Event-Chain Monte Carlo for Yang-Mills SU(N) lattice field theory I : Design and proof of concept

Benoît Blossier,[a,1] **Manon Michel,**[b,2] **Yacob Ozdalkiran**[a,3]

[a] *Laboratoire de Physique des 2 Infinis Irène Joliot-Curie, CNRS/IN2P3, Université Paris-Saclay, 91405 Orsay Cedex, France*

[b] *Laboratoire de Mathématiques Blaise Pascal UMR 6620, CNRS, Université Clermont-Auvergne, Aubière, France*

Abstract: We develop two implementations of the Event-Chain Monte Carlo (ECMC) algorithm for Yang-Mills SU(N) lattice gauge theories with the Wilson action. These algorithms consist in a succession of local ballistic updates intersped with stochastic events, resulting in an irreversible and rejection-free Markov process. The resulting dynamics satisfy global balance, ensuring the correct equilibrium distribution. The algorithms are formulated for general SU(N) Yang-Mills theories with Wilson action and implemented for the case $N = 3$. Numerical tests on four-dimensional lattices show that standard gauge observables, such as the mean plaquette, agree with results obtained using conventional Monte Carlo algorithms. These results provide a first validation of ECMC as a viable sampling scheme for Yang-Mills lattice gauge theories.

[1] blossier@ijclab.in2p3.fr
[2] manon.michel@uca.fr
[3] ozdalkiran@ijclab.in2p3.fr

Contents

1. Introduction

Lattice gauge theory provides the only systematically improvable framework for the non-perturbative study of Quantum Chromodynamics (QCD). In this approach, expectation values of observables are computed by sampling of gauge field configurations distributed according to the Boltzmann factor appearing in the Euclidean path integral. In the case of pure gauge theories described by the Wilson action [1] and its $O(a^2)$ improved formulations, a variety of Markov Chain Monte Carlo (MCMC) algorithms have been developed over the past decades, including local update schemes such as the Heatbath [2] and over-relaxation [3] algorithms, as well as global approaches such as Hybrid Monte Carlo (HMC) [4] when fermions are present. While these methods have proven highly successful, they remain affected by critical slowing down as the lattice spacing is finer and finer [5, 6]. In particular, autocorrelation time of the topological charge diverges in the continuum limit, meaning that topological modes become increasingly difficult to sample efficiently.

Motivated by these limitations, a considerable effort has been devoted to exploring alternative sampling strategies that modify the dynamical properties of the underlying Markov process. Among them, Open Boundary Conditions (OBC) [7] and Parallel Tempering on Boundary Conditions (PTBC) [8], biasing and density-of-states techniques like PT-MetaD [9] and Log-Likelyhood Ratio (LLR) [10], as well as Master-Field simulations [11] and generative approaches via Normalizing Flows [12]. While promising, these methods remain active areas of research and development to fully overcome topological freezing.

Another direction explored in this paper consists in relaxing the requirement of detailed balance while preserving global balance, the only necessary condition to ensure the invariance of the target probability distribution. Algorithms in this class generate irreversible Markov chains exhibiting dynamical persistency in configuration space, which can lead to significantly improved sampling efficiency [13]. Among such methods, Event-Chain Monte Carlo (ECMC) [14] has received increasing attention in recent years. Originally introduced for systems of classical particles, ECMC is based on a lifted Markov process in which the dynamics proceeds through continuous updates interrupted by stochastic "events". The resulting algorithm satisfies global balance but violates detailed balance. Thus it allows for persistent directed motion in configuration space. ECMC has been successfully applied to a variety of statistical physics models, including hard-sphere systems [15], lattice spin models [16], bosonized one-dimensional quantum systems [17] and asymptotically free lattice field theories [18] where substantial reductions in autocorrelation times have been observed.

The extension of Event-Chain Monte Carlo algorithm to lattice gauge theories presents several conceptual and practical challenges. Gauge fields take values in compact non-Abelian Lie groups and the lattice action couples link variables through closed paths. It leads to a highly constrained configuration space. Designing continuous updates interspersed with stochastic events that respect the group structure while maintaining the correct equilibrium distribution is therefore non-trivial. Although Event-Chain algorithms have been explored in several lattice field theory contexts [18], a formulation of ECMC suitable for Yang-Mills gauge theories remains largely unexplored.

In this paper we present an adaptation of Event-Chain Monte Carlo for Yang-Mills SU(N) lattice gauge theories regularized by the Wilson action. The construction relies on the factorization of the gauge action into plaquette contributions and on continuous updates of link variables along

a one-parameter subgroups of the gauge group permitted by the XY-Embedding [19].

We develop two variations of the Event-Chain algorithm for SU(N) lattice gauge theories and discuss their implementation in the case of SU(3), which is the gauge lattice QCD in quenched approximation. The first variation, denoted Reflective ECMC, can easily be generalized to any lattice field theory with factorizable action but allows backtracking. We built on the Forward generalization [20] to propose a second variant that exploits symmetries of the system itself to avoid any backtracking. More precisely, this second Forward version uses the structure of the Wilson action and in particular the symmetry properties of closed loops, thus extending naturally to $O(a^2)$ improved actions.

The paper is organized as follows. Section 2 presents the formalism of Yang-Mills SU(N) lattice gauge theory for completeness and introduces the Wilson action with the relevant objects. Section 3 presents the Event-Chain algorithm using a discrete infinitesimal approximation for clarity, followed by the details of our two implementations, namely Reflective ECMC in subsection 3.5 and Forward ECMC in subsection 3.6. Section 4 provides the formulation of our algorithms in the rigorous Piecewise Deterministic Markov Process (PDMP) formalism and the proof of convergence. Finally, section 5 reports first numerical tests for quenched lattice QCD defined by the Wilson action, it presents the mean value and autocorrelation time of the plaquette (a short-distance observable) and compares results to existing algorithms to ensure correctness of our implementations. We conclude in Section 6. Appendix A provides details of the runs presented in Section 5.

2. Wilson lattice gauge theory

We consider a pure SU(N) lattice gauge theory defined on an Euclidean time four-dimensional hyper cubic lattice Λ with periodic boundary conditions. The fundamental degrees of freedom are group-valued link variables

$$U_{x,\mu} \in \mathrm{SU}(N), \tag{2.1}$$

associated with oriented links connecting a lattice site x to a neighboring site $x + \hat{\mu}$, where $\mu = 1, \ldots, 4$ denotes the lattice directions and $\hat{\mu}$ the associated canonical base vector. From now on we assume the lattice spacing $a = 1$ for clear notations. Gauge transformations act locally at lattice sites according to

$$U_{x,\mu} \;\rightarrow\; \Omega_x \, U_{x,\mu} \, \Omega^{\dagger}_{x+\hat{\mu}}, \qquad \Omega_x \in \mathrm{SU}(N). \tag{2.2}$$

By definition, any gauge-invariant observable remains unchanged under that transformation.

The elementary gauge-invariant objects are the plaquettes, defined as traces of the ordered product of four link variables around a square of length 1:

$$U_{x,\mu\nu} = U_{x,\mu} U_{x+\hat{\mu},\nu} U^{\dagger}_{x+\hat{\nu},\mu} U^{\dagger}_{x,\nu}. \tag{2.3}$$

In the Wilson regularization, the dynamics of the theory is governed by the action

$$S[U] = \beta \sum_{x \in \Lambda} \sum_{\mu < \nu} \left(1 - \frac{1}{N} \mathrm{Re\,Tr} \left[U_{x,\mu\nu} \right] \right), \tag{2.4}$$

where $\beta = 2N/g^2$ is the inverse bare coupling. The partition function is given by

$$Z = \int \mathcal{D}U \, \mathrm{e}^{-S[U]}, \tag{2.5}$$

where $\mathcal{D}U = \prod_{x,\mu} \mathrm{d}U_{x,\mu}$, the product of Haar measures on $\mathrm{SU}(N)$.

Expectation values of observables $O[U]$ are computed as

$$\langle O \rangle = \frac{1}{Z} \int \mathcal{D}U \, O[U] \, \mathrm{e}^{-S[U]}. \tag{2.6}$$

In lattice simulations these expectation values are estimated through importance sampling of gauge configurations distributed according to π such that

$$\mathrm{d}\pi(U) = \frac{1}{Z} \mathrm{e}^{-S[U]} \mathcal{D}U. \tag{2.7}$$

Direct sampling of independent gauge configurations distributed according to (2.7) cannot be done because of the huge dimensions of the configuration space and the complex form of the action. To sample gauge configurations efficiently, one uses Markov Chain algorithms designed to iteratively evolve an initial state towards the equilibrium distribution by performing either local updates such as Heatbath [2] or Metropolis [21], which update individual links, or perform global moves like Hybrid Monte Carlo (HMC) [4] which propose transitions for the entire lattice simultaneously. The Event-Chain algorithm is a local algorithm, however able to produce collective updates thanks to the persistency of the generated dynamics.

For algorithms that update individual link variables, it is convenient to express the action in terms of the local contribution associated with a given link $U_{x,\mu}$. When updating a link $U_{x,\mu} \to U'_{x,\mu}$, the difference in action depends only on the links of the surrounding plaquettes

$$\Delta S(U_{x,\mu} \to U'_{x,\mu}) = -\frac{\beta}{N} \operatorname{Re} \operatorname{Tr} \left[(U'_{x,\mu} - U_{x,\mu}) V_{x,\mu} \right], \tag{2.8}$$

where the local term depends on the neighboring links through the so-called staple matrix

$$V_{x,\mu} = \sum_{\nu \neq \mu} \left(U_{x+\hat{\mu},\nu} U^{\dagger}_{x+\hat{\nu},\mu} U^{\dagger}_{x,\nu} + U^{\dagger}_{x+\hat{\mu}-\hat{\nu},\nu} U^{\dagger}_{x-\hat{\nu},\mu} U_{x-\hat{\nu},\nu} \right) = \sum_{j=1}^{2(d-1)} V^{j}_{x,\mu}. \tag{2.9}$$

Figure 1 provides a geometric representation of the staples. Thus each link interacts only with other links in $2(d-1)$ surrounding plaquettes: $V^{j}_{x,\mu}$ is the staple matrix corresponding to the j-th plaquette attached to the link $U_{x,\mu}$. Such a factorization of the action into local interaction terms plays a central role in the construction of update algorithms. In particular, we can decompose the Boltzmann weight into contributions that correspond to individual plaquettes, a property that will be exploited in the formulation of the Event-Chain Monte Carlo dynamics described in next sections.

3. Event-Chain Monte Carlo

ECMC is a continuous-time and rejection-free Markov Chain Monte Carlo scheme. Its implementation takes place in several steps. The appropriate formalism to define an Event-Chain algorithm is the Piecewise Deterministic Markov Process (PDMP) formalism [22] presented in section 4, but for the sake of pedagogy we present in this section the algorithm using an infinitesimal discrete Markov Chain approximation.

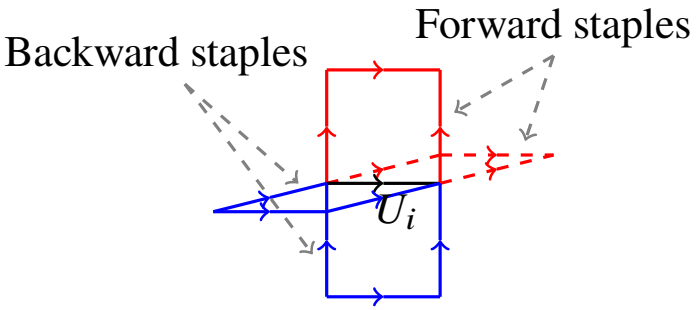


Figure 1: A link U_i (in black) and four of its surrounding staples. Staples in blue are backward staples (i.e. of type $U_1^\dagger U_2^\dagger U_3$) and forward staples (i.e. of type $U_1 U_2^\dagger U_3^\dagger$) are depicted in red. Each staple (or each plaquette if we also count the link U_i) accounts for an independent factor of the factorized Metropolis filter (cf. section 3.1).

3.1. Factorization of the Metropolis filter

First, the variation of action during the update of an element of the gauge configuration, in our case a link, needs to be separated in a sum of factors depending on the surrounding degrees of freedom of the system. This allows us to factorize the Metropolis filter in a product of independent factors.

Let us denote from now on the links by a single index $i = 1, 2, ..., N_l$ with N_l the number of links. In the case of Wilson action, eq. (2.8) and (2.9) yields a decomposition

$$\Delta S(U_i \to U_i') = \sum_{j=1}^{2(d-1)} \Delta S_j(U_i \to U_i') \tag{3.1}$$

with

$$\Delta S_j(U_i \to U_i') = -\frac{\beta}{N} \operatorname{Re}\operatorname{Tr}\left[(U_i' - U_i)V_i^j\right]. \tag{3.2}$$

The regular Metropolis filter [23] is defined by

$$p(U_i \to U_i') = \min\left(1, \mathrm{e}^{-\Delta S(U_i \to U_i')}\right). \tag{3.3}$$

Using the decomposition (3.1) we write the factorized Metropolis filter [24] instead defined by

$$p^{\text{fact}}(U_i \to U_i') = \prod_{j=1}^{2(d-1)} \min\left(1, \mathrm{e}^{-\Delta S_j(U_i \to U_i')}\right). \tag{3.4}$$

Although the factorized Metropolis filter yields a smaller acceptance probability than the conventional Metropolis criterion, we can express the interaction of a link with its neighboring plaquettes as a sum of independent factors. This property is crucial for ECMC dynamics as it allows each plaquette contribution to be treated as an independent interaction term.

3.2. Updates and parametrization of SU(N)

After factorizing the Metropolis filter, we need a way to explore SU(N) during the link updates. We choose the XY-Embedding framework [19] as it can be easily generalized to any N. We will see in subsection 3.4 that it also makes the event generation part of the algorithm analytical and efficient.

Following [19], we update continuously the selected link variable through left multiplication parametrized as

$$U_i \to U_i(\theta) = R\mathrm{e}^{\mathrm{i}\epsilon\,\theta T} R^\dagger U_i, \tag{3.5}$$

where

$$\mathrm{i}T = \mathrm{diag}(\mathrm{i}, -\mathrm{i}, 0, \ldots, 0) \tag{3.6}$$

is a generator of $\mathfrak{su}(N)$ algebra, that are anti-hermitian traceless $N \times N$ matrices, θ is the evolution angle of the continuous update, R a SU(N) matrix chosen randomly according to the Haar measure and $\epsilon \in \{-1, 1\}$ the direction of the update. Regular resampling of R assures the ergodic exploration of SU(N).

The difference in action caused by an update of θ on the link i reads

$$\Delta S_{i,R,\epsilon}(\theta) = \sum_{j=1}^{2(d-1)} -\frac{\beta}{N} \mathrm{Re}\,\mathrm{Tr}\left[R\mathrm{e}^{\mathrm{i}\theta T} R^\dagger U_i V_i^j\right]. \tag{3.7}$$

In particular, this continuous updating process can be expressed as the limit of successive infinitesimal updates of angle $\mathrm{d}\theta$, each accepted with the factorized Metropolis probability

$$p^{\mathrm{fact}}(U_i \to U_i(\mathrm{d}\theta)) = \prod_{j=1}^{2(d-1)} \min\left(1, \mathrm{e}^{-\partial_\theta S^j_{i,R,\epsilon}(0)\mathrm{d}\theta}\right) = \prod_{j=1}^{2(d-1)} \mathrm{e}^{-[\partial_\theta S^j_{i,R,\epsilon}(0)]^+ \mathrm{d}\theta}, \tag{3.8}$$

where each factor corresponds to the contribution of a single plaquette that includes the link i and $[x]^+ = \max(0, x)$. The associated action increments at order $\mathrm{d}\theta$ are

$$\partial_\theta S^j_{i,R,\epsilon}(0) = \epsilon \frac{\beta}{N} \mathrm{Im}\,\mathrm{Tr}\left[RTR^\dagger U_i V_i^j\right]. \tag{3.9}$$

The Event-Chain is the infinitesimal limit of the following Markov Chain algorithm : we apply successive infinitesimal updates of angle $\mathrm{d}\theta$ on the chosen link i as long as all the factors of (3.4) accept the update. The evolution continues until an *lifting event* (or *lift*) occurs. Lifting events arise from the rejection of the infinitesimal update by one of the factors of the acceptance probability. A lift proceeds as follows : when a staple V_i^j rejects the update, we stop updating the link i and start updating a link k belonging to the staple V_i^j instead. When $\mathrm{d}\theta \to 0$, we can show [14] that the probability of two factors or more rejecting an update goes to zero. Therefore the rejecting factor is uniquely determined.

During a lift we can also change the values of ϵ and R, but in order to ensure convergence towards the right equilibrium distribution (2.7) we need to make sure that our algorithm respects global balance. In case of SU(N) Yang-Mills lattice field theory regularized by Wilson action, we found two possible lift rules fulfilling global balance. We detail them in subsections 3.5 and 3.6.

We also need to define a total displacement angle $\theta_{\mathrm{refresh}}$ upon which we break the chain and choose randomly with uniform probability a new link and a new R matrix to ensure ergodicity of the algorithm.

Sampling of gauge configuration happens after a fixed total displacement angle $\theta_{\mathrm{sample}} > \theta_{\mathrm{refresh}}$. Setting a fixed total displacement angle for sampling ensures that we do not sample at events, as this would introduce a bias easily seen in the values of observables.

3.3. Global balance and lifted state space

The global balance equation is the main requirement to ensure that a Markov Chain algorithm converges towards the right probability distribution π on a configuration space Ω. In case of ECMC algorithms, we consider instead an extended version of the configuration space Ω called the *lifted state space*. For SU(N) lattice gauge theory on a 4 dimensional lattice of N_l links, we consider the lift variables

$$(i, R, \epsilon) \in \mathcal{V} = [\![1, N_l]\!] \times \mathrm{SU}(N) \times \{-1, 1\}, \tag{3.10}$$

which denote respectively the index of the link currently being updated, the rotation applied on the $\mathfrak{su}(N)$ generator used in the XY-embedding updates (3.5) and the direction of the update. A configuration in the lifted state space is then characterized by the tuple

$$W = (U, i, R, \epsilon) \in \mathrm{SU}(N)^{N_l} \times \mathcal{V}, \tag{3.11}$$

with $U \in \Omega \coloneqq \mathrm{SU}(N)^{N_l}$ the gauge configuration. Denoting μ the product of uniform probability distributions on the lift variables of space $\mathcal{V}$, the target probability distribution on the lifted state space is $\pi \otimes \mu$.

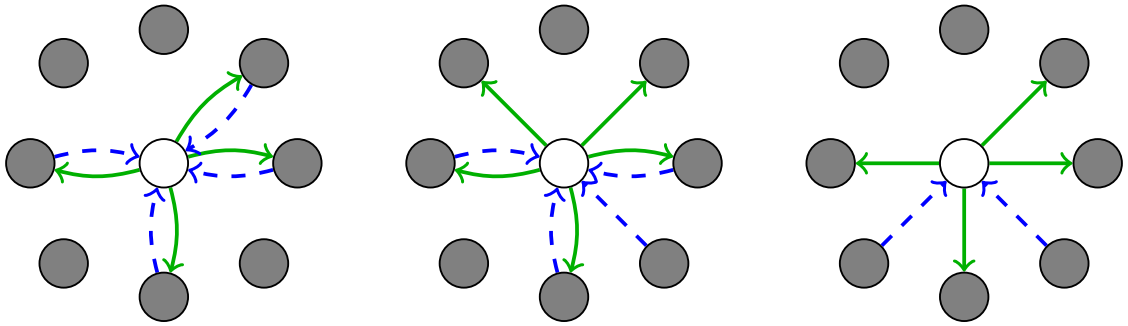
Given a configuration space (in our case $\Omega \times \mathcal{V}$) and a transition operator T, the global balance equation states that for any lifted configuration, the sum of the probability flows to enter this state during the chain, so-called incoming probability flows, must equate the sum of the probabilities of departing from this state during the chain, so-called outgoing probability flows. Hence we write for any $W \in \Omega \times \mathcal{V}$

$$\sum_{W' \in \Omega \times \mathcal{V}} \pi \otimes \mu(W) T(W \to W') = \pi \otimes \mu(W) = \sum_{W' \in \Omega \times \mathcal{V}} \pi(W') T(W' \to W). \tag{3.12}$$

Usual MCMC algorithms enforce a stronger version called *detailed* balance on Ω where the equality (3.12) is term-wise. The purpose of ECMC is to fulfill global balance without the detailed version in the lifted configuration space. It ensures a ballistic exploration of the configuration space instead of a diffusive one. Figure 2 depicts the different ways algorithms fulfill (3.12). Algorithms such as Metropolis, Heatbath and HMC verify the detailed balance, while both algorithms presented in section 3.5 and 3.6 preserve a maximally irreversible global balance in the lifted state space. We show convergence of our ECMC algorithms using the rigorous PDMP formalism in section 4.2.

3.4. Event-driven approach

The Event-Chain algorithm generates a continuous-time Markov process in the lifted configuration space but that can be exactly simulated by an event-driven approach [14, 15, 25] yielding the discrete sequences of events. Event angles θ^j_{event} are sampled for each farctor j and represent a realization of the Event-Chain where the j-th factor accepted the update of the link up to θ^j_{event} and then rejected the update. Factorization of the Metropolis filter allows us to treat each factor independently. We can then update in a single step the link with an angle $\theta_{\text{event}} = \min_{j=1,\dots,2(d-1)} \theta^j_{\text{event}}$, i.e. up to the point where one of the factors rejects the update. This update of the chosen link i between two lifts, in a purely deterministic way and in a single step, does achieve a substantial speed-up with respect to usual MCMC algorithms with infinitesimal updates such as Metropolis algorithm. This allows



(a) Detailed (b) Partially irreversible (c) Maximally irreversible

Figure 2: Illustration of detailed balance (2a), partially irreversible global balance (2b) and maximally irreversible global balance (2c). We depict the incoming (in blue, dashed) and outgoing (in green) probability flows into a configuration of the considered state space Ω.

for a ballistic exploration of the configuration space, by opposition to the diffusive exploration of MCMC algorithms designed with local updates.

We can sample event angles for each member j of the factorized Metropolis filter by solving for θ^j_{event} with u drawn uniformly in $(0, 1]$ [14–16, 25]

$$-\ln(u) = \int_0^{\theta^j_{\text{event}}} [\partial_\theta S^j_{i,R,\epsilon}(\theta)]^+ \mathrm{d}\theta. \tag{3.13}$$

The equation (3.13) can be retrieved in the infinitesimal approximation of the Event-chain as detailed in [16]. Using the XY-Embedding, updating the links with (3.5) we obtain an embedded Wilson action (we write only the θ-dependent part)

$$S^j_{i,R,\epsilon}(\theta) = -\frac{\beta}{N}(A^j_i \cos(\theta) + B^j_i \epsilon \sin(\theta)) = -\frac{\beta}{N} C^j_i \cos\big(\theta - \theta^j\big), \tag{3.14}$$

with

$$A^j_i = \mathrm{Re}[(R^\dagger U_i V^j_i R)_{11} + (R^\dagger U_i V^j_i R)_{22}], \tag{3.15}$$

$$B^j_i = \mathrm{Im}[(R^\dagger U_i V^j_i R)_{11} - (R^\dagger U_i V^j_i R)_{22}], \tag{3.16}$$

$$C^j_i = \sqrt{(A^j_i)^2 + (B^j_i)^2}, \tag{3.17}$$

$$\theta^j_i = \arctan\Big(\epsilon B^j_i / A^j_i\Big). \tag{3.18}$$

The analytical resolution of (3.13) made possible with the form of (3.14) ensures an efficient sampling of event angles and a continuous exploration of SU(N) for each link visited by the chain.

3.5. Reflective ECMC lift rule

After updating the link up to the angle θ_{event}, we then proceed to *lift* to another state of the lifted state space by changing the lift variables $(i, \epsilon) \to (k, \hat{\epsilon})$ with k a link of the plaquette P^j responsible for θ_{event}. We lift towards a link k of this plaquette with probability

$$\lambda_{(i,\epsilon)\to(k,\hat{\epsilon})} = \frac{|\partial_\theta S^j_{k,R}(0)|}{\sum_{l\in P^j} |\partial_\theta S^j_{l,R}(0)|} \delta(\hat{\epsilon} + \mathrm{sign}(\partial_\theta S^j_{k,R}(0))), \tag{3.19}$$

	Forward		Backward	
k	R_k	ϵ_k	R_k	ϵ_k
2	$U_1^\dagger R$	$-\epsilon$	$U_3^\dagger U_4 R$	ϵ
3	$U_4^\dagger R$	ϵ	$U_4 R$	ϵ
4	R	ϵ	$U_4 R$	$-\epsilon$

Table 1: Lifting rules for R_k and ϵ_k depending on the plaquette type and the link k position.

using the notation $\partial_\theta S^j_{k,R} = \frac{\beta}{N} \operatorname{Im} \operatorname{Tr}\left[T R^\dagger U_i V_i^j R\right] \mathrm{d}\theta$. This lift rule is a variant of [26] where we enable lifts to the same link i with opposite update direction ϵ. This addition is mandatory for the conservation of global balance (cf. section 4.2) as the system does not exhibit the same symmetries as [26]. Indeed, the non-Abelian nature of SU(N) yields

$$\sum_{i \in P^j} \partial_\theta S_i^j(0) \neq 0. \tag{3.20}$$

This lifting rule can therefore induce backtracking. However, this lift rule is general and not tied to the Wilson action.

Then we repeat the same steps with the new link k selected and $\hat{\epsilon} = -\operatorname{sign}(\partial_\theta S^j_{k,R}(0))$.

3.6. Forward ECMC lift rule

The reflective lift rule previously discussed generates backtracking due to the non-Abelian nature of SU(N). It is however possible to exploit an intrinsic symmetry of the considered system, namely the invariance of the Wilson action under charge conjugation and the geometry of the plaquette, i.e. a closed path. This enables the definition of an alternative lifting rule which completely suppresses backtracking. When an event arises, we choose one of the links k of the plaquette P^j involved in the event, $k \neq i$, with uniform probability $1/3$ and perform the lift

$$(U, i, R, \epsilon) \to (U, k, R_k, \epsilon_k). \tag{3.21}$$

(R_k, ϵ_k) depends wether P^j is a forward or a backward plaquette (cf. Fig. 4) and on the position of link k within this plaquette.

Let's index the current link of the chain with $i = 1$ and the other links of the rejecting plaquette with subsequent integers such that $P^j = U_1 U_2 U_3^\dagger U_4^\dagger$ if the plaquette is forward for U_1 and $P^j = U_1 U_2^\dagger U_3^\dagger U_4$ if the plaquette is backward. The corresponding lifting rules are detailed in table 1.

Thanks to the suppression of backtracking, this irreversible dynamic induces an extra speed-up with respect to the reflective dynamics shown in Section 5. The frequent updates of R also enables a more efficient exploration of SU(N). The proof of the conservation of global balance in this framework is detailed in section 4.2 and relies on the fact that for each (i, k) in plaquette P^j, a lift from (i, R, ϵ) to (k, R_k, ϵ_k) verifies $\partial_\theta S^j_{k,R_k,\epsilon_k}(0) = -\partial_\theta S^j_{i,R,\epsilon}(0)$.

This lifting rule is action specific, but an Event-Chain algorithm with similar irreversible lift rules can be implemented with improved action containing traces of closed loops such as the

Lüscher-Weisz [27] or the Iwasaki [28] actions in case of quenched gauge lattice QCD. Those actions are also factorizable and for a lift from link i towards a link k of a factor j, namely a trace of a closed loop containing i and k, we can always define (R_k, ϵ_k) such that $\partial_\theta S^j_{k,R_k,\epsilon_k}(0) = -\partial_\theta S^j_{i,R,\epsilon}(0)$, using hermitian conjugation and cyclic permutations inside the trace.

3.7. Summary

To summarize the Event-Chain algorithm with XY-embedding:

1. Define a total displacement length Θ_{sample} upon which the sample is saved, and a total displacement length $\Theta_{\text{refresh}} \ll \Theta_{\text{sample}}$ upon which we refresh the matrix R, the direction ϵ and the link to update.

2. Initialize total displacement counters θ_{refresh} and θ_{sample} to zero.

3. Choose a link i, a matrix $R \in \text{SU}(N)$ and $\epsilon \in \{-1, 1\}$ randomly with uniform probability.

4. Generate a reject angle θ^j_{event} for each of the plaquettes P^j, $j = 1, 2, ..., 2(d-1)$ attached to the chosen link. The smallest angle $\theta^j_{\text{event}} = \theta_{\text{event}}$ will be the angle chosen for the update. (it means that the corresponding plaquette j was the first one to reject a move)

5. Update the link i according to (3.5) with $\theta = \min(\theta_{\text{event}}, \Theta_{\text{sample}} - \theta_{\text{sample}}, \Theta_{\text{refresh}} - \theta_{\text{refresh}})$.

   (a) if $\theta = \theta_{\text{event}}$, update the total displacement counters $\theta_{\text{refresh}} \to \theta_{\text{refresh}} + \theta$ and $\theta_{\text{sample}} \to \theta_{\text{sample}} + \theta$ and go to step 6.

   (b) if $\theta = \Theta_{\text{refresh}} - \theta_{\text{refresh}}$, update $\theta_{\text{refresh}} \to 0$, $\theta_{\text{sample}} \to \theta_{\text{sample}} + \theta$ and go to step 3.

   (c) if $\theta = \Theta_{\text{sample}} - \theta_{\text{sample}}$, save the configuration as a sample, set $\theta_{\text{sample}} \to 0$, $\theta_{\text{refresh}} \to \theta_{\text{refresh}} + \theta$. Then redo step 5 with $\theta_{\text{event}} \to \theta_{\text{event}} - \theta$.

6. Lift by choosing one of the links k of P^j :

   - with uniform probability if we use the forward lift rule. Also change (R, ϵ) according to Table 1,
   - with probability (3.19) if we use the reflective lift rule. Also change $\epsilon \to -\text{sign}(\partial_\theta S^j_{k,R}(0))$.

   and repeat from step 4.

Figure 3 depicts the flow of the algorithm. The refreshment steps straightforwardly ensures the ergodicity of such ECMC algorithm. It is also possible to split the refresh of each of the lift variables (i, R, ϵ) by defining a counter θ and a total displacement angle Θ for each, but the gain in autocorrelation time is minor with respect to the large increase of the cost of fine tuning those parameters.

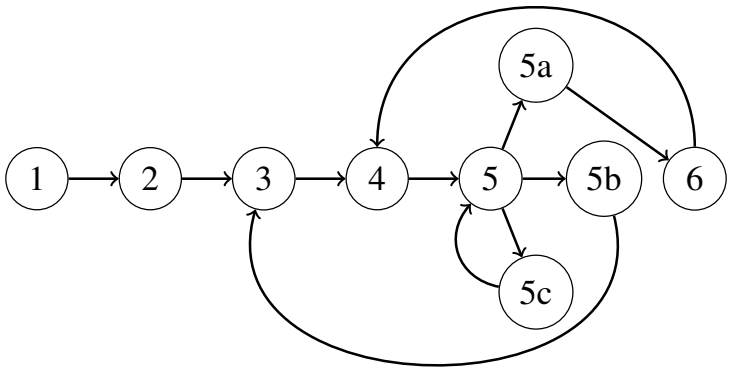


Figure 3: Execution flow of the ECMC algorithm. The numbered nodes correspond to the steps outlined in the summary of the algorithm.

4. PDMP formalism

The efficient formalism to define an ECMC algorithm is the Piecewise Deterministic Markov Process (PDMP) framework [22, 29]. Starting from any lifted state $(U, v) \in \Omega \times \mathcal{V}$, an Event-Chain can be seen as a process $\{U(\theta), v(\theta)\}_{\theta \geq 0}$ on the lifted configuration space $\Omega \times \mathcal{V}$. $U(\theta) = (U_1(\theta), U_2(\theta), ..., U_{N_l}(\theta)) \in \Omega$ denotes the evolution of a gauge configuration with N_l links, starting from $U(0) \equiv U$, during the ECMC process and $v(\theta) = (i(\theta), R(\theta), \epsilon(\theta)) \in \mathcal{V}$ the evolution of the lift variables, starting from $V(0) \equiv V$, during the ECMC process. We can then characterize an Event-Chain algorithm with an infinitesimal generator $\mathcal{A}$. This framework allows a definition of the ECMC process directly as a continuous-time process, without resorting to the infinitesimal Markov Chain limit, and yields a simple condition that the stationary distribution must satisfy.

4.1. Definitions

Following [22], we define the infinitesimal generator $\mathcal{A}$ by

$$\mathcal{A}f = \lim_{\theta \to 0} \frac{\mathbb{E}_{U,v}[f(U(\theta), v(\theta)) - f(U, v)]}{\theta}. \tag{4.1}$$

The ECMC process can be decomposed in two parts : a deterministic flow ϕ corresponding to the macroscopic updates between two lifts and stochastic events leading to lift and refreshment events affecting the lift variables.

In our case, the deterministic flow corresponds to the updates (3.5) of the chosen link. Thus we define for $\theta \geq 0$

$$(\dot{U}, \dot{v}) = \phi(U, v) = (\phi_1(U, v), \phi_2(U, v)), \tag{4.2}$$

with the dot denoting derivation with respect to θ and

$$\begin{aligned} \phi_1(U, v) &= (0, 0, ..., \underbrace{\epsilon R \mathrm{i} T R^\dagger U_i}_{\text{coordinate } i}, 0, ..., 0) \quad \text{if } v = (i, R, \epsilon), \\ \phi_2(U, v) &= 0. \end{aligned} \tag{4.3}$$

The events are characterized by the event rate $\lambda : \Omega \times \mathcal{V} \to \mathbb{R}_+$ and a Markov kernel Q defined on $\Omega \times \mathcal{V} \times \mathcal{B}(\Omega \times \mathcal{V})$ which sums up the different lifting rules. We voluntarily omit the boundary kernel Q^b describing the refresh process for the sake of clarity as it does not intervene in the invariance computation [22].

The general form of the generator $\mathcal{A}$ for an Event-Chain is [29]

$$\mathcal{A}f(U,v) = \langle \phi, \nabla_{U,v} f(U,v) \rangle + \lambda(U,v) \int_{\Omega\times\mathcal{V}} (f(U',v') - f(U,v)) Q[(U,v),(\mathrm{d}U',\mathrm{d}v')], \quad (4.4)$$

with the first term coding for the deterministic drift and the second term for changes due to the stochastic events determined by the Markov kernel Q.

In our case, for a lifted state (U,v), the event rate is the sum of event rates λ^j due to all the plaquettes P^j attached to link i. We can write, with $v = (i,R,\epsilon)$,

$$\lambda(U,v) = \sum_{j=1}^{2(d-1)} \lambda^j(U,v) = \sum_{j=1}^{2(d-1)} [\partial_\theta S_v^j(0)]^+ = \sum_{j=1}^{2(d-1)} [\langle \phi, \nabla_{U_i} S_v^j \rangle]^+. \quad (4.5)$$

with $\langle A, B\rangle = \mathrm{Re}\,\mathrm{Tr}\left[A^\dagger B\right]$ the usual scalar product on $\mathrm{SU}(N)$ and $\nabla_{U_i} S_v^j$ denotes the gradient of the action with respect to the matrix elements of U_i. Formally, it corresponds to the Lie derivative of the action along the algorithmic time θ. Note that the other links of configuration U have no impact in the scalar product because of (4.3).

The Markov kernel Q can also be decomposed in a weighted sum of Markov Q^j coding for the lifts in plaquette P^j. We have

$$Q((U,v),(\mathrm{d}U',\mathrm{d}v')) = \sum_{j=0}^{2(d-1)} \underbrace{\frac{\lambda^j(U,v)}{\lambda(U,v)}}_{\substack{\text{Probability for } P^j \\ \text{to trigger an event}}} Q^j[(U,v),(\mathrm{d}U',\mathrm{d}v')] \quad (4.6)$$

We can then write (4.4) as

$$\mathcal{A}f(U,v) = \langle \phi, \nabla_{U,v} f(U,v) \rangle + \int_{\Omega\times\mathcal{V}} (f(U',v') - f(U,v)) \sum_{j=1}^{2(d-1)} \lambda^j(U,v) Q^j[(U,v),(\mathrm{d}U',\mathrm{d}v')], \quad (4.7)$$

The definition of Q^j depends on the lifting scheme used. In case of reflective ECMC, we have

$$Q^j_{\text{reflective}}[(U,v),(\mathrm{d}U',\mathrm{d}v')] = \sum_{k\in P^j} \frac{|\partial_\theta S^j_{k,R}(0)|}{\sum_{l\in P^j} |\partial_\theta S^j_{l,R}(0)|} \delta_U(\mathrm{d}U') \delta_{(k,R,-\mathrm{sign}(\partial_\theta S^j_{k,R}(0)))}(\mathrm{d}v'), \quad (4.8)$$

and in the forward ECMC case we have

$$Q^j_{\text{forward}}[(U,v),(\mathrm{d}U',\mathrm{d}v')] = \sum_{k\in P^j\setminus\{i\}} \frac{1}{3} \delta_U(\mathrm{d}U') \delta_{(k,R_k,\epsilon_k)}(\mathrm{d}v'), \quad (4.9)$$

with R_k, ϵ_k defined in table 1.

4.2. Invariance of the target probability distribution

The invariance of the target probability distribution is expressed as continuous-time variant of the global balance equation (3.12) written as

$$\int_{\Omega\times\mathcal{V}} \mathcal{A}f(X)\pi(X)\mathrm{d}X = 0. \tag{4.10}$$

Following the derivations of [29], a sufficient condition for algorithms defined with factorized event rates (cf. (4.5)) is written for each factor j and any lifted state (U, v)

$$[\partial_\theta S_v^j(0)]^- \mu(v) = \int_{\mathcal{V}} \mathrm{d}v' \mu(v')[\partial_\theta S_{v'}^j(0)]^+ Q^j[(U, v'), (U, v)], \tag{4.11}$$

with $[x]^- = \max(0, -x)$ the negative part of x, such that $x = [x]^+ - [x]^-$.

Condition (4.11) dictates that any local dissipation caused by the deterministic drift must be perfectly counterbalanced by redistribution performed by the Markov kernels during lift events. Heuristically, the stochastic events act as probability regulators that perpetually replenish the target measure wherever the continuous deterministic flow tends to drain it.

Given the expressions (4.8) and (4.9) of the Markov kernel used in both implementations, showing that this condition holds is straightforward. For the reflective ECMC kernel, by definition, $Q^j_{\text{reflective}}[(U, v'), (U, v)] \neq 0$ only if v is such that updating from the lifted state (U, v) decreases the action. Thus we have for the l.h.s. of (4.11) $[\partial_\theta S_v^j(0)]^- = |\partial_\theta S_v^j(0)|$.

Furthermore, denoting $v = (i, R, \epsilon)$, we have

$$Q^j_{\text{reflective}}[(U, v'), (U, v)] \neq 0 \Leftrightarrow v' \in P^j \times \{R\} \times \{-\text{sign}(\partial_\theta S^j_{i,R}(0))\} \coloneqq \mathcal{P}_v. \tag{4.12}$$

This yields

$$\begin{aligned} &\int_{\mathcal{V}} \mathrm{d}v' \mu(v')[\partial_\theta S_{v'}^j(0)]^+ Q^j_{\text{reflective}}[(U, v'), (U, v)] \\ &\quad = \sum_{v'\in\mathcal{P}_v} \mu(v')[\partial_\theta S_{v'}^j(0)]^+ \frac{|\partial_\theta S_v^j(0)|}{\sum_{l\in P^j} |\partial_\theta S^j_{l,R}(0)|} \end{aligned} \tag{4.13}$$

We then have $\mu(v') = \mu(v)$, as the probability distribution is uniform on $\mathcal{V}$, and $\sum_{l\in P^j} |\partial_\theta S^j_{l,R}(0)| = \sum_{v'\in\mathcal{P}_v} [\partial_\theta S_{v'}^j(0)]^+$, ensuring the realization of condition (4.11).

For the forward lift rule, the kernel is non zero if $v' \in \{(k, R_k, \epsilon_k), k \in P^j \setminus \{i\}\} \coloneqq \mathcal{P}_v$, which contains 3 elements (the 3 links of the corresponding staple). The selection rules of table 1 ensure $[\partial_\theta S_{v'}^j(0)]^+ = [\partial_\theta S_v^j(0)]^-$. The r.h.s. of condition (4.11) becomes

$$\begin{aligned} &\int_{\mathcal{V}} \mathrm{d}v' \mu(v')[\partial_\theta S_{v'}^j(0)]^+ Q^j_{\text{forward}}[(U, v'), (U, v)] \\ &\quad = \sum_{v'\in\mathcal{P}_v} \mu(v')[\partial_\theta S_{v'}^j(0)]^+ \left(\sum_{k\in P^j\setminus\{i\}} \frac{1}{3}\delta_{(k,R_k,\epsilon_k)}(v) \right) \\ &\quad = \mu(v) \sum_{v'\in\mathcal{P}_v} \frac{1}{3}[\partial_\theta S_v^j(0)]^- \\ &\quad = \mu(v)[\partial_\theta S_v^j(0)]^-. \end{aligned} \tag{4.14}$$

β	1	2	3	4	5	6	7	8	9
θ_{sample}	32 768	16 384	10 649	7700	6062	5242	4751	4423	4096

Table 2: Values of θ_{sample} required to define a sweep depending on β for a 8^4 lattice.

We have thus proven that the generators of reflective and forward ECMC algorithms leave the target probability distribution $\pi \otimes \mu$ invariant. Thanks to refreshment events, ergodicity is ensured [22] and therefore this shows the correct convergence of the algorithms in the lifted state space.

5. Implementation for SU(3)

We can now generate configurations in pure gauge lattice QCD with Wilson action using both ECMC implementations and compare the measured value of observables to other sampling algorithms. In this proof-of-concept work, we use small lattice volumes of points to show correct thermalization and convergence towards the equilibrium distribution without the need for parallelization. Parallelization of ECMC for lattice QCD is not straightforward because of the stochastic nature of the lift process and it will be addressed in a forthcoming publication.

The plaquette measurements have been obtained using both implementations of ECMC and quasi-Heatbath on 8^4 lattices with periodic boundary conditions and different values of β. We define the mean plaquette of a gauge configuration of volume V as

$$P = \frac{1}{18V} \sum_{x \in \Lambda} \text{Re Tr} \left[U_{x,\mu\nu} \right] \tag{5.1}$$

with Λ the set of lattice sites. Measurements of the mean plaquette were performed every sweep. In order to compare ECMC to sequential algorithms such as Heatbath or Metropolis, we need to choose θ_{sample} such that the number of events between two samples is equal to the number of links. This value depends on the size of the lattice and on the inverse bare coupling β. The values chosen are detailed in Table 2. High values of θ_{sample} for low β are explained by the form of the factorized Metropolis filter (3.4) and the computation of θ_{event} (3.13) which allows for larger update angles when the bare coupling is low.

The analysis of the measures is performed with the Γ-method [30]. For a thermalized run yielding a finite number of values $(P_1, P_2, ..., P_{N_s})$ of the mean plaquette, we define $C(\tau)$ the autocorrelation at lag τ as

$$C(\tau) = \frac{1}{N-\tau} \sum_{i=1}^{N-\tau} (P_i - \bar{P})(P_{i+\tau} - \bar{P}) \tag{5.2}$$

where $\bar{P} = \frac{1}{N} \sum_{i=1}^{N} P_i$ is the sample mean. The normalized autocorrelation function is then given by $\rho(\tau) = C(\tau)/C(0)$. The crucial quantity for the error estimation is the integrated autocorrelation time, defined as

$$\tau_{\text{int}} = \frac{1}{2} + \sum_{\tau=1}^{\infty} \rho(\tau) \tag{5.3}$$

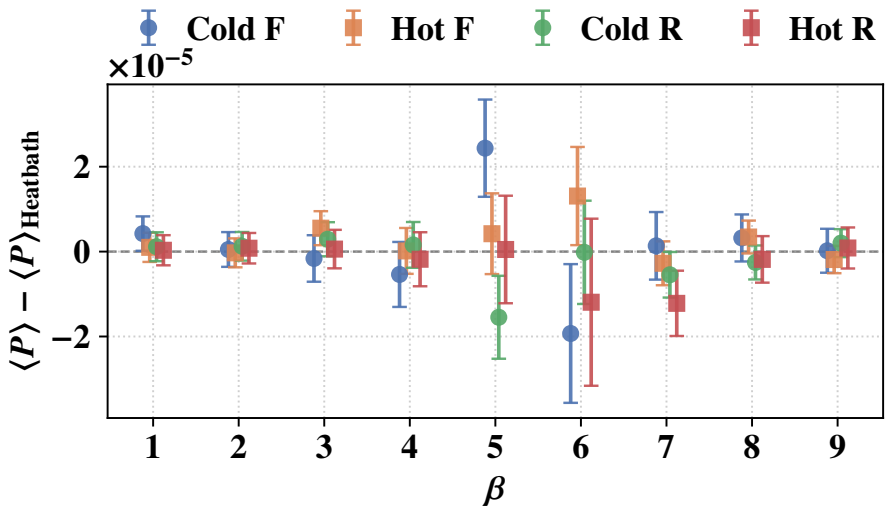


Figure 4: Comparison of residuals of the average plaquette with respect to Heatbath for reflective (R) and forward (F) ECMC implementations, with hot and cold starts.

In practice, the sum is truncated at a self-consistent window W to avoid the accumulation of noise from the tail of $\rho(\tau)$ where the signal-to-noise ratio is low. The variance of the sample mean is then correctly estimated by

$$\sigma^2_{\bar{P}} = \frac{C(0)}{N} \times 2\tau_{\text{int}} \tag{5.4}$$

This method accounts for the correlations between successive configurations in the Markov Chain, which otherwise would lead to a significant underestimation of the statistical uncertainties.

In Section 5.1, we report the average plaquette values obtained across a range of β for both hot and cold starts of the ECMC implementations. These results are compared against Heatbath benchmarks to show algorithmic convergence. Section 5.2 presents the comparison of autocorrelation times both in sweeps and CPU time between both implementations of ECMC and Heatbath.

5.1. Convergence

To assess convergence towards the target distribution π for both ECMC implementations, we generate chains starting with a cold (every link to identity) and a hot (all links chosen at random in SU(3)) configuration. We also generate measurements using Heatbath. Comparison of the values of the average plaquette in all cases with Heatbath are shown in Figure 4.

We observe a very good agreement up to 1×10^{-5} with no visible bias, confirming a correct convergence. Tables that collect the values obtained and statistical consistency with respect to Heatbath are available in Appendix A.

5.2. Comparison of autocorrelation times

Although the mean plaquette is a short-distance observable which has been shown to decouple from the slow modes of the theory [5] and not subject to critical slowing down, the study of its autocorrelation can give us a first idea of the behavior of ECMC around the deconfinement phase transition with respect to other algorithms.

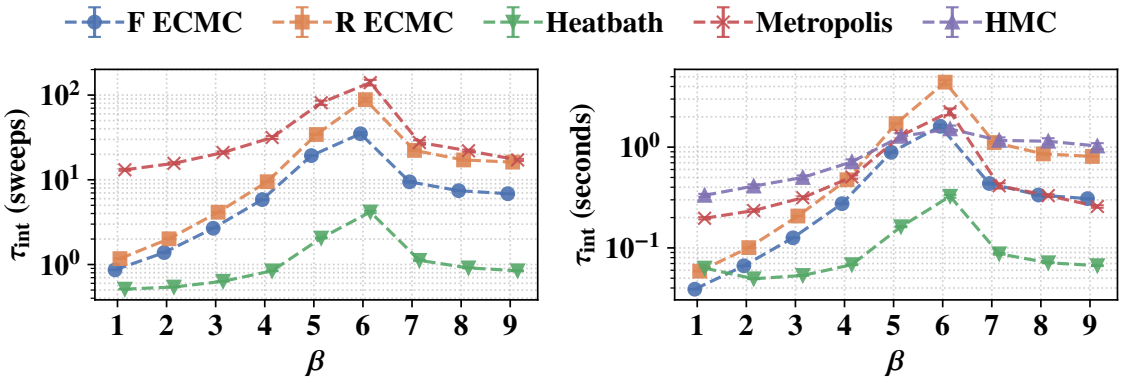


Figure 5: Comparison of autocorrelation times in sweeps (left) and in CPU time (right) as a function of β between reflective (R) and forward (F) ECMC implementations, Heatbath, Metropolis and HMC.

Figure 5 illustrates the evolution of the integrated autocorrelation times as functions of β. It reveals a consistent behavior across all algorithms, with a pronounced peak near $\beta = 6$, that is close to the deconfinement phase transition point $\beta_c(N_t = 8) \sim 6.06$ [31]. Both ECMC variants achieve superior sampling efficiency compared to the Metropolis algorithm in number of sweeps, though they do not reach the performance level of the Heatbath algorithm. When compared to Metropolis and HMC[1] algorithms, forward ECMC achieves significantly lower CPU autocorrelation times away from the phase transition, while all three algorithms reach comparable efficiency in its vicinity. A direct comparison between the two ECMC implementations further reveals that the forward lifting rule yields more efficient decorrelation than its reflective counterpart, with a twofold speedup in the vicinity of the deconfinement transition. This improvement is consistent with the suppression of backtracking inherent to the forward dynamics, which enables a more thorough exploration of the configuration space.

The performance gap with respect to Heatbath is not surprising, as the latter constitutes a highly optimized baseline: by sampling new link variables directly from the local conditional distribution, it achieves near-complete renewal of the local degrees of freedom at each update step. As such, Heatbath provides a natural lower bound for the autocorrelation time of local observables, against which the rejection-free yet potentially more ballistic dynamics of ECMC must be assessed.

It should nonetheless be emphasized that the primary advantage of ECMC is expected to manifest in the sampling of long-range observables, such as the topological charge, which are notoriously affected by critical slowing down (CSD) in conventional algorithms [5, 6]. A thorough investigation of these slow modes requires substantially larger statistics and lattice volumes, and consequently demands a fully parallelized implementation of the algorithm. Both the parallelization strategy and the subsequent analysis of non-local observables lie beyond the scope of this proof-of-concept study and will be addressed in a forthcoming publication.

Finally, it is worth noting that ECMC and Heatbath exhibit comparable computational costs per sweep, as both algorithms involve numerically intensive operations – including evaluations

[1] We have set a trajectory lengh of molecular dynamics $\tau = 3$ in our numerical test.

of trigonometric and exponential functions in addition to staple computations – in contrast to the Metropolis algorithm, which relies solely on elementary arithmetic and staple calculations. As a result, reflective ECMC is less efficient than Metropolis in CPU time, while the forward variant stays better. Also, the efficiency advantage of the forward ECMC implementation over the reflective one is preserved in terms of CPU time, while the gap with Heatbath remains unchanged. As the present results were obtained using a basic implementation of ECMC, a thorough optimization of the ECMC sweep is expected to further improve its computational efficiency relative to classical algorithms in terms of CPU time.

6. Conclusion

In this paper we have presented an alternative class of algorithms to simulate pure lattice gauge theories. So-called Event-Chain Monte-Carlo, they relax the detailed balance condition of the standard Markov-Chain Monte-Carlo algorithms like Metropolis or Heatbath. They respect the global balance and are rejection-free. The conceptual price to pay is an extension of the configuration space to a space with lift variables. They mix ballistic motion in configuration space with a stochastic evolution in the lift variables space. We were able to design two variants that are characterized by different transition rates in the lift variable space. They satisfy the invariance condition of PDMP and, thanks to symmetries of the Wilson gauge action, one of them is without backtracking. We have shown that distribution simulated by ECMC converge to the equilibrium one, whatever the bare coupling β is. A numerical comparison of the plaquette autocorrelation time with the one obtained with Metropolis, Heatbath and Hybrid Monte-Carlo indicates that the forward variant of ECMC has a better efficiency than Metropolis and HMC in terms of CPU. In a companion paper we will address parallelization of ECMC and its ability to mitigate critical slowing down of the topological charge.

Acknowledgments

This work was performed using computational resources from the “Mésocentre” computing center of Université Paris-Saclay, CentraleSupélec and École Normale Supérieure Paris-Saclay[2] supported by CNRS and Région Île-de-France. M.M. acknowledges the support of the French ANR under the grant ANR-20-CE46-0007 (*SuSa* project).

[2] https://mesocentre.universite-paris-saclay.fr

A. Details of the runs

The numerical results for the mean plaquette $\langle P \rangle$ and the corresponding integrated autocorrelation times τ_{int} are summarized in the following tables for different values of the coupling constant β.

For each algorithm, we report the estimated mean value where the statistical uncertainty is calculated using the Γ-method [30]. The efficiency of the sampling is quantified by τ_{int}, expressed in units of sweeps. To verify the convergence and the statistical consistency of the various ECMC implementations (starting from both hot and cold configurations) against the reference Heatbath algorithm, we provide the Z_H-score. It is defined as:

$$Z_H = \frac{|\langle P \rangle_{\text{algo}} - \langle P \rangle_{\text{Heathbath}}|}{\sqrt{\sigma^2_{\text{algo}} + \sigma^2_{\text{Heathbath}}}} \tag{A.1}$$

where σ denotes the statistical error of the respective algorithm. A Z_H-score significantly lower than 3 indicates that the results are statistically compatible, confirming that all implementations sample the correct distribution.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.0601333(24)	0.8621(67)	1.09	768 497
Forward ECMC	Hot	0.0601300(11)	0.8667(33)	0.33	3 548 984
Refl. ECMC	Cold	0.0601301(10)	1.1713(42)	0.37	5 731 472
Refl. ECMC	Hot	0.0601293(14)	1.1787(57)	0.12	2 967 986
Heatbath	Cold	0.0601288(32)	0.5125(40)	-	256 500

Table 3: Comparison for $\beta = 1$.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.1288111(27)	1.385(12)	0.15	1 081 496
Forward ECMC	Hot	0.1288103(16)	1.3819(69)	0.04	3 370 985
Refl. ECMC	Cold	0.1288118(15)	2.0162(96)	0.40	5 583 473
Refl. ECMC	Hot	0.1288114(19)	2.009(12)	0.27	3 218 986
Heatbath	Cold	0.1288105(30)	0.5428(41)	-	346 500

Table 4: Comparison for $\beta = 2$.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.2050398(44)	2.663(33)	0.26	909 997
Forward ECMC	Hot	0.2050468(23)	2.694(18)	1.41	3 406 484
Refl. ECMC	Cold	0.2050442(23)	4.156(28)	0.75	5 199 974
Refl. ECMC	Hot	0.2050419(31)	4.129(37)	0.16	2 785 487
Heatbath	Cold	0.2050412(33)	0.6342(54)	-	378 500

Table 5: Comparison for $\beta = 3$.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.2904918(65)	5.915(99)	0.69	1 050 496
Forward ECMC	Hot	0.2904971(36)	5.778(56)	0.01	3 357 485
Refl. ECMC	Cold	0.2904986(36)	9.536(92)	0.29	5 590 473
Refl. ECMC	Hot	0.2904953(49)	9.45(12)	0.29	2 931 487
Heatbath	Cold	0.2904971(40)	0.8422(88)	-	391 000

Table 6: Comparison for $\beta = 4$.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.400444(10)	19.58(41)	2.09	2 140 990
Forward ECMC	Hot	0.4004246(80)	19.04(32)	0.43	3 412 484
Refl. ECMC	Cold	0.4004042(83)	33.63(57)	1.68	5 648 472
Refl. ECMC	Hot	0.400421(12)	34.67(81)	0.04	2 951 486
Heatbath	Cold	0.4004205(51)	2.054(23)	-	906 997

Table 7: Comparison for $\beta = 5$.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.594201(15)	34.8(24)	1.16	1 376 495
Forward ECMC	Hot	0.5942349(94)	34.0(16)	1.32	3 381 485
Refl. ECMC	Cold	0.594220(10)	68.6(34)	0.00	5 692 972
Refl. ECMC	Hot	0.594214(17)	94.4(73)	0.30	2 959 486
Heatbath	Cold	0.5942199(64)	4.17(14)	-	909 496

Table 8: Comparison for $\beta = 6$.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.6716472(71)	9.50(35)	0.10	908 496
Forward ECMC	Hot	0.6716447(35)	8.88(18)	0.35	3 403 984
Refl. ECMC	Cold	0.6716410(39)	18.11(40)	1.02	5 642 972
Refl. ECMC	Hot	0.6716382(54)	18.09(53)	1.26	2 945 487
Heatbath	Cold	0.6716465(37)	1.122(23)	-	399 998

Table 9: Comparison for $\beta = 7$.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.7206891(48)	7.34(13)	0.52	1 062 496
Forward ECMC	Hot	0.7206901(27)	7.335(78)	1.03	3 441 484
Refl. ECMC	Cold	0.7206834(29)	14.20(16)	0.69	5 625 972
Refl. ECMC	Hot	0.7206875(39)	14.32(22)	0.29	3 185 985
Heatbath	Cold	0.7206862(28)	0.9140(98)	-	399 998

Table 10: Comparison for $\beta = 8$.

Algorithm	Start	Plaquette $\langle P \rangle$	τ_{int}	Z_H	Samples
Forward ECMC	Cold	0.7561657(46)	6.71(14)	0.04	776 498
Forward ECMC	Hot	0.7561646(22)	6.731(69)	0.40	3 411 484
Refl. ECMC	Cold	0.7561677(24)	12.71(14)	0.54	5 679 972
Refl. ECMC	Hot	0.7561697(33)	12.54(18)	0.95	2 956 486
Heatbath	Cold	0.7561659(23)	0.8443(87)	-	399 998

Table 11: Comparison for $\beta = 9$.

References

[1] K.G. Wilson, *Confinement of quarks*, .

[2] A.D. Kennedy and B.J. Pendleton, *Improved heatbath method for Monte Carlo calculations in lattice gauge theories*, .

[3] M. Creutz, *Overrelaxation and Monte Carlo simulation*, .

[4] S. Duane, A.D. Kennedy, B.J. Pendleton and D. Roweth, *Hybrid Monte Carlo*, .

[5] S. Schaefer, R. Sommer and F. Virotta, *Critical slowing down and error analysis in lattice QCD simulations*, 1009.5228.

[6] B. Allés, G. Boyd, M. D'Elia, A.D. Giacomo and E. Vicari, *Hybrid Monte Carlo and topological modes of full QCD*, hep-lat/9607049.

[7] M. Lüscher and S. Schaefer, *Lattice QCD with open boundary conditions and twisted-mass reweighting*, 1206.2809.

[8] B. Joo, A.C. Irving, J.C. Sexton, B. Pendleton, S.M. Pickles, Z. Sroczynski et al., *Parallel Tempering in Lattice QCD with O(a)-Improved Wilson Fermions*, hep-lat/9810032.

[9] T. Eichhorn, G. Fuwa, C. Hoelbling and L. Varnhorst, *Parallel Tempered Metadynamics*, in *Proceedings of The 41st International Symposium on Lattice Field Theory — PoS(LATTICE2024)*, p. 061, DOI [2503.09747].

[10] G. Cossu, B. Lucini, R. Pellegrini and A. Rago, *Ergodicity of the LLR method for the Density of States*, 1710.06250.

[11] A. Francis, P. Fritzsch, M. Lüscher and A. Rago, *Master-field simulations of O(a)-improved lattice QCD: Algorithms, stability and exactness*, 1911.04533.

[12] R. Abbott, M.S. Albergo, A. Botev, D. Boyda, K. Cranmer, D.C. Hackett et al., "Normalizing flows for lattice gauge theory in arbitrary space-time dimension." 10.48550/arXiv.2305.02402.

[13] P. Diaconis, S. Holmes and R.M. Neal, *Analysis of a nonreversible Markov chain sampler*, .

[14] M. Michel, S.C. Kapfer and W. Krauth, *Generalized event-chain Monte Carlo: Constructing rejection-free global-balance algorithms from infinitesimal steps*, 1309.7748.

[15] E.P. Bernard, W. Krauth and D.B. Wilson, *Event-chain Monte Carlo algorithms for hard-sphere systems*, 0903.2954.

[16] M. Michel, J. Mayer and W. Krauth, *Event-chain Monte Carlo for classical continuous spin models*, 1508.06541.

[17] O. Bouverot-Dupuis, A. Rosso and M. Michel, *Bosonized one-dimensional quantum systems through enhanced event-chain Monte Carlo*, 2503.11577.

[18] M. Hasenbusch and S. Schaefer, *Testing the event-chain algorithm in asymptotically free models*, 1806.11460.

[19] G. Mana, A. Pelissetto and A.D. Sokal, *Multi-Grid Monte Carlo via XY Embedding. II. Two-Dimensional $SU(3)$ Principal Chiral Model*, hep-lat/9610021.

[20] M. Michel, A. Durmus and S. Sénécal, *Forward Event-Chain Monte Carlo: Fast Sampling by Randomness Control in Irreversible Markov Chains*, .

[21] M. Creutz, *Monte Carlo study of quantized SU(2) gauge theory*, .

[22] A. Monemvassitis, A. Guillin and M. Michel, *PDMP characterisation of event-chain Monte Carlo algorithms for particle systems*, 2208.11070.

[23] N. Metropolis and S. Ulam, *The Monte Carlo method*, 18139350.

[24] W. Krauth, *Event-chain Monte Carlo: Foundations, applications, and prospects*, 2102.07217.

[25] E. Peters, *Event-Chain Monte Carlo: The Global-Balance Breakthrough* 10.48550/arXiv.2602.07199.

[26] J. Harland, M. Michel, T.A. Kampmann and J. Kierfeld, *Event-chain Monte Carlo algorithms for three- and many-particle interactions*, 1611.09098.

[27] M. Luscher and P. Weisz, *On-shell improved lattice gauge theories*, .

[28] Y. Iwasaki, "Renormalization Group Analysis of Lattice Theories and Improved Lattice Action. II – four-dimensional non-abelian SU(N) gauge model." 10.48550/arXiv.1111.7054.

[29] T. Guyon, A. Guillin and M. Michel, "Necessary and sufficient symmetries in Event-Chain Monte Carlo with generalized flows and Application to hard dimers." 10.48550/arXiv.2307.02341.

[30] U. Wolff, *Monte Carlo errors with less errors*, hep-lat/0306017.

[31] G. Boyd, J. Engels, F. Karsch, E. Laermann, C. Legeland, M. Lutgemeier et al., *Thermodynamics of SU(3) lattice gauge theory*, .