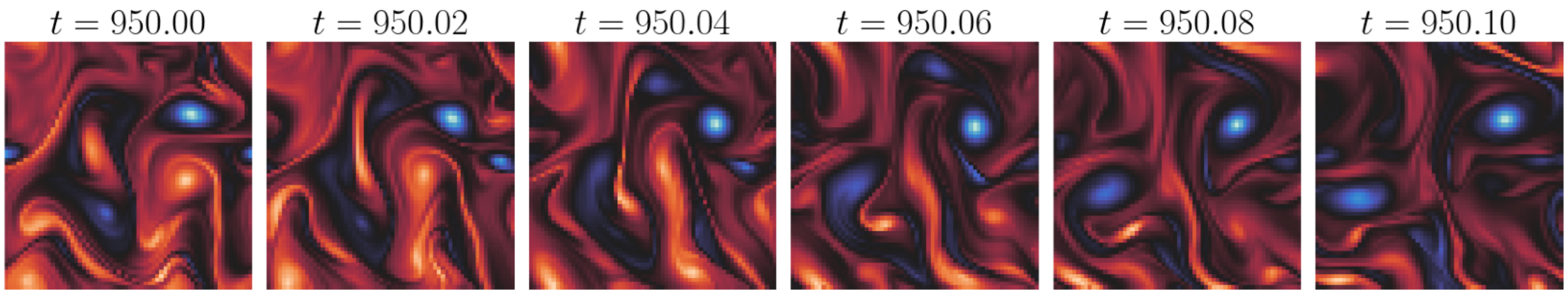
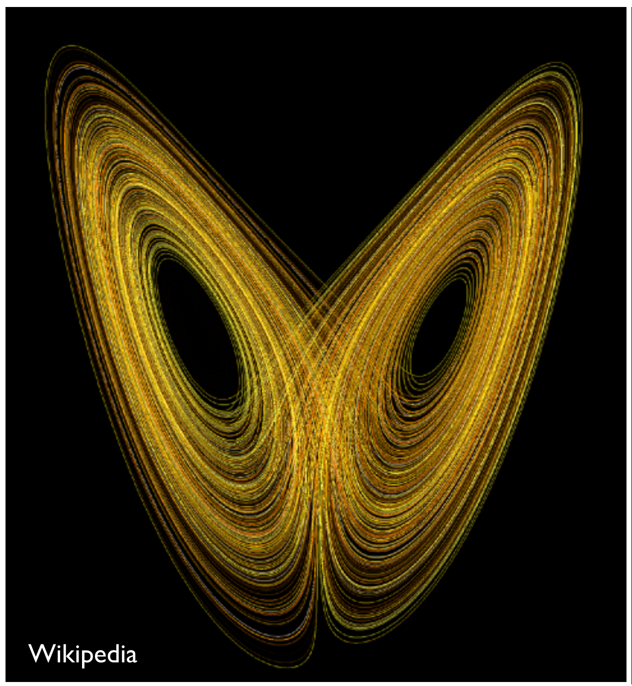
Conditional Gaussian Koopman Network — A Deep Learning Digital Twins Framework for Stochastic Modeling, Forecast, and Data Assimilation

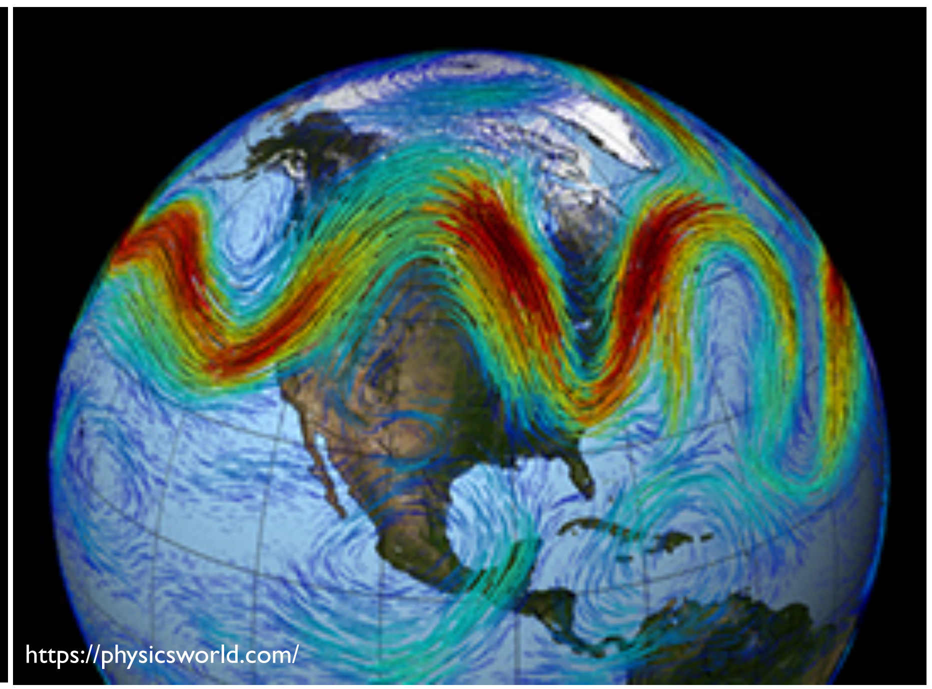
Chuanqi Chen[a], **Zhongrui Wang**[b], Nan Chen[b], Jinlong Wu[a]

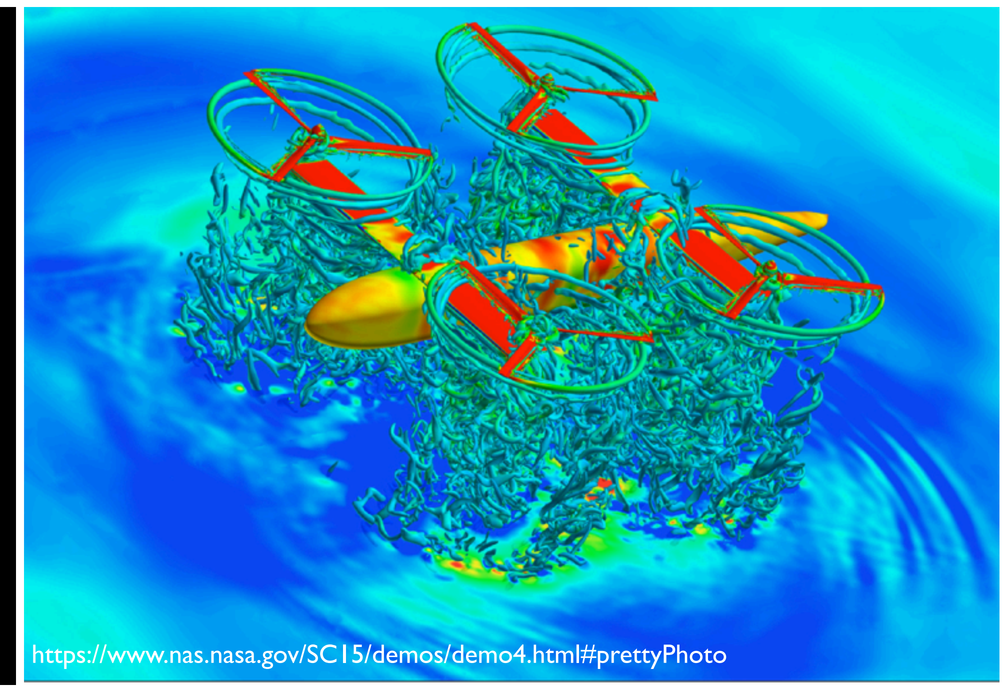
[a]*Department of Mechanical Engineering, University of Wisconsin–Madison*

[a]*Department of Mathematics, University of Wisconsin–Madison*



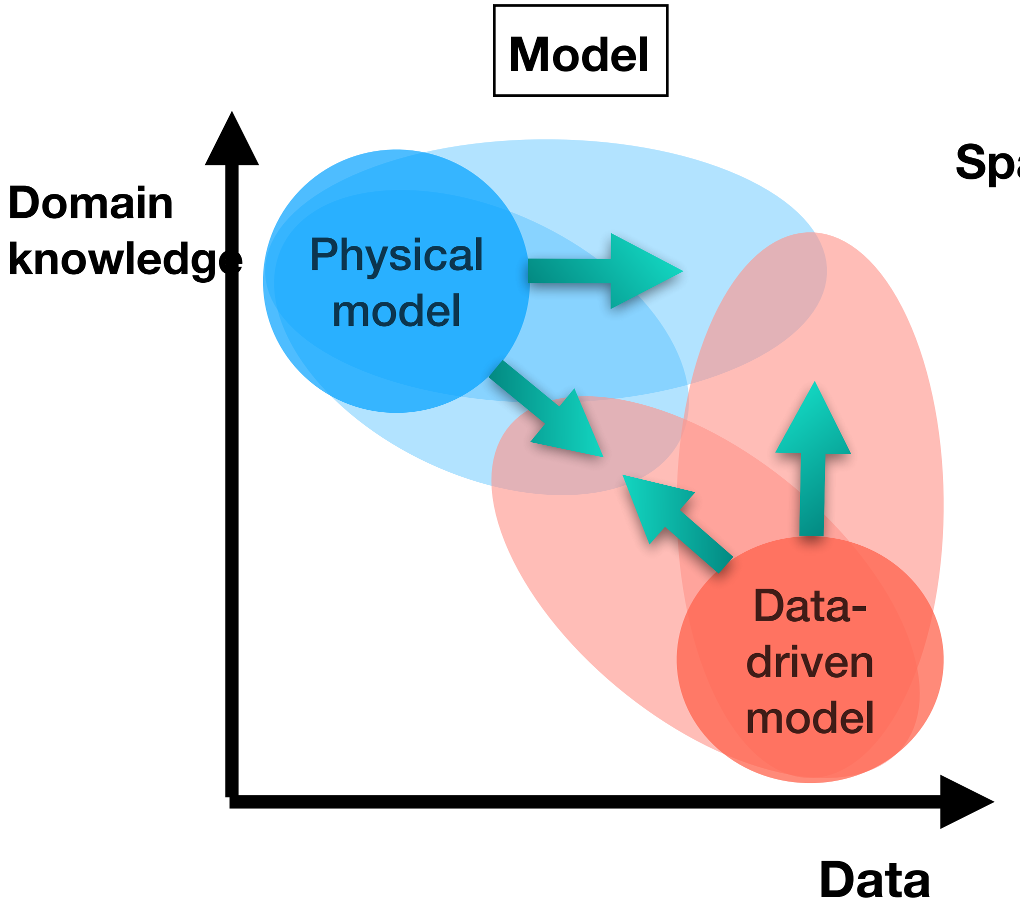


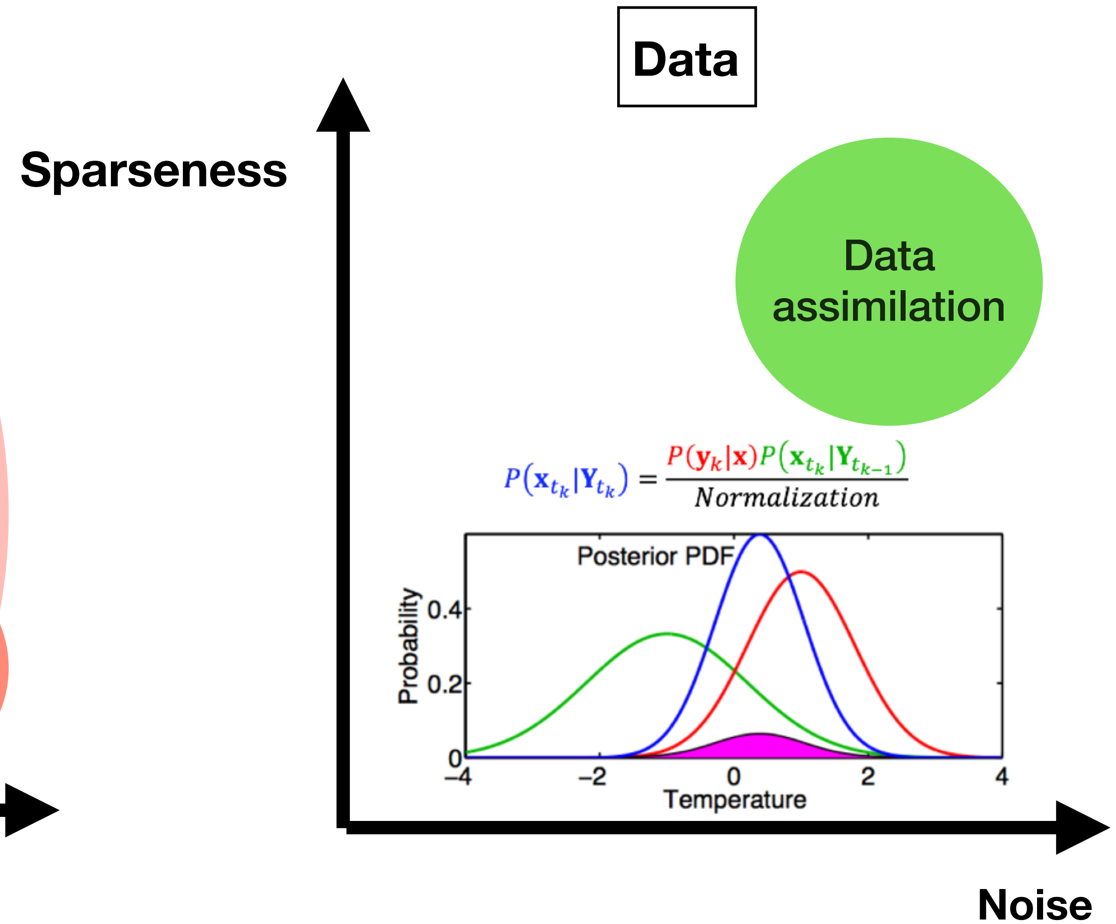




- **Complex dynamical systems**: nonlinear, chaotic, multi-scale, turbulent, intermittent, non-Gaussian; common in fluid dynamics, geophysics, neuroscience, material science…
- **Goal**:
  - Make predictions —> models
  - Use observations to improve predictions —> data assimilation (DA)

DA is widely used in real-time forecasting, parameter estimation, imputation, and optimal control.





- Physical model:
  - Based on governing equations derived from first principles (interpretable)
  - May require strong assumptions; Usually computationally expensive (e.g., NWP)
- Data-driven model:
  - Works with governing equations unknown (but lack of interpretability)
  - Computational efficient; May require a large amount of data (can be sparse and noisy in reality)
- Data assimilation is especially useful when data is sparse and noisy
  - Combing data with existing models, DA can recover complete data, with reduced uncertainty
  - As new observations become available, DA can utilize this information to improve real-time predictions

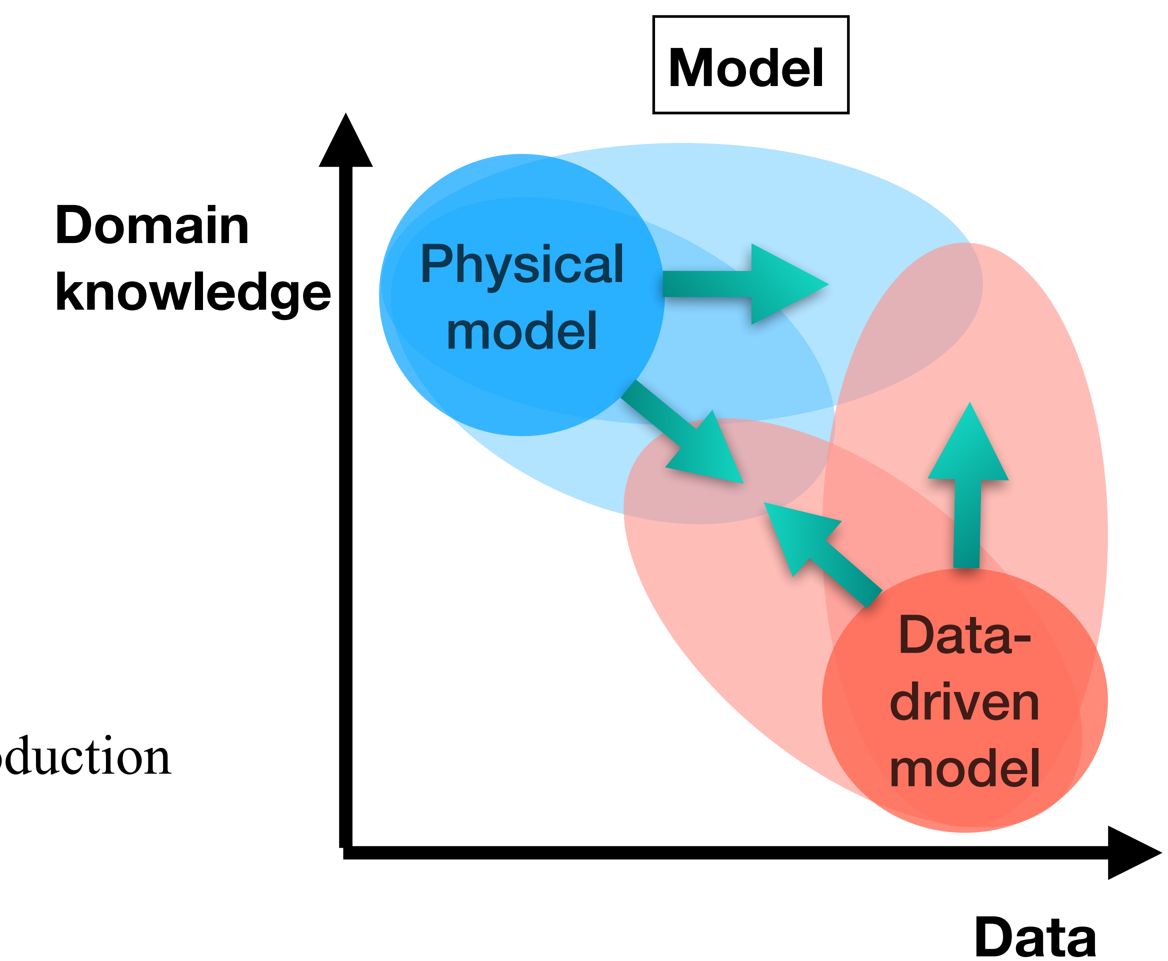
Data-driven models for dynamical systems (SciML)

- *Reduced-order models*: linear stochastic models; dimension reduction; spectral truncation
- *Dynamics identification*: SINDy via sparse regression (Brunton, 2016), causation entropy-based identification (Chen, 2023)
- *Operator learning*: Neural ODE (Chen, 2019); <u>Fourier Neural Operator</u> (FNO; Li, 2021) (Challenging but powerful)

Combination with physical models:

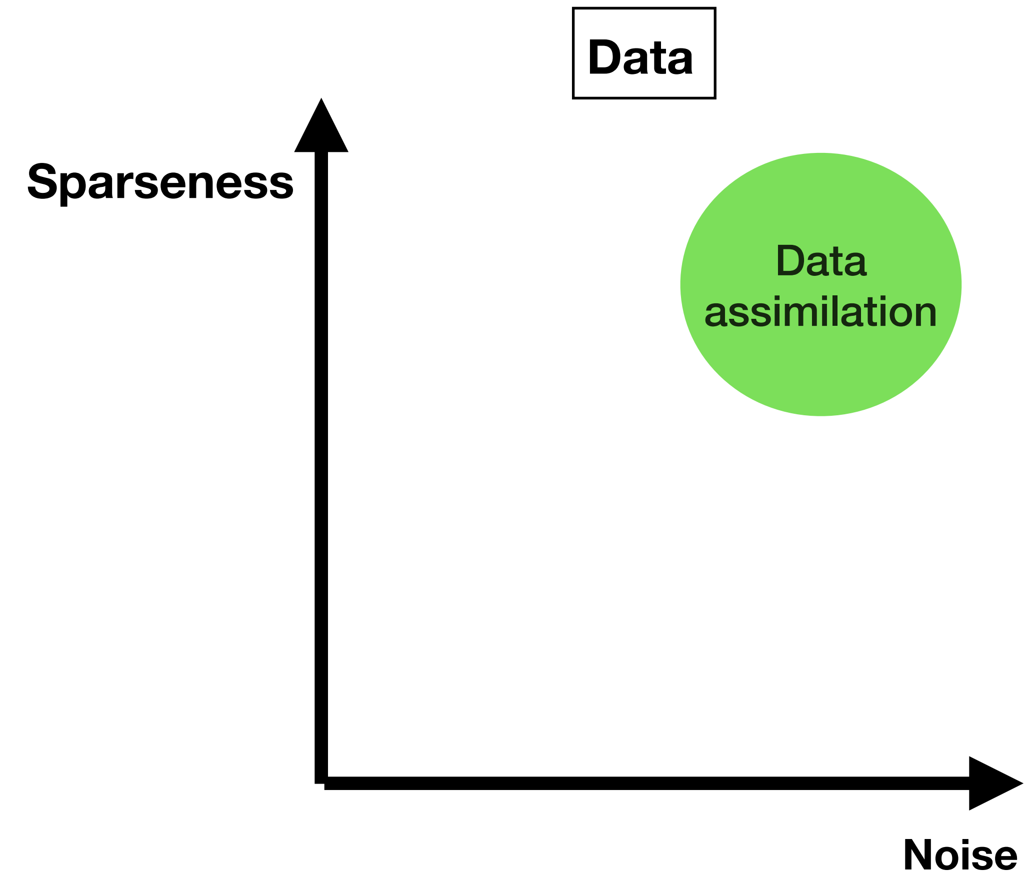
- *Residual learning*: closure models (Levine and Stuart, 2022)
- *Physics-informed machine learning*: PINNs (Raissi, 2019)

[Review: Learning dynamical systems from data: An introduction to physics-guided deep learning (Yu, 2024)]



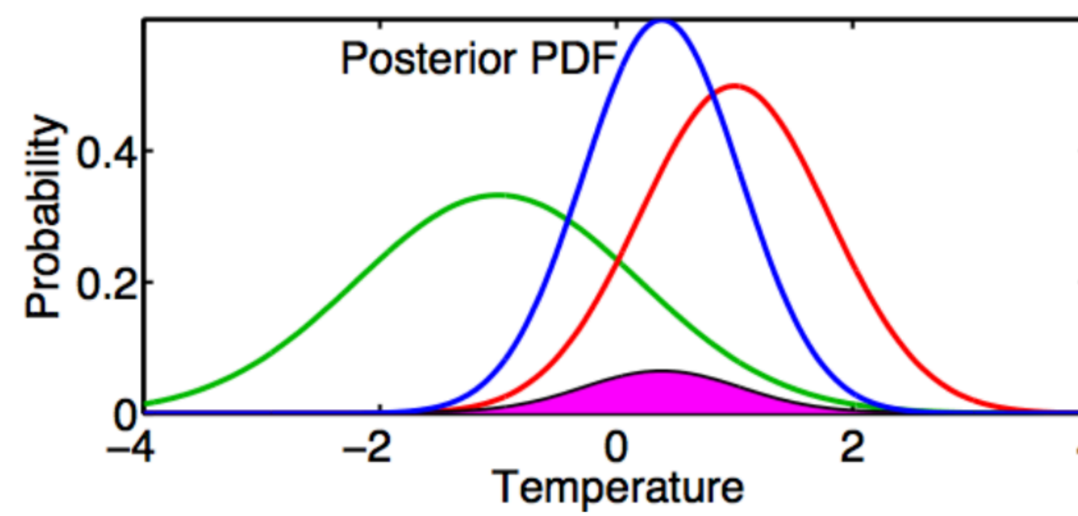
DA for using observations to improve predictions

- DA typically works in scenarios where observations are partial and noisy
- **Challenges:**
  - nonlinearity
  - non-Gaussianity
  - high computational costs due to high dimensionality



- Linear Gaussian case: Kalman Filter
- Efforts to tackle **nonlinearity** and **non-Gaussianity**:
  - Ensemble Kalman Filter (EnKF; Evensen, 1994)
  - 4D-Variational methods (4DVar; Le Dimet and Talagrand, 1986)
  - Particle Filter (PF; van Leeuwen, 2009)
  - Conditional Gaussian Filter (Chen and Majda, 2016)
  - Quantile-conserving Ensemble Filter (Anderson, 2023)

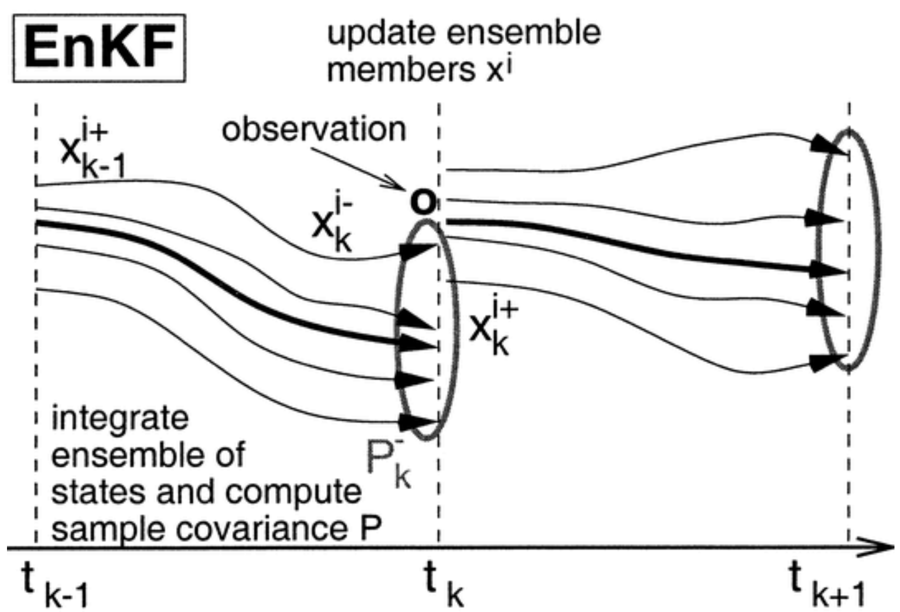
$$P(\mathbf{x}_{t_k}|\mathbf{Y}_{t_k}) = \frac{P(\mathbf{y}_k|\mathbf{x})P(\mathbf{x}_{t_k}|\mathbf{Y}_{t_{k-1}})}{Normalization}$$



Bayes rule (1D Gaussian case).
Image from slides of Jeff Anderson (NCAR)

Combining SciML with DA

- ML for DA
    - ML as cheap surrogate models to generate ensembles or using ML to estimates error covariance statistics (still adheres to an imperfect DA method)
    - Pure data-driven end-to-end DA (UQ is difficult for point-wise prediction; Computation cost can be high for measure transport.)
    - Model error correction

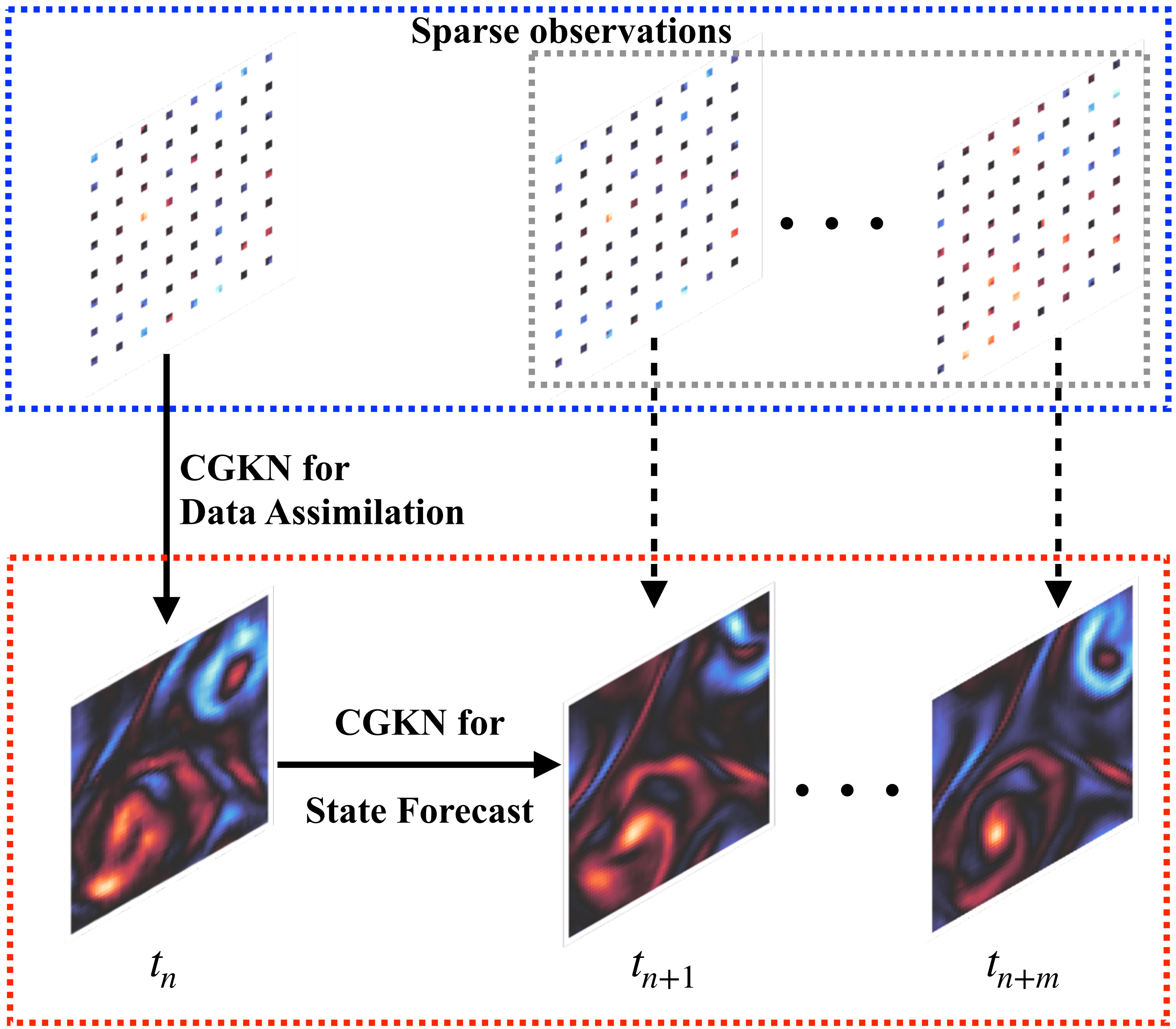


[Reichle, R. H., 2002]

- DA for ML
    - DA analysis as better training data
    - Corrects ML model predictions
    - Derivative-free optimization: Ensemble Kalman Inversion (Iglesias, 2013)

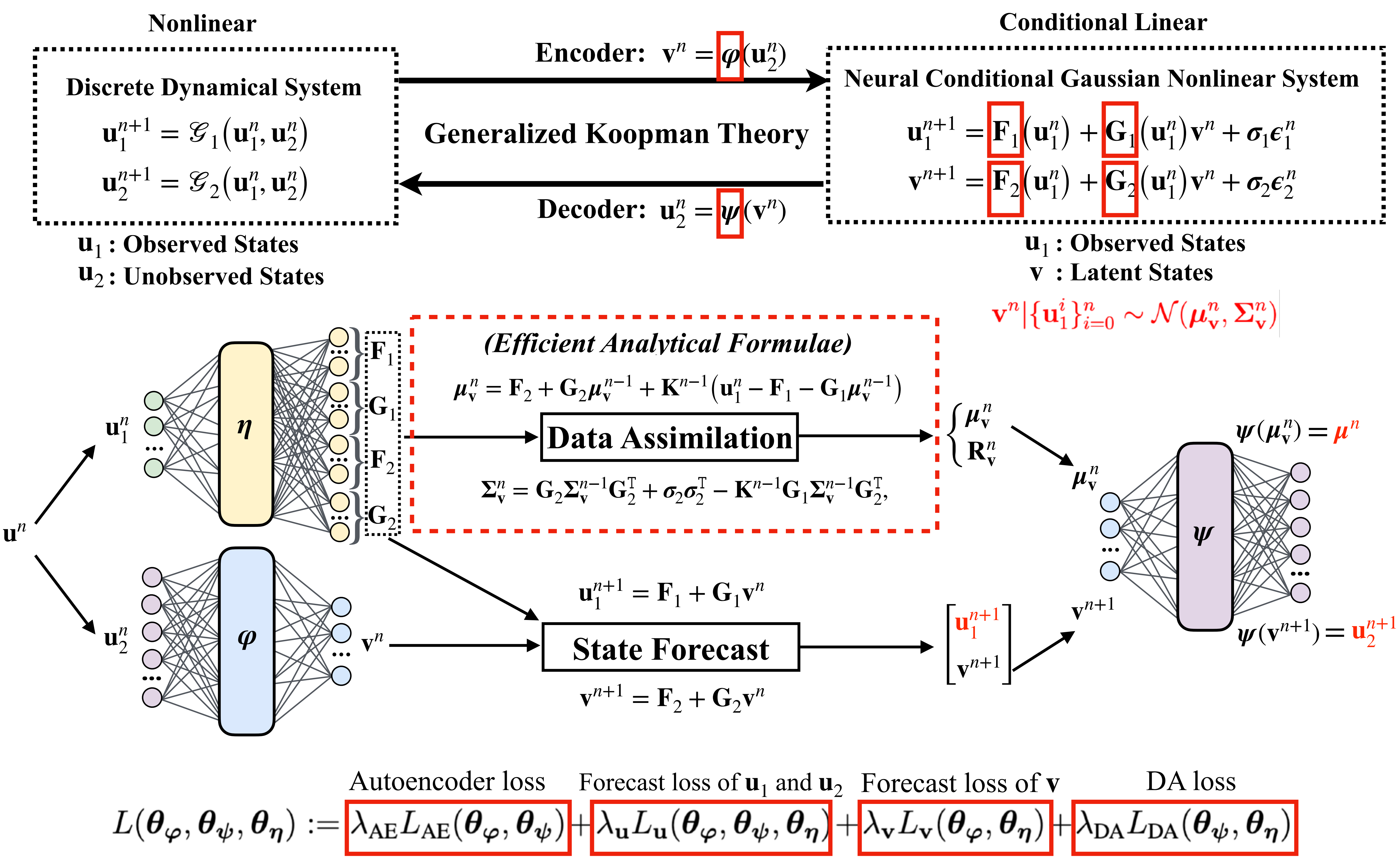
[Review: Machine learning with data assimilation and uncertainty quantification for dynamical systems: a review (Cheng et al., 2023)]

Conditional Gaussian Koopman Network (CGKN)



- A **unified framework** of SciML and DA, to learn surrogate models that performs **efficient prediction** and **DA (with UQ)** for **nonlinear partially observed** dynamical systems.

CGKN (Discrete-time)



Heuristics from Koopman theory:

Nonlinear

Discrete Dynamical System

$$\mathbf{u}_1^{n+1} = \mathscr{G}_1\left(\mathbf{u}_1^n, \mathbf{u}_2^n\right)$$
$$\mathbf{u}_2^{n+1} = \mathscr{G}_2\left(\mathbf{u}_1^n, \mathbf{u}_2^n\right)$$

$\mathbf{u}_1$ **: Observed States**
$\mathbf{u}_2$ **: Unobserved States**

Neural Networks

Encoder: $\mathbf{v}^n = \boxed{\boldsymbol{\varphi}}(\mathbf{u}_2^n)$

Generalized Koopman Theory

Decoder: $\mathbf{u}_2^n = \boxed{\boldsymbol{\psi}}(\mathbf{v}^n)$

Conditional Linear

Neural Conditional Gaussian Nonlinear System

$$\mathbf{u}_1^{n+1} = \boxed{\mathbf{F}_1}\left(\mathbf{u}_1^n\right) + \boxed{\mathbf{G}_1}\left(\mathbf{u}_1^n\right)\mathbf{v}^n + \boldsymbol{\sigma}_1\boldsymbol{\epsilon}_1^n$$
$$\mathbf{v}^{n+1} = \boxed{\mathbf{F}_2}\left(\mathbf{u}_1^n\right) + \boxed{\mathbf{G}_2}\left(\mathbf{u}_1^n\right)\mathbf{v}^n + \boldsymbol{\sigma}_2\boldsymbol{\epsilon}_2^n$$

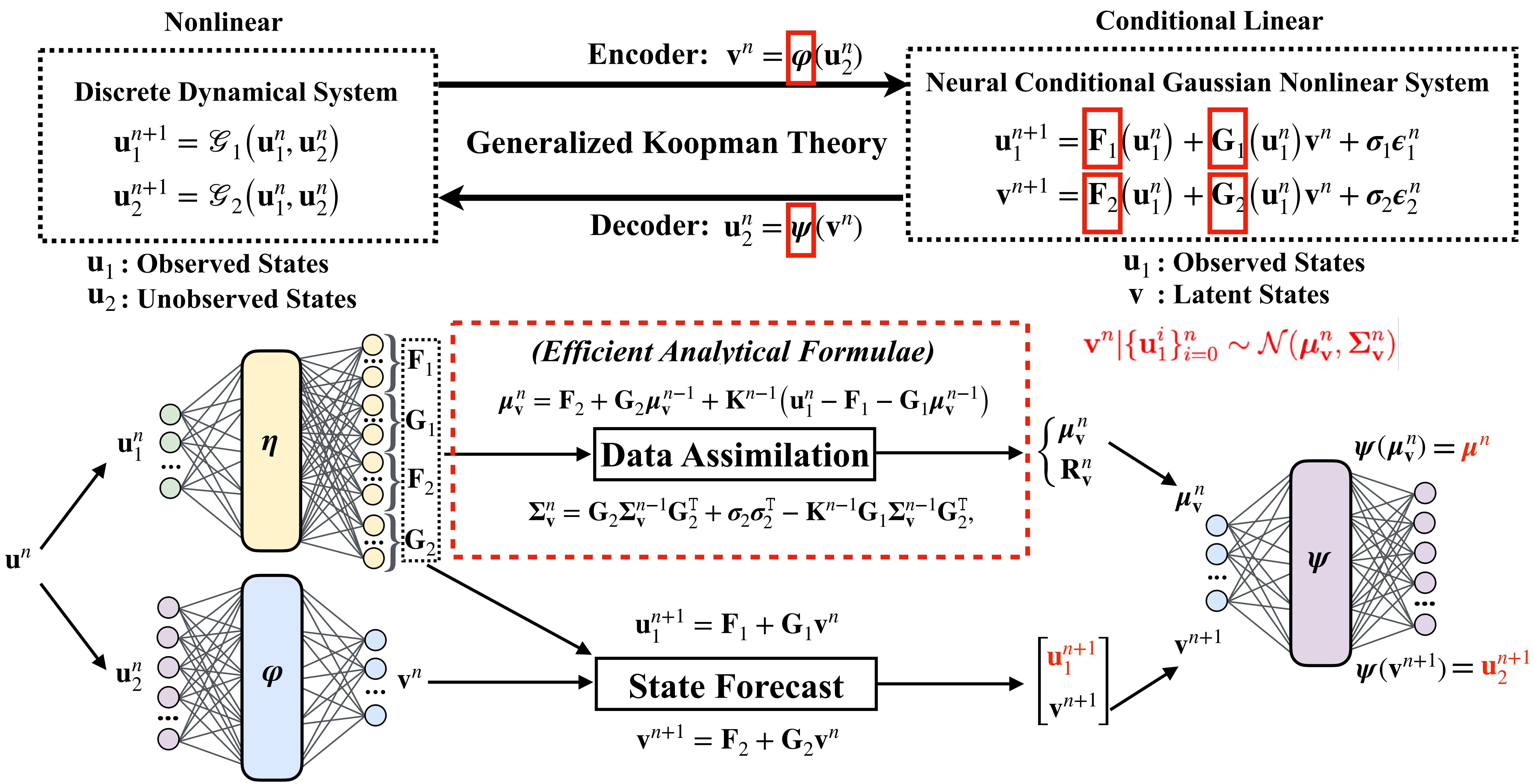
$\mathbf{u}_1$ **: Observed States**
$\mathbf{v}$ **: Latent States**

$$\mathbf{v}^n|\{\mathbf{u}_1^i\}_{i=0}^n \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{v}}^n, \boldsymbol{\Sigma}_{\mathbf{v}}^n)$$

- Instead of directly applying Koopman theory, CGKN only seeks for embeddings of the unobserved states $\mathbf{u}_2$.
- The latent representation $\mathbf{v}$ is conditional linear given $\mathbf{u}_1$ being observed, which leads to a neural conditional Gaussian nonlinear system (CGNS) that has analytic DA formulae.

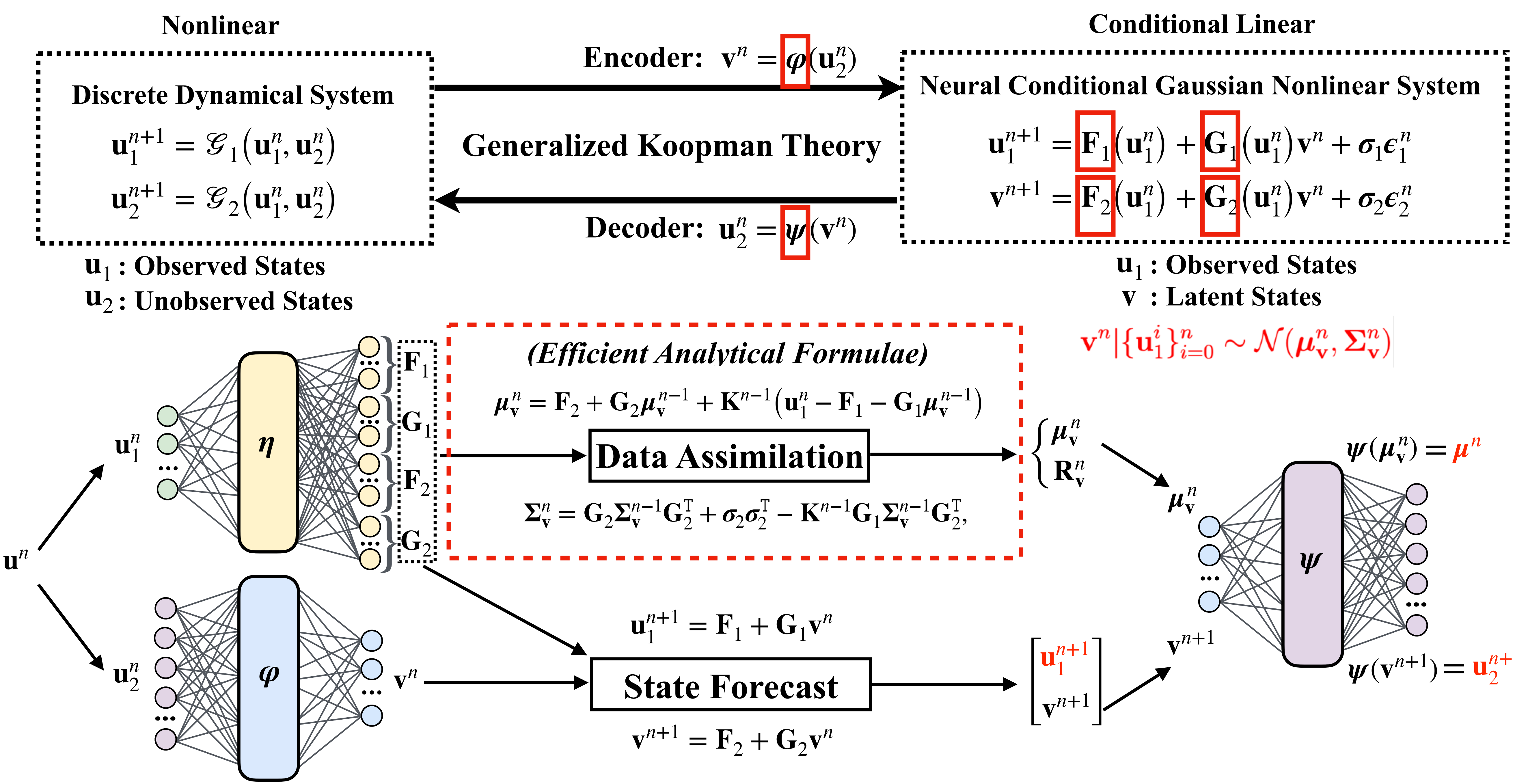
[Koopman, 1931; Brunton, 2022]

CGKN (Discrete-time)



- State forecast and DA are performed in the latent space (with reduced dimension)

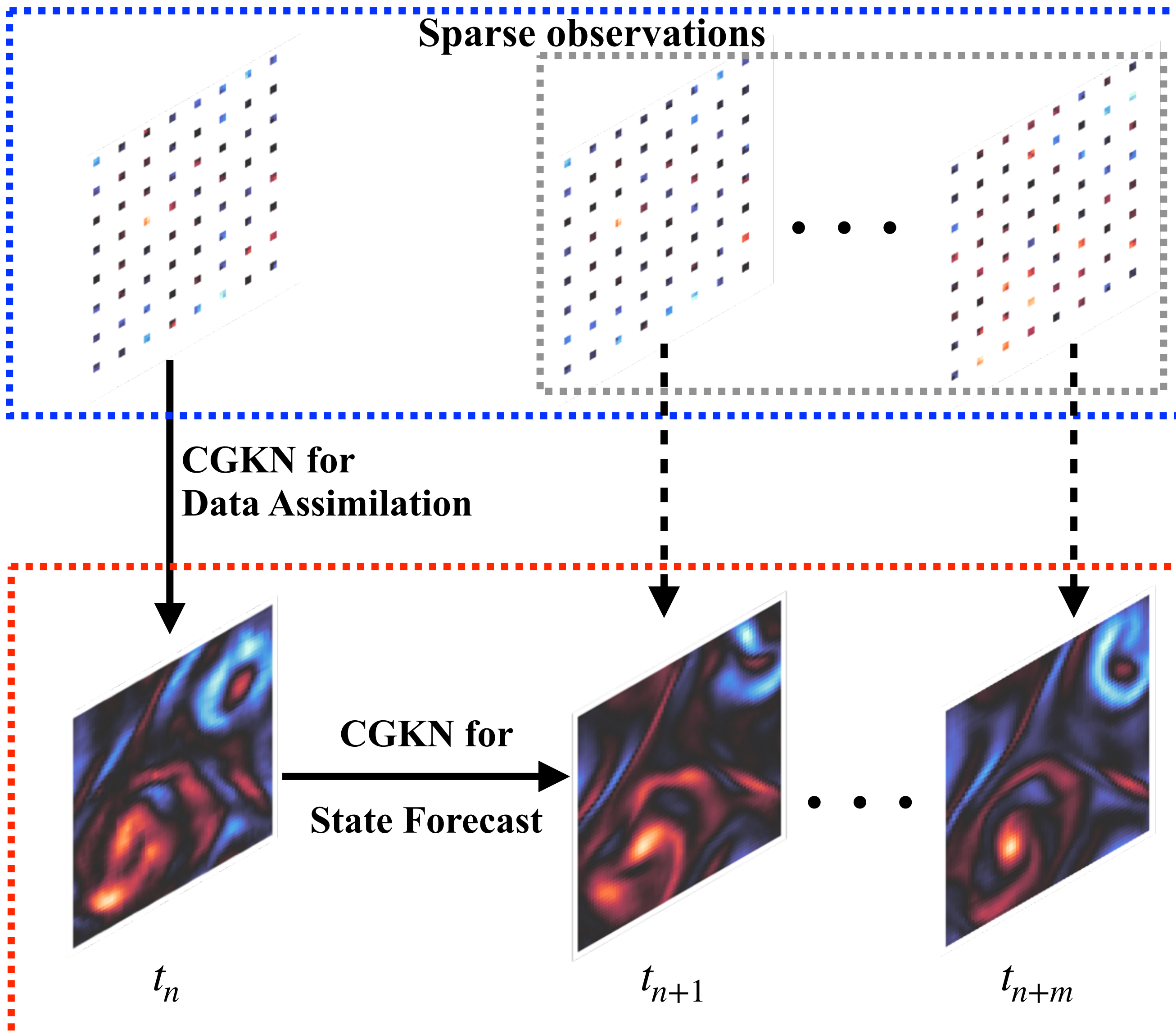
CGKN (Discrete-time)



$$L(\boldsymbol{\theta}_\varphi, \boldsymbol{\theta}_\psi, \boldsymbol{\theta}_\eta) := \underbrace{\boxed{\lambda_{\mathrm{AE}} L_{\mathrm{AE}}(\boldsymbol{\theta}_\varphi, \boldsymbol{\theta}_\psi)}}_{\text{Autoencoder loss}} + \underbrace{\boxed{\lambda_{\mathbf{u}} L_{\mathbf{u}}(\boldsymbol{\theta}_\varphi, \boldsymbol{\theta}_\psi, \boldsymbol{\theta}_\eta)}}_{\text{Forecast loss of } \mathbf{u}_1 \text{ and } \mathbf{u}_2} + \underbrace{\boxed{\lambda_{\mathbf{v}} L_{\mathbf{v}}(\boldsymbol{\theta}_\varphi, \boldsymbol{\theta}_\eta)}}_{\text{Forecast loss of } \mathbf{v}} + \underbrace{\boxed{\lambda_{\mathrm{DA}} L_{\mathrm{DA}}(\boldsymbol{\theta}_\psi, \boldsymbol{\theta}_\eta)}}_{\text{DA loss}}$$

- DA formulae is part of the model structure as inductive bias, and learning is constrained by DA loss.

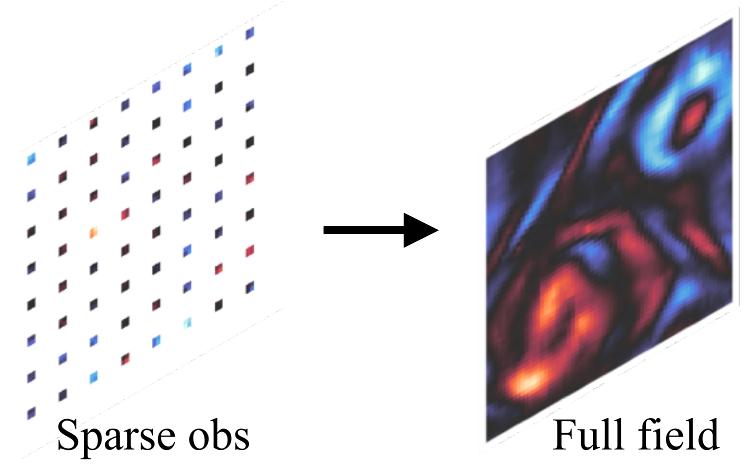
CGKN (Discrete-time)



- Make predictions with only **partially observed initial conditions**, and **improve predictions as new observations come**
- *Analytic DA* formulae for conditional Gaussian systems:
  - Ensures **accuracy** and **efficiency** of DA (scalable to high dimensions)
  - Introduces **inductive bias** to model structure, and constrains the learning
  - Also facilitates the downstream tasks like **uncertainty quantification (UQ)**

Numerical tests

- **Tested cases (PDEs):**
  - Viscous Burgers' equation: a nonlinear PDE of fluid mechanics, notable for modeling **shock** waves
  - Kuramoto-Sivashinsky equation: a nonlinear PDE exhibiting **chaotic** behavior
  - Navier-Stokes equations: PDEs of fluid dynamics that shows **turbulence**



- **Compared benchmarks:**
  - Discrete-time CGKN
  - Ensemble Kalman Filter (EnKF)
  - Direct spatial interpolation

(For DA comparison)

  - Deep neural network (DNN)
  - Convolutional neural network (CNN)
  - Fourier neural operator (FNO) [Li, 2021]

(For predication comparison)

Numerical tests

Summary of numerical results (MSEs; one-step error for forecast/time averaged error for DA)

Methods \ Examples	Viscous Burgers Equation		Kuramoto–Sivashinsky Equation		Navier–Stokes Equations	
	Forecast Error	DA Error	Forecast Error	DA Error	Forecast Error	DA Error
CGKN	7.5683e-04	7.5037e-04	1.1042e-02	2.4927e-02	1.9754e+01	6.0940e+01
EnKF	—	5.8125e-04	—	2.4882e-02	—	6.9010e+01
Interpolation	—	1.3514e-02	—	4.3097e-01	—	1.2844e+02
DNN	6.4816e-03	—	4.7332e-02	—	1.0936e+02	—
CNN	2.3727e-03	—	2.6111e-02	—	3.0600e+01	—
FNO	3.9715e-04	—	5.4859e-03	—	1.7129e+01	—

Computational efficiency compared to EnKF:
- ~ 600 times faster for VBE;
- ~ 125 times faster for KSE;
- ~ 300 times faster for NSE;

Different neural network models have similar number of parameters.

- **Compared benchmarks:**
  - Discrete-time CGKN
  - Ensemble Kalman Filter (EnKF)
  - Direct spatial interpolation

(For DA comparison)

  - Deep neural network (DNN)
  - Convolutional neural network (CNN)
  - Fourier neural operator (FNO)

(For predication comparison)

Navier-Stokes Equations: 2-D turbulent PDE

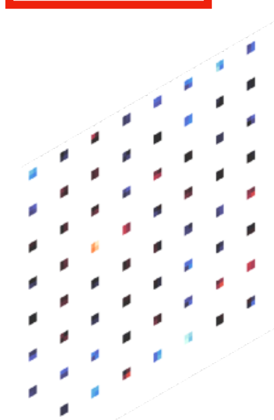
$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u}\cdot\nabla\mathbf{u} + \nu\nabla^2\mathbf{u} - \frac{1}{\rho}\nabla p + \mathbf{f}, \qquad \mathbf{u}(\mathbf{x},t) \in \mathbb{R}^2$$

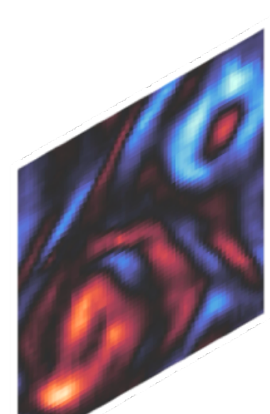
$$\nabla\cdot\mathbf{u} = 0, \qquad \mathbf{x} \in \Omega \subset \mathbb{R}^2, \quad t \in [0, L_t]$$

Settings

- Parmeters: $\nu = 10^{-4}$, $\rho = 1$, $\mathbf{f} = [100\sin(8y), 0]^{\mathrm{T}}$, $\Omega = [0,1]^2$, $L_t = 1000$,
- Truth generated with $\Delta x = \Delta y = 1/256$ $\Delta t = 10^{-3}$
- Trading and testing data are sub-sampled to $\Delta x = \Delta y = 1/64$. $\Delta t = 0.01$
- Training data: 800 time units; test data: 200 time units
- DA time steps $t_{N_l} = 20$ time units (2000 steps) , EnKF ensemble size=100
- State forecast time steps $t_{N_s} = \Delta t = 0.01$ time units for training
- Number of parameters $\#\boldsymbol{\theta}_\varphi = 187265$ $\#\boldsymbol{\theta}_\psi = 187265$ $\#\boldsymbol{\theta}_\eta = 82240$

- Observed states: **8x8 out of 64x64** (uniformly distributed) with observational noise ~ $\mathcal{N}(0, 2.5^2)$; unobserved states: 120 out of 128
- Encoder $\varphi : \mathbb{R}^{64\times 64} \mapsto \boxed{\mathbb{R}^{16\times 16}}$ decoder $\psi : \mathbb{R}^{16\times l6} \mapsto \mathbb{R}^{64\times 64}$


Sparse obs


Full field

Navier-Stokes Equations: 2-D turbulent PDE

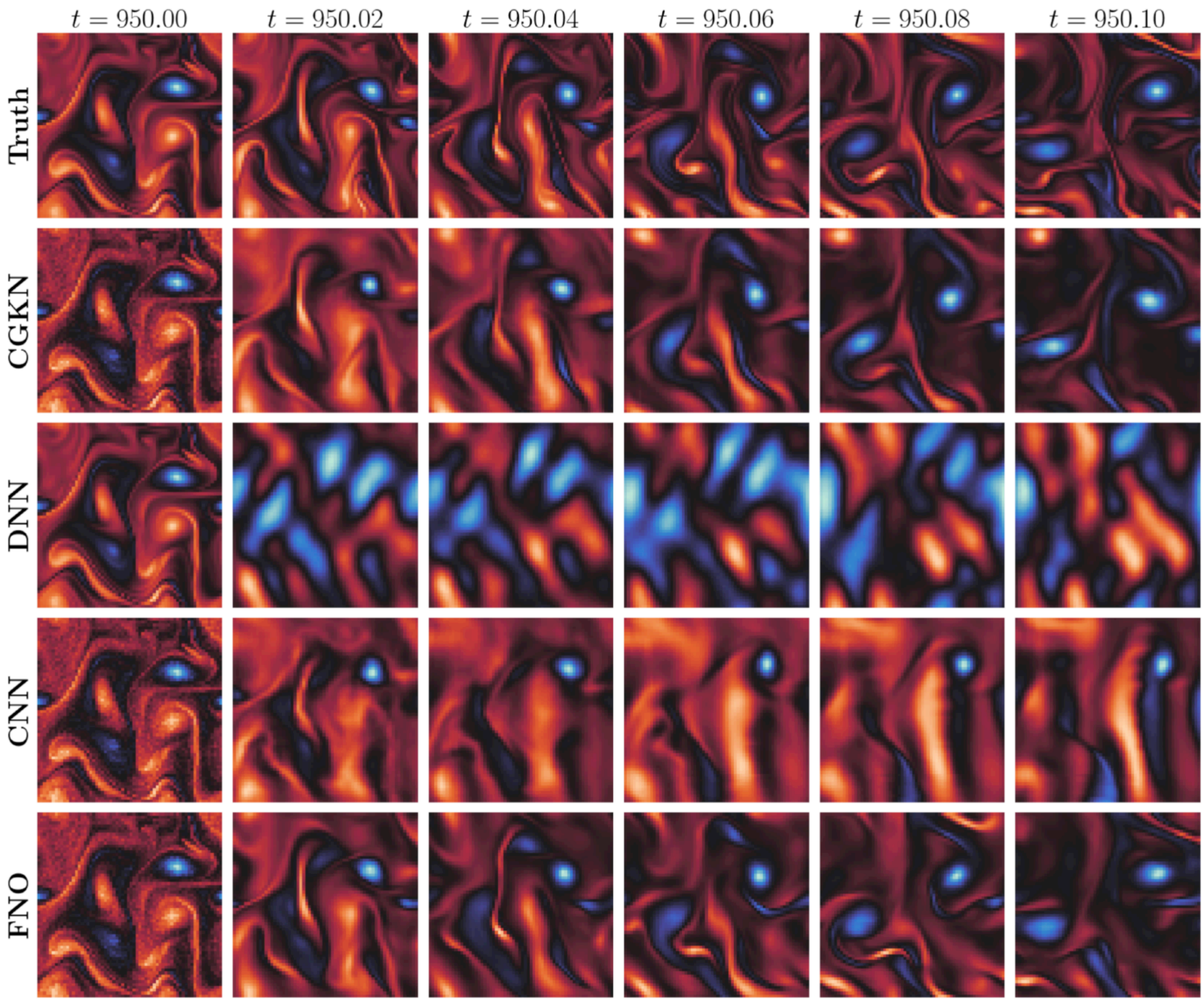


Figure 3.9: Results of state forecast for the Navier-Stokes equations. The evolution of the true vorticity field is compared with the predictive vorticity field from various models including CGKN, DNN, CNN, and FNO.

Navier-Stokes Equations: 2-D turbulent PDE

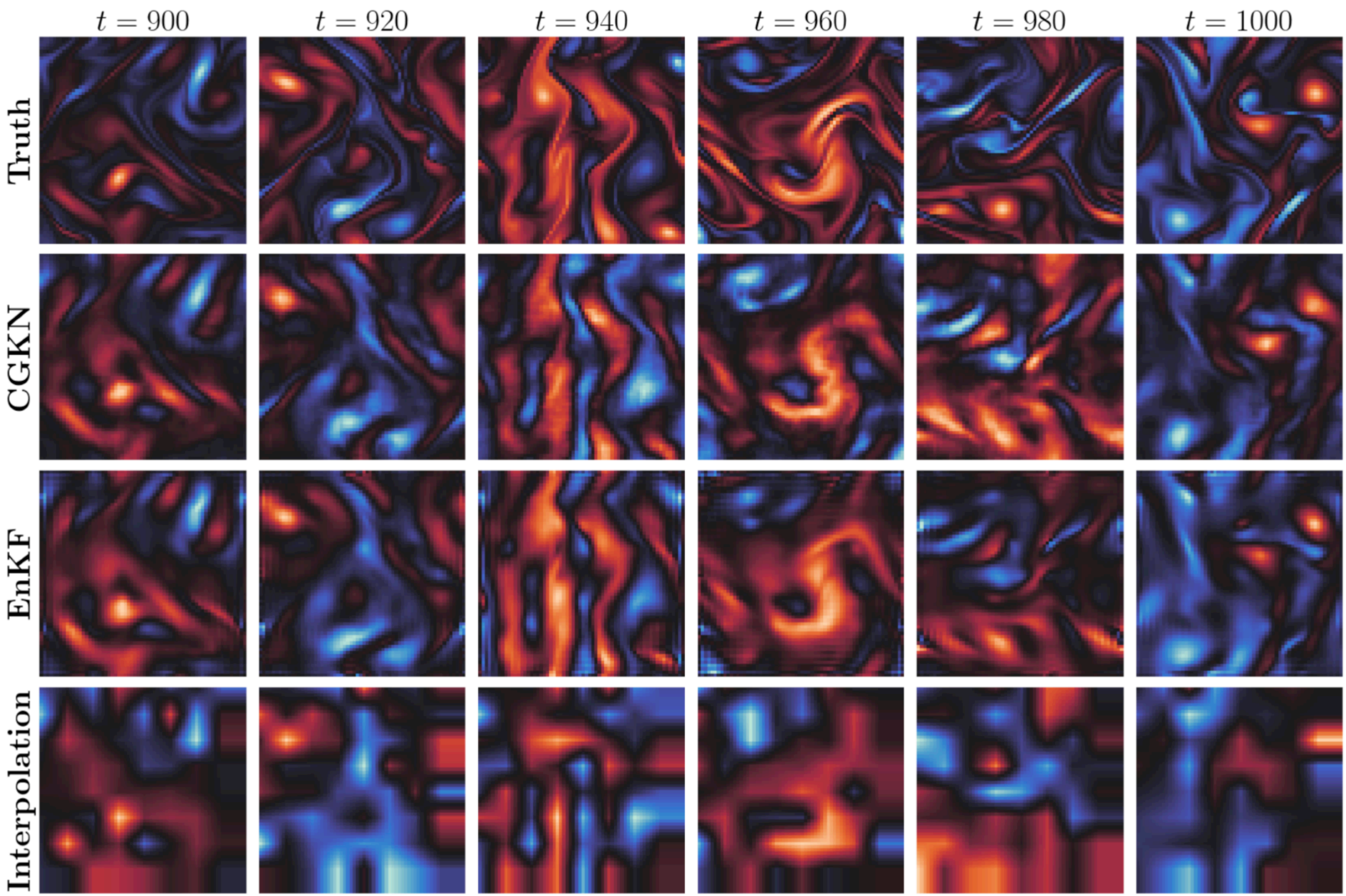


Figure 3.10: Spatiotemporal evolution of the DA results for the Navier-Stokes equations. Heatmaps of each row display the vorticity fields for the true simulation, the DA posterior means from CGKN and EnKF, and the interpolation result.

Summary

- The discrete-time CGKN is a **unified deep learning framework** for **efficient state forecast** and **DA**.
- The **efficiency** comes from the *analytic DA formulae* of conditional Gaussian structure, and the *lower dimension of embedded latent space*. Compared to the continuous-time CGKN, the discrete-time version further improves the efficiency by *allowing larger time steps*.
- For predictions in realistic scenarios, **partial noisy observations can be used** by DA to provide initial conditions or improve the real-time forecast.
- The analytic DA formulae is incorporated into the ML model structure as inductive bias, and constrains the learning via loss function, leading to a **unified framework for SciML and DA**.

[Modeling partially observed nonlinear dynamical systems and efficient data assimilation via discrete-time conditional Gaussian Koopman network, *Computer Methods in Applied Mechanics and Engineering*, 2025]