Faster Parallel MCMC: Metropolis Adjustment Is Best Served Warm

Jakob Robnik
Physics Department,
University of California at Berkeley,
Berkeley, CA 94720, USA

Uroš Seljak
Physics Department,
University of California at Berkeley
and Lawrence Berkeley National Laboratory,
Berkeley, CA 94720, USA

Abstract

Despite the enormous success of Hamiltonian Monte Carlo and related Markov Chain Monte Carlo (MCMC) methods, sampling often still represents the computational bottleneck in scientific applications. Availability of parallel resources can significantly speed up MCMC inference by running a large number of chains in parallel, each collecting a single sample. However, the parallel approach converges slowly if the chains are not initialized close to the target distribution (cold start). Theoretically this can be resolved by initially running MCMC without Metropolis-Hastings adjustment to quickly converge to the vicinity of the target distribution and then turn on adjustment to achieve fine convergence. However, no practical scheme uses this strategy, due to the difficulty of automatically selecting the step size during the unadjusted phase. We here develop Late Adjusted Parallel Sampler (LAPS), which is precisely such a scheme and is applicable out of the box. LAPS takes advantage of ensemble-based hyperparameter adaptation to estimate the bias at each iteration and converts it to the appropriate step size. We show that LAPS consistently and significantly outperforms ensemble adjusted methods such as MEADS or ChESS and the optimization-based initializer Pathfinder on a variety of standard benchmark problems. LAPS typically achieves two orders of magnitude lower wall-clock time than the corresponding sequential algorithms such as NUTS.

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300. Copyright 2026 by the author(s).

1 INTRODUCTION

Markov Chain Monte Carlo (MCMC) methods (Metropolis et al., 1953) are a workhorse of Bayesian statistical inference, with applications ranging from from economics and social science Gelman et al. (1995), to high energy physics (Duane et al., 1987) and machine learning (Neal, 1996). They are also widely employed for performing high-dimensional integrals in quantum physics (Gattringer and Lang, 2010), molecular chemistry (Leimkuhler and Matthews, 2015), statistical mechanics, and in many other fields. In all of these applications, the goal of MCMC is to estimate expectation values $\mathbb{E}_p[f] = \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$, where $p(\boldsymbol{x})$ is a probability density distribution of parameters $\boldsymbol{x} \in \mathbb{R}^d$. This is achieved by generating M samples $\{\boldsymbol{x}^m\}_{m=1}^M$ from the probability density distribution $p(\boldsymbol{x})$ and to estimate expectation values of interest as $\mathbb{E}_p[f(\boldsymbol{x})] \approx \frac{1}{M}\sum_{m=1}^M f(\boldsymbol{x}^m)$. We focus on situation where a (smooth) gradient of the target distribution $\nabla \log p(\boldsymbol{x})$ is available, either analytically, or via automatic differentiation (Griewank and Walther, 2008). This is often the case in fields like Bayesian statistics (Štrumbelj et al., 2024; Carpenter et al., 2017), machine learning (Baydin et al., 2018), lattice quantum problems (Gattringer and Lang, 2010), and cosmology (Campagne et al., 2023).

Traditionally, MCMC is run in the sequential mode, where one (or a few) long chains are used to generate samples (Margossian and Gelman, 2024). The chain first undergoes a burn-in phase where it converges from the initial state to the stationary distribution. After the burn-in, the samples are collected and the expectation value error decays as the square root of the effective sample size, which is the total number of samples, reduced by the sample correlation length (Gelman et al., 1995). However, the sampling phase can be very long in wall-clock time for expensive likelihoods or complicated geometries, which is not practical and can even be prohibitive for a typical workflow.

Parallel sampling. Following the recent computing advances enabling accessibility of parallel resources, such as GPUs, TPUs and CPU clusters, a new paradigm has emerged (Sountsov et al., 2024; Hoffman and Sountsov, 2022; Lao et al., 2020): instead of running a few very long chains, where each chain produces many samples, run a large number of chains in parallel, such that each chain only collects one sample. This strategy is known as ensemble MCMC (Hoffman and Sountsov, 2022; Goodman and Weare, 2010) or many-short-chains regime (Margossian et al., 2024). In the ensemble approach, it is essentially only the burn-in phase that matters; after it finishes, the distribution of the chain positions is already the stationary distribution, and the error is only determined by the number of chains. However, this approach is sensitive to the initialization of the chains, and can converge slowly if the chains are not initialized close to the target distribution (Durmus and Eberle, 2023).

Late adjustment. It is theoretically well-understood (Altschuler and Chewi, 2024; Durmus and Eberle, 2023) that Metropolis-Hastings (MH) adjusted MCMC excels when the initial distribution is close to the target distribution (warm start), but suffers if the initial distribution is far from the stationary distribution (cold start), because the acceptance probability is significantly lower away from the typical set (Durmus and Eberle, 2023). Unadjusted methods (no MH accept-reject step) are complementary in this regard: they are fast to converge from a cold start to the vicinity of the stationary distribution, but struggle to achieve fine convergence, since they are biased and need to control the bias by the small step size (Bou-Rabee and Schuh, 2023; Bou-Rabee and Eberle, 2023; Camrud et al., 2024). One can take the best of both worlds by using an unadjusted method to converge from the cold start to the warm start and then use an adjusted method to achieve fine convergence (Durmus and Eberle, 2023; Altschuler and Chewi, 2024). This method gives theoretically best mixing time to high accuracy from a cold start to date. However, these developments are theoretical and assume global information about the target that is not available in practice, so it is not immediately clear how to implement this algorithm and in particular, how to select its step size during the unadjusted phase. This will be the focus of the present work.

Our contributions. We develop a first practical parallel algorithm that initially performs unadjusted sampling and then uses MH adjustment to achieve fine convergence. The resulting algorithm, termed Late-Adjusted Parallel Sampler (LAPS) can be applied out of the box, the user only needs to provide the target density and the initialization of the chains. We show on standard benchmark problems that it significantly outperforms other turnkey ensemble samplers such as ChESS-HMC (Hoffman et al., 2021) or MEADS (Hoffman and Sountsov, 2022), in some cases by more than an order of magnitude. It also achieves two orders of magnitude faster convergence than sequential algorithms such as NUTS (Hoffman and Gelman, 2014) or MCLMC (Robnik et al., 2024b). The algorithm is implemented in JAX (Schoenholz et al., 2020), specifically in Blackjax (Cabezas et al., 2024) and comes with tutorials on how to use it[1][2]. Blackjax can easily be combined with models written in probabilistic programming languages such as Numpyro (Phan et al., 2019) or Tensorflow Probability (Dillon et al., 2017).

2 BACKGROUND

Parallel MCMC. The strategy is to evolve an ensemble of M chains. Let $(\boldsymbol{x}_t^m, \boldsymbol{u}_t^m)$ be the state of m-th chain at iteration t, where $\boldsymbol{x}_t^m$ is the position in the parameter space and $\boldsymbol{u}_t^m$ is the velocity. Further let $\rho_t(\boldsymbol{x}, \boldsymbol{u})$ be the underlying ensemble probability density distribution, that is, $(\boldsymbol{x}_t^m, \boldsymbol{u}_t^m)$ are exact samples from $\rho_t(\boldsymbol{x}, \boldsymbol{u})$. Each chain is an independent Markov process,

$$(\boldsymbol{x}_{t+1}^m, \boldsymbol{u}_{t+1}^m) = \varphi(\boldsymbol{x}_t^m, \boldsymbol{u}_t^m | \beta_t), \tag{1}$$

where φ is an update kernel and β_t are hyperparameters of the kernel. These updates can be applied in parallel due to independence.

Choice of the kernel. State-of-the-art kernels for sampling from differentiable distributions include Hamiltonian Monte Carlo (HMC; Duane et al. (1987); Neal (2011)) and underdamped Langevin Monte Carlo (LMC; Leimkuhler and Matthews (2015)). In both, the velocity (or equivalently momentum) is enforced to have a standard Gaussian marginal. Recently, Microcanonical Langevin Monte Carlo (MCLMC; Robnik et al. (2024b); Robnik and Seljak (2024)) has been proposed, where the norm of the velocity is instead kept fixed. The proposed benefit is increased stability to large gradients and faster convergence, due to the more deterministic dynamics. Indeed, large improvements over the state-of-the art have also been observed in practice (Simon-Onfroy et al., 2025; Crespi et al., 2025; Sommer et al., 2024). Speed of convergence is of particular interest in the present work, so we will adopt MCLMC, rather than the more standard HMC or LMC. We perform an ablation study in Appendix D. In

[1] Blackjax: `https://blackjax-devs.github.io/sampling-book/algorithms/laps.html`.

[2] Code for reproducing the results is also public: `https://github.com/reubenharry/sampler-benchmarks`

ideal, continuous-time MCLMC, the kernel would be chosen in such a way that $(\boldsymbol{x}_s, \boldsymbol{u}_s) = \varphi(\boldsymbol{x}_0, \boldsymbol{u}_0 | s, \eta)$ exactly solves the following stochastic differential equation (SDE):

$$d\boldsymbol{x}_s = \boldsymbol{u}_s \, ds, \tag{2}$$
$$d\boldsymbol{u}_s = (I - \boldsymbol{u}_s \boldsymbol{u}_s^T)\Big(\frac{\nabla \log p(\boldsymbol{x}_s)}{d-1} \, ds + \eta \, d\boldsymbol{W}\Big)$$

with initial condition $(\boldsymbol{x}_0, \boldsymbol{u}_0)$. The ensemble distribution ρ_t then provably converges to the target distribution (Robnik and Seljak, 2024). Here, $\boldsymbol{W}$ is the Wiener process (Øksendal, 2003) and η is a hyperparameter that determines the strength of stochasticity. In the discretized version of Equation (2) (see Equation (23)) we will parameterize the strength of stochasticity by a parameter L rather than η, such that $\eta = \sqrt{2/Ld}$ in the limit of small step size. This is a convenient parameterization because it makes L directly comparable with with the trajectory length in HMC (Robnik and Seljak, 2024).

Metropolis test. In practice, we have to numerically construct a kernel that only approximately solves Equation (2) by a splitting scheme (Robnik and Seljak, 2024), as reviewed in Appendix E. One then has two ways of dealing with the approximation error:

- **Unadjusted:** use this approximate solution as a kernel $\varphi(\cdot|\epsilon, L)$ and ensure that the step size ϵ is sufficiently small to make the approximation error tolerable.
- **Adjusted:** construct the kernel by performing N steps with the approximate solution $\varphi(\cdot|\epsilon, L)$ and wrap them by a Metropolis-Hastings accept/reject step, followed by refreshing the velocity from its marginal distribution (Robnik et al., 2025). This construction is reviewed in Appendix F. It guarantees that the ensemble exactly converges to the target distribution.

Unadjusted approach is faster to converge from the cold start to the warm start, while the adjusted approach is faster to achieve fine convergence. LAPS combines the best of both worlds, by switching on adjustment once the unadjusted sampling stops making sufficient progress.

Hyperparameter adaptation. In either case, the kernel has two hyperparameters, the step size ϵ and the effective trajectory length L. In adjusted phase, the number of steps per proposal N is an additional hyperparameter. To tune those parameters automatically we will exploit Ensemble Chain Adaptation (ECA; Gilks et al. (1994)), the idea that the ensemble averages at iteration t,

$$\mathbb{E}_{\rho_t}[f] \approx \frac{1}{M} \sum_{m=1}^{M} f(\boldsymbol{x}_t^m), \tag{3}$$

can be used to inform the hyperparameters in iteration $t+1$. This breaks the complete independence of the chains and introduces an asymptotic bias, which is small if the number of chains is large. To eliminate this bias one can either adopt a scheme from (Hoffman and Sountsov, 2022) or freeze the hyperparameter tuning at some point, as we do here. Several ECA algorithms have been developed for HMC or LMC, among these ChEES-HMC (Hoffman et al., 2021), SNAPER (Sountsov and Hoffman, 2022), MEADS (Hoffman and Sountsov, 2022) and MALT (Riou-Durand et al., 2023). Another option is NUTS (Hoffman and Gelman, 2014) run in a parallel mode, where each chain is run independently. These samplers are Metropolis adjusted during the entire process and are typically initializd with an optimizer. A popular choice is the Pathfinder algorithm (Zhang et al., 2022), an L-BFGS-based optimizer that is designed to terminate before the optimizer would start moving away from the typical set and towards the mode of the distribution. Our main novelty is that we initialize with an unadjusted sampler.

3 UNADJUSTED INITIALIZATION

We here develop a scheme to determine the hyperparameters during the unadjusted phase of the algorithm. We emphasize that the purpose of this stage is not to guarantee convergence to the target distribution, but only to approach it as fast as possible.

3.1 Step size

Stationary distribution p_ϵ of the unadjusted kernel $\varphi(\cdot|\epsilon, L)$ differs from the target distribution p. This is called the asymptotic bias. With larger step size we are moving towards the stationary distribution faster, but also the asymptotic bias is larger. We expect the optimal step size to be time-dependent: when we are further away from stationarity, the asymptotic bias is less important, and the advantage of making larger steps prevails. Closer to the stationary distribution, smaller step size will have to be used to achieve fine convergence. In Appendix A we provide theoretical justification for this intuition and demonstrate that the optimal step size schedule should be such that the asymptotic bias $D(p, p_\epsilon)$ is smaller than current total bias $D(p, \rho_t)$ by some a time-independent factor, $D(p, p_{\epsilon_t}) = CD(p, \rho_t)$. Here D is some metric on the

space of distributions. Adopting this prescription in practice presents several challenges:

Total bias. Computing distances such as Wasserstein distance or the total variation distance is hard, even when the exact samples are given, let alone in the present situation, where samples from $p(\boldsymbol{x})$ are the final goal of the algorithm. Instead, we will construct a proxy for the distance, which is based on a certain class of summary statistics, for which the ground truth is known and is independent of $p(\boldsymbol{x})$. We will examine the equipartition condition (Röver et al., 2023): the matrix

$$V_{ij}(p, \rho_t) = \mathbb{E}_{\rho_t}[-(x_i - \mathbb{E}_{\rho_t}[x_i])\partial_j \log p(\boldsymbol{x})] \quad (4)$$

equals the identity when $\rho_t = p$, as can be seen by the integration by parts:

$$\begin{aligned} V_{ij}(p, p) &= -\int (x_i - \mathbb{E}_p[x_i])\partial_j[\log p(\boldsymbol{x})]p(\boldsymbol{x})d\boldsymbol{x} \\ &= -\int (x_i - \mathbb{E}_p[x_i])\partial_j p(\boldsymbol{x})d\boldsymbol{x} \\ &= \delta_{ij}\int p(\boldsymbol{x})d\boldsymbol{x} = \delta_{ij}. \end{aligned} \quad (5)$$

We use this fact to construct

$$\widetilde{D}(p, \rho_t) = \frac{1}{d}\operatorname{Tr}\big\{(I - V(p, \rho_t))(I - V(p, \rho_t))^T\big\}. \quad (6)$$

$\widetilde{D}$ is not a metric on the space of distributions, because it is only based on certain summary statistics which do not define the full distribution uniquely. Nevertheless, in Appendix B we show that it is a divergence on the space of zero-mean Gaussian distributions. Crucially, $\widetilde{D}(\rho_t, p)$ can be calculated using the ECA formalism (3) with negligible computational overhead, as shown in Appendix B.

Proportionality constant. The optimal constant $0 < C < 1$ is not known and can depend on the problem at hand. In this work we will fix it to $C = 0.025$. We perform ablation studies in Appendix D, which demonstrate that this value works well for the problems considered here and the performance is stable for a large range of other values.

Asymptotic bias. The challenge is how to estimate asymptotic bias $D(p, p_\epsilon)$ without knowing the ground truth or even converging to the asymptotic regime. Dynamics, such as HMC, LMC and MCLMC have a quantity called the energy which does not change with time for the exact solution. A step with the approximate dynamics, however, changes the energy by amount $\Delta(\boldsymbol{x}, \boldsymbol{u}|p, \epsilon)$. Exact form of Δ is given in Appendix E. Crucially, it only depends on $\log p(\boldsymbol{x})$ and $\nabla \log p(\boldsymbol{x})$, which are both a side product of the kernel update computation, so the energy change can be computed with negligible computational overhead. Of interest will be the variance of the energy change over the ensemble, specifically the quantity Energy Error Variance per Dimension (EEVPD):

$$\mathrm{EEVPD}(\rho|p, \epsilon) = \mathrm{Var}_{\rho(\boldsymbol{x},\boldsymbol{u})}[\Delta(\boldsymbol{x}, \boldsymbol{u}|p, \epsilon)]/d, \quad (7)$$

where the variance is $\mathrm{Var}_\rho[f] = \mathbb{E}_\rho[(f - \mathbb{E}_\rho[f])^2]$. EEVPD is a useful quantity, because it provides an upper bound for the asymptotic bias (Robnik et al., 2024a) through

$$\widetilde{D}(p, p_\epsilon) \leq F^{-1}(\mathrm{EEVPD}(p_\epsilon|p, \epsilon)), \quad (8)$$

where $F(\widetilde{D}) = 4\widetilde{D}^{3/2}/(1 + \widetilde{D}^{1/2})^2$ is a monotonically increasing function. This bound has been demonstrated rigorously for Gaussian distributions, but also numerically shown to typically hold for non-Gaussian distributions. (Robnik et al., 2024a) also show that EEVPD scales roughly as $\mathrm{EEVPD} \propto \epsilon^6$.

In summary: optimal step size schedule should maintain an asymptotic bias that is smaller than the total bias by some fixed constant. We estimate the total bias by the equipartiton condition and the asymptotic bias by the energy error and then select a step size according to the optimal schedule condition. This results in the following scheme for determining the step size ϵ_{t+1}, given the ensemble at iteration ρ_t.

1. Calculate $\widetilde{D}(p, \rho_t)$ from Equation (6).
2. Convert it to the desired asymptotic bias by $\widetilde{D}(p, p_{\epsilon_t}) = C\widetilde{D}(p, \rho_t)$.
3. Convert the desired asymptotic bias to the desired $\mathrm{EEVPD} = F(\widetilde{D}(p, p_{\epsilon_t}))$.
4. Estimate $EEVPD(\rho_t|p, \epsilon_t)$ associated with the current step. Equation (3) is used to compute the variance.
5. Using the approximate scaling of EEVPD with the step size determine the step size that should be used in the next iteration to ensure the desired EEVPD, $\epsilon_{t+1} = \epsilon_t\big(\mathrm{EEVPD}/\mathrm{EEVPD}(\rho_t|p, \epsilon_t)\big)^{1/6}$.

Figure 1 shows that with this procedure (i) EEVPD succesfully traces its desired value (ii) step size decays which translates to a rapid decay in bias.

3.2 Momentum decoherence scale

Noise added to the dynamics can speed-up mixing by ensuring ergodicity. However, the noise strength

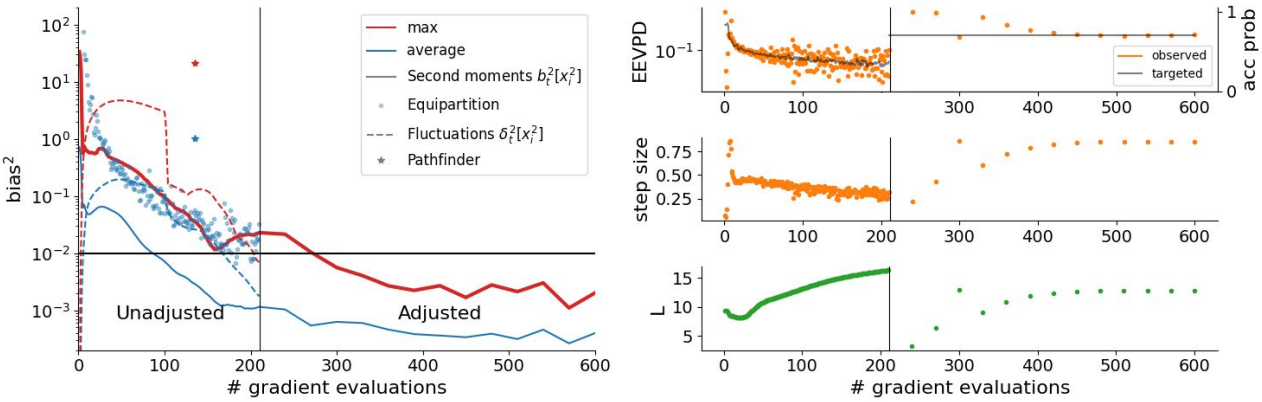

Figure 1: LAPS convergence for the German Credit problem. Left panel: bias of the second moments relative to the ground truth, $b^2_{\max}$ (red) and b^2_{avg} (blue), equipartition loss (blue dots) and the second-moment fluctuations (dotted lines) are shown as a function of the number of gradient evaluations. The latter two can be estimated when the ground truth is not known. In this example, they reflect the actual bias quite well. The switch from the unadjusted to the adjusted phase is shown by a vertical line. In this case, the unadjusted phase already comes very close to the desired accuracy, but adjusted method further improves on that. Note that the final ensemble is not biased, the residual error is caused by the finite number of chains. Pathfinder's second-moment bias is shown with stars. Right panels: convergence of hyperparameters step size and L is shown in the lower two panels. Step size is tuned based on the quantities derived from the energy error, shown in the upper panel. EEVPD is used in the unadjusted phase, acceptance rate in the adjusted phase. Targeted EEVPD and acceptance rate are shown in grey. Acceptance rate is fixed, while targeted EEVPD is adaptively changed, based on the current bias estimate from the equipartition loss. Note that the step size in the adjusted phase converges within a few steps, thanks to the bisection method.

should not be too large, so that the dynamics can efficiently explore the posterior. Optimal time scale before the momentum direction decoheres was shown to be comparable to the typical scale of the posterior Robnik et al. (2024b), suggesting the prescription

$$L_t = \alpha \left(\sum_{i=1}^{d} \mathrm{Var}_{\rho_t}[x_i] \right)^{1/2}. \tag{9}$$

In Robnik et al. (2024b), $\alpha = 1$ was used as default, while here we find that a larger value $\alpha = 2$ works better, corresponding to the more deterministic dynamics. See the ablation studies in Appendix D.

4 SWITCH TO ADJUSTMENT

Unadjusted sampler with a decaying step size is in principle capable of converging to the exact target distribution. However, we had to resort to several approximations in the previous section, because it is in practice difficult to have control over the asymptotic bias. But even compared to a perfect unadjusted step size schedule, the adjusted are faster to achieve fine convergence.

We will take a pragmatic approach in determining when to turn on adjustment: we will monitor summary statistics and determine when their values are no longer changing significantly, signaling that the chains are close to stationarity. We define a relative fluctuation of a summary statistic $\mathbb{E}[f]$ as

$$\delta_t[f] = \sigma_t[f]/\mu_t[f], \tag{10}$$

where average and standard deviation are computed in a moving window of length T:

$$\mu_t[f] = \frac{1}{T} \sum_{s=t-T}^{T} \mathbb{E}_{\rho_s}[f] \tag{11}$$

$$\sigma_t^2[f] = \frac{1}{T-1} \sum_{s=t-T}^{T} (\mathbb{E}_{\rho_s}[f] - \mu_t[f])^2.$$

These can be implemented in a memory-efficient way by performing an online average, so there is no need to store T states of the chain on each device.

If chains were stationary and in the target distribution, we would expect $\delta[f] = \mathrm{Var}_p[f]^{1/2} M^{-1/2} / \mathbb{E}_p[f] \sim M^{-1/2}$, so it should be very small if the ensemble is large. A small value of δ thus indicates that we are approaching equilibrium. For observables, we will take $f(\boldsymbol{x}) = x_i^2$ and terminate the unadjusted phase when all $\delta_t[x_i^2]$ fall below a certain threshold. We fix the

threshold to 0.01 and the moving window size to the 20% of the total sampling time.

Diagonal preconditioning helps to reduce the condition number if the specified target has parameters with different posterior widths, and is thus a common practice in MCMC (Sountsov and Hoffman, 2022). At the end of the unadjusted phase, we perform diagonal preconditioning by a coordinate transformation $y_i(\boldsymbol{x}) = x_i/\mathrm{Var}_{\rho_t}[x_i]^{1/2}$.

5 ADJUSTED SAMPLING

We now turn on Metropolis adjustment and adapt the hyperparameters to the new regime using ECA.

5.1 Step size

In the adjusted phase we tune the step size to achieve a desired average acceptance rate (Neal, 2011). Because the number of chains is large, the average acceptance rate estimate has a very small noise. As a consequence, stochastic optimization routines, such as dual averaging (Hoffman and Gelman, 2014) are not needed. Instead, the acceptance rate as a function of step size $a(\epsilon)$ is practically a monotonically decreasing function and we can simply use bisection to find the root of $a(\epsilon) - a_{\text{targeted}}$. First, we double (halve) the step size in each iteration if the initial step size is too small (too large) until we have a bracketing interval, i.e. two values of ϵ that bracket the root. We then run bisection until the acceptance rate equals the desired acceptance rate, up to some tolerance (3% by default). Thereafter the hyperparameters are fixed, which eliminates the bias caused by ECA (which is in typically unobservably small). Figure 1 illustrates that this procedure converges within a few steps. In our experiments, bisection was faster and more stable than dual averaging from Hoffman and Gelman (2014). Extending the argument from Beskos et al. (2013) and Neal (2011), Betancourt et al. (2015) argues that the optimal acceptance rate for product targets with HMC is in the range $60\% - 80\%$ for second-order integrators and $80\% - 87\%$ for fourth-order integrators. The same argument applies for MCLMC (Robnik et al., 2025). Slightly larger values of acceptance rate are typically more stable numerically (Phan et al., 2019), so we default the target acceptance rate to 70% for second-order integrators and 90% for fourth-order integrators. Ablation studies in Appendix D demonstrate that these choices have marginal impact on the performance.

5.2 Momentum decoherence scale

We do not adapt the trajectory length in this phase, because the phase is typically quite short. We fix 15 integration steps per trajectory, i.e. trajectory length in between full refreshments is $L_{\text{full}} = 15\epsilon$. We set the L in partial refreshments from Equation (23) to $L = 1.25\, L_{\text{full}}$, as recommended in Riou-Durand and Vogrinc (2023); Riou-Durand et al. (2023).

6 EXPERIMENTS

We test LAPS on several benchmark problems, taken from the Inference gym (Sountsov et al., 2020):

- Banana is a two-dimensional, banana shaped target, shown in Figure 3.
- Ill conditioned Gaussian is as 100-dimensional Gaussian with randomly oriented covariance matrix and eigenvalues drawn from the gamma distribution. Its condition number is around 10^5.
- German credit sparse logistic regression (GC) is a 51-dimensional Bayesian hierarchical model, where logistic regression is used to model the approval of the credit based on the information about the applicant.
- Brownian Motion (Brownian) is a 32-dimensional hierarchical problem, where Brownian motion with unknown innovation noise is fitted to the noisy and partially missing data.
- Item Response theory (IRT) is a 501-dimensional hierarchical problem where students' ability is inferred, given the test results.
- Stochastic Volatility (SV) is a 2519-dimensional hierarchical non-Gaussian random walk fit to the S&P500 returns data.

We follow Hoffman and Sountsov (2022) and define the bias of the expectation value $\mathbb{E}[f]$ as

$$b_t^2[f] = \frac{(\mathbb{E}_{\rho_t}[f] - \mathbb{E}_p[f])^2}{\mathrm{Var}_p[f]}, \tag{12}$$

where ground truth expectation values $\mathbb{E}_p[f]$ and $\mathrm{Var}_p[f] = \mathbb{E}_p[(f(\boldsymbol{x}) - \mathbb{E}_p[f])^2]$ are computed analytically for the Ill Conditioned Gaussian and by very long sequential NUTS runs for the other targets. As a measure of convergence we will either take the largest or the average second-moment bias across parameters:

$$b^2_{\max,t} \equiv \max_{1\le i\le d} b_t^2[x_i^2] \qquad b^2_{\text{avg},t} \equiv \frac{1}{d}\sum_{i=1}^{d} b_t^2[x_i^2]. \tag{13}$$

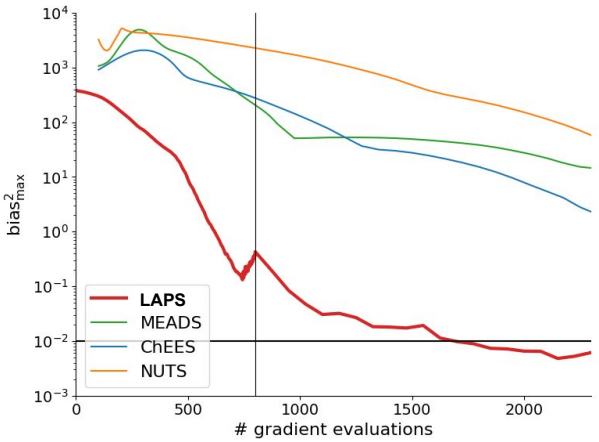

Figure 2: Second-moment bias as a function of the number of gradient evaluations on the Stochastic Volatility problem. LAPS converges significantly faster than the state-of-the-art ensemble samplers, which are shown for comparison. Note that in this case, the unadjusted phase does not quite achieve the desired accuracy requirements and the adjusted phase is essential for efficient convergence.

Both biases are of interest in applications and have been used as the posterior convergence metric. We use the max definition to directly compare with the ensemble algorithms from (Hoffman and Sountsov, 2022) and the average definition to directly compare with sequential algorithms from Robnik et al. (2024b).

In typical applications, computing the gradients $\nabla \log p(\boldsymbol{x})$ dominates the total sampling cost, so we take the number of gradient evaluations as a proxy of wall-clock time. As in (Hoffman and Sountsov, 2022), we measure a sampler's performance as the number of gradient calls n needed to achieve low error, $b^2_{\text{max}} < 0.01$.

In table 1, we compare LAPS to NUTS, ChEES-HMC, and MEADS as implemented in Hoffman and Sountsov (2022). Being MH adjusted, these samplers initially converge slowly, which these address by first running ADAM for 100 steps (Hoffman and Sountsov, 2022). We add these 100 gradient calls to the total cost of sampling in Table 1 and Figure 2. Note that ADAM initialization can be very far from the truth for hierarchical parameters. All samplers in this test use $M = 4096$ chains. In Appendix D we show that LAPS performance does not degrade if a smaller number of chains is used, as long as the number of chains is larger than $128 - 256$.

LAPS outperforms other samplers on all benchmark problems, in some cases by an order of magnitude.

Table 1: Grads to low max bias ($b^2_{\text{max}} = 0.01$), normalized per chain, lower is better. LAPS is compared to the state-of-the-art ensemble algorithms. LAPS is the best performing algorithm in all cases, by factors between 2 to 20.

	NUTS	ChEES	MEADS	LAPS
Banana	320	264	390	**17**
Gaussian	9846	5138	5520	**308**
GC	2716	1168	610	**300**
Brownian	2159	600	410	**206**
IRT	1162	537	790	**185**
SV	4173	3080	2860*	**1325**

* This is the number reported in Table 2 of Hoffman and Sountsov (2022). It is inconsistent with their Figure 7, which suggests it should be around 4200. Our Figure 2 is based on their Figure 7.

Convergence of hyperparameters in LAPS is shown in Figure 1 and 2, where other samplers also also shown for comparison. We also compared against the ensemble version of Pathfinder (Algorithm 2 in Zhang et al. (2022)). Pathfinder failed to converge for most of our benchmark problems. It only converged for the 2d Banana problem, where it reached $b^2_{\text{max}} = 0.02$ in 67 gradient evaluations, and for the German Credit problem, where it reached $b^2_{\text{max}} = 21.5$ in 135 gradient evaluations, as shown in Figure 1. Thus, even when it does not fail completely, the unadjusted phase of LAPS obtains a better approximation to the posterior significantly faster.

We also compare LAPS to sequential algorithms, NUTS, and MCLMC. Two quantities of interest are the total number of gradient calls to low bias (which is proportional to the total number of CPU/GPU hours) and the number of gradient calls per chain (which is proportional to the wall-clock time). In this setting, we use $M = 256$ chains for LAPS. The results are shown in Table 2. In terms of the total number of gradient calls, sequential algorithms are faster, but in some cases the difference is not large, demonstrating that in these cases MCLMC cold start burn-in cost is comparable to the cost of achieving an independent sample from a warm start. However, in terms of wall-clock time, LAPS is typically one to two orders of magnitude faster than sequential MCLMC and two to three orders of magnitude faster than NUTS.

7 CONCLUSIONS

Parallel MCMC is gaining popularity in the recent years due to the increased availability of parallel resources such as CPU clusters, GPUs and TPUs. It is especially attractive when likelihood evaluations are

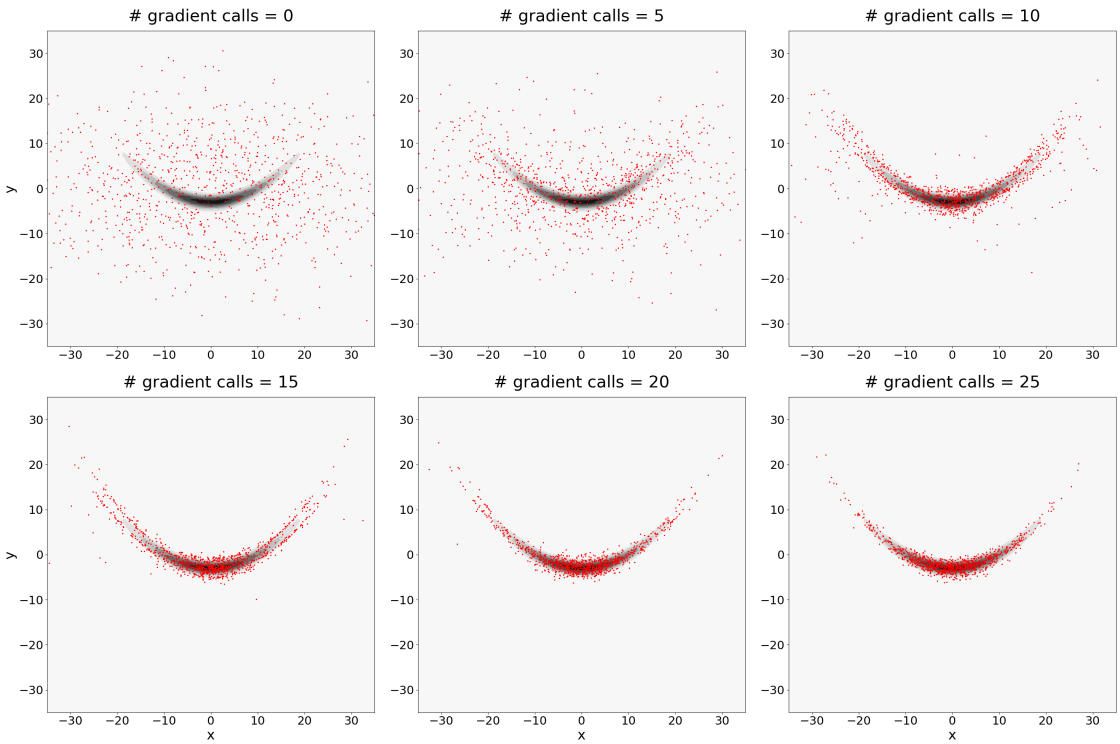

Figure 3: LAPS posterior on the Banana target, as a function of the number of gradient calls (per chain). The location of chains is shown in red, the target density in black. As can be seen, LAPS converges to the posterior in just around 20 gradient calls.

Table 2: Grads to low average bias ($b^2_{\text{avg}} = 0.01$). LAPS with 256 chains is compared to the state-of-the-art sequential algorithms. LAPS (total) counts gradient calls from all the chains, LAPS counts gradient calls per chain. If the computation is not parallelized across the chains, the runtime is determined by the total number of gradient calls and the sequential algorithms are typically faster. However, if the computation is parallelized, the number of gradient calls per chain determines the runtime and LAPS is significantly faster.

	NUTS (SEQ.)	MCLMC (SEQ.)	LAPS (TOTAL)	LAPS
BANANA	64,200	5000	4400	17
GAUSSIAN	113,900	33,500	68,600	270
GC	10,400	4200	28,700	110
BROWNIAN	5400	1700	17,900	70
IRT	7300	1600	21,000	80
SV	29,900	10,000	281,600	1100

time consuming, as it can significantly reduce the wall-clock time. However, its cost can easily be dominated by the speed of convergence from the cold start.

We have formulated the Late Adjusted Parallel Sampler, a scheme that initializes sampling with an unadjusted sampler. The key challenge of unadjusted sampling is determining the step size that ensures small asymptotic bias. We have argued that the optimal step size should be time dependent: initially it should be large and become smaller as the chains approach the target distribution and better precision is needed. We have devised a scheme that achieves this by monitoring the equipartition condition, as a proxy for convergence, and appropriately adapting the step size. We have demonstrated on standard benchmark problems that the resulting algorithm outperforms the state of the art parallel samplers by up to an order of magnitude.

Robustness Construction of the unadjusted phase of the algorithm is inspired by the results that are largely based on the Gaussian assumption, which is a clear limitation. For example, it is possible to construct pathological cases where $\widetilde{D}(p, \rho_t)$ is zero (or very small) but the distributions are not close. This would

cause the algorithm to use a step size that is too short and thus progress slowly. On the other hand we have observed that for the distributions with long tails (such as the Banana example) the $\widetilde{D}$ is an overly stringent metric and even for exact i.i.d. samples decays only slowly with the number of samples. This causes the step size to become too large and the ensemble gets stuck, limited by the discretization bias. The result of both of these failure modes is that the ensemble reaches a stationary distribution which is not equal to the target distribution. However, since the expectation values settle at this point in both cases, the adjusted phase, which does not rely on these assumptions is triggered. We find that this makes the scheme robust, i.e. in the worst case, the adjusted phase is triggered.

Similarly, EEVPD $\propto \epsilon^6$ is not required to hold exactly for the algorithm to work, important is the feedback loop which makes sure that the step size is reduced if the EEVPD is larger than expected and to increase if it is smaller than expected, so the more broadly valid assumption of monotonic growth of EEVPD with the stepsize is sufficient. The upper left panel of Figure 1 shows that this feedback loop works well for a non-Gaussian distribution and demonstrates that the observed EEVPD (orange) tracks its desired value (grey). In general, we have tested the algorithm on various non-Gaussian distributions and the performance and the stability of quantities like the EEVPD indicate that the algorithm is robust to the assumptions.

Future work Several parallel methods such as sequential Monte Carlo, annealing and parallel tempering approach the problem of multi-modal distributions. These approaches are agnostic with respect to the kernel being used, so they are compatible with LAPS. For example in annealing, one could run LAPS at each temperature level, to bring the distribution of particles from the previous temperature level to the new stationary distribution as quickly as possible.

Equipartition was used in LAPS as a convergence metric, but this is not the only option. One alternative is the split-$\hat{R}$ statistics (Margossian et al., 2024), which could also be useful for determining when to switch on adjustment.

Finally, extensive testing on a variety of real-world problems with heavy tails, multimodality or complex geometry will help establish the LAPS strategy in the broader context.

Acknowledgements

We thank Reuben Cohn-Gordon for many discussions, comments on the manuscript and help with Blackjax. We thank Nawaf Bou-Rabee, Bob Carpenter and Siddharth Mitra for useful discussions. This material is based upon work supported in part by the Heising-Simons Foundation grant 2021- 3282 and by NSF CSSI grant award number 2311559.

References

Altschuler, J. M. and Chewi, S. (2024). Faster High-accuracy Log-concave Sampling via Algorithmic Warm Starts. *J. ACM*, 71(3):24:1–24:55.

Baydin, A. G., Pearlmutter, B. A., Radul, A. A., and Siskind, J. M. (2018). Automatic differentiation in machine learning: A survey. *Journal of Machine Learning Research*, 18.

Beskos, A., Pillai, N., Roberts, G., Sanz-Serna, J.-M., and Stuart, A. (2013). Optimal tuning of the hybrid Monte Carlo algorithm. *Bernoulli*, 19(5A):1501–1534.

Betancourt, M. J., Byrne, S., and Girolami, M. (2015). Optimizing The Integrator Step Size for Hamiltonian Monte Carlo. arXiv:1411.6669 [stat].

Bou-Rabee, N. and Eberle, A. (2023). Mixing time guarantees for unadjusted Hamiltonian Monte Carlo. *Bernoulli*, 29(1):75–104.

Bou-Rabee, N. and Schuh, K. (2023). Convergence of unadjusted Hamiltonian Monte Carlo for mean-field models. *Electronic Journal of Probability*, 28(none):1–40.

Cabezas, A., Corenflos, A., Lao, J., and Louf, R. (2024). Blackjax: Composable bayesian inference in jax.

Campagne, J.-E., Lanusse, F., Zuntz, J., Boucaud, A., Casas, S., Karamanis, M., Kirkby, D., Lanzieri, D., Li, Y., and Peel, A. (2023). JAX-COSMO: An End-to-End Differentiable and GPU Accelerated Cosmology Library. *The Open Journal of Astrophysics*, 6:10.21105/astro.2302.05163. arXiv:2302.05163 [astro-ph].

Camrud, E., Durmus, A., Monmarché, P., and Stoltz, G. (2024). Second order quantitative bounds for unadjusted generalized Hamiltonian Monte Carlo. arXiv:2306.09513 [math].

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. (2017). Stan: A Probabilistic Programming Language. *Journal of Statistical Software*, 76:1–32.

Crespi, A., Bonici, M., Loureiro, A., Ruiz-Zapatero, J., Sladoljev, I., Li, Z., Bayer, A., Millea, M., and Seljak, U. (2025). Flinch: A Differentiable Framework for Field-Level Inference of Cosmological parameters from curved sky data. arXiv:2510.26691 [astro-ph].

Dillon, J. V., Langmore, I., Tran, D., Brevdo, E., Vasudevan, S., Moore, D., Patton, B., Alemi, A., Hoffman, M., and Saurous, R. A. (2017). Tensorflow distributions. *arXiv preprint arXiv:1711.10604*. Also part of TensorFlow Probability library.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222.

Durmus, A. O. and Eberle, A. (2023). Asymptotic bias of inexact Markov Chain Monte Carlo methods in high dimension. arXiv:2108.00682 [cs, math, stat].

Gattringer, C. and Lang, C. B. (2010). *Quantum Chromodynamics on the Lattice: An Introductory Presentation*, volume 788 of *Lecture Notes in Physics*. Springer Berlin Heidelberg, Berlin, Heidelberg.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (1995). *Bayesian Data Analysis*. Chapman and Hall/CRC, New York.

Gilks, W. R., Roberts, G. O., and George, E. I. (1994). Adaptive Direction Sampling. *Journal of the Royal Statistical Society Series D: The Statistician*, 43(1):179–189.

Goodman, J. and Weare, J. (2010). Ensemble samplers with affine invariance. *Communications in Applied Mathematics and Computational Science*, 5(1):65–80.

Griewank, A. and Walther, A. (2008). Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, Second Edition. Society for Industrial and Applied Mathematics. Edition: Second.

Hernández-Sánchez, M., Kitaura, F.-S., Ata, M., and Vecchia, C. D. (2021). Higher Order Hamiltonian Monte Carlo Sampling for Cosmological Large-Scale Structure Analysis. *Monthly Notices of the Royal Astronomical Society*, 502(3):3976–3992. arXiv:1911.02667 [astro-ph, physics:hep-lat].

Hoffman, M., Radul, A., and Sountsov, P. (2021). An Adaptive-MCMC Scheme for Setting Trajectory Lengths in Hamiltonian Monte Carlo. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pages 3907–3915. PMLR.

Hoffman, M. D. and Gelman, A. (2014). The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(47):1593–1623.

Hoffman, M. D. and Sountsov, P. (2022). Tuning-Free Generalized Hamiltonian Monte Carlo. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pages 7799–7813. PMLR.

Hutchinson, M. F. (1989). A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076.

Lao, J., Suter, C., Langmore, I., Chimisov, C., Saxena, A., Sountsov, P., Moore, D., Saurous, R. A., Hoffman, M. D., and Dillon, J. V. (2020). tfp.mcmc: Modern Markov Chain Monte Carlo Tools Built for Modern Hardware.

Leimkuhler, B. and Matthews, C. (2015). *Molecular Dynamics: With Deterministic and Stochastic Numerical Methods*, volume 39 of *Interdisciplinary Applied Mathematics*. Springer International Publishing, Cham.

Margossian, C. C. and Gelman, A. (2024). For how many iterations should we run Markov chain Monte Carlo? arXiv:2311.02726 [stat].

Margossian, C. C., Hoffman, M. D., Sountsov, P., Riou-Durand, L., Vehtari, A., and Gelman, A. (2024). Nested R^: Assessing the Convergence of Markov Chain Monte Carlo When Running Many Short Chains. *Bayesian Analysis*, -1(-1):1–28.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*, volume 118 of *Lecture Notes in Statistics*. Springer, New York, NY.

Neal, R. M. (2011). *MCMC using Hamiltonian dynamics*. arXiv:1206.1901 [physics, stat].

Omelyan, I. P., Mryglod, I. M., and Folk, R. (2002). Optimized Verlet-like algorithms for molecular dynamics simulations. *Physical Review E*, 65(5):056706.

Omelyan, I. P., Mryglod, I. M., and Folk, R. (2003). Symplectic analytically integrable decomposition algorithms: classification, derivation, and application to molecular dynamics, quantum and celestial mechanics simulations. *Computer Physics Communications*, 151(3):272–314.

Phan, D., Pradhan, N., and Jankowiak, M. (2019). Composable Effects for Flexible and Accelerated Probabilistic Programming in NumPyro. arXiv:1912.11554 [stat].

Riou-Durand, L., Sountsov, P., Vogrinc, J., Margossian, C., and Power, S. (2023). Adaptive Tuning for Metropolis Adjusted Langevin Trajectories. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pages 8102–8116. PMLR.

Riou-Durand, L. and Vogrinc, J. (2023). Metropolis Adjusted Langevin Trajectories: a robust alternative to Hamiltonian Monte Carlo. arXiv:2202.13230 [stat].

Robnik, J., Cohn-Gordon, R., and Seljak, U. (2024a). Black-box unadjusted Hamiltonian Monte Carlo. *arXiv preprint arXiv:2412.08876*.

Robnik, J., Cohn-Gordon, R., and Seljak, U. (2025). Metropolis Adjusted Microcanonical Hamiltonian Monte Carlo.

Robnik, J., De Luca, G. B., Silverstein, E., and Seljak, U. (2024b). Microcanonical Hamiltonian Monte Carlo. *The Journal of Machine Learning Research*, 24(1):311:14696–311:14729.

Robnik, J. and Seljak, U. (2024). Fluctuation without dissipation: Microcanonical Langevin Monte Carlo. In *Proceedings of the 6th Symposium on Advances in Approximate Bayesian Inference*, pages 111–126. PMLR.

Röver, L., von Campe, H., Herzog, M. P., Kuntz, R. M., and Schäfer, B. M. (2023). Partition function approach to non-Gaussian likelihoods: physically motivated convergence criteria for Markov chains. *Monthly Notices of the Royal Astronomical Society*, 526(1):473–482.

Schoenholz, S. S., Cubuk, E. D., et al. (2020). Jax, m.d.: A framework for differentiable physics. arXiv preprint arXiv:1912.04232.

Simon-Onfroy, H., Lanusse, F., and Mattia, A. d. (2025). Benchmarking field-level cosmological inference from galaxy redshift surveys. arXiv:2504.20130 [astro-ph].

Sommer, E., Robnik, J., Nozadze, G., Seljak, U., and Rügamer, D. (2024). Microcanonical Langevin Ensembles: Advancing the Sampling of Bayesian Neural Networks.

Sountsov, P., Carroll, C., and Hoffman, M. D. (2024). Running Markov Chain Monte Carlo on Modern Hardware and Software. arXiv:2411.04260 [stat].

Sountsov, P. and Hoffman, M. D. (2022). Focusing on Difficult Directions for Learning HMC Trajectory Lengths. arXiv:2110.11576 [stat].

Sountsov, P., Radul, A., and contributors (2020). Inference gym.

Steeg, G. V. and Galstyan, A. (2021). Hamiltonian Dynamics with Non-Newtonian Momentum for Rapid Sampling. arXiv:2111.02434 [physics].

Takaishi, T. and de Forcrand, P. (2006). Testing and tuning symplectic integrators for Hybrid Monte Carlo algorithm in lattice QCD. *Physical Review E*, 73(3):036706. arXiv:hep-lat/0505020.

Zhang, L., Carpenter, B., Gelman, A., and Vehtari, A. (2022). Pathfinder: Parallel quasi-Newton variational inference. *Journal of Machine Learning Research*, 23(306):1–49.

Øksendal, B. (2003). Stochastic Differential Equations. In Øksendal, B., editor, *Stochastic Differential Equations: An Introduction with Applications*, pages 65–84. Springer, Berlin, Heidelberg.

Štrumbelj, E., Bouchard-Côté, A., Corander, J., Gelman, A., Rue, H., Murray, L., Pesonen, H., Plummer, M., and Vehtari, A. (2024). Past, Present and Future of Software for Bayesian Inference. *Statistical Science*, 39(1):46–61.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. **[Yes]** The algorithms are described in detail in Sections 3, 4 and 5. Further details are given in Appendices C E and F.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. **[Yes]** Provided in Section C.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. **[Yes]** Provided in Introduction.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. **[Yes]** See Section A.
   (b) Complete proofs of all theoretical results. **[Yes]** See Section A.
   (c) Clear explanations of any assumptions. **[Yes]** See Section A.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). **[Yes]** Provided in Introduction.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). **[Yes]** The LAPS hyperparameters are provided in the main text with ablation studies in Appendix D.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to

the random seed after running experiments multiple times). **[Yes]** The measure of convergence is defined in Section 6. Statistical errors are usually very small due to the large number of chains and are therefore not displayed. One exception is the ablation study where we reduce the number of chains, see Figure 9. In this case we report the errors by repeating the experiments with different random seeds.

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). **[Yes]** Provided in Appendix C.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator if your work uses existing assets. **[Yes]** Baseline sampling algorithms are cited in Section 6 and so are the inference models used for comparison.

   (b) The license information of the assets, if applicable. **[Not Applicable]**

   (c) New assets either in the supplemental material or as a URL, if applicable. **[Yes]** LAPS implementation is provided in Blackjax.

   (d) Information about consent from data providers/curators. **[Not Applicable]**

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. **[Not Applicable]**

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots. **[Not Applicable]** Crowdsourcing or research with human subjects was not performed.

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. **[Not Applicable]** Crowdsourcing or research with human subjects was not performed.

   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. **[Not Applicable]** Crowdsourcing or research with human subjects was not performed.

Supplementary Materials

A Optimal step size schedule

In this section we provide theoretical guidance for determining the step size schedule in the unadjusted phase of the algorithm. We here assume that the trajectory length L is fixed and will only consider the schedule for the step size ϵ. Let's denote by K_ϵ the transition operator corresponding to the Markov kernel $\varphi(\cdot|\epsilon)$. We have dropped the dependence on L for shorter notation. Let's denote the stationary distribution of K_ϵ by p_ϵ, that is, $K_\epsilon\, p_\epsilon = p_\epsilon$. We construct the Markov chain by applying the Markov kernel with step size ϵ_t in iteration t, such that the chain distribution evolves as $\rho_{t+1} = K_{\epsilon_t}\rho_t$. The core of the ECA framework is that ϵ_t may be informed by ρ_t.

We will quantify convergence as a distance $D(\rho_t, p)$ between the chain distribution and the target distribution. Here, D is a metric on the space of distributions, for example, the Wasserstein metric or the total variation distance. Since each kernel K_{ϵ_t} application performs T/ϵ_t steps, we define the per-step rate of convergence as

$$R_t = \frac{D(\rho_t, p) - D(\rho_{t+1}, p)}{T/\epsilon_t}. \tag{14}$$

In this section we will make the following assumptions about the Markov kernel and its asymptotic bias:

- **Assumption 1.** Asymptotic bias can be bounded by

$$D(p, p_\epsilon) \leq c_{asym}\epsilon^\kappa, \tag{A1}$$

  where the constant $c_{asym} > 0$ and order $\kappa > 0$ do not depend on the step size.

- **Assumption 2.** Dynamics is contracting towards the biased limit:

$$D(K_\epsilon\rho, p_\epsilon) \leq (1-c)D(\rho, p_\epsilon), \tag{A2}$$

  for any distribution ρ and step size ϵ. Contraction constant $0 < c < 1$ does not depend on the step size.

For example, for unadjusted HMC with the leapfrog integrator and the target distribution that is log-convex ($(-\nabla \log p(\boldsymbol{x}) + \nabla \log p(\boldsymbol{y}))\cdot(\boldsymbol{x}-\boldsymbol{y}) \geq \|\boldsymbol{x}-\boldsymbol{y}\|^2$) and has bounded second and third derivatives ($\|\nabla^2 \log p(\boldsymbol{x})\| \leq M_2$ and $\|\nabla^3 \log p(\boldsymbol{x})\| \leq M_3$), these assumptions hold with $\kappa = 2$, $c_{asym} = C(M_2, M_3, L)D/c$, $D = d + \int \|\boldsymbol{x}\| p(\boldsymbol{x}) d\boldsymbol{x} + \int \|\boldsymbol{x}\|^2 p(\boldsymbol{x}) d\boldsymbol{x}$ and $c = mL^2/4$.

We have the following result:

Theorem 1. *Given Assumptions A1 and A2 and a step size schedule, which is recursively constructed in such a way that it satisfies*

$$c_{asym}\epsilon_t^\kappa = \frac{c}{(2-c)(\kappa+1)} D(\rho_t, p), \tag{15}$$

the chain is guaranteed to converge to the target distribution p:

$$D(\rho_{t+1}, p) \leq (1 - c\frac{\kappa}{\kappa+1}) D(\rho_t, p). \tag{16}$$

This schedule is optimal in the sense that it maximizes the lower bound for the rate of convergence R_t.

Proof. Using Assumption A2 and applying triangle inequality twice gives

$$\begin{aligned} D(\rho_{t+1}, p) &\leq D(\rho_{t+1}, p_\epsilon) + D(p, p_\epsilon) \\ &\leq (1-c)D(\rho_t, p_\epsilon) + D(p, p_\epsilon) \\ &\leq (1-c)D(\rho_t, p) + (2-c)D(p, p_\epsilon). \end{aligned}$$

We note that a sufficient condition for the convergent chain is $D(p, p_\epsilon) < \frac{c}{2-c} D(\rho_t, p)$. Applying Assumption A1 and inserting the step size schedule (15) gives

$$\begin{aligned} D(\rho_{t+1}, p) &\leq (1-c)D(\rho_t, p) + (2-c)c_{asym}\epsilon_t^\kappa \\ &\leq (1 - c\frac{\kappa}{\kappa+1})D(\rho_t, p). \end{aligned}$$

The rate of convergence is lower bounded by

$$R_t \geq \frac{\epsilon_t}{T}\big(cD(\rho_t, p) - (2-c)c_{asym}\epsilon_t^\kappa\big).$$

The lower bound is maximized for the unique ϵ_t for which

$$\frac{\partial R_t}{\partial \epsilon_t} = \frac{1}{T}\big(cD(\rho_t, p) + (2-c)(\kappa+1)c_{asym}\epsilon_t^\kappa\big) = 0,$$

which gives the schedule (15). □

This result suggests that the optimal step size schedule should control the asymptotic bias to be by a constant factor smaller than the total bias.

Numerical illustration In Figure 4 we compare the adaptive step size used in LAPS with the fixed step size strategy, all other elements being equal. It can be seen that the adaptive step size schedule outperforms any fixed given fixed step size: large steps are initially fast, but soon reach the asymptotic bias, while small step sizes are slow to converge.

B Equipartition conditon

B.1 Computation

Full rank. Naively computing $\widetilde{D}$ from Equation (6) requires $\mathcal{O}(d^2M)$ operations or $\mathcal{O}(dM^2)$ by some rearranging of the sums. This is prohibitively expensive. However, it can be reduced to $\mathcal{O}(dM)$ (so $\mathcal{O}(d)$ per chain) by the Hutchinson's trick (Hutchinson, 1989): a vector of random variables $\boldsymbol{z}$ with $\mathbb{E}[z_i z_j] = \delta_{ij}$ is introduced and inserted it in the trace to obtain

$$\widetilde{D} = \frac{1}{d}\sum_{i,k,l=1}^{d} \mathbb{E}[(I-E)_{ik} z_k (I-E)_{il} z_l] \approx \mathbb{E}\left[\left|\frac{1}{dM}\sum_{m=1}^{M} \boldsymbol{z} - (\boldsymbol{\nabla}\mathcal{L}(\boldsymbol{x}^m)\cdot\boldsymbol{z})\boldsymbol{x}^m\right|^2\right], \tag{17}$$

where the ECA approximation (3) was used in the second step (which is bias-free and with variance decaying as $1/M$). Two popular choices for the $\boldsymbol{z}$ distribution are Gaussian distribution $\boldsymbol{z} \sim \mathcal{N}(0, I)$ and the Rademacher distribution, we pick the latter. We have checked that using 100 realizations of the $\boldsymbol{z}$ variable ensures sub-percent error in the $\widetilde{D}$ computation.

Diagonal. Run-time overhead of the above calculation is negligible, for all but the simplest problems, because it only requires minimal interaction between the chains (computing 100 expectation values). This is typically negligible compared to the gradient evaluation cost. However, it does introduce the cost of storing 100 d-dimensional $\boldsymbol{z}$ vectors. In some applications this is prohibitive, as the model computation already takes the entire GPU. We therefore also consider a simplification of $\widetilde{D}$ which only considers the diagonal terms of the equipartition matrix:

$$\widetilde{D}_{\text{diag}} = \frac{1}{d}\sum_{i=1}^{d}(1 - V_{ii})^2. \tag{18}$$

Evaluating $\widetilde{D}_{\text{diag}}$ does not introduce the memory overhead. We will therefore use $\widetilde{D}_{\text{diag}}$ as default. In Table 4 we show that the performance does not degrade relative to the full rank equipartition matrix.

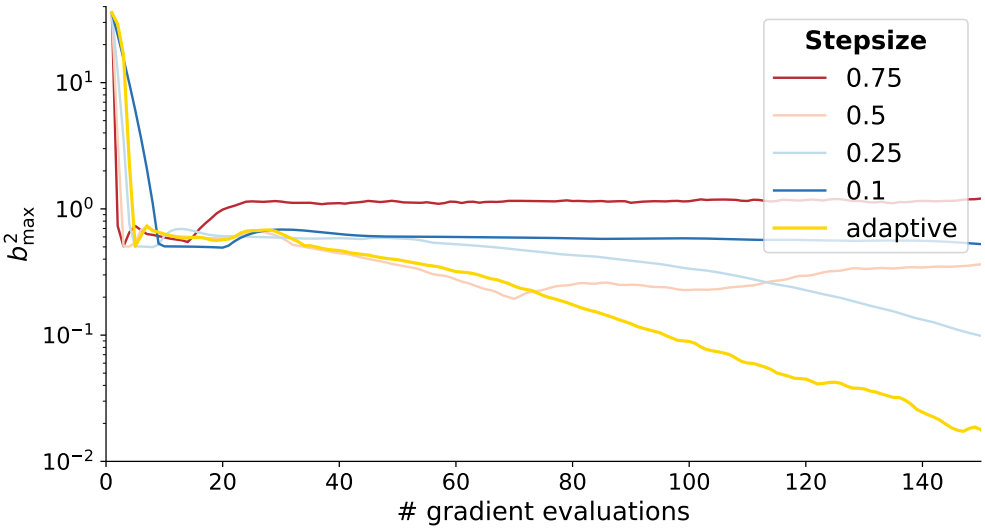

Figure 4: Bias as a function of the number of gradient evaluations with the unadjusted MCLMC dynamics for the German Credit problem. Adaptive step size adaptation used in LAPS (yellow) is compared with a fixed step size strategy for various step size choices. Large step size strategy (red) is initially very fast, but is not able to achieve fine convergence. Small step size (blue) is slow to converge but has a smaller asymptotic bias. LAPS adaptively reduces the step size, combining the best of all worlds.

B.2 Divergence property

We will here show that $\widetilde{D}(p, p')$ is a divergence on the space of zero-mean Gaussian distributions, meaning that it is non-negative and positive if and only if $p = p'$. Similarly $\widetilde{D}_{\text{diag}}$ is a divergence on the space of zero-mean uncorrelated Gaussian distributions. Zero-mean Gaussian distributions are completely determined by their covariance matrix Σ, we will denote them by $\mathcal{N}(\boldsymbol{x}|\Sigma)$. The equipartition matrix in this case simplifies to:

$$V_{ij}(\mathcal{N}, \mathcal{N}') = \int x_i \partial_j \big(\frac{1}{2}\boldsymbol{x}^T \Sigma^{-1} \boldsymbol{x}\big) \mathcal{N}(\boldsymbol{x}|\Sigma') d\boldsymbol{x} = [\Sigma' \Sigma^{-1}]_{ij},$$

meaning that

$$\widetilde{D}(\mathcal{N}, \mathcal{N}') = \frac{1}{d} \operatorname{Tr}\big\{(I - \Sigma'\Sigma^{-1})(I - \Sigma'\Sigma^{-1})^T\big\} = \frac{1}{d}\|I - \Sigma'\Sigma^{-1}\|_F^2 \tag{19}$$

$$\widetilde{D}_{\text{diag}}(\mathcal{N}, \mathcal{N}') = \frac{1}{d}\sum_{i=1}^{d}(1 - \Sigma'_{ii}/\Sigma_{ii})^2, \tag{20}$$

where $\|\cdot\|_F$ is the Frobenious norm. $\widetilde{D}(\mathcal{N}, \mathcal{N}')$ is evidently positive (because the norm is positive) and zero if and only if $I - \Sigma'\Sigma^{-1} = 0$, which is equivalent to $\Sigma = \Sigma'$. Covariance matrix uniquely parameterizes the zero-mean Gaussian distribution so this demonstrates that $\widetilde{D}$ is a divergence on this space. Similarly Equation (20) is a sum of positive terms, so it is zero if and only if all of the terms are zero, which is equivalent to $\Sigma'_{ii} = \Sigma_{ii}$ for all i. Thus, $\widetilde{D}_{\text{diag}}$ is a divergence on the space of uncorrlated Gaussian distributions, because $\{\Sigma_{ii}\}_{i=1}^d$ uniquely determine the uncorrelated Gaussian distributions.

We note that Equation (20) coincides with the covariance matrix error as defined in (Robnik et al., 2024a).

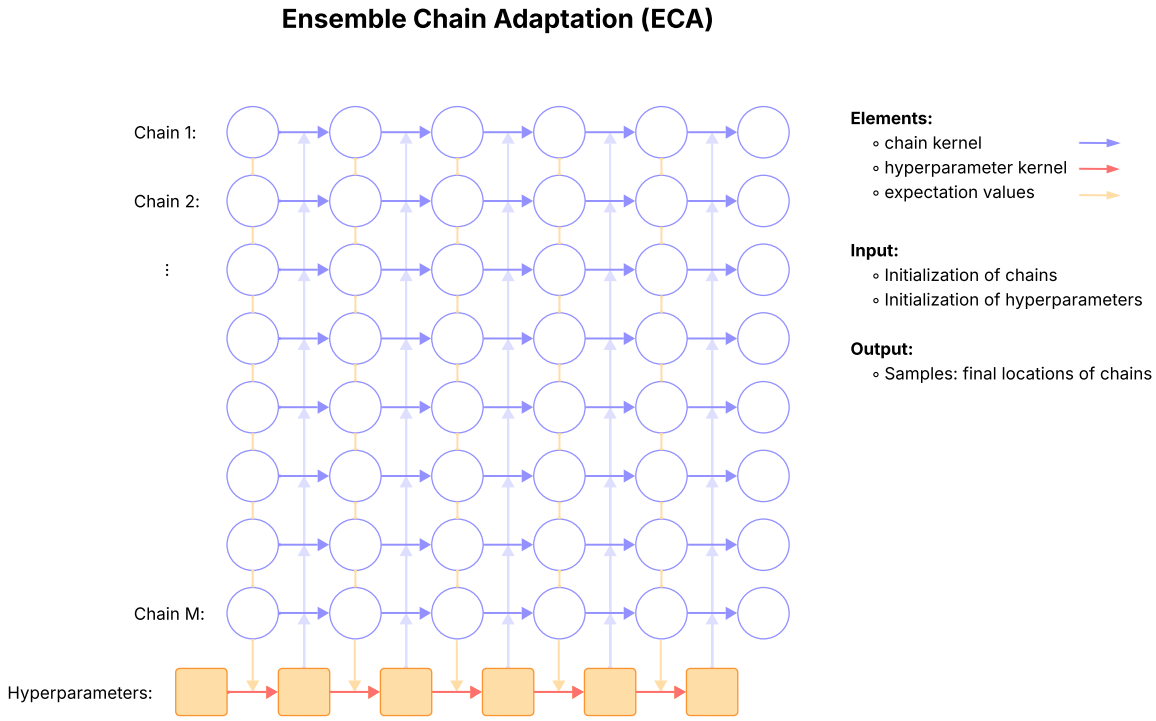

Figure 5: Schematic of the Ensemble Chain Adaptation (ECA) algorithm that we adopt. M chains are evolved, time goes in the horizontal direction. In each step, the locations of all the chains are used to compute expectation values (orange arrow) which are used to update the hyperparameters (red arrow). Updated hyperparameters specify the kernel (pale purple arrow) which updates the chains in parallel (solid purple arrows).

C LAPS details

We here provide some further details and clarifications of the LAPS algorithm.

Graphical illustration. Hyperparameters in LAPS are updated by taking averages over the chains. This is illustrated in Figure 5.

Pseudocode is shown in Algorithm 1. The first while loop corresponds to the unadjusted phase, and the second to the adjusted phase. The MCLMC dynamics updates for different chains are independent and can be executed in parallel. The hyperparameters are then updated using expectation values over the ensemble.

Memory complexity The memory complexity of the algorithm is $\mathcal{O}(d)$ per chain, where d is the dimensionality of the parameter space on which the inference is performed. Complexity is independent of the number of steps.

Architecture. The experiments were run on the NVIDIA A100 GPU (40GB).

Initialization The initial positions of the chains are provided by the user. In the experiments in this work, we initialize them by drawing samples from the prior distribution, as defined by the Inference Gym models. We initialize the velocity to be aligned with the gradient (and it has unit norm in MCLMC).

D Ablation studies

We here study the performance of the algorithm as a function of various algorithmic choices.

Input:
differentiable unnormalized log-probability density $\log p(\boldsymbol{x})$,
initial particle positions $\{\boldsymbol{x}^m\}_{m=1}^M$,
initial particle velocities $\{\boldsymbol{u}^m\}_{m=1}^M$
Output: samples $\{\boldsymbol{x}^m\}_{m=1}^M$.

$\epsilon \leftarrow 0.01\sqrt{d}$
$L \leftarrow$ Equation (9) $(\{\boldsymbol{x}^m\})$
$t \leftarrow 0$
repeat
  $\boldsymbol{x}^m, \boldsymbol{u}^m, \Delta^m \leftarrow$ MCLMC update (21)$(\boldsymbol{x}^m, \boldsymbol{u}^m, \epsilon, L)$
  EEVPD $= \frac{1}{M}\sum_{i=1}^M (\Delta^m)^2/d$
  $\widetilde{D} \leftarrow$ Equation (6)$(\{\boldsymbol{x}^m\})$
  $\text{EEVPD}_{\text{wanted}} \leftarrow F(C\widetilde{D})$, see Equation (8).
  $\epsilon \leftarrow \epsilon\big(\text{EEVPD}_{\text{wanted}}/\text{EEVPD}\big)^{1/6}$
  $L \leftarrow$ Equation (9) $(\{\boldsymbol{x}^m\})$
  $\delta[x_i^2] \leftarrow$ Equation (10) $(\{\boldsymbol{x}^m\})$
  $t \leftarrow t+1$
until $\max_i \delta[x_i^2] > 0.01$ and $t <$ maxiter
ADAPT $\leftarrow$ True
while $t <$ maxiter **do**
  $\boldsymbol{x}^m, \boldsymbol{u}^m, a^m \leftarrow$ MAMS update $(\boldsymbol{x}^m, \epsilon)$, see Section F.
  $a = \frac{1}{M}\sum_{i=1}^M a^m$
  ADAPT $\leftarrow$ ADAPT $\wedge\, |a - a_{targeted}| > 0.03$
  if ADAPT **then**
    $\epsilon \leftarrow$ Bisection $(\epsilon, a - a_{\text{targeted}})$
  end if
end while

Algorithm 1: LAPS Pseudocode

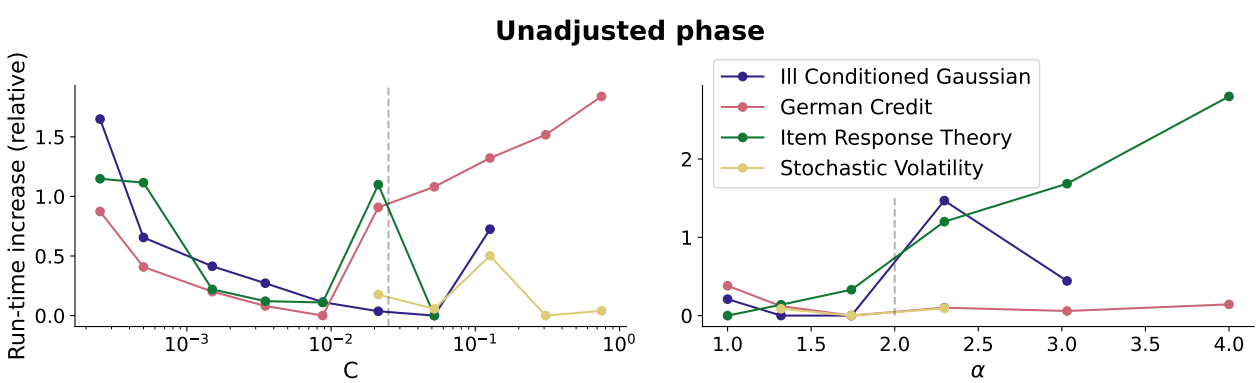

Figure 6: Ablation study of the unadjusted phase hyperparameters C and α. Run-time increase (as measured by the number of gradient calls to low bias, $b^2_{\max} < 0.01$) is measured relative to the optimal performance in this grid search. The default hyperparameter values are shown by vertical dashsed lines.

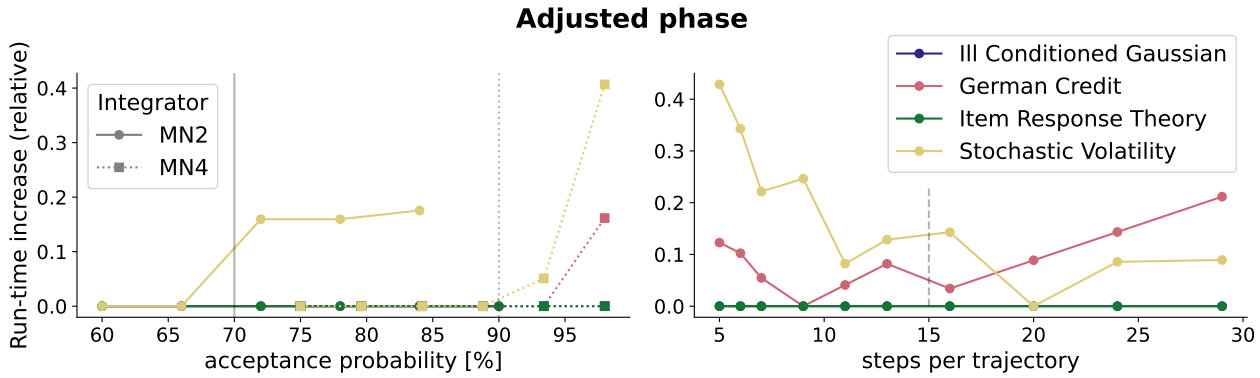

Figure 7: Ablation study of the adjusted phase hyperparameters acceptance probability and the number of steps per trajectory. Run-time increase (as measured by the number of gradient calls to low bias, $b^2_{\max} < 0.01$) is measured relative to the optimal performance in this grid search. The default hyperparameter values are shown by vertical dashsed lines. Acceptance rate results are also shown as a function of the integrator used (fourth order MN4 versus second order MN2).

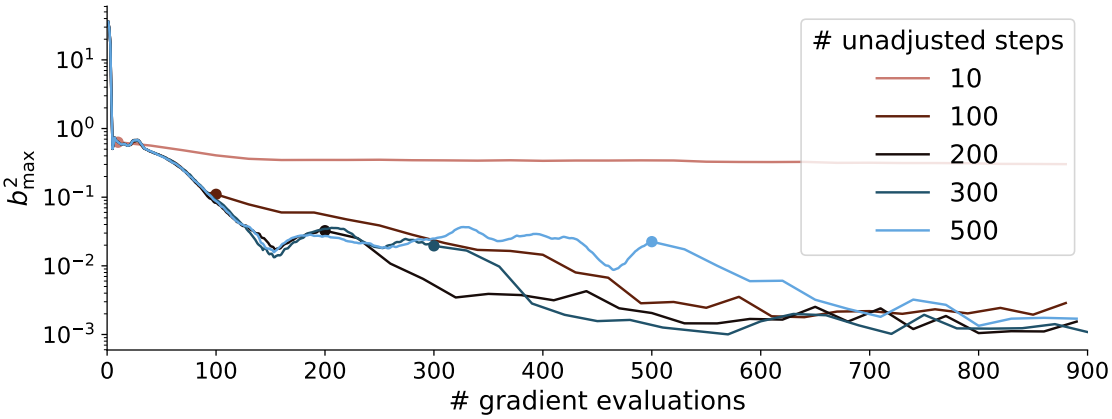

Figure 8: Dependence of the performance on the number of unadjusted steps. Bias as a function of the number of total gradient calls is shown for the German Credit problem. Different colors correspond to different number of steps in the unadjusted phase. The switch to adjustment is denoted by a dot. The adjusted method on its own (10 unadjusted steps) struggles with the cold start and does not converge in a reasonable time. If a sufficient number of unadjusted steps is taken (100) the method converges, but the adjusted part takes an unnecessarily long time. On the other hand the unadjusted method on its own does not converge to the extremely low bias, which is only achieved once the adjustment is turned on, as can be when 500 unadjusted steps are taken (light blue). In LAPS, the switch to adjustment is determined automatically by monitoring the summary statistics and in this case occurs after 200 gradient calls, which is close to optimal.

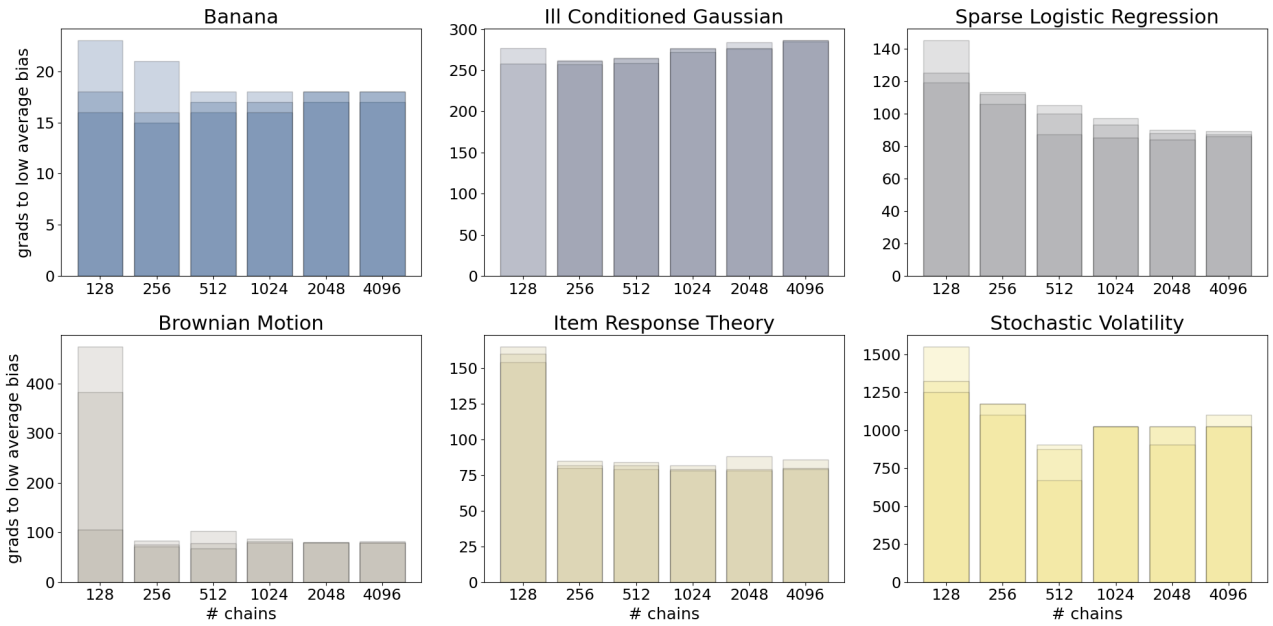

Figure 9: Number of gradient calls to low bias ($b^2_{\text{avg}} < 0.01$) as a function of number of chains M. Different color shades correspond to runs with different random seeds. It can be seen that performance stays roughly constant as long as $M > 128 - 256$, depending on the target.

Unadjusted phase. We study the performance as a function of C (ratio between the asymptotic bias and the total bias) and α (the proportionality constant relating the typical posterior size and the momentum decoherence scale L). The results are shown in Figure 6. These results demonstrate that the performance is stable in a wide range of hyperparameters. Results for some of the models could be improved for some of the models (German Credit and Item Response Theory) by a factor of two if a smaller value of C was used.

Adjusted phase. We study the performance as a function of the acceptance probability and the number of steps per trajectory. The results are shown in Figure 7. The acceptance probability ablations are shown for both MN2 and MN4 integrators, demonstrating that the choice of the integrator does not play a large role. Figure 7 shows that the performance is very stable with respect to the adjusted phase hyperparameters. For some models (Ill Conditioned Gaussian and Item Response Theory) the unadjusted phase already achieved close to desired accuracy, so these hyperparameters practically do not play a role. In all cases the change of performance in the reasonable range of hyperparameters is on the order of 10%.

Is the adjusted phase necessary? We study how does the time when we switch on adjustment impact performance. This is shown in Figure 8 for the German Credit problem. Turning on adjustment too soon dramatically slows down convergence, as expected. Unadjusted sampler on the other hand reaches a stationary state at some point and does not improve further. Adjusted method is needed for the final decay to extremely low error (determined by the variance due to the finite number of chains). Without the adjusted phase the sampler fails to achieve low error for some problems, for example for the Stochastic Volatility model.

Is the unadjusted phase necessary? We compare LAPS with and without the unadjusted phase. The algorithm with only the adjusted phase is a parallel version of the Metropolis adjusted microcanonical dynamics (MAMS). We explore two options, with initializing the sampling with 100 ADAM steps (as in (Hoffman and Sountsov, 2022)) or without. In both cases we give MAMS an (unfair) advantage of giving it the same diagonal mass matrix preconditioning that LAPS would use in the adjusted phase (i.e. the one obtained from samples after the LAPS unadjusted phase), so that we do not need to develop a custom adaptive preconditioner. We have checked that this produces better results than omitting the preconditioning and we expect it to be better than the custom preconditioner (which would be based on more biased samples in the initial steps). The results are shown

Table 3: Grads to low max bias ($b^2_{\text{max}} = 0.01$), normalized per chain, lower is better. - indicates that sampler did not converge after 2000 gradient calls. LAPS is compared with only the adjusted phase, i.e., the parallel version of MAMS. MAMS is shown to be less reliable and often fails to converge in reasonable time.

	MAMS	MAMS with ADAM	LAPS
Banana	390	670	17
Ill conditioned Gaussian	-	-	308
German Credit	270	280	300
Item Response Theory	-	1000	185
Stochastic Volatility	-	-	1325

Table 4: Ratio of the number of gradient calls to low bias for the full rank equipartition versus the diagonal one. The differences are marginal, suggesting that diagonal version, which is less memory intensive, may be used.

	Gaussian	German Credit	IRT	Stoch. Volatility
Relative run-time (full rank / diagonal)	1.003	0.954	1.102	1.000

in Table 3. As can be seen, the unadjusted initialization (LAPS) gives a much more stable performance and is in many problems the only method that converges. When all methods converge, LAPS is typically significantly faster, the only exception is German Credit when performance of all methods is similar. For example, LAPS is a factor of 18-23 times faster than MAMS on Banana. This results suggest that both adjusted and unadjusted phases are critical for the good performance.

Number of chains We study the performance as a function of the number of chains M. The results are shown in Figure 9. The performance is roughly independent of the number of chains, as long as $M > 128 - 256$, depending on the target. Lower M cannot work, because the number of samples obtained by LAPS is equal to M. Even if these are perfectly independent, unbiased samples, their accuracy is limited by the variance. $b^2_{\text{avg}} = 0.01$ accuracy roughly corresponds to 100 effective samples (Hoffman and Sountsov, 2022), so a lower number of chains typically cannot achieve this performance.

Equipartition estimation In the rest of the work we used the diagonal version of $\widetilde{D}$ as bias proxy in the unadjusted phase. Here we explore if this degrades the performance relative to the full rank $\widetilde{D}$. Table 4 compares the the number of gradient calls to low bias ($b^2_{\text{max}} < 0.01$) for both approaches. It demonstrates that on the problems considered here, the differences are marginal. Note that this also holds for the Ill Conditioned Gaussian, whose covariance matrix is randomly orientated and therefore is not biased to be close to diagonal. This suggests that there is no need to use the full rank bias proxy.

Dynamics We tried replacing the microcanonical dynamics with the more standard underdamped Langevin dynamics. We found that the unadjusted underdamped Langevin dynamics fails to aid the sampler and the scheme is significanlty slower than LAPS with the microcanonical dynamics. For example, LAPS scheme with underdamped Langevin converges to in 650 gradient calls per chain on the Item response theory. Only running the adjusted underdamped dynamics (i.e. MALT) converges in 610 gradient calls, so comparable. It is also comparable to HMC with ChESS (537 gradient calls) and underdamped Langevin with MEADS (790 gradient calls). LAPS with the microcanonical dynamics on the other hand converges significantly faster (185 gradient calls). The conclusion is that microcanonical dynamics is important for the success of our scheme.

E Integrators

In this appendix we review the numerical integrators used to approximate Equation (2). We employ operator splitting schemes that alternate between deterministic and stochastic updates (Leimkuhler and Matthews, 2015). The resulting algorithms are time-reversible, stable, and suitable for use inside a Metropolis–Hastings wrapper (Robnik et al., 2025).

Table 5: Integrators considered in this work. The integrator is defined by its coefficients, which are given in the last column. The coefficients that can be inferred from the symmetric structure or normalization are not explicitly provided.

Name	Order	Gradients/step	Best for	Coefficients
LF	2	1	low accuracy	$b_1 = 0.5$
MN2	2	2	high accuracy, low dimensionality	$b_1 = 0.1931833275$ $a_1 = 0.5$
MN4	4	5	high accuracy, high dimensionality	$b_1 = 0.0839831526$ $b_2 = 0.6822365335$ $a_1 = 0.2539785108$ $a_2 = -0.032302867$

First, we split the SDE in a stochastic and determinisitc part and approximate the combined solution by an Euler-Maruyama composition,

$$\varphi(\cdot|\epsilon, L) = \Phi^O_{\epsilon/2,L} \circ \Phi_\epsilon \circ \Phi^O_{\epsilon/2,L}. \tag{21}$$

Here, ϵ is the step size, Φ^O is the stochastic update, and Φ_ϵ is the deterministic update.

The deterministic update is obtained by further splitting the differential equation to the part that updates the velocity and the part that updates the position. The exact solutions of these two equations are called the position update Φ^A_ϵ and the velocity update Φ^B_ϵ. The combined solution is approximated by alternating these two updates:

$$\Phi_\epsilon = \Phi^B_{b_1\epsilon}\, \Phi^A_{a_1\epsilon}\, \Phi^B_{b_2\epsilon}\, \Phi^A_{a_2\epsilon} \cdots \Phi^B_{b_K\epsilon}. \tag{22}$$

Different integrators correspond to different choices of coefficients $\{a_k, b_k\}$. In all cases we require the normalization $\sum_{k=1}^K b_k = \sum_{k=1}^{K-1} a_k = 1$ and the symmetric composition ($a_k = a_{K-k}$, $b_k = b_{K+1-k}$), because it ensures time-reversibility and cancels odd-order terms in the local error expansion, thereby improving accuracy. Specifically, we consider three schemes, summarized in Table E. Each scheme trades accuracy for computational cost in a different way:

- Leapfrog (LF; Leimkuhler and Matthews (2015)) requires only 1 gradient evaluation per step. It is preferred when low accuracy is sufficient, i.e. in the unadjusted phase of the algorithm.
- Second order minimal norm integrator (MN2; Omelyan et al. (2002, 2003)) is optimized for accuracy, and comes with two gradient evaluations per step. This makes it the preferred choice in the adjusted phase of the algorithm, when we are trying to achieve fine convergence.
- Fourth order minimal norm integrator (MN4; Omelyan et al. (2002, 2003)) is more costly, as it requires five gradient calls per step. However, unlike the other two integrators, it is fourth order accurate, meaning that its local approximation error scales as ϵ^4. This means that reducing the step size has a much larger impact on the accuracy than for the second order integrators. This is crucial for the adjusted phase if the target is high-dimensional (Takaishi and de Forcrand, 2006; Hernández-Sánchez et al., 2021). We adopt it for $d > 200$.

For completeness, we below explicitly state the Φ^O, Φ^A and Φ^B updats.

Stochastic update. We follow (Robnik et al., 2024a) and write the stochastic update as

$$\Phi^O_{\epsilon,L}(\boldsymbol{x}, \boldsymbol{u}) = \left(\boldsymbol{x},\ \frac{c_1\boldsymbol{u} + c_2\boldsymbol{Z}}{\|c_1\boldsymbol{u} + c_2\boldsymbol{Z}\|}\right), \tag{23}$$

where $\boldsymbol{Z} \sim \mathcal{N}(0, I/\sqrt{d})$ is a d-dimensional vector, $c_1 = e^{-\epsilon/L}$, and $c_2 = \sqrt{1 - c_1^2}$.

Position update. At fixed velocity, the dynamics from Equation (2) reduces to

$$\frac{d}{dt}\boldsymbol{x}_t = \boldsymbol{u},$$

which has the trivial solution

$$\Phi_\epsilon^A : \quad (\boldsymbol{x}, \boldsymbol{u}) \mapsto (\boldsymbol{x} + \epsilon \boldsymbol{u}, \boldsymbol{u}). \tag{24}$$

The energy change induced by this update is $\Delta(\boldsymbol{x}, \boldsymbol{u}|p, \epsilon) = -\log p(\boldsymbol{x} + \epsilon \boldsymbol{u}) + \log p(\boldsymbol{x})$

Velocity update. At fixed position, the deterministic part of the dynamics from Equation (2) reduces to

$$\tfrac{d}{dt}\boldsymbol{u}_t = (I - \boldsymbol{u}_t \boldsymbol{u}_t^T)\, \frac{\nabla \log p(\boldsymbol{x})}{d-1}.$$

This is the nonlinear Riccati vector equation. However, it can be reduced to a second-order linear ODE by introducing the variable $\boldsymbol{y}_t$ via $\boldsymbol{u}_t = \frac{d}{dt}\boldsymbol{y}_t/(\boldsymbol{g} \cdot \boldsymbol{y}_t)$. One obtains (Steeg and Galstyan, 2021)

$$\tfrac{d^2}{dt^2}\boldsymbol{y} = (\boldsymbol{g}\boldsymbol{g}^T)\boldsymbol{y},$$

which is a linear system with analytic solution. We have denoted $\boldsymbol{g} = \frac{\nabla \log p(\boldsymbol{x})}{d-1}$ and $\boldsymbol{e} = \frac{\boldsymbol{g}}{\|\boldsymbol{g}\|}$. Solving the linear equation and translating back to the $\boldsymbol{u}$ variable, one arrives at the closed-form update

$$\Phi_\epsilon^B : \quad (\boldsymbol{x}, \boldsymbol{u}) \mapsto (\boldsymbol{x}, \frac{\boldsymbol{u} + \big(\sinh\delta + (\boldsymbol{e}\cdot\boldsymbol{u})(\cosh\delta - 1)\big)\boldsymbol{e}}{\cosh\delta + (\boldsymbol{e}\cdot\boldsymbol{u})\sinh\delta}), \tag{25}$$

where $\delta = \epsilon\|\nabla \log p(\boldsymbol{x})\|/(d-1)$. Thus the velocity rotates smoothly in the plane spanned by $\boldsymbol{u}$ and $\boldsymbol{g}$, while maintaining unit norm. The energy change induced by this update is (Steeg and Galstyan, 2021):

$$\Delta(\boldsymbol{x}, \boldsymbol{u}|p, \epsilon) = (d-1)\log\Big\{\cosh\delta + (\boldsymbol{e}\cdot\boldsymbol{u})\sinh\delta\Big\}. \tag{26}$$

F Metropolis adjusted microcanonical dynamics

In this appendix we review the Metropolis-adjusted microcanonical dynamics (MAMS; Robnik et al. (2025)) that LAPS uses in the adjusted phase of the algorithm. MAMS kernel is constructed by the following steps:

1. Start from $(\boldsymbol{x}, \boldsymbol{u})$.
2. Resample $\boldsymbol{u}$ from a uniform distribution on the unit sphere. Specifically $\boldsymbol{u} = Z/\|Z\|$, with $Z \sim \mathcal{N}(0, I)$.
3. Apply N steps of the splitting integrator with step size ϵ, including partial velocity refreshment at each step, to obtain $(\boldsymbol{x}', \boldsymbol{u}')$.
4. Compute the energy error Δ accumulated in this integration. Each position update $\boldsymbol{x} \to \boldsymbol{x}'$ contributes $-\log p(\boldsymbol{x}')/p(\boldsymbol{x})$ to the energy error. Each velocity update contributes $(d-1)\log\Big\{\cosh\delta + (\boldsymbol{e}\cdot\boldsymbol{u})\sinh\delta\Big\}$, where $\delta = \epsilon\|\nabla \log p(\boldsymbol{x})\|/(d-1)$ and $\boldsymbol{e} = \nabla \log p(\boldsymbol{x})/\|\nabla \log p(\boldsymbol{x})\|$, see Section E.
5. Accept the proposal with probability $\min(1, e^{-\Delta})$, otherwise remain at $(\boldsymbol{x}, \boldsymbol{u})$.

The stationary distribution of this kernel is exactly the target distribution, the integrator approximation error is eliminated by the Metropolis–Hastings (MH) step.