“Sequential Neural Likelihood: Fast Likelihood-free Inference with Autoregressive Flows”

CSC696

Shanrui Zhang

Overview

1.Likelihood-free inference

2.Neural density estimator

3.Sequential Neural Likelihood(SNL)

4.Experiments

5.Result

6.Discussion

Likelihood free inference

1.Bayesian inference $p(x \mid \theta)\, p(\theta)$,likelihood $p(x \mid \theta)$ computationally infeasible in general

2.Approximate Bayesian Computation, Synthetic Likelihood

- require only the ability to generate data from the simulator
- simulate the model repeatedly, and use the simulated data to build estimates of the parameter posterior
- improves as the number of simulations increases, hard to compute
- especially if the simulator is expensive to run
- achieve a good trade-off accuracy and simulation cost.

Sequential Neural Likelihood (SNL)

- Train a Masked Autoregressive Flow on simulated data->$p(x|\theta)$
- Serves as an accurate model of the likelihood function
- During training, a Markov Chain Monte Carlo sampler selects the next Sequential Neural Likelihood
- Fast Likelihood-free Inference with Autoregressive Flows batch of simulations to run using the most up-to-date estimate of the likelihood function

Simulator model

- Takes a vector of parameters θ => Output a data vector x
- Sample $p(x \mid \theta)$ by running the program
- X and a prior distribution $p(\theta)$
- Estimating $p(\theta \mid X) \propto p(X \mid \theta)\, p(\theta)$.

Conditional neural density estimator

- Parametric model q_φ (such as a neural network) controlled by a set of parameters φ
- Input: pair of datapoints (u, v)
- Output :conditional probability density $q_\varphi(u \mid v)$
- Training data: $\{Un, Vn\}_{1:N}$
- Maximizing the total log probability $\sum_n \log q_{\boldsymbol{\phi}}(\mathbf{u}_n \mid \mathbf{v}_n)$
- $q_\varphi(u \mid v)$ will learn to approximate the conditional $p(u \mid v)$.

Approximate posterior using Neural density estimator

- we obtain a set of samples $\{\theta_n, X_n\}_{1:N}$ from the joint distribution $p(\theta, x)$, by $\theta_n \sim p(\theta)$ and $X_n \sim p(x \mid \theta_n)$ for $n = 1, . . . , N$.
- we train q_φ using $\{\theta_n, X_n\}_{1:N}$ as training data in order to obtain a global approximation of $p(\theta \mid x)$.
- Large number of simulations required enough training data accurate posterior fit
- Expensive

Sequential Neural Posterior Estimation(SNPE)

- Reducing the number of simulations needed by conditional neural density estimation
- Generate parameter samples θ_n from a proposal $\tilde{p}(\theta)$ instead of the prior $p(\theta)$
- Finds a good proposal $\tilde{p}(\theta)$ by training the estimator q_φ over a number of rounds, whereby in each round $\tilde{p}(\theta)$ is taken to be the approximate posterior obtained in the round before
- For its neural density estimator, SNPE uses a Mixture Density Network, a feedforward neural network
- Input x output a Gaussian mixture over θ

Problems of SNPE

- Parameter samples follow $\tilde{p}(\theta)$ instead of $p(\theta)$
- Adjust posterior or proposed samples
- SNPE-A
- SNPE-B

SNPE-A

- Posterior $q_{\varphi}(\theta \mid X_0)$ is adjusted
- Dividing it by $\tilde{p}(\theta)$ and multiplying it by $p(\theta)$.
- SNPE-A restricts $\tilde{p}(\theta)$ to be Gaussian; since $q_{\varphi}(\theta \mid X_0)$ is a Gaussian mixture
- Problem: $\tilde{p}(\theta)$ happens to have a smaller variance than any of the components of $q_{\varphi}(\theta \mid X_0)$, the division yields a Gaussian with negative variance, from which the algorithm is unable to recover and thus is forced to terminate prematurely

SNPE-B

- Adjust parameter samples θ_n
- assigning them weights $w_n = p(\theta_n)/\tilde{p}(\theta_n)$
- During training, the weighted log likelihood $\sum_n w_n \log q_\phi(\boldsymbol{\theta}_n \mid \mathbf{x}_n)$ is used instead of the total log likelihood $\sum_n \log q_\phi(\mathbf{u}_n \mid \mathbf{v}_n)$
- Compared to SNPE-A, this method does not require the proposal $\tilde{p}(\theta)$ to be Gaussian, and it does not suffer from negative variances
- However, the weights can have high variance, which may result in high-variance gradients and instability during training.

Sequential Neural Likelihood(SNL)

- Avoids bias by proposal
- Learn likelihood instead of posterior

Sequential Neural Likelihood(SNL)

- Samples $\{\theta_n, X_n\}_{1:N}$ from the joint distribution $p(\theta, x)$, by $\theta_n \sim p\tilde{}(\theta)$ and $X_n \sim p(x \mid \theta_n)$ for $n = 1, \ldots, N$.
- Define $p\tilde{}(\theta, x) = p(x \mid \theta)\, \tilde{p}(\theta)$ to be the joint distribution of each pair (θ_n, X_n).
- Train a conditional neural density estimator $q_\varphi(x \mid \theta)$,

Sequential Neural Likelihood(SNL)

- Max total log likelihood $\sum_n \log q_\phi(\mathbf{x}_n \mid \boldsymbol{\theta}_n)$
- Approximately equivalent to maximizing $$\mathbb{E}_{\tilde{p}(\boldsymbol{\theta},\mathbf{x})}(\log q_\phi(\mathbf{x} \mid \boldsymbol{\theta})) = - \mathbb{E}_{\tilde{p}(\boldsymbol{\theta})}(D_{\mathrm{KL}}(p(\mathbf{x} \mid \boldsymbol{\theta}) \,\|\, q_\phi(\mathbf{x} \mid \boldsymbol{\theta}))) + \mathrm{const}$$
- Kullback–Leibler divergence $D_{\mathrm{KL}}(\cdot \,\|\, \cdot)$
- Maximum when KL is zero :$q_\varphi(x \mid \theta) = p(x \mid \theta)$ for all θ such that $\tilde{p}(\theta) > 0$
- Approximate the likelihood in the support of the proposal, regardless of the shape of the proposal.
- The way we propose parameters does not bias learning the likelihood asymptotically

Sequential Neural Likelihood(SNL)

- the proposal $\tilde{p}(\theta)$ controls where $q_\varphi(x \mid \theta)$ will be most accurate.
- In a parameter region where $\tilde{p}(\theta)$ is high, there will be a high concentration of training data, hence $p(x \mid \theta)$ will be approximated better.
- Final goal is estimating the posterior $p(\theta \mid X_0)$,use proposal that is high in regions of high posterior density

Sequential Neural Likelihood(SNL)

- Train q_φ multiple rounds, similar with SNL, but train on all simulations obtained up to each round

- More training data in each round, $q_\varphi(x \mid \theta)$ becomes a more accurate model=> $\hat{p}_r(\theta \mid X_0)$ gets closer to the exact posterior

Algorithm 1: Sequential Neural Likelihood (SNL)

Input : observed data $\mathbf{x}_o$, estimator $q_{\boldsymbol{\phi}}(\mathbf{x} \mid \boldsymbol{\theta})$, number of rounds R, simulations per round N

Output : approximate posterior $\hat{p}(\boldsymbol{\theta} \mid \mathbf{x}_o)$

set $\hat{p}_0(\boldsymbol{\theta} \mid \mathbf{x}_o) = p(\boldsymbol{\theta})$ and $\mathcal{D} = \{\}$
for $r = 1 : R$ **do**
  for $n = 1 : N$ **do**
    sample $\boldsymbol{\theta}_n \sim \hat{p}_{r-1}(\boldsymbol{\theta} \mid \mathbf{x}_o)$ with MCMC
    simulate $\mathbf{x}_n \sim p(\mathbf{x} \mid \boldsymbol{\theta}_n)$
    add $(\boldsymbol{\theta}_n, \mathbf{x}_n)$ into $\mathcal{D}$
  (re-)train $q_{\boldsymbol{\phi}}(\mathbf{x} \mid \boldsymbol{\theta})$ on $\mathcal{D}$ and set
  $\hat{p}_r(\boldsymbol{\theta} \mid \mathbf{x}_o) \propto q_{\boldsymbol{\phi}}(\mathbf{x}_o \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})$
return $\hat{p}_R(\boldsymbol{\theta} \mid \mathbf{x}_o)$

Sequential Neural Likelihood(SNL)

- Choice of the neural density estimator $q_\varphi(X|\ \theta)$:conditional Masked Autoregressive Flow
- Perform well in a variety of general-purpose density estimation tasks
- MAF: transformation of a standard Gaussian density N (0, I) through a series of K autoregressive functions f1, . . . , fK each of which depends on θ

$$\mathbf{x} = \mathbf{z_K} \quad \text{where} \quad \begin{aligned} &\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ &\mathbf{z}_k = f_k(\mathbf{z}_{k-1}, \boldsymbol{\theta}). \end{aligned}$$

Conditional Masked Autoregressive Flow

$$\mathbf{x} = \mathbf{z_K} \quad \text{where} \quad \begin{array}{l} \mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ \mathbf{z}_k = f_k(\mathbf{z}_{k-1}, \boldsymbol{\theta}). \end{array}$$

- Each f_k is a bijection with a lower-triangular Jacobian matrix, and is implemented by a Masked Autoencoder for Distribution Estimation conditioned on θ
- By change of variables, the conditional density is given by

$$q_{\boldsymbol{\phi}}(\mathbf{x} \,|\, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{z}_0 \,|\, \mathbf{0}, \mathbf{I}) \prod_k \left| \det\left(\frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right) \right|^{-1}.$$

Experiments

- Neural Likelihood (NL)

(SNL without simulation guiding)

- SNPE-A
- SNPE-B
- Synthetic Likelihood (SL)
- Sequential Monte Carlo ABC (SMC-ABC)

Results

- toy model with complex posterior(fast)
- Lotka–Volterra model from ecology(slow)

A toy model with complex posterior

1.θ is 5-dimensional

2.x is a set of four 2-dimensional points (or an 8-dimensional vector) sampled from a Gaussian

3.mean m_θ and covariance matrix S_θ are functions of θ:

$$\theta_i \sim \mathcal{U}(-3,3) \quad \text{for} \quad i = 1,\ldots,5 \tag{3}$$

$$\mathbf{m}_{\boldsymbol{\theta}} = (\theta_1, \theta_2) \tag{4}$$

$$s_1 = \theta_3^2, \quad s_2 = \theta_4^2, \quad \rho = \tanh(\theta_5) \tag{5}$$

$$\mathbf{S}_{\boldsymbol{\theta}} = \begin{pmatrix} s_1^2 & \rho s_1 s_2 \\ \rho s_1 s_2 & s_2^2 \end{pmatrix} \tag{6}$$

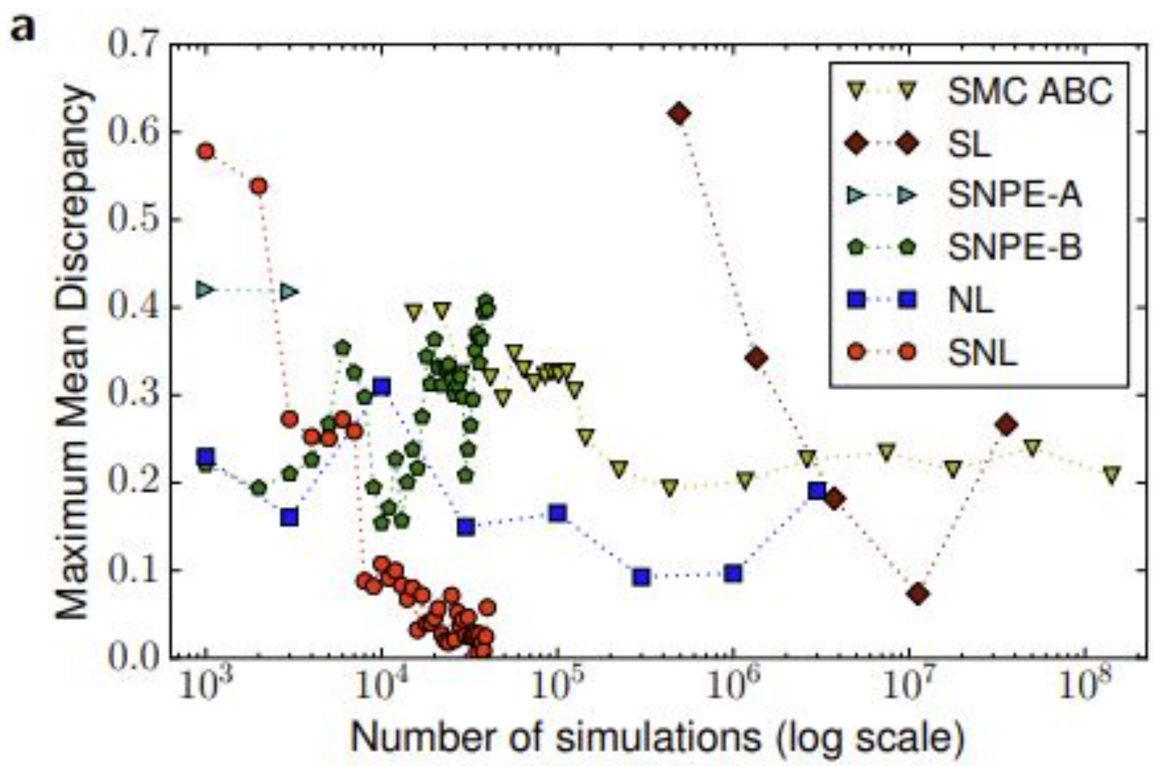
$$\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_4) \quad \text{where} \quad \mathbf{x}_j \sim \mathcal{N}(\mathbf{m}_{\boldsymbol{\theta}}, \mathbf{S}_{\boldsymbol{\theta}}). \tag{7}$$

A toy model with complex posterior

- Y:Maximum Mean Discrepancy between the approximate posterior of each method and the true posterior

- X:Total number of simulations used

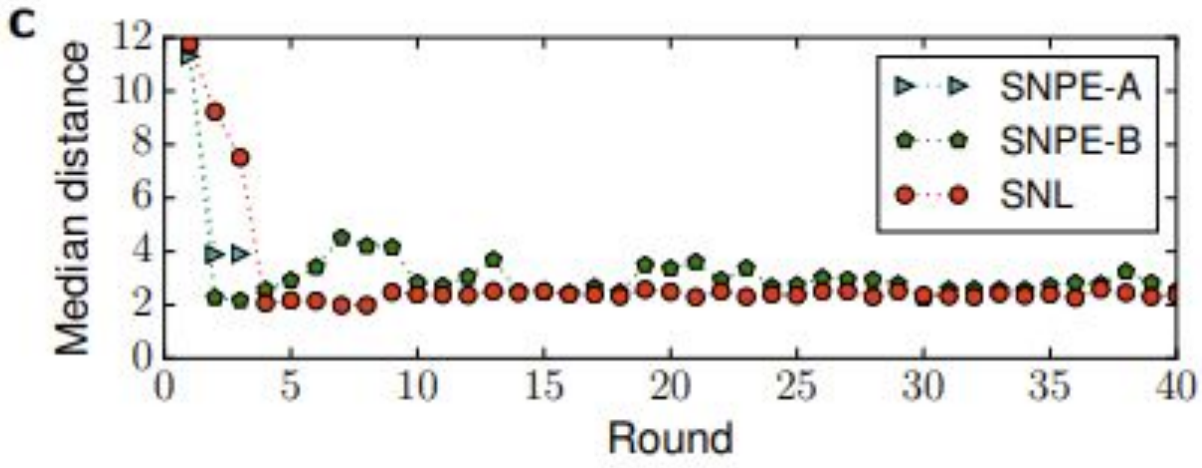
- Left corner best trade-off between accuracy and simulation cost



Median distance between simulated and observed data for each round

- this plot we can assess convergence, and determine the minimum number of rounds to run for
- SNL has lower median distance compared to SNPE-B

- evidence that SNPE-B has not estimated the posterior accurately enough (as also shown in the left plot).



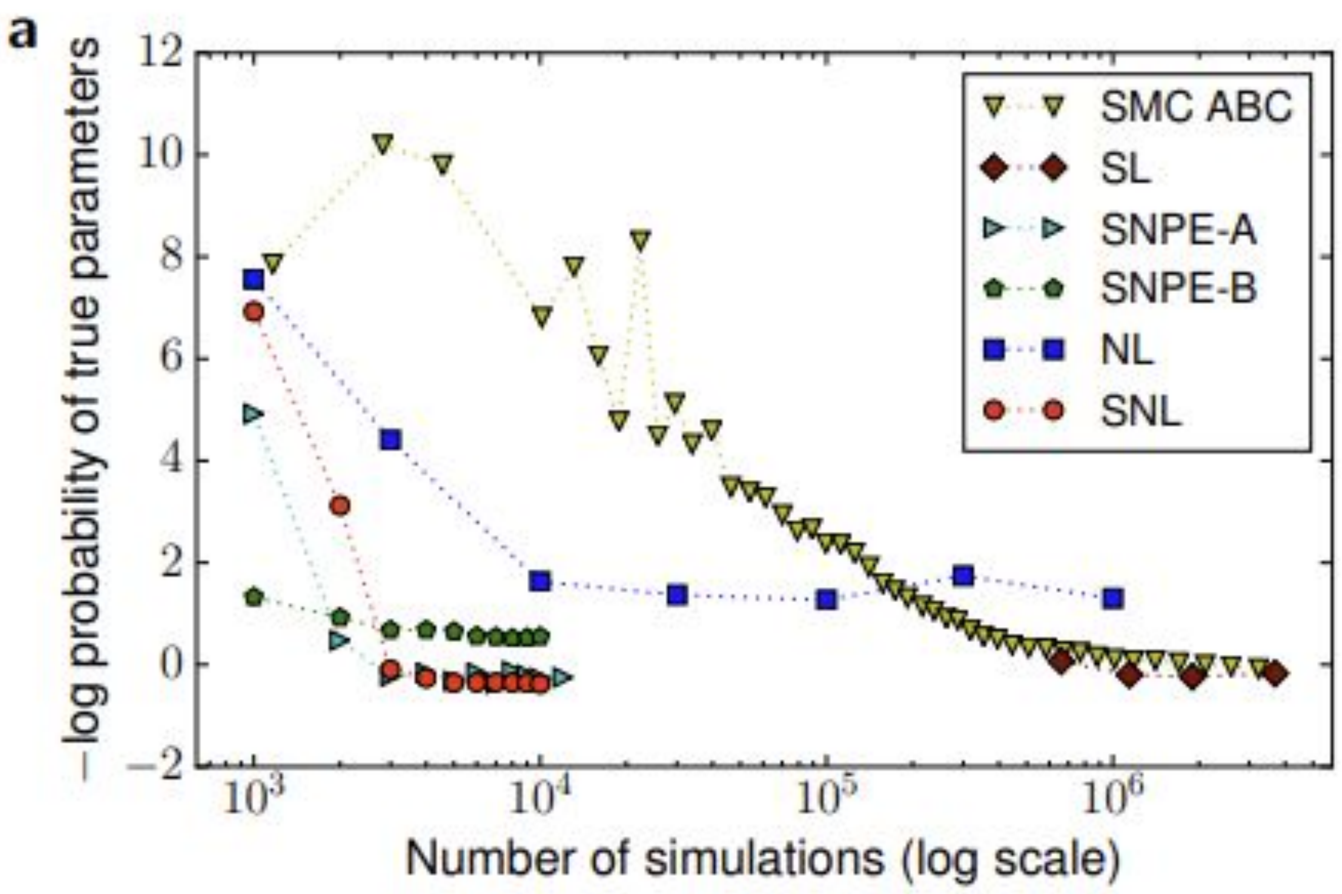
Lotka–Volterra population model

- Markov jump process that models the interaction of a population of predators with a population of prey

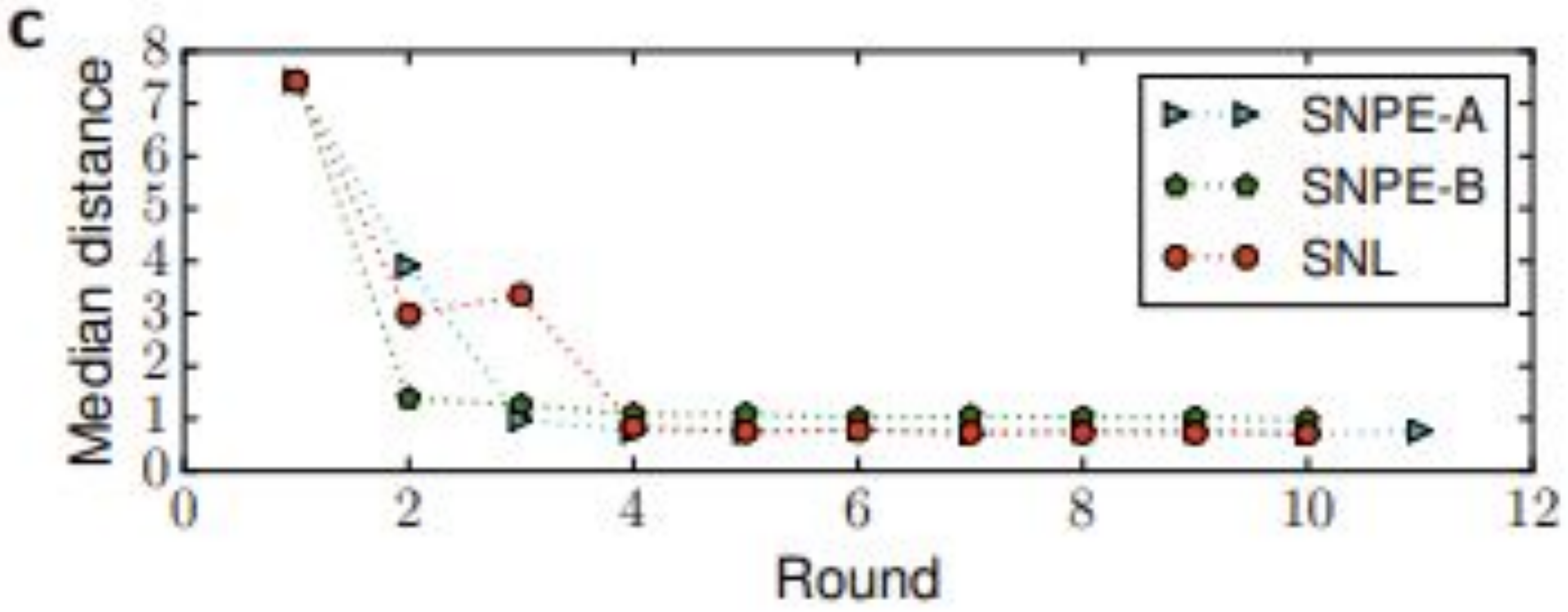
- Four parameters θ, which control the rate of (a) predator births, (b) predator deaths, (c) prey births, and (d) predator-prey interactions

Lotka–Volterra population model

- SNL and SNPE-A perform the best
- SNPE-B is less accurate



Lotka–Volterra population model



- SNPE-A and SNL have a lower median distance
- SNPE-B has not estimated the posterior accurately enough

Discussion

- Performance and robustness of SNL

- Scaling to high-dimensional data

A potential strategy for scaling SNL up to high dimensions is exploiting the structure of the data

- Learning the likelihood vs the posterior

learning the likelihood can often be easier than learning the posterior, and it does not depend on the choice of proposal

a model of the likelihood can be reused with different priors, and is in itself an object of interest that can be used for identifiability analysis [40] or hypothesis testing