MCMC in the Cloud

A parallelised implementation of the Metropolis-Hastings algorithm

Presented by :
Youssef Ghoname
Lynda Djellouli
Arel Akaunu

Introduction

Problem:

Slow MCMC for complex models (e.g. cosmology, logistic regression)

Solution:

Parallel MCMC using ensemble samplers

Relevance:

Cloud computing enables fast sampling for expensive posteriors

Motivation for Parallelisation

Bayesian inference requires sampling from complex posteriors

Traditional Metropolis-Hastings is sequential and computationally expensive

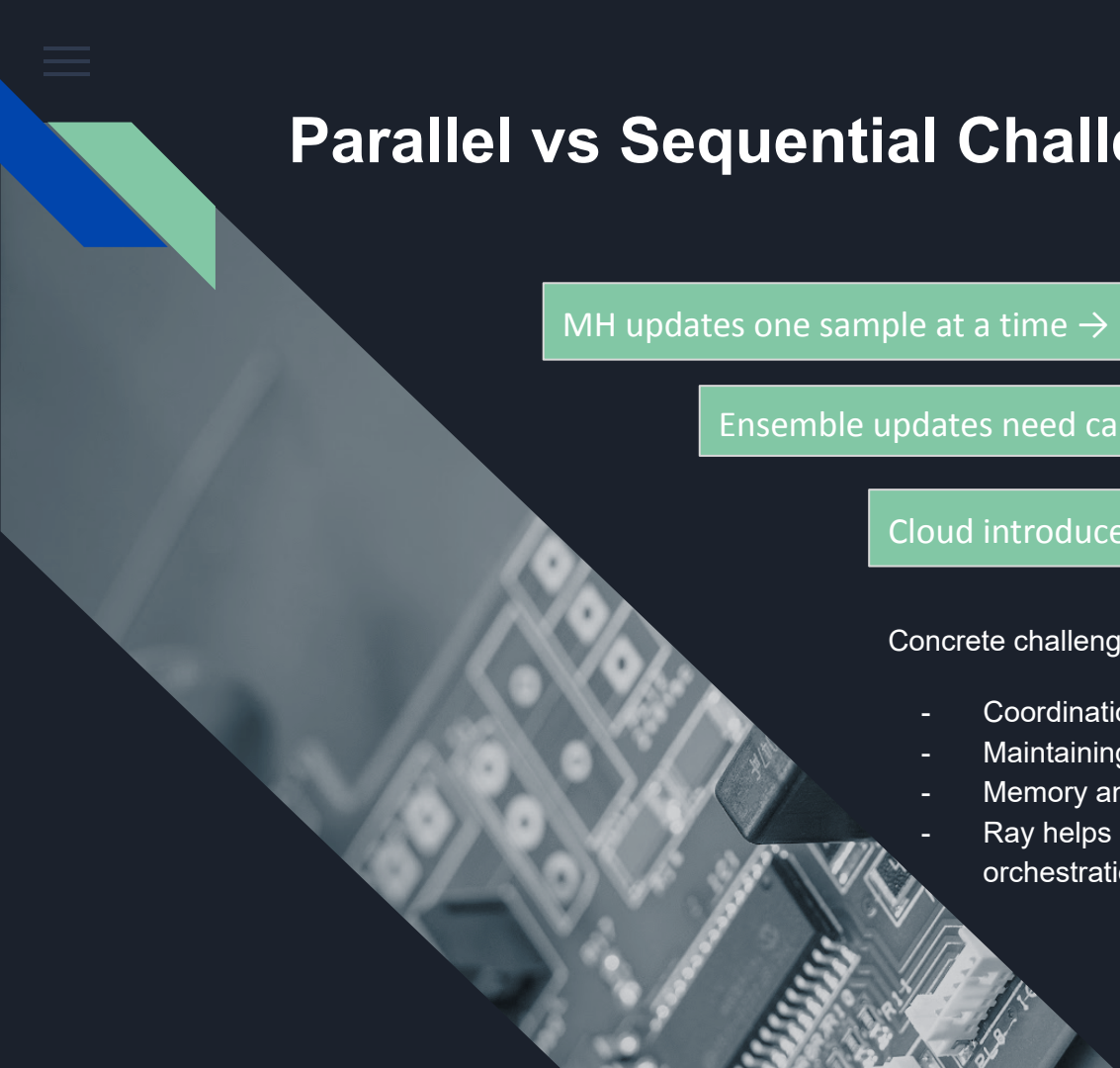
Goodman & Weare ensemble allows parallel updates

Cloud environments provide the needed scale

Implementation Overview

1. Sequential stretch move (Goodman & Weare)
2. Foreman-Mackey's emcee with two-group update
3. Ray parallelism: many chains with different seeds
4. CosmoHammer-style architecture: modular and cloud-friendly
5. Mock cosmological model + real logistic regression

Parallel vs Sequential Challenges



MH updates one sample at a time → hard to parallelisec

Ensemble updates need careful coordination (stretch move)

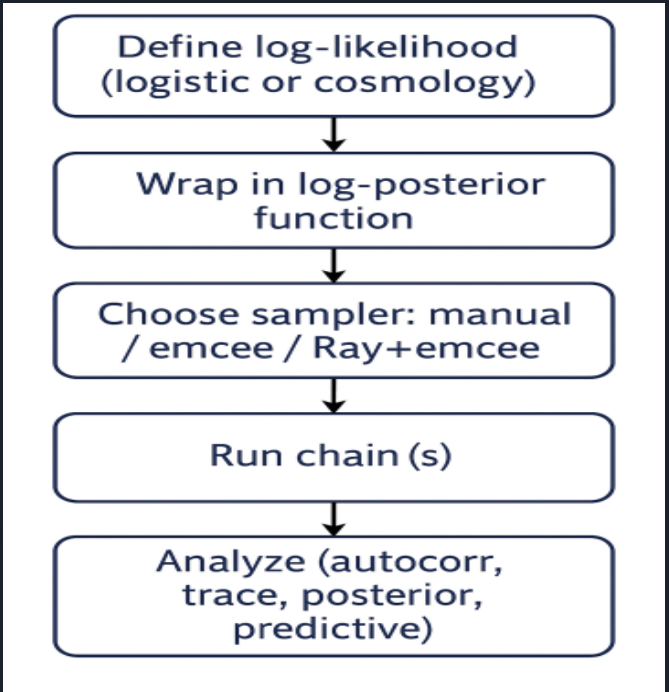
Cloud introduces new challenges (e.g. process failures)

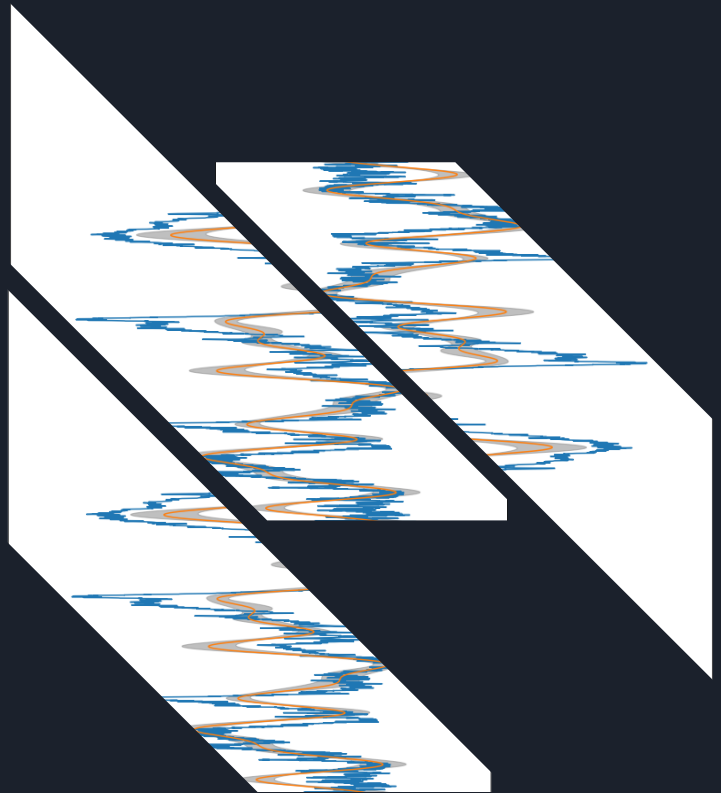
Concrete challenges:

- Coordination between walkers in ensemble sampling
- Maintaining detailed balance in parallel updates
- Memory and process management in cloud environments
- Ray helps abstract these challenges, but overhead and orchestration matter

Code Architecture

Diagram of pipeline

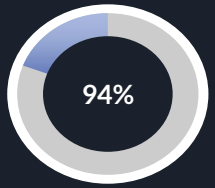




Results & Discussion

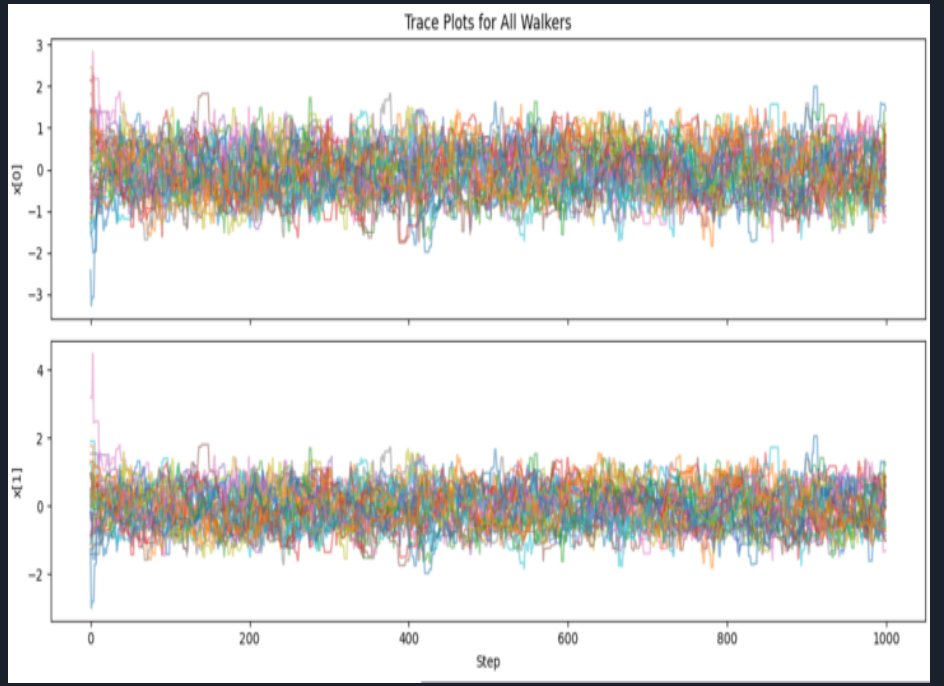
1. Posterior Predictive Accuracy

- Sampled from the posterior
- Predicted on a new test set
- Averaged predictions to compute accuracy



2. Trace Plot

- Display the sampled values of each parameter over time.
- The trace plots demonstrate good mixing and convergence.
  - Stable fluctuations
  - Fast mixing (no sticking)
  - No major drifts or trends



Results & Discussion

3. Autocorrelation Time & Acceptance Fraction

Chain	Autocorrelation Time (τ)	Acceptance Fraction
0	[35.57, 34.68, 30.16]	0.646
1	[34.63, 34.53, 34.92]	0.645
2	[38.47, 34.42, 38.20]	0.647
3	[35.43, 34.73, 34.26]	0.644

Used `sampler.get_autocorr_time()` and `sampler.acceptance_fraction` for quantitative diagnostics:

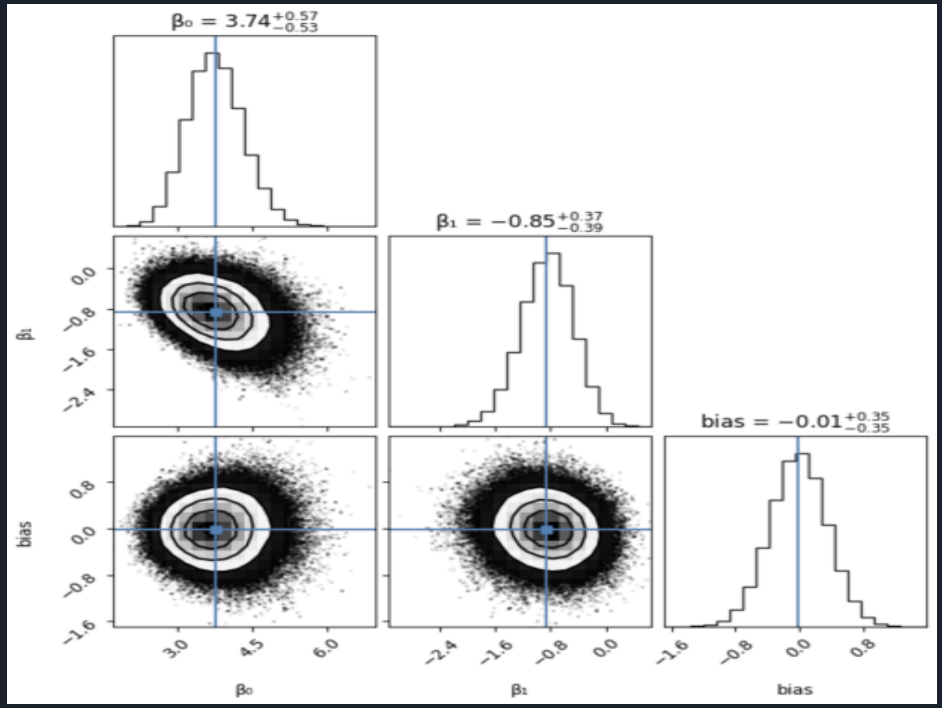
- All acceptance rates are within the optimal range (0.2–0.6, with ensemble samplers tolerating up to ~0.7).
- Autocorrelation times are moderate, indicating fairly efficient sampling with good mixing.

Results & Discussion

4. Corner plot and posterior stats (β_0, β_1, bias)

The corner plot shows :

- Joint and marginal distributions of β_0, β_1, and bias
- Median values and 95% credible intervals



Printed summary:

β_0: mean=3.761, std=0.554, 95% CI=[2.744, 4.912]

β_1: mean=-0.858, std=0.381, 95% CI=[-1.636, -0.141]

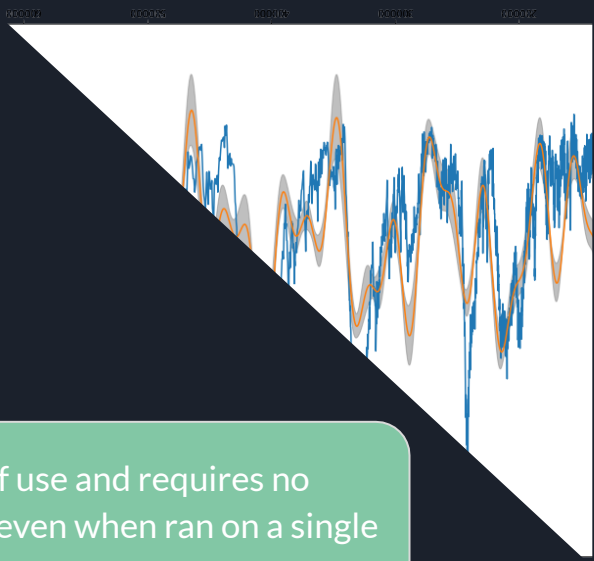
bias: mean=-0.010, std=0.357, 95% CI=[-0.716, 0.690]

Results & Discussion

5. Comparison table: sequential vs Ray-parallel

Method	Parallelism	Tuning Required	Runtime	Predictive Accuracy
Goodman & Waere	NO	NO	MEDIUM	Not measured
Foreman-Mackey (EMCEE)	LIMITED	NO	FAST	93-94%
Akeret (Ray + EMCEE)	YES	NO	FASTER	94%

Results & Discussion



Sequential sampling with Goodman & Weare required ~10× more runtime for same convergence → Parallel version 10–20x faster

Emcee improves ease of use and requires no tuning; highly scalable, even when ran on a single core

Ray enables fully independent chains with shared configuration, scalable to any cloud setup which is a good match with Akeret et al.'s goals

Cloud parallelism pays off if likelihood is expensive (like CAMB)

Results & Discussion

Sequential vs. Parallel

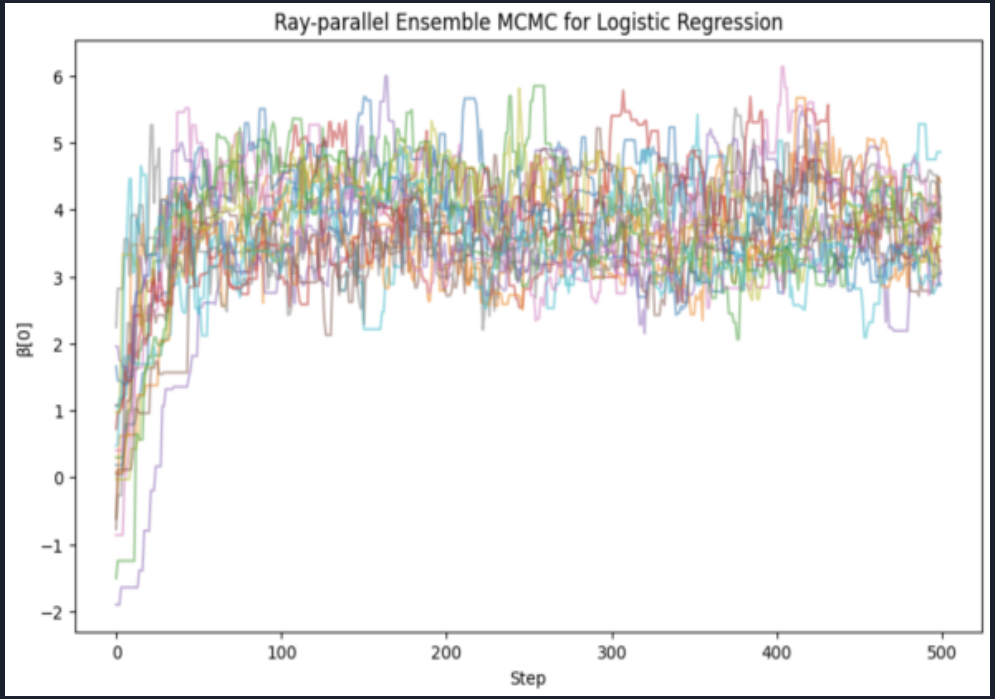
Method	Runtime	Dataset	Notes
Sequential	1.28 sec	Small Gaussian	Runs fast due to low complexity
Parallel (+ Ray)	18.18 sec	Small Gaussian	Overhead dominates; slower overall

On small, low-cost datasets (e.g., 2D Gaussian), the sequential implementation of Goodman & Weare completes faster than the parallel version. This is because Ray incurs setup and inter-process communication overheads that outweigh the benefits of parallelism when the likelihood is computationally cheap.

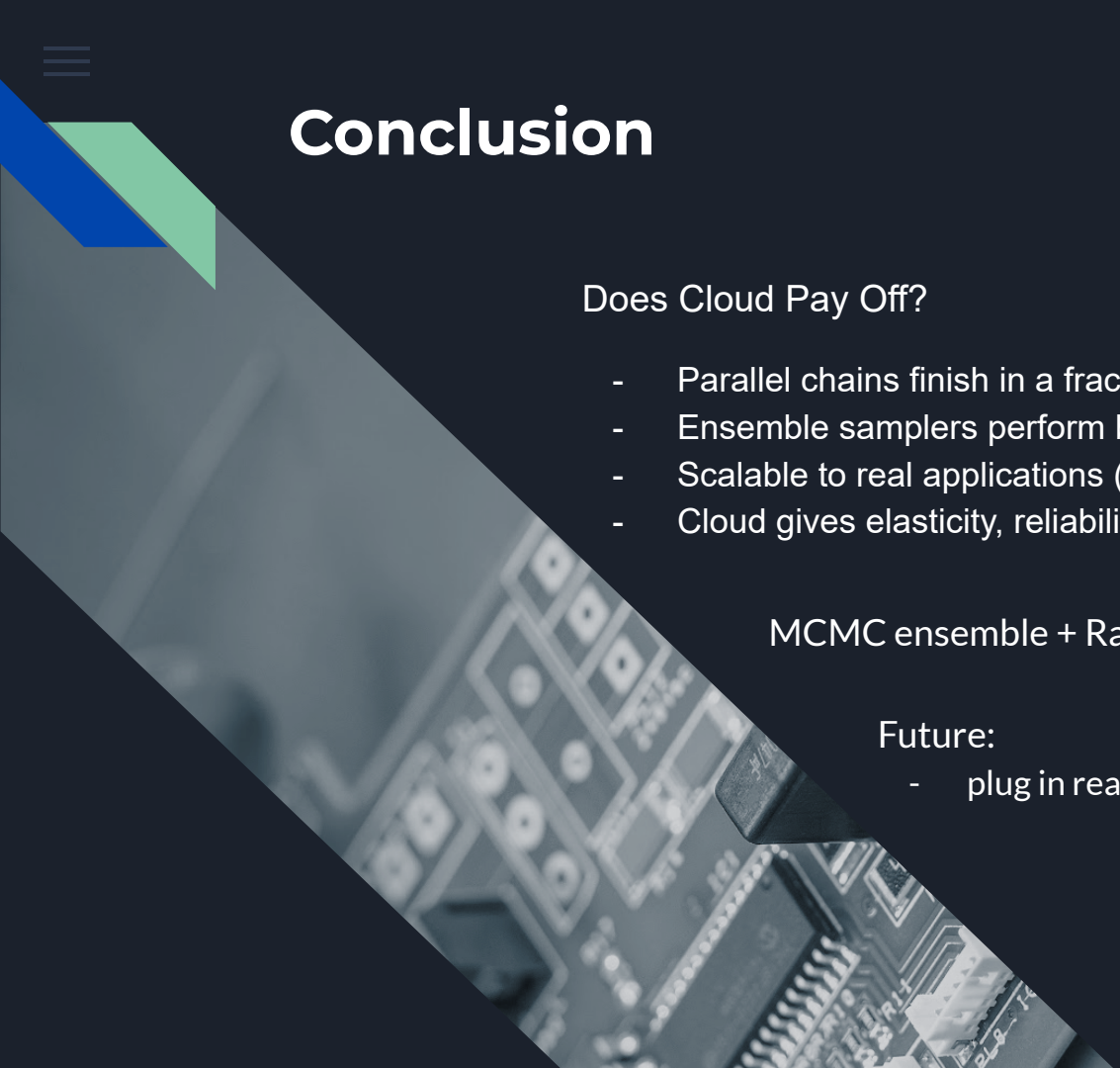
Results & Discussion

Sequential vs. Parallel

This trace plot of β_0 across all parallel walkers shows convergence after roughly 50–100 steps. The walkers mix well, and the chain stabilizes, confirming good sampling behavior even under parallel execution using Ray.



Conclusion



Does Cloud Pay Off?

- Parallel chains finish in a fraction of time
- Ensemble samplers perform better than MH without tuning
- Scalable to real applications (e.g. CAMB likelihoods)
- Cloud gives elasticity, reliability, and reproducibility

MCMC ensemble + Ray = scalable, cloud-ready Bayesian inference

Future:

- plug in real cosmological models or GPU-enabled backends