The Journal of Open Source Software

visualCaseGen: An SMT-based Experiment Configurator for Community Earth System Model

Alper Altuntas[1], **Isla R. Simpson**[1], **Scott D. Bachman**[1], **Manish Venumuddula**[1], **Samuel Levis**[1], **Brian Dobbins**[1], **William J. Sacks**[1], **and Gokhan Danabasoglu**[1]

1 U.S. National Science Foundation National Center for Atmospheric Research (NSF NCAR), Boulder, CO

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Mengqi Zhao

Reviewers:

- @lars2015
- @fzhao70

Submitted: 01 July 2025
Published: unpublished

Summary

`visualCaseGen` is a graphical user interface (GUI) that streamlines the setup of Community Earth System Model (CESM) experiments. While CESM supports simulations across a wide range of spatial and temporal scales and complexities (Danabasoglu et al., 2020), configuring non-standard experiments can be technically demanding and time-consuming.

To simplify this process, `visualCaseGen` offers a Jupyter-based interface that guides users through each configuration stage and automates key tasks. It features a constraint solver based on satisfiability modulo theories (SMT) (De Moura & Bjørner, 2011), which validates settings in real time by analyzing dependencies, detecting conflicts, and explaining incompatibilities. A complementary *Stage Mechanism* structures the workflow to manage variable precedence and to enforce constraints in a targeted and efficient manner.

The frontend enables users to explore standard configurations or define custom experiments, with tools for editing input files such as ocean grids, bathymetry, and land properties. By reducing setup burden and the potential for errors, `visualCaseGen` makes CESM more accessible, particularly for hierarchical modeling (Maher et al., 2019), idealized experiments (Polvani et al., 2017), and custom coupled simulations, thereby allowing researchers to focus on scientific goals rather than technical challenges.

Statement of need

Setting up custom CESM experiments requires navigating complex compatibility constraints across components, physics options, parameterizations, and model grids. These non-standard setups often involve mixing and matching settings, modifying code, editing runtime parameter files, maintaining consistency, and resolving conflicts. This process is time-consuming, error-prone, and can take weeks to produce a working configuration.

A study by Wu et al. (2021) exemplifies these challenges. The authors configured an idealized CESM experiment to explore atmosphere-ocean interactions using two aquaplanet models: one with no continents, and another with a pole-to-pole land strip to investigate effects on Hadley circulation, equatorial upwelling, and precipitation (Figure 1). Achieving this required extensive manual effort: modifying the CESM codebase, generating custom inputs, adjusting parameters, consulting domain experts, and iterating through trial and error.

`visualCaseGen` was developed to address these barriers. As an interactive GUI, it eliminates manual steps and provides an intuitive workflow for building CESM configurations efficiently.

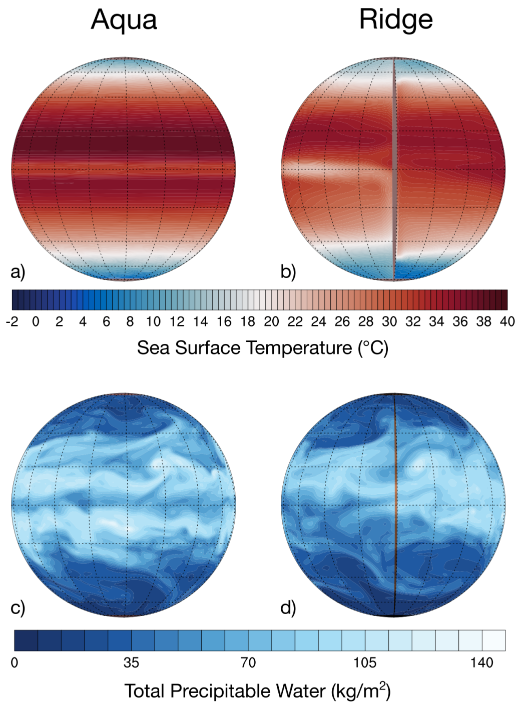

Figure 1: Sea surface temperature and precipitable water distribution from Aqua and Ridge planet simulations using CESM (Wu et al., 2021).

Constraint Solver

One of the main challenges in configuring CESM experiments is ensuring compatibility among model settings, such as components, physics, grids, and parameterizations, which often have strict interdependencies. `visualCaseGen` addresses this challenge by integrating an SMT-based constraint solver, built using the `Z3` solver (De Moura & Bjørner, 2008), which offers a robust Python API and supports multiple parameter types (integers, reals, booleans, and strings), enabling specification and enforcement of complex constraints in CESM configurations.

In `visualCaseGen`, constraints are specified as key-value pairs, where the key represents a `Z3` logical expression defining a condition, and the value is the error message displayed when the constraint is violated. These constraints enforce compatibility rules and prevent invalid model configurations. Below are three example constraints with increasing complexity, demonstrating how the SMT solver can enforce simple value bounds, conditional dependencies, and more complex multi-component rules:

```
LND_DOM_PFT >= 0.0:
    "PFT/CFT must be set to a nonnegative number",

Implies(OCN_GRID_EXTENT=="Regional", OCN_CYCLIC_X=="False"):
    "Regional ocean domain cannot be reentrant (due to an ESMF limitation.)",

Implies(And(COMP_OCN=="mom", COMP_LND=="slnd", COMP_ICE=="sice"),
        OCN_LENY<180.0):
    "If LND and ICE are stub, custom MOM6 grid must exclude poles "
    "to avoid singularities in open water.",
```

Why Use a Constraint Solver?

Configuring CESM is inherently a constraint satisfaction problem (CSP) that can become computationally complex as the number of configuration variables grows. For this reason, we use an SMT solver, which offers several advantages:

- **Detecting Hidden Conflicts:** Individual constraints may be satisfied independently, yet their combination can lead to conflicts that are nontrivial to detect manually.
- **Preventing Dead-Ends:** The solver guides users away from unsatisfiable configurations.
- **Enabling Constraint Analysis:** The solver can determine if constraints are satisfiable, identify unreachable options, and detect redundancies.
- **Scalability and Efficiency:** The solver efficiently manages large, complex constraint sets.

The Stage Mechanism

A core backend concept in `visualCaseGen` is the Stage Mechanism, which breaks the CESM configuration into consecutive stages, each grouping related variables that can be adjusted together. Stages activate dynamically based on user selections. This mechanism also enables the constraint solver to incrementally apply only relevant constraints per stage, which improves performance and responsiveness.

The stage pipeline (see Figure 2) defines the sequence of stages and variable precedence, with earlier stages having higher priority. Variables may appear in multiple stages if they are not on the same path. A critical requirement is that the pipeline forms a directed acyclic graph (DAG) to prevent cycles and inconsistencies.

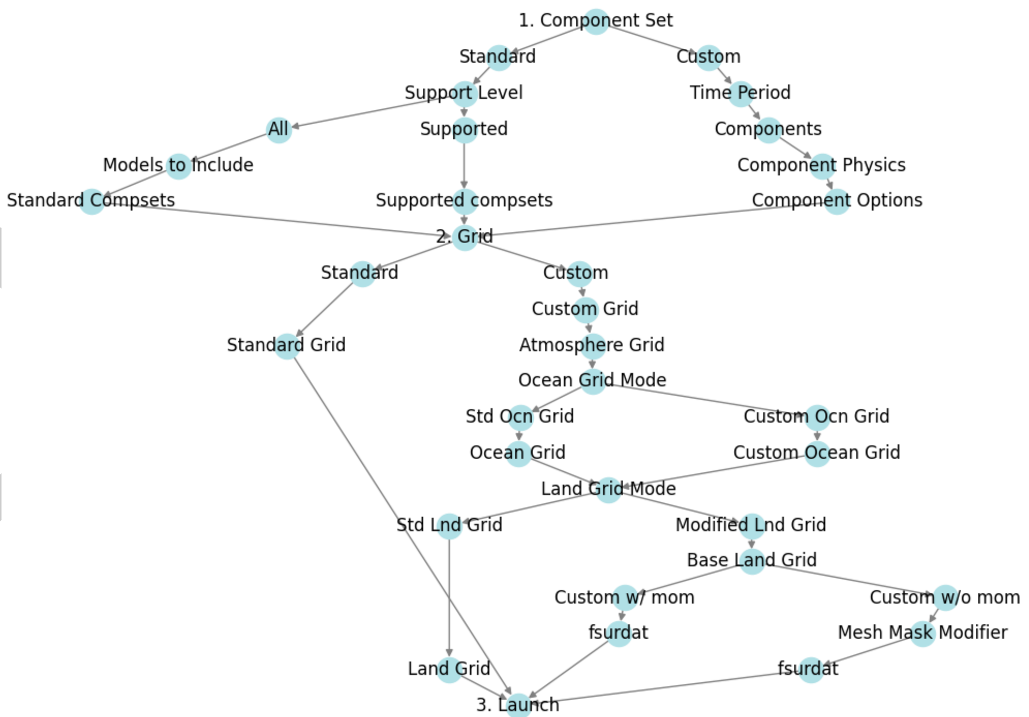

Figure 2: The stage pipeline, starting with the "Component Set" stage and ending with the "Launch" stage. The user follows a path along this pipeline based on their modeling needs and selections.

Constraint Graph and its Traversal

Using the stage pipeline and constraints, `visualCaseGen` builds a constraint graph (Figure 3) where nodes are configuration variables and directed edges represent dependencies, pointing from higher to lower-precedence variables.

When the user makes a selection, the graph is traversed breadth-first from that variable, identifying all affected variables. Traversal stops at variables unaffected by the selection. This dynamic traversal, guided by user input, stage hierarchy, and constraints, allows real-time feedback by re-evaluating constraints and adjusting options to prevent invalid configurations and ensure consistency.

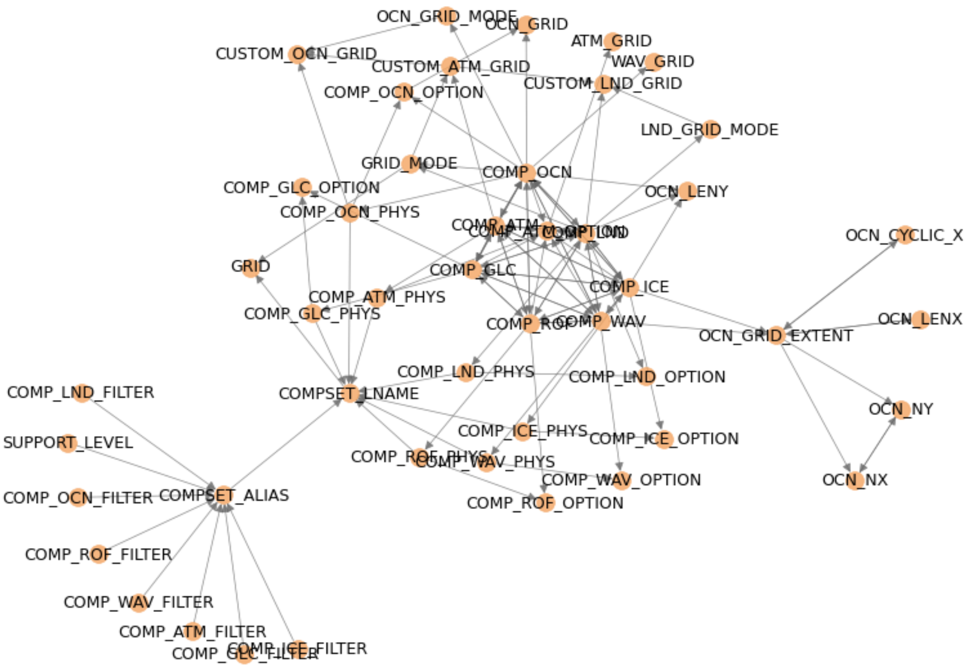

Figure 3: The visualCaseGen constraint graph.

Frontend

The `visualCaseGen` frontend provides an interactive interface for configuring CESM experiments. Built with Jupyter `ipywidgets`, it can operate on local machines, HPC clusters, and cloud environments. This portability and flexibility allows researchers to configure CESM experiments efficiently, whether prototyping lightweight simulations on personal computers or running sophisticated applications on remote supercomputing systems.

Figure 4 displays an example stage from the `visualCaseGen` GUI, where users can select the individual models to be coupled in their CESM experiment. As the user makes selections, the GUI dynamically updates available options by crossing out incompatible choices, ensuring that only valid configurations are selectable. This interactive feedback guides the user toward compatible model setups.

The Journal of Open Source Software

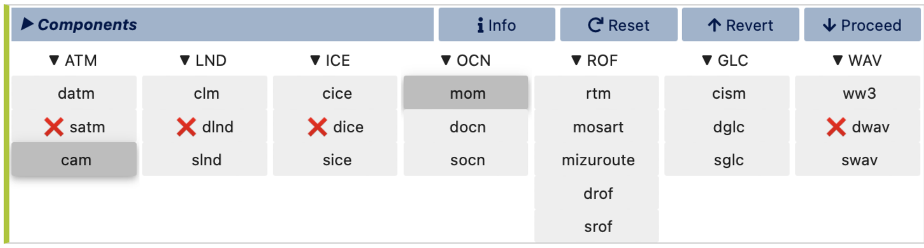

Figure 4: The "Components" stage.

At any stage, users can click on crossed-out options to see a brief explanation of why that choice is incompatible with their current selections (Figure 5). This helps users make informed adjustments.

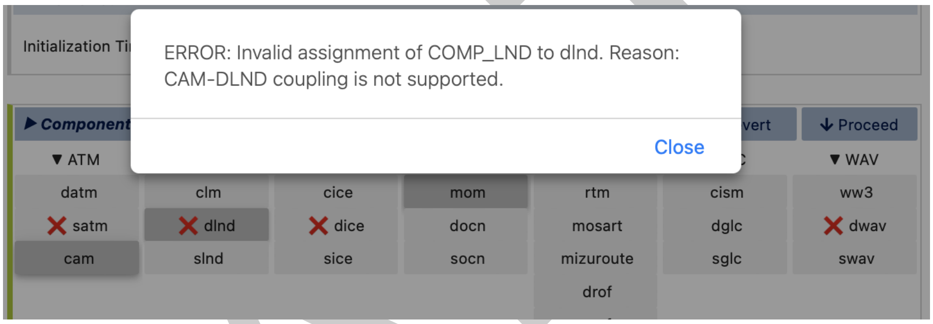

Figure 5: Interactive feedback in incompatible choices.

As another example of streamlining model customization, Figure 6 shows the TopoEditor widget that comes with `visualCaseGen`. This tool allows users to interactively modify ocean bathymetry, enhancing customizability and ease of use.

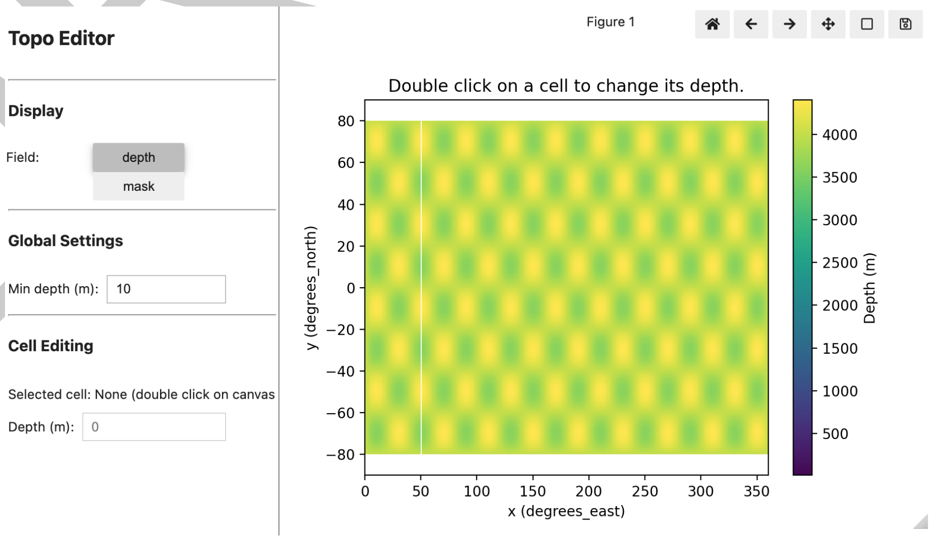

Figure 6: TopoEditor widget

Remarks

`visualCaseGen` can significantly speed up custom CESM experiment setup by automating much of the configuration process. Instead of manual edits, users define setups via an interactive GUI, mixing components, physics, and parameterizations, and generating custom input files. The SMT-based solver ensures only valid settings are chosen. While complex cases may require fine-tuning, initial working configurations can be created in hours rather than weeks. To our knowledge, `visualCaseGen` is the first tool to integrate SMT-based constraint solving into Earth system model configuration, demonstrating how formal methods can aid this process.

Acknowledgements

This work was supported by the NSF Cyberinfrastructure for Sustained Scientific Innovation (CSSI) program under award number 2004575.

The NSF National Center for Atmospheric Research (NCAR) is a major facility sponsored by the NSF under Cooperative Agreement No. 1852977.

References

Danabasoglu, G., Lamarque, J.-F., Bacmeister, J., Bailey, D., DuVivier, A., Edwards, J., Emmons, L., Fasullo, J., Garcia, R., Gettelman, A., & others. (2020). The community earth system model version 2 (CESM2). *Journal of Advances in Modeling Earth Systems*, *12*(2), e2019MS001916. https://doi.org/10.1029/2019MS001916

De Moura, L., & Bjørner, N. (2008). Z3: An efficient SMT solver. *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 337–340. https://doi.org/10.1007/978-3-540-78800-3_24

De Moura, L., & Bjørner, N. (2011). Satisfiability modulo theories: Introduction and applications. *Communications of the ACM*, *54*(9), 69–77. https://doi.org/10.1145/1995376.1995394

Maher, P., Gerber, E. P., Medeiros, B., Merlis, T. M., Sherwood, S., Sheshadri, A., Sobel, A. H., Vallis, G. K., Voigt, A., & Zurita-Gotor, P. (2019). Model hierarchies for understanding atmospheric circulation. *Reviews of Geophysics*, *57*(2), 250–280. https://doi.org/10.1029/2018RG000607

Polvani, L., Clement, A., Medeiros, B., Benedict, J., & Simpson, I. (2017). When less is more: Opening the door to simpler climate models, eos, 98. *Eos, Transactions American Geophysical Union*, *99*(3), 15–16. https://doi.org/10.1029/2017EO079417

Wu, X., Reed, K. A., Wolfe, C. L., Marques, G. M., Bachman, S. D., & Bryan, F. O. (2021). Coupled aqua and ridge planets in the community earth system model. *Journal of Advances in Modeling Earth Systems*, *13*(4), e2020MS002418. https://doi.org/10.1029/2020MS002418