A likelihood-free approach to Estimate Parameter Posterior Distribution Using Sequential Neural Networks

Bosi Hou, Jonathon Rubin
Department of Statistics, Department of Mathematics, University of Pittsburgh

CONTEXT:

- Computational models often have unknown **parameters** that we need to fit data
- Recent work uses machine learning to train a neural network to estimate suitable probability distribution of **parameters**
- This idea can significantly reduce computational cost in fitting new data

GOALS:

- Train neural networks on data from Rosenzweig-McArthur population dynamics model
- Utilize trained networks to estimate posterior distribution on parameters given new data
- Compare the performance of different training methods

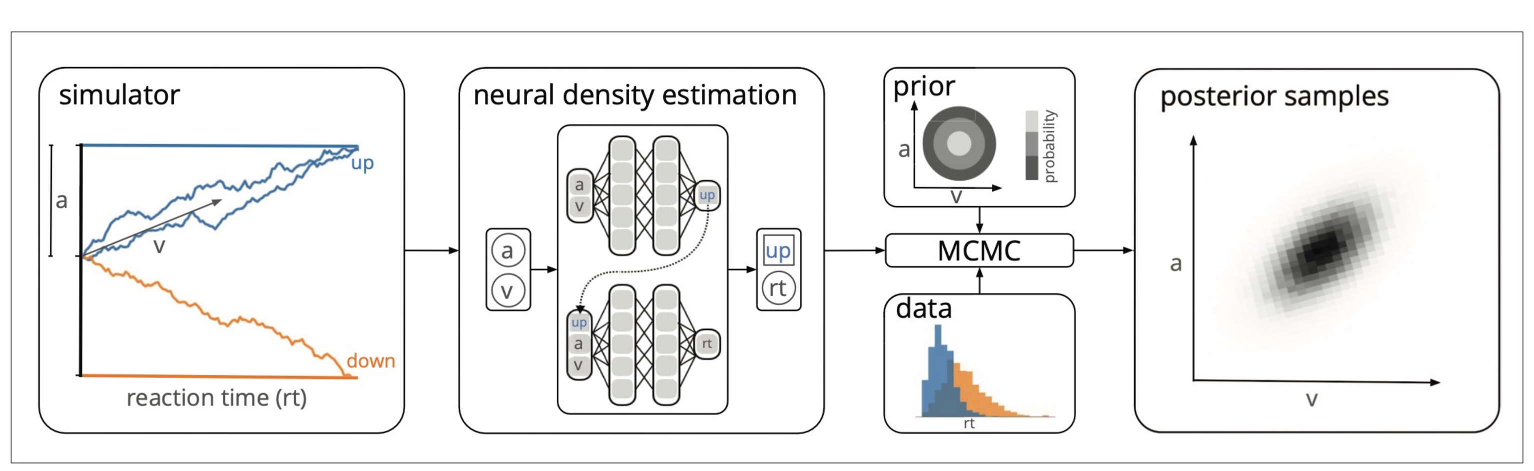
BACKGROUND:

- Density Estimation Neural Network (DENN) is a special type of neural network designed to estimate probability density
- DENN applies Bayesian framework, which is demonstrated as follows:

$$\theta \sim p(\theta) \quad x \sim f(x|\theta)$$
$$\theta|x \propto f_n(\boldsymbol{x}|\theta)p(\theta)$$

- Traditional DENN requires explicit likelihood function and output a parametric density model. However, likelihood function is often untenable in real-world setting.
- We introduce a simulation-based model which provides faster inference and does not require likelihood function.
- The model use SNN (Sequential Neural Network) for estimation. Typical workflow is presented as the following:

Estimating posterior distribution of parameters of Drift Diffusion Model



Specify simulator   Train SNN   Sample from the posterior for analysis

METHODOLOGY:

SNN Set-up

- We employ SNN with the following three elements
  - **Prior**: a prior distribution of parameter set
  - **Simulator:** relationship between parameters and data
  - **Training method:** The type of SNN using for network training
- The trained network takes observation data as argument and output posterior samples of parameters.

$$Observation\ data \xrightarrow{SNN} posterior\ sample$$

Training methods: SNLE & SNPE

- **SNLE** (Sequential Neural Posterior Estimation) and **SNPE** (Sequential Neural Likelihood Estimation) are often good choice to do the work
- Both use synthetic data for sequential network training. Specifically, they obtain data from the current estimate of the likelihood and uses this data to train a new estimate of the likelihood function.
- The difference lies on the target. SNLE focus on maximizing the likelihood during training; SNPE focus on estimate an actual shape of posterior distribution

Data production from R-M model

- The Rosenzweig-MacArthur predator-prey dynamics model is defined by the following ODE system:

$$\begin{cases} \dfrac{dx}{dt} = rx\left(1-\dfrac{x}{K}\right) - \dfrac{axy}{x+h} \\ \dfrac{dy}{dt} = \dfrac{abxy}{x+h} - my \end{cases}$$

- m, h, r, a, b, k are parameters. When the following condition meets, the model will present oscillations

$$h < K\frac{ab-m}{ab+m}, \ m < ab$$

- we replace individual parameters with a CIR stochastic processes, specified as follows:

$$dp = \gamma(\bar{p} - p)dt + \sigma\sqrt{p}dW$$

- $\bar{p}$ is baseline value of one parameters. For training, we choose $m = 0.6, h = 0.15, r = 1, a = 2, b = 0.5, k = 1$.
- from each oscillation cycle, we compute three features: X_{amp}, Y_{amp}, T, where amp stands for amplitude, and T stands for oscillation period.

$$X_{amp} = \max(X) - \min(X)$$
$$Y_{amp} = \max(Y) - \min(Y)$$
$$T = \sum \Delta t$$

SNN training & posterior sampling

- **Prior:** giving oscillation regime, we choose $[\mathrm{m}, \mathrm{h}]\ s.t.\, m + h \leq 0.8$. We define a box-uniform prior and map the data outside oscillation regime
- **Simulator:**

$$[m, h] \xrightarrow{yields} [X_{amp}, Y_{amp}, T]$$

- **Network training:**

$$prior(\theta)\ \&\ [X_{amp}, Y_{amp}, T] \xrightarrow{SNLE/SNPE} p(\theta|[X_{amp}, Y_{amp}, T])$$

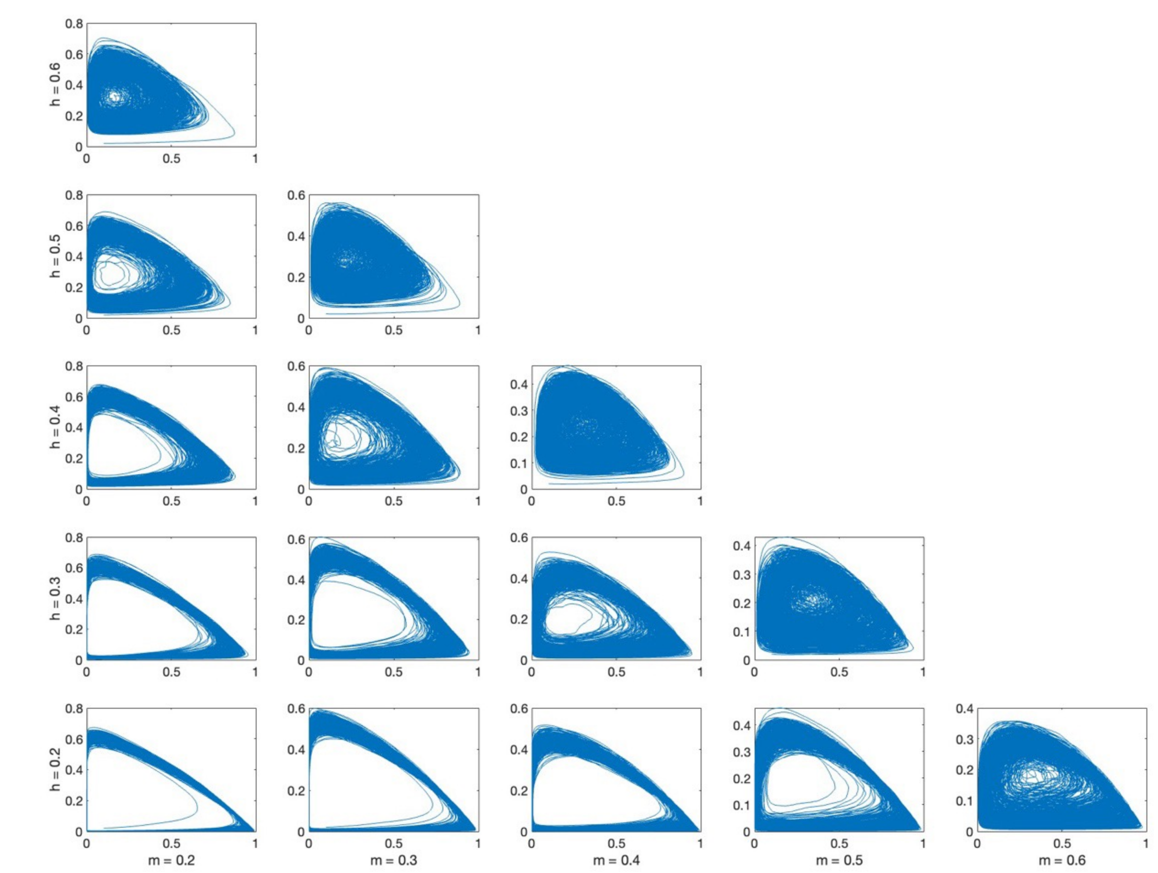
- **Obtaining posterior samples:**

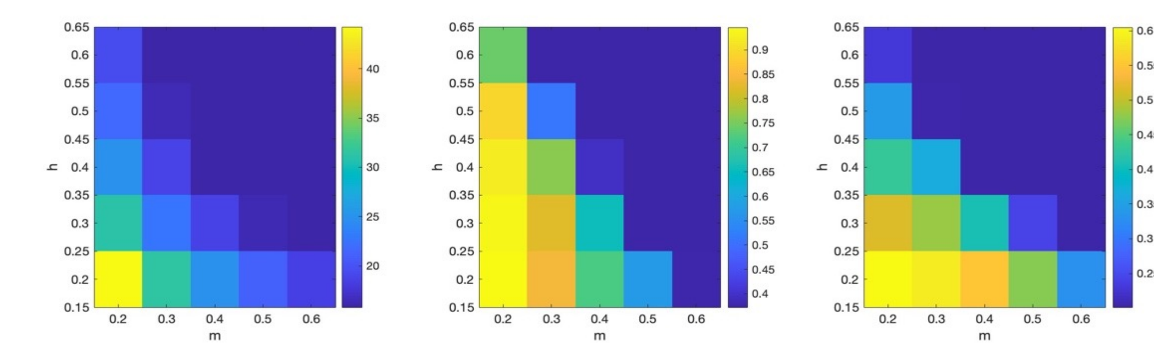
$$p(\theta|[X_{amp}, Y_{amp}, T]) \xrightarrow{MCMC\ sampling} \theta \mid observed\ [X_{amp}, Y_{amp}, T]$$

Note that we generate the data before actual training. The simulator is thus reduced to a data fetcher from data files. This provides even faster training and inference compared to proceeding simulation within the network

RESULTS:

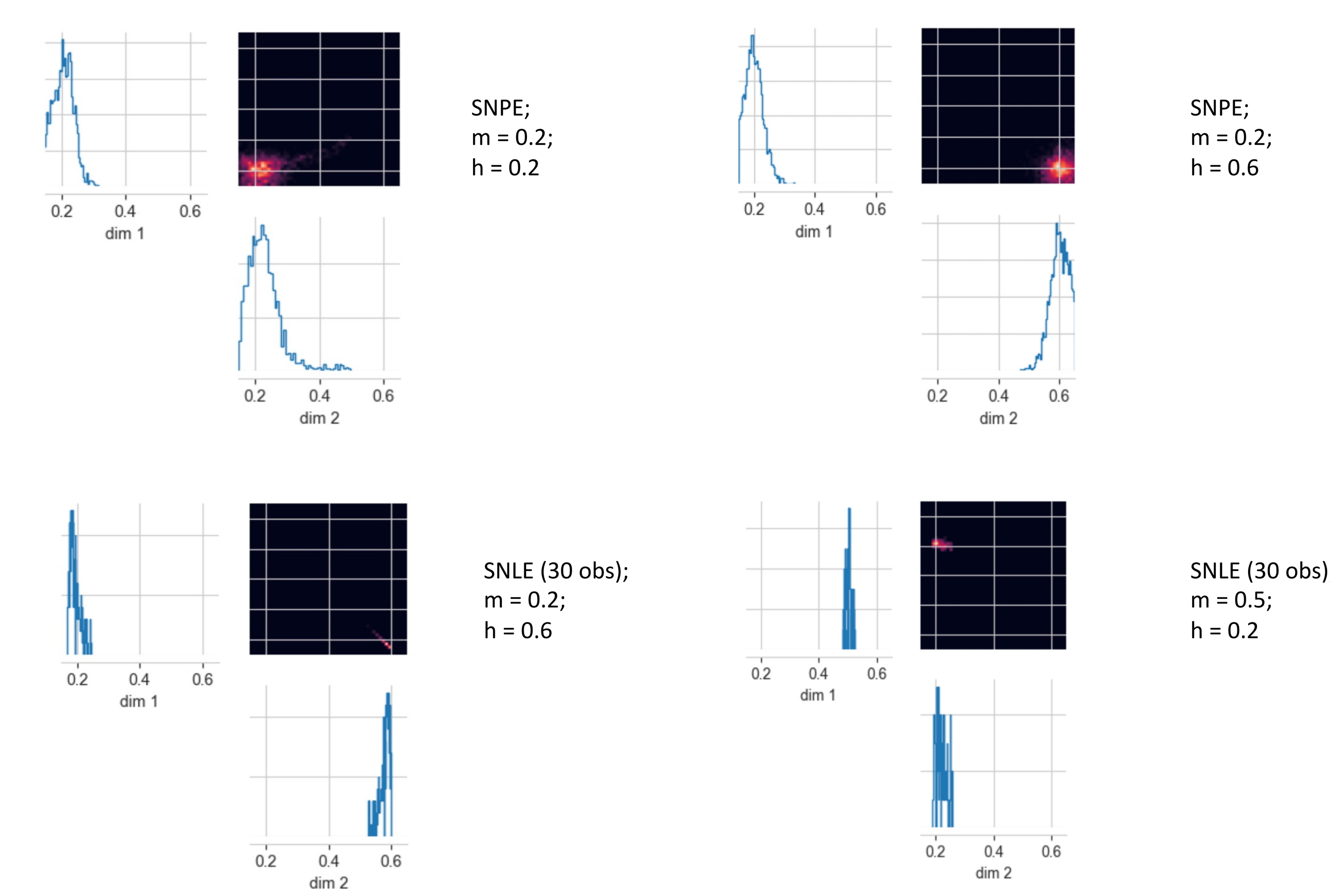
- Our data demonstrates good quality from network training





- The pair plot of posterior samples shows good prediction accuracy

  dim1 = m, dim2 = h



CONCLUSION:

- SNPE provides pointwise estimation, which is much faster than SNLE; SNLE allows multiple observations for posterior sampling, which is less sensitive to noise
- Prior distribution determines the parameter region to sample from, but ought not to affect the information given fitted data
- It is feasible to finish simulation process in minutes by pre-processing data and reducing the simulator in SNN to a data-fetcher

REFERENCE:

[1] Rubin et al., *HAL Open Science*, 2021
[2] Bishop, *NCRG,* Aston University, 1994
[3] Boelts et al, *eLife,* 2022
[4] Papamakarios and Murray, NIPS, 2016
[5] Papamakarios et al., PMLR, 2019