Time-Dependent Statistical Finite Element Problems: Unstructured Data, Unknown Observation Operators, and Inversion

Connor Duffin

Department of Engineering
University of Cambridge

University of Warwick 2023, 2 February 2024

Collaborators and References

Alex Glyn-Davies (Cambridge)

Deniz Akyıldız (Imperial)

Ed Cripps (UWA)

Thomas Stemler (UWA)

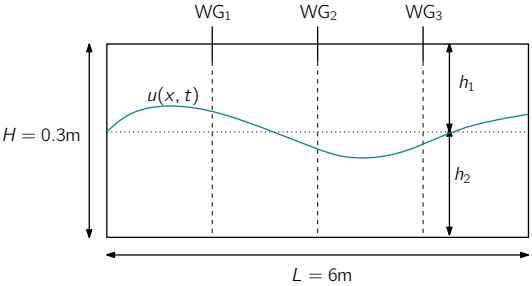
Mark Girolami (Cambridge)

1. Connor Duffin et al. "Statistical Finite Elements for Misspecified Models". *Proceedings of the National Academy of Sciences* (Jan. 2021).
2. Alex Glyn-Davies et al. *Φ-DVAE: Physics-Informed Dynamical Variational Autoencoders for Unstructured Data Assimilation*, July 2023. arXiv: `2209.15609 [physics, stat]`.

Talk Structure

1. Problem Motivation.
2. Statistical Finite Element Methodology.
3. Case Study: Internal Waves Experimental Data.
4. Extending statFEM for Unstructured Data: Φ-DVAE.
5. Case Studies: Φ-DVAE inversion.

Motivation



- Internal waves flow between layers of density-varying water (mean depths h_1, h_2), in a tank of length L and total depth $H = h_1 + h_2$.
- KdV equation models the internal wave profile $u(s,t)$ (deviations from rest):

$$\partial_t u + \alpha u u_s + \beta u_{sss} + c u_s = 0.$$

- Can we synthesise this PDE with measurements $\mathbf{y}_{1:n} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ obtained at wave gauges (labelled above)?

Motivation and Goals

However, the KdV equation,

$$\partial_t u + \alpha u u_s + \beta u_{sss} + c u_s = 0,$$

is an idealised model of reality, and may not match the data.

- Parameters: α drives nonlinear advection, β drives dispersion, and c is latent advection: well-known and able to be calculated!
- How to use the data to correct for model mismatch?
- Let $u_n = u(s, n\Delta_t)$ (the model at some discretised time n). To correct for this mismatch we will estimate the posterior $p(u_n \mid \mathbf{y}_{1:n}, \Lambda)$.
- How can we construct this filtering distribution?

Underlying dynamical process

- Use stochastic dynamics. Take the KdV equation:

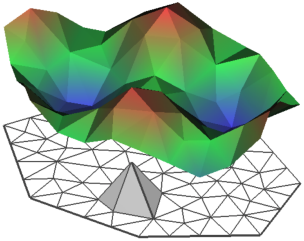
$$\partial_t u + \alpha u u_s + \beta u_{sss} + c u_s = \xi_\theta,$$

  we have $u := u(s,t)$, $s \in \Omega$, $t \in [0,T]$.
- Model uncertainty with a Gaussian process (with θ known):

$$\xi_\theta(s,t) \sim \mathcal{GP}(0, \delta(t-t') \cdot k_\theta(s,s')), \quad k_\theta(s,s') = \rho^2 \exp(-\|s-s'\|^2/(2\ell^2)).$$

In practise we need to discretize: we use FEM.

- Construct a mesh, Ω_h, with mesh-size h: discrete approximation to Ω.
- Use finite elements on the mesh Ω_h, $u(s, n\Delta_t) \approx \sum_{i=1}^{n_u} u_i \phi_i(s)$.
- Hat-functions $\implies \phi_i(s_j) = \delta_{ij}$.
- FEM coefficients $\mathbf{u}(t) = (u_1(t), \ldots, u_{n_u}(t))$.



FEM (from Bakka, H. (2019), arXiv:1803.03765).

Underlying dynamical process: an (implied) prior distribution

- Time discretization: let $\mathbf{u}_n = (u_1(n\Delta_t), \ldots, u_{n_u}(n\Delta_t))$; use Crank-Nicolson for stability.
- Important! We conduct inference over the *basis function coefficients* $\implies$ discrete representation of the process.
- So, on the basis function coefficients we have a Markov process:

$$\mathcal{M}(\mathbf{u}_n, \mathbf{u}_{n-1}) = \mathbf{e}^n, \quad \mathbf{e}^n \sim \mathcal{N}(0, \Delta_t \mathbf{G}),$$

  for $\mathbf{G}_{ij} = \langle \phi_i, \langle k_\theta(\cdot,\cdot), \phi_j \rangle \rangle$. This is our discretized version of KdV!
- This provides the transition densities $p(\mathbf{u}_n \mid \mathbf{u}_{n-1}, \Lambda)$, which in turn provides the prior distribution $p(\mathbf{u}_n \mid \Lambda)$.

This begs the question: how to embed data within this model structure? Nonlinear filtering methods...

Underlying dynamical process: an (implied) prior distribution

- Time discretization: let $\mathbf{u}_n = (u_1(n\Delta_t), \ldots, u_{n_u}(n\Delta_t))$; use Crank-Nicolson for stability.
- Important! We conduct inference over the *basis function coefficients* $\implies$ discrete representation of the process.
- So, on the basis function coefficients we have a Markov process:

$$\mathcal{M}(\mathbf{u}_n, \mathbf{u}_{n-1}) = \mathbf{e}^n, \quad \mathbf{e}^n \sim \mathcal{N}(0, \Delta_t \mathbf{G}),$$

  for $\mathbf{G}_{ij} = \langle \phi_i, \langle k_\theta(\cdot,\cdot), \phi_j \rangle \rangle$. This is our discretized version of KdV!
- This provides the transition densities $p(\mathbf{u}_n \mid \mathbf{u}_{n-1}, \Lambda)$, which in turn provides the prior distribution $p(\mathbf{u}_n \mid \Lambda)$.

This begs the question: how to embed data within this model structure? Nonlinear filtering methods...

Model-data synthesis

With data $\mathbf{y}_n \in \mathbb{R}^{n_y}$, we write $\mathbf{y}_n = \mathbf{H}\mathbf{u}_n + \varepsilon_n$.

- Observation operator $\mathbf{H} : \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$.
- Noisy measurements $\varepsilon_n \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I})$.

And this gives the state-space model

$$\begin{aligned} \mathcal{M}(\mathbf{u}_n, \mathbf{u}_{n-1}) = \mathbf{e}^n, \quad \mathbf{e}^n \sim \mathcal{N}(\mathbf{0}, \Delta_t \mathbf{G}), \quad &\text{(Transition)} \\ \mathbf{y}_n = \mathbf{H}\mathbf{u}_n + \varepsilon_n, \quad \varepsilon_n \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I}). \quad &\text{(Observation)} \end{aligned}$$

Compute the posterior $p(\mathbf{u}_n \mid \mathbf{y}_{1:n}, \Lambda)$ using nonlinear filtering methods: extended Kalman filter (ExKF).

Extended Kalman filtering

Let's start with $p(\mathbf{u}_{n-1} \mid \mathbf{y}_{1:n-1}, \Lambda) = \mathcal{N}(\mathbf{m}_{n-1}, \mathbf{C}_{n-1})$. Then:

1. Prediction step:

$$\hat{\mathbf{m}}_n \text{ solves } \mathcal{M}(\hat{\mathbf{m}}_n, \mathbf{m}_{n-1}) = 0,$$

$$\hat{\mathbf{C}}_n = \mathbf{J}_n^{-1} \left(\mathbf{J}_{n-1} \mathbf{C}_{n-1} \mathbf{J}_{n-1}^\top + \Delta_t \mathbf{G} \right) \mathbf{J}_n^{-\top},$$

   where $\mathbf{J}_n = \frac{\partial \mathcal{M}}{\partial \mathbf{u}_n}(\hat{\mathbf{m}}_n, \mathbf{m}_{n-1})$. So $p(\mathbf{u}_n \mid \mathbf{y}_{1:n-1}, \Lambda) = \mathcal{N}(\hat{\mathbf{m}}_n, \hat{\mathbf{C}}_n)$.

2. Update step:

$$\mathbf{m}_n = \hat{\mathbf{m}}_n + \left(\mathbf{H}\hat{\mathbf{C}}_n \right)^\top \left(\mathbf{H}\hat{\mathbf{C}}_n \mathbf{H}^\top + \sigma_n^2 \mathbf{I} \right)^{-1} (\mathbf{y}_n - \mathbf{H}\hat{\mathbf{m}}_n)$$

$$\mathbf{C}_n = \hat{\mathbf{C}}_n - \left(\mathbf{H}\hat{\mathbf{C}}_n \right)^\top \left(\mathbf{H}\hat{\mathbf{C}}_n \mathbf{H}^\top + \sigma_n^2 \mathbf{I} \right)^{-1} \mathbf{H}\hat{\mathbf{C}}_n.$$

   Then $p(\mathbf{u}_n \mid \mathbf{y}_{1:n}, \Lambda) = \mathcal{N}(\mathbf{m}_n, \mathbf{C}_n)$.

Aside: scaling to high dimensions

ExKF works for low-dimensional systems but is not scalable! How to scale the method?

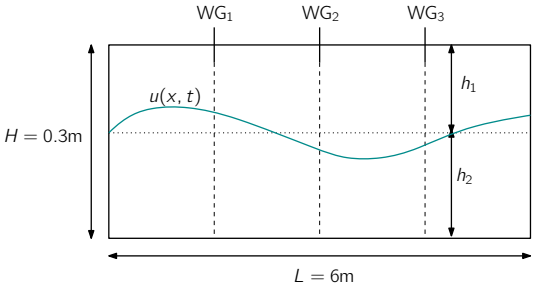
- To compute posterior $p(\mathbf{u}_n \mid \mathbf{y}_{1:n}, \Lambda)$ we use a low-rank Extended Kalman filter (LR-ExKF).
- Idea: $\mathcal{GP}$ covariance matrices (typically) only need a few dominant modes (eigenvector/value pairs) to describe the system. Leverage this inside of ExKF.
- LR-ExKF constructs approximate measure $p(\mathbf{u}_n \mid \mathbf{y}_{1:n}, \Lambda) = \mathcal{N}(\mathbf{m}_n, \mathbf{L}_n\mathbf{L}_n^\top)$, $\mathbf{m}_n \in \mathbb{R}^{n_u}$, $\mathbf{L}_n \in \mathbb{R}^{n_u \times k}$, $k \ll n_u$[1].
- Low-rank approximation is optimal in the ℓ^2 sense so UQ is sensible (not the case with, e.g., EnKF).

[1] Connor Duffin et al. "Low-Rank Statistical Finite Elements for Scalable Model-Data Synthesis". *Journal of Computational Physics* (Aug. 2022).

Case study: KdV experimental data

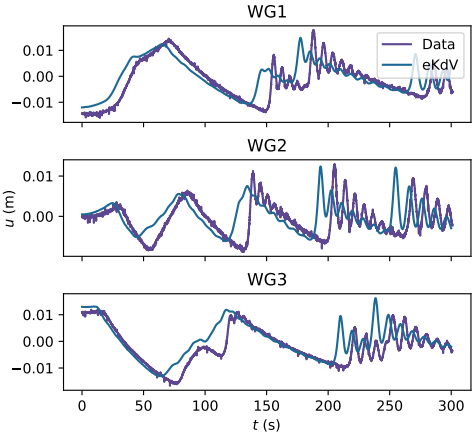
Case study: experimental data

Recall the experimental setup introduced in the first slide:

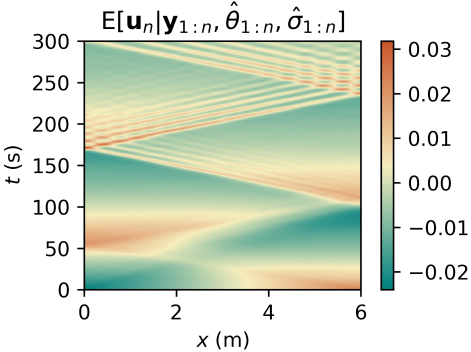


Apply statFEM to compute posterior $p(\mathbf{u}_n \mid \mathbf{y}_{1:n}, \Lambda)$ given the observations at each timestep. Observations $y_n = (u_n^{\mathrm{WG}_1}, u_n^{\mathrm{WG}_2}, u_n^{\mathrm{WG}_3})$, taking each of the $n_T = 1001$ timesteps for $0 \leq t \leq 300$ s.

Case Study: internal waves

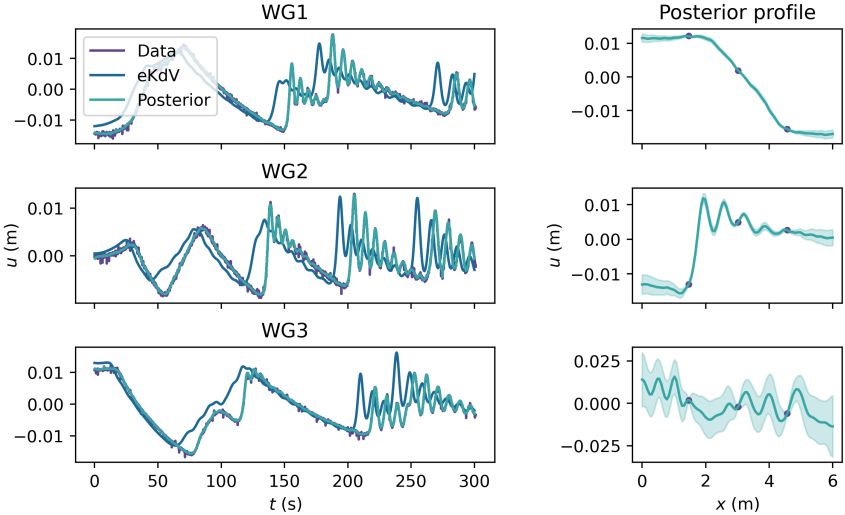


Experimental data and prior mean, up to time $t = 300$s.



KdV posterior mean across space-time grid.

Case study: internal waves



StatFEM posterior measure $p(\mathbf{u}_n \mid \mathbf{y}_{1:n}, \Lambda)$ for the KdV equation: posterior at WG locations (left); posterior wave profile $u(s,t)$ for $t = \{75, 150, 225\}$ s (right).

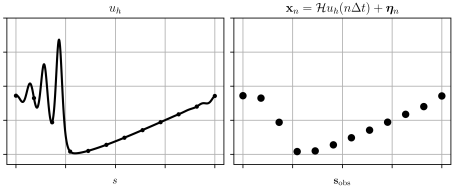
Case Study: Review and Conclusions

- Assimilated data with KdV equation: allows for physics-informed interpolator, with an interpretable posterior distribution.
- Uncertainty quantification is sensible and enables the calibration of simpler physical models with potentially sparse data.
- Next question: what if we don't know the observation operator?

Extending statFEM to Unkown Observation Operators

Known Observation Operator

Phenomena



Known observation operator for KdV

Mechanistic Model

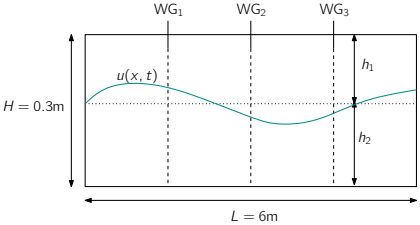
- We know that statFEM gives us transition densities of the form $p(\mathbf{u}_n \mid \mathbf{u}_{n-1}, \Lambda)$.
- These are derived from mechanistic descriptions (PDEs).

- Classical, known observation model.
- Can synthesise observed data with physical model

Observation Model

$$p(\mathbf{x}_n|\mathbf{u}_n) = \mathcal{N}(\mathbf{H}\mathbf{u}_n, \mathbf{R}),$$

$$\mathbf{H} : \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}.$$



Unknown Indirect Observation Operator

What if observation operator is unknown?

- That is, what if $\mathbf{y}_n = \mathcal{G}_\theta(\mathbf{u}_n) + \varepsilon_n$, for some learnable function $\mathcal{G}_\theta(\cdot)$.
- Use neural nets to learn this embedding from unstructured data into known mechanistic description.
- Mechanistic information used to identify the embedding: not to learn approximations to solution fields.
- Example: process is recorded with video camera, multi-channel recordings are taken (e.g., audio data).

How can we synthesise the phenomena with the mechanistic representation when we do not have an observation model?

Unknown Indirect Observation Operator

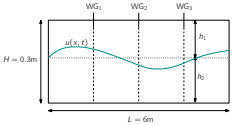
What if observation operator is unknown?

- That is, what if $\mathbf{y}_n = \mathcal{G}_\theta(\mathbf{u}_n) + \varepsilon_n$, for some learnable function $\mathcal{G}_\theta(\cdot)$.
- Use neural nets to learn this embedding from unstructured data into known mechanistic description.
- Mechanistic information used to identify the embedding: not to learn approximations to solution fields.
- Example: process is recorded with video camera, multi-channel recordings are taken (e.g., audio data).

How can we synthesise the phenomena with the mechanistic representation when we do not have an observation model?

Unknown observation operator: some examples

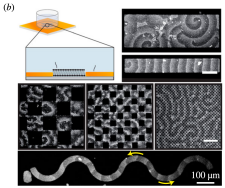
Measurement | **Phenomena** | **Mechanistic Representation**



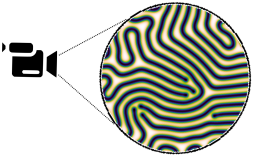
Observed internal wave.

Example: Korteweg-de Vries equation:

$$\partial_t u + \alpha u u_s + \beta u_{sss} + c u_s = 0.$$



Observed species concentrations.

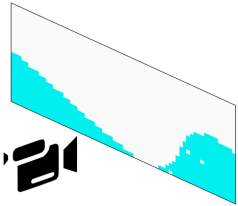


Example: Gray-Scott equation:

$$\partial_t u = D_u \nabla^2 u - uv^2 + F(1-u),$$
$$\partial_t v = D_v \nabla^2 v + uv^2 - (F+k)v.$$

Unknown Observation Operator

Phenomena



Mechanistic Representation

KdV equation:

$$\partial_t u + \alpha u u_s + \beta u_{sss} + c u_s = 0.$$

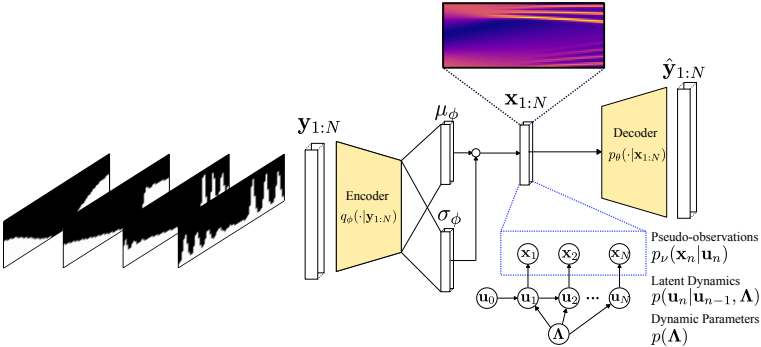
- Observation operator can be approximated with deep neural networks.
- We *posit* an observation operator of the form:

$$p(\mathbf{y}_n | \mathbf{u}_n) = \mathcal{N}(\mathcal{G}_\phi(\mathbf{u}_n), \mathbf{R}), \quad \mathcal{G} : \mathbb{R}^{n_u} \to \mathbb{R}^{n_y \times n_c}.$$

- Learn this embedding of the data to observations of the mechanistic system in a variational inference framework.

Φ-DVAE

- Phenomenological data received through time: $\mathbf{y}_{1:N}$ (e.g., video feeds).
- Encoded to latent mechanistic observations $\mathbf{x}_{1:N}$ using a variational autoencoder (VAE).
- Mechanistic representation embedded into latent space, driving latent stochastic dynamics with statFEM.



Φ-DVAE: Probabilistic Model Structure

We propose the following hierarchical probabilistic model:

- Parameter prior: $\Lambda \sim p(\Lambda)$.
- Transition density: $\mathbf{u}_n \mid \mathbf{u}_{n-1}, \Lambda \sim p(\mathbf{u}_n \mid \mathbf{u}_{n-1}, \Lambda)$ (assumed known form).
- Pseudo-observations: $\mathbf{x}_n \mid \mathbf{u}_n \sim p(\mathbf{x}_n \mid \mathbf{u}_n)$ (assumed known form).
- Decoder distribution: $\mathbf{y}_n \mid \mathbf{x}_n \sim p_\theta(\mathbf{y}_n \mid \mathbf{x}_n)$.

Following VAEs, we also introduce the "encoder" variational approximation, $q_\phi(\mathbf{x}_{1:N} | \mathbf{y}_{1:N}) = \mathcal{N}(\mu_\phi, \sigma_\phi)$, and the parameter posterior $p(\Lambda \mid \mathbf{y}_{1:N}) \approx q_\lambda(\Lambda)$.

How can we conduct joint parameter inference over $\{\Lambda, \theta, \phi\}$?

Φ-DVAE: Probabilistic Model Structure

We propose the following hierarchical probabilistic model:

- Parameter prior: $\Lambda \sim p(\Lambda)$.
- Transition density: $\mathbf{u}_n \mid \mathbf{u}_{n-1}, \Lambda \sim p(\mathbf{u}_n \mid \mathbf{u}_{n-1}, \Lambda)$ (assumed known form).
- Pseudo-observations: $\mathbf{x}_n \mid \mathbf{u}_n \sim p(\mathbf{x}_n \mid \mathbf{u}_n)$ (assumed known form).
- Decoder distribution: $\mathbf{y}_n \mid \mathbf{x}_n \sim p_\theta(\mathbf{y}_n \mid \mathbf{x}_n)$.

Following VAEs, we also introduce the "encoder" variational approximation, $q_\phi(\mathbf{x}_{1:N} | \mathbf{y}_{1:N}) = \mathcal{N}(\mu_\phi, \sigma_\phi)$, and the parameter posterior $p(\Lambda \mid \mathbf{y}_{1:N}) \approx q_\lambda(\Lambda)$.

How can we conduct joint parameter inference over $\{\Lambda, \theta, \phi\}$?

Φ-DVAE: Variational Lower Bound

- Encoding, decoding, and model parameters are all jointly learnt through optimising a variational lower bound.
- Evidence lower bound provides a tractable target for optimisation:

$$\log p(\mathbf{y}_{1:N}) \geq \mathbb{E}_{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})}\left[\log \frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})}{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})} + \mathbb{E}_{q_\lambda}\left[\log p(\mathbf{x}_{1:N} \mid \Lambda) + \log \frac{p(\Lambda)}{q_\lambda(\Lambda)}\right]\right].$$

- First term: encoder/decoder. Second term: pseudo-observations

$$p(\mathbf{x}_{1:N} \mid \Lambda) = \int p(\mathbf{u}_{1:N}, \mathbf{x}_{1:N} \mid \Lambda)\, d\mathbf{u}_{1:N}.$$

Marginalising over the dynamics acts as a "physics informed regulariser". Third term: variational parameter posterior KL divergence.

Φ-DVAE: Case studies

Φ-DVAE: Simulation Studies

- We now go through a selection of simulation studies using Φ-DVAE.
- We look at (variational) parameter inference and filtering inference, $p(\mathbf{u}_{1:n}|\mathbf{x}_{1:n})$.
- We look at 2 particular systems: the classic Lorenz-63 system, and the (hopefully, now-familiar) KdV equation.
- We simulate synthetic data consisting of velocity fields, for the Lorenz-63 case, and video data, for the KdV case. These are our $\mathbf{y}_{1:N}$.
- We aim to learn the mapping from $\mathbf{y}_{1:N} \to \mathbf{x}_{1:N}$, thus inferring the latent state $\mathbf{u}_n$, conditioned on $\mathbf{y}_{1:n}$.

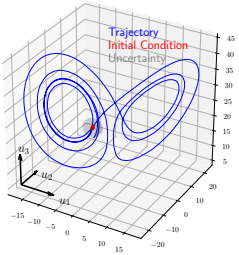
Lorenz-63 Dynamical System: Illustrative Example

Data $\mathbf{y}_{1:N}$ are simulated velocity field measurements, which are modulated by the first dimension of a latent stochastic Lorenz-63 system:

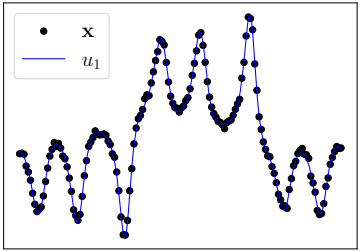
$$du_1 = -\sigma u_1 + \sigma u_2 + dw_1, \quad du_2 = -u_1 u_3 + r u_1 - u_2 + dw_2, \quad du_3 = u_1 u_2 - b u_3 + dw_3,$$

so now $\Lambda = \sigma$, $p(\Lambda) = \mathcal{N}(30, 5^2)$, and $q_\lambda(\Lambda) = \mathcal{N}(\mu_\lambda, \sigma_\lambda^2)$.

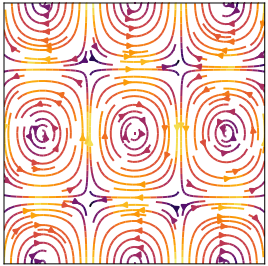
Latent Dynamics: $\mathbf{u}_{1:N}$



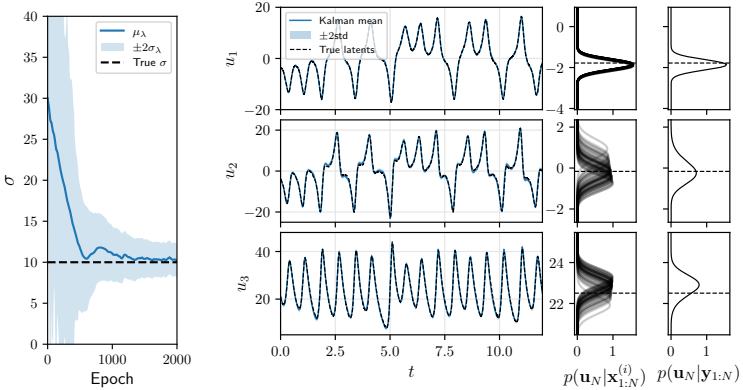
Pseudo-Observations: $\mathbf{x}_{1:N}$



Velocity Field: $\mathbf{y}_N$

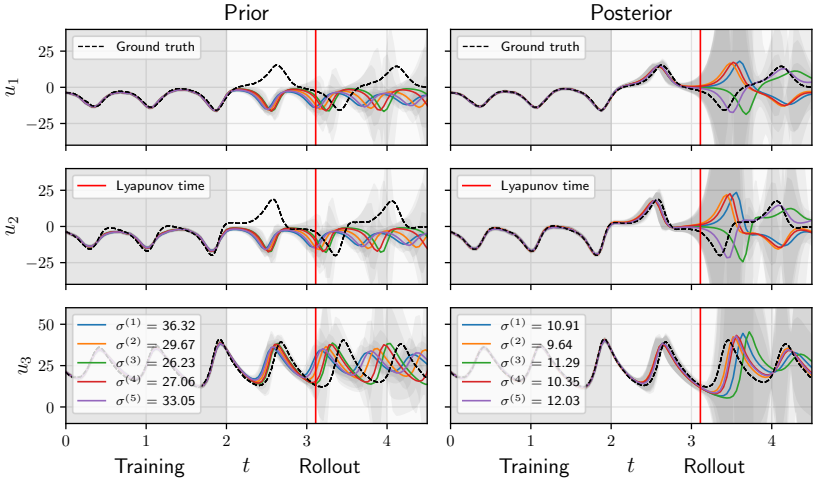


Lorenz-63: State and Parameter Inference



Left: "trace plot" of parameter variational distribution $q_\lambda(\mathbf{\Lambda}) = \mathcal{N}(\mu_\lambda, \sigma^2_\lambda)$, with mean (blue) and ± 2 standard deviations (blue fill). Right: filtering inference for latent states $\mathbf{u}_{1:N}$, where the filtering distribution $p(\mathbf{u}_n|\mathbf{x}_{1:n})$ is plotted with the ground truth $\mathbf{u}_n^{\text{true}}$.

Lorenz-63: Rolling Out Beyond Training



"Rollout": training time indicated with grey-fill, with (left) showing samples generated with the prior (left), and the posterior (right) variational distribution $q_\lambda(\cdot)$.

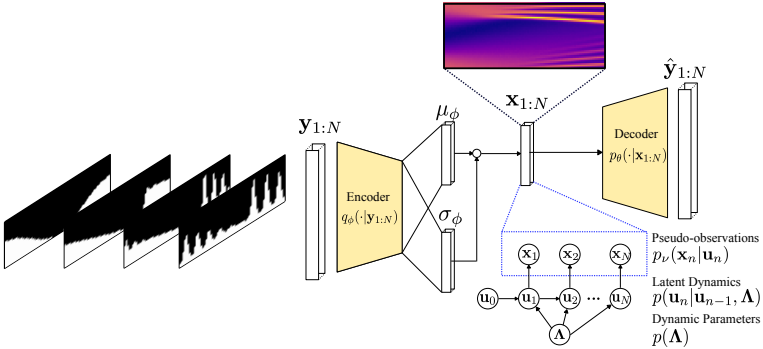
KdV: Learning the Observation Operator and Drag Coefficient

- In this final example we return to KdV: we generate synthetic video data (a sequence of images), giving our $\mathbf{y}_{1:N}$, from a governing KdV equation:

$$\partial_t u + \alpha u u_x + \beta u_{xxx} + \nu u = \xi_\theta.$$

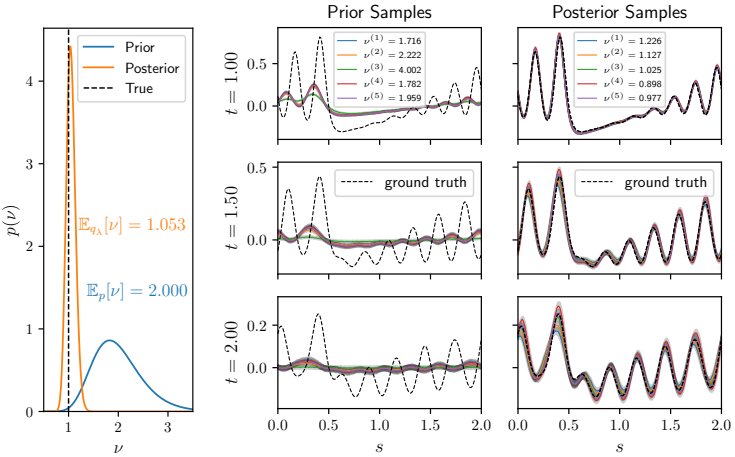
  We jointly estimate the embedding and the drag coefficient ν, so $\Lambda = \nu$, $p(\Lambda) = \mathcal{LN}(2, 0.5^2)$, $q_\lambda(\Lambda) = \mathcal{LN}(\mu_\lambda, \sigma_\lambda^2)$.
- Weakly-informative log-normal prior for the drag coefficient as $\nu > 0$.
- Encoding and decoding networks are MLPs with 3 hidden layers of width 128.

KdV: Learning the Observation Operator and Drag Coefficient



A reminder: video frames $\mathbf{y}_{1:N}$ are encoded to pseudo-observations $\mathbf{x}_{1:N}$ of a latent dynamical system with a known transition density $p(\mathbf{u}_n \mid \mathbf{u}_{n-1}, \Lambda)$. Φ-DVAE infers the encoder $q_\phi(\cdot)$, the decoder $p_\theta(\cdot \mid \mathbf{y}_{1:N})$, *and* parameters $q_\lambda(\cdot)$.

KdV: Results with drag coefficient estimation



Results for KdV with drag: (left) comparison of prior and variational posterior for model parameter $\nu = 1$. Right: latent filtering distribution for prior and posterior parameter estimates.

Conclusions

- Statistical FEM construction allows for the construction of physics-informed interpolators which naturally include nonstationarity.
- Enables interpolation and inference in sparse data settings.
- Interpretable and relatively robust framework to conduct inference with some latent dynamical process.
- Φ-DVAE enables the use of such methods when the mapping to observations is not known, and, when the parameters may be uncertain.
- Additional work on Langevin dynamics samplers for static problems (Akyildiz et al.).
- Current work focusses on applications (shallow water flows) and on generalising the framework to enable parameter estimation.

Thanks!

References I

Akyildiz, Ömer Deniz, et al. "Statistical Finite Elements via Langevin Dynamics". *SIAM/ASA Journal on Uncertainty Quantification* 10, no. 4 (Dec. 2022): 1560–1585. https://doi.org/10.1137/21M1463094.

Bakka, Haakon. "How to Solve the Stochastic Partial Differential Equation That Gives a Mat\'ern Random Field Using the Finite Element Method". *arXiv:1803.03765 [stat]* (Oct. 2019). arXiv: 1803.03765 [stat].

Duffin, Connor, et al. "Low-Rank Statistical Finite Elements for Scalable Model-Data Synthesis". *Journal of Computational Physics* 463 (Aug. 2022). https://doi.org/10.1016/j.jcp.2022.111261.

– . "Statistical Finite Elements for Misspecified Models". *Proceedings of the National Academy of Sciences* 118, no. 2 (Jan. 2021). https://doi.org/10.1073/pnas.2015006118.

References II

Glyn-Davies, Alex, et al. *Φ-DVAE: Physics-Informed Dynamical Variational Autoencoders for Unstructured Data Assimilation*, arXiv:2209.15609, July 2023. https://doi.org/10.48550/arXiv.2209.15609. arXiv: 2209.15609 [physics, stat].