Environment and Climate Change Canada
Environnement et Changement climatique Canada

Canada

Enhancing high-resolution simulations of the global ocean with machine learning using SmartSim

Andrew Shao

Andrew.Shao@canada.ca

Environment and Climate Change Canada

National Center for Atmospheric Research
Scott Bachman
Gustavo Marques (NCAR)

Hewlett Packard Enterprise
Sam Partee
Alessandro Rigazzi
Matt Ellis
Benjamin Robbins

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Hewlett Packard Enterprise

How do we leverage advances in machine-learning in numerical simulations?

- Machine-learning and traditional, numerical scientific codes come from two different approaches
  - ML/Data Science: Give me enough data and I'll solve all your problems
  - Numerical Codes: Give me enough Taylor expansions and I'll solve all your problems
- Combining the two requires reconciling the **philosophical** differences
  - Creativity and collaboration amongst domain scientists, data scientists, and computer scientists
- Combining the two requires reconciling **technical** differences
  - Python, arguably, the lingua franca of machine learning (some C++)
    - What about C or (gasp) Fortran?!
  - How can we use machine learning at scale on heterogeneous architectures?

What is a climate model?

- Land, ocean, atmosphere, and cryosphere components all coupled together
- Governing equations
  - Fluid dynamics
  - Thermodynamics
  - Chemistry
  - Biology
- Generally parallelized via domain or spectral decomposition
- Freely running, not constrained by observations

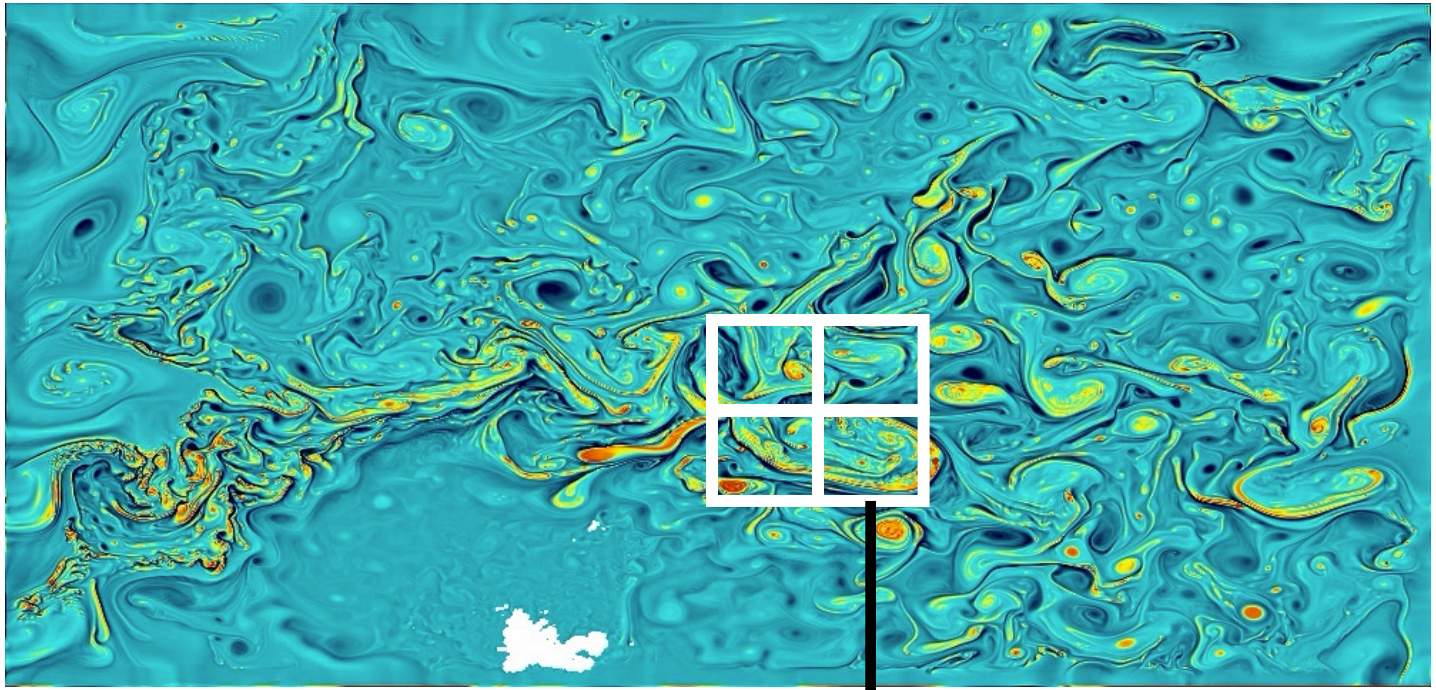
Image courtesy of NOAA

Climate modelling from an HPC perspective

- Horizontal discretization
  - 360x210x50 Ocean/ice
  - 128x64x49 atmosphere/land
- Temporal discretization
  - 1 hour: Ocean/ice
  - 15 minutes: Atmosphere
- A climate change experiment
  - 2000 years spinup
  - 150 years for the historical (1850-2015)
  - 85 years for 21st century
  - Multiple ensemble members to account for internal variability

- IPCC Report published on Monday
- CanESM5's contribution
  - 300,000 years of simulation
  - 40PB of data on tape
  - 500TB data being served
  - 150 million CPU hours
- **CanESM5 is one of the lowest resolution models**
  - Significant, resolution-resolution biases in the ocean
  - To resolve the largest scales of oceanic turbulence: ~1000x more expensive
  - Next-generation model expected to be about 64x more expensive (quadruple the resolution)

What turbulence in the ocean looks like



Big whorls have little whorls
Which feed on their velocity
And little whorls have lesser whorls
And so on to viscosity.

-Lewis Fry Richardson

Turbulence is pretty… **IMPORTANT**

1 white square: typical size of an ocean grid box
(simulation here is 50x finer in each direction)

4 white squares: roughly sets the size of the physical feature that the model can resolve

Machine-learning approaches to turbulence parameterizations

Reynolds averaged, incompressible, rotating, Boussinesq equations

$$\frac{D\overline{\mathbf{u}}}{Dt} + \nabla \cdot \overline{\mathbf{u}' \otimes \mathbf{u}'} + \boldsymbol{\Omega} \times \overline{\mathbf{u}} = -\nabla \overline{p} - \overline{b} \nabla Z + \nu \nabla^2 \overline{\mathbf{u}}$$ Momentum

$$\frac{D\overline{b}}{Dt} + \nabla \cdot \overline{\mathbf{u}' b'} = \kappa \nabla^2 \overline{b}$$ Thermodynamics

$$\nabla \cdot \overline{\mathbf{u}} = 0$$ Continuity

Parameterizations need to capture the eddy terms $\overline{\mathbf{u}' \otimes \mathbf{u}'}$ $\overline{\mathbf{u}' b'}$ using the 'resolved' terms

- Brute force (pure data science approach):
  - Replace the eddy terms wholesale with a trained model
    - Con: Constraining the predictions with known behaviors?
- Our approach (hybrid data science/physics) approach
  - Use existing turbulence theory to give the **functional form**
  - Use Machine Learning to predict **the strength**

b. Non-eddy-resolving models

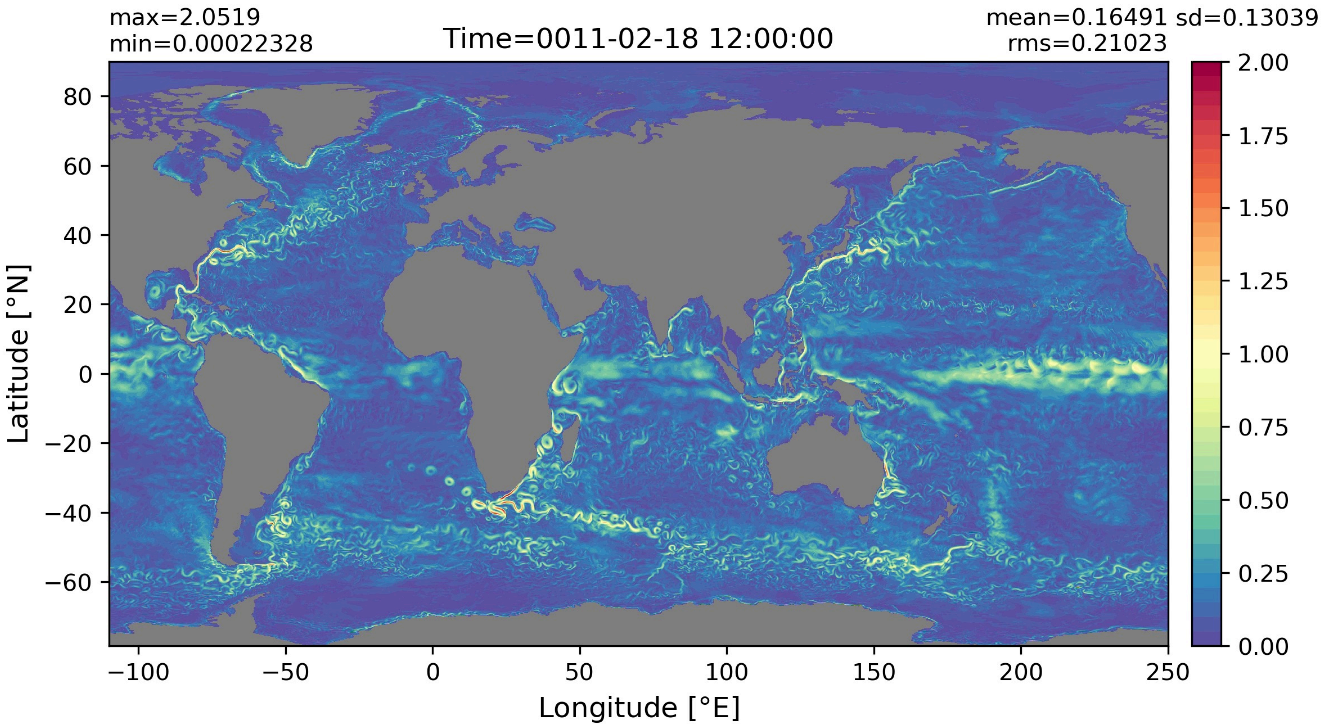
Therefore, the eddy mixing can be represented in approximate non-eddy-resolving models by the equation

$$\frac{\partial^2 h}{\partial t \partial \rho} + \nabla_\rho \cdot \left(\frac{\partial h}{\partial \rho} \mathbf{u} \right) + \nabla_\rho \cdot \mathbf{F} = 0. \qquad (6)$$

The famous Gent-McWilliams (1990)

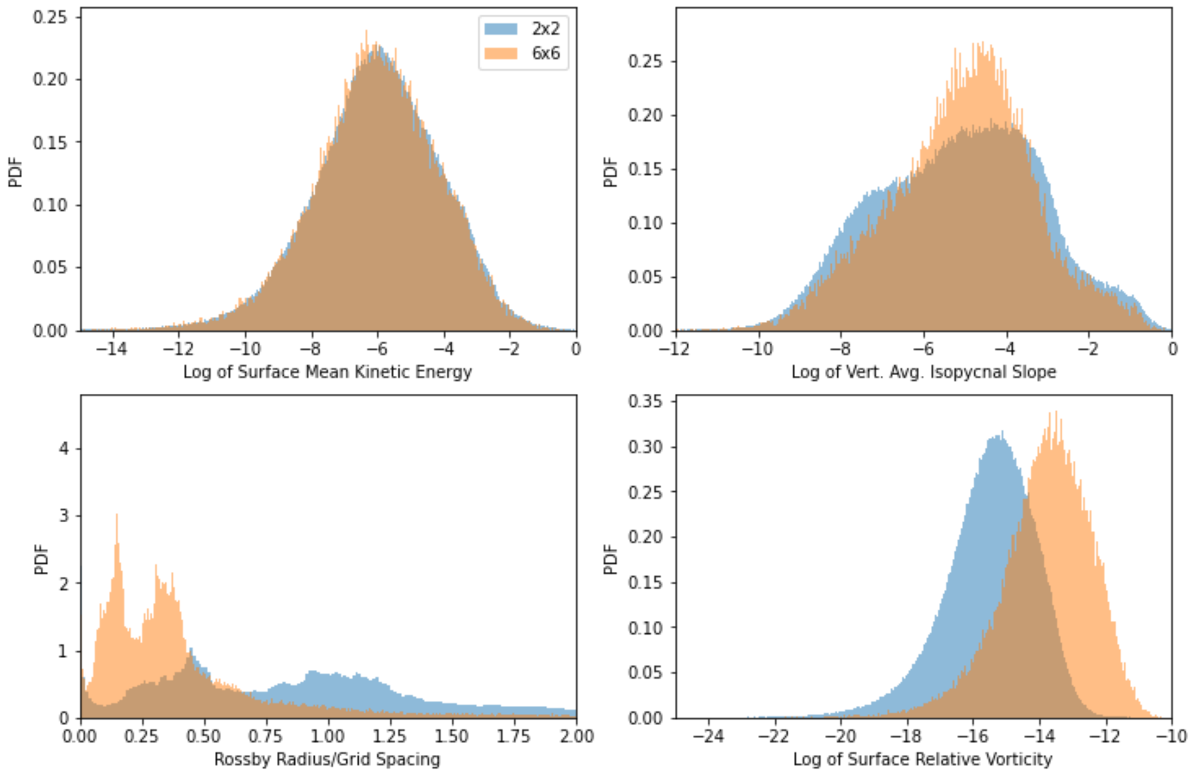
Enhancing Gent-McWilliams with a Neural Network

- Gent-McWilliams effectively 'smooths' horizontal density gradients using a specified diffusivity/coefficient
- Coefficient can be calculated from Eddy Kinetic Energy (EKE)
- Existing bleeding-edge prognostic equation for calculating EKE in a coarse resolution model *OMITS* or poorly approximates key terms
- **Our approach:** Replace this equation with a neural network
    - Generate training data using a 10km resolution MOM simulation which 'resolves' eddy fields and coarsen
    - Predict EKE from 'resolvable' fields
        - DO NOT ENCODE SPATIAL INFORMATION
    - Predict EKE grid cell by grid cell
    - Transform EKE into a diffusivity using known equations



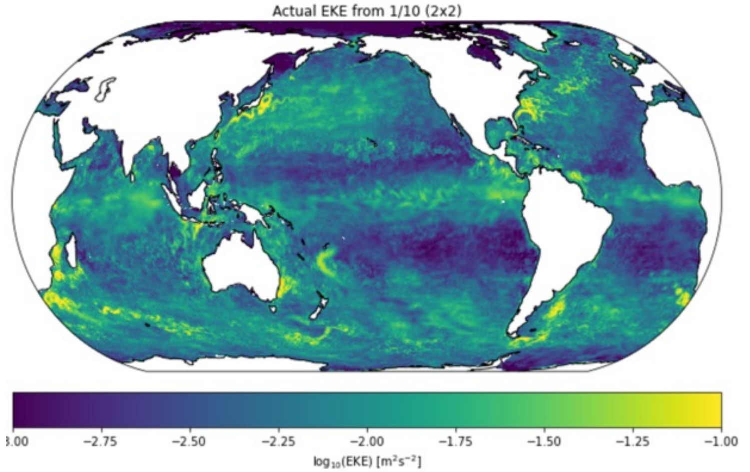
Training and validating the neural network: ResNet with custom loss

Distributions of training features

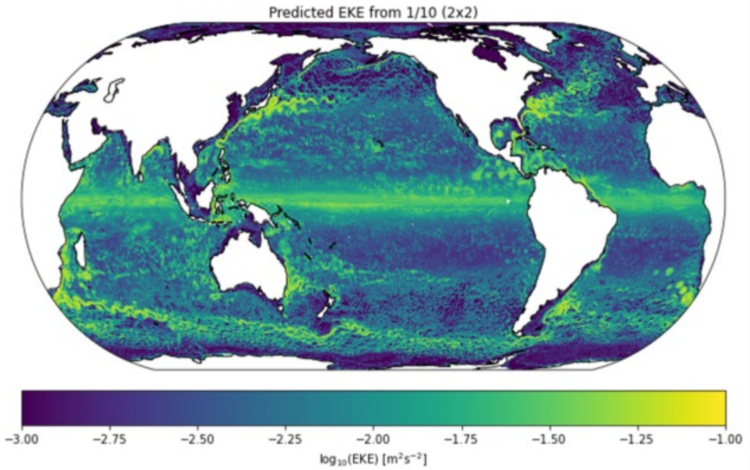


Coarsening the data mimics Reynold's averaging and what the targeted (low-res) simulation is able to resolve

"Truth"



Machine learning

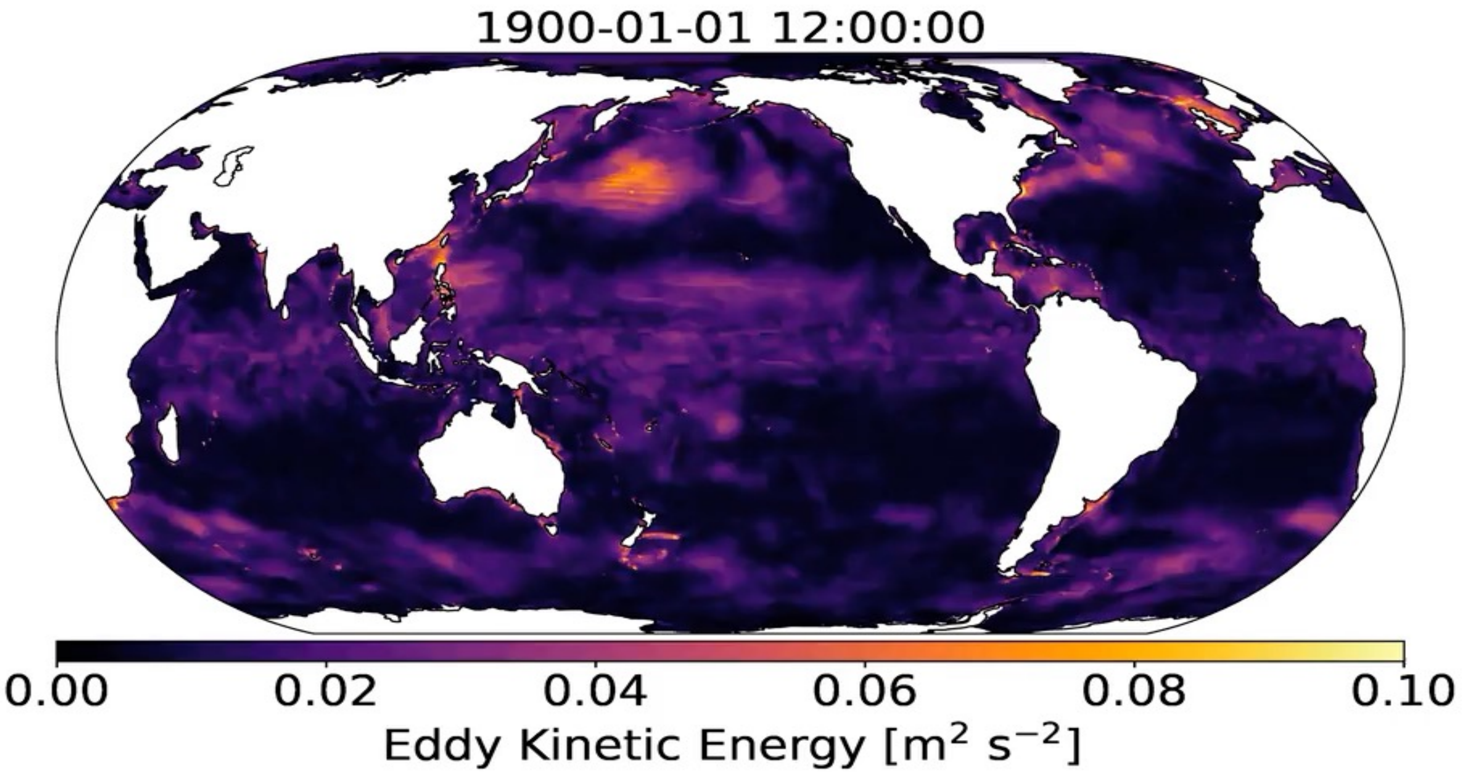


Targeted simulation to improve

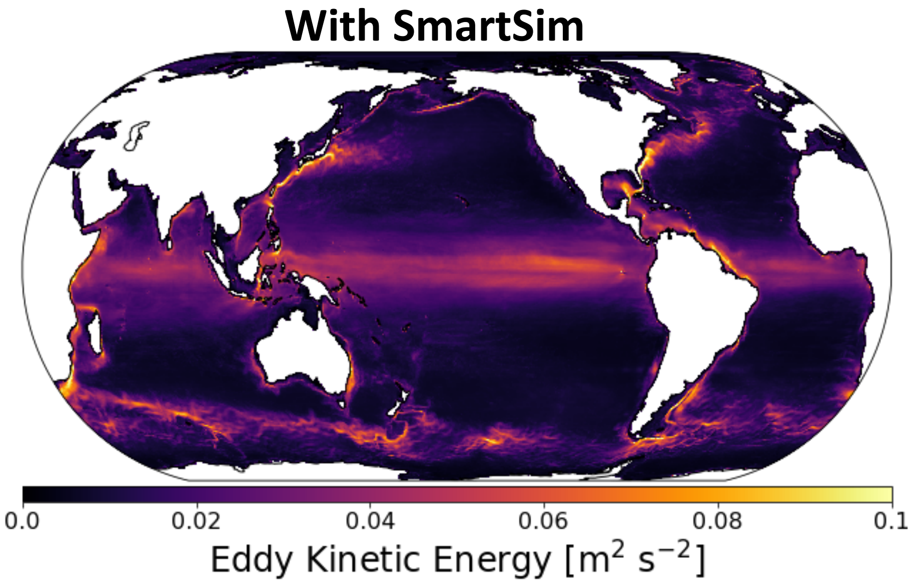
- Modular Ocean Model version 6
  - Numerics:
    - Finite volume, staggered C-grid
    - Logically rectangular, non-rectangular horizontal grid (1440x1080, ~30km)
    - 75 vertical levels, arbitrary Lagrangian-Eulerian vertical coordinate
    - Split Timestep:
      - ‘Fast’ depth-averaged dynamics: 20s
      - ‘Slow’ layer-by-layer dynamics: 900s
      - ‘Slow’ tracer transport and thermodynamics: 2 hours
- Applications
  - Coupled Model Intercomparison Project version 6
  - Regional modelling (climate downscaling)
  - Coupled, operational weather prediction
  - Hindcasts (i.e. trying to simulate the past)
  - Fundamental climate/oceanography research

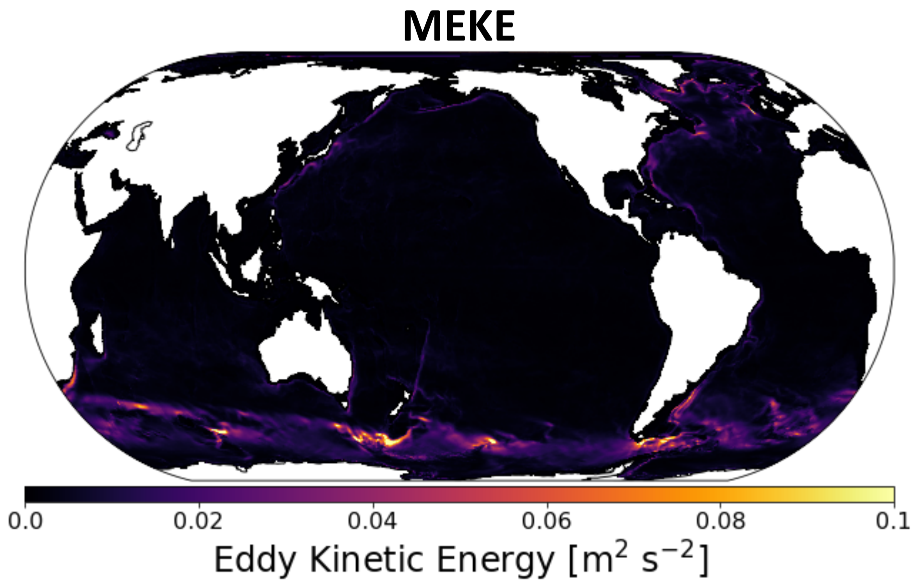
Applying this in high-res (for a climate model) 30km simulation using SmartSim (to be discussed)

Key features:
- Online Inference (every 3 hour timestep)
- Inference done grid cell by grid cell



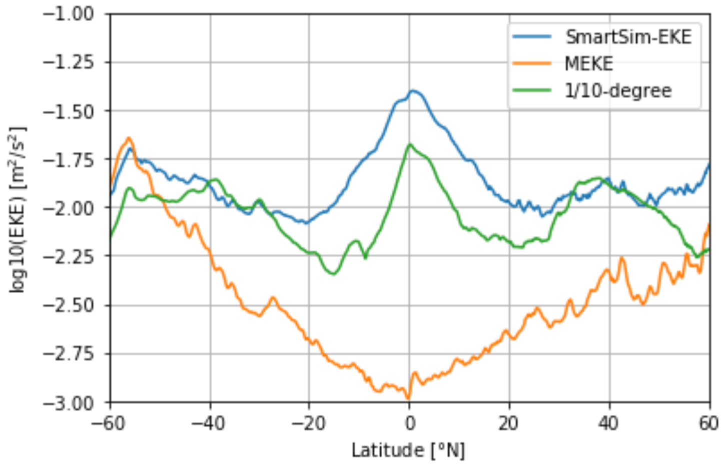
Evaluating EKE in an ensemble of simulations



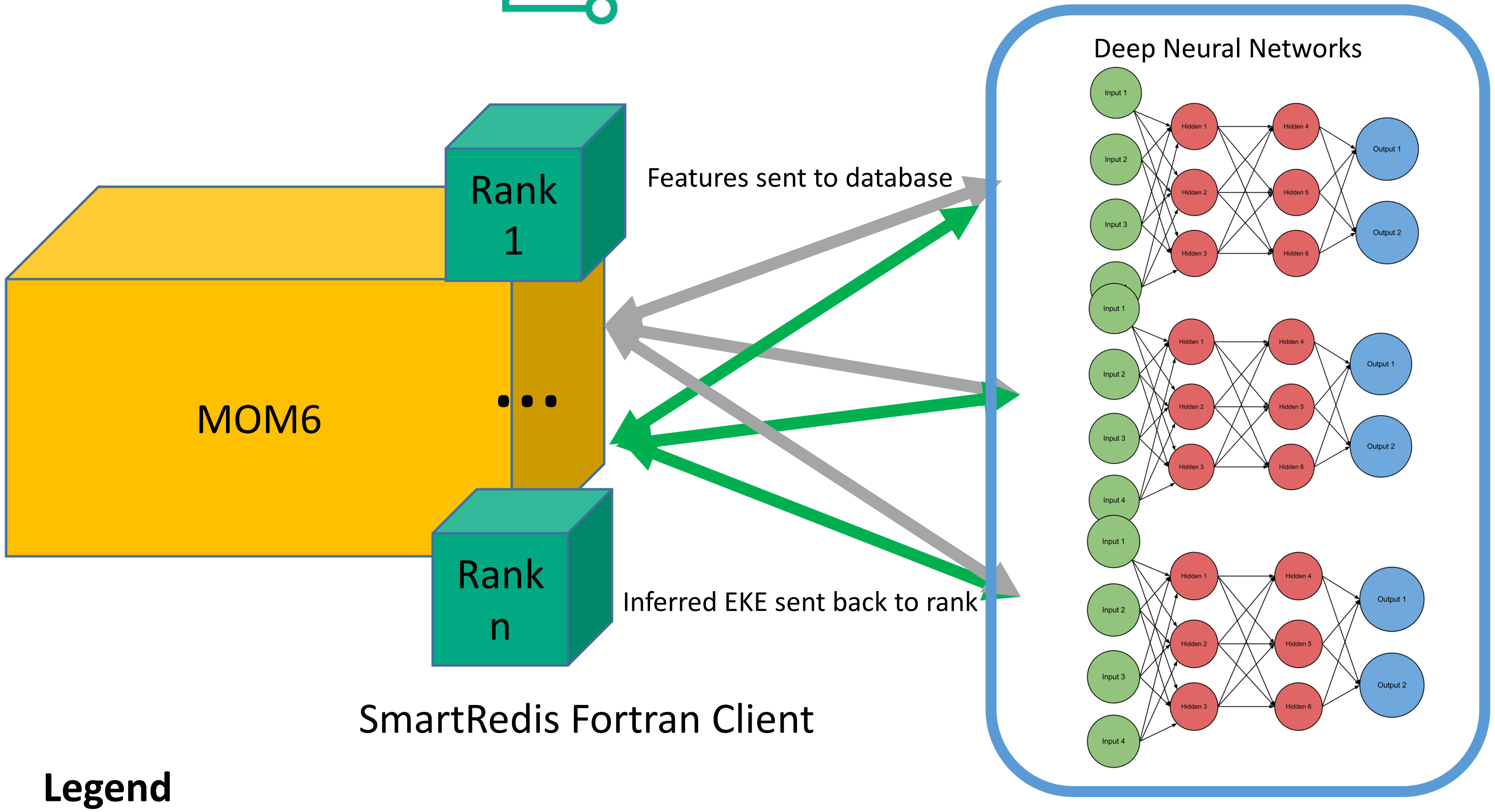


Computational Metrics

- 10,920 cores (910 per member)
- 16 Database nodes with V100s
- 20-year integration (1.5M core hours)
- No crashes or instabilities
- About **1 trillion inferences** (2 million/second)
- 6% slower than the prognostic MEKE
  - Running 1/10-degree (10km): 15x slower



SMARTSIM: How we did it!



RedisAI distributed ‘database’ nodes with GPUs

Legend

- MOM6 sends input features
- DNN models return EKE values

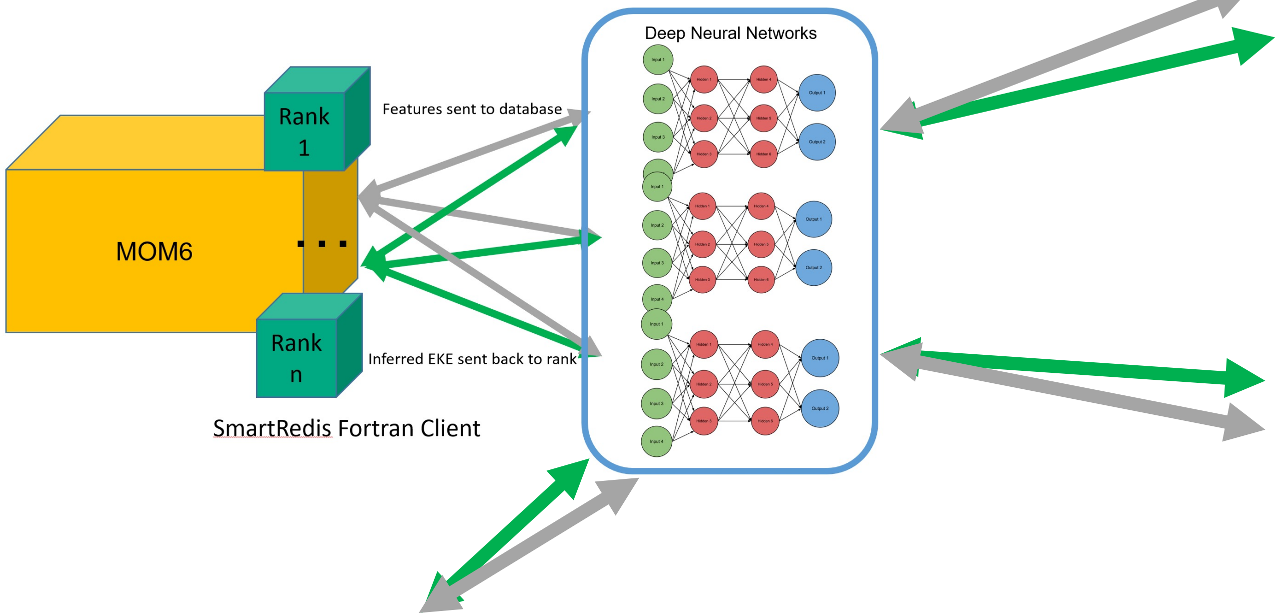
- SmartSim orchestrates ensembles of simulations, the database, and other components (Python, C, C++, or Fortran)
- Clients allow model to stream data in and out of their simulation
- Data stored in a key/value datastore
- RedisAI database allows for inference on GPU/TPU-enabled nodes
  - All major machine learning packages are supported (tensorflow, keras, sci-kit learn, ONXX, etc.)
- Database is scalable across and within nodes
- **Computations can be done in Fortran, Python, C, and C++ without needing to write extensive code**
  - **Fortran-based model will interact only with the Fortran client**

Instrumenting MOM6

ON MOM6 INITIALIZATION ON EVERY MPI RANK	
• Initialize the SmartRedis client used to communicate with the database	`call client%initialize(.true.)`
• Create a unique identifier for every subdomain	`write(key_suffix, '(A,I6.6)'),PE_here()`
• Load a previously trained machine-learning model into the database	`call client%set_model_from_file(...)`
EVERY MOM6 TIMESTEP ON EVERY MPI RANK	
• Put a state variable (e.g. surface relative vorticity) into the database	`call client%put_tensor( ... )`
• Predict EKE using the machine-learning model	`call client%run_model( ... )`
• Retrieve the EKE prediction from the database	`call client%unpack_tensor( ... )`

- MOM6 only ever calls native Fortran (2013) code
- All inference is done in the (remote) database
  - No need to continually update a Fortran implementation of a neural network architecture
- MOM6 nodes do NOT need a local accelerator

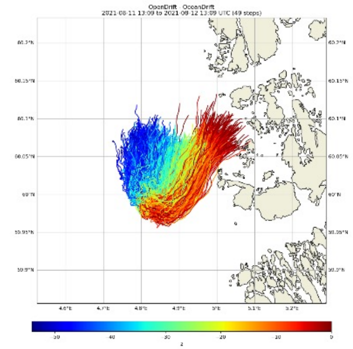
New computing ecosystems enabled by SmartSim



- Real-time data monitoring
- Rapid code prototyping

- Two-way human interaction with the simulation
- Analysis/visualization with diskless I/O
- Direct computational steering

Particle Tracking
- OceanParcels
- OpenDrift



In closing …

- Conclusions
  - Combined data science and ‘classical’ physics approaches to improve a global-scale, realistic ocean climate simulation
  - Replaced a prognostic equation of the model that has an first-order effect on the simulation accuracy
  - Ensemble of ocean runs was done on a heterogeneous cluster with high CPU and GPU utilization
  - SmartSim is a lightweight solution for creating multilingual computing ecosystems at scale
  - Centralizing data and communication in a distributed database opens up novel data and human-guided simulations
- Ongoing work
  - Continued, open-source development of SmartSim (https://github.com/CrayLabs/)
  - Multicentennial integrations with SmartSim-EKE: Neural Network pruning? Other, simpler architectures
  - NCAR collaboration using SmartSim, machine-learning, and CESM for hurricane tracking and intelligent model sampling in fully-coupled climate model
  - Incorporation of SmartSim into NEMO and MOM6

Additional Information

- Interested in using SmartSim for your application?
  - Contact me at Andrew.Shao@canada.ca
  - CrayLabs Slack: https://join.slack.com/t/craylabs/shared_invite/zt-nw3ag5z5-5PS4tIXBfufu1bIvvr71UA

- Arxiv paper: https://arxiv.org/abs/2104.09355