## Data-Driven Subgrid-Scale Modeling: Stability, Extrapolation & Interpretation

Pedram Hassanzadeh

Yifei Guan (postdoc)

Adam Subel (undergrad student  $\rightarrow$  PhD at Courant)

Ashesh Chattopadhyay (PhD student)





**Rice University** 

KITP 2021

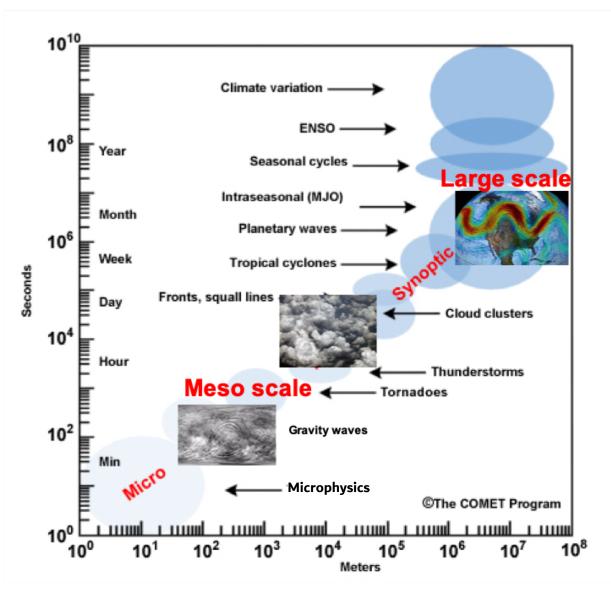






#### **Climate, Weather Extremes & Turbulence:**

spatio-temporal, *multi-scale*, *multi-physics*, high-dimensional & chaotic ...



**X**: large/slow-scale variables The main variables of interest

#### **Y:** small/fast-scale variables Influence the spatio-temporal variability of X

## **Traditional approach:**

Coarse-resolution numerical solver + physics-based subgrid-scale (SGS) model

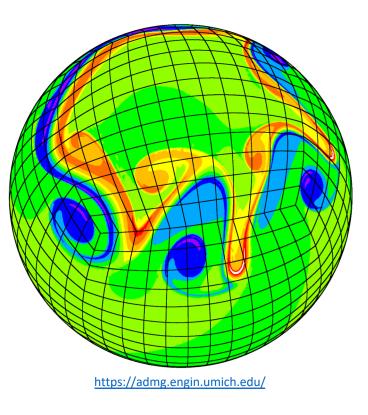
Large-scale processes

 $\dot{X} = \mathbf{F}(X, \mathbf{P}(X))$ 

solved numerically at O(10)-O(100)km resolutions

Closure for SGS processes (parameterization)

 $Y = \mathbf{P}(X)$ 



## **ML-based** approach:

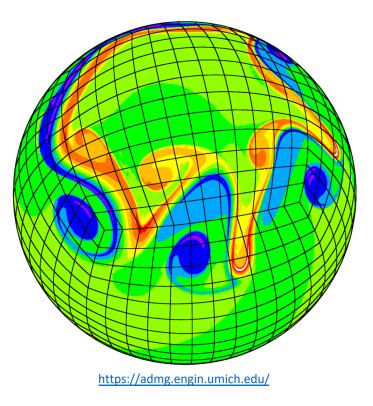
Coarse-resolution numerical solver + data-driven subgrid-scale (SGS) model

Large-scale processes

 $\dot{X} = \mathbf{F}(X, \mathbf{NN}(X))$ 

solved numerically at O(10)-O(100)km resolutions

Data-driven closure for SGS processes (*data-driven parameterization*, DD-P)



Y = NN(X)

Main focus of this talk: *Non-parametric* DD-P

## Using ML for modeling weather/climate/turbulence: Questions, challenges & opportunities

• How to use ML?

*Non-parametric* DD-P

• How to choose the ML method?

Dealing with poor (high-quality) data regimes
Incorporating physics/PDEs' properties

- Interpretability
- Generalization (i.e., extrapolation: to different forcing ....)

Instability: blow-up in coupled (ML+numerical solver) models

## Using ML for modeling weather/climate/turbulence: Questions, challenges & opportunities

- How to use ML?
- How to choose the ML method?

Dealing with poor (high-quality) data regimes

Incorporating physics/PDEs' properties

Interpretability

Generalization (i.e., extrapolation)

transfer learning

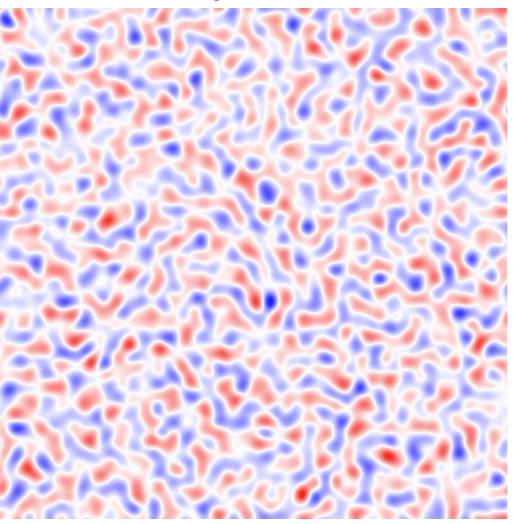
Instability: blow-up in coupled (ML+numerical solver) models

## **Test case: 2D Turbulence**

$$\frac{\partial\omega}{\partial t} + J(\omega,\psi) = \frac{1}{Re} \nabla^2 \omega + f^0$$

 $\nabla^2 \psi = -\omega$ 

 $\omega$ : Vorticity  $\psi$ : Streamfunction  $J(\omega, \psi)$ : Jacobian Re:Reynold number f: Forcing Direct numerical simulation (DNS) Re=32000; grid=2048 x 2048



## Large-Eddy Simulation (LES)

$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\text{Re}} \nabla^2 \omega$$
Gaussian
$$\frac{\partial \overline{\omega}}{\partial t} + J(\overline{\omega}, \overline{\psi}) = \frac{1}{\text{Re}} \nabla^2 \overline{\omega} + \underbrace{\left[J(\overline{\omega}, \overline{\psi}) - \overline{J(\omega, \psi)}\right]}_{\Pi = \nabla \times (\nabla \cdot \tau^{SGS}) \cdot \hat{z}}$$

Re=32000 DNS grid = 2048 x 2048 , time step =  $\Delta t$ LES grid = 256 x 256 , time step =  $10\Delta t$  Gaussian filter + coarse graining

Guan, Chattopadhyay, Subel & Hassanzadeh, Stable a posteriori LES of 2D turbulence with convolutional neural networks: backscattering analysis and generalization to higher Re via transfer learning, in revision arXiv: 2102.11400

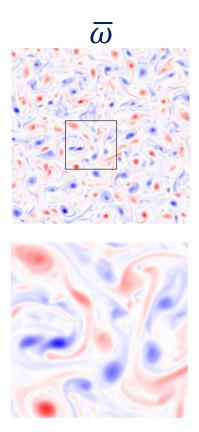
physics-based parameterization: Smagorinsky's model (1963)  $\Pi = v_e \nabla^2 \overline{\omega}$ 

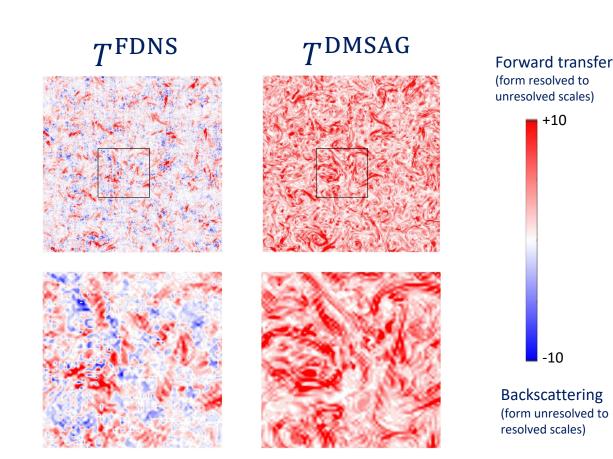
data-driven, non-parametric parameterization (DD-P):  $\Pi = NN(\overline{\omega}, \overline{\psi})$ 

#### Major shortcoming of many physics-based models: Only diffusive, not accounting for *backscattering*

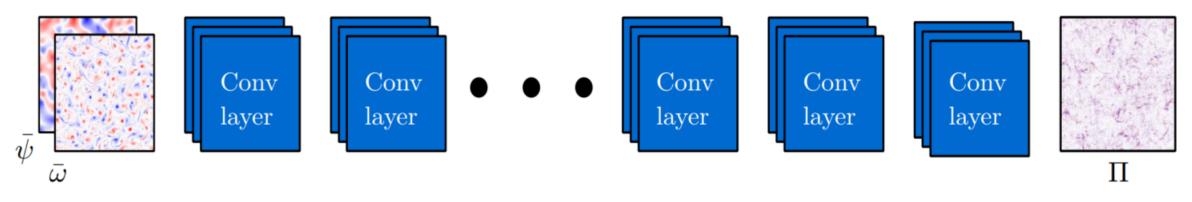
 $T = \prod \nabla^2 \overline{\omega}$  T: subgrid-scale transfer

 $T^{\text{DSMAG}} = v_e \ \nabla^2 \overline{\omega} \ \nabla^2 \overline{\omega} \ge 0$ (Dynamic Smagorinsky, Germano et al. 1991)





## Spatially non-local, non-parametric DD-P using CNNs



10 layers (64 filters, 5 x 5) + ReLU + no pooling/upsampling

#### **Training:** find $\theta$ that minimizes

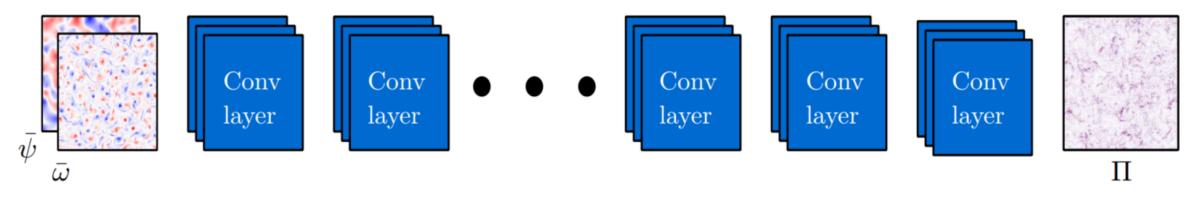
$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} \left( \Pi_n - CNN(\bar{\psi}_n, \bar{\omega}_n, \theta) \right)^2$$

#### **Offline testing:**

 $\Pi(x,y) = CNN(\overline{\psi},\overline{\omega},\theta)$ 

- Physics agnostic neural network architecture
- Physics agnostic loss function
- Deterministic parameterization
- Memoryless parameterization
- No uncertainty quantification (UQ)
- Use clean (noise-free) DSN data for training

## Spatially non-local, non-parametric DD-P using CNNs



10 layers (64 filters, 5 x 5) + ReLU + no pooling/upsampling

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} \left( \prod_{n} - CNN(\overline{\psi}_{n}, \overline{\omega}_{n}, \theta) \right)^{2}$$

#### Training dataset:

From 7 DNS runs started from random initial conditions

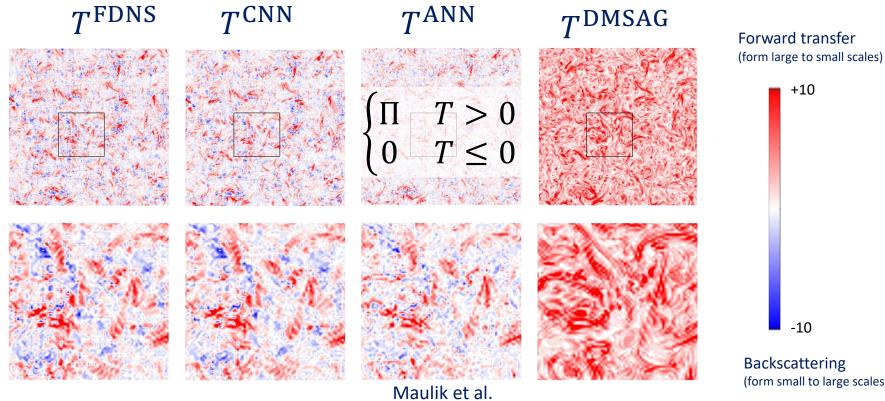
#### Validation dataset:

From 3 DNS runs started from random initial conditions

#### **Testing dataset:**

From 5 DNS runs started from random initial conditions <sup>11</sup>

## A priori (offline) test of DD-P



2019 JFM

(form small to large scales)

 $c = corr{\Pi^{FDNS}, \Pi^{model}}$  averaged over 100 samples

	SMAG	DSMAG	ANN	CNN
Correlation coefficient c	0.55	0.55	0.86	0.93

## Stability of a posteriori (coupled) LES model?

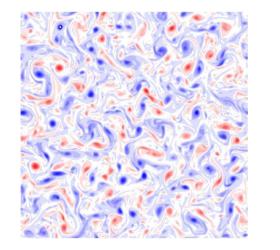
A priori accuracy of the CNN-based DD-P & the fate of coupled LES run as a function of the numer of training samples, *N* 

N	500	1000	10000	30000	50000
С	0.78	0.83	0.90	0.92	0.93
Fate	Unstable	Unstable	Unstable	Stable	Stable

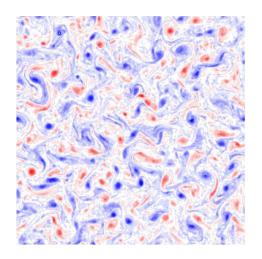
- Backscattering is harder to learn data drivenly when the training set is small
- Speculation: Disproportionally low accuracy for backscattering is the reason for instabilities

## Accuracy of a posteriori (online) LES with DD-P

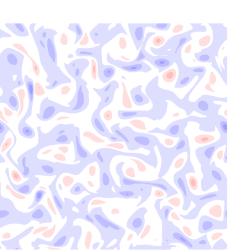
**FDNS** 

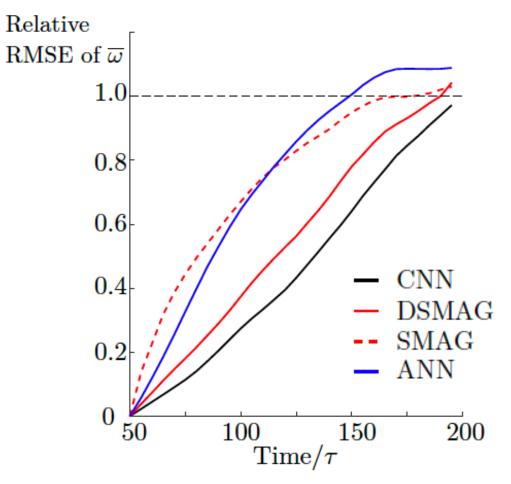


**LES-CNN** 

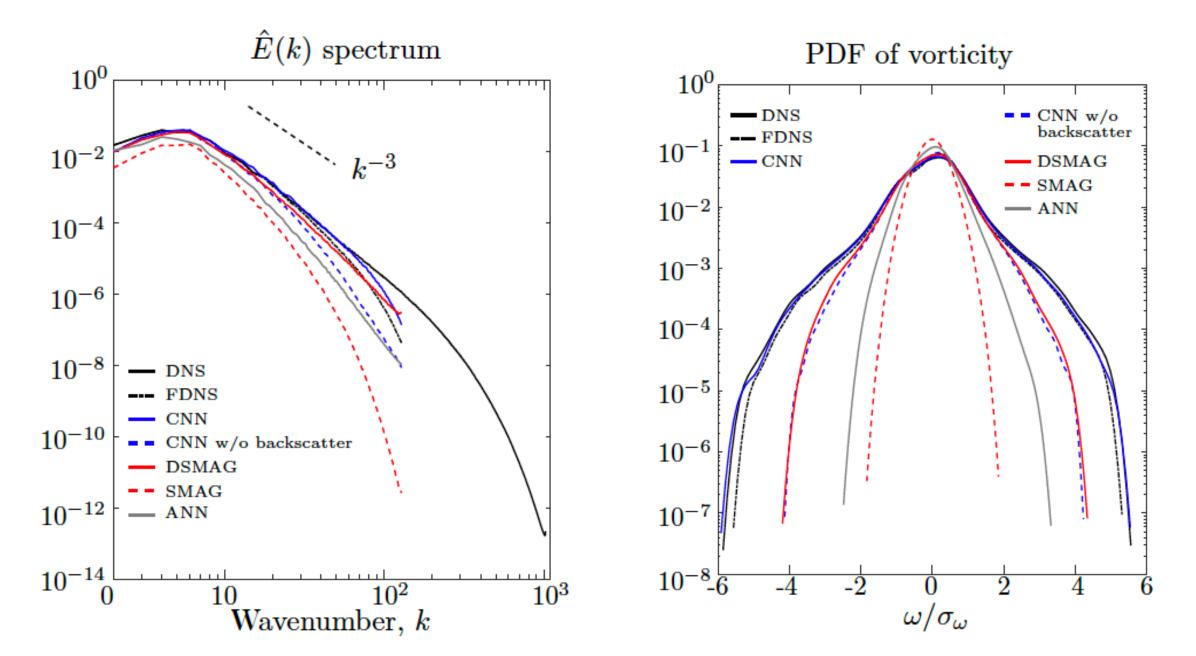


**LES-SMAG** 





## Accuracy of a posteriori (online) LES with DD-P



## Using ML for modeling weather/climate/turbulence: Questions, challenges & opportunities

• Best ways to use ML?

*Non-parametric* DD-P

• How to choose the ML method?

Dealing with poor (high-quality) data regimes
Incorporating physics/PDEs' properties

- Interpretability
- Generalization (i.e., extrapolation: to different forcing ....)

Instability: blow-up in coupled (ML+numerical solver) models

Instabilities could be due to inaccuracies resulting from small training set

#### Physics-agnostic CNNs: Some of the shortcomings in the small-data regime

Small- vs big-data regimes: not just the number of samples, but also *inter-sample* correlations

- Data augmentation: build symmetries into the input samples
- Equivariant CNNs: physics built into the architecture
- Physics-constrained loss functions:  $\langle \overline{\omega} \Pi \rangle^{\text{FDNS}} = \langle \overline{\omega} \Pi \rangle^{\text{CNN}}$

Physics-constrained learning: Same performance with 50 and 2000 samples

Guan, Subel, Chattopadhyay, & Hassanzadeh, *Learning physics-constrained data-driven subgrid-scale closures in the small-data regime for stable large-eddy simulations,* to be submitted soon

Subel, Chattopadhyay, Guan & Hassanzadeh, Data-driven subgrid-scale modeling of forced Burgers turbulence using deep learning with generalization to higher Reynolds numbers via transfer learning, Physics of Fluids (2021)

## Using ML for modeling weather/climate/turbulence: Questions, challenges & opportunities

- Best ways to use ML?
- How to choose the ML method?

Dealing with poor (high-quality) data regimes

Incorporating physics/PDEs' properties

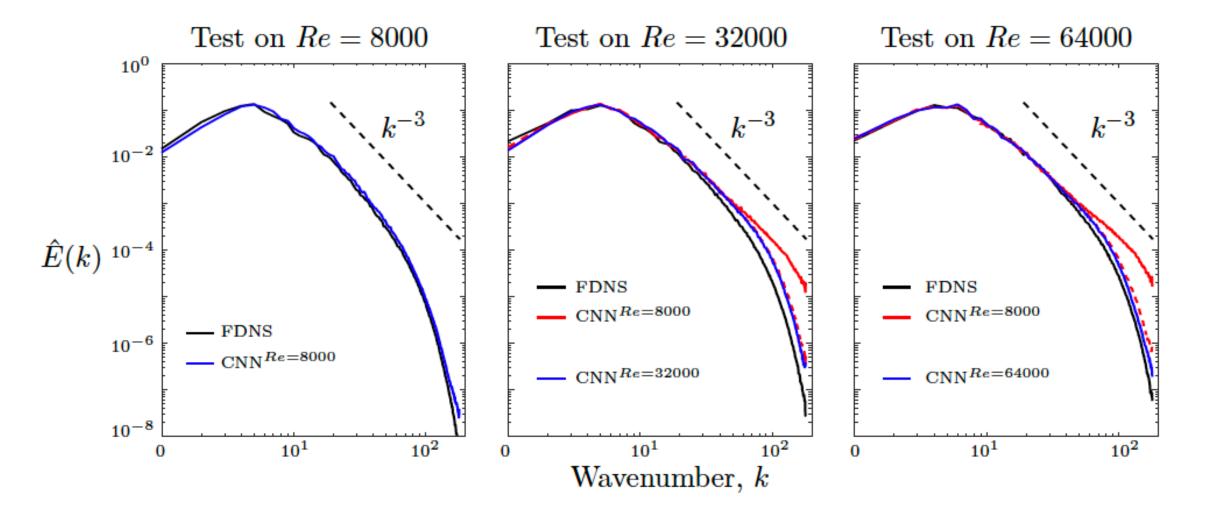
Interpretability

Generalization (i.e., extrapolation)

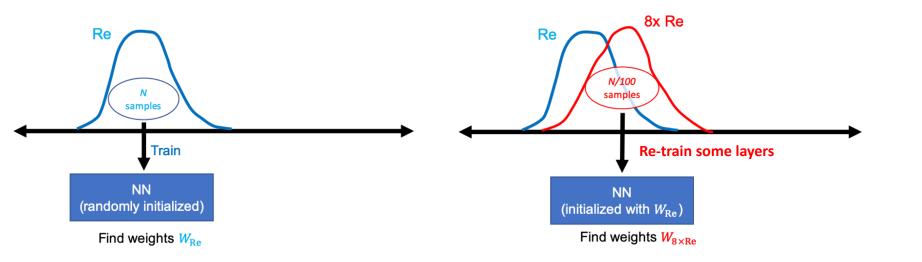
transfer learning

Instability: blow-up in coupled (ML+numerical solver) models

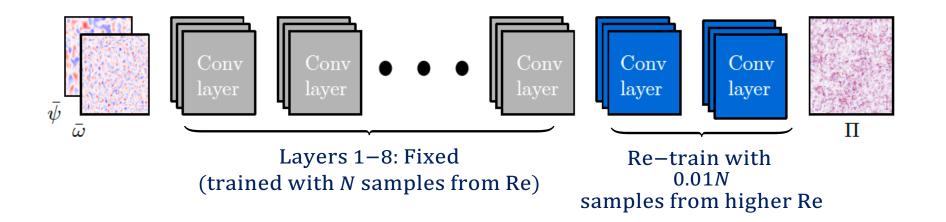
#### **DD-P** does not extrapolate to higher *Re*



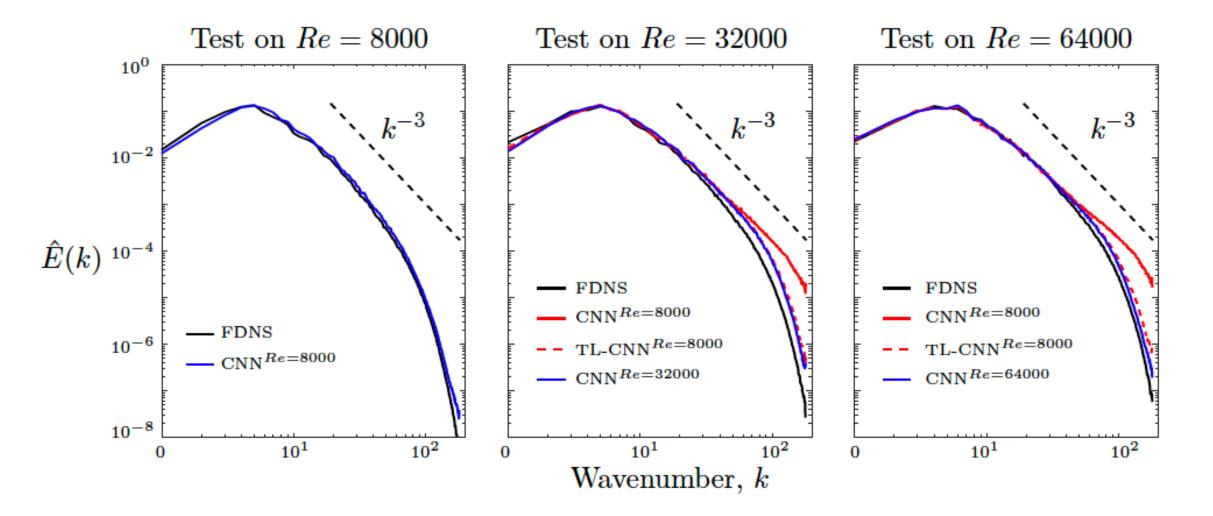
## Generalization to higher Re via transfer learning



Chattopadhyay, Subel & Hassanzadeh, Data-driven super-parameterization using deep learning: Experimentation with multiscale Lorenz 96 systems and transfer learning. J. Advances in Modeling Earth Systems (2020)



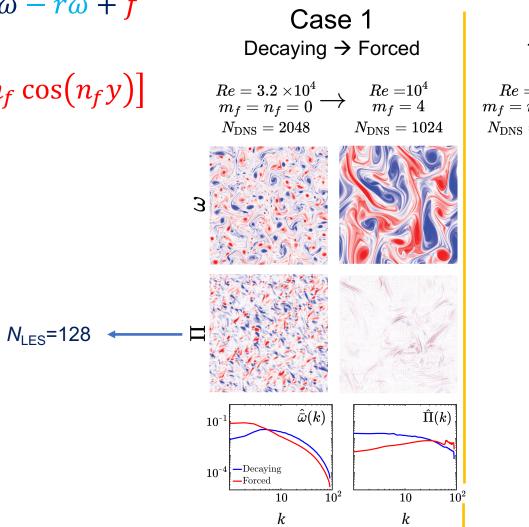
#### Generalization to higher Re via transfer learning



Guan, Chattopadhyay, Subel & Hassanzadeh, Stable a posteriori LES of 2D turbulence with convolutional neural networks: backscattering analysis and generalization to higher Re via transfer learning, in revision <u>arXiv: 2102.11400</u>

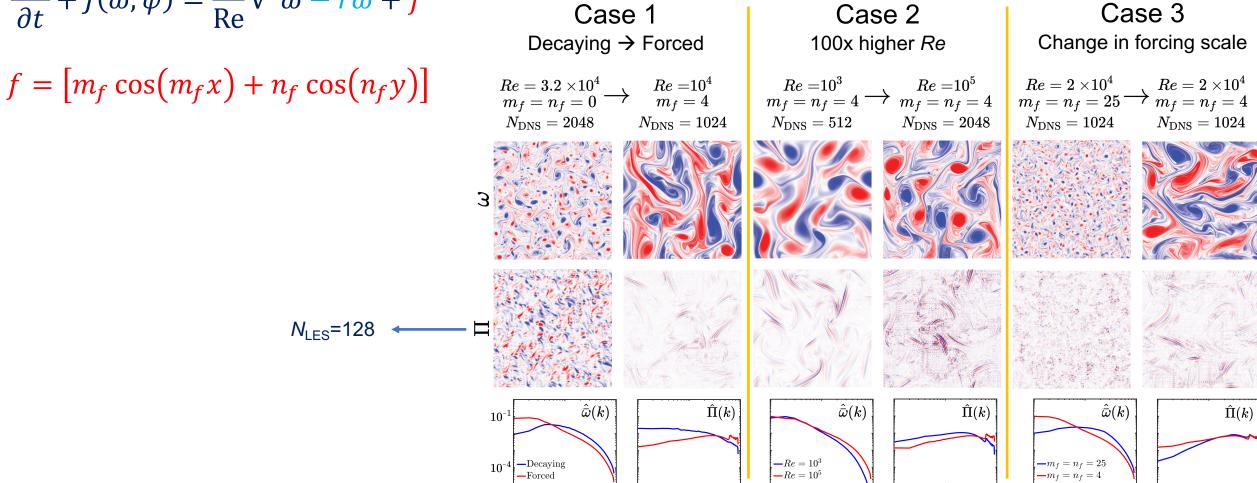
$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\text{Re}} \nabla^2 \omega - r\omega + f$$

 $f = [m_f \cos(m_f x) + n_f \cos(n_f y)]$ 



# Case 2<br/>100x higher ReCase 3<br/>Change in forcing scale $Re = 10^3$ <br/> $m_f = n_f = 4$ <br/> $N_{\text{DNS}} = 512$ $Re = 10^5$ <br/> $m_f = n_f = 4$ <br/> $N_{\text{DNS}} = 2048$ $Re = 2 \times 10^4$ <br/> $m_f = n_f = 25$ <br/> $N_{\text{DNS}} = 1024$

$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\text{Re}} \nabla^2 \omega - r\omega + f$$



10

 $10^2$ 

10

 $10^{2}$ 

10

 $10^{2}$ 

10

 $10^2$ 

 $10^{2}$ 

10

 $10^{2}$ 

10

BNN: Base CNN trained with *N* samples from the base or target system

TLNN: Retrained BNN with N/10 samples from the target system

$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\text{Re}} \nabla^2 \omega - r\omega + f$$

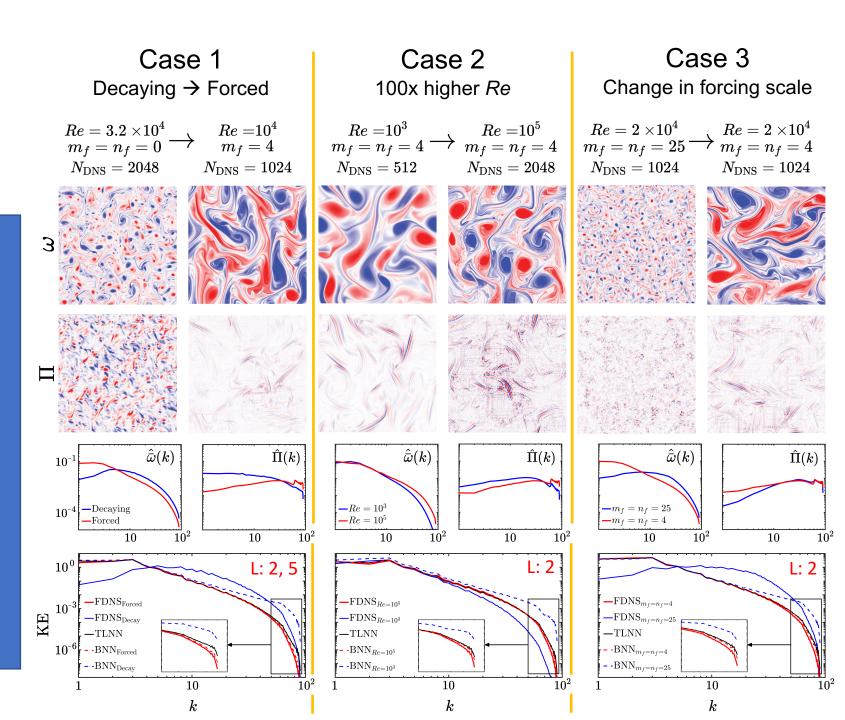
 $f = [m_f \cos(m_f x) + n_f \cos(n_f y)]$ 

#### **Questions:**

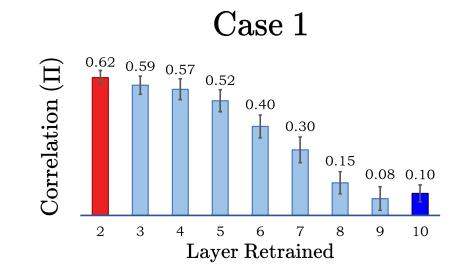
- What is learnt during transfer learning?
- For a given BNN and new system, what are the optimal layer(s) to retrain?

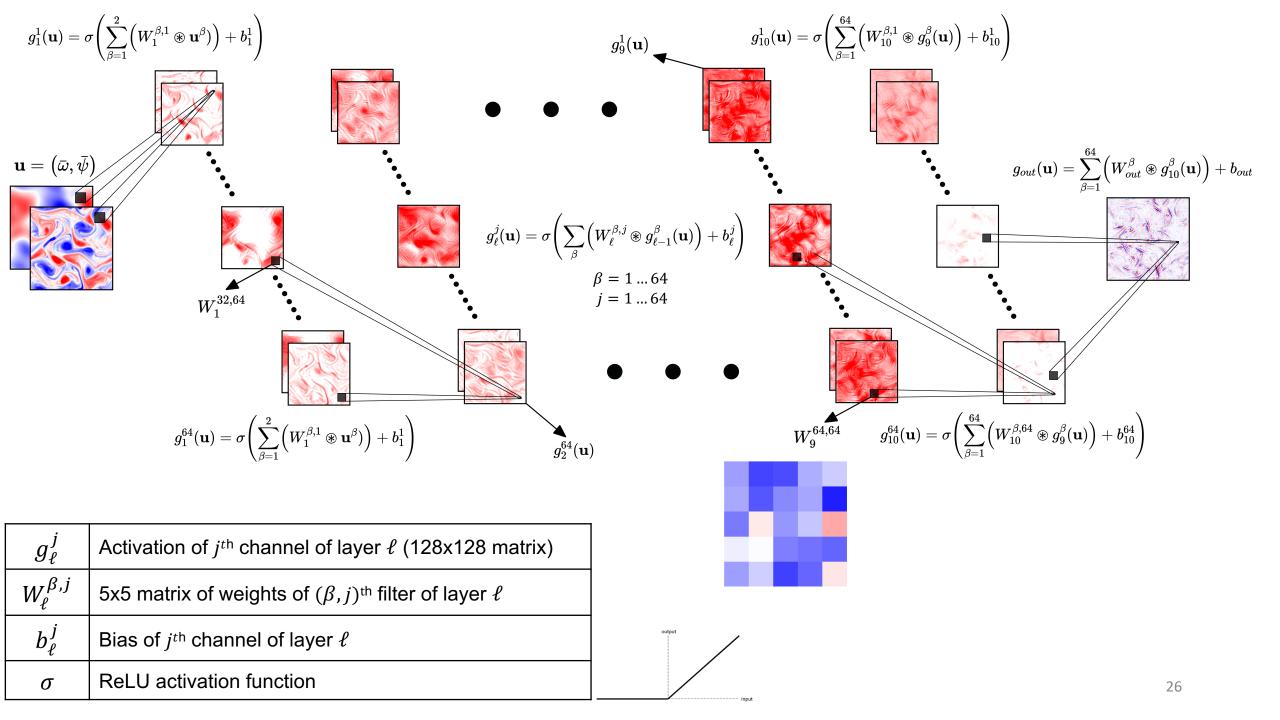
#### **Ultimate goals:**

- A framework to guide TL
- Build more accurate TLNNs with fewer retraining samples

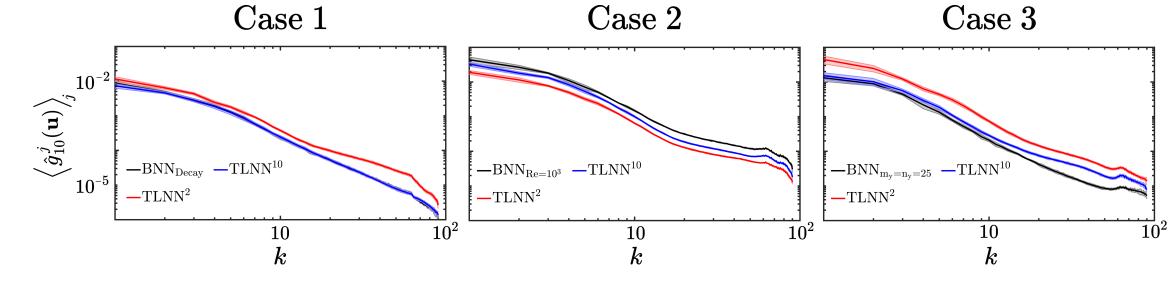


#### **Best layers to re-train? The shallowest ones**



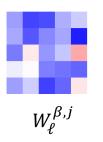


# Re-training deeper layers leads to small/no changes in the output, particularly at large scales



$$g_{\ell}^{j} = \sigma \left( \sum_{\beta} \left( W_{\ell}^{\beta,j} \circledast g_{\ell-1}^{\beta} \right) + b_{\ell}^{j} \right) \qquad h_{\ell}^{j} = \sum_{\beta} \left( W_{\ell}^{\beta,j} \circledast g_{\ell-1}^{\beta} \right) + b_{\ell}^{j}$$

$$\hat{g}_{\ell}^{j} = \sum_{\alpha} \left( e^{-i\left(k_{x}x_{\alpha} + k_{y}y_{\alpha}\right)} \right) \circledast \hat{h}_{\ell}^{j}(m,n)$$
$$(x_{\alpha}, y_{\alpha}) \in \left\{ (x, y) \left| h_{\ell}^{j}(x, y) > 0 \right\}$$

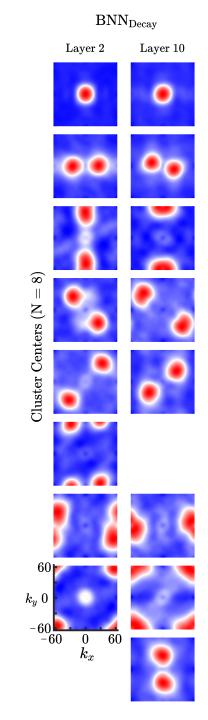


#### What are the filters?

Spectra of BNN filters

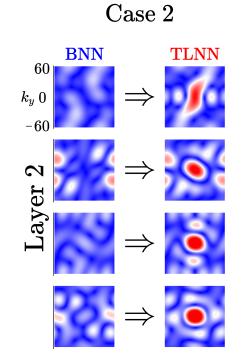
 $\widehat{W}_{\ell}^{eta,j}$ 

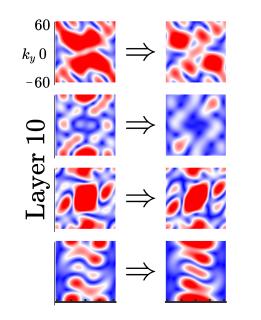
 $\beta = 1, 2 \dots 64$  $j = 1, 2 \dots 64$ 



#### How do filters change during transfer learning?

Isolated the 4 filters that have changed the most after transfer learning (based on Frobenius norm)





#### How can we determine *a priori* what layers can change the most for new data? Calculate the *loss landscape* of the BNN layers for new data

$$\theta_{\ell} = \left\{ W_{\ell}^{\beta, j}, b_{\ell} \right\}$$

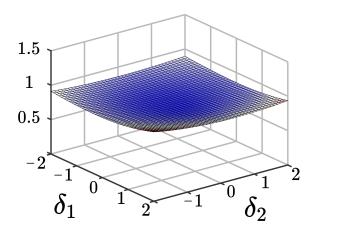
Pick layer(s): L

 $(\Theta_1, \Theta_2)$ : two random, normalized matrices

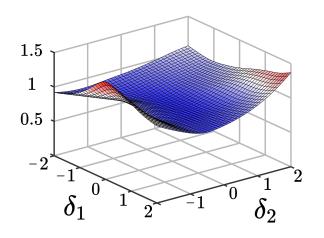
Perturb  $\theta_L$ :  $\tilde{\theta}_L = \theta_L + \delta_1 \Theta_1 + \delta_2 \Theta_2$ 

Compute  $\mathcal{L}(\boldsymbol{u}^{new}, \theta_{\ell \neq L}, \tilde{\theta}_L)$ 

Li et al. (2018) arXiv: 1712.09913 Krishnapriyan et al. (2021) arXiv:2109.01050



Layer 2



Layer 10



**Questions & answers** (for this application & this NN architecture):

- 1) What is learnt during transfer learning?
  - A number of new spectral filters (mostly low-pass filters)

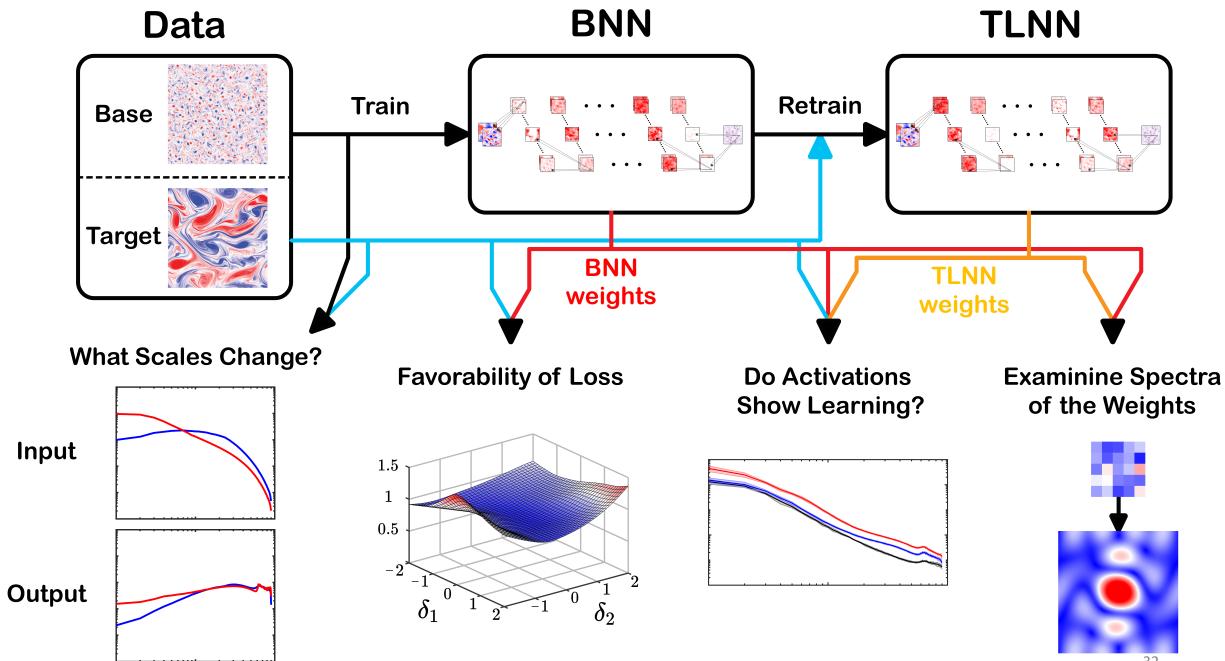
2) For a given BNN and new system, what are the optimal layer(s) to retrain?- Shallowest layers, because the main difference between *base* and *new* data is in large scales

#### **Other Applications & architectures?**

- Applications so far: SGS modeling & fully data-driven forecasting
  - Changing parameters or blending a *large* training dataset with a *smaller, higher-fidelity* dataset
- (1) Still, likely a number of new spectral filters
- (2) Would depend on base/new data differences, may depend on NN architecture

#### **Ultimate goals:**

- A framework to guide TL
- Build more accurate TLNNs with fewer retraining samples



**Questions & answers** (for this application & this NN architecture):

- 1) What is learnt during transfer learning?
  - A number of new spectral filters (mostly low-pass filters)

2) For a given BNN and new system, what are the optimal layer(s) to retrain?- Shallowest layers, because the main difference between *base* and *new* data is in large scales

#### **Other Applications & architectures?**

- Applications so far: SGS modeling & fully data-driven forecasting
  - Changing parameters or blending large, low-fidelity data with small, high-fidelity data
- (1) Still, likely a number of new spectral filters
- (2) Would depend on base/new data differences, may depend on NN architecture

#### **Ultimate goals:**

A framework to guide TL

Build more accurate TLNNs with fewer retraining samples