# Everything flows

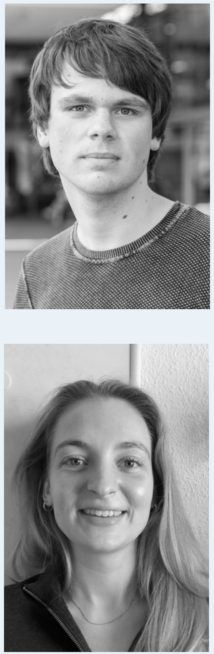
## Bridging fluid flows and data assimilation through entropy and stochasticity

Benjamin Sanderse

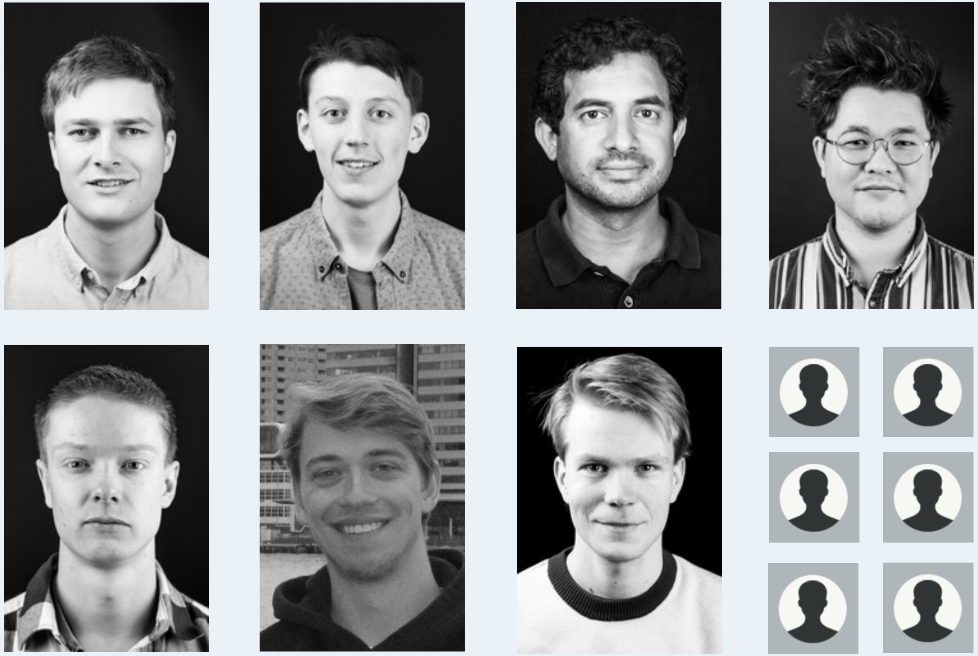
IPAM, April 16, 2026

CWI TU/e

# The Scientific Computing group @CWI



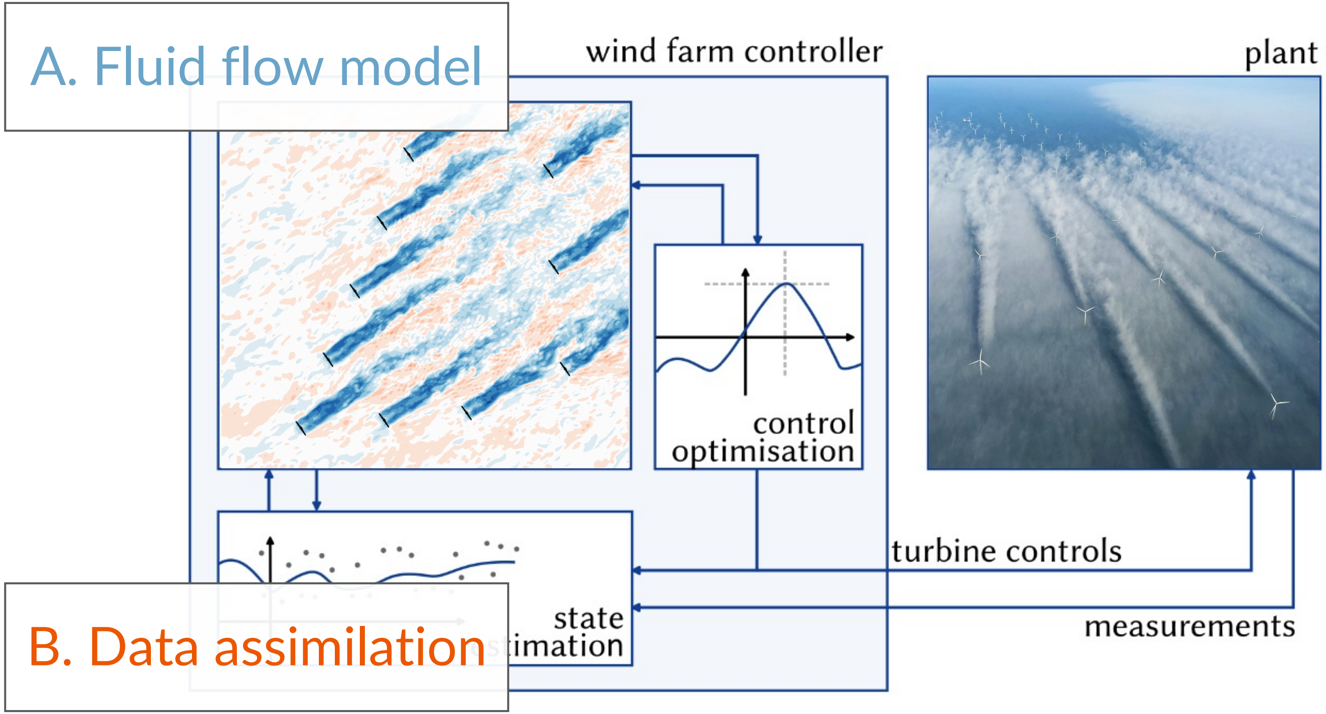
MSc students



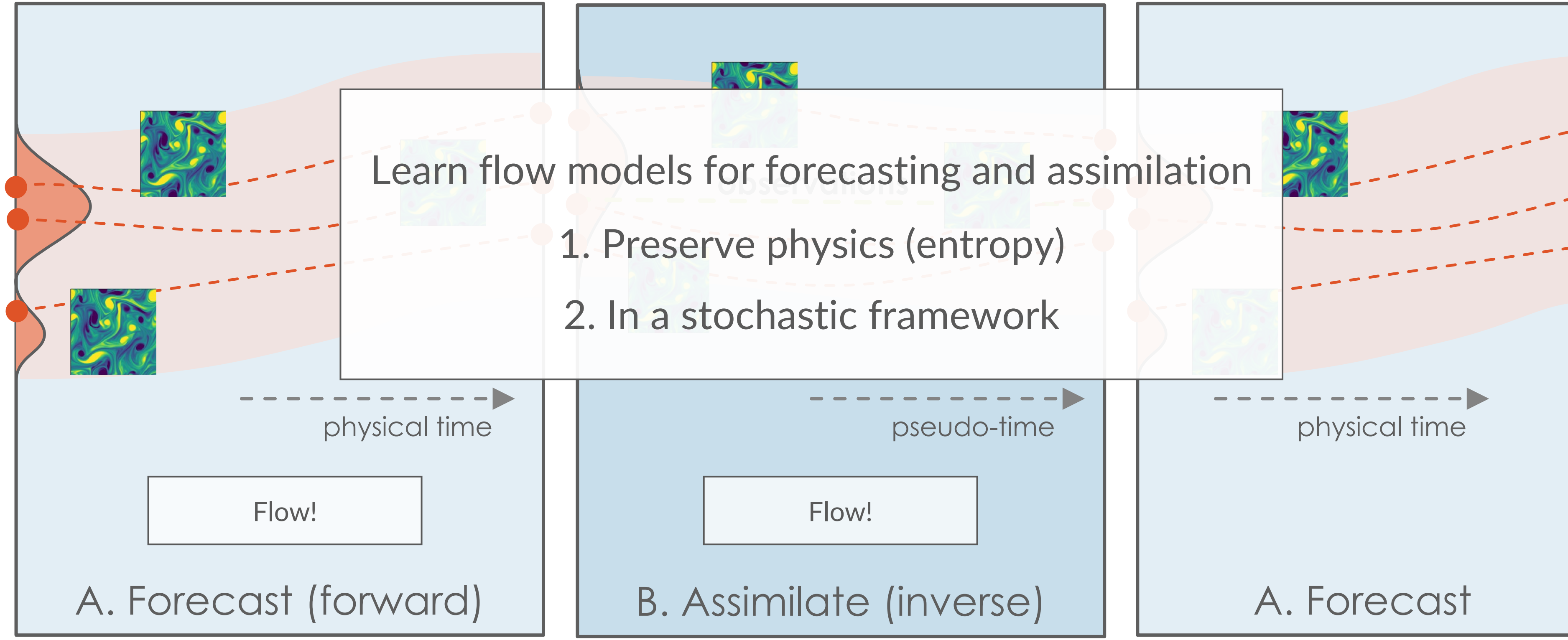
PhD candidates & postdocs

Staff

# Fluid flows in energy applications



# Summary: probabilistic forecasting and assimilation




Learn flow models for forecasting and assimilation

1. Preserve physics (entropy)

2. In a stochastic framework

## A. Fluid flow model

**True dynamics**

Navier-Stokes (non-linear PDE)

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho\boldsymbol{v}\\ E\end{pmatrix} + \nabla\cdot\begin{pmatrix}\rho\boldsymbol{v}\\ p\boldsymbol{I}+\rho\boldsymbol{v}\otimes\boldsymbol{v}\\ \boldsymbol{v}(E+p)\end{pmatrix} = \nabla\cdot\begin{pmatrix}0\\ \boldsymbol{\tau}\\ \boldsymbol{\tau}\cdot\boldsymbol{v}+\boldsymbol{q}\end{pmatrix}$$

Reduce dimension:
1. Discretize
2. Filter: remove small scales
3. Close (e.g. neural net)

**Learnt model**

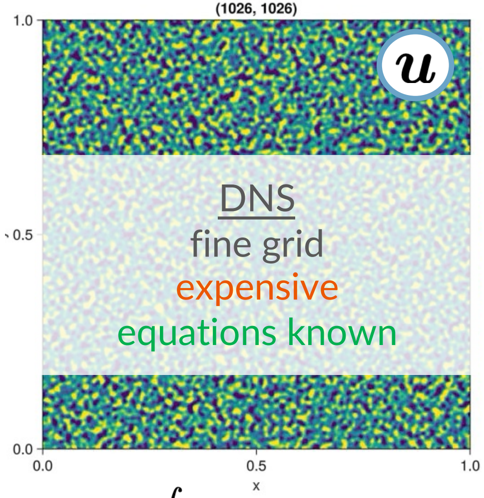
Neural LES model

$$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}t} = \boldsymbol{f}(\bar{\boldsymbol{u}}) + \boldsymbol{m}(\bar{\boldsymbol{u}};\theta)$$

**Structure (entropy)**

Stable and physical solutions

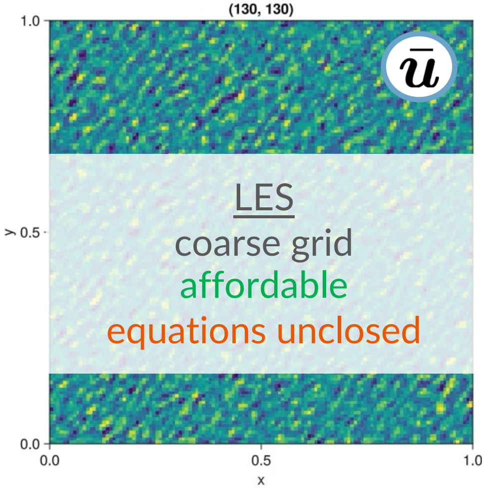
$$s = -\rho\ln\left(\frac{p}{\rho^{\gamma}}\right) \qquad \frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega} s\,\mathrm{d}\Omega \leq 0$$




$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{u})$$

~ $10^9$ DOF

$$\bar{\boldsymbol{u}} = \int \boldsymbol{u}(\xi,t)G(x,\xi)\mathrm{d}\xi$$




$$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}t} = \overline{\boldsymbol{f}(\boldsymbol{u})} \neq \boldsymbol{f}(\bar{\boldsymbol{u}})$$

# The closure problem - interpretations

DNS: $\boldsymbol{u}$

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{u})$$

LES: $\bar{\boldsymbol{u}}$

$$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}t} = \boldsymbol{f}(\bar{\boldsymbol{u}}) + \boldsymbol{m}(\bar{\boldsymbol{u}};\theta)$$

$$\boldsymbol{m}(\bar{\boldsymbol{u}};\theta) \approx \mathcal{C}(\bar{\boldsymbol{u}},\boldsymbol{u}) = \overline{\nabla\cdot(\boldsymbol{u}\otimes\boldsymbol{u})} - \nabla\cdot(\bar{\boldsymbol{u}}\otimes\bar{\boldsymbol{u}})$$

- **Learning missing physics** (model discovery, model error)

- **PDE-constrained optimization problem**: minimize loss function s.t. PDE holds

$$\mathcal{L}_{\boldsymbol{\theta}}^{\text{prio}}(\bar{\boldsymbol{u}}^*,\boldsymbol{u}^*) = \|\boldsymbol{m}(\bar{\boldsymbol{u}}^*;\theta) - \boldsymbol{C}(\boldsymbol{u}^*,\bar{\boldsymbol{u}}^*)\|^2 \qquad \mathcal{L}_{\boldsymbol{\theta}}^{\text{post}}(\bar{\boldsymbol{u}}_{\boldsymbol{\theta}},\bar{\boldsymbol{u}}^*) = \|\bar{\boldsymbol{u}}_{\boldsymbol{\theta}} - \bar{\boldsymbol{u}}^*\|^2$$

- **Inverse problem**: non-uniqueness, ill-posed

# SciML approaches for closure models

Foundations of Data Science

doi:10.3934/fods.2024043

## SCIENTIFIC MACHINE LEARNING FOR CLOSURE MODELS IN MULTISCALE PROBLEMS: A REVIEW

Benjamin Sanderse*[1], Panos Stinis[2], Romit Maulik[3] and Shady E. Ahmed[2]

[1]Scientific Computing Group, Centrum Wiskunde & Informatica, Science Park 123, Amsterdam, The Netherlands

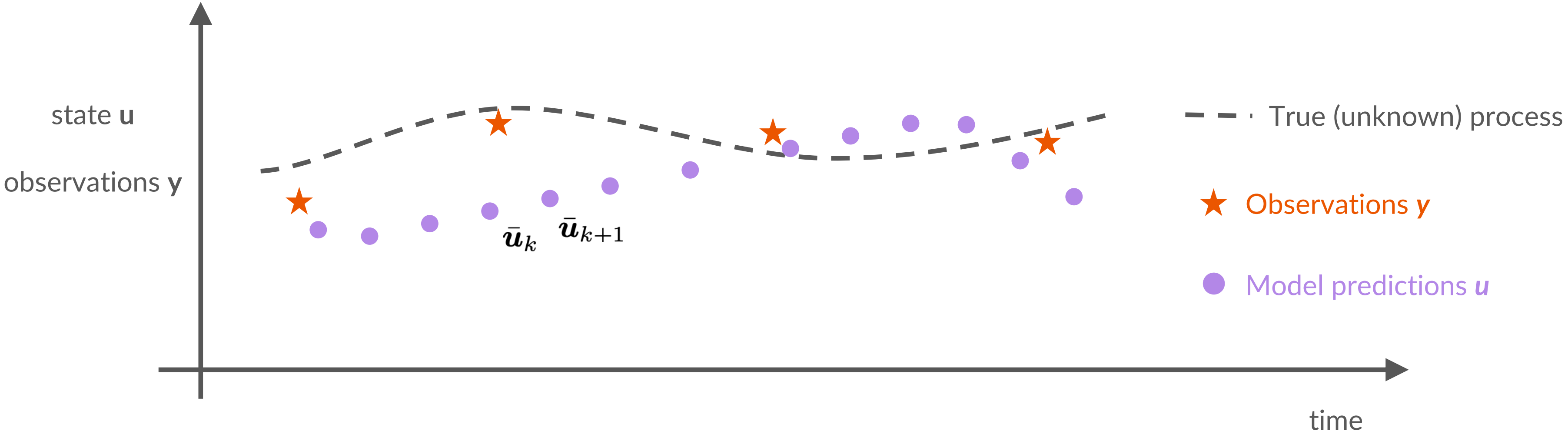
[2]Advanced Computing, Mathematics and Data Division, Pacific Northwest National Laboratory, Richland, 99352, WA, USA

[3]College of Information Sciences and Technology, The Pennsylvania State University, Westgate Building, University Park, 16802, PA, USA

Abstract. Closure problems are omnipresent when simulating multiscale systems, where some quantities and processes cannot be fully prescribed despite their effects on the simulation's accuracy. Recently, scientific machine learning approaches have been proposed as a way to tackle the closure problem, combining traditional (physics-based) modeling with data-driven (machine-learned) techniques, typically through enriching differential equations with neural networks. This paper reviews the different reduced model forms, distinguished by the degree to which they include known physics, and the different objectives of a priori and a posteriori learning. The importance of adhering to physical laws (such as symmetries and conservation laws) in choosing the reduced model form and choosing the learning method is discussed. The effect of spatial and temporal discretization and recent trends toward discretization-invariant models are reviewed. In addition, we make the connections between closure problems and several other research disciplines: inverse problems, Mori-Zwanzig theory, and multi-fidelity methods. In conclusion, much progress has been made with scientific machine learning approaches for solving closure problems, but many challenges remain. In particular, the generalizability and interpretability of learned models is a major issue that needs to be addressed further.

# Data assimilation

- The learned model $\boldsymbol{m}(\bar{\boldsymbol{u}};\theta)$ is trained on limited "offline" DNS data and needs calibration
- The state might need corrections, too



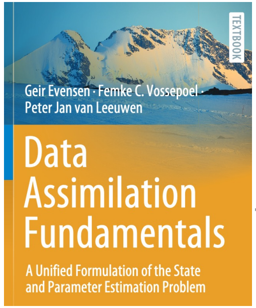
# Data assimilation

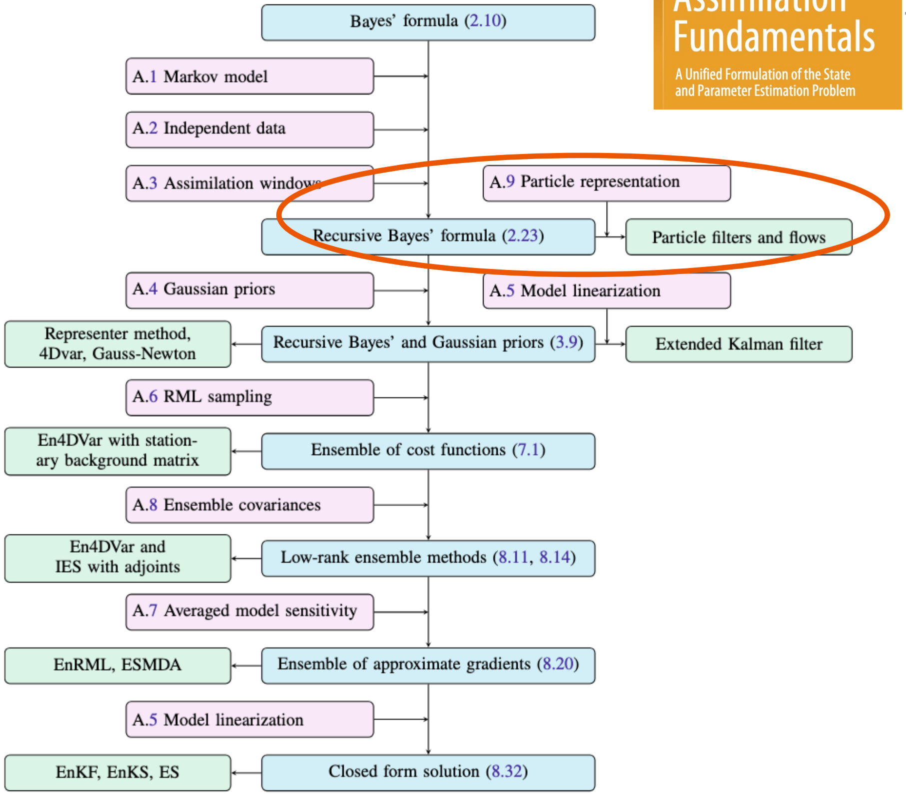
- Interested in **filtering distribution:**

$$p(\bar{\boldsymbol{u}}^{n+1}|\boldsymbol{y}^{1:n+1}) \propto p(\boldsymbol{y}^{n+1}|\bar{\boldsymbol{u}}^{n+1})p(\bar{\boldsymbol{u}}^{n+1}|\boldsymbol{y}^{1:n})$$

- Data assimilation = Bayesian inference

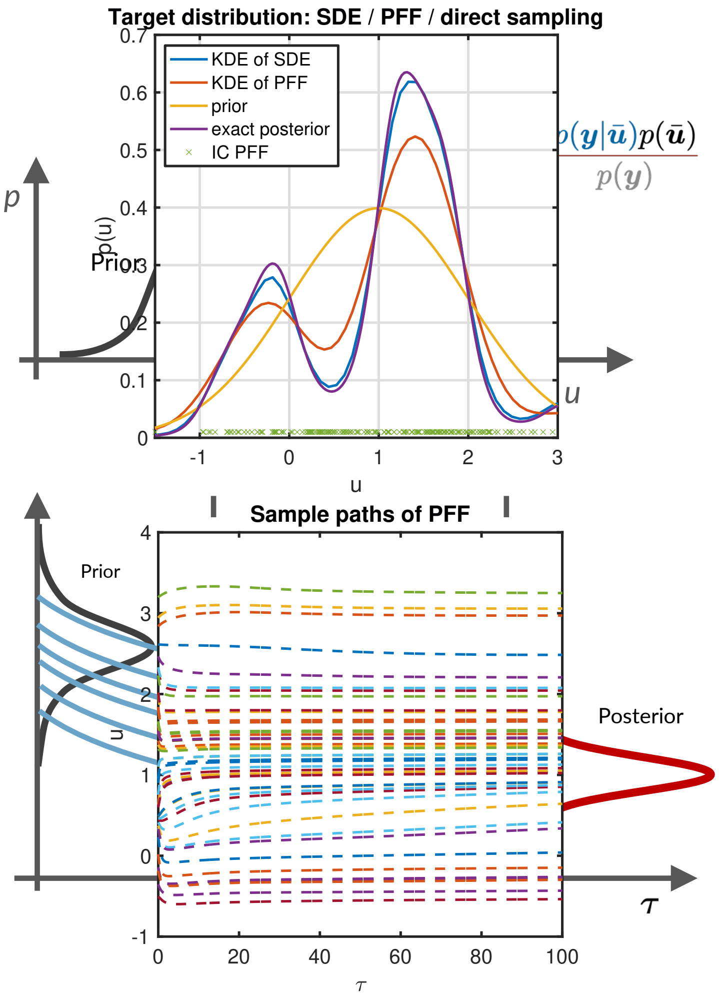
- Use particles, solve Bayes in pseudo-time

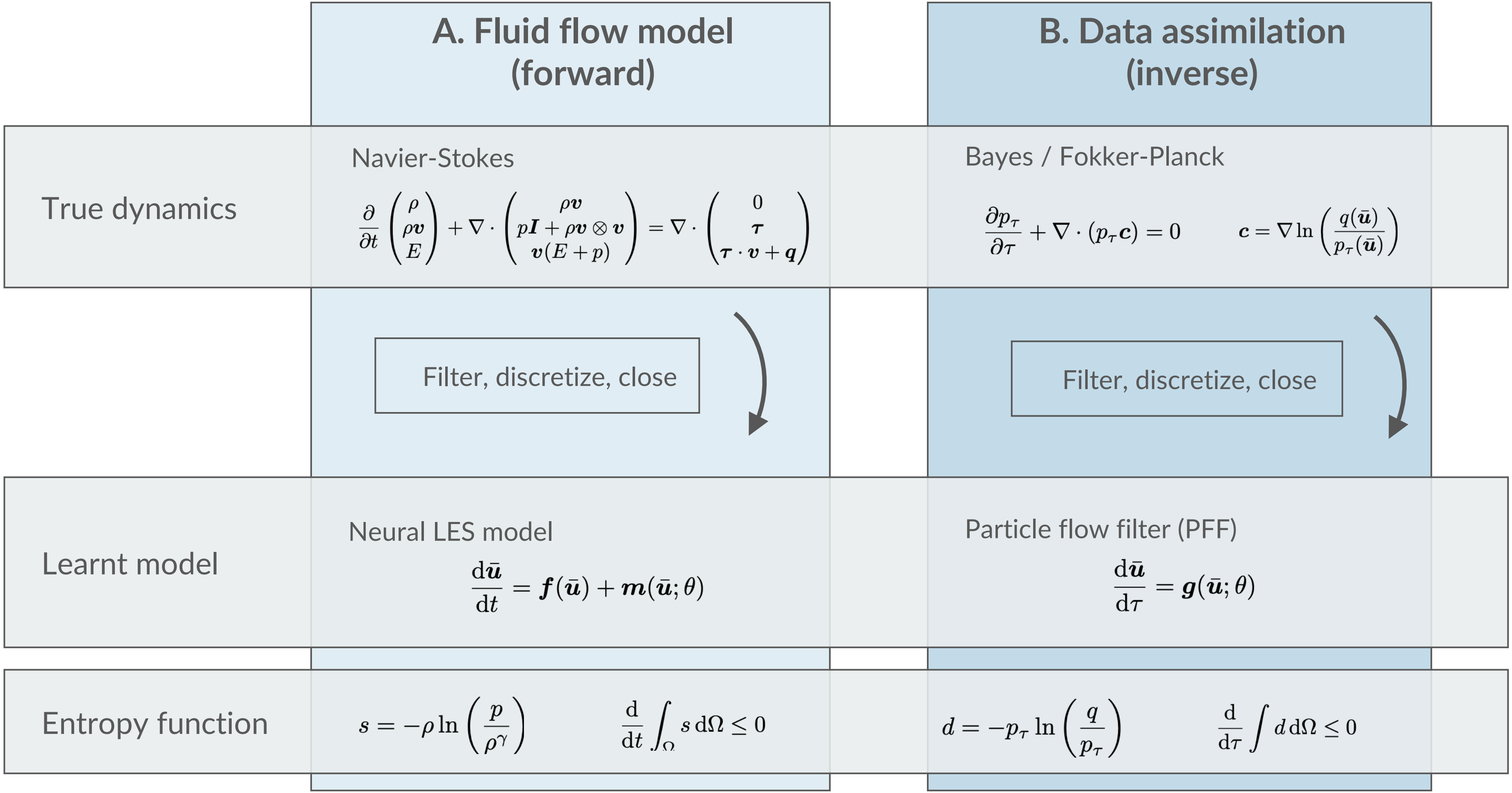


# B. Data assimilation

| | |
|---|---|
| True dynamics | Bayes -> Fokker-Planck (linear!) $$\frac{\partial p_\tau}{\partial \tau} + \nabla \cdot (p_\tau \boldsymbol{c}) = 0 \qquad \boldsymbol{c} = \nabla \ln \left( \frac{q(\bar{\boldsymbol{u}})}{p_\tau(\bar{\boldsymbol{u}})} \right)$$ |
| | Reduce dimension<br>1. Filtering formulation<br>2. Discretize (particles)<br>3. Close (kernel) |
| Learnt model | Particle flow filter, flow matching, neural ODE, … : $$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}\tau} = \boldsymbol{g}(\bar{\boldsymbol{u}}; \theta)$$ |
| Structure (entropy) | Distance to posterior (KL-divergence) $$d = -p_\tau \ln \left( \frac{q}{p_\tau} \right) \qquad \frac{\mathrm{d}D_{\mathrm{KL}}}{\mathrm{d}\tau} = -\int p_\tau \boldsymbol{c} \cdot \boldsymbol{c} \, \mathrm{d}\Omega \leq 0$$ |




A. Fluid flow model (forward)
B. Data assimilation (inverse)
True dynamics
Navier-Stokes
$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho\boldsymbol{v}\\ E\end{pmatrix} + \nabla\cdot\begin{pmatrix}\rho\boldsymbol{v}\\ p\boldsymbol{I}+\rho\boldsymbol{v}\otimes\boldsymbol{v}\\ \boldsymbol{v}(E+p)\end{pmatrix} = \nabla\cdot\begin{pmatrix}0\\ \boldsymbol{\tau}\\ \boldsymbol{\tau}\cdot\boldsymbol{v}+\boldsymbol{q}\end{pmatrix}$
Bayes / Fokker-Planck
$\frac{\partial p_\tau}{\partial \tau} + \nabla\cdot(p_\tau \boldsymbol{c}) = 0$
$\boldsymbol{c} = \nabla \ln\left(\frac{q(\bar{\boldsymbol{u}})}{p_\tau(\bar{\boldsymbol{u}})}\right)$
Filter, discretize, close
Filter, discretize, close
Learnt model
Neural LES model
$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}t} = \boldsymbol{f}(\bar{\boldsymbol{u}}) + \boldsymbol{m}(\bar{\boldsymbol{u}};\theta)$
Particle flow filter (PFF)
$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}\tau} = \boldsymbol{g}(\bar{\boldsymbol{u}};\theta)$
Entropy function
$s = -\rho \ln\left(\frac{p}{\rho^\gamma}\right)$
$\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega s\,\mathrm{d}\Omega \leq 0$
$d = -p_\tau \ln\left(\frac{q}{p_\tau}\right)$
$\frac{\mathrm{d}}{\mathrm{d}\tau}\int d\,\mathrm{d}\Omega \leq 0$

# Structure preservation: entropy

*"You should call it entropy, for two reasons.*
*In the first place your uncertainty function has been used in statistical mechanics under that name, so it already has a name.*

*In the second place, and more important,* ***no one really knows what entropy is, so in a debate you will always have the advantage****."*

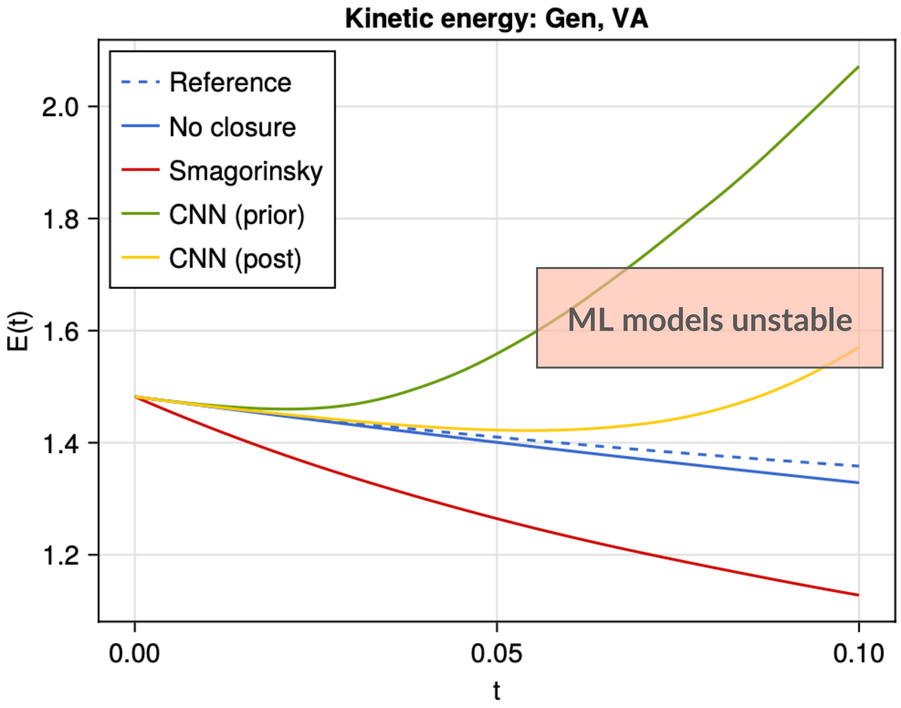
Von Neumann's advice to Shannon

# Neural closure models suffer from instability

LES: $$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}t} = \boldsymbol{f}(\bar{\boldsymbol{u}}) + \boldsymbol{m}(\bar{\boldsymbol{u}}; \theta)$$

Instabilities occur:

- even with a-posteriori training
- even with perfect (DNS) training data
- even without chaotic behavior



# Learning flow models

**Structure-preserving parameterization**

Discretization consistency

NN architectures that preserve symmetries

Hard constraints

**Probabilistic models via stochastic differential equations**

Change loss function: a-posteriori learning

Soft constraints

# Preserve structure: entropy

- Conservation law:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \frac{\partial \boldsymbol{f}(\boldsymbol{u})}{\partial x} = \frac{\partial \boldsymbol{f}^v(\boldsymbol{u})}{\partial x} \quad \times \eta(\boldsymbol{u}) = \frac{\partial s}{\partial \boldsymbol{u}}$$

- **Convex**
  - Pr

ility condition

Note "math - physics clash":

! Mathematical entropy **decreases**

! Energy can be an entropy function

$\int_\Omega$

- Conservation (inviscid):
- Dissipation (shocks/diffusion):

$$\frac{dS[\boldsymbol{u}]}{dt} = 0$$

$$\frac{dS[\boldsymbol{u}]}{dt} \leq 0$$

$$S[\boldsymbol{u}] := \int_\Omega s(\boldsymbol{u})dx$$

# Example: compressible Euler

$$\sigma(\boldsymbol{u}) = \ln(p/\rho^\gamma)$$

$$p(\boldsymbol{u}) = \left(E - \frac{1}{2}u^2\right)(\gamma - 1)$$

- PDE (hyperbolic)

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho \\ \rho u \\ E\end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix}\rho u \\ \rho u^2 + p \\ (E+p)u\end{bmatrix} = 0$$

- Entropy function

$$s(\boldsymbol{u}) = \frac{-\rho\,\sigma}{\gamma - 1}$$

- Entropy variables

$$\eta(\boldsymbol{u}) = \frac{\partial s}{\partial \boldsymbol{u}} = \begin{bmatrix}\frac{\gamma-\sigma}{\gamma-1} - \frac{\rho u^2}{2p} \\ \rho u/p \\ -\rho/p\end{bmatrix}$$

- Global **entropy conservation**

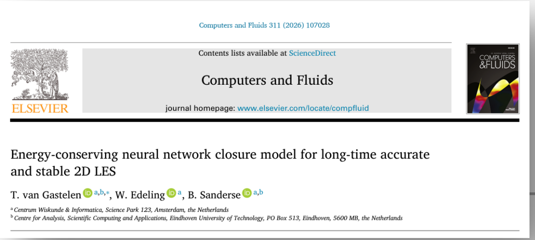
$$\frac{d\mathcal{S}[\boldsymbol{u}]}{dt} = \frac{d}{dt}\int_\Omega \frac{-\rho\,\sigma}{\gamma - 1}dx = 0$$

# Why entropy? (physics)

Stability

Physically admissible solutions

# Structure-preserving closure (1)


Computers and Fluids 311 (2026) 107028

Contents lists available at ScienceDirect

Computers and Fluids

journal homepage: www.elsevier.com/locate/compfluid

ELSEVIER

Energy-conserving neural network closure model for long-time accurate and stable 2D LES

T. van Gastelen[a,b,*], W. Edeling[a], B. Sanderse[a,b]

[a] Centrum Wiskunde & Informatica, Science Park 123, Amsterdam, the Netherlands

[b] Centre for Analysis, Scientific Computing and Applications, Eindhoven University of Technology, PO Box 513, Eindhoven, 5600 MB, the Netherlands

- LES: $$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}t} = \boldsymbol{f}(\bar{\boldsymbol{u}}) + \boldsymbol{m}(\bar{\boldsymbol{u}};\theta)$$
- New skew-symmetric **parameterization:**

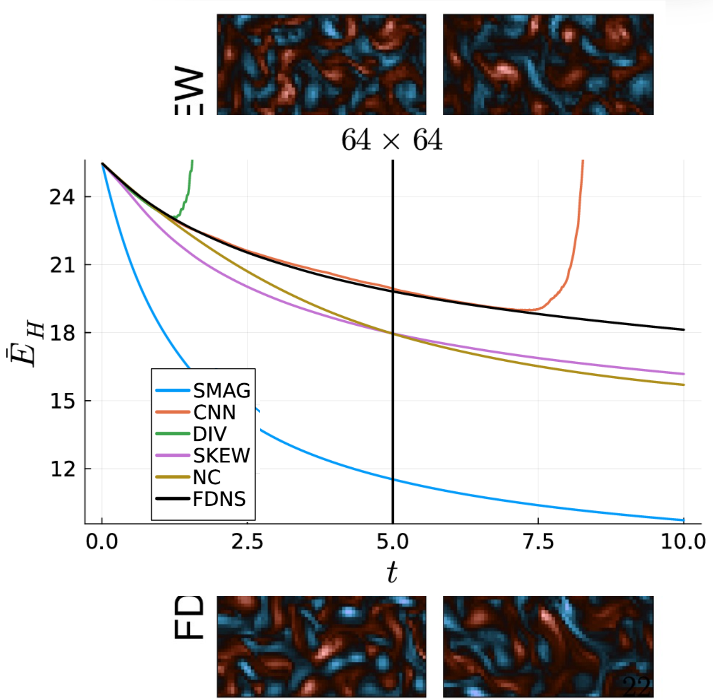
$$\boldsymbol{m}(\bar{\boldsymbol{u}};\theta) = (\boldsymbol{K} - \boldsymbol{K}^T)\bar{\boldsymbol{u}} - \boldsymbol{Q}^T\boldsymbol{Q}\bar{\boldsymbol{u}}$$

neural networks: $\boldsymbol{K} = \boldsymbol{K}(\bar{\boldsymbol{u}};\theta) \quad \boldsymbol{Q} = \boldsymbol{Q}(\bar{\boldsymbol{u}};\theta)$

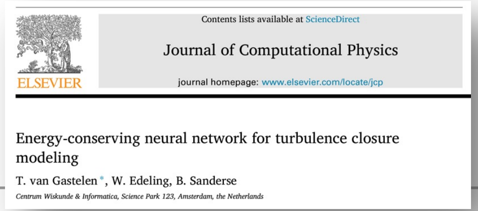
- Entropy stability guaranteed: $\bar{\boldsymbol{u}}^T\boldsymbol{m}(\bar{\boldsymbol{u}};\theta) \leq 0$

Compatibility condition satisfied by construction

- Similar parameterization possible for data assimilation (?)



# Structure-preserving closure (2)


Contents lists available at ScienceDirect

Journal of Computational Physics

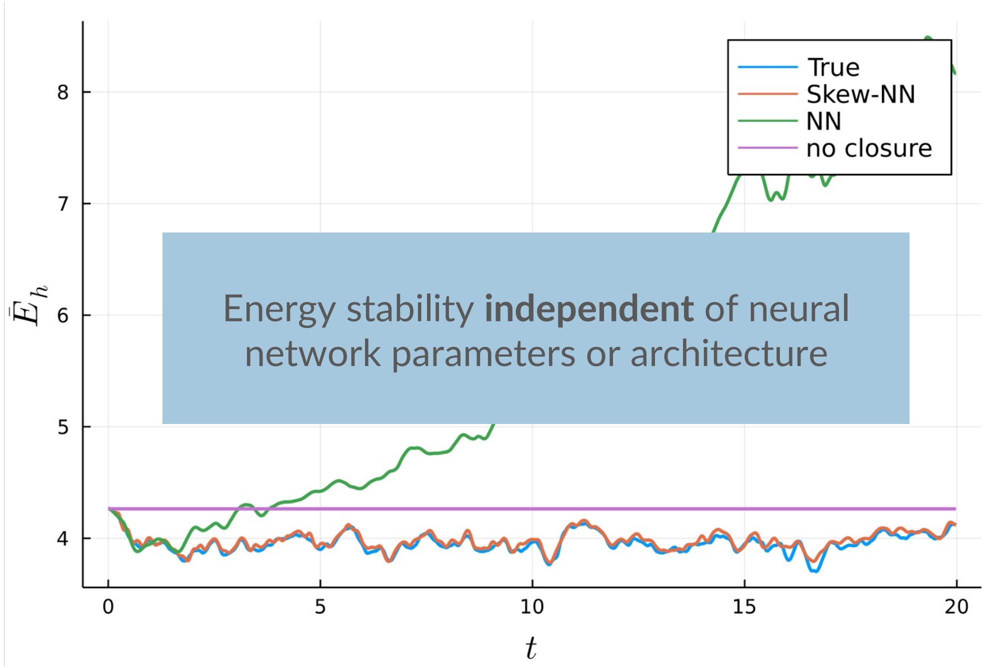
journal homepage: www.elsevier.com/locate/jcp

ELSEVIER

Energy-conserving neural network for turbulence closure modeling

T. van Gastelen *, W. Edeling, B. Sanderse

Centrum Wiskunde & Informatica, Science Park 123, Amsterdam, the Netherlands




Energy stability **independent** of neural network parameters or architecture

## Extended neural closure model

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\bar{\mathbf{v}}\\ \mathbf{s}\end{bmatrix} = \begin{bmatrix}\mathbf{f}(\bar{\mathbf{v}})\\ \mathbf{0}\end{bmatrix} + \begin{bmatrix}\mathbf{m}^v_\theta(\bar{\mathbf{v}},\mathbf{s})\\ \mathbf{m}^s_\theta(\bar{\mathbf{v}},\mathbf{s})\end{bmatrix}$$

$$\begin{bmatrix}\mathbf{m}^v_\theta(\bar{\mathbf{v}},\mathbf{s})\\ \mathbf{m}^v_\theta(\bar{\mathbf{v}},\mathbf{s})\end{bmatrix} = \mathcal{K}_\theta(\bar{\mathbf{v}},\mathbf{s})\begin{bmatrix}\bar{\mathbf{v}}\\ \mathbf{s}\end{bmatrix}$$

**Skew-symmetric** "matrix" $\mathcal{K}$ leads to exact energy conservation:

$$\frac{\mathrm{d}\bar{E}_h(\bar{\mathbf{v}})}{\mathrm{d}t} + \frac{\mathrm{d}\frac{1}{2}\|\mathbf{s}\|^2}{\mathrm{d}t} = 0$$

# Discretization matters!


Exact expressions for the unresolved stress in a finite-volume based large-eddy simulation

Syver Døving Agdestein[a,b,*], Roel Verstappen[c], Benjamin Sanderse[a,b]

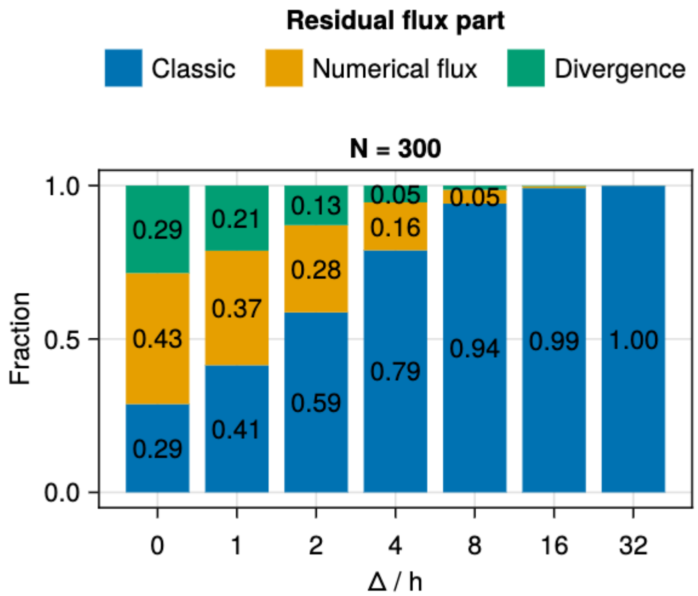
[a] Scientific Computing Group, Centrum Wiskunde & Informatica, Science Park 123, Amsterdam, 1098 XG, The Netherlands
[b] Centre for Analysis, Scientific Computing and Applications, Eindhoven University of Technology, PO Box 513, Eindhoven, 5600 MB, The Netherlands
[c] Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen , Nijenborgh 9, Groningen, 9747 AG, The Netherlands

- LES is a combined physical-numerical model with **two filters:**
  - A **physical** filter
  - A **discretization-induced** filter

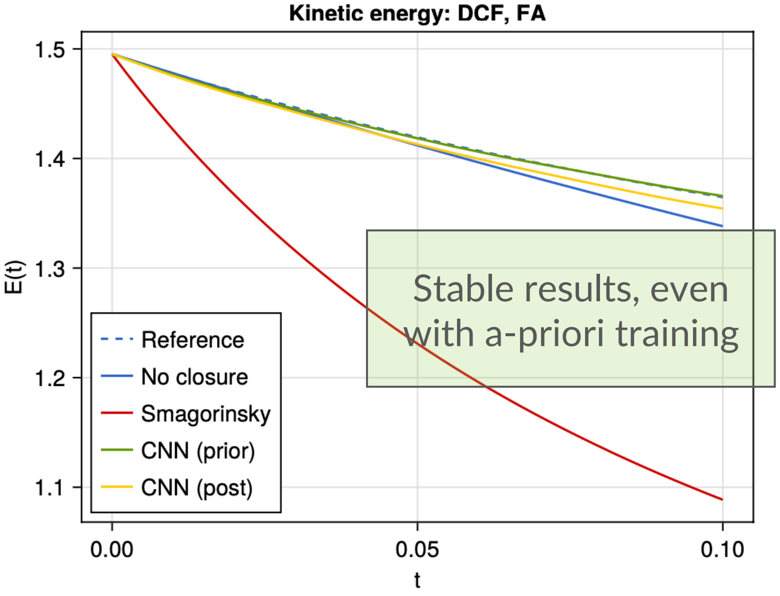
- The closure expression consists of **physical terms and discretization terms**

$$\boldsymbol{m}(\bar{\boldsymbol{u}}; \theta) \approx \mathcal{C}(\bar{\boldsymbol{u}}, \boldsymbol{u}) = \overline{\nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u})} - \nabla \cdot (\bar{\boldsymbol{u}} \otimes \bar{\boldsymbol{u}})$$

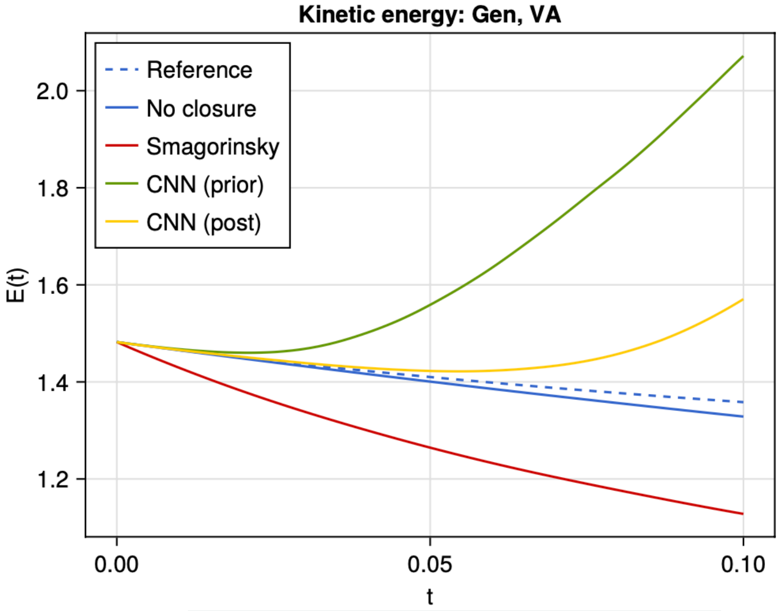
- A good closure model should incorporate both



# New discretization-consistent filter

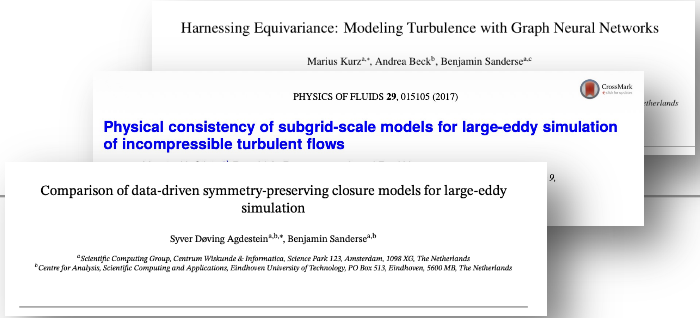



Discretization-consistent filter




Standard filter

# Other structure: symmetries


Harnessing Equivariance: Modeling Turbulence with Graph Neural Networks

Marius Kurz[a,*], Andrea Beck[b], Benjamin Sanderse[a,c]

PHYSICS OF FLUIDS 29, 015105 (2017)

Physical consistency of subgrid-scale models for large-eddy simulation of incompressible turbulent flows

Comparison of data-driven symmetry-preserving closure models for large-eddy simulation

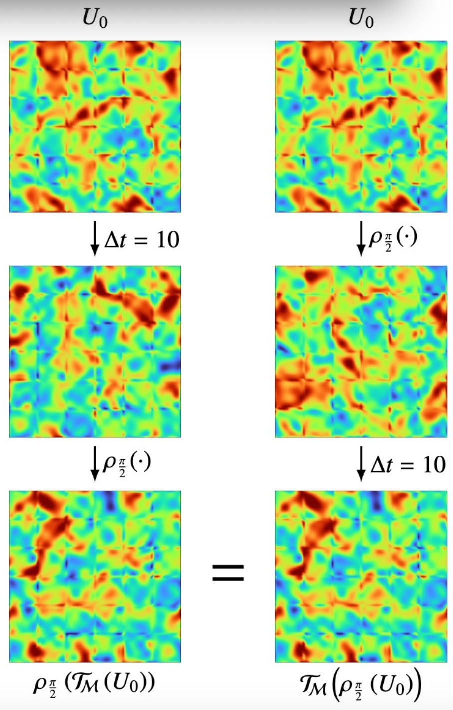
Syver Døving Agdestein[a,b,*], Benjamin Sanderse[a,b]

[a] Scientific Computing Group, Centrum Wiskunde & Informatica, Science Park 123, Amsterdam, 1098 XG, The Netherlands
[b] Centre for Analysis, Scientific Computing and Applications, Eindhoven University of Technology, PO Box 513, Eindhoven, 5600 MB, The Netherlands

- **Symmetries** can be incorporated into NNs:
  - Group-equivariant CNNs
  - Graph NNs
  - **Tensor-basis NNs:**

$$\boldsymbol{m}(\bar{\boldsymbol{u}};\theta) = \nabla \cdot \tau^{\mathrm{TB}}, \quad \tau^{\mathrm{TB}} = \Delta^2 \sum_k \alpha_k(\lambda)\mathbf{T}^{(k)}$$

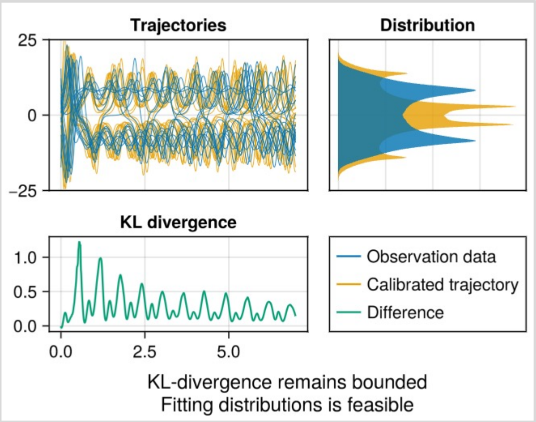
- **Many open questions**
  - Which symmetries are most important?
  - Continuous vs. discrete symmetries

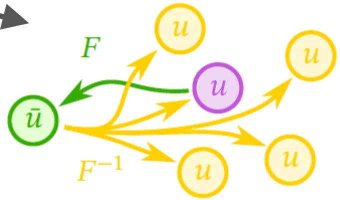


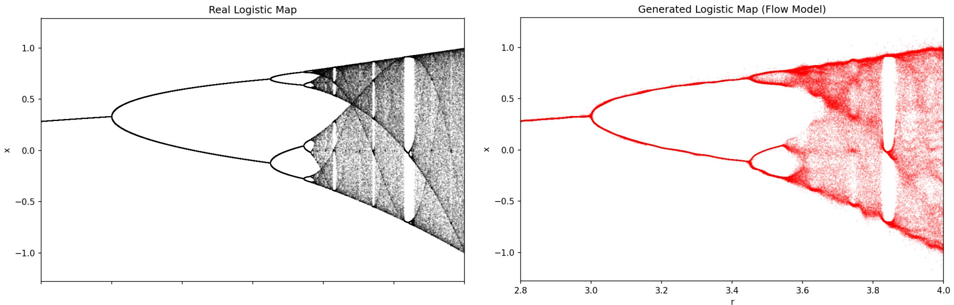
# Structure-preserving stochastic models

# Why stochastic?

- Turbulence is chaotic, interested in **statistics**
- Effect of **non-uniqueness**
- **Uncertainty quantification**
- Model **missing physics**
- Explore distributions, bifurcations



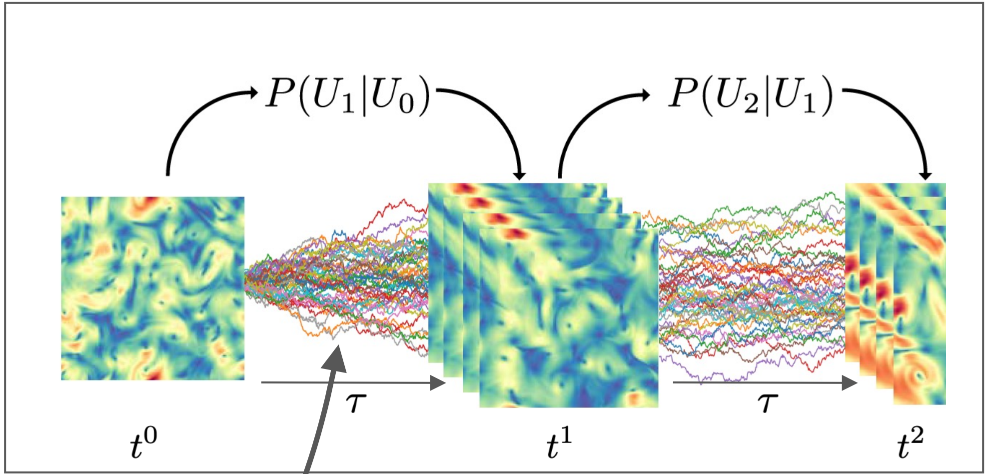
# Stochastic forecasting for turbulence

- Each time step, solve an SDE in pseudo-time
- Simulate **ensemble**
- Need to find drift + diffusion

Generative modeling

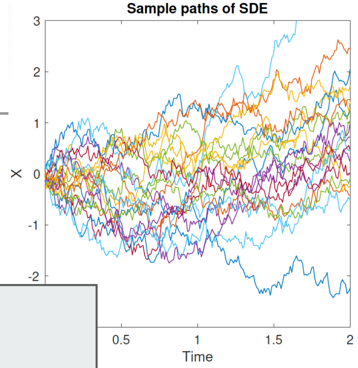
⇔

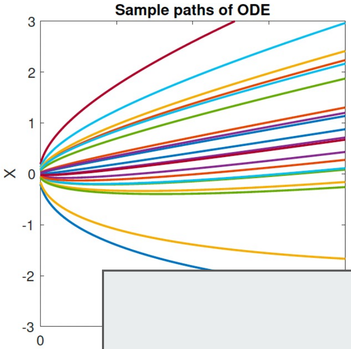
Finding SDEs




$$\mathrm{d}\bar{\boldsymbol{u}}_t = (\boldsymbol{f}(\bar{\boldsymbol{u}}_t) + \boldsymbol{m}_\theta(\bar{\boldsymbol{u}}_t))\mathrm{d}t + \sigma(t)\mathrm{d}\boldsymbol{W}_t$$

# The “trinity”




## SDE

$$\mathrm{d}\boldsymbol{x}_t = \boldsymbol{b}(\boldsymbol{x}_t)\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}_t$$

## ODE

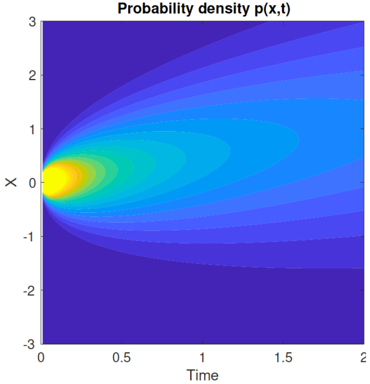
$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{c}(\boldsymbol{x}, t)$$

$$\boldsymbol{c}(\boldsymbol{x}, t) = \boldsymbol{b}(\boldsymbol{x}) - \frac{1}{2} g^2(t) \nabla_{\boldsymbol{x}} \ln p(\boldsymbol{x}, t)$$

implied distribution

## Fokker-Planck

$$\frac{\partial p(\boldsymbol{x}, \boldsymbol{t})}{\partial t} = -\nabla \cdot (\boldsymbol{b}(\boldsymbol{x}) p(\boldsymbol{x}, t)) + \nabla \cdot (\frac{1}{2} g^2(t) \nabla p(\boldsymbol{x}, t))$$



# Finding the drift in the SDE

Stochastic interpolants (SI) [1]:

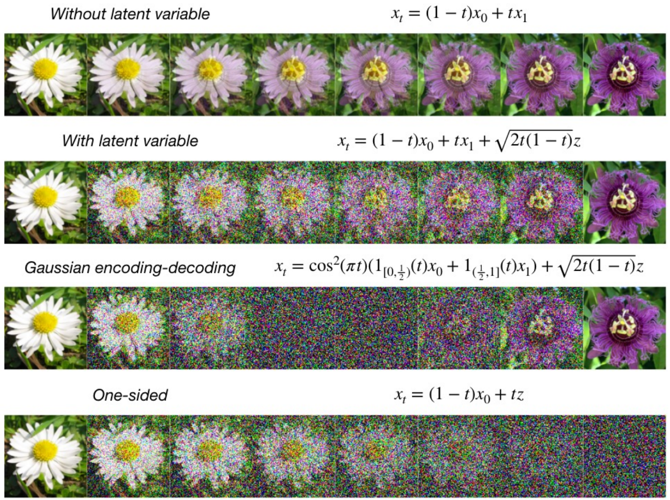
- Interpolate between flow fields
- Related to flow matching [2]

Finding the drift is **similar to a-priori training** (no adjoints etc.):

$$L(\theta) = \int_0^1 \mathbb{E}\left[ |\boldsymbol{b}_\theta(\boldsymbol{I}_\tau(\boldsymbol{x}_0, \boldsymbol{x}_1), \tau) - \boldsymbol{R}_\tau(\boldsymbol{x}_0, \boldsymbol{x}_1)|^2 \right] \mathrm{d}\tau$$

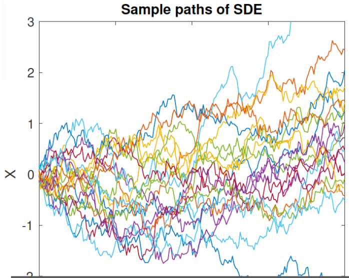
$$\boldsymbol{x}_\tau \approx \boldsymbol{I}_\tau(\boldsymbol{x}_0, \boldsymbol{x}_1) = \alpha_\tau \boldsymbol{x}_0 + \beta_\tau \boldsymbol{x}_1 + \gamma_\tau \boldsymbol{W}_\tau$$

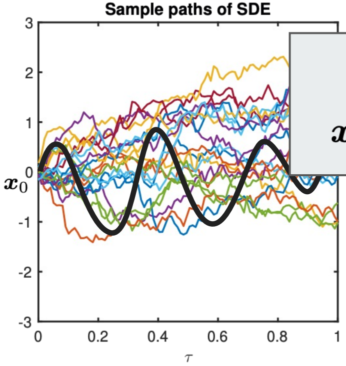
$$\mathrm{d}\bar{\boldsymbol{u}}_t = (\boldsymbol{f}(\bar{\boldsymbol{u}}_t) + \boldsymbol{m}_\theta(\bar{\boldsymbol{u}}_t))\mathrm{d}t + \sigma(t)\mathrm{d}\boldsymbol{W}_t$$





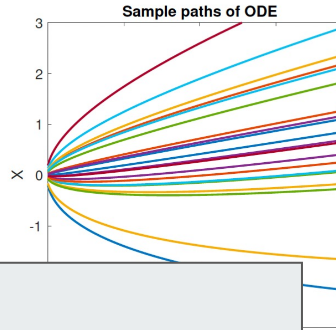
[1] Albergo, Michael S., Nicholas M. Boffi, and Eric Vanden-Eijnden. "Stochastic interpolants: A unifying framework for flows and diffusions." arXiv:2303.08797 (2023).
[2] Y. Lipman, M. Tancik, and J. Lu, Flow matching for generative modelling, arXiv:2210.02747, 2022.




## Stochastic interpolant

$$\boldsymbol{x}_\tau \approx \boldsymbol{I}_\tau(\boldsymbol{x}_0, \boldsymbol{x}_1) = \alpha_\tau \boldsymbol{x}_0 + \beta_\tau \boldsymbol{x}_1 + \gamma_\tau \boldsymbol{W}_\tau$$




## SDE

$$\mathrm{d}\boldsymbol{x}_t = \boldsymbol{b}(\boldsymbol{x}_t)\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}_t$$

## ODE

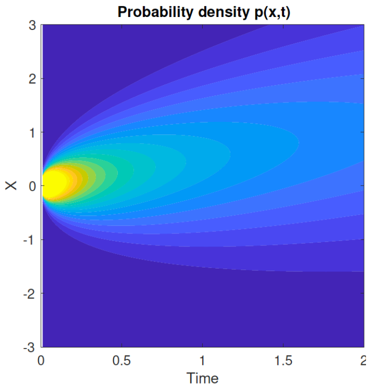
$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{c}(\boldsymbol{x}, t)$$

$$\boldsymbol{c}(\boldsymbol{x}, t) = \boldsymbol{b}(\boldsymbol{x}) - \frac{1}{2} g^2(t) \nabla_{\boldsymbol{x}} \ln p(\boldsymbol{x}, t)$$

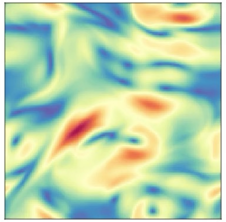
implied distribution

## Fokker-Planck

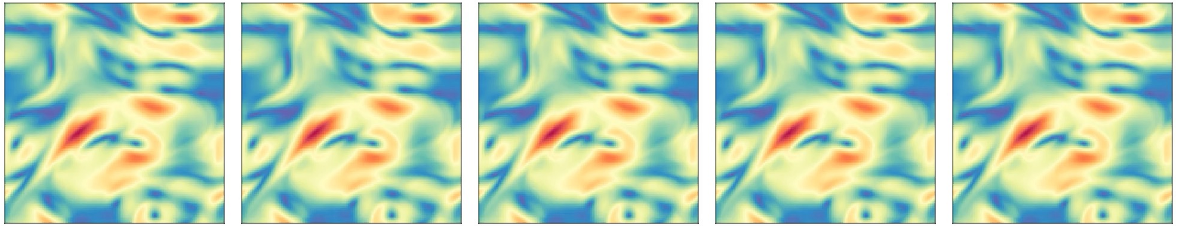
$$\frac{\partial p(\boldsymbol{x}, \boldsymbol{t})}{\partial t} = -\nabla \cdot (\boldsymbol{b}(\boldsymbol{x}) p(\boldsymbol{x}, t)) + \nabla \cdot (\frac{1}{2} g^2(t) \nabla p(\boldsymbol{x}, t))$$



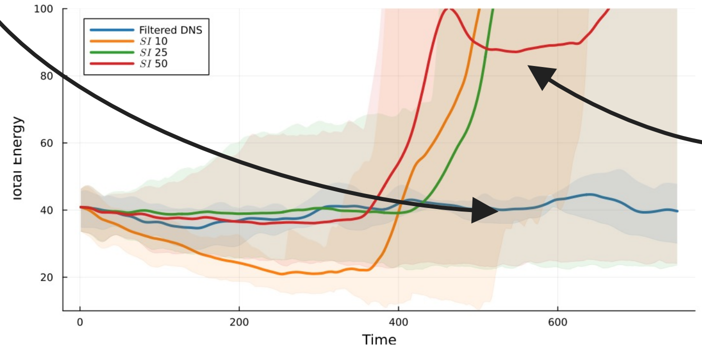
# Vanilla stochastic interpolant results



Ground truth
(filtered DNS)

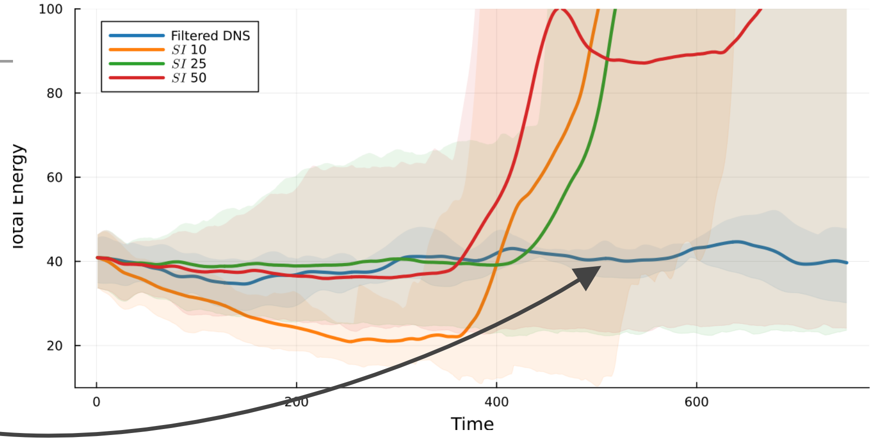


Example realizations




Vanilla SIs have **unphysical energy behavior**

# Structure-preserving SI




- Should not enforce energy stability on individual trajectories
  - Flow only statistically energy stable

- **Idea**: tune the $\alpha_\tau, \beta_\tau$ coefficients for energy consistency:

$$L_{\text{energy}}(\alpha_\tau, \beta_\tau) = \int_0^1 \mathbb{E}_{(\boldsymbol{x}_0, \boldsymbol{x}_1)} \left[ \mathrm{d} \left[ \frac{1}{2} ||\boldsymbol{I}_\tau(\boldsymbol{x}_0, \boldsymbol{x}_1)||_2^2 \right] - k_\tau(\boldsymbol{x}_0, \boldsymbol{x}_1) \right] \mathrm{d}\tau$$

Need Ito's lemma

Desired rate of change

# Structure-preserving SI


Physics-aware generative models for turbulent fluid flows through energy-consistent stochastic interpolants

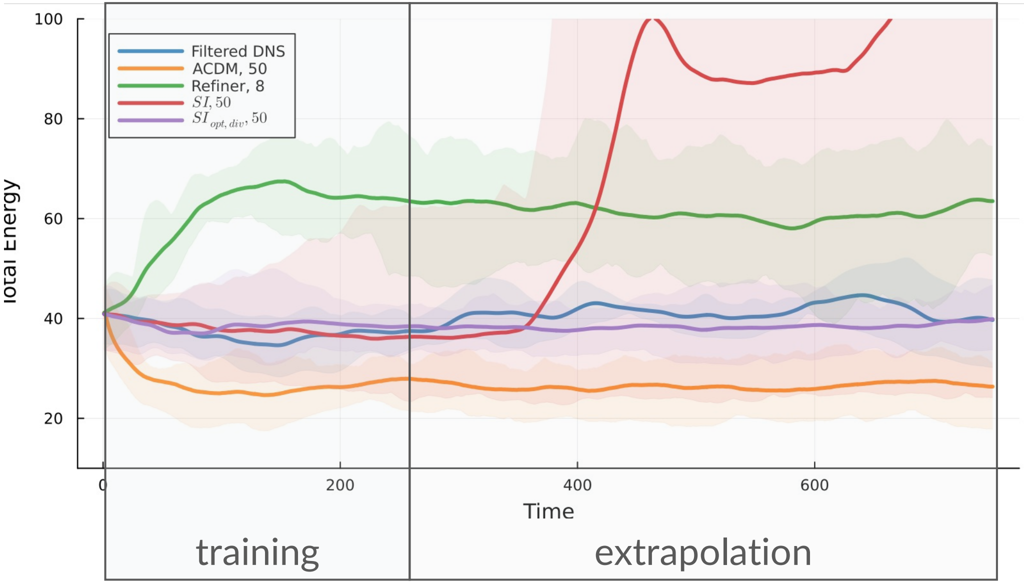
Nikolaj T. Mücke[a], Benjamin Sanderse[a,b]

[a] Scientific Computing, Centrum Wiskunde & Informatica, Science Park 123, Amsterdam, 1098 XG, The Netherlands
[b] Centre for Analysis, Scientific Computing and Applications, Eindhoven University of Technology, PO Box 513, Eindhoven, 5600 MB, The Netherlands

- New parameterization of interpolant promotes stability
- Long roll-outs possible

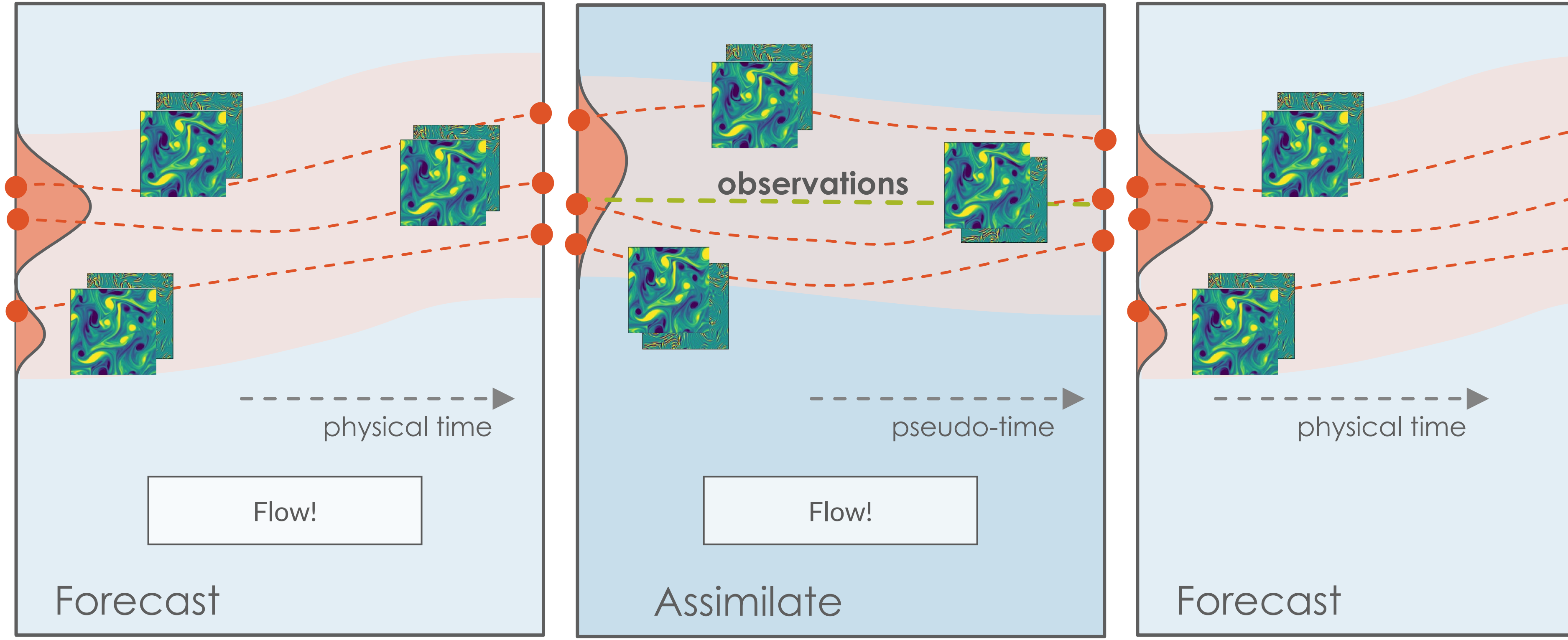
Implications for data assimilation:

- Stochastic interpolants can be adapted to Bayesian inference
- Stable and physics aware inference
- Combined framework for forward and inverse problem



| | A. Fluid flow model | B. Data assimilation |
|---|---|---|
| True dynamics | Navier-Stokes<br>$\frac{\partial}{\partial t}\begin{pmatrix}\rho \\ \rho \boldsymbol{v} \\ E\end{pmatrix} + \nabla \cdot \begin{pmatrix}\rho \boldsymbol{v} \\ p\boldsymbol{I} + \rho \boldsymbol{v} \otimes \boldsymbol{v} \\ \boldsymbol{v}(E+p)\end{pmatrix} = \nabla \cdot \begin{pmatrix}0 \\ \boldsymbol{\tau} \\ \boldsymbol{\tau} \cdot \boldsymbol{u} + \boldsymbol{q}\end{pmatrix}$ | Bayes / Fokker Planck<br>$\frac{\partial p_\tau}{\partial \tau} + \nabla \cdot (p_\tau \boldsymbol{c}) = 0 \qquad \boldsymbol{c} = \nabla \ln\left(\frac{q(\bar{\boldsymbol{u}})}{p_\tau(\bar{\boldsymbol{u}})}\right)$ |
| Learnt model | Neural LES model<br>$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}t} = \boldsymbol{f}(\bar{\boldsymbol{u}}) + \boldsymbol{m}(\bar{\boldsymbol{u}};\theta)$ | Particle flow filter<br>$\frac{\mathrm{d}\bar{\boldsymbol{u}}}{\mathrm{d}\tau} = \boldsymbol{g}(\bar{\boldsymbol{u}};\theta)$ |
| Structure (entropy) | Stable and admissible solutions | Distance to target (posterior) |
| Stochastic | Probabilistic (UQ); model error; chaoticity; regularization<br>$\mathrm{d}\bar{\boldsymbol{u}}_t = (\boldsymbol{f}(\bar{\boldsymbol{u}}_t) + \boldsymbol{m}_\theta(\bar{\boldsymbol{u}}_t))\mathrm{d}t + \sigma(t)\mathrm{d}\boldsymbol{W}_t$ | Explore phase space; ugly posteriors; bifurcations<br>$\mathrm{d}\bar{\boldsymbol{u}}_\tau = \boldsymbol{g}_\theta(\bar{\boldsymbol{u}})\mathrm{d}\tau + \sigma(\tau)\mathrm{d}\boldsymbol{W}_\tau$ |

# Everything flows



# Connecting to plasma fusion

- Approximations (averaging, collisions, etc.) made in fusion models (Vlasov-Poisson/Maxwell/gyrokinetic) introduce model error that can (should?) be modelled stochastically

- The forward kinetic equation in fusion is a nonlinear FP equation with an associated entropy function
  -> *implied SDE?*
  -> *even tighter integration of forward and inverse problem?*

- Learned (chaotic!) models require calibration (data assimilation), *especially in the case of control*