Oceananigans.jl

breakthrough *resolution*, *memory*, and *energy efficiency* in global ocean simulations

Navid Constantinou

@navidcy

Australian National University

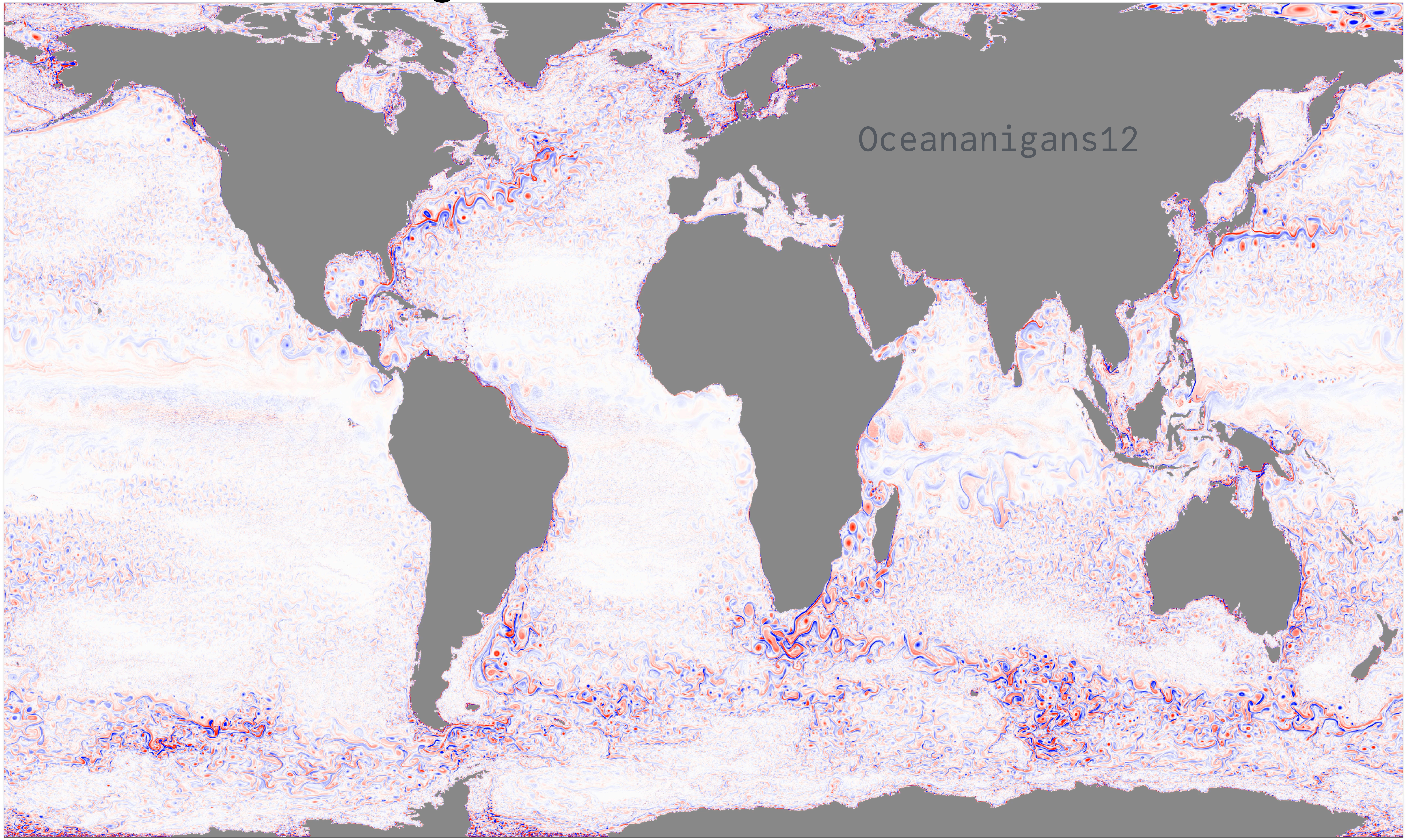
Massachusetts Institute of Technology

Simone Silvestri

Gregory Wagner

and also all
Ocean Core
CliMA Dev Team

CliMA CLIMATE MODELING ALLIANCE



near-global (75°S–75°N) ocean simulation at 1/12° horizontal resolution, 48 vertical levels @ 68 Nvidia A100 achieving 10 simulated years per day

GitHub

julia

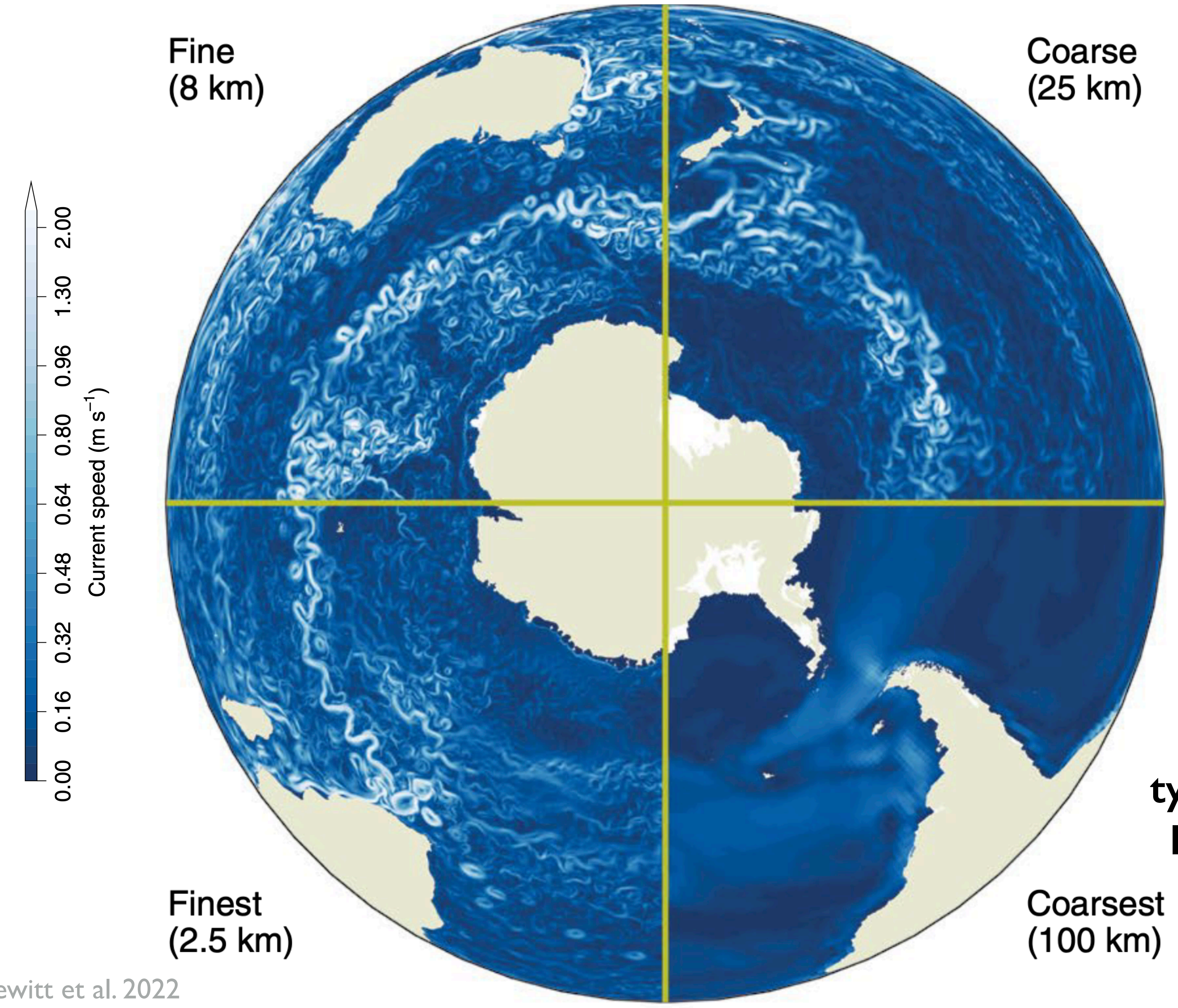
NVIDIA

GPU

breakthrough resolution — why do we keep pushing for higher resolution?

turbulent flows develop fine-scale features… the more we resolve, the more we see

remember Adele's talk on Wed



typically used in
IPCC & CMIP

Hewitt et al. 2022

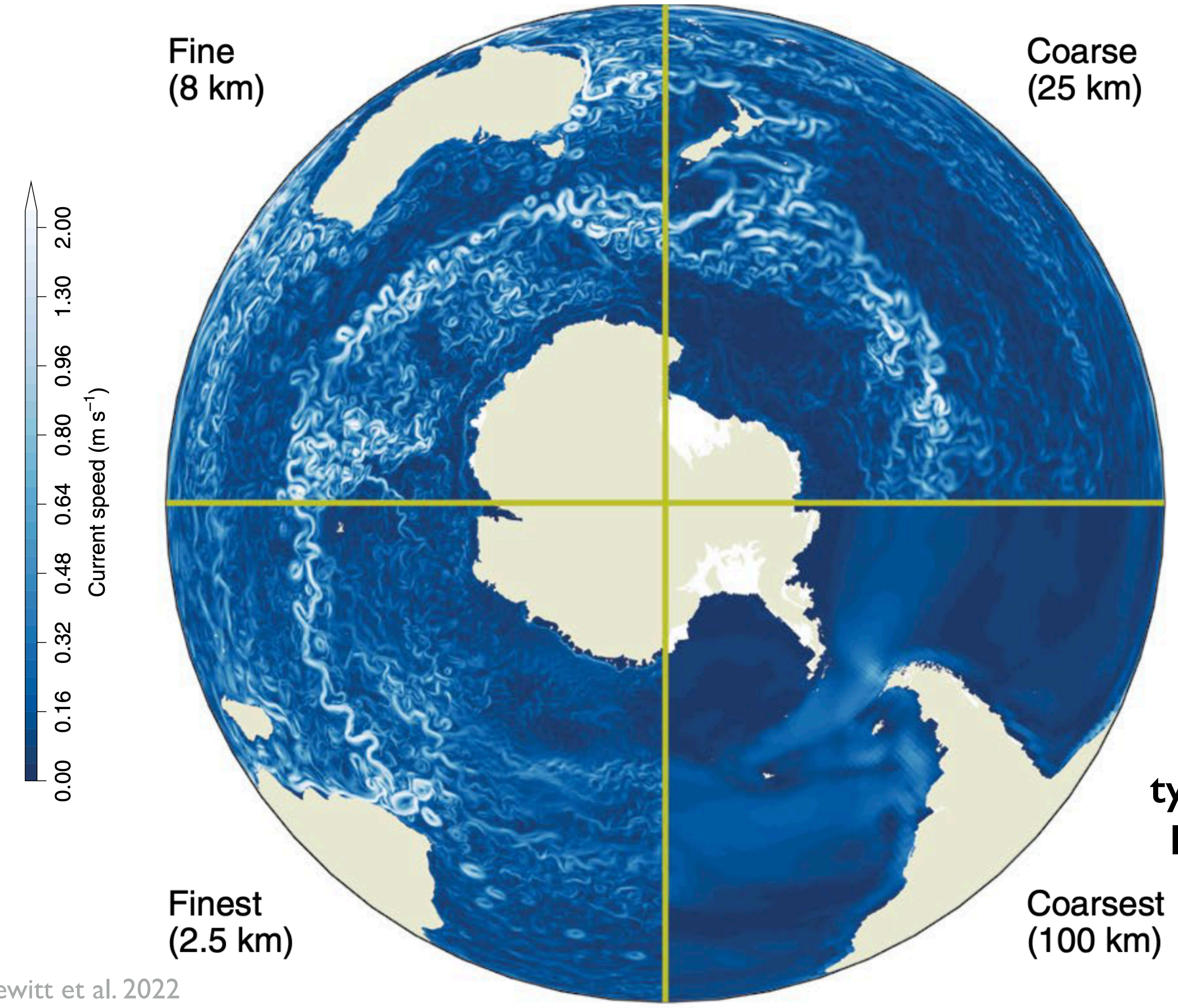
flows at metre-scales or even 10-100km (eddies 🌀) shape the global ocean circulation at century timescales!

small and short-lived flow features affect global circulation over decades-centuries

breakthrough resolution — why do we keep pushing for higher resolution?

turbulent flows develop fine-scale features… the more we resolve, the more we see

remember Adele's talk on Wed



typically used in IPCC & CMIP

Hewitt et al. 2022

"From little things big things grow"
Paul Kelly

"From little eddies large-scale ocean circulation grows"
adage

flows at metre-scales or even 10-100km (eddies 🌀) shape the global ocean circulation at century timescales!

small and short-lived flow features affect global circulation over decades-centuries

Climate Modeling Alliance

CliMA
CLIMATE MODELING ALLIANCE

Caltech
JPL
Jet Propulsion Laboratory
California Institute of Technology
MIT

`ClimateModeling v2.0`

…building a ***new Earth system model***
that leverages recent advances in the computational and data sciences
to learn directly from a wealth of Earth observations from space and the ground.

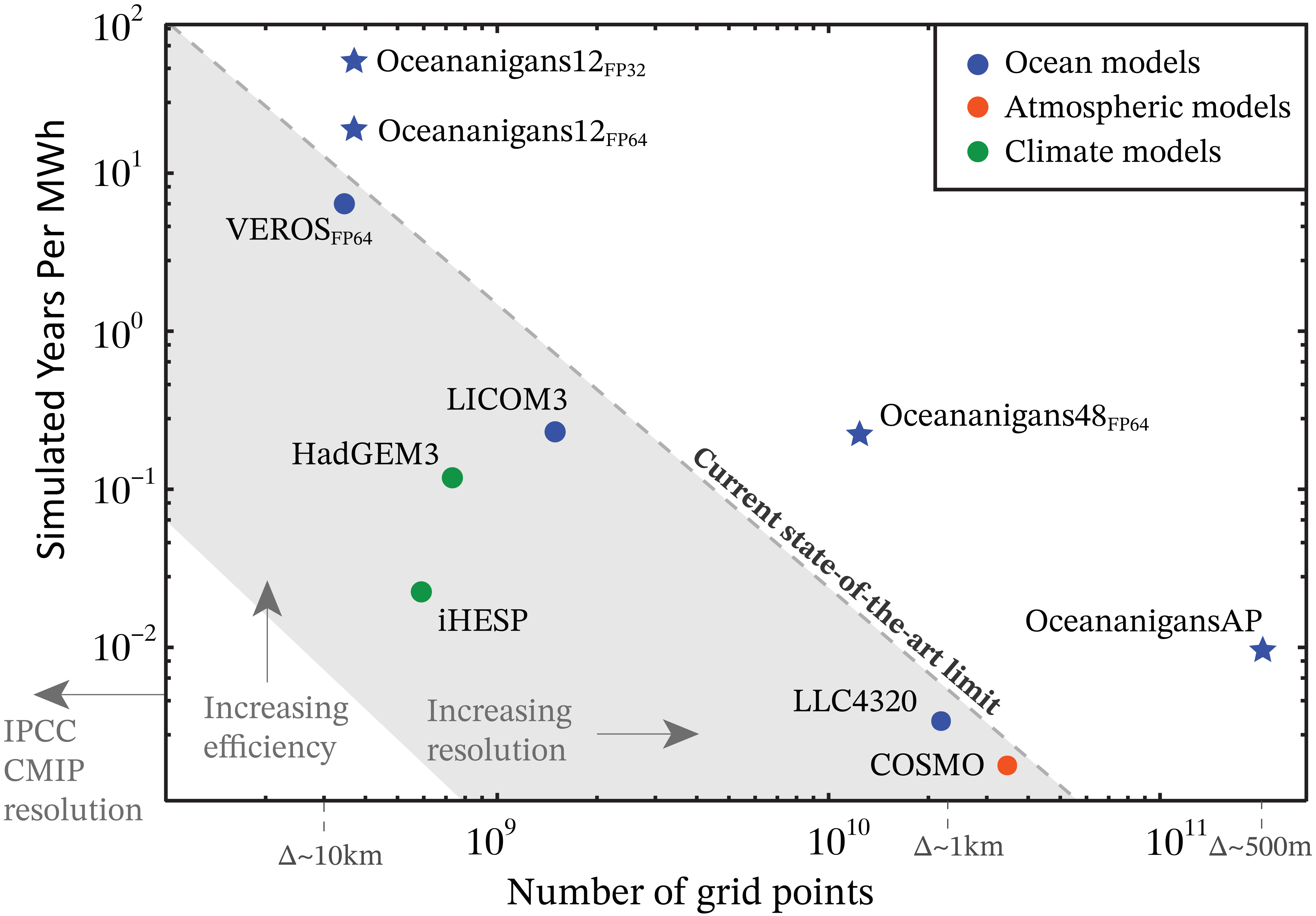
Andre Souza | Chris Hill | Simone Silvestri | Greg Wagner | Raffaele Ferrari | Jean Michel Campin | Navid Constantinou | John Marshall

Ocean Model Dev Team

Grace O'Neil | Xin-Kai Lee | M. G. | Ali Ramadhan | Ulyana Piterbarg | Adeline Hillier | Valentin Churavy

Oceananigans is the ocean core of Clima's Earth System model

pushing boundaries: Oceananigans on the big scene



near-global (75°S–75°N)

★ Oceananigans12

1/12° horizontal resolution
48 vertical levels
(video at title slide)

★ Oceananigans48

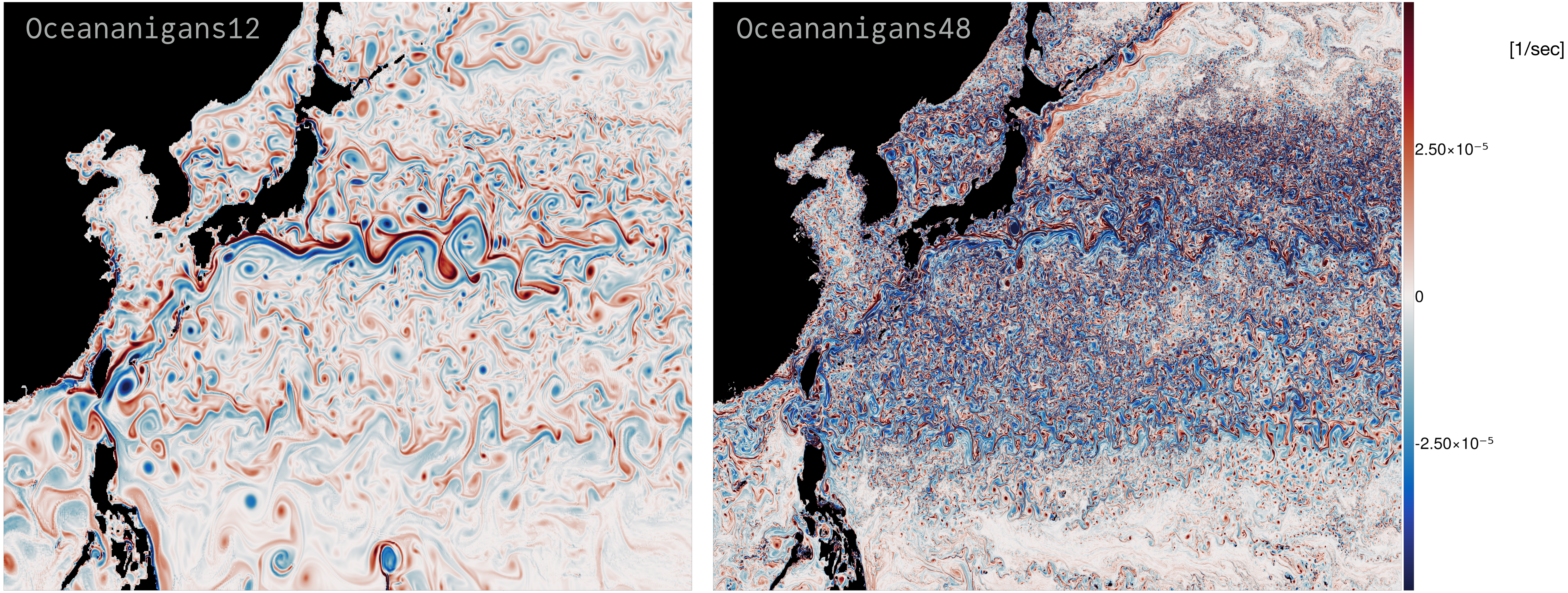
1/48° horizontal resolution
100 vertical levels

★ OceanangansAP

idealised Aqua-Planet + 2-continent config
1/192° horizontal resolution
100 vertical levels

breakthrough energy efficiency

surface relative vorticity



$^1/_{12}°$ horizontal resolution + 48 vertical levels
~ 10^9 grid points

10 Simulated Years Per Day
@ 68 Nvidia A100 GPUs

$^1/_{48}°$ horizontal resolution + 100 vertical levels
~ $5\text{x}10^{10}$ points

1 Simulated Year Per Day
@ 576 Nvidia A100 GPUs

but why is Oceananigans *so fast* and *so efficient?*

fast compute

Oceananigans is written from scratch for GPUs
(we didn't port a CPU-based model onto GPU)

memory leanness

minimise temporary array creations
loop over the domain as few times as possible

scalability

compute is fast, communication is expensive

novel algorithm for fast 2D motions (barotropic solver) allows
effective overlap of communication & computation

breakthrough memory efficiency

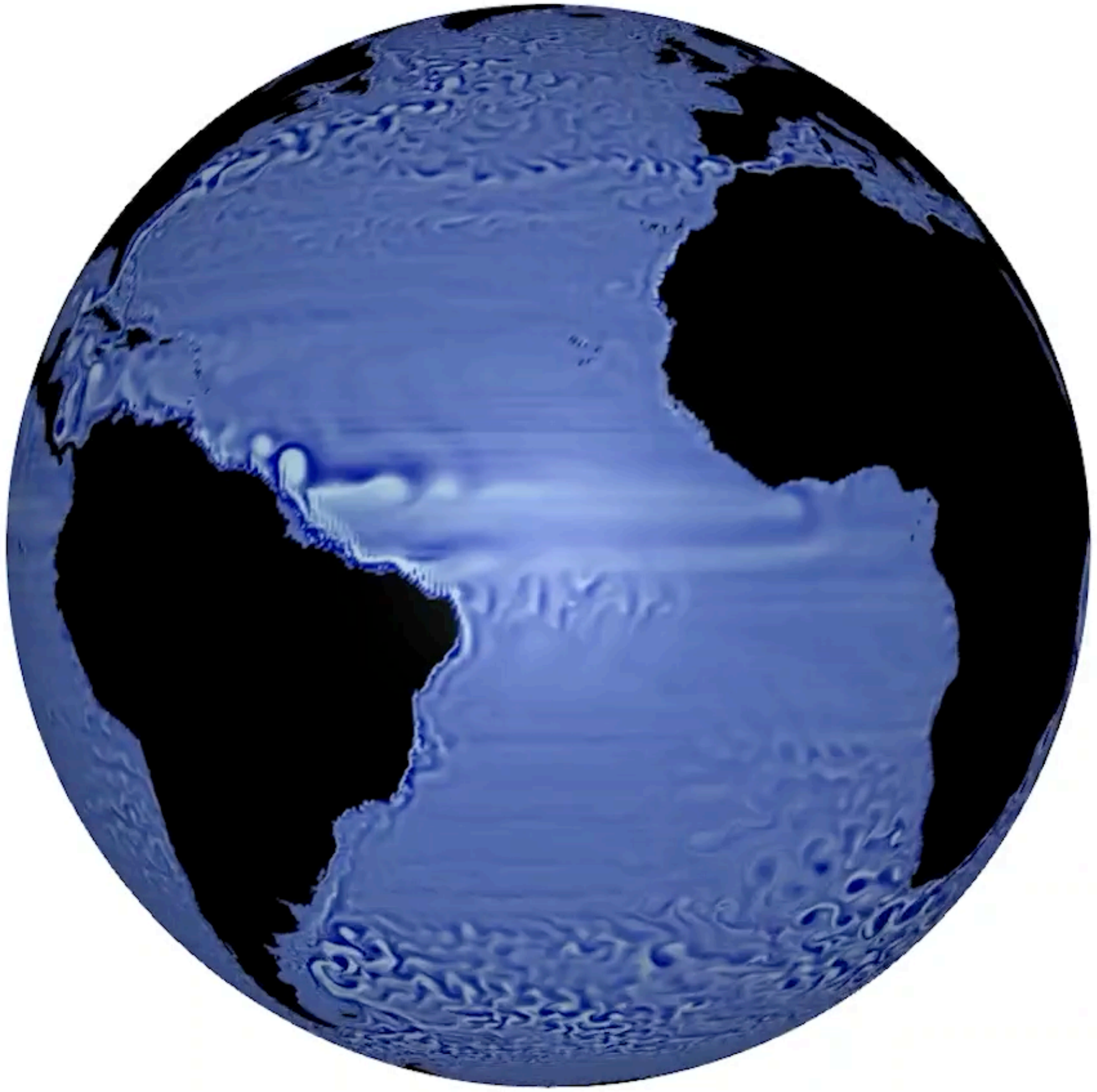
eddy permitting

$^{1}/_{4}°$ horizontal resolution

48 vertical levels

10 GB memory footprint

fits ***easily*** on 1 Nvidia V100 GPU



surface relative vorticity

jupyter

memory leanness

advection-diffusion
of temperature

$$\partial_t T = \underbrace{\underbrace{- \boldsymbol{u} \cdot \nabla T}_{\text{advection}} + \underbrace{\partial_z \left(\kappa \partial_z T \right)}_{\text{diffusion}}}_{\text{rhs}}$$

"classic" Fortran-style
temp arrays on CPU are cheap

temporary array ⟶
temporary array ⟶
temporary array ⟶

```
u_nabla_T = calculate_advection(u, v, w, T)
diff_T = calculate_diffusion(κ, T)
rhs_T = - u_nabla_T + diff_T
```

memory leanness

advection-diffusion
of temperature

$$\partial_t T = \underbrace{\underbrace{- \boldsymbol{u} \cdot \nabla T}_{\text{advection}} + \underbrace{\partial_z \left(\kappa \partial_z T \right)}_{\text{diffusion}}}_{\text{rhs}}$$

"classic" Fortran-style
temp arrays on CPU are cheap

temporary array ⟶ temporary array ⟶ temporary array ⟶

```
u_nabla_T = calculate_advection(u, v, w, T)
diff_T = calculate_diffusion(κ, T)
rhs_T = - u_nabla_T + diff_T
```

Oceananigans
GPU-friendly kernel fusion
no memory allocation

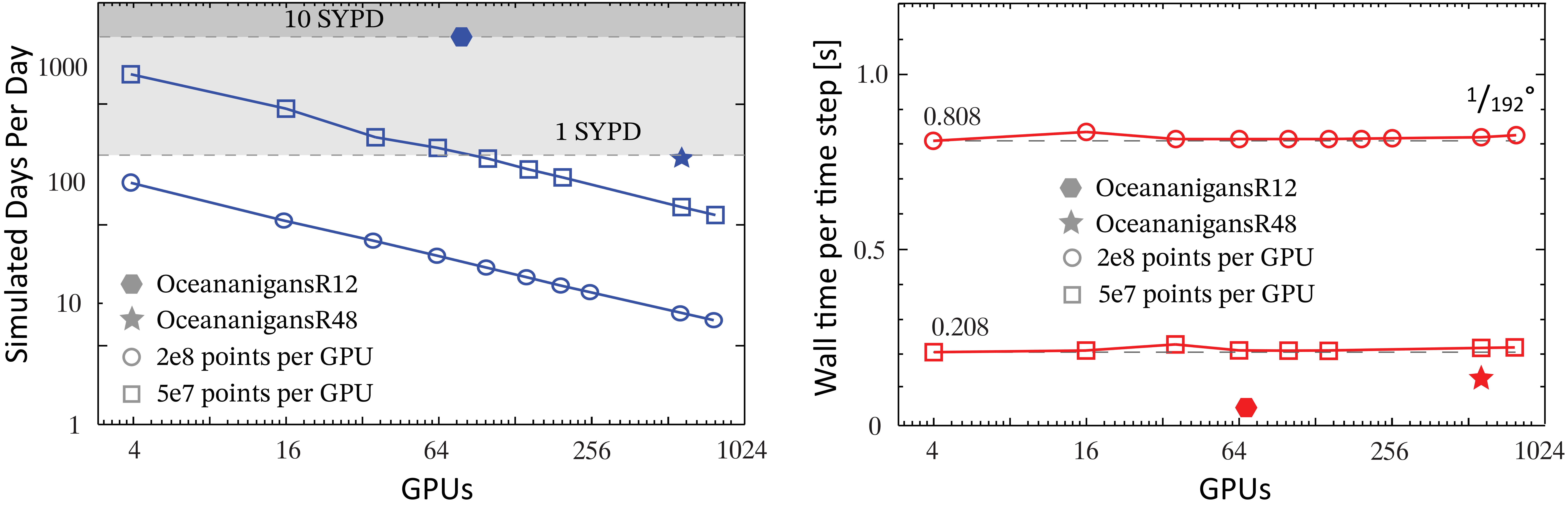
```
tendency_T(i, j, k) = - u_nabla_T(i, j, k) + diffusion(i, j, k)

@kernel function calculate_tendency_T!(rhs_T)
    i, j, k = @index(Global, NTuple)
    rhs_T[i, j, k] = tendency_T(i, j, k)
end
```

we load as few kernels as possible; only one for the tendency of each flow field
we loop over the grid once for every flow field

how do we scale?

weak scalings



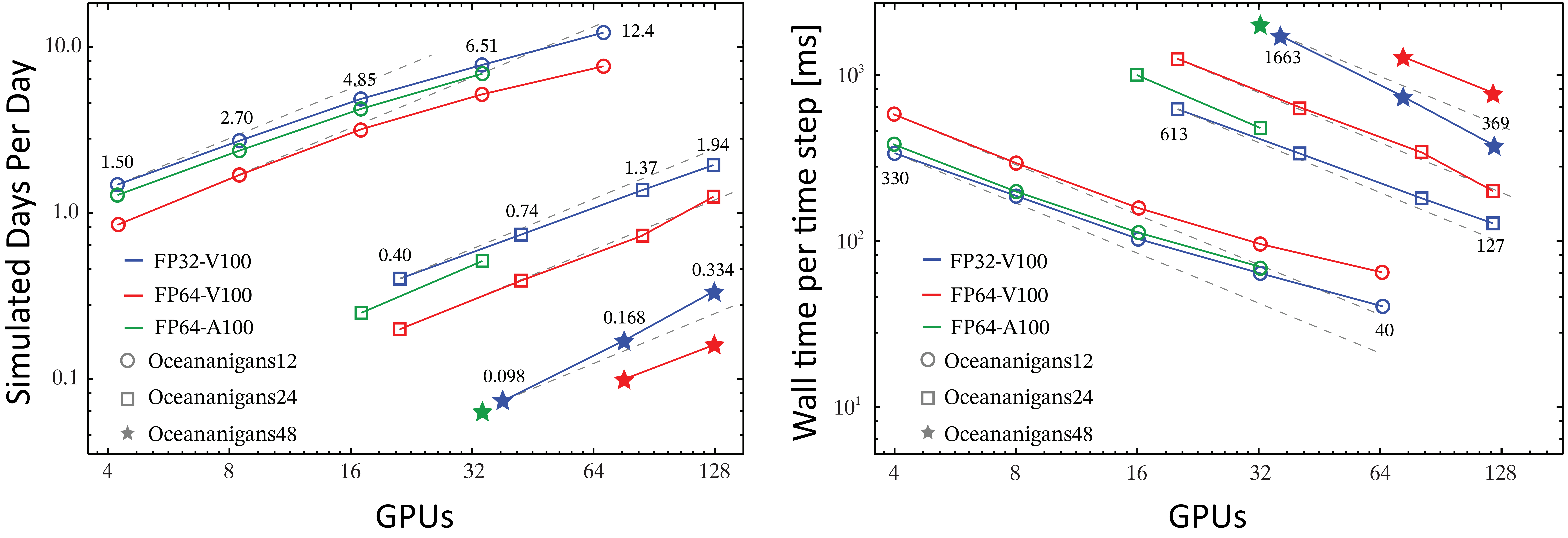
100 Simulated Years Per Day at $^1/_2°$ resolution (paleoclimate) @ 1 Nvidia V100 GPU

10 Simulated Years Per Day at $^1/_{12}°$ resolution @ 68 Nvidia A100 GPUs

1 Simulated Year Per Day at $^1/_{48}°$ resolution @ 576 Nvidia A100 GPUs

how do we scale?

strong scalings



100 Simulated Years Per Day at $^1/_2$° resolution (paleoclimate) @ 1 Nvidia V100 GPU

10 Simulated Years Per Day at $^1/_{12}$° resolution @ 68 Nvidia A100 GPUs

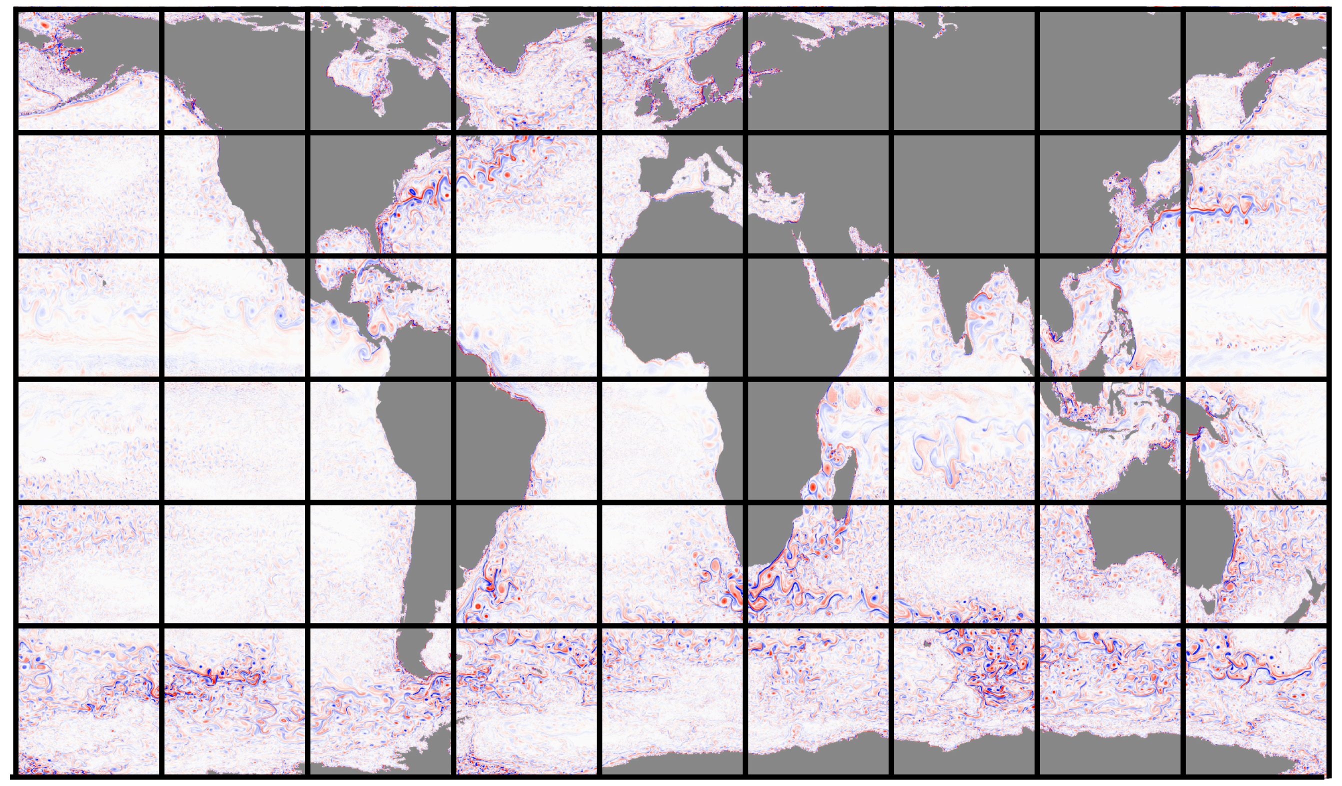
1 Simulated Year Per Day at $^1/_{48}$° resolution @ 576 Nvidia A100 GPUs

scalability

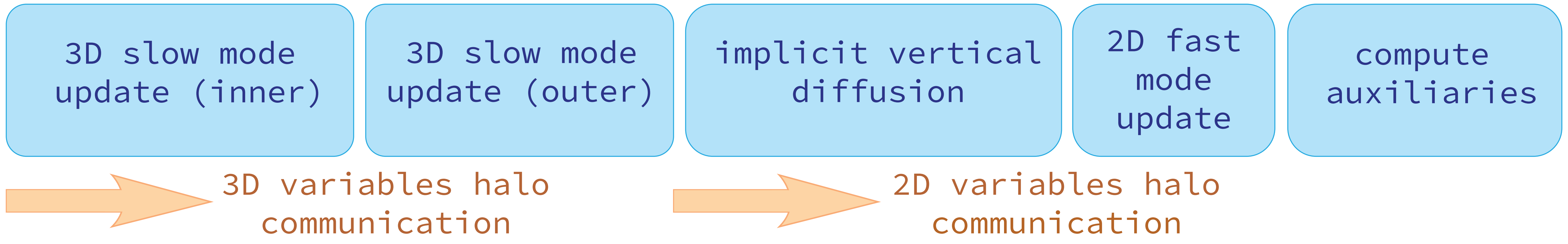
ocean flow motions

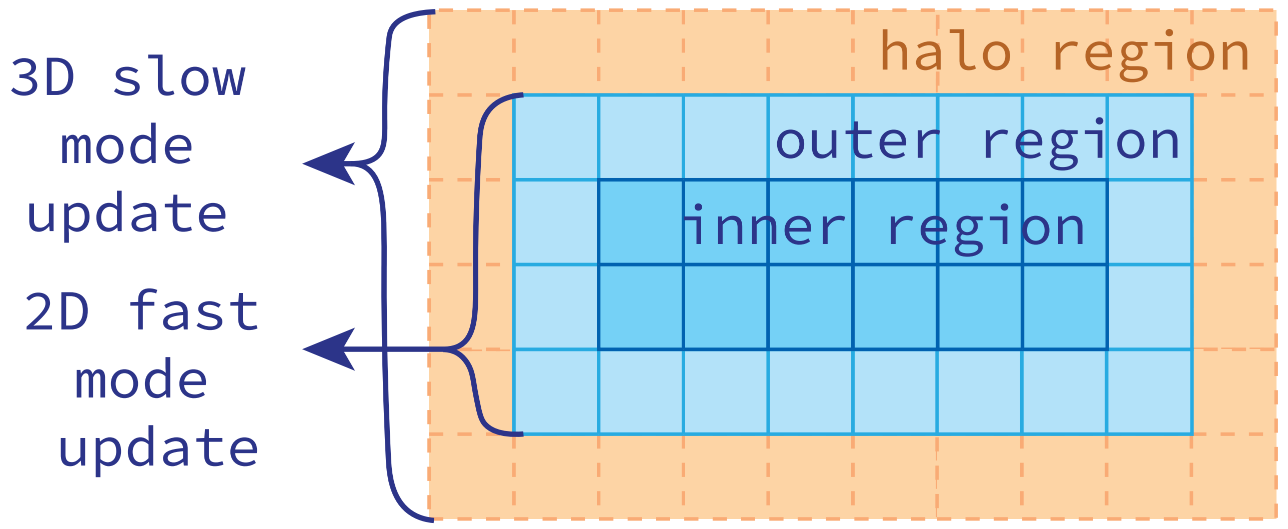
2D that vary fast O(seconds)

3D that vary slower O(minutes)



1 timestep = 1 3D timestep + many (eg 20-40) 2D timesteps





novel algorithm minimises communication in the expense of more computation (cheap on GPU)

👉 allows **overlap** of 2D solver with other tasks

conclusions

Oceananigans enables global ocean modelling
with unprecedented performance on GPUs

👉 start from scratch with GPU in mind

👉 software design enables memory leanness

👉 novel algorithm allows overlap of computation & communication

thanks

CliMA CLIMATE MODELING ALLIANCE http://clima.caltech.edu

github.com/CliMA/Oceananigans.jl GitHub