Ali Ramadhan, Chris Hill (advisor), John Marshall (advisor), Alan Edelman (advisor)

Massachusetts Institute of Technology

Oceananigans.jl

Improving climate model accuracy with fast and friendly geophysical fluid dynamics on GPUs

Summary

Oceananigans.jl is a fast and friendly software package for the numerical simulation of incompressible, stratified, rotating fluid flows on CPUs and GPUs. Oceananigans.jl is fast and flexible enough for research yet simple enough for students and first-time programmers. Oceananigans.jl is being developed as part of the Climate Modeling Alliance project for the simulation of small-scale ocean physics at high-resolution that affect the evolution of Earth's climate.

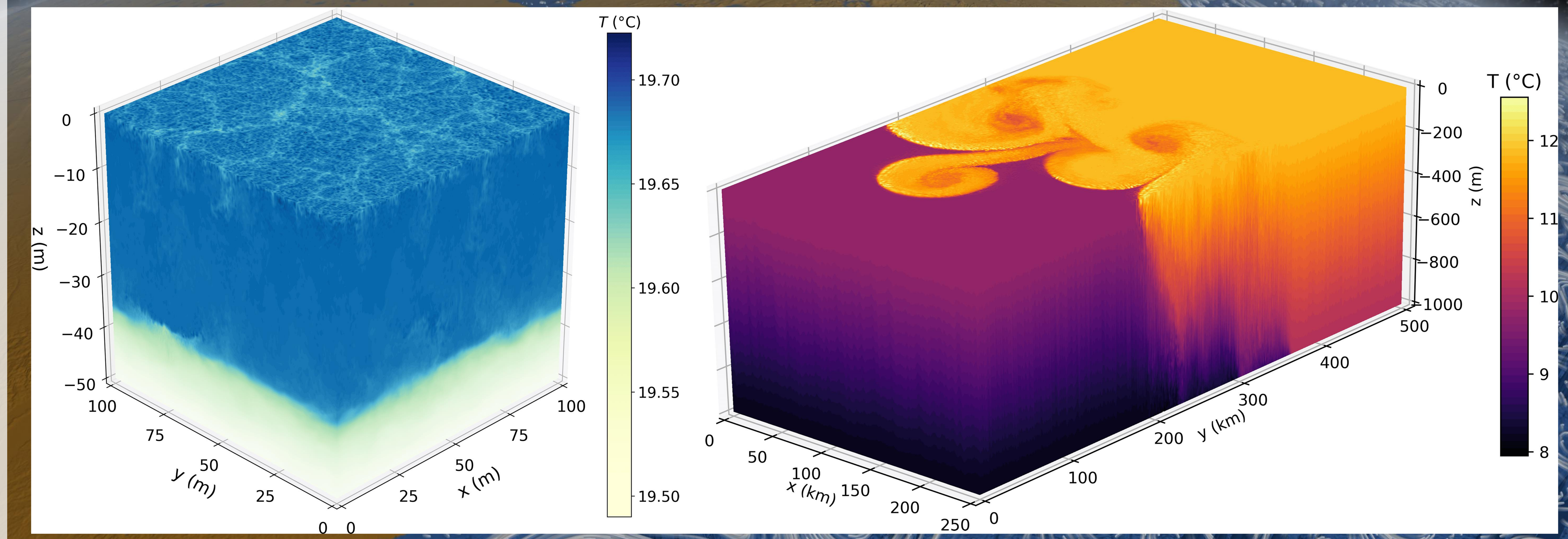
Oceananigans.jl is designed for high-resolution simulations in idealized geometries and supports direct numerical simulation, large eddy simulation, arbitrary numbers of active and passive tracers, and linear and nonlinear equations of state for seawater. Under the hood, Oceananigans.jl employs a finite volume algorithm similar to that used by the Massachusetts Institute of Technology general circulation model.

Why Julia?

Oceananigans.jl leverages the Julia programming language to implement high-level, low-cost abstractions, a friendly user interface, and a high-performance model in one language and a common code base for execution on the CPU or GPU with Julia's native GPU compiler. Because Julia is a high-level language, development is streamlined and users can flexibly specify model configurations, set up arbitrary diagnostics and output, extend the code base, and implement new features. Configuring a model with `architecture=CPU()` or `architecture=GPU()` will execute the model on the CPU or GPU. By pinning a simulation script against a specific version of Oceananigans, simulation results are reproducible up to hardware differences.

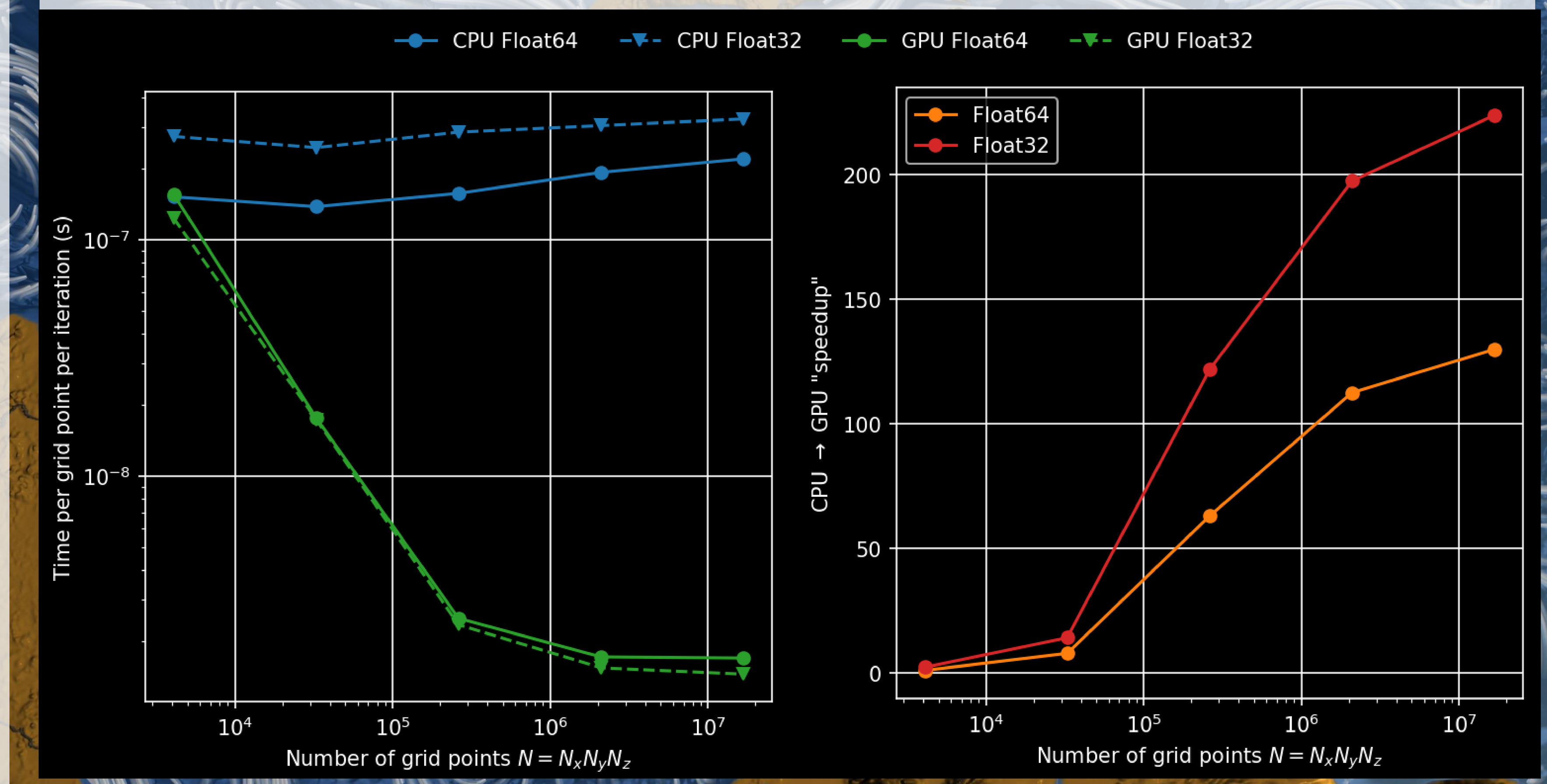
Advantages

1. Makes computational fluid dynamics, ocean modeling, and large eddy simulation more accessible. Easy to set up and use. Fortran is no longer being taught.
2. Simulations are set up via scripts. Flexible and easy to extend and share.
3. Continuous integration testing against analytic solutions and published results.
4. Makes efficient use of GPUs. An Nvidia V100 GPU can replace 150 CPU cores. Using GPUs is 300% more cost-effective than using CPUs on Google Cloud

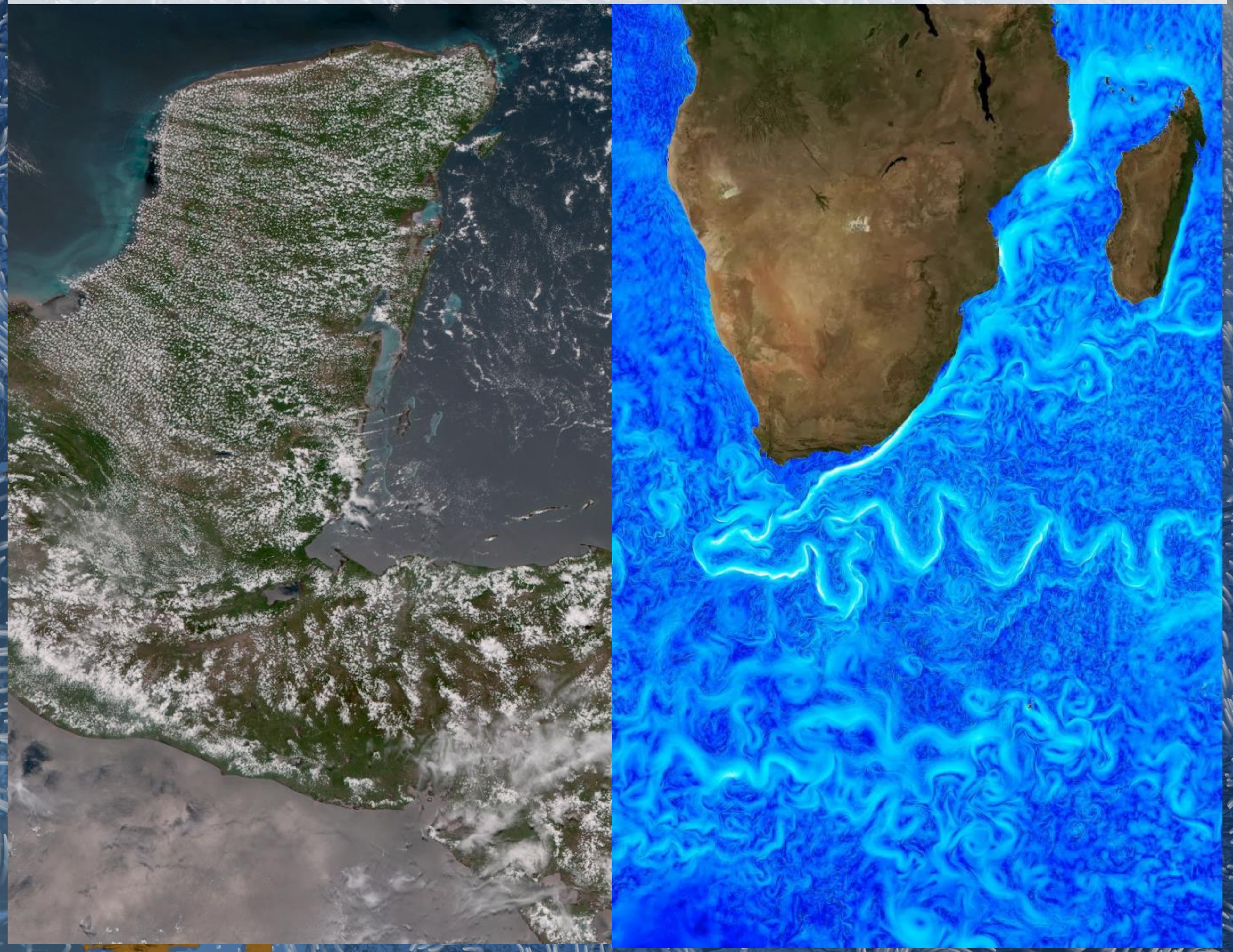


Performance

Performance benchmarks show significant speedups when running on a GPU. Large simulations on an Nvidia Tesla V100 GPU require ~1 nanosecond per grid point per iteration. GPU simulations are therefore roughly 3x more cost-effective than CPU simulations on cloud computing platforms such as Google Cloud. A GPU with 32 GB of memory can time-step models with ~150 million grid points assuming five fields are being evolved; for example, three velocity components and tracers for temperature and salinity. These performance gains permit the long-time integration of realistic simulations, such as large eddy simulation of oceanic boundary layer turbulence over a seasonal cycle or the generation of training data for turbulence parameterizations in Earth system models Short-term future development plans include support for distributed parallelism with CUDA-aware MPI as well as topography.



The main sources of uncertainty in the accuracy of climate models arise from the extreme difficulty of modeling small-scale physical processes such as clouds in the atmosphere (left) and mesoscale eddies, the "weather" of the ocean (right).



Background credit: NASA/Goddard Space Flight Center Scientific Visualization Studio

https://github.com/CliMA/Oceananigans.jl