Two-staged neural likelihood surrogate:

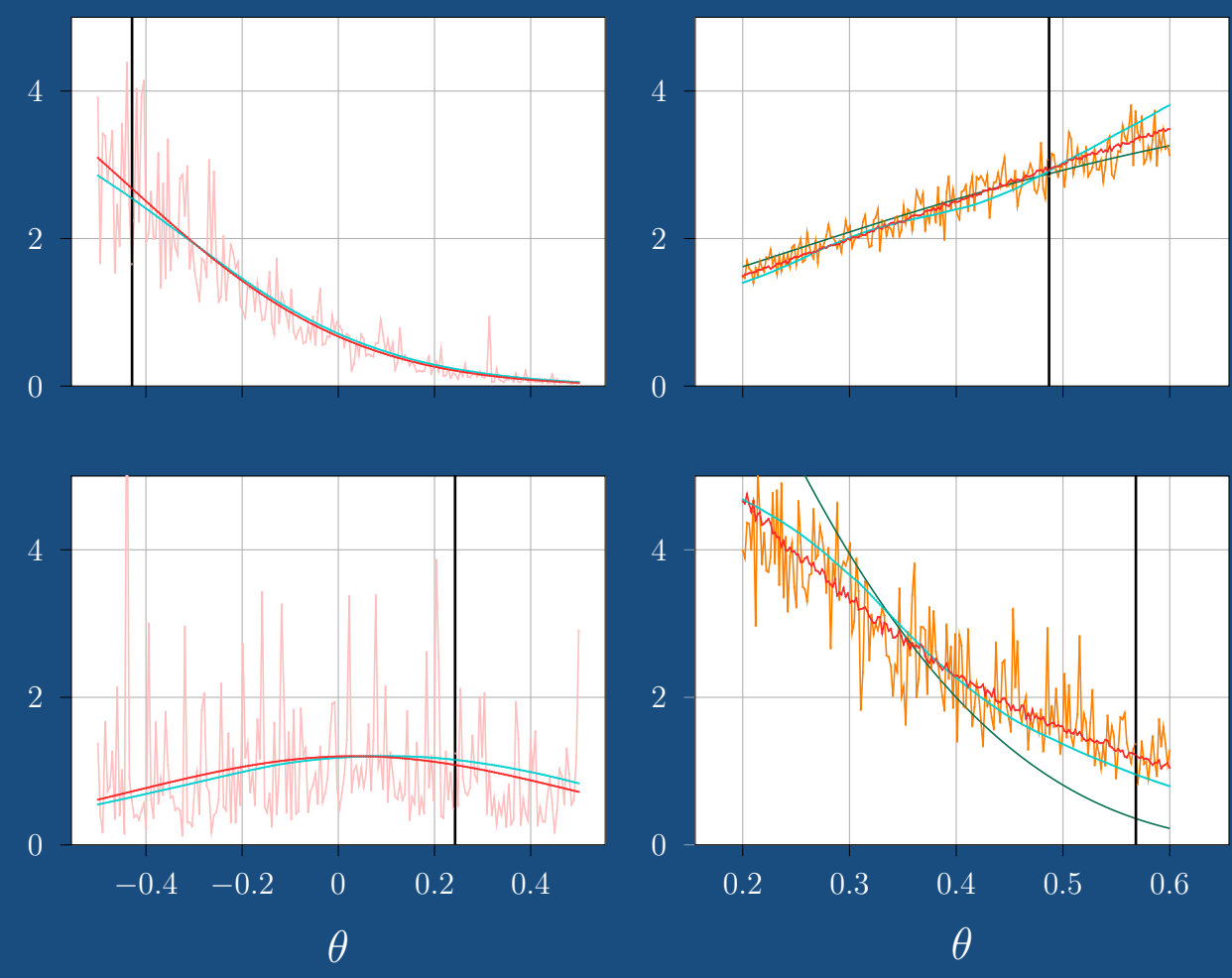
1. Apply a deep density method of the parametric log-transformed Fokker–Planck equations
2. Learn the neural surrogate for the log-normalizer

Results in our two numerical examples. Arrows indicate whether lower is better (↓) or higher is better (↑).

	Ornstein–Uhlenbeck				Bistable			
	LLE (↓)	MAPE (↓)	PLL (↑)	Time	LLE (↓)	MAPE (↓)	PLL (↑)	Time
PF 10^3	5.001	0.219	-158.346	(4.1m)	0.017	0.042	-16.017	(1.7m)
PF 10^4	0.772	0.157	-157.347	(15m)	0.006	0.018	**-16.006**	(5.3m)
PF 10^5	0.146	0.099	-157.047	(1.2h)	**0.005**	**0.010**	-16.007	(39m)
PF 10^6	0.034	0.056	**-156.983**	(16h)	-	-	-	(-)
EKF	-	-	-	-	0.163	0.024	-16.043	(51s)
EnKF 10^3	1.207	0.177	-157.443	(5.0m)	0.071	0.058	-16.031	(1.2m)
EnKF 10^4	0.104	0.100	-157.068	(16m)	0.049	0.034	-16.024	(1.8m)
EnKF 10^5	**0.020**	0.055	-156.983	(1.3h)	0.047	0.023	-16.024	(10m)
Ours	0.077	**0.024**	-157.054	(**0.3s**)	0.009	0.027	-16.017	(**0.3s**)

Ornstein–Uhlenbeck posterior

Bistable posterior



PF 10^3 PF 10^4 EKF Ours Reference True Parameter

Posterior densities for two parameter samples and accompanying observation chains for both examples.

KASPER BÅGMARK FILIP RYDIN

CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

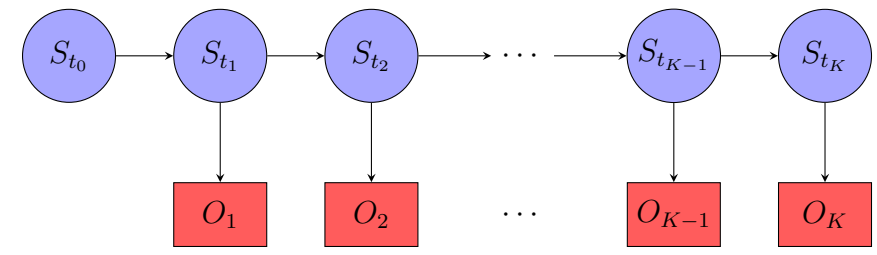
WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM

Neural likelihood surrogates for parameter inference via log-density PDE

Bayesian parameter estimation

We assume an underlying state space model

$$\begin{aligned} &\text{(State)} && \mathrm{d}S_t = \mu(S_t,\theta)\,\mathrm{d}t + \sigma(S_t,\theta)\,\mathrm{d}B_t, \quad t \in (0,T] \\ &&& S_0 \sim p_0(\cdot \mid \theta) \\ &\text{(Observations)} && O_k = h(S_{t_k},\theta,\xi_k), \quad k = 1,\dots,K \end{aligned}$$



Task: Learn a surrogate for $p(\theta \mid O_{1:K})$

Stage I

- Find the probability density $p(S_{t_k} \mid O_{1:k},\theta)$ (**filtering density**)

Log-density PDE formulation

The filtering density $p = (p_k)_{k=0}^K$, where $p_k \in C^{1,2}([t_k,t_{k+1}]\times\mathbb{R}^d;\mathbb{R})$, associated to ($S$, O) and conditional on θ, satisfies the Fokker–Planck equation with discrete measurement updates. Hence the log formulation, with $v_k = -\log p_k$, satisfies for $k = 0,\dots,K$ and $t \in (t_k,t_{k+1}]$

$$\begin{aligned} &\text{(Prediction)} && v_k(t \mid O_{1:k},\theta) = v_k(t_k \mid O_{1:k},\theta) \\ &&& \qquad + \int_{t_k}^t \left(A^\theta + F^\theta_{\log}\right) v_k(s \mid O_{1:k},\theta)\,\mathrm{d}s \\ &\text{(Update)} && v_{k+1}(t_{k+1} \mid O_{1:k+1},\theta) = v_k(t_{k+1} \mid O_{1:k},\theta) \\ &&& \qquad - \log\left(L(O_{k+1},\theta)\right) + \log p(O_{k+1} \mid O_{1:k},\theta). \end{aligned}$$

where the likelihood L is tractable

$$L(o,x,\theta) := p(O_k = o \mid S_{t_k} = x,\theta).$$

- Approximate the Prediction PDE by a deep BSDE method

Stage II

From Bayes' formula, the log-posterior can be decomposed as

$$\log p(\theta \mid O_{1:K}) = \log p(\theta) + \sum_{k=1}^K \log p(O_k \mid O_{1:k-1},\theta).$$

- Learn a surrogate for each $\log p(O_k \mid O_{1:k-1},\theta)$.

Note: This is the normalizing term in the log-density Update.

We train a network with supervised labels, produced by learned filtering densities from Stage 1.

Algorithm 1 Training pipeline

Input: $p_0(\cdot \mid \theta)$, $\mu(\theta),\sigma(\theta),h(\theta)$, prior $p(\theta)$
Initialize $\bar{v}_0 = -\log p_0$
for $k = 0,\dots,K-1$ **do**
  Stage I: conditional log-filter
  Initialize $u,(\bar{w}_n)_{n=0}^{N-1}$
  for $\ell = 0,\dots,L-1$ **do**
    Sample $\theta^{(m)} \sim p(\theta)$
    Sample $(\mathcal{X}_n^{k,(m)})_{0:N}$, $O_{1:k}^{(m)}$ with $\theta^{(m)}$
    Compute prediction $\mathcal{Y}^{k,(m)}$
    Target $\bar{v}_k(\mathcal{X}^{k,(m)},O_{1:k}^{(m)},\theta^{(m)})$
    Update $u,(\bar{w}_n)$ via MSE loss
  end for
  Set $u_k^* = u$
  Stage II: log-normalizer surrogate
  for $\ell = 0,\dots,L-1$ **do**
    Sample $\theta^{(m)}$, $O_{1:k+1}^{(m)}$
    Target $C(O_{1:k+1}^{(m)},\theta^{(m)})$
    Update Φ_{k+1} via MSE loss
  end for
  $\bar{v}_{k+1} = u_k^* - \log L + \Phi_{k+1}$
end for
Output: $(\bar{v}_k,\Phi_k)_{k=1}^K$