Diffusion posterior sampling for simulation-based inference in tall data settings

Gabriel V. Cardoso (CMAP - Ecole Polytechnique)

In collaboration with: J. Linhart, A. Gramfort, S. Le Corff, PLC Rodrigues.

March 25, 2024

1. Simulation-based inference

2. Score Based generative modelling

3. "Tall" Simulation-based inference with Score based generative models

Simulation-based inference

We observe data $\mathcal{D} := \{(x_1, \theta_1), \cdots, (x_n, \theta_n)\}$ from a simulator $(p(x|\theta))$.

Simulation-based inference

We observe data $\mathcal{D} := \{(x_1, \theta_1), \cdots, (x_n, \theta_n)\}$ from a simulator ($p(x|\theta)$).

Given prior information on θ, via $\lambda(\theta)$, our goal is to sample the posterior for a new set of observations $x_1^\star, \cdots, x_n^\star$:

$$p(\theta | x_1^\star, \cdots, x_n^\star) \,.$$

Simulation-based inference

We observe data $\mathcal{D} := \{(x_1, \theta_1), \cdots, (x_n, \theta_n)\}$ from a simulator $(p(x|\theta))$.

Given prior information on θ, via $\lambda(\theta)$, our goal is to sample the posterior for a new set of observations $x_1^\star, \cdots, x_n^\star$:

$$p(\theta|x_1^\star, \cdots, x_n^\star) \,.$$

We assume we **cannot evaluate the simulator likelihood** $p(x|\theta)$. (It can be result of an ODE, SDE or some complicated fuction of θ!)

Generative models and SBI

Learn $p(\theta|x)$ from the dataset $\mathcal{D}$ using

- Conditional normalizing flows[1],
- Score based generative models[2].

[1] George Papamakarios et al. "Sequential Neural Likelihood: Fast Likelihood-free Inference with Autoregressive Flows". In: 89 (). Ed. by Kamalika Chaudhuri and Masashi Sugiyama, pp. 837–848.

[2] Louis Sharrock et al. "Sequential Neural Score Estimation: Likelihood-Free Inference with Conditional Score Based Diffusion Models". In: (), Tomas Geffner et al. "Compositional Score Modeling for Simulation-based Inference". In: ().

Generative models, SBI and "tall" data

How to use the generative model for $p(\theta|x)$ to sample from $p(\theta|x_1^\star, \cdots, x_n^\star)$?

Score and sampling: Langevin algorithm

Let $\Theta_0 \sim \mu_0$ and

$$\Theta_t = \Theta_{t-1} + \gamma \nabla \log \pi(\Theta_{t-1}) + \sqrt{2\gamma}\epsilon_t \,.$$

For $\delta > 0$, appropriate choices[3] of γ and t lead to $W_2^2(\mathcal{L}(\Theta_t), \pi) < \delta$.

[3] Alain Durmus et al. "Analysis of Langevin Monte Carlo via Convex Optimization". In: *Journal of Machine Learning Research* 20.73 (), pp. 1–46.

Score matching

Let $\mathrm{s}_\psi(\cdot)$ be a neural network, $\psi \in \Psi \subset \mathbb{R}^d$.

Score matching

Let $\mathrm{s}_\psi(\cdot)$ be a neural network, $\psi \in \Psi \subset \mathbb{R}^d$.

Score Matching

$$\underset{\psi \in \Psi}{\operatorname{argmin}} \, \mathbb{E}_{\Theta \sim \pi} \left[\left\| \mathrm{s}_\psi(\Theta) - \nabla \log \pi(\Theta) \right\|^2 \right] \tag{1}$$

Score matching: Learning from data.

- Implicit Score Matching[4]

[4] Aapo Hyvärinen and Peter Dayan. "Estimation of non-normalized statistical models by score matching.". In: *Journal of Machine Learning Research* 6.4 ().

[5] Pascal Vincent. "A connection between score matching and denoising autoencoders". In: *Neural computation* 23.7 (), pp. 1661–1674.

Score matching: Learning from data.

- Implicit Score Matching[4]
- DSM

Denoise Score Matching[5]

If $\Theta_\sigma = \Theta + \sigma\epsilon$ with $\Theta \sim \pi$, $\epsilon \sim \mathcal{N}(0, \mathrm{Id})$, $\pi_\sigma = \mathcal{L}(\Theta_\sigma)$ then (2) (for π_σ) is equivalent to

$$\underset{\psi \in \Psi}{\operatorname{argmin}} \, \mathbb{E}_{\Theta \sim \pi, \epsilon \sim \mathcal{N}(0,\mathrm{Id})} \left[\| \mathrm{s}_\psi(\Theta + \sigma\epsilon) - \underbrace{(-\sigma^{-1}\epsilon)}_{\nabla \log \mathbb{P}(\Theta + \sigma\epsilon | \Theta)} \|^2 \right]. \quad (2)$$

[4] Aapo Hyvärinen and Peter Dayan. "Estimation of non-normalized statistical models by score matching.". In: *Journal of Machine Learning Research* 6.4 ().

[5] Pascal Vincent. "A connection between score matching and denoising autoencoders". In: *Neural computation* 23.7 (), pp. 1661–1674.

Score Matching: not enough

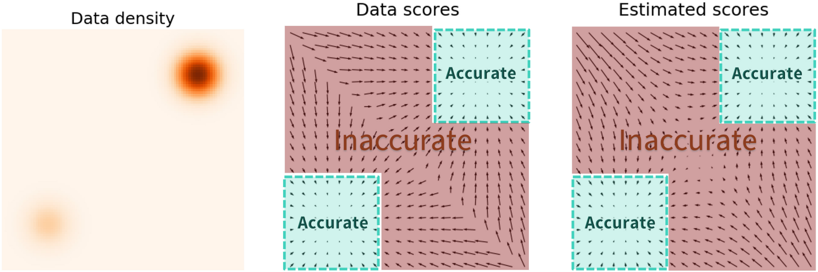


Figure: Taken from `https://yang-song.net/blog/2021/score/`.

Multi-level perturbation

Consider $0 < \sigma_1 < \cdots < \sigma_T$, $\Theta_0 \sim \pi$ and for $t \in \{1, \cdots, T\}$

$$\Theta_t = \Theta_{t-1} + \sqrt{\sigma_t^2 - \sigma_{t-1}^2}\epsilon_t \, ,$$

where $\epsilon_t \sim \mathcal{N}(0, \mathrm{Id})$.

Multi-level perturbation

Consider $0 < \sigma_1 < \cdots < \sigma_T$, $\Theta_0 \sim \pi$ and for $t \in \{1, \cdots, T\}$

$$\Theta_t = \Theta_{t-1} + \sqrt{\sigma_t^2 - \sigma_{t-1}^2}\epsilon_t \,,$$

where $\epsilon_t \sim \mathcal{N}(0, \mathrm{Id})$.

Figure: Noising an image with increasing Gaussian noise. Taken from `https://yang-song.net/blog/2021/score/`.

Multi-level perturbation

Consider $0 < \sigma_1 < \cdots < \sigma_T$, $\Theta_0 \sim \pi$ and for $t \in \{1, \cdots, T\}$

$$\Theta_t = \Theta_{t-1} + \sqrt{\sigma_t^2 - \sigma_{t-1}^2}\epsilon_t \, ,$$

where $\epsilon_t \sim \mathcal{N}(0, \mathrm{Id})$.

Figure: Noising an image with increasing Gaussian noise. Taken from `https://yang-song.net/blog/2021/score/`.

Then $\mathcal{L}(\Theta_t) = p_t$ and we can jointly approximate the scores of $\{p_t\}_{t=1}^T$ by:

$$\underset{\psi \in \Psi}{\operatorname{argmin}} \sum_{t=1}^{T} \varkappa_t^2 \mathbb{E}_{\Theta \sim \pi, \epsilon \sim \mathcal{N}(0,\mathrm{Id})} \left[\| \mathrm{s}_\psi(\Theta + \sigma_t \epsilon, \sigma_t) + \sigma_t^{-1} \epsilon \|^2 \right] \, .$$

Score based generative modelling

Current score based generative modelling consists of approximatively sampling backward from p_T to p_1 by exploiting $\{\mathrm{s}_\psi(\cdot, \sigma_t)\}_{t=1}^T$.

[6] Yang Song and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution". In: *Advances in neural information processing systems* 32 ().

[7] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations". In.

[8] Tero Karras et al. "Elucidating the Design Space of Diffusion-Based Generative Models". In.

[9] Jiaming Song et al. "Denoising Diffusion Implicit Models". In.

Score based generative modelling

Current score based generative modelling consists of approximatively sampling backward from p_T to p_1 by exploiting $\{\mathrm{s}_\psi(\cdot, \sigma_t)\}_{t=1}^{T}$.

Several options available: Sequential Langevin[6], SDE[7], ODE[8], "Markov Chain"[9].

[6] Yang Song and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution". In: *Advances in neural information processing systems* 32 ().

[7] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations". In.

[8] Tero Karras et al. "Elucidating the Design Space of Diffusion-Based Generative Models". In.

[9] Jiaming Song et al. "Denoising Diffusion Implicit Models". In.

Score based generative modelling: DDIM

Goal: "Pass" from Θ_t to Θ_{t-1}.

[10] $m(\theta_0, \theta_t) = \theta_0 + \frac{\sigma_{t-1}^2}{\sigma_t^2}(\theta_t - \theta_0)$ and $\sigma_{t-1|t,0}^2 = (\sigma_t^2 - \sigma_{t-1}^2)\frac{\sigma_{t-1}^2}{\sigma_t^2}$.

[11] Jiaming Song et al. "Denoising Diffusion Implicit Models". In.

Score based generative modelling: DDIM

Goal: "Pass" from Θ_t to Θ_{t-1}.

Bridge

$$p_{t-1|t,0}(\theta_{t-1}|\theta_t, \theta_0) = \frac{p_{t|t-1}(\theta_t|\theta_{t-1})p_{t-1|0}(\theta_{t-1}|\theta_0)}{p_{t|0}(\theta_t|\theta_0)} = \mathcal{N}(\theta_{t-1}; m(\theta_0, \theta_t), \sigma^2_{t-1|t,0}\,\mathrm{Id})\,.$$

10

---

[10] $m(\theta_0, \theta_t) = \theta_0 + \frac{\sigma^2_{t-1}}{\sigma^2_t}(\theta_t - \theta_0)$ and $\sigma^2_{t-1|t,0} = (\sigma^2_t - \sigma^2_{t-1})\frac{\sigma^2_{t-1}}{\sigma^2_t}$.

[11] Jiaming Song et al. "Denoising Diffusion Implicit Models". In.

Score based generative modelling: DDIM

Goal: "Pass" from Θ_t to Θ_{t-1}.

Bridge

$$p_{t-1|t,0}(\theta_{t-1}|\theta_t, \theta_0) = \frac{p_{t|t-1}(\theta_t|\theta_{t-1})p_{t-1|0}(\theta_{t-1}|\theta_0)}{p_{t|0}(\theta_t|\theta_0)} = \mathcal{N}(\theta_{t-1}; m(\theta_0, \theta_t), \sigma^2_{t-1|t,0}\,\mathrm{Id})\,.$$

[10]

DDIM Backward kernel

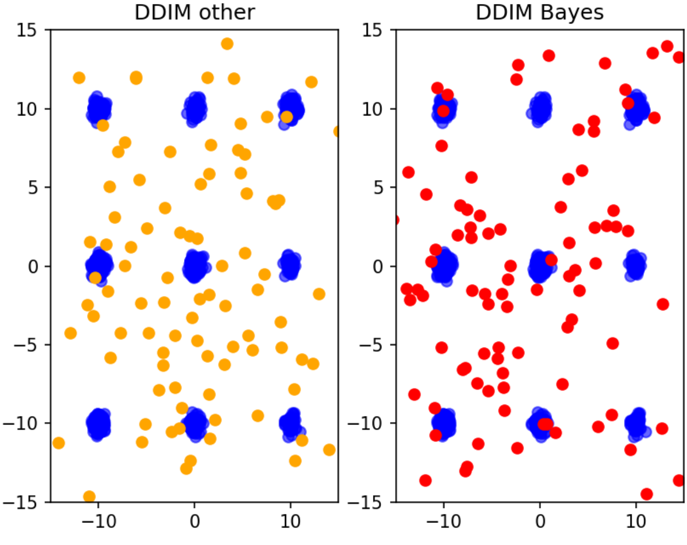
As $\mathbb{E}\left[\Theta_0|\Theta_t = \theta_t\right] = \theta_t + \sigma_t^2 \nabla \log p_t(\theta_t)$, define[11],

$$\overleftarrow{p}_{t-1|t}(\theta_{t-1}|\theta_t) = p_{t-1|t,0}(\theta_{t-1}|\theta_t, \theta_0 = \theta_t + \sigma_t^2 \mathrm{s}_\psi(\theta_t, \sigma_t))\,.$$

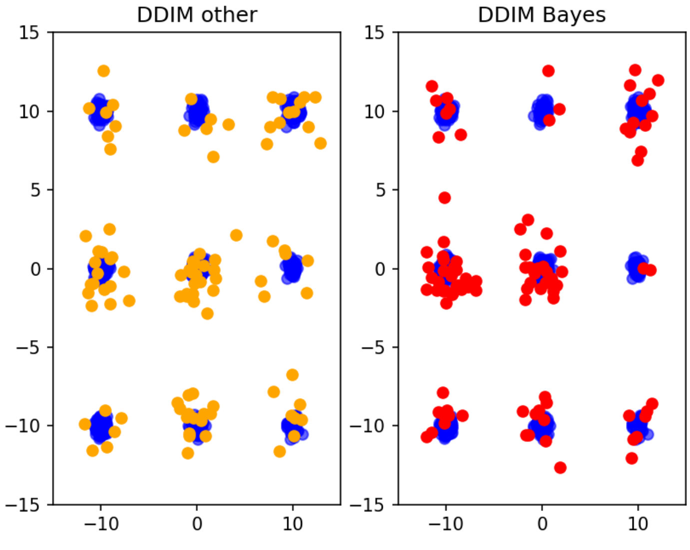
[10] $m(\theta_0, \theta_t) = \theta_0 + \frac{\sigma^2_{t-1}}{\sigma^2_t}(\theta_t - \theta_0)$ and $\sigma^2_{t-1|t,0} = (\sigma_t^2 - \sigma^2_{t-1})\frac{\sigma^2_{t-1}}{\sigma^2_t}$.

[11] Jiaming Song et al. "Denoising Diffusion Implicit Models". In.

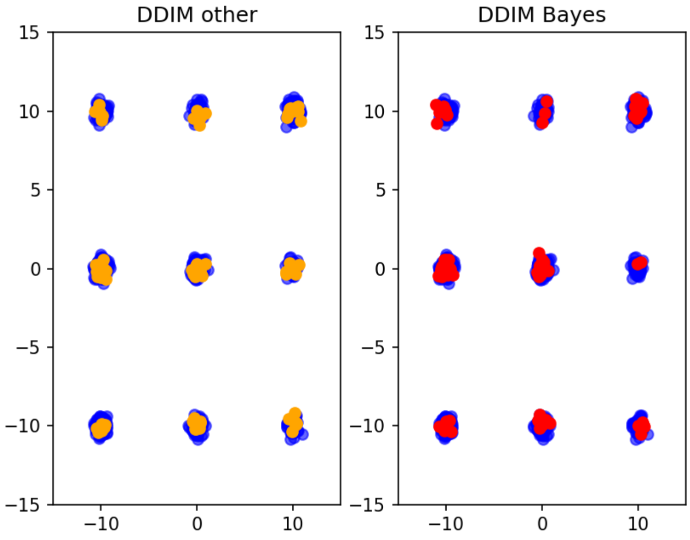
Score based generative modelling: DDIM



Score based generative modelling: DDIM



Score based generative modelling: DDIM



Diffusion for SBI[12]

In SBI, $\pi = p(\theta|x)$ by learning the scores conditionaly on x:

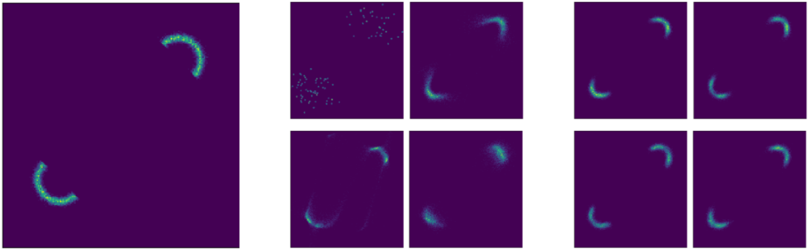
$$\underset{\psi \in \Psi}{\text{argmin}} \sum_{t=1}^{T} \varkappa_t^2 \mathbb{E}_{(X,\theta) \sim p(X|\theta)\lambda(\theta), \epsilon \sim \mathcal{N}(0, \text{Id})} \left[\| \mathrm{s}_\psi(\theta + \sigma_t \epsilon, X, \sigma_t) + \sigma_t^{-1} \epsilon \|^2 \right] .$$

Then,

$$\mathrm{s}_\psi(\theta_t, x, \sigma_t) \approx \nabla_{\theta_t} \log p_t(\theta_t|x) = \nabla_{\theta_t} \log \int \mathcal{N}(\theta_t; \theta, \sigma_t^2 \, \text{Id}) p(\theta|x) \mathrm{d}\theta \, .$$

[12] Tomas Geffner et al. "Compositional Score Modeling for Simulation-based Inference". In: (), Louis Sharrock et al. "Sequential Neural Score Estimation: Likelihood-Free Inference with Conditional Score Based Diffusion Models". In: ().

Diffusion for SBI



(a) True posterior.

(b) Existing algorithms: SMC-ABC (top left), SNLE (top right), SNPE (bottom left), SNRE (bottom right).

(c) Our algorithms: NLSE (top left), NPSE (top right), SNLSE (bottom left), SNPSE (bottom right).

Figure: Figure taken from [7][13]

[13] Louis Sharrock et al. "Sequential Neural Score Estimation: Likelihood-Free Inference with Conditional Score Based Diffusion Models". In: ().

"Tall" score: Geffner solution

Following[14] consider $\{\tilde{\pi}_t\}_{t=1}^T$ such that

$$\nabla \log \tilde{\pi}_t(\theta_t) = (1-n)\nabla \log \overleftarrow{p}_t^{\lambda}(\theta_t) + \sum_{i=1}^{n} \underbrace{\nabla \log \overleftarrow{p}_t(\theta_t | x_i^\star)}_{\mathrm{s}_\psi(\theta_t, x_j^\star, \sigma_t)} .$$

[14] Tomas Geffner et al. "Compositional Score Modeling for Simulation-based Inference". In: ().

"Tall" score: Geffner solution

Following[14] consider $\{\tilde{\pi}_t\}_{t=1}^T$ such that

$$\nabla \log \tilde{\pi}_t(\theta_t) = (1-n)\nabla \log \overleftarrow{p}_t^{\lambda}(\theta_t) + \sum_{i=1}^{n} \underbrace{\nabla \log \overleftarrow{p}_t(\theta_t|x_i^\star)}_{\mathrm{s}_\psi(\theta_t, x_j^\star, \sigma_t)} .$$

But $\boxed{\tilde{\pi}_t(\theta_t|x_{1:n}^\star) \neq p_t(\theta_t|x_{1:n}^\star) = \int \mathcal{N}(\theta_t; \theta, \sigma_t^2 \, \mathrm{Id}) p(\theta|x_{1:n}^\star) \mathrm{d}\theta \ !}$

[14] Tomas Geffner et al. "Compositional Score Modeling for Simulation-based Inference". In: ().

"Tall" score: Geffner solution

Following[14] consider $\{\tilde{\pi}_t\}_{t=1}^{T}$ such that

$$\nabla \log \tilde{\pi}_t(\theta_t) = (1-n)\nabla \log \overleftarrow{p}_t^{\lambda}(\theta_t) + \sum_{i=1}^{n} \underbrace{\nabla \log \overleftarrow{p}_t(\theta_t | x_i^\star)}_{\mathrm{s}_\psi(\theta_t, x_j^\star, \sigma_t)} .$$

But $\boxed{\tilde{\pi}_t(\theta_t | x_{1:n}^\star) \neq p_t(\theta_t | x_{1:n}^\star) = \int \mathcal{N}(\theta_t; \theta, \sigma_t^2 \, \mathrm{Id}) p(\theta | x_{1:n}^\star) \mathrm{d}\theta \ !}$

Available samplers:

Sequential Langevin ✓, SDE✗, ODE✗, "Markov Chain"✗.

[14] Tomas Geffner et al. "Compositional Score Modeling for Simulation-based Inference". In: ().

Approximating the posterior score

Tall posterior score

$$\nabla_\theta \log \overleftarrow{p}_t(\theta \mid x^\star_{1:n}) = \overbrace{(1-n)\nabla_\theta \log \overleftarrow{p}^\lambda_t(\theta) + \sum_{j=1}^{n} \nabla_\theta \log \overleftarrow{p}_t(\theta \mid x^\star_j)}^{\text{Same as in Geffner!}}$$

$$+ \nabla_\theta \log L_\lambda(\theta, x^\star_{1:n}),$$

with $L_\lambda(\theta, x^\star_{1:n}) := \int \overleftarrow{p}^\lambda_{0|t}(\theta_0|\theta)^{1-n} \prod_{j=1}^n \overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j)\mathrm{d}\theta_0$.

Approximating $L_\lambda(\theta, x^\star_{1:n})$

$$L_\lambda(\theta, x^\star_{1:n}) := \int \overleftarrow{p}^{\lambda}_{0|t}(\theta_0|\theta)^{1-n} \prod_{j=1}^{n} \overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j)\mathrm{d}\theta_0 \,.$$

[15] Benjamin Boys et al. "Tweedie moment projected diffusions for inverse problems". In: *arXiv preprint arXiv:2310.06721* ().

[16] Jiaming Song et al. "Pseudoinverse-Guided Diffusion Models for Inverse Problems". In.

Approximating $L_\lambda(\theta, x^\star_{1:n})$

$$L_\lambda(\theta, x^\star_{1:n}) := \int \overleftarrow{p}^{\lambda}_{0|t}(\theta_0|\theta)^{1-n} \prod_{j=1}^{n} \overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j)\mathrm{d}\theta_0 \,.$$

Second order approximation

$$\overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j) \approx \mathcal{N}(\theta_0; \mu_t(\theta, x^\star_j), \Sigma_t(\theta, x^\star_j)) \,.$$

[15] Benjamin Boys et al. "Tweedie moment projected diffusions for inverse problems". In: *arXiv preprint arXiv:2310.06721* ().

[16] Jiaming Song et al. "Pseudoinverse-Guided Diffusion Models for Inverse Problems". In.

Approximating $L_\lambda(\theta, x^\star_{1:n})$

$$L_\lambda(\theta, x^\star_{1:n}) := \int \overleftarrow{p}^{\lambda}_{0|t}(\theta_0|\theta)^{1-n} \prod_{j=1}^{n} \overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j)\mathrm{d}\theta_0 \,.$$

Second order approximation

$$\overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j) \approx \mathcal{N}(\theta_0; \mu_t(\theta, x^\star_j), \Sigma_t(\theta, x^\star_j))\,.$$

$$\mu_t(\theta, x^\star_j) := \mathbb{E}\left[\Theta_0|\Theta_t = \theta, x^\star_j\right] = \theta + \sigma^2 \nabla \log \overleftarrow{p}_t(\theta|x^\star_j).$$

- JAC[15]: $\Sigma_t(\theta, x) := \nabla_\theta \mu_t(\theta, x)$.
- COV[16]: $\Sigma_t(\theta, x) := \sigma_t^2 \, \mathrm{Id} + \mathrm{Cov}(\Theta_0|x)$.

---

[15] Benjamin Boys et al. "Tweedie moment projected diffusions for inverse problems". In: *arXiv preprint arXiv:2310.06721* ().

[16] Jiaming Song et al. "Pseudoinverse-Guided Diffusion Models for Inverse Problems". In.

Approximating $L_\lambda(\theta, x^\star_{1:n})$

Under

$$\overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j) \approx \mathcal{N}(\theta_0; \mu_t(\theta, x^\star_j), \Sigma_t(\theta, x^\star_j))\,.$$

Approximating $L_\lambda(\theta, x^\star_{1:n})$

Under

$$\overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j) \approx \mathcal{N}(\theta_0; \mu_t(\theta, x^\star_j), \Sigma_t(\theta, x^\star_j))\,.$$

$$L_\lambda(\theta, x^\star_{1:n}) := \int \underbrace{\overleftarrow{p}^{\lambda}_{0|t}(\theta_0|\theta)^{1-n} \prod_{j=1}^{n} \overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j)}_{\text{Product of Gaussian pdfs!}} \,\mathrm{d}\theta_0\,.$$

Approximating $L_\lambda(\theta, x^\star_{1:n})$

Under

$$\overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j) \approx \mathcal{N}(\theta_0; \mu_t(\theta, x^\star_j), \Sigma_t(\theta, x^\star_j))\,.$$

$$L_\lambda(\theta, x^\star_{1:n}) := \int \underbrace{\overleftarrow{p}^{\lambda}_{0|t}(\theta_0|\theta)^{1-n} \prod_{j=1}^{n} \overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j)}_{\text{Product of Gaussian pdfs!}} \,\mathrm{d}\theta_0\,.$$

Let $\ell_\lambda(\theta, x^\star_{1:n})$ be the resulting approximation.
Under appropriate conditions, we can calculate (autograd) $\nabla \ell_\lambda(\theta, x^\star_{1:n}) \approx \nabla L_\lambda(\theta, x^\star_{1:n})$

Approximating $L_\lambda(\theta, x^\star_{1:n})$

Under

$$\overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j) \approx \mathcal{N}(\theta_0; \mu_t(\theta, x^\star_j), \Sigma_t(\theta, x^\star_j)) \,.$$

$$L_\lambda(\theta, x^\star_{1:n}) := \int \underbrace{\overleftarrow{p}^{\lambda}_{0|t}(\theta_0|\theta)^{1-n} \prod_{j=1}^{n} \overleftarrow{p}_{0|t}(\theta_0|\theta, x^\star_j)}_{\text{Product of Gaussian pdfs!}} \, \mathrm{d}\theta_0 \,.$$

Let $\ell_\lambda(\theta, x^\star_{1:n})$ be the resulting approximation.
Under appropriate conditions, we can calculate (autograd) $\nabla \ell_\lambda(\theta, x^\star_{1:n}) \approx \nabla L_\lambda(\theta, x^\star_{1:n})$

Highly unstable! (i.e, does not work at all)

Approximating $L_\lambda(\theta, x^\star_{1:n})$

Lemma (Score approximation [17])

We can write

$$\nabla_\theta \log \overleftarrow{p}_t(\theta \mid x^\star_{1:n}) = \Lambda(\theta)^{-1} \sum_{j=1}^{n} \Sigma_{t,j}^{-1}(\theta) \nabla_\theta \log \overleftarrow{p}_t(\theta \mid x^\star_j)$$
$$+ (1-n)\Lambda(\theta)^{-1} \Sigma_{t,\lambda}^{-1}(\theta) \nabla_\theta \log \overleftarrow{p}^{\lambda}_t(\theta) + F(\theta, x^\star_{1:n}),$$

and F satisfies

$$\nabla_\theta \Sigma_{t,j}(\theta) = 0 \quad \textit{and} \quad \nabla_\theta \Sigma_{\lambda,t}(\theta) = 0 \Rightarrow F(\theta, x^\star_{1:n}) = 0$$

.

[17] $\Lambda(\theta) = \sum_{j=1}^{n} \Sigma_{t,j}^{-1}(\theta) + (1-n)\Sigma_{t,\lambda}^{-1}(\theta)$.

Results

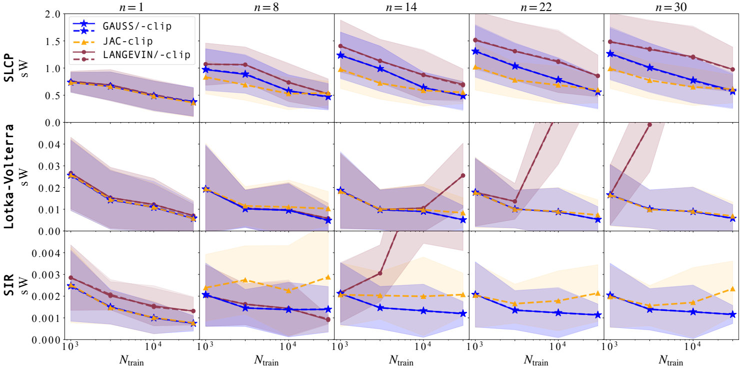


Figure: sW distance as a function of $N_{\text{train}} \in [10^3, 3.10^3, 10^4, 3.10^4]$ between the samples obtained by each algorithm and the true tall posterior distribution $p(\theta \mid x^{\star}_{1,n})$ (for $n \in [1, 8, 14, 22, 30]$). Mean and std over 20 different parameters $\theta^{\star} \sim \lambda(\theta)$.

Conclusion and Perspectives

- Is there a better second order approximation of $p_{0|t}$? Variational Inference?
- Can we approximate L_λ by MCMC (or something else)?
- Link between $p(\theta|x)$ and approximation error?