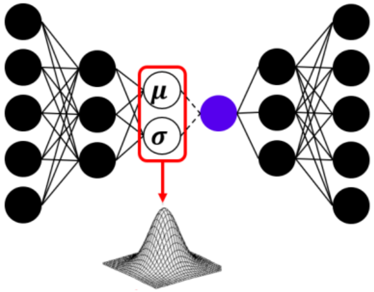
Probabilistic Models for Sequential Data

杨帆

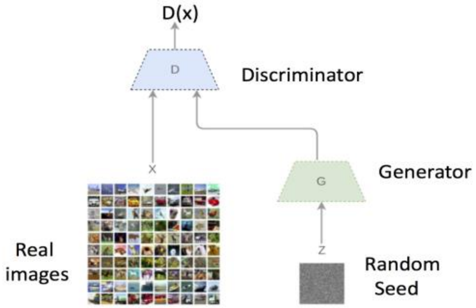
2018年7月19日

Motivations For This Talk (1)

- Some trends in machine learning: make ML to be
  - Robust to uncertain and adversarial inputs
  - Unsupervised, semi-supervised or self-supervised
- Key tools: ***probabilistic generative models***
  - Explicitly model the uncertainty in data
  - Recent advances make them scalable to large datasets and complex data distributions
    - E.g., two popular frameworks: VAE and GAN



Variational Autoencoders



Generative Adversarial Networks

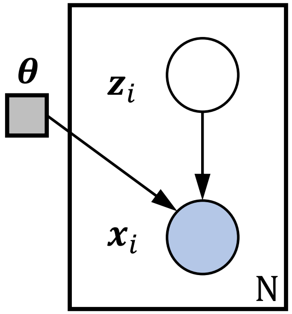
Motivations For This Talk (2)

- Last time we introduced the **Variational Autoencoders**
  - They assume a deep latent gaussian generative model

$$\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbb{I}), \qquad i = 1, \dots, N,$$

$$\boldsymbol{x}_i | \boldsymbol{z}_i \sim \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{z}_i), {\boldsymbol{\sigma}_{\boldsymbol{\theta}}}^2(\boldsymbol{z}_i)\right), \quad i = 1, \dots, N.$$

where $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\cdot)$ and $\boldsymbol{\sigma}_{\boldsymbol{\theta}}(\cdot)$ are neural networks



- For sequential data, we may simply plug CNN or RNN and play
  - However, deep latent gaussian models are not so natural for ***sequential data***
- Today we will introduce several probabilistic models specially designed for sequential data
  - Enable us do prediction and inference in a principled way

Outline

- Short (Re)Introduction to VI and VAE
- State Space Models and Stochastic RNNs
- Variational Inference of SSM
- (Variational) Sequential Monte Carlo
- Conclusions

Outline

- **Short (Re)Introduction to VI and VAE**
- State Space Models and Stochastic RNNs
- Variational Inference of SSM
- (Variational) Sequential Monte Carlo
- Conclusions

Some Clarifications

- **M**aximum **L**ikelihood **E**stimation (**MLE**)
  - θ are treated as unknown static parameters

$$\theta_{MLE} = \operatorname*{argmax}_{\theta} \, p_{\theta}(\mathcal{D})$$

- **M**aximum **A P**osterior estimation (**MAP**)
  - θ are treated as latent variables and we have the prior $p(\theta)$

$$\theta_{MAP} = \operatorname*{argmax}_{\theta} \, p(\theta|\mathcal{D})$$

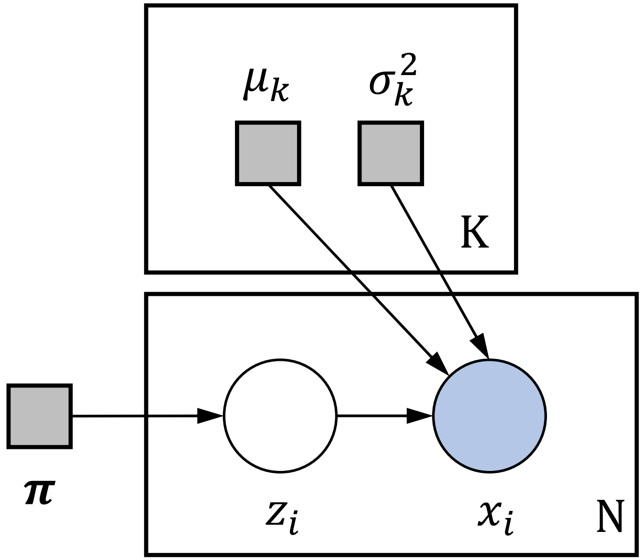
where

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$$

- Nonparametric Bayes: Gaussian process, Dirichlet process, …
  - a class of models in which the number of parameters grows with sample size

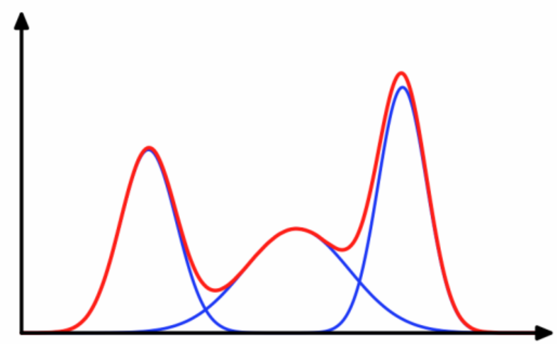
Probabilistic Generative Models

- Example: Gaussian Mixture Models



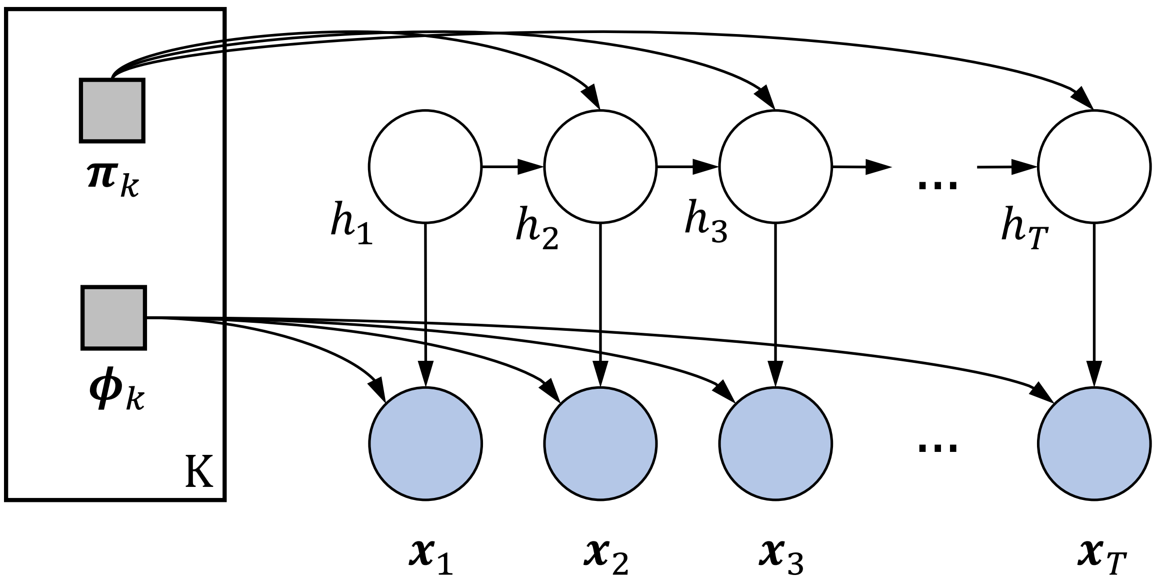
$$z_i \sim \text{categorical}(\pi_1, \dots, \pi_K), \quad i = 1, \dots, N,$$

$$x_i | z_i \sim \mathcal{N}\left(\mu_{z_i}, \sigma_{z_i}^2\right) \qquad i = 1, \dots, N.$$



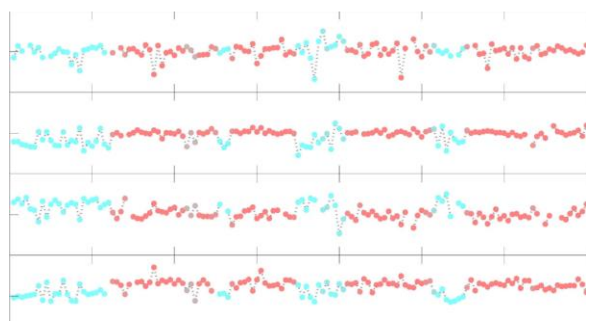
Probabilistic Generative Models

- Example: Hidden Markov Models



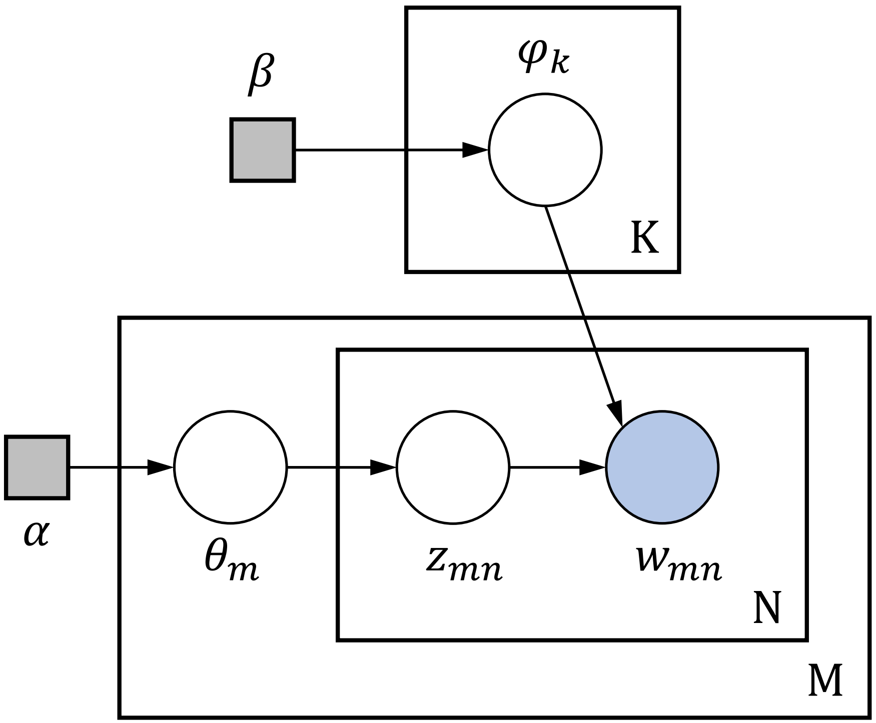
$$h_t \sim \text{categorical}\left(\boldsymbol{\pi}_{h_{t-1}}\right)$$

$$\boldsymbol{x}_t | h_t \sim F\left(\boldsymbol{\phi}_{h_t}\right)$$



Probabilistic Generative Models

- Example: Latent Dirichlet Allocation (fully Bayesian)

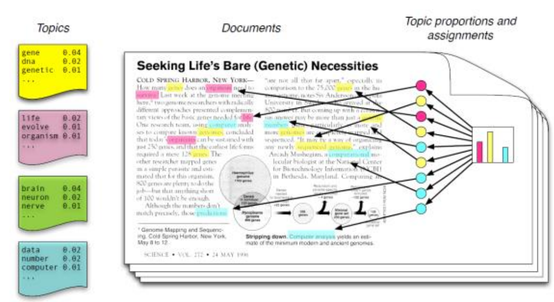


$$\varphi_k \sim Dir(\beta), \quad k = 1, \ldots, K,$$

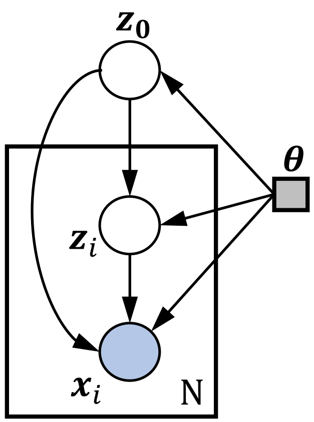
$$\theta_m \sim Dir(\alpha), \quad m = 1, \ldots, \mathrm{M},$$

$$z_{mn} \big| \theta_m \sim \text{categorical}(\theta_m),$$

$$w_{mn} \big| \boldsymbol{\varphi}, z_{mn} \sim \text{categorical}\left(\varphi_{z_{mn}}\right)$$



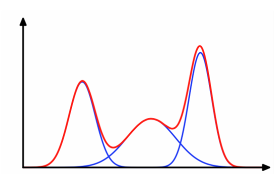
Basic Tasks in Probabilistic Generative Models

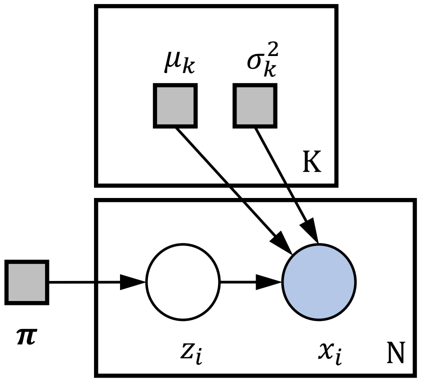


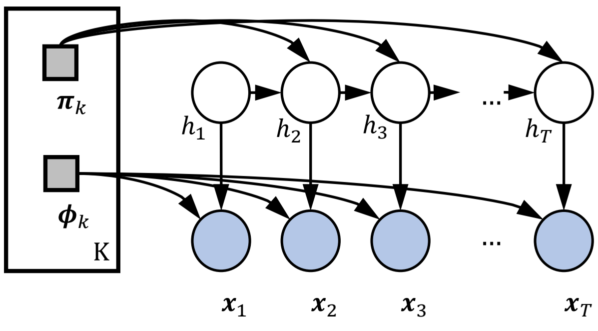
- **Parameter Learning**: fit the model to the dataset
  - **M**aximum **L**ikelihood **E**stimation for the parameters $\boldsymbol{\theta}$

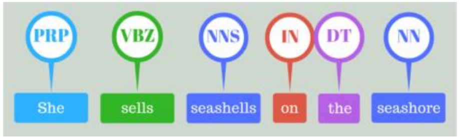
$$\boldsymbol{\theta}_{MLE} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ p_{\boldsymbol{\theta}}(\boldsymbol{x}) \qquad p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{x})}$$

- **Inference**: compute unknown probability distributions
  - Marginal distribution of observations $\boldsymbol{x}$
  - Posterior distribution of latent variable $\boldsymbol{z}$





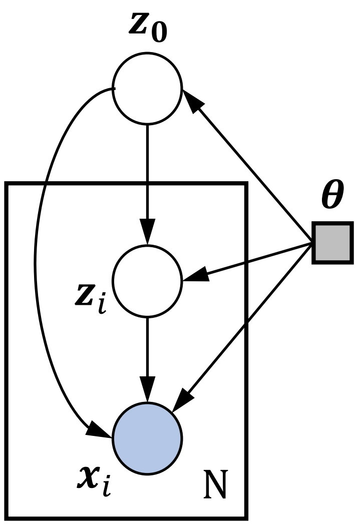




The Difficulty of Learning and Inference

- Posterior distribution of latent variables $\boldsymbol{z} = \{\boldsymbol{z}_i\},\ i = 0 \ldots \mathrm{N}$
- Marginal distribution of observations $\boldsymbol{x} = \{\boldsymbol{x}_i\},\ i = 1 \ldots \mathrm{N}$
- MLE for model parameters $\boldsymbol{\theta}$: $\boldsymbol{\theta}_{MLE} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ p_{\boldsymbol{\theta}}(\boldsymbol{x})$

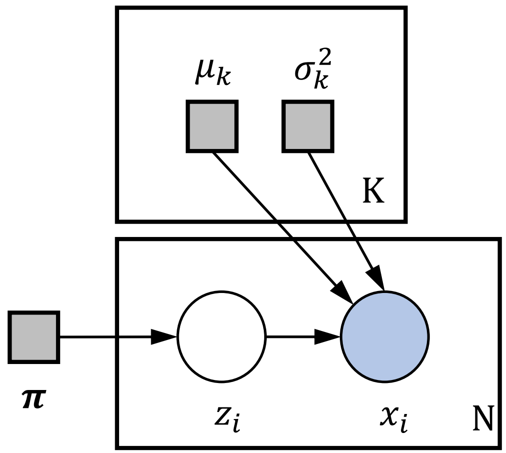
$$p_\theta(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{p_\theta(\boldsymbol{x})} = \frac{\overbrace{p_\theta(\boldsymbol{x}|\boldsymbol{z})}^{\text{likelihood}}\overbrace{p_\theta(\boldsymbol{z})}^{\text{prior}}}{\underbrace{\int p_\theta(\boldsymbol{x},\boldsymbol{z})d\boldsymbol{z}}_{\text{marginal/evidence}}}$$



For many models, the integral (or sum) is **intractable**:

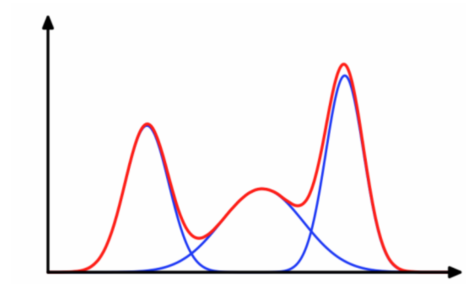
- Unavailable in closed form, or
- Requires exponential time to compute.

A Not-So-Good Example: Univariate GMM



$$z_i \sim \text{categorical}(\pi_1, \dots, \pi_K), \quad i = 1, \dots, N,$$

$$x_i | z_i \sim \mathcal{N}\left(\mu_{z_i}, \sigma_{z_i}^2\right) \qquad i = 1, \dots, N.$$



$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = \prod_{i=1}^{N} p_{\boldsymbol{\theta}}(x_i, z_i), \qquad \text{where} \quad \boldsymbol{\theta} = \{\pi_k, \mu_k, \sigma_k^2 \colon k = 1 \dots K\}$$

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = \prod_{i=1}^{N} \sum_{z_i=1}^{K} p_{\boldsymbol{\theta}}(z_i) p_{\boldsymbol{\theta}}(x_i | z_i)$$

$$= \prod_{i=1}^{N} \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(x;\ \mu_k, \sigma_k^2)$$

$$O(K^N)$$

Variational Inference

- Task: computing the posterior of latent variables $\boldsymbol{z}$: $p(\boldsymbol{z}|\boldsymbol{x})$
- **V**ariational **I**nference turns inference into an ***optimization*** problem
  - Set up a ***family*** of approximate densities $\mathbb{Q} = \{q_\phi\}$ over the latent variables $\boldsymbol{z}$
  - Find the member $q^*(\boldsymbol{z}|\boldsymbol{x}) \in \mathbb{Q}$ that is closest to the exact posterior

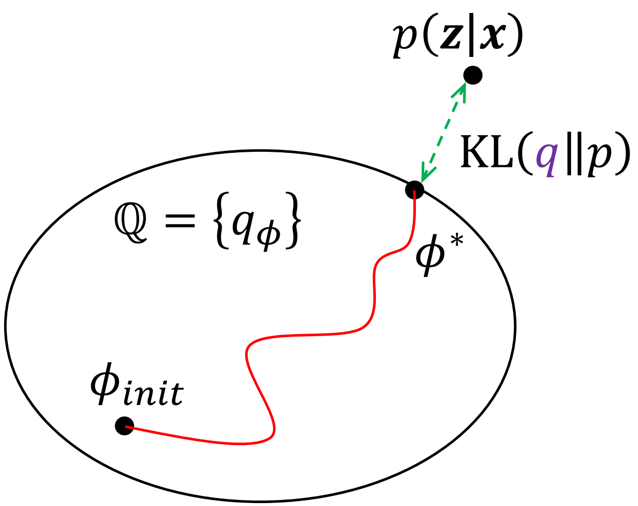
- Specifically, given a family of distributions $\mathbb{Q} = \{q_\phi\}$, try to find:

$$q^*(\boldsymbol{z}|\boldsymbol{x}) = \underset{q_\phi \in \mathbb{Q}}{\arg\min} \, \mathrm{KL}\big(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \| p(\boldsymbol{z}|\boldsymbol{x})\big)$$

equivalent to find:

$$\phi^* = \underset{\phi}{\arg\min} \, \mathrm{KL}\big(q_\phi(\boldsymbol{z}) \| p(\boldsymbol{z}|\boldsymbol{x})\big)$$

$$= \underset{\boldsymbol{\phi}}{\mathbf{arg\,max}} \, \mathbf{ELBO}\big(\boldsymbol{q_\phi}\big)$$



Maximizing the ELBO

- Variational Inference: maximizing the <u>E</u>vidence <u>L</u>ower <u>Bo</u>und
  - Maximizing the ELBO is equivalent to minimizing the KL divergence

$$\log p(\boldsymbol{x}) = \downarrow \mathrm{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \| p(\boldsymbol{z}|\boldsymbol{x})\right) + \ \mathbf{ELBO}\left(q_\phi\right) \uparrow$$

$$\geq \mathbf{ELBO}\left(q_\phi\right) = \mathbb{E}_{q_\phi}\left[\log \frac{p(\boldsymbol{z},\boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]$$

$$\int q_\phi(\boldsymbol{z}|\boldsymbol{x})(\dots)d\boldsymbol{z}$$

- We want to optimize the ELBO using stochastic gradient ascent
- The key is how to take the gradient ∇_ϕ of the **intractable** expectation
  - The general REINFORCE gradient provides an estimator (with high variance)

$$\nabla \mathbb{E}_p[f(\dots)] = \mathbb{E}_p[\nabla \log p(x)\, f(\dots)] \approx \sum_{i=1}^{M} \nabla \log p(x_i)\, f(\dots), \qquad x_i \sim p(x)$$

Another Usage of the ELBO

- Another task is finding the MLE for $\boldsymbol{\theta}$: $\boldsymbol{\theta}_{MLE} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ p_{\boldsymbol{\theta}}(\boldsymbol{x})$
- Recall that $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is generally **intractable** for complex models
- Instead, we can maximize the ELBO, which is a lower bound of $p_{\boldsymbol{\theta}}(\boldsymbol{x})$

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathrm{KL}\big(q_{\phi}(\boldsymbol{z}|\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})\big) + \mathbf{ELBO}\big(q_{\phi}\big)$$

$$\geq \mathbf{ELBO}\big(q_{\phi}\big) = \mathbb{E}_{q_{\phi}}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}, \boldsymbol{x})}{q_{\phi}(\boldsymbol{z}|\boldsymbol{x})}\right]$$

- So the ELBO provides an optimization target for both $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$
  - Variational parameters ϕ and model/generative parameters θ

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{q_{\phi}}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{z}, \boldsymbol{x}) - \log q_{\phi}(\boldsymbol{z}|\boldsymbol{x})\right]$$

  - We want to take the gradients $\nabla_{\boldsymbol{\phi},\boldsymbol{\theta}}\mathcal{L}$ and optimize these parameters

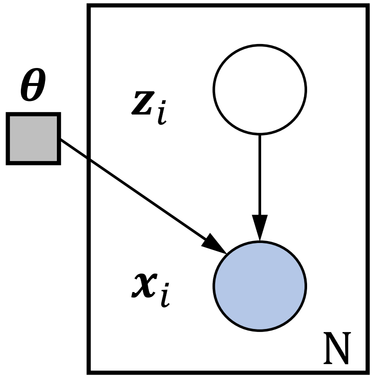
Variational Autoencoder: Model and ELBO

- VAE assumes a **D**eep **L**atent **G**aussian **M**odel

$$\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbb{I}), \qquad i = 1, \dots, N,$$

$$\boldsymbol{x}_i | \boldsymbol{z}_i \sim \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{z}_i), {\boldsymbol{\sigma}_{\boldsymbol{\theta}}}^2(\boldsymbol{z}_i)\right), \quad i = 1, \dots, N.$$

where $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\cdot)$ and $\boldsymbol{\sigma}_{\boldsymbol{\theta}}(\cdot)$ are neural networks



- Here the global ELBO can be factorized into many local ELBOs
  - One ELBO for each individual data point

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \sum_{i=1}^{N} \mathbb{E}_{q_{\phi_i}}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}_i, \boldsymbol{x}_i)}{q_{\boldsymbol{\phi}_i}(\boldsymbol{z}_i | \boldsymbol{x}_i)}\right] = \sum_{i=1}^{N} \mathcal{L}_i(\boldsymbol{\phi}_i, \boldsymbol{\theta})$$

where $\boldsymbol{\phi} = \{\boldsymbol{\phi}_i : i = 1 \dots N\}$

VAE: Optimization of the ELBO (1)

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \sum_{i=1}^{N} \mathcal{L}_i(\boldsymbol{\phi}_i, \boldsymbol{\theta}) = \sum_{i=1}^{N} \mathbb{E}_{q_{\boldsymbol{\phi}_i}}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}_i, \boldsymbol{x}_i)}{q_{\boldsymbol{\phi}_i}(\boldsymbol{z}_i|\boldsymbol{x}_i)}\right]$$

$$\nabla_{\boldsymbol{\phi}_i}\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \nabla_{\boldsymbol{\phi}_i}\mathcal{L}_i(\boldsymbol{\phi}_i, \boldsymbol{\theta}), \qquad i = 1 \dots N$$

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) \approx \frac{N}{S}\sum_{s=1}^{S} \nabla_{\boldsymbol{\theta}}\mathcal{L}_{i_s}(\boldsymbol{\phi}_{i_s}, \boldsymbol{\theta})$$

- They introduced a low-variance gradient estimator for the ELBO
  - Called *reparameterization trick*, or *path-wise gradient estimator*

$$\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}_i}\mathbb{E}_{q_{\boldsymbol{\phi}_i}}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}_i, \boldsymbol{x}_i)}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)}\right] = \mathbb{E}_{\mathcal{N}(\boldsymbol{\epsilon})}\left[\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}_i}\log \frac{p_{\boldsymbol{\theta}}(t(\boldsymbol{\epsilon}, \boldsymbol{\phi}_i), \boldsymbol{x}_i)}{q_{\boldsymbol{\phi}_i}(t(\boldsymbol{\epsilon}, \boldsymbol{\phi}_i)|\boldsymbol{x}_i)}\right]$$

$$\approx \frac{1}{M}\sum_{k=1}^{M} \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}_i}\log \frac{p_{\boldsymbol{\theta}}(t(\boldsymbol{\epsilon}_k, \boldsymbol{\phi}_i), \boldsymbol{x}_i)}{q_{\boldsymbol{\phi}_i}(t(\boldsymbol{\epsilon}_k, \boldsymbol{\phi}_i)|\boldsymbol{x}_i)}, \qquad \boldsymbol{\epsilon}_k \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{1})$$

Assume:

$$\boldsymbol{z}_i \sim q_{\boldsymbol{\phi}_i}(\boldsymbol{z}_i|\boldsymbol{x}_i)$$

$$\Leftrightarrow$$

$$\boldsymbol{z}_i = t(\boldsymbol{\epsilon}, \boldsymbol{\phi}_i, \boldsymbol{x}_i),$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, \mathbf{1})$$

VAE: Optimization of the ELBO (2)

$$\mathcal{L}(\boldsymbol{\phi},\boldsymbol{\theta}) = \sum_{i=1}^{N} \mathcal{L}_i(\boldsymbol{\phi}_i,\boldsymbol{\theta}) = \sum_{i=1}^{N} \mathbb{E}_{q_{\boldsymbol{\phi}_i}}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}_i,\boldsymbol{x}_i)}{q_{\boldsymbol{\phi}_i}(\boldsymbol{z}_i|\boldsymbol{x}_i)}\right]$$

$$\nabla_{\boldsymbol{\phi}_i}\mathcal{L}(\boldsymbol{\phi},\boldsymbol{\theta}) = \nabla_{\boldsymbol{\phi}_i}\mathcal{L}_i(\boldsymbol{\phi}_i,\boldsymbol{\theta}), \qquad i = 1 \dots N$$

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\phi},\boldsymbol{\theta}) \approx \frac{N}{S}\sum_{s=1}^{S} \nabla_{\boldsymbol{\theta}}\mathcal{L}_{i_s}(\boldsymbol{\phi}_{i_s},\boldsymbol{\theta})$$

- There are N local variational parameters $\boldsymbol{\phi}_i$ to be optimized
  - For large datasets N is large, this is too expensive
- VAE uses a shared ***inference network*** to predict $\boldsymbol{\phi}_i$ from $\boldsymbol{x}_i$ instead
  - $g(\boldsymbol{x}_i) = \boldsymbol{\phi}_i$
  - Also called *recognition network*
  - this technique is known as **A**mortized **V**ariational **I**nference

VAE: Optimization of the ELBO (3)

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \sum_{i=1}^{N} \mathcal{L}_i\left(g_{\boldsymbol{\phi}}(\boldsymbol{x}_i), \boldsymbol{\theta}\right) = \sum_{i=1}^{N} \mathbb{E}_{q_{g_{\boldsymbol{\phi}}(\boldsymbol{x}_i)}}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}_i, \boldsymbol{x}_i)}{q_{g_{\boldsymbol{\phi}}(\boldsymbol{x}_i)}(\boldsymbol{z}_i | \boldsymbol{x}_i)}\right]$$

$$\nabla_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) \approx \frac{N}{S} \sum_{s=1}^{S} \nabla_{\boldsymbol{\phi}} \mathcal{L}_{i_s}\left(g_{\boldsymbol{\phi}}(\boldsymbol{x}_{i_s}), \boldsymbol{\theta}\right)$$

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) \approx \frac{N}{S} \sum_{s=1}^{S} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{i_s}\left(g_{\boldsymbol{\phi}}(\boldsymbol{x}_{i_s}), \boldsymbol{\theta}\right)$$

- We treat the parameters of the inference network g as $\boldsymbol{\phi}$
  - Instead of $\boldsymbol{\phi} = \{\boldsymbol{\phi}_i : i = 1 \dots N\}$
- The reparameterization trick gives the gradients $\nabla_{\boldsymbol{\phi}_i}$ for $\boldsymbol{\phi}_i$
- Since $g_{\boldsymbol{\phi}}(\boldsymbol{x}_i) = \boldsymbol{\phi}_i$, $\nabla_{\boldsymbol{\phi}_i}$ can be propagated back into $g_{\boldsymbol{\phi}}$

Contributions of VAE

- Proposed the reparameterization tricks (a.k.a. path-wise estimator)
  - which yield a low-variance gradient estimator for variational inference
- Introduced the inference network (a.k.a. *recognition network*)
  - This scheme is also called *amortized variational inference*
    - In traditional VI, we have to optimize local variational parameters for every datapoint
    - In amortized VI, we optimize shared parameters of the inference network instead

$$\boldsymbol{z}_i \big| \boldsymbol{x}_i \sim q_\phi(\boldsymbol{z}_i | \boldsymbol{x}_i) = \mathcal{N}(\boldsymbol{\mu}_i, {\boldsymbol{\sigma}_i}^2) = \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}_i), {\boldsymbol{\sigma}_{\boldsymbol{\phi}}}^2(\boldsymbol{x}_i)\mathbb{I}\right)$$

where $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\cdot)$ and $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\cdot)$ are neural networks

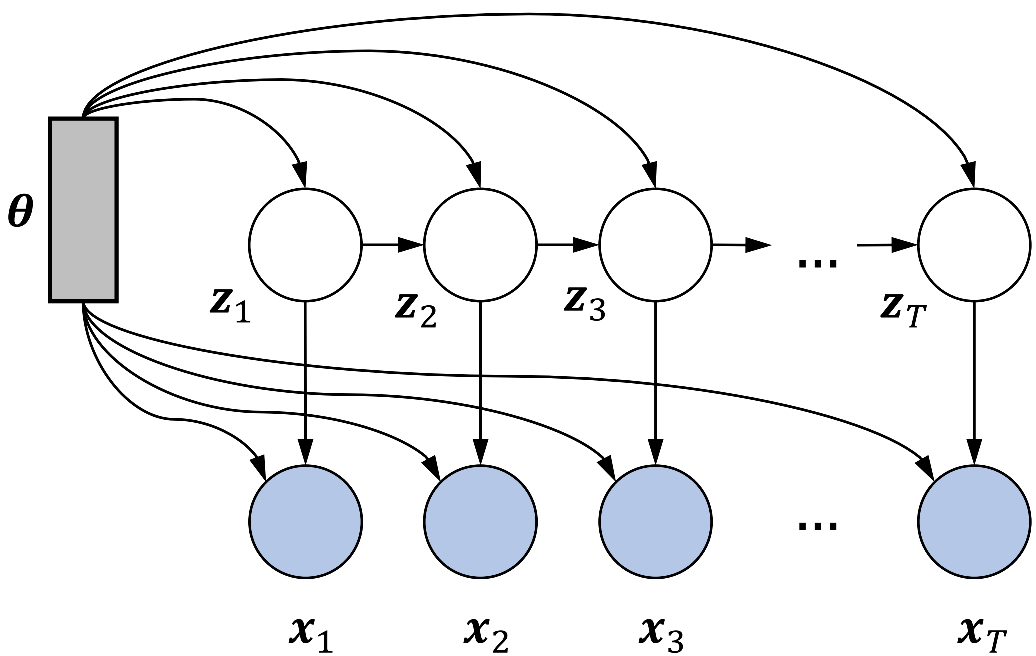
- Co-training of variational parameters and generative parameters

Outline

- Short (Re)Introduction to VI and VAE
- **State Space Models and Stochastic RNNs**
- Variational Inference of SSM
- (Variational) Sequential Monte Carlo
- Conclusions

State Space Models

- **SSM** are a general class of probabilistic sequential models
  - Usually we assume that the Markovian property holds for state transitions
    - Non-Markovian transitions can be convert to Markovian transitions by memorization



$$\boldsymbol{z}_1 \sim \pi_\theta(\cdot)$$

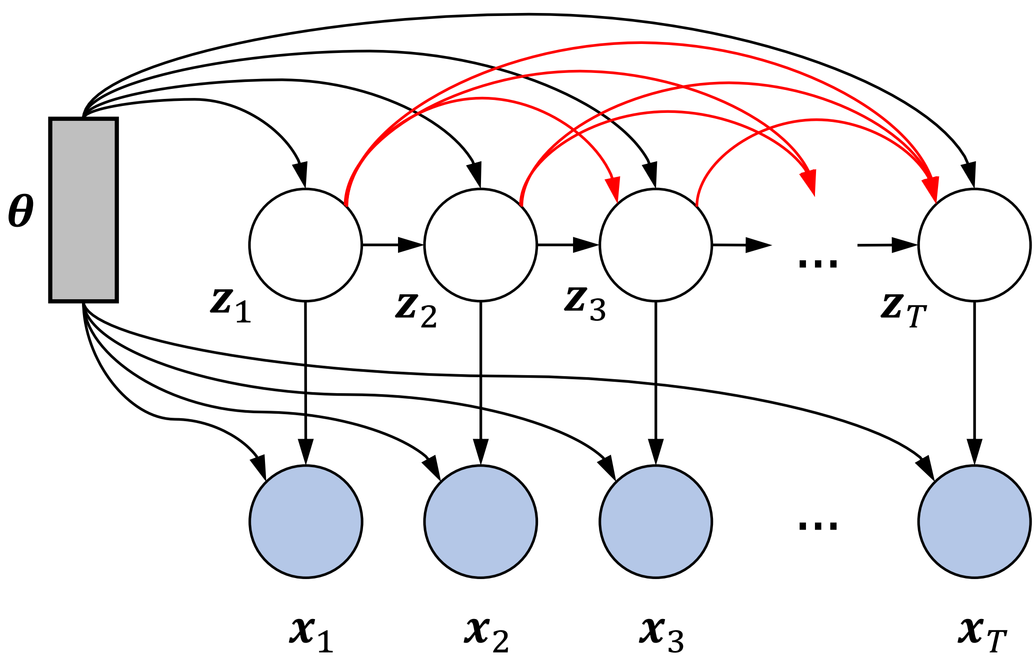
$$\boldsymbol{z}_t | \boldsymbol{z}_{t-1} \sim f_\theta(\boldsymbol{z}_{t-1})$$

$$\boldsymbol{x}_t | \boldsymbol{z}_t \sim g_\theta(\boldsymbol{z}_t)$$

$$\text{where } t = 2 \dots T$$

State Space Models: Non-Markovian Variant

- **SSM** are a general class of probabilistic sequential models
  - Usually we assume that the Markovian property holds for state transitions
    - Non-Markovian transitions can be convert to Markovian transitions by memorization
  - However, explicitly modelling the non-Markovian property can be beneficial



$$\boldsymbol{z}_1 \sim \pi_\theta(\cdot)$$

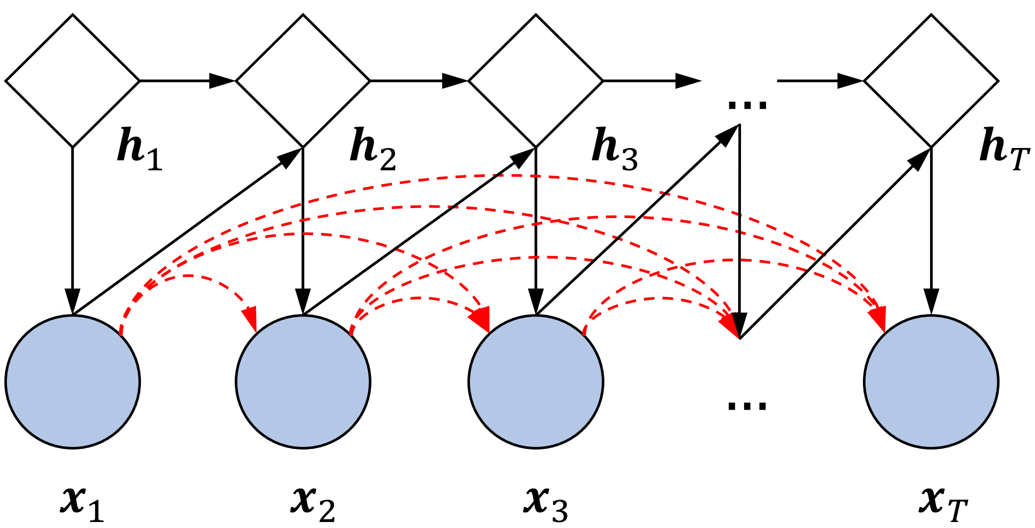
$$\boldsymbol{z}_t \big| \boldsymbol{z}_{1:t-1} \sim f_\theta(\boldsymbol{z}_{1:t-1})$$

$$\boldsymbol{x}_t \big| \boldsymbol{z}_t \sim g_\theta(\boldsymbol{z}_t)$$

where $t = 2 \dots T$

Recurrent Neural Network

- In general, RNN is a deterministic model
- However in language modeling, RNN is exactly a probabilistic model
  - The output at each timestep is a softmax, measured by cross-entropy loss
- A.k.a. ***autoregressive*** models, in which there are no **latent** variables
  - order sensitive, not-so-good at modeling stochastic transitions



$$\boldsymbol{x}_1 \sim g_\theta(\boldsymbol{h}_1)$$

$$\boldsymbol{h}_t = \mathrm{LSTM}_\theta(\boldsymbol{h}_{t-1}, \boldsymbol{x}_{t-1})$$

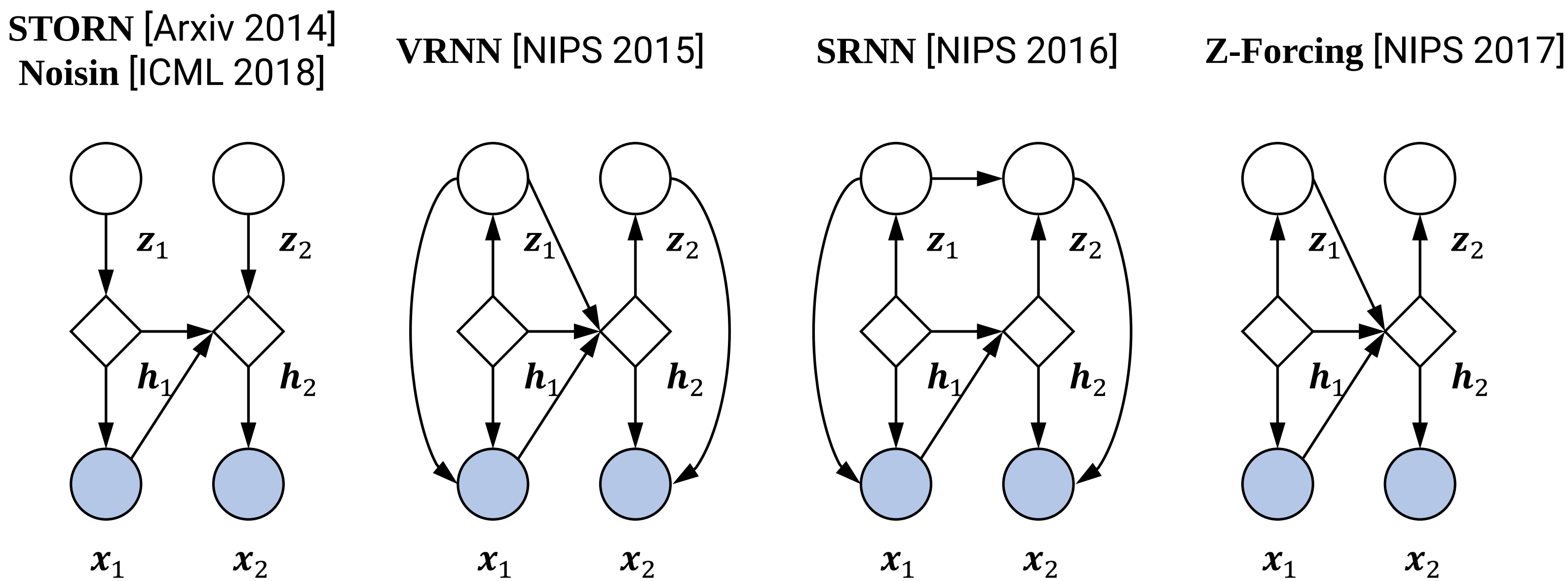
$$\boldsymbol{x}_t | \boldsymbol{x}_{1:t-1} \sim g_\theta(\cdot\, | \boldsymbol{h}_t)$$

$$\text{where } \ t = 2 \dots T$$

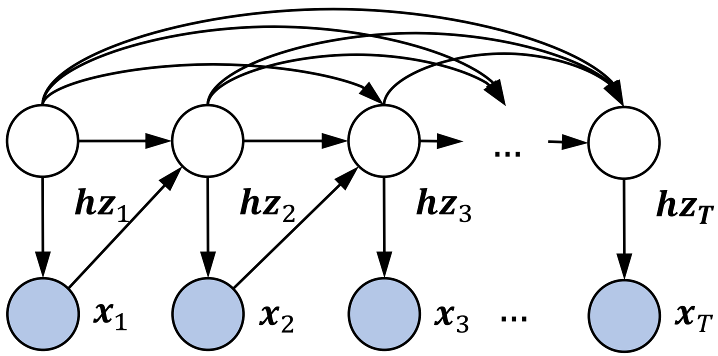
the actual conditional dependences are shown in red dashed lines

Stochastic RNNs

- Some work focused on injecting random variables into RNNs



After merging
the deterministic & stochastic state
at each time step,
almost all of them resemble this:



Two Elegant Models that I Like

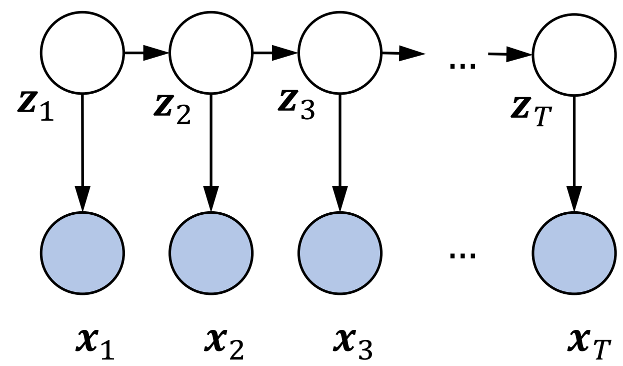
- Deep Markov Model

$$z_1 \sim \pi_\theta(\cdot)$$

$$z_t | z_{t-1} \sim \mathcal{N}\left(\mu_\theta(z_{t-1}), \sigma_\theta^2(z_{t-1})\right)$$

$$x_t | z_t \sim g_\theta(\cdot \mid z_t)$$

where $\mu_\theta(\cdot)$ and $\sigma_\theta^2(\cdot)$ are (gated) neural networks



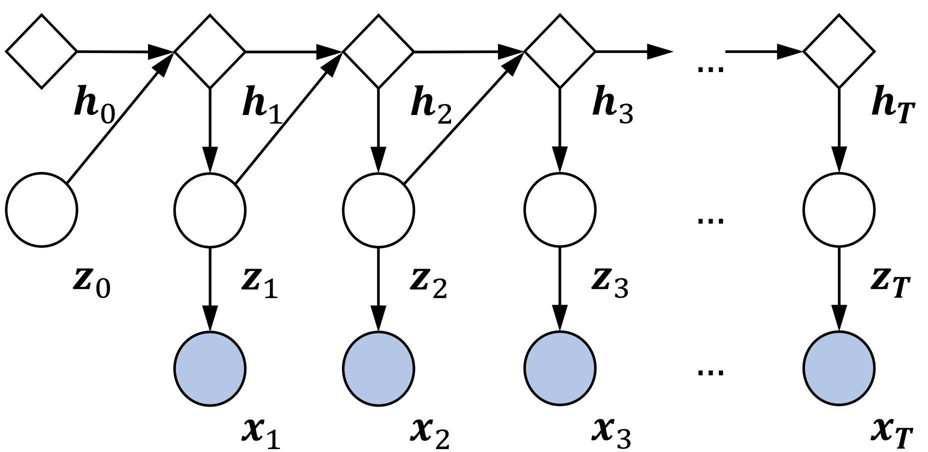
- LSTM State Space Model

$$z_0 \sim \pi_\theta(\cdot)$$

$$h_t = \mathrm{LSTM}_\theta(h_{t-1}, z_{t-1})$$

$$z_t | z_{1:t-1} \sim f_\theta(\cdot \mid h_t)$$

$$x_t | z_t \sim g_\theta(\cdot \mid z_t)$$

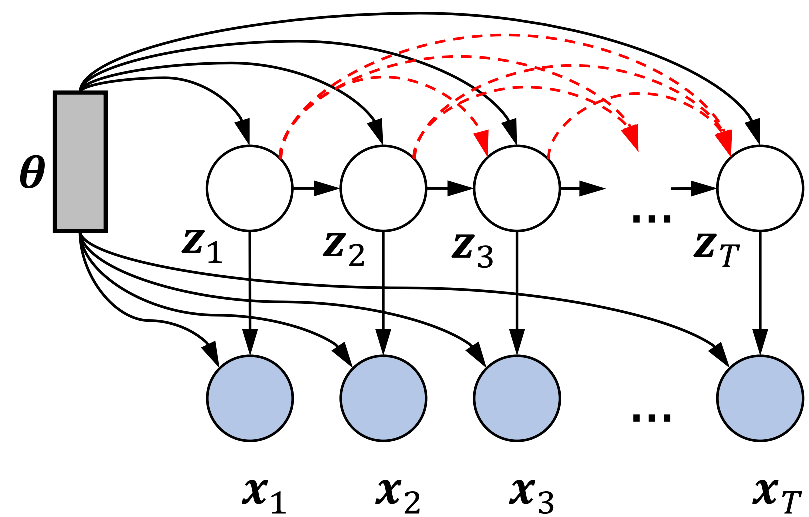


Structured Inference Networks for Nonlinear State Space Models. Krishnan, et al. AAAI 2017
State Space LSTM Models with Particle MCMC Inference. Zheng, et al. Arxiv 2017

Using Probabilistic Sequential Models

- Given these models, our basic tasks are:
  - Model learning: MLE for the model parameters $\boldsymbol{\theta}$: $\boldsymbol{\theta}_{MLE} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ p_{\boldsymbol{\theta}}(\boldsymbol{x})$
  - Bayesian inference: posterior distribution of latent variables $\boldsymbol{z}$: $p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})$
  - Prediction: conditional distribution of future states: $p_{\boldsymbol{\theta}}(\boldsymbol{z}_{T+1:T+L}|\boldsymbol{x}_{1:T})$
    - And future observations: $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{T+1:T+L}|\boldsymbol{x}_{1:T})$

- Next: do learning and inference using
  - Variational Inference
  - (Variational) Sequential Monte Carlo

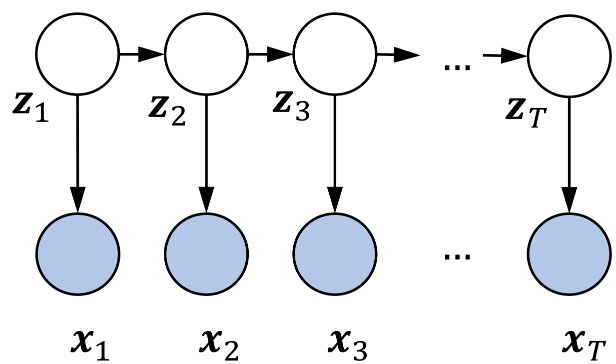


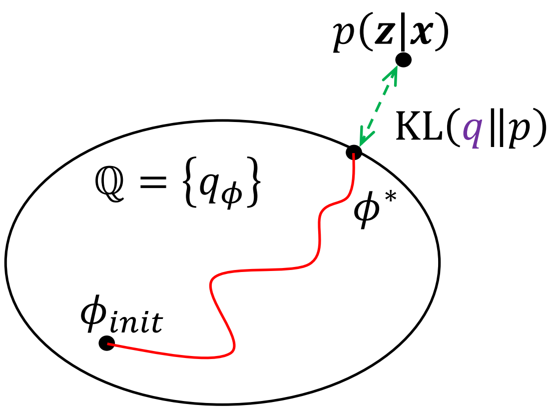
Outline

- Short (Re)Introduction to VI and VAE
- State Space Models and Stochastic RNNs
- **Variational Inference of SSM**
- (Variational) Sequential Monte Carlo
- Conclusions

Variational Inference of SSM

- Recall that in VI, we need to design a variational family $\mathbb{Q} = \{q_\phi(\boldsymbol{z}|\boldsymbol{x})\}$
  - Or equivalently, a family of inference networks in VAE
  - A good variational family should be expressive enough
    - To include the true posterior or some element close enough to the posterior
  - We can look at the factorized form of true posterior and imitate it



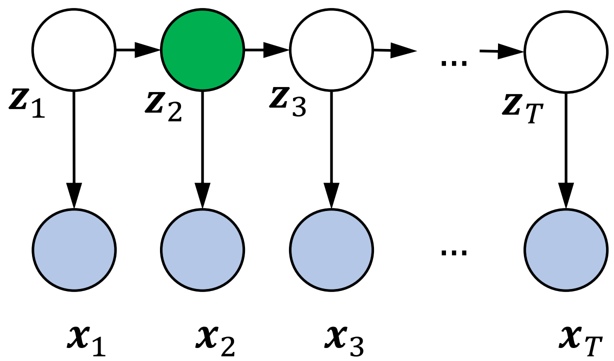


$$p_\theta(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T}) = \pi_\theta(\boldsymbol{z}_1) g_\theta(\boldsymbol{x}_1|\boldsymbol{z}_1) \prod_{t=2}^{T} f_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) g_\theta(\boldsymbol{x}_t|\boldsymbol{z}_t)$$

$$p_\theta(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) = p_\theta(\boldsymbol{z}_1|\boldsymbol{x}_{1:T}) \prod_{t=2}^{T} p_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \boldsymbol{x}_{t:T})$$

True Posterior for SSMs

- Using the standard **d-separation** criterion for Bayesian networks:



$$\boldsymbol{z}_{t+1} \perp \boldsymbol{z}_{1:t-1} \,|\, \boldsymbol{z}_t$$

$$\boldsymbol{z}_t \perp \boldsymbol{x}_{1:t-1} \,|\, \boldsymbol{z}_{t-1}$$

$$p_\theta(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})$$

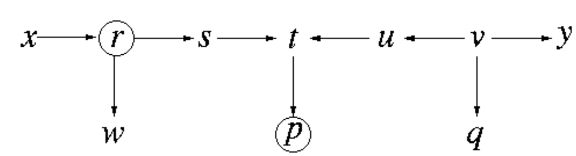
$$= p_\theta(\boldsymbol{z}_1|\boldsymbol{x}_{1:T}) \prod_{t=2}^{T} p_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{x}_{1:T})$$

$$= p_\theta(\boldsymbol{z}_1|\boldsymbol{x}_{1:T}) \prod_{t=2}^{T} p_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \boldsymbol{x}_{1:T})$$

$$= p_\theta(\boldsymbol{z}_1|\boldsymbol{x}_{1:T}) \prod_{t=2}^{T} p_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \boldsymbol{x}_{t:T})$$

information from the **future** is crucial

Rule 1: x and y are d-connected if there is an unblocked path between them.
Rule 2: x and y are d-connected, conditioned on a set Z of nodes, if there is a collider-free path between x and y that traverses no member of Z.
Rule 3: If a collider is a member of the conditioning set Z, or has a descendant in Z, then it no longer blocks any path that traces this collider.



Inference Network for SSM

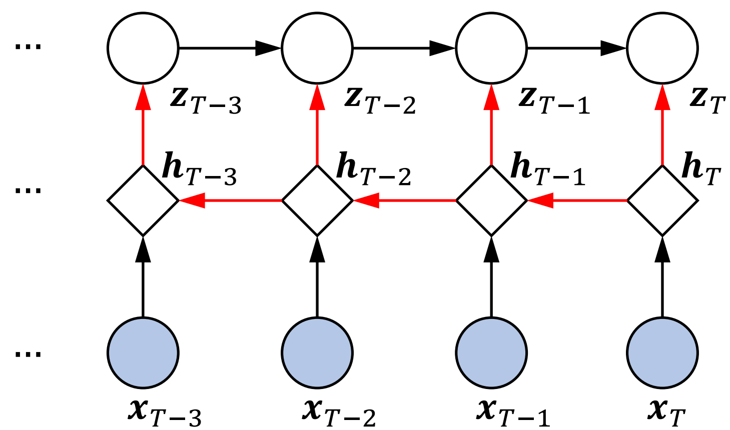
- Let the form of $q_\phi(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})$ be exactly the same as $p_\theta(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})$:

$$p_\theta(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) = p_\theta(\boldsymbol{z}_1|\boldsymbol{x}_{1:T})\prod_{t=2}^{T} p_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{t-1},\boldsymbol{x}_{t:T})$$

$$q_\phi(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) = q_\phi(\boldsymbol{z}_1|\boldsymbol{x}_{1:T})\prod_{t=2}^{T} q_\phi(\boldsymbol{z}_t|\boldsymbol{z}_{t-1},\boldsymbol{x}_{t:T})$$

$$\text{where} \quad q_\phi(\boldsymbol{z}_t|\boldsymbol{z}_{t-1},\boldsymbol{x}_{t:T}) = \mathcal{N}\left(\mu_\phi(\boldsymbol{z}_{t-1},\boldsymbol{x}_{t:T}), \sigma_\phi^2(\boldsymbol{z}_{t-1},\boldsymbol{x}_{t:T})\right)$$

- How to implement $q_\phi(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})$? Well, it looks like a **Backward** **RNN** !
  - $\mu_\phi(\cdot)$ and $\sigma_\phi^2(\cdot)$ use backward RNNs as their backbones



Learning and Inference

- We have constructed the variational family backboned by RNNs
- So we have the ELBO:

$$\mathbf{ELBO}(q_{\phi}) = \mathbb{E}_{q_{\phi}}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}, \boldsymbol{x})}{q_{\phi}(\boldsymbol{z}|\boldsymbol{x})}\right]$$

- Now we can perform stochastic gradient ascent in both $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$
  - Using the gradient estimator provided by reparameterization tricks
  - The ELBO can be factorized to reduce the variance of gradient estimator

- Given the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, we can:
  - Generating new sequences (using generative parameters $\boldsymbol{\theta}$)
  - Performing inference (using variational parameters $\boldsymbol{\phi}$)
  - Predicting the future from current observations (using both $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$)

Structured Inference Networks for Nonlinear State Space Models. Krishnan, et al. AAAI 2017

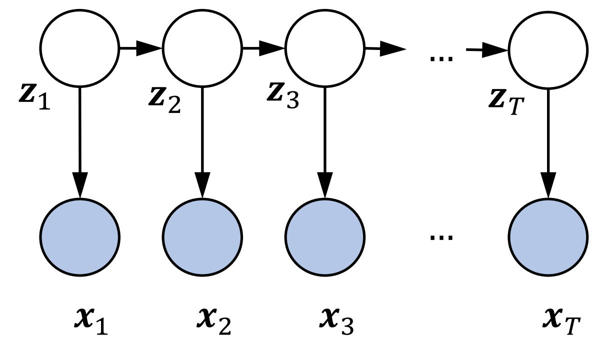
Outline

- Short (Re)Introduction to VI and VAE
- State Space Models and Stochastic RNNs
- Variational Inference of SSM
- **(Variational) Sequential Monte Carlo**
- Conclusions

Sequential Monte Carlo

- **SMC** methods are standard tools for nonlinear dynamical systems
  - a.k.a. **P**article **F**ilters
  - Widely used in many fields, including physics, engineering, robotics, finance, …
- Consider the state space model

$$p_\theta(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T})$$
$$= \mu_\theta(\boldsymbol{z}_1) g_\theta(\boldsymbol{x}_1|\boldsymbol{z}_1) \prod_{t=2}^{T} f_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) g_\theta(\boldsymbol{x}_t|\boldsymbol{z}_t)$$



- PFs are mainly used for solving the ***filtering*** & ***smoothing*** problem
  - Filtering: the posterior distribution of current state: $\{p_{\boldsymbol{\theta}}(\boldsymbol{z}_t|\boldsymbol{x}_{1:t})\}_{t\geq 1}$
  - Smoothing: the posterior distribution of past states: $\{p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:t}|\boldsymbol{x}_{1:t})\}_{t\geq 1}$
    - or $p_{\boldsymbol{\theta}}(\boldsymbol{z}_{m:n}|\boldsymbol{x}_{1:t}), 1 \leq m \leq n < t$
- While SMCs can be used in more general case
  - e.g., non-Markovian SSM

(most literature writes $\boldsymbol{x} \rightarrow \boldsymbol{y}$ instead of $\boldsymbol{z} \rightarrow \boldsymbol{x}$)

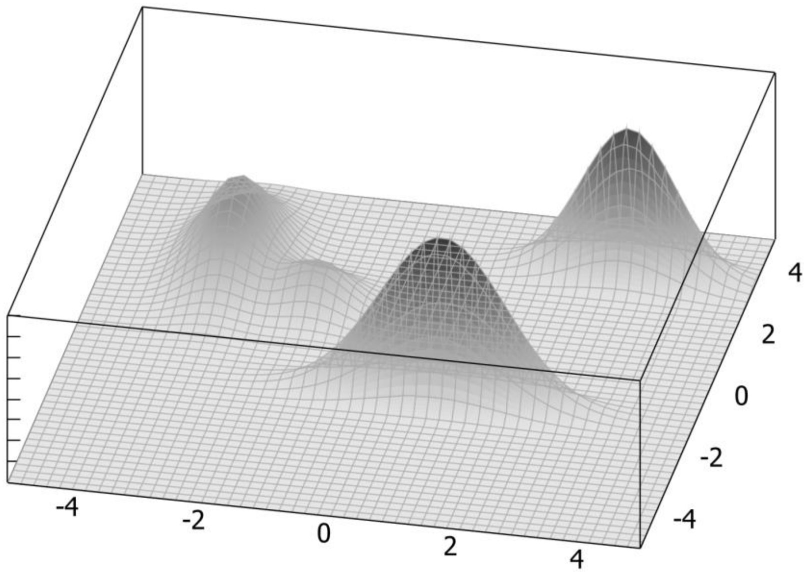
Monte Carlo Basics (1)

- Monte Carlo approximation of the intractable density π

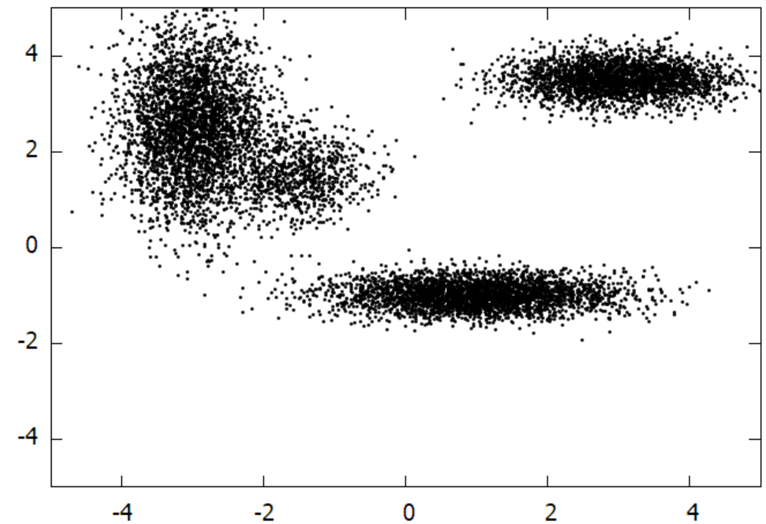
$$\mathbb{E}_{\pi}[f(x)] = \int \pi(x) f(x)\, dx \approx \mathbb{E}_{\hat{\pi}}[f(x)] = \frac{1}{N}\sum_{i=1}^{N} f(x_i)\,, \qquad x_i \sim \pi(x), i = 1 \dots N$$

  - i.e., approximate π by its empirical distribution $\hat{\pi}$: $\hat{\pi}(x) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}(x)$

PDF



Samples

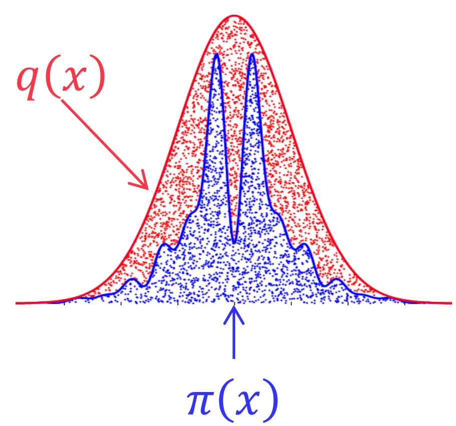


Monte Carlo Basics (2)

- MC assumes that it is possible to generate $x_i \sim \pi(x), i = 1 \dots N$
  - i.e., to generate i.i.d. samples from the target distribution
  - However, we only have access to uniform PRNGs
  - For some simple distributions, this can be done by inverting the CDF
    - Inverting the CDF: $u \sim \mathbf{U}(0,1)$, then $F^{-1}(u) \sim \pi(x)$

- For distributions w/o tractable CDF, **rejection sampling** can be used

given proposal $q(x)$ s.t. $\pi(x) \le Cq(x)$:
for $i = 1 \dots N$:
    repeat
        sample $\tilde{x} \sim q(x)$
        sample $u \sim \mathbf{U}(0,1)$
    until $u \le \pi(\tilde{x})/Cq(\tilde{x})$
    set $x_i = \tilde{x}$



  - the *acceptance probability* decays exponentially as the dimension increases

Importance Sampling

$$\mathbb{E}_\pi[f(x)] = \int \pi(x) f(x)\, dx = \int q(x) \frac{\pi(x)}{q(x)} f(x)\, dx = \mathbb{E}_q\left[\frac{\pi(x)}{q(x)} f(x)\right]$$

$$\approx \frac{1}{N}\sum_{i=1}^{N} \omega(x_i) f(x_i), \qquad \omega(x) = \frac{\pi(x)}{q(x)}, \qquad x_i \sim q(x), i = 1 \dots N$$

- **I**mportance **S**ampling also uses a proposal distribution that is easy to sample
- Rather than discarding some samples, IS makes use of all samples but assigns individual ***importance weights*** to them

- How about if we can only evaluate $\tilde{\pi}(x)$ rather than $\pi(x) = \frac{\tilde{\pi}(x)}{\int \tilde{\pi}(x)\, dx}$?
  - Let $\tilde{\omega}(x) = \tilde{\pi}(x)/q(x),\ Z = \int \tilde{\pi}(x)\, dx$. Then:

$$Z = \int \tilde{\pi}(x)\, dx = \int q(x) \frac{\tilde{\pi}(x)}{q(x)}\, dx = \mathbb{E}_q[\tilde{\omega}(x)] \approx \frac{1}{N}\sum_{i=1}^{N} \tilde{\omega}(x_i), \qquad x_i \sim q(x)$$

$$\frac{1}{N}\omega(x_i) = \frac{1}{N}\frac{\pi(x_i)}{q(x_i)} = \frac{1}{N}\frac{1}{Z}\frac{\tilde{\pi}(x_i)}{q(x_i)} \approx \frac{\tilde{\omega}(x_i)}{\sum_{k=1}^{N}\tilde{\omega}(x_k)} \triangleq w_i, \qquad \mathbb{E}_\pi[f(x)] \approx \sum_{i=1}^{N} w_i f(x_i)$$

self-normalized importance sampling estimator

Sequential Importance Sampling (1)

- Aim: approximate a sequence of target distributions $\{\pi_t(x_{1:t})\}_{t\geq 1}$
  - $\pi_1(x_1), \pi_2(x_{1:2}), \pi_3(x_{1:3})$, …
    - e.g., in state space models: $p_{\boldsymbol{\theta}}(\boldsymbol{z}_1|\boldsymbol{x}_1), p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:2}|\boldsymbol{x}_{1:2}), p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:3}|\boldsymbol{x}_{1:3})$, …
- IS: approximate $\mathbb{E}_{\pi_t}(f_t)$ by N weighted samples $\left\{x_{1:t}^{(i)}: i = 1 \dots N\right\}$:

$$\mathbb{E}_{\pi_t}(f_t) \approx \sum_{i=1}^{N} w_t^{(i)} f_t\left(x_{1:t}^{(i)}\right)$$

$$Z_t = \int \tilde{\pi}_t(x_{1:t})\, dx_{1:t} \approx \frac{1}{N}\sum_{i=1}^{N} \widetilde{\omega}\left(x_{1:t}^{(i)}\right)$$

- To generate such weighted samples, we need to sampling from a sequence of proposal distributions: $\{q_t(x_{1:t})\}_{t\geq 1}$
  - Randomly sampling from high-dimensional distributions can be inefficient
    - Requiring numerous samples to achieve desired accuracy & computationally expensive
- Basic idea of SIS: reuse $\left\{x_{1:t}^{(i)}: i = 1 \dots N\right\}$ to build $\left\{x_{1:t+1}^{(i)}: i = 1 \dots N\right\}$

Sequential Importance Sampling (2)

- **SIS** uses a structured proposal to reuse samples:

$$\begin{aligned} q_t(x_{1:t+1}) &= q_t(x_{1:t})q_{t+1}(x_{t+1}|x_{1:t}) \\ &= q_1(x_1)q_2(x_2|x_1)q_3(x_3|x_{1:2}) \dots q_{t+1}(x_{t+1}|x_{1:t}) \end{aligned}$$

- Given $x_{1:t}^{(i)} \sim q_t(x_{1:t})$, sampling $x_{t+1}^{(i)} \sim q_{t+1}\left(x_{t+1}\middle|x_{1:t}^{(i)}\right)$ to obtain $x_{1:t+1}^{(i)}$:

$$x_{1:t+1}^{(i)} = \left\langle x_{1:t}^{(i)}, x_{t+1}^{(i)} \right\rangle$$

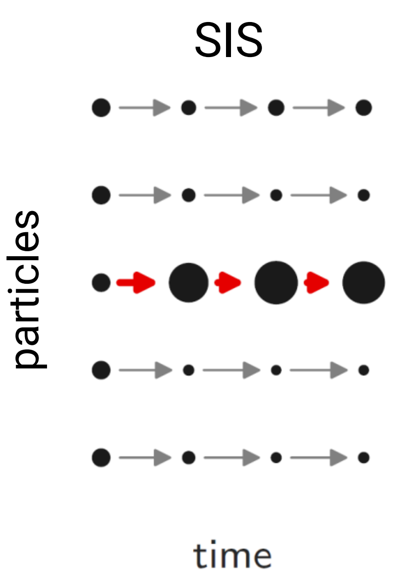
- And update the unnormalized weights as:

$$\tilde{\omega}_{t+1}(x_{1:t+1}) = \frac{\tilde{\pi}_{t+1}(x_{1:t+1})}{q_{t+1}(x_{1:t+1})} = \frac{\tilde{\pi}_{t+1}(x_{1:t+1})}{q_t(x_{1:t})q_{t+1}(x_{t+1}|x_{1:t})}$$

$$= \frac{\tilde{\pi}_t(x_{1:t})}{q_t(x_{1:t})} \frac{\tilde{\pi}_{t+1}(x_{1:t+1})}{\tilde{\pi}_t(x_{1:t})q_{t+1}(x_{t+1}|x_{1:t})} = \tilde{\omega}_t(x_{1:t}) \frac{\tilde{\pi}_{t+1}(x_{1:t+1})}{\tilde{\pi}_t(x_{1:t})q_{t+1}(x_{t+1}|x_{1:t})}$$

Sequential Importance Sampling (3)

- The overall SIS algorithm



input: number of samples N,
unnormalized target distributions $\{\tilde{\pi}_t(x_{1:t})\}_{t\geq 1}$,
proposals $q_1(\cdot)$ and $\{q_t(\cdot\,|x_{1:t-1})\}_{t>1}$

for $i = 1 \ldots N$:

sample $x_1^{(i)} \sim q_1(\cdot)$, set $\tilde{\omega}_1\left(x_1^{(i)}\right) = \frac{\tilde{\pi}_1\left(x_1^{(i)}\right)}{q_1\left(x_1^{(i)}\right)}$

for $t = 2 \ldots T$:

for $i = 1 \ldots N$:

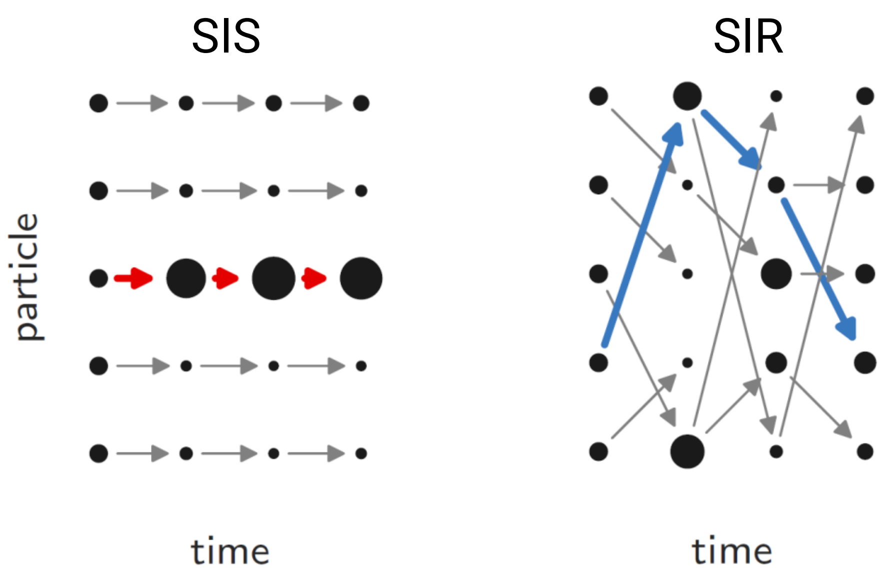
sample $x_t^{(i)} \sim q_t\left(\cdot\,\middle|x_{1:t-1}^{(i)}\right)$, set $x_{1:t}^{(i)} = \left\langle x_{1:t-1}^{(i)}, x_t^{(i)}\right\rangle$

set $\tilde{\omega}_t\left(x_{1:t}^{(i)}\right) = \tilde{\omega}_{t-1}\left(x_{1:t-1}^{(i)}\right)\frac{\tilde{\pi}_t\left(x_{1:t}^{(i)}\right)}{\tilde{\pi}_{t-1}\left(x_{1:t-1}^{(i)}\right)q_t\left(x_t^{(i)}|x_{1:t-1}^{(i)}\right)}$

output: $\left\{\left(x_{1:T}^{(i)}, w_T^{(i)}\right) : i = 1 \ldots N\right\}$, $\hat{Z}_T = \frac{1}{N}\sum_{i=1}^{N}\tilde{\omega}_T\left(x_{1:T}^{(i)}\right)$

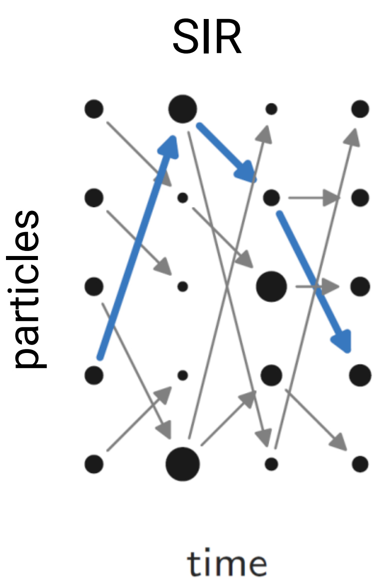
Sequential Importance Resampling

- Problem of SIS: the variance of $\left\{\overline{\omega}_t^{(i)}\right\}$ tends to grow as t increases
  - All the mass will concentrate on a few samples
    - Converge to $\overline{\omega}_t^{(i)} = 1$ for some i and others being zero
- Resampling to rescue: **S**equential **I**mportance **R**esampling
  - Using resampling to focus on promising samples/trajectories/particles



Sequential Monte Carlo

- The general SMC algorithm



input: number of samples N,
unnormalized target distributions $\{\tilde{\pi}_t(x_{1:t})\}_{t\geq 1}$,
proposals $q_1(\cdot)$ and $\{q_t(\cdot\,|x_{1:t-1})\}_{t>1}$

for $i = 1 \dots N$:

sample $x_1^{(i)} \sim q_1(\cdot)$, set $\tilde{\omega}_1\left(x_1^{(i)}\right) = \frac{\tilde{\pi}_1\left(x_1^{(i)}\right)}{q_1\left(x_1^{(i)}\right)}$, $w_1^{(i)} = \frac{\tilde{\omega}_1\left(x_1^{(i)}\right)}{\sum_{k=1}^{N}\tilde{\omega}_1\left(x_1^{(k)}\right)}$

for $t = 2 \dots T$:

for $i = 1 \dots N$:

sample ancestor index $a_{t-1}^{(i)} \sim \text{categorical}\left(\cdot\,\middle|w_{t-1}^{(i)}, \dots, w_{t-1}^{(N)}\right)$

sample $x_t^{(i)} \sim q_t\left(\cdot\,\middle|x_{1:t-1}^{a_{t-1}^{(i)}}\right)$, set $x_{1:t}^{(i)} = \left\langle x_{1:t-1}^{a_{t-1}^{(i)}}, x_t^{(i)}\right\rangle$

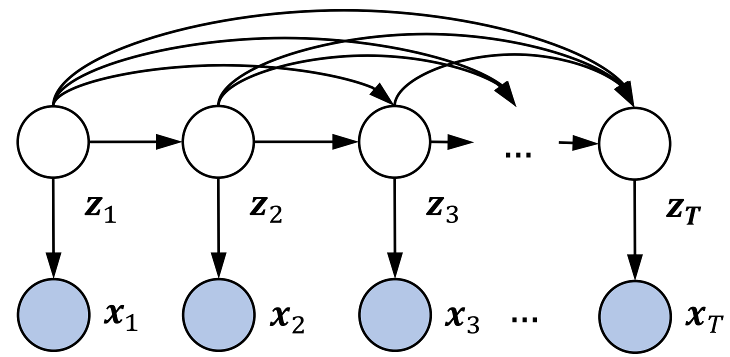
set $w_t^{(i)} \propto \tilde{\omega}_t\left(x_{1:t}^{(i)}\right) = \tilde{\pi}_t\left(x_{1:t}^{(i)}\right) / \left[\tilde{\pi}_{t-1}\left(x_{1:t-1}^{a_{t-1}^{(i)}}\right) q_t\left(x_t^{(i)}\middle|x_{1:t-1}^{a_{t-1}^{(i)}}\right)\right]$

output: $\left\{\left(x_{1:T}^{(i)}, w_T^{(i)}\right) : i = 1 \dots N\right\}$, $\hat{Z}_T = \prod_{t=1}^{T}\left[\frac{1}{N}\sum_{i=1}^{N}\tilde{\omega}_t\left(x_{1:t}^{(i)}\right)\right]$

Sequential Monte Carlo for SSM

- Consider the non-Markovian SSM

$$p_\theta(\boldsymbol{z}_{1:T}, \boldsymbol{x}_{1:T}) = \mu_\theta(\boldsymbol{z}_1) g_\theta(\boldsymbol{x}_1|\boldsymbol{z}_1) \prod_{t=2}^{T} f_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}) g_\theta(\boldsymbol{x}_t|\boldsymbol{z}_t)$$



- We set the unnormalized target distributions for SMC to be:

$$\pi_t(\cdot) = p_\theta(\boldsymbol{z}_{1:t}|\boldsymbol{x}_{1:t}) = \frac{p_\theta(\boldsymbol{z}_{1:t}, \boldsymbol{x}_{1:t})}{p_\theta(\boldsymbol{x}_{1:t})} = \frac{\tilde{\pi}_t(\cdot)}{Z_t}$$

- Now:

$$\tilde{\pi}_1(\cdot) = \mu_\theta(\boldsymbol{z}_1) g_\theta(\boldsymbol{x}_1|\boldsymbol{z}_1)$$

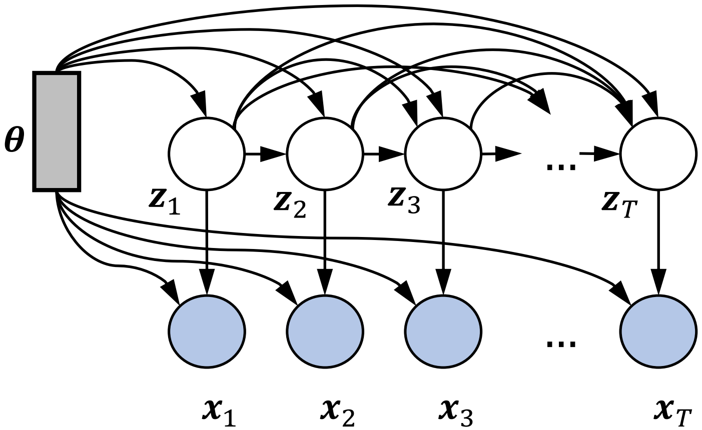
$$\frac{\tilde{\pi}_t(\cdot)}{\tilde{\pi}_{t-1}(\cdot)} = \frac{p_\theta(\boldsymbol{z}_{1:t}, \boldsymbol{x}_{1:t})}{p_\theta(\boldsymbol{z}_{1:t-1}, \boldsymbol{x}_{1:t-1})} = f_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}) g_\theta(\boldsymbol{x}_t|\boldsymbol{z}_t)$$

- And the proposals are structured as: $q_t(\cdot\,|x_{1:t-1}) = q_t(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{x}_{1:t})$

SMC for Parameter Learning

- Given the model

$$p_\theta(\boldsymbol{z}_{1:T}, \boldsymbol{x}_{1:T}) \\ = \mu_\theta(\boldsymbol{z}_1) g_\theta(\boldsymbol{x}_1|\boldsymbol{z}_1) \prod_{t=2}^{T} f_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}) g_\theta(\boldsymbol{x}_t|\boldsymbol{z}_t)$$



- We want to learn the MLE for θ: $\boldsymbol{\theta}_{MLE} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T})$

- Two basic approaches: gradient accent & **E**xpectation-**M**aximization

(1) Gradient accent: optimizing the likelihood estimated by SMC

$$\hat{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}) = \prod_{t=1}^{T} \left[\frac{1}{N} \sum_{i=1}^{N} \widetilde{\omega}_t^{(i)} \right]$$

(2) EM: using the SMC *smoothing* approximation at the E step

$$\mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})}[\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{z}_{1:T}, \boldsymbol{x}_{1:T})] \approx \sum_{i=1}^{N} w_i \mathcal{L}\left(\boldsymbol{\theta}, \boldsymbol{z}_{1:T}^{(i)}, \boldsymbol{x}_{1:T}\right)$$

ELBO and Monte Carlo Objectives

- Recall the ELBO in VI:

$$\mathbf{ELBO}(q_\phi) = \mathbb{E}_{q_\phi}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{z}, \boldsymbol{x})}{q_\phi(\mathbf{z}|\boldsymbol{x})}\right] \leq \log \mathbb{E}_{q_\phi}\left[\frac{p_{\boldsymbol{\theta}}(\mathbf{z}, \boldsymbol{x})}{q_\phi(\mathbf{z}|\boldsymbol{x})}\right]$$

$$= \log \int q_\phi(\mathbf{z}|\boldsymbol{x}) \frac{p_{\boldsymbol{\theta}}(\mathbf{z}, \boldsymbol{x})}{q_\phi(\mathbf{z}|\boldsymbol{x})} d\mathbf{z} = \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$$

- Note that $\mathbb{E}_{q_\phi}\left[\frac{p_{\boldsymbol{\theta}}(\mathbf{z},\boldsymbol{x})}{q_\phi(\mathbf{z}|\boldsymbol{x})}\right] = p_\theta(\boldsymbol{x})$
  - i.e., $\frac{p_{\boldsymbol{\theta}}(\mathbf{z},\boldsymbol{x})}{q_\phi(\mathbf{z}|\boldsymbol{x})}$ is an unbiased estimator of $p_\theta(\boldsymbol{x})$
- **M**onte **C**arlo **O**bjectives generalize the ELBO
  - Given an unbiased estimator $\hat{p}_\theta(\boldsymbol{x})$ of $p_\theta(\boldsymbol{x})$: $\mathbb{E}[\hat{p}_\theta(\boldsymbol{x})] = p_\theta(\boldsymbol{x})$, the MCO is:

$$\mathbf{MCO}(\cdot) = \mathbb{E}[\log \hat{p}_\theta(\boldsymbol{x})] \leq \log \mathbb{E}[\hat{p}_\theta(\boldsymbol{x})] = \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$$

Variational SMC (1)

- **M**onte **C**arlo **O**bjectives: given $\mathbb{E}[\hat{p}_\theta(\boldsymbol{x})] = p_\theta(\boldsymbol{x})$

$$\mathbf{MCO}(\cdot) = \mathbb{E}[\log \hat{p}_\theta(\boldsymbol{x})] \leq \log \mathbb{E}[\hat{p}_\theta(\boldsymbol{x})] = \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$$

- Recall that SMC provides an estimator for the margin distribution:

$$\hat{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}) = \prod_{t=1}^{T} \left[\frac{1}{N} \sum_{i=1}^{N} \widetilde{\omega}_t^{(i)}\right]$$

- It can be shown that $\hat{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T})$ is unbiased

$$\mathbb{E}_{Q_{\text{SMC}}}[\hat{p}_\theta(\boldsymbol{x}_{1:T})] = p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}), \qquad \text{where } \; Q_{\text{SMC}}(x_{1:T}^{1:N}, a_{1:T-1}^{1:N}) =$$

$$\left(\prod_{i=1}^{N} q_{1,\phi}\left(x_1^{(i)}\right)\right) \prod_{t=2}^{T} \prod_{i=1}^{N} q_{t,\phi}\left(x_t^{(i)} \middle| x_{1:t-1}^{a_{t-1}^{(i)}}\right) \cdot \text{categorical}\left(a_{t-1}^{(i)} \middle| w_{t-1}^{1:N}\right)$$

Variational SMC (2)

- So we can construct a MCO using SMC:

$$\mathbf{MCO}_{\mathrm{SMC}}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}_{1:T}) = \mathbb{E}_{Q_{\mathrm{SMC}}}[\log \hat{p}_\theta(\boldsymbol{x}_{1:T})]$$

$$= \mathbb{E}_{Q_{\mathrm{SMC}}}\left[\prod_{t=1}^{T}\left(\frac{1}{N}\sum_{i=1}^{N}\widetilde{\omega}_t\left(\boldsymbol{x}_{1:t}^{(i)}\right)\right)\right]$$

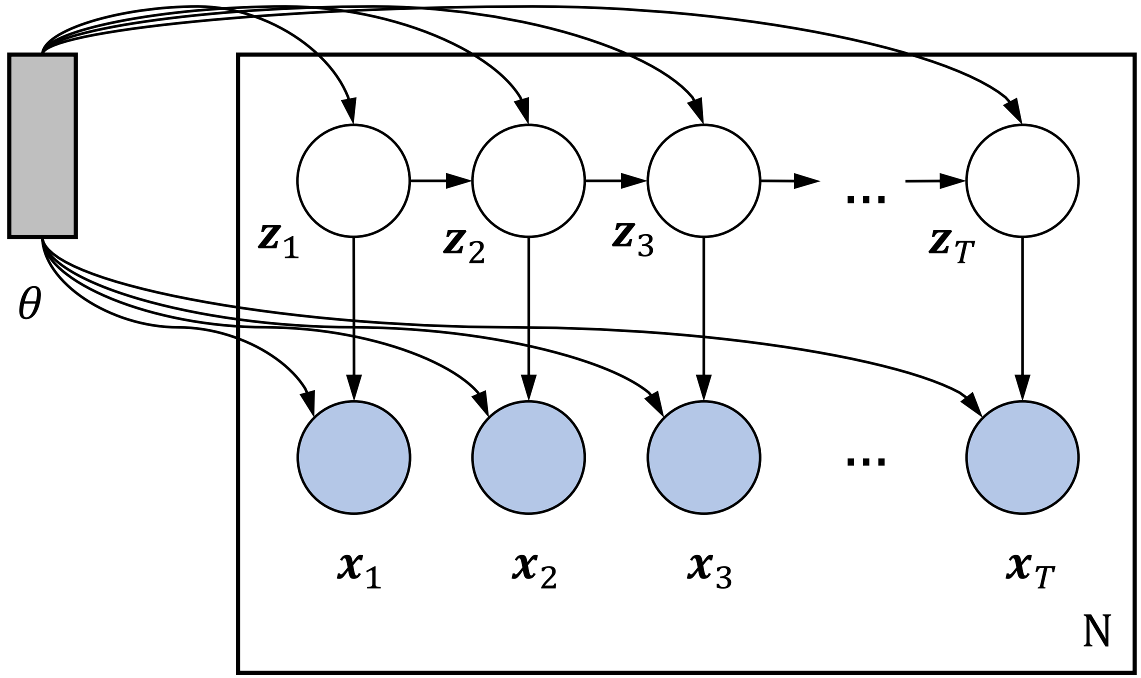
  - $\boldsymbol{\theta}$ are the model parameters, $\boldsymbol{\phi}$ are parameters of the proposal distribution

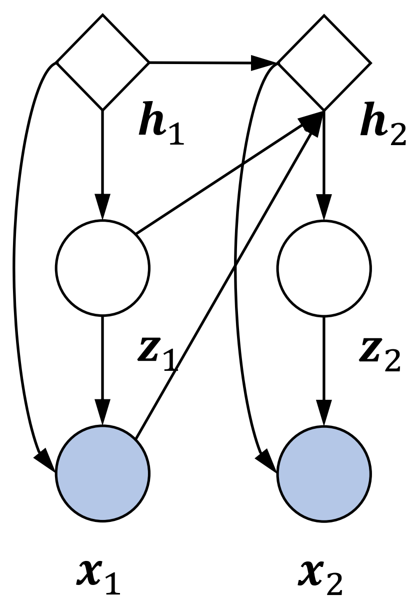
- Now by optimizing this objective with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, we can learn the model parameters and the proposals together
  - This objective is (much) tighter than ELBO: $\mathbf{MCO}_{\mathrm{SMC}} \to \log p_\theta(\boldsymbol{x}_{1:T})$ as $N \to \infty$
  - In practice, it leads to better models and faster convergence
  - The learned proposals can be very useful

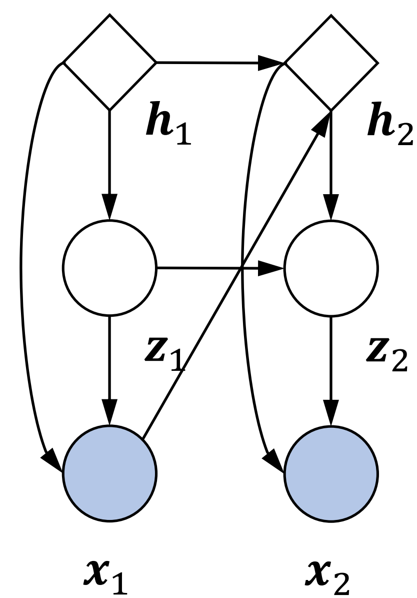
Outline

- Short (Re)Introduction to VI and VAE
- State Space Models and Stochastic RNNs
- Variational Inference of SSM
- (Variational) Sequential Monte Carlo
- **Conclusions**

Backup Slides


θ
z1
z2
z3
...
zT
x1
x2
x3
...
xT
N


h1
h2
z1
z2
x1
x2


h1
h2
z1
z2
x1
x2