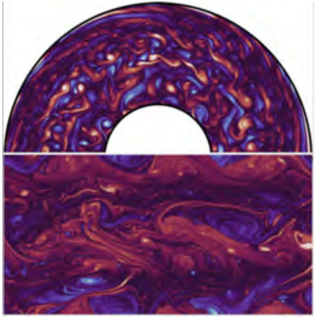


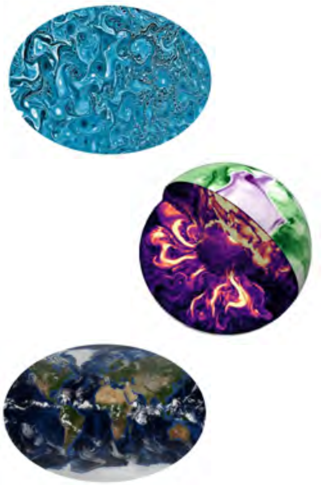
Learning subgrid-scale models for turbulent rotating convection: towards realistic timescales simulations

Workshop "Artificial Intelligence for HPC@Exascale"

Hugo Frezat Alexandre Fournier Thomas Gastine

October 2, 2024

Turbulent components of Earth systems



Sub-mesoscale permitting, cloud-resolving and geodynamo (credits: N. Schaeffer) simulations.

In geophysical systems:

- Ocean: mixing, boundary layers.
- Atmosphere: convection, clouds, gravity waves.
- Earth's core: geodynamo.

Turbulent state:

- Large range of structures.
- Non-linear interactions.
- Chaotic: sensitive to initial conditions.

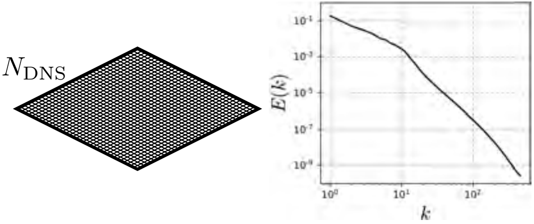
In simulations:

- Governed by Navier-Stokes equations (fluid motion), and induction (magnetic field).
- Discretized on a grid.

Computational limitations: hybrid modeling

Direct numerical simulation (DNS):

$$\frac{\partial \mathbf{y}}{\partial t} = f(\mathbf{y})$$



Grids on domain length L and corresponding energy spectrum.

Grid domain and spacing:

- Not reachable in realistic scenarios.

Reduced equations (LES):

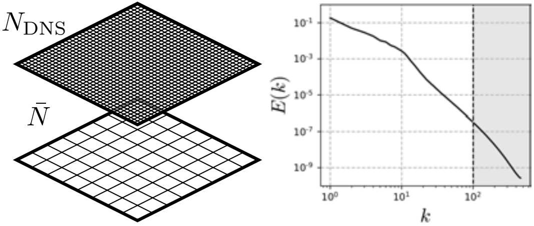
- Universal small-scale dynamics.
- Applying projection $\mathcal{T}(\mathbf{y}) = \bar{\mathbf{y}}$.
- Typically using a filter.

$$\frac{\partial \bar{\mathbf{y}}}{\partial t} = f(\bar{\mathbf{y}}) + \underbrace{\tau(\mathbf{y})}_{\mathcal{T}(f(\mathbf{y})) - f(\mathcal{T}(\mathbf{y}))}$$

Computational limitations: hybrid modeling

Direct numerical simulation (DNS):

$$\frac{\partial \mathbf{y}}{\partial t} = f(\mathbf{y})$$



Grids on domain length L and corresponding energy spectrum.

Grid domain and spacing:

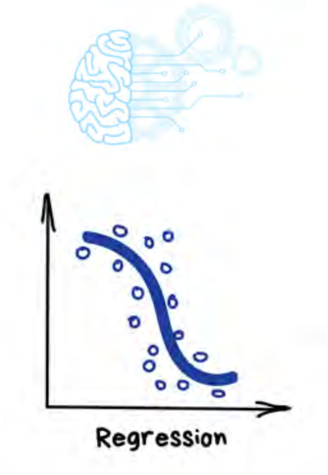
- Not reachable in realistic scenarios.

Reduced equations (LES):

- Universal small-scale dynamics.
- Applying projection $\mathcal{T}(\mathbf{y}) = \bar{\mathbf{y}}$.
- Typically using a filter.

$$\frac{\partial \bar{\mathbf{y}}}{\partial t} = f(\bar{\mathbf{y}}) + \underbrace{\tau(\mathbf{y})}_{\mathcal{T}(f(\mathbf{y})) - f(\mathcal{T}(\mathbf{y}))}$$

Machine learning in physical sciences



Predicting the subgrid term using machine learning is a regression problem.

Subgrid modeling in physics:

- Open-problem (1963-.).
- Models designed from well-known functions (PDEs).

Scientific Machine Learning (SciML):

- Recent field ($\sim$2018-.).
- Parametric functions (neural networks for e.g.).
- Models designed as a supervised learning problem.
- Using data from DNS.

Outlines

1. Designing **learning strategies** for stable simulations

   Ideal domains: uniform grid and periodic boundary conditions.

2. **State-based** learning in the annulus

   Not so ideal domains: non-uniform grid and rigid boundary conditions.

3. Beyond spectral: neural operators

   Interaction with current solvers.

4. Conclusion

Designing learning strategies for stable simulations

Subgrid-scale challenges in QG dynamics

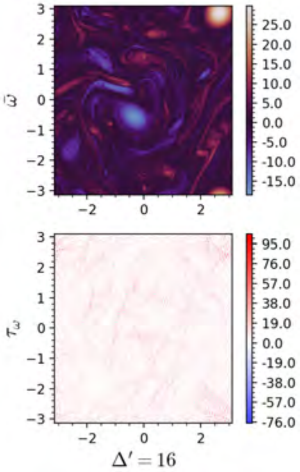
Example: quasi-geostrophic turbulence

$$\partial_t \omega + \underbrace{J(\psi, \omega)}_{\text{advection}} = \underbrace{\nu \nabla^2 \omega}_{\text{diffusion}} - \underbrace{\mu \omega}_{\text{drag}} - \underbrace{\beta \partial_x \psi}_{\text{latitude variation}} + \underbrace{F}_{\text{forcing}}$$

A simplified rotating geophysical surface system:

- Vorticity equation.
- Two-dimensional.
- 1 layer.

$$\partial_t \bar{\omega} + J(\bar{\psi}, \bar{\omega}) = \nu \nabla^2 \bar{\omega} - \mu \bar{\omega} - \beta \partial_x \bar{\psi} + \bar{F} + \underbrace{J(\bar{\psi}, \bar{\omega}) - \overline{J(\psi, \omega)}}_{\tau_\omega}$$

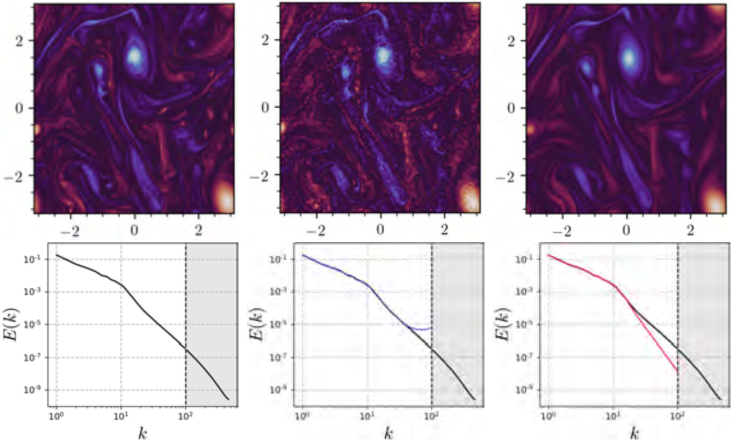


Example of reduced vorticity and SGS term.

Subgrid-scale challenges in QG dynamics

Potential difficulties:

- Accumulation of small-scale energy: numerical instabilities.
- Incorrect representation of the unresolved dynamics.



Difficulties in SGS modeling for two-dimensional turbulent systems.

State of the art: historical

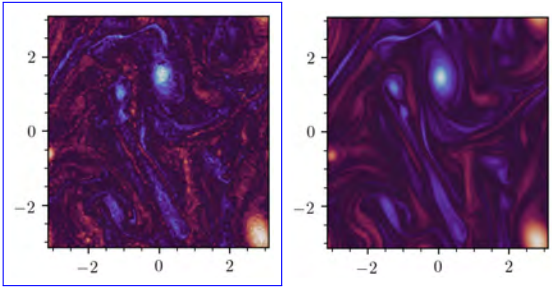
"Historical" – or physical turbulence models (*Sagaut, 2006*):

- Mathematical developments (*Clark et al., 1979*): **Structural**.

	Structural
Stability	-
Forward	+
Backward	+

Potential difficulties:

- Accumulation of small-scale energy: numerical instabilities.
- Incorrect representation of the unresolved dynamics.



Difficulties in SGS modeling for two-dimensional turbulent systems.

State of the art: historical

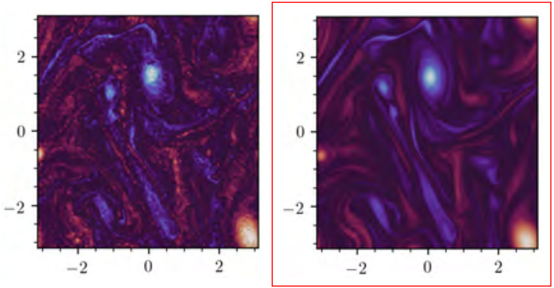
"Historical" – or physical turbulence models (*Sagaut, 2006*):

- Mathematical developments (*Clark et al., 1979*): **Structural**.
- First principles (*Smagorinsky, 1963, Leith, 1996*): **Functional**.

	Structural	**Functional**
Stability	-	+
Forward	+	-
Backward	+	-

Potential difficulties:

- Accumulation of small-scale energy: numerical instabilities.
- Incorrect representation of the unresolved dynamics.



Difficulties in SGS modeling for two-dimensional turbulent systems.

State of the art: machine learning

Current models:

- Exclusive on stability **and** correct transfers.
- Machine learning as an alternative (*Brunton et al., 2020*).

Solving a problem from data:

- Inputs $\bar{\mathbf{y}}$.
- Output τ.
- Model $\mathcal{M} : \bar{\mathbf{y}} \rightarrow \tau$.
- "Static".

Sub-grid modelling for two-dimensional turbulence using neural networks

R. Maulik[1], O. San[1]†, A. Rasheed[2], P. Vedula[3]

[1]School of Mechanical & Aerospace Engineering, Oklahoma State University, Stillwater, OK 74078, USA

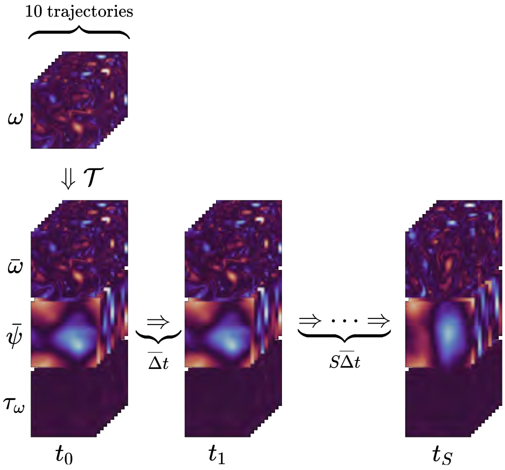
[2]CSE Group, Applied Mathematics and Cybernetics, SINTEF Digital, N-7465 Trondheim, Norway

[3]School of Aerospace & Mechanical Engineering, The University of Oklahoma Norman, OK 73019, USA

Initial experiments on two-dimensional turbulence (*Maulik et al., 2019*).

	Structural	Functional	ML
Stability	-	+	-
Forward	+	-	++
Backward	+	-	++

Numerical setup



Data generation pipeline.

Numerical solver g:

- Pseudo-spectral (Fourier **doubly periodic**).
- Cutoff filter (wavenumbers truncation).
- DNS 2048^2, reduced 128^2.
- $S = 3000$ samples.

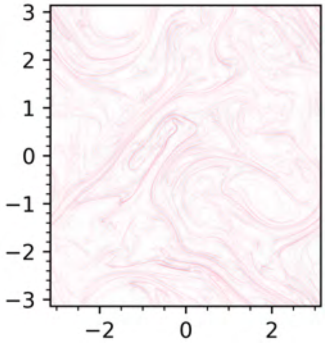
Learning models:

- From literature (not detailed here).
- **Equivalent** NN architectures.

Turbulence evaluation metrics

a priori metrics

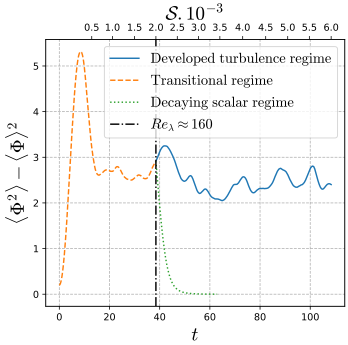
Prediction of the missing term on a **fixed time-step**.



Instantaneous subgrid contribution.

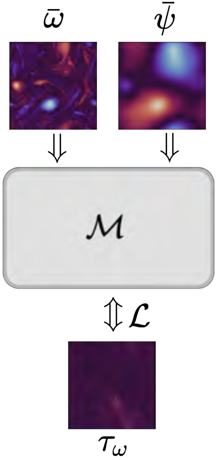
a posteriori metrics

Prediction of the simulation's trajectory over a **temporal horizon**.



Temporal evolution of kinetic energy.

a priori learning



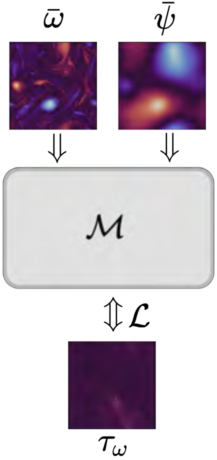
Instantaneous loss computation.

Instantaneous (classical) loss

$$\mathcal{L} := \left\langle \ell(\mathcal{M}(\bar{\psi}, \bar{\omega}), \tau_\omega) \right\rangle_{\mathbf{x}}$$

- Optimize only on the **next** temporal increment $t + \Delta t$.
- **Not perfect**: errors can either lead to stable or unstable predictions.

a priori learning



Instantaneous loss computation.

a priori turbulence "metrics" (Pope, 2000)

$$\ell_{\text{prio}} := \underbrace{(\mathcal{M}(\bar{\psi}, \bar{\omega}) - \tau_\omega)^2}_{\text{Squared error}}$$

$$\vdots$$

$$\ell_{\text{prio}} := \underbrace{\tau_\omega(\log \tau_\omega - \mathcal{M}(\bar{\psi}, \bar{\omega}))}_{\text{KL divergence}}$$

- "Optimal" in *a priori* evaluations.
- Improved (non-interpretable) structural model.

a posteriori learning

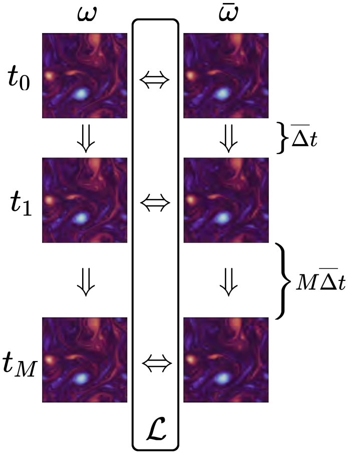
a posteriori loss

$$\mathcal{L} := \langle \ell(\bar{\mathbf{y}}_{\text{pred}}(t), \mathbf{y}(t)) \rangle_{\mathbf{x},\mathbf{t}}$$

$$\bar{\mathbf{y}}_{\text{pred}} \equiv \{\bar{\omega}_{\text{pred}}, \bar{\psi}_{\text{pred}}, \mathcal{M}\}$$

$$\mathbf{y} \equiv \{\omega, \psi, \tau_\omega\}$$

- **Temporal** component in loss function.
- Required to form a **continuous** trajectory.
- Allows for a larger class of evaluation metrics (encompass *a priori* if $|\mathbf{t}| = 1$ and loss only uses τ_ω).



Temporal loss computation.

a posteriori learning

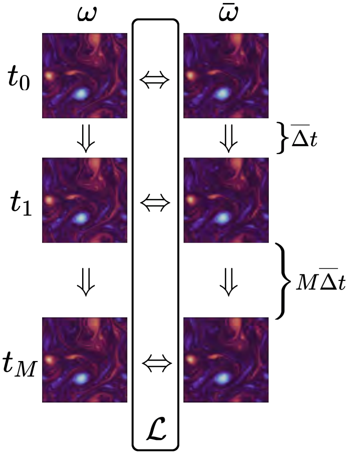
a posteriori turbulence "metrics" (Pope, 2000)

$$\ell_{\text{post}} := \underbrace{E_{\text{pred}}(k) - E(k)}_{\text{Energy spectrum}} : \text{ statistical}$$

$$\vdots$$

$$\ell_{\text{post}} := \underbrace{(\bar{\omega}_{\text{pred}}(t) - \mathcal{T}(\omega(t)))^2}_{\text{Vorticity squared error}} : \text{ local}$$

- "Optimal" in *a posteriori* evaluations.
- Depends on the temporal horizon t (limited, here $M = 25$).

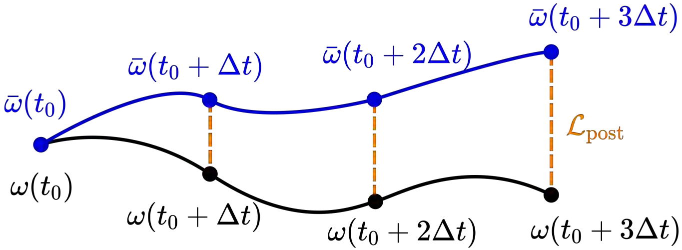


Temporal loss computation.

a posteriori learning: in practice

Optimizing for future quantities:

- Same (resolved) initial conditions.
- Perform temporal integrations **during training** (M discrete timesteps).
- Target fields can be pre-computed.

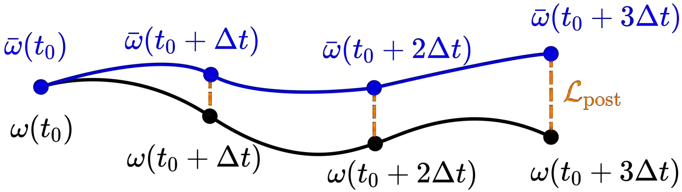


Visual sketch of an *a posteriori* training on one trajectory.

a posteriori learning: in practice

Optimizing for future quantities:

- Same (resolved) initial conditions.
- Perform temporal integrations **during training** (M discrete timesteps).
- Target fields can be pre-computed.

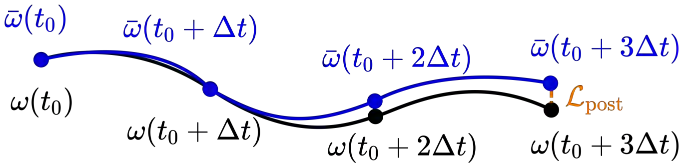


Visual sketch of an *a posteriori* training on one trajectory.

a posteriori learning: in practice

Optimizing for future quantities:

- Same (resolved) initial conditions.
- Perform temporal integrations **during training** (M discrete timesteps).
- Target fields can be pre-computed.



Visual sketch of an *a posteriori* training on one trajectory.

Technical bits

Gradient-based mathematical optimization:

for $\mathcal{M}(\mathbf{y} \mid \theta) : \arg\min_{\theta} \mathcal{L}$ involves $\theta_{n+1} = \theta_n - \gamma \nabla_\theta \mathcal{L}$

a priori loss gradient:

$$
\begin{aligned}
&\nabla_\theta \ell_{\text{prio}}(\mathcal{M}, \tau_\omega) \\
&\quad = \frac{\partial \ell_{\text{prio}}}{\partial \tau_\omega} \frac{\partial \tau_\omega}{\partial \theta} + \frac{\partial \ell_{\text{prio}}}{\partial \mathcal{M}} \frac{\partial \mathcal{M}}{\partial \theta} \\
&\quad = \frac{\partial \ell_{\text{prio}}}{\partial \mathcal{M}} \frac{\partial \mathcal{M}}{\partial \theta}
\end{aligned}
$$

a posteriori loss gradient:

$$
\begin{aligned}
&\nabla_\theta \ell_{\text{post}}(\bar{\mathbf{y}}_{\text{pred}}(t), \mathbf{y}(t)) \\
&\quad = \frac{\partial \ell_{\text{post}}}{\partial \mathbf{y}} \frac{\partial \mathbf{y}}{\partial \theta} + \frac{\partial \ell_{\text{post}}}{\partial \bar{\mathbf{y}}_{\text{pred}}} \frac{\partial \bar{\mathbf{y}}_{\text{pred}}}{\partial \theta} \\
&\quad = \frac{\partial \ell_{\text{post}}}{\partial \bar{\mathbf{y}}_{\text{pred}}} \left(\int_{t_0}^{t} \frac{\partial g}{\partial \theta} + \frac{\partial \mathcal{M}}{\partial \theta} \, \mathrm{d}t' \right)
\end{aligned}
$$

a priori vs *a posteriori*: losses

Technical bits

Gradient-based mathematical optimization:

for $\mathcal{M}(\mathbf{y}\,|\,\theta)$: $\arg\min_{\theta} \mathcal{L}$ involves $\theta_{n+1} = \theta_n - \gamma \nabla_\theta \mathcal{L}$

a priori loss gradient:

$$
\begin{aligned}
\nabla_\theta \ell_{\text{prio}}(\mathcal{M}, \tau_\omega) \\
&= \frac{\partial \ell_{\text{prio}}}{\partial \tau_\omega}\frac{\partial \tau_\omega}{\partial \theta} + \frac{\partial \ell_{\text{prio}}}{\partial \mathcal{M}}\frac{\partial \mathcal{M}}{\partial \theta} \\
&= \frac{\partial \ell_{\text{prio}}}{\partial \mathcal{M}}\frac{\partial \mathcal{M}}{\partial \theta}
\end{aligned}
$$

a posteriori loss gradient:

$$
\begin{aligned}
\nabla_\theta \ell_{\text{post}}(\bar{\mathbf{y}}_{\text{pred}}(t), \mathbf{y}(t)) \\
&= \frac{\partial \ell_{\text{post}}}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \theta} + \frac{\partial \ell_{\text{post}}}{\partial \bar{\mathbf{y}}_{\text{pred}}}\frac{\partial \bar{\mathbf{y}}_{\text{pred}}}{\partial \theta} \\
&= \frac{\partial \ell_{\text{post}}}{\partial \bar{\mathbf{y}}_{\text{pred}}}\left(\int_{t_0}^{t} \underbrace{\frac{\partial g}{\partial \theta}}_{\text{Not available}} + \frac{\partial \mathcal{M}}{\partial \theta}\, \mathrm{d}t' \right)
\end{aligned}
$$

a priori vs *a posteriori*: losses

a posteriori learning: implementation

Gradient of the solver w.r.t. model parameters:

- Estimates using numerical derivatives.
- Manually implement adjoint.
- Automatic generation tools.
- **Implementation using auto-differentiation languages or libraries.**
- **Deep Differentiable Emulators** (*Frezat et al., 2024, Nonnenmacher and Greenberg, 2021, Hatfield et al., 2021*)

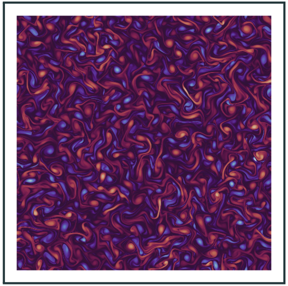
Gradient-free methods: not explored but active field.

a posteriori loss gradient:

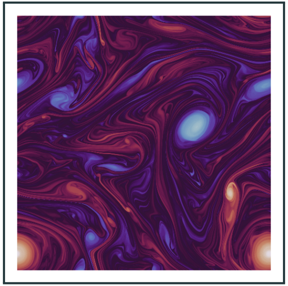
$$\frac{\partial \ell_{\text{post}}}{\partial \bar{\mathbf{y}}_{\text{pred}}} \left(\int_{t_0}^{t} \underbrace{\frac{\partial g}{\partial \theta}}_{\text{Not available}} + \frac{\partial \mathcal{M}}{\partial \theta} \, \mathrm{d}t' \right)$$

Differentiable programming libraries in Julia and Python.

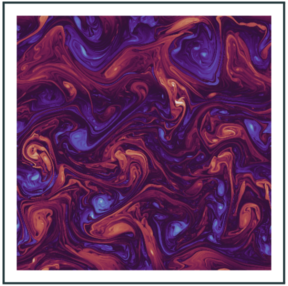
Numerical experiments



Decaying turbulence (*McWilliams, 1984*)

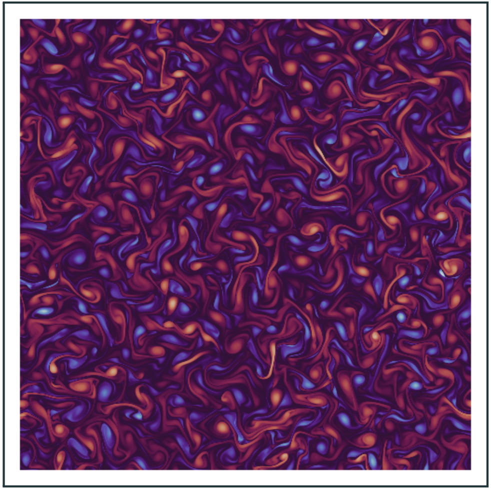


Forced turbulence (*Graham et al., 2013*)

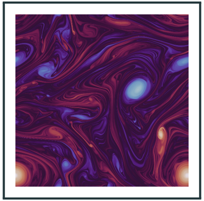


Beta-plane on topography (*Thompson, 2009*)

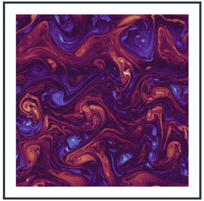
Numerical experiments: decaying turbulence



Decaying turbulence (*McWilliams, 1984*)



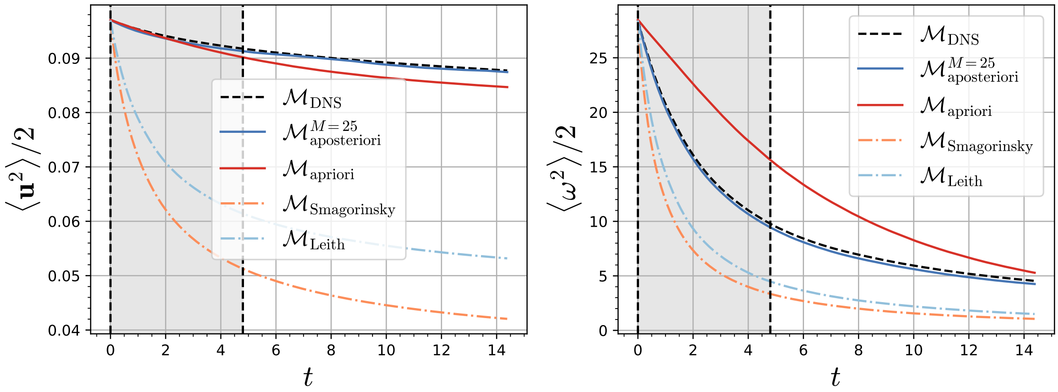
Forced turbulence
(*Graham et al., 2013*)



Beta-plane on topography
(*Thompson, 2009*)

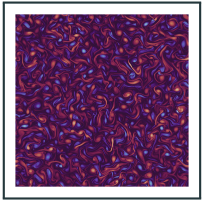
Numerical experiments: decaying turbulence

Energy (left) and enstrophy (right) in decaying turbulence.

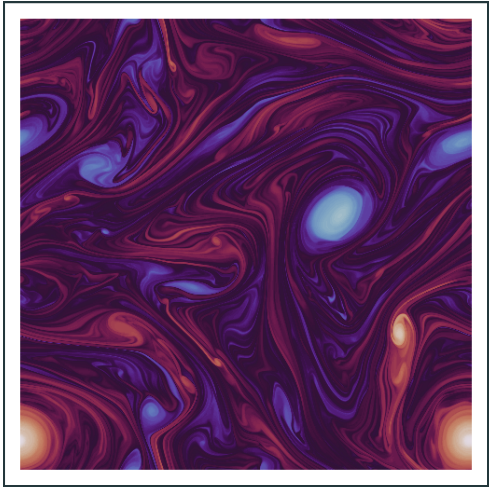


Unsteady generalization 3 times larger than training horizon.

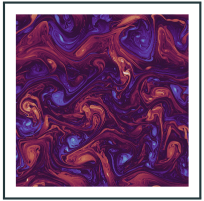
Numerical experiments: forced turbulence



Decaying turbulence (*McWilliams, 1984*)



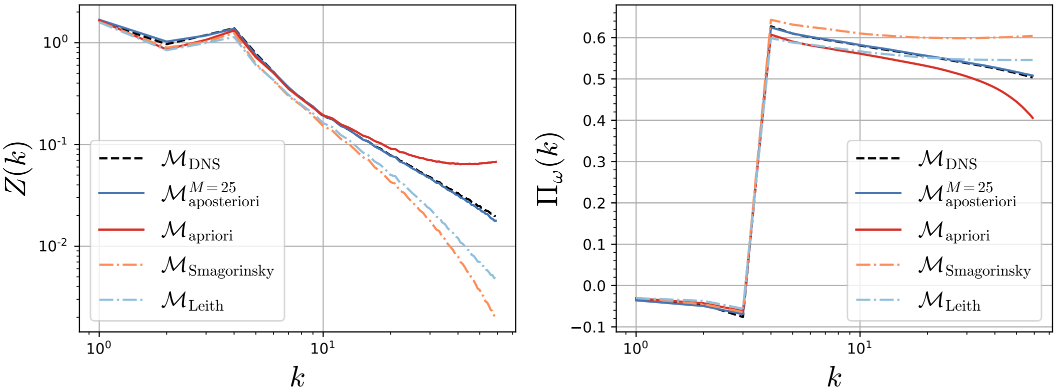
Forced turbulence (*Graham et al., 2013*)



Beta-plane on topography (*Thompson, 2009*)

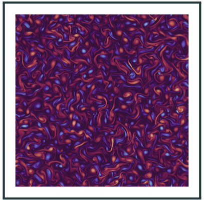
Numerical experiments: forced turbulence

Enstrophy spectrum (left) and fluxes (right) in forced turbulence.

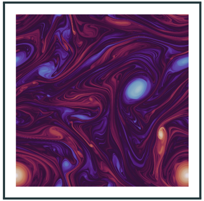


Statistical quantities matching DNS in long-term simulations (18k iterations).

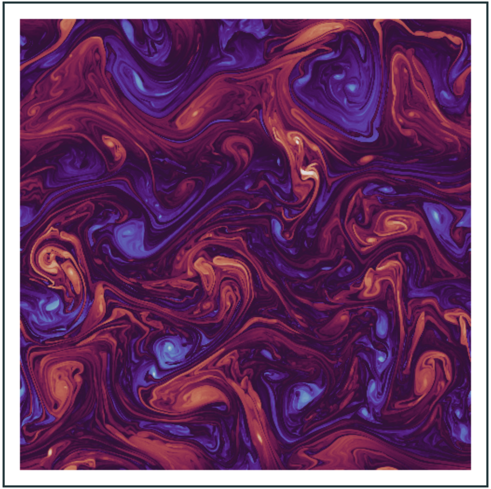
Numerical experiments: beta-plane on topography



Decaying turbulence (*McWilliams, 1984*)

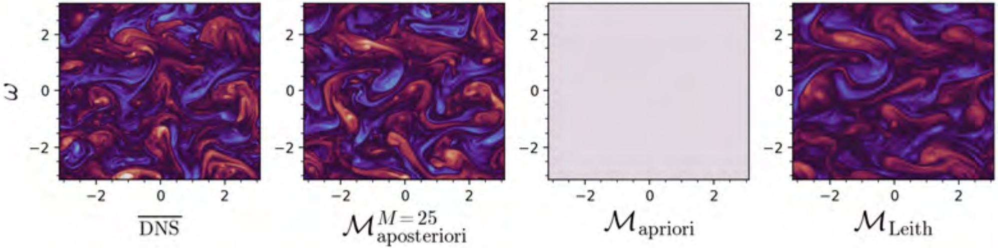


Forced turbulence (*Graham et al., 2013*)



Beta-plane on topography (*Thompson, 2009*)

Numerical experiments: beta-plane on topography



Beta-plane on topography (*Thompson, 2009*)

Conclusion

Results:

- Improved **long-term** stability with **small-term** training.
- Flexibility of the loss function (not explored).
- Higher performance for similar complexity.

Potential limitations:

- Generalization capabilities (w.r.t. configuration).
- Training time overhead complexity (technical).
- Relying on solver gradient availability (applicability).

State-based learning in the annulus

1. Designing **learning strategies** for stable simulations

2. **State-based** learning in the annulus

3. Beyond spectral: neural operators

4. Conclusion

A spectral case for planetary interiors

Example: rotating spherical QG convection

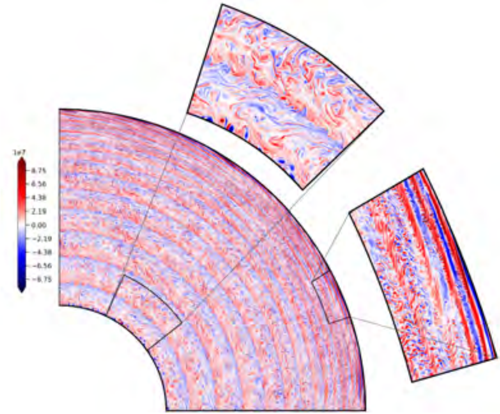
$$\frac{\partial \omega}{\partial t} + \nabla \cdot (\mathbf{u}\omega) = \Delta\omega + \frac{2}{E}\beta u_s - \frac{Ra}{Pr}\frac{1}{s_o}\frac{\partial \theta}{\partial \phi} - \Upsilon\omega$$

$$\frac{\partial \overline{u_\phi}}{\partial t} + \overline{u_s\omega} = \Delta\overline{u_\phi} - \frac{\overline{u_\phi}}{s^2} - \Upsilon\overline{u_\phi}$$

$$\frac{\partial T}{\partial t} + \nabla \cdot (\mathbf{u}T) = \frac{1}{Pr}\Delta T$$

Potential difficulties:

- **Coupled correction terms to learn (2 + 1 axisymmetric)**.
- Grid inhomogeneity.
- Presence of boundaries.



Example of vorticity field from the code `pizza` (credits *T. Gastine*).

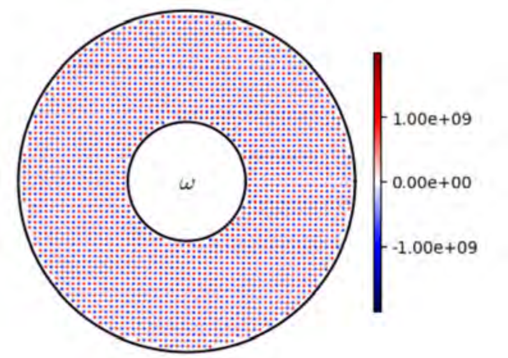
A spectral case for planetary interiors: almost removing coupled terms

Example: forced spherical QG

$$\frac{\partial \omega}{\partial t} + \nabla \cdot (\mathbf{u}\omega) = \Delta \omega + \frac{2}{E}\beta u_s + F - \Upsilon \omega$$
$$\frac{\partial \overline{u_\phi}}{\partial t} + \overline{u_s \omega} = \Delta \overline{u_\phi} - \frac{\overline{u_\phi}}{s^2} - \Upsilon \overline{u_\phi}$$

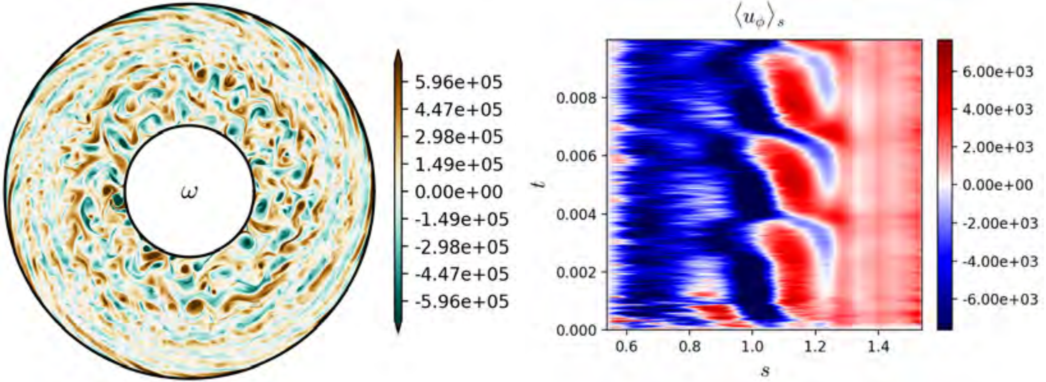
Potential difficulties:

- ~~Coupled correction terms to learn (2 + 1 axisymmetric)~~.
- Grid inhomogeneity.
- Presence of boundaries.



Vorticity "pumps" forcing pattern F (*Lemasquerier et al., 2023*).

A spectral case for planetary interiors: almost removing coupled terms



Example of vorticity field (left) and radially averaged azimutal velocity with varying β (spherical container) with $E = 3 \times 10^{-7}$.

A spectral case for planetary interiors: inhomogeneous filter commutation

Example: forced spherical QG

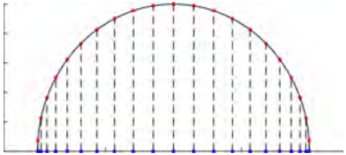
$$\frac{\partial \omega}{\partial t} + \nabla \cdot (\mathbf{u}\omega) = \Delta\omega + \frac{2}{E}\beta u_s + F - \Upsilon\omega$$
$$\frac{\partial \overline{u_\phi}}{\partial t} + \overline{u_s\omega} = \Delta\overline{u_\phi} - \frac{\overline{u_\phi}}{s^2} - \Upsilon\overline{u_\phi}$$

Potential difficulties:

- ~~Coupled correction terms to learn (2 + 1 axisymmetric)~~.
- **Grid inhomogeneity.**
- Presence of boundaries.

Spectral method:

- Discretization for azimutal direction ϕ with **Fourier (periodic) basis**.
- Discretization for radial direction s with **Chebyshev polynomials**.



GL nodes: equispaced semi-circle projected on the line.

A spectral case for planetary interiors: inhomogeneous filter commutation

Example: forced spherical QG

$$\frac{\partial \omega}{\partial t} + \nabla \cdot (\mathbf{u}\omega) = \Delta\omega + \frac{2}{E}\beta u_s + F - \Upsilon\omega$$
$$\frac{\partial \overline{u_\phi}}{\partial t} + \overline{u_s\omega} = \Delta\overline{u_\phi} - \frac{\overline{u_\phi}}{s^2} - \Upsilon\overline{u_\phi}$$

Potential difficulties:

- ~~Coupled correction terms to learn (2 + 1 axisymmetric)~~.
- ~~Grid inhomogeneity.~~ (for *a posteriori* learning)
- Presence of boundaries.

Filtering in polynomial spaces:

- With Fourier, **grid points are equidistant**: filters commute w.r.t. partial derivatives.
- In radial direction, SGS term contains **some commuting error** (*see Yalla et al., 2021*).

$$\frac{\partial \bar{\mathbf{y}}}{\partial t} = f(\bar{\mathbf{y}}) + \underbrace{\tau(\mathbf{y})}_{\neq \mathcal{T}(f(\mathbf{y})) - f(\mathcal{T}(\mathbf{y}))}$$

- Not clear, but **verified empirically**.
- Impossible to construct "exact" objective for a (*a priori*) training.

A spectral case for planetary interiors: boundary-preserving basis

Example: forced spherical QG

$$\frac{\partial \omega}{\partial t} + \nabla \cdot (\mathbf{u}\omega) = \Delta\omega + \frac{2}{E}\beta u_s + F - \Upsilon\omega$$

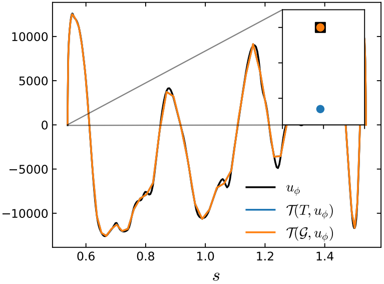
$$\frac{\partial \overline{u_\phi}}{\partial t} + \overline{u_s\omega} = \Delta\overline{u_\phi} - \frac{\overline{u_\phi}}{s^2} - \Upsilon\overline{u_\phi}$$

Potential difficulties:

- ~~Coupled correction terms to learn (2 + 1 axisymmetric)~~.
- ~~Grid inhomogeneity.~~ (for *a posteriori* learning)
- **Presence of boundaries.**

Effect of filtering on boundaries :

$u_s = u_\phi = 0$ or $\psi = \dfrac{\partial \psi}{\partial s} = 0$ at $s = s_i, s_o$,



Radial slice of u_ϕ with different filtering methods.

A spectral case for planetary interiors: boundary-preserving basis

Example: forced spherical QG

$$\frac{\partial \omega}{\partial t} + \nabla \cdot (\mathbf{u}\omega) = \Delta\omega + \frac{2}{E}\beta u_s + F - \Upsilon\omega$$
$$\frac{\partial \overline{u_\phi}}{\partial t} + \overline{u_s \omega} = \Delta\overline{u_\phi} - \frac{\overline{u_\phi}}{s^2} - \Upsilon\overline{u_\phi}$$

Potential difficulties:

- ~~Coupled correction terms to learn (2 + 1 axisymmetric)~~.
- ~~Grid inhomogeneity.~~ (for *a posteriori* learning)
- ~~Presence of boundaries.~~

Truncating in Galerkin basis:

1. Transform Chebyshev coefficients to the corresponding Galerkin basis:

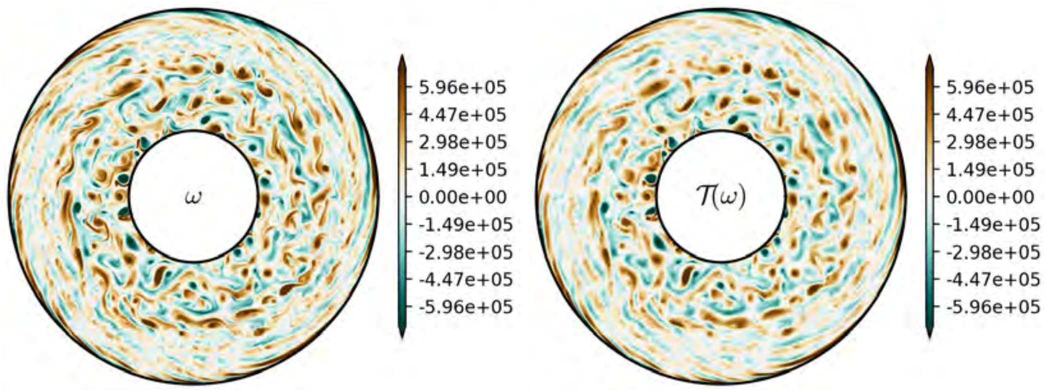
$$\mathcal{G}_n(x) = G^{-1} T_n(x)$$

2. Truncate largest Galerkin coefficients $\mathcal{G}_m(x), m \leq n$.
3. Transform back to Chebyshev coefficients:

$$T_m(x) = G\mathcal{G}_m(x)$$

To avoid non-zero divergence, truncate ψ and recompute u_s, u_ϕ and ω.

A spectral case for planetary interiors: truncated vorticity



Vorticity field from DNS at $N_m = 641, N_r = 321$ (left) and from Galerkin truncation at $N_m = 129, N_r = 65$ (right).

Learning setup

Non-optimal architecture

- 500 samples.
- Temporal horizon N = 25.
- Predictions in spectral space.
- Variant of ConvNext (*Liu et al., 2023*).
- Complex activation function:

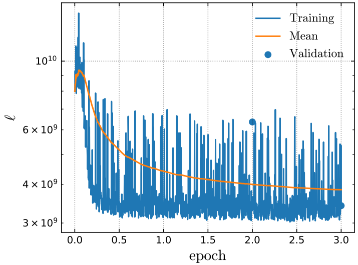
$$\mathrm{modrelu}(z) = \mathrm{relu}(|z|+1)\frac{z}{|z|+\epsilon}$$

- Spherical enstrophy loss:

$$\ell(\omega,\hat{\omega}) = \sum_{m=1}^{N_m}\int_{s_i}^{s_o}\left(|\hat{\omega}_s^m|^2 - |\omega_s^m|^2\right)s\,\mathrm{d}s$$

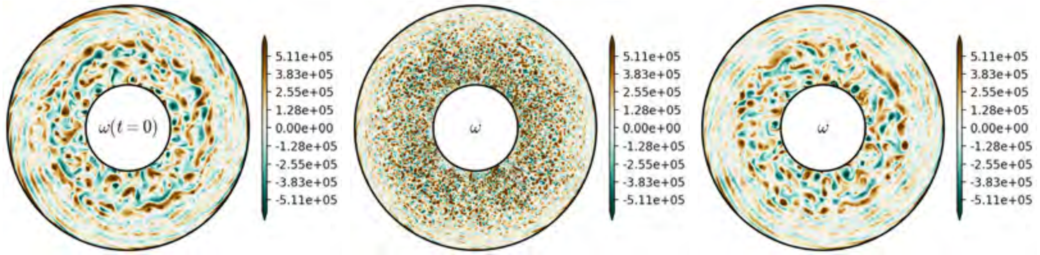
Preliminary results:

- Only 500 samples (20% of the expected full dataset).



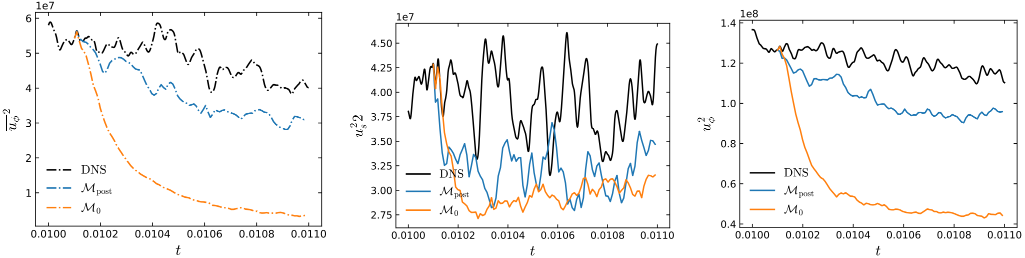
a posteriori loss on limited dataset.

Results



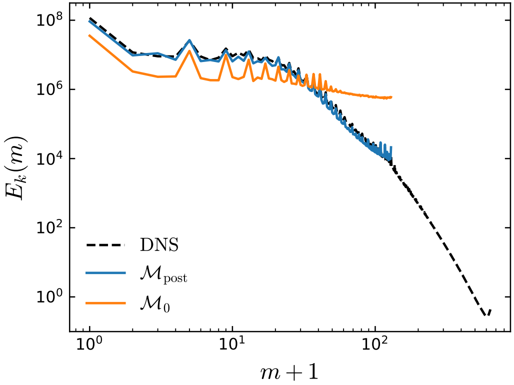
Vorticity field from truncated DNS (left), from simulation truncated resolution without model (middle) and with *a posteriori*-learned model (right) after 25k iterations.

Results



Velocity integrals evolution for 25k iterations.

Results



Time-averaged energy spectrum for 25k iterations.

Takeaway / Next steps

a posteriori learning seems more natural when approaching time-dependent PDE problems.

On the "hybrid" side:

- Evaluate "classical" models: Hyperdiffusion, Smagorinsky (*Matsui and Buffett, 2012*).
- Similar methodology on spherical QG convection (coupled terms, **separate models (?)**).
- Full dynamo system (**spherical harmonics**, 5 coupled terms).

Limiting factor: differentiable banded/sparse linear solvers.

On the "neural" side:

- Explore **neural operators** as an alternative numerical method.

Thanks

Thanks for your attention.