Parallel computations for Metropolis Markov chains with Picard maps

Sebastiano Grazzi
with Giacomo Zanella
Department of Decision Science, Bocconi University, Milan

July 26, 2025

Funded by the European Union
European Research Council
Established by the European Commission

Outline

- **Overview**
- **Picard Map** Φ for **Markov chain simulation**
- **Main theoretical results**
- **Simulations**
- (Technical Appendix, only if time allows) **Contraction of** Φ

S. Grazzi, G. Zanella, ***Parallel computations for Metropolis Markov chains with Picard maps.*** **arXiv:2506.09762**

Overview

Zeroth-order Parallel Sampling

- **Objective**: Sample from a distribution $\pi(\mathrm{d}x) = C\exp(-V(x))\mathrm{d}x$ on $\mathcal{X} = \mathbb{R}^d$, for some unknown constant C.
  - Motivation: Bayesian Inference, statistical physics,...

Zeroth-order Parallel Sampling

- **Objective**: Sample from a distribution $\pi(\mathrm{d}x) = C\exp(-V(x))\mathrm{d}x$ on $\mathcal{X} = \mathbb{R}^d$, for some unknown constant C.
  - Motivation: Bayesian Inference, statistical physics,...
- **Setting** :
  - **Zeroth-order methods**: point-wise evaluation of V (and **not** ∇V, which is typical for first-order methods)
  - **Parallel computing**: $K \geq 1$ processors that can work in parallel to execute the task

Zeroth-order Parallel Sampling

- **Objective**: Sample from a distribution $\pi(\mathrm{d}x) = C\exp(-V(x))\mathrm{d}x$ on $\mathcal{X} = \mathbb{R}^d$, for some unknown constant C.
  - Motivation: Bayesian Inference, statistical physics,...
- **Setting** :
  - **Zeroth-order methods**: point-wise evaluation of V (and **not** ∇V, which is typical for first-order methods)
  - **Parallel computing**: $K \geq 1$ processors that can work in parallel to execute the task

Performance

(Parallel round) complexity: number of point-wise evaluations of V per parallel processor in order to obtain samples close to π (e.g. in total variation).

- **Important quantities**: dimension d, number of processors K.

Diagram Parallel sampling

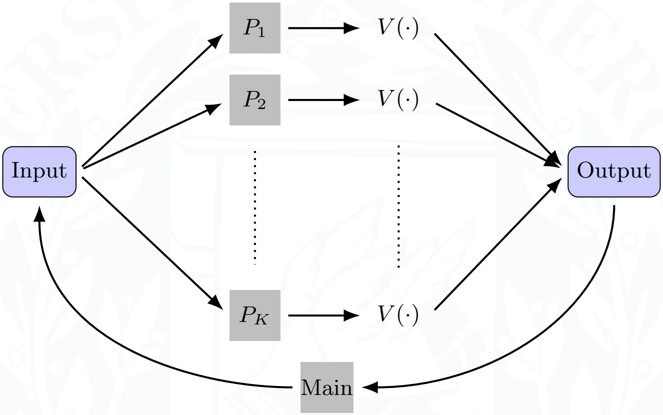


Figure 1: One parallel iteration of the algorithm

Markov chain Monte Carlo

- **Approach:** Markov chain Monte Carlo i.e. **simulate** a Markov chain

$$X_{i+1} = X_i + f(X_i, W_i)\,, \qquad\qquad i = 0, 1, \ldots \tag{1}$$

  whose limiting distribution coincides with π, for some i.i.d random variables $W_0, W_1, \ldots$

[1] the notation $\mathcal{O}$ ignores constants and log terms

Markov chain Monte Carlo

- **Approach:** Markov chain Monte Carlo i.e. **simulate** a Markov chain

$$X_{i+1} = X_i + f(X_i, W_i)\,, \qquad i = 0, 1, \ldots \tag{1}$$

  whose limiting distribution coincides with π, for some i.i.d random variables $W_0, W_1, \ldots$
- Random Walk Metropolis (RWM): $W = (Z, U)$, $U \sim \mathcal{U}([0,1])$, $Z \sim \mathcal{N}(0, \xi^2)$,

$$f(x, W) = ZB(x, U, Z) \quad \text{with } B(x, u, z) = \mathbb{1}\left(\pi(x+z)/\pi(x) \geq u\right).$$

[1] the notation $\mathcal{O}$ ignores constants and log terms

Markov chain Monte Carlo

- **Approach:** Markov chain Monte Carlo i.e. **simulate** a Markov chain

$$X_{i+1} = X_i + f(X_i, W_i)\,, \qquad i = 0, 1, \dots \tag{1}$$

  whose limiting distribution coincides with π, for some i.i.d random variables $W_0, W_1, \dots$

- Random Walk Metropolis (RWM): $W = (Z, U)$, $U \sim \mathcal{U}([0,1])$, $Z \sim \mathcal{N}(0, \xi^2)$,

$$f(x, W) = ZB(x, U, Z) \quad \text{with } B(x, u, z) = \mathbb{1}\left(\pi(x+z)/\pi(x) \geq u\right).$$

- **State of the art** for log-concave distributions and sequential algorithms ($K = 1$) with complexity $\mathcal{O}(d)$[1] (Andrieu et al. 2024).

[1] the notation $\mathcal{O}$ ignores constants and log terms

Markov chain Monte Carlo

- **Approach:** Markov chain Monte Carlo i.e. **simulate** a Markov chain

$$X_{i+1} = X_i + f(X_i, W_i) \,, \qquad i = 0, 1, \ldots \tag{1}$$

  whose limiting distribution coincides with π, for some i.i.d random variables $W_0, W_1, \ldots$
- Random Walk Metropolis (RWM): $W = (Z, U)$, $U \sim \mathcal{U}([0,1])$, $Z \sim \mathcal{N}(0, \xi^2)$,

$$f(x, W) = ZB(x, U, Z) \quad \text{with } B(x, u, z) = \mathbb{1}\left(\pi(x+z)/\pi(x) \geq u\right).$$

- **State of the art** for log-concave distributions and sequential algorithms ($K = 1$) with complexity $\mathcal{O}(d)$[1] (Andrieu et al. 2024).

How do we parallelize the recursion in (1)*, given its sequential nature?*

[1] the notation $\mathcal{O}$ ignores constants and log terms

Zeroth order parallel sampling

- **Previous attempts:**
  - **Pre-fetching**: computes V in each future potential state of the Markov chain for $j \geq 1$ steps ahead.
    - Caveat: number of potential states grows exponentially with j.

Bocconi

Zeroth order parallel sampling

- **Previous attempts:**
  - **Pre-fetching**: computes V in each future potential state of the Markov chain for $j \geq 1$ steps ahead.
    - Caveat: number of potential states grows exponentially with j.
  - **Multiple-try**: at each interation, simulates K proposal states and computes V in each state.
    - Caveat: $\text{Gap}(\text{Multiple-try}) \leq \text{Gap}(\text{RWM}) \log(K)$ (Pozza and Zanella 2024).

Zeroth order parallel sampling

- **Previous attempts:**
  - **Pre-fetching**: computes V in each future potential state of the Markov chain for $j \geq 1$ steps ahead.
    - Caveat: number of potential states grows exponentially with j.
  - **Multiple-try**: at each interation, simulates K proposal states and computes V in each state.
    - Caveat: $\mathrm{Gap}(\text{Multiple-try}) \leq \mathrm{Gap}(\text{RWM}) \log(K)$ (Pozza and Zanella 2024).

$\rightarrow$ Complexity $\mathcal{O}(d/log(K)) \rightarrow K$ has to grow exponentially with d

Bocconi

Zeroth order parallel sampling

- **Previous attempts:**
  - **Pre-fetching**: computes V in each future potential state of the Markov chain for $j \geq 1$ steps ahead.
    - Caveat: number of potential states grows exponentially with j.
  - **Multiple-try**: at each interation, simulates K proposal states and computes V in each state.
    - Caveat: $\text{Gap}(\text{Multiple-try}) \leq \text{Gap}(\text{RWM}) \log(K)$ (Pozza and Zanella 2024).

  $\rightarrow$ Complexity $\mathcal{O}(d/log(K)) \rightarrow K$ has to grow exponentially with d
- **Best scenarios**: no waste of computational power: from $\mathcal{O}(d)$ to $\mathcal{O}(d/K)$.

Zeroth order parallel sampling

- **Previous attempts:**
  - **Pre-fetching**: computes V in each future potential state of the Markov chain for $j \geq 1$ steps ahead.
    - Caveat: number of potential states grows exponentially with j.
  - **Multiple-try**: at each interation, simulates K proposal states and computes V in each state.
    - Caveat: $\text{Gap}(\text{Multiple-try}) \leq \text{Gap}(\text{RWM}) \log(K)$ (Pozza and Zanella 2024).

  → Complexity $\mathcal{O}(d/log(K)) \to K$ has to grow exponentially with d
- **Best scenarios**: no waste of computational power: from $\mathcal{O}(d)$ to $\mathcal{O}(d/K)$.
- **Preview of our results:**

Algorithm	complexity	K	method
Sequential algorithm	$\mathcal{O}(d)$	1	exact
Online Picard	$\mathcal{O}(\sqrt{d})$	$\mathcal{O}(\sqrt{d})$	exact
Approx. Online Picard	$\mathcal{O}(1)$	$\mathcal{O}(d)$	approximate

Bocconi

Picard map for Markov chain simulation

Bocconi

Picard Map

- X is also defined by

$$X_i = X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell)\,, \qquad i = 1, 2, \ldots\,. \tag{2}$$

Picard Map

- X is also defined by

$$X_i = X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell)\,, \qquad\qquad i = 1, 2, \ldots\,. \tag{2}$$

Picard map Φ

$\Phi\colon \mathcal{X}^{K+1} \times \mathcal{W}^K \mapsto \mathcal{X}^{K+1}$ takes as input a trajectory X and outputs a new trajectory $X' = (X'_0, \ldots, X'_K) = \Phi(X, W)$ defined as

$$X'_i = \Phi_i(X, W) = \begin{cases} X_0 & i = 0 \\ X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell) & 0 < i \le K\,. \end{cases}$$

Bocconi

Picard Map

- X is also defined by

$$X_i = X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell)\,, \qquad\qquad i = 1, 2, \ldots\,. \tag{2}$$

Picard map Φ

$\Phi\colon \mathcal{X}^{K+1} \times \mathcal{W}^K \mapsto \mathcal{X}^{K+1}$ takes as input a trajectory X and outputs a new trajectory $X' = (X'_0, \ldots, X'_K) = \Phi(X, W)$ defined as

$$X'_i = \Phi_i(X, W) = \begin{cases} X_0 & i = 0 \\ X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell) & 0 < i \leq K\,. \end{cases}$$

- The K calls to the function f **can be executed in parallel.**

Bocconi

Picard Map

- X is also defined by

$$X_i = X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell)\,, \qquad i = 1, 2, \ldots\,. \tag{2}$$

Picard map Φ

$\Phi\colon \mathcal{X}^{K+1} \times \mathcal{W}^K \mapsto \mathcal{X}^{K+1}$ takes as input a trajectory X and outputs a new trajectory $X' = (X'_0, \ldots, X'_K) = \Phi(X, W)$ defined as

$$X'_i = \Phi_i(X, W) = \begin{cases} X_0 & i = 0 \\ X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell) & 0 < i \leq K\,. \end{cases}$$

- The K calls to the function f **can be executed in parallel.**
- Given $W \in \mathcal{W}^K$,
  - $x \to \Phi(x, W)$ is deterministic.
  - the **fixed point** X satisfying $X = \Phi(X, W)$ is the solution to (2).

Picard Map

- X is also defined by

$$X_i = X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell)\,, \qquad\qquad i = 1, 2, \dots\,. \tag{2}$$

Picard map Φ

$\Phi \colon \mathcal{X}^{K+1} \times \mathcal{W}^K \mapsto \mathcal{X}^{K+1}$ takes as input a trajectory X and outputs a new trajectory $X' = (X'_0, \dots, X'_K) = \Phi(X, W)$ defined as

$$X'_i = \Phi_i(X, W) = \begin{cases} X_0 & i = 0 \\ X_0 + \sum_{\ell=0}^{i-1} f(X_\ell, W_\ell) & 0 < i \leq K\,. \end{cases}$$

- The K calls to the function f **can be executed in parallel.**
- Given $W \in \mathcal{W}^K$,
  - $x \to \Phi(x, W)$ is deterministic.
  - the **fixed point** X satisfying $X = \Phi(X, W)$ is the solution to (2).
- Compute X as the limit of the recursion $X^{(j)} = \Phi(X^{(j-1)}, W)$ for $j = 1, 2, \dots$

Diagram Picard recursion

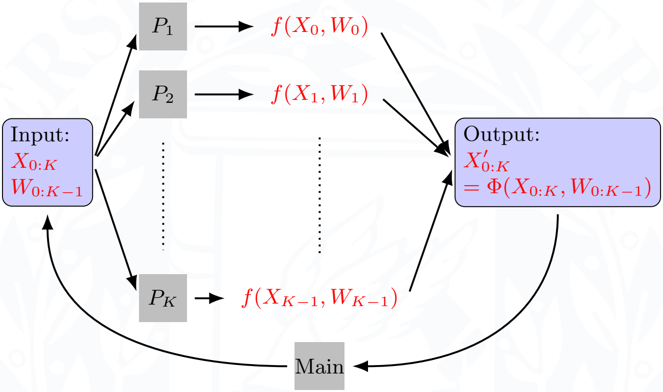


Figure 2: One parallel iteration of Picard recursion

Illustration Picard map

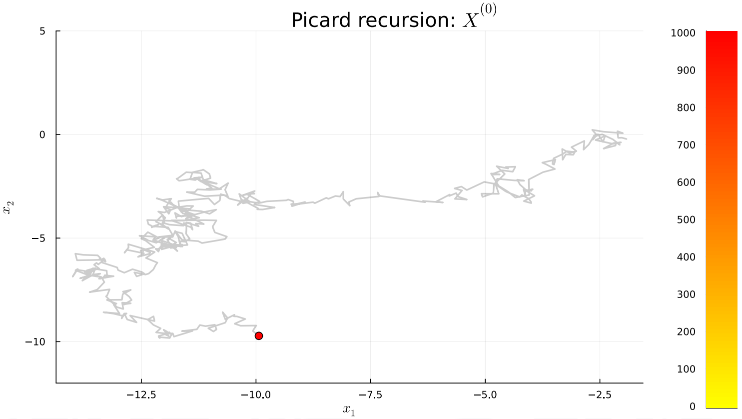


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

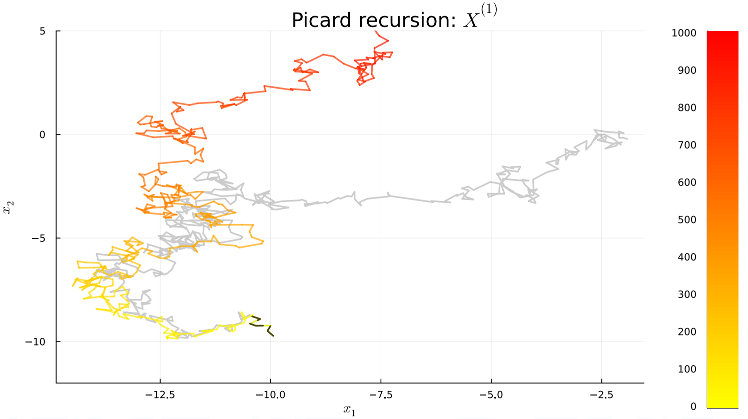


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

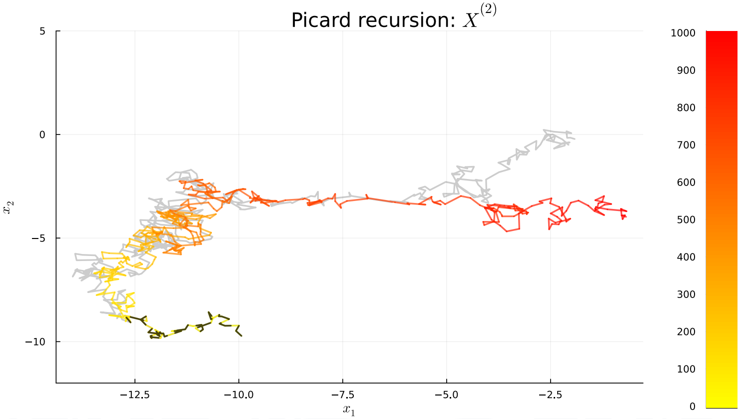


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

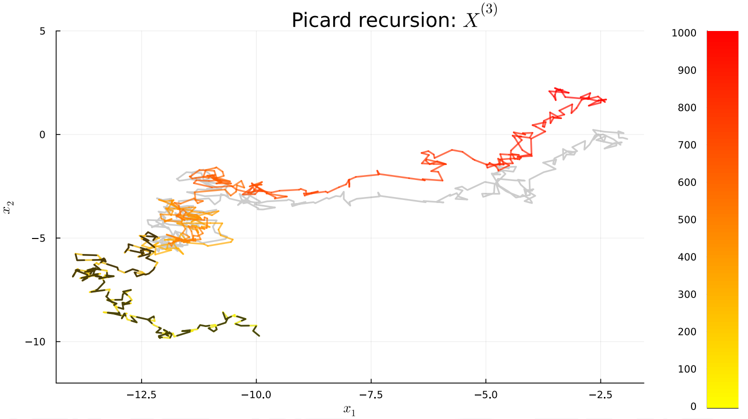


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

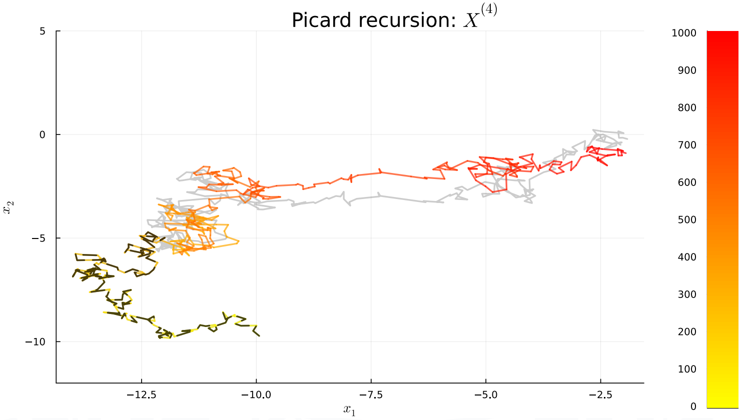


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Bocconi

Illustration Picard map

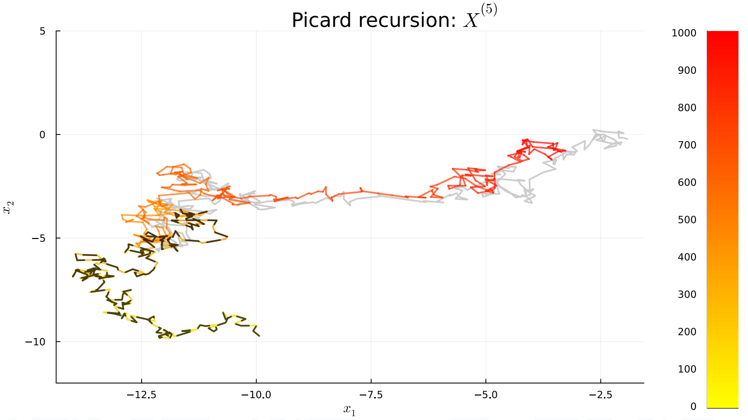


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Bocconi

Illustration Picard map

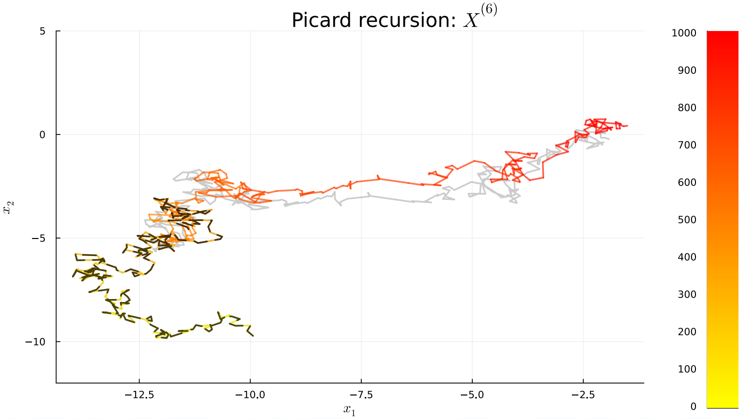


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Bocconi

Illustration Picard map

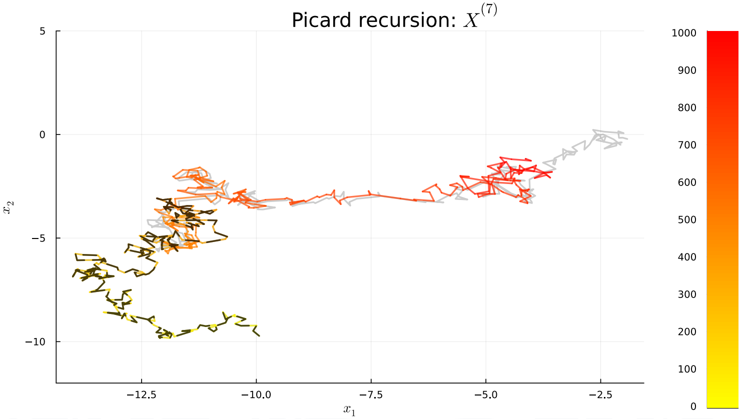


Figure 3: $X_1^{(i)}, X_2^{(i)}, \dots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \dots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

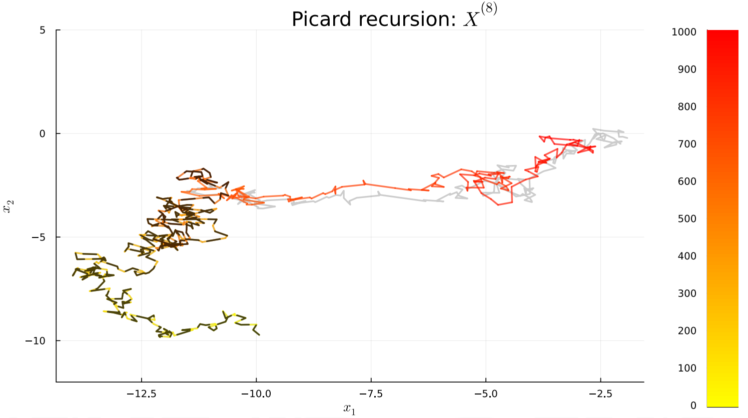


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

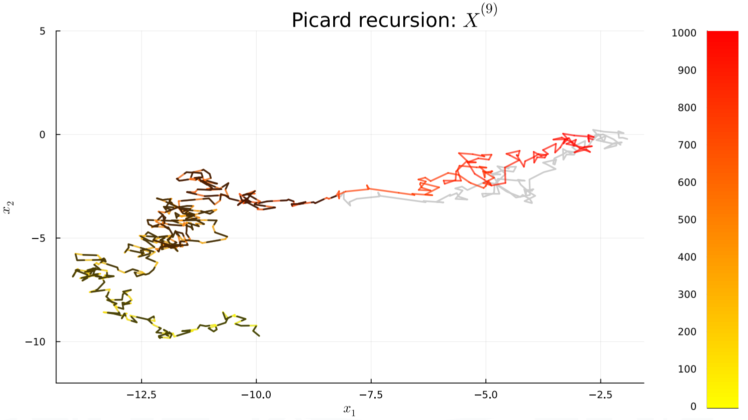


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

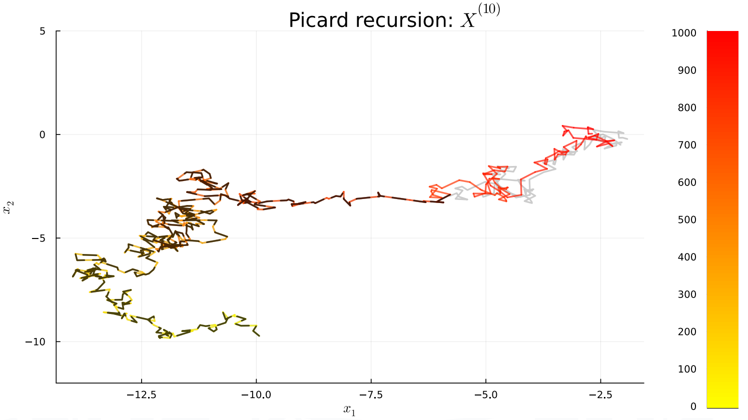


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

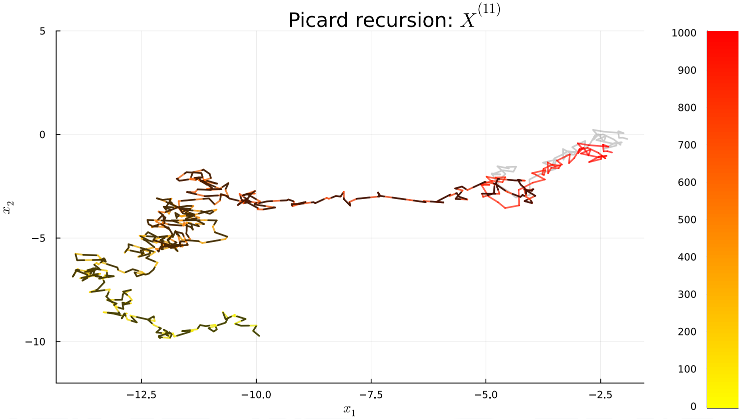


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Illustration Picard map

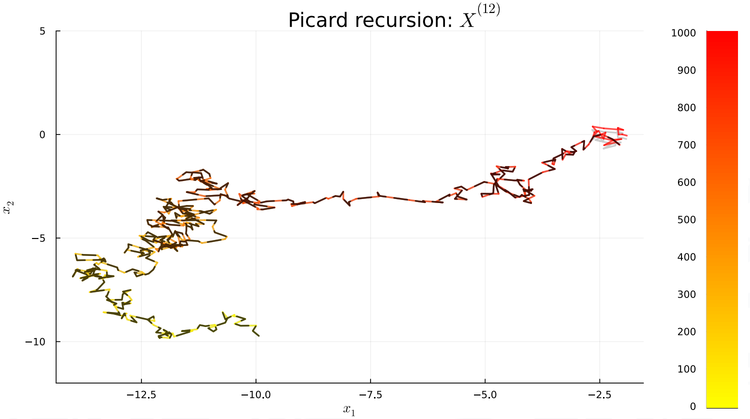


Figure 3: $X_1^{(i)}, X_2^{(i)}, \ldots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \ldots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Bocconi

Illustration Picard map

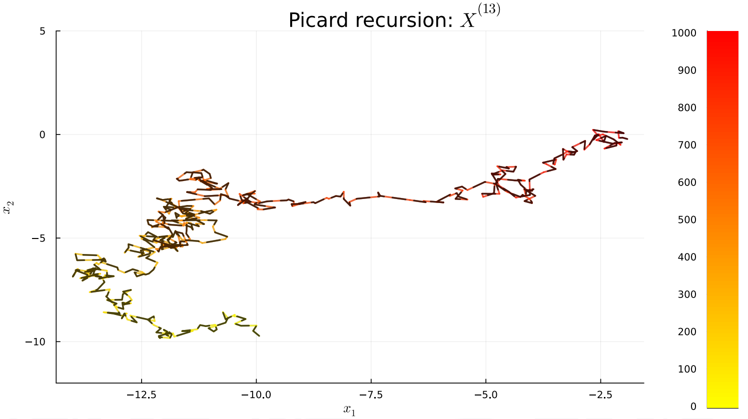


Figure 3: $X_1^{(i)}, X_2^{(i)}, \dots$ of the Picard recursion for $K = 1000$ steps applied to a $d = 100$ dimensional RWM Markov chain. Gray line: Fixed point $X_1, \dots, X_K$. The dashed line corresponds to the part of the trajectory that has converged to its fixed point.

Picard map Φ

- We simulated $K = 1000$ steps of RWM in $j = 13 \ll K$ (parallel) iterations, *trading the use of parallel computing for a shorter run-time.*

Picard map Φ

- We simulated $K = 1000$ steps of RWM in $j = 13 \ll K$ (parallel) iterations, *trading the use of parallel computing for a shorter run-time.*
- The Picard map was previously considered for ODE/PDE:

Model	N/d	$x \to f(x)$ or $x \mapsto f(x, w)$	Method
ODE/PDE	large	smooth	approx.
SDE (ULA, diff. models)	constant	smooth	approx.
RWM	constant	piecewise constant	exact

Picard map Φ

- We simulated $K = 1000$ steps of RWM in $j = 13 \ll K$ (parallel) iterations, *trading the use of parallel computing for a shorter run-time.*
- The Picard map was previously considered for ODE/PDE:

Model	N/d	$x \to f(x)$ or $x \mapsto f(x, w)$	Method
ODE/PDE	large	smooth	approx.
SDE (ULA, diff. models)	constant	smooth	approx.
RWM	constant	piecewise constant	exact

- **Blessing of dimensionality for RWM**: the convergence of Picard for RWM improves in high dimensions as all increments become approximately orthogonal with each other.

Picard map Φ

- We simulated $K = 1000$ steps of RWM in $j = 13 \ll K$ (parallel) iterations, *trading the use of parallel computing for a shorter run-time.*
- The Picard map was previously considered for ODE/PDE:

Model	N/d	$x \to f(x)$ or $x \mapsto f(x, w)$	Method
ODE/PDE	large	smooth	approx.
SDE (ULA, diff. models)	constant	smooth	approx.
RWM	constant	piecewise constant	exact

- **Blessing of dimensionality for RWM**: the convergence of Picard for RWM improves in high dimensions as all increments become approximately orthogonal with each other.
- Piecewise constant $x \mapsto f(x, w)$:
  - The contraction of the Picard map for RWM is **non-standard**.
  - $X \mapsto \Phi(X, W)$ for RWM is **constant in a neighborhood of its fixed point**.
  - The fixed point of Φ can be reached exactly.

Online Picard Algorithm (OPA)

- **Goal**: generalize Picard for N steps of the Markov chain with $K \leq N$ processors.

Online Picard Algorithm (OPA)

- **Goal**: generalize Picard for N steps of the Markov chain with $K \leq N$ processors.

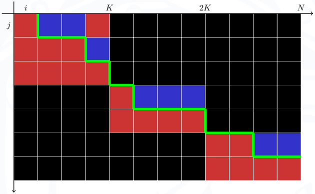


Figure 4: The color of the (j, i) entry represents the state of the ith step: ■ for $f(X_i^{(j)}, W_i) = f(X_i^{(j-1)}, W_i)$ (correct guess), ■ for $f(X_i^{(j)}, W_i) \neq f(X_i^{(j-1)}, W_i)$ (error). ■ where no processor has been allocated. ▬: number of steps simulated according to RWM.

Online Picard Algorithm (OPA)

- **Goal**: generalize Picard for N steps of the Markov chain with $K \leq N$ processors.

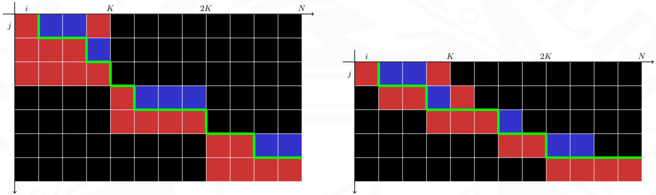


Figure 4: The color of the (j, i) entry represents the state of the ith step: ■ for $f(X_i^{(j)}, W_i) = f(X_i^{(j-1)}, W_i)$ (correct guess), ■ for $f(X_i^{(j)}, W_i) \neq f(X_i^{(j-1)}, W_i)$ (error). ■ where no processor has been allocated. ▬: number of steps simulated according to RWM.

Online Picard Algorithm (OPA)

- **Goal**: generalize Picard for N steps of the Markov chain with $K \leq N$ processors.

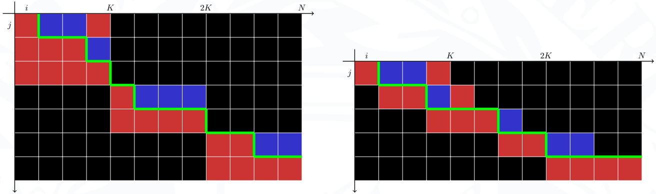


Figure 4: The color of the (j, i) entry represents the state of the ith step: for $f(X_i^{(j)}, W_i) = f(X_i^{(j-1)}, W_i)$ (correct guess), for $f(X_i^{(j)}, W_i) \neq f(X_i^{(j-1)}, W_i)$ (error). where no processor has been allocated. : number of steps simulated according to RWM.

- **Key challenges for analyzing the convergence:**
  - For each (j, i) square: probability of (error) vs (correct guess).

Bocconi

Online Picard Algorithm (OPA)

- **Goal**: generalize Picard for N steps of the Markov chain with $K \leq N$ processors.

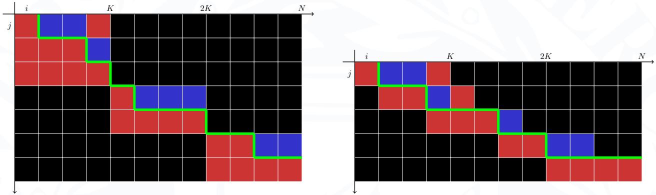


Figure 4: The color of the (j, i) entry represents the state of the ith step: ■ for $f(X_i^{(j)}, W_i) = f(X_i^{(j-1)}, W_i)$ (correct guess), ■ for $f(X_i^{(j)}, W_i) \neq f(X_i^{(j-1)}, W_i)$ (error). ■ where no processor has been allocated. ▬: number of steps simulated according to RWM.

- **Key challenges for analyzing the convergence:**
  - For each (j, i) square: probability of ■ (error) vs ■ (correct guess).
  - For each row j: probability of a strike of $n > 1$ consecutive ■ (or equivalently the probability of the first ■).

Theoretical results

Probability of an error (■)

- Technical assumption: V is L-smooth and Hessian-Lipschitz.

Bocconi

Probability of an error (■)

- Technical assumption: V is L-smooth and Hessian-Lipschitz.

(Simplified) Theorem 1

After $j \geq \log(d)$ steps we have

$$\mathbb{P}(f(X_i^{(j)}, W_i) \neq f(X_i, W_i)) = \mathcal{O}(\frac{i}{d}), \qquad i \leq K.$$

- After $j \geq \log(d)$ iterations, the probability of ■ at square (j, i) is $\mathcal{O}(i/d)$

Bocconi

Probability of an error (■)

- Technical assumption: V is L-smooth and Hessian-Lipschitz.

(Simplified) Theorem 1

After $j \geq \log(d)$ steps we have

$$\mathbb{P}(f(X_i^{(j)}, W_i) \neq f(X_i, W_i)) = \mathcal{O}(\frac{i}{d}), \qquad i \leq K.$$

- After $j \geq \log(d)$ iterations, the probability of ■ at square (j, i) is $\mathcal{O}(i/d)$
- The probability goes to 0 for $d \to \infty$.

Probability of an error (■)

- Technical assumption: V is L-smooth and Hessian-Lipschitz.

(Simplified) Theorem 1

After $j \geq \log(d)$ steps we have

$$\mathbb{P}(f(X_i^{(j)}, W_i) \neq f(X_i, W_i)) = \mathcal{O}(\frac{i}{d}), \qquad i \leq K.$$

- After $j \geq \log(d)$ iterations, the probability of ■ at square (j, i) is $\mathcal{O}(i/d)$
- The probability goes to 0 for $d \to \infty$.
- The probability is controlled only for $K \leq \mathcal{O}(d)$.

Complexity OPA

- $T_{K,N}$: number of iterations of OPA for simulating N steps with K processors

Complexity OPA

- $T_{K,N}$: number of iterations of OPA for simulating N steps with K processors

(Simplified) Theorem 2

For all $N \in \mathbb{N}$, $K = \mathcal{O}(\sqrt{d})$, we have that $T_{K,N} = \mathcal{O}(\frac{N}{K})$ with high probability.

Complexity OPA

- $T_{K,N}$: number of iterations of OPA for simulating N steps with K processors

(Simplified) Theorem 2

For all $N \in \mathbb{N}$, $K = \mathcal{O}(\sqrt{d})$, we have that $T_{K,N} = \mathcal{O}(\frac{N}{K})$ with high probability.

- By the union bound, the first ■ happens at step i with probability $\mathcal{O}(i^2/d)$. $\rightarrow$ $K =$ ~~$\mathcal{O}(d)$~~ $\mathcal{O}(\sqrt{d})$.

Complexity OPA

- $T_{K,N}$: number of iterations of OPA for simulating N steps with K processors

(Simplified) Theorem 2

For all $N \in \mathbb{N}$, $K = \mathcal{O}(\sqrt{d})$, we have that $T_{K,N} = \mathcal{O}(\frac{N}{K})$ with high probability.

- By the union bound, the first ■ happens at step i with probability $\mathcal{O}(i^2/d)$. $\rightarrow$ $K =$ ~~$\mathcal{O}(d)$~~ $\mathcal{O}(\sqrt{d})$.

Corollary 1 (Complexity OPA)

For log-concave distributions, the Online Picard algorithm with $K = \mathcal{O}(\sqrt{d})$ outputs a random variable X, with $\|\mathcal{L}(X) - \pi\|_{\mathrm{TV}} \leq \epsilon$ after

$$J = \mathcal{O}\left(\frac{L}{m}\sqrt{d}\,\mathrm{polylog}(\epsilon^{-1})\right) \quad \text{parallel iterations.}$$

- Corollary 1 was obtained by combining Theorem 2 with known mixing time bounds of RWM (Andrieu et al. 2024)

Approximate OPA (1)

- Bottleneck OPA: even though the probability of ■ at step i is $\mathcal{O}(i/d)$ (Theorem 1), the first ■ happens much earlier at step $i = \mathcal{O}(\sqrt{d})$.

Approximate OPA (1)

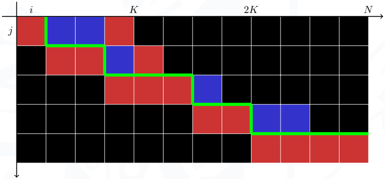
- Bottleneck OPA: even though the probability of ■ at step i is $\mathcal{O}(i/d)$ (Theorem 1), the first ■ happens much earlier at step $i = \mathcal{O}(\sqrt{d})$.
- However, the probability of having at most r-fraction of ■, $r \in (0, 1)$ is $\mathcal{O}(d)$.

Approximate OPA (1)

- Bottleneck OPA: even though the probability of ■ at step i is $\mathcal{O}(i/d)$ (Theorem 1), the first ■ happens much earlier at step $i = \mathcal{O}(\sqrt{d})$.
- However, the probability of having at most r-fraction of ■, $r \in (0, 1)$ is $\mathcal{O}(d)$.
- In the Approximate OPA we tolerates a fixed (and small) percentage $r \in [0, 1]$ of errors:

Approximate OPA (1)

- Bottleneck OPA: even though the probability of ■ at step i is $\mathcal{O}(i/d)$ (Theorem 1), the first ■ happens much earlier at step $i = \mathcal{O}(\sqrt{d})$.
- However, the probability of having at most r-fraction of ■, $r \in (0, 1)$ is $\mathcal{O}(d)$.
- In the Approximate OPA we tolerates a fixed (and small) percentage $r \in [0, 1]$ of errors:



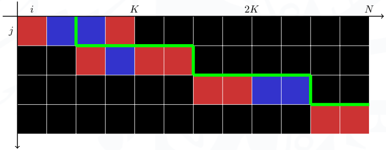


Figure 5: Illustration of OPA (left) vs AOPA with $r = 50\%$ (right). The color of the (j, i) entry represents the state of the ith step: ■ for $f(X_i^{(j)}, W_i) = f(X_i^{(j-1)}, W_i)$ (correct guess), ■ for $f(X_i^{(j)}, W_i) \neq f(X_i^{(j-1)}, W_i)$ (error). ■ where no processor has been allocated.

Approximate OPA (2)

- For $r = 0$, we recover OPA.

Approximate OPA (2)

- For $r = 0$, we recover OPA.
- For at least $(1 - r)$-fraction of increments we have $f(X_i^{(j)}, W_i) = X_{i+1}^{(j)} - X_i^{(j)}$.

Bocconi

Approximate OPA (2)

- For $r = 0$, we recover OPA.
- For at least $(1 - r)$-fraction of increments we have $f(X_i^{(j)}, W_i) = X_{i+1}^{(j)} - X_i^{(j)}$.
- The error is bounded by the size of Gaussian noise for the remaining steps.

Approximate OPA (2)

- For $r = 0$, we recover OPA.
- For at least $(1 - r)$-fraction of increments we have $f(X_i^{(j)}, W_i) = X_{i+1}^{(j)} - X_i^{(j)}$.
- The error is bounded by the size of Gaussian noise for the remaining steps.
- $T_{K,N}^{(r)}$: number of iterations of AOPA with tolerance $r \in (0, 1)$ for simulating N steps with K processors

Approximate OPA (2)

- For $r = 0$, we recover OPA.
- For at least $(1 - r)$-fraction of increments we have $f(X_i^{(j)}, W_i) = X_{i+1}^{(j)} - X_i^{(j)}$.
- The error is bounded by the size of Gaussian noise for the remaining steps.
- $T_{K,N}^{(r)}$: number of iterations of AOPA with tolerance $r \in (0, 1)$ for simulating N steps with K processors

(Simplified) Theorem 3

For all $N \in \mathbb{N}$, $K = \mathcal{O}(d)$, we have that $T_{K,N}^{(r)} = \mathcal{O}(\frac{N}{K})$ with high probability.

Approximate OPA (2)

- For $r = 0$, we recover OPA.
- For at least $(1 - r)$-fraction of increments we have $f(X_i^{(j)}, W_i) = X_{i+1}^{(j)} - X_i^{(j)}$.
- The error is bounded by the size of Gaussian noise for the remaining steps.
- $T_{K,N}^{(r)}$: number of iterations of AOPA with tolerance $r \in (0, 1)$ for simulating N steps with K processors

(Simplified) Theorem 3

For all $N \in \mathbb{N}$, $K = \mathcal{O}(d)$, we have that $T_{K,N}^{(r)} = \mathcal{O}(\frac{N}{K})$ with high probability.

these results suggest a complexity $\mathcal{O}(1)$ with $K = \mathcal{O}(d)$ for AOPA.

Approximate OPA (2)

- For $r = 0$, we recover OPA.
- For at least $(1-r)$-fraction of increments we have $f(X_i^{(j)}, W_i) = X_{i+1}^{(j)} - X_i^{(j)}$.
- The error is bounded by the size of Gaussian noise for the remaining steps.
- $T_{K,N}^{(r)}$: number of iterations of AOPA with tolerance $r \in (0,1)$ for simulating N steps with K processors

(Simplified) Theorem 3

For all $N \in \mathbb{N}$, $K = \mathcal{O}(d)$, we have that $T_{K,N}^{(r)} = \mathcal{O}(\frac{N}{K})$ with high probability.

these results suggest a complexity $\mathcal{O}(1)$ with $K = \mathcal{O}(d)$ for AOPA.

- **More results can be found in the paper**:
  - As $X_0 \to \infty$, the probability of ■ $\to 0$ ($\to$ Instantaneous convergence in the tails).

Approximate OPA (2)

- For $r = 0$, we recover OPA.
- For at least $(1-r)$-fraction of increments we have $f(X_i^{(j)}, W_i) = X_{i+1}^{(j)} - X_i^{(j)}$.
- The error is bounded by the size of Gaussian noise for the remaining steps.
- $T_{K,N}^{(r)}$: number of iterations of AOPA with tolerance $r \in (0,1)$ for simulating N steps with K processors

(Simplified) Theorem 3

For all $N \in \mathbb{N}$, $K = \mathcal{O}(d)$, we have that $T_{K,N}^{(r)} = \mathcal{O}(\frac{N}{K})$ with high probability.

these results suggest a complexity $\mathcal{O}(1)$ with $K = \mathcal{O}(d)$ for AOPA.

- **More results can be found in the paper**:
  - As $X_0 \to \infty$, the probability of ■ $\to 0$ ($\to$ Instantaneous convergence in the tails).
  - All convergence results also apply to Metropolis within Gibbs.

Bocconi

Approximate OPA (2)

- For $r = 0$, we recover OPA.
- For at least $(1 - r)$-fraction of increments we have $f(X_i^{(j)}, W_i) = X_{i+1}^{(j)} - X_i^{(j)}$.
- The error is bounded by the size of Gaussian noise for the remaining steps.
- $T_{K,N}^{(r)}$: number of iterations of AOPA with tolerance $r \in (0, 1)$ for simulating N steps with K processors

(Simplified) Theorem 3

For all $N \in \mathbb{N}$, $K = \mathcal{O}(d)$, we have that $T_{K,N}^{(r)} = \mathcal{O}(\frac{N}{K})$ with high probability.

these results suggest a complexity $\mathcal{O}(1)$ with $K = \mathcal{O}(d)$ for AOPA.

- **More results can be found in the paper**:
  - As $X_0 \to \infty$, the probability of $\blacksquare \to 0$ ($\to$ Instantaneous convergence in the tails).
  - All convergence results also apply to Metropolis within Gibbs.
  - For Metropolis within Gibbs, we have instantaneous convergence for isotropic Gaussian targets, suggesting better performance for well-conditioned targets.

Simulations

Logistic Regression

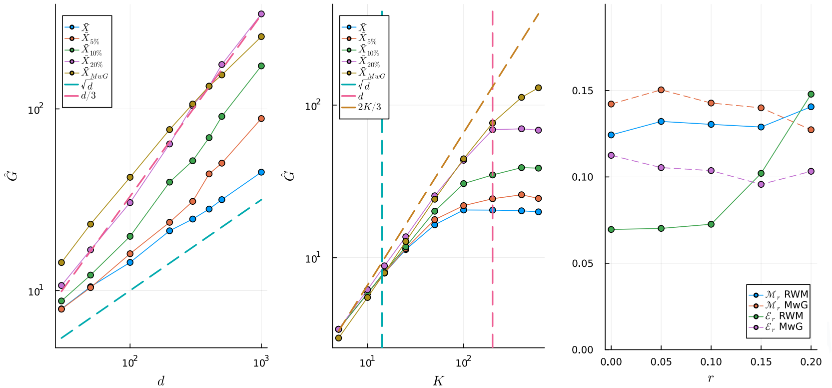


Figure 6: Performance of OPA ($\bar{X}$) and its AOPA ($\bar{X}_r,\ r = 0\%, \ldots, 20\%$) applied to RWM and MwG ($\bar{X}_{MwG}$). Average speedup $\hat{G} = N/T_{K,N}$, $K = d$, $d = 10^2, \ldots, 10^3$ (Left panel) and $d = 200$, $K = 2, 3, \ldots, 10^3$. Right panel: Average error on 1st $(\mathcal{M}_r)$ and 2nd $(\mathcal{E}_r)$ moment estimation for the AOPA with $r = 0\%, \ldots, 20\%$

Bocconi

Conclusion

- **Recap**:

Algorithm	complexity	K	method
Sequential algorithm	$\mathcal{O}(d)$	1	exact
Online Picard	$\mathcal{O}(\sqrt{d})$	$\mathcal{O}(\sqrt{d})$	exact
Approx. Online Picard	$\mathcal{O}(1)$	$\mathcal{O}(d)$	approximate

Bocconi

Conclusion

- **Recap**:

Algorithm	complexity	K	method
Sequential algorithm	$\mathcal{O}(d)$	1	exact
Online Picard	$\mathcal{O}(\sqrt{d})$	$\mathcal{O}(\sqrt{d})$	exact
Approx. Online Picard	$\mathcal{O}(1)$	$\mathcal{O}(d)$	approximate

- The algorithm is **simple**, offering promising directions to **parallelize computations** for Bayesian problems with black-box, expensive models and no access to gradient information.

Bocconi

Conclusion

- **Recap**:

Algorithm	complexity	K	method
Sequential algorithm	$\mathcal{O}(d)$	1	exact
Online Picard	$\mathcal{O}(\sqrt{d})$	$\mathcal{O}(\sqrt{d})$	exact
Approx. Online Picard	$\mathcal{O}(1)$	$\mathcal{O}(d)$	approximate

- The algorithm is **simple**, offering promising directions to **parallelize computations** for Bayesian problems with black-box, expensive models and no access to gradient information.

- **Follow-ups**:
  - Control of the approximation on π of AOPA

Conclusion

- **Recap**:

Algorithm	complexity	K	method
Sequential algorithm	$\mathcal{O}(d)$	1	exact
Online Picard	$\mathcal{O}(\sqrt{d})$	$\mathcal{O}(\sqrt{d})$	exact
Approx. Online Picard	$\mathcal{O}(1)$	$\mathcal{O}(d)$	approximate

- The algorithm is **simple**, offering promising directions to **parallelize computations** for Bayesian problems with black-box, expensive models and no access to gradient information.

- **Follow-ups**:
  - Control of the approximation on π of AOPA
  - Extend our results for other Markov chains with piecewise constant increments $x \mapsto f(x, w)$ such as **Laplace-Hamiltonian** and **Barker Metropolis**.

Conclusion

- **Recap**:

Algorithm	complexity	K	method
Sequential algorithm	$\mathcal{O}(d)$	1	exact
Online Picard	$\mathcal{O}(\sqrt{d})$	$\mathcal{O}(\sqrt{d})$	exact
Approx. Online Picard	$\mathcal{O}(1)$	$\mathcal{O}(d)$	approximate

- The algorithm is **simple**, offering promising directions to **parallelize computations** for Bayesian problems with black-box, expensive models and no access to gradient information.

- **Follow-ups**:
  - Control of the approximation on π of AOPA
  - Extend our results for other Markov chains with piecewise constant increments $x \mapsto f(x, w)$ such as **Laplace-Hamiltonian** and **Barker Metropolis**.
  - Develop more advanced algorithms combining **"cheap" predictions** with **Picard maps** (e.g. Parareal framework)

- Andrieu, Christophe et al. (2024). “Explicit convergence bounds for Metropolis Markov chains: isoperimetry, spectral gaps and profiles”. In: *The Annals of Applied Probability* 34.4, pp. 4022–4071.
- Grazzi, Sebastiano and Giacomo Zanella (2025). *Parallel computations for Metropolis Markov chains with Picard maps*. arXiv: 2506.09762 [stat.CO]. URL: https://arxiv.org/abs/2506.09762.
- Pozza, Francesco and Giacomo Zanella (2024). “On the fundamental limitations of multiproposal Markov chain Monte Carlo algorithms”. In: *arXiv preprint arXiv:2410.23174*.

Contraction of the Picard map

Lemma 1

Under Assumptions 3, for every $x, y \in \mathcal{X}$,

$$\mathbb{P}(f(x, W) \neq f(y, W)) \leq \frac{hL^{1/2}}{d^{1/2}} \left(\sqrt{\frac{2}{\pi}} + \frac{h\gamma}{2} \right) \|x - y\|, \qquad W \sim \nu.$$

Lemma 2

Under Assumption 2 and for all $x, y \in \mathcal{X}^{K+1}$ with $x_0 = y_0, w_0 \in \mathcal{W}$ and $1 < i \leq d$,

$$\mathbb{E}[\|\Phi_i(x, W) - \Phi_i(y, W)\|^2] \leq \frac{15h^2}{L} \sum_{\ell=1}^{i-1} \left(\mathbb{P}(f(x_\ell, W_\ell) \neq f(y_\ell, W_\ell)) + \delta(d)\right).$$

Bocconi

Lemma 1

Under Assumptions 3, for every $x, y \in \mathcal{X}$,

$$\mathbb{P}(f(x,W) \neq f(y,W)) \leq \frac{hL^{1/2}}{d^{1/2}} \left(\sqrt{\frac{2}{\pi}} + \frac{h\gamma}{2} \right) \|x - y\|, \qquad W \sim \nu.$$

Lemma 2

Under Assumption 2 and for all $x, y \in \mathcal{X}^{K+1}$ with $x_0 = y_0, w_0 \in \mathcal{W}$ and $1 < i \leq d$,

$$\mathbb{E}[\|\Phi_i(x,W) - \Phi_i(y,W)\|^2] \leq \frac{15h^2}{L} \sum_{\ell=1}^{i-1} \left(\mathbb{P}(f(x_\ell, W_\ell) \neq f(y_\ell, W_\ell)) + \delta(d) \right).$$

Lemma 3

Let $A^{(j)} = \max_{\ell \leq i} \mathbb{P}(f(X_\ell^{(j)}, W_\ell) \neq f(X_\ell, W_\ell)), j \in \{0, 1, \dots\}$. Under Assumption 2, we have

$$A^{(j+1)} \leq \sqrt{c_0 \frac{i}{d} \left(A^{(j)} + \delta(d) \right)}. \tag{3}$$

Bocconi

Lemma 1

Under Assumptions 3, for every $x, y \in \mathcal{X}$,

$$\mathbb{P}(f(x,W) \neq f(y,W)) \leq \frac{hL^{1/2}}{d^{1/2}} \left(\sqrt{\frac{2}{\pi}} + \frac{h\gamma}{2} \right) \|x - y\|, \qquad W \sim \nu.$$

Lemma 2

Under Assumption 2 and for all $x, y \in \mathcal{X}^{K+1}$ with $x_0 = y_0, w_0 \in \mathcal{W}$ and $1 < i \leq d$,

$$\mathbb{E}[\|\Phi_i(x,W) - \Phi_i(y,W)\|^2] \leq \frac{15h^2}{L} \sum_{\ell=1}^{i-1} (\mathbb{P}(f(x_\ell, W_\ell) \neq f(y_\ell, W_\ell)) + \delta(d)) .$$

Lemma 3

Let $A^{(j)} = \max_{\ell \leq i} \mathbb{P}(f(X_\ell^{(j)}, W_\ell) \neq f(X_\ell, W_\ell)), j \in \{0, 1, \dots\}$. Under Assumption 2, we have

$$A^{(j+1)} \leq \sqrt{c_0 \frac{i}{d} \left(A^{(j)} + \delta(d)\right)}. \tag{3}$$

Lemma 4

Let $(a_j)_{j=0,1,\dots}$ be a non-negative sequence satisfying $a_0 = 1$ and $a_{j+1} \leq b\sqrt{a_j + \epsilon}$ for some fixed $b, \epsilon > 0$ and all $j \geq 0$. Then $a_j \leq b^2 + \epsilon + 2^{-j}$ for all $j \geq 0$.

Lemma 1

Under Assumptions 3, for every $x, y \in \mathcal{X}$,

$$\mathbb{P}(f(x, W) \neq f(y, W)) \leq \frac{hL^{1/2}}{d^{1/2}} \left(\sqrt{\frac{2}{\pi}} + \frac{h\gamma}{2} \right) \|x - y\|, \qquad W \sim \nu.$$

Lemma 2

Under Assumption 2 and for all $x, y \in \mathcal{X}^{K+1}$ with $x_0 = y_0, w_0 \in \mathcal{W}$ and $1 < i \leq d$,

$$\mathbb{E}[\|\Phi_i(x, W) - \Phi_i(y, W)\|^2] \leq \frac{15h^2}{L} \sum_{\ell=1}^{i-1} (\mathbb{P}(f(x_\ell, W_\ell) \neq f(y_\ell, W_\ell)) + \delta(d)) .$$

Lemma 3

Let $A^{(j)} = \max_{\ell \leq i} \mathbb{P}(f(X_\ell^{(j)}, W_\ell) \neq f(X_\ell, W_\ell)), j \in \{0, 1, \dots\}$. Under Assumption 2, we have

$$A^{(j+1)} \leq \sqrt{c_0 \frac{i}{d} \left(A^{(j)} + \delta(d) \right)}. \tag{3}$$

Lemma 4

Let $(a_j)_{j=0,1,\dots}$ be a non-negative sequence satisfying $a_0 = 1$ and $a_{j+1} \leq b\sqrt{a_j + \epsilon}$ for some fixed $b, \epsilon > 0$ and all $j \geq 0$. Then $a_j \leq b^2 + \epsilon + 2^{-j}$ for all $j \geq 0$.

Lemma 3 and 4 implies Theorem 1, i.e.

$$\mathbb{P}(f(X_i^{(j)}, W_i) \neq f(X_i, W_i) \mid X_0 = x_0, W_0 = w_0) \leq c_0 \frac{i}{d} + \delta(d) + 2^{-j}$$

Bocconi