OPENMDAO DEVELOPMENT UPDATE

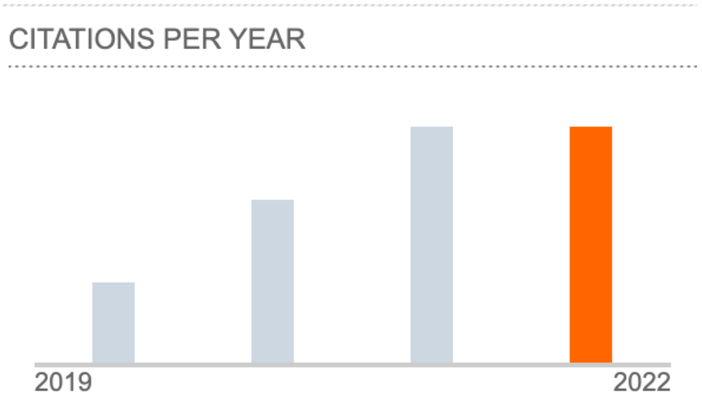
Rob Falck

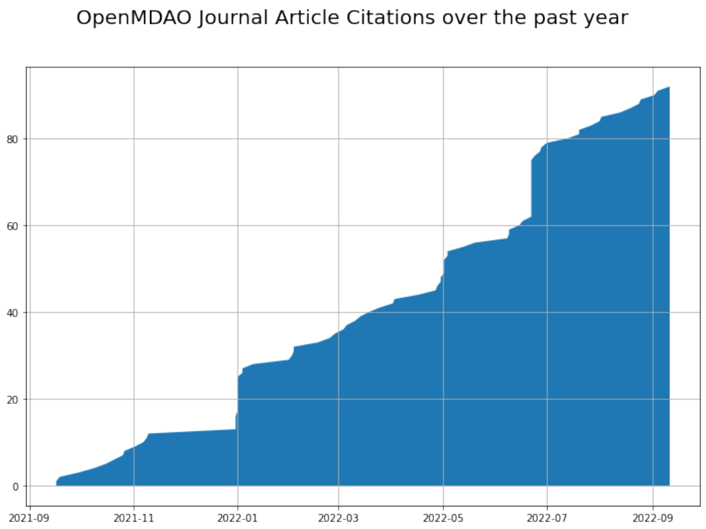
Development Team Lead

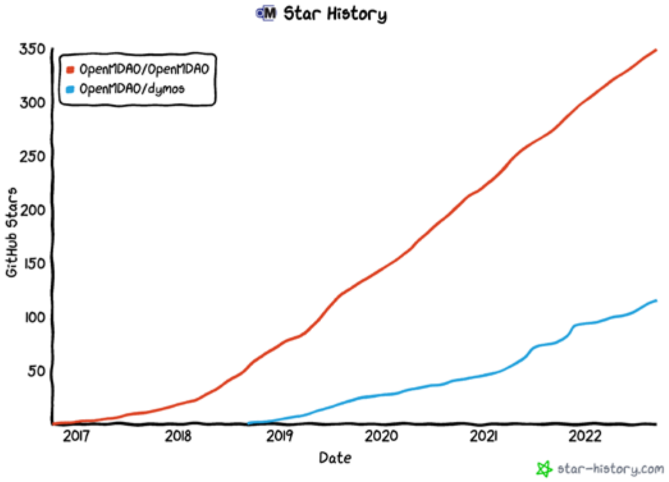
DEVELOPMENT TEAM

- John Jasa
- Tad Kollar
- Ken Moore
- Bret Naylor
- Kaushik Ponnapalli
- Steve Ryan
- Herb Schilling

ASSESSING OUR IMPACT







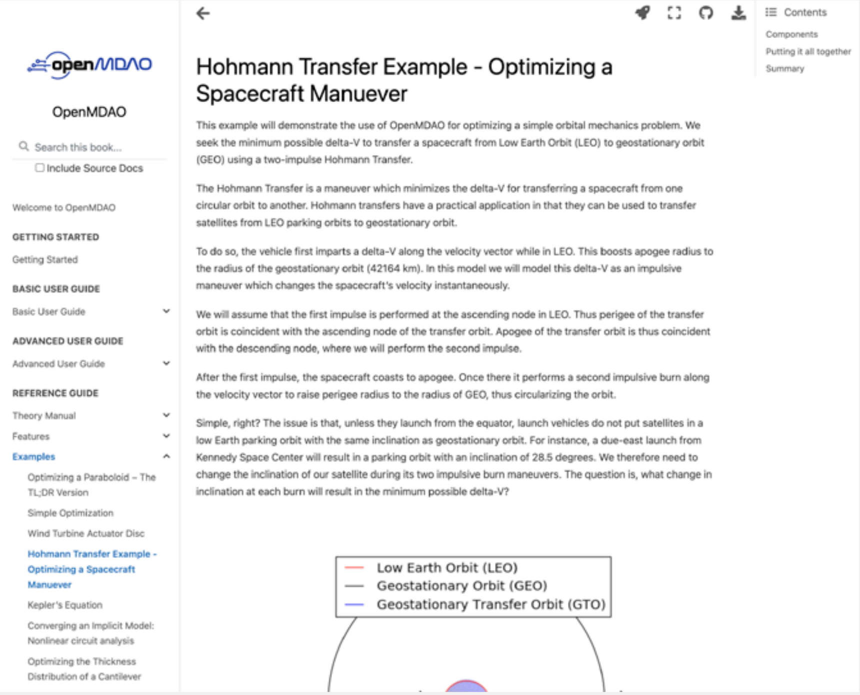
FOCUS OF RECENT DEVELOPMENT

LOWERING BARRIERS TO ENTRY

- Executable Documentation
- Practical MDO with OpenMDAO notebooks and videos
- build_pyoptsparse

EXECUTABLE DOCUMENTATION

- Rewrote documentation using jupyter-book
  - Notebooks as documentation
  - Documentation can test itself as part of our CI process
  - Users can test OpenMDAO examples on Google Colab without the need to install anything locally.



Hohmann Transfer Example - Optimizing a Spacecraft Manuever

This example will demonstrate the use of OpenMDAO for optimizing a simple orbital mechanics problem. We seek the minimum possible delta-V to transfer a spacecraft from Low Earth Orbit (LEO) to geostationary orbit (GEO) using a two-impulse Hohmann Transfer.

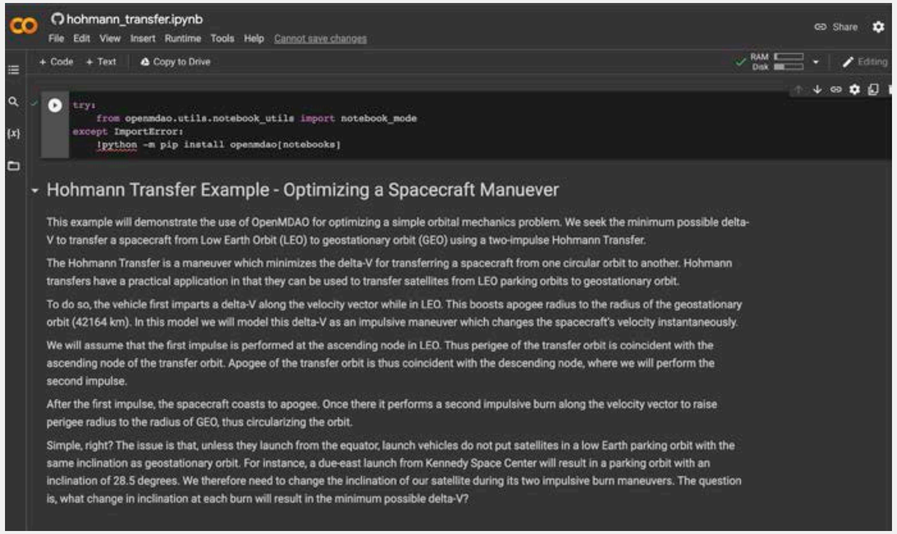
The Hohmann Transfer is a maneuver which minimizes the delta-V for transferring a spacecraft from one circular orbit to another. Hohmann transfers have a practical application in that they can be used to transfer satellites from LEO parking orbits to geostationary orbit.

To do so, the vehicle first imparts a delta-V along the velocity vector while in LEO. This boosts apogee radius to the radius of the geostationary orbit (42164 km). In this model we will model this delta-V as an impulsive maneuver which changes the spacecraft's velocity instantaneously.

We will assume that the first impulse is performed at the ascending node in LEO. Thus perigee of the transfer orbit is coincident with the ascending node of the transfer orbit. Apogee of the transfer orbit is thus coincident with the descending node, where we will perform the second impulse.

After the first impulse, the spacecraft coasts to apogee. Once there it performs a second impulsive burn along the velocity vector to raise perigee radius to the radius of GEO, thus circularizing the orbit.

Simple, right? The issue is that, unless they launch from the equator, launch vehicles do not put satellites in a low Earth parking orbit with the same inclination as geostationary orbit. For instance, a due-east launch from Kennedy Space Center will result in a parking orbit with an inclination of 28.5 degrees. We therefore need to change the inclination of our satellite during its two impulsive burn maneuvers. The question is, what change in inclination at each burn will result in the minimum possible delta-V?



```
try:
    from openmdao.utils.notebook_utils import notebook_mode
except ImportError:
    !python -m pip install openmdao[notebooks]
```

Hohmann Transfer Example - Optimizing a Spacecraft Manuever

This example will demonstrate the use of OpenMDAO for optimizing a simple orbital mechanics problem. We seek the minimum possible delta-V to transfer a spacecraft from Low Earth Orbit (LEO) to geostationary orbit (GEO) using a two-impulse Hohmann Transfer.

The Hohmann Transfer is a maneuver which minimizes the delta-V for transferring a spacecraft from one circular orbit to another. Hohmann transfers have a practical application in that they can be used to transfer satellites from LEO parking orbits to geostationary orbit.

To do so, the vehicle first imparts a delta-V along the velocity vector while in LEO. This boosts apogee radius to the radius of the geostationary orbit (42164 km). In this model we will model this delta-V as an impulsive maneuver which changes the spacecraft's velocity instantaneously.

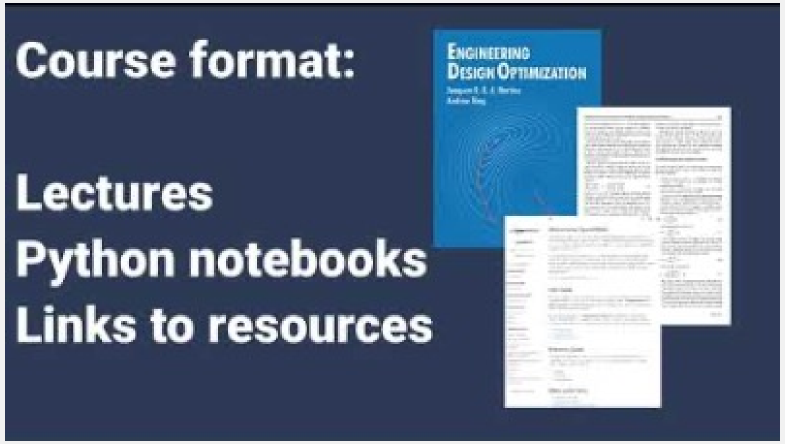
We will assume that the first impulse is performed at the ascending node in LEO. Thus perigee of the transfer orbit is coincident with the ascending node of the transfer orbit. Apogee of the transfer orbit is thus coincident with the descending node, where we will perform the second impulse.

After the first impulse, the spacecraft coasts to apogee. Once there it performs a second impulsive burn along the velocity vector to raise perigee radius to the radius of GEO, thus circularizing the orbit.

Simple, right? The issue is that, unless they launch from the equator, launch vehicles do not put satellites in a low Earth parking orbit with the same inclination as geostationary orbit. For instance, a due-east launch from Kennedy Space Center will result in a parking orbit with an inclination of 28.5 degrees. We therefore need to change the inclination of our satellite during its two impulsive burn maneuvers. The question is, what change in inclination at each burn will result in the minimum possible delta-V?

PRACTICAL MDO COURSE

- John Jasa has joined our team and has done an amazing job at producing tutorial notebooks and corresponding videos.
- These videos provide common lessons that we find ourselves teaching to users.



Course format:

Lectures

Python notebooks

Links to resources

BUILD_PYOPTSPARSE

- The MDOLab provides an amazing tool in the form of pyoptsparse.
- We were finding it difficult to build [pyoptsparse]() in a way that could provide IPOPT as an option for users.
- Easier to build pyoptsparse with support for
  - IPOPT
  - ParOPT
  - SNOPT (if source is available locally)
- https://github.com/OpenMDAO/build_pyoptsparse

OpenMDAO / **build_pyoptsparse** Public

REDUCING USER PAIN

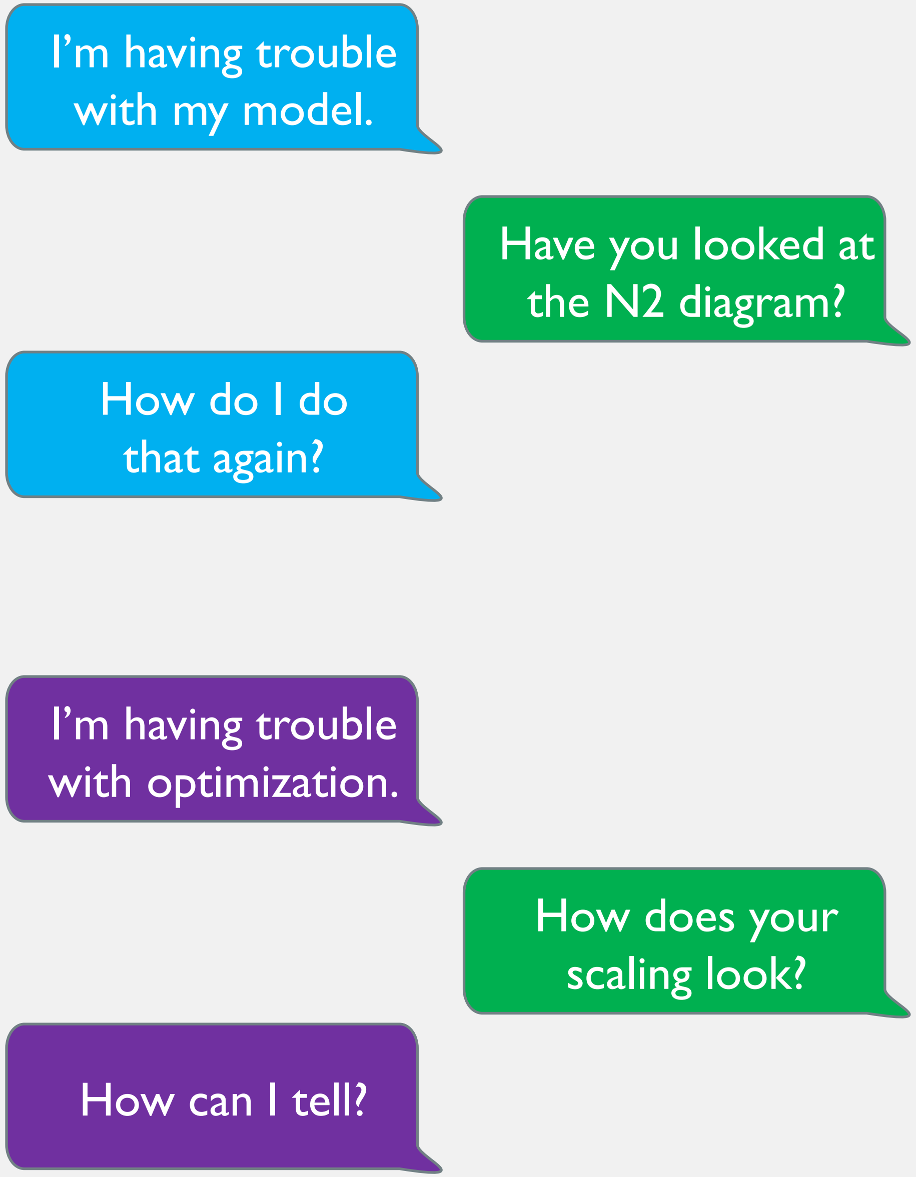
- Reports
- Visualization Tools
- Performance Improvements

REPORT GENERATION

- We provide information that users typically need.
  - N2 diagram for connectivity
  - Scaling report

1. Many users don't know that these exist, let alone how to get them.
   - Why aren't we just doing this all the time?

- Provide more rich feedback via HTML than standard output.
  - We shouldn't limit ourselves to ascii (it's not 1989 anymore)
  - Standard output often gets swamped by solver or optimizer output.
  - In the future we'll be putting more of our standard output in reports.


I'm having trouble with my model.
Have you looked at the N2 diagram?
How do I do that again?
I'm having trouble with optimization.
How does your scaling look?
How can I tell?

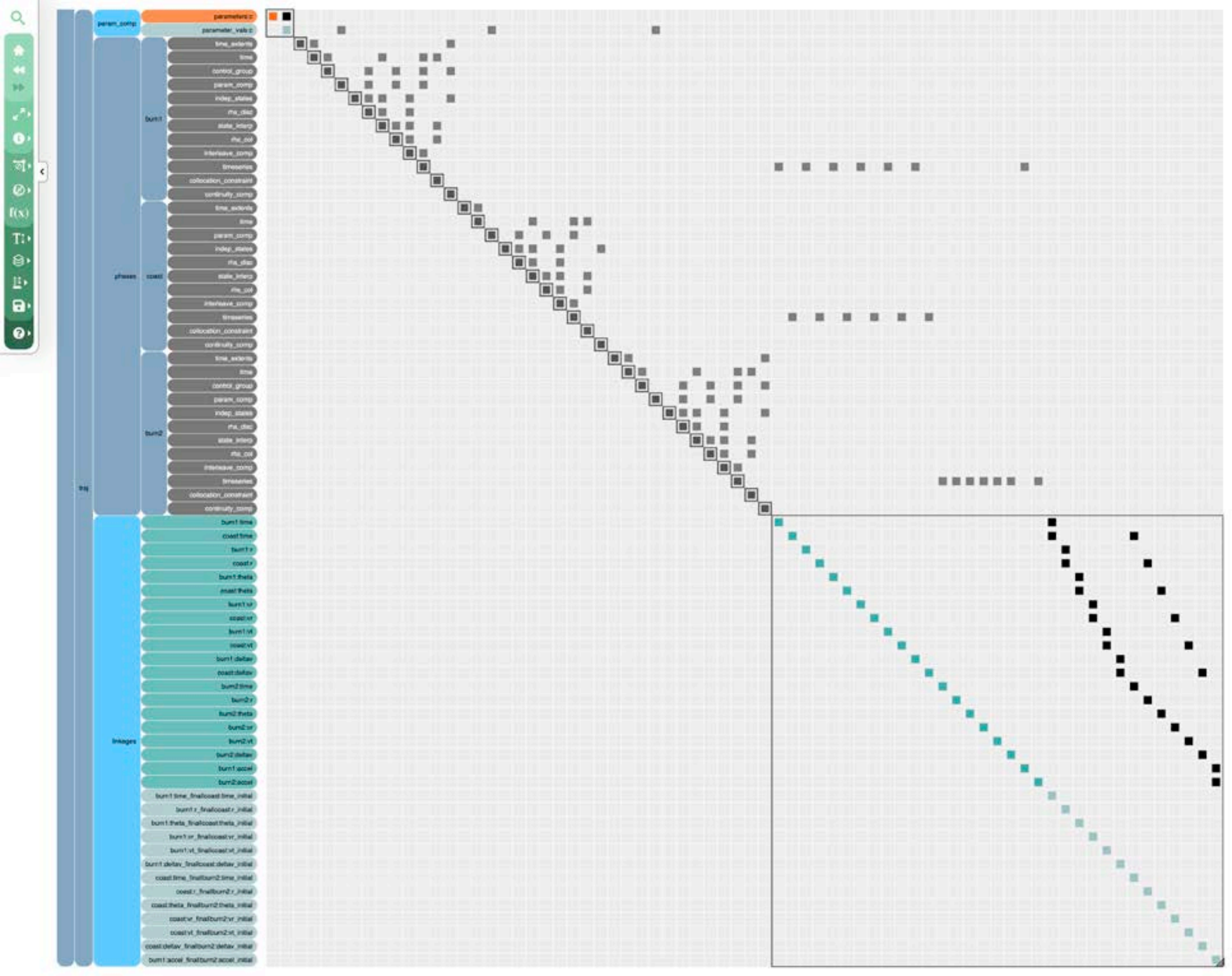
REPORT GENERATION

Name	Description	Triggered by	During	Pre/Post
scaling*	Driver scaling report	Driver	_compute_totals	post
optimizer*	Optimization summary	Problem	run_driver	post
connections	Connections viewer	Problem	final_setup	post
total coloring*	Total coloring	Driver	get_coloring	post
n2*	N2	Problem	final_setup	post
checks	Config checks	Problem	final_setup	post
summary	Model summary	Problem	final_setup	post

* report generated automatically. Others need to be specified.

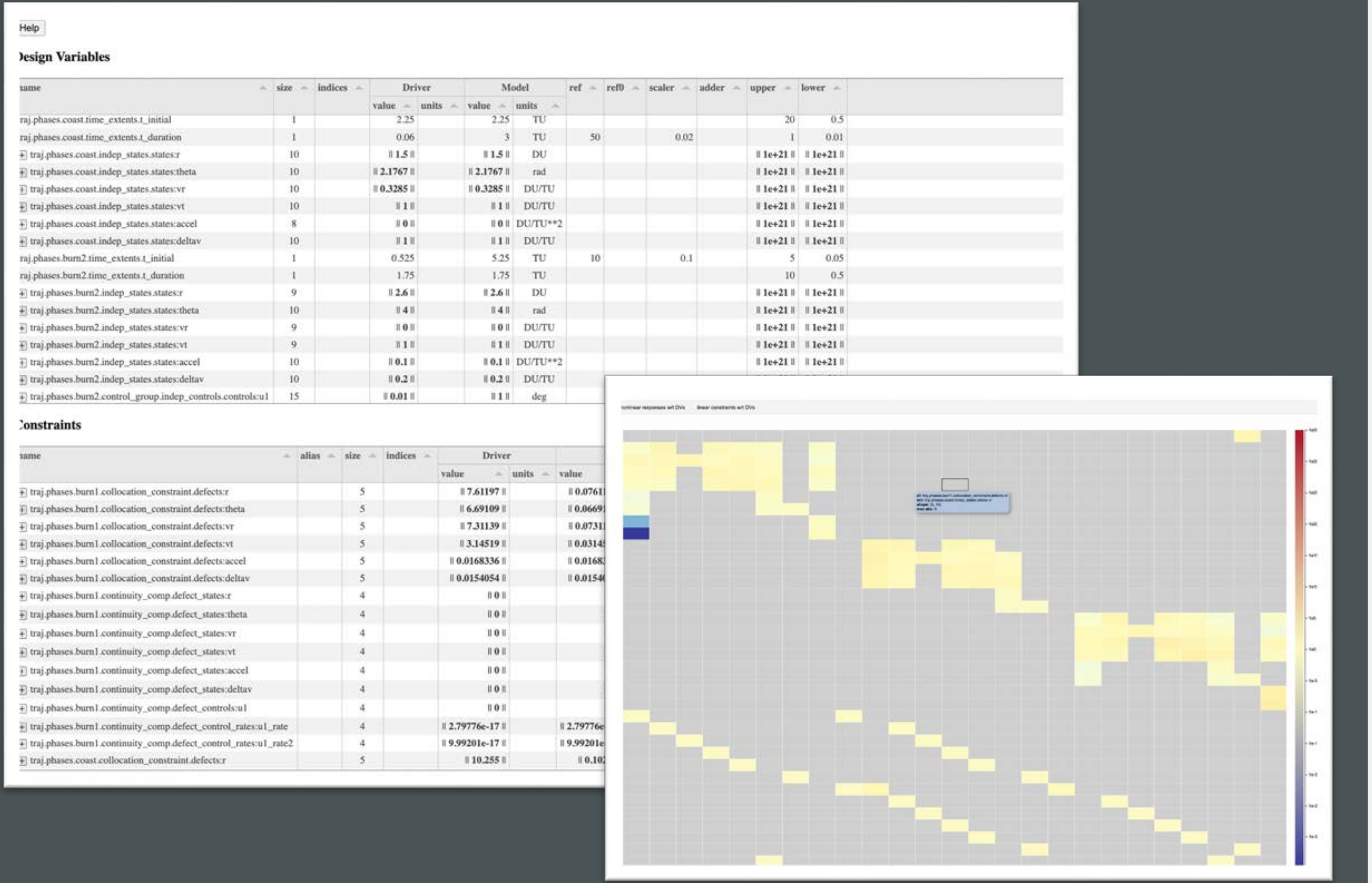
- Reports placed in the reports subdirectory by default.
- Users can design and implement their own reports.
- Users can choose individual reports, mute them all, and choose the destination directory.

THE N2 REPORT



- This probably needs no introduction.
- It’s an extremely useful tool and now it’s just generated in the process of executing a model.

THE SCALING REPORT



Help

)esign Variables

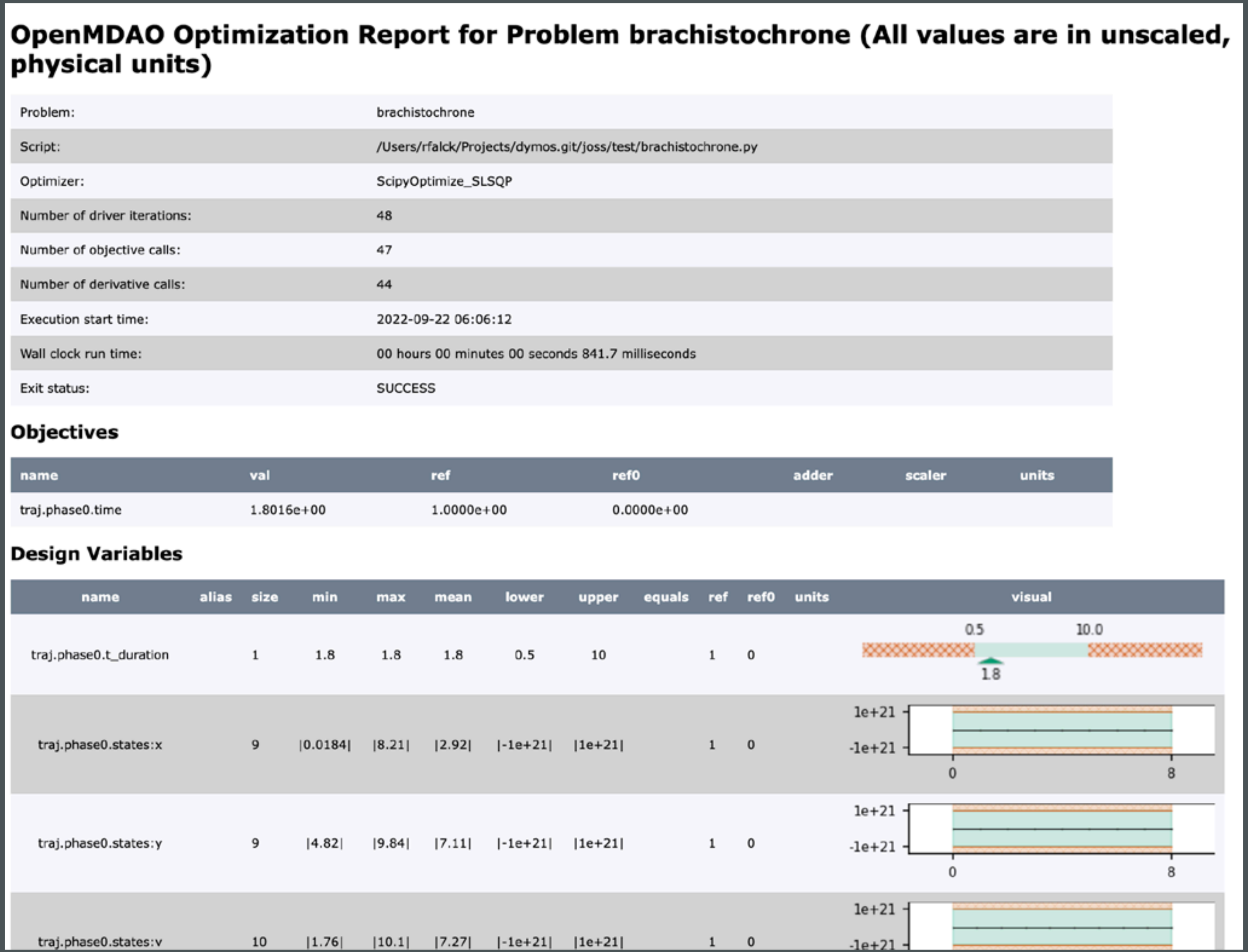
ıame	size	indices	Driver		Model		ref	ref0	scaler	adder	upper	lower
			value	units	value	units						
raj.phases.coast.time_extents.t_initial	1		2.25		2.25	TU					20	0.5
raj.phases.coast.time_extents.t_duration	1		0.06		3	TU	50		0.02		1	0.01
+ traj.phases.coast.indep_states.states:r	10		‖ 1.5 ‖		‖ 1.5 ‖	DU					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.coast.indep_states.states:theta	10		‖ 2.1767 ‖		‖ 2.1767 ‖	rad					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.coast.indep_states.states:vr	10		‖ 0.3285 ‖		‖ 0.3285 ‖	DU/TU					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.coast.indep_states.states:vt	10		‖ 1 ‖		‖ 1 ‖	DU/TU					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.coast.indep_states.states:accel	8		‖ 0 ‖		‖ 0 ‖	DU/TU**2					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.coast.indep_states.states:deltav	10		‖ 1 ‖		‖ 1 ‖	DU/TU					‖ 1e+21 ‖	‖ 1e+21 ‖
raj.phases.burn2.time_extents.t_initial	1		0.525		5.25	TU	10		0.1		5	0.05
raj.phases.burn2.time_extents.t_duration	1		1.75		1.75	TU					10	0.5
+ traj.phases.burn2.indep_states.states:r	9		‖ 2.6 ‖		‖ 2.6 ‖	DU					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.burn2.indep_states.states:theta	10		‖ 4 ‖		‖ 4 ‖	rad					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.burn2.indep_states.states:vr	9		‖ 0 ‖		‖ 0 ‖	DU/TU					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.burn2.indep_states.states:vt	9		‖ 1 ‖		‖ 1 ‖	DU/TU					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.burn2.indep_states.states:accel	10		‖ 0.1 ‖		‖ 0.1 ‖	DU/TU**2					‖ 1e+21 ‖	‖ 1e+21 ‖
+ traj.phases.burn2.indep_states.states:deltav	10		‖ 0.2 ‖		‖ 0.2 ‖	DU/TU						
+ traj.phases.burn2.control_group.indep_controls.controls:u1	15		‖ 0.01 ‖		‖ 1 ‖	deg						

:onstraints

ıame	alias	size	indices	Driver		
				value	units	value
+ traj.phases.burn1.collocation_constraint.defects:r		5		‖ 7.61197 ‖		‖ 0.0761
+ traj.phases.burn1.collocation_constraint.defects:theta		5		‖ 6.69109 ‖		‖ 0.0669
+ traj.phases.burn1.collocation_constraint.defects:vr		5		‖ 7.31139 ‖		‖ 0.0731
+ traj.phases.burn1.collocation_constraint.defects:vt		5		‖ 3.14519 ‖		‖ 0.0314
+ traj.phases.burn1.collocation_constraint.defects:accel		5		‖ 0.0168336 ‖		‖ 0.0168
+ traj.phases.burn1.collocation_constraint.defects:deltav		5		‖ 0.0154054 ‖		‖ 0.0154
+ traj.phases.burn1.continuity_comp.defect_states:r		4		‖ 0 ‖		
+ traj.phases.burn1.continuity_comp.defect_states:theta		4		‖ 0 ‖		
+ traj.phases.burn1.continuity_comp.defect_states:vr		4		‖ 0 ‖		
+ traj.phases.burn1.continuity_comp.defect_states:vt		4		‖ 0 ‖		
+ traj.phases.burn1.continuity_comp.defect_states:accel		4		‖ 0 ‖		
+ traj.phases.burn1.continuity_comp.defect_states:deltav		4		‖ 0 ‖		
+ traj.phases.burn1.continuity_comp.defect_controls:u1		4		‖ 0 ‖		
+ traj.phases.burn1.continuity_comp.defect_control_rates:u1_rate		4		‖ 2.79776e-17 ‖		‖ 2.79776e
+ traj.phases.burn1.continuity_comp.defect_control_rates:u1_rate2		4		‖ 9.99201e-17 ‖		‖ 9.99201e
+ traj.phases.coast.collocation_constraint.defects:r		5		‖ 10.255 ‖		‖ 0.10

- See at a glance the impact of scaling on variables and constraints in the optimizer’s context.

THE OPTIMIZER REPORT



OpenMDAO Optimization Report for Problem brachistochrone (All values are in unscaled, physical units)

Problem:	brachistochrone
Script:	/Users/rfalck/Projects/dymos.git/joss/test/brachistochrone.py
Optimizer:	ScipyOptimize_SLSQP
Number of driver iterations:	48
Number of objective calls:	47
Number of derivative calls:	44
Execution start time:	2022-09-22 06:06:12
Wall clock run time:	00 hours 00 minutes 00 seconds 841.7 milliseconds
Exit status:	SUCCESS

Objectives

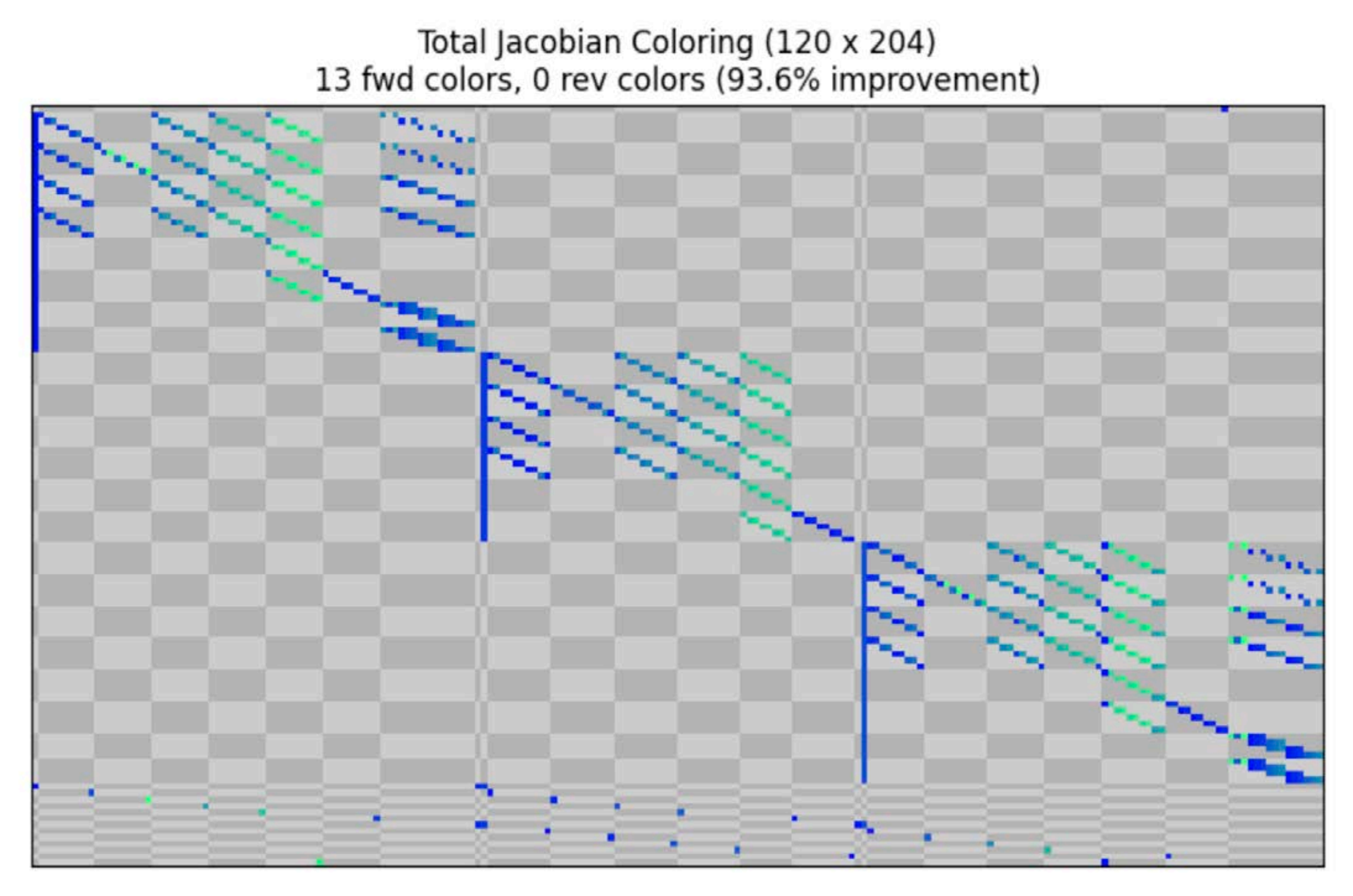
name	val	ref	ref0	adder	scaler	units
traj.phase0.time	1.8016e+00	1.0000e+00	0.0000e+00			

Design Variables

name	alias	size	min	max	mean	lower	upper	equals	ref	ref0	units	visual
traj.phase0.t_duration		1	1.8	1.8	1.8	0.5	10		1	0		
traj.phase0.states:x		9	|0.0184|	|8.21|	|2.92|	|-1e+21|	|1e+21|		1	0		
traj.phase0.states:y		9	|4.82|	|9.84|	|7.11|	|-1e+21|	|1e+21|		1	0		
traj.phase0.states:v		10	|1.76|	|10.1|	|7.27|	|-1e+21|	|1e+21|		1	0		

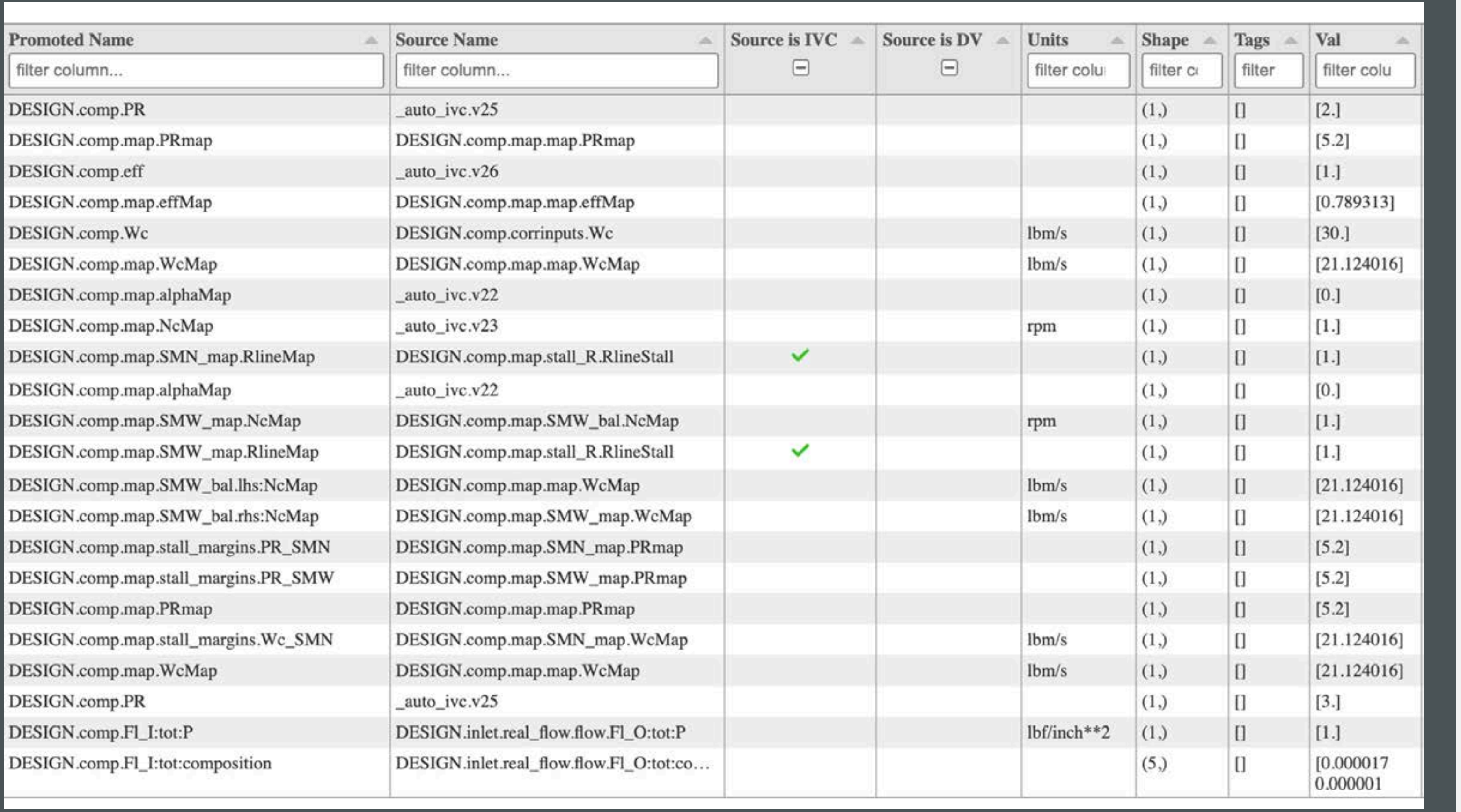
- Summary of all optimization variables, and the current optimizer settings.

THE TOTAL COLORING REPORT



- See the sparsity of your jacobian and how OpenMDAO is solving the derivatives.
- Each ”color” corresponds to a linear solve necessary to compute the total derivatives.

THE INPUTS REPORT



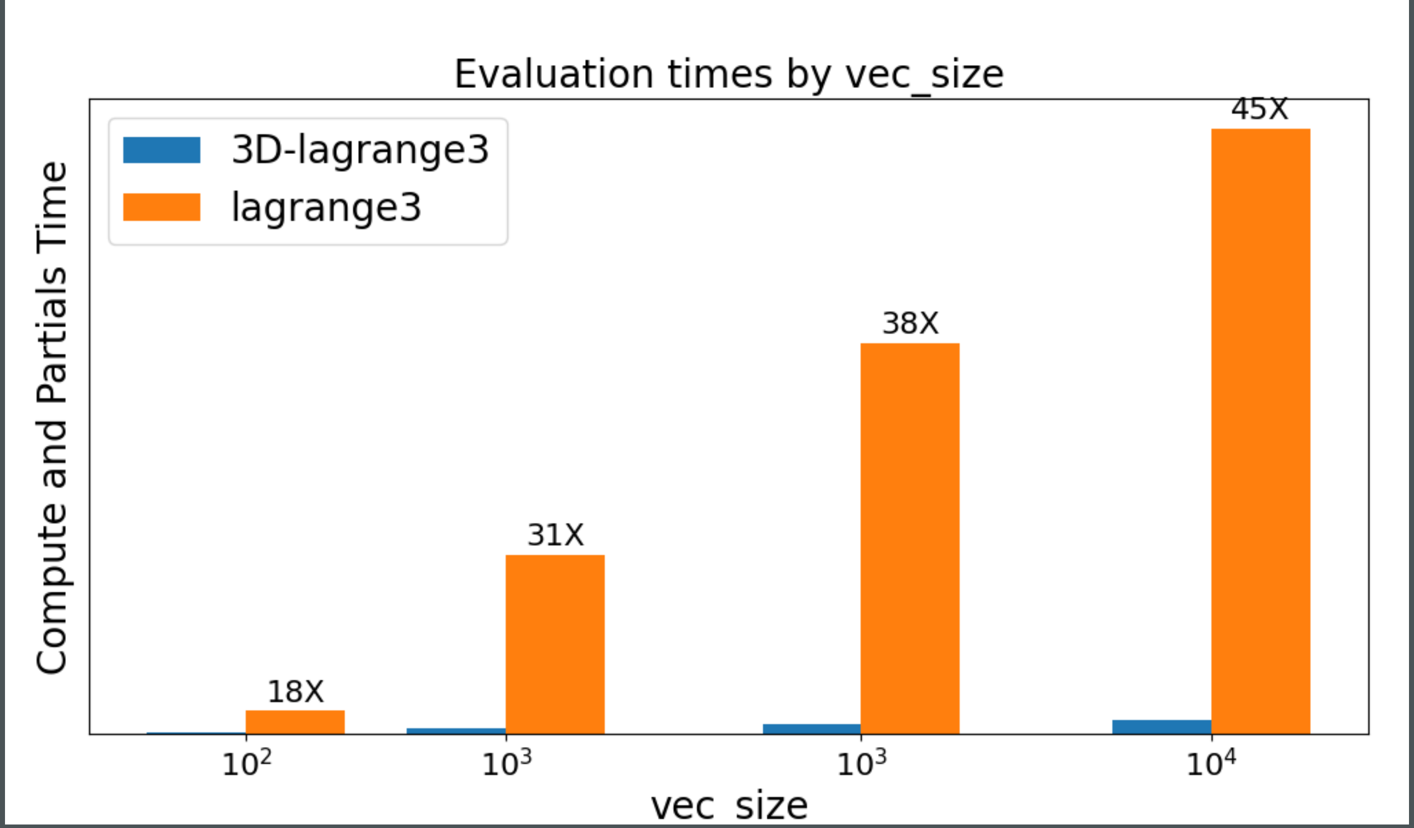
Promoted Name	Source Name	Source is IVC	Source is DV	Units	Shape	Tags	Val
filter column...	filter column...	⊟	⊟	filter colu	filter co	filter	filter colu
DESIGN.comp.PR	_auto_ivc.v25				(1,)	[]	[2.]
DESIGN.comp.map.PRmap	DESIGN.comp.map.map.PRmap				(1,)	[]	[5.2]
DESIGN.comp.eff	_auto_ivc.v26				(1,)	[]	[1.]
DESIGN.comp.map.effMap	DESIGN.comp.map.map.effMap				(1,)	[]	[0.789313]
DESIGN.comp.Wc	DESIGN.comp.corrinputs.Wc			lbm/s	(1,)	[]	[30.]
DESIGN.comp.map.WcMap	DESIGN.comp.map.map.WcMap			lbm/s	(1,)	[]	[21.124016]
DESIGN.comp.map.alphaMap	_auto_ivc.v22				(1,)	[]	[0.]
DESIGN.comp.map.NcMap	_auto_ivc.v23			rpm	(1,)	[]	[1.]
DESIGN.comp.map.SMN_map.RlineMap	DESIGN.comp.map.stall_R.RlineStall	✓			(1,)	[]	[1.]
DESIGN.comp.map.alphaMap	_auto_ivc.v22				(1,)	[]	[0.]
DESIGN.comp.map.SMW_map.NcMap	DESIGN.comp.map.SMW_bal.NcMap			rpm	(1,)	[]	[1.]
DESIGN.comp.map.SMW_map.RlineMap	DESIGN.comp.map.stall_R.RlineStall	✓			(1,)	[]	[1.]
DESIGN.comp.map.SMW_bal.lhs:NcMap	DESIGN.comp.map.map.WcMap			lbm/s	(1,)	[]	[21.124016]
DESIGN.comp.map.SMW_bal.rhs:NcMap	DESIGN.comp.map.SMW_map.WcMap			lbm/s	(1,)	[]	[21.124016]
DESIGN.comp.map.stall_margins.PR_SMN	DESIGN.comp.map.SMN_map.PRmap				(1,)	[]	[5.2]
DESIGN.comp.map.stall_margins.PR_SMW	DESIGN.comp.map.SMW_map.PRmap				(1,)	[]	[5.2]
DESIGN.comp.map.PRmap	DESIGN.comp.map.map.PRmap				(1,)	[]	[5.2]
DESIGN.comp.map.stall_margins.Wc_SMN	DESIGN.comp.map.SMN_map.WcMap			lbm/s	(1,)	[]	[21.124016]
DESIGN.comp.map.WcMap	DESIGN.comp.map.map.WcMap			lbm/s	(1,)	[]	[21.124016]
DESIGN.comp.PR	_auto_ivc.v25				(1,)	[]	[3.]
DESIGN.comp.Fl_I:tot:P	DESIGN.inlet.real_flow.flow.Fl_O:tot:P			lbf/inch**2	(1,)	[]	[1.]
DESIGN.comp.Fl_I:tot:composition	DESIGN.inlet.real_flow.flow.Fl_O:tot:co…				(5,)	[]	[0.000017 0.000001

- Similar to list_inputs in HTML format.
- Sortable, filterable columns to answer questions like
  - *What inputs does the user need to be providing to this model?*
  - *Are all of these inputs connected to the same IVC?*

PERFORMANCE IMPROVEMENTS

- Vectorized, fixed-dimension interpolants
- Coloring improvements
- Efficiency approvements to apply_linear calls under LinearBlockGS and LinearBlockJac

FIXED-DIMENSION INTERPOLATIONS

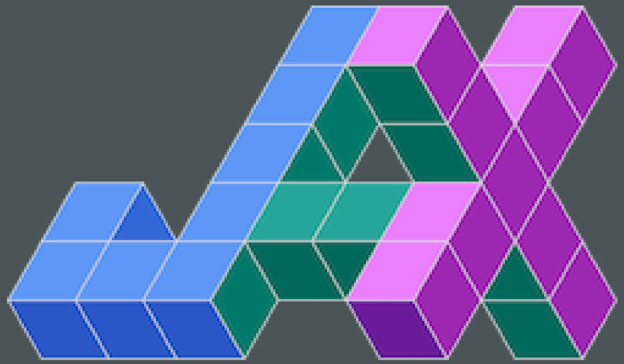


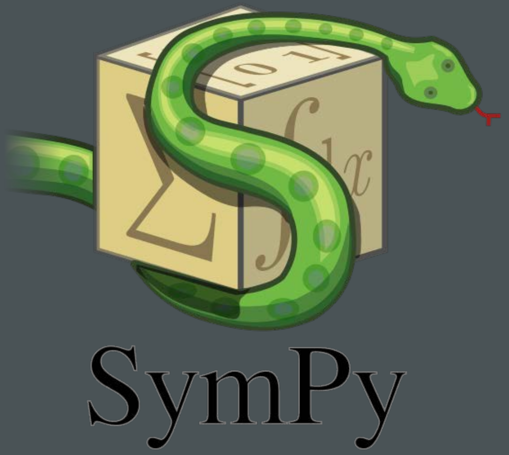
- Vectorized, fixed-dimension interpolation algorithms significantly increase speed.

- Discontinuing support for the Fortran-based MBI interpolation tool.

FUTURE DEVELOPMENT EFFORTS

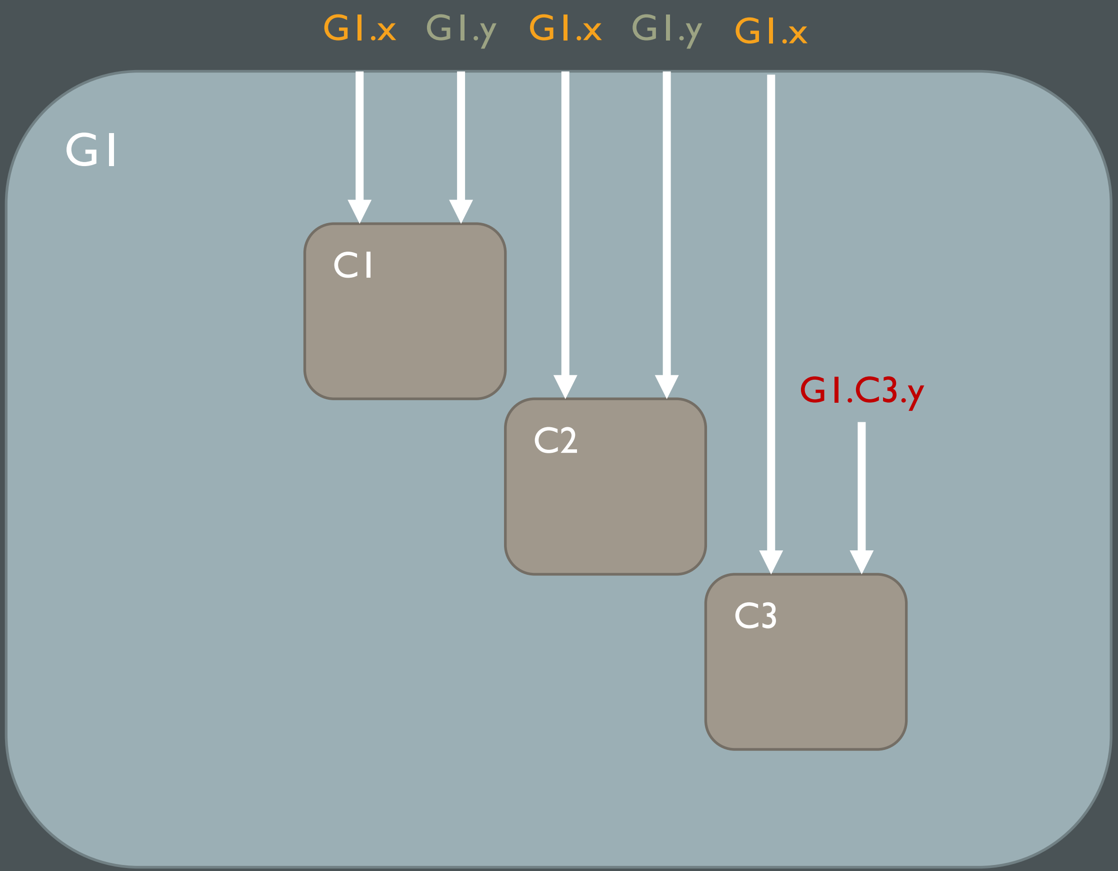
AUTOMATIC DIFFERENTIATION





- ExplicitFuncComp and ImplicitFuncComp can use jax.
- Sympy source-code transformation
- Other possible paths?

MANAGING COMPLEXITY
DEALING WITH CONNECTIONS



- OpenMDAO enables large, complex models.
- Dealing with connections increases risk of user error.
  - Forgotten connections, automatically assigned to AutoIVC unexpectedly.
  - Inputs intended to be promoted to the same variable but forgotten.
- Solutions?
  - Better feedback – The inputs report.
  - Some method of bundling connections?
    - Something like a Simulink bus?
  - Development of best practices?

COMMON INTERFACE

jupyter

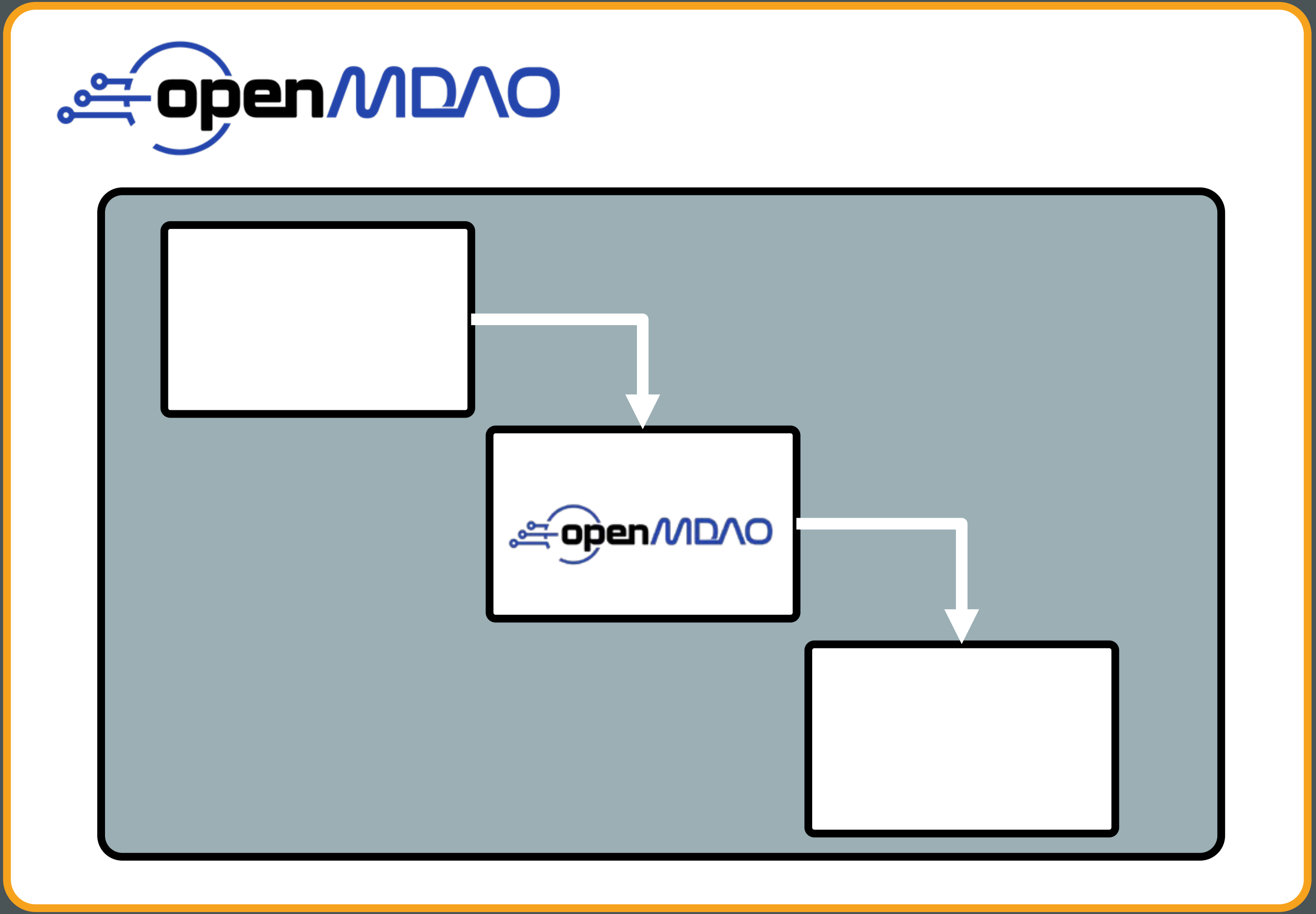
.txt

.py

.json

- Several teams seem to be working on the best practice for providing input for a problem that hides code.
- Let analysts and system engineers change select inputs without dealing with code or diving into many files.
  - Notebooks are one way
  - Should we support a canonical way or let users handle it?

SUBPROBLEM INTERFACE



- OpenMDAO Problem where run_model and compute_totals are used within the compute and compute_partials methods of a component.

- Can improve performance by "hiding" inputs and outputs that are irrelevant to the outside problem.

- There are some challenges with rolling-your-own solution that might make a canonical OpenMDAO approach preferable.

2ND DERIVATIVES (HESSIANS)

- Leveraging second derivatives can dramatically improve convergence.
- Both IPOPT and newer versions of SNOPT can utilize Hessians.
- Requires extension to MAUD upon which OpenMDAO is based.
- Reliable, efficient AD is a prerequisite.

QUESTIONS?