Distributed Monte Carlo with application to large-scale Bayesian inference

Ecole Polytechnique
Lagrange Mathematical and Computing Research Center

V. Plassier, A. Durmus, E. Moulines

April 5, 2023

Uncertainty Quantification

Uncertainty in ML

- Representing uncertainty is critical to decision making.
- Machine (and especially Deep) learning models typically lack a representation of uncertainty [overconfident and miscalibrated]
- Objective: quantify uncertainty in a trustworthy way.

Methods for reasoning and making decisions under uncertainty are an important building block of accurate, reliable, and interpretable machine learning systems. In many applications - ranging from supply chain planning to medical diagnosis to autonomous driving - faithfully assessing uncertainty can be as important as obtaining high accuracy. *Ermon, 2018*

"Classical" supervised ML problem

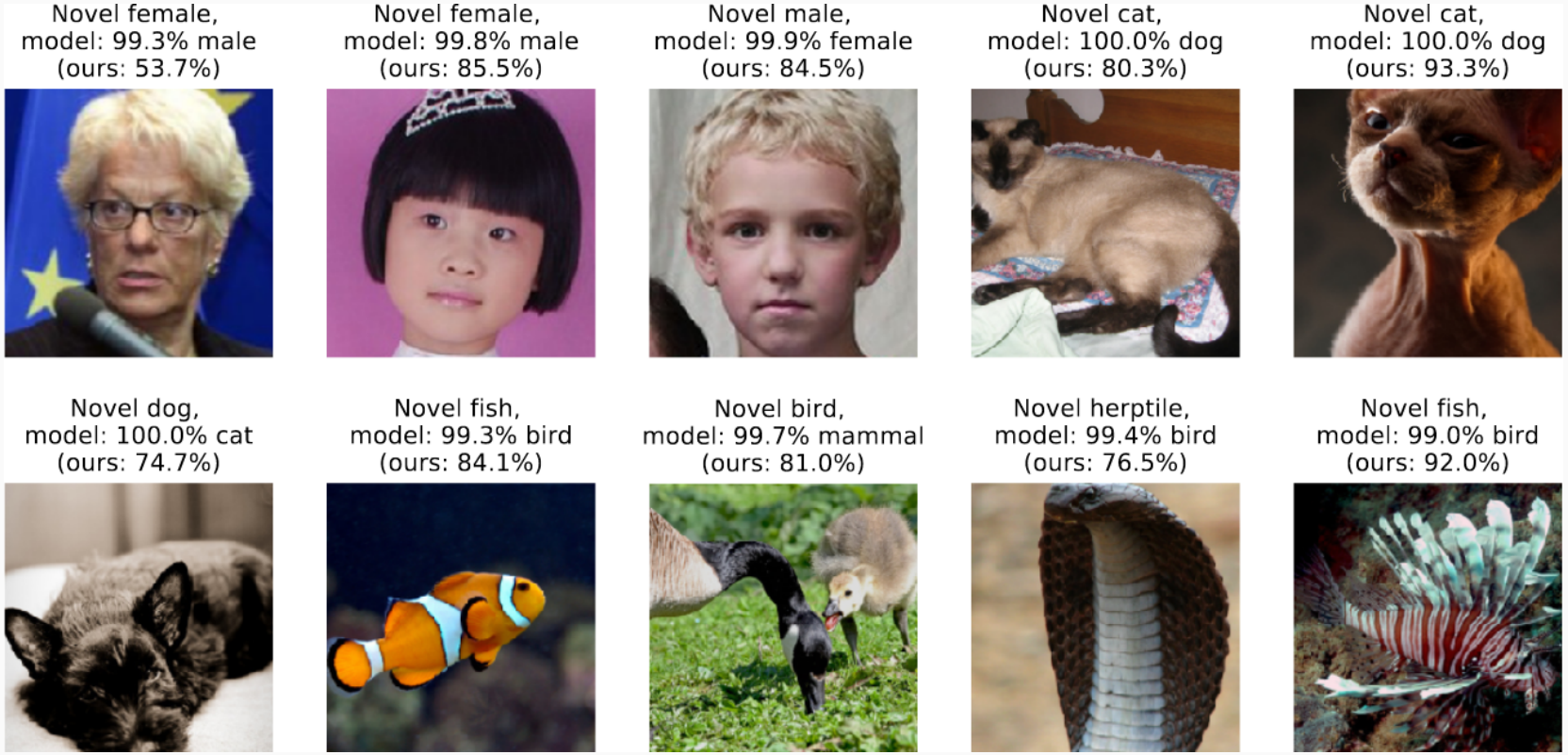
- Training set $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^{n}$, where y_i is the response (dependent variable) and x_i the explanatory variables (or attributes, independent variables).
- Statistical model $\mathscr{F} = \{(x, y) \to p(y|x;\theta) : \theta \in \mathbb{R}^d\}$
- Parameter estimation

$$\widehat{\theta} \approx \arg\min_{\theta \in \mathbb{R}^d} \left\{ -\frac{1}{|\mathscr{D}|} \sum_{(x,y) \in \mathscr{D}} \log p(y|x;\theta) + \lambda J(\theta) \right\}$$

- **Predictive distribution** $\approx p(y|x, \widehat{\theta})$

Overconfidence

Classical **training** method for Neural Network $\Longrightarrow$ **poor predictive uncertainty** $p(y|x;\widehat{\theta})$



Calibration metrics

classification task with $C \geq 2$ possible classes. For a sample x, the quantity $\widehat{p}(y|x)$ is a probabilistic prediction, $y \in \{1, \ldots, C\}$.

Set $\widehat{y}(x) = \text{argmax}_y \, \widehat{p}(y|x)$ and $\widehat{p}(x) = \max_y \widehat{p}(y \mid x)$.

- Log Likelihood - on test set: $-\sum_{(x,y) \in \mathcal{T}} \log \widehat{p}(y \mid x)$

Calibration metrics

classification task with $C \geq 2$ possible classes. For a sample x, the quantity $\widehat{p}(y|x)$ is a probabilistic prediction, $y \in \{1, \ldots, C\}$.

Set $\widehat{y}(x) = \text{argmax}_y \, \widehat{p}(y|x)$ and $\widehat{p}(x) = \max_y \widehat{p}(y \mid x)$.

- Brier Score (BS):

$$\frac{1}{|\mathcal{T}|} \sum_{(x,y) \in \mathcal{T}} \left(\mathbb{1}_{\{y = \widehat{y}(x)\}} - \widehat{p}(x) \right)^2$$

Calibration metrics

classification task with $C \geq 2$ possible classes. For a sample x, the quantity $\widehat{p}(y|x)$ is a probabilistic prediction, $y \in \{1, \ldots, C\}$.

Set $\widehat{y}(x) = \text{argmax}_y \, \widehat{p}(y|x)$ and $\widehat{p}(x) = \max_y \widehat{p}(y \mid x)$.

- Expected Calibration Error: measures the discrepancy between prediction confidence and empirical accuracy. For a partition $0 = c_0 < \cdots < c_M = 1$ of the unit interval and training data $\mathcal{T}$ set $B_m = \{x \in \mathcal{T}, c_{m-1} < \widehat{p}(x) \leq c_m\}$.

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{|\mathcal{T}|} |\text{acc}_m - \text{conf}_m|$$

where

$$\text{acc}_m = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}_{\{\widehat{y}(x_i) = y_i\}} \quad \text{and} \quad \text{conf}_m = \frac{1}{|B_m|} \sum_{i \in B_m} \widehat{p}(x_i).$$

ECE

For a partition $0 = c_0 < \cdots < c_M = 1$ of the unit interval and training data $\mathcal{T}$ set $B_m = \{x \in \mathcal{T}, c_{m-1} < \widehat{p}(x) \leq c_m\}$.

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{|\mathcal{T}|} |\text{acc}_m - \text{conf}_m|$$

where

$$\text{acc}_m = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}_{\{\widehat{y}(x_i) = y_i\}} \quad \text{and} \quad \text{conf}_m = \frac{1}{|B_m|} \sum_{i \in B_m} \widehat{p}(x_i).$$

- Perfect calibration: $\text{acc}_m = \text{conf}_m$.
- reliability curve: $\text{acc}_m - \text{conf}_m$ versus conf_m .
- Perfect calibrated model:flat reliability curve. For under-confident (resp. over-confident) model reliability curves prominently lies above (resp. below) the flat line $\text{acc}_m - \text{conf}_m = 0$

Bayesian approach

Bayesian inference in AI promises

- improved predictions,
- reliable uncertainty estimates,
- principled model comparison,

naturally supporting active learning, continual learning, and decision-making under uncertainty.

- The Bayesian Deep Learning community - with many different flavors !- is very active (10^5 results on google scholar on "Bayesian deep learning" !)
- Many applications range from astrophysics to automatic diagnosis of diabetic retinopathy, advertising click-through rate prediction, fluid dynamics modeling, active learning... and much more

Bayesian approach

In Bayesian modeling, the model uncertainty is formalized as a probability distribution over the model parameters θ, while the data uncertainty is formalized as a probability distribution over the model outputs y^*, given a parameterized model f_θ.

- Training set $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^n$
- **Statistical model** $\mathscr{F} = \{(x, y) \to p(y|x;\theta) : \theta \in \mathbb{R}^d\}$
- **Prior** $p(\theta)$: does not take any information but the general knowledge on θ into account.
- **Likelihood** on $\mathscr{D}$ distribution predicted by a model parameterized with θ.
  - $p(\mathscr{D}|\theta) = \prod_{(x,y)\in\mathscr{D}} p(y|x,\theta)$
- posterior distribution - Bayes theorem

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

Bayesian approach

- In "classical ML", a single best setting of the parameters is chosen by minimizing training error + regularization.
- In "Bayesian ML", the method infers a posterior distribution.
- Model prediction is computed by Bayesian model averaging:

$$\underbrace{p(\mathrm{y}|x,\mathscr{D})}_{\text{posterior predictive distribution}} = \int_{\theta} \underbrace{p(\mathrm{y}|x,\theta)}_{\text{prediction}} \underbrace{p(\theta|\mathscr{D})}_{\text{posterior}} \mathrm{d}\theta$$

Bayesian Model Average

- BMA is compelling in Bayesian deep learning because the posterior over the parameters for a DNN capture many complementary solutions corresponding to different parameter (same spirit: deep ensembles, SWAG, MC dropout).
- BMA for neural networks cannot be evaluated in closed form, so one must resort to approximate inference.
- BMA approximation is challenging due to high-dimensional and demanding posterior: multimodal posterior, unusual model structure... Need also to take into account computational and memory constraints.
- Many unresolved questions on BMA !

Other solutions

- Variational Approximations provide Gaussian approximations to the posterior (multimodality ?).
- Laplace approximation or SWA-Gaussian (SWAG) - which computes the first moment of stochastic gradient descent (SGD) iterates with a modified learning rate schedule -,
- Successful methods such as MC-dropout or deep ensembles provides ensemble but have no "natural" Bayesian interpretation.

Unadjusted Langevin dynamics

- **Existence** & **uniqueness**:
  - $\nabla \log p(\cdot|\mathscr{D})$ locally Lipschitz
  - $\langle \nabla \log p(\theta|\mathscr{D}) - \nabla \log p(\vartheta|\mathscr{D}), \theta - \vartheta \rangle \geq \rho \|\theta - \vartheta\|^2 - b$

$$\theta_t \sim p(\cdot|\mathscr{D}) \quad \boxed{\leftrightarrow} \quad \mathrm{d}\theta_t = -\nabla \log p(\theta_t|\mathscr{D})\mathrm{d}t + \sqrt{2}\mathrm{d}B_t$$

$\Longrightarrow p(\cdot|\mathscr{D})$ is the unique invariant probability measure

Unadjusted Langevin dynamics

- Discretized Langevin diffusion.
  - Euler-Maruyama scheme

$$\theta_{k+1} = \theta_k - \gamma \nabla \log p(\theta_k | \mathcal{D}) + \sqrt{2\gamma}\mathcal{N}(0, \mathrm{I}_d)$$

  - ⚠Gradient Descent (Gd)

$$\theta_{k+1} = \theta_k - \gamma \nabla \log p(\theta_k | \mathcal{D})$$

- Unadjusted Langevin dynamics (ULA) is biased because the Metropolis-Hastings correction step is omitted.

Some properties of Langevin dynamics

Assumptions.

- $\log p(\cdot|\mathcal{D}) \in \mathscr{C}_2(\mathbb{R}^d)$
- $\log p(\cdot|\mathcal{D})$ m-strongly convex
- $\nabla \log p(\cdot|\mathcal{D})$ L-smooth

Theoretical guarantee.

$$W_2{}^2\left(\mathscr{L}(\theta_k), p(\cdot|\mathcal{D})\right) \leq \underbrace{(1-\gamma m)^k \left\{\|\theta_0 - \theta_\star\|^2 + \frac{2d}{m}\right\}}_{\text{Variance } k \to \infty} + \underbrace{\mathrm{O}(\gamma)}_{\text{Bias } \gamma \to 0}$$

How to choose the parameters? $\Rightarrow$ **Mixing-time**

$$\text{Find } \mathsf{k}_\star, \qquad W_2\left(\mathscr{L}(\theta_{\mathsf{k}_\star}), p(\cdot|\mathcal{D})\right) \leq \varepsilon$$

Solve

$$(1-\gamma m)^{\mathsf{k}_\star} \left\{\|\theta_0 - \theta_\star\|^2 + \frac{2d}{m}\right\} \leq \varepsilon^2 \qquad \mathrm{O}(\gamma) \leq \varepsilon^2$$

$\Rightarrow$ implies the choices of $\mathsf{k}_\star$, γ

Some properties of Langevin dynamics

Assumptions.

- $\log p(\cdot|\mathcal{D}) \in \mathscr{C}_2(\mathbb{R}^d)$
- $\log p(\cdot|\mathcal{D})$ m-strongly convex
- $\nabla \log p(\cdot|\mathcal{D})$ L-smooth

Theoretical guarantee.

$$W_2{}^2\left(\mathscr{L}(\theta_k), p(\cdot|\mathcal{D})\right) \leq \underbrace{(1-\gamma m)^k \left\{\|\theta_0 - \theta_\star\|^2 + \frac{2d}{m}\right\}}_{\text{Variance } k \to \infty} + \underbrace{\mathrm{O}(\gamma)}_{\text{Bias } \gamma \to 0}$$

How to choose the parameters? $\Rightarrow$ **Mixing-time**

$$\text{Find } \mathsf{k}_\star, \qquad W_2\left(\mathscr{L}(\theta_{\mathsf{k}_\star}), p(\cdot|\mathcal{D})\right) \leq \varepsilon$$

Solve

$$(1-\gamma m)^{\mathsf{k}_\star} \left\{\|\theta_0 - \theta_\star\|^2 + \frac{2d}{m}\right\} \leq \varepsilon^2 \qquad \mathrm{O}(\gamma) \leq \varepsilon^2$$

$\Rightarrow$ implies the choices of $\mathsf{k}_\star$, γ

Some properties of Langevin dynamics

Assumptions.

- $\log p(\cdot|\mathcal{D}) \in \mathscr{C}_2(\mathbb{R}^d)$
- $\log p(\cdot|\mathcal{D})$ m-strongly convex
- $\nabla \log p(\cdot|\mathcal{D})$ L-smooth

Theoretical guarantee.

$$W_2{}^2\left(\mathscr{L}(\theta_k), p(\cdot|\mathcal{D})\right) \leq \underbrace{(1-\gamma m)^k \left\{\|\theta_0 - \theta_\star\|^2 + \frac{2d}{m}\right\}}_{\text{Variance } k \to \infty} + \underbrace{\mathrm{O}(\gamma)}_{\text{Bias } \gamma \to 0}$$

How to choose the parameters? $\Rightarrow$ **Mixing-time**

$$\text{Find } \mathsf{k}_\star, \qquad W_2\left(\mathscr{L}(\theta_{\mathsf{k}_\star}), p(\cdot|\mathcal{D})\right) \leq \varepsilon$$

Solve

$$(1-\gamma m)^{\mathsf{k}_\star} \left\{\|\theta_0 - \theta_\star\|^2 + \frac{2d}{m}\right\} \leq \varepsilon^2 \qquad \mathrm{O}(\gamma) \leq \varepsilon^2$$

$\Rightarrow$ implies the choices of $\mathsf{k}_\star$, γ

Stochastic Gradient Langevin Dynamics

- Computational bottleneck the complexity of the gradient evaluation $\nabla \log p(\theta_k|\mathscr{D})$ scales proportionally to the number of observations, unfeasible in the "Big Data" limit.
- Welling and Teh (2011) replaced the "full" gradient with a SG estimation on minibatches. This algorithm, Stochastic Gradient Langevin dynamics (SGLD), has become an important MCMC algorithm in Bayesian inference for large data sets.

Stochastic Gradient Langevin Dynamics

- The analysis of SGLD and its finite sample performance has attracted a wealth of contributions Ma et al. (2015); Teh et al. (2016); Brosse et al. (2018)
- These works show that the use of the stochastic gradient has its price: The resulting estimate of the gradient is still unbiased,but its variance could negate the computational advantages of SGLD Belomestny et al. (2021)

Stochastic Gradient Langevin Dynamics

- Several proposals have been made to reduce the variance of the stochastic gradient estimate of the "full" gradient, inspired methods used in stochastic optimization
  - Stochastic Average Gradient (SAG and SAGA)
  - Stochastic Variance Reduced Gradient (SVRG)
  - SPIDER methods
- Other variance reduction approaches include various subsampling schemes and constructing alternative estimates for the gradient (see, for instance, Baker et al. (2019))

Distributed / Federated

The surge of massive data has led to significant interest in distributed algorithms for scaling computations in the context of machine learning and optimization

- **Several** workers
- Local **datasets** $\mathscr{D}_i$
- Local **loss** $\mathrm{Loss_i}(\theta, \mathscr{D}_i)$

Distributed / Federated context

- Data is split among agents / workers.
- Distributed: data are typically centralized and "distributed" among workers. Federated: data are collected by agents, which do not share + communication constraints

Single machine:

$$\text{Loss}(\theta, \mathscr{D}) = \sum_{(x,y)\in\mathscr{D}} \ell(\text{model}_\theta(x), y)$$

Multiple machines:

$$\text{Loss}(\theta, \mathscr{D}) = \sum_{i=1}^{b} \text{Loss}_i(\theta, \mathscr{D}_i)$$

- Local posterior distribution

$$\rightsquigarrow p_i(\theta \mid \mathscr{D}_i) \propto \exp\{-\text{Loss}_i(\theta, \mathscr{D}_i)\}$$

Objective:

Sample parameter in large dimension

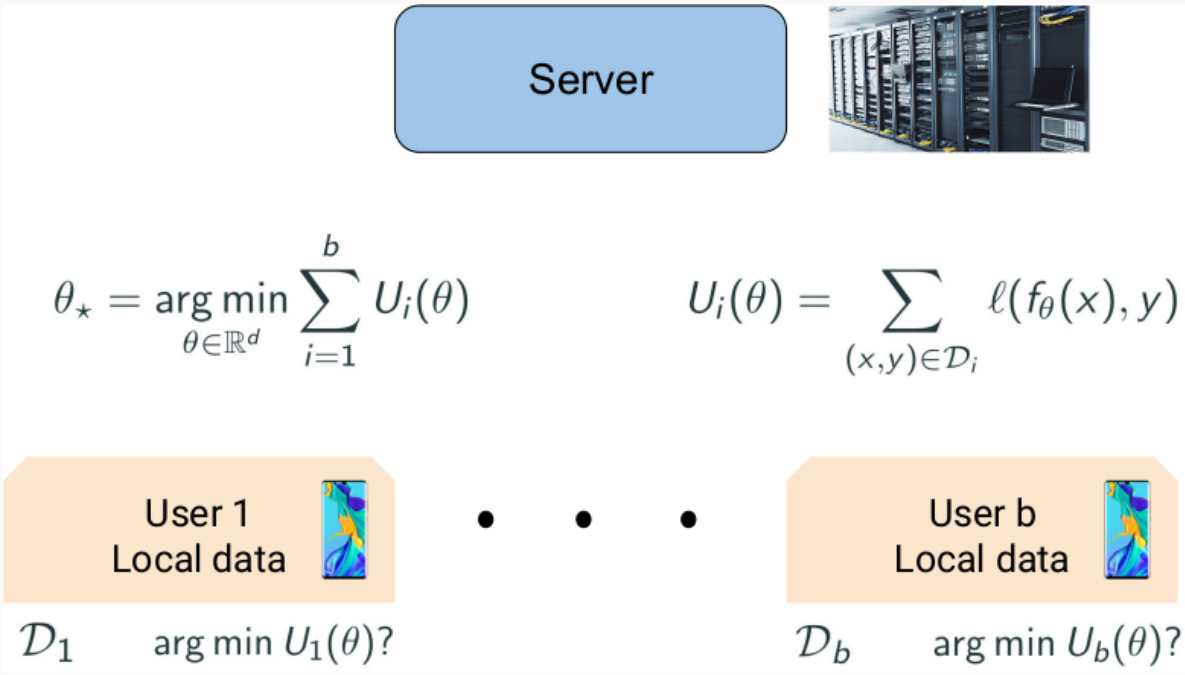
$$\theta_k \sim \underbrace{p(\theta|\mathscr{D})}_{\text{posterior}} \propto \prod_{i=1}^{b} \underbrace{p_i(\theta|\mathscr{D}_i)}_{\text{local posterior}} \propto \underbrace{\exp\{-\text{Loss}(\theta, \mathscr{D})\}}_{\text{like with a single machine}}$$

Qlsd: Quantized Langevin Dynamics for Bayesian federated learning

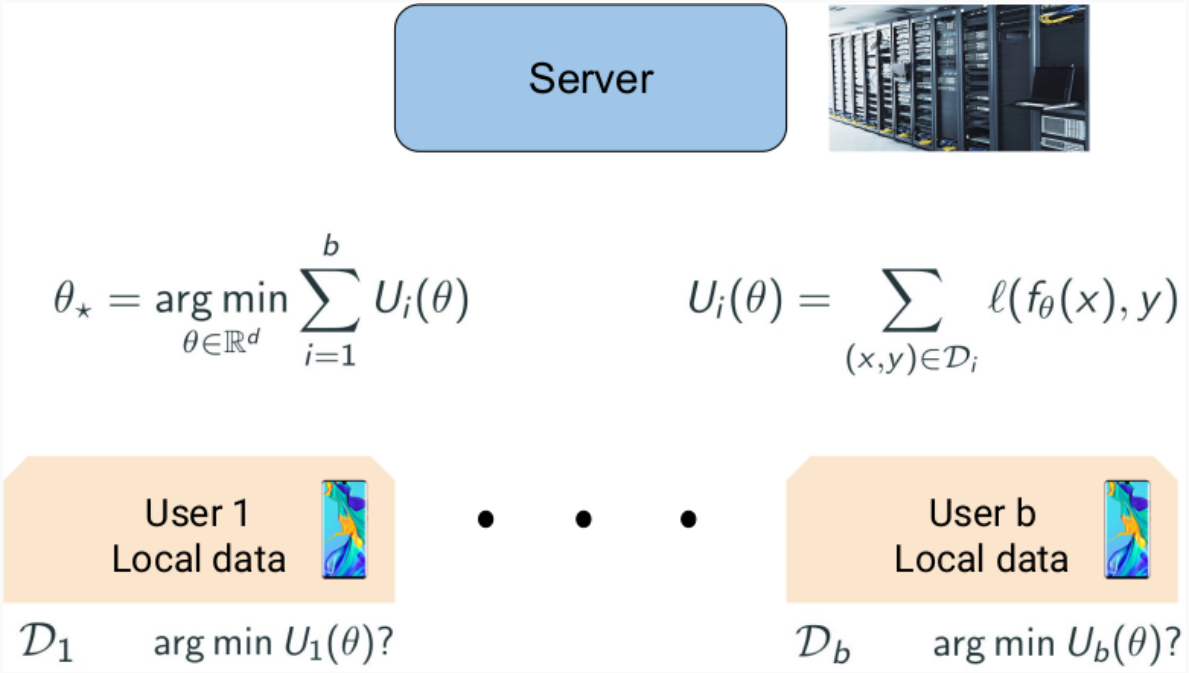
Federated Learning

- **Collaborate** to learn a parameter
- **Constraints**:
  - **Data privacy**
  - **Communication constraints**
  - **Different data distribution** on each client

How it works

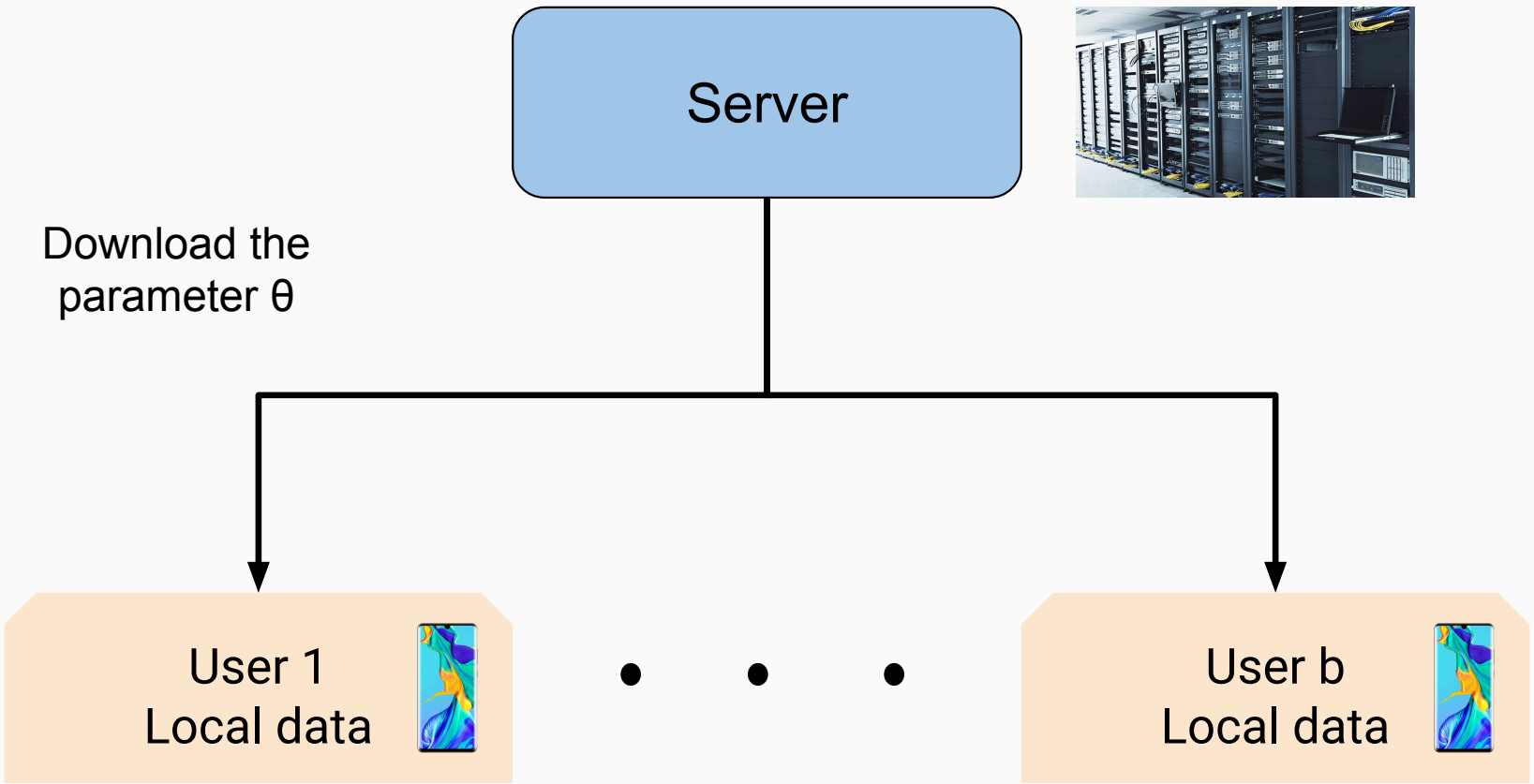

Server
$\theta_\star = \arg\min_{\theta \in \mathbb{R}^d} \sum_{i=1}^{b} U_i(\theta)$
$U_i(\theta) = \sum_{(x,y)\in\mathcal{D}_i} \ell(f_\theta(x), y)$
User 1
Local data
User b
Local data
$\mathcal{D}_1$
$\arg\min U_1(\theta)$?
$\mathcal{D}_b$
$\arg\min U_b(\theta)$?

How it works

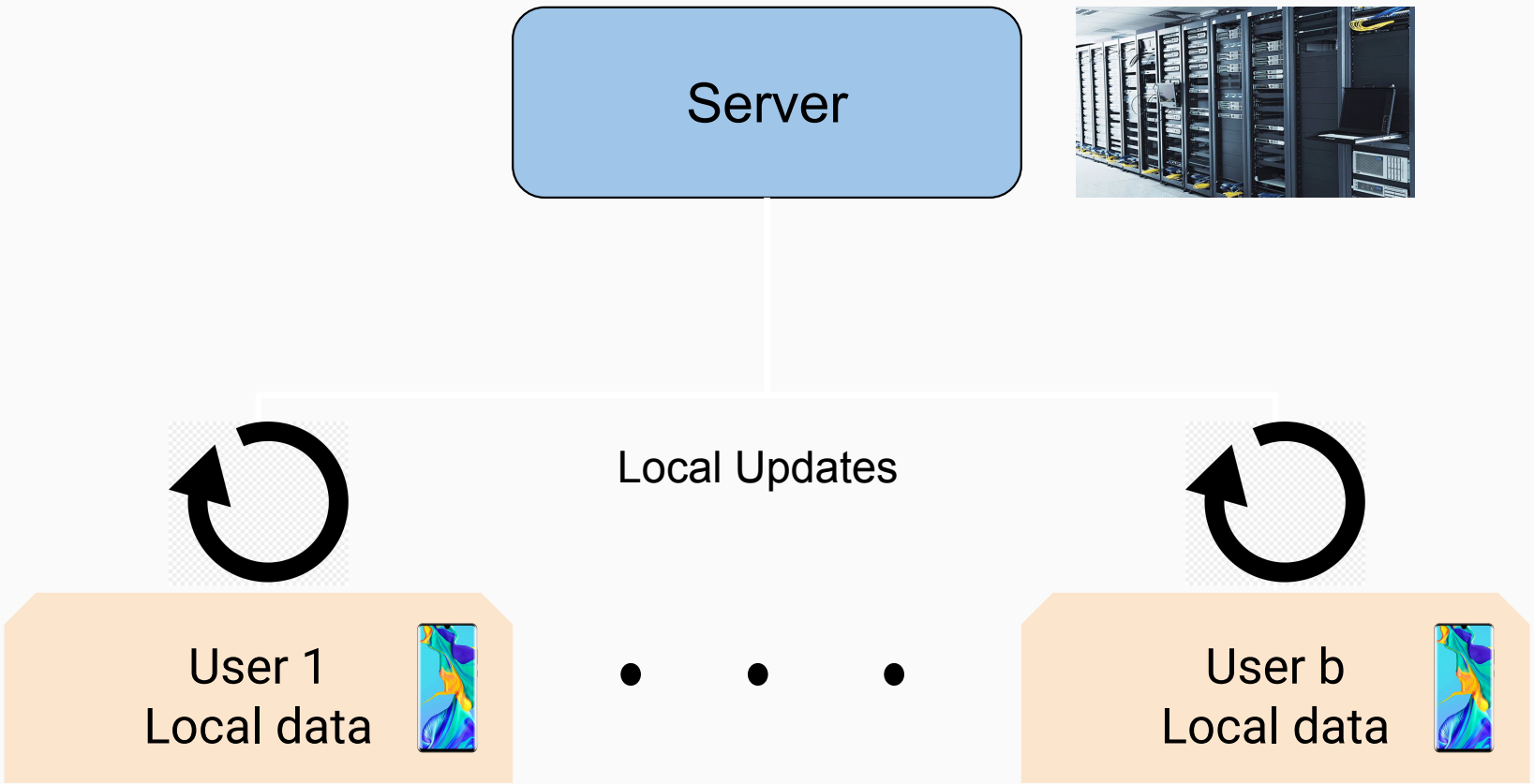


- What is the difference with distributed computing?
  - **Statistical heterogeneity**, **Unbalanced** data
  - Massively distributed data
  - Communication bottleneck
  - Partial participation)

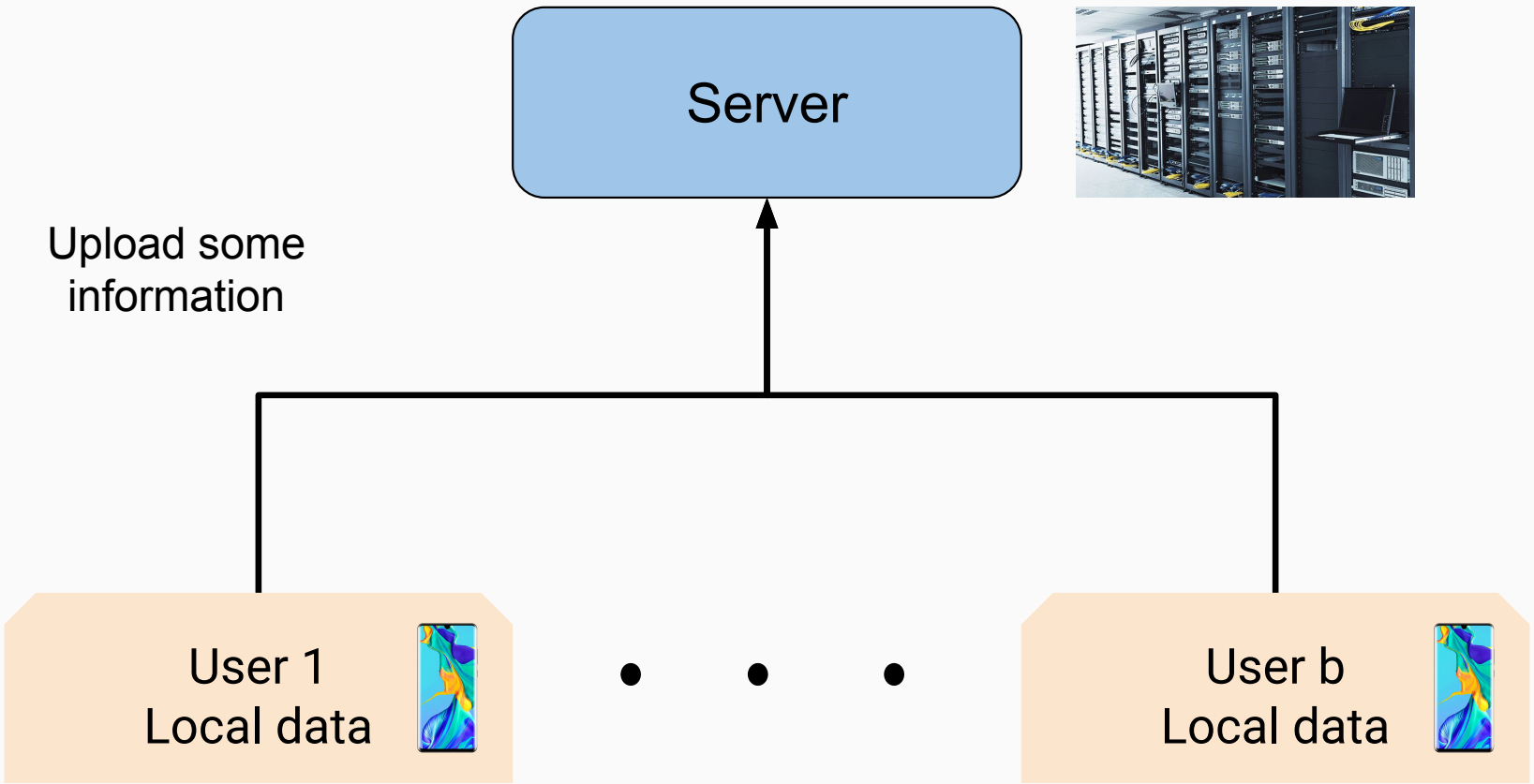
How it works


Server
Download the parameter θ
User 1
Local data
User b
Local data

How it works



How it works


Server
Upload some information
User 1
Local data
User b
Local data

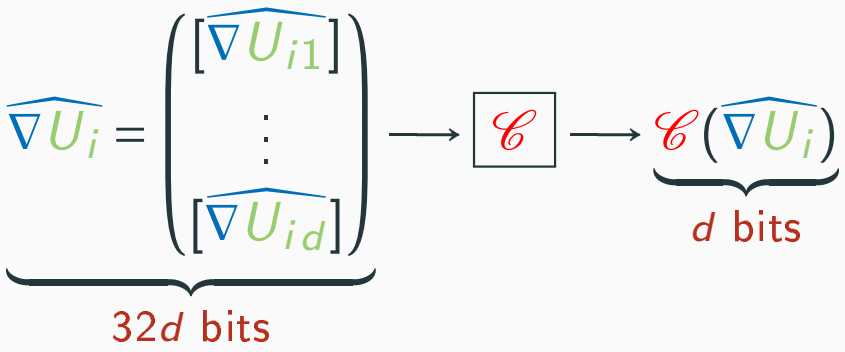
Classical algorithms

Method	FedAvg	Qsgd
Article	McMahan et al. (2017)	Alistarh et al. (2017)
Num local iter	$\mathbb{N}^*$	1
Compression	No	Yes

Philosphy.

- FedAvg: workers compute parameters (several local steps), the supervisor implements a global parameter update
- Qsgd: One popular way to reduce this cost has been to perform lossy compression of the gradients. The workers compute gradients, compressed gradients are transmitted to the supervisor which performs parameter update.

(Random) Compression operator



Assumptions. The exists $\omega > 0$, such that for all $x \in \mathbb{R}^d$,

$$\begin{cases} \mathbb{E}[\mathscr{C}(x)] = x \\ \mathbb{E}\left[\|\mathscr{C}(x) - x\|^2\right] \leq \omega \|x\|^2 \end{cases}$$

Stochastic scalar quantization

For any $x \in \mathbb{R}^d$ with $x \neq 0$, $\mathscr{C}(x)$ is defined as

$$\mathscr{C}(x_i) = \|x\|_2 \cdot \operatorname{sign}(x_i) \cdot \xi_i(x, s),$$

where $\xi_i(x, s)$'s are independent random variables defined as follows. Let $0 \leq \ell < s$ be an integer such that $|x_i| / \|x\|_2 \in [\ell/s, (\ell+1)/s]$. That is, $[\ell/s, (\ell+1)/s]$ is the quantization interval corresponding to $|x_i| / \|x\|_2$. Then

$$\xi_i(x, s) = \begin{cases} \ell/s & \text{with probability } 1 - p(|x_i|/\|x\|_2, s) \\ (\ell+1)/s & \text{otherwise.} \end{cases}$$

where, for any $a \in [0, 1]$,

$$p(a, s) = as - \ell.$$

If $x = 0$, then we define $Q(\boldsymbol{v}, s) = 0$.

QSGD

Algorithm 1: Qsgd

Initialize $\theta_0 \in \mathbb{R}^d$

for $k = 0$ ***to*** $K-1$ **do**

  // In parallel on the b clients

  for $i \in \{1, \dots, b\}$ **do**

    Send $\mathscr{C}\left(\widehat{\nabla U_i(\theta_k)}\right)$

  // On the central server

  Set $\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^n \mathscr{C}\left(\widehat{\nabla U_i(\theta_k)}\right)$

Output: θ_K

⚠ Drawback: Because of heterogeneity, $\nabla U_i(\theta_k) \nrightarrow 0$ the norm of the compressed gradient can be large $\Longrightarrow$ slow convergence

Solution: Use a memory term in the compression Horváth et al. (2019)

Memory term

- Each device keeps a local memory term $\eta_k^{(i)} \in \mathbb{R}^d$
- Upload $\mathscr{C}\left(\underbrace{\widehat{\nabla U_i(\theta_k)} - \eta_k^{(i)}}_{\text{tends to zero}}\right)$ instead of $\mathscr{C}\left(\underbrace{\widehat{\nabla U_i(\theta_k)}}_{\neq 0 \text{ due to heterogeneity}}\right)$
- Since $\mathbb{E}\left[\|\mathscr{C}(x) - x\|^2\right] \leq \omega \|x\|^2 \Rightarrow$ transfer $\|x\| \ll 1$ to reduce quantization error

In practice

Convergence results:

Client	$\eta_k^{(i)} \to \nabla U_i(\theta_\star)$
Server	$\eta_k \to 0$

Algorithm. **Update the memory term at each iteration**

- On the clients

$$\text{Send } \mathscr{C}\left(\widehat{\nabla U_i(\theta_k)} - \eta_k^{(i)}\right)$$

$$\eta_{k+1}^{(i)} = \eta_k^{(i)} + \alpha \mathscr{C}\left(\widehat{\nabla U_i(\theta_k)} - \eta_k^{(i)}\right)$$

- On the central server

$$\text{Update } \theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{n} \mathscr{C}\left(\widehat{\nabla U_i(\theta_k)} - \eta_k^{(i)}\right) - \gamma \eta_k$$

$$\eta_{k+1} = \eta_k + \alpha \sum_{i=1}^{n} \mathscr{C}\left(\widehat{\nabla U_i(\theta_k)} - \eta_k^{(i)}\right)$$

- Closed loop adaptation: no need to transmit local memory terms

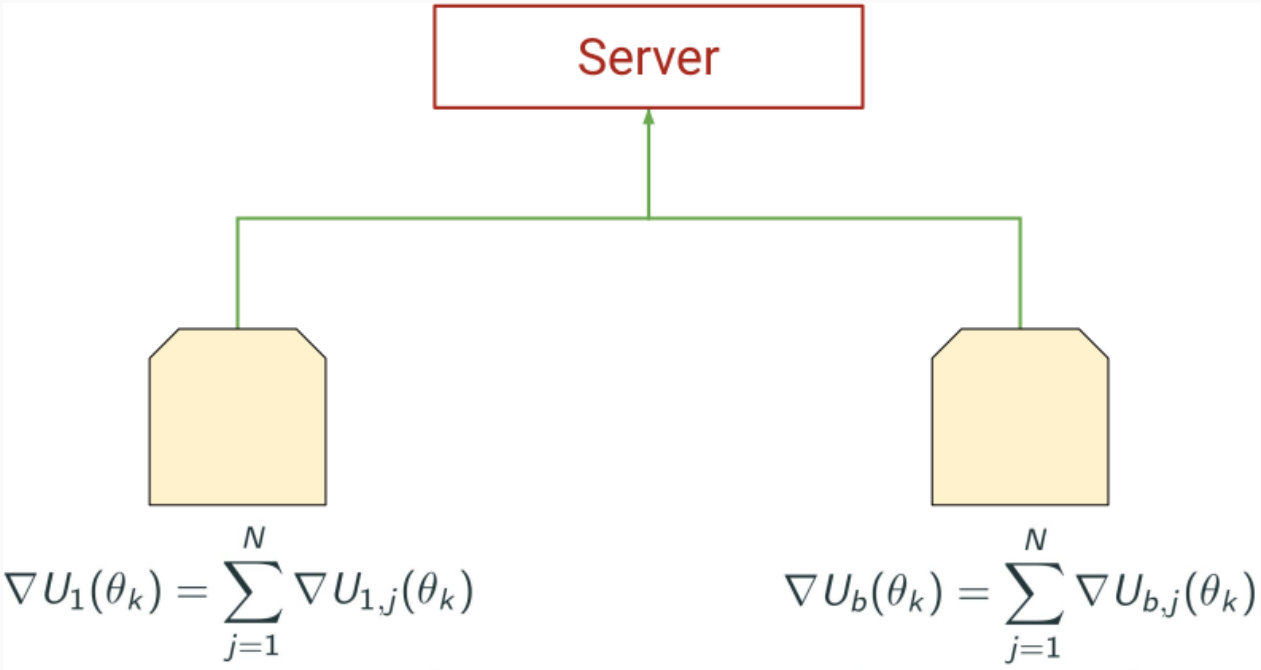
First possibility

- Lsd$^{\#}$

$$\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{n} \nabla U_i(\theta_k) + \sqrt{2\gamma}\mathcal{N}(0, \mathrm{I}_d)$$

- Deep Learning scenario

$$U_i = \sum_{j \in \mathcal{D}_i} U_{i,j} \qquad U_{i,j}(\theta_k) = \ell(f_{\theta_k}(x_j), y_j)$$

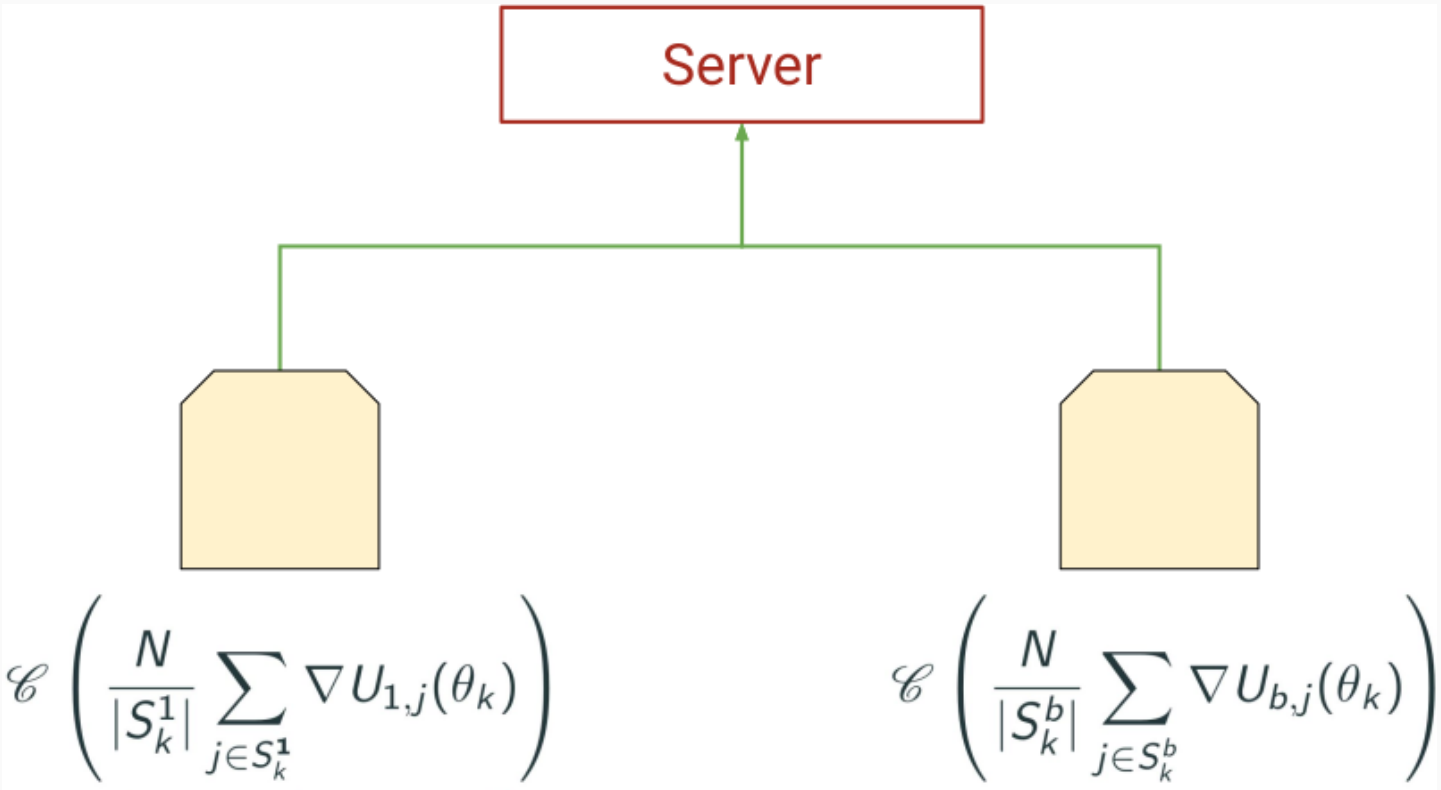


- LSD$^{\#}$
  - ↪ Computational cost? ☹
- LSD$^{\#}$ + Mini-batch
  - ↪ Computational cost? ☺
  - ↪ Communication constraint? ☹
- LSD$^{\#}$ + Mini-batch + Compression
  - ↪ Computational cost? ☺
  - ↪ Communication constraint? ☺

Qlsd

Qlsd$^{\#}$ = *Quantized Langevin Stochastic Dynamics*$^{\#}$

$$\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{b} \mathscr{C}\left(\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \nabla U_{i,j}(\theta_k) \right) + \sqrt{2\gamma}\,\mathcal{N}(0, \mathrm{I}_d)$$



Algorithm

Algorithm 2: Qlsd$^{\#}$

Initialize $\theta_0 \in \mathbb{R}^d$

Algorithm

Algorithm 2: Qlsd$^{\#}$

Initialize $\theta_0 \in \mathbb{R}^d$

for $k = 0$ ***to*** $K-1$ **do**

  for $i \in \{1, \dots, b\}$ *// In parallel on the b clients* **do**

    Set $\boxed{g_k^i = \mathscr{C}\left(\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \nabla U_{i,j}(\theta_k)\right)}$

    Send g_k^i to the server

Algorithm

Algorithm 2: Qlsd$^{\#}$

Initialize $\theta_0 \in \mathbb{R}^d$

for $k = 0$ ***to*** $K-1$ **do**

  for $i \in \{1,\dots,b\}$ *// In parallel on the b clients* **do**

    Set $g_k^i = \mathscr{C}\left(\frac{N}{|S_k^i|}\sum_{j\in S_k^i}\nabla U_{i,j}(\theta_k)\right)$

    Send g_k^i to the server

  // On the central server

  Set $\boxed{\theta_{k+1} = \theta_k - \gamma\sum_{i=1}^b g_k^i + \sqrt{2\gamma}\mathcal{N}(0, I_d)}$

Output: $(\theta_k)_{k=0}^K$

Algorithm

Algorithm 2: Qlsd$^{\#}$

Initialize $\theta_0 \in \mathbb{R}^d$

for $k = 0$ ***to*** $K-1$ **do**

  for $i \in \{1,\dots,b\}$ *// In parallel on the b clients* **do**

    Set $g_k^i = \mathscr{C}\left(\frac{N}{|S_k^i|}\sum_{j\in S_k^i}\nabla U_{i,j}(\theta_k)\right)$

    Send g_k^i to the server

  // On the central server

  Set $\theta_{k+1} = \theta_k - \gamma\sum_{i=1}^b g_k^i + \sqrt{2\gamma}\mathcal{N}(0,\mathrm{I}_d)$

Output: $(\theta_k)_{k=0}^K$

Algorithm 3: Qsgd

Initialize $\theta_0 \in \mathbb{R}^d$

for $k = 0$ ***to*** $K-1$ **do**

  for $i \in \{1,\dots,b\}$ **do**

    Set $g_k^i = \mathscr{C}\left(\frac{N}{|S_k^i|}\sum_{j\in S_k^i}\nabla U_{i,j}(\theta_k)\right)$

  Set $\theta_{k+1} = \theta_k - \gamma\sum_{i=1}^b g_k^i$

Output: θ_K

Theoretical result

Assumptions.

- The potential U is $\mathtt{m}$- strongly convex , $\mathtt{L}$- Lipschitz
- The compression $\mathscr{C}$ is unbiased and $\mathbb{E}\left[\|\mathscr{C}(x)-x\|^2\right] \leq \omega\|x\|^2$
- There exists $\widetilde{\mathtt{m}} \geq 0$,
  $\|\nabla U_{i,j}(\theta_2)-\nabla U_{i,j}(\theta_1)\|^2 \leq \widetilde{\mathtt{m}}\langle \nabla U_{i,j}(\theta_2)-\nabla U_{i,j}(\theta_1), \theta_2-\theta_1\rangle$

Results

- $\exists \psi > 0,\ \forall \gamma < \psi,\ \exists A_\gamma^{\#}, B_\gamma^{\#} > 0$
- $\forall \mathscr{L}(\theta_0) \in \mathscr{P}_2\left(\mathbb{R}^d\right)$

Theoretical result

Assumptions.

- The potential U is $\mathtt{m}$- strongly convex , $\mathtt{L}$- Lipschitz
- The compression $\mathscr{C}$ is unbiased and $\mathbb{E}\left[\|\mathscr{C}(x)-x\|^2\right] \leq \omega\|x\|^2$
- There exists $\widetilde{\mathtt{m}} \geq 0$,
  $\|\nabla U_{i,j}(\theta_2)-\nabla U_{i,j}(\theta_1)\|^2 \leq \widetilde{\mathtt{m}}\langle \nabla U_{i,j}(\theta_2)-\nabla U_{i,j}(\theta_1), \theta_2-\theta_1\rangle$

Results

- $\exists \psi > 0,\ \forall \gamma < \psi,\ \exists A_\gamma^\#, B_\gamma^\# > 0$
- $\forall \mathscr{L}(\theta_0) \in \mathscr{P}_2\left(\mathbb{R}^d\right)$

$$W_2{}^2\left(\mathscr{L}(\theta_k), p(\cdot|\mathscr{D})\right) \leq \underbrace{(1-\gamma \mathtt{m}/2)^k}_{\text{Contraction term}} \overbrace{W_2{}^2\left(\mathscr{L}(\theta_0), p(\cdot|\mathscr{D})\right)}^{\text{Distance between the initialization and the target}}$$
$$+\gamma B_\gamma^\# + \gamma^2 A_\gamma^\# (1-\mathrm{m}\gamma/2)^{k-1} k \int_{\mathbb{R}^d} \|\theta-\theta_\star\|^2 \mathscr{L}(\theta_0)(\mathrm{d}\theta)$$

Theoretical result

Assumptions.

- The potential U is $\mathtt{m}$- strongly convex , $\mathtt{L}$- Lipschitz
- The compression $\mathscr{C}$ is unbiased and $\mathbb{E}\left[\|\mathscr{C}(x)-x\|^2\right] \leq \omega\|x\|^2$
- There exists $\widetilde{\mathtt{m}} \geq 0$,
  $\|\nabla U_{i,j}(\theta_2)-\nabla U_{i,j}(\theta_1)\|^2 \leq \widetilde{\mathtt{m}}\langle \nabla U_{i,j}(\theta_2)-\nabla U_{i,j}(\theta_1), \theta_2-\theta_1\rangle$

Results

- $\exists \psi > 0,\ \forall \gamma < \psi,\ \exists A_\gamma^{\#}, B_\gamma^{\#} > 0$
- $\forall \mathscr{L}(\theta_0) \in \mathscr{P}_2\left(\mathbb{R}^d\right)$

Heterogeneity + Discretization error

$$W_2{}^2\left(\mathscr{L}(\theta_k), p(\cdot|\mathscr{D})\right) \leq (1-\gamma \mathtt{m}/2)^k W_2{}^2\left(\mathscr{L}(\theta_0), p(\cdot|\mathscr{D})\right) + \overbrace{\gamma B_\gamma^{\#}}$$
$$+\gamma^2 A_\gamma^{\#}(1-\mathtt{m}\gamma/2)^{k-1} k \int_{\mathbb{R}^d} \|\theta-\theta_\star\|^2 \mathscr{L}(\theta_0)(\mathrm{d}\theta)$$

Theoretical result

Assumptions.

- The potential U is $\mathtt{m}$- strongly convex , $\mathtt{L}$- Lipschitz
- The compression $\mathscr{C}$ is unbiased and $\mathbb{E}\left[\|\mathscr{C}(x)-x\|^2\right] \leq \omega\|x\|^2$
- There exists $\widetilde{\mathtt{m}} \geq 0$,
  $\|\nabla U_{i,j}(\theta_2)-\nabla U_{i,j}(\theta_1)\|^2 \leq \widetilde{\mathtt{m}}\langle \nabla U_{i,j}(\theta_2)-\nabla U_{i,j}(\theta_1), \theta_2-\theta_1\rangle$

Results

- $\exists \psi > 0,\ \forall \gamma < \psi,\ \exists A_\gamma^\#, B_\gamma^\# > 0$
- $\forall \mathscr{L}(\theta_0) \in \mathscr{P}_2\left(\mathbb{R}^d\right)$

$$W_2{}^2\left(\mathscr{L}(\theta_k), p(\cdot|\mathscr{D})\right) \leq (1-\gamma\mathtt{m}/2)^k W_2{}^2\left(\mathscr{L}(\theta_0), p(\cdot|\mathscr{D})\right) + \gamma B_\gamma^\# + \underbrace{\gamma^2 A_\gamma^\# (1-\mathtt{m}\gamma/2)^{k-1} k \int_{\mathbb{R}^d} \|\theta-\theta_\star\|^2 \mathscr{L}(\theta_0)(\mathrm{d}\theta)}_{\text{Mini-batch + Compression}}$$

Sketch of proof.

- Based on couplings

Initialization	$\nu_0 \sim p(\cdot\|\mathscr{D})$	$\theta_0 \sim \mathscr{L}(\theta_0)$
k-th iteration	$\nu_{k\gamma} \sim p(\cdot\|\mathscr{D})$	$\theta_k \sim \mathscr{L}(\theta_k)$

- Update

$$\begin{cases} \mathrm{d}\nu_t = -\nabla U(\nu_t)\mathrm{d}t + \sqrt{2}\mathrm{d}B_t \\ \theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{b} \mathscr{C}\left(\widehat{\nabla U_i}(\theta_k)\right) + \sqrt{2}(B_{\gamma(k+1)} - B_{\gamma k}) \end{cases}$$

- Main idea $\rightsquigarrow$ Find a contraction

$$\mathbb{E}^{\mathscr{F}_k}\left[\|\nu_{\gamma(k+1)} - \theta_{k+1}\|^2\right] \leq A\|\nu_{k\gamma} - \theta_k\|^2 + \gamma^2 B\|\theta_k - \theta_\star\|^2 + \gamma^2 C - \gamma D\left\langle \nu_{k\gamma} - \theta_k, \nabla U(\nu_{k\gamma}) - \nabla U(\theta_k)\right\rangle$$

- Wasserstein distance $\rightsquigarrow$ infimum over couplings between $\mathscr{L}(\theta_k)$ & $p(\cdot|\mathscr{D})$

$$W_2{}^2\left(\mathscr{L}(\theta_k), p(\cdot|\mathscr{D})\right) \leq \mathbb{E}\left[\|\nu_{\gamma k} - \theta_k\|^2\right]$$

Analysis

Drawback.

- When $\gamma \propto N^{-1}$

$$\liminf_{N\to\infty} \gamma B_\gamma^{\#} > 0$$

Solution: Variance-reduction scheme.

- **Fixed-point** based on the minimizer $\theta_\star = \arg\min U$

$$\widehat{\nabla U}_i(\theta) = \frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta) - \nabla U_{i,j}(\theta_\star)\}$$

- **Biased** operator

$$\mathbb{E}\left[\widehat{\nabla U}_i(\theta)\right] = \nabla U_i(\theta) - \nabla U_i(\theta_\star) \neq \nabla U_i(\theta)$$

- Qlsd*:

$$\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{b} \mathscr{C}\left(\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta_k) - \nabla U_{i,j}(\theta_\star)\} \right) + \sqrt{2\gamma}\mathcal{N}(0, \mathrm{I}_d)$$

Qlsd*

- Update scheme

$$\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{b} \mathscr{C}\left(\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta_k) - \nabla U_{i,j}(\theta_\star)\} \right) + \sqrt{2\gamma}\mathcal{N}(0, \mathrm{I}_d)$$

Heterogeneous datasets $\mathscr{D}_1, \ldots, \mathscr{D}_b$

- $\lim_{N\to\infty} \gamma B_\gamma^\star = 0$ when $\gamma \propto N^{-1} \to 0$
- $B_\gamma^\star$ does no longer depend on the heterogeneity !

Qlsd*

- Update scheme

$$\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{b} \mathscr{C}\left(\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta_k) - \nabla U_{i,j}(\theta_\star)\} \right) + \sqrt{2\gamma}\mathcal{N}(0, \mathrm{I}_d)$$

Drawback.

- Difficult estimation of $\theta_\star$, especially in a FL context 

Qlsd*

- Update scheme

$$\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{b} \mathscr{C}\left(\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta_k) - \nabla U_{i,j}(\theta_\star)\} \right) + \sqrt{2\gamma}\mathcal{N}(0, \mathrm{I}_d)$$

Drawback.

- Difficult estimation of $\theta_\star$, especially in a FL context ☹

Solution: Variance-reduction scheme without $\theta_\star$.

- Svrg : variance reduction ☺
- Memory Term: heterogeneity
- QLSD^{++}:

$$g_k^i = \underbrace{\mathscr{C}}_{\text{Compression}}\left(\left[\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta_k) - \nabla U_{i,j}(\zeta_k)\} + h_k^i - \eta_k^i \right] \right)$$

Qlsd*

- Update scheme

$$\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{b} \mathscr{C}\left(\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta_k) - \nabla U_{i,j}(\theta_\star)\} \right) + \sqrt{2\gamma}\mathcal{N}(0, \mathrm{I}_d)$$

Drawback.

- Difficult estimation of $\theta_\star$, especially in a FL context ☹

Solution: Variance-reduction scheme without $\theta_\star$.

- Svrg : variance reduction ☺
- Memory Term: heterogeneity
- QLSD^{++}:

$$g_k^i = \mathscr{C}\left(\left[\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \left\{ \nabla U_{i,j}(\theta_k) \underbrace{-\nabla U_{i,j}(\zeta_k)}_{\text{Control Variate}} \right\} \underbrace{+h_k^i}_{\text{Control Variate}} - \eta_k^i \right] \right)$$

Qlsd*

- Update scheme

$$\theta_{k+1} = \theta_k - \gamma \sum_{i=1}^{b} \mathscr{C}\left(\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta_k) - \nabla U_{i,j}(\theta_\star)\} \right) + \sqrt{2\gamma}\mathcal{N}(0, \mathrm{I}_d)$$

Drawback.

- Difficult estimation of $\theta_\star$, especially in a FL context ☹

Solution: Variance-reduction scheme without $\theta_\star$.

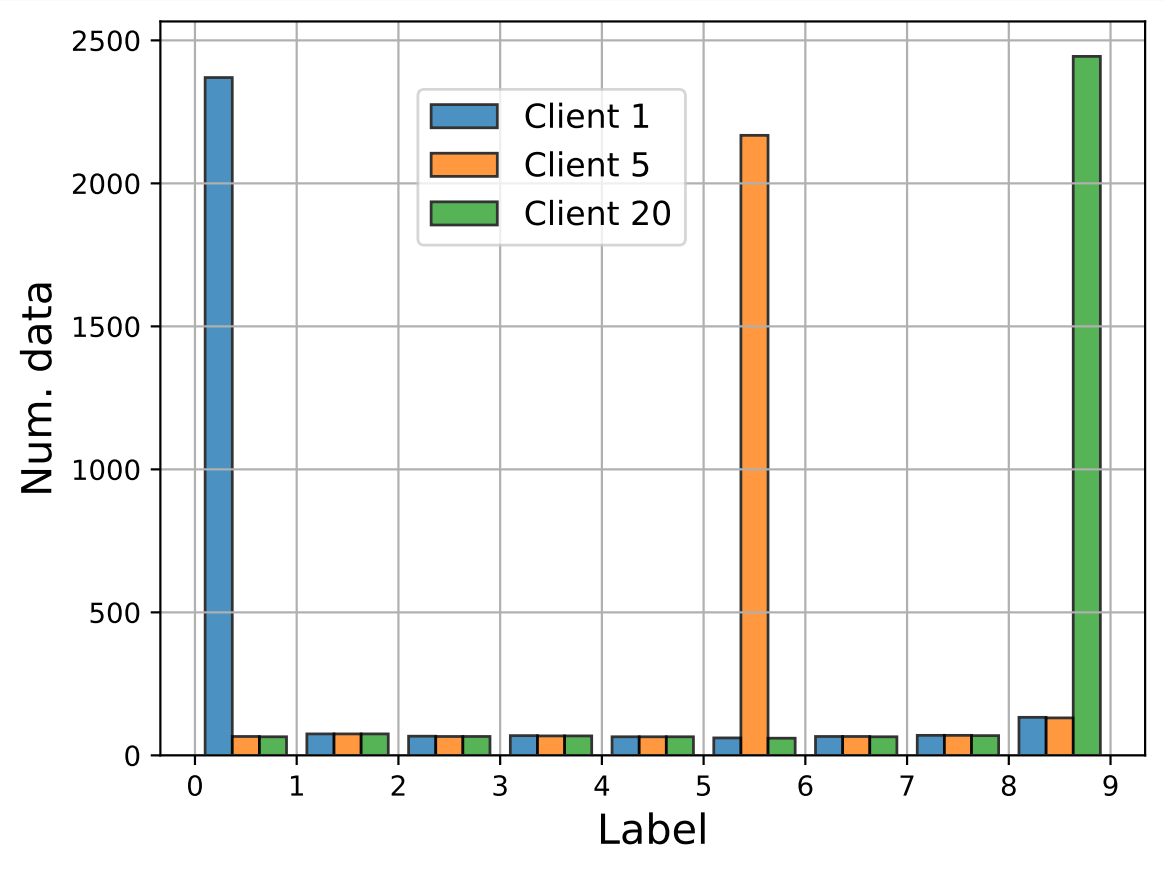
- Svrg : variance reduction ☺
- Memory Term: heterogeneity
- QLSD^{++}:

$$g_k^i = \mathscr{C}\left(\left[\frac{N}{|S_k^i|} \sum_{j \in S_k^i} \{\nabla U_{i,j}(\theta_k) - \nabla U_{i,j}(\zeta_k)\} + h_k^i \underbrace{- \eta_k^i}_{\text{Memory term}} \right] \right)$$

Summary

- 1 We proposed Qlsd$^{\#}$
- 2 Bias issue $\liminf_{N\to\infty} \gamma B_\gamma^{\#} > 0$ when $\gamma \propto N^{-1} \to 0$
- 3 $\Rightarrow$ Qlsd*: control variates using $\theta_\star = \arg\min U$.
- 4 $\hookrightarrow$ hard to compute
- 5 $\Rightarrow$ Qlsd^{++}: memory term $\to$ heterogeneity & control variates $\to$ fixes bias when $\gamma \propto N^{-1} \to 0$

Highly heterogeneous dataset



Method	SGLD	Qlsd	Qlsd PP	Qlsd^{++}	Qlsd^{++} PP	FedBe	FSGLD
Accuracy	99.1	98.8	98.3	98.8	98.7	43.5	98.5
$10^2\times$ ECE	0.577	0.916	1.57	0.692	0.930	7.51	2.65
$10^2\times$ BS	1.38	1.98	2.23	1.91	2.18	66.6	2.64
$10^2\times$ nNLL	2.86	4.15	4.82	4.11	4.65	139	6.19
Weight Decay	5	5	5	5	5	0	5
Batch Size	64	64	64	64	64	64	64
Learning rate	1e-07	1e-07	1e-07	1e-07	1e-07	1e-02	1e-07
Local steps	N/A	1	1	1	1	250	16
Burn-in	100epch.	1e04	1e04	1e04	1e04	N/A	1e04
Thinning	1	500	500	500	500	N/A	500
Training	1e03epch.	1e05it.	1e05it.	1e05it.	1e05it.	N/A	1e05it.

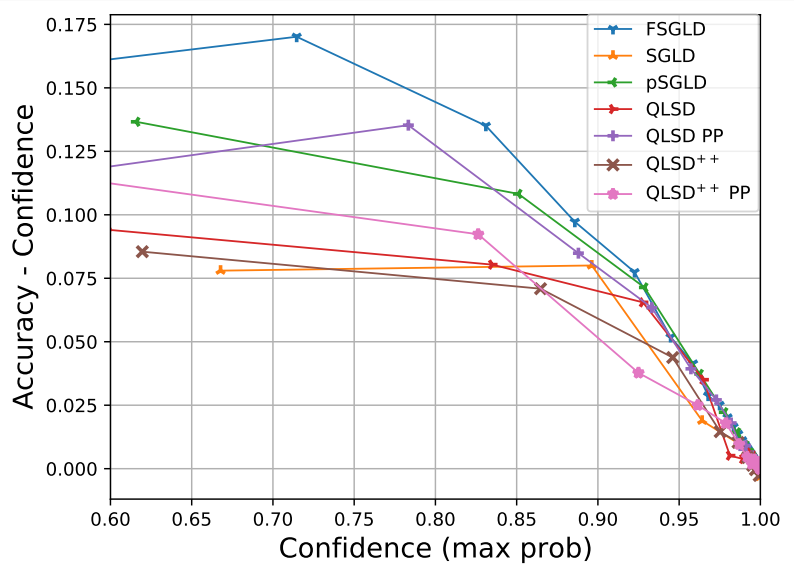
Calibrations scores.

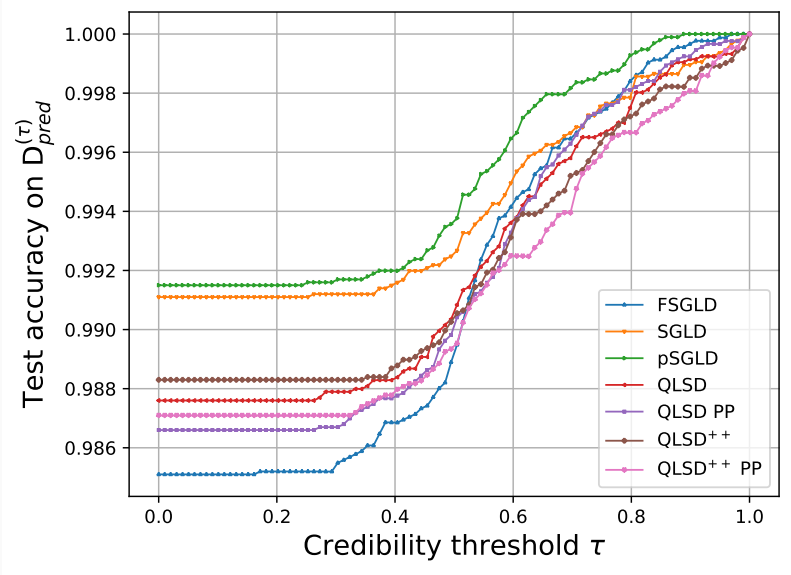
- Difference between the confidence level of a prediction and its accuracy

$$\mathrm{ECE} \approx \mathbb{E}_{(x,y)}\left[\left|\mathbb{P}_{(x',y')}\left(y_{\mathrm{pred}}(x') = y' \,\middle|\, p(y_{\mathrm{pred}'}(x')|x') = p(y_{\mathrm{pred}}(x)|x)\right) - p(y_{\mathrm{pred}}(x)|x)\right|\right].$$

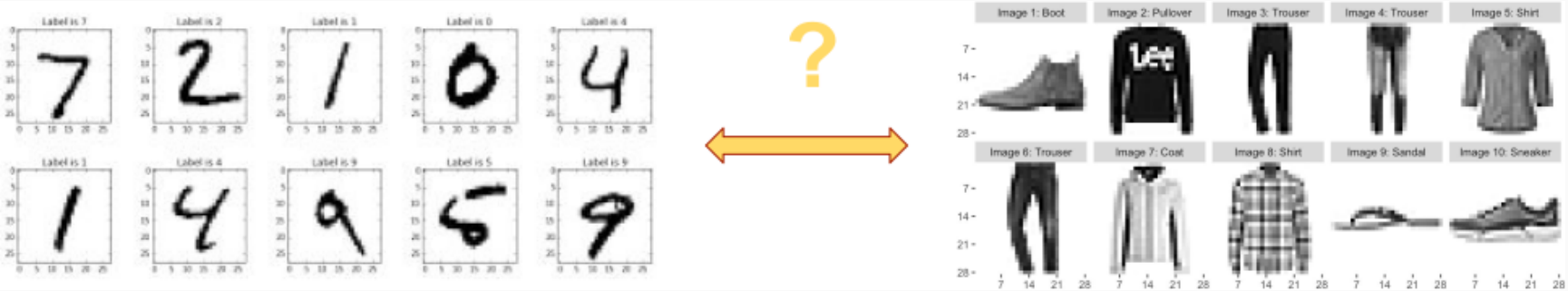
Calibration results.

- Good ECE ⇒ **Confidence** ∼ **Accuracy**

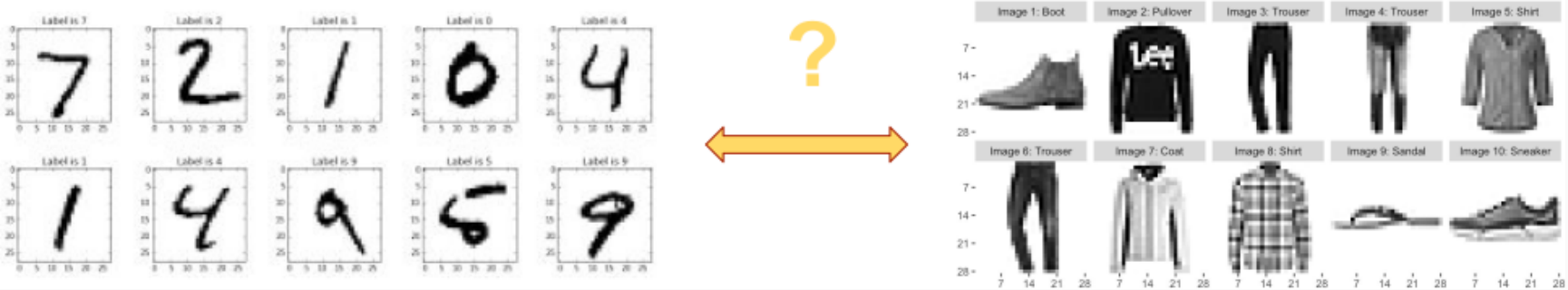


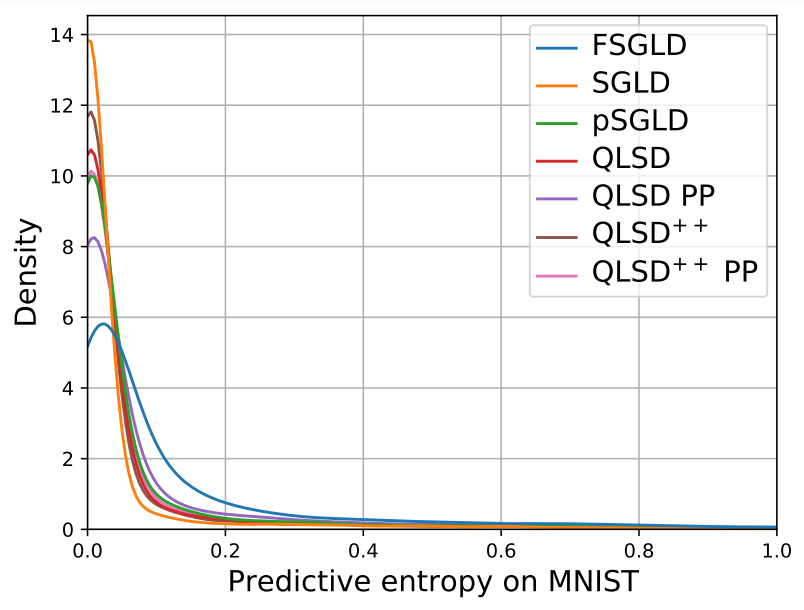


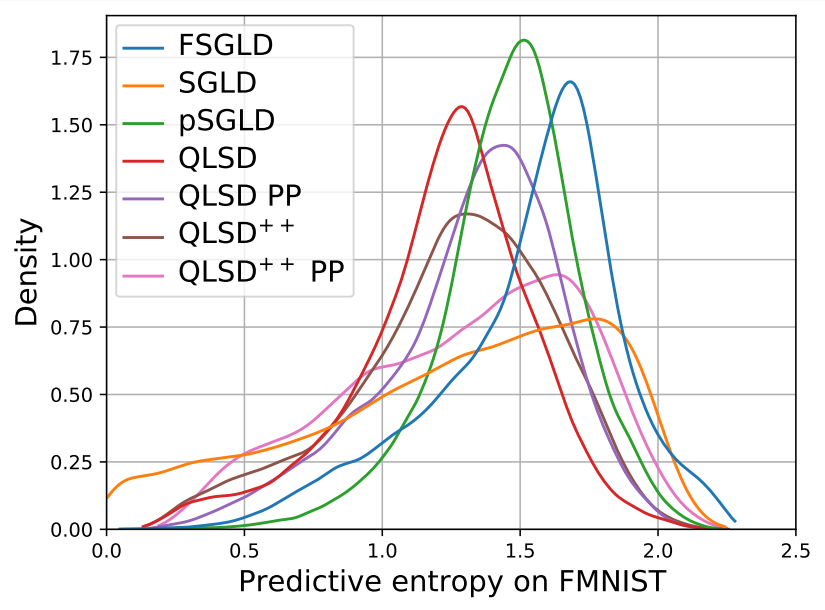
Out-Of-Distribution detection.



Out-Of-Distribution detection.







Dg-Lmc: A Turn-key and Scalable Synchronous Distributed MCMC Algorithm

Objective.

- Sample parameters $(\theta_k)_{k\in\mathbb{N}}$ in high dimension

$$\theta_k \sim p(\cdot \mid \mathscr{D}) \propto \prod_{i=1}^{b} p_i(\cdot \mid \mathscr{D}_i)$$

Objective.

- Sample parameters $(\theta_k)_{k\in\mathbb{N}}$ in high dimension

$$\theta_k \sim p(\cdot \mid \mathscr{D}) \propto \prod_{i=1}^{b} p_i(\cdot \mid \mathscr{D}_i)$$

→ target posterior distribution

$$p(\theta \mid \mathscr{D}) \propto \prod_{i=1}^{b} p_i(\theta \mid \mathscr{D}_i)$$

Objective.

- Sample parameters $(\theta_k)_{k\in\mathbb{N}}$ in high dimension

$$\theta_k \sim p(\cdot \mid \mathcal{D}) \propto \prod_{i=1}^{b} p_i(\cdot \mid \mathcal{D}_i)$$

→ target posterior distribution

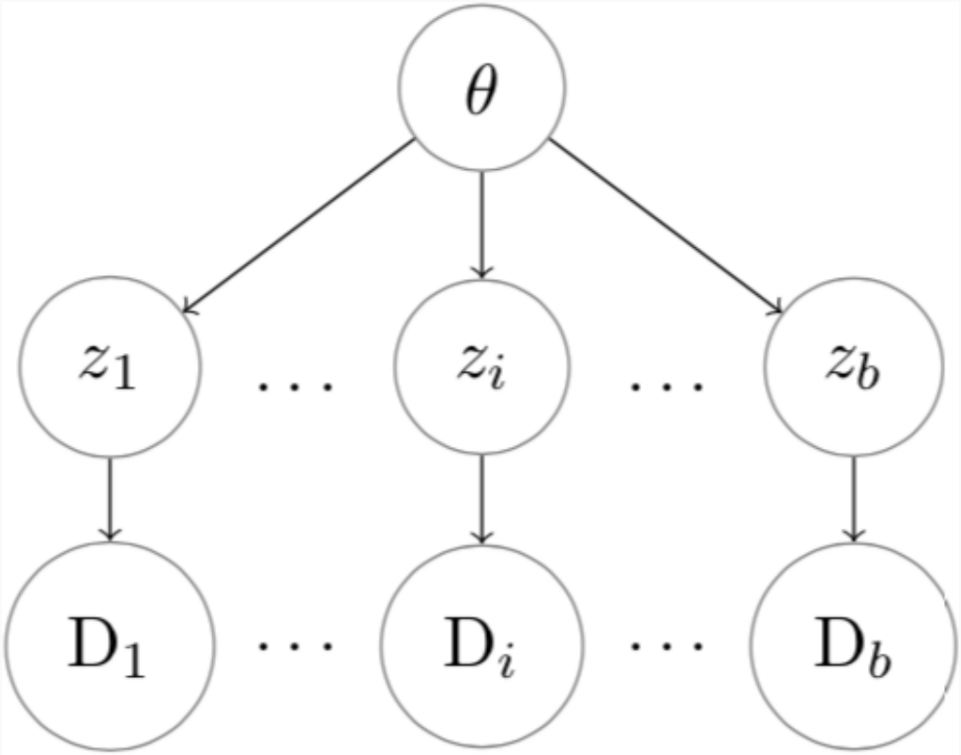
$$p(\theta \mid \mathcal{D}) \propto \prod_{i=1}^{b} p_i(\theta \mid \mathcal{D}_i)$$

Problem:

- Each **worker** only have access to $\mathcal{D}_i$
- Cope with **communication bottleneck**
- How to create a **distributed algorithm**?

Hierarchical model - herding

Idea: Introduce auxiliary variables $z_1, \dots z_b$ to distribute the calculations



⚠Key idea . Vono et al. (2019, 2020)

- Server updates θ using the auxiliary variables $z_1, \dots, z_b$
- Workers updates z_i using the local data $\mathcal{D}_i$

Hierarchical model - herding

Replace the local posterior, tolerance parameter $\boldsymbol{\rho}_i > 0$

$$p_i(\theta|\mathscr{D}_i) \propto p_i(\theta)p_i(\mathscr{D}_i|\theta) \longrightarrow p_i(\theta)\underbrace{p_i(\mathscr{D}_i|z_i)g_{\boldsymbol{\rho}_i}(z_i,\theta)}_{\text{ensure} \approx p_i(\mathscr{D}_i|\theta)}$$

Hierarchical model - herding

Replace the local posterior , tolerance parameter $\boldsymbol{\rho}_i > 0$

$$p_i(\theta|\mathscr{D}_i) \propto p_i(\theta)p_i(\mathscr{D}_i|\theta) \longrightarrow p_i(\theta)\underbrace{p_i(\mathscr{D}_i|z_i)g_{\boldsymbol{\rho}_i}(z_i,\theta)}_{\text{ensure} \approx p_i(\mathscr{D}_i|\theta)}$$

Remarks:

- Objective: $p_i(\mathscr{D}_i|z_i) \approx p_i(\mathscr{D}_i|\theta)$
- An elementary choice

$$g_{\boldsymbol{\rho}_i}(z_i,\theta) = \mathcal{N}(z_i|\theta, \boldsymbol{\rho}_i \mathrm{I}_d)$$

Hierarchical model - herding

Replace the local posterior , tolerance parameter $\boldsymbol{\rho}_i > 0$

$$p_i(\theta|\mathscr{D}_i) \propto p_i(\theta)p_i(\mathscr{D}_i|\theta) \longrightarrow p_i(\theta)\underbrace{p_i(\mathscr{D}_i|z_i)g_{\boldsymbol{\rho}_i}(z_i,\theta)}_{\text{ensure} \approx p_i(\mathscr{D}_i|\theta)}$$

Remarks:

- Objective: $p_i(\mathscr{D}_i|z_i) \approx p_i(\mathscr{D}_i|\theta)$
- An elementary choice

$$g_{\boldsymbol{\rho}_i}(z_i,\theta) = \mathcal{N}(z_i|\theta,\boldsymbol{\rho}_i \mathrm{I}_d)$$

- Target $p_{\boldsymbol{\rho}}(\theta, z_{1:b}|\mathscr{D})$

$$p_{\boldsymbol{\rho}}(\theta, z_{1:b}|\mathscr{D}) \propto \prod_{i=1}^{b} \left\{ p_i(\theta) \underbrace{p_i(\mathscr{D}_i|z_i)\mathcal{N}(z_i|\theta,\boldsymbol{\rho}_i \mathrm{I}_d)}_{\text{replace } p(\mathscr{D}_i|\theta)} \right\}$$

Hierarchical model - herding

Replace the local posterior , tolerance parameter $\boldsymbol{\rho}_i > 0$

$$p_i(\theta|\mathscr{D}_i) \propto p_i(\theta)p_i(\mathscr{D}_i|\theta) \longrightarrow p_i(\theta)\underbrace{p_i(\mathscr{D}_i|z_i)g_{\boldsymbol{\rho}_i}(z_i,\theta)}_{\text{ensure} \approx p_i(\mathscr{D}_i|\theta)}$$

Remarks:

- Objective: $p_i(\mathscr{D}_i|z_i) \approx p_i(\mathscr{D}_i|\theta)$
- An elementary choice

$$g_{\boldsymbol{\rho}_i}(z_i,\theta) = \mathcal{N}(z_i|\theta,\boldsymbol{\rho}_i \mathrm{I}_d)$$

- Target $p_{\boldsymbol{\rho}}(\theta, z_{1:b}|\mathscr{D})$

$$p_{\boldsymbol{\rho}}(\theta, z_{1:b}|\mathscr{D}) \propto \prod_{i=1}^{b} \left\{ p_i(\theta) \underbrace{p_i(\mathscr{D}_i|z_i)\mathcal{N}(z_i|\theta,\boldsymbol{\rho}_i \mathrm{I}_d)}_{\text{replace } p(\mathscr{D}_i|\theta)} \right\}$$

Hierarchical model - herding

Replace the local posterior , tolerance parameter $\boldsymbol{\rho}_i > 0$

$$p_i(\theta|\mathscr{D}_i) \propto p_i(\theta)p_i(\mathscr{D}_i|\theta) \longrightarrow p_i(\theta)\underbrace{p_i(\mathscr{D}_i|z_i)g_{\boldsymbol{\rho}_i}(z_i,\theta)}_{\text{ensure} \approx p_i(\mathscr{D}_i|\theta)}$$

Remarks:

- Objective: $p_i(\mathscr{D}_i|z_i) \approx p_i(\mathscr{D}_i|\theta)$
- An elementary choice

$$g_{\boldsymbol{\rho}_i}(z_i,\theta) = \mathcal{N}(z_i|\theta,\boldsymbol{\rho}_i \mathrm{I}_d)$$

- Target $p_{\boldsymbol{\rho}}(\theta, z_{1:b}|\mathscr{D})$

$$p_{\boldsymbol{\rho}}(\theta, z_{1:b}|\mathscr{D}) \propto \prod_{i=1}^{b} \left\{ p_i(\theta) \underbrace{p_i(\mathscr{D}_i|z_i)\mathcal{N}(z_i|\theta,\boldsymbol{\rho}_i \mathrm{I}_d)}_{\text{replace } p(\mathscr{D}_i|\theta)} \right\}$$

Gibbs Sampler

1. In parallel on each worker

$$z_i^{(t)} | \theta^{(t-1)} \sim p_i(\mathcal{D}_i | \cdot) \mathcal{N}(\cdot | \theta^{(t-1)}, \boldsymbol{\rho}_i \mathrm{I}_d)$$

2. On the central server $\rightarrow$ consensus step

$$\theta^{(t)} | z_{1:b}^{(t)} \sim p_i(\cdot) \prod_{i=1}^{b} \mathcal{N}(z_i^{(t)} | \cdot, \boldsymbol{\rho}_i \mathrm{I}_d)$$

Gibbs Sampler

1. In parallel on each worker

$$z_i^{(t)} | \theta^{(t-1)} \sim p_i(\mathscr{D}_i | \cdot) \mathcal{N}(\cdot | \theta^{(t-1)}, \boldsymbol{\rho}_i \mathrm{I}_d)$$

2. On the central server $\rightarrow$ consensus step

$$\theta^{(t)} | z_{1:b}^{(t)} \sim p_i(\cdot) \prod_{i=1}^{b} \mathcal{N}(z_i^{(t)} | \cdot, \boldsymbol{\rho}_i \mathrm{I}_d)$$

⚠ The server update on the server does not require $\mathscr{D}_1, \ldots, \mathscr{D}_b$

Gibbs Sampler

1. In parallel on each worker

$$z_i^{(t)} | \theta^{(t-1)} \sim p_i(\mathscr{D}_i | \cdot) \mathcal{N}(\cdot | \theta^{(t-1)}, \boldsymbol{\rho}_i \mathrm{I}_d)$$

2. On the central server $\rightarrow$ consensus step

$$\theta^{(t)} | z_{1:b}^{(t)} \sim p_i(\cdot) \prod_{i=1}^{b} \mathcal{N}(z_i^{(t)} | \cdot, \boldsymbol{\rho}_i \mathrm{I}_d)$$

⚠ The server update on the server does not require $\mathscr{D}_1, \ldots, \mathscr{D}_b$

⚠ Convergence: $\theta \rightsquigarrow p_{\boldsymbol{\rho}}(\cdot | \mathscr{D})$

Gibbs Sampler

1. In parallel on each worker

$$z_i^{(t)} | \theta^{(t-1)} \sim p_i(\mathscr{D}_i | \cdot) \mathcal{N}(\cdot | \theta^{(t-1)}, \boldsymbol{\rho}_i \mathrm{I}_d)$$

2. On the central server → consensus step

$$\theta^{(t)} | z_{1:b}^{(t)} \sim p_i(\cdot) \prod_{i=1}^{b} \mathcal{N}(z_i^{(t)} | \cdot, \boldsymbol{\rho}_i \mathrm{I}_d)$$

⚠ The server update on the server does not require $\mathscr{D}_1, \ldots, \mathscr{D}_b$

⚠ Convergence: $\theta \rightsquigarrow p_{\boldsymbol{\rho}}(\cdot | \mathscr{D})$

⚠ Step 1: difficult → Langevin Monte Carlo.

Dg-Lmc

1. In parallel on each worker (N_i Langevin local steps)
   - $y_i^{(0)} = z_i^{(t-1)}$
   - For all $k = 1, \dots, N_i$,

$$y_i^{(k)} = (1 - \gamma_i/\rho_i) y_i^{(k-1)} + \gamma_i \nabla \log p_i(\mathcal{D}_i | y_i^{(k-1)}) + (\gamma_i/\rho_i)\theta^{(t-1)} + \sqrt{2\gamma_i}\mathcal{N}(0_d, \mathrm{I}_d)$$

   - $z_i^{(t)} = y_i^{N_i}$

Dg-Lmc

1. In parallel on each worker (N_i Langevin local steps)
   - $y_i^{(0)} = z_i^{(t-1)}$
   - For all $k = 1, \ldots, N_i$,

$$y_i^{(k)} = (1 - \gamma_i/\rho_i) y_i^{(k-1)} + \gamma_i \nabla \log p_i(\mathcal{D}_i | y_i^{(k-1)}) + (\gamma_i/\rho_i)\theta^{(t-1)} + \sqrt{2\gamma_i}\mathcal{N}(0_d, \mathrm{I}_d)$$

   - $z_i^{(t)} = y_i^{N_i}$

2. On the central server

$$\theta^{(t)} | z_{1:b}^{(t)} \sim \prod_{i=1}^{b} \left\{ p_i(\cdot) \mathcal{N}(\cdot | z_{1:b}^{(t)}, \rho_i \mathrm{I}_d) \right\}$$

Dg-Lmc

1. In parallel on each worker (N_i Langevin local steps)
   - $y_i^{(0)} = z_i^{(t-1)}$
   - For all $k = 1, \ldots, N_i$,

$$y_i^{(k)} = (1 - \gamma_i/\rho_i) y_i^{(k-1)} + \gamma_i \nabla \log p_i(\mathcal{D}_i | y_i^{(k-1)}) + (\gamma_i/\rho_i)\theta^{(t-1)} + \sqrt{2\gamma_i}\mathcal{N}(0_d, \mathrm{I}_d)$$

   - $z_i^{(t)} = y_i^{N_i}$

2. On the central server

$$\theta^{(t)} | z_{1:b}^{(t)} \sim \prod_{i=1}^{b} \left\{ p_i(\cdot) \mathcal{N}(\cdot | z_{1:b}^{(t)}, \rho_i \mathrm{I}_d) \right\}$$

⚠Step 1: Langevin discretization $\rightsquigarrow p_\rho \to p_{\rho,\gamma,N}$.

Non-asymptotic theoretical guarantees

Assumptions

- H1: $\log p_i(\cdot|\mathcal{D})$ is m_i-strongly concave
- H2: $\nabla \log p_i(\cdot|\mathcal{D})$ is M_i-Lipschitz
- H3: $\nabla^2 \log p_i(\cdot|\mathcal{D})$ is differentiable, L_i-Lipschitz

Parameters

- ρ_i : tolerance parameter
- γ_i : discretization time-step in Langevin Monte Carlo
- N_i : number of local updates before communication

Theoretical guarantees

- **Theorem** Dg-Lmc $\rightsquigarrow$ t-iterations (with some abuses in notation !)

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq \underbrace{W_2(\mathscr{L}(\theta_t), p_{\boldsymbol{\rho},\gamma,N})}_{\text{Algorithm efficiency}} + \underbrace{W_2(p_{\boldsymbol{\rho},\gamma,N}, p_{\boldsymbol{\rho}})}_{\text{Langevin discretization}} + \underbrace{W_2(p_{\boldsymbol{\rho}}, p(\cdot|\mathscr{D}))}_{\text{herding error}}$$

Theoretical guarantees

- **Theorem** Dg-Lmc $\rightsquigarrow$ t-iterations (with some abuses in notation !)

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq \underbrace{W_2(\mathscr{L}(\theta_t), p_{\boldsymbol{\rho},\gamma,N})}_{\text{Algorithm efficiency}} + \underbrace{W_2(p_{\boldsymbol{\rho},\gamma,N}, p_{\boldsymbol{\rho}})}_{\text{Langevin discretization}} + \underbrace{W_2(p_{\boldsymbol{\rho}}, p(\cdot|\mathscr{D}))}_{\text{herding error}}$$

  - H1 & H2

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq C_0(1-\kappa_{\boldsymbol{\rho},\gamma})^t + \frac{C_1}{\boldsymbol{\rho}}\sqrt{d\gamma} + C_2 d\boldsymbol{\rho}$$

Theoretical guarantees

- **Theorem** Dg-Lmc $\rightsquigarrow$ t-iterations (with some abuses in notation !)

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq \underbrace{W_2(\mathscr{L}(\theta_t), p_{\boldsymbol{\rho},\gamma,N})}_{\text{Algorithm efficiency}} + \underbrace{W_2(p_{\boldsymbol{\rho},\gamma,N}, p_{\boldsymbol{\rho}})}_{\text{Langevin discretization}} + \underbrace{W_2(p_{\boldsymbol{\rho}}, p(\cdot|\mathscr{D}))}_{\text{herding error}}$$

  - H1 & H2

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq C_0(1-\kappa_{\boldsymbol{\rho},\gamma})^t + \frac{C_1}{\boldsymbol{\rho}}\sqrt{d\gamma} + C_2 d\boldsymbol{\rho}$$

  - H1 & H2 & H3

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq C_0(1-\kappa_{\boldsymbol{\rho},\gamma})^t + \frac{C_1}{\boldsymbol{\rho}}\sqrt{d\gamma(\gamma+\boldsymbol{\rho}^4)} + C_2 d\boldsymbol{\rho}$$

Theoretical guarantees

- **Theorem** Dg-Lmc $\rightsquigarrow$ t-iterations (with some abuses in notation !)

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq \underbrace{W_2(\mathscr{L}(\theta_t), p_{\rho,\gamma,N})}_{\text{Algorithm efficiency}} + \underbrace{W_2(p_{\rho,\gamma,N}, p_\rho)}_{\text{Langevin discretization}} + \underbrace{W_2(p_\rho, p(\cdot|\mathscr{D}))}_{\text{herding error}}$$

  - H1 & H2

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq C_0(1-\kappa_{\rho,\gamma})^t + \frac{C_1}{\rho}\sqrt{d\gamma} + C_2 d\rho$$

  - H1 & H2 & H3

$$W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq C_0(1-\kappa_{\rho,\gamma})^t + \frac{C_1}{\rho}\sqrt{d\gamma(\gamma+\rho^4)} + C_2 d\rho$$

⚠Advantage: $\rho \ll 1$ without explosion of $W_2(p_{\rho,\gamma,N}, p_\rho)$

Mixing-Time

How to use the algorithm?

- Choose the hyperparameters : ρ, γ, N and the number of global iterations : t

Mixing-Time

How to use the algorithm?

- Choose the hyperparameters : ρ, γ, N and the number of global iterations : t

Objective:

- Sample a parameter θ_t satisfying: $W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq \varepsilon$

Mixing-Time

How to use the algorithm?

- Choose the hyperparameters : ρ, γ, N and the number of global iterations : t

Objective:

- Sample a parameter θ_t satisfying: $W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq \varepsilon$

$$\underbrace{W_2(p_\rho, p(\cdot|\mathscr{D}))}_{\rho \lesssim d^{-1}\varepsilon} \leq \frac{\varepsilon}{3} \quad \underbrace{W_2(\mathscr{L}(\theta_t), p_{\rho\gamma N})}_{\gamma, N \text{ choices}} \leq \frac{\varepsilon}{3} \quad \underbrace{W_2(p_{\rho\gamma N}, p_\rho)}_{t \gtrsim -\kappa_{\rho,\gamma}^{-1} \log \varepsilon} \leq \frac{\varepsilon}{3}$$

Mixing-Time

How to use the algorithm?

- Choose the hyperparameters : ρ, γ, N and the number of global iterations : t

Objective:

- Sample a parameter θ_t satisfying: $W_2(\mathscr{L}(\theta_t), p(\cdot|\mathscr{D})) \leq \varepsilon$

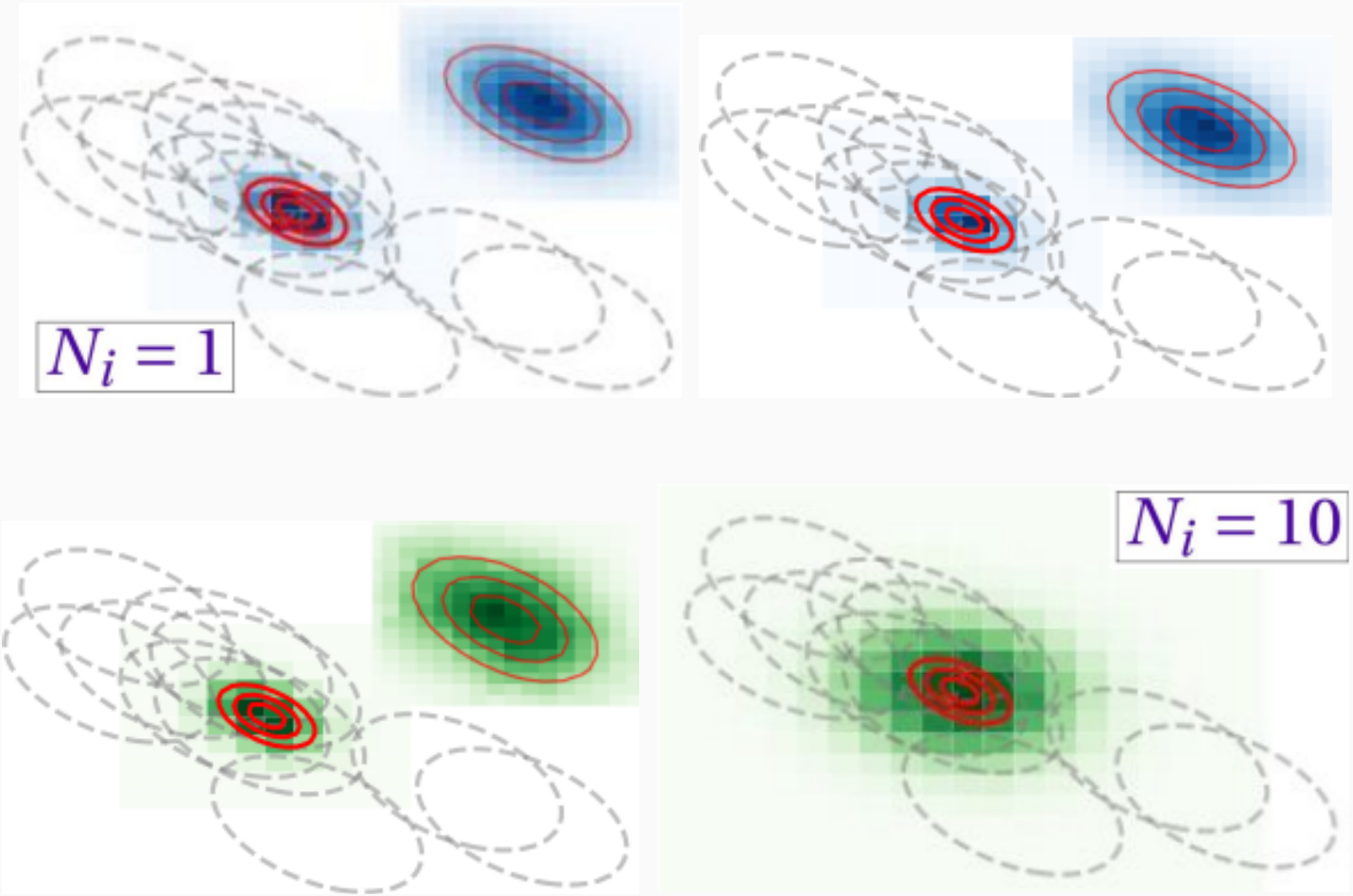
$$\underbrace{W_2(p_\rho, p(\cdot|\mathscr{D}))}_{\rho \lesssim d^{-1}\varepsilon} \leq \frac{\varepsilon}{3} \quad \underbrace{W_2(\mathscr{L}(\theta_t), p_{\rho\gamma N})}_{\gamma, N \text{ choices}} \leq \frac{\varepsilon}{3} \quad \underbrace{W_2(p_{\rho\gamma N}, p_\rho)}_{t \gtrsim -\kappa_{\rho,\gamma}^{-1} \log \varepsilon} \leq \frac{\varepsilon}{3}$$

Assumptions	ρ_ε	γ_ε	N_ε	Nb. grad. eval.
H1-H2	d^{-1}	d^{-3}	d	$d^3 \log(d)$
	ε	ε^4	ε^{-2}	$\varepsilon^{-4} \lvert\log(\varepsilon)\rvert$
H1-H2-H3	d^{-1}	d^{-2}	1	$d^2 \log(d)$
	ε	ε^2	1	$\varepsilon^{-2} \lvert\log(\varepsilon)\rvert$

Toy Gaussian example

Dg-Lmc Vs D-Sgld (Distributed Stochastic Gradient Langevin Dynamics)

True posterior Vs Approximate posterior

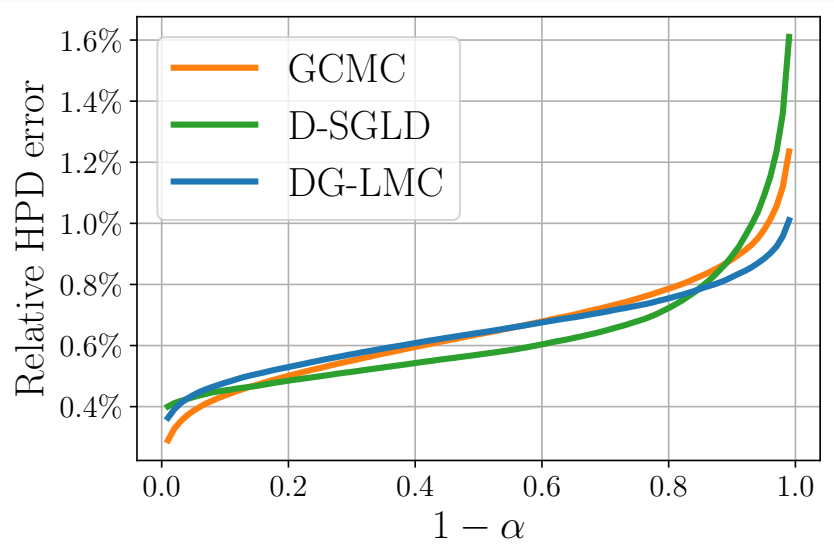


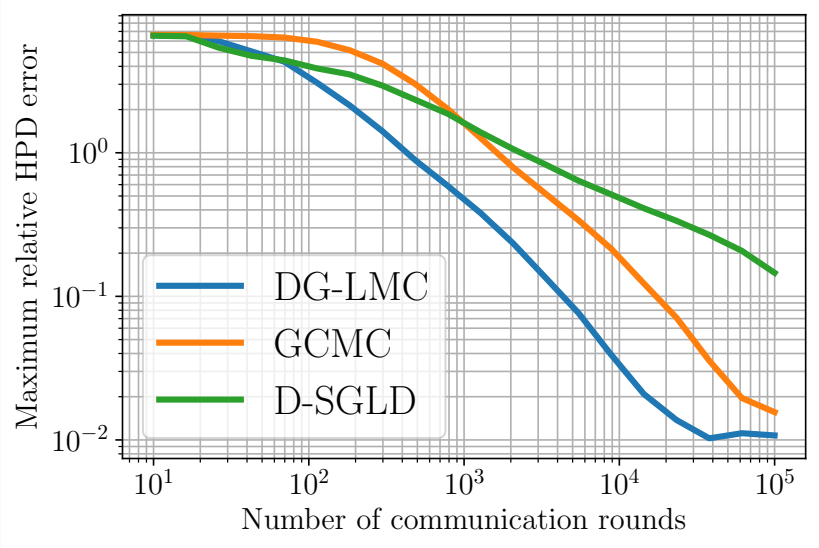
Highest posterior density

HPD region estimation

$$\begin{cases} \mathscr{C}_\alpha = \{\theta \in \mathbb{R}^d : -\log p(\theta \mid \mathscr{D}) \leq \eta_\alpha)\} \\ \eta_\alpha \text{ such that } \int_{\mathscr{C}_\alpha} p(\theta \mid \mathscr{D}) \mathrm{d}\theta = 1 - \alpha \end{cases}$$

HDP error: $\left|\eta_\alpha - \eta_\alpha^{\text{true}}\right| / \eta_\alpha^{\text{true}}$

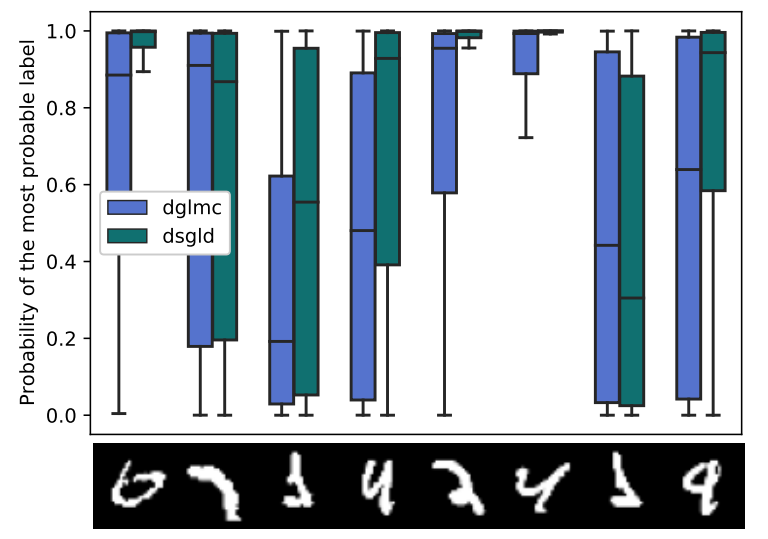


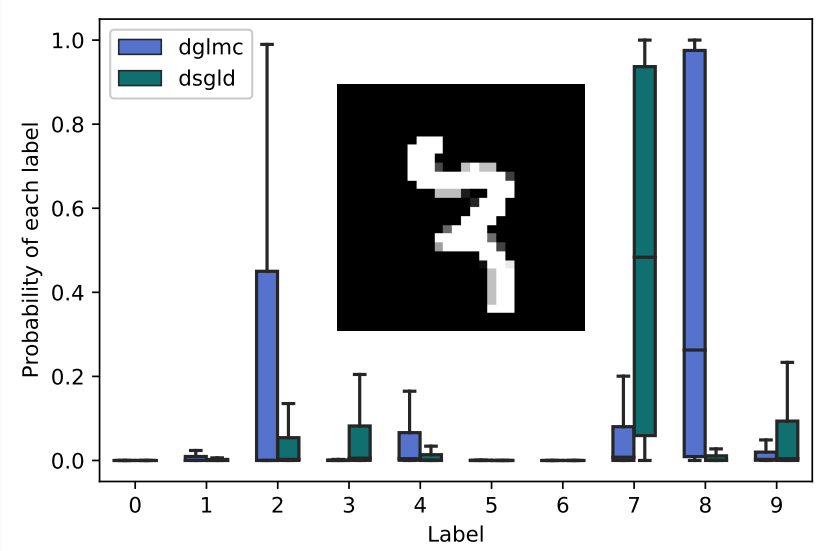


Bayesian neural network

Boxplots on $\{p(y \mid x, \theta)\}_{\theta \sim p(\cdot \mid \mathcal{D})}$.

Dg-Lmc $\approx$ D-Sgld but more robust to the choice of hyperparameters.





Hyperparameters tuning : guidelines

- Theoretically, the optimal choice

$$N\gamma \asymp \frac{m\rho^2}{(\rho M + 1)^2}$$

- Experimentally, good empirical results

$$\rho \approx \frac{1}{M} \qquad \gamma \ll \rho$$

FALD: Stochastic Averaging Langevin Dynamics

Federated Averaging Langevin Dynamics (FALD).

- FALD, proposed in Deng et al. (2021), is an extension to the Bayesian setting of FedAvg (McMahan et al., 2017).
- The updates performed on the ith client define a sequence of local parameters $(X_k^i)_{k\in\mathbb{N}}$ which are transmitted according to some preset schedule (which is deterministic in Deng et al. (2021) and is random in this work) to a central server.
- The central server averages the local parameters to update the global parameter. This global parameter is transmitted back to each client, and is used as a starting point of a new round of local iterations.

Local iteration (on each client)

Each client i performs one step of the Langevin Monte Carlo algorithm (Roberts and Tweedie, 1996) with a stochastic gradient associated with its local potential:

$$G^i_{k+1} = \widehat{\nabla} U^i_{k+1}(X^i_k),$$
$$X^i_{k+1} = X^i_k - \gamma G^i_{k+1} + \sqrt{2\gamma}\, Z^i_{k+1},$$

where $\gamma > 0$ and for $x \in \mathbb{R}^d$, $\widehat{\nabla} U^i_{k+1}(x)$ is an unbiased estimator of $\nabla U^i(x)$ given by

$$\widehat{\nabla} U^i_{k+1} = \omega^i \nabla U^0 + (N_i/n_i) \textstyle\sum_{j \in S^i_{k+1}} \nabla U^{i,j},$$

where

- $(S^i_k)_{k \in \mathbb{N}^\star}$ is a sequence of i.i.d. uniform random subsets of $[N_i]$ of cardinal number n_i.
- $(Z^i_k)_{k \in \mathbb{N}^*}$, $i \in [b]$ are sequence of i.i.d Gaussian random variables which might be correlated across the agents and the central server.

A local update.

1. With probability $p_c \in (0,1]$,
   - the ith client communicates its parameter X^i_{k+1} to the central server which in turns broadcasts the average $X_{k+1} = b^{-1} \sum_{i \in [b]} X^i_{k+1}$.
   - each client updates its parameter as $X^i_{k+1} = X_{k+1}$.
2. When no communication is performed, each client updates its parameter as $X^i_{k+1} = X^i_{k+1}$.

The local recursions defined by FALD can be written for $i \in [b]$ and $k \geq 0$ as

$$X^i_{k+1} = (1 - B_{k+1}) X^i_{k+1} + (B_{k+1}/b) \sum_{j \in [b]} X^j_{k+1} ,$$

where $(B_k)_{k \in \mathbb{N}^*}$ is a sequence of i.i.d. Bernoulli random variables with parameter p_c.

For $k \geq 1$, denote by $\mu^{(\mathrm{F})}_k$ the distribution of the average parameter

$$X_k = (1/b) \sum_{i \in [b]} X^i_k .$$

Assumptions

- For any $i \in [b]$, U^i is continuously differentiable. In addition, there exist $m, L > 0$ such that for any $i \in [b]$, the function U^i is L-smooth and m-strongly convex, *i.e.*, for any $x, x' \in \mathbb{R}^d$,

$$(m/2)\|x' - x\|^2 \leq U^i(x') - U^i(x) - \left\langle \nabla U^i(x), x' - x \right\rangle \leq (L/2)\|x' - x\|^2.$$

- For any $i \in [b]$, $(\{\widehat{\nabla} U^i_k\}_{i \in [b]})_{k \in \mathbb{N}}$ are i.i.d. unbiased estimates of $\{\nabla U^i\}_{i \in [b]}$. In addition, there exists $L \geq 0$ such that for any $x, x' \in \mathbb{R}^d$ we have

$$\mathbb{E}\left[\|\widehat{\nabla} U^i_k(x') - \widehat{\nabla} U^i_k(x)\|^2\right] \leq L^2 \|x' - x\|^2 .$$

Key quantities

Denote by $x_\star$ the minimizer of $\sum_{i\in[b]} U^i$ which exists and is unique under the strong convexity assumption. We define

$$\mathsf{V}_\pi = \int_{\mathbb{R}^d} \mathsf{Var}\{b^{-1}\textstyle\sum_{i\in[b]} \widehat{\nabla} U_1^i(x)\}\pi(\mathrm{d}x),$$

$$\mathsf{V}_\star = \mathsf{Var}\{b^{-1}\textstyle\sum_{i\in[b]} \widehat{\nabla} U_1^i(x_\star)\},$$

the average of the stochastic gradient variance under the stationary distribution π and at the minimum $x_\star$, respectively.

The statistical heterogeneity between the clients is quantified by

$$\mathsf{H} = b^{-1}\textstyle\sum_{i\in[b]} \|\nabla U^i(x_\star)\|^2.$$

Theorem (Simplified)

there exist $\gamma > 0$, *such that for any* $\gamma \in (0, \gamma]$, $k \in \mathbb{N}$, $X_0 \sim \mu_0 \in \mathscr{P}_2(\mathbb{R}^d)$, *we have*

$$W_2{}^2(\mu_k^{(\mathrm{F})}, \pi) \lesssim (1 - \gamma m/8)^k \, \mathsf{I}(\mu_0) + \frac{\gamma}{b} d + \frac{\gamma^2 (1 - p_c)}{p_c{}^2} \left\{ \mathsf{H} + p_c \mathsf{V}_\star + \frac{d}{b} \right\} + \frac{\gamma (1 - \tau)(1 - b^{-1}) d}{p_c},$$

Take away

- **Bayesian** $\leftrightarrow$ **uncertainty quantification**
- Lot of **theoretical** results $\approx$ optimization, but **stronger assumptions**
- Algorithms **sample** $\theta_1, \ldots, \theta_k \sim p(\theta|\mathscr{D}) \neq \theta_\star$ optimization

Conclusion.

- **Introduce 3 algorithms** for distributed Bayesian inference & $\int h(\theta) p(\theta|\mathscr{D}) \mathrm{d}\theta$.
- **Theoretically** grounded methodology.
- **Numerically** well-founded.

References

Alistarh, D., D. Grubic, J. Li, R. Tomioka, and M. Vojnovic (2017). Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in Neural Information Processing Systems 30*.

Baker, J., P. Fearnhead, E. B. Fox, and C. Nemeth (2019). Control variates for stochastic gradient mcmc. *Statistics and Computing 29*(3), 599–615.

Belomestny, D., L. Iosipoi, E. Moulines, A. Naumov, and S. Samsonov (2021). Variance reduction for dependent sequences with applications to stochastic gradient mcmc. *SIAM/ASA Journal on Uncertainty Quantification 9*(2), 507–535.

Brosse, N., A. Durmus, and E. Moulines (2018). The promises and pitfalls of stochastic gradient langevin dynamics. *Advances in Neural Information Processing Systems 31*.

Deng, W., Y.-A. Ma, Z. Song, Q. Zhang, and G. Lin (2021). On convergence of federated averaging langevin dynamics. *arXiv preprint arXiv:2112.05120*.

Bibliography ii

Horváth, S., D. Kovalev, K. Mishchenko, S. Stich, and P. Richtárik (2019). Stochastic distributed learning with gradient quantization and variance reduction. *arXiv preprint arXiv:1904.05115*.

Ma, Y.-A., T. Chen, and E. Fox (2015). A complete recipe for stochastic gradient mcmc. *Advances in neural information processing systems 28*.

McMahan, B., E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR.

Roberts, G. O. and R. L. Tweedie (1996, 12). Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli 2*(4), 341–363.

Teh, Y. W., A. H. Thiery, and S. J. Vollmer (2016). Consistency and fluctuations for stochastic gradient langevin dynamics. *The Journal of Machine Learning Research 17*(1), 193–225.

Vono, M., N. Dobigeon, and P. Chainais (2019). Split-and-augmented gibbs sampler—application to large-scale inference problems. *IEEE Transactions on Signal Processing 67*(6), 1648–1661.

Bibliography iii

Vono, M., N. Dobigeon, and P. Chainais (2020). Asymptotically exact data augmentation: Models, properties, and algorithms. *Journal of Computational and Graphical Statistics 30*(2), 335–348.

Welling, M. and Y. W. Teh (2011). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688. Citeseer.