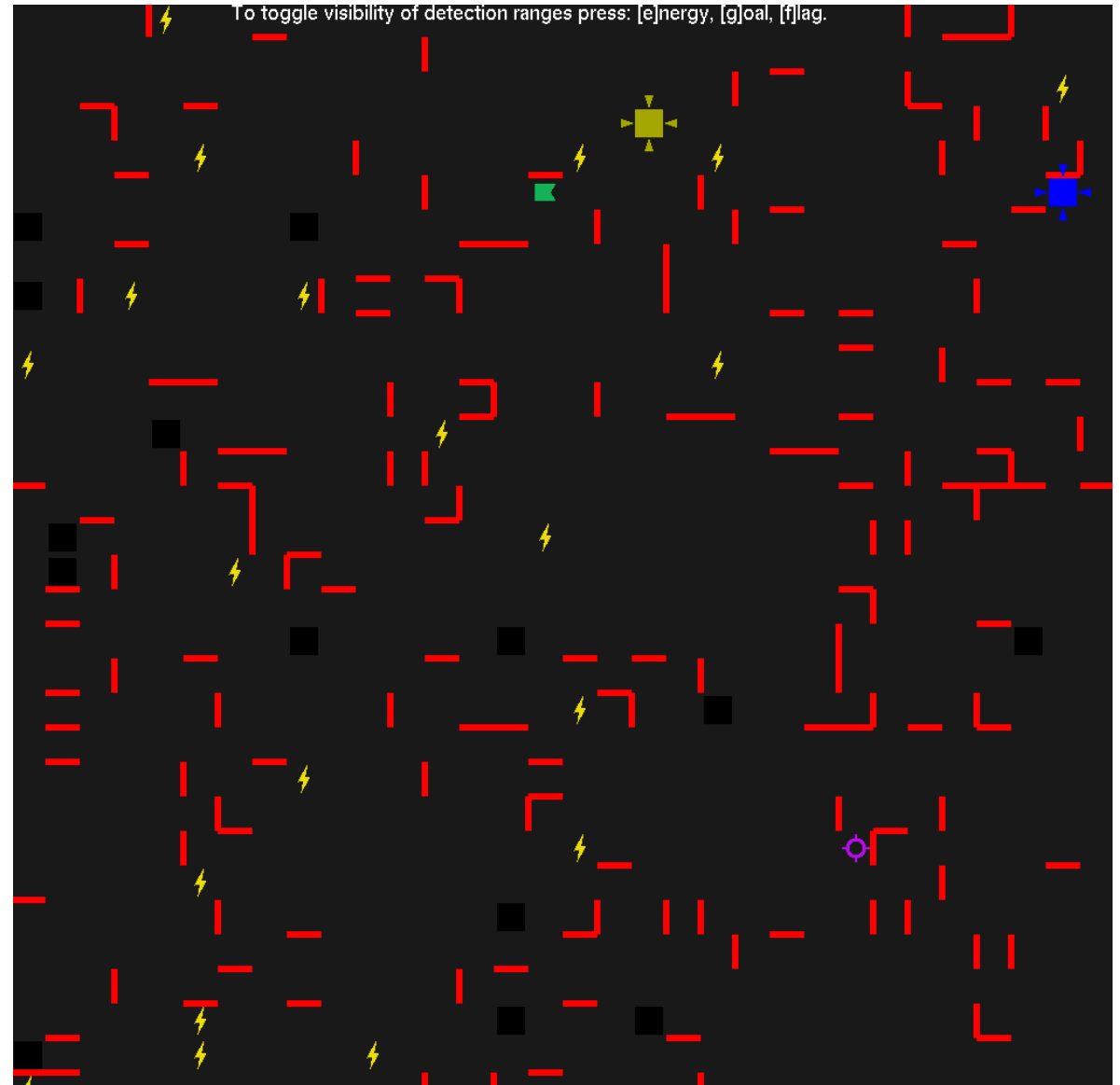
Goals:

- To learn how to write PLEXIL plans
- To fix and improve the Robosim simulator's graphics and functionality
- To help others who try to learn PLEXIL by providing an in-depth tutorial for the language.



Additions

- Graphics update for visual clarity
- Addition of a secondary objective in the form of a flag that can be picked up by the robot.
- Fix several bugs including one where the robot could move even without power!



End Result

- In-depth PLEXIL tutorial that makes use of the improved Robosim simulator to teach a new user PLEXIL.

Intro to PLEXIL

Plexil (Plan Execution Interchange Language) is a language used to represent plans of automation. To help new users learn the language without having a system to automate, we have created the robosim simulator. This tutorial will guide you through learning Plexil using this simulator.

Throughout this tutorial links to the [Plexil Documentation](#) will be provided so you can read more about the topics we explore.

Outline

1. Starting the simulation
2. Your first action and command
3. Compile and run a PLEXIL plan
4. Moving a robot
5. Adding Intelligence
6. Library calls
7. Further exploration

1. Starting the simulation

Once you have Plexil installed you can navigate to `$PLEXIL_HOME/examples/robosim` and run:

```
$ ./robosim start
```

This command runs an IPC server and the robosim simulation in the background of your terminal. You should see the robosim application launch, this is the environment we'll be working with for this tutorial.

To restart the simulation or stop both the simulation and the IPC server run

```
$ ./robosim restart
```

or

```
$ ./robosim stop
```

Troubleshooting

If the application fails to launch you may have to run

```
$ make
```

Project 2 – Compilation Script Improvements

Goals:

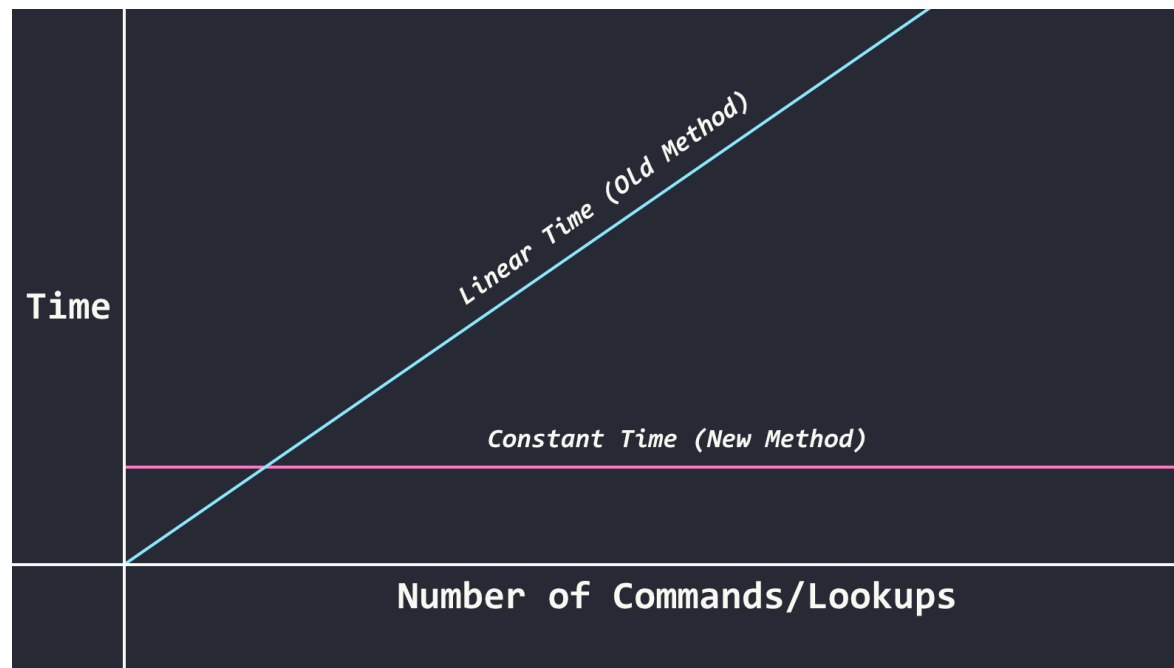
- To improve the PLEXIL compilation script to allow the compilation of many PLEXIL files with a single command.

```
bryce@bwc-plexil:~/plexil/test/TestExec-regression-test/plans$ plexilc *.ple
Compiling file: 'AncestorReferenceTest.ple'
Compiling file: 'ArrayAssignmentWithFailure.ple'
Compiling file: 'ArrayEquality.ple'
Compiling file: 'AssignFailureWithConflict.ple'
Compiling file: 'AssignmentFailureTest.ple'
Compiling file: 'AssignToParentExit.ple'
Compiling file: 'CommandCleanupTest.ple'
Compiling file: 'InactiveAncestorInvariantTest.ple'
Compiling file: 'Increment-test.ple'
Compiling file: 'lib1.ple'
Compiling file: 'lib2.ple'
Compiling file: 'libcall.ple'
Compiling file: 'NoChildFailedTest.ple'
Compiling file: 'NonLocalExit.ple'
Compiling file: 'repeat2.ple'
Compiling file: 'RoundTest.ple'
Compiling file: 'unknown_lookup.ple'
bryce@bwc-plexil:~/plexil/test/TestExec-regression-test/plans$ plexilc AssignmentFailureTest.pl
AssignmentFailureTest.ple  AssignmentFailureTest.plx
bryce@bwc-plexil:~/plexil/test/TestExec-regression-test/plans$ plexilc AssignmentFailureTest.ple lookup1.pli unknown_lookup.ple
Compiling file: 'AssignmentFailureTest.ple'
Compiling file: 'lookup1.pli'
Using schema ~/plexil/schema/core-plexil.rnc
Compiling file: 'unknown_lookup.ple'
bryce@bwc-plexil:~/plexil/test/TestExec-regression-test/plans$
```

Project 3 – Lookup/Command Handling

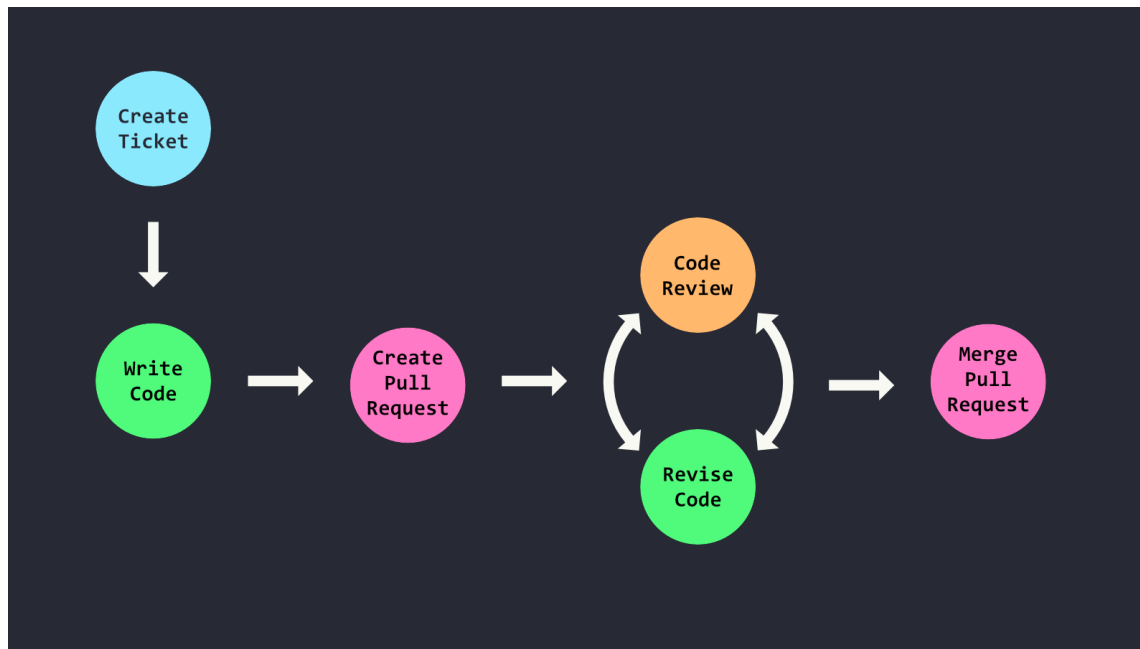
Goals:

- The lookup and command systems are used by PLEXIL plans to send and receive information and commands from the autonomous system. By letting the PLEXIL Executive do more of the work, developing these interfaces can be made easier.
- Shift lookups and commands from an if/else conditional approach to an efficient, hash-table, and handler based approach for dispatching commands and lookups.



Lessons Learned

- How to write autonomy plans.
- Improved knowledge of coding in C++
- How to work around project constraints like our use of the C++98 standard.
- How to create and work on tickets to track coding progress.
- How to use code reviews to discuss and debug code changes with mentors.



Acknowledgments

Mentors: Dr. Jeremy Frank, Chuck Fry, and Michael Dalal for all their guidance throughout my internship.

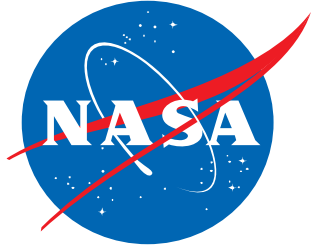
Funding: Universities Space Research Association

Teammates: Devin Wright, Albert Kutsyy, and Alina Kim

References

- Conversations with: Chuck Fry, Michael Dalal, and Dr. Jeremy Frank
- PLEXIL Website/Documentation: <http://plexil.sourceforge.net/>
- Dowek, Gilles et al. “A Formal Analysis Framework for PLEXIL.” (2007).
- Dowek, Gilles et al. “A SMALL-STEP SEMANTICS OF PLEXIL” (2008-11).

Questions?



RSE ROVER PLATFORM

Summer 2020

Thomas Cannon, University of California, Santa Cruz

Nathaniel Benz, ARC-TI, Mentor

- ▶ 10 hobby grade RC rovers
- ▶ ArduRover Autopilot + Raspberry Pi companion computer running NASA's Core Flight Software (cFS)
- ▶ Simulated lunar environment
- ▶ Currently untitled, goes by many names (RSE Rover Platform, Multi Rover Project, Mini Rover Project)

WHAT IS THE RSE ROVER PLATFORM?

- ▶ Relatively cheap and easy to put together
- ▶ Allows us to build platform for testing multi rover autonomy
- ▶ Many tasks multiple rovers can solve better than an individual one:
 - ▶ Determine most efficient path to goal
 - ▶ Minimize battery usage
 - ▶ Quickly map out targets for exploration
 - ▶ Avoid damage to hardware by transmitting info between rovers
 - ▶ Radio ranging, determine rover positions

WHY IS THE PROJECT IMPORTANT?

- ▶ Must use simulation due to COVID19
- ▶ Using Microsoft Research's AirSim simulator
 - ▶ Not used by NASA before
 - ▶ Built on Unreal Engine 4 (UE4), a game engine developed by Epic Games
 - ▶ UE4 has high quality graphics, which we will use for computer vision research
 - ▶ Lots of prebuilt content available, allowing faster prototyping of simulation environments

SIMULATION



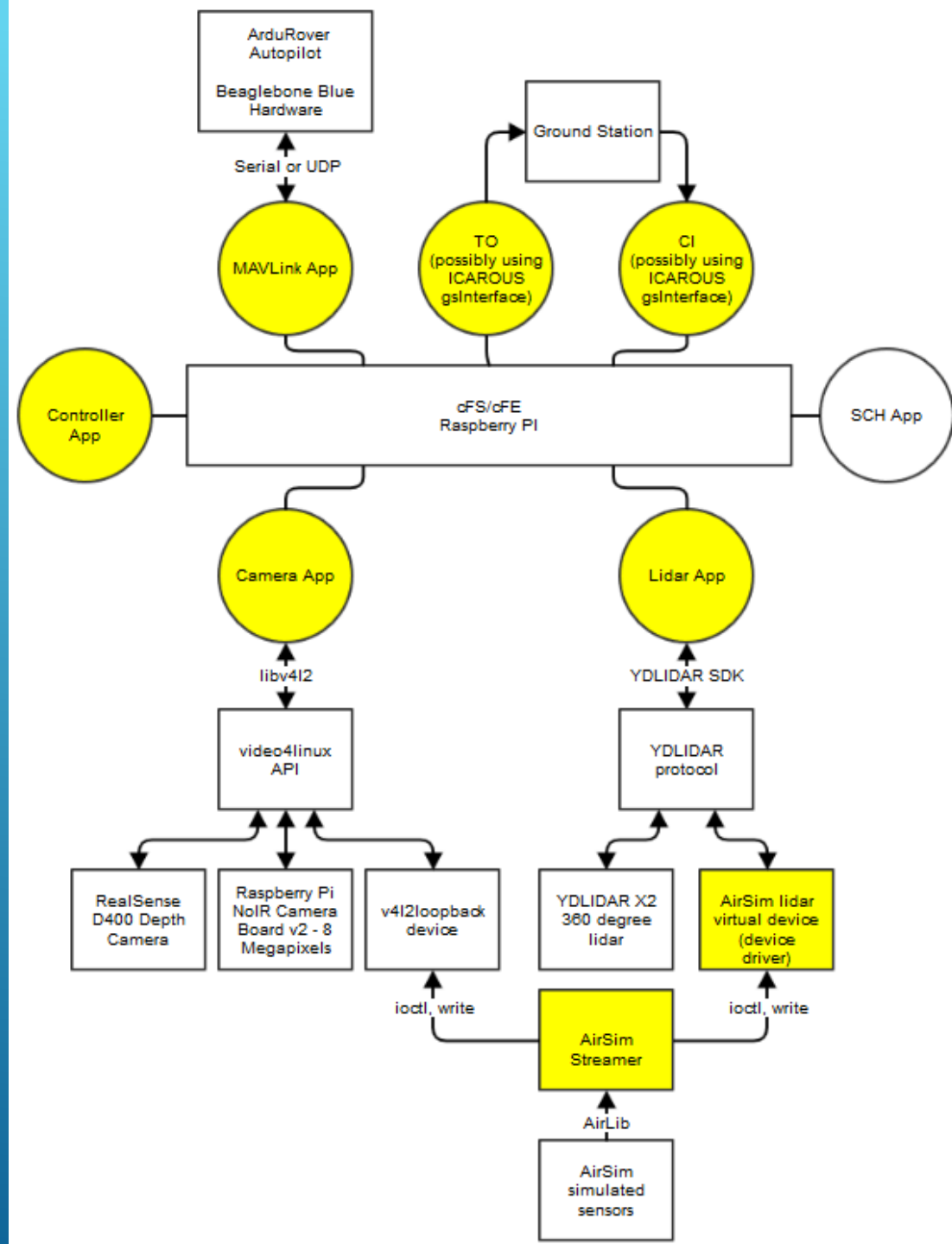


Screenshot taken by author

- ▶ Internship focused on setting up simulator and foundation rover software for future research
- ▶ Created basic rover model based on rover measurements, imported into AirSim environment
- ▶ Script for exporting from FreeCAD into UE4
- ▶ Scripts for setting up Software in the Loop (SITL) environment
- ▶ Initial rover software using cFS

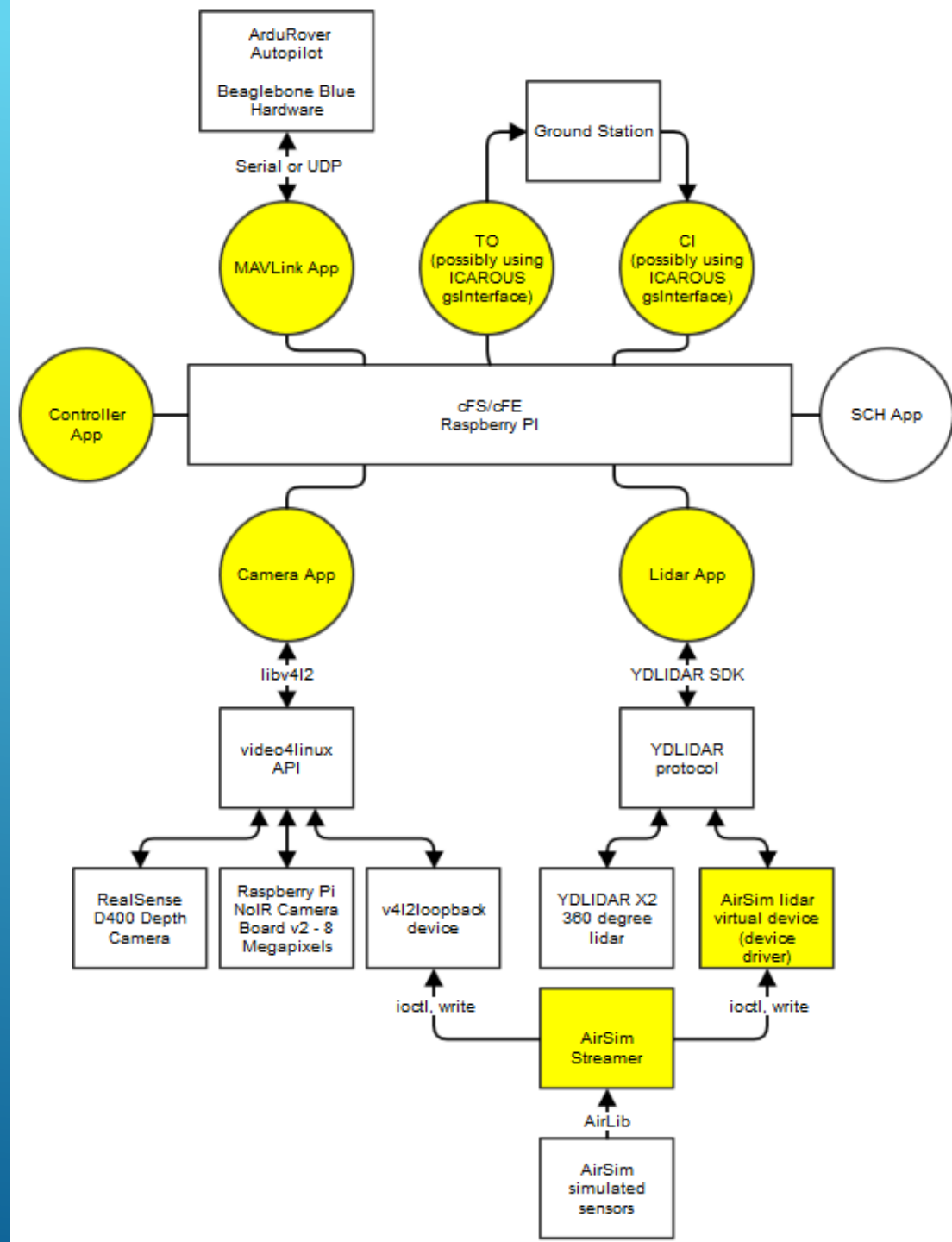
WHAT DID I WORK ON?

- ▶ Using cFS running on Linux (Raspberry Pi)
- ▶ Yellow components are project specific



ROVER SOFTWARE

- ▶ Camera App
- ▶ Modified ICAROUS MAVLink App
- ▶ AirSim Streamer
- ▶ Lidar App (in progress)
- ▶ AirSim lidar virtual device (in progress)



COMPONENTS I CREATED

- ▶ Put together the physical rovers
- ▶ Finish adding sensor I/O apps
- ▶ Add Telemetry Out (TO) and Command In (CI) apps
- ▶ Begin multi rover research

WHAT'S NEXT?



WHAT DID I LEARN?

- cFS and tools
- CMake, Doxygen
- AirSim, Unreal and FreeCAD
- Linux device driver basics

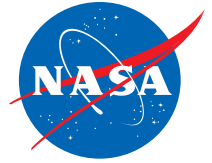
- ▶ Nathan Benz – Mentor
- ▶ Ritchie Lee – cFS design advice
- ▶ Rory Lipkis – Developed simulated environments, AirSim help
- ▶ Scott Christa – Advice about simulating sensors
- ▶ Krystine Carrington – Advice about cFS Camera App
- ▶ Universities Space Research Association (USRA) and ARC Internship Coordinators

ACKNOWLEDGEMENTS

- ▶ Please feel free to contact me:
 - ▶ Thomas W. Cannon (ARC-TI)[UNIVERSITIES SPACE RESEARCH ASSOCIATION]
 - ▶ thomas.w.cannon@nasa.gov (until September 18th)
 - ▶ thwcanno@ucsc.edu
 - ▶ (408)630-0454

QUESTIONS?

A decorative graphic consisting of several parallel white lines of varying lengths, slanted upwards from left to right, located in the bottom right corner of the slide.



Aircraft System Identification and Flight Control Optimization Curriculum Development in Cal Poly Pomona

Sung Hyeok Cho

University: Cal Poly Pomona

Mentor: Kenny K. Cheung, Aeroflightdynamics

Summer 2020

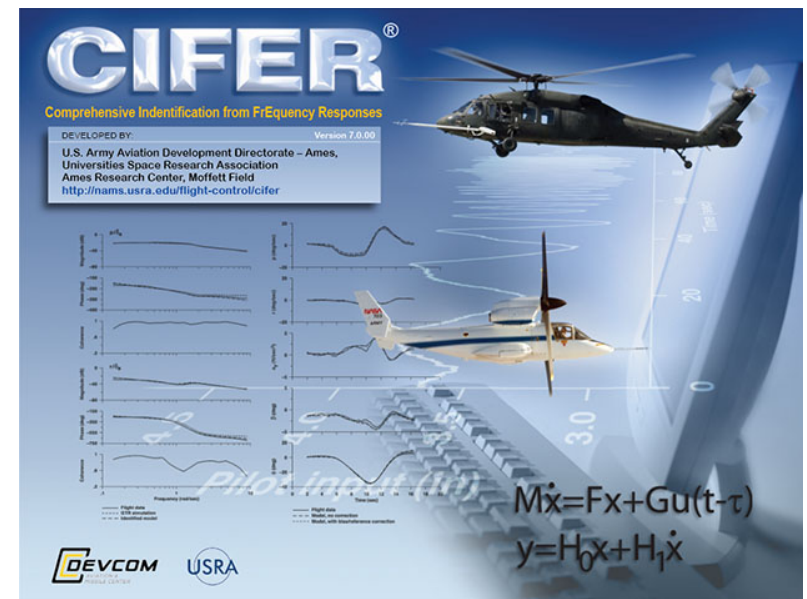
Introduction

- CPP Aerospace Engineering curriculum includes many of the standard topics
- With emphasis on learn-by-doing philosophy, CPP UAS Lab conducts research and supports project teams
 - Collision detection and avoidance
 - Indoor search and rescue using UASs
 - Collaboration between multiple unmanned vehicles
 - Various senior projects on UASs



Goals of Curriculum Development

- Development of graduate and undergraduate level courses on Aircraft System Identification and Flight Control Optimization using **CIFER**[®] and **CONDUIT**[®] software suites
- Introduction of the elements of these topics into the existing courses
- Integration of **CIFER**[®] and **CONDUIT**[®] software capabilities into UAS Lab

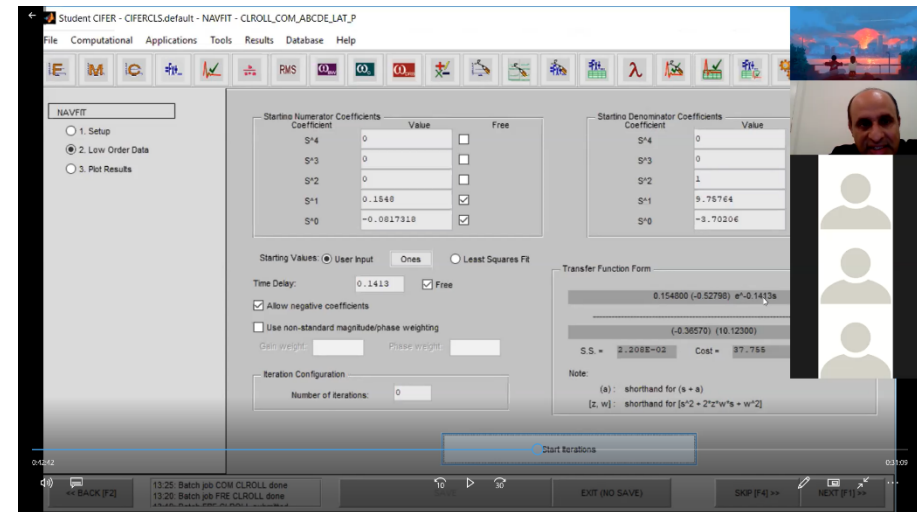


Developed Curriculum

- Graduate and undergraduate courses are now taught in Cal Poly Pomona
 - Aircraft System Identification (ARO 5430, ARO 4430)
 - Flight Control Optimization (ARO 5520, ARO 4520)
- Elements of these courses were introduced to Aircraft Stability and Control and Aerospace Feedback Control Systems classes for undergraduate students

Aircraft System Identification Course

- ARO 5430 (Graduate Level) and ARO 4430 (Undergraduate Level)
- 13 students have taken ARO 5430 (Summer 2020)
- Topics taught:
 - Instrumentation and data collection
 - Frequency domain identification technique
 - SISO and MIMO model identification
 - UAV system identification applications



Flight Control Design Course

- Course Number: ARO 5520 (Graduate Level) and ARO 4520 (Undergraduate Level)
- 10 students have taken ARO 5520 (Fall 2018)
- Topics taught:
 - Design specifications and source of design requirements
 - Simulation requirements for design
 - Preliminary design
 - Design tradeoffs
 - UAV control system design application

Outcomes of Curriculum Development

- Collaboration with USRA has added new capabilities to the UAS Lab of Cal Poly Pomona
- These capabilities helped faculty and student conduct state-of-the-art research on system identification and flight control design
- System identification and control design became a new area of focus for the UAS Lab
 - Senior project for identification and optimal control design of blended wing body UAV
 - Two graduate students and several undergraduate students are involved in other projects

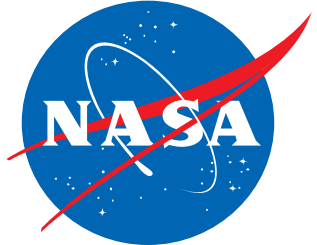


Acknowledgments

- Universities Space Research Association
- Cal Poly Pomona Aerospace Engineering department



Questions





PLEXIL Decompiler

Alina Kim, University of California, Irvine

Fry, Charles; Frank, Jeremy; Dalal, Michael; Iatauro, Michael, ARC-TI

Summer 2020



What is PLEXIL?

- A highly deterministic language
- Used primarily in automation
- Allows highly discrete control over the execution flow of the language
 - Typical control structures like if / while / for are all external constructs
- Used to automate the K10 Rover, the Mars Drill, and used to demonstrate automation on the International Space Station



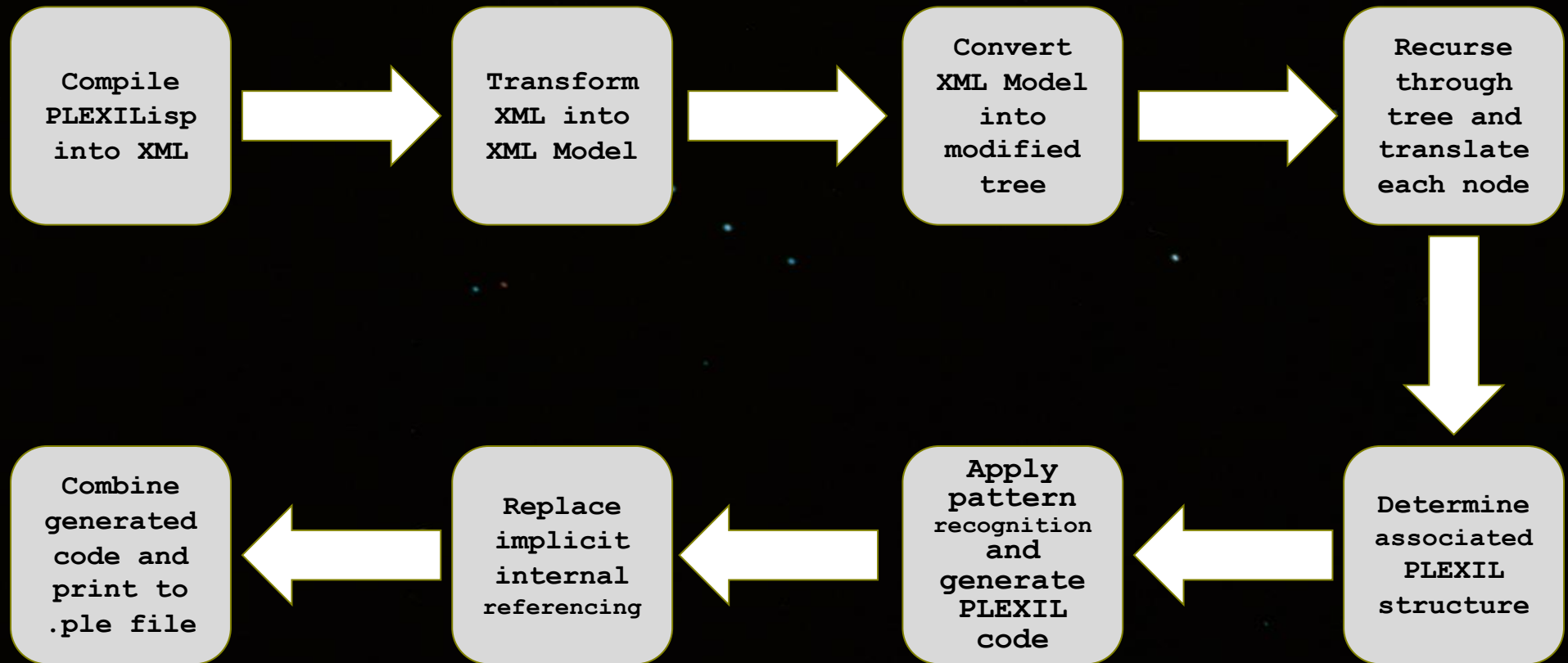
Problem Space

- PLEXIL plans (.ple files) compile into XML (.plx)
- There are few different “versions” of PLEXIL
 - Standard Plexil, written in a C-like syntax
 - Plexilisp, written in a lisp-like syntax
 - Extended Plexil, an XML representation that the aforementioned versions compile to
 - Core Plexil, an XML format which is understood by the PLEXIL Executive, and is expanded to from Extended Plexil
- Plexilisp is now deprecated, and there is a need to convert existing Plexilisp files back into Standard Plexil
- There are no maintained ways to translate in between languages
- However, they all ultimately compile to the same Core Plexil XML format

Proposed Solutions

- Brute force: Manually translating all existing Plexilisp into Standard Plexil
 - Straightforward
 - Not scalable
- Translator: A program to translate Plexilisp into Standard Plexil
 - Relatively time-consuming endeavor
 - Only highly specific applications
- Decompiler: A program to translate Core Plexil into Standard Plexil
 - About as difficult as writing a translator
 - More widely applicable; will still be useful moving forward

Approach



Results

- Coverage extends beyond original goals
 - Effectively all PLEXIL structures are covered by the Decompiler
 - Only a couple minor pattern recognition bugs exist
- Capable of decompiling virtually any properly compiled, error-free code
 - There exist a few edge-cases due to the way naming conventions are handled in PLEXIL
- Decompiles to Standard Plexil, but most of the structures are reduced to their underlying condition-based logic
- Approximately 75% of compilable Plexilisp files have now been successfully translated to Standard Plexil
- Also implemented a program to navigate through XML trees like a directory to aid with the development process

Moving Forward

- Decompile completely into Standard Plexil while either reducing all structures to their condition-driven logic, or recreating all of their original structures for the sake of consistency
- Make pattern recognition suite modular, allowing it to be extended alongside the Plexil language's evolution in the future
- Automate decompilation script

Takeaways

- Through this internship, I had the opportunity to work on a specified, large-scale project that required a good deal of software engineering
- Had to learn the inner workings of PLEXIL in an in-depth manner
- Gained experience working on the ground level of a programming language
- Learned about working with a mentor-- striving to balance out the help I receive with the help that I can give

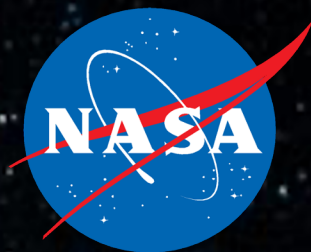
Acknowledgements

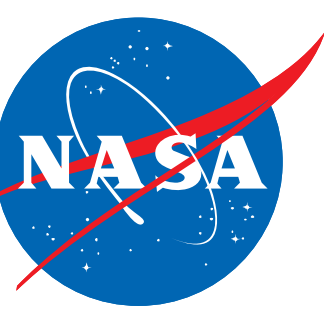
- Thank you to my mentors-- Michael, Chuck, The Other Michael, and Jeremy, for helping me throughout this process, providing guidance and encouraging me to ask for help when I needed it
- Shoutout to my teammates as well for always having something interesting to say and/or report c:

References

- “Langley Formal Methods Program • César Muñoz • PLEXIL.” NASA, NASA, shemesh.larc.nasa.gov/fm/PLEXIL/.
- “Plexil Wiki.” Plexil, plexil.sourceforge.net/wiki.

Questions?

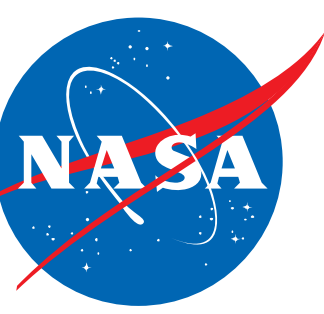




CFD Smorgasbord

Sonic-Booms, Computational Geometry, and Hypersonic Aerodynamics

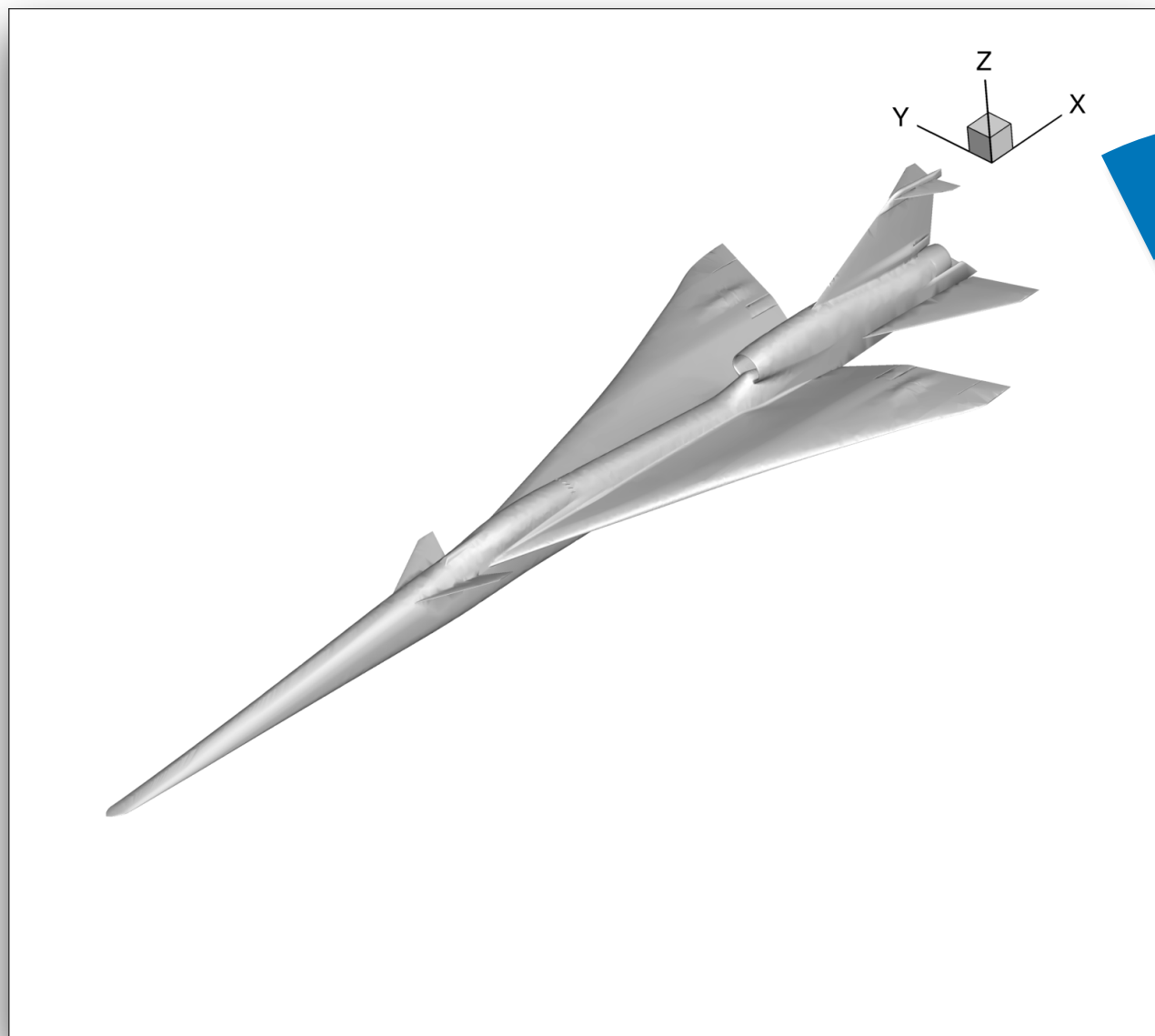
**Alex Kleb, University of Michigan, Ann Arbor
Summer 2020**



Mini Projects

- sBOOM/PCBoom Comparison
 - Error convergence
- Cart3D Geometry Manipulation Tools
 - Increased functionality
 - Python wrapper
- Hypersonic Glider Aerodynamic Analysis
 - Engineering Sketch Pad
 - Cart3D performance

Sonic Boom Prediction

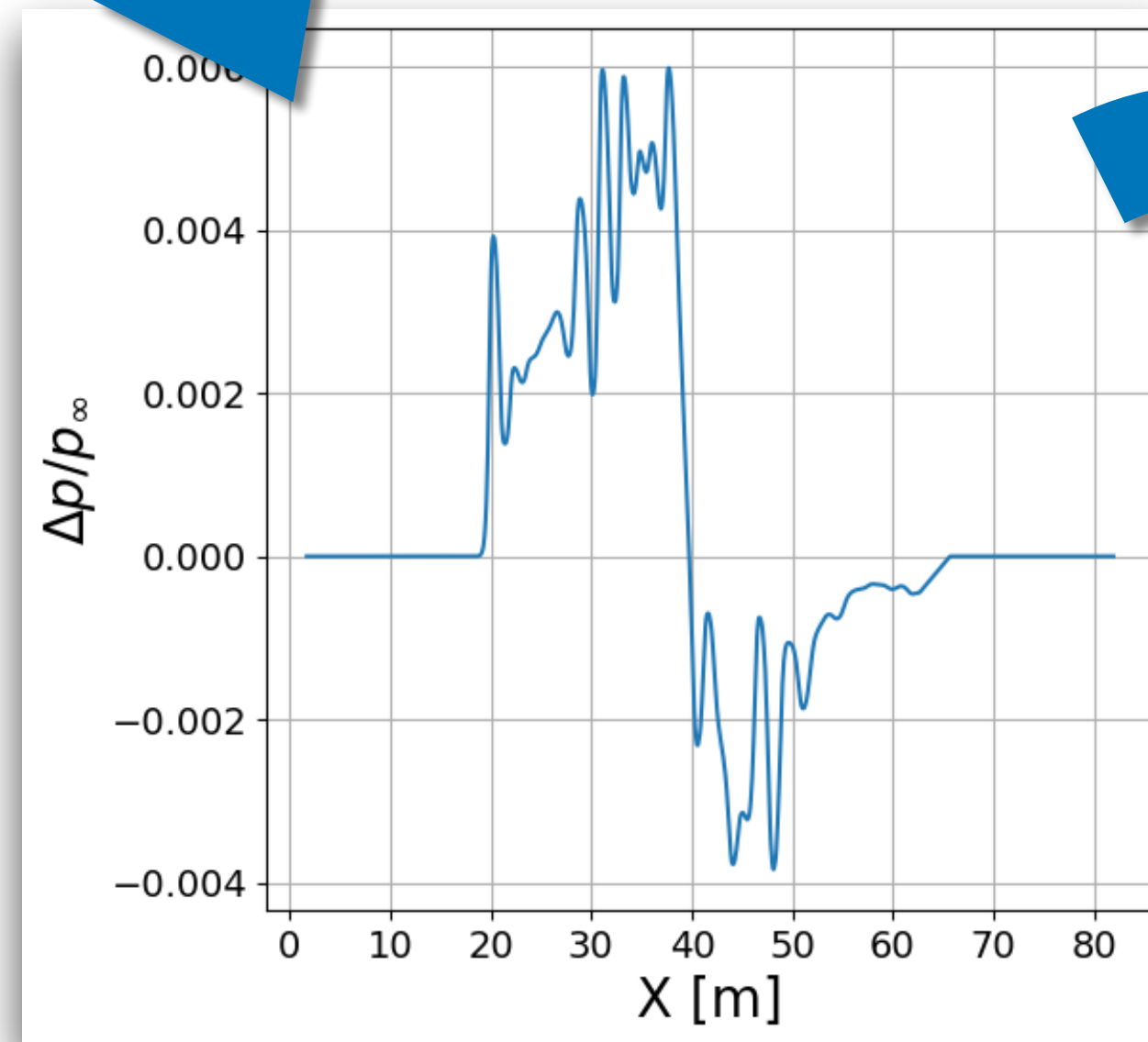


Geometry

C608 standard atmosphere on-track case from the 3rd AIAA Sonic Boom Prediction Workshop (SBPW)

1. Flow Solve

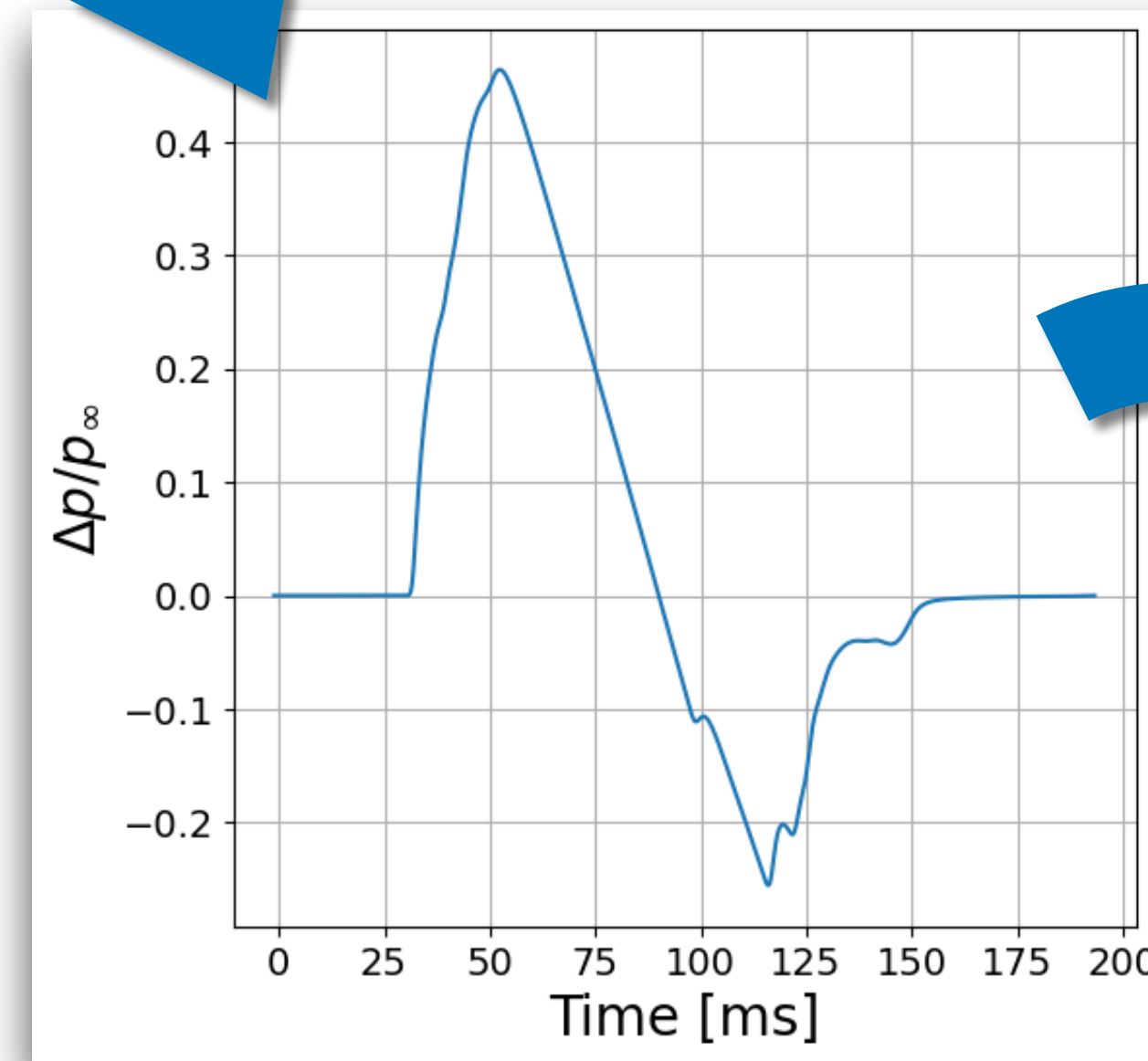
- Cart3D
- FUN3D



Near-field Pressure

2. Atmospheric Propagation

- PCBoom v6.7.1
- sBOOM v2.83



Ground Pressure

3. Apply Filters

- adloud
- sBOOM

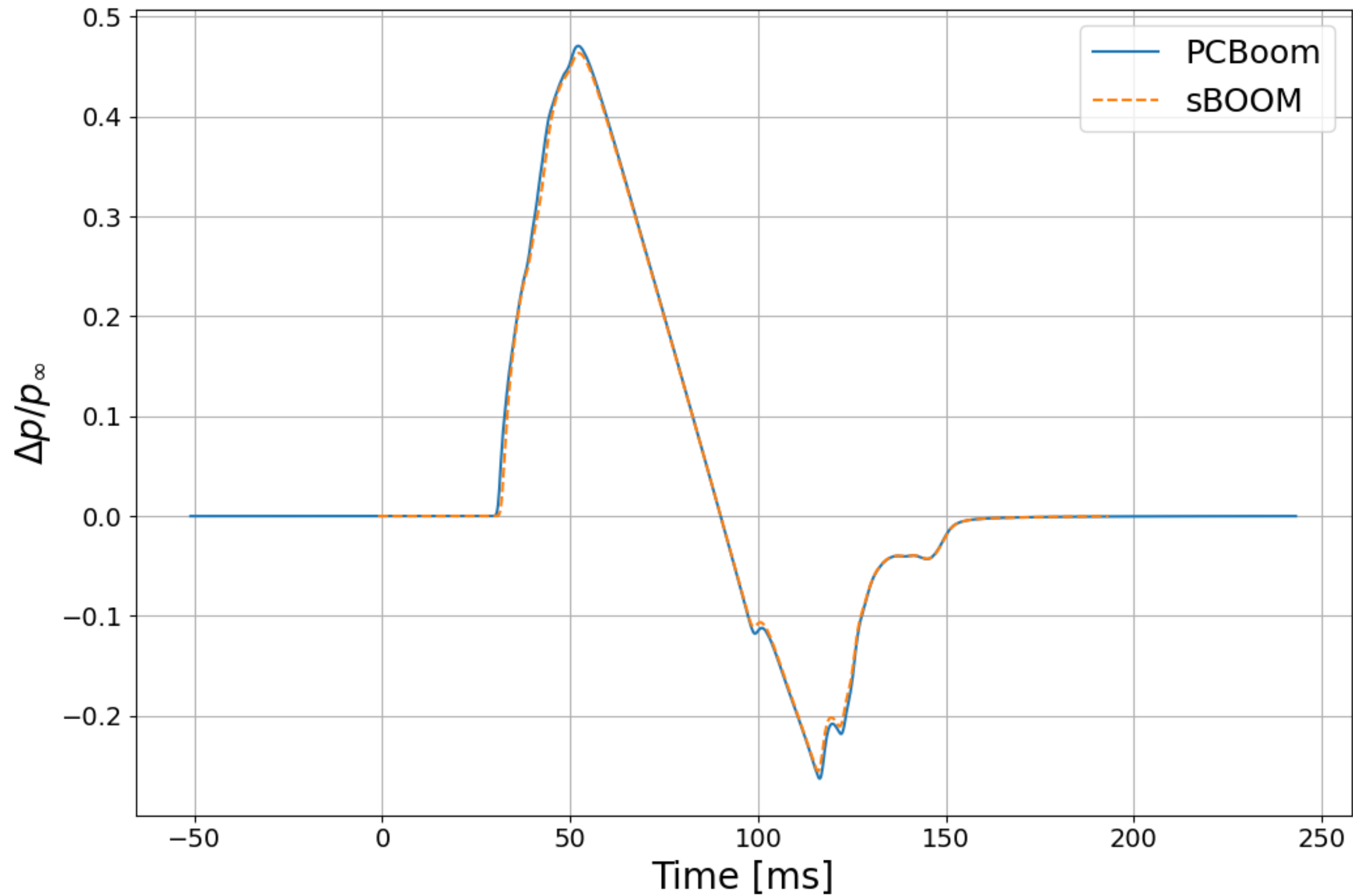
ASEL: 61.0 dB
BSEL: 74.8 dB
CSEL: 90.2 dB
PL: 75.3 dB

Noise Metrics

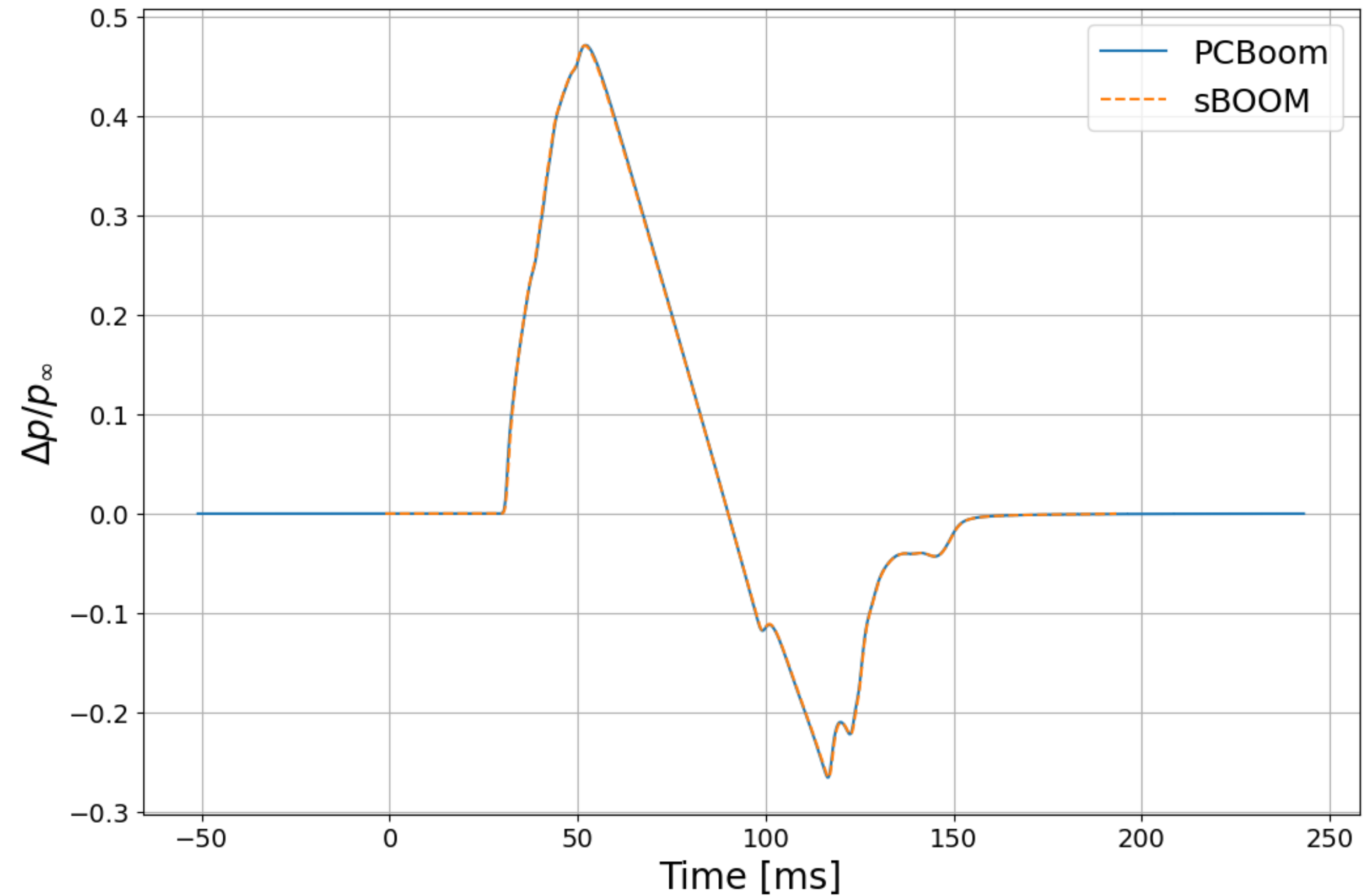
Ground Signal

Signals show good agreement at low sampling frequencies

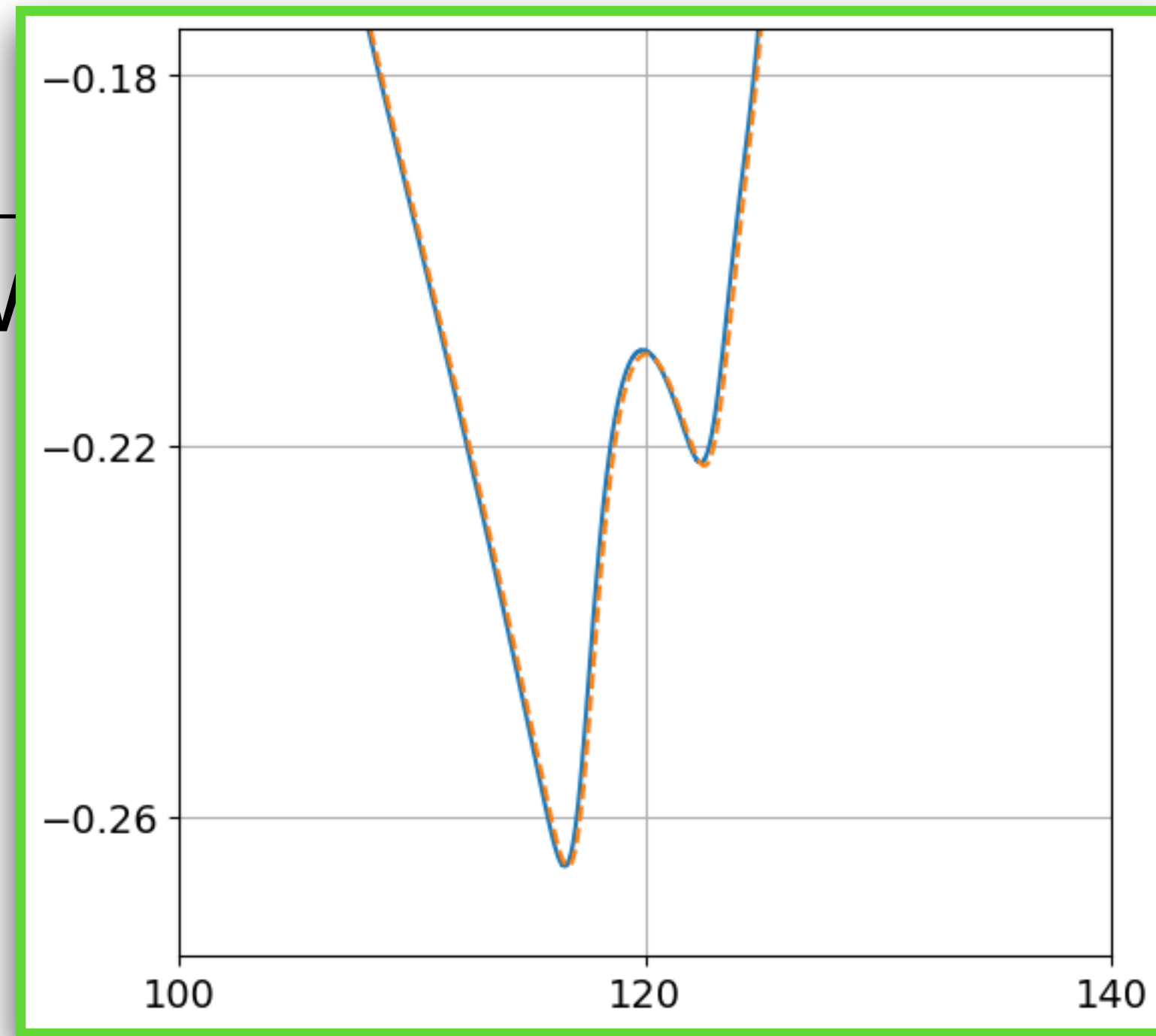
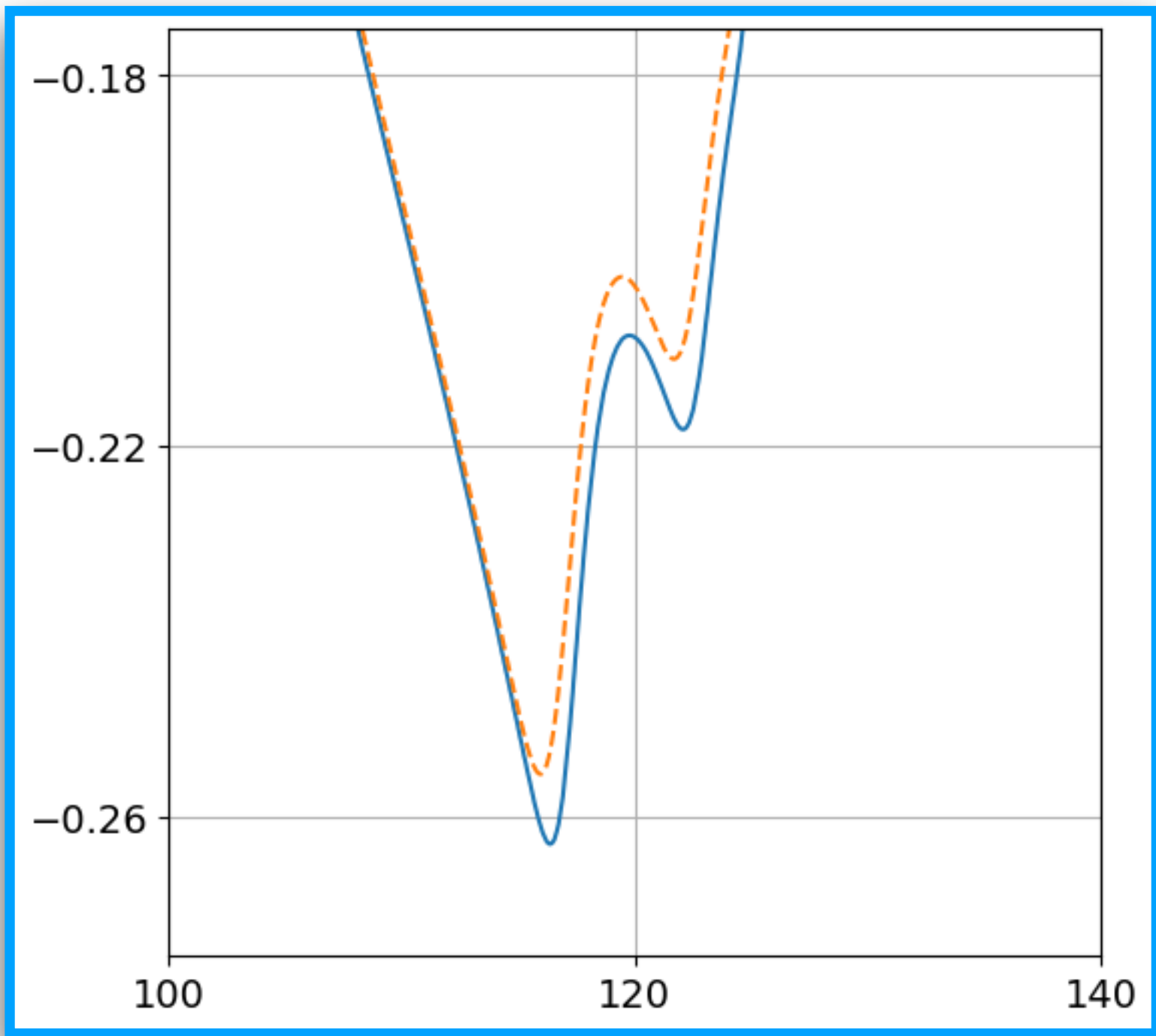
20 kHz (4,000 Points)



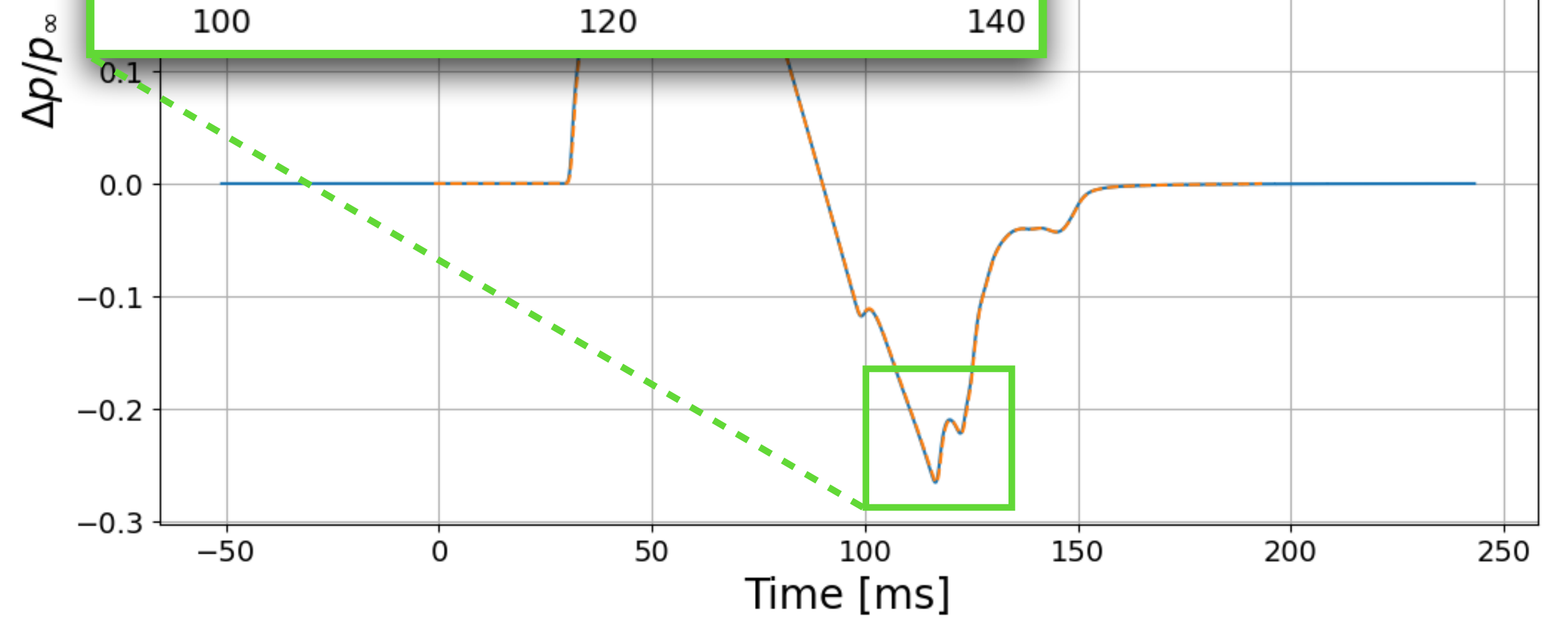
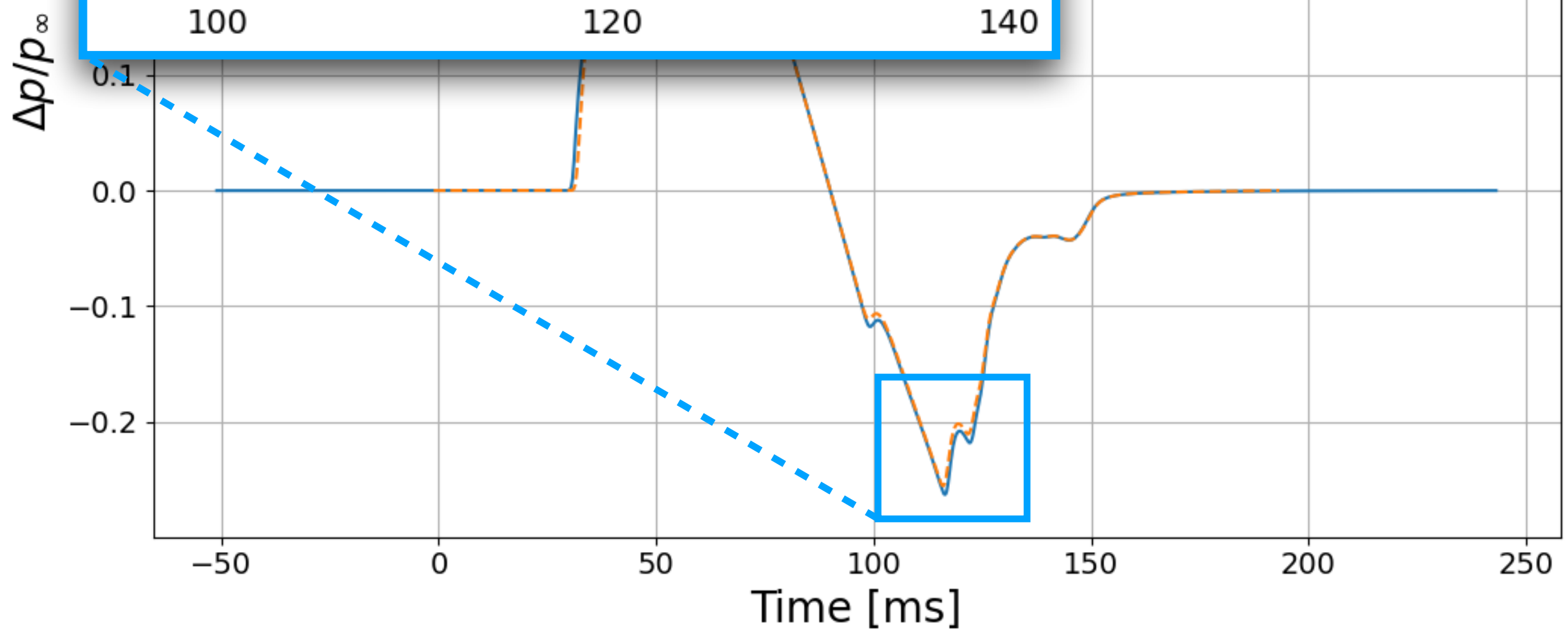
350 kHz (68,000 Points)



reement at low

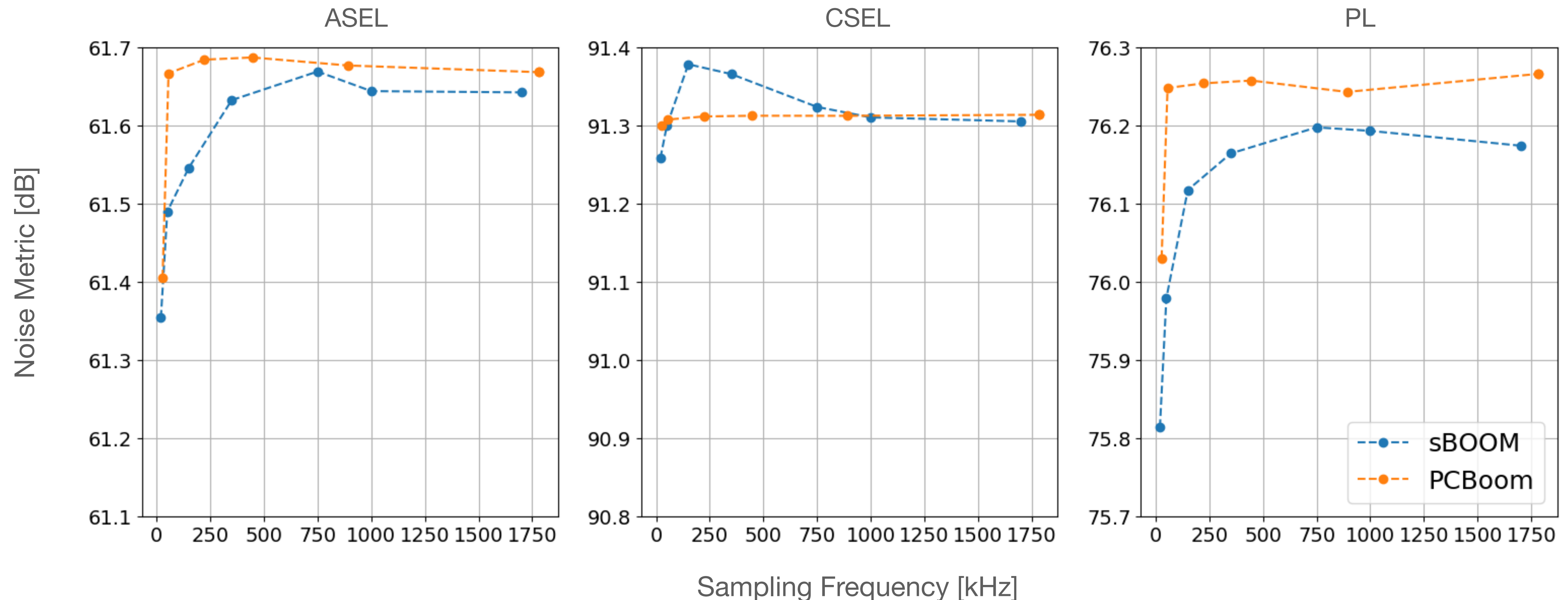


nts)



On-track Metric Convergence

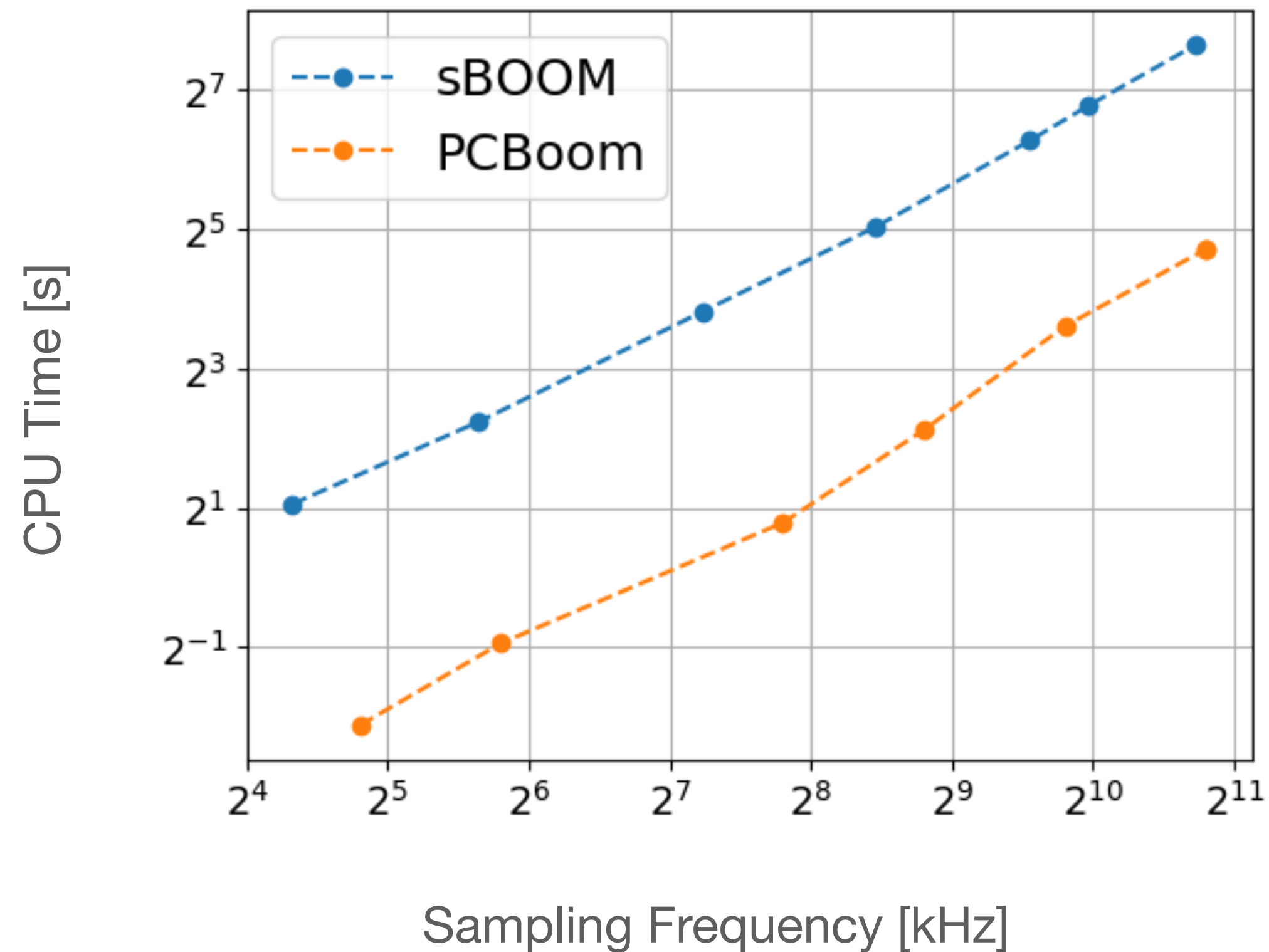
- Calculated with adloud (LCASB release 2009.02.27)
- PCBoom converging to ± 0.05 dB much faster than sBOOM
- PCBoom and sBOOM agree within ± 0.1 on a sufficiently fine mesh



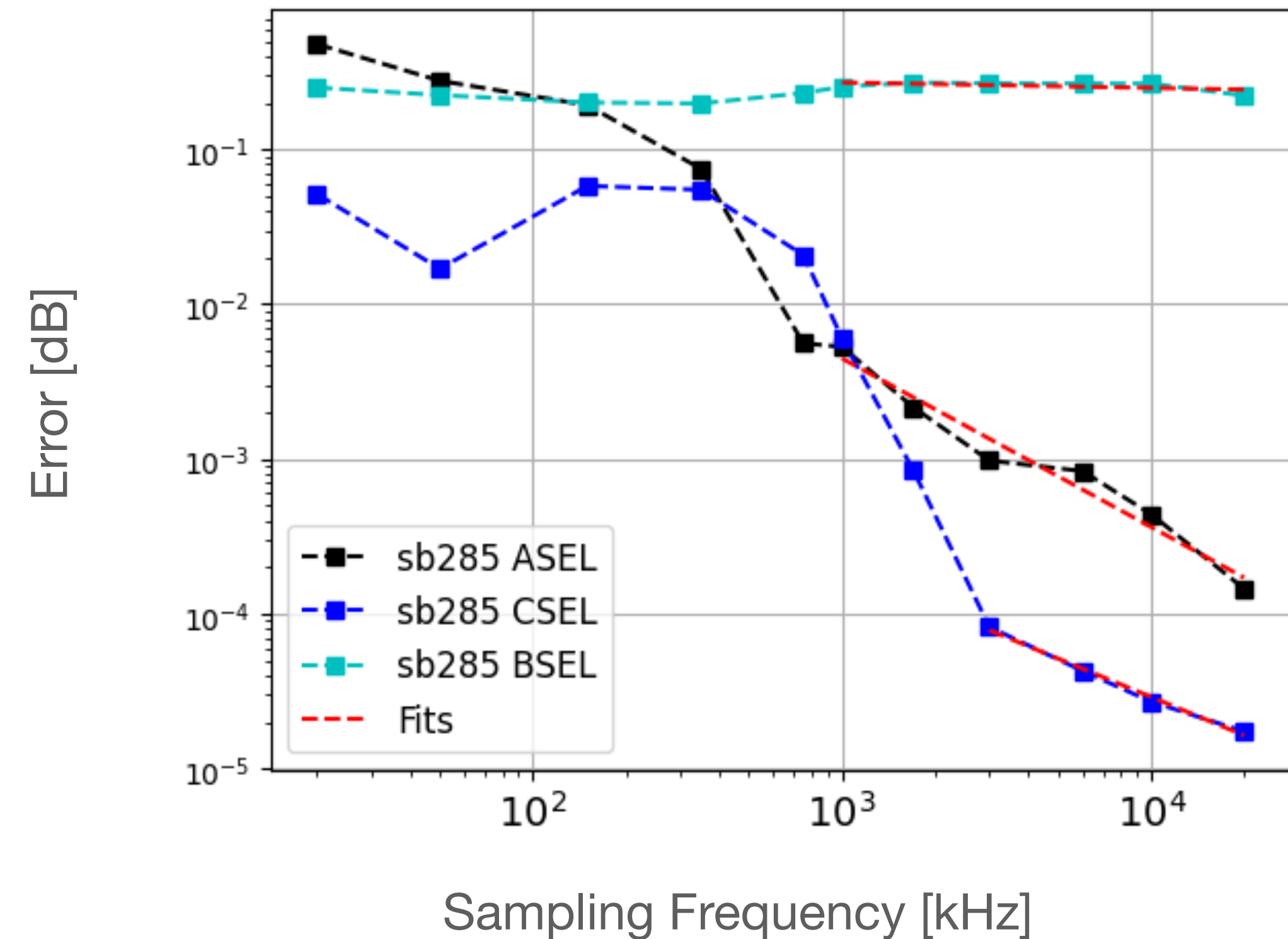
On-track Runtime and Mesh Convergence

- PCBoom consistently runs ~8 times faster than sBOOM
- PCBoom metrics did not converge with mesh refinement, sBOOM v2.85 does for two metrics
- 40,000 kHz run used as an exact solution with an L_1 error approximation

Runtime



sBOOM v2.85 Mesh Convergence



Cart3D Geometry Manipulation: Trix

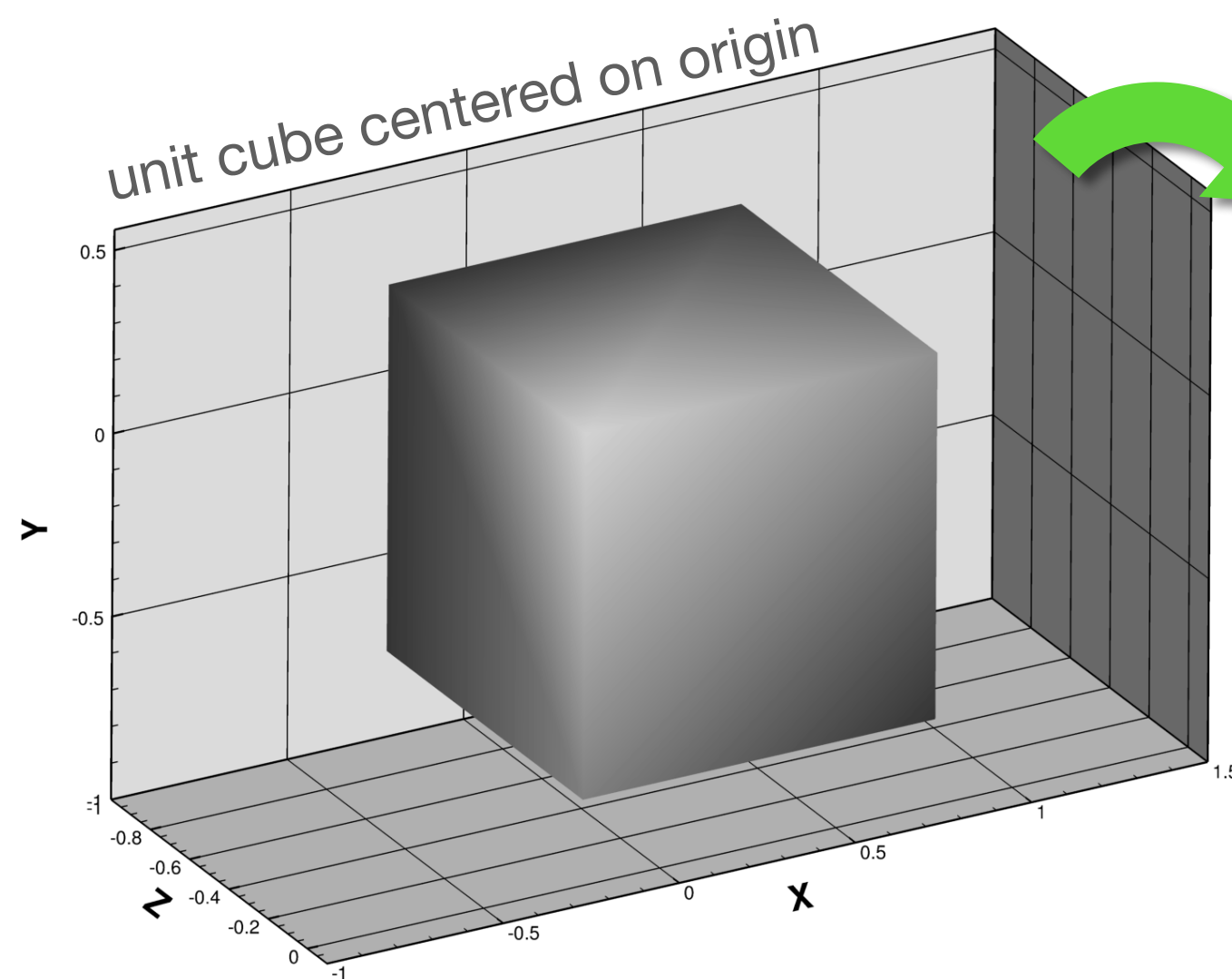
A Swiss army knife of geometry manipulations

Pros

- Performs a wide range of geometry manipulations: scaling, translating, rotating and mirroring
- Takes in different file formats

Cons

- No options to modify specific parts of the geometry
- No options for arbitrary axis rotations
- All manipulations must be preformed in the same order every time



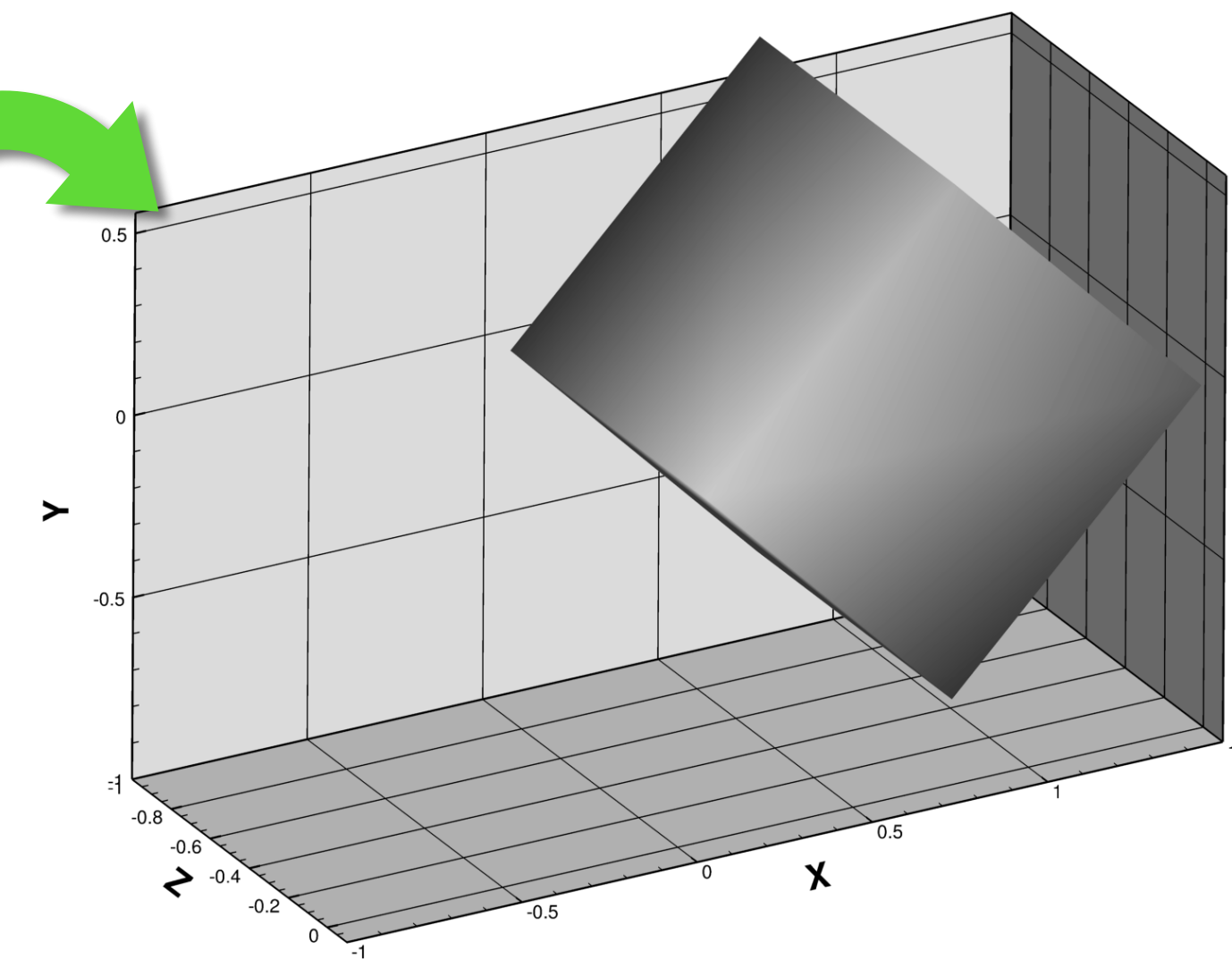
```
trix -cx 0.5 -cy -0.5 -rz 45 -T cube.tri
```

Center of Rotation: $\langle 0.5, -0.5, 0 \rangle$

Input File

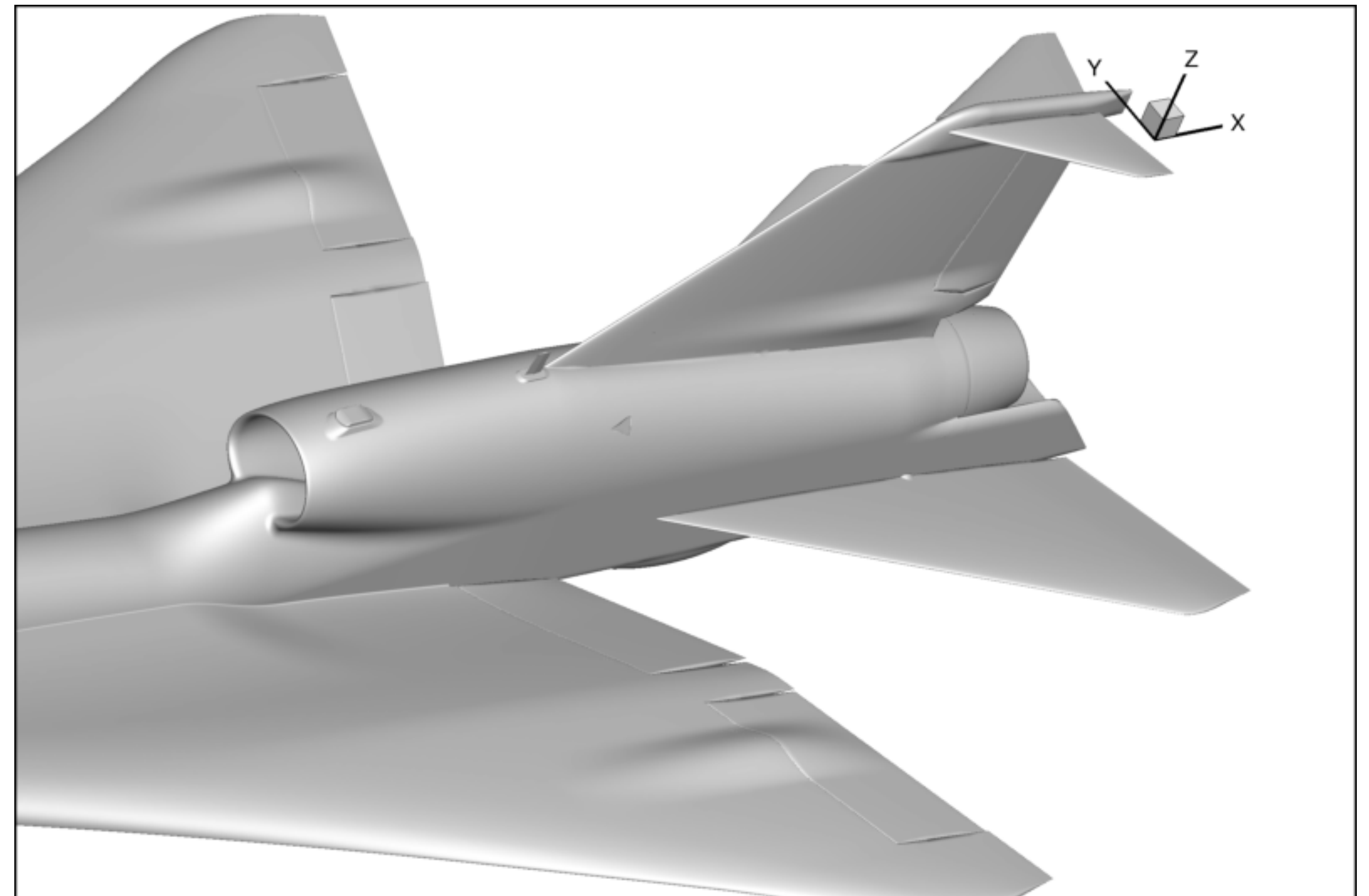
Rotate about z axis: 45°

Write out Tecplot File



Improved Functionality

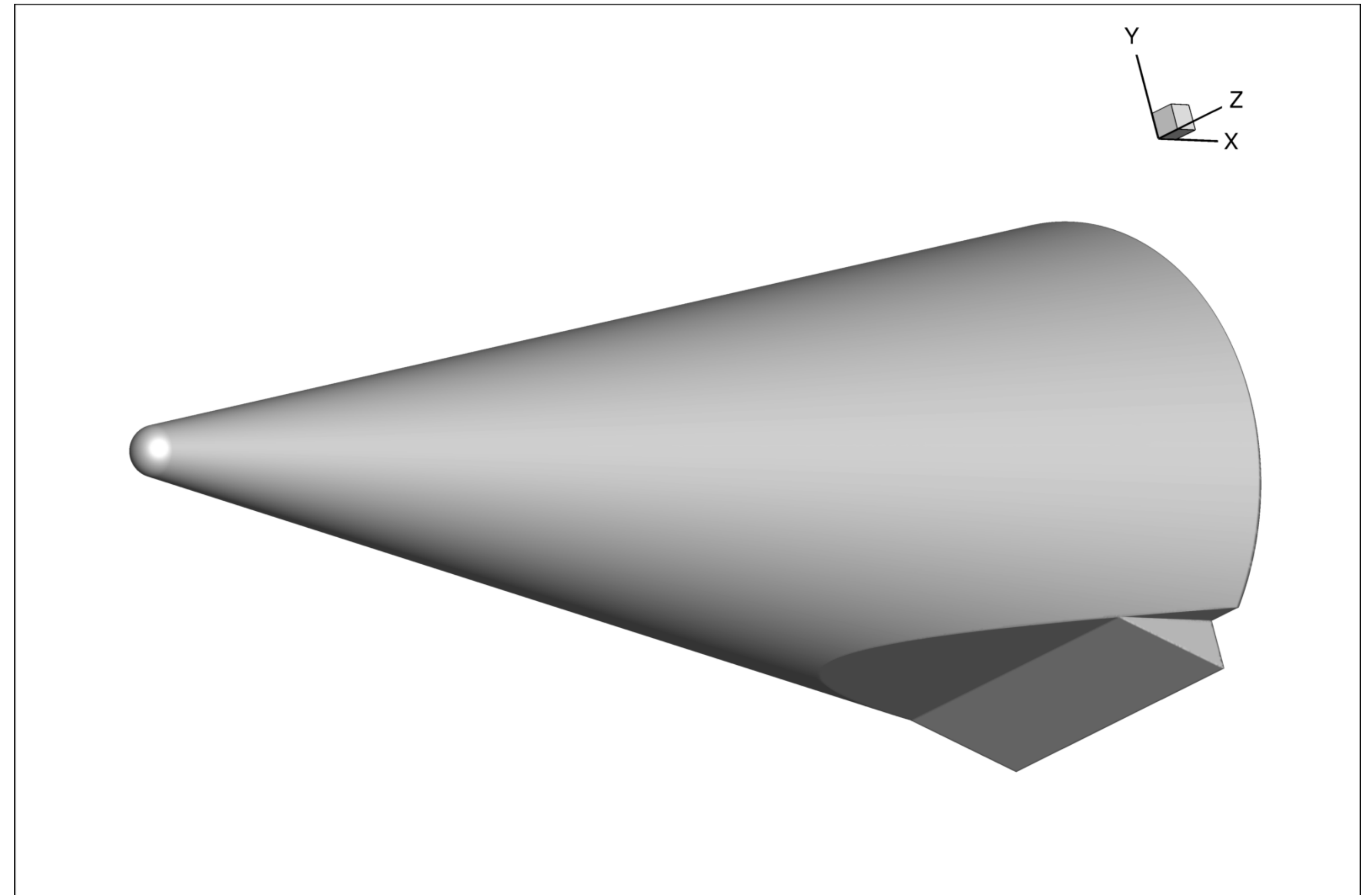
- Added individual component manipulation and arbitrary axis rotations
- Created a unit testing framework using Python's pytest and unittest
- Implemented a python wrapper for easier use



Old Flap Deflection Method	New Flap Deflection Method
<ol style="list-style-type: none"> 1. Separate flap from geometry 2. Use trix to translate the flap to the origin and rotate the flap 3. Use trix to translate the flap back 4. Reattach flap to original geometry 	<ol style="list-style-type: none"> 1. Use trix to rotate the flap about its hinge line

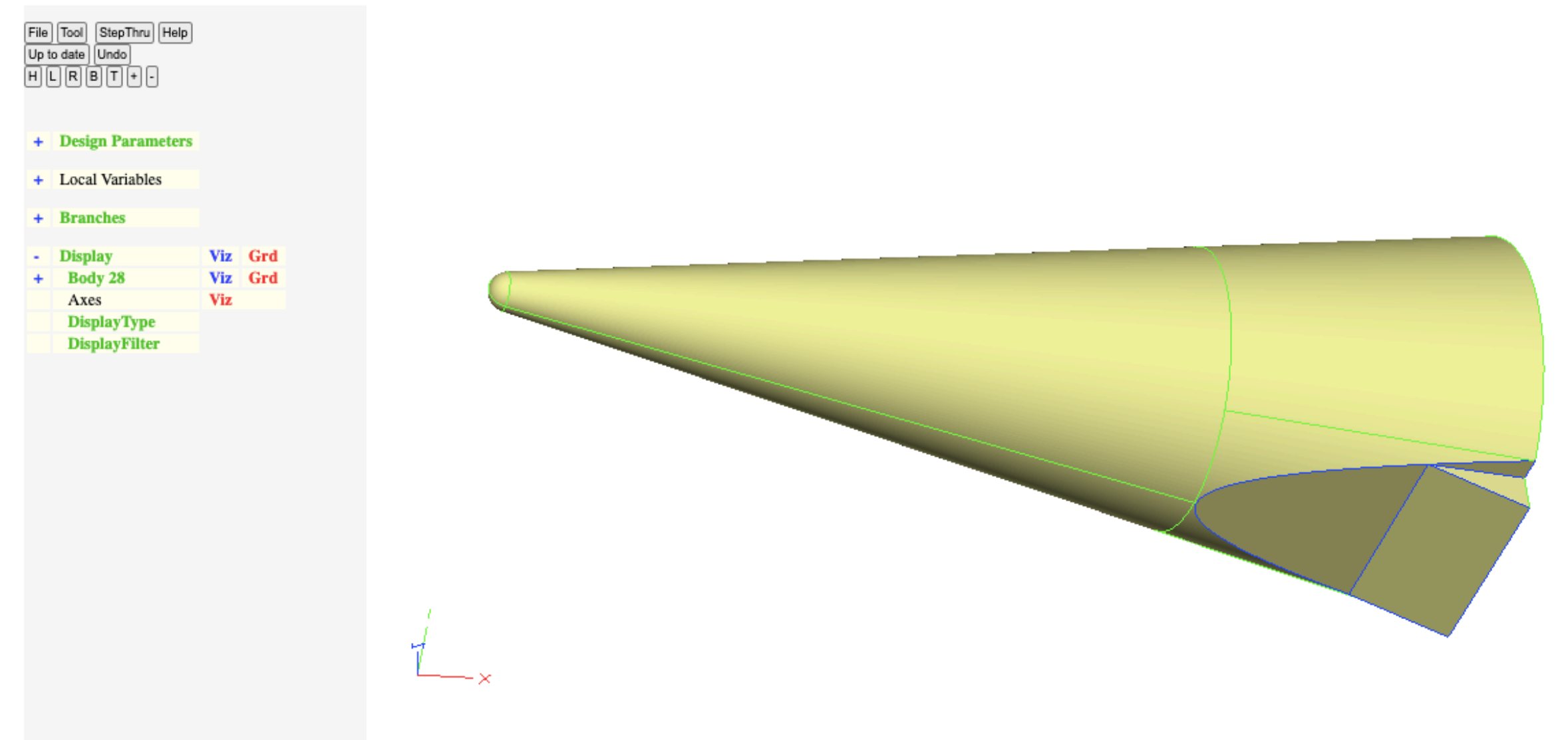
Hypersonic Glider Analysis

- Hypersonic glider configuration from Oberkampf and Aeschliman
- Geometry created with Engineering Sketch Pad (ESP)
- Compared Cart3D, laminar FUN3D solver, a Modified Newtonian model, and wind tunnel results

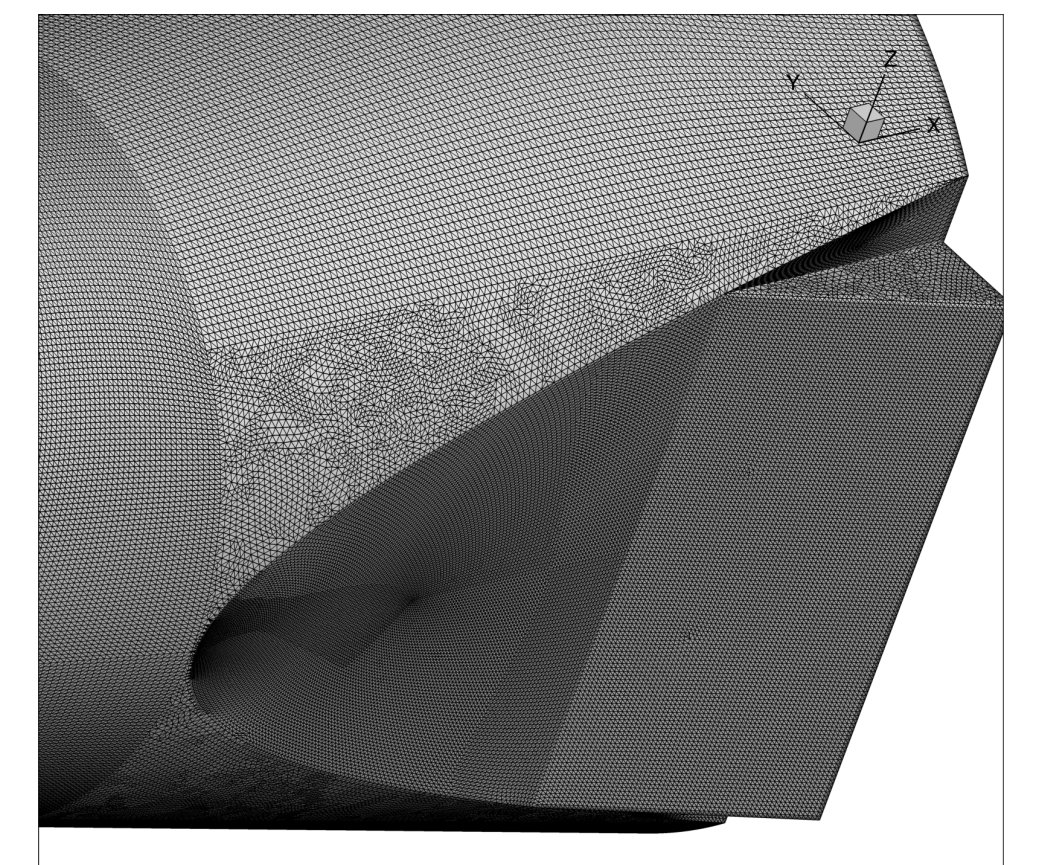
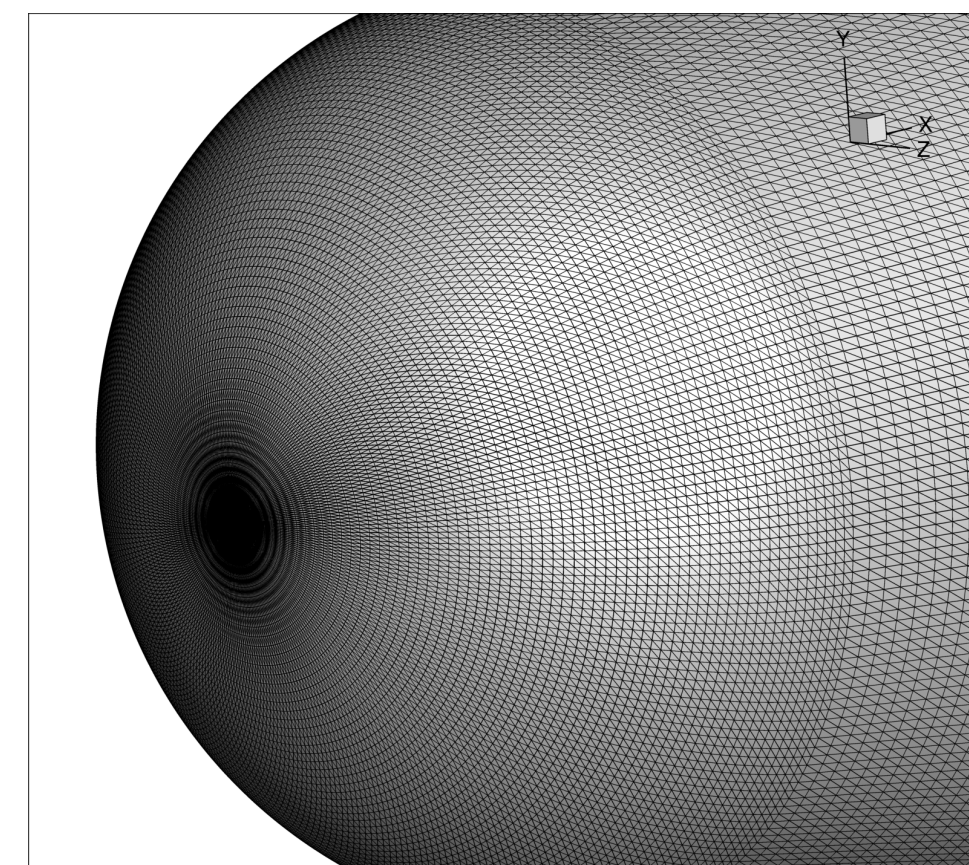


ESP

- An aerospace Multidisciplinary design optimization (MDO) tool
- Parametric geometry modeler: OpenCSM
- Surface tessellation: Electronic Geometry Aircraft Design System (EGADS)
- Set grid size and Cart3D component numbers in ESP
- Cart3D tri file created with egads2cart

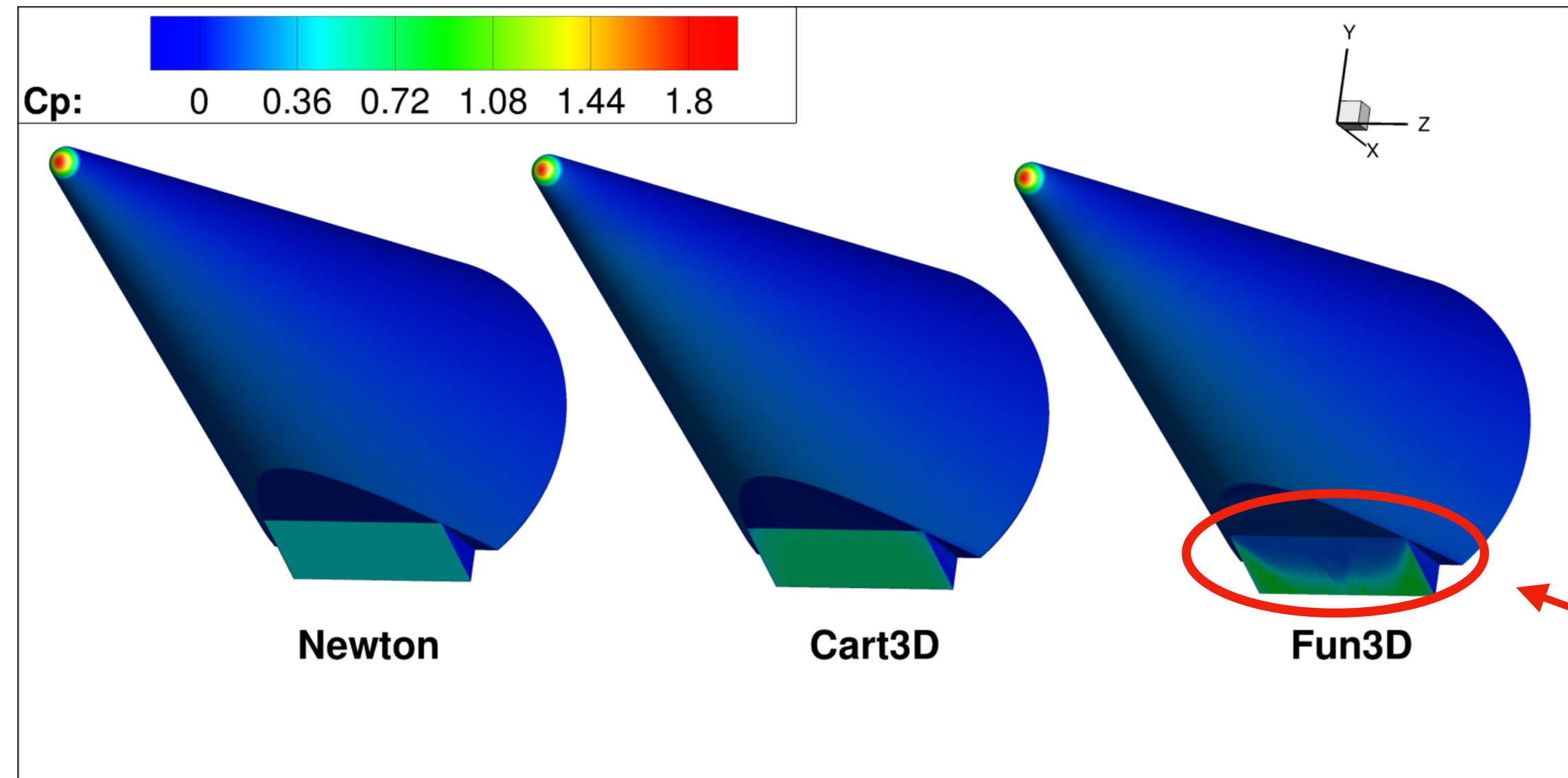


Parameter	Description
R	Radius of the sphere at the tip of the sphere-cone
L	Length of the sphere-cone from the sphere tip to the base
θ	Half-angle of the Cone
Rcut	Fraction of the base radius to put the control plane slice
Put	Length of the flap as a fraction of the parabola created from the slice
ϕ	Angle of the flap



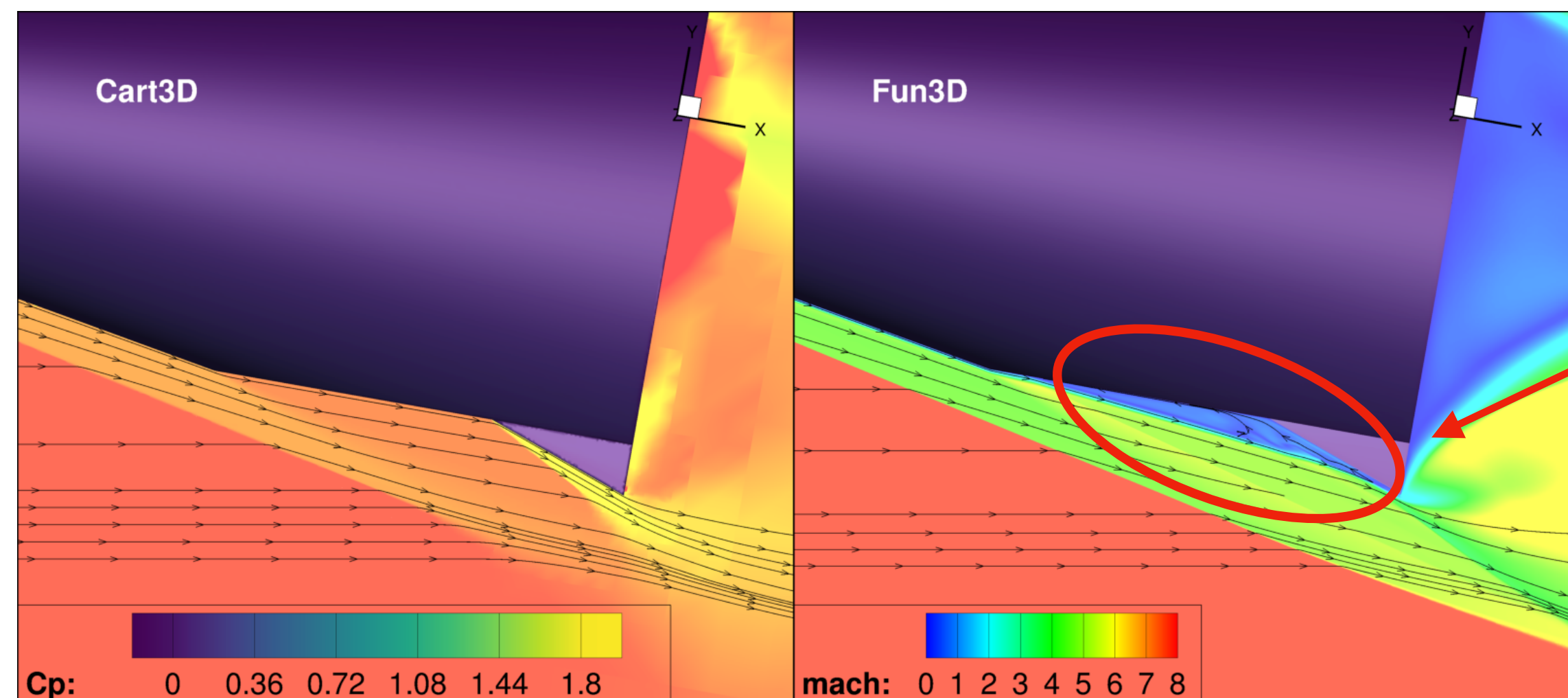
10° Angle of Attack with 20° Flap Deflection

- Generally good agreement between solvers
- FUN3D showing the best results
- Challenging case due to large region of separated flow
- Suspect a lucky coincidence for the Newtonian model results
- Error measured with respect to wind tunnel experiment



Region of Separated Flow

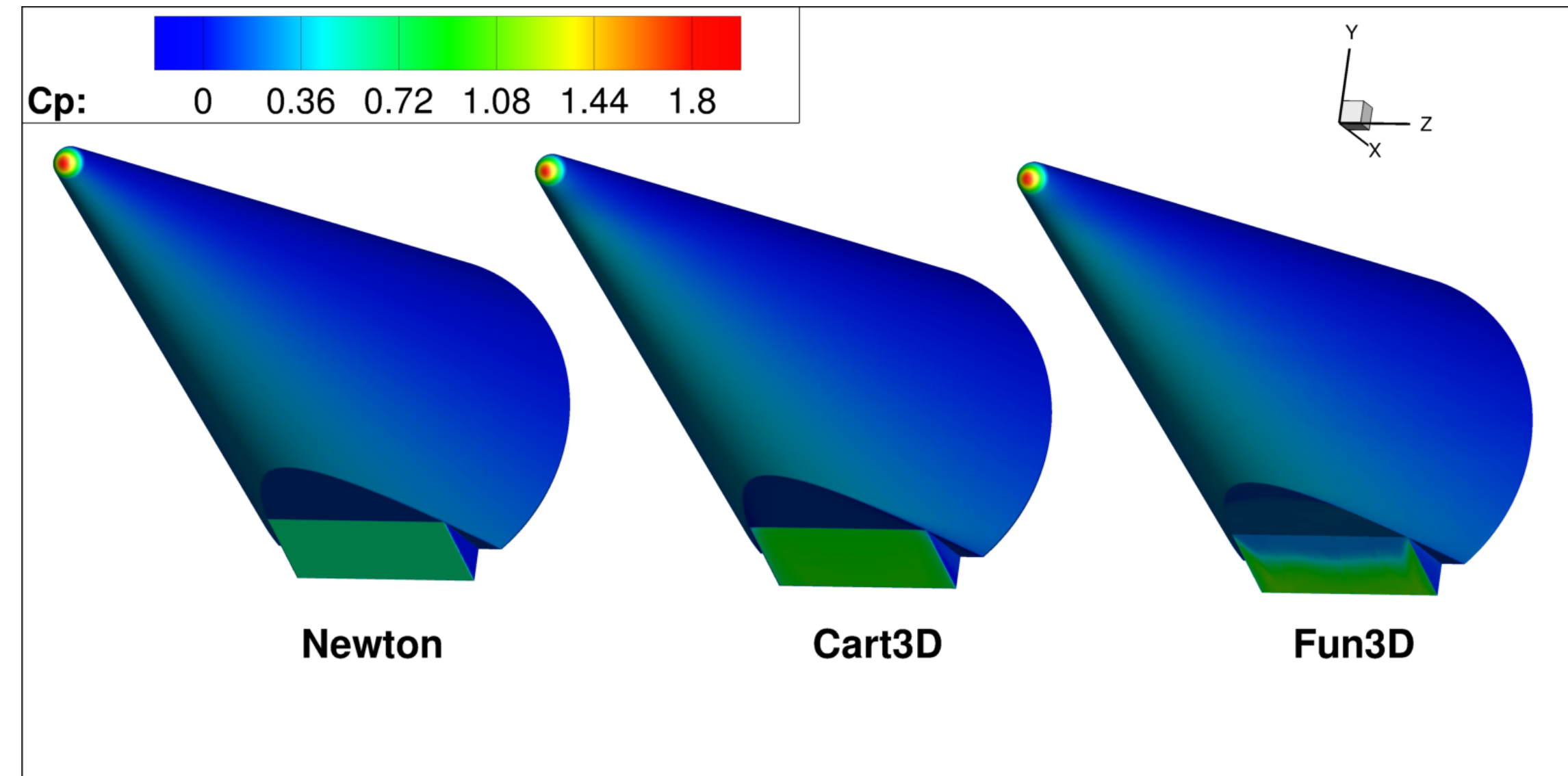
Parameters	Cart3D Error %	FUN3D Error %	Newton Error %
C_A	6.61	3.88	7.68
C_N	9.06	1.14	4.04
C_M	37.8	11.1	7.21
X_{Cp}	7.95	2.49	11.0



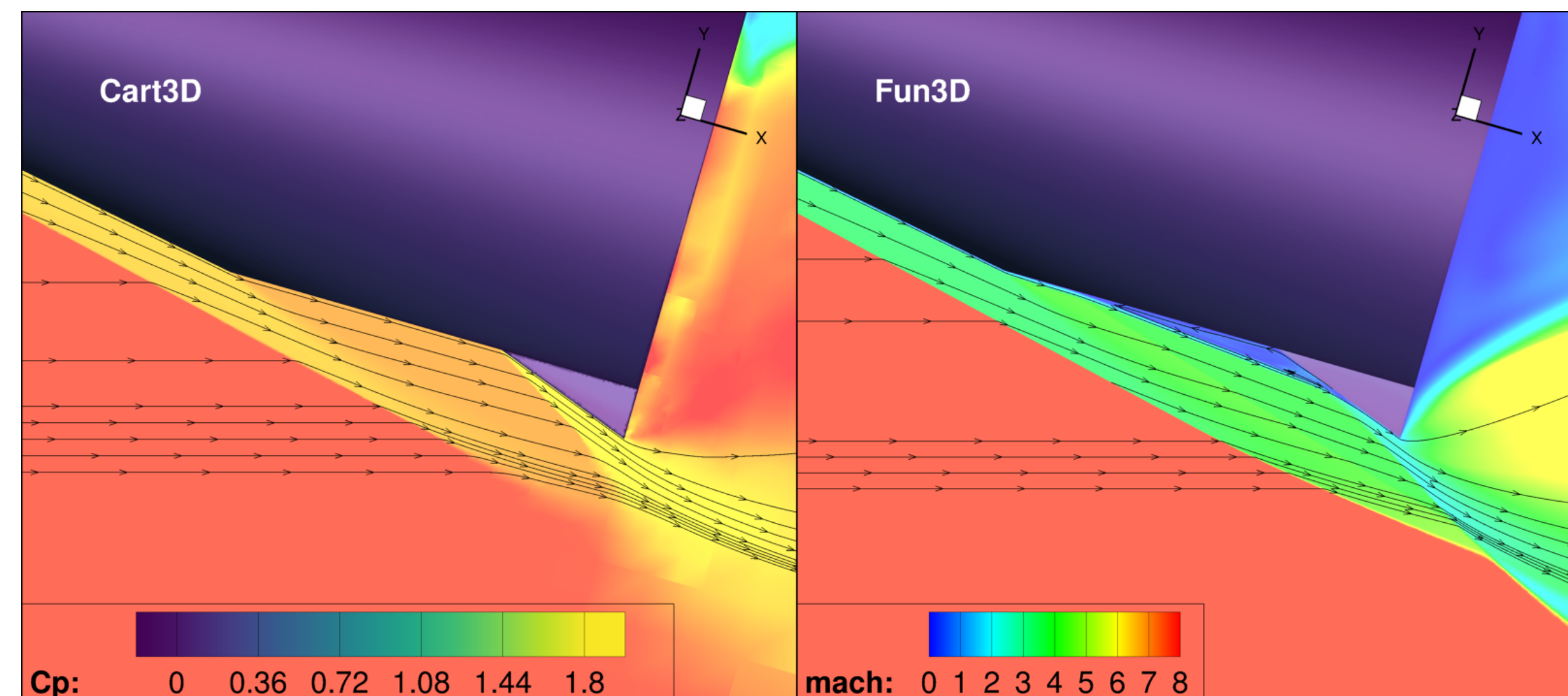
Region of Separated Flow

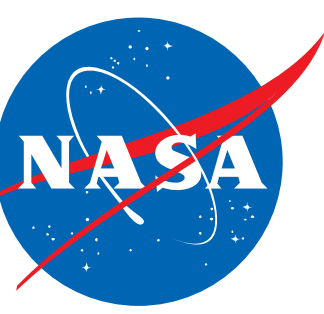
16° Angle of Attack with 20° Flap Deflection

- Better results compared to the 10° AoA case
- Less separation on the control surface
- Confirmed that the good Newtonian model on the 10° is for the wrong reasons
- Error measured with respect to wind tunnel experiment



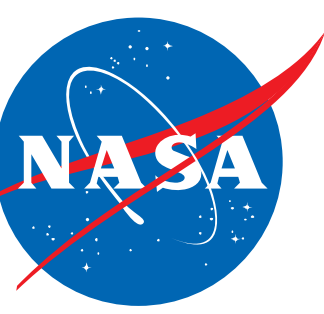
Parameters	Cart3D Error %	FUN3D Error %	Newton Error %
C_A	1.24	0.06	11.9
C_N	5.56	0.29	9.13
C_M	18.9	3.74	8.69
X_{Cp}	10.6	0.97	13.4





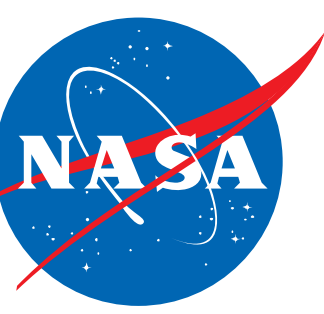
Conclusion

- sBOOM/PCBoom Comparison
 - PCBoom runs much faster than sBOOM, but unable to show noise metric mesh convergence
- Cart3D Geometry Manipulation Tools
 - Increased functionality
 - Python wrapper
- Hypersonic Glider Aerodynamic Analysis
 - Engineering Sketch Pad works well with Cart3D
 - Demonstrated robust and efficient predictions with Cart3D, but accuracy may be compromised in regions of separated flow



Acknowledgments

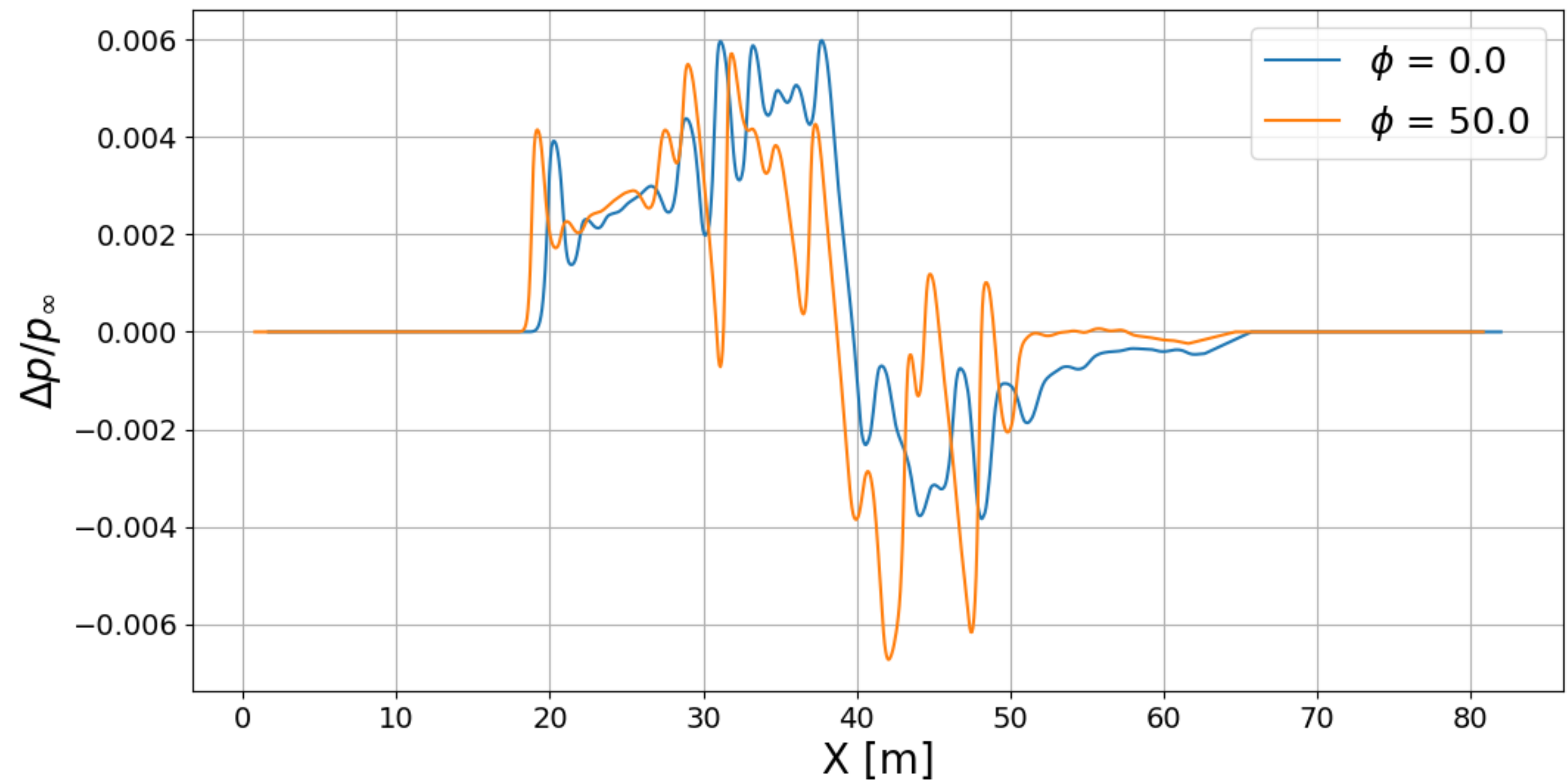
- Marian Nemec and Mike Aftosmis for giving me this opportunity and helping me out along the way
- Sriram Rallabhandi for help trouble shooting sBOOM and identifying the reasons for differences with PCBoom
- Bill Wood and Bil Kleb for help setting up the FUN3D portion of the hypersonic glider study
- Funding was provided by the ARMD Commercial Supersonic Technology Project
- Computing resources were provided by the NASA High-End Computing Program



Questions?

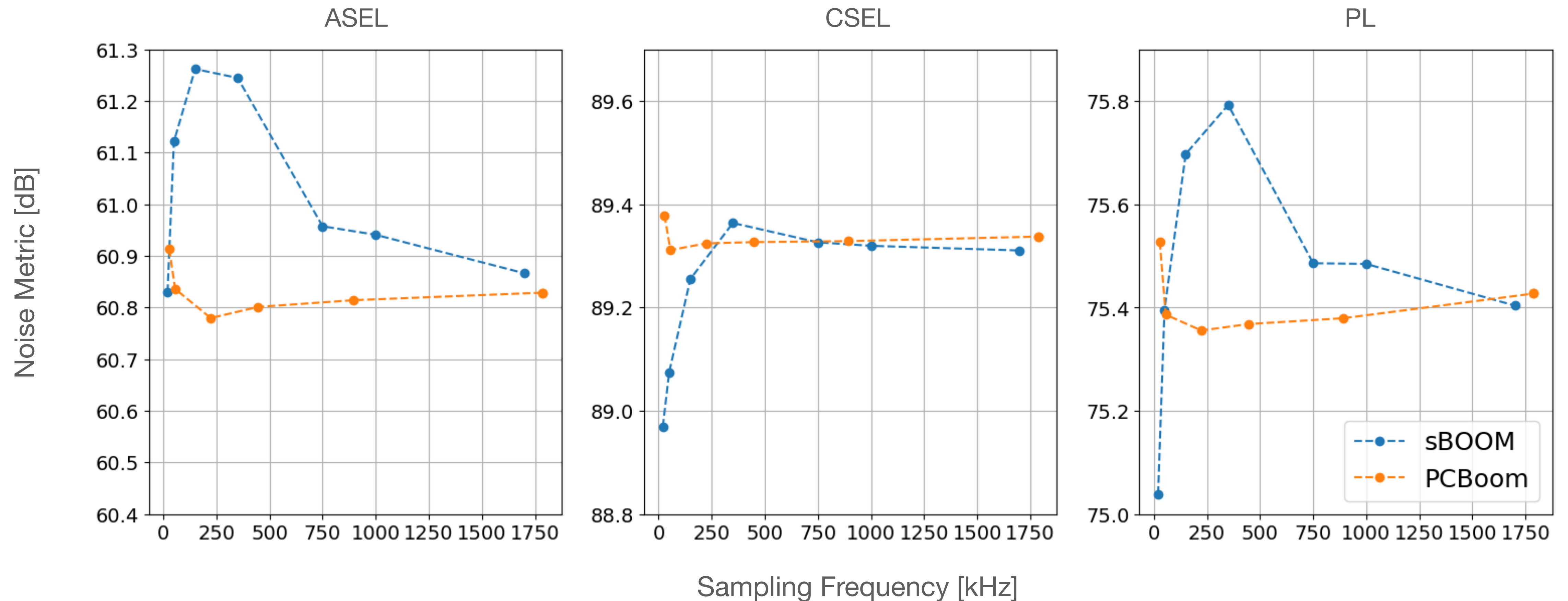
Case Setup

- C608 standard atmosphere case from the 3rd AIAA Sonic Boom Prediction Workshop (SBPW)
- PCBoom v6.7.1
- sBOOM v2.83
- Linear pressure interpolation
- Data validation from SBPW presentations



Off-track Metric Convergence

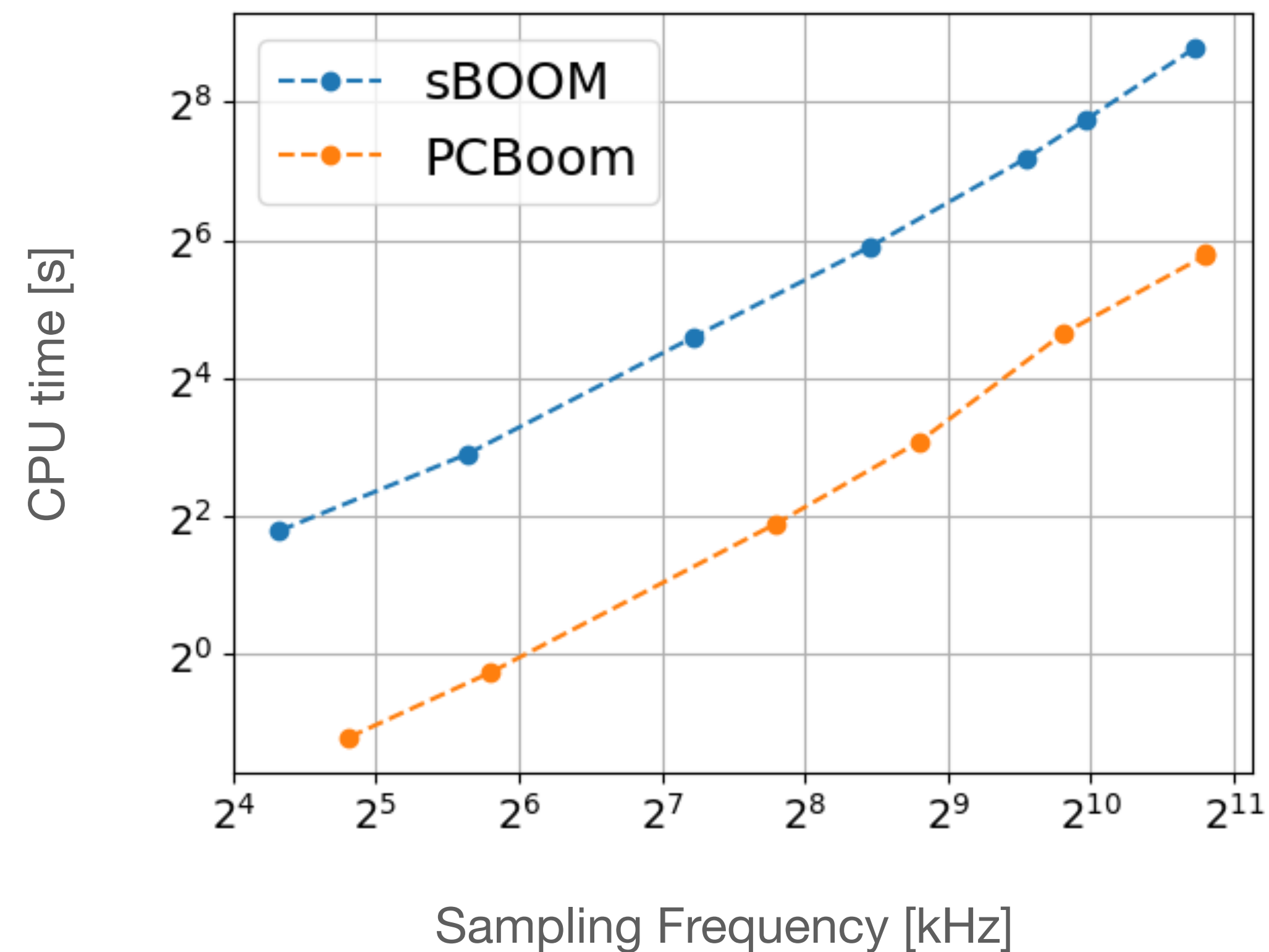
- Both have more trouble converging off-track
- PCBoom still converging to ± 0.05 dB faster than sBOOM



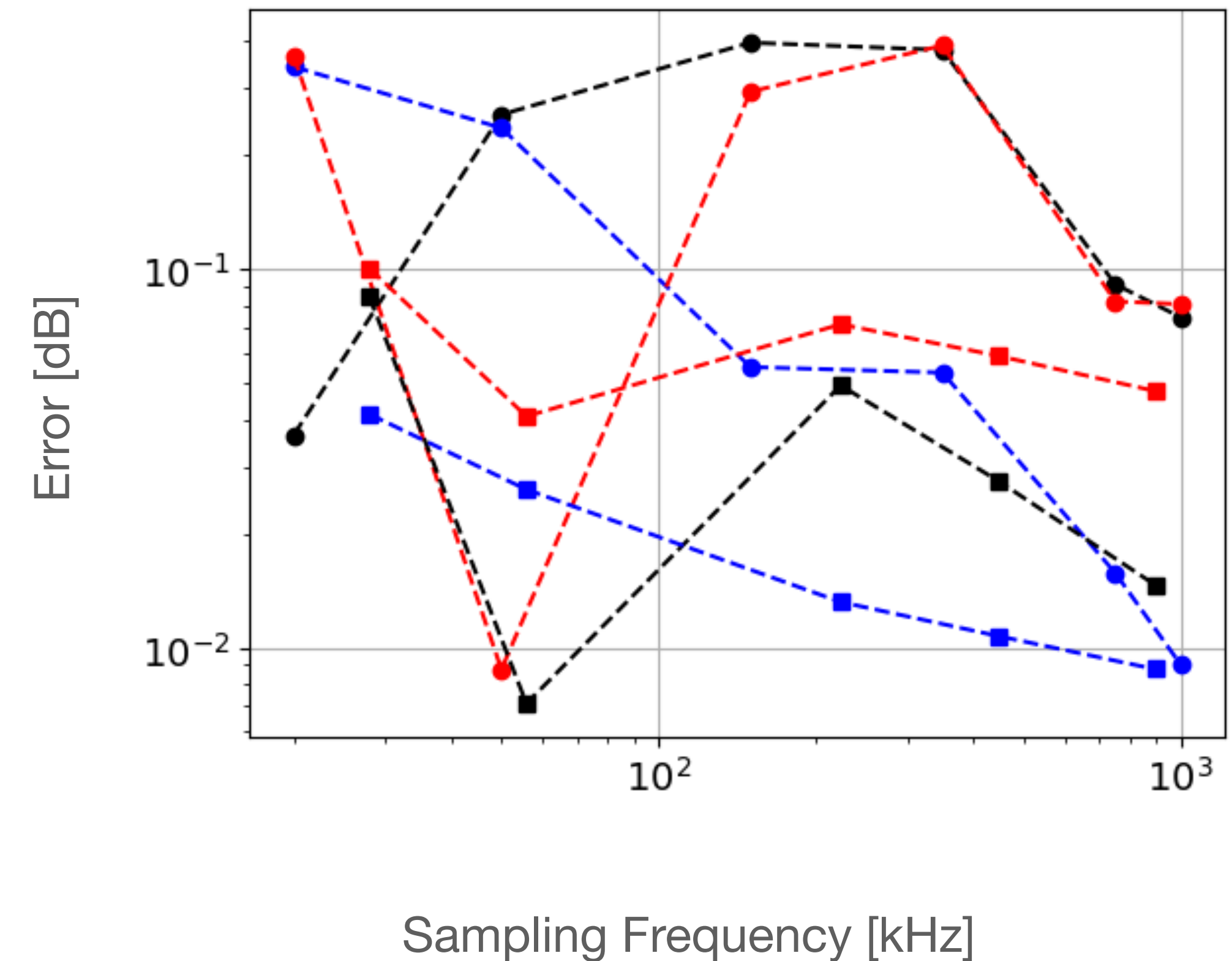
Off-track Runtime and Convergence

- 1700 kHz was used as the exact value in an L_1 error calculation due to frequency limitations
- Lack of any mesh convergence for either program

Runtime

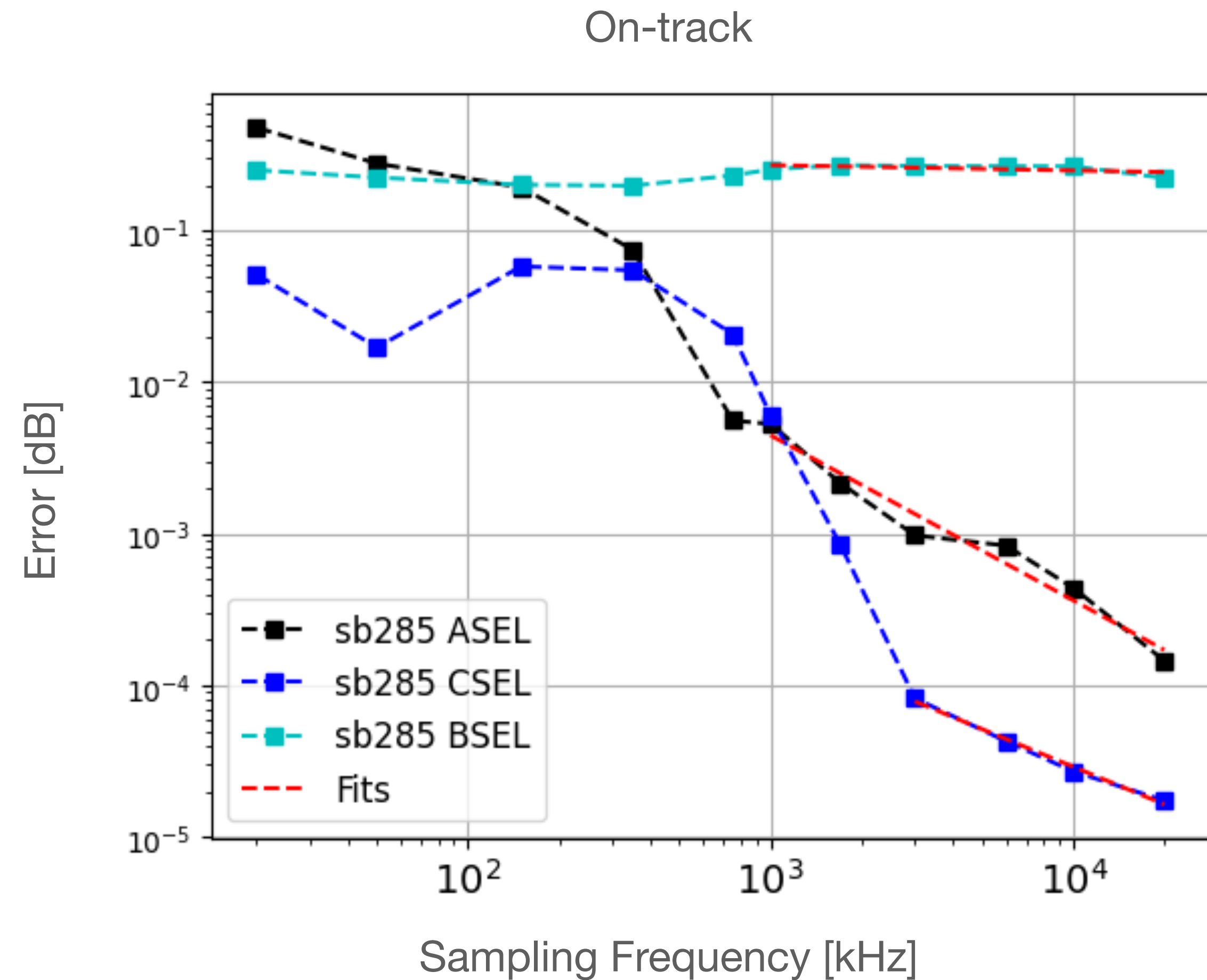


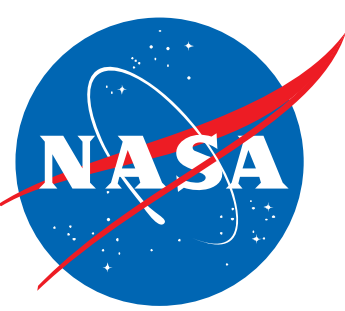
Mesh Convergence



sBOOM v2.85 Mesh Convergence

- No longer an upper bound on frequency — 40,000 kHz exact solution
- Internal sBOOM noise metric filters used instead of adloud
- PL not available at high frequencies
- On-track Convergence Rates
 - ASEL: 1.088
 - BSEL: 0.035
 - CSEL: 0.823





Trix Unit Testing Framework Example

Input File for Generating Tests

File Created to Run tests

```

1 split HEADER -rg_TESTS
2 # Basic test for rotation when cx cy cz is at origin
3 rg_tail_origin dp
4 ../../trix -rg 1 0 0 30 -tri cube.tri
5 -0.5 (sqrt(3)-1)/4 (sqrt(3)+1)/4
6 -0.5 (-sqrt(3)-1)/4 (sqrt(3)-1)/4
7 -0.5 (-sqrt(3)+1)/4 (-sqrt(3)-1)/4
8 -0.5 (sqrt(3)+1)/4 (-sqrt(3)+1)/4
9 0.5 (sqrt(3)-1)/4 (sqrt(3)+1)/4
10 0.5 (-sqrt(3)-1)/4 (sqrt(3)-1)/4
11 0.5 (-sqrt(3)+1)/4 (-sqrt(3)-1)/4
12 0.5 (sqrt(3)+1)/4 (-sqrt(3)+1)/4
13 # Basic test for off origin axis rotation
14 rg_tail_off_origin dp
15 ../../trix -rg 0.5 0 0 90 -cx 0.5 -cy 0.5 -tri cube.tri
16 0.0 0.5 0
17 0.0 0.5 -1
18 1.0 0.5 -1
19 1.0 0.5 0
20 0.0 -0.5 0
21 0.0 -0.5 -1
22 1.0 -0.5 -1
23 1.0 -0.5 0

```

```

1 """
2 Tests generated from add_tests.py
3 Cases: all.inpt2
4 2020-08-17 09:49:24.486749
5 """
6
7 import pytest, os
8 from trix_test_helpers import read_points, compare_points, read_sens
9 from math import pi, sqrt, cos, tan, sin, radians
10
11 #=====
12 #                               END HEADER
13 #
14 #                               BEGIN -rg_TESTS
15 #=====
16
17 # Basic test for rotation when cx cy cz is at origin
18 def test_rg_tail_origin():
19     expected = [ [-0.5, (sqrt(3)-1)/4, (sqrt(3)+1)/4],
20                 [-0.5, (-sqrt(3)-1)/4, (sqrt(3)-1)/4],
21                 [-0.5, (-sqrt(3)+1)/4, (-sqrt(3)-1)/4],
22                 [-0.5, (sqrt(3)+1)/4, (-sqrt(3)+1)/4],
23                 [0.5, (sqrt(3)-1)/4, (sqrt(3)+1)/4],
24                 [0.5, (-sqrt(3)-1)/4, (sqrt(3)-1)/4],
25                 [0.5, (-sqrt(3)+1)/4, (-sqrt(3)-1)/4],
26                 [0.5, (sqrt(3)+1)/4, (-sqrt(3)+1)/4] ]
27
28     os.system('../../trix -rg 1 0 0 30 -tri cube.tri')
29     os.rename('Components_000_c3d.tri', 'rg_tail_origin.tri')
30     actual = read_points('rg_tail_origin.tri')
31
32     missed, error_msg = compare_points(expected, actual)
33
34     assert len(missed) == 0, error_msg
35
36     os.remove('rg_tail_origin.tri')
37

```

Running Tests

```

alexs-air:example akleb$ ./add_tests_pytest.py test_all.py all.inpt2
Writing tests to file: test_all.py ...
Adding a split between: HEADER and -rg_TESTS ...
=> Writing case rg_tail_origin ...
=> Writing case rg_tail_origin_dp ...
=> Writing case rg_tail_off_origin ...
=> Writing case rg_tail_off_origin_dp ...
alexs-air:example akleb$ pytest
===== test session starts =====
platform darwin -- Python 3.7.7, pytest-5.4.3, py-1.9.0, pluggy-0.13.1
rootdir: /Users/akleb/nasa/cart3d_v1.5.5_OSX64_ICC--18.10.02/trixs/example
collected 4 items

test_all.py .... [100%]

===== 4 passed in 0.09s =====
alexs-air:example akleb$

```

Python Wrapper

- Uses already written Trix source code
- Can preform geometry manipulations in any order desired
- Reduces time spent in file I/O

```

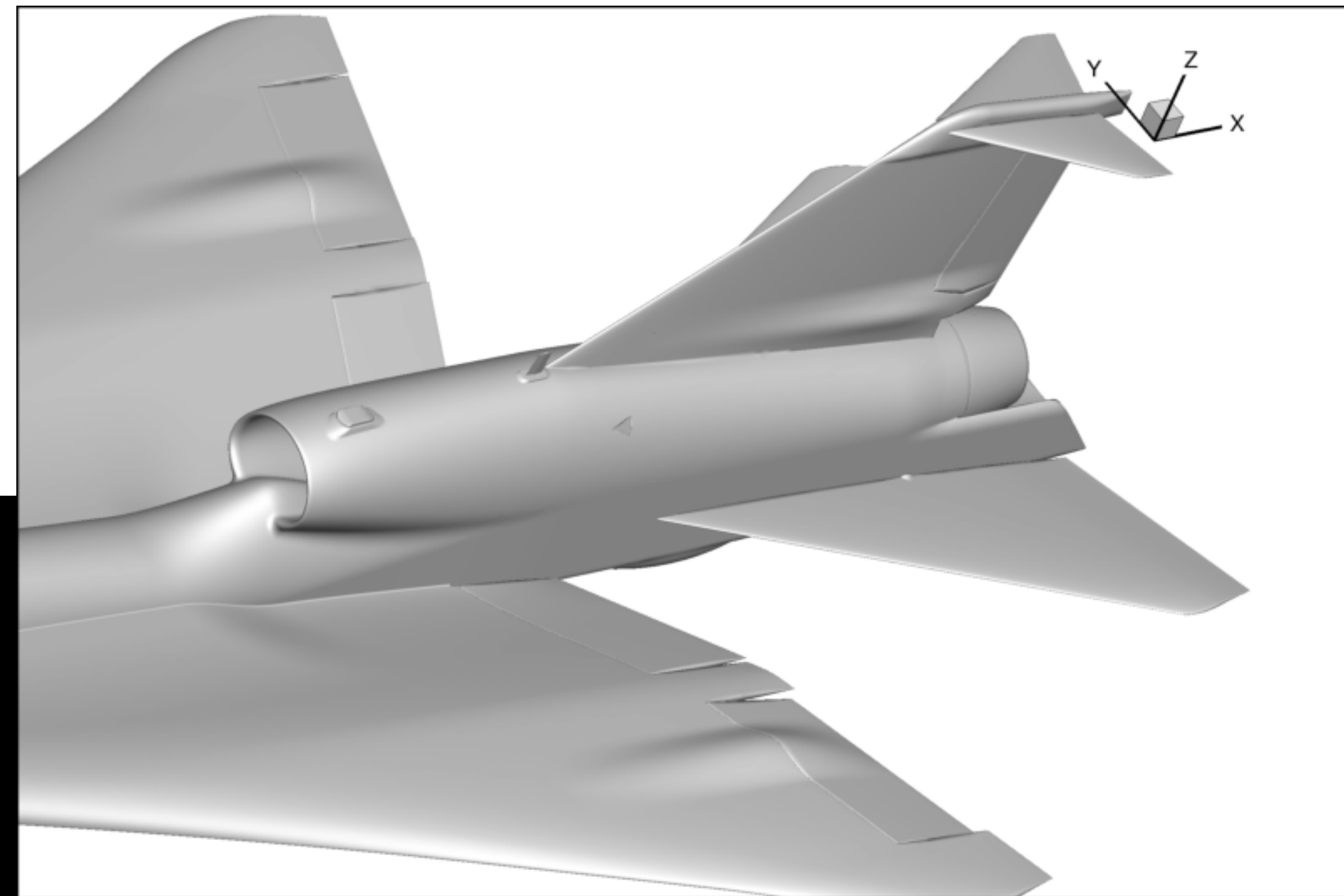
1  #!/usr/bin/env python3
2  """
3  Alex Kleb
4  Script for animating flaps
5  """
6
7  import tecplot, trix, os, sys
8
9  # Create geometry files
10 if "-m" in sys.argv:
11     try:
12         os.mkdir('geoms')
13         os.chdir('geoms')
14     except FileExistsError:
15         print("geoms/ already exists, cleaning out ...")
16         os.chdir('geoms')
17         for filename in os.listdir():
18             os.remove(filename)
19
20     hinge = [[25.195972, -4.048937, 0.559805], [24.474211, -2.047063, 0.576095]]
21
22     c612 = trix.tsTriangulation('../c612_case01_m.i.tri')
23     c612.write_tec(str(0).zfill(5))
24
25     j = 0
26     c612.select([9])
27     rg = [list(range(0,-36, -5)), list(range(-35, 36, 5))]
28     for i in range(7):
29         c612.arb_rotate(-5, hinge[0], hinge[1])
30         c612.write_tec(str(j).zfill(5))
31         j += 1
32     for i in range(14):
33         c612.arb_rotate(5, hinge[0], hinge[1])
34         c612.write_tec(str(j).zfill(5))
35         j += 1
36     for i in range(7):
37         c612.arb_rotate(-5, hinge[0], hinge[1])
38         c612.write_tec(str(j).zfill(5))
39         j += 1
40
41     os.chdir('../')
42
43

```

```

43
44 if '-p' in sys.argv:
45     try:
46         os.chdir('geoms')
47     except OSError:
48         print("no geometry folder found, run with '-m'")
49     for filename in os.listdir():
50         name = filename.split('_')[0]
51
52         dataset = tecplot.data.load_tecplot(filename, read_data_option=0)
53         frame = tecplot.active_frame()
54         frame.width = 12
55         frame.load_stylesheet("../tail.sty")
56         tecplot.export.save_png(name+".png")
57
58     os.chdir('../')
59
60 try:
61     os.chdir('geoms')
62     os.system('convert -delay 20 -loop 0 *png trix-short.gif')
63 except:
64     print("geoms/ does not exist, run with -m and -p")
65

```



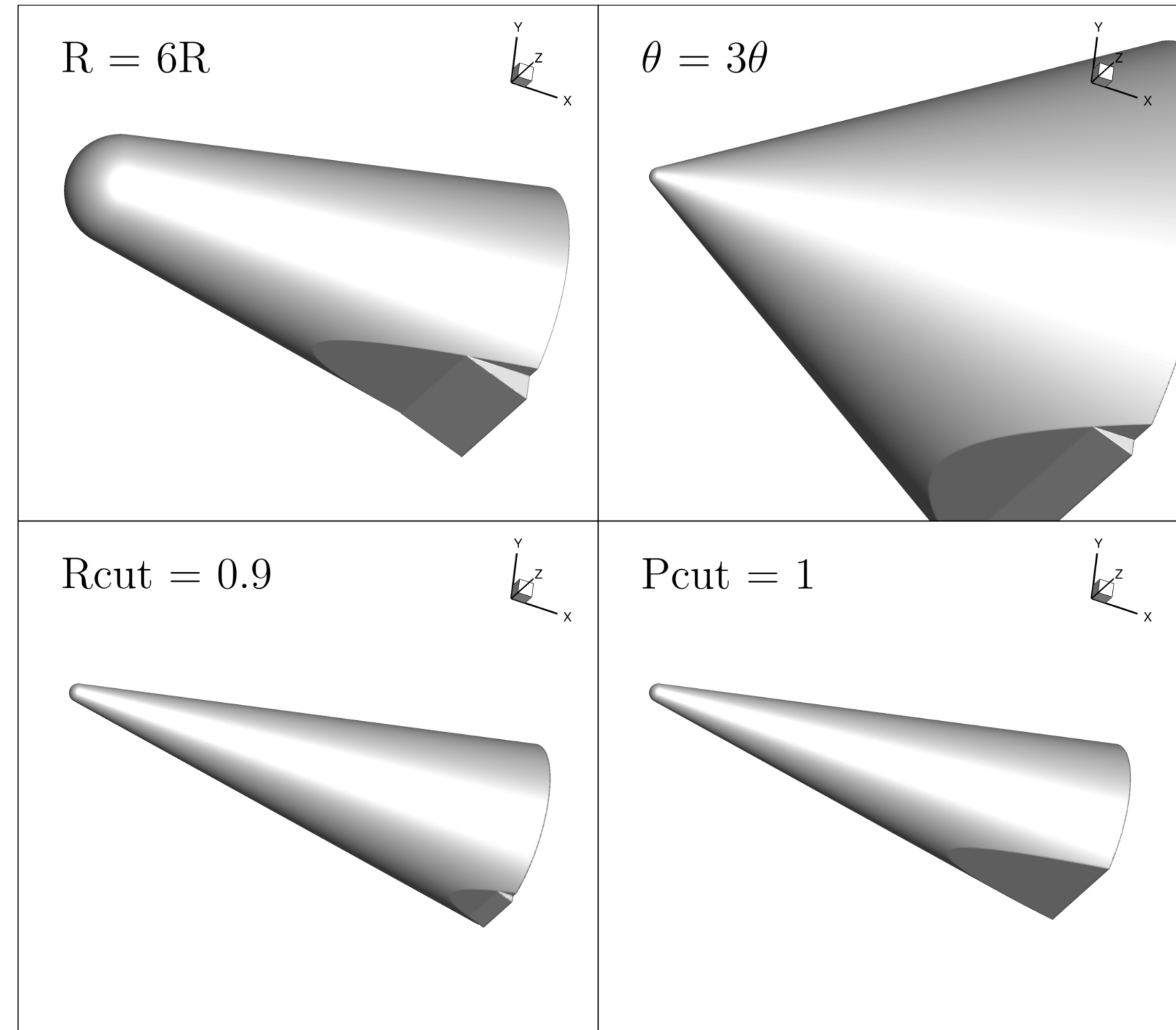
Parametric Modeling with ESP

- Robust file for geometry creation
- Rapid geometry prototyping

```

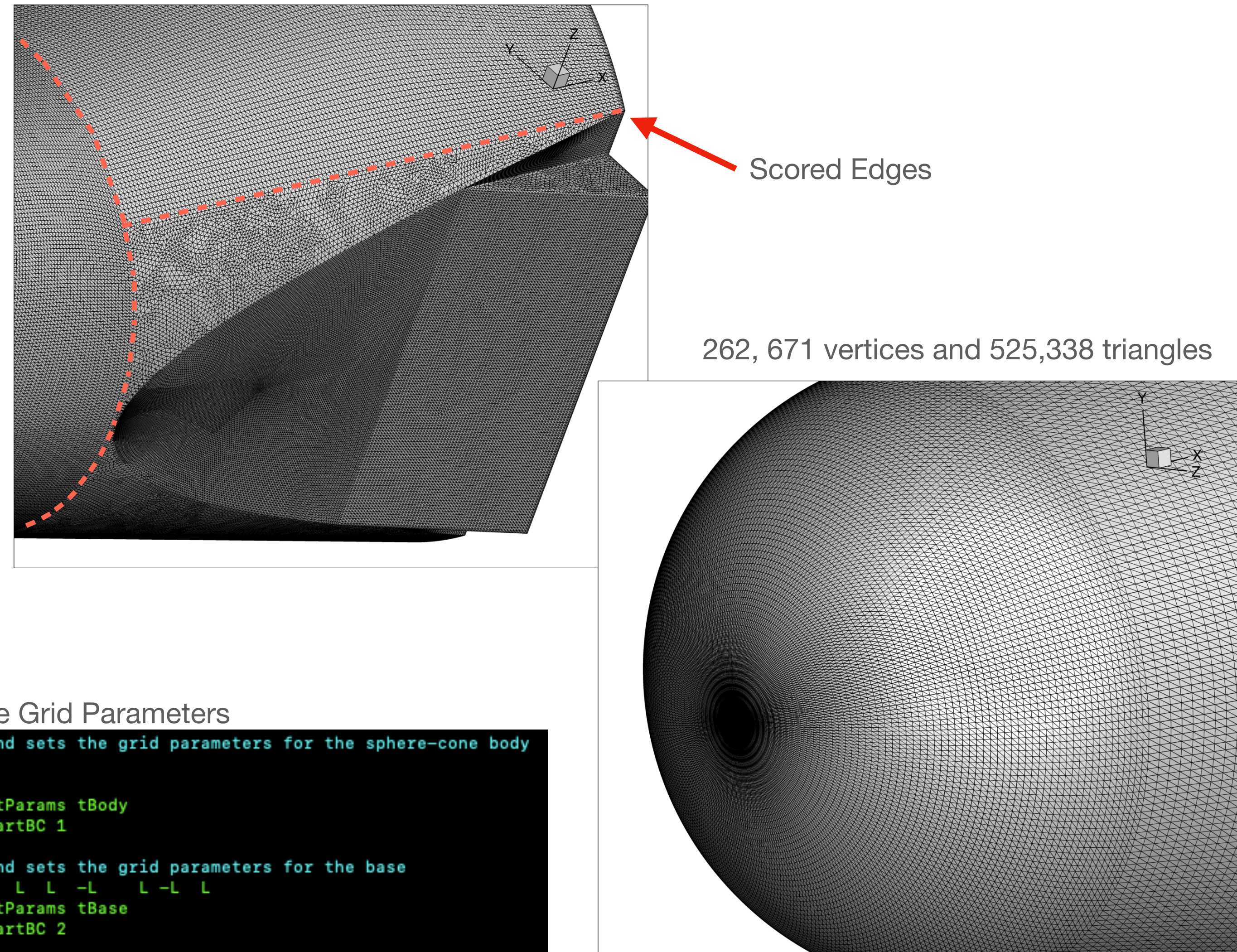
1 # Design Parameters
2 DESPMTR R 1/60 # Sphere tip radius
3 DESPMTR L 10.391/12 # Length of Glider
4 DESPMTR theta 10 # Parametric Cone Angle
5 DESPMTR Rcut 0.7251944005658573 # % of the radius at which to put the plane slice
6 DESPMTR Pcut 1/3 # % of the max parabola length at which to put the flap
7 DESPMTR phi 20 # Angle of the flap
8
9 # Grid Parameters
10 DIMENSION tBody 1 3 0
11 SET tBody "0.004/sqrt(2); 0.01/sqrt(2); 2.0/sqrt(2)"
12
13 DIMENSION tBase 1 3 0
14 SET tBase "0.008/sqrt(2); 0.01/sqrt(2); 10.0/sqrt(2)"
15
16 # the radius of the base of the cone
17 SET Rmax R*cosd(theta)+(L-R*(1+sin(theta)))*tand(theta)
18
19 # Sketch the profile to revolve
20 SKBEG 0.0 0.0 0.0
21 LINSEG L 0.0 0.0
22 LINSEG L Rmax 0.0
23 LINSEG R*(1-sind(theta)) R*cosd(theta) 0.0
24 CIRARC R*(1-cosd((90-theta)/2)) R*sind((90-theta)/2) 0.0 \
25 0.0 0.0 0.0
26 SKEND
27
28 REVOLVE 0 0 0 1 0 0 180
29
30 STORE halfBody
31 RESTORE halfBody
32
33 MIRROR 0 0 1
34 RESTORE halfBody
35 UNION
36
37 # Makes the plane cut only if the Rcut is in the range [0, 1)
38 IFTHEN Rcut LT 1 AND Rcut GE 0
39 # Make the plane cut for the flap
40 BOX 0.0 Rcut*Rmax -Rmax L Rmax 2*Rmax
41 SUBTRACT
42 ENDIF
43

```



Surface Tessellation

- ESP uses the Electronic Geometry Aircraft Design System (EGADS)
- .tParams attribute determines grid size
- Easy to encourage structured grids
- egads2cart converts surface tessellation to Cart3D tri file
- CartBC attribute determines component numbers



Defining the Grid Parameters

```

9 # Grid Parameters
10 DIMENSION tBody 1 3 0
11 SET tBody "0.004/sqrt(2); 0.01/sqrt(2); 2.0/sqrt(2)"
12
13 DIMENSION tBase 1 3 0
14 SET tBase "0.008/sqrt(2); 0.01/sqrt(2); 10.0/sqrt(2)"
15

```

Applying the Grid Parameters

```

44 # Selects and sets the grid parameters for the sphere-cone body
45 SELECT BODY
46 SELECT FACE
47 ATTRIBUTE .tParams tBody
48 ATTRIBUTE CartBC 1
49
50 # Selects and sets the grid parameters for the base
51 SELECT FACE L L -L L -L L
52 ATTRIBUTE .tParams tBase
53 ATTRIBUTE CartBC 2
54

```

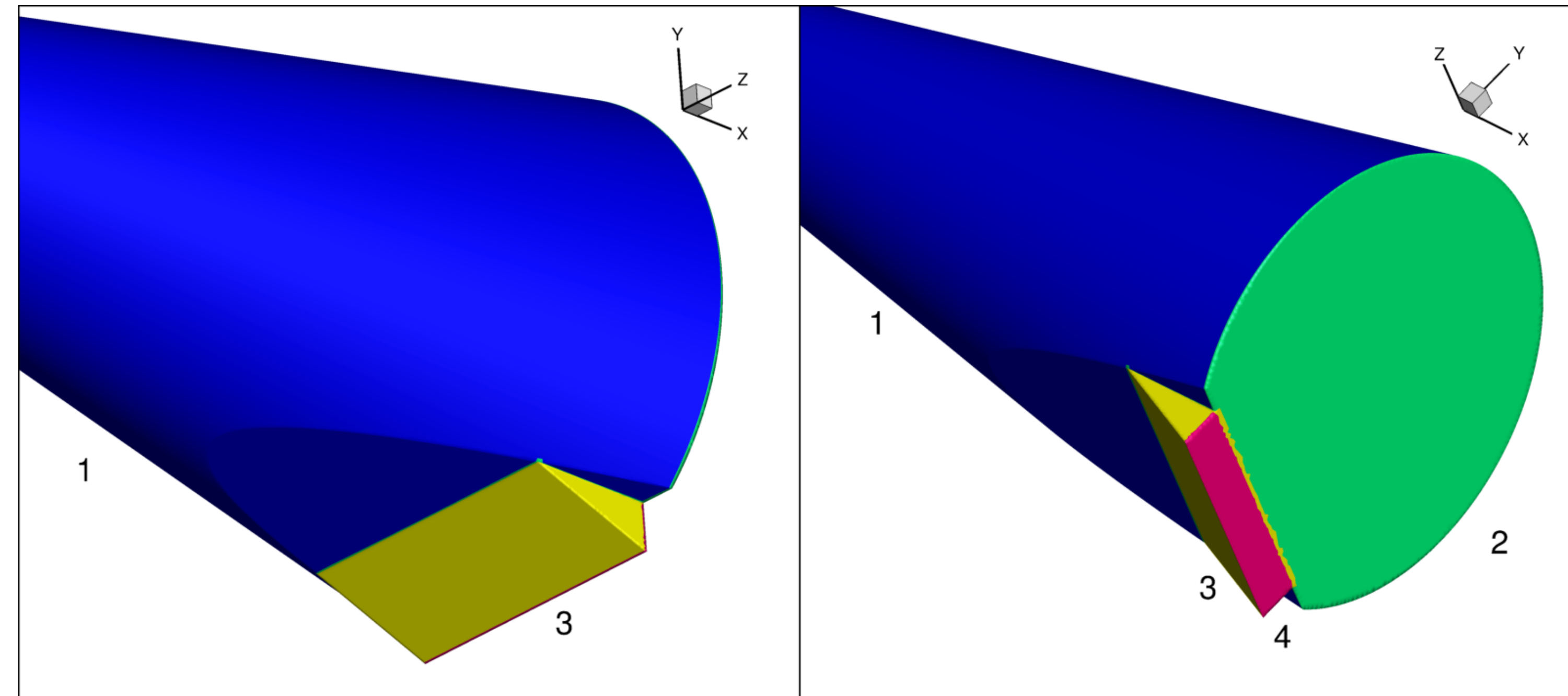
Flow Solver Setups

- Mach 7.84, $Re_L 1.86 \times 10^{16}$
- Cart3D goal-based mesh adaptation
- FUN3D feature-based mesh adaptation
- Canonical, 10° AoA, and 16° AoA cases

Cart3D Functional:

$$F(J_H) = C_{A, \text{Body}} + C_{N, \text{Body}} + 0.1(C_{A, \text{Base}} + C_{N, \text{Base}})$$

Parameters of Interest	Description
C_A	Fore-body axial force coefficient
C_N	Normal force Coefficient
C_M	Pitching moment about $x/L = 0.5$
X_{Cp}	Center of pressure

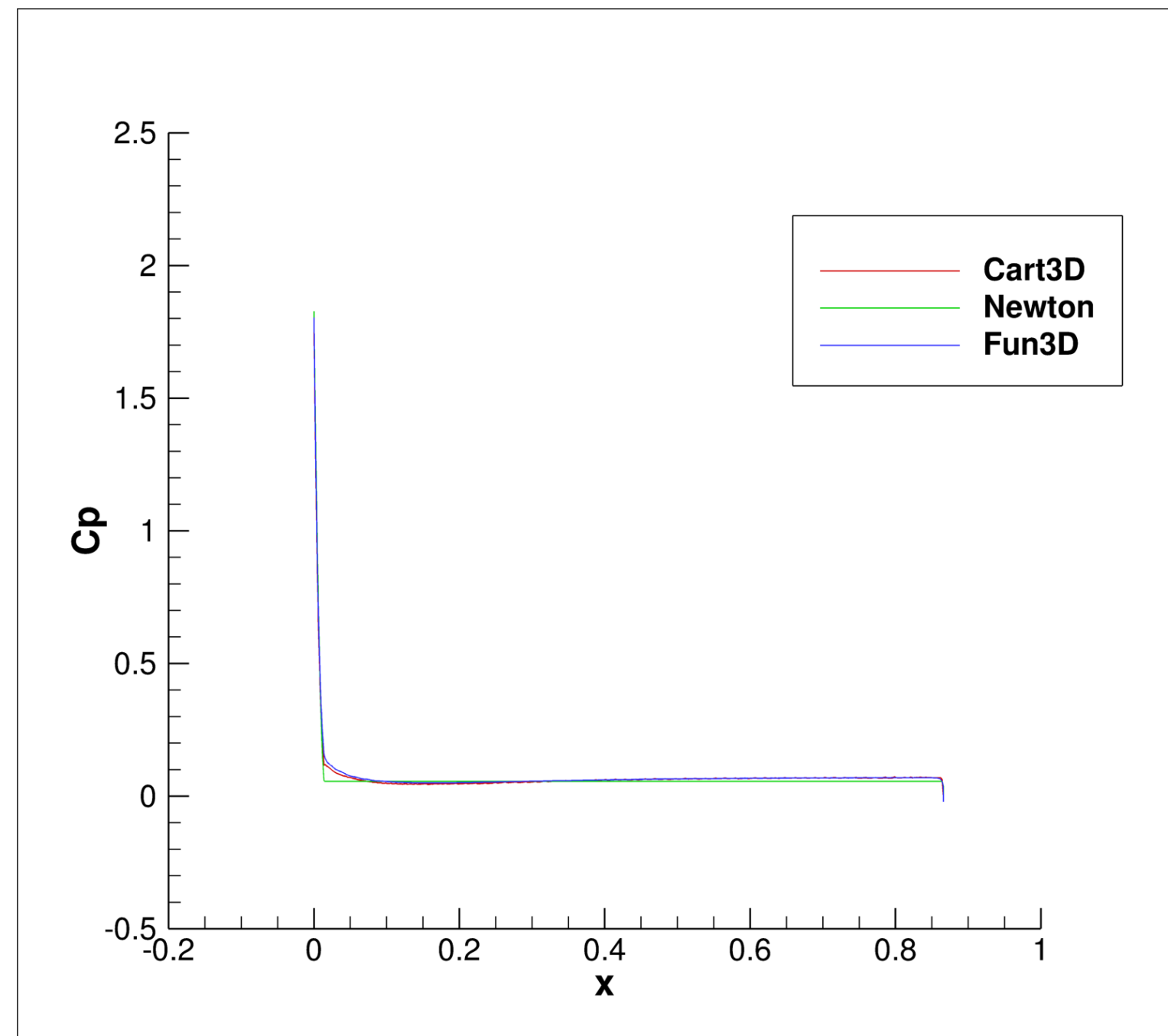


Body refers to components 1 and 3

Base refers to components 1 and 3

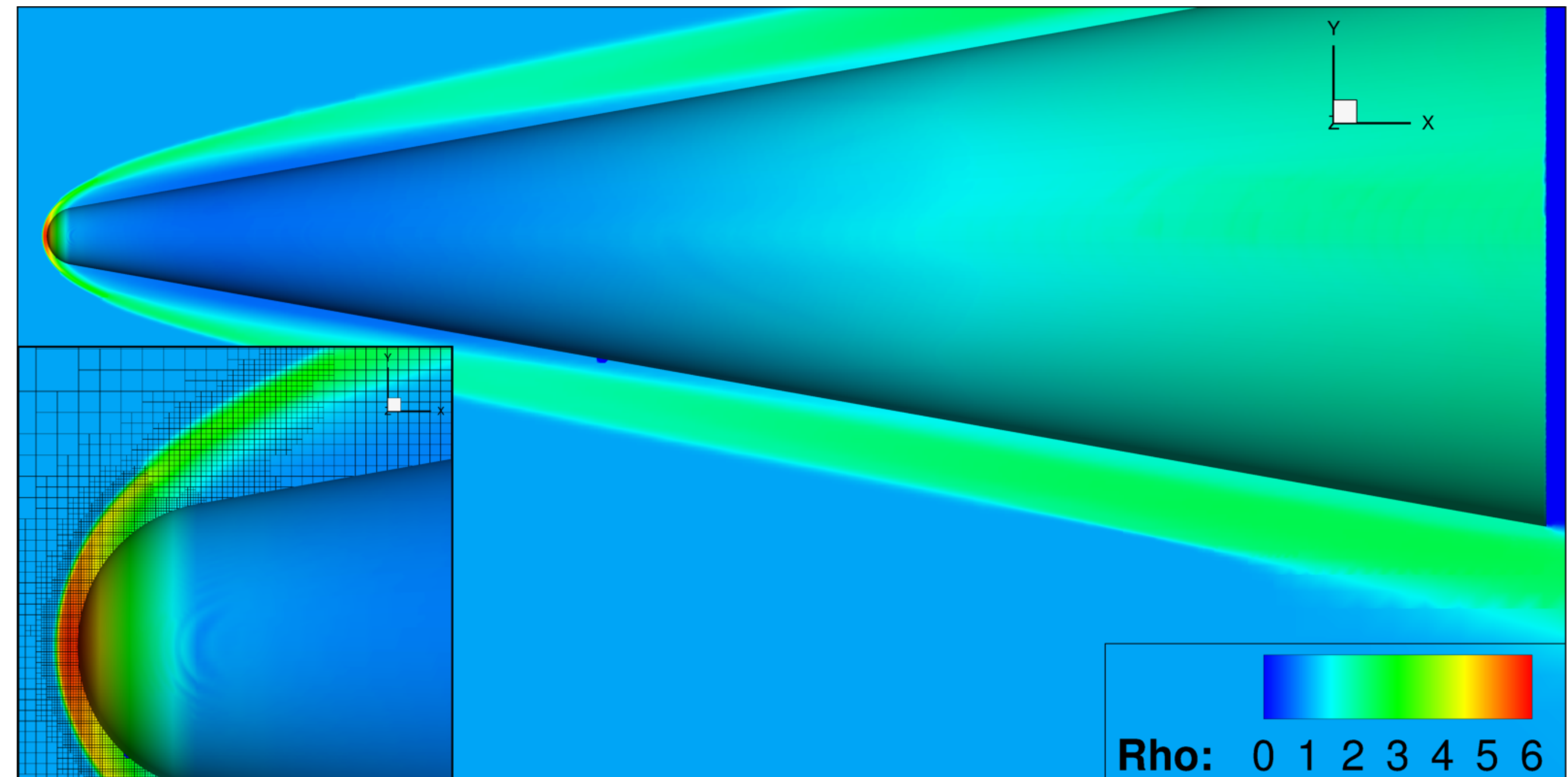
Canonical Case

- Good agreement between all three flow solvers
- Confirmed Cart3D functional choice



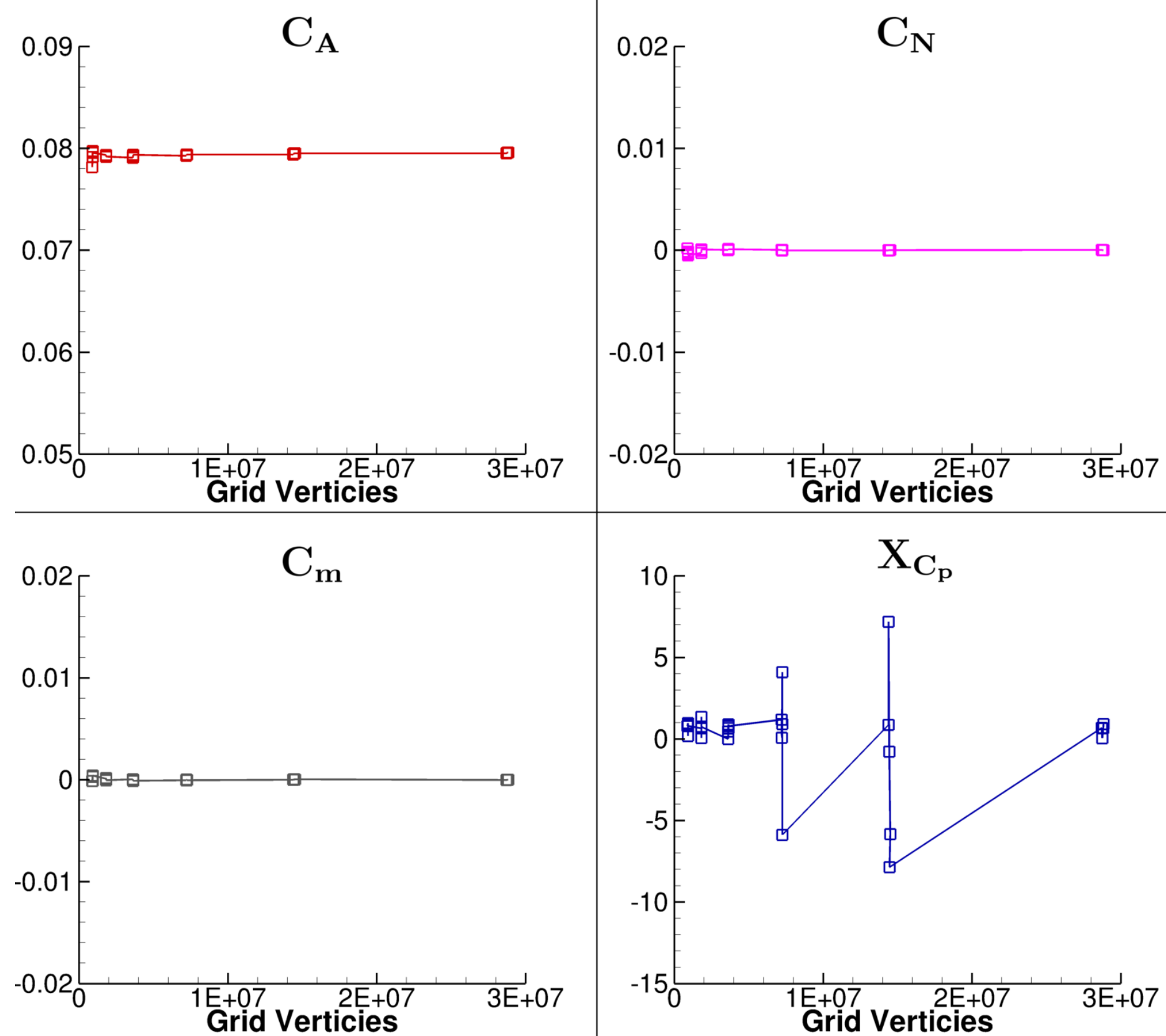
Cart3D Functional:

$$F(J_H) = C_{A, \text{Body}} + C_{N, \text{Body}} + 0.1(C_{A, \text{Base}} + C_{N, \text{Base}})$$

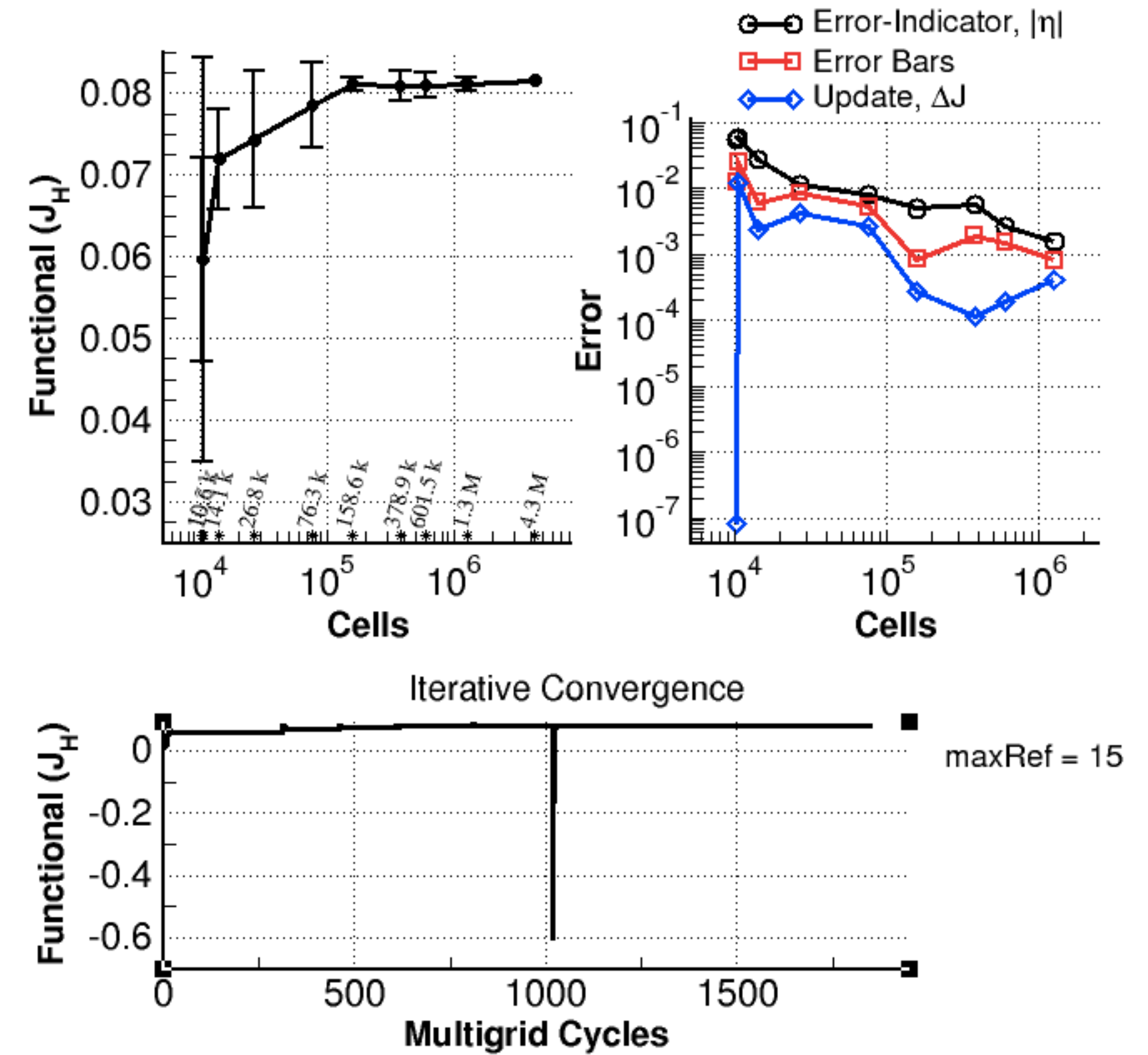


Canonical Case Mesh Convergence

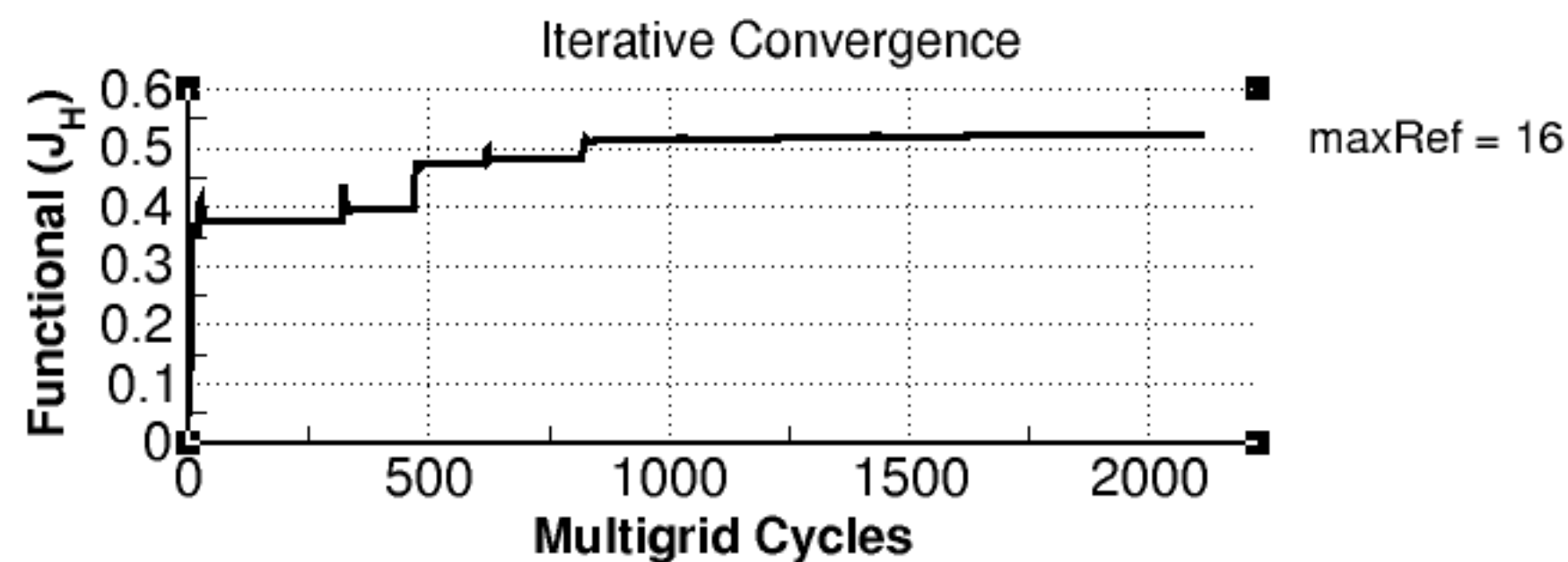
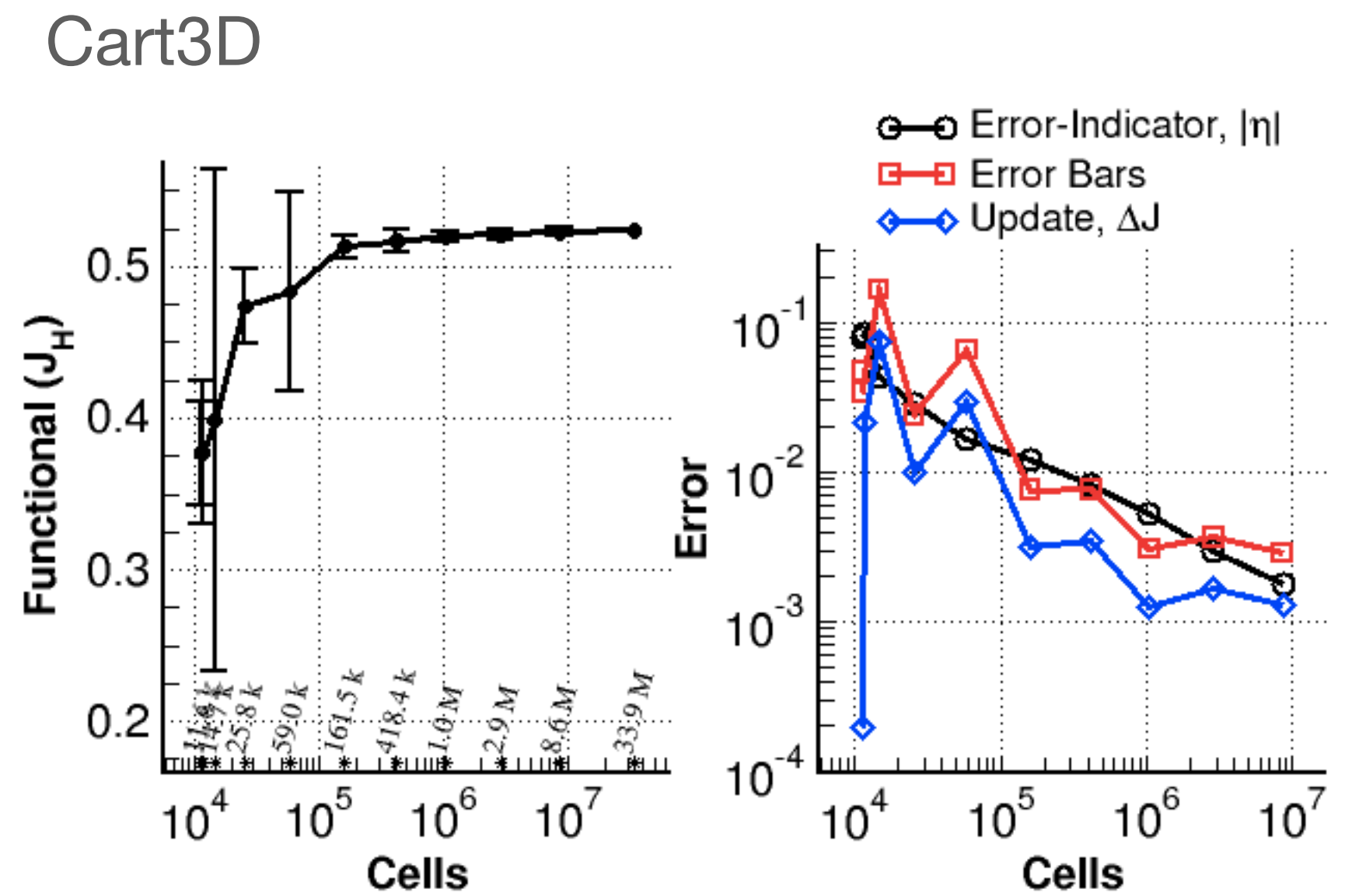
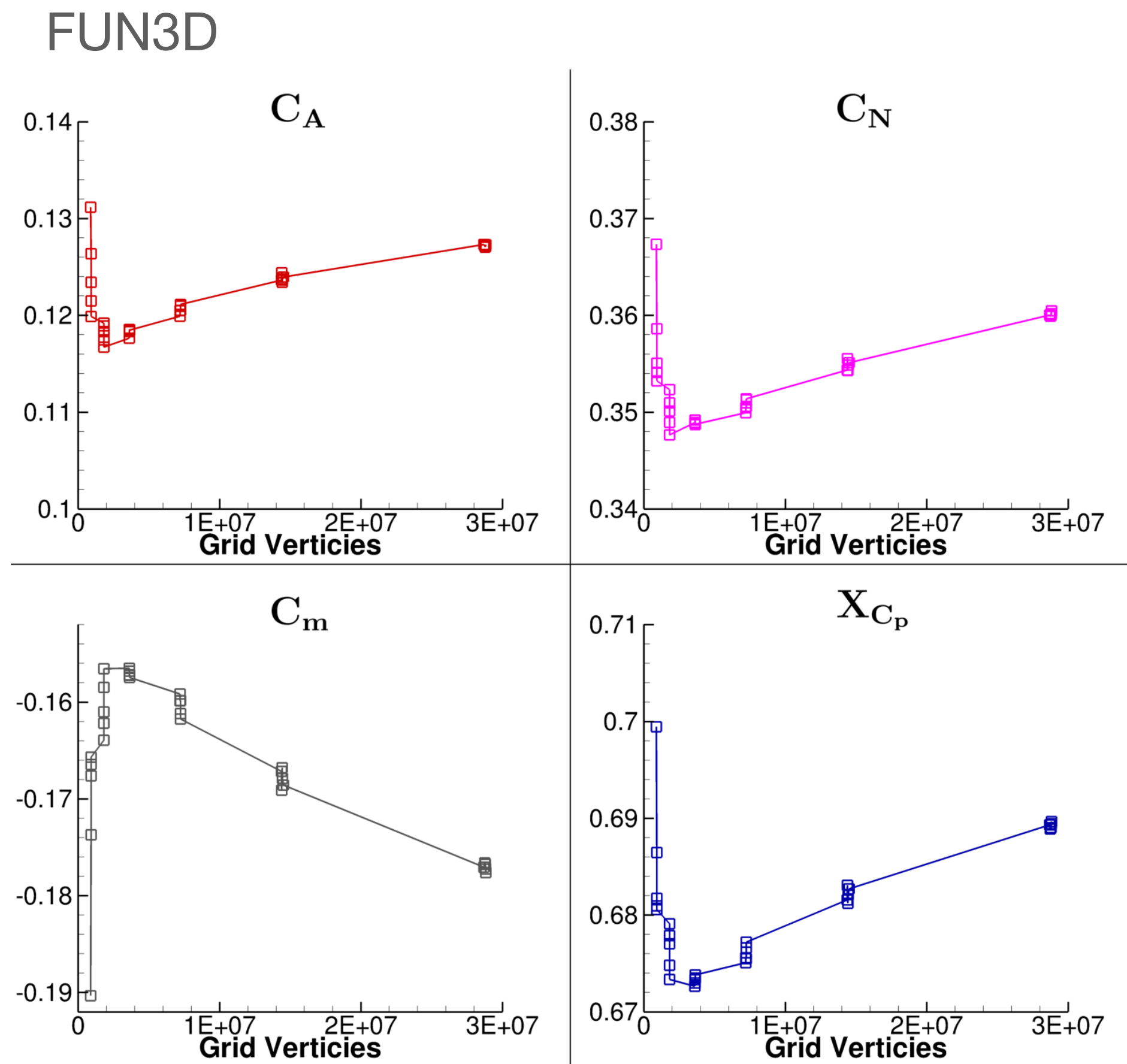
FUN3D



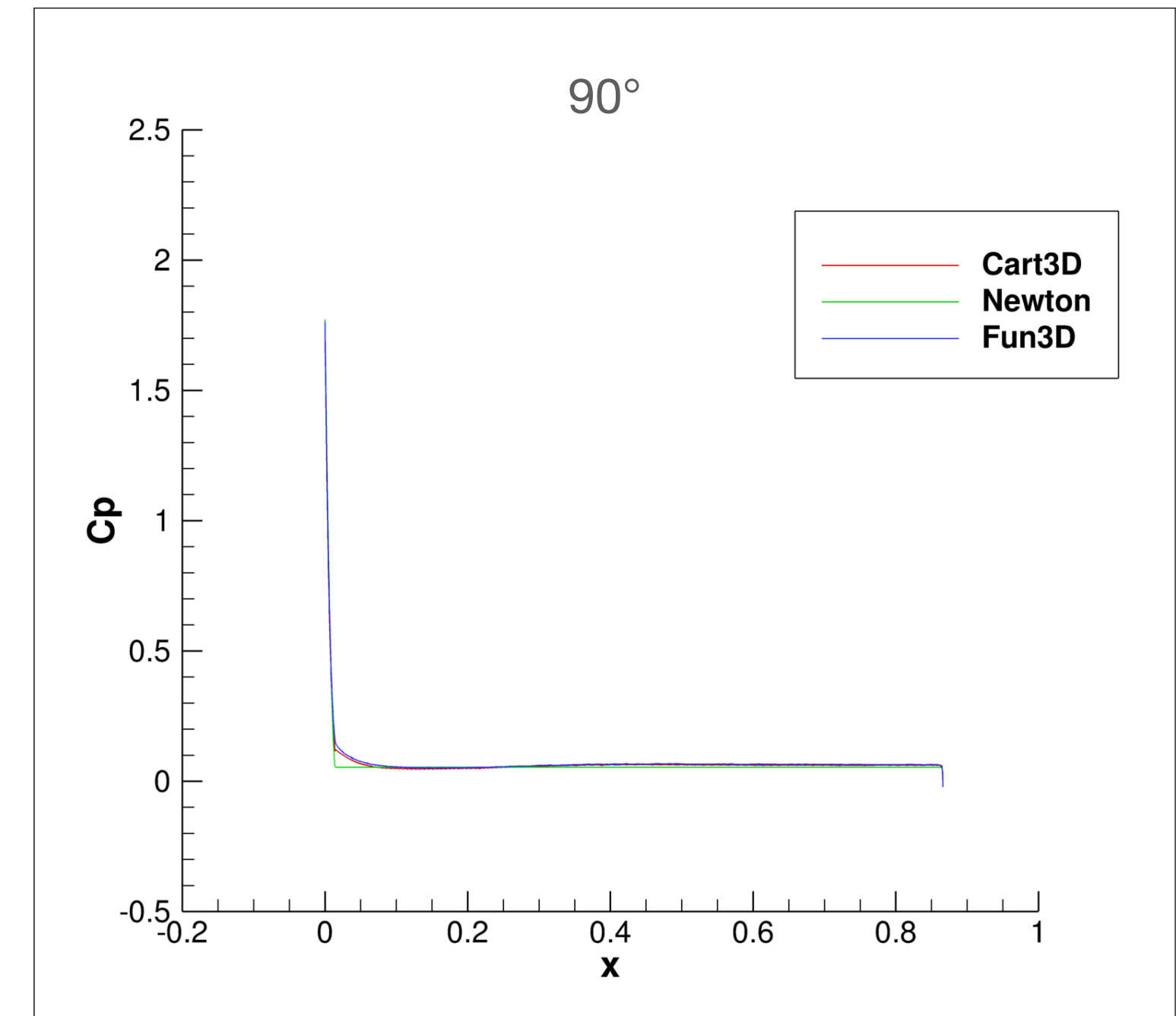
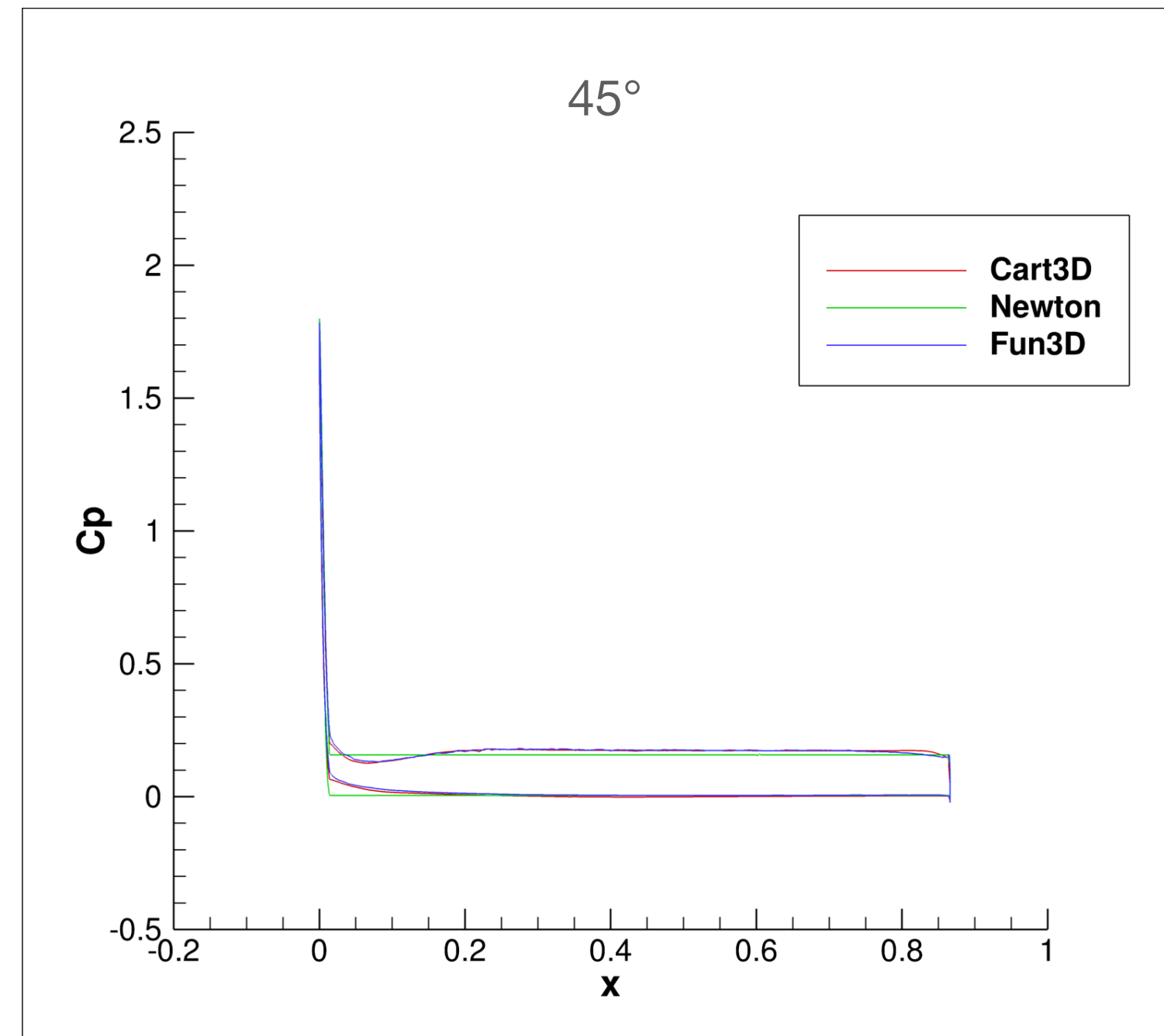
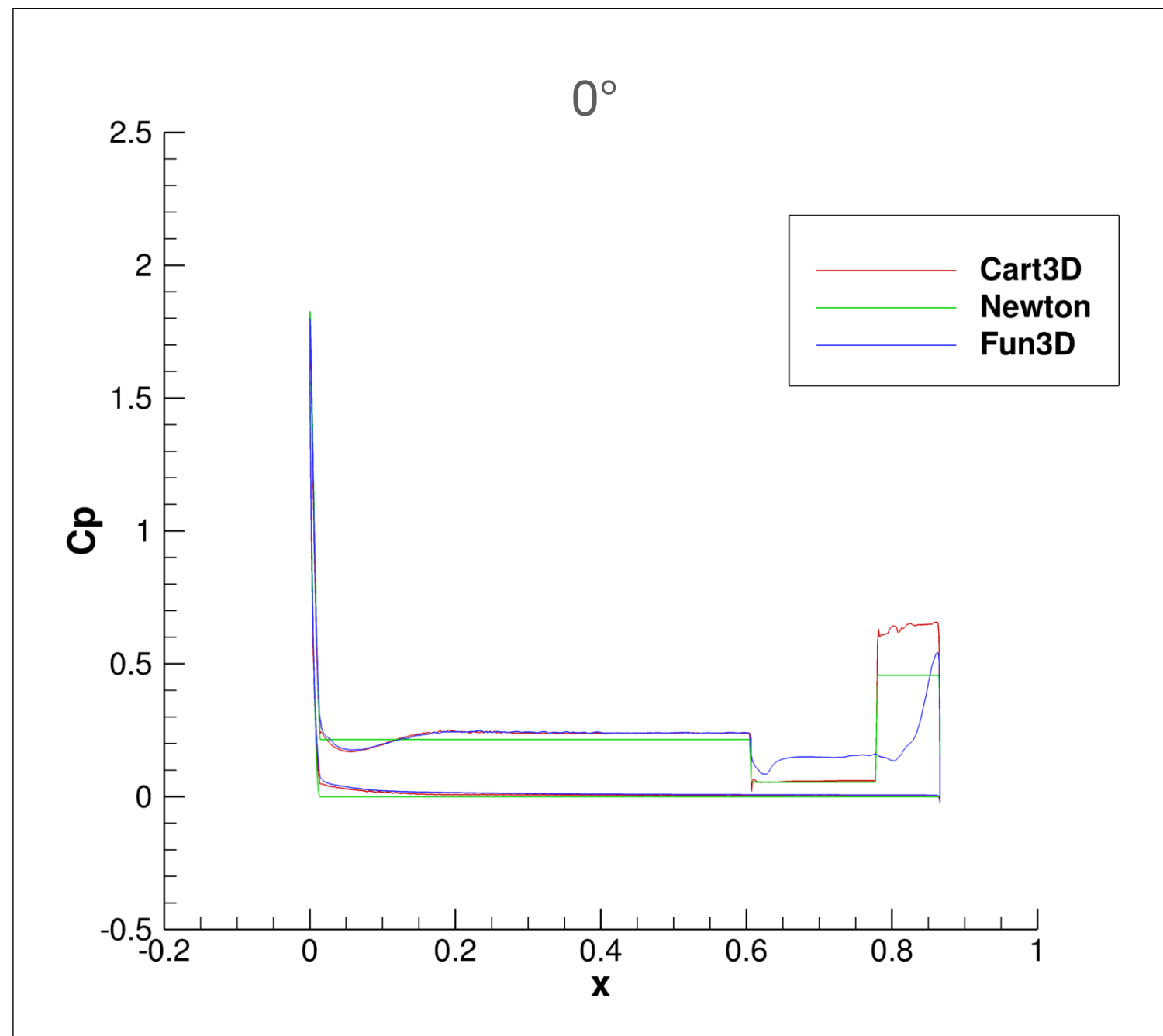
Cart3D



10° AoA Mesh Convergence

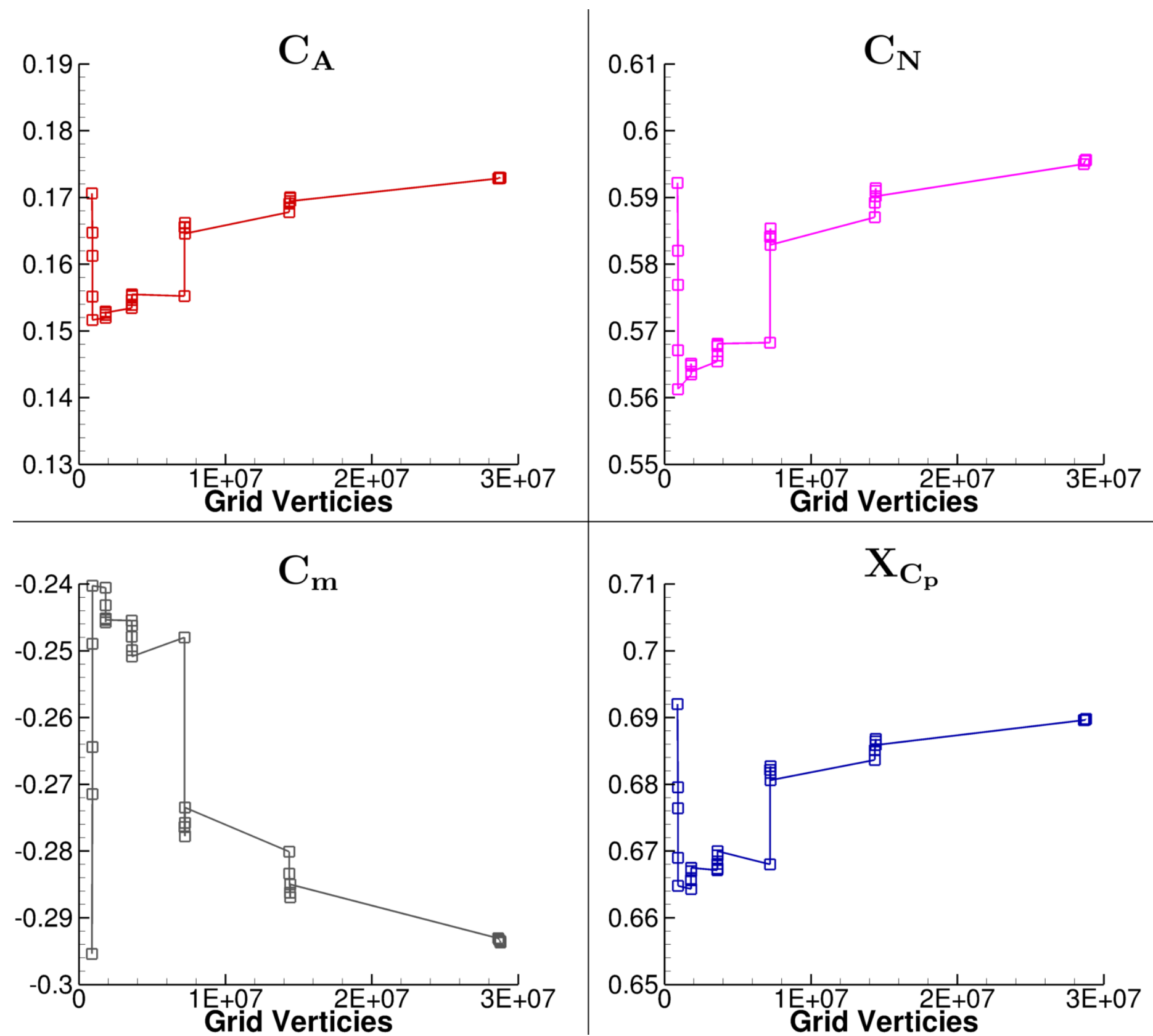


10° AoA Cp Profiles

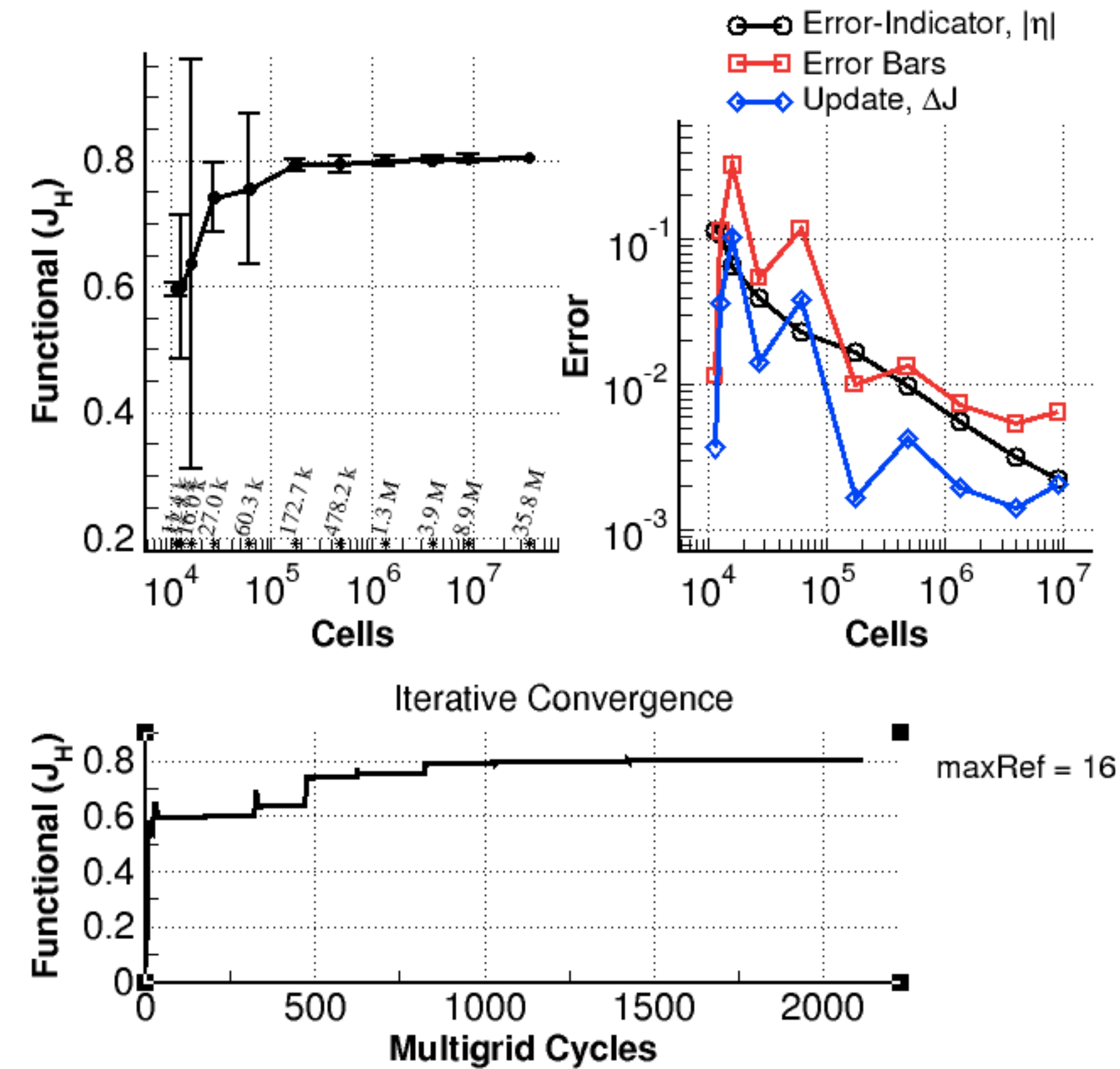


16° AoA Mesh Convergence

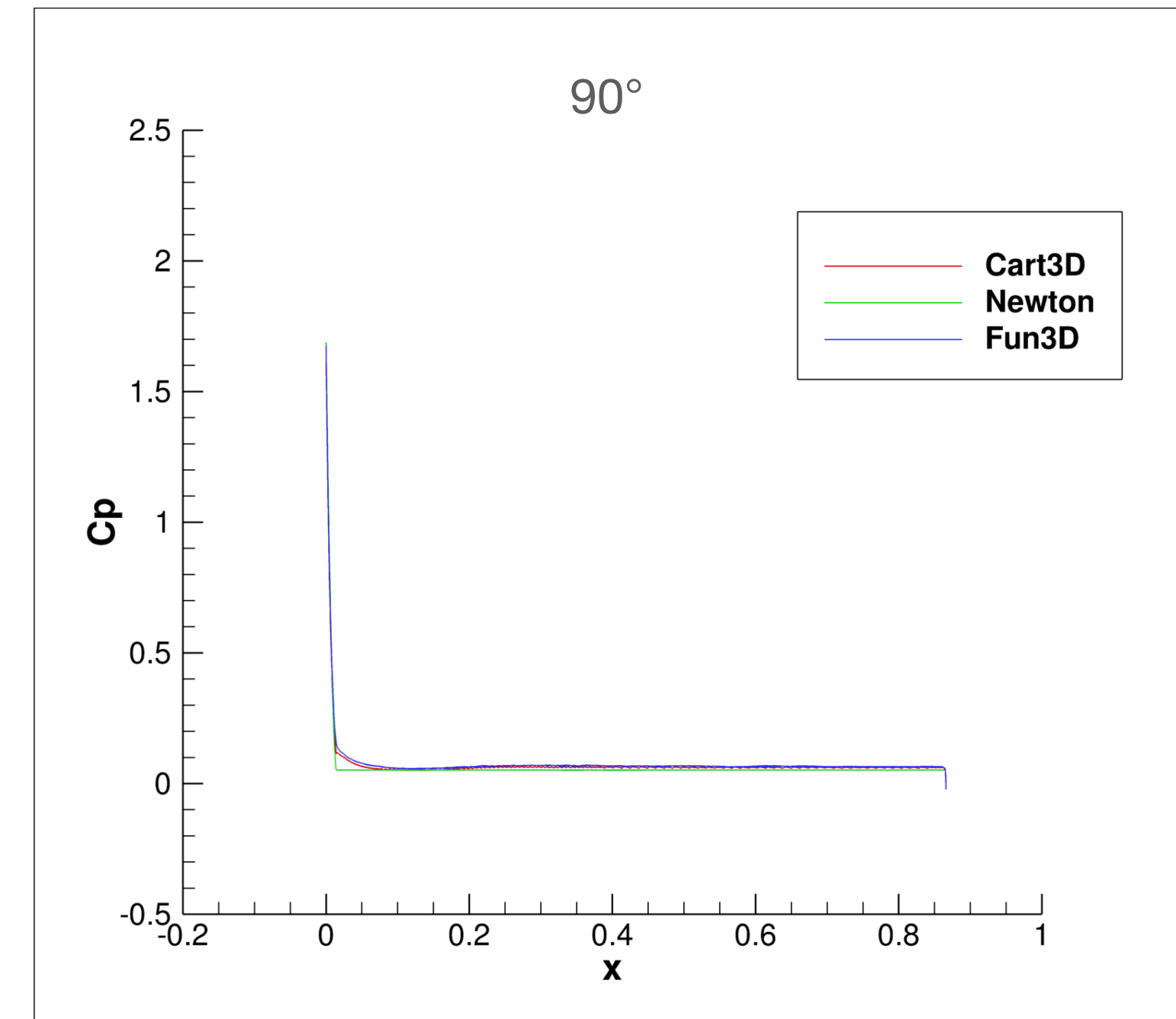
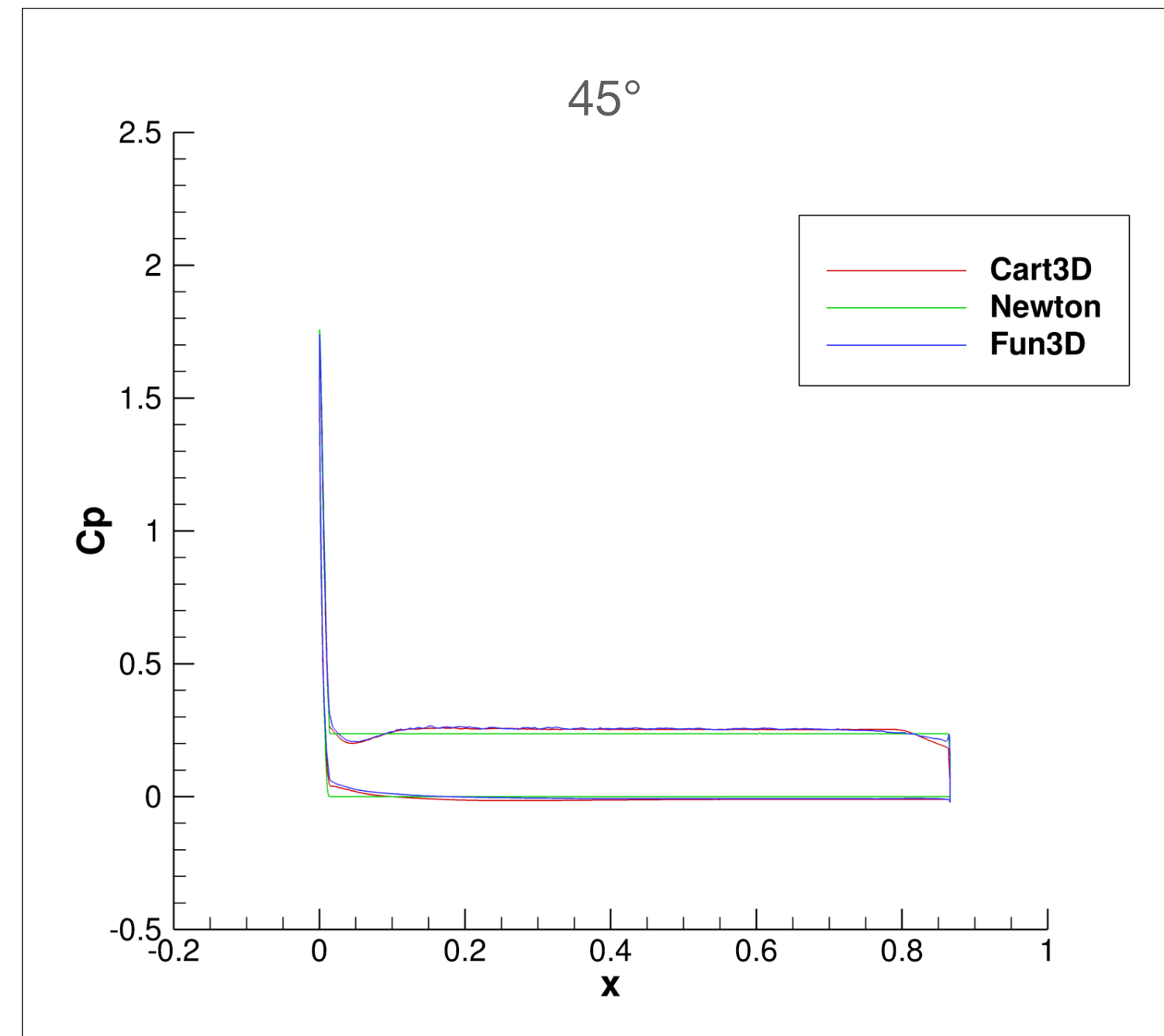
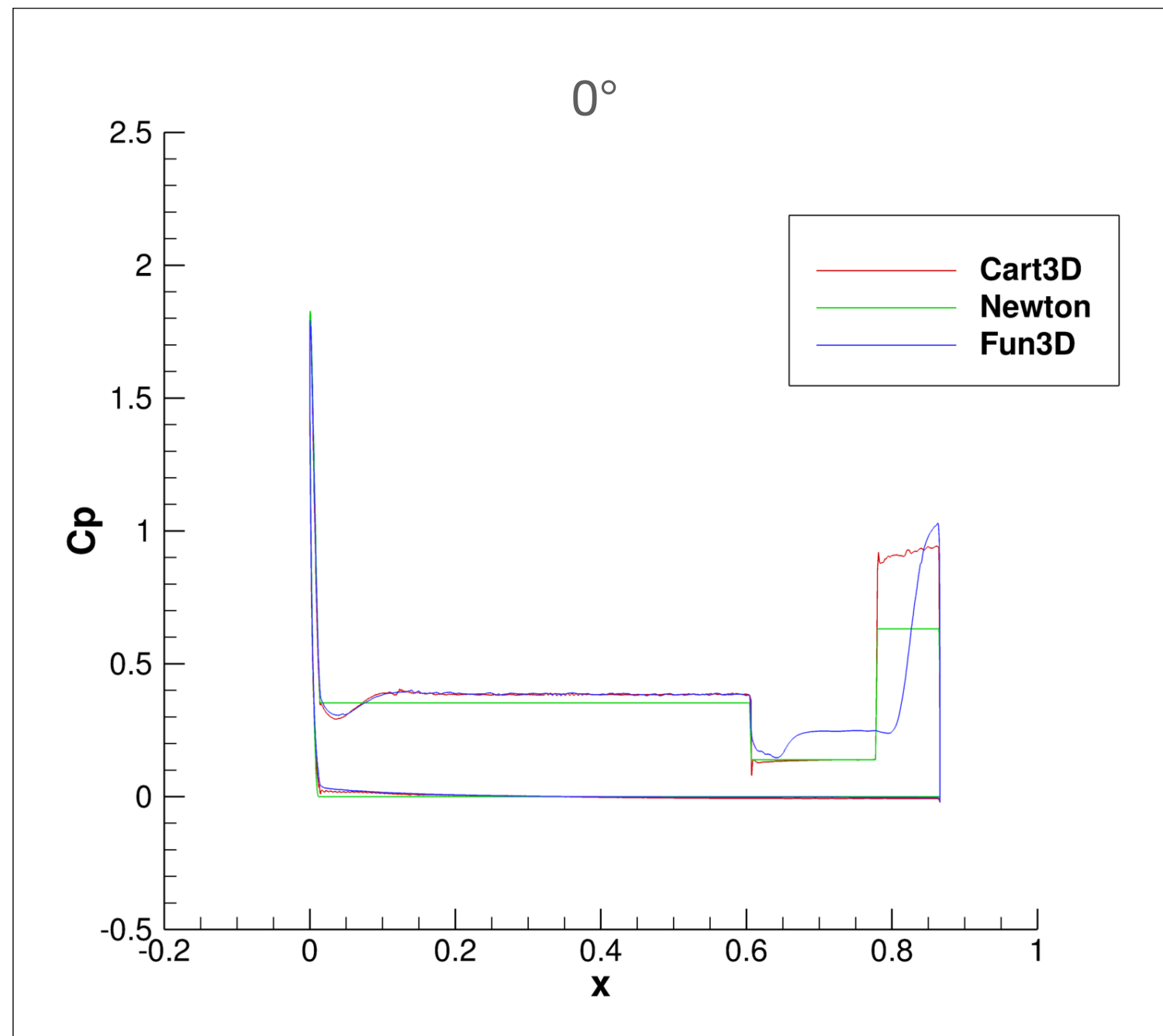
FUN3D



Cart3D



16° AoA Cp Profiles

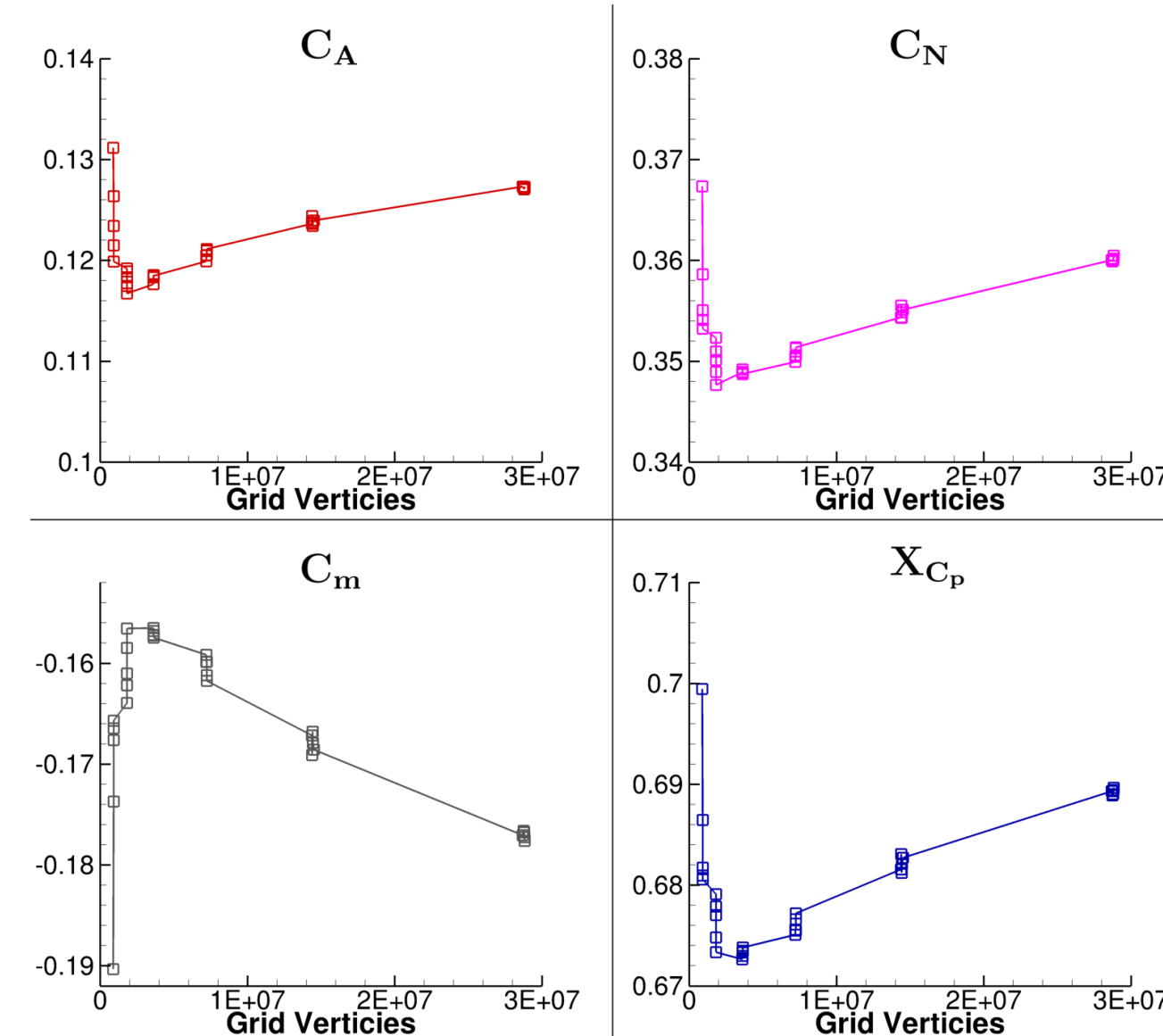


Runtimes and Grid Sizes

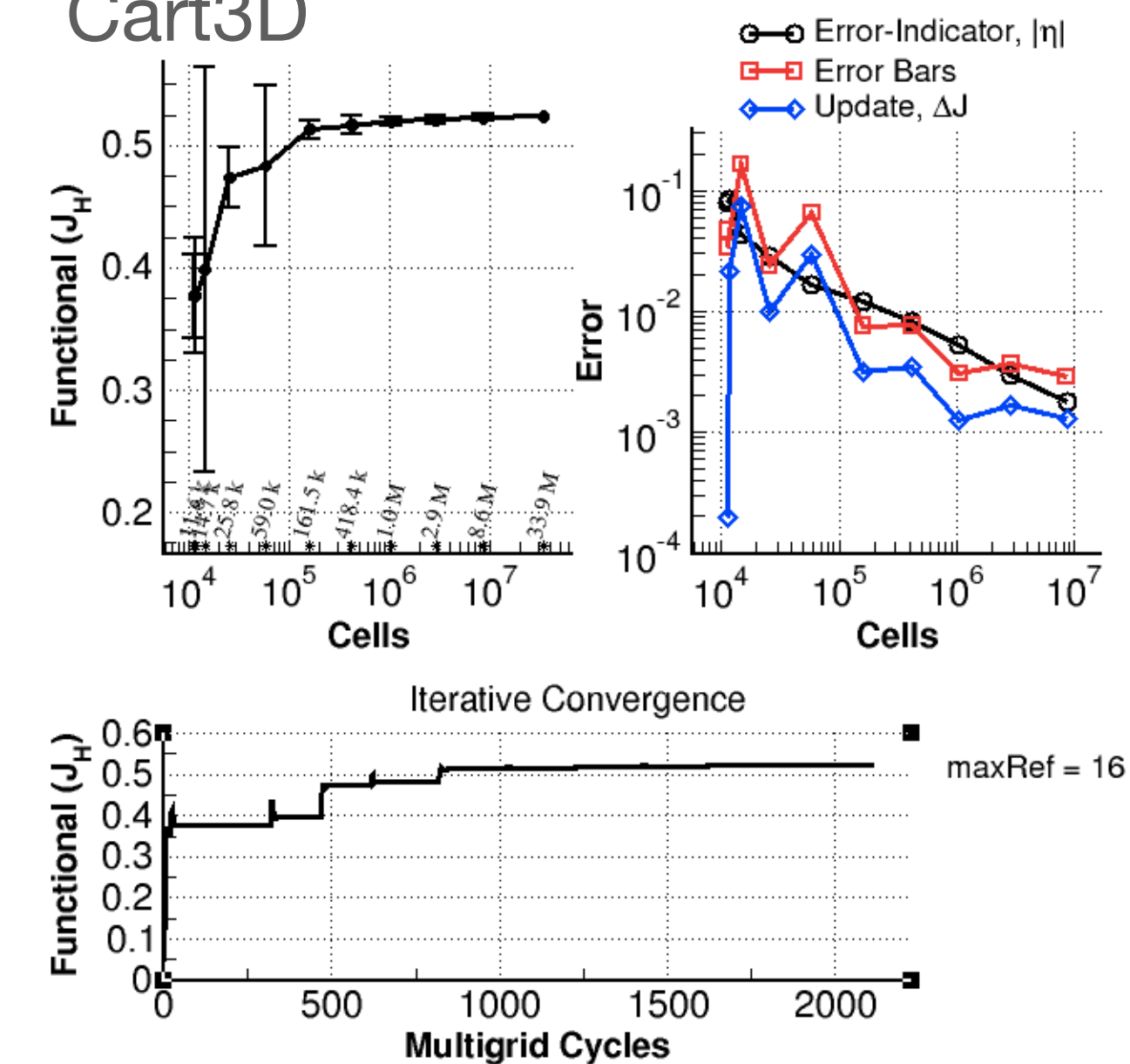
- FUN3D takes significantly more resources to run
- Cart3D is able to get broad-stroke results with quicker turnaround

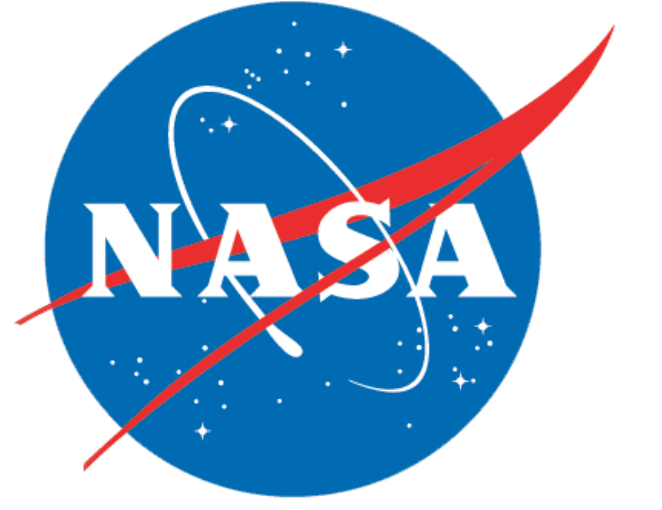
	Cart3D	FUN3D
Runtime	04:31	05:11
Nodes/Cores	1/40	50/800
Final Grid Size	34 million cells	168 million cells

FUN3D



Cart3D





Robust Autonomy for Ocean Worlds Exploration

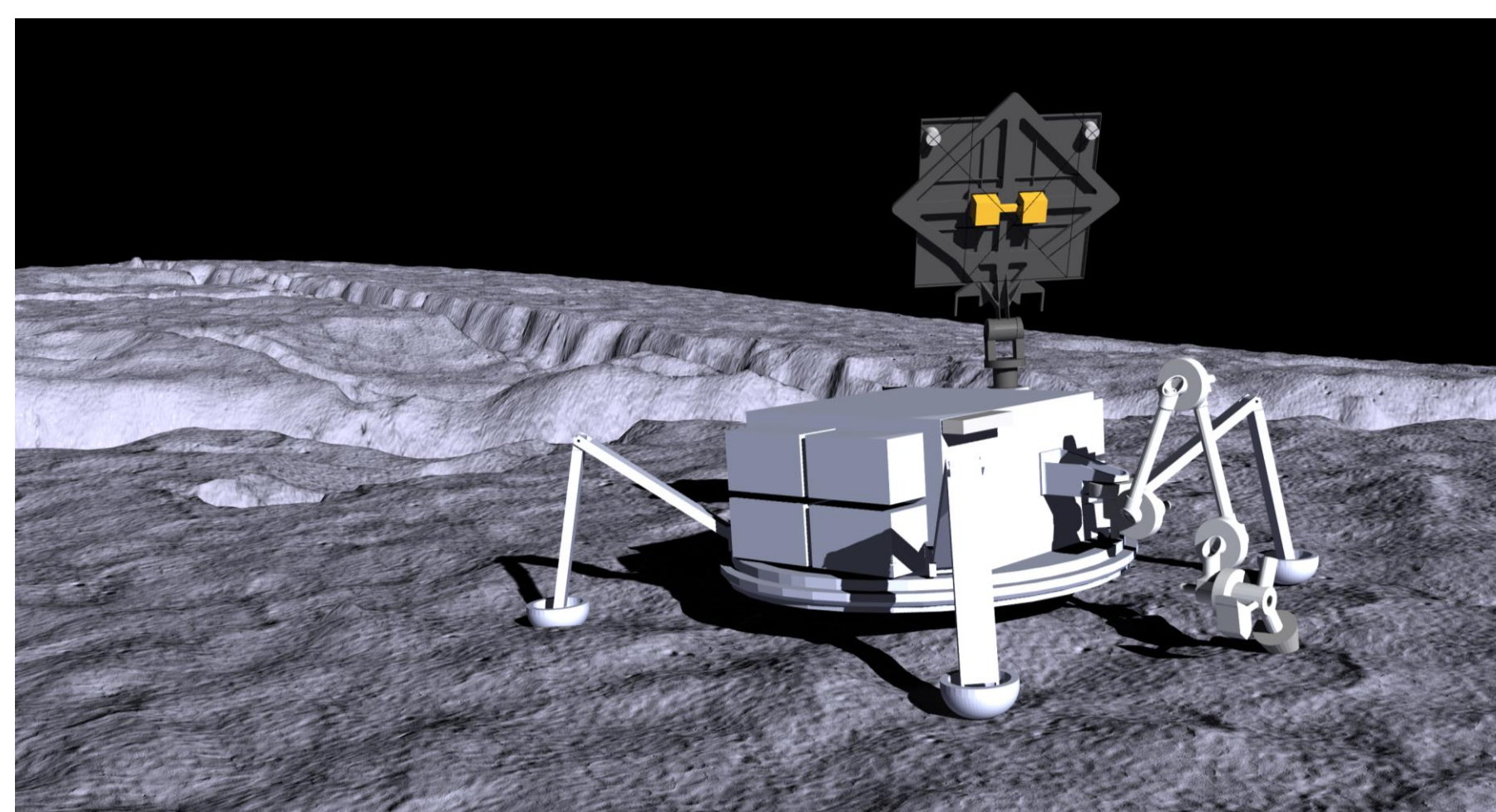
1. Abstract

Higher degrees of autonomy are a driving factor in advancements in robotics and space travel. While support exists to handle software and hardware errors, systems to recover from unexpected reboots are extremely rare. This is a desirable functionality for robotic systems since reboots may occur for any number of reasons (radiation, power, temperature) but while a program's state may reset after a reboot, any real-world effects will persist. Further, spacecraft are frequently in situations where waiting for ground control is time consuming or dangerous – the round-trip communication time with Europa is 1.5 hours¹, a lander can't see earth during the 42 hour night¹, and current proposals for a lander have no means to recharge the batteries², making waiting for communications mission-threatening. I have developed an extension to the PLEXIL language to allow information to persist across boots through a system built on "checkpoints". The result is an intuitive, semantically clear, and concise system where PLEXIL plans can retrieve information about earlier crashes, determine which points in execution had been reached, and store information persistently across reboots. I then used this system to modify the OceanWATERS reference mission plan to handle unexpected reboots. With these changes, the high-level autonomy can safely resume the mission without relying on ground control in the case of a systems failure.

2. Background

Plan Execution Interchange Language (PLEXIL) is a high-level programming language for representing plans for automation. The PLEXIL executive is a technology to execute plans on real or simulated systems. It allows interaction with the outside world through a system of interfaces – called adapters, where commands are sent to external programs and the plan can call lookups (with arguments) to query external state.

Ocean Worlds Autonomy Testbed for Exploration Research & Simulation (OceanWATERS) is a simulation of a lander on Europa. It has been developed as a testbed to aid the development of software for missions to ocean worlds such as Europa or Enceladus. PLEXIL plans are used to provide onboard autonomy for the lander and demonstrate lander features. An output from the simulator is pictured below³.



3. Objectives

- Extend PLEXIL executive to allow plans to determine if they had crashed
- Allow plans to determine what operations have previously been completed
- Use PLEXIL changes to implement crash recovery in OceanWATERS

Albert Kutsyy | University of Cambridge
Mentors: Michael Dalal, Dr. Jeremy Frank
Intelligent Systems Division (TI)

4. PLEXIL Enhancement

I have implemented a checkpoint-based information persistence system as a PLEXIL adapter. PLEXIL plans can asynchronously execute commands, which are simply passed to the relevant adapter. Similarly, plans can perform lookups (with arguments), which are also passed to the relevant adapter. I have determined that these facilities are sufficient for all persistence functions and created the Checkpoint Adapter.

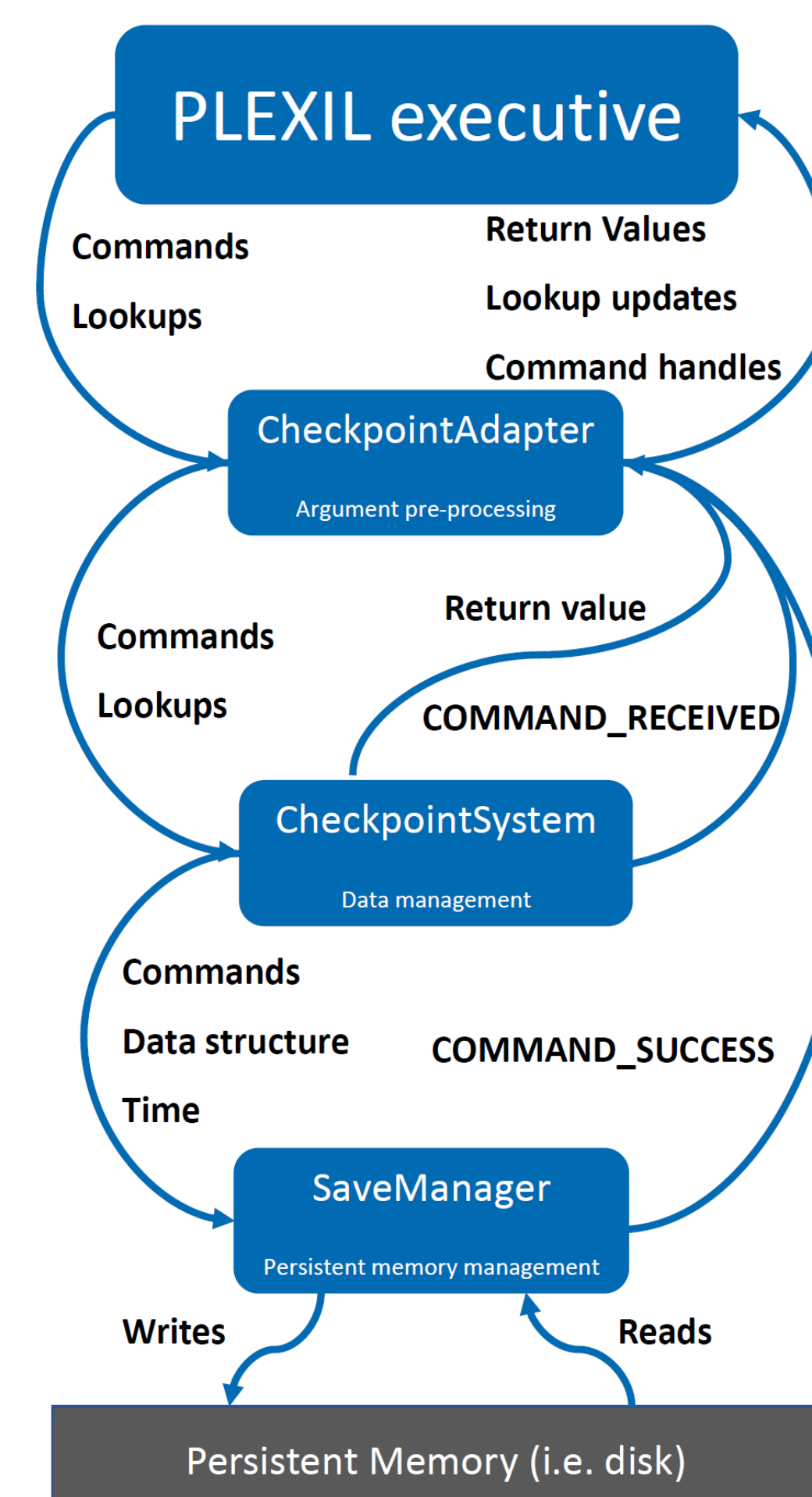
Three commands and many lookups are exposed to the user. *SetCheckpoint(name,value,info)* allows users to set or modify a checkpoint, *SetBootOK(value)* labels the current boot as safe to exit (automatically called on safe exit), and *Flush()* guarantees the current state of all checkpoints is saved to persistent memory. The user can also query the state, last modification time, and information associated with each checkpoint, both in the current boot and historically. Additionally, the user can query the number of logged boots, whether boots safely exited, and the startup/last active time for each boot. Several convenience functions are also provided as lookups.

Each checkpoint is associated with a boot number, either the current (boot 0) or a historical boot. The adapter utilizes PLEXIL command handles to indicate the state of commands – returning the handle "COMMAND_SUCCESS" upon saving the changes to persistent memory. The time and method of writing to memory, as well as the maximum number of saved boots is controlled by the SaveManager, a modularized subsystem which can be replaced to facilitate different persistence strategies.

5. Architecture

The Checkpoint Adapter consists of three major components – the adapter itself, the underlying checkpoint system, and the save manager. The adapter receives commands from the PLEXIL executive, replaces missing arguments with their default values, and calls the appropriate function in the checkpoint system. The checkpoint system maintains a data structure that tracks checkpoints and boots. After each command is processed, it indicates to the adapter that the command has been received and. The save manager can view the data structure maintained by the checkpoint system and is responsible for writing to persistent memory.

There are no constraints on when the save manager will write to persistent memory other than during a Flush command, making the checkpoint adapter flexible for use in different environments. A diagram of the adapter is pictured right.



6. OceanWATERS Modifications

The structure of the checkpoint system makes implementation a straightforward procedure. In the OceanWATERS Europa reference mission, I have added a block of start-up checks which run if there has been a crash – verifying the lander hasn't moved, checking the clock accuracy, and informing Earth of the crash, if possible. The mission sets a "completed" checkpoint after each operation and skips any operation for which a "completed" checkpoint exists. Each operation is then responsible for managing its own resumption. For example, the TakePanorama operation logs the position of the last-taken photo to a checkpoint. If the new panorama ordered is compatible (based on the requested imaging zone and the time), it resumes the panorama from the last photograph.

7. Conclusions

The PLEXIL additions have performed exactly as expected under all tests, including manual edge-case exploration and an automated stress-tester. Persistent memory usage is less than one kilobyte per boot, and mechanisms are in place for the save manager to optionally delete old boot information. The checkpoint adapter has proven to be a powerful addition and can allow for safe crash handling with minimal user overhead and no external crash-detection software. My additions to OceanWATERS have demonstrated that the checkpoint adapter can be easily added to existing plans to create robust but concise crash handling. The OceanWATERS additions have performed as expected. They allow the Europa reference mission to conduct a series of safety checks and then resume the last unfinished operation in a safe manner.

8. Acknowledgements

Even as a remote intern, it has been a great experience to be a part of the NASA team. I would like to especially thank Michael Dalal and Chuck Fry for their constant guidance and mentorship throughout the summer. I would also like to thank Dr. Jeremy Frank and Dr. Laurence Edwards for providing this internship opportunity.

9. References

1. Erickson, Kristen, et al. "In Depth | Europa." NASA Solar System, NASA, 19 Dec. 2019, solarsystem.nasa.gov/moons/jupiter-moons/europa/in-depth/.
2. Hand, K.P., Murray, A.E., Garvin, J.B., Brinckerhoff, W.B., Christner, B.C., Edgett, K.S., Ehlmann, B.L., German, C.R., Hayes, A.G., Hoehler, T.M., Horst, S.M., Lunine, J.I., Nealon, K.H., Paranicas, C., Schmidt, B.E., Smith, D.E., Rhoden, A.R., Russell, M.J., Templeton, A.S., Willis, P.A., Yingst, R.A., Phillips, C.B., Cable, M.L., Craft, K.L., Hofmann, A.E., Nordheim, T.A., Pappalardo, R.P., and the Project Engineering Team (2017): Report of the Europa Lander Science Definition Team. Posted February, 2017.
3. "Ocean Worlds Autonomy Testbed for Exploration Research & Simulation." GitHub, NASA, 2020, github.com/nasa/ow_simulator.
4. PLEXIL Wiki, 2020, http://plexil.sourceforge.net/wiki/index.php/Main_Page.
5. "Confluence project wikis". Internal documentation. NASA. n.d. Online.

Extending simulation testbed for autonomous vehicle research with differential-steering rover dynamics and virtual NASA Ames environment

Rory Lipkis, Stanford University
Ritchie Lee, ARC-TI (mentor)



Autonomous vehicle control

- Can be implemented with neural network controller:
 - Input: visual data from cameras (and possible additional sensors)
 - Preprocessing (downsampling and whitening)
 - Convolutional and hidden layers
 - Output: action / control input
- Drawback: "opaqueness"
 - Difficult to understand or explain controller behavior in terms of learned parameters
 - Impossible make strong guarantees about performance or safety

Adaptive stress testing (AST)

- Goal: detect and understand failure modes of a neural network controller
- Uses reinforcement learning with the objective of rewarding failure
- Computes perturbations to camera input that are maximally likely to induce controller failure
- Requires large number of test runs for comprehensive exploration
- Simulation necessary in order to reduce testing time and avoid potential vehicle damage accrued during live testing

Unreal Engine

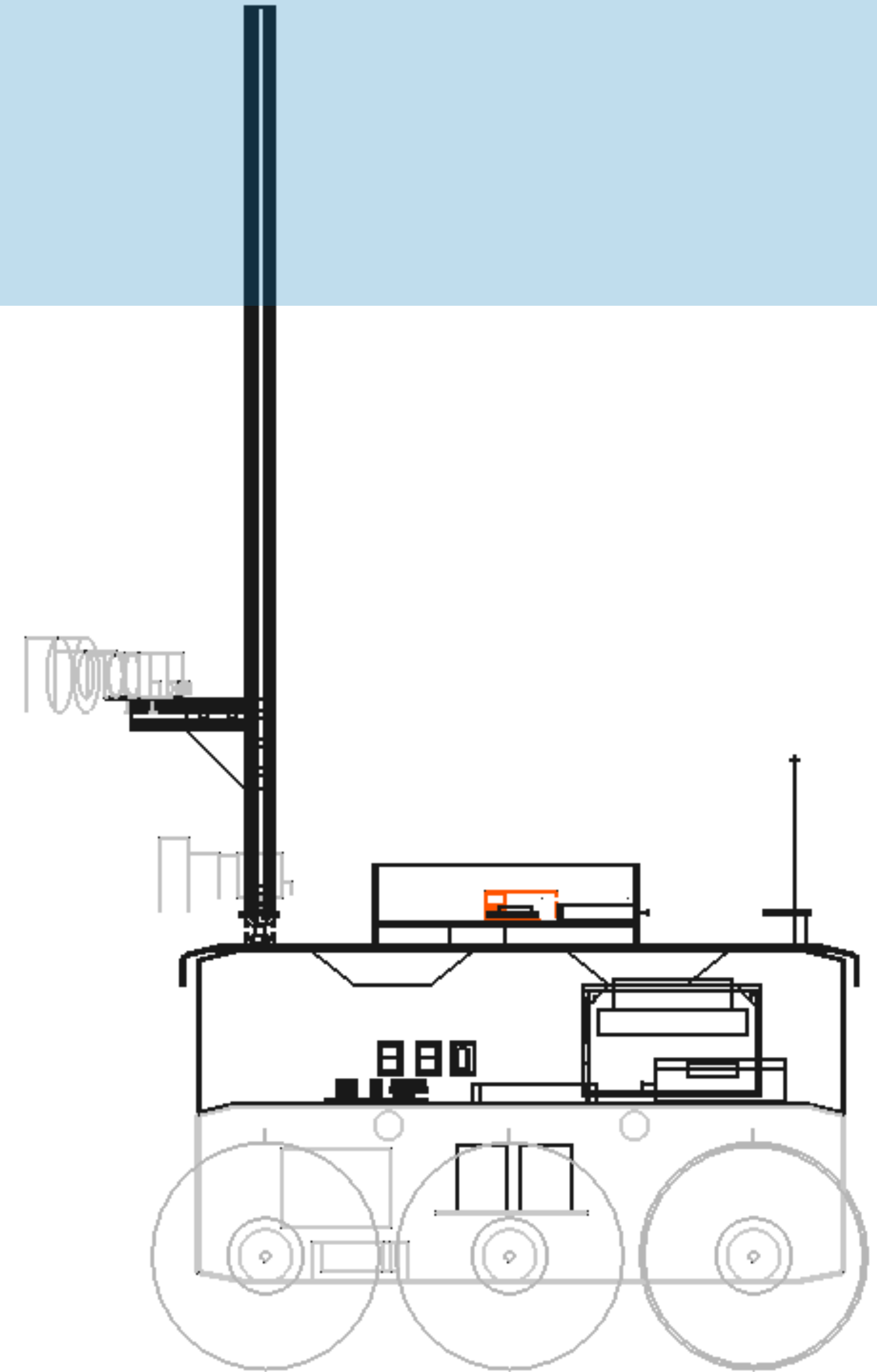
- Simulation platform
- Developed for video game design
 - Optimized graphics processing
 - High degree of modularity
- Allows generation of ultra-realistic camera inputs
- Simplifies creation of realistic and complex environments

Microsoft AirSim

- Exposes vehicle control API for use in control algorithms
- Camera inputs can be gathered and made available to external programs
- Control inputs can be computed by external programs and sent back to simulated vehicle
- Previously supported vehicles:
 - Car
 - UAV

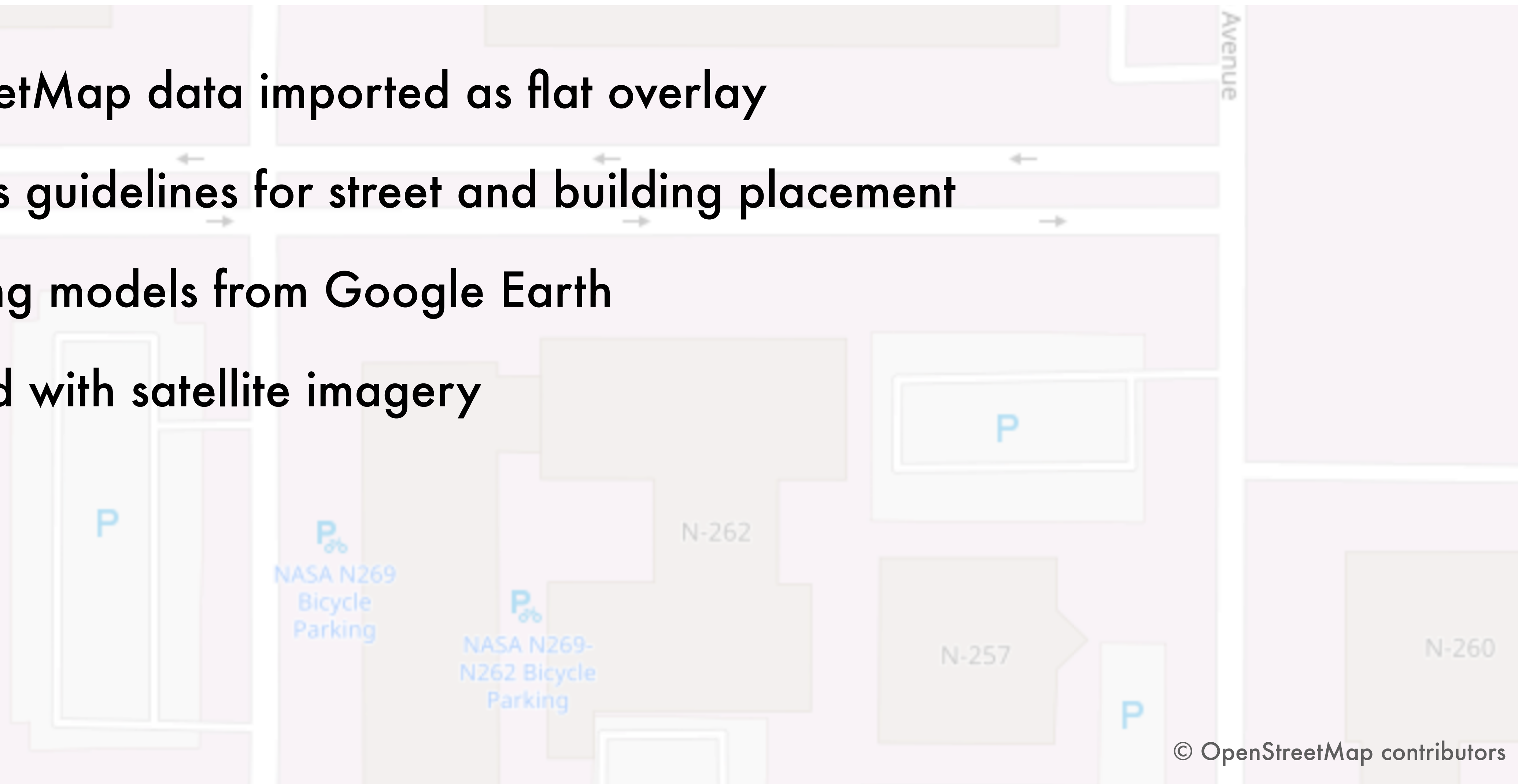
Rover dynamics

- Car:
 - Torque applied to front and rear axles
 - Steering accomplished by pivoting front wheels
 - Inherently coupled to forward motion
- Differential wheeled rover:
 - Torque applied to wheels individually
 - Steering independent of forward motion
 - Vehicle capable of turning in place



Realistic environment development

- OpenStreetMap data imported as flat overlay
 - Provides guidelines for street and building placement
- 3D building models from Google Earth
 - Textured with satellite imagery



Accomplished

- Extension of AirSim functionality to allow simulation of n -wheeled rovers with differential steering
- Creation of functional Unreal Engine rover pawn from existing 3D model
 - Complete specification and implementation of rover dynamics
 - Control interface callable with AirSim APIs
- Creation of virtual NASA Ames environment (vicinity of Intelligent Systems division) for vehicle simulation

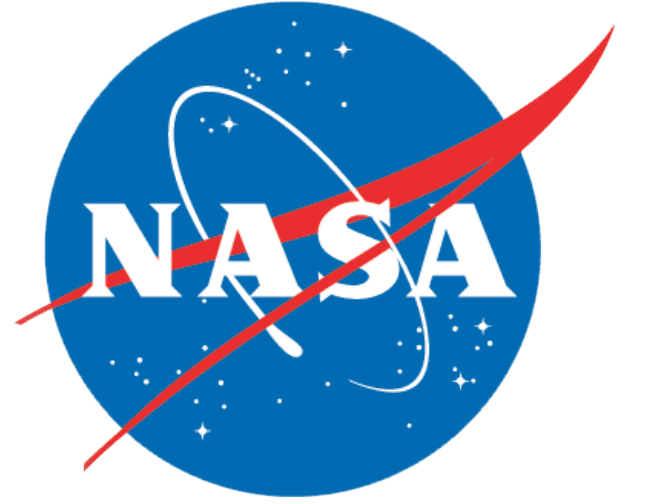
Remaining work

- Demonstrate full rover control from external C++ program
- Fully integrate rover into existing AirSim framework as distinct specifiable vehicle type for use by others
 - Refactoring code
 - Simplifying rover module to expedite compilation
- Increase simulation clock rate to allow for testing at scale

Acknowledgments

- Ritchie Lee
- Haley Feck
- Thomas Cannon





Data Storage Backend Server

Chi Kei Loi | San Jose State University | NIFS Intern

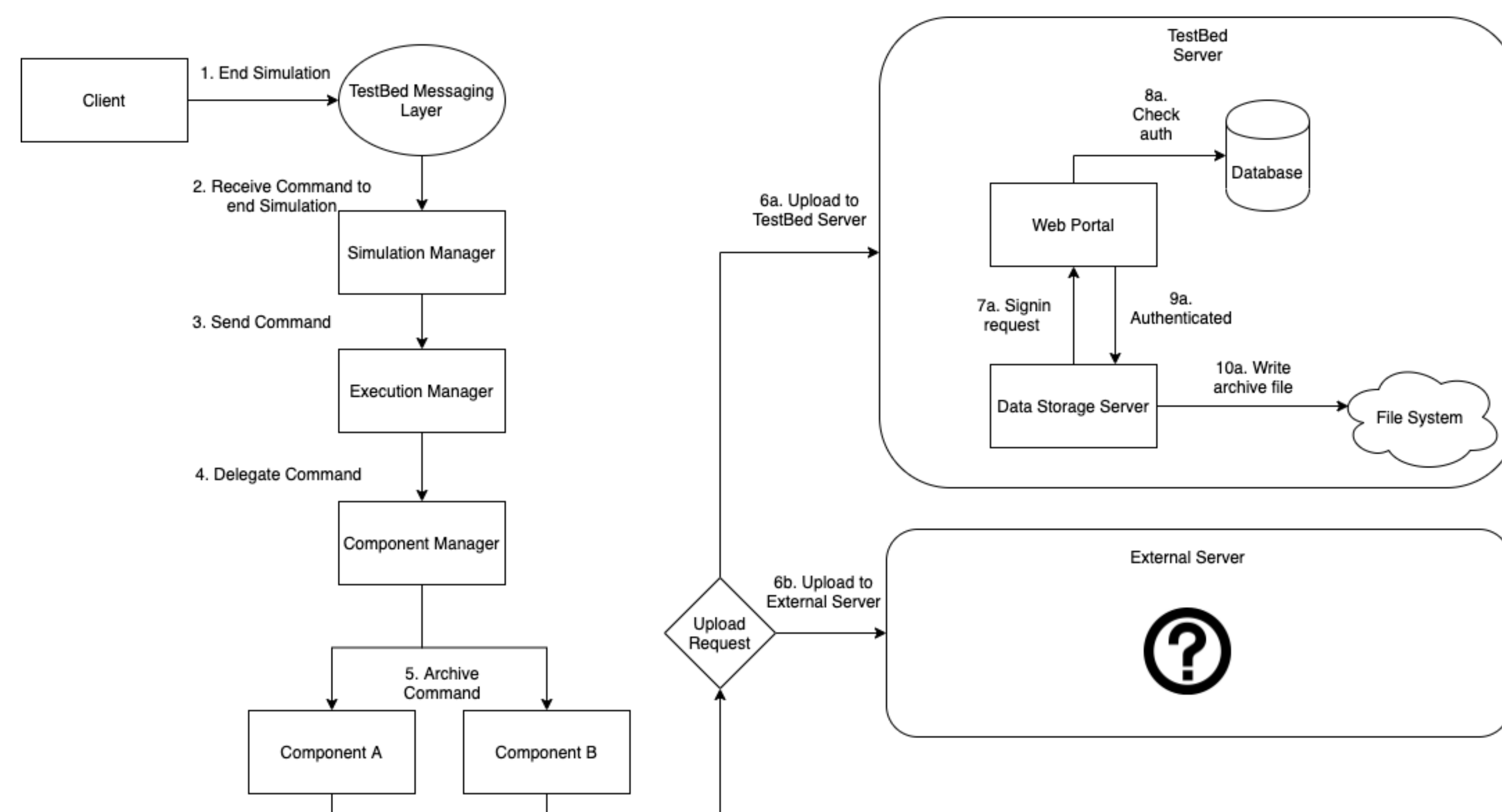
Background

Air traffic simulation is the main research area of SMART-NAS TestBed (SNTB). Being capable of storing simulation outputs is one of important feature that help researchers and developers to analyze the simulation result, so that they can learn from each simulation by tuning different parameters. Thus, we propose a new backend server to bookkeep each simulation output by storing them into a persistent file system in our lab machine. The data storage server has four endpoints to support upload, list-sim, list-file, download functionally. Each endpoint is protected by an authentication token to ensure that only registered TestBed users can use this server. The files uploaded by users are temporarily stored on our server for a maximum of X days. After X days, an email notification will be sent to the user so that the user can act on the expiring file. Lastly, hosting a separate backend service will provide lighter server load and data organization flexibility. It will also eliminate the need of hosting multiple service on every team that collaborate with Testbed.

Objectives

1. Provide API to upload, list, and download archived simulation data
2. Notify user when their files exceed temporary day limit
3. Ensure API is fully tested by writing test cases for each endpoint

Software Architecture



Implementation

Several open source libraries are used to implement this back-end server. The main libraries used are Node.js, Express.js and Multer. Node.js and Express.js are tools for handling http routing, and Multer is used to handle the upload and download of compressed files. In addition, this project uses ECMAScript 6 and confirms google style guides to make the code more modern and readable.



express



Testing

Unit Testing and Integration Testing are used in this project. To ensure each point performs correctly, each test cases must test against the http request parameters and queries using framework such as Mocha and Chai. In addition, a temporary file system and http interceptor are used to simulate the required services, so that developers can test against the required services without creating test fixtures.

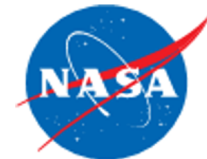
Future work

1. Provide cloud storage feature using third party service
2. Optimize upload download speed for concurrent environment

Acknowledgements

A special thanks to my mentor Kee Palopo and Huu Huynh, as well as Chok Fung Lai, Alan Lee, Jimmy Nguyen and Phu Huynh for their support and guidance throughout this project.





Smart Mobility Website

Andrew Moffatt, Umesh Krishnamurthy

Cal Poly Pomona, UC Merced

Mentors: Dr. Michael Stewart, Dr. Paul Fast, Dr. David Bell

Summer 2020

Overview

- Design Smart Mobility website for NASA's flight safety review process for testing at NASA Ames Research Center
- This will fast-track how teams will prepare for flight tests and support current work on Unmanned Aircraft System Traffic Management (UTM) and Small Unmanned Aerial Systems
- The current review process is very obtuse and needlessly complex. People who want to carry out flight missions are often unsure about whose approval they need to do so.
- A well-designed and intuitive website will have all necessary materials in one easily accessible location and completed documents will be sent directly to relevant personnel for review.

UAVs in Firefighting



Image source: <https://www.powerdms.com/blog/fire-department-drone-policies/>

UAVs in Search & Rescue



Image source: <https://www.youtube.com/watch?v=GkJ2NjQHys>

UAV Type vs. Mission Type

- Different types of UAVs are best-suited for different mission scenarios
- Multi-rotor UAVs are best for surveillance and imaging missions
- Fixed-wing UAVs are best for flying long distances and carrying heavy payloads
- We need a way of assessing what type of UAV is best for a given mission



<https://ukfiremag.mdmpublishing.com/selecting-a-drone-for-emergency-response-airframe-selection/>



<https://www.commercialdroneprofessional.com/silicon-valleys-new-fixed-wing-uav-designed-for-lidar-completes-test-flights/>

Testing Environments

Untethered UAV flight tests



Tethered UAV flight tests



UAVs and Ames

- CROB multirotor
- Tigershark small fixed wing
- Elios 2 indoor wind tunnel inspection
- Coastal mapping
- sUAS operations around rockets?

Flight Review Process

- Research branch needs a UAV for a mission. Discuss the UAV needed with flight ops
 - Why do you need this drone? How big should it be?
- Once the aircraft is purchased and delivered, JO takes possession and oversees assembly
- Proceed to flight test. Provisional approval for 4 tests:
 - Indoor tethered
 - Indoor untethered
 - Outdoor tethered
 - Outdoor untethered
- Deliver an oral presentation to flight ops about the mission

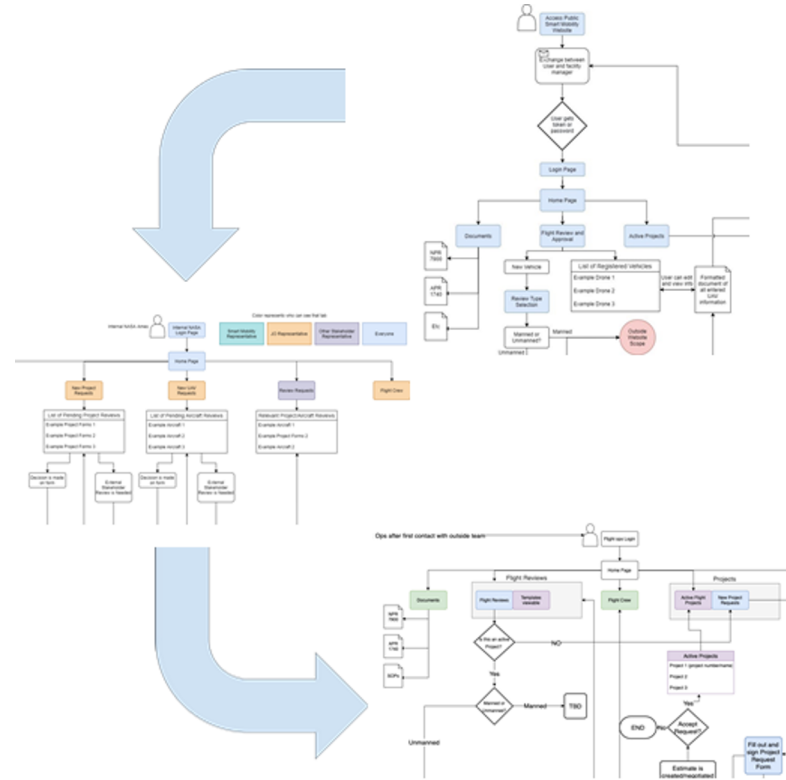
Challenges

- Importance of testing
- Review types and existing system
 - AFSRB
 - FRRB
 - Indoor
 - Tethered
 - Part 107
- Scope of Website to address issues

Methods

- Gather information on review process and relevant documentation
- Clearly define scope and goals of website project
- Create or improve existing user roadmaps
- Design page layout for user information input
- Prototype website layout and look

Example of iteration on website layout



Stakeholder Interviews

- Michael Stewart, Chief of Flight Operations
- David Satterfield, Chief of Maintenance
- Jonas Jonsson, Research Engineer
- Zachary Roberts, Senior Simulation Engineer
- David Murakami, Aerospace Engineer in Fluid Mechanics
- Nick Cramer, Project Manger for Spacecraft Autonomy
- Mark Sumich, Range Safety Officer in Aviation Systems
- Frank Aguilera, Chair of Airworthiness Review


Interview Findings

- There are too many people to contact, and requesters don't always know *whom* they need to contact.
- The presentations for airworthiness review often run several hours
- Approval often takes several months, making it very easy to miss deadlines
- Mounting deadlines result in lost time and funding
- Required documents are hard to find
- A public-facing calendar of what has already been scheduled would be great
- Email and in-person conversations are easy to forget, meaning reviewers can lose track of documents
- The long review process can turn away external contractors who may end up looking elsewhere to conduct flights

Design


- User interface prototyping
- Data collection interface
- Design site to clearly label sites and vehicles with appropriate information
- Mockup shows how user will input data
- Demonstrates navigation features
- Work to make as simple as possible to enter information
- Content of needed information is work in progress

Revise Submission




Matrice 600 (NASA Ames)
UAV: Un-Modified Off the Shelf
Mission (s)UAS Outdoor Test Range: Imaging (Valid for 8/20/2020 - 10/15/2020)
Flight Crew: Albert Taylor (External Pilot), Sarah Raymond (NASA RSO)
Part 107 Review APPROVED

Revise Submission




S1000 (External)
UAV: Modified Off the Shelf
Mission (Location TBD): Package Delivery
Flight Crew: TBD
AFSBB REVIEW INCOMPLETE

Revise Submission



Believer (External)
UAV: Modified Off the Shelf
Mission (s)UAS Outdoor Test Range: Fire Detection/Suppression (Valid for 8/20/2020 - 10/15/2020)
Flight Crew: TBD
Part 107 Review PENDING

New Submission



UAS System Description

- Size & Weight
- Propulsion & Power
- Performance
- Communications & Control
- Redundancies
- Technical Operations
- Flight Operations
- Security
- Range Safety
- Emergency Procedures

About Help Contact

Size and Weight

1) Size of UA. If you enclosed the UA in a box in its flight configuration (e.g. extended arms and props), how big would it be in **cm** (height x width x depth)?

Height: cm

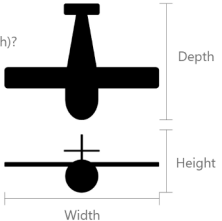
Width: cm

Depth: cm

2) What is the UA basic operating weight: kg

3) What is the UA maximum payload capacity: kg

4) What is the UA maximum gross takeoff weight: kg



PREV SAVE NEXT

Discussion/Conclusions

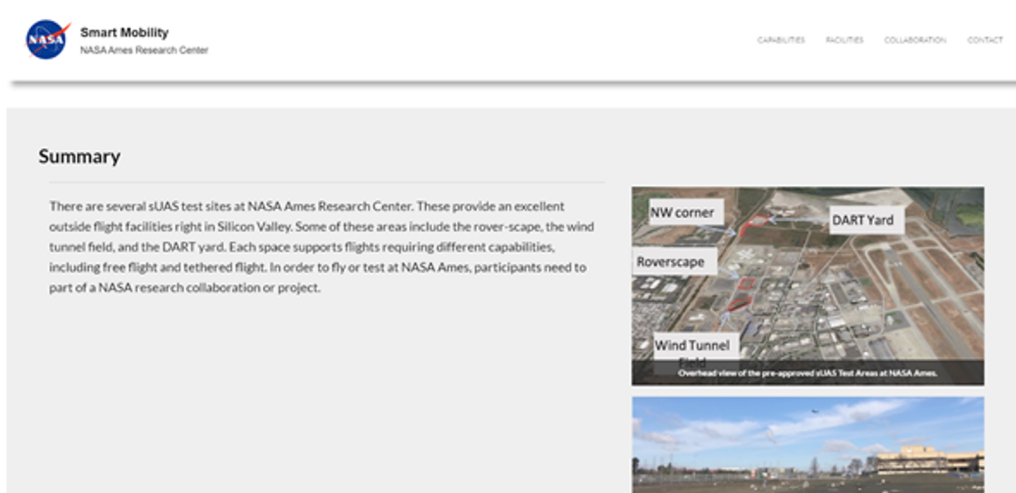
- Gathers and centralizes UAV and mission information
- Creates tool for facility managers and existing users to more easily fly at NASA Ames sites
- Streamlining process will make it easier to bring in new and more project teams both internal to NASA and external



Image Source: <https://www.nasa.gov/subject/9566/unmanned-aircraft/>

Future Research & Implications

- In process of finalizing mockups
- Translating the mockups and designs to URL test site
- Refine website road map
- Prototype internal facility manager website
- Gather more information and feedback on needed functionality



The screenshot shows the NASA Smart Mobility website. The header includes the NASA logo, the text "Smart Mobility" and "NASA Ames Research Center", and navigation links for "CAPABILITIES", "FACILITIES", "COLLABORATION", and "CONTACT". The main content area is titled "Summary" and contains a paragraph of text. To the right of the text is an aerial photograph of the test sites with labels for "NW corner", "DART Yard", "Roverscape", and "Wind Tunnel". Below the photograph is a smaller image showing a ground-level view of the facility.

Summary

There are several sUAS test sites at NASA Ames Research Center. These provide an excellent outside flight facilities right in Silicon Valley. Some of these areas include the rover-scape, the wind tunnel field, and the DART yard. Each space supports flights requiring different capabilities, including free flight and tethered flight. In order to fly or test at NASA Ames, participants need to part of a NASA research collaboration or project.

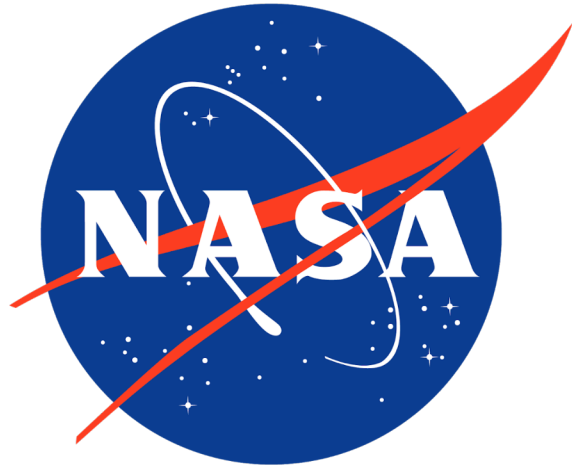
Overhead view of the pre-approved sUAS Test Areas at NASA Ames.

Lessons Learned

- Basics of Adobe XD
- The beginning steps in Web design
- How to gauge user needs in the design process
- How to simplify complex procedures
- How competitive and fast paced the private UAV market is
- Adapting to remote work

Acknowledgments

- Dr. David Bell, NAMS Director, USRA
- Michael Stewart, NASA
- Jonas Jonsson, NASA
- Zachary Roberts, NASA
- Saba Hussain, USRA
- Zeal Panchal, USRA
- NAMS RD Student Program





Comparison of Bounding Box and Semantic Segmentation Neural Networks for Trail Recognition

Abed Musaffar^a, Nathan Scheinkman^b, Dr. Adrian Agogino^c

^a Department of Mechanical Engineering, University of California Santa Barbara, CA

^b Department of Mechanical Engineering, California Polytechnic University, San Luis Obispo, CA

^c Research Scientist, NASA Ames Research Center, Moffett Field, CA

ABSTRACT

Vehicle navigation based on image sensors is an important part of autonomy in transportation. While significant research has been performed in urban environments, less has been performed in non-urban environments that are more relevant to domains such as search and rescue and navigation on planetary missions. This poster compares two approaches to this problem 1) Semantic segmentation, and 2) Object detection. Both these methods are trained using hand-labeled data on an unlabeled dataset from the Swiss alps. Performance results from each approach on a “field test” show that they are promising methods for non-urban navigation.

OBJECTIVES

- Create a trail recognition dataset using both bounding boxes and semantic segmentation
- Train a trail recognition model using both YOLOv3 and Google Deeplab and comparatively analyze their performance
- Fine-tune the models in order to maximize performance while preserving ability for general use
- Field-test models using videos and eventually life feed to test feasibility for future self-guided drone implementation

BACKGROUND

In 2015, a team of researchers used a simple classification model to train a drone for semi autonomous flight. However, the model they used would crash the drone if left unattended for more than ten minutes due to model inaccuracies. With the advent of modern one-step inference models such as Google’s Deeplab and Darknet’s YOLOv3, it becomes possible to build more complex yet lightweight computer vision models. These models offer the opportunity for increased accuracy and detail, and when combined, open the possibility of redundancy and combined accuracy in model predictions and applications.

BOUNDING BOX



- LIBRARY: Keras
- MODEL: YOLOv3
- mAP: 0.90
- Although this model was able to train and accurately classify the trails, the issues found were that it was difficult to determine how to label the trail (entire trail or center of it). Despite this setback, the model was very robust, accurately identifying unseen trails, even those dissimilar to the training data

SEMANTIC SEGMENTATION



LIBRARY: Tensorflow
MODEL: Deeplab
mIOU: 0.81

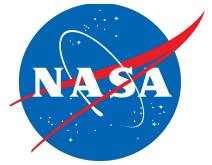
This model produced detailed and verifiable output. On trails similar to the training dataset, it could quickly and reliably “draw” the trail in real time with very good accuracy, as shown above. However, it is much less robust. It struggled to identify any trails dissimilar to the training data.

CONCLUSION

The results suggest the best solution to be at the crossroads of the above prediction methods. Using semantic segmentation to create trail masks will give valuable information about the trail shape/size while a bounding box can then give information about trail navigation (such as going left, right, straight). It is important to note that the implementation of both these models could result in a more complex and longer inference and training.

REFERENCES

1. Redmon, J., & Farhadi, A. (2018). Yolo3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.
2. Giusti, A., Guzzi, J., Cireşan, D. C., He, F. L., Rodríguez, J. P., Fontana, F., ... & Scaramuzza, D. (2015). A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2), 661-667.
3. <https://github.com/experiencor/keras-yolo3>
4. <https://github.com/tensorflow/models/tree/master/research/deeplab>
5. Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 801-818).



Have We Tested Enough? Using Bayesian Deep Learning to Estimate Test Coverage

Chelsea Sidrane, Stanford University

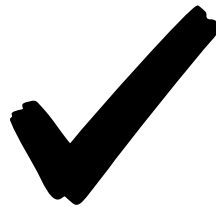
Mentor: Ritchie Lee, Code TI, RSE Group
Summer 2020

Introduction

- When you are doing testing, it's difficult to know if you have tested the system *enough*
If you find a failure, maybe – but what if you don't find a failure?

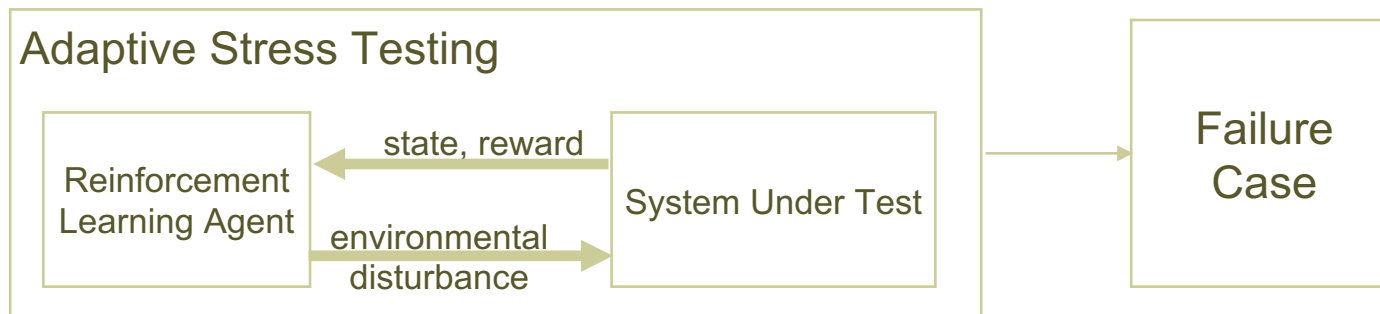


- In this project I attempt to produce a *coverage guarantee* for the system under test using Bayesian deep learning



Literature Review

- **Adaptive Stress Testing** is a technique used to find failures in complex, black-box systems [1, 2]
- Adaptive stress testing formulates testing as a **reinforcement learning** problem, where the reinforcement learning agent controls any stochasticity or environmental disturbances in the system under test in attempt to find failure cases



This approach is flexible and scalable, but doesn't give any **testing coverage** guarantees

Research Questions

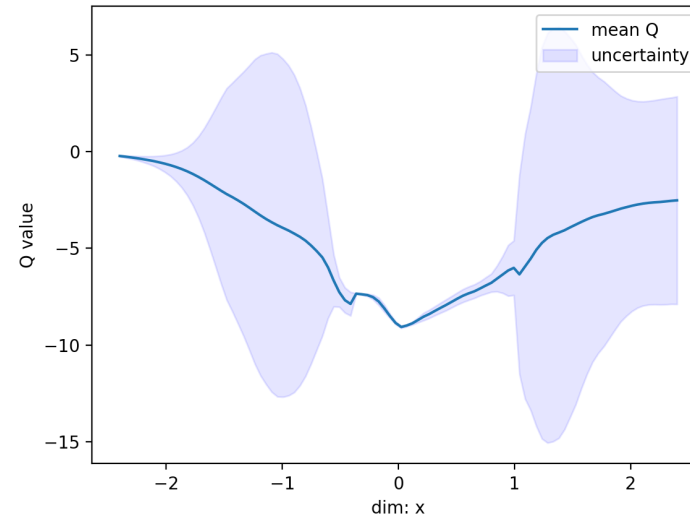
- Can Bayesian deep learning be used to produce reliable estimates of testing coverage?
- Specifically, can this method accurately differentiate regions of the system that are **safe** from regions that require **more testing**?

Methods

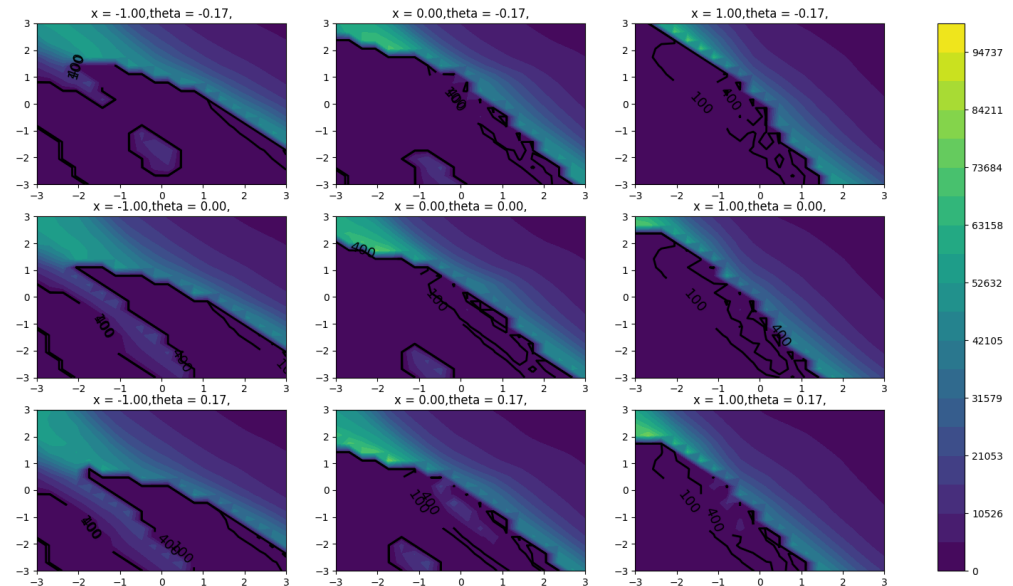
- In this project, approximate Bayesian deep reinforcement learning with the SUNRISE method is used [3]
- Multiple neural networks are used in an **ensemble**
 - Each network is trained on a different, randomly sampled, subset of the data
 - The variance across the ensemble of networks is taken as the **confidence level**
 - **low variance == high confidence**
 - **high variance == low confidence**
 - **More testing** is required in the regions of state space where the **confidence is low**

Findings

- Qualitatively, the method produces Bayesian-like confidence estimates
- We are still in the process of verifying the accuracy of these confidence estimates, i.e. where the neural network ensemble is very confident, does the output match the ground truth?



Q contour plot: stdx axis: xdot, y axis: theta_dot



Discussion/Conclusions

- This direction of work seems promising
- The field of testing needs more work combining scalable algorithms with guarantees

Future Research & Implications

- If this method is successful, it can be used anywhere that adaptive stress testing is currently used, and can give test engineers more confidence in the results of adaptive stress testing
- One future direction of work is to use policies produced by adaptive stress testing to more accurately estimate the failure rate of a system under test in nominal conditions

Lessons Learned

- I learned that research prototyping can produce unexpected results

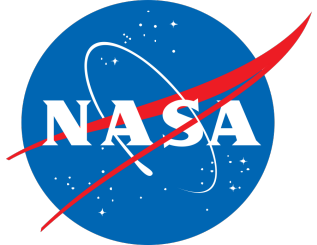
Acknowledgments

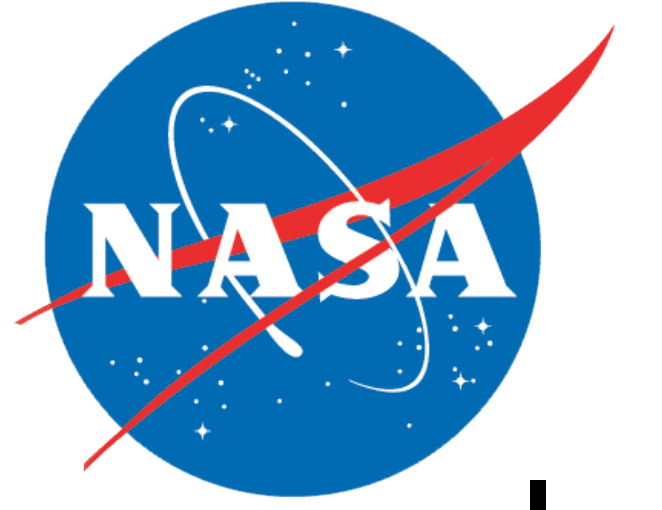
- Thanks to my mentors, Ritchie Lee and Susmit Jha, to the whole RSE group, and to my funding source, the Universities Space Research Association

References

- [1] Lee, R., Kochenderfer, M. J., Mengshoel, O. J., Brat, G. P., & Owen, M. P. (2015, September). Adaptive stress testing of airborne collision avoidance systems. In *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)* (pp. 6C2-1). IEEE.
- [2] Corso, A., Moss, R. J., Koren, M., Lee, R., & Kochenderfer, M. J. (2020). A Survey of Algorithms for Black-Box Safety Validation. *arXiv preprint arXiv:2005.02979*.
- [3] Lee, K., Laskin, M., Srinivas, A., & Abbeel, P. (2020). SUNRISE: A Simple Unified Framework for Ensemble Learning in Deep Reinforcement Learning. *arXiv preprint arXiv:2007.04938*.

Thanks! Any Questions?





Preliminary Characterization of 3D Printed Zeolite 13X Sorbents for Carbon Dioxide Removal

Yanizbeth Yambo-Chevere¹, Nathan Doi², Tra-My Justine Richardson^{3*}

University of Puerto Rico – Rio Piedras¹, San Jose State University², NASA Ames Research Center³

Abstract

The Carbon Dioxide Removal Assembly (CDRA) aboard the International Space Station (ISS) uses pelletized zeolite molecular sieves to remove carbon dioxide (CO₂) from cabin air. CDRA is currently sufficient for low earth orbit missions, but the sorbent bed remains plagued with routine maintenance due to dust generation and thermal inefficiencies stemming from the use sorbent pellets. Recently, 3D printed structured sorbents for CO₂ removal have been investigated as an alternative to pelletized sorbents [1-3]. Here at NASA Ames Research Center, we are currently investigating the potential use of 3D printed zeolite 13X based structured sorbents to replace conventional sorbent pellets to alleviate these concerns for future long-term manned missions where resupply is unavailable. 3D printed zeolite 13X samples showed similar surface morphology to their pellet form and carbon dioxide adsorption capacity for the 3D printed samples was slightly less than the pellet but comparable. TGA plots revealed that the organic binders added to the 3D printing paste are

Objective

3D print zeolite 13X based sorbent structures (ribbons) and physically characterize the printed samples for comparison against commercial zeolite 13X pellets and 3D printed structures prepared and reported by Thakkar et al [1].

Method

In this study, zeolite 13X sorbent structures were printed using an Ultimaker® 3D printer and the R4 zeolite mixture recipe reported by Thakkar *et al.* [1]. The samples were characterized via scanning electron microscopy (SEM), thermogravimetric analysis (TGA) (TA Instruments – TGA 5500), and a porosity and surface area analyzer (Micrometrics ASAP 2020).

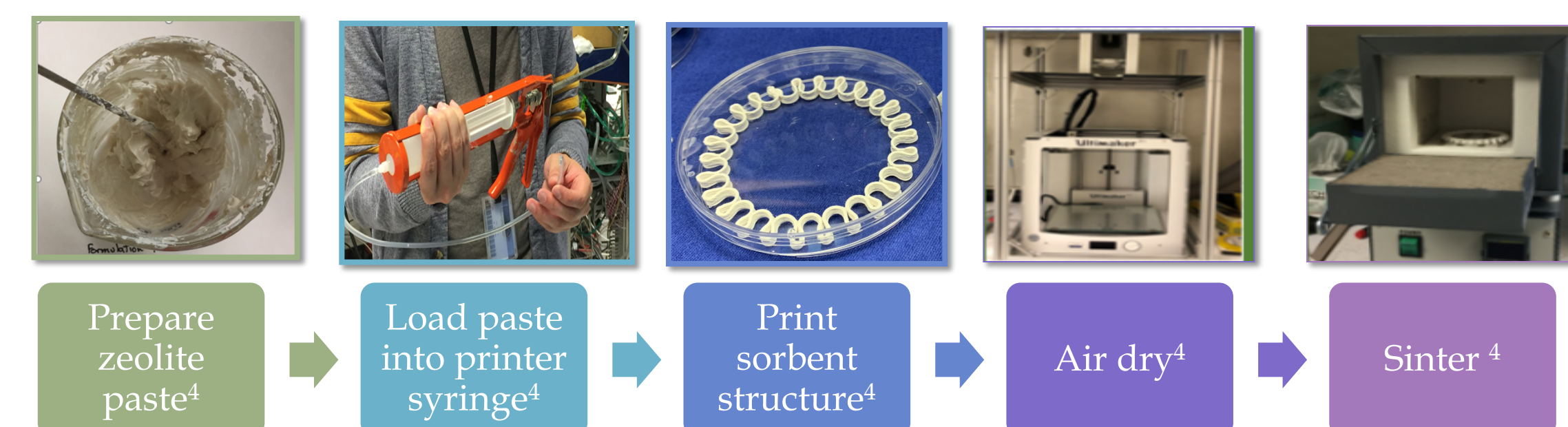


Figure 1: Methodology for 3D printed 13X zeolite monoliths. ⁴

Results & Discussion

- As shown by SEM results, the surface morphology of the zeolite 13X pellet (left) and 3D printed sample (middle and right) appear to be similar, suggesting that the 3D printing process did not alter the structure.
- According to carbon dioxide isotherms measured for 3D printed zeolite 13X sorbents, the sample with 47.5% solids mass has the highest CO₂ adsorption capacity and BET surface area
- 3D printed zeolite 13X sorbents have slightly lower CO₂ adsorption capacity than their pellet form.

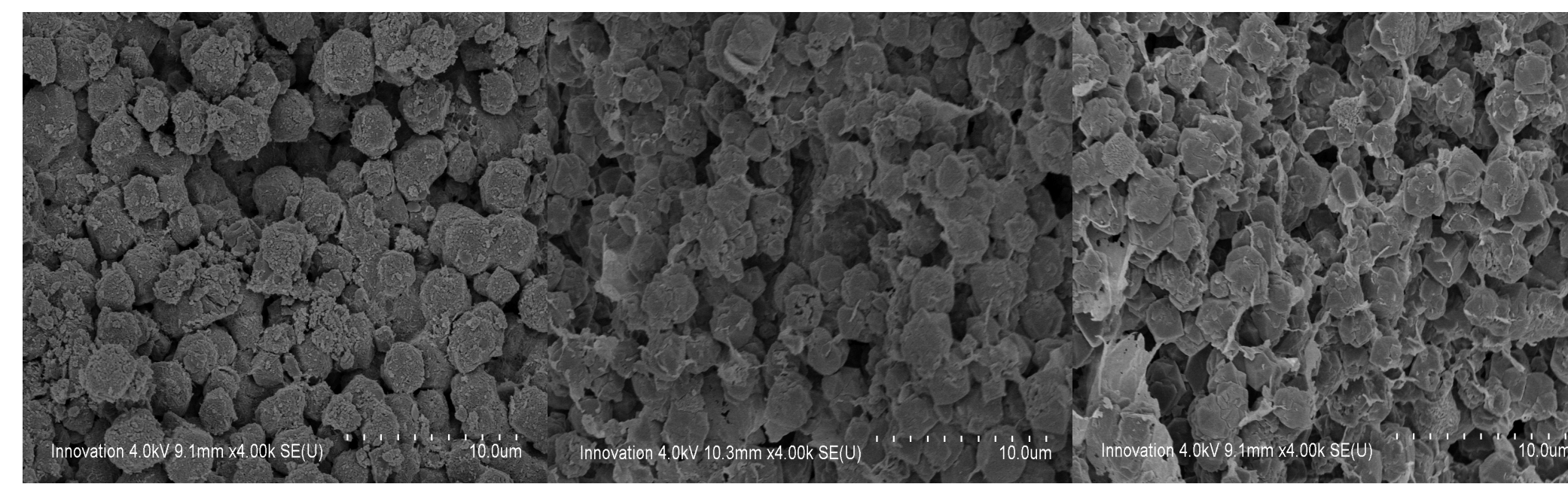


Figure 2: SEM images of the 13X pellets and R4-13X printed pellets. ⁴

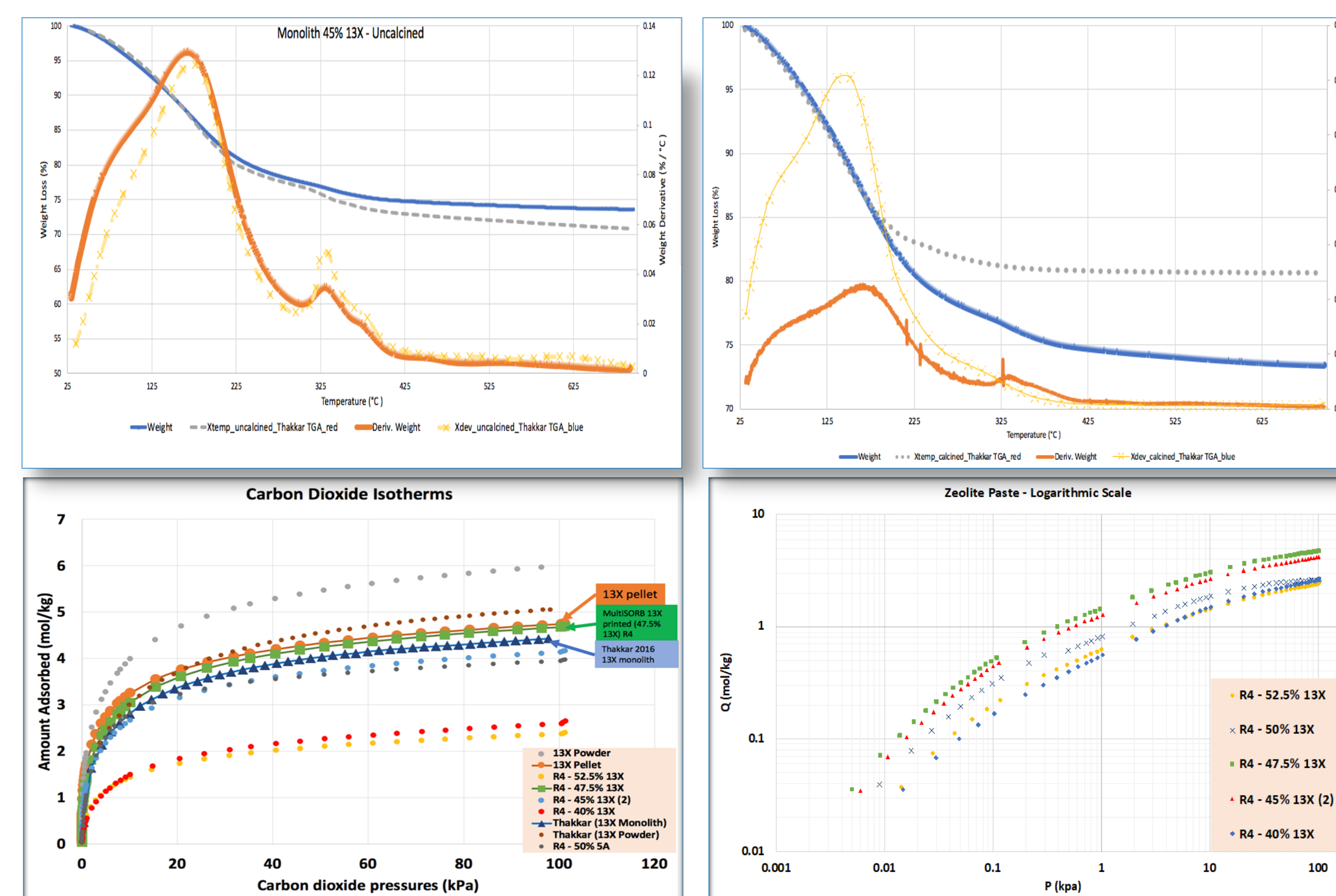


Figure 3: TGA analysis and CO₂ isotherm, on 3D printed multifunctional sorbent. ⁴

Percent Solids	BET Surface Area	Pore Volume	Pore Diameter(Å)	CO ₂ Adsorption Capacity
40%	134.3242	-	6.914	0.3039
45%	220.6183	0.005654	6.356	0.7745
47.5%	250.5108	0.008566	6.396	0.8827
50%	153.8685	0.006719	6.518	0.5631
52.5%	123.5304	-	6.592	0.3648

Conclusions and Future Work

- 3D printed zeolite 13X sorbents in this study had slightly less, but comparable, CO₂ adsorption capacity in comparison to the commercial zeolite 13X pellets.
- Additionally, CO₂ adsorption capacity for our 3D printed sorbents were higher than published values.
- 3D printed zeolite 13X sorbent structures show potential but require further optimization of the paste formulation, thermal and fluid modeling, sorbent geometry, and heater selection and integration.
- Continue to collaborate with Robocasting® to produce high resolution and accurate prints.
- Conduct a literature review and determine candidate heating materials including 3D printable heaters.

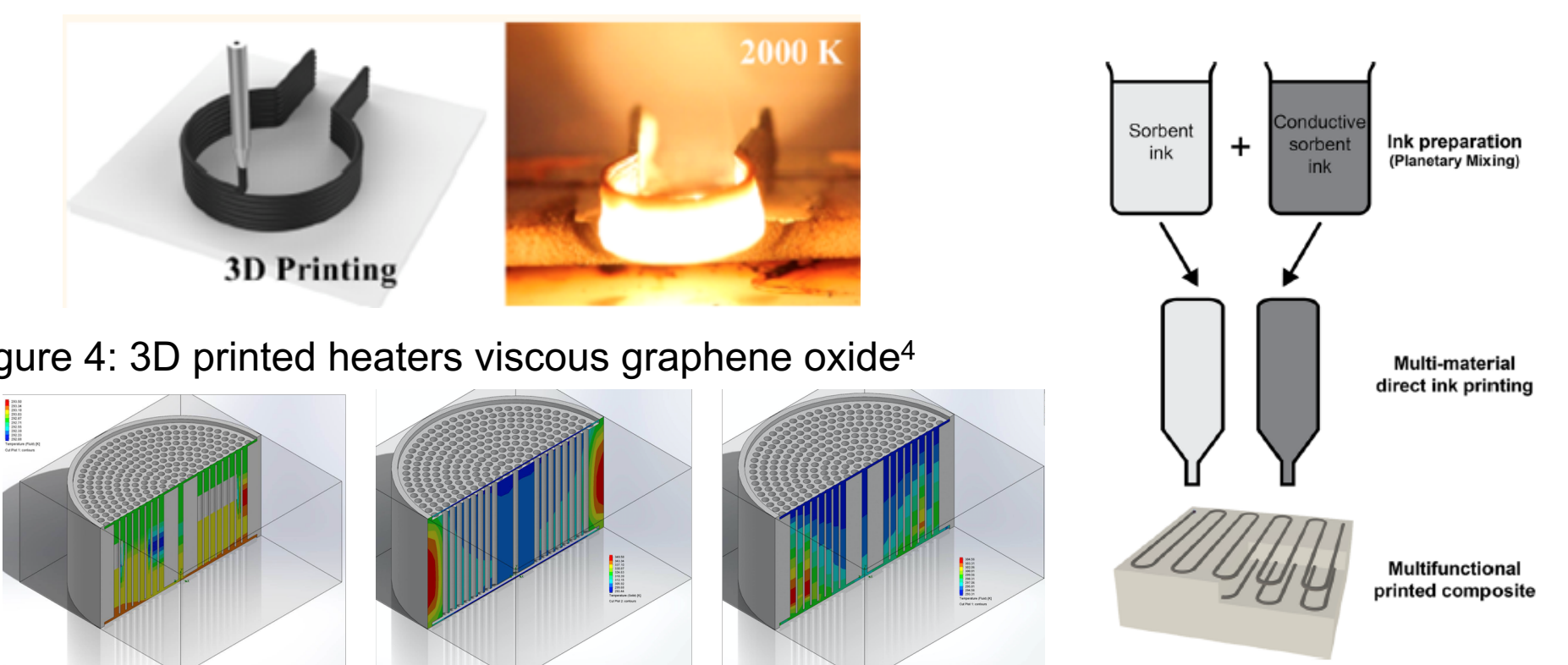


Figure 4: 3D printed heaters viscous graphene oxide ⁴

Figure 5: Embedded heaters preliminary thermal modeling ⁴

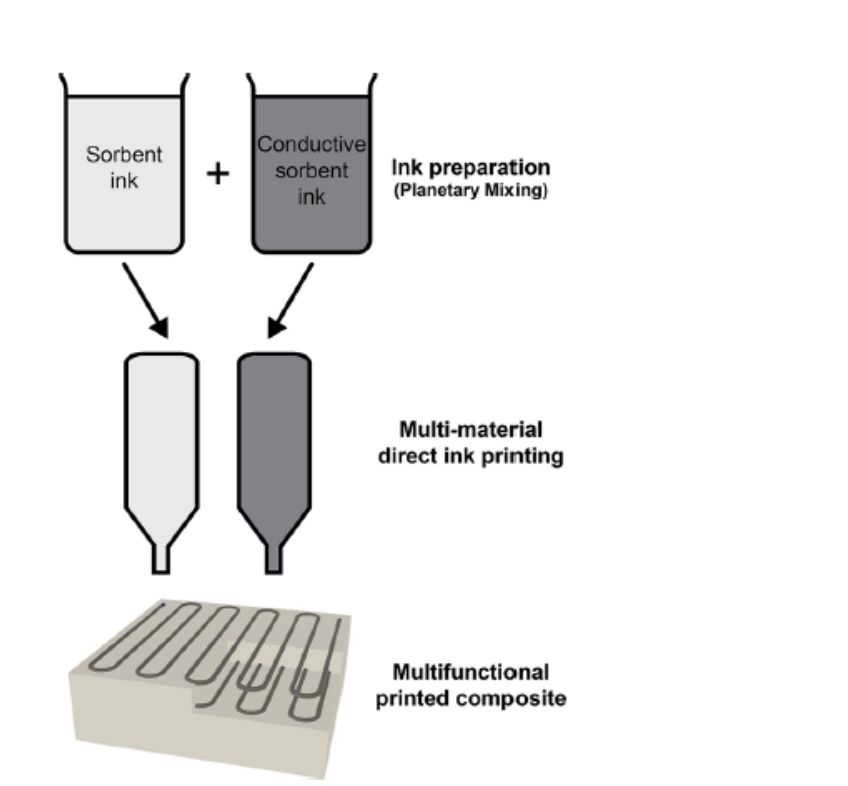


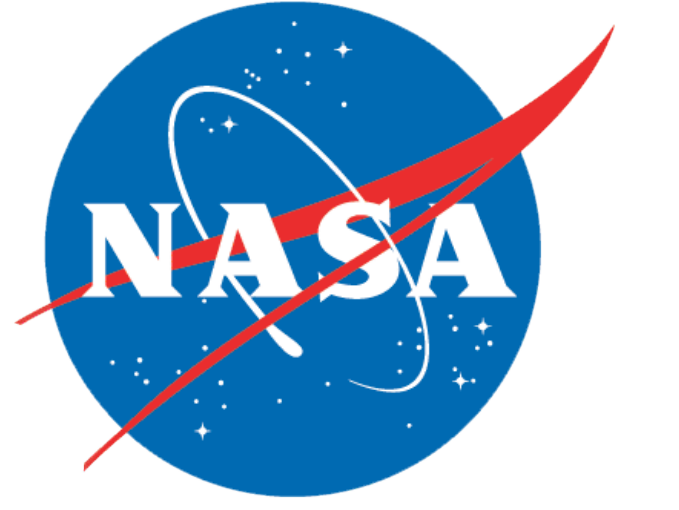
Figure 6: Process for multimaterial printing to form a multisorb from conductive and nonconductive CO₂ sorbent inks. ³

References

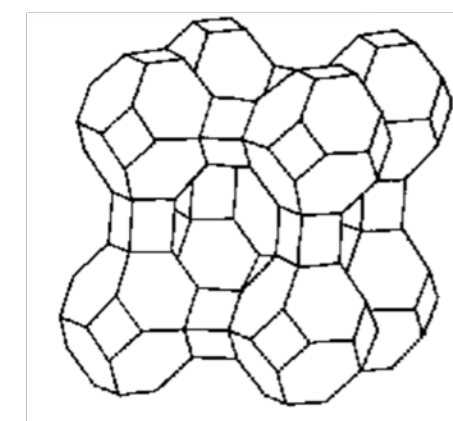
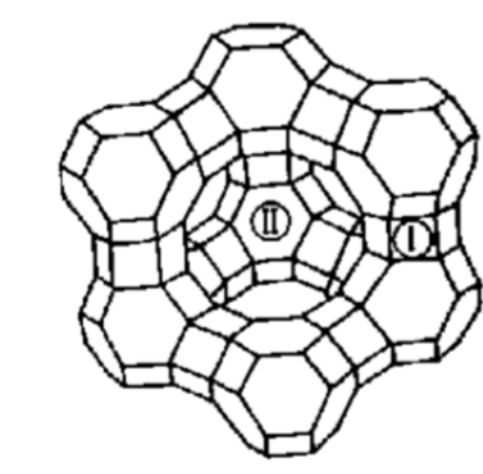
- Thakkar, H.; Eastman, S.; Hajari, A.; Rowanghi, A. A.; Knox, J. C.; Rezaei, F. 3D-Printed Zeolite Monoliths for CO₂ Removal from Enclosed Environments. *ACS Applied Materials & Interfaces* **2016**, *8* (41), 27753–27761.
- Thakkar, H.; Lawson, S.; Rowanghi, A.A. and Rezaei, F., 2018. Development of 3D-printed polymer-zeolite composite monoliths for gas separation. *Chemical Engineering Journal*, *348*, pp.109-116.
- Thompson, J.F., Bellerjeau, C., Marinick, G., Osio-Norgaard, J., Evans, A., Carry, P., Street, R.A., Petit, C. and Whiting, G.L., 2019. Intrinsic Thermal Desorption in a 3D Printed Multifunctional Composite CO₂ Sorbent with Embedded Heating Capability. *ACS Applied Materials & Interfaces*, *11*(46), pp.43337-43343.
- Richardson, J. T., Avila G., Yau A., Castellanos J., Jan D., *Multifunctional Sorbent (MultiSORB) HRHR FY19 Year End Review*. Nov. 2019. PowerPoint Presentation
- https://en.wikipedia.org/wiki/Methyl_cellulose#/media/File:Methyl_cellulose.png
- https://en.wikipedia.org/wiki/Polyvinyl_alcohol#/media/File:Polyvinyl_Alcohol_Structural_Formula_V1.svg

Acknowledgements

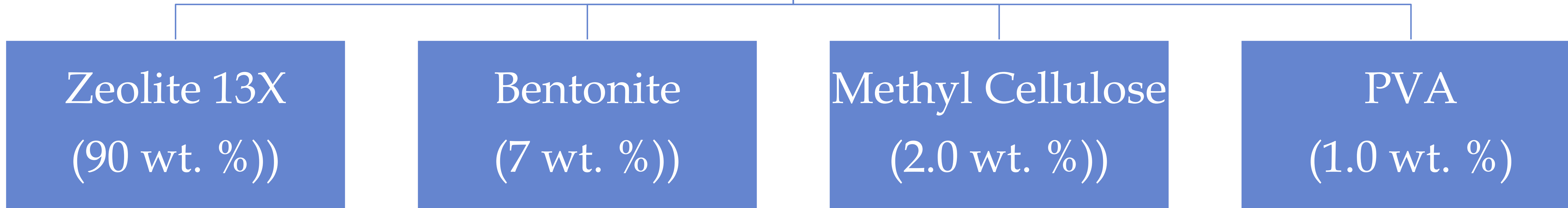
We wanted to especially thank our mentor, Tra-My Justine Richardson, and teammates for their tremendous support and guidance. Additionally, we would like to thank Dr. Eduardo Nicolau for his help. Also, we want to thank our assigned internship coordinator, Haley Feck, for her counseling and direction through the summer. Finally, we want to recognize the NASA MIRO-Puerto Rico Space Partnership for Research, Education and Training (PR-SPRINT) program under grant # 80NSSC19M0236, for funding and supporting this exceptional NASA internship.

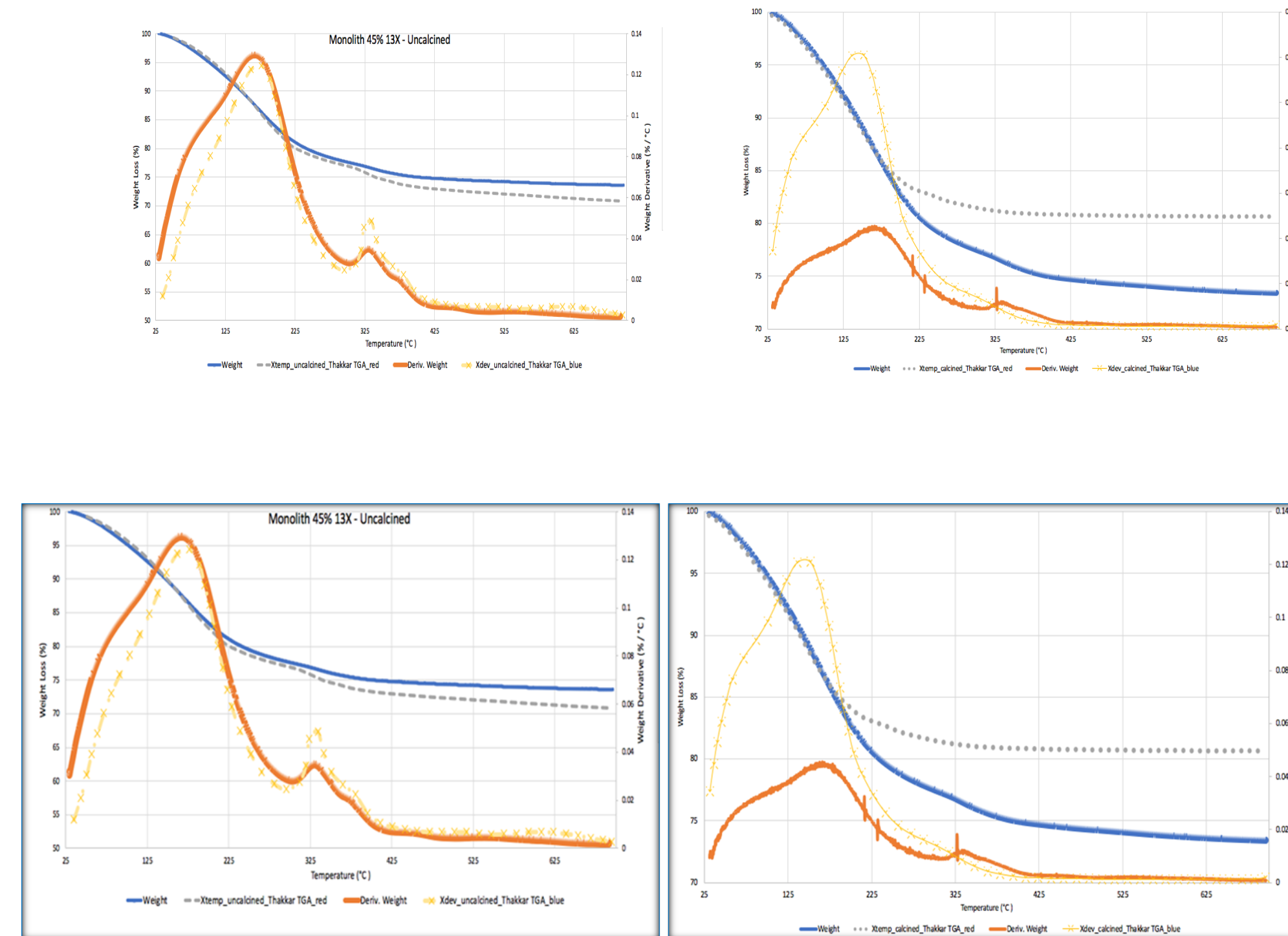
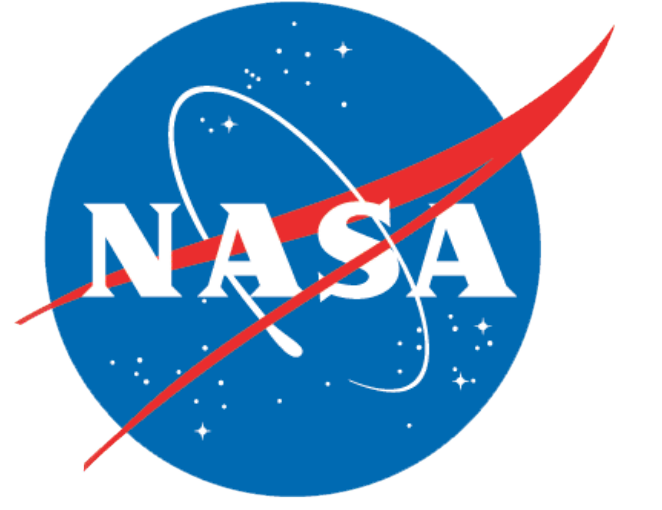


Pictures



R4 Zeolite 13X
Paste





Percent Solids	BET Surface Area	Pore Volume	Pore Diameter(Å)	CO2 Adsorption Capacity
40%	134.3242	-	6.914	0.3039
45%	220.6183	0.005654	6.356	0.7745
47.5%	250.5108	0.008566	6.396	0.8827
50%	153.8685	0.006719	6.518	0.5631
52.5%	123.5304	-	6.592	0.3648