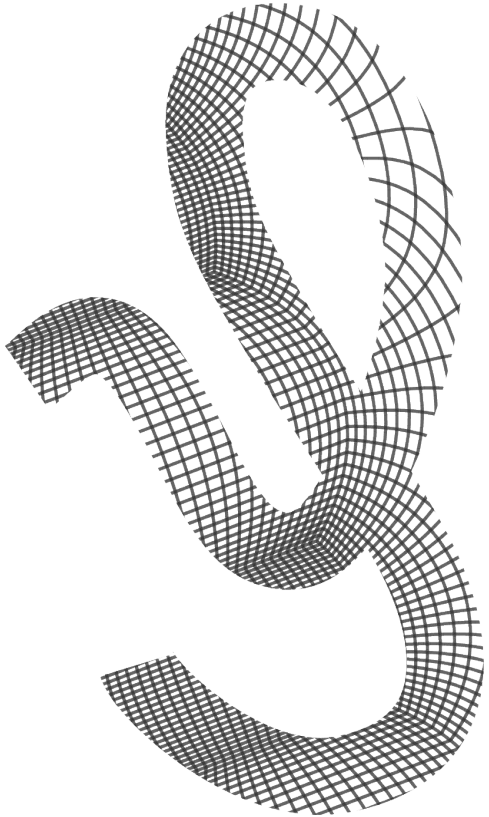


The pysces dynamical cores: CAM-SE and HOMME in JAX

Owen Hughes, (University of Michigan)

Three scientific problems that motivate this work:

Model tuning

$$\min_{\mathrm{u}} \sum_{(x_n, x_{n+1} \in \mathrm{dataset})} \mathrm{ForecastError}(F(x_n; \mathrm{u}), x_{n+1})$$

Less constrained model params

Three scientific problems that motivate this work:

Model tuning

$$\min_{\mathrm{u}} \sum_{(x_n, x_{n+1} \in \mathrm{dataset})} \mathrm{ForecastError}(F(x_n; \mathrm{u}), x_{n+1})$$

Less constrained model params

Forecast model

Three scientific problems that motivate this work:

Model tuning

$$\min_{\mathrm{u}} \sum_{(x_n, x_{n+1} \in \mathrm{dataset})} \mathrm{ForecastError}(F(x_n; \mathrm{u}), x_{n+1})$$

What's the optimal choice for parameters that we have more freedom to change?

Less constrained model params

Three scientific problems that motivate this work:

Variational data assimilation:

minimize

x ∈ model adjustment

$$J(\delta \mathbf{x}_0) = \frac{1}{2}\delta \mathbf{x}_0^T \mathbf{B}^{-1} \delta \mathbf{x}_0 + \frac{1}{2}\sum_{i=0}^{n} (\mathbf{H}_i(\delta \mathbf{x}_i) - \mathbf{d}_i)^T \mathbf{R}_i^{-1} (\mathbf{H}_i(\delta \mathbf{x}_i) - \mathbf{d}_i) \quad (1)$$

Courtesy of: https://www.ecmwf.int/en/newsletter/175/earth-system-science/linearised-physics-heart-ecmwfs-4d-var

Three scientific problems that motivate this work:

Variational data assimilation:

minimize

x ∈ model adjustments

$$J(\delta \mathbf{x}_0) = \frac{1}{2}\delta \mathbf{x}_0^T \mathbf{B}^{-1} \delta \mathbf{x}_0 + \frac{1}{2}\sum_{i=0}^{n} (\mathbf{H}_i(\delta \mathbf{x}_i) - \mathbf{d}_i)^T \mathbf{R}_i^{-1} (\mathbf{H}_i(\delta \mathbf{x}_i) - \mathbf{d}_i) \quad (1)$$

Requires:

$$\nabla_{\delta \mathbf{x}_0} J = \mathbf{B}^{-1} \delta \mathbf{x}_0 + \frac{1}{2}\sum_{i=0}^{n} \mathbf{M}^T (t_i, t_0)\, \mathbf{H}_i^T \mathbf{R}_i^{-1} (\mathbf{H}_i(\delta \mathbf{x}_i) - \mathbf{d}_i) = 0 \quad (2)$$

Adjoint ("backwards mode") of forecast model

Courtesy of: https://www.ecmwf.int/en/newsletter/175/earth-system-science/linearised-physics-heart-ecmwfs-4d-var

Three scientific problems that motivate this work:

Variational data assimilation:

minimize

$$J(\delta \mathbf{x}_0) = \frac{1}{2}\delta \mathbf{x}_0^T \mathbf{B}^{-1} \delta \mathbf{x}_0 + \frac{1}{2}\sum_{i=0}^{n} (\mathbf{H}_i(\delta \mathbf{x}_i) - \mathbf{d}_i)^T \mathbf{R}_i^{-1} (\mathbf{H}_i(\delta \mathbf{x}_i) - \mathbf{d}_i) \quad (1)$$

x = model adjustments

What's the optimal state that reconciles prediction and observation?

Requires*:

$$\nabla_{\delta \mathbf{x}_0} J = \mathbf{B}^{-1} \delta \mathbf{x}_0 + \frac{1}{2}\sum_{i=0}^{n} \mathbf{M}^T (t_i, t_0)\, \mathbf{H}_i^T \mathbf{R}_i^{-1} (\mathbf{H}_i(\delta \mathbf{x}_i) - \mathbf{d}_i) = 0 \quad (2)$$

Adjoint ("backwards mode") of forecast model

Courtesy of: https://www.ecmwf.int/en/newsletter/175/earth-system-science/linearised-physics-heart-ecmwfs-4d-var

Three scientific problems that motivate this work:

Hybrid physics/ML models:

$$\min_{\phi} \sum_{(x_n, x_{n+1}) \in \text{dataset}} \mathcal{D}(\mathrm{F}(x_n; \phi), x_{n+1})$$

Neural network params

Discrepancy function

Hybrid physics/ml forecast operator

Atmosphere state

Three scientific problems that motivate this work:

Hybrid physics/ML models:

$$\min \sum \mathcal{D}(\mathrm{F}(x_n;\phi), x_{n+1})$$

What's the optimal neural network for adjusting a physical model to fit "ground truth"

Neural network params

Discrepancy function

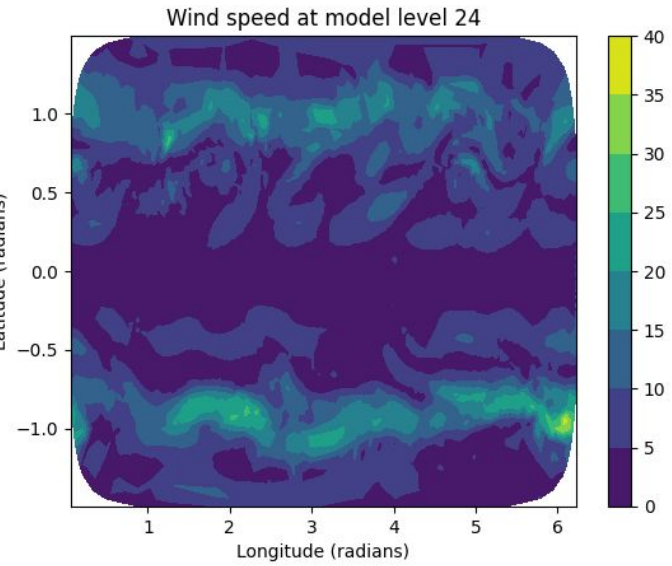
Atmosphere state

Hybrid physics/ml
forecast operator

Optimization is a really powerful mathematical way to phrase problems, and it shows up everywhere.

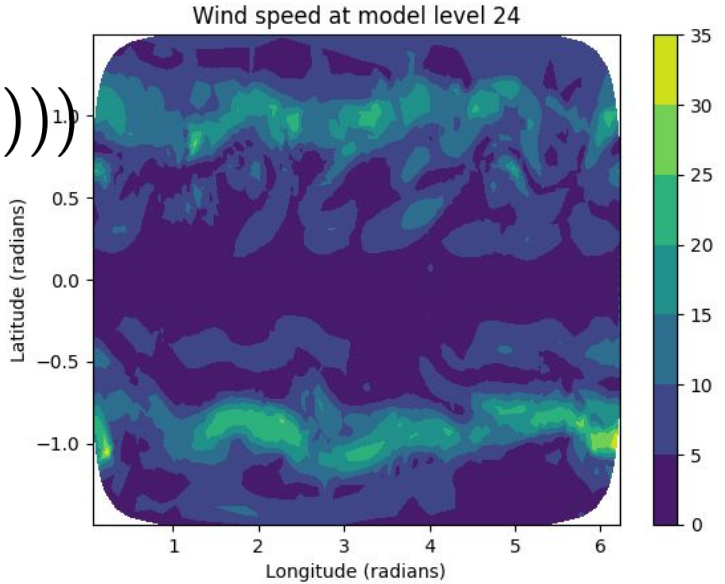
Fact:
You really should do optimization using a gradient of your cost function if it's humanly feasible to do so.

Time n



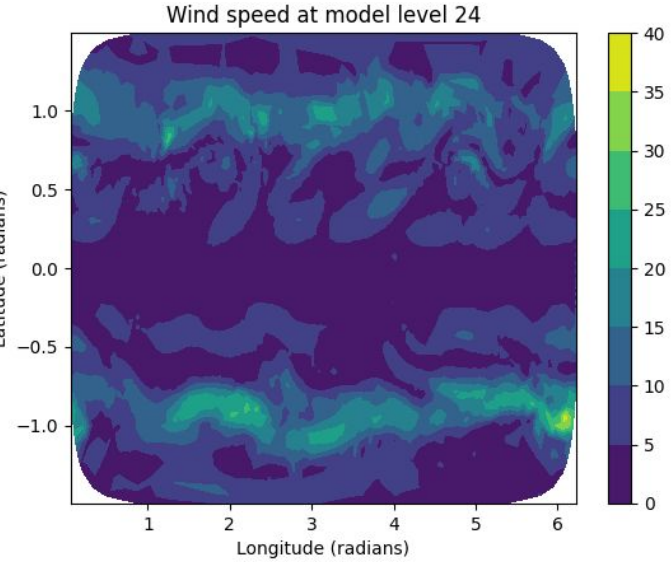
$$\tilde{x}_{n+3} = \mathrm{F}(\mathrm{F}(\mathrm{F}(x_n)))$$

Time n+3

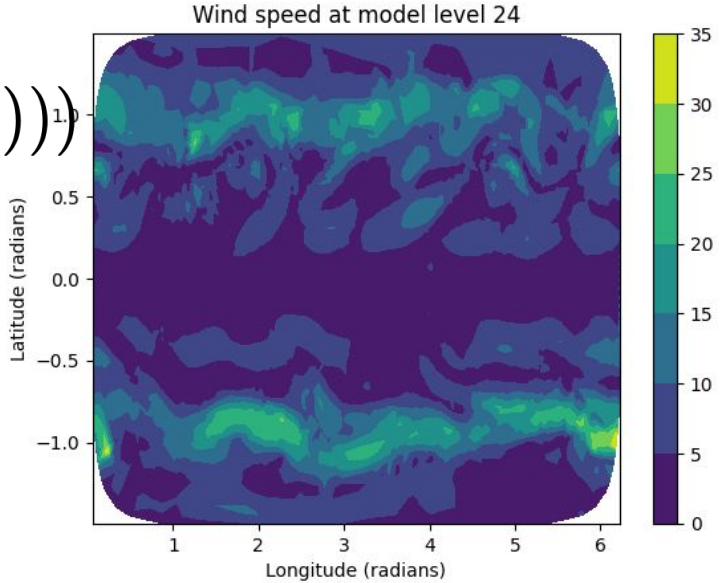


Time n

Time n+3



$$\tilde{x}_{n+3} = \mathrm{F}(\mathrm{F}(\mathrm{F}(x_n)))$$



Adjoint allows us to calculate good search direction to minimize/maximize a cost function $C(\tilde{x}_{n+k})$ by efficiently calculating $\nabla_{x_n} C(\tilde{x}_{n+k})$.

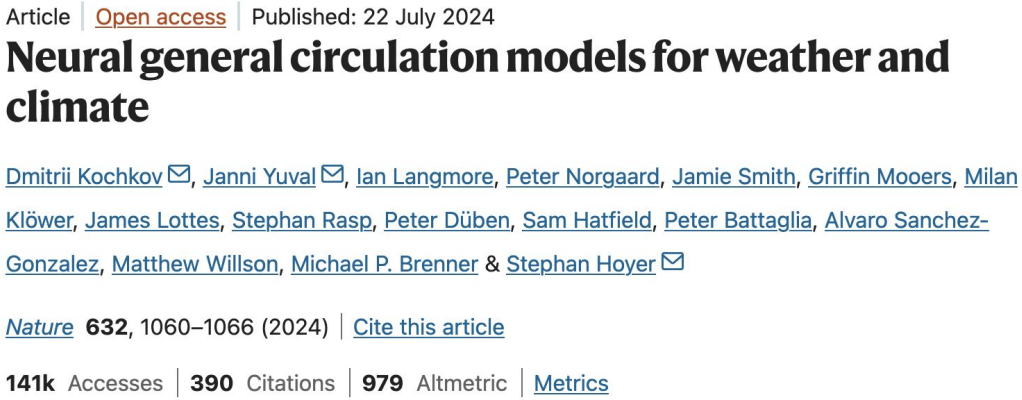
Adjoint:

- linearize your model
- take the transpose of the typical jacobian (tangent linear model)
- Use it for almost all optimization tasks.

JAX is a python library that automatically calculates these differentials for Numpy operations on CPU/GPU/TPU:

$$\nabla_{x_n} C(\tilde{x}_{n+k}) \longrightarrow$$ `jax.grad(C)(x_npk)`

NeuralGCM: JAX dynamical core + ML physics



Article | Open access | Published: 22 July 2024

Neural general circulation models for weather and climate

Dmitrii Kochkov ✉, Janni Yuval ✉, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, Sam Hatfield, Peter Battaglia, Alvaro Sanchez-Gonzalez, Matthew Willson, Michael P. Brenner & Stephan Hoyer ✉

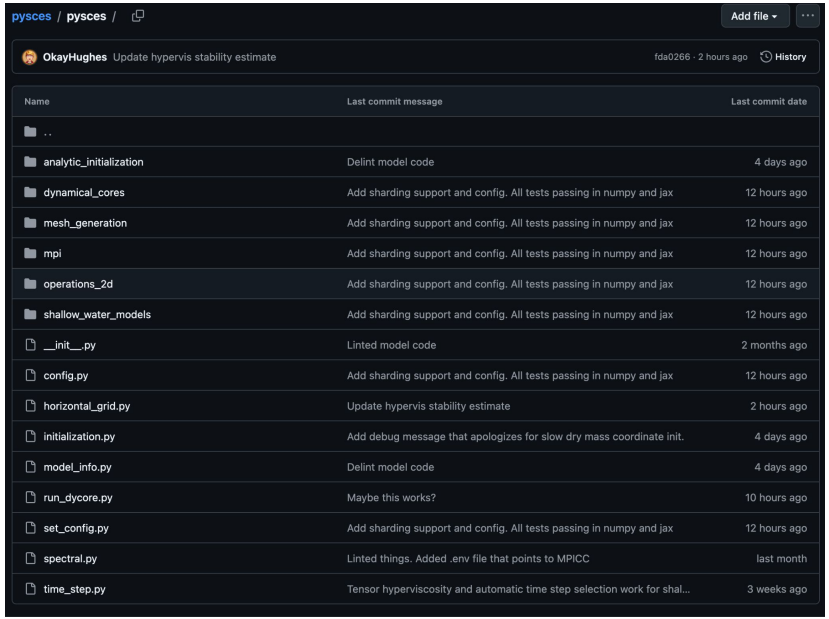
Nature **632**, 1060–1066 (2024) | Cite this article

141k Accesses | **390** Citations | **979** Altmetric | Metrics

- Google reimplemented the IFS spectral model in JAX, and coupled it to a column-wise deep neural network model.
- IFS model:
  - Quasi-uniform effective grid resolution. No grid refinement.
  - Pretty old school
  - STILL A GOOD DYNAMICAL CORE

pysces is a CPU/GPU automatically differentiable port of CAM-SE and HOMME (sister dynamical core)

Source code



https://github.com/OkayHughes/pysces

pysces is a CPU/GPU automatically differentiable port of CAM-SE and HOMME (sister dynamical core)

Source code

pysces / pysces /

Add file

OkayHughes Update hypervis stability estimate — fda0266 · 2 hours ago — History

Name	Last commit message	Last commit date
..		
analytic_initialization	Delint model code	4 days ago
dynamical_cores	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
mesh_generation	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
mpi	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
operations_2d	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
shallow_water_models	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
__init__.py	Linted model code	2 months ago
config.py	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
horizontal_grid.py	Update hypervis stability estimate	2 hours ago
initialization.py	Add debug message that apologizes for slow dry mass coordinate init.	4 days ago
model_info.py	Delint model code	4 days ago
run_dycore.py	Maybe this works?	10 hours ago
set_config.py	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
spectral.py	Linted things. Added .env file that points to MPICC	last month
time_step.py	Tensor hyperviscosity and automatic time step selection work for shal...	3 weeks ago

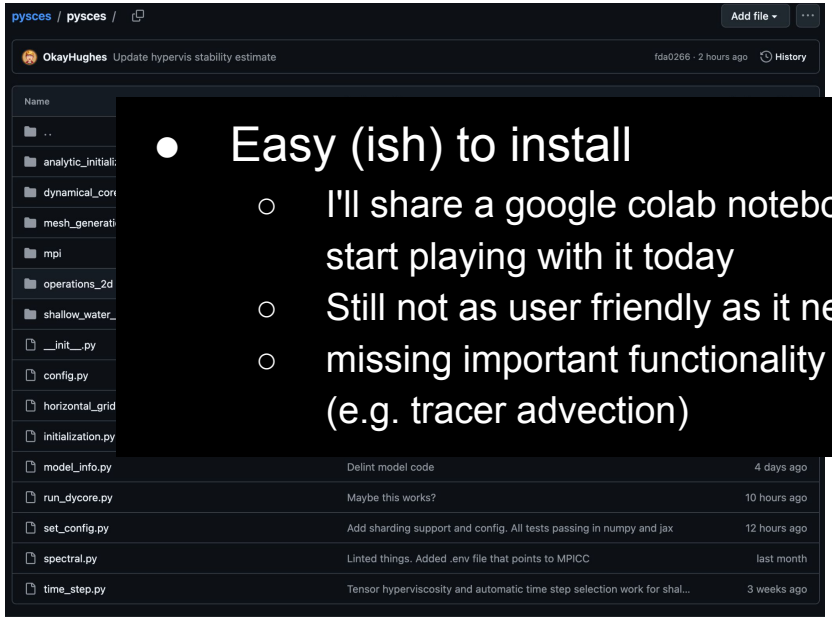
Extensive unit tests (nested, looks less exhaustive than it is)

Name	Last commit message	Last commit date
..		
fast_tests	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
mpi_tests (fast_tests)	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
reference_implementations	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
slow_tests	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
test_data	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
.env	Linted things. Added .env file that points to MPICC	last month
__init__.py	Refactors cubed_sphere.py to be callable as a module.	5 months ago
context.py	Add sharding support and config. All tests passing in numpy and jax	12 hours ago
run_tests.sh	Update readme, test script.	last month
test_spectral.py	Linted test code, and got rid of cube_definitions.py	2 months ago

https://github.com/OkayHughes/pysces

pysces is a CPU/GPU automatically differentiable port of CAM-SE and HOMME (sister dynamical core)

Source code



Extensive unit tests
(nested, looks less exhaustive than it is)

- Easy (ish) to install
  - I'll share a google colab notebook at the end of this presentation so you can start playing with it today
  - Still not as user friendly as it needs to be.
  - missing important functionality that's still not tested well enough to release (e.g. tracer advection)

https://github.com/OkayHughes/pysces

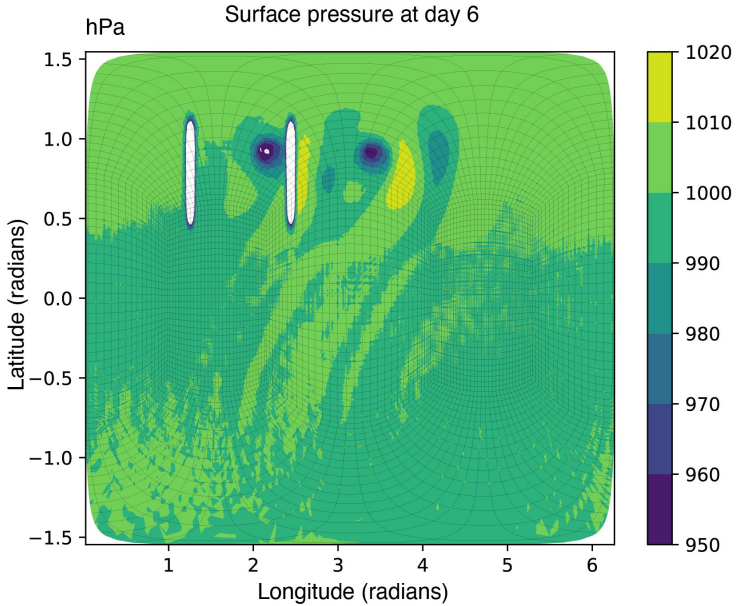
What does the model do right now?

- Support for CAM-SE dry mass coordinate (with variable Rgas thermo)
  - Temperature-based thermodynamic variable
- Support for HOMME moist mass coordinate (shallow hydrostatic)
  - Hamiltonian-based thermodynamic variable
- Support for variable-resolution grids
  - Tensor-hyperviscosity based on Guba *et al*. (2014)
- CAM-like substepping
- MPI-based parallelism
  - not differentiable
  - multiple-host capabilities untested, but should work VERY QUICKLY
- JAX sharding based parallelism
  - Differentiable
  - multiple-host capabilities are WIP
  - Exposes processor connectivity info to compiler through array dim.
    - Compiler may not presently exploit this, but this should be very possible.

Design priorities

- Model should be versatile, and still have value if JAX becomes unmaintained
  - We use JAX-numpy interoperability so the same code can be run with or without JAX installed on your system.
  - An earlier version of the code showed that PyTorch could also be used for parallelism.
    - There are a number of reasons that isn't a priority
- Model should be able to be run easily, and scale to a modest number of processors. This is not a replacement for CAM-SE, HOMME(XX)
- Untested code is unfinished code.
- People keep figuring out how to compile Numpy-style array operations.
  - If we phrase our performance-critical code that way, there will still be a way to make it fast in 5 years, and probably run it on whatever accelerator is fashionable.

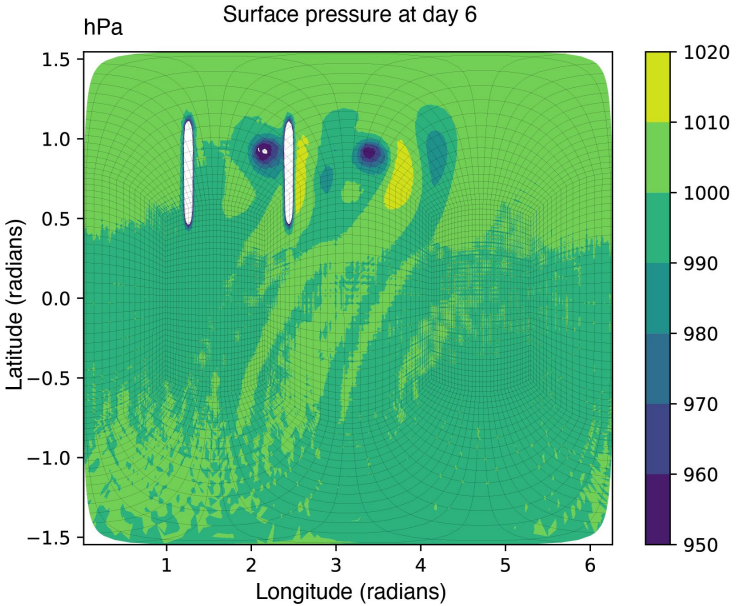
The model works:



Topographic baroclinic wave (Hughes and Jablonowski, 2023).

Grid spacing vary from ne60 to ne15.

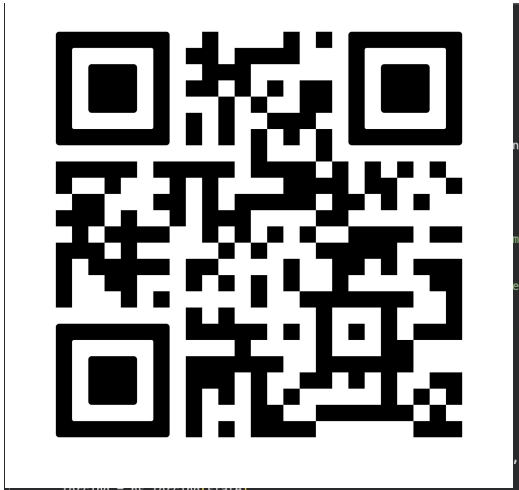
The model works:



Topographic baroclinic wave (Hughes and Jablonowski, 2023)

Grid spacing vary from ne60 to ne15.

You can try it out in a Colab (online jupyter notebook)



Conclusions:

If you're interested in any of the three problems I mentioned at the start of this talk, the model should be quite useful once I get tracer advection tested. Also, I've only partially tested that the automatic gradients do what they're advertised to do (seem to be working on one GPU).

CAM-SE runs on GPUs. You could start porting an intermediate complexity CAM configuration to python tomorrow.

I want to make this a community model. Right now, things will break. If you think the model will be useful to you, try using it and then open issues on the repo: https://github.com/OkayHughes/pysces.

PLEASE: help me explore if this can be a community model

- If you fall into one of these three groups, please consider filling out a google form
  - You want to be updated when this model hits v0.1.0, and has basically all the functionality of a dynamical core
  - You want to provide input on where you should take the model next. Should I focus on:
    - Non-hydrostatic capabilities?
    - Semi-lagrangian tracer advection?
    - Finite Volume physics grid (pg2 grids)
    - Standardized disk IO
  - You want to schedule a quick < 30 minute meeting to talk about an idea that pysces might be useful for and would like to talk through what you would need
- If you know someone who might like to use this model,

Appreciation + gratitude

- I wrote and designed this codebase basically entirely myself, with a couple of high-level (but rather life saving) suggestions from Dhamma Kimpara (NSF NCAR).
- That said, writing this code would have been impossible without mentorship from Drs. Oksana Guba and Mark Taylor (SNL), the masterminds behind the adiabatic dynamics side of HOMME
- Dr. Peter Lauritzen (NSF NCAR) helped me quickly get up to speed on what CAM modelers find valuable about CAM-SE, and how physics coupling should work.
- Professor Christiane Jablonowski, who's still helping me find my niche in the atmospheric modeling world.

Questions? Did I miss something crucial?