ITW 2024

Session: Modeling and network processing

Learning long-range infrasound propagation using neural operators

Christophe Millet[1,2], Fanny Lehmann[3]

Elodie Noëlé[2]

[1] ENS Paris-Saclay, 91190 Gif-sur-Yvette, France

[2] CEA, DAM, DIF, 91297 Arpajon, France

[3] ETH AI Center, ETH Zürich, Swiss

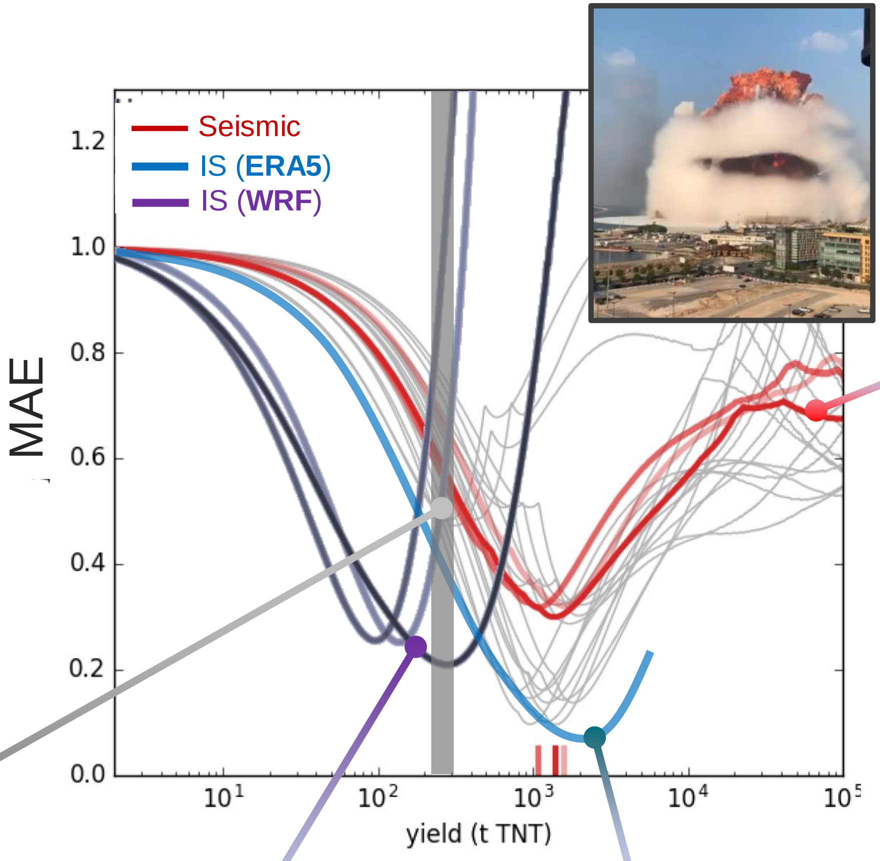
0. Motivation

A result and a challenge

You get what you pay for
Millet *et al.*, ITW2023

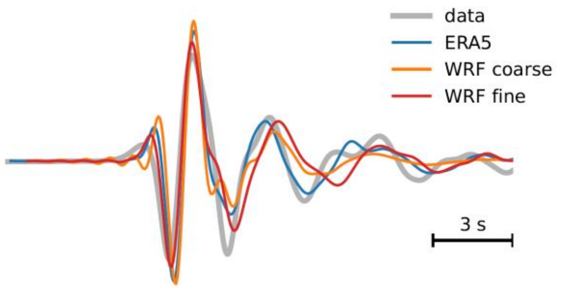
Yield estimates (Beirut, 2020)

- Using **Green**'s functions of the wave equation (SEM3D) + signals.
  - $W_S \sim W_{IS}$ with ERA5.
  - Adding small-scale fluctuations alters W_{IS} by $O(10)$.
- Using a compressible flow solver and videos of **shock dynamics**.

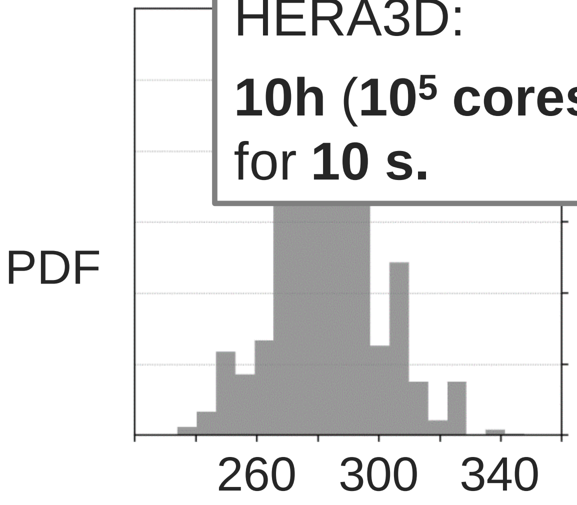


Seismic signals

InfraSound (IS)

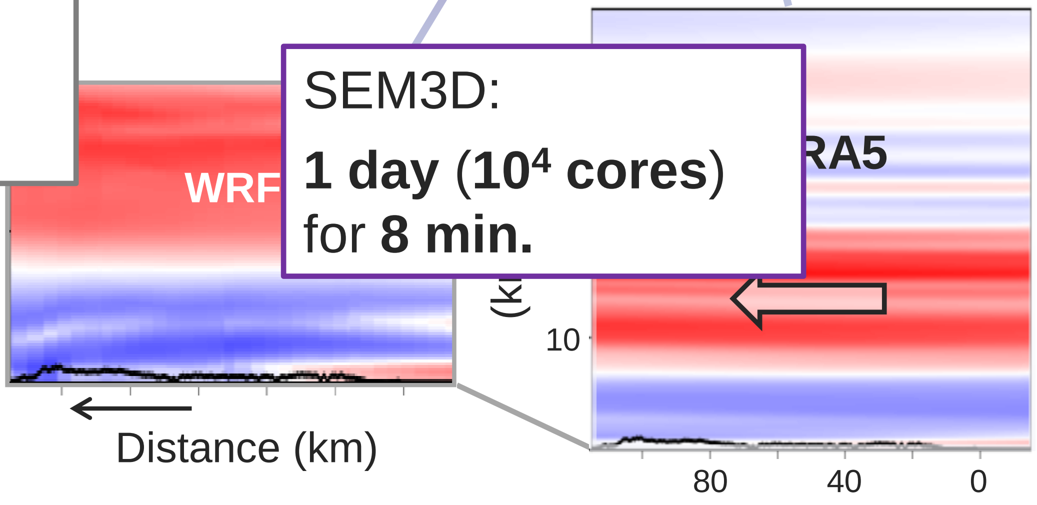


+



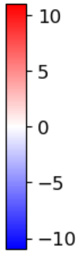
HERA3D:

10h (10^5 cores)
for **10 s.**



SEM3D:

1 day (10^4 cores)
for **8 min.**



Wind (m.s^{-1})

Can data-driven techniques help?

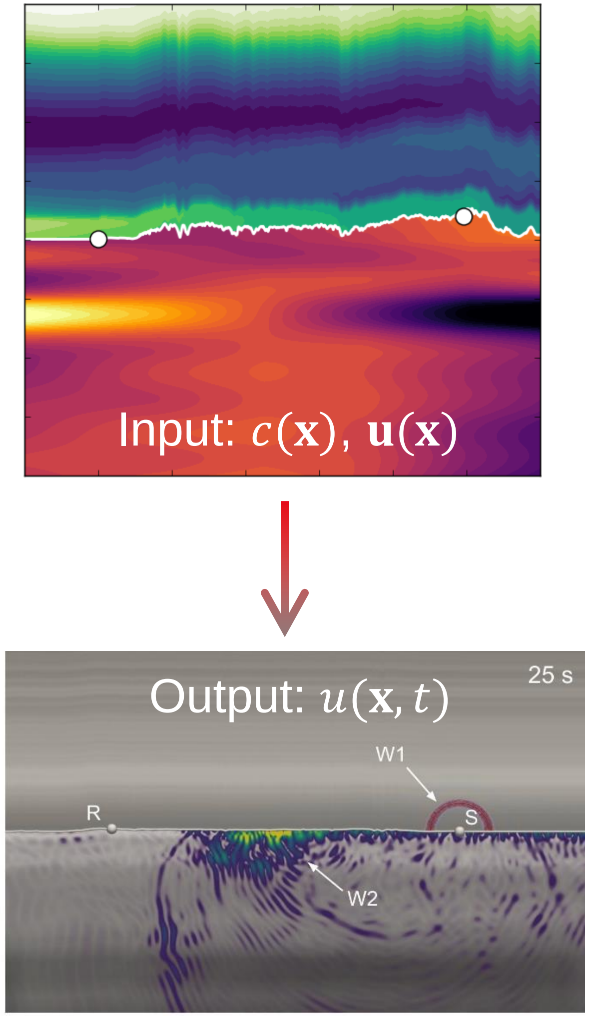
- **Limitations of ML in scientific modeling**
  - Most NNs (CNNs) are not specifically designed for computing a **Green function** or solving PDEs.

CNN filters

  - Databases are often **biased** (small training set size N).
  - NNs are often **over-parameterized** (DOF$\gg N$), making them excellent interpolators, but with limited extrapolation capabilities.
- **Challenges and objectives**
  - Map multi-scale fields to waveforms and/or TLs despite spectral bias, which can hinder accurate capture of fine-scale features:

$$\text{NN: } c_{\text{eff}}(\mathbf{x}) \mapsto u(\mathbf{x}, t)$$

  - Ensure robust extrapolation for out-of-sample conditions, which is essential for real-world applications and unexpected scenarios.



1. Neural Operators

Basics and intuition

Neural Operators

●○○○○ Kernel methods

* Zongyi Li, Kovachki *et al.*, 2021.

- **Intuition (t is ignored for simplicity)**
  - If G is the Green function of a parametric PDE, then:

$$u(\mathbf{x}) = \int G(\mathbf{x},\mathbf{y}) f(\mathbf{y})\, \mathrm{d}\mathbf{y}.$$

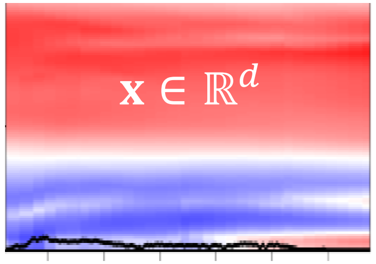
  - G is modelled as a kernel $\kappa_{\boldsymbol{\theta}}$ defined by a NN with parameters $\boldsymbol{\theta}$:

$$G(\mathbf{x},\mathbf{y}) \simeq \kappa_{\boldsymbol{\theta}}\big(\mathbf{x},\mathbf{y},c(\mathbf{x}),c(\mathbf{y})\big).$$

- **Neural Operator (NO)**
  - For any $\mathbf{v}\colon \mathbb{R}^d \to \mathbb{R}^m$, a NO is defined by

$$\mathbf{K}_{\boldsymbol{\theta}}\mathbf{v}(\mathbf{x}) = \int \kappa_{\boldsymbol{\theta}}\big(\mathbf{x},\mathbf{y},c(\mathbf{x}),c(\mathbf{y})\big)\mathbf{v}(\mathbf{y})d\mathbf{y}\,.$$

  - Four variations: Graph NO, multipole GNO, low-rank NO and **Fourier NO***.
  - **FNO** (convolution kernel) $\kappa_{\boldsymbol{\theta}}\big(\mathbf{x},\mathbf{y},c(\mathbf{x}),c(\mathbf{y})\big) = \kappa_{\boldsymbol{\theta}}(\mathbf{x}-\mathbf{y}) \Rightarrow \mathbf{K}_{\boldsymbol{\theta}}\mathbf{v}(\mathbf{x}) = \kappa_{\boldsymbol{\theta}} * \mathbf{v}.$



$\mathcal{F}$

NO

$\mathcal{F}^{-1}$

Waveforms
$u(\mathbf{x}_\perp, t)$

$(\mathbf{x}_\perp, t) \in \mathbb{R}^d$

Neural Operators

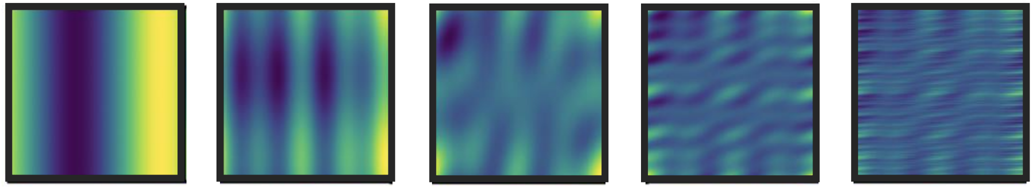
○●○○○ Architecture

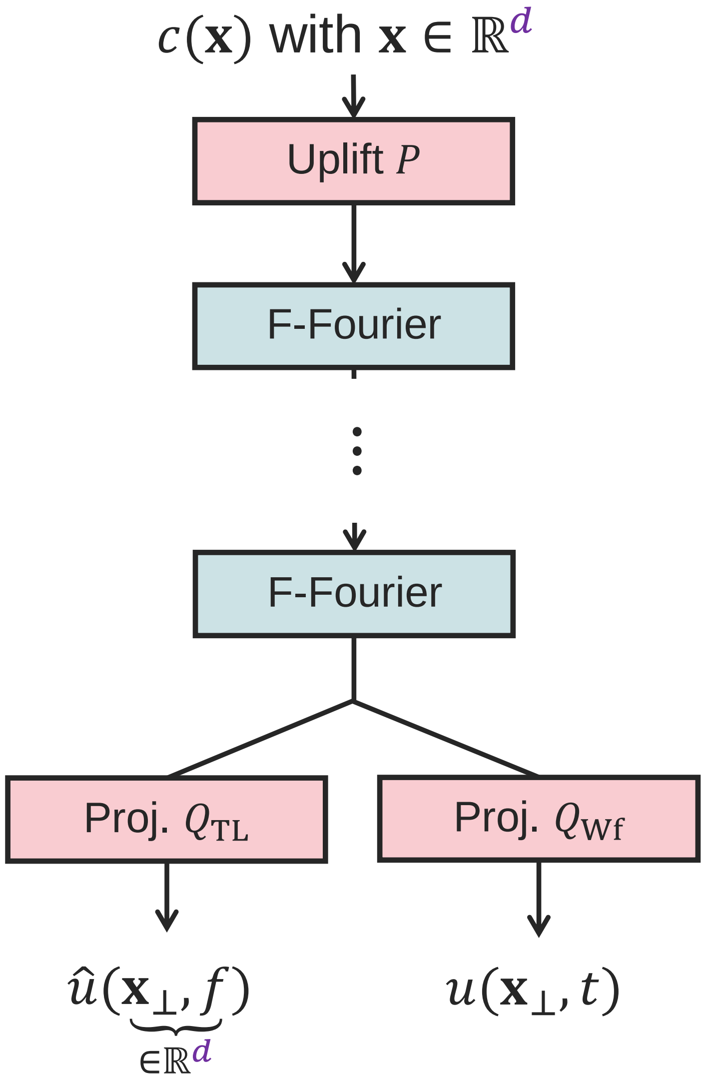
- **Architecture**
  - The mapping is learnt iteratively: $\begin{pmatrix} c(\mathbf{x}) \\ \mathbf{x} \end{pmatrix} \xrightarrow[P]{} \mathbf{v}_0 \xrightarrow[F_1]{} \cdots \xrightarrow[F_L]{} \mathbf{v}_L \xrightarrow[Q]{} u(\mathbf{x})$
  - P is an uplift layer, Q is a projection layer and Fourier layers F_l are defined by: $\mathbf{v}_l = \sigma_l\big([\mathbf{W}^l + \mathbf{K}(c)]\mathbf{v}_{l-1} + \mathbf{b}^l\big)$.
- **Kernel (non-local) integral operators**
  - Assuming K is a convolution kernel, the convolution theorem leads to:

$$\mathbf{K}(c)\mathbf{v} = \mathcal{F}^{-1}\big(\underbrace{\mathcal{F}(\boldsymbol{\kappa})}_{\mathbf{R}}\,\mathcal{F}(\mathbf{v})\big)$$

  - The weights ($\mathbf{R} \in \mathbb{C}^{N\times M}$) are learnt inside each layer.

Fourier filters





Neural Operators

○○●○○ FNO usage

* https://arxiv.org/abs/2402.01868

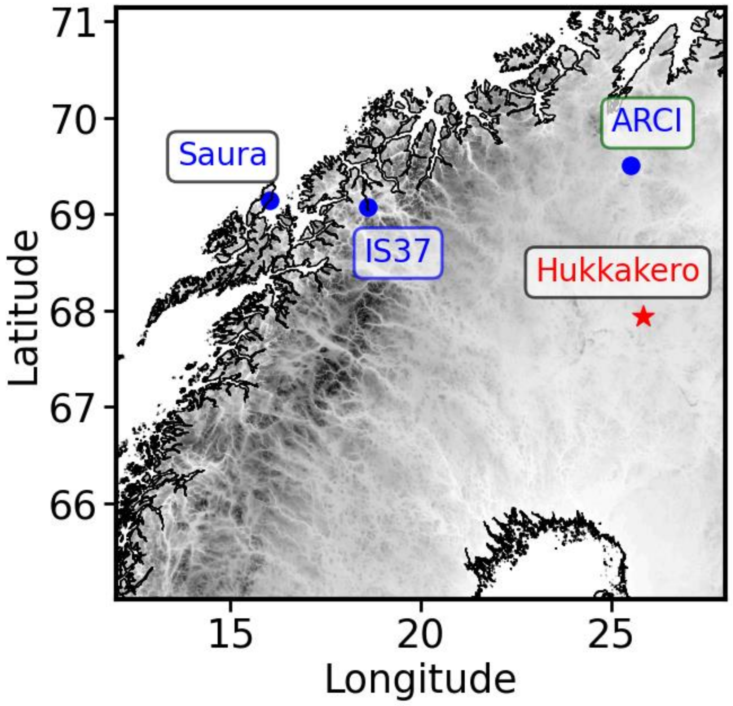
- **Training: leverage limited dataset by combining real observations with synthetic data.**
  - Few real-world observations ($\sim 10^2$) form the foundation. We augment this data with synthetic signals created via simulations.
  - **Data augmentation:** for each event, 10^2 GW fields are generated to produce 10^2 synthetic waveforms

    $\Rightarrow$ input-output pairs $\{c_i, u_i\}_{i=1}^{P}$ with $P \sim 10^4$ (20% validation).
  - **Hybrid optimization** (Rathore, ICML2024*).

- **Datasets for evaluation**

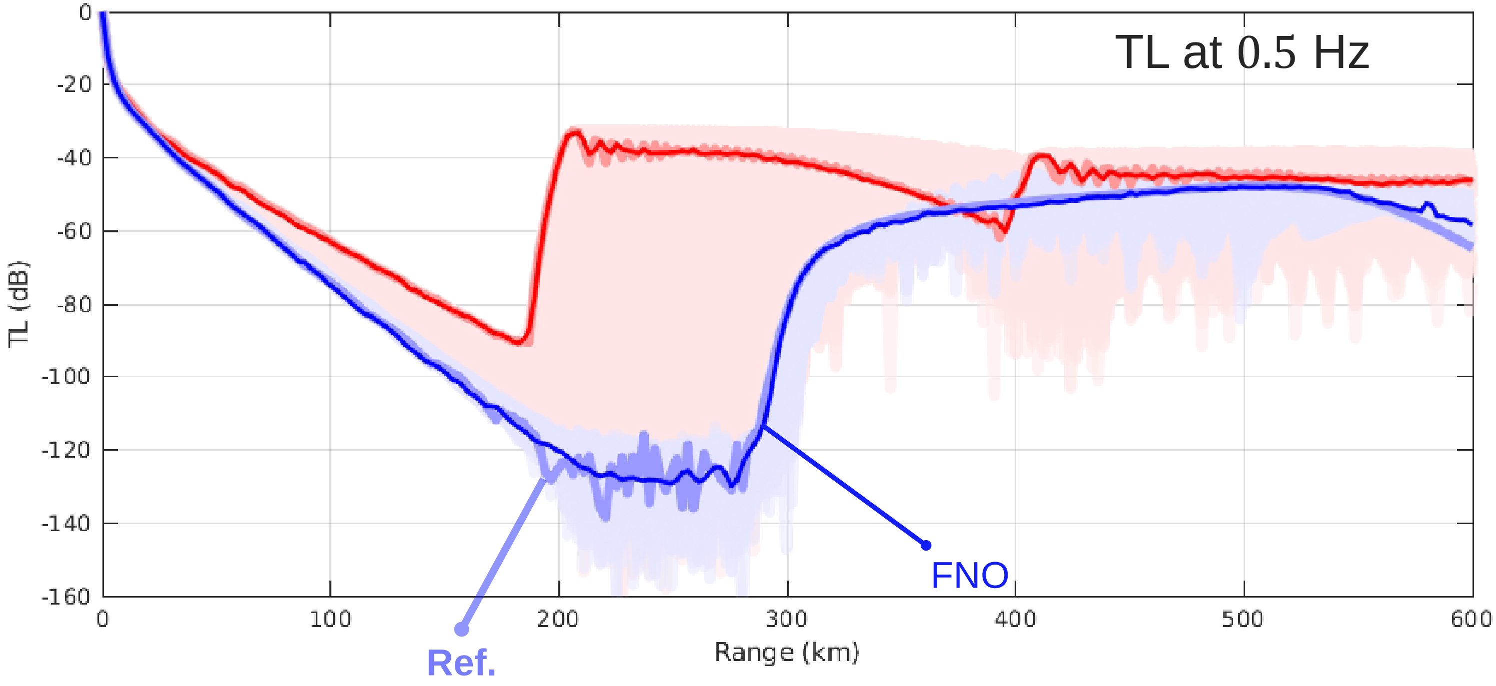
  ❶ **Sanity check**: Idealized conditions for c_i with ducting effects, allowing model performance evaluation under simplified scenarios and optimization of FNO layers (σ, N and M).

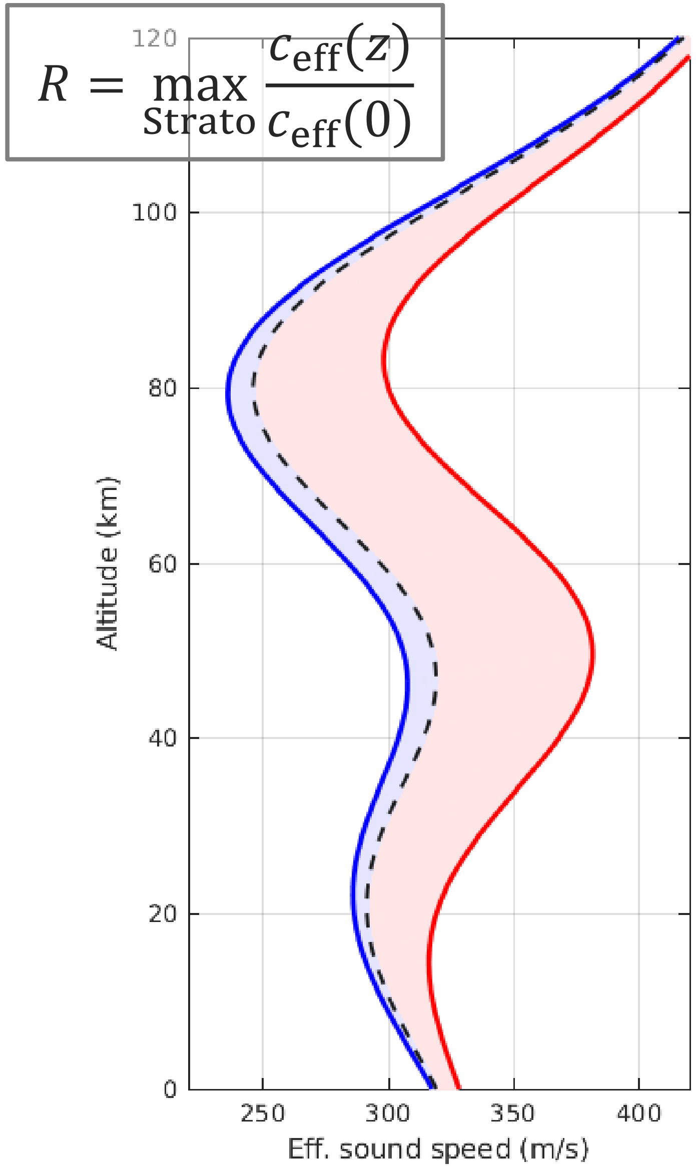
  ❷ **Real-world case:** Dataset of waveforms recorded at IS37 during ammunition explosion campaigns in Hukkakero (2014–2024).



Neural operators

○○○●○ Validation on new profiles (case 1)





- Spatial evolution of TL is well reproduced for various sound speed ratios (from $R < 1$ to $R > 1$).
- Small-scale features are more difficult to predict.
- 2 layers, $N = 192$ neurons and $M = 80$ modes $\Rightarrow$ $\boxed{6.10^6 \text{ parameters.}}$

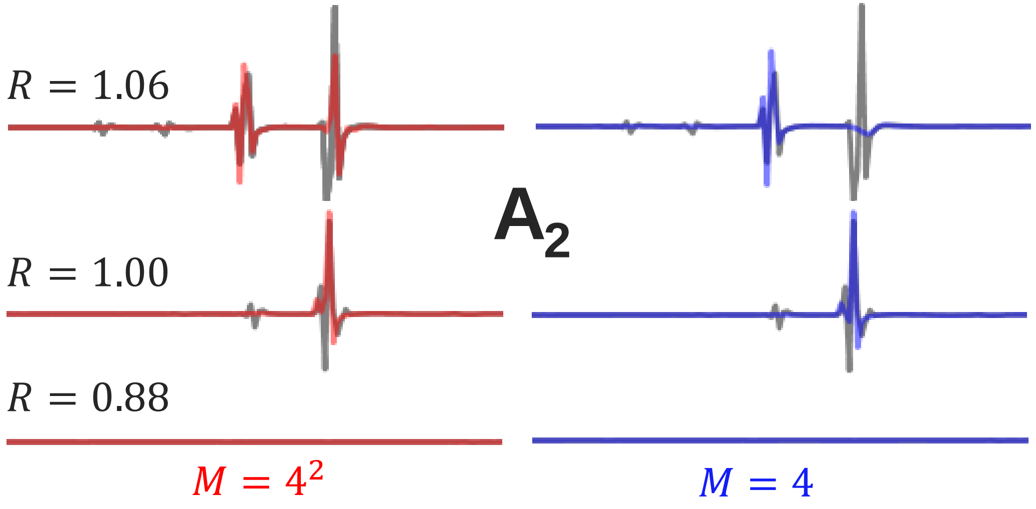
overparameterized

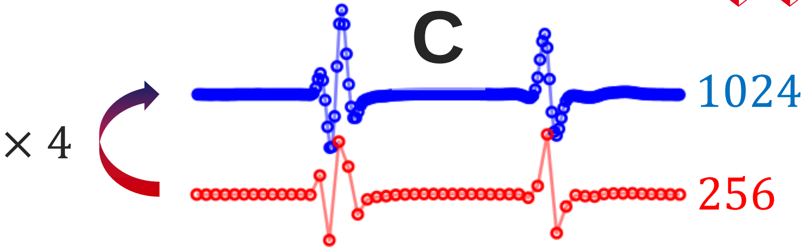
Neural operators

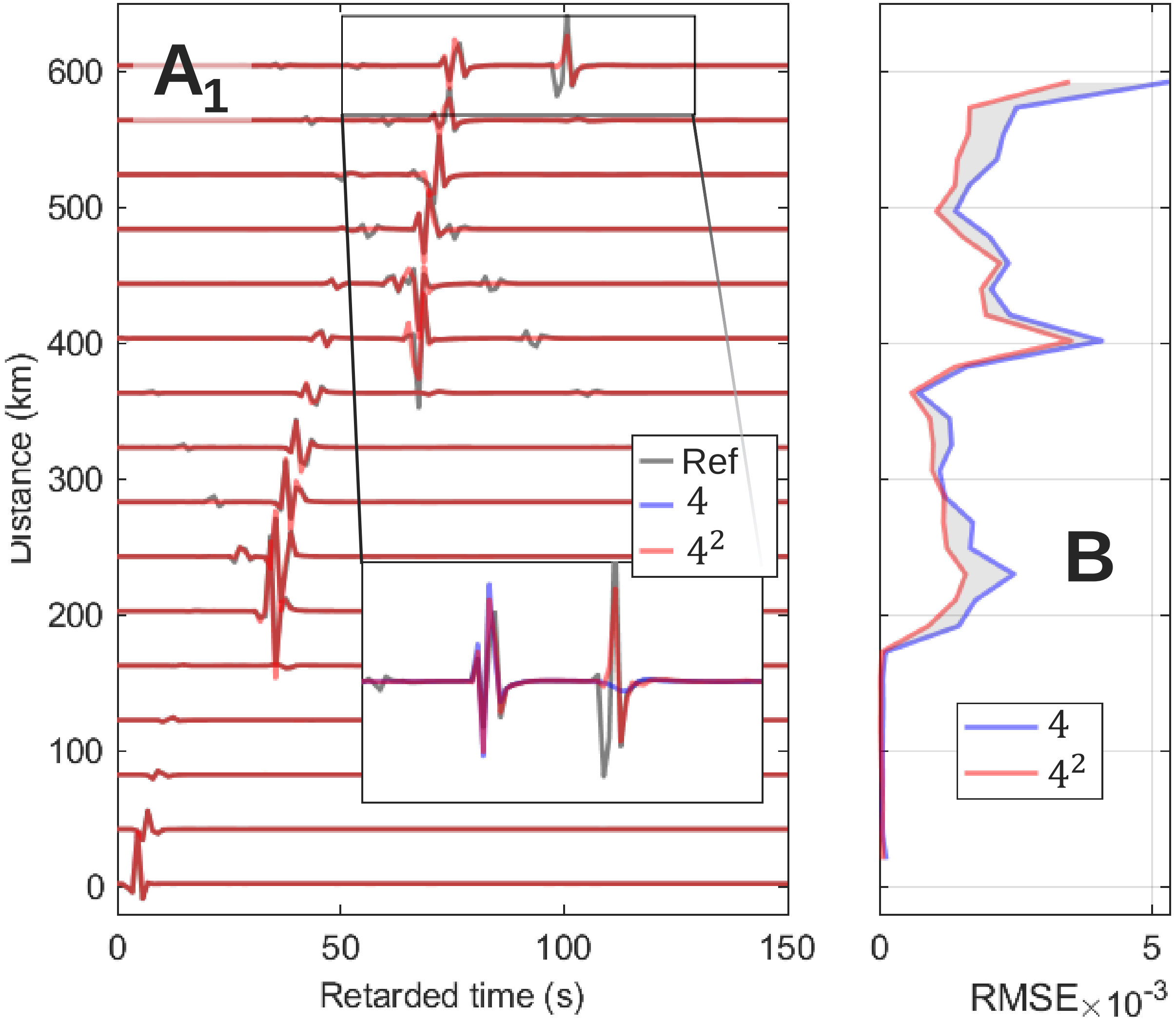
○○○○● On the # of modes

Impact of modes (48 neurons/layer, 4 layers**)**

- Main stratospheric arrivals are reproduced for $R \geq 1$ unless M is too small ⇨ **A_1, A_2.**
- The higher the # of modes, the smaller the RMSE (4 vs 4^2 for $R = 1.06$) ⇨ **B**.
- **Resolution invariance:** (256 vs 1024 for resolution of $c_{\mathrm{eff}}(z)$, with $M = 4^2$) ⇨ **C.**







2. Gravity waves

A random multiwave model

Modeling fluctuations

●○○○ Basics on GWs

Eliassen-Palm flux and GW drag

- **EP flux:**

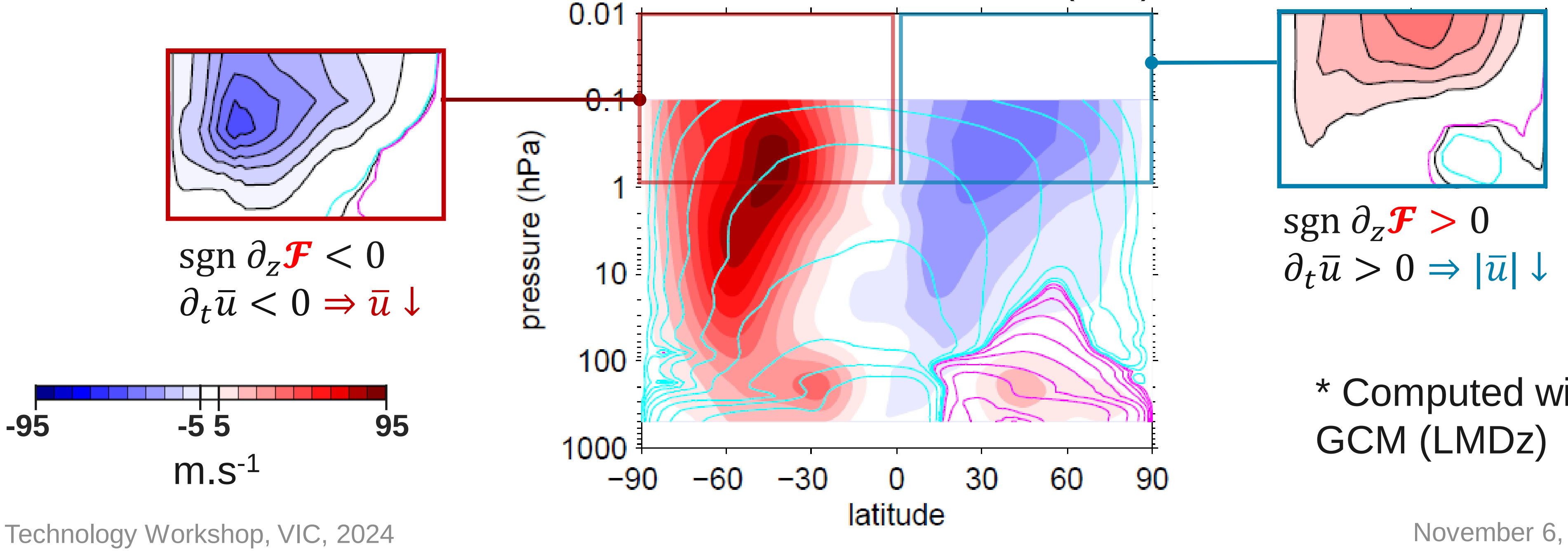
$$\mathbf{F} \propto -\rho_0 \overline{\mathbf{u}'w'} \text{ [Pa].}$$

- **GW drag (GWD):**

$$\rho \partial_t \bar{\mathbf{u}} = \partial_z \mathbf{F} \text{ [m.s}^{-1}\text{.day}^{-1}\text{].}$$

- $\mathbf{F}$ ~ direction of GW propagation.
- $\mathrm{div}(\mathbf{F})$ gives a force (/unit mass) acting on the mean flow.

Impact on the zonal-mean wind



$\mathrm{sgn}\, \partial_z \mathcal{F} < 0$

$\partial_t \bar{u} < 0 \Rightarrow \bar{u} \downarrow$

Zonal-mean GWD*

$\Delta = 4$ m.s^{-1}.d^{-1}

$\mathrm{sgn}\, \partial_z \mathcal{F} > 0$

$\partial_t \bar{u} > 0 \Rightarrow |\bar{u}| \downarrow$

* Computed with a GCM (LMDz)

Modeling fluctuations

○●○○ A multiwave model

* Ribstein, Millet, Lott, JAMES, 2022.

General formalism for GWs

- Use a stochastic Fourier series of individual monochromatic wave packets:

$$\begin{pmatrix}\mathbf{u}'\\ T'\end{pmatrix} = \sum_{j=1}^{J} C_j \begin{pmatrix}\hat{\mathbf{u}}_j(z)\\ \hat{T}_j(z)\end{pmatrix} e^{i(\mathbf{k}_j.\mathbf{x}-\omega_j t)},$$

C_j: intermittency coefficient.

- EP flux evolution is given by*:

$$\mathbf{F}_j(z_{l+1}) = -\frac{\mathbf{k}_j}{k_j}\underbrace{\Theta[\hat{\omega}_j^2(z_l)]}_{1}\min[\underbrace{F_j(z_l)e^{-\mu\Delta z}}_{2},\underbrace{F_j^c(z_{l+1})}_{3}]$$

1 $\mathbf{F}_j = \mathbf{0}$ above critical levels ($\hat{\omega} = \omega - \mathbf{k}_j.\mathbf{u}$).
2 is reduced by diffusion.
3 is limited by that of a saturated wave.

Random variables

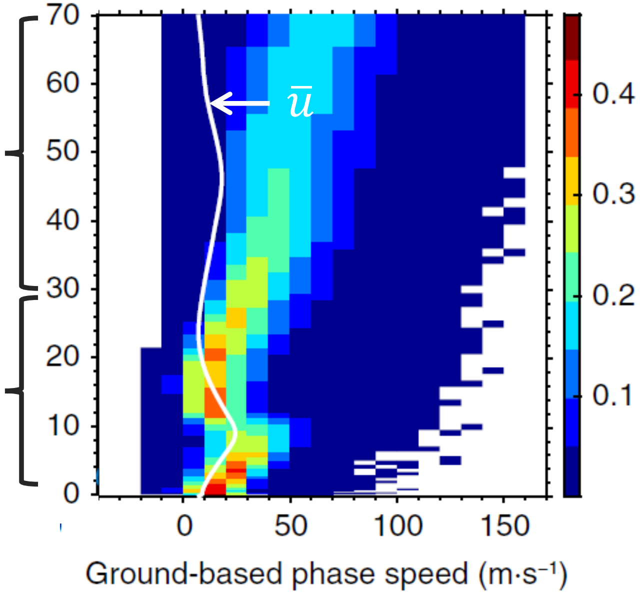
- $\mathbf{k}_j$, $\hat{\omega}_j$ and z_j are random.
  - $\|\mathbf{k}_j\| \sim \mathrm{U}(k_{\min}, k_{\max})$, $k_{\max}$ related to Δx.
  - Direction of $\mathbf{k}_j \sim \mathrm{U}(0,2\pi)$.
  - Phase speed: $c_j \sim \mathrm{N}(0, \sigma_c)$

Ex.: Contribution of $\#j$ to $\sum_i C_i^2 \partial_z \mathbf{F}_i$ (GWD).

Oct., 45-75°N

Amplitude saturation (3)

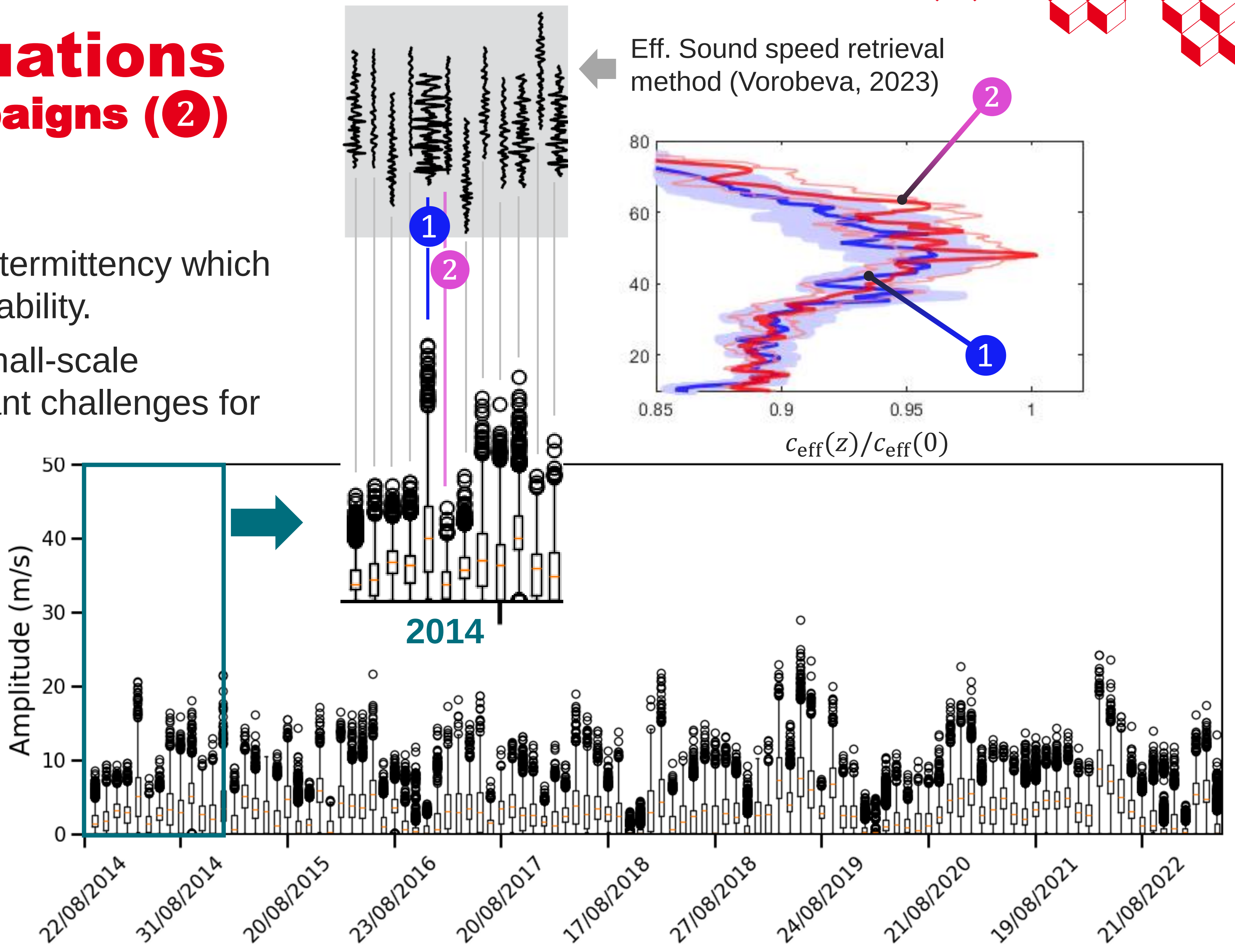
Effect of critical levels (1)



Modeling fluctuations

○○●●○ Hukkakero campaigns (2)

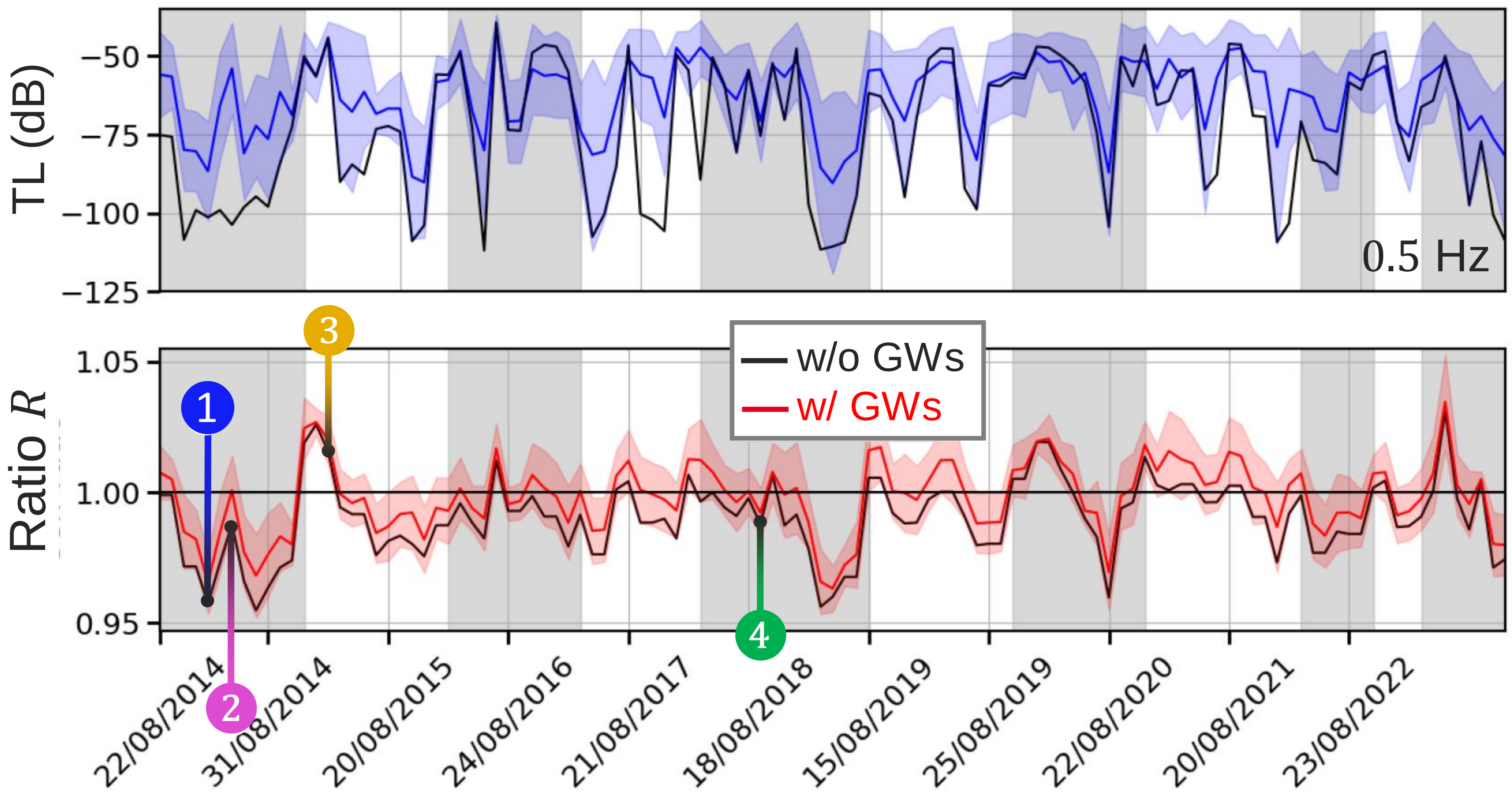
- **Variability & stochasticity**
  - GWs are characterized by intermittency which is subject to **day-to-day** variability.
  - Each GW field introduces small-scale structures that pose significant challenges for neural networks to capture.



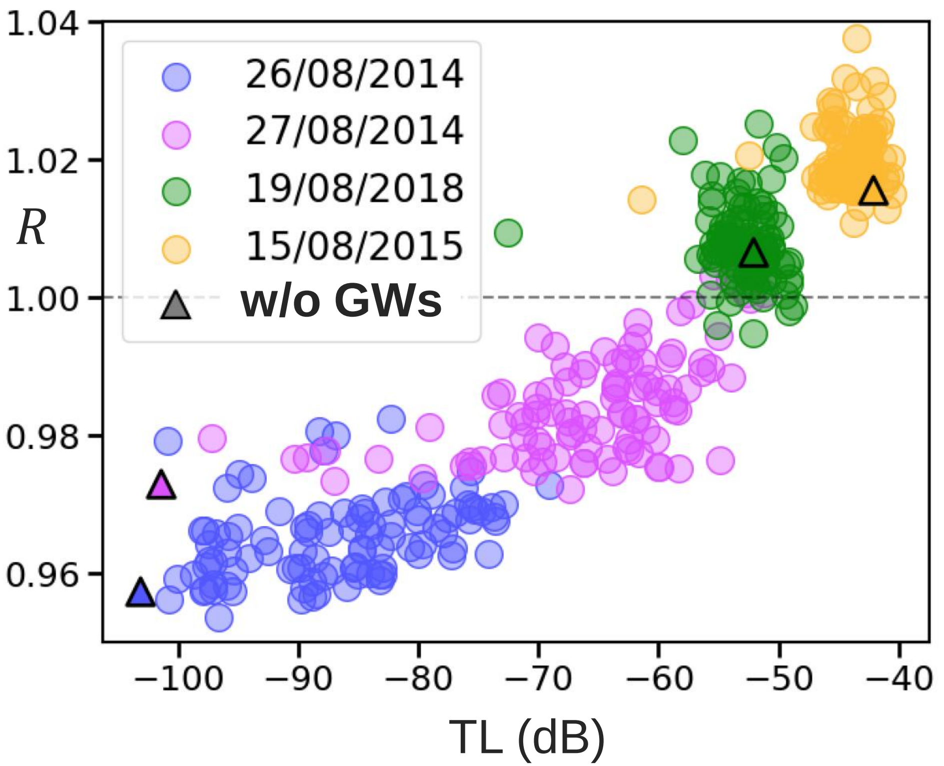
- **Training set**
  - 113 events $\times 10^2$ GW fields.
  - Mean atmospheric specification: ERA5.
  - Fine tuning is essential for obtaining good convergence of the loss (MAE+MSE).

Modeling fluctuations

○○○● Impact of GWs on TLs (➋)



$$R = \max_{\text{Strato}} \frac{c_{\text{eff}}(z)}{c_{\text{eff}}(0)}$$



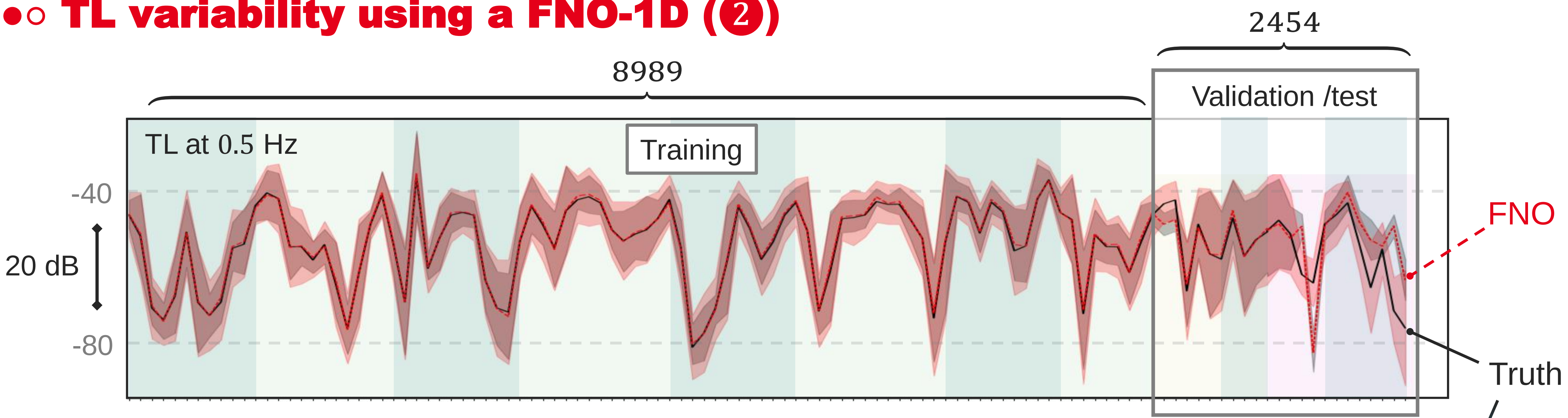
- GWs increase $c_{\text{eff}}(z)/c_{\text{eff}}(0)$ in the stratosphere.
- For $R < 1$, GWs introduce a bias in the TL statistics, causing the mean value to shift away from that obtained without GWs.

3. Preliminary results

TLs and waveforms

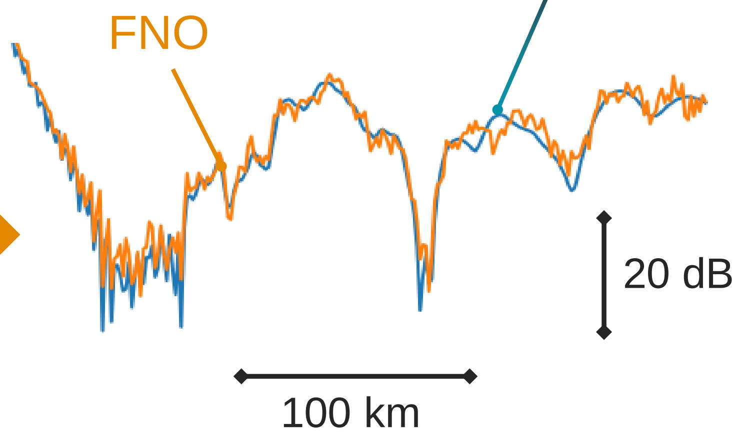
Preliminary results

●○ TL variability using a FNO-1D (❷)



- Dispersion ($\pm\sigma$) due to GWs and day-to-day variability of TL at IS37 are correctly reproduced during training (6 yrs × 10 events/yr).
- Predictions for the last 4 years are associated with errors that increase over time, reaching a few dB, except for specific # of GWs.
- 6 FNO layers with 128 neurons/layer, 64 modes. New frequencies can be added using Q_{TL}.

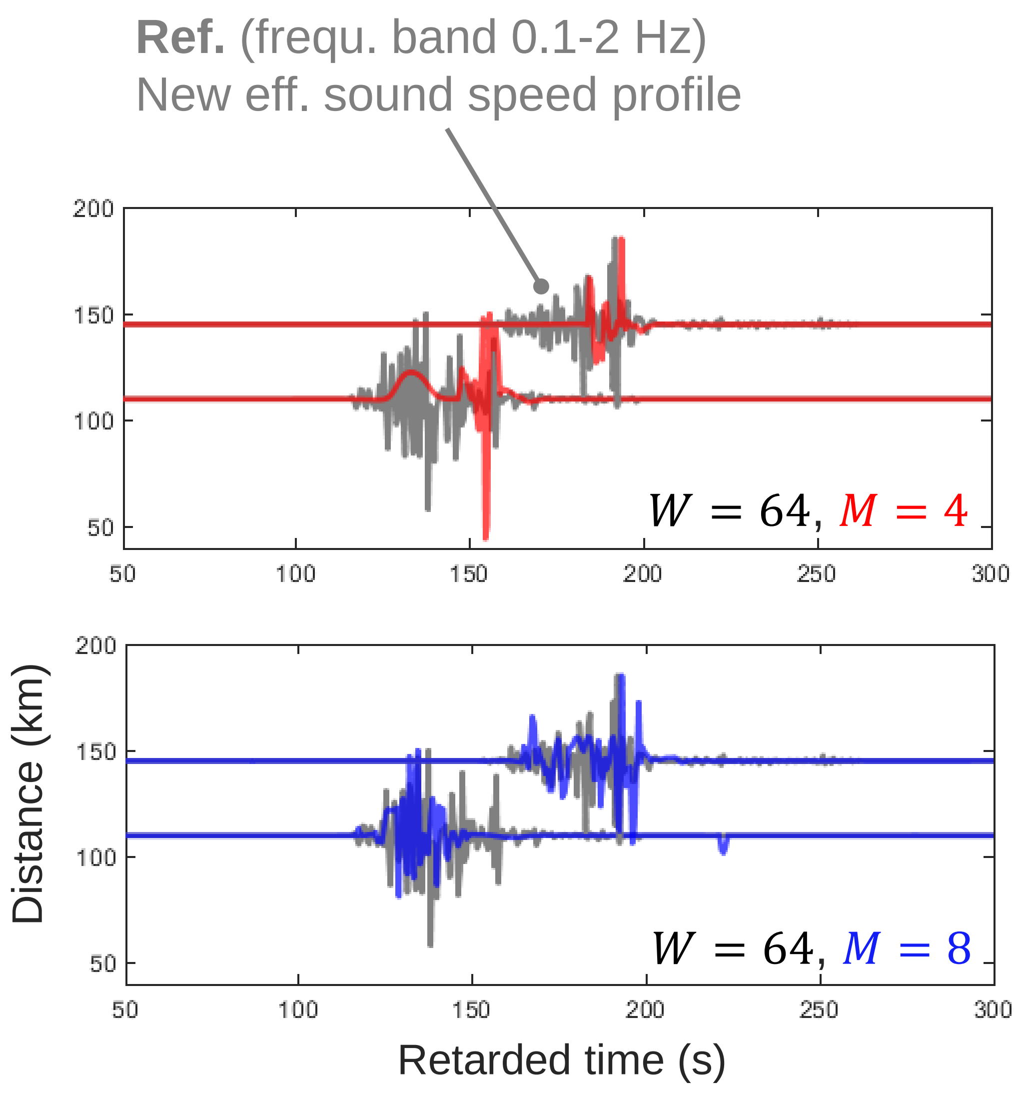
Training is completed in less than 1h.



Preliminary results

○● Waveforms using a FNO-2D (❷)

- **Predictions in the shadow zone**
  - Diffraction in the shadow zone is effectively reproduced overall with moderate N and M (64 and 8, resp. ⇨ $1.8\ 10^6$ dof).
  - **But too much complexity!** The small size of the dataset (10^4) limits efficient learning of GW-IS interactions at the smallest wavelengths.
- **Variability & stochasticity**
  - **Scaling invariance**: to minimize memory space, the training is conducted with signals of size 512, while validation is performed on oversampled signals (1024).
  - **Stability:** by reducing M, the number of parameters can be decreased, allowing the training to be completed in a few hours (training size of 10^4).

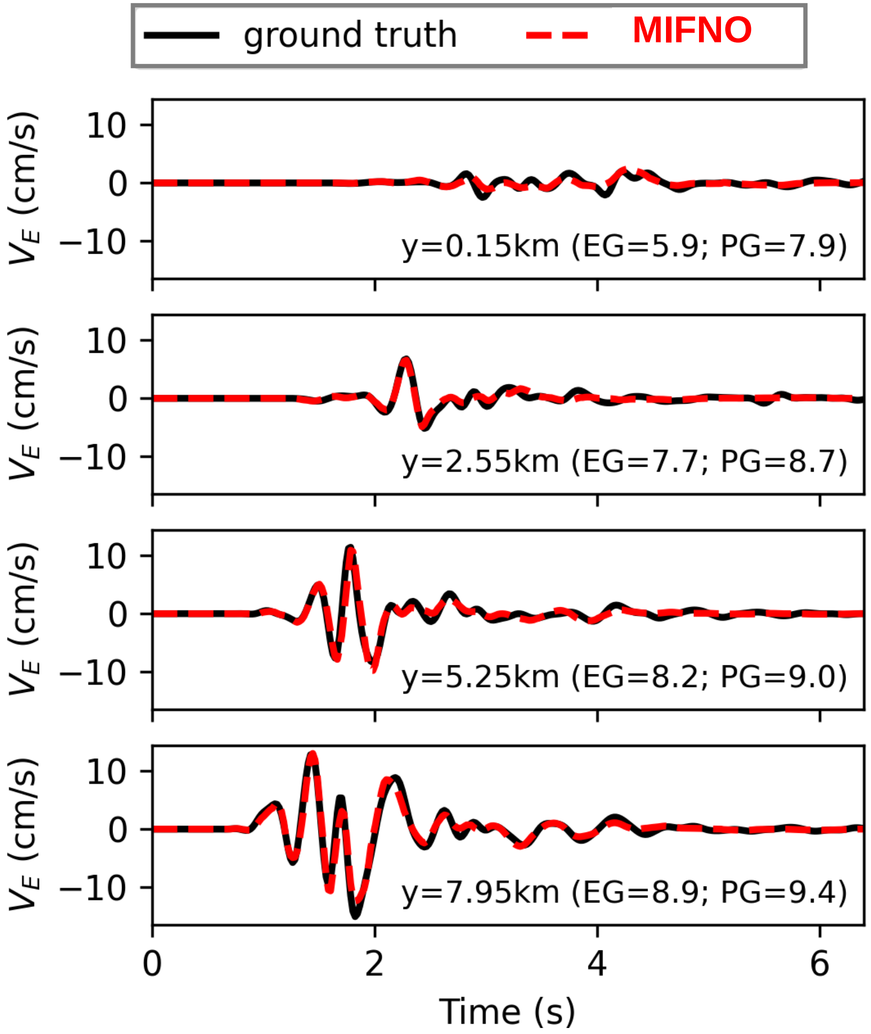


Conclusion

• take-home messages

[1]arxiv: 2203.07404
[2]**M**ultiple-**I**nput **FNO** (arxiv: 2404.10115)

- **Homogeneity and stability**
  - **FNO**s capture the influence of underlying patterns on waveforms and TLs across varying distances and time scales with a **same architecture**.
  - Unlike **PINN**s, FNOs maintain stability without violating causality[1], making them a robust choice for long-range wave propagation.
- **Adaptability to new data**
  - FNOs generate solutions with accuracy that scales with the number of modes and is independent of discretization.
  - Source-dependence can be taken into account[2].
- **Perspective:** large-scale validation, involving out-of-distribution GW fields (intermittency) ⇨ challenge: CPU costs associated with building waveforms datasets.



⇨ Source characterization using interpretable dynamic GNN
CM, X. Cassagnou, M. Mougeot

⇨ Bayesian inference using neural operators
E. Noëlé, CM, F. Lehmann

HPC Workshop for Nuclear Explosion Monitoring

3 – 5 December 2024

Any questions?

christophe.millet@cea.fr
christophe.millet@lmd.ipsl.fr

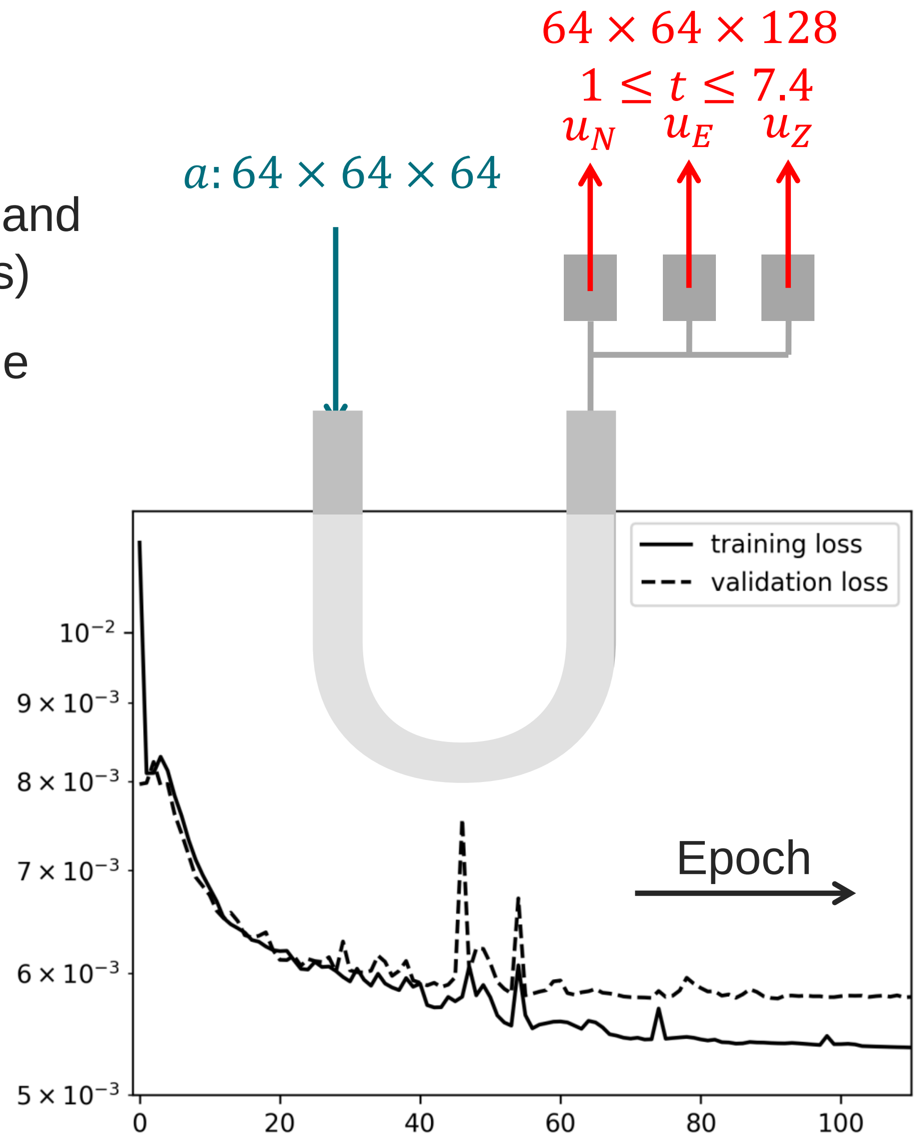
U-FNO for deep learning of seismic waves
Architecture

Parameters of U-FNO

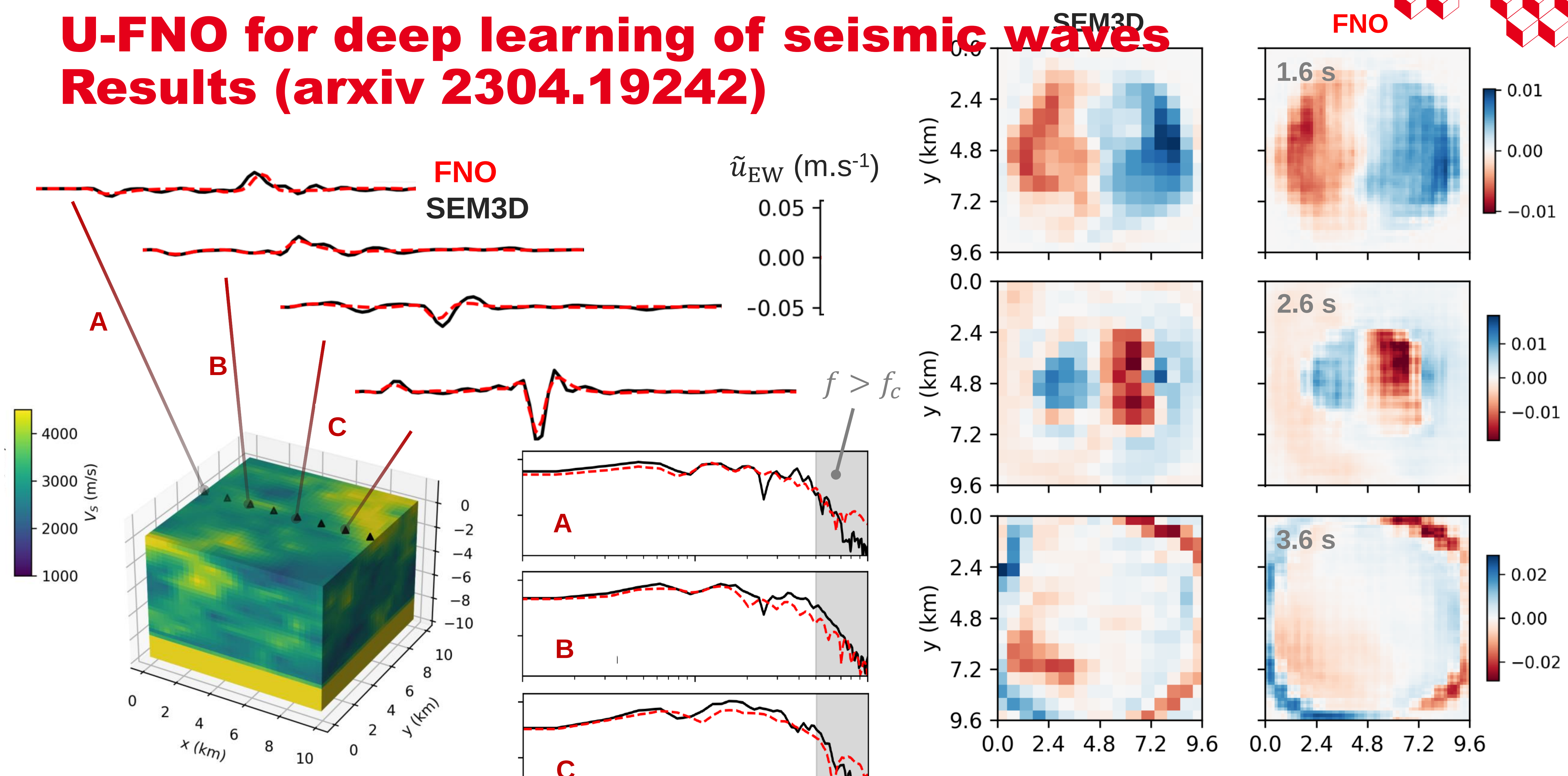
- 8 (4+4) Fourier layers $\mathbf{F}$, two fully-connected layers for $\mathbf{P}$, $\mathbf{Q_E}$, $\mathbf{Q_N}$ and $\mathbf{Q_Z}$ (projection operators for E-W, N-S and vertical displacements)
- Physical dimensions are ↘ in the encoder (64^3 to 8^3) and ↗ in the decoder (8^3 to $64^2 \times \underbrace{128}_{t}$).

Test case

- a: S-wave velocity field in a 9.6^3 km³ cube (a = matrices 64^3) obtained from adding von Karman random fields in layers.
- u: SEM3D using hexahedral mesh with elements of size 300 m ($f_c = 5$ Hz). Seismic source (moment tensor) is placed in the middle of the bottom layer. 256 virtual sensors equally spaced at $z = 0$.
- Computational cost : $3.10^4 \times 50$ min (64 CPUs) for SEM3D and 11 h (4 Nvidia A100 GPUs) for training of the U-FNO.



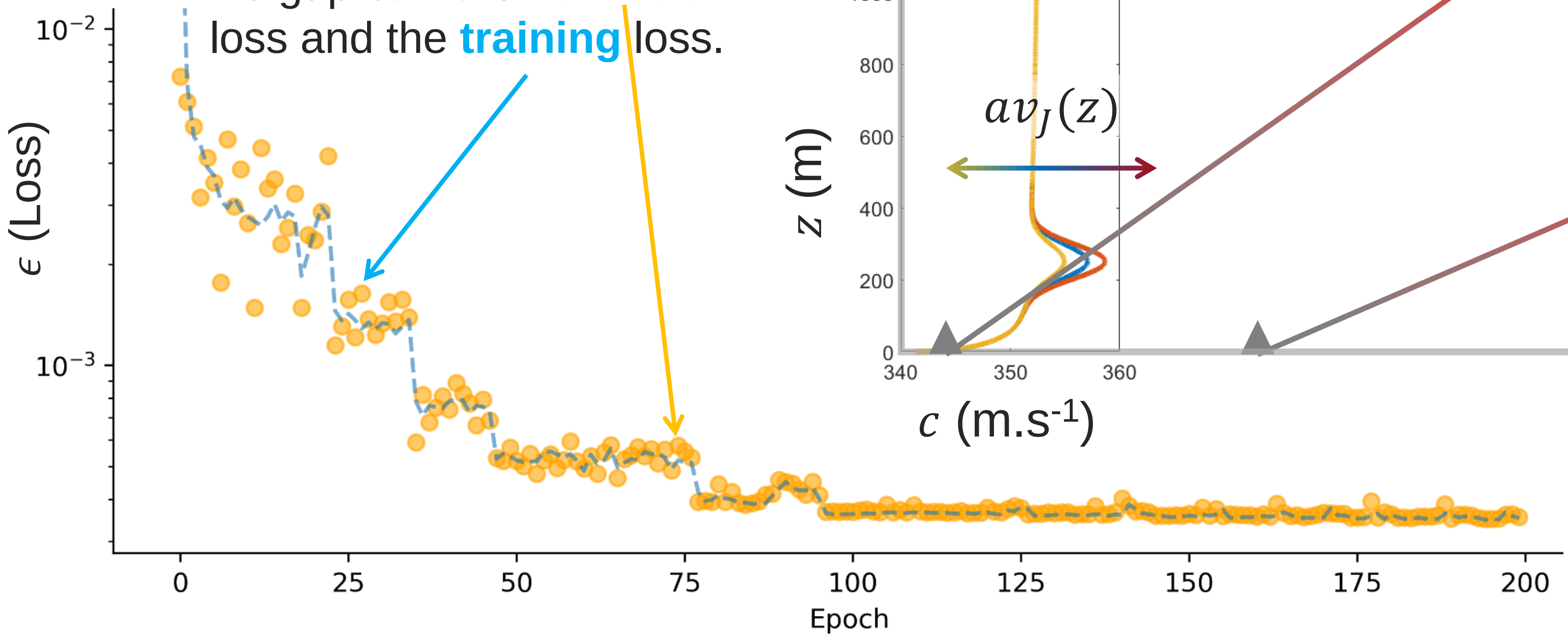
U-FNO for deep learning of seismic waves
Results (arxiv 2304.19242)

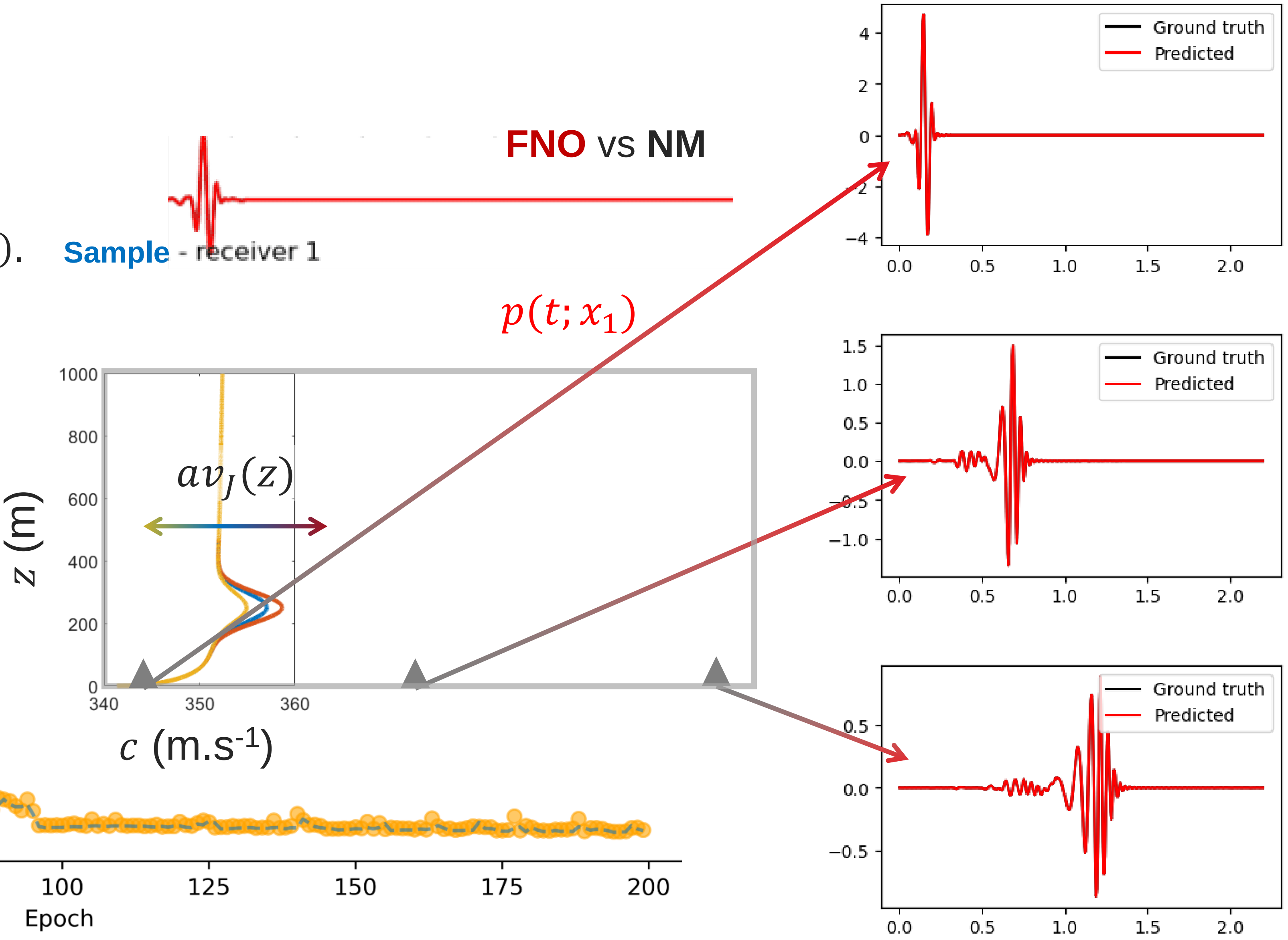


cea

- The U-FNO can learn the relationship btw. $c(z)$ and $p(t)$.
- Good generalization.
- No gap btw. the **validation** loss and the **training** loss.

FNO vs **NM**



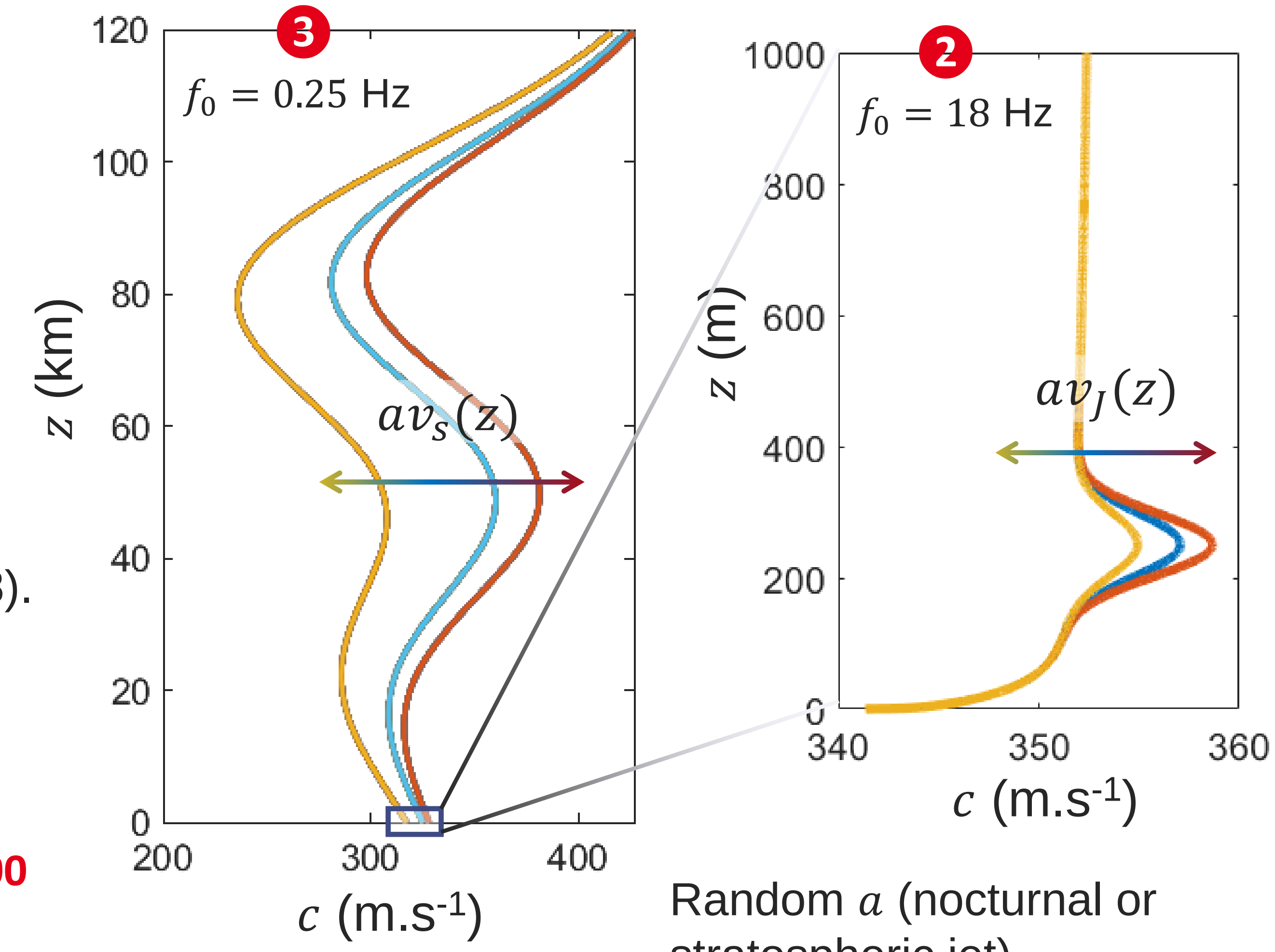


Parameters of FNO

- 4 or 6 Fourier layers $\mathbf{F}$, two fully-connected layers for $\mathbf{P}$ and $\mathbf{Q}$, 48 neurones/layer.
- U-FNO: Physical dimensions are ↘ in the encoder ($\times\, 10^{-1}$) and ↗ in the decoder.

Test cases ⇨ 2, 3

- **INPUT**: $c(x, z)$ is given by a matrix (100^2 for 2 and 256×31 for 3) obtained using simplified models (GMM, Waxler *et al*., 2008).
- **OUTPUT**: waveforms are computed with a normal mode-based technique. Source at 1 km: Kinney model with f_0.
- Comput. cost for training: 45 min for 2 **(1500 profiles)** and 2 h for 3 **(4000)**. Pred.:**1 ms.**



Random a (nocturnal or stratospheric jet).

Causality Aware PINNs

Loss function (general case)

- For a system of Q PDEs $\mathcal{F}_j(\mathbf{u};\lambda)=0$ (with $\mathbf{u}\in\mathbb{R}^Q$)

$$\epsilon^{\Omega}=\sum_{j=1}^{Q}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\sum_{k=1}^{N_k}\epsilon_{jik}^2\right]$$

with $\epsilon_{ji}=\mathcal{F}_j(\mathbf{u}(\mathbf{x}_i,t_k);\lambda)$

  - The residuals at t_{k+1} are minimized even if the prediction at t_k and previous times are inaccurate → violates temporal causality.

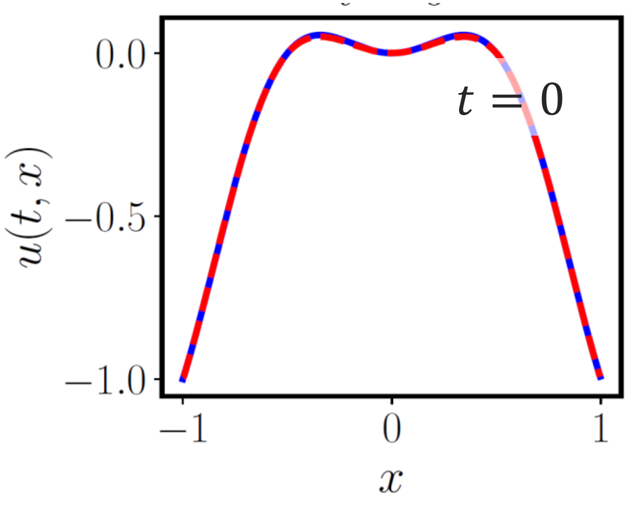
Illustrative example

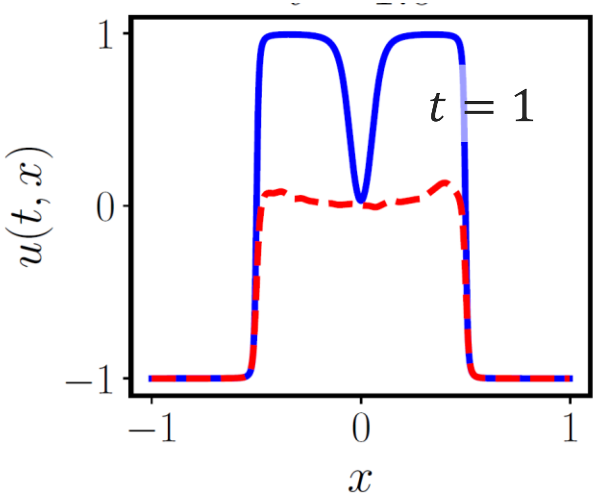
- 1D Allen Cahn equation:

$$\partial_t u-\epsilon^2\partial_{xx}u+f(u)=0,$$

$u(x,0)=x^2\cos(\pi x)$, $u(t,-1)=u(t,1)$ (same for u_x).

  - Results for $\epsilon=0.01$ and $f(u)=5(u^3-u)$. Difficult to solve with PINN, re-sampling strategies (Wight & Zhao, 2020).





$N_j=2.5\ 10^4$, Iter 2.10^5, 4hl, 4n/l, $\tanh(.)$

Causality Aware PINNs

NTK method : We seek to minimize $\epsilon(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} L\left(u_{\boldsymbol{\theta}}\left(\mathbf{x}^i\right), y^i\right)$.
The weights are obtained by solving ($t = \mathrm{Iter}$):

$$\boldsymbol{\theta}'(t) = -\frac{1}{N}\sum_i \nabla_{\boldsymbol{\theta}(t)} u_{\boldsymbol{\theta}(t)}\left(\mathbf{x}^i\right) \nabla_u L(u, y^i).$$

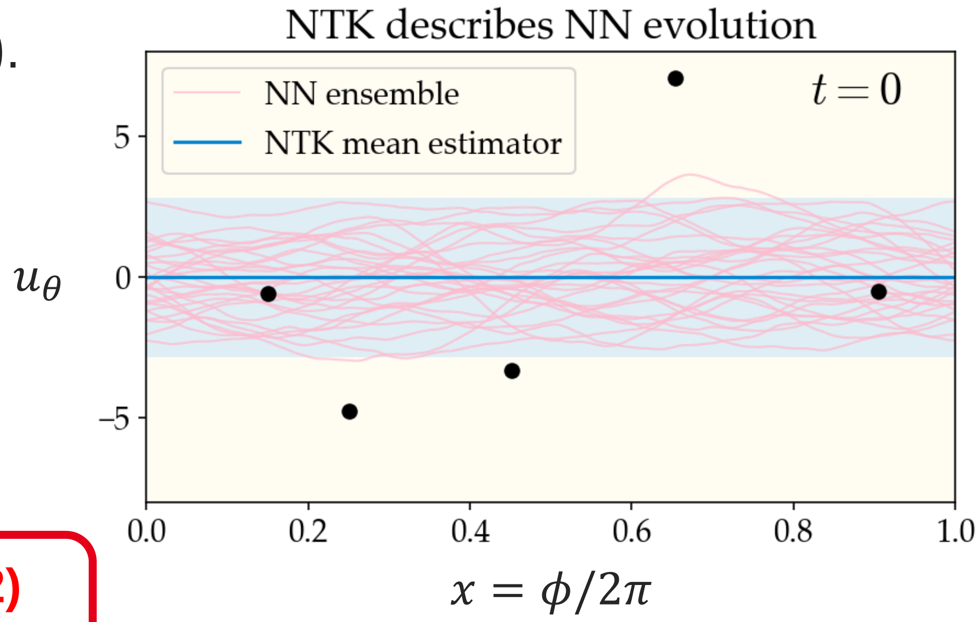
From $\partial_t u_{\boldsymbol{\theta}(t)}(\mathbf{x}) = \left(\nabla_{\boldsymbol{\theta}(t)} u_{\boldsymbol{\theta}(t)}(\mathbf{x})\right)^\top \boldsymbol{\theta}'(t)$ we get:

$$\partial_t u_{\boldsymbol{\theta}(t)}(\mathbf{x}) = -\frac{1}{N}\sum_i \underbrace{\left(\nabla_{\boldsymbol{\theta}(t)} u_{\boldsymbol{\theta}(t)}(\mathbf{x})\right)^\top \nabla_{\boldsymbol{\theta}(t)} u_{\boldsymbol{\theta}(t)}\left(\mathbf{x}^i\right)}_{k_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{x}^i)} \nabla_u L(u, y^i).$$

For $n_1, \dots, n_L \to \infty$ **(1)** $k_\infty(\mathbf{x}, \mathbf{x}^i)$ is **deterministic** at $t = 0$ and **(2)** $k_\infty(\mathbf{x}, \mathbf{x}^i)$ remains **constant** as t increases.

For $L(u, y) = (u - y)^2$ we obtain $\partial_t \mathbf{Y}_{\boldsymbol{\theta}(t)} = 2/N\left(\mathbf{Y} - \mathbf{Y}_{\boldsymbol{\theta}(t)}\right)\mathbf{K}_\infty$,
where $\left(\mathbf{Y}_{\boldsymbol{\theta}(t)}\right)_i = u_{\boldsymbol{\theta}}(\mathbf{x}^i)$ and $(\mathbf{K}_\infty)_{ij} = k_\infty(\mathbf{x}^i, \mathbf{x}^j)$

$$\mathbb{E}\left[\mathbf{Y}_{\boldsymbol{\theta}(t)}\right] = \mathbf{Y}(\mathbf{I} - e^{-t\mathbf{K}_\infty}).$$



At $t = 0$, an ensemble of wide NNs is a zero-mean GP; for $t > 0$, the ensemble evolves according to the NTK.

$u_\theta \in \mathbb{R}$ trained on inputs drawn from S^1.

⇒The larger $u \in \mathrm{Sp}(\mathbf{K}_\infty)$ is, the faster the NN learns in the direction of the eigenvector.

Causality Aware PINNs

Kernel at time t over points $\mathbf{x}_i$

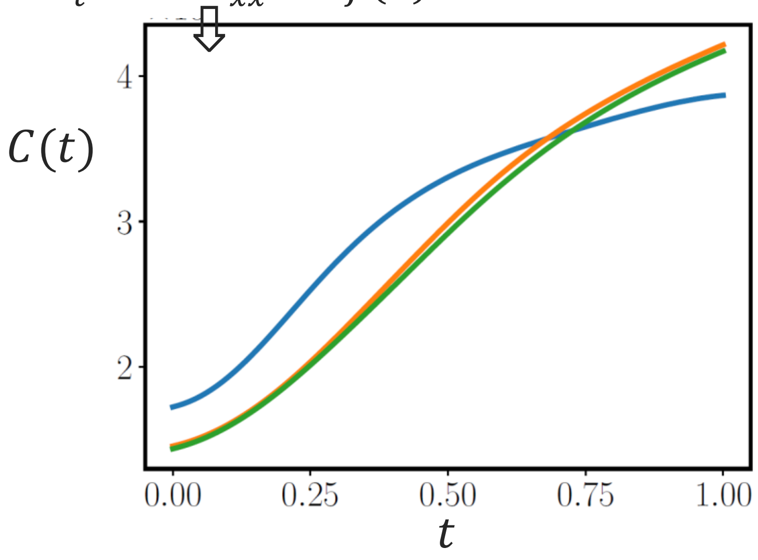
$$\left(\mathbf{K}_\infty(t)\right)_{ij} = \nabla_{\boldsymbol{\theta}} R(t, \mathbf{x}_i)^\top \nabla_{\boldsymbol{\theta}} R\left(t, \mathbf{x}_j\right).$$

Constant (as $n_l \to \infty, \forall l$) during training, the rate of (exponential) convergence being (Wang *et al*., 2022):

$$C(t) = \frac{\mathrm{tr}(\mathbf{K}_\infty(t))}{N_{\mathbf{x}}}.$$

Allen-Cahn Equation

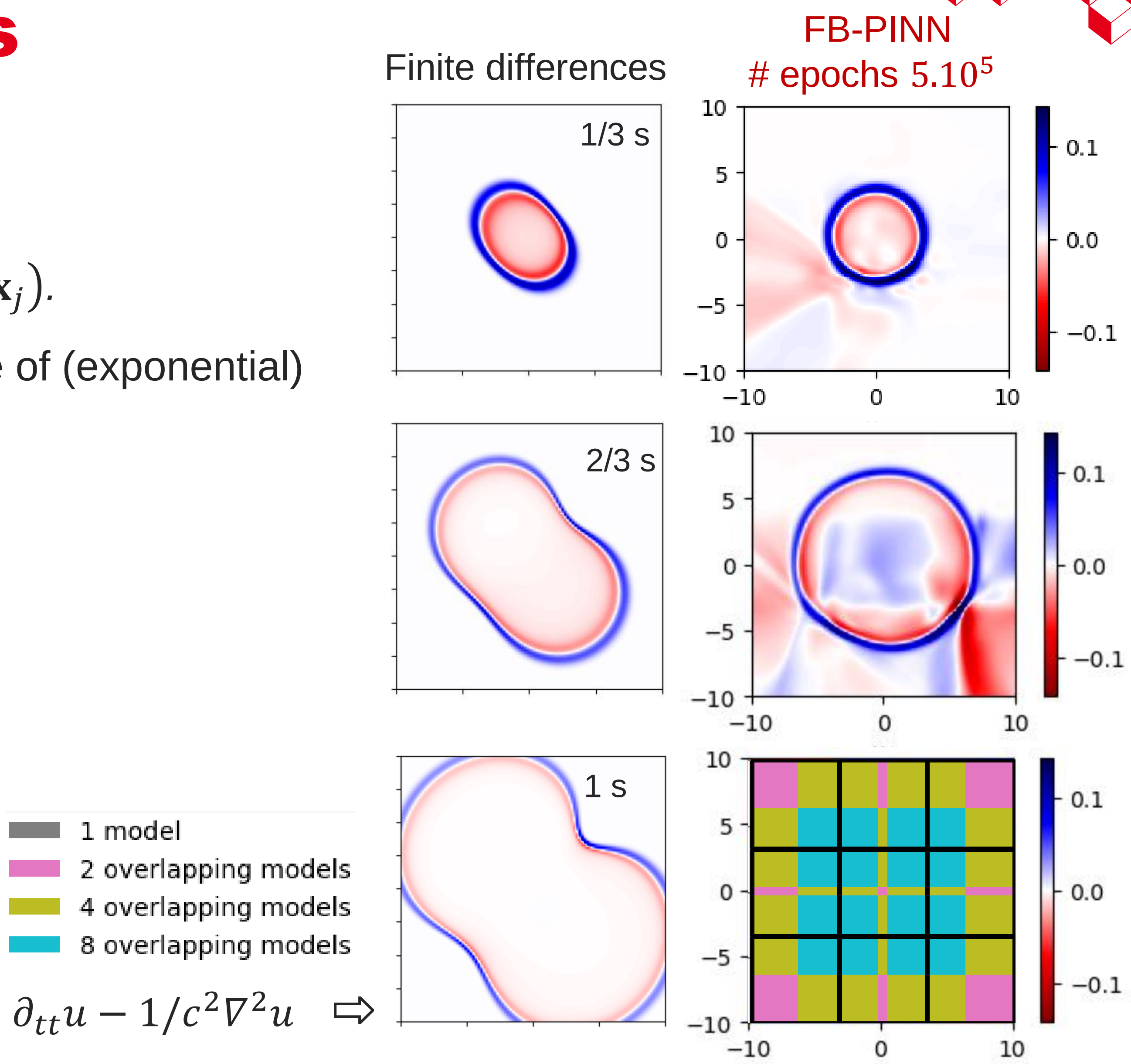
$\partial_t u - \epsilon^2 \partial_{xx} u + f(u) = 0$



Iter. 10^4 et 10^5

Iter. 10^3

$R = \partial_{tt} u - 1/c^2 \nabla^2 u$



Predictions

●○○○ A quick view on statistics