# Asynchronous Stochastic Quasi-Newton MCMC for Non-Convex Optimization


Umut Şimşekli[1], Çağatay Yıldız[2], Thanh Huy Nguyen[1], Gaël Richard[1], A. Taylan Cemgil[3]

1: Télécom ParisTech, Paris, France 2: Aalto University, Espoo, Finland 3: Boğaziçi University, Istanbul, Turkey



Supported by ANR (ANR-16-CE23-0014), TÜBİTAK (116E580), the industrial chair Machine Learning for Big Data from Télécom ParisTech


## Introduction & Motivation

- Distributed L-BFGS [1]: promising algorithm for solving:

$$\theta^\star = \arg\min_{\theta \in \mathbb{R}^d} \Big\{ U(\theta) \triangleq \sum_{i=1}^{N_Y} U_i(\theta) \Big\}$$

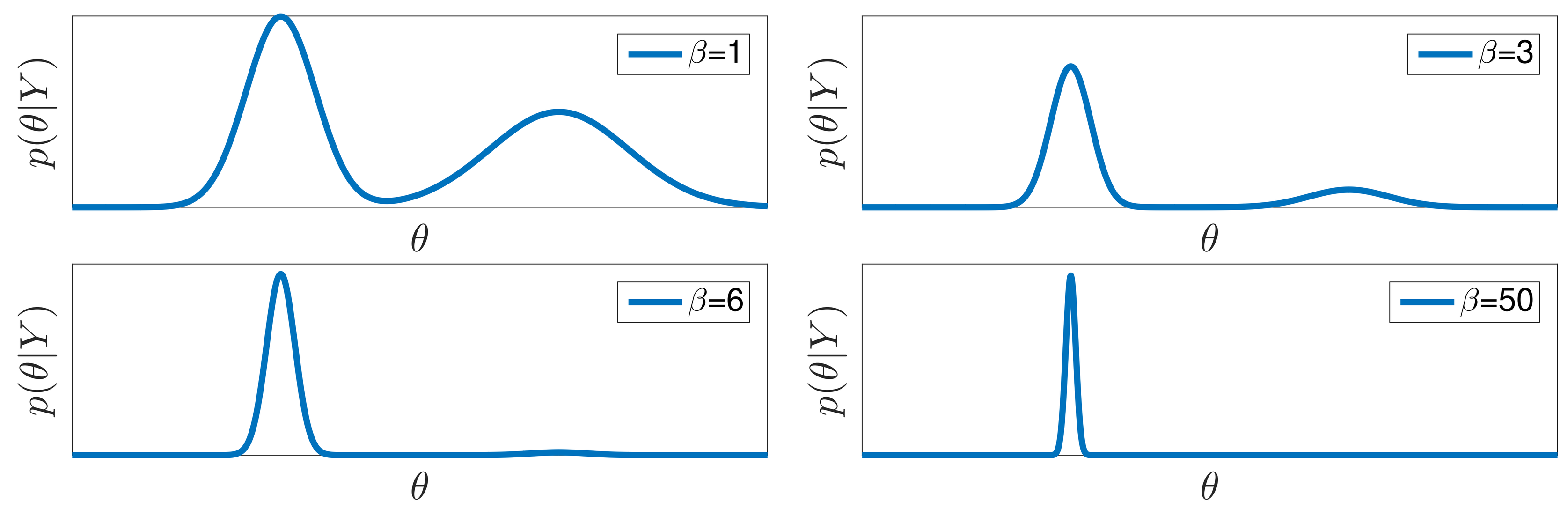
– Its computational efficiency might be improved by reducing the synchronization overhead

- **Goal**: Develop **asynchronous distributed L-BFGS**
- **Approach**: Sample from a **tempered posterior**:

$$\theta^\star \approx \theta \sim \{ p(\theta|Y) \propto \exp(-\beta U(\theta)) \}$$

– $Y$: dataset, $\beta$: inverse temperature

– When $\beta \to \infty$: $p(\theta|Y) \to \delta_{\theta^\star}$ (global optimum)




– For large $\beta$, a sample from $p(\theta|Y)$ will be close to $\theta^\star$

– Develop an **L-BFGS-based, asynch. distributed MCMC algorithm** for sampling from $p(\theta|Y)$

## Technical Background

- **The L-BFGS algorithm:** [2]

– $\theta_n = \theta_{n-1} - h H_n \nabla U(\theta_{n-1})$

– $H_n$: approximate inverse Hessian

– Requires iterate and gradient differences:

$(\theta_n - \theta_{n-1}), \quad (\nabla U(\theta_n) - \nabla U(\theta_{n-1}))$

- **Stochastic Gradient MCMC:** [3]

$$\theta_n = \theta_{n-1} - h \nabla \tilde{U}_n(\theta_{n-1}) + \sqrt{2h/\beta} Z_n$$

– $\nabla \tilde{U}$: stochastic gradient

– $Z_n \sim \mathcal{N}(0, I)$, Gaussian noise

– Discretization of: $d\theta_t = -\nabla U(\theta_t) dt + \sqrt{2/\beta} dW_t$

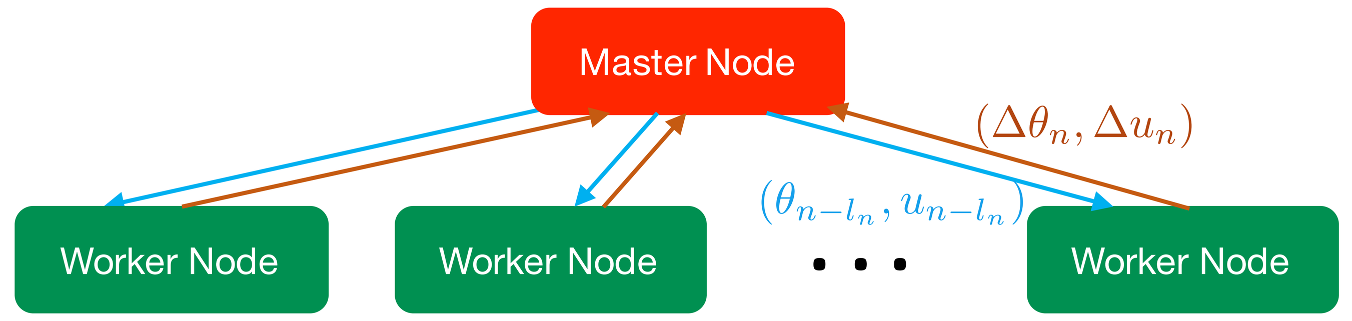
+ $W_t$ : Brownian motion

+ Stationary measure $\propto \exp(-\beta U(\theta))$

+ Recently used for optimization [4]

## Method

- **Architecture:** Master node + independent Worker nodes




- **Master node:** Store the up-to-date sample

$$u_{n+1} = u_n + \Delta u_{n+1}, \qquad \theta_{n+1} = \theta_n + \Delta\theta_{n+1}$$

- **Worker nodes:** L-BFGS computations

$$\Delta u_{n+1} \triangleq - h' H_{n+1}(\theta_{n-l_n}) \nabla \tilde{U}(\theta_{n-l_n}) - \gamma' u_{n-l_n} + \sqrt{2h'\gamma'/\beta} Z_{n+1}$$
$$\Delta\theta_{n+1} \triangleq H_{n+1}(\theta_{n-l_n}) u_{n-l_n}$$

– $u_n$: momentum variable, $h'$: step-size, $\gamma'$: friction

– $l_n$: the 'delay' at iteration $n$, $\max_n l_n = l_{\max}$

– $H_n$: L-BFGS matrix: computed on *local* variables of a worker

- **SGD-momentum:** $\beta \to \infty$, $l_{\max} = 0$, $H_n(\theta) = I$

## Non-Asymptotic Analysis

- Start with the stochastic differential equation:

$$dp_t = \Big[ (1/\beta)\Gamma_t(\theta_t) - H_t(\theta_t)\nabla U(\theta_t) - \gamma p_t \Big] dt + \sqrt{2\gamma/\beta}\, dW_t$$
$$d\theta_t = H_t(\theta_t) p_t dt$$

– $\Gamma$: partial derivatives of $H_t$

– Invariant measure with density $\propto \exp(-\beta U(\theta) - (\beta/2) p^\top p)$

- Euler discretization: ($h$: step-size)

$$p_{n+1} = p_n - h H_n(\theta_n) \nabla U(\theta_n) - h\gamma p_n + \frac{h}{\beta} \Gamma_n(\theta_n) + \sqrt{2h\gamma/\beta} Z_{n+1}$$
$$\theta_{n+1} = \theta_n + h H_n(\theta_n) p_n$$

– Discard $\Gamma$, set $u_n \triangleq h p_n$, $\gamma' \triangleq h\gamma$, $h' \triangleq h^2$, use delayed iterates → proposed algorithm

- Analyze the **ergodic error**: $\mathbb{E}[\hat{U}_N - U^\star]$:

– $\hat{U}_N \triangleq \frac{1}{N}\sum_{n=1}^N U(\theta_n)$, $U^\star \triangleq \min_\theta U(\theta)$, $\bar{U}_\beta \triangleq \int \theta p(\theta|Y)\, d\theta$

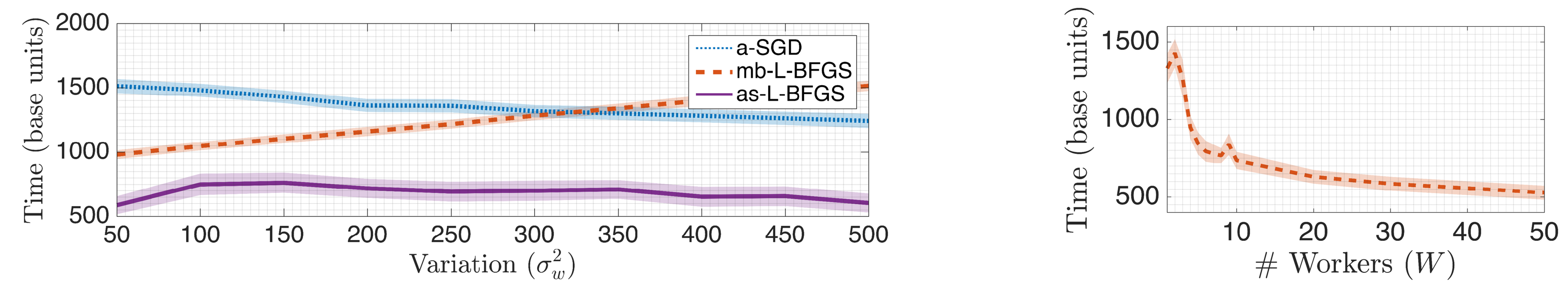
– Approach: $\mathbb{E}[\hat{U}_N - U^\star] = \mathbb{E}[\hat{U}_N - \bar{U}_\beta] + [\bar{U}_\beta - U^\star]$

## Experiments

- **Synthetic data:** $\theta \sim \mathcal{N}(0, I)$, $Y_i|\theta \sim \mathcal{N}(a_i^\top \theta, \sigma_x^2)$, $i \in [\![1, N_Y]\!]$

– Simulated distributed environment in MATLAB

– Explicit control on comp. times, delay, node variability

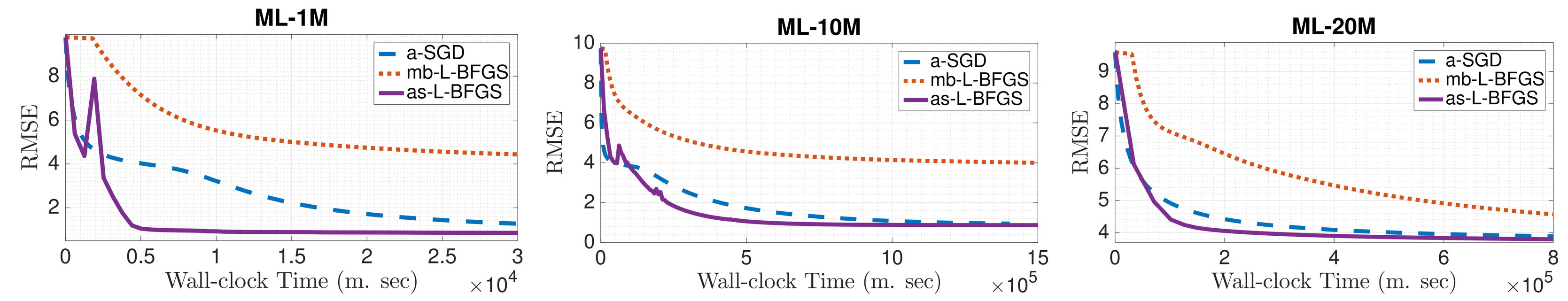



- **Large-scale matrix factorization:**

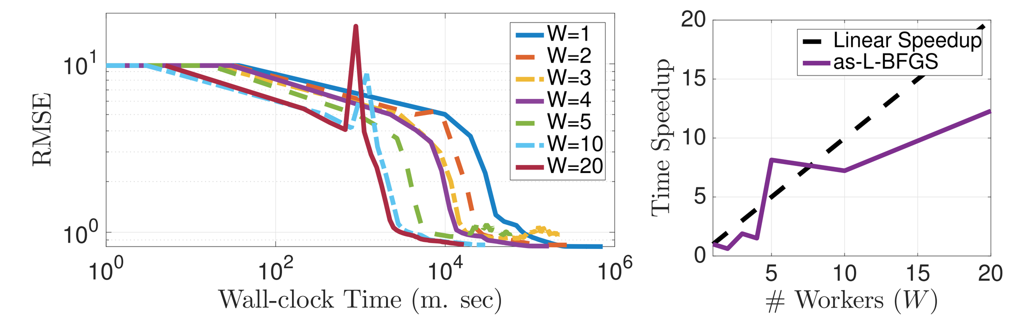
– $F_{rk} \sim \mathcal{N}(0,1)$, $G_{ks} \sim \mathcal{N}(0,1)$, $Y_{rs}|F,G \sim \mathcal{N}(\sum_k F_{rk} G_{ks}, 1)$

– C++ code with OpenMPI on a cluster with 500 computers

– MovieLens 1M, 10M, 20M datasets




– Sub-linear speed up:




- **Main assumptions:**

– $\nabla U$, $H_n$: Lipschitz, $H_n$: bounded 2nd order derivatives

– Stochastic gradients: $\mathbb{E}\|\nabla_\theta U(\theta) - \nabla_\theta \tilde{U}(\theta)\|^2 \le \sigma$

– Second-order moments: $\int_{\mathbb{R}^d} \|\theta\|^2 p(\theta|Y)\, d\theta \le C_\beta/\beta$

**Theorem 1**

The ergodic error of the proposed algorithm is bounded as follows:

$$\big|\mathbb{E}\hat{U}_N - U^\star\big| = \mathcal{O}\Big( \frac{1}{Nh} + \max(1, l_{\max}) h + \frac{1}{\beta} \Big)$$

## References


[1] Albert S Berahas, Jorge Nocedal, and Martin Takác. A multi-batch L-BFGS method for machine learning. In *Advances in Neural Information Processing Systems*, pages 1055–1063, 2016.

[2] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer, Berlin, 2006.

[3] M. Welling and Y. W Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *International Conference on Machine Learning*, pages 681–688, 2011.

[4] M. Raginsky, A. Rakhlin, and M. Telgarsky. Non-convex learning via stochastic gradient Langevin dynamics: a nonasymptotic analysis. In *Proceedings of the 2017 Conference on Learning Theory*, volume 65, pages 1674–1703, 2017.