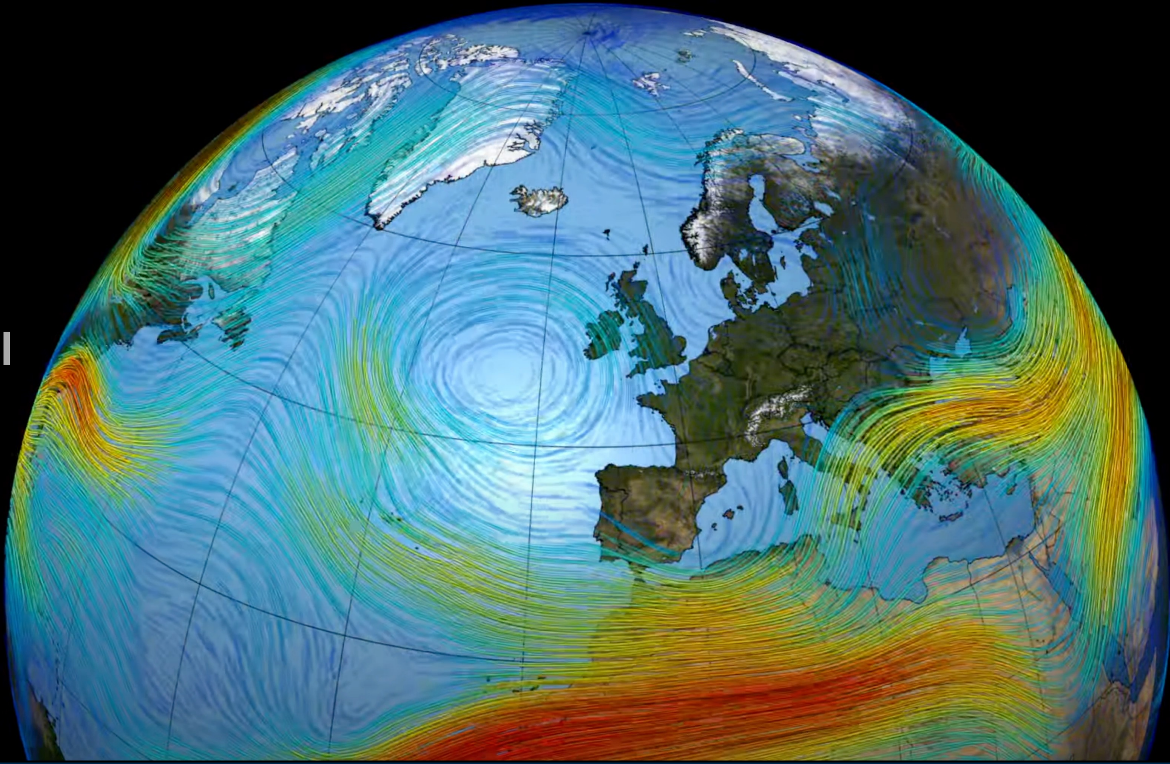
Learning Stochastic Dynamics with Neural Networks to study Zonal Jets

Ira Shokar[1,2], Peter Haynes[1] & Rich Kerswell[1]

[1]DAMTP, Cambridge; [2] UKRI AI4ER CDT

Project Goal:

A Deep Learning approach to deriving a reduced-order model of stochastically forced atmospheric zonal jets, that provides a speed-up in emulating the jets, over numerical integration.



UNIVERSITY OF CAMBRIDGE

LIFD Workshop March 2023

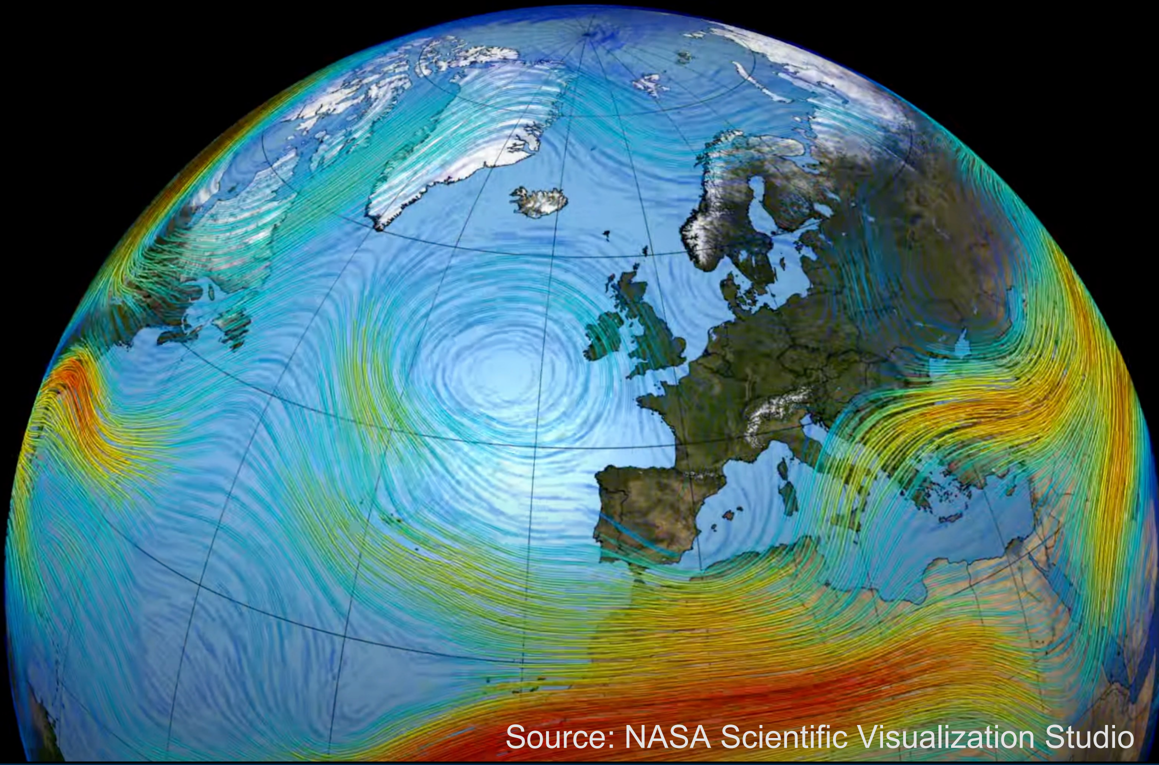
AI4ER

Motivation: Planetary Zonal Jets

- Jet streams have a major influence over regional weather patterns, transporting quantities such as momentum and heat and tracers, such as ozone and water vapour.

- Within CMIP6 projections there are biases in the representation of jets[1].

- The computational expense of GCMs results in requiring many processes to be parameterised.



Source: NASA Scientific Visualization Studio

[1] Dorrington et al. doi:10.5194/wcd-3-505-2022

System of Study: *Barotropic, stochastically forced turbulent flow on a beta-plane.*

- Starting with the shallow water equations we neglect stratification and solve the system on a 2D plane with periodic boundary conditions.
- We incorporate planetary rotation by adopting a beta-plane approximation.
- Parameterise the turbulence due to baroclinic instabilities with a stochastic forcing - ξ:

$$\frac{\partial \zeta}{\partial t} + J(\psi, \zeta) + \beta \frac{\partial \psi}{\partial x} = \xi - \mu\zeta + \upsilon_n \nabla^{2n} \zeta$$

System of Study: *Barotropic, stochastically forced turbulent flow on a beta-plane.*

- Starting with the shallow water equations we neglect stratification and solve the system on a 2D plane with periodic boundary conditions.
- We incorporate planetary rotation by adopting a beta-plane approximation.
- Parameterise the turbulence due to baroclinic instabilities with a stochastic forcing - ξ:

$$\frac{\partial \zeta}{\partial t} + J(\psi, \zeta) + \beta \frac{\partial \psi}{\partial x} = \xi - \mu\zeta + \upsilon_n \nabla^{2n}\zeta$$

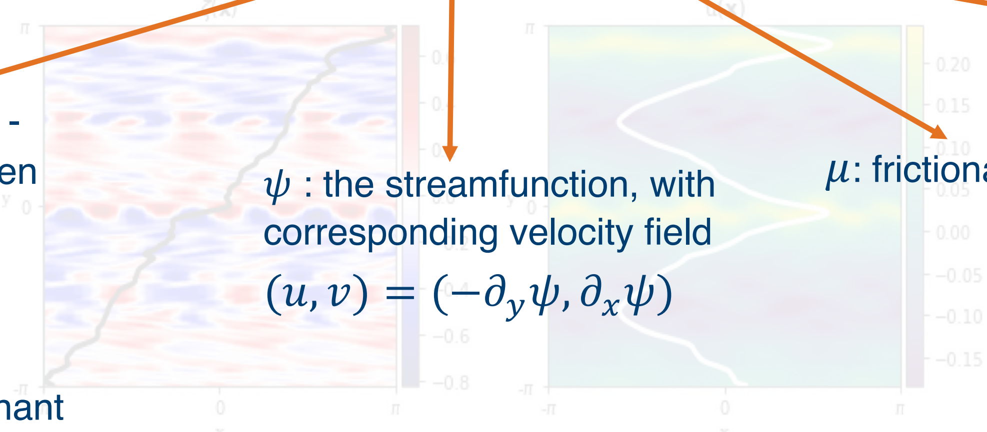
$\zeta = \nabla^2\psi$: the relative vorticity

ν: hyperviscosity

β : the Rossby parameter - the latitudinal gradient when linearising the Coriolis parameter, $f = f_0 + \beta y$

ψ : the streamfunction, with corresponding velocity field $(u, v) = (-\partial_y\psi, \partial_x\psi)$

μ: frictional damping rate

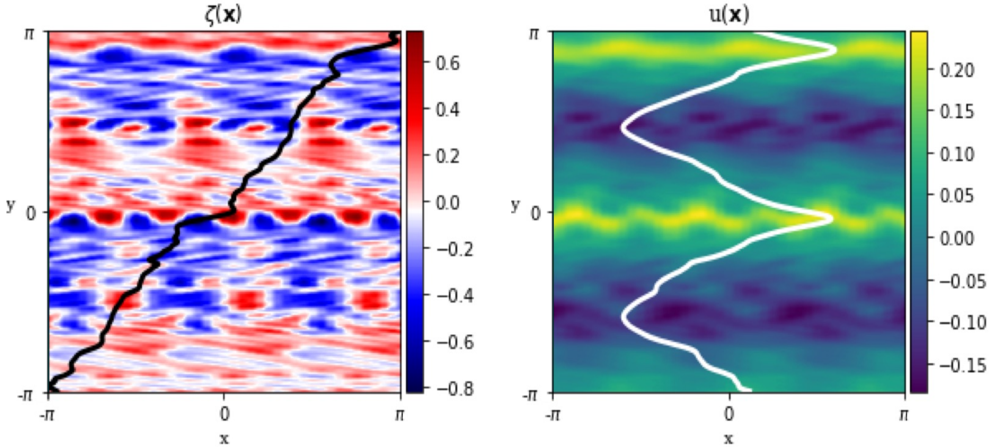


- J: the Jacobian determinant

System of Study: *Barotropic, stochastically forced turbulent flow on a beta-plane.*

- Starting with the shallow water equations we neglect stratification and solve the system on a 2D plane with periodic boundary conditions.
- We incorporate planetary rotation by adopting a beta-plane approximation.
- Parameterise the turbulence due to baroclinic instabilities with a stochastic forcing - ξ:

$$\frac{\partial \zeta}{\partial t} + J(\psi, \zeta) + \beta \frac{\partial \psi}{\partial x} = \xi - \mu\zeta + \upsilon_n \nabla^{2n} \zeta$$



System of Study: *Barotropic, stochastically forced turbulent flow on a beta-plane.*

- Studying zonally-oriented flows, we perform a Reynolds decomposition, to obtain an EOM for the zonally-averaged zonal velocity ($U(y,t) = \overline{u}(y,t) = u(x,y,t) - u'(x,y,t)$):

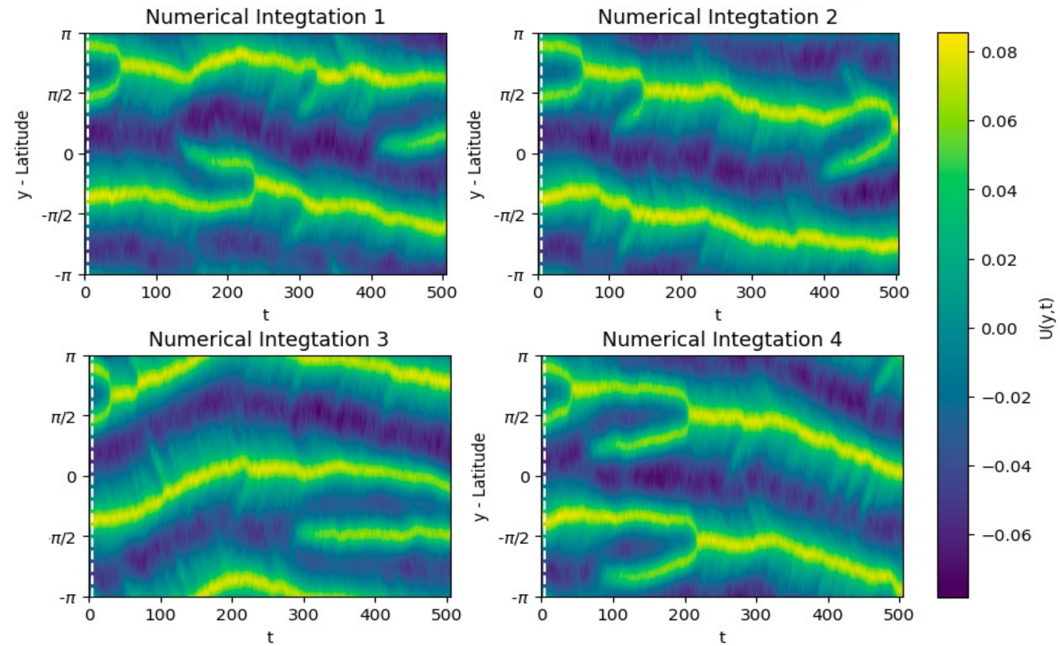
$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial y}(\overline{u'v'}) = -\mu U + \nu_n \frac{\partial^{2n}}{\partial y^{2n}} U$$

System of Study: *Barotropic, stochastically forced turbulent flow on a beta-plane.*

- Studying zonally-oriented flows, we perform a Reynolds decomposition, to obtain an EOM for the zonally-averaged zonal velocity ($U(y,t) = \overline{u}(y,t)$ $= u(x,y,t) - u'(x,y,t)$):

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial y}\left(\overline{u'v'}\right) = -\mu U + \nu_n \frac{\partial^{2n}}{\partial y^{2n}} U$$

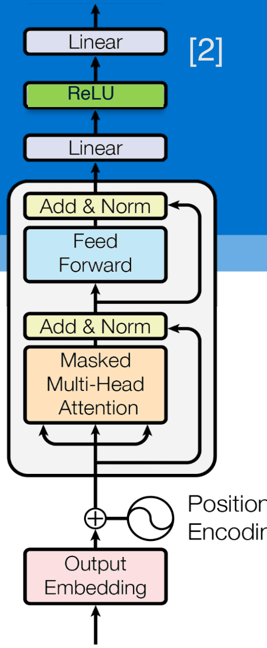
- Zonal Jets exhibit wandering, merging and nucleating behaviour.
- Can ML learn the underlying dynamics given only $U(y,t)$, implicitly parameterising fluctuation fields (u' ,v')?



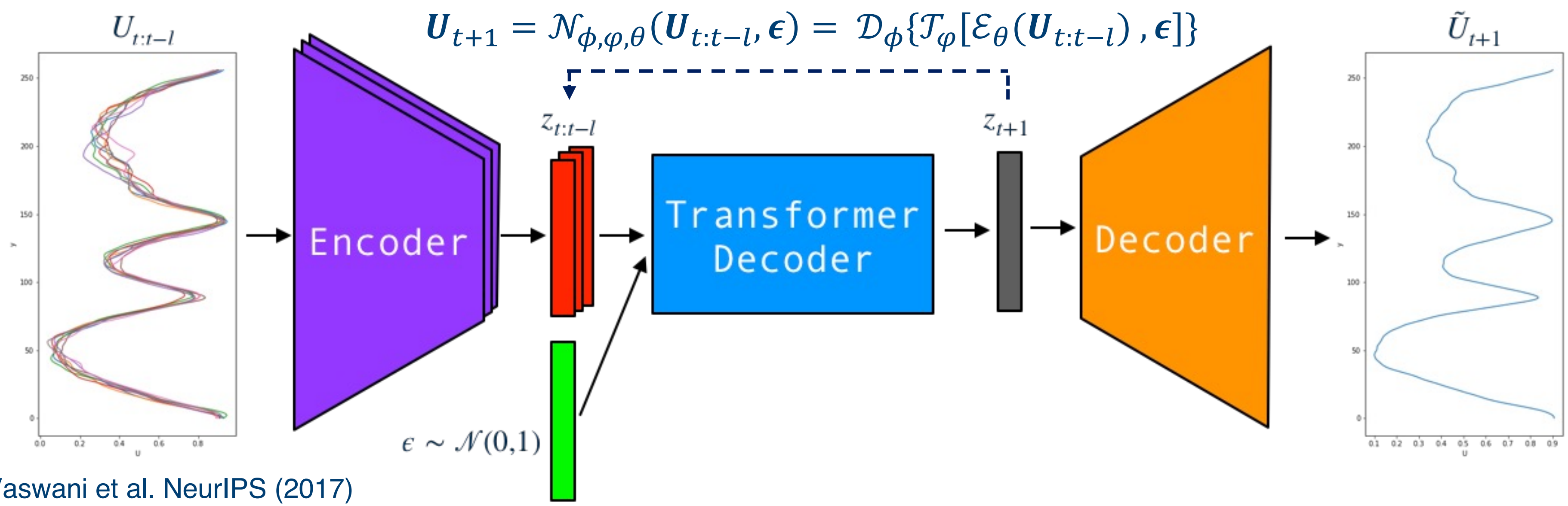
Manifold Learning

- The equations of motion lie on a manifold, $\mathcal{M}$, with a lower-degrees of freedom than the input fields $D_{\mathcal{M}} \ll D$.
- We want to learn a mapping to this latent space, $\boldsymbol{Z}_{t:t-l} = \mathcal{E}_{\theta}(\boldsymbol{U}_{t:t-l})$, induce the forcing and evolve the system in time, $\boldsymbol{Z}_{t+1} = \mathcal{T}_{\varphi}(\boldsymbol{Z}_{t:t-l}, \boldsymbol{\epsilon})$, before mapping back to the observed space, $\boldsymbol{U}_{t+1} = \mathcal{D}_{\phi}(\boldsymbol{Z}_{t+l})$.

Manifold Learning



- The equations of motion lie on a manifold, $\mathcal{M}$, with a lower-degrees of freedom than the input fields $D_{\mathcal{M}} \ll D$.
- We want to learn a mapping to this latent space, $\boldsymbol{Z}_{t:t-l} = \mathcal{E}_\theta(\boldsymbol{U}_{t:t-l})$, induce the forcing and evolve the system in time, $\boldsymbol{Z}_{t+1} = \mathcal{T}_\varphi(\boldsymbol{Z}_{t:t-l}, \boldsymbol{\epsilon})$, before mapping back to the observed space, $\boldsymbol{U}_{t+1} = \mathcal{D}_\phi(\boldsymbol{Z}_{t+l})$.

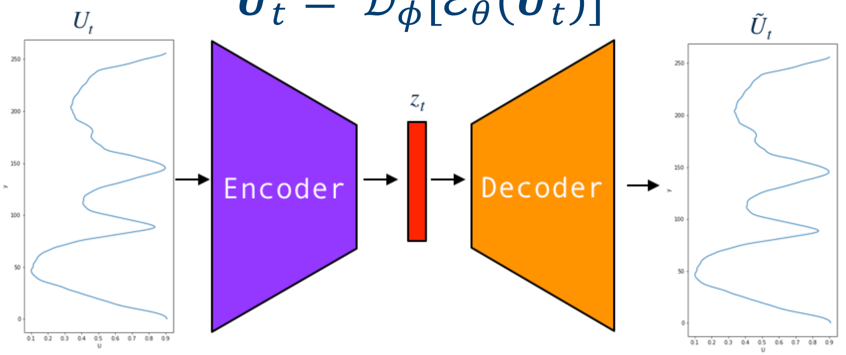


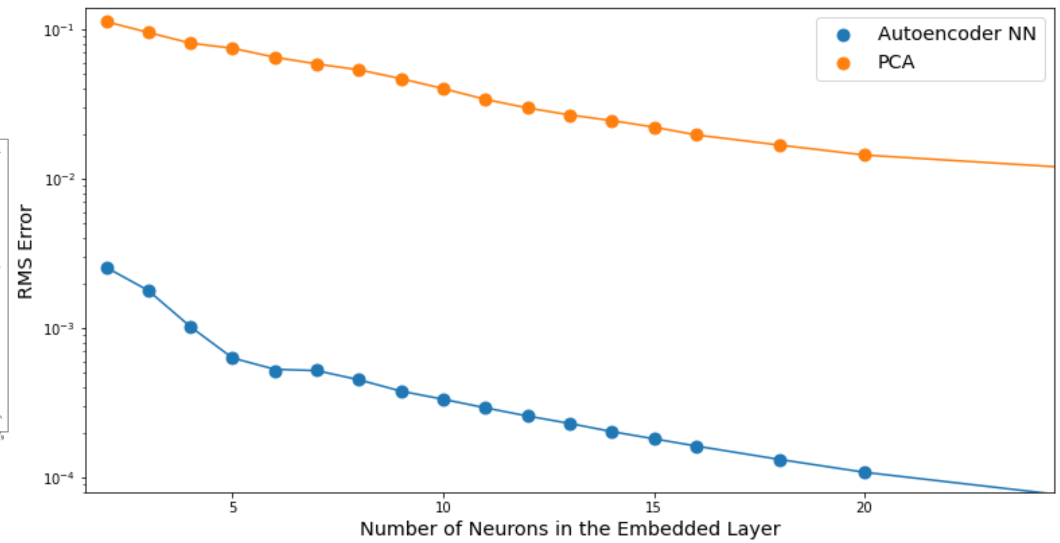
[2] Vaswani et al. NeurIPS (2017)

Manifold Learning

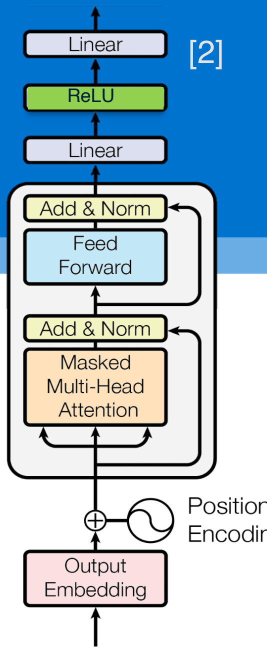
- The equations of motion lie on a manifold, $\mathcal{M}$, with a lower-degrees of freedom than the input fields $D_{\mathcal{M}} \ll D$.
- SVD/PCA or POD only capture linear manifolds, while Autoencoders use nonlinear dimensionality reduction.
- Compare spatial reduction of snapshots.

$$\boldsymbol{U}_t = \mathcal{D}_\phi[\mathcal{E}_\theta(\boldsymbol{U}_t)]$$



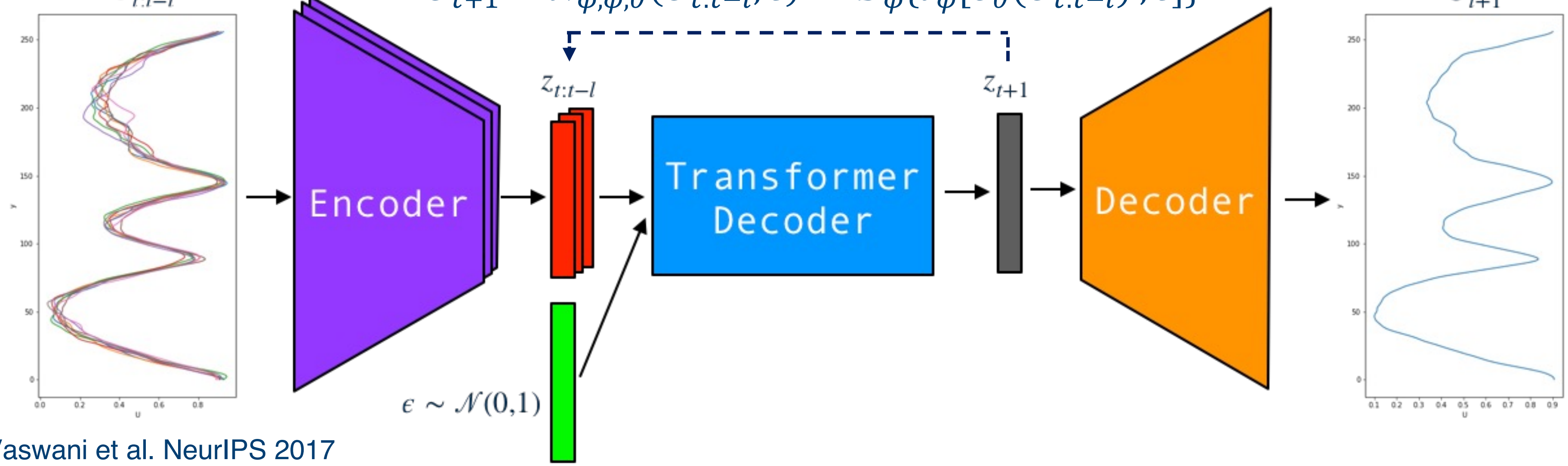


Manifold Learning



- The equations of motion lie on a manifold, $\mathcal{M}$, with a lower-degrees of freedom than the input fields $D_{\mathcal{M}} \ll D$.
- We want to learn a mapping to this latent space, $\boldsymbol{Z}_{t:t-l} = \mathcal{E}_\theta(\boldsymbol{U}_{t:t-l})$, induce the forcing and evolve the system in time, $\boldsymbol{Z}_{t+1} = \mathcal{T}_\varphi(\boldsymbol{Z}_{t:t-l}, \boldsymbol{\epsilon})$, before mapping back to the observed space, $\boldsymbol{U}_{t+1} = \mathcal{D}_\phi(\boldsymbol{Z}_{t+l})$.

$$\boldsymbol{U}_{t+1} = \mathcal{N}_{\phi,\varphi,\theta}(\boldsymbol{U}_{t:t-l}, \boldsymbol{\epsilon}) = \mathcal{D}_\phi\{\mathcal{T}_\varphi[\mathcal{E}_\theta(\boldsymbol{U}_{t:t-l}), \boldsymbol{\epsilon}]\}$$



[2] Vaswani et al. NeurIPS 2017

Objective Function

Continuous Ranked Probability Score (CRPS)[3]/Energy Score[4][5]:

$$\frac{1}{m}\sum_{i=1}^{m}\left\|\widetilde{U}_i - U\right\|^2 - \frac{1}{2m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\left\|\widetilde{U}_i - \widetilde{U}_j\right\|^2$$

[3] Matheson et al, Management Science (1976) , [4] Gneiting et al. doi: 10.1198/016214506000001437 (2012)
[5] Pacchiardi et al. doi: 10.48550/arXiv.2112.08217 (2022)

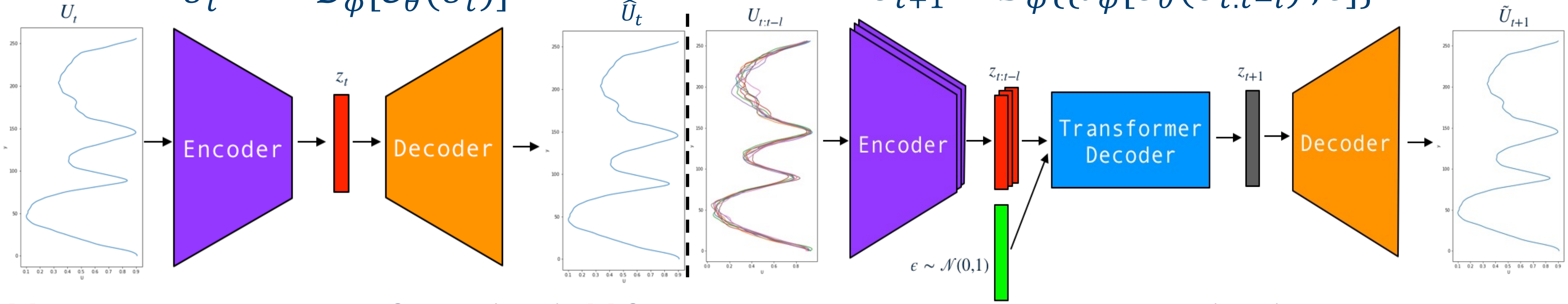
Objective Function

Continuous Ranked Probability Score (CRPS)[3]/Energy Score[4][5]:

$$\frac{1}{m}\sum_{i=1}^{m}\|\tilde{U}_i - U\|^2 - \frac{1}{2m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\|\tilde{U}_i - \tilde{U}_j\|^2$$

$$\mathcal{L}_{AE}(\theta,\phi) = ES(\tilde{U}_{t+1}^{(i)},\ U_{t+1}^{(i)}) + \gamma\|\hat{U}_t - U_t\|^2\,; \qquad \mathcal{L}_T(\varphi) = ES(\tilde{U}_{t+1}^{(i)},\ U_{t+1}^{(i)})$$

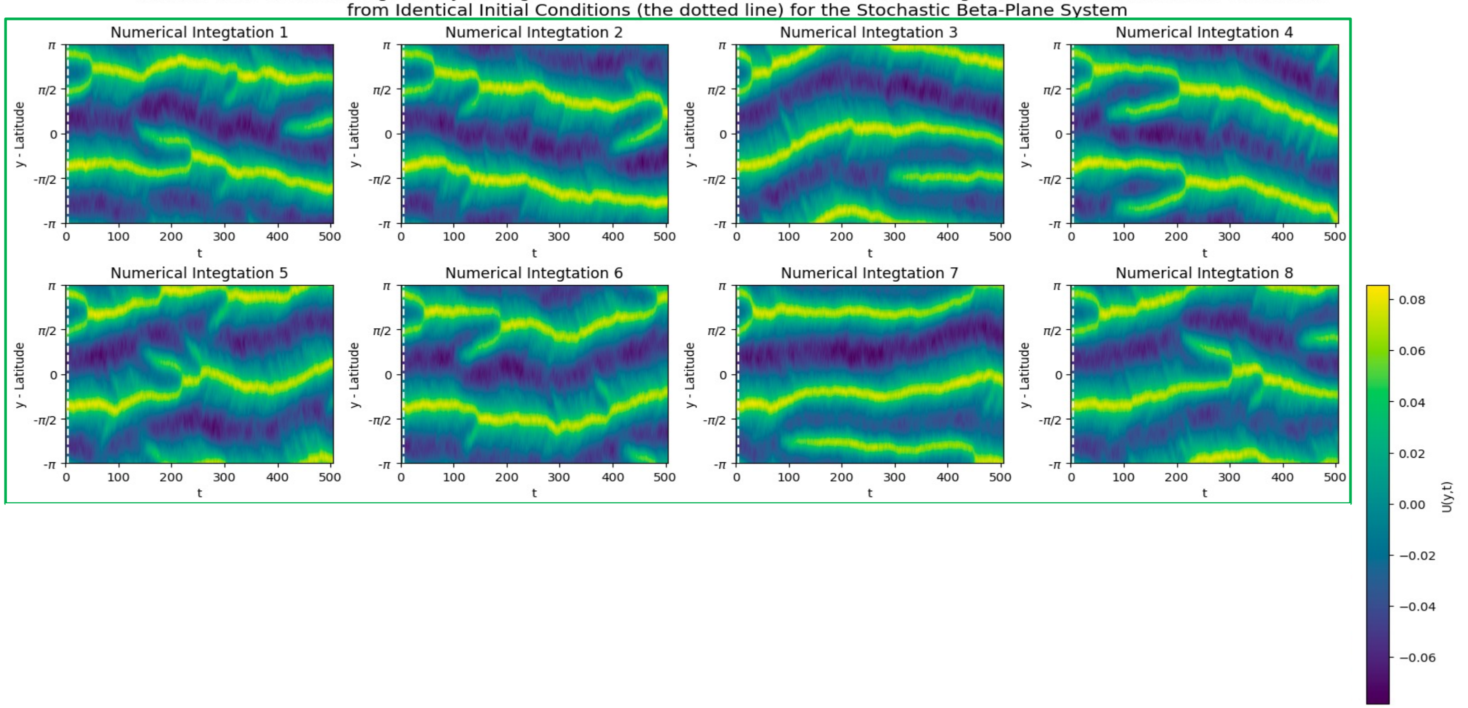
$$\hat{U}_t = \mathcal{D}_\phi[\mathcal{E}_\theta(U_t)] \qquad \tilde{U}_{t+1} = \mathcal{D}_\phi\{\{\mathcal{T}_\varphi[\mathcal{E}_\theta(U_{t:t-l}),\epsilon]\}$$



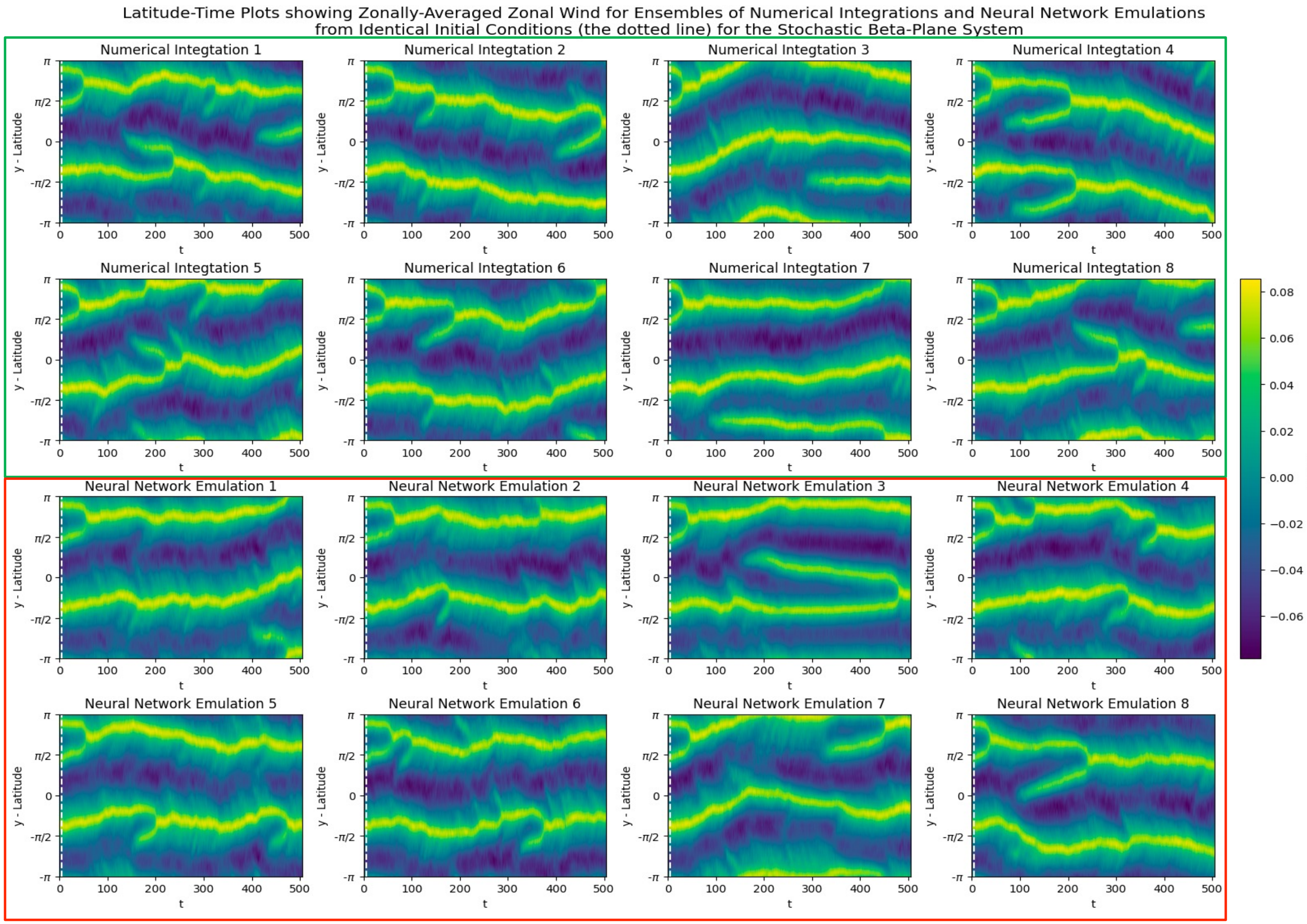
[3] Matheson et al, Management Science (1976) , [4] Gneiting et al. doi: 10.1198/016214506000001437 (2012)
[5] Pacchiardi et al. doi: 10.48550/arXiv.2112.08217 (2022)

Results - Emulations

Latitude-Time Plots showing Zonally-Averaged Zonal Wind for Ensembles of Numerical Integrations and Neural Network Emulations from Identical Initial Conditions (the dotted line) for the Stochastic Beta-Plane System



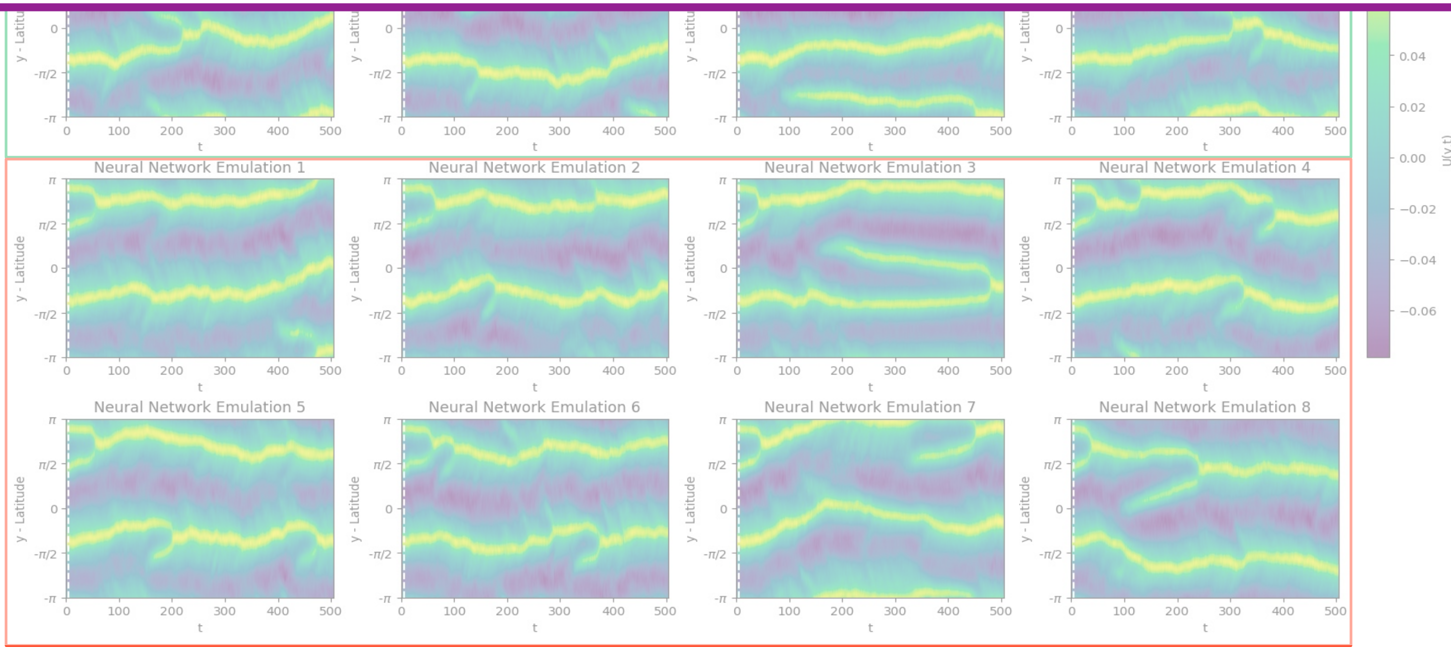
Results - Emulations



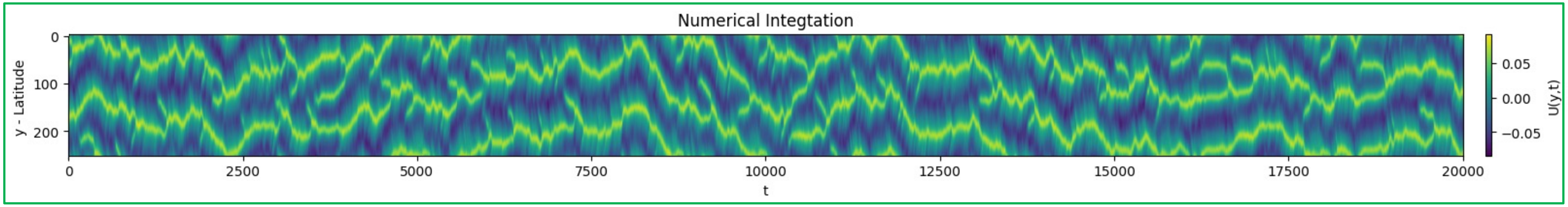
Results - Emulations

- Time to generate 500 time-steps using numerical integration*: **~180 minutes**
- Time to generate 500 time-steps using Deep Learning: **~ 5.1 milliseconds**
- Speed-up factor: **~2,112,000**

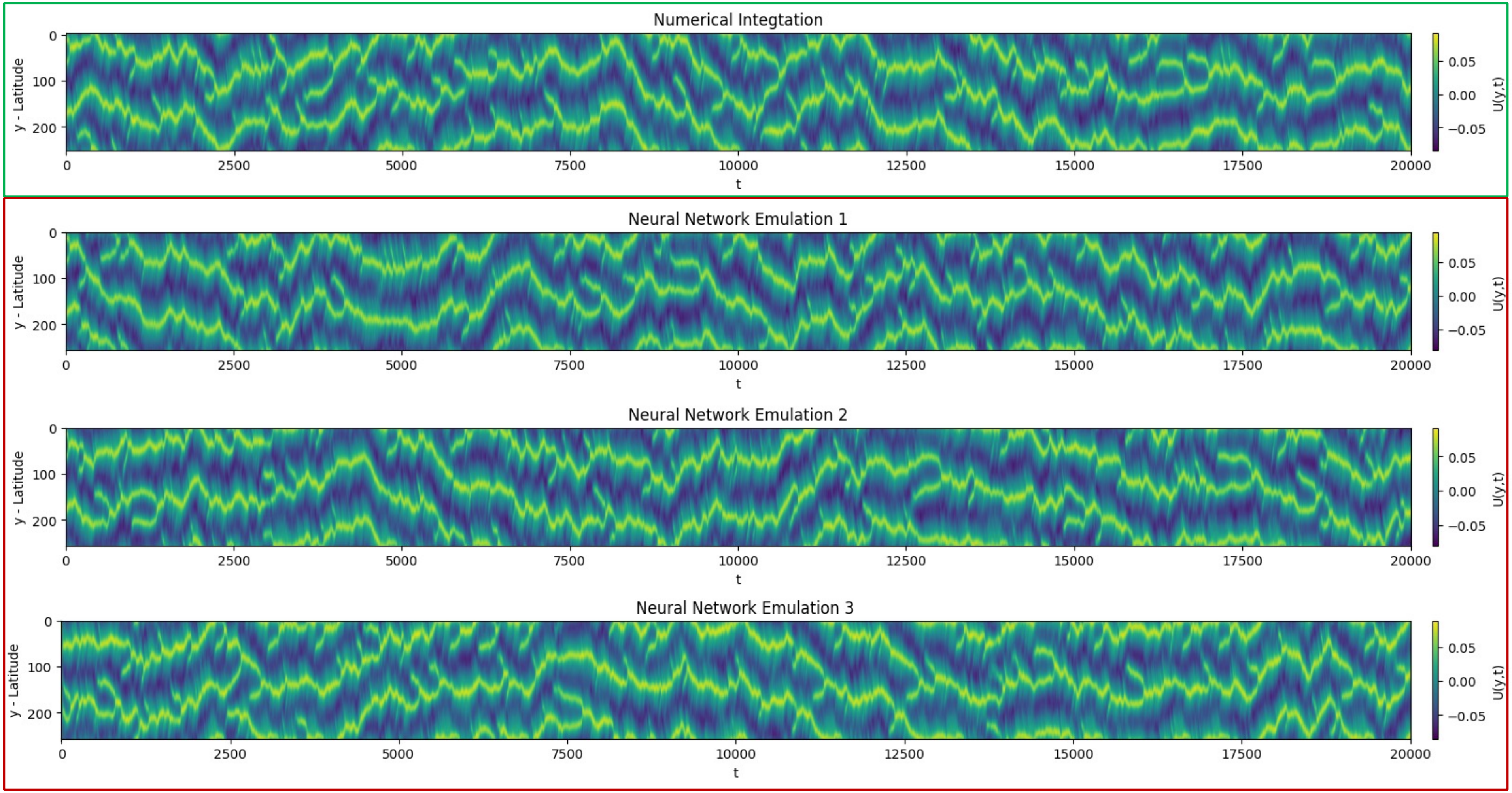
**system solved with time discretisation of $2.5x10^{3}$ per output step*



Stability - Long Time Emulations



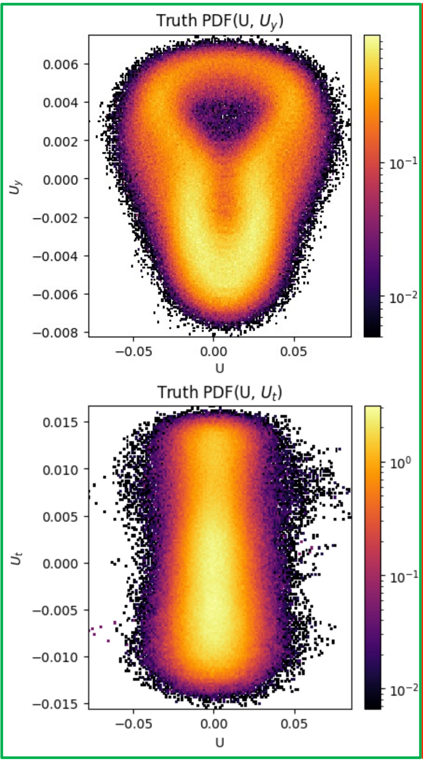
Stability - Long Time Emulations

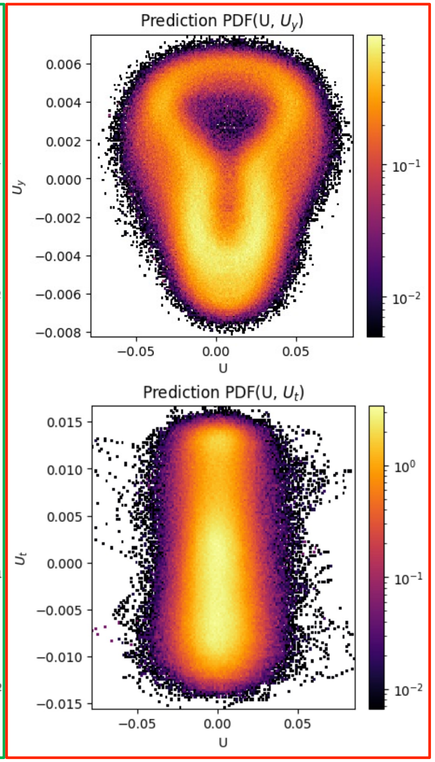


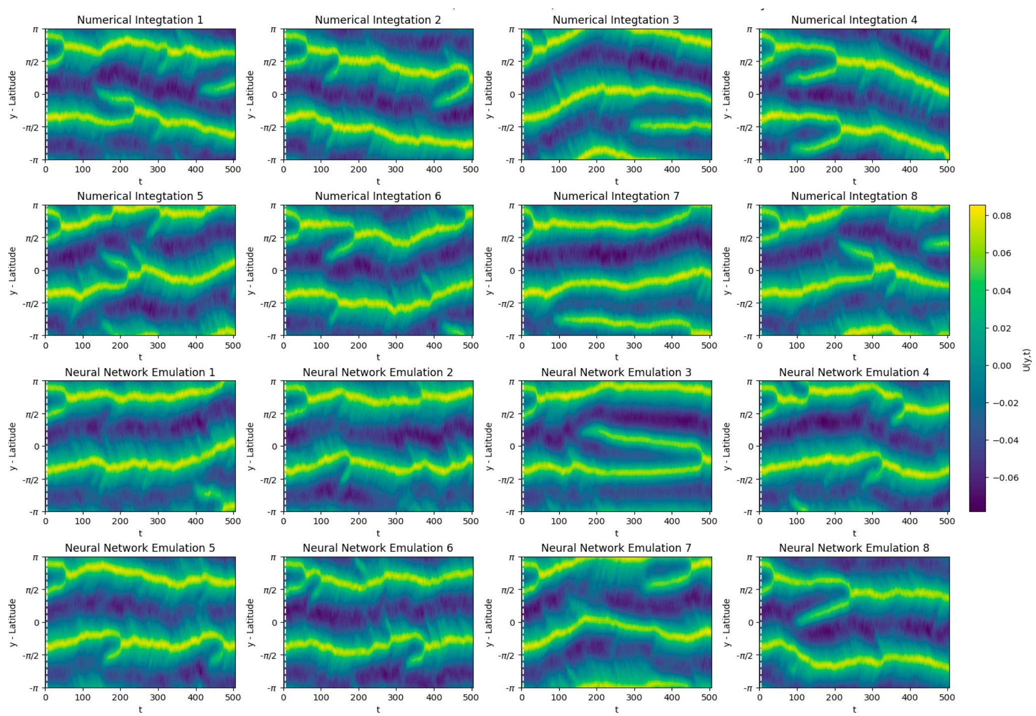
Evaluation – PDF of Temporal and Spatial Derivates

$P(U, Uy, Ut)$

$P(\widetilde{U}, \widetilde{U}_y, \widetilde{U}_t)$

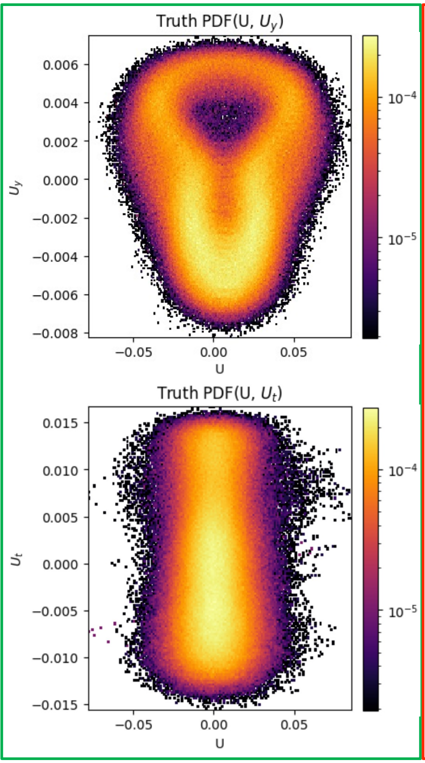




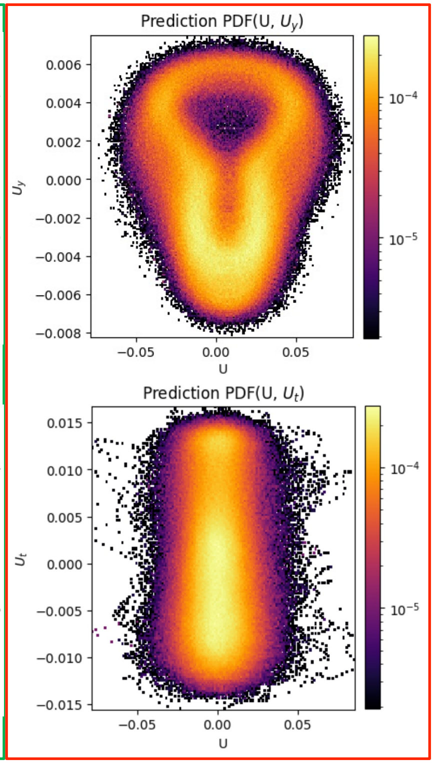


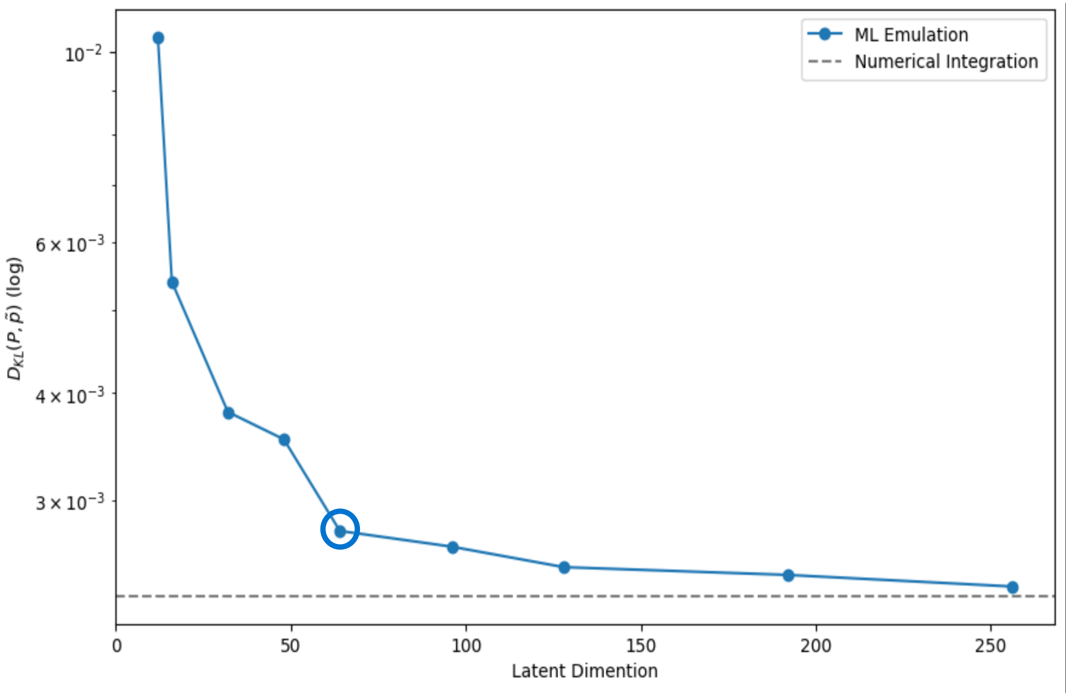
Evaluation – PDF of Temporal and Spatial Derivates

$P(U, Uy, Ut)$

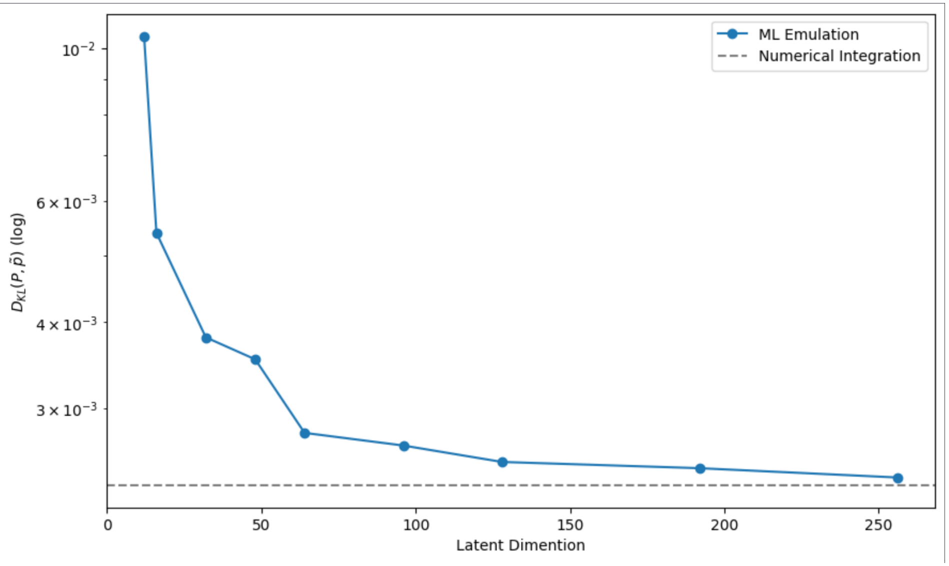


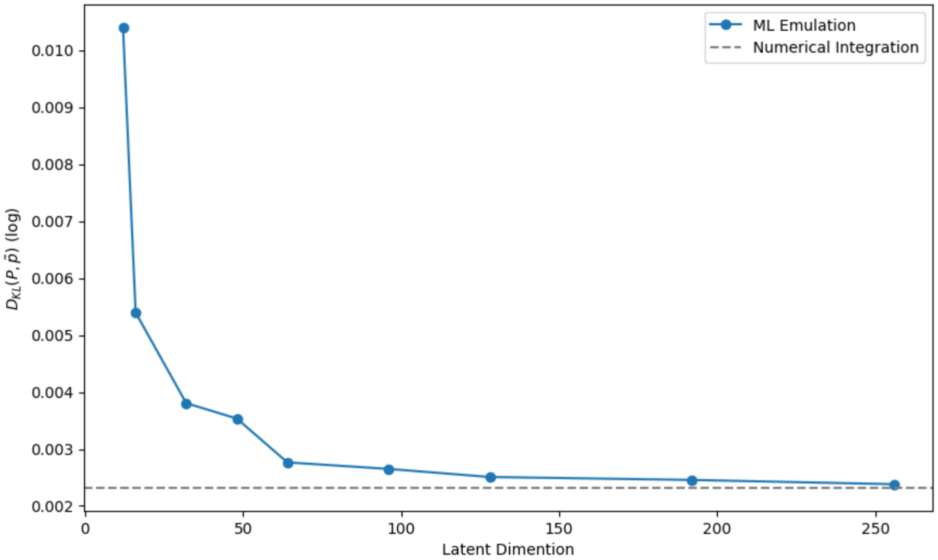
$P(\widetilde{U}, \widetilde{U}_y, \widetilde{U}_t)$





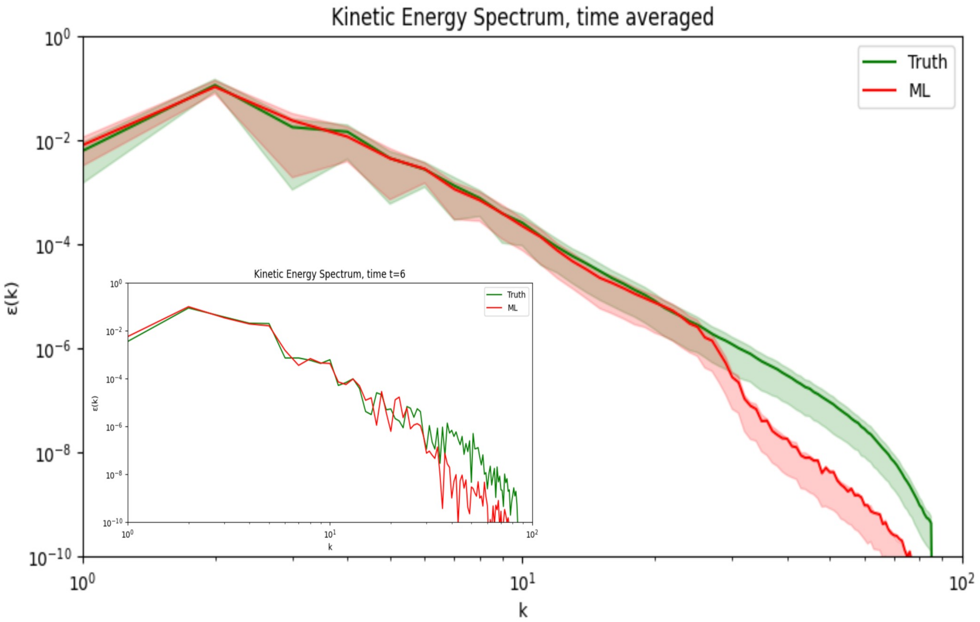
Evaluation – PDF of Temporal and Spatial Derivates





Evaluation – Energy Spectra and Jet Frequency

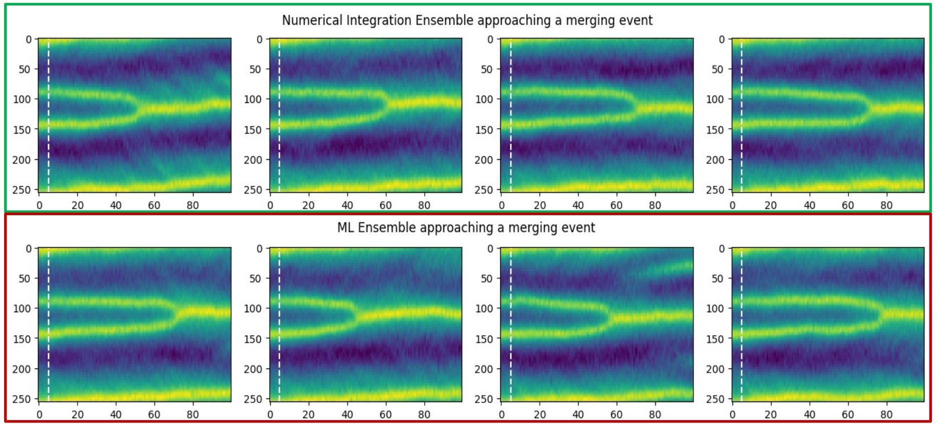
Comparing instantaneous and time-averaged energy spectra.

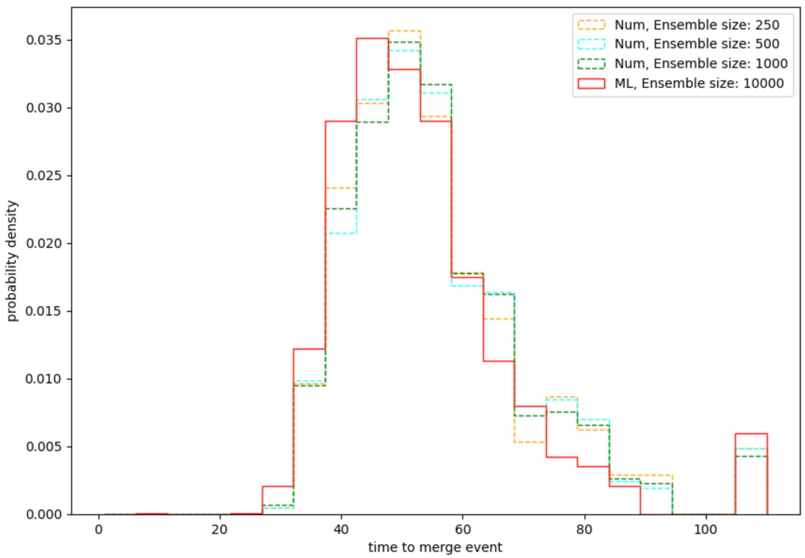


Spectral Bias in Generative Models[6]

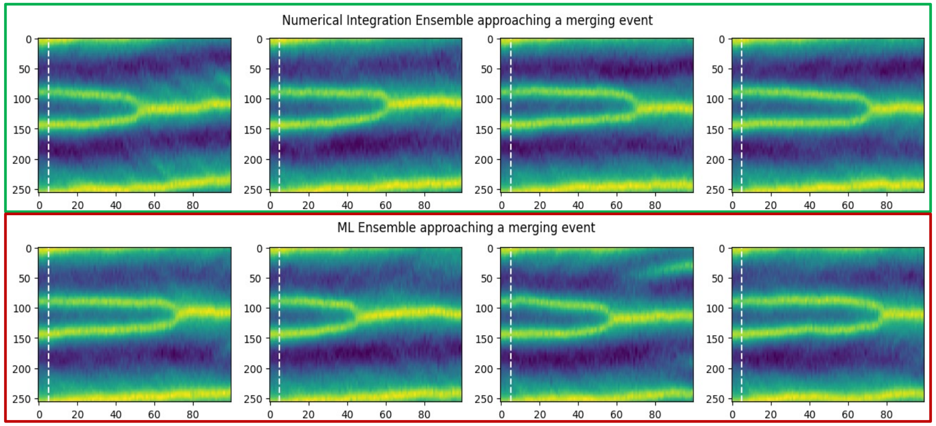
[6] Schwarz et al. NeurIPS (2022)

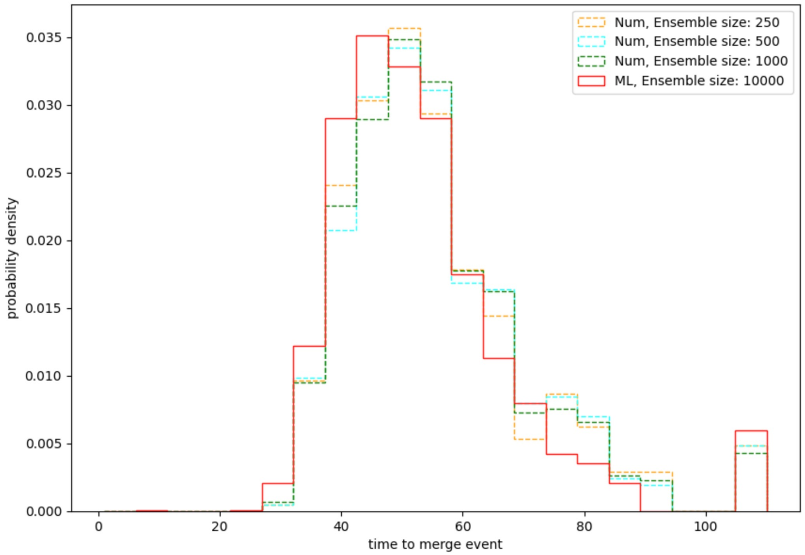
Evaluation – Time to Merging Event Distributions

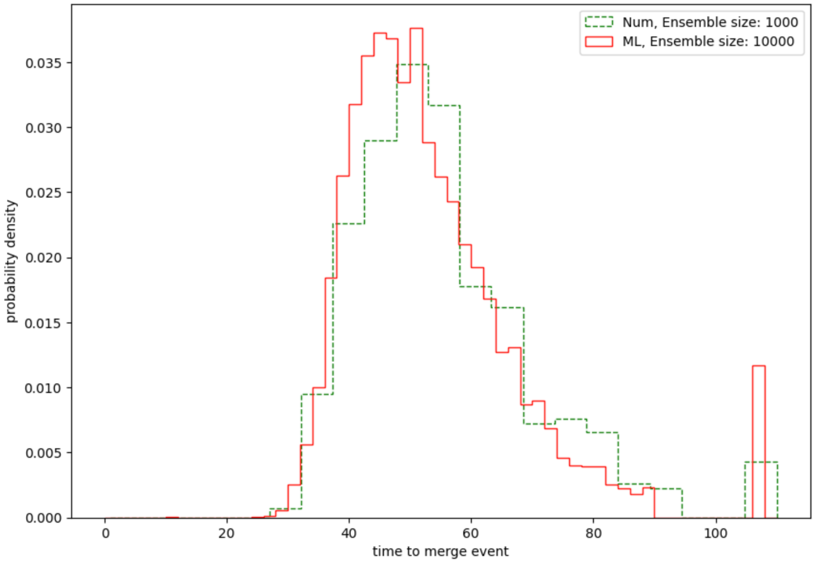




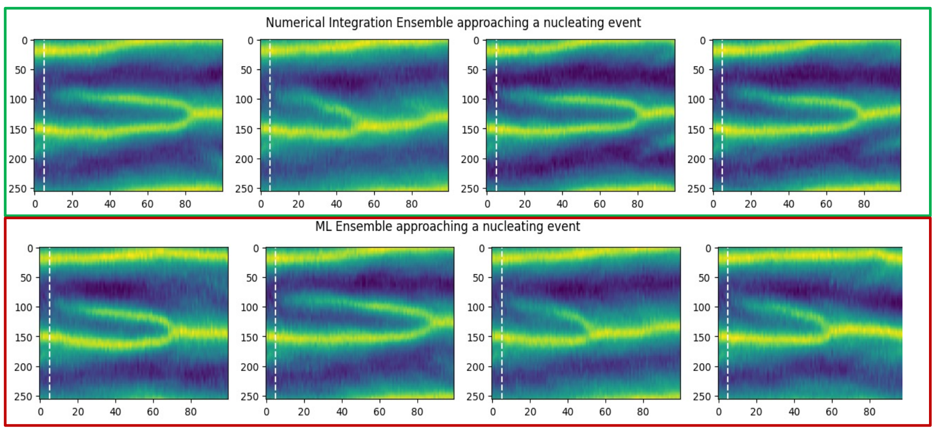
Evaluation – Time to Merging Event Distributions

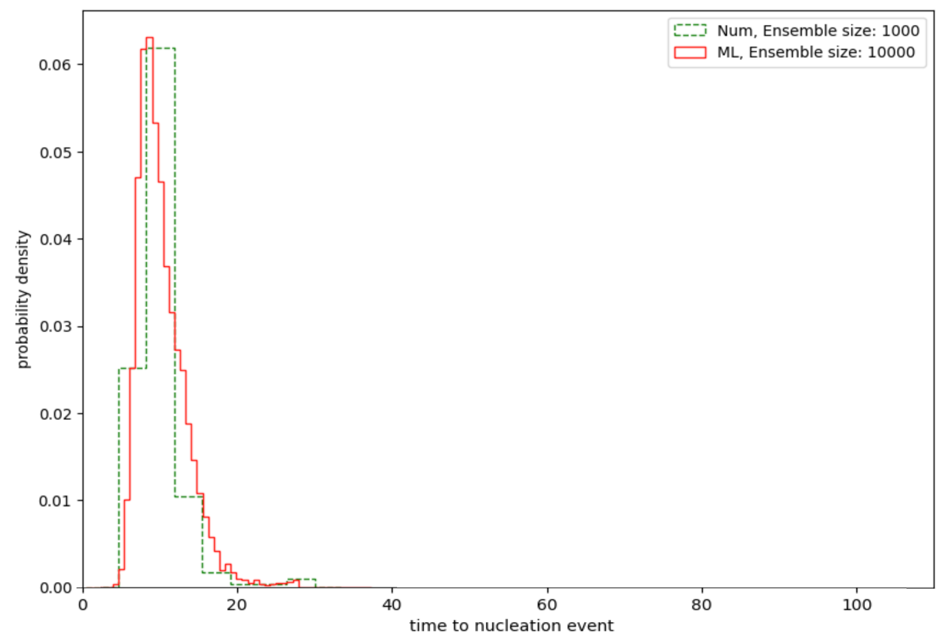






Evaluation – Time to Nucleating Event Distributions





Next Steps

Exciting Questions

- Can we quantify if some states are more ‘stable’ than others?
- What can the latent representation tell us about the dynamics of the system?
- What can the latent representation tell us about how the ML model has learned the system?
- What can the transformer attention weights tell us?

Future Applications

- Move to the 2-D case to model $u(x, y, t) = U + u'$, or model u', as a parameterisation.
- Model a two-layer/ multi-layer system in z that generates its own turbulence without requiring of stochastic forcing?
  - Will this now chaotic 3-D model still be best captured by a probabilistic ML approach despite being deterministic?
- Ultimately, model more complex/realistic geophysical processes, eg. parameterisations used in operational forecasts to reduce their computational cost.