Particle-MALA and Particle-mGRAD: Gradient-based MCMC methods for high-dimensional state-space models

Adrien Corenflos[1] and Axel Finke[2]

[1]Department of Electrical Engineering and Automation, Aalto University, Finland
adrien.corenflos@aalto.fi
[2]Department of Mathematical Sciences, Loughborough University, UK
a.finke@lboro.ac.uk

January 26, 2024

State-of-the-art methods for Bayesian inference in state-space models are (a) *conditional sequential Monte Carlo (CSMC)* algorithms; (b) sophisticated 'classical' MCMC algorithms like *MALA*, or *mGRAD* from Titsias and Papaspiliopoulos (2018). The former propose N particles at each time step to exploit the model's 'decorrelation-over-time' property and thus scale favourably with the time horizon, T, but break down if the dimension of the latent states, D, is large. The latter leverage gradient-/prior-informed local proposals to scale favourably with D but exhibit sub-optimal scalability with T due to a lack of model-structure exploitation. We introduce methods which combine the strengths of both approaches. The first, *Particle-MALA*, spreads N particles locally around the current state using gradient information, thus extending MALA to $T > 1$ time steps and $N > 1$ proposals. The second, *Particle-mGRAD*, additionally incorporates (conditionally) Gaussian prior dynamics into the proposal, thus extending the mGRAD algorithm to $T > 1$ time steps and $N > 1$ proposals. We prove that Particle-mGRAD interpolates between CSMC and Particle-MALA, resolving the 'tuning problem' of choosing between CSMC (superior for highly informative prior dynamics) and Particle-MALA (superior for weakly informative prior dynamics). We similarly extend other 'classical' MCMC approaches like *auxiliary MALA*, *aGRAD*, and *preconditioned Crank–Nicolson–Langevin (PCNL)* to $T > 1$ time steps and $N > 1$ proposals. In experiments, for both highly and weakly informative prior dynamics, our methods substantially improve upon both CSMC and sophisticated 'classical' MCMC approaches.

1 Introduction

1.1 Feynman–Kac models

The aim of this work is to construct efficient *Markov chain Monte Carlo (MCMC)* updates for sampling from a continuous *joint smoothing* distribution $\pi_T(\mathbf{x}_{1:T})$ on $\mathcal{X}^T$, where $\mathcal{X} \coloneqq \mathbb{R}^D$ and where for any $t \leq T$, we have defined the following distributions (termed *filters*):

$$\pi_t(\mathbf{x}_{1:t}) \propto \prod_{s=1}^{t} Q_s(\mathbf{x}_{s-1:s}). \tag{1}$$

Here, $Q_t(\mathbf{x}_{t-1:t}) > 0$ is differentiable and can be evaluated point-wise. Throughout this work, we use the convention that quantities with 'time' subscripts $t \leq 0$ or $t > T$ should be ignored, so that, e.g., $Q_1(x_{0:1}) \equiv Q_1(\mathbf{x}_1)$ and $Q_{T+1}(\mathbf{x}_{T:T+1}) \equiv 1$. We will frequently work with some decomposition

$$Q_t(\mathbf{x}_{t-1:t}) = M_t(\mathbf{x}_t|\mathbf{x}_{t-1})G_t(\mathbf{x}_{t-1:t}),$$

such that

- $M_t(\,\cdot\,|\mathbf{x}_{t-1})$ is a density (w.r.t. a suitable version of the Lebesgue measure) and also defines a Markov transition kernel called *mutation kernel*;
- $G_t(\mathbf{x}_{t-1:t}) > 0$ is called *potential function.*

We assume that these densities and potential functions are differentiable and that they (as well as their gradients) can be evaluated point-wise. Motivated by the following example, we will sometimes refer to $M_{1:T}(\mathbf{x}_{1:T}) \coloneqq \prod_{t=1}^T M_t(\mathbf{x}_t|\mathbf{x}_{t-1})$ as the *prior dynamics* of the *latent states* $\mathbf{x}_{1:T}$ and $G_{1:T}(\mathbf{x}_{1:T}) \coloneqq \prod_{t=1}^T G_t(\mathbf{x}_{t-1:t})$ as the *likelihood.*

Example 1 (state-space model). *One important special case of Feynman–Kac models are* state-space models. *A state-space model is a bivariate Markov chain $(\mathbf{x}_t, \mathbf{y}_t)_{t\geq 1}$ on $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \coloneqq \mathbb{R}^D$ and $\mathcal{Y} \coloneqq \mathbb{R}^{D'}$, with initial density $p(\mathbf{x}_1, \mathbf{y}_1) = f_1(\mathbf{x}_1)g_1(\mathbf{y}_1|\mathbf{x}_1)$ and transition densities $p(\mathbf{x}_t, \mathbf{y}_t|\mathbf{x}_{t-1}) = f_t(\mathbf{x}_t|\mathbf{x}_{t-1})g_t(\mathbf{y}_t|\mathbf{x}_t)$ (w.r.t. a suitable version of the Lebesgue measure). State-space models assume that only a measurements $\mathbf{y}_{1:T}$ can be observed whilst the Markov chain $(\mathbf{x}_t)_{t\geq 1}$ (often representing the evolution of the phenomenon of interest) is latent. The joint smoothing distribution then encodes our knowledge of the latent states $\mathbf{x}_{1:T}$ given the available data $\mathbf{y}_{1:T}$:*

$$\pi_T(\mathbf{x}_{1:T}) \coloneqq p(\mathbf{x}_{1:T}|\mathbf{y}_{1:T}) \propto \prod_{t=1}^{T} f_t(\mathbf{x}_t|\mathbf{x}_{t-1})g_t(\mathbf{y}_t|\mathbf{x}_t).$$

One possible way of casting such a state-space model as a Feynman–Kac model (there are others) is then to take $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = f_t(\mathbf{x}_t|\mathbf{x}_{t-1})$ and $G_t(\mathbf{x}_{t-1:t}) = g_t(\mathbf{y}_t|\mathbf{x}_t)$. In this case, $M_{1:T}(\mathbf{x}_{1:T}) = p(\mathbf{x}_{1:T})$, $G_{1:T}(\mathbf{x}_{1:T}) = p(\mathbf{y}_{1:T}|\mathbf{x}_{1:T})$, and $\pi_t(\mathbf{x}_{1:t}) = p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$, for $t \leq T$.

1.2 Sampling the latent states

Performing inference about the latent states $\mathbf{x}_{1:T}$ requires calculating expectations of the form $\mathbb{E}_{\mathbf{x}_{1:T}\sim\pi_T}[\varphi(\mathbf{x}_{1:T})]$, for some integrable test function $\varphi\colon \mathcal{X}^T \to \mathbb{R}$. Unfortunately, such expectations do not admit closed-form expressions in most realistic problems and must be approximated by some Monte Carlo estimate $\frac{1}{I}\sum_{i=1}^I \varphi(\mathbf{x}_{1:T}[i])$ using samples $(\mathbf{x}_{1:T}[i])_{i=1}^I$ (approximately) distributed according to π_T. These often come from some MCMC algorithm targeting π_T.

'Classical' MCMC methods. Unfortunately, simple MCMC approaches like the *independent Metropolis–Hastings (IMH)* algorithm (Hastings, 1970) perform poorly if the problem size: $D \times T$, is large due the difficulty of constructing efficient *global* (a.k.a. *independent*) proposal distributions in high dimensions. To circumvent this difficulty, MCMC algorithms with *local* moves like the *random-walk Metropolis (RWM)* algorithm (Metropolis et al., 1953), propose a new state of the Markov chain near the current state. By decreasing the proposal scale at a suitable rate with the problem size, the RWM algorithm can circumvent this curse of dimension (Roberts et al., 1997). Further improved performance can be achieved by exploiting

- *gradient information,* i.e. by including gradients of the log-likelihood or log-target density into the proposal as in the *Metropolis-adjusted Langevin algorithm (MALA)* from Besag (1994) and in the *auxiliary MALA (aMALA)* from Titsias and Papaspiliopoulos (2018); and *additionally*

- *prior information,* i.e. by explicitly incorporating the prior dependence structure into the proposal as in the *preconditioned Crank–Nicolson–Langevin (PCNL)* and related algorithms (see, e.g., Cotter et al., 2013, and references therein) or in the *marginal gradient (mGRAD)* and *auxiliary gradient (aGRAD)*[1] algorithms from Titsias (2011); Titsias and Papaspiliopoulos (2018).

Figure 1a illustrates that 'classical' MCMC algorithms can scale favourably with D.

However, 'classical' MCMC algorithms are agnostic to the 'decorrelation-over-time' structure of the target distribution $\pi_T(\mathbf{x}_{1:T})$, i.e., to the fact that, for suitably regular models, the correlation of x_s and $\mathbf{x}_t$ under $\pi(\mathbf{x}_{1:T})$ decays with $|t-s|$. For example, for the simple RWM algorithm and MALA, the *step size* $\delta > 0$ (i.e., proposal variance) would need to decrease at a suitable rate with T ($\delta \in \mathrm{O}((DT)^{-1})$ and $\delta \in \mathrm{O}((DT)^{-1/3})$, respectively) even if the model was completely independent across time steps (Roberts and Rosenthal, 2001). Therefore, it stands to reason that the scaling of 'classical' MCMC methods like MALA, PCNL or mGRAD/aGRAD with the time horizon T could be improved by empowering them to exploit this model structure.

CSMC methods. Another popular π_T-invariant MCMC-kernel, P_{CSMC}, is induced by running the *CSMC* algorithm proposed in the seminal work Andrieu et al. (2010); Whiteley (2010). Given the current state $\mathbf{x}_{1:T} \in \mathcal{X}^T$ of the Markov chain (then called the *reference path*) this algorithm generates $\tilde{\mathbf{x}}_{1:T} \sim P_{\mathrm{CSMC}}(\,\cdot\,|\mathbf{x}_{1:T})$ as follows, where we write $[n] \coloneqq \{1, \dots, n\}$ and $[n]_0 \coloneqq [n] \cup \{0\}$:

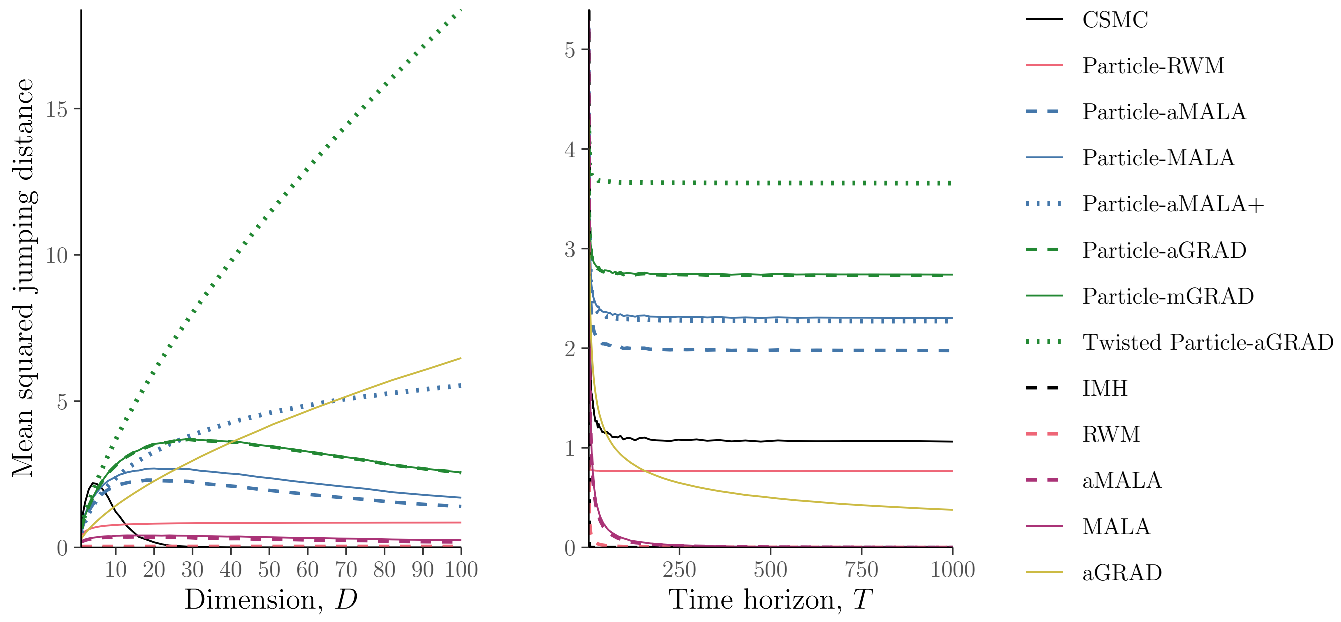
1. For $t = 1, \dots, T$, sample some index k_t from a uniform distribution on $[N]_0$; set $\mathbf{x}_t^{k_t} \coloneqq \mathbf{x}_t$ and sample the remaining *particles* $\mathbf{x}_t^{-k_t} \coloneqq (\mathbf{x}_t^0, \dots, \mathbf{x}_t^{k_t-1}, \mathbf{x}_t^{k_t+1}, \dots, \mathbf{x}_t^N)$ conditionally independently such that for $n \neq k_t$,

$$\mathbf{x}_t^n \sim M_t(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n}), \tag{2}$$

   for *ancestor indices* $a_{t-1}^n \in [N]_0$ whose rôle is explained later.

2. Return $\tilde{\mathbf{x}}_{1:T} \coloneqq (\mathbf{x}_1^{l_1}, \dots, \mathbf{x}_t^{l_T})$, for indices $l_1, \dots, l_T \in [N]_0$ sampled from an appropriate distribution.

[1]Throughout this work, 'aGRAD' refers more specifically to the 'aGrad-z' algorithm from Titsias and Papaspiliopoulos (2018).



(a) Empirical scaling with D for fixed time horizon $T = 25$.

(b) Empirical scaling with T for fixed state dimension $D = 10$.

Figure 1: Toy linear-Gaussian state-space model with $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{x}_{t-1}, \lambda\mathbf{I})$ and $G_t(\mathbf{x}_{t-1:t}) = \mathrm{N}(\mathbf{y}_t; \mathbf{x}_t, \mathbf{I})$, where $\mathbf{I}$ is the $(D \times D)$-identity matrix and $\lambda = 1$. For a fair comparison, all methods use $N + 1 = 32$ particles. The step sizes are: $(TD)^{-1}$ for (multi-proposal) RWM, D^{-1} for Particle-RWM, $(TD)^{-1/3}$ for (multi-proposal) aMALA/MALA/aGRAD, and $D^{-1/3}$ for the remaining (i.e., new) methods. Panel a illustrates that as D increases, some 'classical' MCMC algorithms (RWM, MALA, aMALA and aGRAD) are stable but the CSMC algorithm breaks down. Conversely, Panel b illustrates that as T increases, the CSMC algorithm is stable in T but all 'classical' MCMC algorithms (IMH, RWM, MALA, aMALA and aGRAD) break down.

Informally, the CSMC algorithm can be interpreted as employing T separate accept–reject steps (one at each time point) which allows it to exploit the 'decorrelation-over-time' property of $\pi_T(\mathbf{x}_{1:T})$ (akin to a 'classical' MCMC algorithm with blocking in the 'time' direction as noted by Singh et al. 2017). For suitably regular problems, the CSMC algorithm therefore scales more favourably with T than 'classical' MCMC approaches as illustrated in Figure 1b.

Unfortunately, as shown in Finke and Thiery (2023), the CSMC algorithm suffers from a curse of dimension in the state dimension D: as D increases, it becomes increasingly likely that $\tilde{\mathbf{x}}_{1:T}$ coincides exactly with $\mathbf{x}_{1:T}$, i.e., the induced MCMC chain gets stuck. This is unsurprising because the CSMC algorithm generalises the IMH algorithm to $T > 1$ time steps and $N > 1$ proposals. Indeed, note that (1) is again an independent (i.e. global) proposal in the sense that it does not depend on the time-t component of the current state of the Markov chain, $\mathbf{x}_t$; and such proposals are known to scale poorly with dimension (due to the difficulty of finding efficient global proposals in high dimensions). The only potential remedy: increasing N exponentially with D, is prohibitively costly.

Existing combinations of 'classical' MCMC and CSMC. To circumvent this problem, Finke and Thiery (2023) introduced the *Particle-RWM*[2] algorithm which scatters the particles locally around the reference path (see also Shestopaloff and Neal, 2018; Malory, 2021, for related

[2]Referred to as 'random-walk CSMC' therein.

approaches). That is, conditional on the reference path $\mathbf{x}_{1:T}$, the remaining particles $\mathbf{x}_t^{-k_t}$ are proposed from a joint distribution under which

$$\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{x}_t, \delta_t \mathbf{I}),$$

for $n \neq k_t$, where $\mathbf{I}$ is the $(D \times D)$-identity matrix. As noted in Tjelmeland (2004), sampling from this joint proposal distribution can be achieved by first sampling an auxiliary variable $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t, \frac{\delta_t}{2}\mathbf{I})$ and then $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{u}_t, \frac{\delta_t}{2}\mathbf{I})$, for $n \neq k_t$. Finke and Thiery (2023) also showed that scaling the step size as $\delta_t \in \mathrm{O}(D^{-1})$ (independently of T) guarantees stability in high dimensions. This is again unsurprising because the Particle-RWM algorithm generalises the RWM algorithm with Gaussian proposals (and proposal variance δ_1) to $T > 1$ time steps and $N > 1$ proposals. Recently, Corenflos and Särkkä (2023) showed that the Particle-RWM algorithm can be viewed as a Gibbs-sampling step for the auxiliary variables $\mathbf{u}_t$ followed by a CSMC update which targets a modified Feynman–Kac model which depends on $\mathbf{u}_{1:T}$, allowing for greater flexibility in the choice proposals. Including related 'pseudo observations' $\mathbf{u}_t$ into CSMC updates had previously been suggested by Murray et al. (2013); Fearnhead and Meligkotsidou (2016); Karppinen and Vihola (2021) but primarily aimed at overcoming the problem that the CSMC algorithm mixes poorly if the initial distribution $M_1(\mathbf{x}_1)$ is diffuse (and potentially also for improving mixing in the presence of 'static' model parameters).

1.3 Contributions

Recall that in the 'classical' MCMC setting, improved performance can often be achieved by enhancing the proposal distribution using gradient or prior information. Thus, in this work, we introduce a methodology which combines the strength of CSMC methods (i.e., exploitation of the 'decorrelation-over-time' property of the target distribution) with the strengths of sophisticated 'classical' MCMC approaches (i.e., gradient-enhanced local proposals).

In the remainder of this section, we detail the contributions of this paper (Table 1 summarises our proposed methodology).

In Section 3, we introduce the following CSMC type methods which propose particles locally around the reference path guided by gradient information:

- **Particle-aMALA.** In Section 3.1, we extend the Particle-RWM algorithm to incorporate gradient information into the proposals. That is, conditional on the reference path $\mathbf{x}_{1:T}$, the remaining particles $\mathbf{x}_t^{-k_t}$ are proposed from a joint distribution under which

$$\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{x}_t + \tfrac{\delta_t}{2}\nabla_{\mathbf{x}_t} \log \pi_t(\mathbf{x}_{1:T}), \delta_t \mathbf{I}), \tag{3}$$

  for $n \neq k_t$. Sampling from this joint proposal can be achieved by first sampling an auxiliary variable $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t} \log \pi_t(\mathbf{x}_{1:t}), \frac{\delta_t}{2}\mathbf{I})$ and then $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{u}_t, \frac{\delta_t}{2}\mathbf{I})$, for $n \neq k_t$. We call this method *Particle-aMALA* because the auxiliary variables $\mathbf{u}_t$ are explicitly included in the space, i.e. they appear in the particle weights, and because the algorithm generalises a version of *auxiliary MALA (aMALA)* from Titsias and Papaspiliopoulos (2018) to $T > 1$ time steps and $N > 1$ proposals.

- **Particle-MALA.** In Section 3.2, we improve Particle-aMALA by marginalising out the auxiliary variables $\mathbf{u}_t$. We call the resulting method *Particle-MALA* because it generalises *MALA* (Besag, 1994) to $T > 1$ time steps and $N > 1$ proposals.

- **Particle-aMALA+.** In Section 3.3, we improve Particle-aMALA differently by replacing the 'filter' gradient $\nabla_{\mathbf{x}_t} \log \pi_t(\mathbf{x}_{1:t})$ in (1.3) with the 'smoothing' gradient $\nabla_{\mathbf{x}_t} \log \pi_T(\mathbf{x}_{1:T})$

which is beneficial when future observations are informative about the current state. We call the resulting method *Particle-aMALA+*.

In Section 4, we consider the special case that the Feynman–Kac model has conditionally Gaussian mutation kernels: $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))$. In this setting, we introduce the following methods which propose particles locally around the reference path guided by both gradient information and prior information:

- **Particle-aGRAD.** In Section 4.1, we propose an algorithm which, conditional on the reference path $\mathbf{x}_{1:T}$, proposes the remaining particles $\mathbf{x}_t^{-k_t}$ from a joint distribution under which
$$\mathbf{x}_t^n \sim \mathrm{N}\Big((\mathbf{I} - \mathbf{A}_t(\mathbf{x}_{t-1}^{a_{t-1}^n}))\mathbf{m}_t(\mathbf{x}_{t-1}^{a_{t-1}^n}) + \mathbf{A}_t(\mathbf{x}_{t-1}^{a_{t-1}^n})[\mathbf{x}_t + \tfrac{\delta_t}{2}\nabla_{\mathbf{x}_t} \log G_t(\mathbf{x}_{t-1:t})], \mathbf{B}_t(\mathbf{x}_{t-1}^{a_{t-1}^n})\Big), \quad (4)$$
for $n \neq k_t$, where $\mathbf{A}_t(\mathbf{x}) \coloneqq (\mathbf{C}_t(\mathbf{x}) + \frac{\delta_t}{2}\mathbf{I})^{-1}\mathbf{C}_t(\mathbf{x})$ and $\mathbf{B}_t(\mathbf{x}) \coloneqq \frac{\delta_t}{2}\mathbf{A}_t(\mathbf{x})^2 + \mathbf{A}_t(\mathbf{x})$. Sampling from this joint proposal can be achieved by first sampling an auxiliary variable $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \frac{\delta_t}{2}\nabla_{\mathbf{x}_t} \log G_t(\mathbf{x}_{t-1:t}), \frac{\delta_t}{2}\mathbf{I})$ and then $\mathbf{x}_t^n \sim M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}^{a_{t-1}^n}; \mathbf{u}_t)$, for $n \neq k_t$, where $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t) = p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t)$ is the fully-adapted auxiliary particle-filter proposal for the state-space model with Gaussian transitions $\mathbf{x}_t|\mathbf{x}_{t-1} \sim M_t(\mathbf{x}_t|\mathbf{x}_{t-1})$ and pseudo observations $\mathbf{u}_t|\mathbf{x}_t \sim \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t; \frac{\delta_t}{2}\mathbf{I})$. We call this the *Particle-aGRAD* algorithm because the auxiliary variables $\mathbf{u}_t$ again appear in the particle weights, and because it generalises the powerful *aGRAD* algorithm from Titsias and Papaspiliopoulos (2018) to $T > 1$ time steps and $N > 1$ proposals.

- **Particle-mGRAD.** In Section 4.2, under the assumption that $\mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t$ and in analogy to Section 3.2, we improve Particle-aGRAD by marginalising out the auxiliary variables $\mathbf{u}_t$. We call the resulting method *Particle-mGRAD* because it generalises the powerful *mGRAD* algorithm from Titsias and Papaspiliopoulos (2018) to $T > 1$ time steps and $N > 1$ proposals.

- **Particle-aGRAD+.** In Section 4.3, in analogy to Section 3.3, we improve Particle-aGRAD by replacing the 'filter-potential' gradients $\nabla_{\mathbf{x}_t} \log G_t(\mathbf{x}_{t-1:t})$ in (1.3) with 'smoothing-potential' gradients $\nabla_{\mathbf{x}_t} \log G_{1:T}(\mathbf{x}_{1:T})$ which may be beneficial if $G_t(\mathbf{x}_{t-1:t})$ varies significantly in $\mathbf{x}_{t-1}$. We call this method *Particle-aGRAD+*.

- **Twisted Particle-aGRAD(+).** In Section 4.4, under the assumption that $\mathbf{m}_t(\mathbf{x}_{t-1}) = \mathbf{F}_t\mathbf{x}_{t-1} + \mathbf{b}_t$ and $\mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t$, we improve Particle-aGRAD and Particle-aGRAD+ by instead using all future auxiliary variables $\mathbf{u}_{t:T}$ to propose $\mathbf{x}_t^n \sim M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}^{a_{t-1}^n}; \mathbf{u}_{t:T})$, for $n \neq k_t$, where $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{t:T}) = p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t:T})$ is the fully *twisted* particle filter proposal for the state-space model with Gaussian transitions and pseudo observations $\mathbf{u}_t$ mentioned above. We call the resulting methods *twisted Particle-aGRAD* and *twisted Particle-aGRAD+*.

In Section 4.6, we prove that Particle-aGRAD and Particle-mGRAD (and their smoothing-gradient/twisted variants) solve the 'tuning' problem of having to choose between:

1. the CSMC algorithm (which proposes particles solely based on the prior dynamics);
2. the Particle-aMALA, Particle-MALA or Particle-aMALA+ (which propose particles solely locally around the reference path).

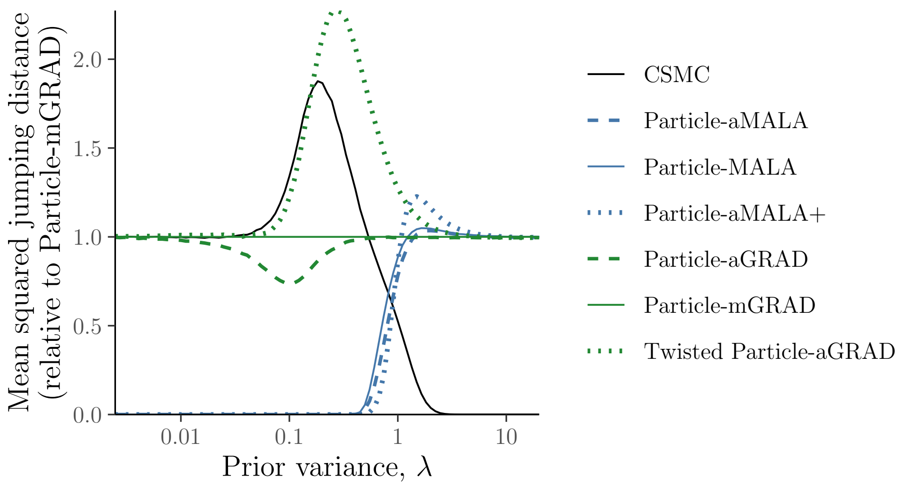


Figure 2: Empirical illustration of the 'interpolation' from Propositions 8 and 9 in the toy linear-Gaussian state-space model from Figure 1 (with $D = T = 10$).

This choice is not always clear: on the one hand, Choice 2 can exhibit superior performance in high dimensions. On the other hand, if the prior dynamics are highly informative then Choice 1 can outperform Choice 2. Specifically, we prove that the following results hold in stationarity and under the simplifying assumption that the model factorises over time, i.e., if G_t, $\mathbf{m}_t$, $\mathbf{C}_t$ (and hence $\mathbf{A}_t$ and $\mathbf{B}_t$ in (1.3)) do not depend on the state at time $t - 1$:

- **Proposition 8.** Particle-aGRAD and Particle-mGRAD reduce to the CSMC algorithm as prior dynamics become *more* informative. Informally, we then have $\mathbf{A}_t \approx \mathbf{0}$ and $\mathbf{B}_t \approx \mathbf{C}_t$ so that (1.3) reduces to (1).

- **Proposition 9.** Particle-aGRAD and Particle-mGRAD reduce to Particle-aMALA and Particle-MALA, respectively, as prior dynamics become *less* informative. Informally, we then have $\mathbf{A}_t \approx \mathbf{I}$ and $\mathbf{B}_t \approx \delta_t \mathbf{I}$ so that (1.3) reduces to (1.3).

Propositions 8 and 9 are illustrated in Figure 2 for a model in which the independence across time-steps is not verified. As a by-product, these propositions show that the aGRAD/mGRAD algorithms from Titsias and Papaspiliopoulos (2018) can be viewed as automatically interpolating between the IMH algorithm and aMALA/MALA, depending on the 'informativeness' of the prior. To our knowledge, this has not been pointed out in the literature. As another by-product, the methodology presented in this section also addresses the 'tuning problem' of having to choose whether to sample the initial latent state $\mathbf{x}_1$ within the CSMC scheme (which is preferable if the prior on the initial state is informative) or to treat it as a 'static' parameter to be sampled separately (which is preferable if this prior is diffuse, see Murray et al., 2013; Fearnhead and Meligkotsidou, 2016; Karppinen and Vihola, 2021).

In Section 5, we demonstrate the performance of our methodology on a high-dimensional multivariate stochastic volatility model, often used as a benchmark in the particle filtering literature. The different methods proposed in this article dramatically improve on existing CSMC and related methods and also on 'classical' MCMC methods in terms of effective sample size for different levels of prior informativeness.

All proofs (e.g., of the fact that the proposed methods leave $\pi_T(\mathbf{x}_{1:T})$ invariant) are deferred to the appendix. Additionally, in Appendix A, we introduce *Particle-PCNL* methods which generalise the *preconditioned Crank–Nicolson–Langevin (PCNL)* algorithm from Cotter et al. (2013) to $T > 1$ time steps and $N > 1$ proposals. The methods proposed in this work and their

Table 1: The methods mentioned in this work (new methods are in *italic*).

Method	**Section**	**Special case if $N = T = 1$**
CSMC[†]	2.1	IMH
Particle-RWM	2.2	RWM
Particle-aMALA	3.1	aMALA
Particle-MALA	3.2	MALA
Particle-aMALA+	3.3	aMALA
Particle-aGRAD	4.1	aGRAD
Particle-mGRAD	4.2	mGRAD
Particle-aGRAD+	4.3	aGRAD
Twisted Particle-aGRAD(+)	4.4	aGRAD
Particle-PCNL & more[‡]	Appendix A	PCNL

[†] In our taxonomy, CSMC could be called 'Particle-IMH'. However, the latter already refers to an entirely different algorithm in Andrieu et al. (2010).
[‡] We again also describe auxiliary-variable, smoothing-gradient ('+') and twisted versions.

special cases if $N = T = 1$ are summarised in Table 1. Note that for $T = 1$ but $N > 1$, our work implies novel *multi-proposal* versions of 'classical' MCMC kernels like MALA, aMALA, mGRAD, aGRAD and PCNL. These may be of independent interest because they can exploit parallel computing architectures for inference in non-dynamic models.

Importantly, and in keeping with existing CSMC methodology, the computational cost of all our proposed algorithms is linear in both T and N, in time and memory.

Finally, the Python code for reproducing our experiments is publicly available at `https://github.com/AdrienCorenflos/particle_mala`. It was written as a library and can be extended to accommodate other models than the ones considered here.

2 Existing methodology

2.1 CSMC (particle extension of IMH)

2.1.1 Algorithm

Assume that we can generate independent and identically distributed (IID) samples from the mutation kernels $M_t(\mathbf{x}_t|\mathbf{x}_{t-1})$. A method for constructing a π_T-invariant MCMC kernel $P_{\text{CSMC}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ is then given by the *CSMC* algorithm from Andrieu et al. (2010) which proposes N particles at each time step to build up an efficient proposal. Algorithm 1 summarises the scheme, where 'w.p.' is short for 'with probabilitity'. We also recursively define the nth surviving particle lineage at time t as

$$\mathbf{x}_{1:t}^{(n)} \coloneqq (\mathbf{x}_{1:t-1}^{(a_{t-1}^n)}, \mathbf{x}_t^n).$$

In particular, therefore, $\mathbf{x}_{t-1:t}^{(n)} = (\mathbf{x}_{t-1}^{a_{t-1}^n}, \mathbf{x}_t^n)$.

Algorithm 1 (CSMC). Given $\mathbf{x}_{1:T} \in \mathcal{X}^T$:

```
1. for $t = 1, \dots, T$,
    a) sample $k_t$ from a uniform distribution on $[N]_0$ and set $\mathbf{x}_t^{k_t} \coloneqq \mathbf{x}_t$,
    b) if $t > 1$, set $a_{t-1}^{k_t} \coloneqq k_{t-1}$ and sample $a_{t-1}^n = i$ w.p. $W_{t-1}^i$, for $n \in [N]_0 \setminus \{k_t\}$,
    c) sample $\mathbf{x}_t^n \sim M_t(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n})$ for $n \in [N]_0 \setminus \{k_t\}$,
    d) for $n \in [N]_0$, set $w_t^n \propto G_t(\mathbf{x}_{t-1:t}^{(n)})$.
    e) for $n \in [N]_0$, set $W_t^n \coloneqq w_t^n / \sum_{m=0}^N w_t^m$;
2. sample $i \in [N]_0 \setminus \{k_T\}$ w.p. $\frac{W_T^i}{1 - W_T^{k_T}}$; set $l_T \coloneqq i$ w.p. $1 \wedge \frac{1 - W_T^{k_T}}{1 - W_T^i}$; otherwise, set $l_T \coloneqq k_T$;
3. for $t = T-1, \dots, 1$, sample $l_t = i \in [N]_0$ w.p. $\frac{W_t^i Q_{t+1}(\mathbf{x}_t^i, \mathbf{x}_{t+1}^{l_{t+1}})}{\sum_{n=0}^N W_t^n Q_{t+1}(\mathbf{x}_t^n, \mathbf{x}_{t+1}^{l_{t+1}})}$;
4. return $\tilde{\mathbf{x}}_{1:T} \coloneqq (\mathbf{x}_1^{l_1}, \dots, \mathbf{x}_t^{l_T})$.
```

Algorithm 1 includes two extensions to the original presentation of the CSMC algorithm in Andrieu et al. (2010):

- Step 2 uses the so-called *forced-move* extension for CSMC algorithms which was proposed in Chopin and Singh (2013) (see also Liu, 1996). The algorithm would still be valid if we instead sampled $l_T = i \in [N]_0$ with probability W_T^i.
- Step 3 is the *backward-sampling* extension from Whiteley (2010). The algorithm would still be valid if we instead set $l_t = a_t^{l_{t+1}}$ (but typically much less efficient, especially if T is large).

Importantly, sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ as described in Algorithm 1 induces a Markov kernel $P_{\text{CSMC}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant. For sufficiently ergodic models, this MCMC kernel can yield highly efficient updates of the sequence of latent states, even if the time horizon T is large (Lee et al., 2020; Karjalainen et al., 2023).

2.1.2 Relationship with 'classical' MCMC algorithms

Interestingly, the CSMC algorithm generalises the classical *IMH* algorithm (Hastings, 1970) in the sense that the former reduces to the latter if $T = N = 1$. This can be seen as follows, where we suppress the 'time' subscript $t = 1$ everywhere to simplify the notation. Given that the current state of the Markov chain is $\mathbf{x} = \mathbf{x}^0$ (we can assume that $k = 0$ without loss of generality), Step 1c of Algorithm 1 proposes $\mathbf{x}^1 \sim M$. The remaining steps return $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^1$ as the new state with acceptance probability $1 \wedge \alpha_{\text{IMH}}(\mathbf{x}^0, \mathbf{x}^1)$, where

$$\alpha_{\text{IMH}}(\mathbf{x}^0, \mathbf{x}^1) \coloneqq \frac{1 - W^0}{1 - W^1} = \frac{G(\mathbf{x}^1)}{G(\mathbf{x}^0)} = \frac{\pi(\mathbf{x}^1)M(\mathbf{x}^0)}{\pi(\mathbf{x}^0)M(\mathbf{x}^1)}.$$

Otherwise, the old state $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^0 = \mathbf{x}$ is returned as the new state.

2.1.3 Breakdown in high dimensions

Unfortunately, as shown in Finke and Thiery (2023), Algorithm 1 suffers from a curse of dimension if D is large (unless the number of proposed particles, N, grows exponentially in D but that is prohibitive). This is not surprising since the IMH algorithm is known to break down in high dimensions (due to the difficulty of finding an efficient global proposal distribution M in high dimensions).

2.2 Particle-RWM

2.2.1 Algorithm

To circumvent the curse of dimension, Finke and Thiery (2023) (see also Shestopaloff and Neal, 2018; Malory, 2021, for related methods) developed the *particle random-walk Metropolis (Particle-RWM)* algorithm which scatters the proposed particles locally around the reference path using Gaussian perturbations as outlined in Algorithm 2.

Algorithm 2 (Particle-RWM). Implement Algorithm 1 but replace the particle proposal (Step 1c) and the weight calculation (Step 1d) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t, \frac{\delta_t}{2}\mathbf{I})$, and $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{u}_t, \frac{\delta_t}{2}\mathbf{I})$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. for $n \in [N]_0$, set $w_t^n \propto Q_t(\mathbf{x}_{t-1:t}^{(n)})$.

Notably, Step 1c marginally samples $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{x}_t, \delta_t \mathbf{I})$, for $n \neq k_t$.

2.2.2 Interpretation as a CSMC update on an extended space

Corenflos and Särkkä (2023) showed that Algorithm 2 can be derived by including the auxiliary variables $\mathbf{u}_t$ into the space and thus considering the extended distribution

$$\pi'_T(\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \coloneqq \pi_T(\mathbf{x}_{1:T}) \prod_{t=1}^{T} \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \tfrac{\delta_t}{2}\mathbf{I}),$$

which admits $\pi_T(\mathbf{x}_{1:T})$ as a marginal and which can be targeted by alternating the following two steps. Given $\mathbf{x}_{1:T} \in \mathcal{X}^T$,

1. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t, \frac{\delta_t}{2}\mathbf{I})$, for $t = 1, \dots, T$;
2. run the CSMC algorithm (Algorithm 1) but with $M_t(\mathbf{x}_t|\mathbf{x}_{t-1})$, $G_t(\mathbf{x}_{t-1:t})$, and $Q_t(\mathbf{x}_{t-1:t})$ replaced by $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t) \coloneqq \mathrm{N}(\mathbf{x}_t; \mathbf{u}_t, \frac{\delta_t}{2}\mathbf{I})$, $G'_t(\mathbf{x}_{t-1:t}) \coloneqq Q_t(\mathbf{x}_{t-1:t})$ and $Q'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_t) \coloneqq M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t) G'_t(\mathbf{x}_{t-1:t})$.

In particular, this shows that sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 2 induces a Markov kernel $P_{\text{Particle-RWM}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.

2.2.3 Relationship with 'classical' MCMC algorithms

The Particle-RWM algorithm generalises the classical (Gaussian) *RWM* algorithm of Metropolis et al. (1953) in the sense that the former reduces to the latter if $T = N = 1$. This can be seen as follows, where we again suppress the 'time' subscript $t = 1$ everywhere to simplify the notation. Given that the current state of the Markov chain is $\mathbf{x} = \mathbf{x}^0$ (we can again assume that $k = 0$ without loss of generality), Step 1c of Algorithm 2 proposes $\mathbf{x}^1 \sim \mathrm{N}(\mathbf{x}^0, \delta \mathbf{I})$. The remaining steps return $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^1$ as the new state with acceptance probability $1 \wedge \alpha_{\text{RWM}}(\mathbf{x}^0, \mathbf{x}^1)$, where

$$\alpha_{\text{RWM}}(\mathbf{x}^0, \mathbf{x}^1) \coloneqq \frac{1 - W^0}{1 - W^1} = \frac{\pi(\mathbf{x}^1)}{\pi(\mathbf{x}^0)}.$$

Otherwise, the old state $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^0 = \mathbf{x}$ is returned as the new state.

2.2.4 Stability in high dimensions

Finke and Thiery (2023) proved that the Particle-RWM algorithm circumvents the curse of dimensionality if the proposal variance is scaled as $\delta_t \in \mathrm{O}(D^{-1})$ (see also Malory, 2021, for a proof for non-Gaussian exchangeable proposals but in the case where the model factorises over time). However, from the literature on classical MCMC algorithms, it is well known that faster convergence rates can be achieved by incorporating gradient information into the proposal (Roberts and Rosenthal, 1998). Thus, in the next section, we extend the Particle-RWM to allow for gradient-informed proposals.

3 Particle extensions of MALA and aMALA

3.1 Particle-aMALA

We now propose *Particle-aMALA*, a method which extends the Particle-RWM algorithm from Finke and Thiery (2023) by allowing for the use of gradient information in the proposal. For the moment, gradients are taken w.r.t. the filtering densities and we employ an indicator $\kappa \in \{0, 1\}$ to permit switching off the use of gradient information.

We now write

$$M'_t(\mathbf{x}_t | \mathbf{x}_{t-1}; \mathbf{u}_t) \coloneqq \mathrm{N}(\mathbf{x}_t; \mathbf{u}_t, \tfrac{\delta_t}{2}\mathbf{I}), \tag{5}$$

$$G'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_t) \coloneqq Q_t(\mathbf{x}_{t-1:t}) \frac{\mathrm{N}(\mathbf{u}_t; \mathbf{x}_t + \kappa \frac{\delta_t}{2} \nabla_{\mathbf{x}_t} \log \pi_t(\mathbf{x}_{1:t}), \frac{\delta_t}{2}\mathbf{I})}{\mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \frac{\delta_t}{2}\mathbf{I})}, \tag{6}$$

as well as $Q'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_t) \coloneqq M'_t(\mathbf{x}_t | \mathbf{x}_{t-1}; \mathbf{u}_t) G'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_t)$, where we note that

$$\nabla_{\mathbf{x}_t} \log \pi_t(\mathbf{x}_{1:t}) = \nabla_{\mathbf{x}_t} \log Q_t(\mathbf{x}_{t-1:t}).$$

A single iteration of the Particle-aMALA is then as follows.

Algorithm 3 (Particle-aMALA). Implement Algorithm 1 but replace the particle proposal (Step 1c) and the weight calculation (Step 1d) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \kappa \frac{\delta_t}{2} \nabla_{\mathbf{x}_t} \log \pi_t(\mathbf{x}_{1:t}), \frac{\delta_t}{2}\mathbf{I})$, and $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{u}_t, \frac{\delta_t}{2}\mathbf{I})$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. for $n \in [N]_0$, set $w_t^n \propto G'_t(\mathbf{x}_{t-1:t}^{(n)}; \mathbf{u}_t)$,

and also replace $Q_{t+1}(\,\cdot\,)$ in the backward kernel in Step 3 by $Q'_{t+1}(\,\cdot\,; \mathbf{u}_{t+1})$.

Step 1c marginally samples $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{x}_t + \kappa \frac{\delta_t}{2} \nabla_{\mathbf{x}_t} \log \pi_t(\mathbf{x}_{1:t}), \delta_t \mathbf{I})$, for $n \neq k_t$. This follows from Lemma 3 in Appendix C.

Proposition 1 (validity of Particle-aMALA). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 3 induces a Markov kernel $P_{\text{Particle-aMALA}}(\tilde{\mathbf{x}}_{1:T} | \mathbf{x}_{1:T})$ which leaves π_T invariant.*

3.2 Particle-MALA

In this section, we analytically integrate out the auxiliary variables $\mathbf{u}_t$ appearing in the weights of the Particle-aMALA. A single iteration of the resulting methodology – which we term the *Particle-MALA* – is as follows, where we write

$$\log H_{t,\boldsymbol{\phi}}(\mathbf{x}, \bar{\mathbf{x}}) \coloneqq \frac{1}{\delta_t}\Big[2\boldsymbol{\phi}^{\mathrm{T}}(\bar{\mathbf{x}} - \mathbf{x}) - \tfrac{N}{N+1}\boldsymbol{\phi}^{\mathrm{T}}\boldsymbol{\phi}\Big]. \tag{7}$$

Algorithm 4 (Particle-MALA). Implement Algorithm 1 but replace the particle proposal (Step 1c) and the weight calculation (Step 1d) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log\pi_t(\mathbf{x}_{1:t}), \frac{\delta_t}{2}\mathbf{I})$, and $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{u}_t, \frac{\delta_t}{2}\mathbf{I})$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. set $\bar{\mathbf{x}}_t \coloneqq \frac{1}{N+1}\sum_{n=0}^{N}\mathbf{x}_t^n$ and, for $n \in [N]_0$,

$$w_t^n \propto Q_t(\mathbf{x}_{t-1:t}^{(n)})H_{t,\kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t^n}\log Q_t(\mathbf{x}_{t-1:t}^{(n)})}(\mathbf{x}_t^n, \bar{\mathbf{x}}_t).$$

Step 1d pre-computes $\bar{\mathbf{x}}_t$ to ensure that the algorithm can still be implemented in $\mathrm{O}(N)$ operations even though the weight of the nth particle now depends on the values of all $N+1$ particles. However, note that the auxiliary variables $\mathbf{u}_t$ no longer appear in the weights.

Remark 1 (Particle-aMALA 'exactly approximates' Particle-MALA). *Note that the Particle-aMALA differs from the Particle-MALA only in the definition of the weights (and the backward-sampling weights). This allows us to interpret the former as a 'noisy' version of the latter. Indeed, write the unnormalised weight of the nth particle at time-t in the Particle-aMALA as $w_t^n(\mathbf{u}_t)$, whilst w_t^n denotes the corresponding weight under the Particle-MALA (which does not depend on the auxiliary variable $\mathbf{u}_t$). Then we have*

$$\frac{w_t^n(\mathbf{u}_t)}{w_t^{k_t}(\mathbf{u}_t)} = \frac{w_t^n}{w_t^{k_t}} \times \frac{q_t^{-n}(\mathbf{u}_t|\mathbf{x}_t^{-n}, \mathbf{x}_t^n; \mathcal{H}_{t-1})}{q_t^{-k_t}(\mathbf{u}_t|\mathbf{x}_t^{-k_t}, \mathbf{x}_t^{k_t}; \mathcal{H}_{t-1})},$$

where $q_t^{-n}(\mathbf{u}_t|\mathbf{x}_t^{-n}, \mathbf{x}_t^n; \mathcal{H}_{t-1}) = \mathrm{N}(\mathbf{u}_t; \bar{\mathbf{x}}_t + \kappa\frac{\delta_t}{2(N+1)}\nabla_{\mathbf{x}_t^n}\log Q_t(\mathbf{x}_{t-1:t}^{(n)}), \frac{\delta_t}{2(N+1)}\mathbf{I})$ is the conditional distribution of $\mathbf{u}_t$ under the joint distribution of all random variables generated by Algorithm 3 up to (and including) time t assuming the reference particle at time t is placed in position n (and $\mathcal{H}_{t-1}$ denotes the history of the particle system, i.e. all particles and ancestor indices up to time $t-1$). This conditional distribution follows from Lemma 3 in Appendix C. In particular, we therefore have

$$\mathbb{E}\left[\frac{w_t^n(\mathbf{u}_t)}{w_t^{k_t}(\mathbf{u}_t)}\right] = \frac{w_t^n}{w_t^{k_t}},$$

where the expectation is taken w.r.t. $q_t^{-k_t}(\mathbf{u}_t|\mathbf{x}_t^{-k_t}, \mathbf{x}_t^{k_t}; \mathcal{H}_{t-1})$. Interestingly, for the Particle-RWM algorithm (recovered by setting $\kappa = 0$), the 'auxiliary' and 'marginal' variants are statistically equivalent.

Proposition 2 (validity of Particle-MALA). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 4 induces a Markov kernel $P_{\text{Particle-MALA}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

3.3 Particle-aMALA+

In this section, we extend the Particle-aMALA in a different manner: we now modify the algorithm so that the proposal distributions incorporate gradients w.r.t. the joint smoothing distribution π_T rather than w.r.t. the filters, π_t. This can be beneficial if there is a significant discrepancy between the marginal distribution of $\mathbf{x}_t$ under the former and the latter as is typically the case if D is large. Indeed, this discrepancy is likely the reason for the decay in performance of Particle-aMALA and Particle-MALA for very large D visible in Figure 1a.

For $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_t)$ and $G'_t(\mathbf{x}_{t-1:t};\mathbf{u}_t)$ still defined as in the Particle-aMALA algorithm (i.e., as in (3.1) and (3.1)), we now write

$$G'_t(\mathbf{x}_{t-2:t};\mathbf{u}_{t-1:t}) \coloneqq G'_t(\mathbf{x}_{t-1:t};\mathbf{u}_t)\frac{\mathrm{N}(\mathbf{u}_{t-1};\mathbf{x}_{t-1}+\kappa\frac{\delta_{t-1}}{2}\nabla_{\mathbf{x}_{t-1}}\log\pi_T(\mathbf{x}_{1:T}),\frac{\delta_{t-1}}{2}\mathbf{I})}{\mathrm{N}(\mathbf{u}_{t-1};\mathbf{x}_{t-1}+\kappa\frac{\delta_{t-1}}{2}\nabla_{\mathbf{x}_{t-1}}\log\pi_{t-1}(\mathbf{x}_{1:t-1}),\frac{\delta_{t-1}}{2}\mathbf{I})},$$

as well as $Q'_t(\mathbf{x}_{t-2:t};\mathbf{u}_{t-1:t}) \coloneqq M'_t(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_t)G'_t(\mathbf{x}_{t-2:t};\mathbf{u}_{t-1:t})$, where we note that

$$\nabla_{\mathbf{x}_t}\log\pi_T(\mathbf{x}_{1:T}) = \nabla_{\mathbf{x}_t}[\log Q_t(\mathbf{x}_{t-1:t}) + \log Q_{t+1}(\mathbf{x}_{t:t+1})].$$

A single iteration of the resulting 'smoothing-gradient' methodology – which we term the *Particle-aMALA+* – is then as follows.

Algorithm 5 (Particle-aMALA+). Implement Algorithm 1 but replace the particle proposal (Step 1c), the weight calculation (Step 1d), and backward sampling (Step 3) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log\pi_T(\mathbf{x}_{1:T}), \frac{\delta_t}{2}\mathbf{I})$, and $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{u}_t, \frac{\delta_t}{2}\mathbf{I})$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. for $n \in [N]_0$, set $w_t^n \propto G'_t(\mathbf{x}_{t-2:t}^{(n)};\mathbf{u}_{t-1:t})$,

3. for $t = T-1,\dots,1$, sample $l_t = i \in [N]_0$ w.p.

$$\frac{W_t^i Q'_{t+1}((\mathbf{x}_{t-1:t}^{(i)},\mathbf{x}_{t+1}^{l_{t+1}});\mathbf{u}_{t:t+1})Q'_{t+2}((\mathbf{x}_t^i,\mathbf{x}_{t+1}^{l_{t+1}},\mathbf{x}_{t+2}^{l_{t+2}});\mathbf{u}_{t+1:t+2})}{\sum_{n=0}^N W_t^n Q'_{t+1}((\mathbf{x}_{t-1:t}^{(n)},\mathbf{x}_{t+1}^{l_{t+1}});\mathbf{u}_{t:t+1})Q'_{t+2}((\mathbf{x}_t^n,\mathbf{x}_{t+1}^{l_{t+1}},\mathbf{x}_{t+2}^{l_{t+2}});\mathbf{u}_{t+1:t+2})}.$$

In Step 3, we recall the convention that any quantity with 'time' index $t > T$ should be ignored, so that $Q'_{T+1} \equiv 1$. Some comments about Algorithm 5 are in order.

- Step 1c marginally samples $\mathbf{x}_t^n \sim \mathrm{N}(\mathbf{x}_t + \kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log\pi_T(\mathbf{x}_{1:T}), \delta_t\mathbf{I})$, for $n \neq k_t$. This is in contrast to the Particle-aMALA and Particle-MALA, whose (marginal) proposal distribution is centred around $\mathbf{x}_t + \kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log\pi_t(\mathbf{x}_{1:t})$.
- Steps 1d and 3 are similar to the weight-calculation and backward-sampling steps in the previous algorithms. The only difference here is that the model is now no longer (first-order) Markov in the sense that the (incremental) weights at time t now also depend on the state at time $t-2$.

Proposition 3 (validity of Particle-aMALA+). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 5 induces a Markov kernel $P_{\text{Particle-MALA}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

3.4 Relationship with other methods

We end this section by relating the proposed algorithms to existing methodologies.

1. **Generalisation of Particle-RWM and RWM.** If $\kappa = 0$, then the algorithms introduced in this section (Particle-aMALA, Particle-MALA and Particle-aMALA+) do not make use of any gradient information and reduce to the Particle-RWM algorithm. In particular, if $T = N = 1$, they thus reduce to the RWM algorithm.

2. **Generalisation of aMALA.** For $\kappa = 1$, the Particle-aMALA (and similarly the Particle-aMALA+) algorithm generalise the *auxiliary MALA (aMALA)* from Titsias and Papaspiliopoulos (2018) in the sense that the former reduces to the latter if $T = N = 1$. This can be seen as follows, where we again suppress the 'time' subscript $t = 1$ everywhere. Given that the current state of the Markov chain is $\mathbf{x} = \mathbf{x}^0$ (we can assume that $k = 0$ without loss of generality), Step 1c of Algorithm 3 first refreshes the auxiliary variable by sampling $\mathbf{u} \sim \mathrm{N}(\mathbf{x}^0 + \frac{\delta}{2}\nabla \log \pi(\mathbf{x}^0), \frac{\delta}{2}\mathbf{I})$ and then proposes $\mathbf{x}^1 \sim \mathrm{N}(\mathbf{u}, \frac{\delta}{2}\mathbf{I})$. The remaining steps return $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^1$ as the new state with acceptance probability $1 \wedge \alpha_{\mathrm{aMALA}}(\mathbf{x}^0, \mathbf{x}^1; \mathbf{u})$, where
$$\alpha_{\mathrm{aMALA}}(\mathbf{x}^0, \mathbf{x}^1; \mathbf{u}) \coloneqq \frac{1 - W^0}{1 - W^1} = \frac{\pi(\mathbf{x}^1)\,\mathrm{N}(\mathbf{u}; \mathbf{x}^1 + \frac{\delta}{2}\nabla \log \pi(\mathbf{x}^1), \frac{\delta}{2}\mathbf{I})\,\mathrm{N}(\mathbf{x}^0; \mathbf{u}, \frac{\delta}{2}\mathbf{I})}{\pi(\mathbf{x}^0)\,\mathrm{N}(\mathbf{u}; \mathbf{x}^0 + \frac{\delta}{2}\nabla \log \pi(\mathbf{x}^0), \frac{\delta}{2}\mathbf{I})\,\mathrm{N}(\mathbf{x}^1; \mathbf{u}, \frac{\delta}{2}\mathbf{I})}.$$
Otherwise, the old state $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^0 = \mathbf{x}$ is returned as the new state. This induces the same Markov chain on $\mathcal{X}$ as the aMALA from Titsias and Papaspiliopoulos (2018) (the only difference relates to a re-centring of the auxiliary variables $\mathbf{u}$ previously discussed in Corenflos and Särkkä (2023) but this does not change the law of the Markov chain on the marginal space which does not include the auxiliary variable).

3. **Generalisation of MALA.** Still taking $\kappa = 1$, the Particle-MALA generalises the *Metropolis-adjusted Langevin algorithm (MALA)* (Besag, 1994) in the sense that the former reduces to the latter if $T = N = 1$. This can be seen as follows, where use the same notational conventions as in the case of aMALA above. Step 1c of Algorithm 4 then marginally proposes $\mathbf{x}^1 \sim \mathrm{N}(\mathbf{x}^0 + \frac{\delta}{2}\nabla \log \pi(\mathbf{x}^0), \delta\mathbf{I})$. The remaining steps return $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^1$ as the new state with acceptance probability $1 \wedge \alpha_{\mathrm{MALA}}(\mathbf{x}^0, \mathbf{x}^1)$, where
$$\alpha_{\mathrm{MALA}}(\mathbf{x}^0, \mathbf{x}^1) \coloneqq \frac{1 - W^0}{1 - W^1} = \frac{\pi(\mathbf{x}^1)\,\mathrm{N}(\mathbf{x}^0; \mathbf{x}^1 + \frac{\delta}{2}\nabla \log \pi(\mathbf{x}^1), \delta\mathbf{I})}{\pi(\mathbf{x}^0)\,\mathrm{N}(\mathbf{x}^1; \mathbf{x}^0 + \frac{\delta}{2}\nabla \log \pi(\mathbf{x}^0), \delta\mathbf{I})}.$$
Otherwise, the old state $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^0 = \mathbf{x}$ is returned as the new state.

   In particular, Remark 1 shows that we can view the aMALA as a 'noisy' version of MALA (as already mentioned in Titsias and Papaspiliopoulos, 2018) because, dropping the time subscript again, by Lemma 3:
$$\alpha_{\mathrm{aMALA}}(\mathbf{x}^0, \mathbf{x}^1; \mathbf{u}) = \alpha_{\mathrm{MALA}}(\mathbf{x}^0, \mathbf{x}^1)\frac{\mathrm{N}(\mathbf{u}; \bar{\mathbf{x}} + \frac{\delta}{4}\nabla \log \pi(\mathbf{x}^1), \frac{\delta}{4}\mathbf{I})}{\mathrm{N}(\mathbf{u}; \bar{\mathbf{x}} + \frac{\delta}{4}\nabla \log \pi(\mathbf{x}^0), \frac{\delta}{4}\mathbf{I})},$$
   where $\bar{\mathbf{x}} = (\mathbf{x}^0 + \mathbf{x}^1)/2$, and hence
$$\mathbb{E}[\alpha_{\mathrm{aMALA}}(\mathbf{x}^0, \mathbf{x}^1; \mathbf{u})] = \alpha_{\mathrm{MALA}}(\mathbf{x}^0, \mathbf{x}^1),$$
   where the expectation is w.r.t. the conditional distribution of $\mathbf{u}$ under the joint distribution of the random variables sampled in Step 1c of the Particle-aMALA, i.e. w.r.t. $\mathrm{N}(\bar{\mathbf{x}} + \frac{\delta}{4}\nabla \log \pi(\mathbf{x}^0), \frac{\delta}{4}\mathbf{I})$. In other words, this algorithm is the same as MALA except that the acceptance ratio is 'randomised' in the sense that it is multiplied by a non-negative random variable whose expectation is 1. Other examples of such algorithms can be found in Ceperley and Dewing (1999); Nicholls et al. (2012); see also Finke (2015, Section 3.3.3) for a discussion as well as Andrieu and Vihola (2016, page 2669) for a simple argument showing that the asymptotic variance of aMALA cannot be smaller than that of MALA.

4 Particle extensions of mGRAD and aGRAD

4.1 Particle-aGRAD

The gradient-informed algorithms (Particle-MALA, etc) developed in Section 3 can be expected to improve upon the Particle-RWM algorithm in the same way that aMALA/MALA improve upon the RWM algorithm. However, they may underperform compared to the CSMC algorithm when the prior dynamics of the latent states are highly informative in the same way that MALA can underperform relative to the IMH algorithm (with prior as proposal) if the prior is highly informative. Additionally, note that the algorithms from Section 3 employ proposals that are *separable* in the sense that, given the reference path, the marginal proposal distribution of $\mathbf{x}_t^n$ does not depend on the ancestor particle $\mathbf{x}_{t-1}^{a_{t-1}^n}$ (that is, separability implies that the weight-calculation and resampling steps could be postponed until *after* all particles have been proposed); such separable proposals can be expected to perform poorly if the latent states are highly correlated across time.

In this section, we further incorporate (conditionally) Gaussian prior dynamics into the particle proposals and thus interpolate between the CSMC algorithm and the gradient-informed algorithms of Section 3. Our construction generalises the aGRAD and mGRAD algorithms of Titsias and Papaspiliopoulos (2018). In particular, the algorithms introduced in this section do not imply separable proposals, i.e., the proposal kernel for particle $\mathbf{x}_t^n$ will generally depend on its ancestor particle $\mathbf{x}_{t-1}^{a_{t-1}^n}$.

Specifically, in this section, we consider the special case of the generic Feynman–Kac model from (1.1) in which we can find a decomposition $Q_t(\mathbf{x}_{t-1:t}) = M_t(\mathbf{x}_t|\mathbf{x}_{t-1})G_t(\mathbf{x}_{t-1:t})$, such that

$$M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1})), \tag{8}$$

is a Gaussian transition density whose mean $\mathbf{m}_t(\mathbf{x}_{t-1})$ and non-singular covariance matrix $\mathbf{C}_t(\mathbf{x}_{t-1})$ may depend on the previous state $\mathbf{x}_{t-1}$, for $t > 1$; and that $M_1(\mathbf{x}_1) = \mathrm{N}(\mathbf{x}_1; \mathbf{m}_1, \mathbf{C}_1)$.

Example 2 (state-space model, continued). *The methods proposed in this section immediately apply with $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) \coloneqq f_t(\mathbf{x}_t|\mathbf{x}_{t-1})$ if the state-space model has conditionally Gaussian dynamics, i.e. if $f_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))$, by taking $G_t(\mathbf{x}_{t-1:t}) = g_t(\mathbf{y}_t|\mathbf{x}_t)$. However, they may often still apply to state-space models with non-Gaussian dynamics via a change of measure, i.e., by taking $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) \coloneqq \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))$ and $G_t(\mathbf{x}_{t-1:t}) = f_t(\mathbf{x}_t|\mathbf{x}_{t-1})g_t(\mathbf{y}_t|\mathbf{x}_t)/\,\mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))$, or through a suitable transformation.*

The first method proposed in this section is termed *Particle-aGRAD*. Conditional on the auxiliary variables $\mathbf{u}_{1:T}$, it can be viewed as a CSMC algorithm whose proposal kernels are those of the fully-adapted auxiliary particle filter for the state-space model defined by the Gaussian transitions $p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))$ from (4.1) and 'pseudo observations' $\mathbf{u}_t$ with $p(\mathbf{u}_t|\mathbf{x}_t) = \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \frac{\delta_t}{2}\mathbf{I})$. We now write

$$\begin{aligned} M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t) &\coloneqq p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t) \\ &\propto \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))\,\mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \tfrac{\delta_t}{2}\mathbf{I}) \\ &\propto \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t'(\mathbf{x}_{t-1}, \mathbf{u}_t), \mathbf{C}_t'(\mathbf{x}_{t-1})), \end{aligned} \tag{9}$$

with

$$\begin{aligned} \mathbf{m}_t'(\mathbf{x}, \mathbf{u}) &\coloneqq \mathbf{m}_t(\mathbf{x}) + \mathbf{A}_t(\mathbf{x})[\mathbf{u} - \mathbf{m}_t(\mathbf{x})], && (10) \\ \mathbf{C}_t'(\mathbf{x}) &\coloneqq (\mathbf{I} - \mathbf{A}_t(\mathbf{x}))\mathbf{C}_t(\mathbf{x}) = \tfrac{\delta_t}{2}\mathbf{A}_t(\mathbf{x}), && (11) \\ \mathbf{A}_t(\mathbf{x}) &\coloneqq (\mathbf{C}_t(\mathbf{x}) + \tfrac{\delta_t}{2}\mathbf{I})^{-1}\mathbf{C}_t(\mathbf{x}), \end{aligned}$$

as well as

$$G'_t(\mathbf{x}_{t-1:t};\mathbf{u}_t) \coloneqq Q_t(\mathbf{x}_{t-1:t})\frac{\mathrm{N}(\mathbf{u}_t;\mathbf{x}_t+\kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log G_t(\mathbf{x}_{t-1:t}),\frac{\delta_t}{2}\mathbf{I})}{M'_t(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_t)}, \tag{12}$$

and $Q'_t(\mathbf{x}_{t-1:t};\mathbf{u}_t) \coloneqq M'_t(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_t)G'_t(\mathbf{x}_{t-1:t};\mathbf{u}_t)$. A single iteration of the Particle-aGRAD algorithm is as follows.

Algorithm 6 (Particle-aGRAD). Implement Algorithm 1 but replace the particle proposal (Step 1c) and the weight calculation (Step 1d) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t+\kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log G_t(\mathbf{x}_{t-1:t}),\frac{\delta_t}{2}\mathbf{I})$, and $\mathbf{x}_t^n \sim M'_t(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n};\mathbf{u}_t)$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. for $n \in [N]_0$, set $w_t^n \propto G'_t(\mathbf{x}_{t-1:t}^{(n)};\mathbf{u}_t)$,

and also replace $Q_{t+1}(\,\cdot\,)$ in the backward kernel in Step 3 by $Q'_{t+1}(\,\cdot\,;\mathbf{u}_t)$.

Proposition 4 (validity of Particle-aGRAD). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 6 induces a Markov kernel $P_{\text{Particle-aGRAD}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

4.2 Particle-mGRAD

In this section, we analytically integrate out the auxiliary variables $\mathbf{u}_t$ which appeared in the weights of the Particle-aGRAD algorithm. Here we consider the case when the covariance matrices appearing in the conditionally Gaussian mutation kernel (4.1) do not depend on the previous state, i.e.,

$$\mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t, \tag{13}$$

which then also implies that $\mathbf{A}_t(\mathbf{x}_{t-1}) = \mathbf{A}_t$. A single iteration of the resulting methodology – which we term the *Particle-mGRAD* algorithm – is as follows, where we write

$$\begin{aligned}\log H_{t,\boldsymbol{\phi}}(\mathbf{x},\mathbf{v},\bar{\mathbf{x}},\bar{\mathbf{v}}) = {} & \tfrac{1}{2}(\mathbf{x}-\mathbf{v})^{\mathrm{T}}((\tfrac{\delta_t}{2}\mathbf{A}_t)^{-1}+\mathbf{G}_t)(\mathbf{x}-\mathbf{v})\\ & - [\tfrac{1}{2}N(\mathbf{x}+\boldsymbol{\phi})^{\mathrm{T}}\mathbf{A}_t + (\mathbf{x}-\mathbf{v})^{\mathrm{T}}]\mathbf{G}_t(\mathbf{x}+\boldsymbol{\phi})\\ & + (N+1)(\bar{\mathbf{x}}-\bar{\mathbf{v}})^{\mathrm{T}}\mathbf{G}_t(\mathbf{v}+\boldsymbol{\phi}),\end{aligned}$$

for $\mathbf{G}_t \coloneqq \frac{2}{\delta_t}(\mathbf{I}+N\mathbf{A}_t)^{-1}$.

Algorithm 7 (Particle-mGRAD). Implement Algorithm 1 but replace the particle proposal (Step 1c) and the weight calculation (Step 1d) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t+\kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log G_t(\mathbf{x}_{t-1:t}),\frac{\delta_t}{2}\mathbf{I})$ and $\mathbf{x}_t^n \sim M'_t(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n};\mathbf{u}_t)$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. set $\bar{\mathbf{x}}_t \coloneqq \frac{1}{N+1}\sum_{n=0}^N \mathbf{x}_t^n$, $\mathbf{v}_t^n \coloneqq (\mathbf{I}-\mathbf{A}_t)\mathbf{m}_t(\mathbf{x}_{t-1}^{a_{t-1}^n})$, $\bar{\mathbf{v}}_t \coloneqq \frac{1}{N+1}\sum_{n=0}^N \mathbf{v}_t^n$, and, for $n \in [N]_0$,

$$w_t^n \propto Q_t(\mathbf{x}_{t-1:t}^{(n)})H_{t,\kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t^n}\log G_t(\mathbf{x}_{t-1:t}^{(n)})}(\mathbf{x}_t^n,\mathbf{v}_t^n,\bar{\mathbf{x}}_t,\bar{\mathbf{v}}_t). \tag{14}$$

Remark 2 (Particle-aGRAD 'exactly approximates' Particle-mGRAD). *In analogue to the relationship between Particle-aMALA and Particle-MALA discussed in Remark 1, Particle-aGRAD is a noisy version of Particle-mGRAD. That is, letting $w_t^n(\mathbf{u}_t)$ and w_t^n be the unnormalised weights under Particle-aGRAD and Particle-mGRAD, respectively, we have*

$$\mathbb{E}\left[\frac{w_t^n(\mathbf{u}_t)}{w_t^{k_t}(\mathbf{u}_t)}\right] = \frac{w_t^n}{w_t^{k_t}},$$

where the expectation is taken with respect to the conditional distribution of $\mathbf{u}_t$ under the joint distribution of all random variables generated by Algorithm 6 up to (and including) time t.

Proposition 5 (validity of Particle-mGRAD). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 7 induces a Markov kernel $P_{\text{Particle-aGRAD}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

4.3 Particle-aGRAD+

While the algorithm of Section 6 incorporates information from the smoothing distribution by merit of not modifying the latent dynamics, it may happen that the potential $G_t(\mathbf{x}_{t-1:t})$ strongly depends on $\mathbf{x}_{t-1}$. In this case, considering the 'myopic' gradient information $\nabla_{\mathbf{x}_t} \log G_t(\mathbf{x}_{t-1:t})$ may not suffice to improve the mixing of the algorithm and information from $\mathbf{x}_{t-1}$ may then be beneficial. Similarly to Section 3.3, in this section, we extend the Particle-aGRAD algorithm to incorporate gradients w.r.t. the 'smoothing potential' $G_{1:T}(\mathbf{x}_{1:T}) = \prod_{t=1}^T G_t(\mathbf{x}_{t-1:t})$ rather than w.r.t. the 'filtering potential' $\prod_{s=1}^t G_s(\mathbf{x}_{s-1:s})$.

For $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t)$ and $G_t'(\mathbf{x}_{t-1:t}; \mathbf{u}_t)$ still defined as in the Particle-aGRAD algorithm (i.e., as in (4.1) and (4.1)), we now write

$$G_t'(\mathbf{x}_{t-2:t}; \mathbf{u}_{t-1:t}) \coloneqq G_t'(\mathbf{x}_{t-1:t}; \mathbf{u}_t) \frac{\mathrm{N}(\mathbf{u}_{t-1}; \mathbf{x}_{t-1} + \kappa \frac{\delta_{t-1}}{2} \nabla_{\mathbf{x}_{t-1}} \log G_{1:T}(\mathbf{x}_{1:T}), \frac{\delta_{t-1}}{2}\mathbf{I})}{\mathrm{N}(\mathbf{u}_{t-1}; \mathbf{x}_{t-1} + \kappa \frac{\delta_{t-1}}{2} \nabla_{\mathbf{x}_{t-1}} \log G_{t-1}(\mathbf{x}_{t-2:t-1}), \frac{\delta_{t-1}}{2}\mathbf{I})},$$

as well as $Q_t'(\mathbf{x}_{t-2:t}; \mathbf{u}_{t-1:t}) \coloneqq M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t) G_t'(\mathbf{x}_{t-2:t}; \mathbf{u}_{t-1:t})$, where we note that

$$\nabla_{\mathbf{x}_t} \log G_{1:T}(\mathbf{x}_{1:T}) = \nabla_{\mathbf{x}_t} \log[G_t(\mathbf{x}_{t-1:t}) + \log G_{t+1}(\mathbf{x}_{t:t+1})].$$

A single iteration of the resulting 'smoothing-gradient' methodology – which we term the *Particle-aGRAD+* algorithm – is as follows.

Algorithm 8 (Particle-aGRAD+). Implement Algorithm 1 but replace the particle proposal (Step 1c), the weight calculation (Step 1d), and backward sampling (Step 3) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \kappa \frac{\delta_t}{2} \nabla_{\mathbf{x}_t} \log G_{1:T}(\mathbf{x}_{1:T}), \frac{\delta_t}{2}\mathbf{I})$, and $\mathbf{x}_t^n \sim M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t)$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. for $n \in [N]_0$, set $w_t^n \propto G_t'(\mathbf{x}_{t-2:t}^{(n)}; \mathbf{u}_{t-1:t})$,

3. for $t = T-1, \dots, 1$, sample $l_t = i \in [N]_0$ w.p.

$$\frac{W_t^i Q_{t+1}'((\mathbf{x}_{t-1:t}^{(i)}, \mathbf{x}_{t+1}^{l_{t+1}}); \mathbf{u}_{t:t+1}) Q_{t+2}'((\mathbf{x}_t^i, \mathbf{x}_{t+1}^{l_{t+1}}, \mathbf{x}_{t+2}^{l_{t+2}}); \mathbf{u}_{t+1:t+2})}{\sum_{n=0}^N W_t^n Q_{t+1}'((\mathbf{x}_{t-1:t}^{(n)}, \mathbf{x}_{t+1}^{l_{t+1}}); \mathbf{u}_{t:t+1}) Q_{t+2}'((\mathbf{x}_t^n, \mathbf{x}_{t+1}^{l_{t+1}}, \mathbf{x}_{t+2}^{l_{t+2}}); \mathbf{u}_{t+1:t+2})}.$$

Note that if $G_t(\mathbf{x}_{t-1:t}) = G_t(\mathbf{x}_t)$ does not depend on $\mathbf{x}_{t-1}$, then the Particle-aGRAD+ algorithm coincides with the Particle-aGRAD algorithm. However, when $G_t(\mathbf{x}_{t-1:t})$ varies highly in $\mathbf{x}_{t-1}$, their behaviours may differ substantially.

Proposition 6 (validity of Particle-aGRAD+). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 8 induces a Markov kernel $P_{\text{Particle-aGRAD+}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

4.4 Twisted Particle-aGRAD(+)

Recall that, conditionally on the auxiliary variables $\mathbf{u}_{1:T}$, the Particle-aGRAD algorithm could be viewed as a CSMC algorithm whose proposal kernels $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_t) = p(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{u}_t)$ are those of the fully-adapted auxiliary particle filter for the state-space model which is defined by the Gaussian transitions $p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t;\mathbf{m}_t(\mathbf{x}_{t-1}),\mathbf{C}_t(\mathbf{x}_{t-1}))$ from (4.1) and observation densities $p(\mathbf{u}_t|\mathbf{x}_t) = \mathrm{N}(\mathbf{u}_t;\mathbf{x}_t,\frac{\delta_t}{2}\mathbf{I})$.

In this section (and in this section only), we make the more restrictive assumption that the transition kernel from (4.1) is not only Gaussian but also affine, i.e.,

$$\mathbf{m}_t(\mathbf{x}_{t-1}) = \mathbf{F}_t\mathbf{x}_{t-1} + \mathbf{b}_t, \quad \text{and} \quad \mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t, \tag{15}$$

for some $\mathbf{F}_t \in \mathbb{R}^{D\times D}$, $\mathbf{b}_t \in \mathbb{R}^D$, and some covariance matrix $\mathbf{C}_t \in \mathbb{R}^{D\times D}$. Under (4.4), we can then go one step further and implement the fully *twisted* particle filter (Whiteley and Lee, 2014; Guarniero et al., 2017; Heng et al., 2020) proposal which conditions on *all* future pseudo observations $\mathbf{u}_{t:T}$. That is, we now write

$$\begin{aligned}
M'_t(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_{t:T}) &\coloneqq p(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{u}_{t:T}) \\
&\propto \int_{\mathcal{X}^{T-t}} \left[\prod_{s=t}^{T} \mathrm{N}(\mathbf{x}_s;\mathbf{F}_s\mathbf{x}_{s-1}+\mathbf{b}_s,\mathbf{C}_s)\,\mathrm{N}(\mathbf{u}_s;\mathbf{x}_s,\tfrac{\delta_s}{2}\mathbf{I})\right]\mathrm{d}\mathbf{x}_{t+1:T} \\
&\propto \mathrm{N}(\mathbf{x}_t;\mathbf{F}'_t\mathbf{x}_{t-1}+\mathbf{b}'_t,\mathbf{C}'_t), \qquad (16)\\
G'_t(\mathbf{x}_{t-1:t};\mathbf{u}_{t:T}) &\coloneqq Q_t(\mathbf{x}_{t-1:t})\frac{\mathrm{N}(\mathbf{u}_t;\mathbf{x}_t+\kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log G_t(\mathbf{x}_{t-1:t}),\frac{\delta_t}{2}\mathbf{I})}{M'_t(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_{t:T})},
\end{aligned}$$

as well as $Q'_t(\mathbf{x}_{t-1:t};\mathbf{u}_{t:T}) \coloneqq M'_t(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_{t:T})G'_t(\mathbf{x}_{t-1:t};\mathbf{u}_{t:T})$. Here, $\mathbf{b}'_t \in \mathbb{R}^D$ and $\mathbf{F}'_t, \mathbf{C}'_t \in \mathbb{R}^{D\times D}$ can be obtained via Kalman-filtering recursions as explained in Appendix B.

A single iteration of the resulting methodology – which we term the *twisted Particle-aGRAD* algorithm – is then exactly as the Particle-aGRAD (Algorithm 6), except that $M'_t(\,\cdot\,|\,\cdot\,;\mathbf{u}_t)$, $G'_t(\,\cdot\,;\mathbf{u}_t)$ and $Q'_t(\,\cdot\,;\mathbf{u}_t)$ from Section 4.1 are replaced by $M'_t(\,\cdot\,|\,\cdot\,;\mathbf{u}_{t:T})$, $G'_t(\,\cdot\,;\mathbf{u}_{t:T})$ and $Q'_t(\,\cdot\,;\mathbf{u}_{t:T})$ from this section. When the potential functions $G_t(\mathbf{x}_{t-1:t})$ vary in $\mathbf{x}_{t-1}$, then we can further construct a *twisted Particle-aGRAD+* algorithm by replacing $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{u}_t)$ in Algorithm 8 and in the denominator of $G'_t(\mathbf{x}_{t-2:t};\mathbf{u}_{t-1:t})$ by $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{u}_{t:T})$.

Proposition 7 (validity of the twisted Particle-aGRAD/Particle-aGRAD+). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via the twisted Particle-aGRAD or twisted Particle-aGRAD+ algorithm induces a Markov kernel which leaves π_T invariant.*

4.5 Relationship with other methods

The algorithms proposed above relate to existing methods as follows.

1. **Generalisation of aGRAD.** For $\kappa = 1$, the Particle-aGRAD algorithm (and similarly the Particle-aGRAD+ algorithm as well as the twisted versions of either) generalises the *auxiliary gradient (aGRAD)* algorithm from Titsias and Papaspiliopoulos (2018, called 'aGrad-z' therein) in the sense that the former reduces to the latter if $T = N = 1$. This can be seen as follows, where we again suppress the 'time' subscript $t = 1$ everywhere so that $\pi(\mathbf{x}) \propto M(\mathbf{x})G(\mathbf{x})$, where $M(\mathbf{x}) = \mathrm{N}(\mathbf{x};\mathbf{m},\mathbf{C})$. Given that the current state of the Markov chain is $\mathbf{x} = \mathbf{x}^0$ (we can assume that $k = 0$ without loss of generality), Step 1c of Algorithm 6 first refreshes the auxiliary variable by sampling $\mathbf{u} \sim \mathrm{N}(\mathbf{x}^0 + \frac{\delta}{2}\nabla\log G(\mathbf{x}^0), \frac{\delta}{2}\mathbf{I})$

and then proposes $\mathbf{x}^1 \sim \mathrm{N}((\mathbf{I}-\mathbf{A})\mathbf{m} + \mathbf{A}\mathbf{u}, \frac{\delta}{2}\mathbf{A})$, for $\mathbf{A} = (\mathbf{C} + \frac{\delta}{2}\mathbf{I})^{-1}\mathbf{C}$. The remaining steps return $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^1$ as the new state with acceptance probability $1 \wedge \alpha_{\mathrm{aGRAD}}(\mathbf{x}^0, \mathbf{x}^1; \mathbf{u})$, where

$$\begin{aligned}\alpha_{\mathrm{aGRAD}}(\mathbf{x}^0, \mathbf{x}^1; \mathbf{u}) &\coloneqq \frac{1 - W^0}{1 - W^1} \\ &= \frac{\pi(\mathbf{x}^1)\,\mathrm{N}(\mathbf{u}; \mathbf{x}^1 + \frac{\delta}{2}\nabla \log G(\mathbf{x}^1), \frac{\delta}{2}\mathbf{I})\,\mathrm{N}(\mathbf{x}^0; (\mathbf{I}-\mathbf{A})\mathbf{m} + \mathbf{A}\mathbf{u}, \frac{\delta}{2}\mathbf{A})}{\pi(\mathbf{x}^0)\,\mathrm{N}(\mathbf{u}; \mathbf{x}^0 + \frac{\delta}{2}\nabla \log G(\mathbf{x}^0), \frac{\delta}{2}\mathbf{I})\,\mathrm{N}(\mathbf{x}^1; (\mathbf{I}-\mathbf{A})\mathbf{m} + \mathbf{A}\mathbf{u}, \frac{\delta}{2}\mathbf{A})}.\end{aligned}$$

Otherwise, the old state $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^0 = \mathbf{x}$ is returned as the new state.

2. **Generalisation of mGRAD.** Still taking $\kappa = 1$, the Particle-mGRAD algorithm generalises the *marginal gradient (mGRAD)* algorithm from Titsias and Papaspiliopoulos (2018) in the sense that the former reduces to the latter if $T = N = 1$. This can be seen as follows, where we use the same notational conventions as in the case of aGRAD above. Step 1c of Algorithm 7 then marginally proposes $\mathbf{x}^1 \sim \mathrm{N}((\mathbf{I}-\mathbf{A})\mathbf{m} + \mathbf{A}[\mathbf{x}^0 + \frac{\delta}{2}\nabla \log G(\mathbf{x}^0)], \mathbf{B})$, where $\mathbf{B} \coloneqq \frac{\delta}{2}\mathbf{A}^2 + \mathbf{A}$. The remaining steps return $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^1$ as the new state with acceptance probability $1 \wedge \alpha_{\mathrm{mGRAD}}(\mathbf{x}^0, \mathbf{x}^1)$, where

$$\alpha_{\mathrm{mGRAD}}(\mathbf{x}^0, \mathbf{x}^1) \coloneqq \frac{1 - W^0}{1 - W^1} = \frac{\pi(\mathbf{x}^1)\,\mathrm{N}((\mathbf{I}-\mathbf{A})\mathbf{m} + \mathbf{A}[\mathbf{x}^1 + \frac{\delta}{2}\nabla \log G(\mathbf{x}^1)], \mathbf{B})}{\pi(\mathbf{x}^0)\,\mathrm{N}((\mathbf{I}-\mathbf{A})\mathbf{m} + \mathbf{A}[\mathbf{x}^0 + \frac{\delta}{2}\nabla \log G(\mathbf{x}^0)], \mathbf{B})}.$$

Otherwise, the old state $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^0 = \mathbf{x}$ is returned as the new state. In particular, by Remark 2, in analogue to Section 3.4, we can again interpret aGRAD as a version of mGRAD with 'randomised' acceptance ratio.

3. **Generalisation of a 'preconditioned' Particle-RWM algorithm.** If $\kappa = 0$, then the Particle-aGRAD and Particle-aGRAD+ algorithms reduce to a method recently proposed in Corenflos and Särkkä (2023, Section 4.3), which can be seen as a 'preconditioned' version of the Particle-RWM algorithm.

4.6 Interpolation between CSMC and Particle-MALA/Particle-aMALA

The Particle-MALA (and related methods) proposed in Section 3 may be outperformed by the CSMC algorithm in the case when the prior dynamics are highly informative – in the same way that MALA may be outperformed by the IMH algorithm (with prior as proposal) if the prior dominates the posterior. For instance, in the extreme case that all the potential functions are constant, the CSMC algorithm proposes N trajectories (in addition to the reference path) that are IID samples from π_T (assuming an adaptive or low-variance conditional resampling scheme is used) while the N trajectories proposed by Particle-MALA are still highly correlated with the reference path.

Put differently, the user is faced with the 'tuning problem' of having to decide between the CSMC algorithm on the one hand and the Particle-MALA (and related methods) on the other hand. In this section, we show that the Particle-mGRAD algorithm resolves this tuning problem in the sense that it can be viewed as interpolating between CSMC and Particle-MALA. Specifically, Proposition 8 shows that Particle-mGRAD reduces to the CSMC algorithm if the prior dynamics are highly informative. Conversely, Proposition 9 shows that Particle-mGRAD reduces to the Particle-MALA if the prior dynamics are uninformative. The same results hold for the auxiliary-variable versions: Particle-aMALA and Particle-aGRAD.

We make the following assumptions (assumed to hold for all $t \in [T]$):

A1 For any $\mathbf{x}_t \in \mathcal{X}$, $\mathbf{m}_t(\mathbf{x}_{t-1}) = \mathbf{m}_t$, $\mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t$ and $G_t(\mathbf{x}_{t-1:t}) = G_t(\mathbf{x}_t)$ are constant in $\mathbf{x}_{t-1}$, with G_t uniformly bounded on $\mathcal{X}$ and $\mathbf{C}_t$ invertible.

A2 There exist $C_0, C_1 \geq 0$ such that $\kappa \|\nabla \log G_t(\mathbf{x}_t)\|_2 \leq C_0 + C_1 \|\mathbf{x}_t\|_2$.

A3 $\max_{d \in [D]} \int_{\mathcal{X}} x_{t,d}^2 G_t(\mathbf{x}_t)\,\mathrm{d}\mathbf{x}_t < \infty$, where $x_{t,d}$ is the dth component of $\mathbf{x}_t$.

Whenever $T > 1$, Assumption **A1** is strong because it requires the Feynman–Kac model to factorise over time. However, we expect that it could be relaxed at the cost of greatly complicating the arguments. Indeed, note that the model used in Figure 2 does not satisfy this assumption. Assumption **A2** is rather mild, e.g. it holds in a state-space model with Gaussian measurement errors.

In the following, for each $t \in [T]$, we will consider a sequence of prior covariance matrices $(\mathbf{C}_{t,k})_{k \geq 1}$. We will therefore add the subscript k to any quantity which depends on $\mathbf{C}_{t,k}$. We also let $\lambda(\mathbf{A})$ denote the set of eigenvalues of some matrix $\mathbf{A}$. The following propositions are proved in Appendix E.

Proposition 8. *For some $D, T, N \geq 1$, assume **A1**–**A2**, and assume that there exists a sequence $(\lambda_k)_{k \geq 1}$ in $(0, \infty)$ with $\max\{\lambda(\mathbf{C}_{1,k}), \ldots, \lambda(\mathbf{C}_{T,k})\} \leq \lambda_k \to 0$ as $k \to \infty$. Then for any $\varepsilon > 0$, there exists a sequence $(F_{T,k})_{k \geq 1}$ of subsets of $\mathcal{X}^T$ with $\lim_{k \to \infty} \pi_{T,k}(F_{T,k}) = 1$ such that*

1. $\sup_{\mathbf{x}_{1:T} \in F_{T,k}} \|P_{\text{Particle-mGRAD},k}(\,\cdot\,|\mathbf{x}_{1:T}) - P_{\text{CSMC},k}(\,\cdot\,|\mathbf{x}_{1:T})\|_{\text{TV}} \in \mathrm{O}(\lambda_k^{(1-\varepsilon)/4})$;
2. $\sup_{\mathbf{x}_{1:T} \in F_{T,k}} \|P_{\text{Particle-aGRAD},k}(\,\cdot\,|\mathbf{x}_{1:T}) - P_{\text{CSMC},k}(\,\cdot\,|\mathbf{x}_{1:T})\|_{\text{TV}} \in \mathrm{O}(\lambda_k^{(1-\varepsilon)/4})$.

Proposition 9. *For some $D, T, N \geq 1$, assume **A1**–**A3**, and assume that there exists a sequence $(\lambda_k)_{k \geq 1}$ in $(0, \infty)$ with $\min\{\lambda(\mathbf{C}_{1,k}), \ldots, \lambda(\mathbf{C}_{T,k})\} \geq \lambda_k \to \infty$ as $k \to \infty$. Then for any $\varepsilon > 0$, there exists a sequence $(F_{T,k})_{k \geq 1}$ of subsets of $\mathcal{X}^T$ with $\lim_{k \to \infty} \pi_{T,k}(F_{T,k}) = 1$ such that*

1. $\sup_{\mathbf{x}_{1:T} \in F_{T,k}} \|P_{\text{Particle-mGRAD},k}(\,\cdot\,|\mathbf{x}_{1:T}) - P_{\text{Particle-MALA},k}(\,\cdot\,|x)\|_{\text{TV}} \in \mathrm{O}(\lambda_k^{-(1-\varepsilon)/4})$;
2. $\sup_{\mathbf{x}_{1:T} \in F_{T,k}} \|P_{\text{Particle-aGRAD},k}(\,\cdot\,|\mathbf{x}_{1:T}) - P_{\text{Particle-aMALA},k}(\,\cdot\,|x)\|_{\text{TV}} \in \mathrm{O}(\lambda_k^{-(1-\varepsilon)/4})$.

As per Sections 2.1.2, 3.4 and 4.5, taking $T = N = 1$ in Propositions 8 and 9 immediately imply that the aGRAD/mGRAD algorithm can be viewed as automatically interpolating between the IMH algorithm with prior as proposal (if the prior is highly informative) and aMALA/MALA (if the prior is highly diffuse). To our knowledge, this interpretation has not been pointed out in the literature. It provides new intuition for the noteworthy performance of aGRAD/mGRAD in Titsias and Papaspiliopoulos (2018).

4.7 Complexity

An iteration of Particle-aGRAD or Particle-aGRAD+ requires computing $T(N+1)$ gain matrices $\mathbf{A}_t(\mathbf{x}_{t-1}^{a_{t-1}^n}) \in \mathbb{R}^{D \times D}$; and all of these, in general, have a cubic cost in the latent-state dimension D. While this may be reasonable for small enough systems and will be helpful for informative likelihoods, the computational quickly outweighs the statistical benefits of the method. However, when the dynamics have additive noise (4.2), $\mathbf{A}_t$ does not depend on $\mathbf{x}_{t-1}$. In this case, only T gain matrices are needed and these can be pre-computed, only paying the cubic cost in the dimension upfront rather than at each iteration.

The same applies for the Particle-mGRAD algorithm for which we always require (4.2) to hold (the auxiliary variables could still be integrated out if (4.2) is relaxed, but only at the cost of a cubic computational complexity in the number of particles).

However, as for the Particle-RWM algorithm and Particle-MALA-type methods, we need to calibrate the step-size parameters δ_t which changes the gain matrices (so that pre-computation is not possible during the calibration stage). Thankfully, because $\mathbf{A}_t$ and $\mathbf{C}_t$ have the same eigenvectors no matter what δ_t is, it is possible to use similar spectral methods as in Titsias and Papaspiliopoulos (2018) to reduce the complexity of changing δ_t to quadratic.

At first sight, the complexity of the twisted Particle-aGRAD seems quadratic in T as the proposal kernel $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t:T}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}'_t\mathbf{x}_{t-1} + \mathbf{b}'_t, \mathbf{C}'_t)$ requires processing $T - t$ auxiliary variables for each time t. However, in Appendix B, we show how $\mathbf{F}'_t$, $\mathbf{b}'_t$ and $\mathbf{C}'_t$ can all be pre-computed based on standard Kalman filter recursions (Kalman, 1960), preserving the linear cost in T and N.

5 Experimental validation and comparison

5.1 Multivariate stochastic volatility model

In this section, we illustrate the efficiency of our methods on a multivariate stochastic volatility model often used as a benchmark for high-dimensional sequential Monte Carlo methodology (see, e.g., Guarniero et al., 2017). This model is a state-space model with a non-linear observation equation:

$$g_t(\mathbf{y}_t|\mathbf{x}_t) = \mathrm{N}(\mathbf{y}_t; \mathbf{0}, \mathrm{diag}(\exp \mathbf{x}_t)),$$

where exp is applied element-wise and where $\mathbf{0}$ is a D-dimensional vector of zeros. The prior on the latent variables is defined through auto-regressive Gaussian dynamics, i.e. for $t > 1$:

$$f_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t) \tag{17}$$

where $\mathbf{m}_t(\mathbf{x}_{t-1}) := \varphi\mathbf{x}_{t-1}$ and $\mathbf{C}_t \in \mathbb{R}^{D\times D}$ has diagonal entries τ and off-diagonal entries $\tau\rho$. The initial distribution $f_1(\mathbf{x}_1) = \mathrm{N}(\mathbf{x}_1; \mathbf{m}_1, \mathbf{C}_1)$ is the stationary distribution under the dynamics (5.1), i.e., $\mathbf{m}_1 := \mathbf{0}$ and $\mathbf{C}_1 := \mathbf{C}_t/(1-\varphi^2)$. Here, $\varphi \in (-1, 1)$ is some autocorrelation coefficient, $\rho \in (-1, 1)$ is some intra-asset correlation coefficient and $\tau > 0$.

Throughout our experiments, we take $\varphi = 0.9$, $\rho = 0.25$, and $\tau \in \{0.1, 0.5, 1, 2\}$. The eigenvalues of $\mathbf{C}_t$ are then proportional to τ, i.e., a small value of τ corresponds to highly informative prior dynamics (as in Proposition 8) while a large value of τ corresponds to weakly informative prior dynamics (as in Proposition 9). To make our observations robust to the choice of data set, for each τ, we simulated $M = 5$ independent sets of $T = 128$ observations from the multivariate stochastic volatility model with $D = 30$, i.e., each state $\mathbf{x}_t$ takes values in $\mathcal{X} = \mathbb{R}^{30}$. To make results more easily comparable, experiments for different values of τ use the same random number generator seed.

5.2 Simulation study setup

In addition to the methods proposed in Sections 3 and 4 – potentially without the use of gradient information by taking $\kappa = 0$ – we consider the following benchmark methods:

1. **CSMC.** The CSMC algorithm with bootstrap proposals (Algorithm 1).
2. **Particle-RWM.** The Particle-RWM algorithm (Algorithm 2) from Finke and Thiery (2023) (the special case of Particle-aMALA/Particle-MALA/Particle-aMALA+ if $\kappa = 0$).

3. **MALA and aMALA.** The N-proposal MALA and aMALA which correspond to the Particle-aMALA and Particle-MALA proposed in this work with a single time step (applied to the path-space representation of the Feynman–Kac model, i.e. with a single $(D \times T)$-dimensional state).

4. **aGRAD.** The N-proposal aGRAD algorithm, which corresponds to the Particle-aGRAD proposed in this work with a single time step (again on the path space). We note that we implemented aGRAD using the auxiliary Kalman perspective of Corenflos and Särkkä (2023), making the method complexity scale linearly with T rather than quadratically with T as in the original version of Titsias and Papaspiliopoulos (2018). We do not compare to mGRAD because computing its particle weights (and hence acceptance ratio) has quadratic complexity in T.

All algorithms use $N + 1 = 32$ particles, and those employing resampling use the conditional 'killing' resampling method (Karppinen et al., 2023), more stable than multinomial resampling, especially with highly informative priors. In each of $M = 5$ independent experiments, algorithms start from the same trajectory generated by a bootstrap particle filter using 32 particles. The samplers run for 10 000 steps to calibrate step-size parameters δ_t, detailed below (note that calibration stabilises much faster). For CSMC, which requires no calibration, the initial 10 000 steps are discarded as warm-up. After calibration, $J = 4$ independent chains start at the final calibration sample, running for $K = 50\,000$ iterations, with the first 5000 discarded as burn-in to decorrelate the chains. Reported statistics are based on these J independent chains.

The step-size parameters δ_t are calibrated for a 75 % acceptance rate, as explained in Appendix F. This slightly exceeds recommendations by, e.g., Roberts and Rosenthal (2001); Titsias and Papaspiliopoulos (2018). This is because we use multiple proposals and the optimal acceptance rate is expected to increase accordingly. Here, 'acceptance rate at time t' refers to the relative frequency of with which the state $\mathbf{x}_t$ is updated. Figure 7 in Appendix G.1 shows stable acceptance rates around 75 % for all methods except CSMC across all time steps. Figure 6 in Appendix G.1 displays calibrated δ_t values.

Experiments ran on a shared computational cluster with identical configurations (32 GB RAM, four processor cores, on shared machines with 2×64-core AMD EPYC 7713 CPUs, clock speed 2.0 GHz). Nonetheless, cluster idiosyncrasies may be present, potentially impacting slower methods like Particle-aMALA+ and Particle-mGRAD.

5.3 Breakdown of CSMC, aMALA and MALA

Our results indicate that CSMC, aMALA, and MALA failed to explore the right regions of the space for all of our chosen levels of informativeness of the latent dynamics ($\tau \in \{0.1, 0.5, 1, 2\}$). Specifically, Figures 8 and 9 in Appendix G.2 show that both the estimated marginal posterior means and also the energy traces of CSMC, aMALA and MALA differ substantially from those of all the other algorithms. Here, 'energy trace' refers to $\log \pi_T(\mathbf{x}_{1:T}) + \text{const}$ computed on the sampled trajectories throughout the sampling procedure. Since CSMC, aMALA and MALA thus do not produce reliable approximations of the distribution of interest, we omit these methods from our discussions in the sequel.

In the remainder of this section, we compare the remaining algorithms in terms of the *effective sample size (ESS)* computed using the method of Vehtari et al. (2021) with $J = 4$ independent chains. We also compare the algorithms in terms of ESS per second (ESS/s). The latter corresponds to the time it would take to obtain a 'perfect' sample using the Markov chain. In the main manuscript, we only show results for the median ESS and averaged over all T time steps. Appendix G.3 shows detailed results for the minimum and maximum ESS and ESS/s (which are qualitatively similar to the median case) separately for each time step $t = 1, \dots, T$.

5.4 Benefits of exploiting gradient information

Figure 3 compares the median ESS ('unnormalised') and median ESS/s ('per second') of Particle-aMALA, Particle-MALA and Particle-aMALA+, i.e., for those methods which do not make any Gaussian assumption about the prior dynamics. Recall that these differ from the baseline: the Particle-RWM algorithm, only in the use of gradient information. Thus, the left panel in Figure 3 illustrates the benefits (in terms of ESS) of exploiting gradient information. Notably:

- the improvement of Particle-MALA over Particle-aMALA is marginal at best. Possibly, the difference between both algorithms decreases with N but this calls for further investigation;
- the 'smoothing-gradient' variant Particle-aMALA+ dominates all other alternatives for all values of τ, with up to three times the performance of Particle-RWM and twice that of the 'filter-gradient' variants Particle-aMALA and Particle-MALA;
- the performance of all shown methods improves as τ increases: this is because the posterior distribution then decorrelates in time, and, therefore, the fact that they all use proposals which are separable (in the sense discussed in Section 4) stops being penalising.

The right panel in Figure 3 shows that the use of gradient information is still beneficial even when accounting for the cost of gradient calculation. However, the relative performance of the gradient-based methods is now less clear: whilst Particle-aMALA+ has the highest sampling efficiency, it incurs additional overheads due to computing twice as many gradients as Particle-aMALA and Particle-MALA and due to dealing with non-Markovian potentials.

5.5 Benefits of exploiting Gaussian prior dynamics

In this section, we demonstrate that exploiting the latent (conditionally) Gaussian dynamics of the model (as done by Particle-aGRAD, Particle-mGRAD and twisted Particle-aGRAD) can improve the sampling efficiency.

First, in Figure 4, we illustrate the performance of those methods which require (at most) *conditionally* Gaussian prior dynamics as in (4.1), i.e., of Particle-aGRAD and Particle-mGRAD (note that the later also requires $\mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t$ (4.2) to retain linear computational complexity in N). In terms of ESS, these methods improve upon the 'filter-gradient' methods Particle-aMALA and Particle-MALA but they are still dominated by the 'smoothing-gradient' method Particle-aMALA+. However, the picture is less clear when accounting for computation time.

Second, in Figure 5, we illustrate the performance of the twisted Particle-aGRAD which requires *unconditionally* Gaussian prior dynamics as in (4.4). As a baseline, we use the aGRAD algorithm from Titsias and Papaspiliopoulos (2018) as it makes the same assumption. The twisted Particle-aGRAD strongly outperforms this baseline and also all the other algorithms. Furthermore, the dominance of the twisted Particle-aGRAD algorithm does not disappear when accounting for the computation time. This is because, in contrast to Particle-aMALA+, its modified model is still Markovian and because it only requires the computation of a single gradient per particle and time step.

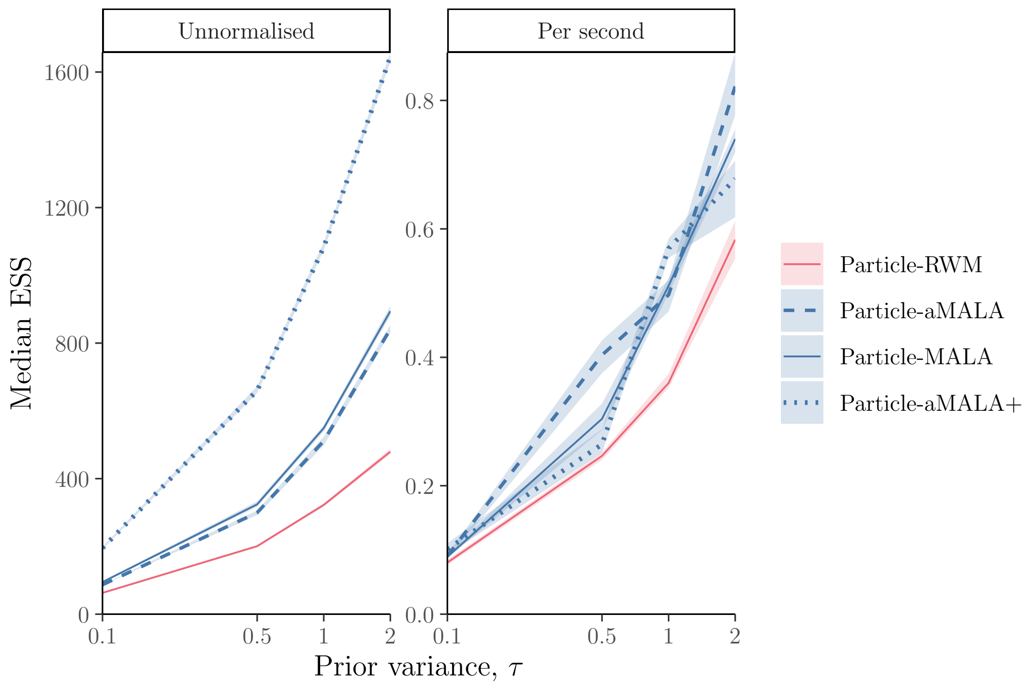


Figure 3: Performance of those proposed methods which do not require (conditionally or unconditionally) Gaussian prior dynamics compared with the existing Particle-RWM algorithm as a baseline.

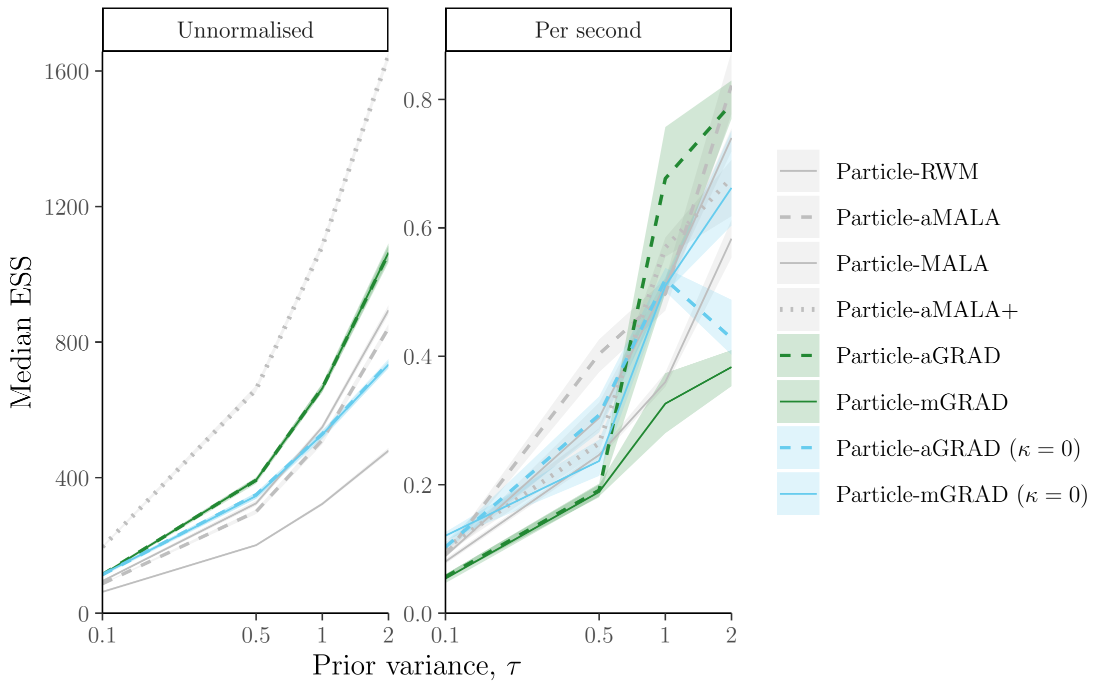


Figure 4: Performance of the proposed methods which require only *conditionally* Gaussian prior dynamics (4.1), i.e., $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))$. The Particle-mGRAD algorithm (with any $\kappa \in \{0, 1\}$) also requires that $\mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t$ is constant to avoid superlinear computational complexity in N. Results that were already shown in the previous figure are greyed out.

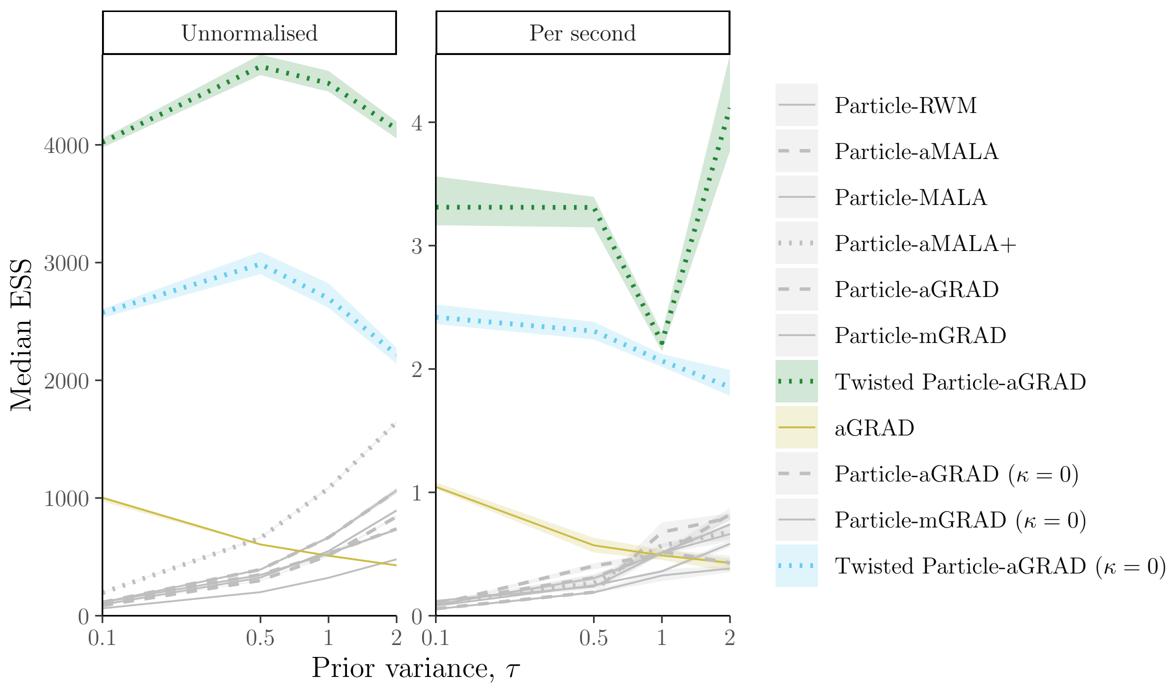


Figure 5: Performance of the proposed methods which require *unconditionally* Gaussian prior dynamics (4.4), i.e., $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}_t\mathbf{x}_{t-1} + \mathbf{b}_t, \mathbf{C}_t)$, compared with aGRAD (which also requires (4.4)) as baseline. Results that were already shown in the previous two figures are greyed out. The abnormally large computation time of the twisted Particle-aGRAD for $\kappa = \tau = 1$ was likely caused by some computational-cluster idiosyncrasies.

6 Conclusion

6.1 Summary

We have proposed a methodology for Bayesian inference about the latent states in high-dimensional state-space models and beyond. Our methodology combines the CSMC algorithm (Andrieu et al., 2010) with sophisticated 'classical' MCMC algorithms like MALA (Besag, 1994), aMALA (Titsias and Papaspiliopoulos, 2018), aGRAD/mGRAD (Titsias, 2011; Titsias and Papaspiliopoulos, 2018) or PCNL (Cotter et al., 2013) to retain *the best of both worlds:*

- from the CSMC algorithm, our methods retain the ability to exploit the model's 'decorrelation-over-time' structure which permits favourable scaling with the number of time steps, T;
- from 'classical' MCMC algorithms, our methods retain the ability to use gradient-informed, local proposals which permits favourable scaling with the dimension of the states, D.

Most of our proposed algorithms (except the 'marginal' ones) leverage an auxiliary-variable perspective recently proposed in Corenflos and Särkkä (2023). We name our algorithms Particle-aMALA, Particle-MALA, Particle-aGRAD, Particle-mGRAD and Particle-PCNL. This is motivated by the fact that if $T = N = 1$ (where $N \in \mathbb{N}$ is the number of particles), they reduce to the 'classical' MCMC algorithms: aMALA, MALA, PCNL, aGRAD and mGRAD, respectively. Furthermore, if $T = 1$ but $N > 1$, our methods constitute novel multi-proposal versions of such 'classical' MCMC algorithms which may themselves be of interest with a view to exploiting parallelisation.

The generalisation of such 'classical' MCMC algorithms to $T > 1$ time steps is, however, not unique. And so we have presented additional variants named Particle-aMALA+ and Particle-aGRAD+ and *twisted* Particle-aGRAD/Particle-aGRAD+. These can be viewed as 'lookahead' methods because their proposals employ 'smoothing' rather than 'filter' gradients or utilise information contained in future auxiliary variables. Notably, if $N = 1$ but $T > 1$, then the IMH, RWM, aMALA and aGRAD algorithm can still be recovered as a special case of slightly modified versions of the CSMC, Particle-RWM, Particle-aMALA+, and twisted Particle-aGRAD+ algorithms (and also of the twisted Particle-aGRAD algorithm if $G_t(\mathbf{x}_{t-1:t})$ is constant in $\mathbf{x}_{t-1}$). Specifically, this modification would entail that the latter use no resampling (i.e., they instead set $a_t^n = n$ for all $n \in [N]_0$ and all $t \in [T-1]$), use *ancestral tracing* instead of backward sampling (i.e., they instead set $l_t = a_t^{l_{t+1}}$ for all $t \in [T-1]$) and use $\delta_1 = \ldots = \delta_T$.

We have further proved that the Particle-aGRAD/Particle-mGRAD algorithms have the desirable property that they naturally recover (a) the CSMC algorithm if the prior dynamics are highly informative (i.e., if the target posterior distribution is dominated by the prior); (b) the Particle-aMALA/Particle-MALA if the prior dynamics are completely uninformative (i.e., if the target posterior distribution is dominated by the likelihood). This property independently helps explain the impressive performance of aGRAD and mGRAD reported in Titsias and Papaspiliopoulos (2018).

Our methods have enabled Bayesian inference in a multivariate stochastic volatility model with $D = 30$ assets and $T = 128$ observations (3840 unknowns in total) in which neither CSMC nor aMALA/MALA gave reliable estimates. In particular, in this application, our twisted Particle-aGRAD algorithm strongly outperformed the existing sophisticated aGRAD algorithm – even when accounting for computation time.

6.2 Limitations

The main limitations of our methods are the same as in all gradient-based 'classical' MCMC algorithms. First, they require continuously differentiable target densities (more precisely, the densities $Q_t(\mathbf{x}_{t-1:t})$ need to be computable and differentiable pointwise). This requirement is slightly softened for the methods of Section 4 where only the likelihood $G_t(\mathbf{x}_{t-1:t})$ is required to be differentiable, at the cost of needing (at least conditionally) Gaussian prior dynamics $M_t(\mathbf{x}_t|\mathbf{x}_{t-1})$. The favourable scaling with the dimension D also typically requires target densities to be sufficiently smooth (see, e.g., Vogrinc and Kendall, 2021, for counterexamples). Second, while it improves mixing properties, locality in MCMC is often detrimental when exploring multi-modal posteriors. This is inherited by our methods which, too, explore the space by local moves.

6.3 Extensions

Our work opens up multiple avenues for further research.

- The algorithms proposed in this work can be extended to more general graphical models, i.e., they can be combined with suitably 'conditional' versions of the divide-&-conquer sequential Monte Carlo algorithm from Lindsten et al. (2017). For instance, for a particular graphical model, such a 'conditional' scheme was recently described in Corenflos et al. (2022, Section 3).

- Particle-aMALA can be incorporated straightforwardly into the methodology from Corenflos et al. (2022) to reduce the computation time per MCMC update from $\mathrm{O}(T)$ to

$\mathrm{O}(\log_2 T)$ (for some fixed dimension D) on parallel architectures. While less directly obvious (because of the non-Markovianity of the auxiliary target), the smoothing-gradient version Particle-aMALA+ is likely parallelisable, too, by simply extending the framework to compute weight functions over three time steps rather than two. It is however less clear that Particle-MALA is parallelisable, as the marginalisation has to be done across two time steps rather than one as presented in Section 3.2.

- All our algorithms can be straightforwardly extended to use other resampling schemes than conditional multinomial resampling, e.g., conditional systematic resampling. In fact, in our experiments, we used the conditional killing resampling which is stable under low-informative likelihoods (Karppinen et al., 2023), a regime that may happen in our case when δ_t takes very small values at calibration time.
- In this work, we have left aside the question of choosing δ_t and have elected to take it to correspond to a 75 % acceptance rate throughout. It is however clear that its optimal value (and the optimal value of the acceptance rate) depends on the number of proposals N and on the dimension D. An optimal-scaling analysis (see Roberts and Rosenthal, 2001, and references therein) of the methods proposed in this work is therefore needed. An optimal-scaling analysis for a related algorithm without backward sampling and without gradient or prior-informed proposals can be found in Malory (2021).
- In Section 4 (in which we propose Particle-aGRAD and variations thereof), we have assumed that the covariance matrices $\mathbf{C}_t(\mathbf{x}_{t-1})$ (or $\mathbf{C}_t$) are non-singular. However, it is worth noting that proposal kernels used by the methods from Section 4 remain valid if the covariance matrices are singular, in the sense that they are *still* absolutely continuous w.r.t. the true dynamics. However, the use of backward sampling is no longer possible for such degenerate dynamics. Instead, one must resort to ancestral tracing, i.e., taking $l_t \coloneqq a_t^{l_{t+1}}$, for $t = T-1, \dots, 1$. However, in this case, N needs to grow with T at a suitable rate which depends on the stability properties of the model, but at least linearly (Andrieu et al., 2018; Lindsten et al., 2015). An alternative is to fix N but decrease the step sizes δ_t with t (which would automatically occur when using adaptation based on acceptance rates as considered in work), as considered in Malory (2021) for a related method.
- Our proposed algorithms consider solely first-order gradient information. A natural extension would therefore be to incorporate second-order expansions or preconditioned and adaptive versions of the Particle-MALA variants. Another obvious direction of study is to extend our methodology to other MCMC kernels, such as the recently proposed Barker's robust proposal (Livingstone and Zanella, 2022), or non-reversible discrete-time kernels such as the discrete bouncy particle sampler Sherlock and Thiery (2022). Other natural extensions would consist of adapting the methodology to non-continuous spaces, e.g., using methods from Zanella (2020); Rhodes and Gutmann (2022), or constrained spaces.

Author contributions

A.C. and A.F. jointly developed the methodology, writing was primarily done by A.F., A.C. implemented and conducted the experiments, after which both A.C. and A.F. edited and reviewed the final manuscript.

References

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342. With discussion.

Andrieu, C., Lee, A., and Vihola, M. (2018). Uniform ergodicity of the iterated conditional SMC and geometric ergodicity of particle Gibbs samplers. *Bernoulli*, 24(2):842–872.

Andrieu, C. and Vihola, M. (2016). Establishing some order amongst exact approximations of MCMCs. *Annals of Applied Probability*, 26(5):2661–2696.

Besag, J. E. (1994). Contribution to the discussion on 'Representations of knowledge in complex systems' by Grenander, U and Miller, M. I.. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 56(4):549–581.

Ceperley, D. M. and Dewing, M. (1999). The penalty method for random walks with uncertain energies. *The Journal of Chemical Physics*, 110(20):9812–9820.

Chopin, N. and Singh, S. S. (2013). On particle Gibbs sampling. *arXiv e-prints*, arXiv:1304.1887v1.

Corenflos, A., Chopin, N., and Särkkä, S. (2022). De-sequentialized Monte Carlo: A parallel-in-time particle smoother. *Journal of Machine Learning Research*, 23(283):1–39.

Corenflos, A. and Särkkä, S. (2023). Auxiliary MCMC and particle Gibbs samplers for parallelisable inference in latent dynamical systems. *arXiv preprint arXiv:2303.00301*.

Cotter, S. L., Roberts, G. O., Stuart, A. M., and White, D. (2013). MCMC methods for functions: Modifying old algorithms to make them faster. *Statistical Science*, 28(3):424–446.

Fearnhead, P. and Meligkotsidou, L. (2016). Augmentation schemes for particle MCMC. *Statistics and Computing*, 26:1293–1306.

Finke, A. (2015). *On Extended State-Space Constructions for Monte Carlo Methods*. PhD thesis, Department of Statistics, University of Warwick, UK.

Finke, A., Doucet, A., and Johansen, A. M. (2016). On embedded hidden Markov models and particle Markov chain Monte Carlo methods. *arXiv e-prints*, arXiv:1610.08962.

Finke, A. and Thiery, A. H. (2023). Conditional sequential Monte Carlo in high dimensions. *The Annals of Statistics*, 51(2):437–463.

Guarniero, P., Johansen, A. M., and Lee, A. (2017). The iterated auxiliary particle filter. *Journal of the American Statistical Association*, 112(520):1636–1647.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Henderson, H. V. and Searle, S. R. (1981). On deriving the inverse of a sum of matrices. *SIAM Review*, 23(1):53–60.

Heng, J., Bishop, A. N., Deligiannidis, G., and Doucet, A. (2020). Controlled sequential Monte Carlo. *The Annals of Statistics*, 48(5):2904 – 2929.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45.

Karjalainen, J., Lee, A., Singh, S. S., and Vihola, M. (2023). Mixing time of the conditional backward sampling particle filter. *arXiv e-prints*, arXiv:2312.17572.

Karppinen, S., Singh, S. S., and Vihola, M. (2023). Conditional particle filters with bridge backward sampling. *Journal of Computational and Graphical Statistics*, 0(0):1–15.

Karppinen, S. and Vihola, M. (2021). Conditional particle filters with diffuse initial distributions. *Statistics and Computing*, 31:1–14.

Lee, A., Singh, S. S., and Vihola, M. (2020). Coupled conditional backward sampling particle filter. *Annals of Statistics*, 48(5):3066–3089.

Lindsten, F., Douc, R., and Moulines, E. (2015). Uniform ergodicity of the particle Gibbs sampler. *Scandinavian Journal of Statistics*, 42(3):775–797.

Lindsten, F., Johansen, A. M., Naesseth, C. A., Kirkpatrick, B., Schön, T. B., Aston, J. A., and Bouchard-Côté, A. (2017). Divide-and-conquer with sequential Monte Carlo. *Journal of Computational and Graphical Statistics*, 26(2):445–458.

Lindsten, F., Jordan, M. I., and Schön, T. B. (2012). Ancestor sampling for particle Gibbs. In *Proceedings of the 2012 Conference on Neural Information Processing Systems*, Lake Tahoe, NV.

Liu, J. S. (1996). Peskun's theorem and a modified discrete-state Gibbs sampler. *Biometrika*, 83(3):681–682.

Livingstone, S. and Zanella, G. (2022). The Barker proposal: Combining robustness and efficiency in gradient-based MCMC. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(2):496–523.

Malory, S. (2021). *Bayesian inference for stochastic processes.* PhD thesis, Lancaster University.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092.

Murray, L. M., Jones, E. M., and Parslow, J. (2013). On disturbance state-space models and the particle marginal Metropolis–Hastings sampler. *SIAM/ASA Journal on Uncertainty Quantification*, 1(1):494–521.

Nicholls, G. K., Fox, C., and Muir Watt, A. (2012). Coupled MCMC with a randomized acceptance probability. *arXiv e-prints*, arXiv:1205.6857.

Rhodes, B. and Gutmann, M. (2022). Enhanced gradient-based MCMC in discrete spaces. *arXiv e-prints*, arXiv:2208.00040.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability*, 7(1):110–120.

Roberts, G. O. and Rosenthal, J. S. (1998). Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1):255–268.

Roberts, G. O. and Rosenthal, J. S. (2001). Optimal scaling for various Metropolis–Hastings algorithms. *Statistical Science*, 16(4):351–367.

Särkkä, S. and Svensson, L. (2023). *Bayesian filtering and smoothing*, volume 17. Cambridge University Press.

Sherlock, C. and Thiery, A. H. (2022). A discrete bouncy particle sampler. *Biometrika*, 109(2):335–349.

Shestopaloff, A. Y. and Neal, R. M. (2018). Sampling latent states for high-dimensional non-linear state space models with the embedded HMM method. *Bayesian Analysis*, 13(3):797–822.

Singh, S. S., Lindsten, F., and Moulines, E. (2017). Blocking strategies and stability of particle Gibbs samplers. *Biometrika*, 104(4):953–969.

Titsias, M. K. (2011). Contribution to the discussion on 'Riemann manifold Langevin and Hamiltonian Monte Carlo methods' by Girolami, M., and Calderhead, b. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 73(2):123–214.

Titsias, M. K. and Papaspiliopoulos, O. (2018). Auxiliary gradient-based sampling algorithms. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 80(4):749–767.

Tjelmeland, H. (2004). Using all Metropolis–Hastings proposals to estimate mean values. preprint 4/2004, Norwegian University of Science and Technology, Trondheim, Norway.

Vehtari, A., Gelman, A., Simpson, D., Carpenter, B., and Bürkner, P.-C. (2021). Rank-normalization, folding, and localization: An improved $\widehat{R}$ for assessing convergence of MCMC (with discussion). *Bayesian Analysis*, 16(2):667–718.

Vogrinc, J. and Kendall, W. S. (2021). Counterexamples for optimal scaling of Metropolis–Hastings chains with rough target densities. *The Annals of Applied Probability*, 31(2):972–1019.

Whiteley, N. (2010). Contribution to the discussion on 'Particle Markov chain Monte Carlo methods' by Andrieu, C., Doucet, A., and Holenstein, R. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):306–307.

Whiteley, N. and Lee, A. (2014). Twisted particle filters. *The Annals of Statistics*, 42(1):115–141.

Zanella, G. (2020). Informed proposals for local MCMC in discrete spaces. *Journal of the American Statistical Association*, 115(530):852–865.

A Particle extensions of PCN(L)

A.1 Particle-aPCNL

In this section, we extend the *preconditioned Crank–Nicolson–Langevin (PCNL)* algorithm (and also the *preconditioned Crank–Nicolson (PCN)* algorithm recovered by setting $\kappa = 0$) (Cotter et al., 2013) to $T > 1$ time steps and $N > 1$. As a by-product, we derive an 'auxiliary-variable' version of PCNL which was mentioned, but not explicitly stated, in Titsias and Papaspiliopoulos (2018). Throughout this section, we assume the prior dynamics are conditionally Gaussian, $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))$ as in (4.1).

Throughout this section, we use the parametrisation of the PCNL algorithm from Titsias and Papaspiliopoulos (2018)[3], i.e., we set

$$\beta_t \coloneqq \frac{2}{2+\delta_t} \in (0,1),$$

where $\delta_t > 0$ is again the step size at time t. Note that this implies that $\frac{1-\beta_t}{\beta_t} = \frac{\delta_t}{2}$.

The first method proposed in this section is termed *Particle-aPCNL*. Conditional on the auxiliary variables $\mathbf{u}_{1:T}$, it can be viewed as a CSMC algorithm whose proposal kernels are those of the fully-adapted auxiliary particle filter for the state-space model defined by the Gaussian transitions $p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t(\mathbf{x}_{t-1}), \mathbf{C}_t(\mathbf{x}_{t-1}))$ from (4.1) and 'pseudo observations' $\mathbf{u}_t$ with $p(\mathbf{u}_t|\mathbf{x}_t) = \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \frac{\delta_t}{2}\mathbf{C}_t(\mathbf{x}_{t-1}))$. We now write

$$\begin{aligned} M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t) &\coloneqq p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t) \\ &\propto M_t(\mathbf{x}_t|\mathbf{x}_{t-1})\,\mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \tfrac{\delta_t}{2}\mathbf{C}_t(\mathbf{x}_{t-1})) \\ &\propto \mathrm{N}(\mathbf{x}_t; \mathbf{m}'_t(\mathbf{x}_{t-1}, \mathbf{u}_t), \mathbf{C}'_t(\mathbf{x}_{t-1})), \end{aligned} \tag{18}$$

with

$$\mathbf{m}'_t(\mathbf{x}_{t-1}, \mathbf{u}_t) \coloneqq \beta_t \mathbf{u}_t + (1-\beta_t)\mathbf{m}_t(\mathbf{x}_{t-1}), \tag{19}$$

$$\mathbf{C}'_t(\mathbf{x}_{t-1}) \coloneqq (1-\beta_t)\mathbf{C}_t(\mathbf{x}_{t-1}), \tag{20}$$

as well as

$$G'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_t) \coloneqq Q_t(\mathbf{x}_{t-1:t}) \frac{\mathrm{N}(\mathbf{u}_t; \mathbf{x}_t + \kappa\frac{\delta_t}{2}\widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1})\nabla_{\mathbf{x}_t}\log G_t(\mathbf{x}_{t-1:t}), \frac{\delta_t}{2}\mathbf{C}_t(\mathbf{x}_{t-1}))}{M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t)}, \tag{21}$$

and $Q'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_t) \coloneqq M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t)G'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_t)$. Here, $\widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1}) \in \mathbb{R}^{D\times D}$ is some preconditioning matrix whose choice is discussed in Section A.5 below.

A single iteration of the Particle-aPCNL algorithm is then as follows.

Algorithm 9 (Particle-aPCNL). Implement Algorithm 1 but replace the particle proposal (Step 1c) and the weight calculation (Step 1d) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \kappa\frac{\delta_t}{2}\widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1})\nabla_{\mathbf{x}_t}\log G_t(\mathbf{x}_{t-1:t}), \frac{\delta_t}{2}\mathbf{C}_t(\mathbf{x}_{t-1}))$, and $\mathbf{x}_t^n \sim M'_t(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n}; \mathbf{u}_t)$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. for $n \in [N]_0$, set $w_t^n \propto G'_t(\mathbf{x}_{t-1:t}^{(n)}; \mathbf{u}_t)$,

and also replace $Q_{t+1}(\,\cdot\,)$ in the backward kernel in Step 3 by $Q'_{t+1}(\,\cdot\,; \mathbf{u}_t)$.

Proposition 10 (validity of Particle-aPCNL). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 9 induces a Markov kernel $P_{\text{Particle-aPCNL}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

[3] In the parametrisation from Cotter et al. (2013), we would have $\delta_t \in [0,2]$ $\beta_t = \frac{2-\delta_t}{2+\delta_t}$

A.2 Particle-PCNL

In this section, in analogy to the Particle-MALA and Particle-mGRAD algorithms from the main manuscript, we analytically integrate out the auxiliary variables $\mathbf{u}_t$ which appeared in the weights of the Particle-aPCNL algorithm. As in the case of the Particle-mGRAD algorithm, we assume that the covariance matrices appearing in the conditionally Gaussian mutation kernel (4.1) do not depend on the previous state, i.e., $\mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t$ (4.2).

A single iteration of the resulting methodology – which we term the *Particle-PCNL* algorithm – is as follows, where we write

$$\begin{aligned}\log H_{t,\boldsymbol{\phi}}(\mathbf{x}, \mathbf{v}, \bar{\mathbf{x}}, \bar{\mathbf{v}}) = {} & \tfrac{1}{2}(\beta_t^{-1} + N + 1)(\mathbf{x} - \mathbf{v})^{\mathrm{T}} \mathbf{G}_t (\mathbf{x} - \mathbf{v}) \\ & - \tfrac{1}{2} N \beta_t (\mathbf{x} + \boldsymbol{\phi})^{\mathrm{T}} \mathbf{G}_t (\mathbf{x} + \boldsymbol{\phi}) \\ & + (N+1)(\bar{\mathbf{x}} - \bar{\mathbf{v}})^{\mathrm{T}} \mathbf{G}_t (\mathbf{v} + \boldsymbol{\phi}) \\ & - (\mathbf{x} - \mathbf{v})^{\mathrm{T}} \mathbf{G}_t (\mathbf{x} + \boldsymbol{\phi}),\end{aligned}$$

for

$$\mathbf{G}_t \coloneqq \frac{\beta_t}{(1-\beta_t)(1+N\beta_t)} \mathbf{C}_t^{-1} = \frac{2(\delta_t + 2)}{\delta_t(\delta_t + 2 + N)} \mathbf{C}_t^{-1}. \tag{22}$$

Algorithm 10 (Particle-PCNL). Implement Algorithm 1 but replace the particle proposal (Step 1c) and the weight calculation (Step 1d) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \kappa \frac{\delta_t}{2} \widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1}) \nabla_{\mathbf{x}_t} \log G_t(\mathbf{x}_{t-1:t}), \frac{\delta_t}{2} \mathbf{C}_t)$, and $\mathbf{x}_t^n \sim M_t'(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n}; \mathbf{u}_t)$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. set $\bar{\mathbf{x}}_t \coloneqq \frac{1}{N+1} \sum_{n=0}^N \mathbf{x}_t^n$, $\mathbf{v}_t^n \coloneqq (1-\beta_t) \mathbf{m}_t(\mathbf{x}_{t-1}^{a_{t-1}^n})$, $\bar{\mathbf{v}}_t \coloneqq \frac{1}{N+1} \sum_{n=0}^N \mathbf{v}_t^n$, and, for $n \in [N]_0$,

$$w_t^n \propto Q_t(\mathbf{x}_{t-1:t}^{(n)}) H_{t, \kappa \frac{\delta_t}{2} \widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1}^{a_{t-1}^n}) \nabla_{\mathbf{x}_t^n} \log G_t(\mathbf{x}_{t-1:t}^{(n)})}(\mathbf{x}_t^n, \mathbf{v}_t^n, \bar{\mathbf{x}}_t, \bar{\mathbf{v}}_t).$$

In the same way as outlined in Remarks 1 and 2, the Particle-aPCNL algorithm can be viewed as an 'exact approximation' of the Particle-PCNL algorithm.

Proposition 11 (validity of Particle-PCNL). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 10 induces a Markov kernel $P_{\text{Particle-PCNL}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

A.3 Particle-aPCNL+

In this section, similar to the Particle-aMALA+ and Particle-aGRAD+ algorithms from the main manuscript, we extend the Particle-aPCNL algorithm to incorporate gradients w.r.t. the 'smoothing' potential $G_{1:T}(\mathbf{x}_{1:T}) = \prod_{t=1}^T G_t(\mathbf{x}_{t-1:t})$ rather than w.r.t. the 'filtering' potential $\prod_{s=1}^t G_s(\mathbf{x}_{s-1:s})$.

For $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t)$ and $G_t'(\mathbf{x}_{t-1:t}, \mathbf{u}_t)$ still defined as in the Particle-aPCNL algorithm (i.e., as in (A.1) and (A.1)), we now write

$$\begin{aligned}& G_t'(\mathbf{x}_{t-2:t}, \mathbf{u}_{1:T}) \\ & \coloneqq G_t'(\mathbf{x}_{t-1:t}, \mathbf{u}_t) \frac{\mathrm{N}(\mathbf{u}_{t-1}; \mathbf{x}_{t-1} + \kappa \frac{\delta_{t-1}}{2} \widetilde{\mathbf{C}}_{t-1}(\mathbf{x}_{t-2}) \nabla_{\mathbf{x}_{t-1}} \log G_{1:T}(\mathbf{x}_{1:T}), \frac{\delta_{t-1}}{2} \mathbf{C}_{t-1}(\mathbf{x}_{t-2}))}{\mathrm{N}(\mathbf{u}_{t-1}; \mathbf{x}_{t-1} + \kappa \frac{\delta_{t-1}}{2} \widetilde{\mathbf{C}}_{t-1}(\mathbf{x}_{t-2}) \nabla_{\mathbf{x}_{t-1}} \log G_{t-1}(\mathbf{x}_{t-2:t-1}), \frac{\delta_{t-1}}{2} \mathbf{C}_{t-1}(\mathbf{x}_{t-2}))},\end{aligned}$$

as well as $Q'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{t-1:t}) \coloneqq M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t) G'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{t-1:t})$, where we note that

$$\nabla_{\mathbf{x}_t} \log G_{1:T}(\mathbf{x}_{1:T}) = \nabla_{\mathbf{x}_t}[\log G_t(\mathbf{x}_{t-1:t}) + \log G_{t+1}(\mathbf{x}_{t:t+1})].$$

A single iteration of the resulting methodology – which we term the *Particle-aPCNL+* algorithm – is as follows.

Algorithm 11 (Particle-aPCNL+). Implement Algorithm 1 but replace the particle proposal (Step 1c), the weight calculation (Step 1d), and backward sampling (Step 3) by

1c. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \kappa \frac{\delta_t}{2} \widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1}) \nabla_{\mathbf{x}_t} \log G_T(\mathbf{x}_{1:T}), \frac{\delta_t}{2} \mathbf{C}_t(\mathbf{x}_{t-1}))$, and $\mathbf{x}_t^n \sim M'_t(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n}; \mathbf{u}_t)$, for $n \in [N]_0 \setminus \{k_t\}$,

1d. for $n \in [N]_0$, set $w_t^n \propto G'_t(\mathbf{x}_{t-2:t}^{(n)}; \mathbf{u}_{t-1:t})$,

3. for $t = T-1, \dots, 1$, sample $l_t = i \in [N]_0$ w.p.

$$\frac{W_t^i Q'_{t+1}((\mathbf{x}_{t-1:t}^{(i)}, \mathbf{x}_{t+1}^{l_{t+1}}); \mathbf{u}_{t:t+1}) Q'_{t+2}((\mathbf{x}_t^i, \mathbf{x}_{t+1}^{l_{t+1}}, \mathbf{x}_{t+2}^{l_{t+2}}); \mathbf{u}_{t+1:t+2})}{\sum_{n=0}^N W_t^n Q'_{t+1}((\mathbf{x}_{t-1:t}^{(n)}, \mathbf{x}_{t+1}^{l_{t+1}}); \mathbf{u}_{t:t+1}) Q'_{t+2}((\mathbf{x}_t^n, \mathbf{x}_{t+1}^{l_{t+1}}, \mathbf{x}_{t+2}^{l_{t+2}}); \mathbf{u}_{t+1:t+2})}.$$

Note that if $G_t(\mathbf{x}_{t-1:t}) = G_t(\mathbf{x}_t)$ does not depend on $\mathbf{x}_{t-1}$, then the Particle-aPCNL+ algorithm coincides with the Particle-aPCNL algorithm. However, when $G_t(\mathbf{x}_{t-1:t})$ varies highly in $\mathbf{x}_{t-1}$, their behaviours may differ substantially.

Proposition 12 (validity of Particle-aPCNL+). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 11 induces a Markov kernel $P_{\text{Particle-aPCNL+}}(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

A.4 Twisted Particle-aPCNL(+)

In analogue to the twisted Particle-aGRAD and twisted Particle-aGRAD+ algorithms, we can again construct 'twisted' versions of the Particle-aPCNL and Particle-aPCNL+ algorithms, under the assumption that $M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}_t\mathbf{x}_{t-1} + \mathbf{b}_t, \mathbf{C}_t)$, i.e., (4.4).

We start with the twisted Particle-aPCNL algorithm. We now write

$$\begin{aligned} M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{t:T}) &\coloneqq p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t:T}) \\ &\propto \int_{\mathcal{X}^{T-t}} \left[\prod_{s=t}^T \mathrm{N}(\mathbf{x}_s; \mathbf{F}_s\mathbf{x}_{s-1} + \mathbf{b}_s, \mathbf{C}_s)\, \mathrm{N}(\mathbf{u}_s; \mathbf{x}_s, \tfrac{\delta_s}{2}\mathbf{C}_t)\right] \mathrm{d}\mathbf{x}_{t+1:T} \\ &\propto \mathrm{N}(\mathbf{x}_t; \mathbf{F}'_t\mathbf{x}_{t-1} + \mathbf{b}'_t, \mathbf{C}'_t), \qquad (23) \\ G'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_{t:T}) &\coloneqq Q_t(\mathbf{x}_{t-1:t}) \frac{\mathrm{N}(\mathbf{u}_t; \mathbf{x}_t + \kappa\frac{\delta_t}{2}\widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1})\nabla_{\mathbf{x}_t}\log G_t(\mathbf{x}_{t-1:t}), \frac{\delta_t}{2}\mathbf{C}_t)}{M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t:T})}, \end{aligned}$$

as well as $Q'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_{t:T}) \coloneqq M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{t:T}) G'_t(\mathbf{x}_{t-1:t}; \mathbf{u}_{t:T})$. Here, $\mathbf{b}'_t \in \mathbb{R}^D$ and $\mathbf{F}'_t, \mathbf{C}'_t \in \mathbb{R}^{D\times D}$ can again be obtained via the Kalman-filtering recursions given in Appendix B.

A single iteration of the resulting methodology – which we term the *twisted Particle-aPCNL* algorithm – is then exactly as the Particle-aPCNL algorithm (Algorithm 9), except that $M'_t(\,\cdot\,|\,\cdot\,; \mathbf{u}_t)$, $G'_t(\,\cdot\,; \mathbf{u}_t)$ and $Q'_t(\,\cdot\,; \mathbf{u}_t)$ from Section A.1 are replaced by $M'_t(\,\cdot\,|\,\cdot\,; \mathbf{u}_{t:T})$, $G'_t(\,\cdot\,; \mathbf{u}_{t:T})$ and $Q'_t(\,\cdot\,; \mathbf{u}_{t:T})$ from this section. When the potential functions $G_t(\mathbf{x}_{t-1:t})$ vary in $\mathbf{x}_{t-1}$, then we can further construct a *twisted Particle-aGRAD+* algorithm by replacing $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t)$ in Algorithm 11 and in the denominator of $G'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{t-1:t})$ by $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t:T})$.

Proposition 13 (validity of the twisted Particle-aPCNL/Particle-aPCNL+). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via the twisted Particle-aPCNL or twisted Particle-aPCNL+ algorithm induces a Markov kernel which leaves π_T invariant.*

A.5 Choice of preconditioning matrix

There is some degree of freedom in choosing the preconditioning matrices $\widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1}) \in \mathbb{R}^{D \times D}$ in the algorithms presented above.

1. A simple option which does not require further assumptions is to take

$$\widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1}) \coloneqq \mathbf{C}_t(\mathbf{x}_{t-1}).$$

2. If we make the stronger model assumption that $\mathbf{m}_t(\mathbf{x}_{t-1}) = \mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{b}_t$ and $\mathbf{C}_t(\mathbf{x}_{t-1}) = \mathbf{C}_t$ (4.4) (which is assumed to hold for the twisted versions of the Particle-aPCNL and Particle-aPCNL+ algorithms anyway), then we could alternatively set

$$\widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1}) = \widetilde{\mathbf{C}}_t \coloneqq \sum_{s=1}^{T} \mathbf{\Sigma}_{s,t}, \tag{24}$$

where $\mathbf{\Sigma}_{s,t} \in \mathbb{R}^{D \times D}$ is the block (s,t) in the covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{TD \times TD}$ of the prior dynamics $M_{1:T}(\mathbf{x}_{1:T})$, i.e.,

$$\mathbf{\Sigma}_{s,t} = \begin{cases} \mathbf{F}_s \mathbf{\Sigma}_{s-1,t}, & \text{if } s > t, \\ \mathbf{\Sigma}_{t,s}^{\mathrm{T}}, & \text{if } s < t, \\ \mathbf{\Sigma}_t, & \text{if } s = t, \end{cases}$$

where $\mathbf{\Sigma}_t$ can be found via the recursion from Step 1 of Algorithm 12 from Appendix B. As discussed below, this specification has the potentially useful implication that the algorithm reduces to an 'auxiliary-variable' version of the PCNL algorithm on the path space in the absence of resampling and backward sampling. Unfortunately, evaluating the preconditioning matrices $\widetilde{\mathbf{C}}_t$ is likely to incur a quadratic computational complexity in T which we prefer to avoid.

3. A compromise (which retains linear computational complexity in T) may be to truncate the above sum by setting

$$\widetilde{\mathbf{C}}_t \coloneqq \sum_{s=(t-L)\vee 1}^{(t+L)\wedge T} \mathbf{\Sigma}_{s,t},$$

for some $L \in [T]_0$ (note that this still requires the model assumption (4.4)).

A.6 Relationship with other methods

The algorithms proposed above relate to existing methods as follows.

1. **Generalisation of aPCNL.** For $\kappa = 1$, the Particle-aPCNL algorithm (and similarly the Particle-aPCNL+ algorithm as well as the twisted versions of either) generalises an *auxiliary preconditioned Crank–Nicolson–Langevin (aPCNL)* algorithm (which was mentioned but not explicitly derived in Titsias and Papaspiliopoulos (2018)) in the sense that the former reduces to the latter if $T = N = 1$. This can be seen as follows, where we again suppress the 'time' subscript $t = 1$ everywhere so that $\pi(\mathbf{x}) \propto M(\mathbf{x})G(\mathbf{x})$, where $M(\mathbf{x}) = \mathrm{N}(\mathbf{x}; \mathbf{m}, \mathbf{C})$. We also take $\widetilde{\mathbf{C}} \coloneqq \mathbf{C}$. Given that the current state of the Markov

chain is $\mathbf{x} = \mathbf{x}^0$ (we can assume that $k = 0$ without loss of generality), Step 1c of Algorithm 9 first refreshes the auxiliary variable by sampling $\mathbf{u} \sim \mathrm{N}(\mathbf{x}^0 + \frac{\delta}{2}\mathbf{C}\nabla \log G(\mathbf{x}^0), \frac{\delta}{2}\mathbf{C})$ and then proposes $\mathbf{x}^1 \sim \mathrm{N}((1-\beta)\mathbf{m} + \beta\mathbf{u}, (1-\beta)\mathbf{C})$. The remaining steps return $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^1$ as the new state with acceptance probability $1 \wedge \alpha_{\text{aPCNL}}(\mathbf{x}^0, \mathbf{x}^1; \mathbf{u})$, where

$$\begin{aligned}
&\alpha_{\text{aPCNL}}(\mathbf{x}^0, \mathbf{x}^1; \mathbf{u}) \\
&\quad\coloneqq \frac{1 - W^0}{1 - W^1} \\
&\quad= \frac{\pi(\mathbf{x}^1)\,\mathrm{N}(\mathbf{u}; \mathbf{x}^1 + \frac{\delta}{2}\mathbf{C}\nabla \log G(\mathbf{x}^1), \frac{\delta}{2}\mathbf{C})\,\mathrm{N}(\mathbf{x}^0; (1-\beta)\mathbf{m} + \beta\mathbf{u}, (1-\beta)\mathbf{C})}{\pi(\mathbf{x}^0)\,\mathrm{N}(\mathbf{u}; \mathbf{x}^0 + \frac{\delta}{2}\mathbf{C}\nabla \log G(\mathbf{x}^0), \frac{\delta}{2}\mathbf{C})\,\mathrm{N}(\mathbf{x}^1; (1-\beta)\mathbf{m} + \beta\mathbf{u}, (1-\beta)\mathbf{C})}.
\end{aligned}$$

Otherwise, the old state $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^0 = \mathbf{x}$ is returned as the new state. If $N = 1$ and $\kappa = 1$ but $T > 1$ then the aPCNL algorithm could still be recovered as a special case of (a slightly modified version of) the twisted Particle-aPCNL+ algorithm (and also of the twisted Particle-aPCNL algorithm if the potential functions $G_t(\mathbf{x}_{t-1:t})$ do not depend on $\mathbf{x}_{t-1}$) if the latter uses no resampling and ancestral tracing instead of backward sampling, and if $\delta_1 = \ldots = \delta_T$ and if the preconditioning matrices are specified via (2). However, we do not recommend this choice of preconditioning matrix as it leads to squared computational complexity in T (also incurred by aPCNL).

2. **Generalisation of PCNL.** Still taking $\kappa = 1$, the Particle-PCNL algorithm generalises the *preconditioned Crank–Nicolson–Langevin (PCNL)* algorithm (Cotter et al., 2013) in the sense that the former reduces to the latter if $T = N = 1$. This can be seen as follows, where we use the same notational conventions as in the case of aPCNL above. Step 1c of Algorithm 10 then marginally proposes $\mathbf{x}^1 \sim \mathrm{N}((1-\beta)\mathbf{m} + \beta[\mathbf{x}^0 + \frac{\delta}{2}\mathbf{C}\nabla \log G(\mathbf{x}^0)], (1-\beta^2)\mathbf{C})$. The remaining steps return $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^1$ as the new state with acceptance probability $1 \wedge \alpha_{\text{PCNL}}(\mathbf{x}^0, \mathbf{x}^1)$, where

$$\begin{aligned}
\alpha_{\text{PCNL}}(\mathbf{x}^0, \mathbf{x}^1) &\coloneqq \frac{1 - W^0}{1 - W^1} \\
&= \frac{\pi(\mathbf{x}^1)\,\mathrm{N}((1-\beta)\mathbf{m} + \beta[\mathbf{x}^1 + \frac{\delta}{2}\nabla \log G(\mathbf{x}^1)], (1-\beta^2)\mathbf{C})}{\pi(\mathbf{x}^0)\,\mathrm{N}((1-\beta)\mathbf{m} + \beta[\mathbf{x}^0 + \frac{\delta}{2}\nabla \log G(\mathbf{x}^0)], (1-\beta^2)\mathbf{C})}.
\end{aligned}$$

Otherwise, the old state $\tilde{\mathbf{x}} \coloneqq \mathbf{x}^0 = \mathbf{x}$ is returned as the new state. In particular, in analogue to Section 3.4, we can again interpret aPCNL as a version of PCNL with 'randomised' acceptance ratio.

B Twisted proposals

In this section, we detail the mutation kernel $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{t:T})$ used by the twisted Particle-aGRAD/Particle-aGRAD+ (4.4) and twisted Particle-aPCNL/Particle-aPCNL+ (A.4) algorithms. This mutation kernel can be thought of as the fully-twisted particle-filter proposal for the state-space model which is defined by the Gaussian transitions $p(\mathbf{x}_t|\mathbf{x}_{t-1}) = M_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}_t\mathbf{x}_{t-1} + \mathbf{b}_t, \mathbf{C}_t)$ from (4.4) and observation densities $p(\mathbf{u}_t|\mathbf{x}_t) = \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \frac{\delta_t}{2}\mathbf{V}_t)$, where we take $\mathbf{V}_t \coloneqq \mathbf{I}$ in the case of the twisted Particle-aGRAD or twisted Particle-aGRAD+ algorithm and $\mathbf{V}_t \coloneqq \mathbf{C}_t$ in the case of the twisted Particle-aPCNL or twisted Particle-aPCNL+ algorithm:

$$M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{t:T}) = p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{1:T}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}'_t\mathbf{x}_{t-1} + \mathbf{b}'_t, \mathbf{C}'_t).$$

General algorithm. Algorithm 12 explains how the twisted-proposal parameters $\mathbf{b}'_t \in \mathbb{R}^D$ and $\mathbf{F}'_t, \mathbf{C}'_t \in \mathbb{R}^{D \times D}$ can be calculated at linear complexity in T, independently of the total number of particles, N. Notably, Algorithm 12 does not require $\mathbf{C}_t$ to be invertible.

Algorithm 12 (twisted-proposal parameters). At the start of an iteration of the twisted Particle-aGRAD, Particle-aGRAD+, Particle-aPCNL or Particle-aPCNL+ algorithm (after having sampled all the auxiliary variables $\mathbf{u}_{1:T}$ upfront – e.g., as in Algorithm 14 from Appendix D.1 – which is possible because these only depend on the reference path),

1. recursively compute the moments of $p(\mathbf{x}_t) = \mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$, for $t = 1, \dots, T$, as

$$\begin{aligned}\boldsymbol{\mu}_t &\coloneqq \mathbf{F}_t \boldsymbol{\mu}_{t-1} + \mathbf{b}_t,\\ \boldsymbol{\Sigma}_t &\coloneqq \mathbf{F}_t \boldsymbol{\Sigma}_{t-1} \mathbf{F}_t^{\mathrm{T}} + \mathbf{C}_t,\end{aligned}$$

   if $t > 1$, and with initial condition $\boldsymbol{\mu}_1 = \mathbf{b}_1$ and $\boldsymbol{\Sigma}_1 = \mathbf{C}_1$,

2. recursively compute the moments of the time-reversed state transition kernels $p(\mathbf{x}_t|\mathbf{x}_{t+1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}_t^{\leftarrow}\mathbf{x}_{t+1} + \mathbf{b}_t^{\leftarrow}, \mathbf{C}_t^{\leftarrow})$, for $t = T-1, \dots, 1$, as

$$\begin{aligned}\mathbf{F}_t^{\leftarrow} &\coloneqq \mathbf{F}_{t+1} \boldsymbol{\Sigma}_t \boldsymbol{\Sigma}_{t+1}^{-1},\\ \mathbf{b}_t^{\leftarrow} &\coloneqq \boldsymbol{\mu}_t - \mathbf{F}_t^{\leftarrow} \boldsymbol{\mu}_{t+1},\\ \mathbf{C}_t^{\leftarrow} &\coloneqq \boldsymbol{\Sigma}_t - \mathbf{F}_t^{\leftarrow} \boldsymbol{\Sigma}_t \mathbf{F}_{t+1}^{\mathrm{T}}.\end{aligned}$$

3. run the Kalman filtering recursion for the time-reversed state-space model (i.e., with observation densities $p(\mathbf{u}_t|\mathbf{x}_t) = \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \frac{\delta_t}{2}\mathbf{V}_t)$, initial distribution $p(\mathbf{x}_t)$ and time-reversed state transitions $p(\mathbf{x}_t|\mathbf{x}_{t+1})$ found in Steps 1 and 2) to compute the moments of $p(\mathbf{x}_t|\mathbf{u}_{t:T}) = \mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_{t|t}^{\leftarrow}, \boldsymbol{\Sigma}_{t|t}^{\leftarrow})$, for $t = T, \dots, 1$,

4. set $\mathbf{F}'_1 \coloneqq \mathbf{0}_{D \times D}$, $\mathbf{b}'_1 \coloneqq \boldsymbol{\mu}_{1|1}^{\leftarrow}$ as well as $\mathbf{C}'_1 \coloneqq \boldsymbol{\Sigma}_{1|1}^{\leftarrow}$, and, for $t = 2, \dots, T$,

$$\begin{aligned}\mathbf{F}'_t &\coloneqq \boldsymbol{\Sigma}_{t|t}^{\leftarrow} \mathbf{F}_{t-1}^{\leftarrow} (\mathbf{C}_{t-1}^{\leftarrow} + \mathbf{F}_{t-1}^{\leftarrow} \boldsymbol{\Sigma}_{t|t}^{\leftarrow} \{\mathbf{F}_{t-1}^{\leftarrow}\}^{\mathrm{T}})^{-1},\\ \mathbf{b}'_t &\coloneqq \boldsymbol{\mu}_{t|t}^{\leftarrow} - \mathbf{F}'_t (\mathbf{F}_{t-1}^{\leftarrow} \boldsymbol{\mu}_{t|t}^{\leftarrow} + \mathbf{b}_{t-1}^{\leftarrow}),\\ \mathbf{C}'_t &\coloneqq (\mathbf{I} - \mathbf{F}'_t \mathbf{F}_{t-1}^{\leftarrow}) \boldsymbol{\Sigma}_{t|t}^{\leftarrow}.\end{aligned}$$

Algorithm 12 is justified by the decomposition

$$\begin{aligned}M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{t:T}) &\propto p(\mathbf{x}_{t-1:t}|\mathbf{u}_{t:T})\\ &= p(\mathbf{x}_{t-1}|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{u}_{t:T})\\ &= \mathrm{N}(\mathbf{x}_t; \mathbf{F}_t^{\leftarrow}\mathbf{x}_{t+1} + \mathbf{b}_t^{\leftarrow}, \mathbf{C}_t^{\leftarrow})\, \mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_{t|t}^{\leftarrow}, \boldsymbol{\Sigma}_{t|t}^{\leftarrow}) \qquad (25)\\ &\propto \mathrm{N}(\mathbf{x}_t; \mathbf{F}'_t \mathbf{x}_{t-1} + \mathbf{b}'_t, \mathbf{C}'_t), \qquad (26)\end{aligned}$$

where, as described in Algorithm 12, $p(\mathbf{x}_t|\mathbf{u}_{t:T}) = \mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_{t|t}^{\leftarrow}, \boldsymbol{\Sigma}_{t|t}^{\leftarrow})$ is the time-t filter for the time-reversed state-space model with the same observation densities $p(\mathbf{u}_t|\mathbf{x}_t) = \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t, \frac{\delta_t}{2}\mathbf{V}_t)$ as before but with initial distribution $p(\mathbf{x}_t) = \mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_T, \boldsymbol{\Sigma}_T)$ and time-reversed state transitions $p(\mathbf{x}_t|\mathbf{x}_{t+1}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}_t^{\leftarrow}\mathbf{x}_{t+1} + \mathbf{b}_t^{\leftarrow}, \mathbf{C}_t^{\leftarrow})$. Thus, in Algorithm 12:

- Step 1 calculates the marginal prior distributions of the states. To see this, note that this step is effectively the Kalman-filter recursion without observations. Note that if the prior dynamics are stationary with stationary distribution $\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then Step 1 can be skipped (because then $\boldsymbol{\mu}_t = \boldsymbol{\mu}$ and $\boldsymbol{\Sigma}_t = \boldsymbol{\Sigma}$, for any $t \in [T]$).

- Step 2 computes the parameters of the time-reversed transition kernels and follows from standard Gaussian algebra (see, e.g., Särkkä and Svensson, 2023, Section A.1) by noting that
$$\begin{bmatrix} \mathbf{x}_t \\ \mathbf{x}_{t+1} \end{bmatrix} \sim \mathrm{N}\left(\begin{bmatrix} \boldsymbol{\mu}_t \\ \boldsymbol{\mu}_{t+1} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_t & \mathbf{F}_{t+1}\boldsymbol{\Sigma}_t \\ \boldsymbol{\Sigma}_t \mathbf{F}_{t+1}^{\mathrm{T}} & \boldsymbol{\Sigma}_{t+1} \end{bmatrix} \right).$$

- Step 4 derives (B) from (B) and corresponds to a single update step of a Kalman filter (Särkkä and Svensson, 2023, Chapter 6, Equation 6.21), where $\mathbf{x}_{t-1}$ plays the rôle of an observation.

Alternative algorithm for invertible covariance matrices. If $\mathbf{C}_t$ is invertible for all $t \in [T]$, then the twisted-proposal parameters can be alternatively computed via Algorithm 13 which may be slightly simpler to implement for some users and which may provide additional numerical advantages in the case of explosive prior dynamics.

Algorithm 13 (twisted-proposal parameters: alternative). At the start of an iteration of the twisted Particle-aGRAD, Particle-aGRAD+, Particle-aPCNL or Particle-aPCNL+ algorithm (after having sampled all the auxiliary variables $\mathbf{u}_{1:T}$ upfront – e.g., as in Algorithm 14 from Appendix D.1 – which is possible because these only depend on the reference path),

1. run the Kalman filtering recursion to compute the moments of $p(\mathbf{x}_t|\mathbf{u}_{1:t-1}) = \mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})$, for $t = 1, \dots, T$,

2. run the Kalman smoothing (a.k.a. Rauch–Tung–Striebel smoothing) recursion to compute the moments of $p(\mathbf{x}_t|\mathbf{u}_{1:T}) = \mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_{t|T}, \boldsymbol{\Sigma}_{t|T})$, for $t = T, T-1, \dots, 1$,

3. for $t \in [T]$, set
$$\begin{aligned} \mathbf{C}'_t &:= [\mathbf{C}_t^{-1} + \boldsymbol{\Sigma}_{t|T}^{-1} - \boldsymbol{\Sigma}_{t|t-1}^{-1}]^{-1}, \\ \mathbf{F}'_t &:= \mathbf{C}'_t[\mathbf{C}_t^{-1}\mathbf{b}_t + \boldsymbol{\Sigma}_{t|T}^{-1}\boldsymbol{\mu}_{t|T} - \boldsymbol{\Sigma}_{t|t-1}^{-1}\boldsymbol{\mu}_{t|t-1}], \\ \mathbf{b}'_t &:= \mathbf{C}'_t\mathbf{C}_t^{-1}\mathbf{F}_t. \end{aligned}$$

Algorithm 13 is justified by the decomposition
$$\begin{aligned} p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{1:T}) &\propto p(\mathbf{x}_{t-1:t}, \mathbf{u}_{1:T}) \\ &= p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{u}_{1:T})p(\mathbf{x}_t|\mathbf{u}_{1:T}) \\ &\propto \frac{p(\mathbf{x}_t|\mathbf{x}_{t-1})}{p(\mathbf{x}_t|\mathbf{u}_{1:t-1})} p(\mathbf{x}_t|\mathbf{u}_{1:T}) \\ &= \frac{\mathrm{N}(\mathbf{x}_t; \mathbf{F}_t\mathbf{x}_{t-1} + \mathbf{b}_t, \mathbf{C}_t)}{\mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})} \mathrm{N}(\mathbf{x}_t; \boldsymbol{\mu}_{t|T}, \boldsymbol{\Sigma}_{t|T}) \\ &\propto \mathrm{N}(\mathbf{x}_t; \mathbf{F}'_t\mathbf{x}_{t-1} + \mathbf{b}'_t, \mathbf{C}'_t). \end{aligned}$$

C Integrating out the auxiliary variables

In this section, we prove a few lemmata which are used in subsequent sections.

- **Lemmata 1 and 2.** Lemmata 1 and 2 derive the determinant and inverse of a certain simple block matrix which appears repeatedly in the remainder of this section and also in Appendix E.
- **Lemma 3.** Lemma 3 will allow us to derive marginal proposal distributions of the various algorithms, i.e., the distribution of $\mathbf{x}_t^{-k_t} = (\mathbf{x}_t^0, \dots, \mathbf{x}_t^{k_t-1}, \mathbf{x}_t^{k_t+1}, \dots, \mathbf{x}_t^N)$ conditional on k_t, $\mathbf{x}_t^{k_t} = \mathbf{x}_t$ and all the particles and ancestor indices with time indices $s < t$, but with the auxiliary variables $\mathbf{u}_{1:T}$ integrated out.
- **Lemma 4.** Lemma 4 will allow us to evaluate the particle weights used in the 'marginal' algorithms (Particle-MALA, Particle-mGRAD and Particle-PCNL) at linear complexity in N although the weight of the nth particle depends on the values of all other particles.

C.1 Properties of a particular block matrix

Let $\mathbf{I}_M$ denote the $(M \times M)$ identity matrix. When $M = D$ we continue to leave out the subscript. Furthermore, let $\mathbf{1}_{M\times N} \in \{1\}^{M\times N}$ and denote a matrix in which every element is 1. For matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{D\times D}$, define the block matrix

$$\mathcal{M}_N(\mathbf{A}, \mathbf{B}) \coloneqq \mathbf{I}_N \otimes \mathbf{A} + \mathbf{1}_{N\times N} \otimes \mathbf{B} = \begin{bmatrix} \mathbf{A}+\mathbf{B} & \mathbf{B} & \dots & \mathbf{B} \\ \mathbf{B} & \mathbf{A}+\mathbf{B} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{B} \\ \mathbf{B} & \dots & \mathbf{B} & \mathbf{A}+\mathbf{B} \end{bmatrix} \in \mathbb{R}^{(DN)\times(DN)}. \quad (27)$$

Lemma 1. *For $N, D \in \mathbb{N}$, let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{D\times D}$. Then,*

$$\det(\mathcal{M}_N(\mathbf{A}, \mathbf{B})) = \det(\mathbf{A})^{N-1} \det(\mathbf{A} + N\mathbf{B}).$$

Proof. Subtracting the last row of $\mathcal{M}_N(\mathbf{A}, \mathbf{B})$ from all other rows and then adding the sum of the first $N-1$ columns to the last column gives the upper-triangular block matrix

$$\begin{bmatrix} \mathbf{A} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{B} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \mathbf{A} & \mathbf{0} \\ \mathbf{B} & \dots & \mathbf{B} & \mathbf{A}+N\mathbf{B} \end{bmatrix}.$$

This proves the result. □

Lemma 2. *For $N, D \in \mathbb{N}$, let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{D\times D}$, such that $\mathbf{A}$ and $(\mathbf{A} + N\mathbf{B})$ are invertible. Then*

$$\mathcal{M}_N(\mathbf{A}, \mathbf{B})^{-1} = \mathcal{M}_N(\mathbf{A}^{-1}, -(\mathbf{A} + N\mathbf{B})^{-1}\mathbf{B}\mathbf{A}^{-1}).$$

Proof. We must have $\mathcal{M}_N(\mathbf{A}, \mathbf{B})\mathcal{M}_N(\mathbf{F}, \mathbf{G}) = \mathbf{I}_{DN}$ and hence

$$(\mathbf{A}+\mathbf{B})(\mathbf{F}+\mathbf{G}) + (N-1)\mathbf{B}\mathbf{G} = \mathbf{I},$$
$$\mathbf{B}(\mathbf{F}+\mathbf{G}) + (\mathbf{A}+\mathbf{B})\mathbf{G} + (N-2)\mathbf{B}\mathbf{G} = \mathbf{0}.$$

This implies $\mathbf{F} = \mathbf{A}^{-1}$ and $\mathbf{G} = -(\mathbf{A} + N\mathbf{B})^{-1}\mathbf{B}\mathbf{A}^{-1}$. □

C.2 Conditional and marginal proposal distributions

In this section, for any tuple $(\mathbf{z}_0, \ldots, \mathbf{z}_N)$ of values in $\mathcal{X} := \mathbb{R}^D$ and any $n \in [N]_0$, we write $\mathbf{z}_{-n} := (\mathbf{z}_0, \ldots, \mathbf{z}_{n-1}, \mathbf{z}_{n+1}, \ldots, \mathbf{z}_N)$. Given some $N \in \mathbb{N}$, $n \in [N]_0$, $\mathbf{x}_n, \boldsymbol{\phi}_n \in \mathcal{X}$, we consider the following joint distribution on $\mathcal{X}^{N+1}$:

$$q_{-n}(\mathbf{u}, \mathbf{x}_{-n}|\mathbf{x}_n) := \mathrm{N}(\mathbf{u}; \mathbf{x}_n + \boldsymbol{\phi}_n, \mathbf{E}) \prod_{\substack{m=0 \\ m \neq n}}^{N} \mathrm{N}(\mathbf{x}_m; \mathbf{v}_m + \mathbf{H}_m \mathbf{u}, \mathbf{D}_m), \tag{28}$$

where, for any $m \in [N]_0$, $\mathbf{v}_m \in \mathcal{X}$ and $\mathbf{H}_m \in \mathbb{R}^{D \times D}$, and $\mathbf{D}_m, \mathbf{E} \in \mathbb{R}^{D \times D}$ are positive definite and symmetric.

To simplify the presentation – and with some abuse of notation since we use the same symbols for tuples and their vectorised versions – we write

$$\mathbf{x}_{-n} := \begin{bmatrix} \mathbf{x}_0 \\ \vdots \\ \mathbf{x}_{n-1} \\ \mathbf{x}_{n+1} \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \quad \mathbf{v}_{-n} := \begin{bmatrix} \mathbf{v}_0 \\ \vdots \\ \mathbf{v}_{n-1} \\ \mathbf{v}_{n+1} \\ \vdots \\ \mathbf{v}_n \end{bmatrix}, \quad \mathbf{H}_{-n} := \begin{bmatrix} \mathbf{H}_0 \\ \vdots \\ \mathbf{H}_{n-1} \\ \mathbf{H}_{n+1} \\ \vdots \\ \mathbf{H}_n \end{bmatrix}, \quad \mathbf{D}_{-n} := \operatorname{diag}\left(\begin{bmatrix} \mathbf{D}_0 \\ \vdots \\ \mathbf{D}_{n-1} \\ \mathbf{D}_{n+1} \\ \vdots \\ \mathbf{D}_n \end{bmatrix}\right),$$

where, in the last expression, diag induces a block-diagonal matrix. With this notation, we can formulate (C.2) equivalently as

$$q_{-n}(\mathbf{u}, \mathbf{x}_{-n}|\mathbf{x}_n) = \mathrm{N}(\mathbf{u}; \mathbf{x}_n + \boldsymbol{\phi}_n, \mathbf{E})\, \mathrm{N}(\mathbf{x}_{-n}; \mathbf{v}_{-n} + \mathbf{H}_{-n}\mathbf{u}, \mathbf{D}_{-n}). \tag{29}$$

Lemma 3. *For any $n \in [N]_0$ and $\mathbf{x}_n \in \mathcal{X}$,*

1. *the marginal distribution of $\mathbf{x}_{-n}$ given $\mathbf{x}_n$ under (C.2) is*

$$q_{-n}(\mathbf{x}_{-n}|\mathbf{x}_n) = \mathrm{N}(\mathbf{x}_{-n}; \boldsymbol{\mu}_{-n}, \boldsymbol{\Sigma}_{-n}),$$

where

$$\begin{aligned} \boldsymbol{\mu}_{-n} &:= \mathbf{v}_{-n} + \mathbf{H}_{-n}(\mathbf{x}_n + \boldsymbol{\phi}_n), \\ \boldsymbol{\Sigma}_{-n} &:= \mathbf{D}_{-n} + \mathbf{H}_{-n}\mathbf{E}\mathbf{H}_{-n}^{\mathrm{T}}; \end{aligned}$$

2. *the conditional distribution of $\mathbf{u}$ given $\mathbf{x}_{-n}$ and $\mathbf{x}_n$ under (C.2) is*

$$q_{-n}(\mathbf{u}|\mathbf{x}_{-n}, \mathbf{x}_n) = \mathrm{N}(\mathbf{u}; \mathbf{x}_n + \boldsymbol{\phi}_n + \mathbf{K}[\mathbf{x}_{-n} - \mathbf{v}_{-n} - \mathbf{H}_{-n}(\mathbf{x}_n + \boldsymbol{\phi}_n)], (\mathbf{I} - \mathbf{K}\mathbf{H}_{-n})\mathbf{E}),$$

where $\mathbf{K} := \mathbf{E}\mathbf{H}_{-n}^{\mathrm{T}}(\mathbf{D}_{-n} + \mathbf{H}_{-n}\mathbf{E}\mathbf{H}_{-n}^{\mathrm{T}})^{-1}$.

Proof. This follows by simple algebra (see, e.g., Särkkä and Svensson, 2023, Appendix A.1). □

Lemma 4. *Assume now that $\mathbf{H}_m = \mathbf{H}$ and $\mathbf{D}_m = \mathbf{D}$, for any $m \in [N]_0$. Then, with the notation from Lemma 3,*

$$q_{-n}(\mathbf{x}_{-n}|\mathbf{x}_n) \propto H_{\boldsymbol{\phi}_n}(\mathbf{x}_n, \mathbf{v}_n, \bar{\mathbf{x}}, \bar{\mathbf{v}}) I((\mathbf{x}_m - \mathbf{v}_m)_{m=0}^{N}),$$

where

1. *$\mathbf{z}_{0:N} \mapsto I(\mathbf{z}_{0:N})$ is invariant under any permutation of its arguments;*

2. *$\bar{\mathbf{x}} := \frac{1}{N+1}\sum_{m=0}^{N}\mathbf{x}_n$, and $\bar{\mathbf{v}} := \frac{1}{N+1}\sum_{m=0}^{N}\mathbf{v}_n$, and*

$$\begin{aligned}
&\log H_{\boldsymbol{\phi}}(\mathbf{x}, \mathbf{v}, \bar{\mathbf{x}}, \bar{\mathbf{v}}) \\
&\quad = \tfrac{1}{2}(\mathbf{x}-\mathbf{v})^{\mathrm{T}}(\mathbf{D}^{-1}+\mathbf{G})(\mathbf{x}-\mathbf{v}) \\
&\qquad - \tfrac{1}{2}N(\mathbf{x}+\boldsymbol{\phi})^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}(\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}(\mathbf{x}+\boldsymbol{\phi}) \\
&\qquad - (N+1)(\bar{\mathbf{x}}-\bar{\mathbf{v}})^{\mathrm{T}}[\mathbf{G}(\mathbf{x}-\mathbf{v}) - (\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}(\mathbf{x}+\boldsymbol{\phi})] \\
&\qquad - (\mathbf{x}-\mathbf{v})^{\mathrm{T}}(\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}(\mathbf{x}+\boldsymbol{\phi}),
\end{aligned}$$

whose evaluation complexity does not depend on N. Here,

$$\begin{aligned}
\mathbf{G} &:= (\mathbf{D}+N\mathbf{HEH}^{\mathrm{T}})^{-1}\mathbf{HEH}^{\mathrm{T}}\mathbf{D}^{-1} &(30)\\
&= \mathbf{D}^{-1}\mathbf{HE}(\mathbf{E}+N\mathbf{EH}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{HE})^{-1}\mathbf{EH}^{\mathrm{T}}\mathbf{D}^{-1}. &(31)
\end{aligned}$$

Proof. The equivalence of (2) and (2) follows from the *push-through identity* (Henderson and Searle, 1981). By assumption, $\boldsymbol{\Sigma}_{-n} = \mathcal{M}_N(\mathbf{D}, \mathbf{HEH}^{\mathrm{T}})$. Thus, Lemma 2 gives

$$\boldsymbol{\Sigma}_{-n}^{-1} = \mathcal{M}_N(\mathbf{D}^{-1}, -\mathbf{G}).$$

In particular, letting $\otimes$ be the Kronecker product, this implies that

$$\begin{aligned}
\boldsymbol{\Sigma}_{-n}^{-1}\mathbf{H}_{-n} &= \mathbf{1}_{N\times 1}\otimes[(\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}], \\
\mathbf{H}_{-n}^{\mathrm{T}}\boldsymbol{\Sigma}_{-n}^{-1}\mathbf{H}_{-n} &= N\mathbf{H}^{\mathrm{T}}(\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}.
\end{aligned}$$

Therefore, defining

$$\mathbf{x} := \begin{bmatrix}\mathbf{x}_0\\ \vdots \\ \mathbf{x}_N\end{bmatrix}, \quad \mathbf{v} := \begin{bmatrix}\mathbf{v}_0\\ \vdots \\ \mathbf{v}_N\end{bmatrix}, \quad \boldsymbol{\Sigma} := \mathcal{M}_{N+1}(\mathbf{D}^{-1}, -\mathbf{G})^{-1},$$

we have

$$
\begin{aligned}
&q_{-n}(\mathbf{x}_{-n}|\mathbf{x}_n) \\
&\quad\propto \exp\Big(-\tfrac{1}{2}\Big[(\mathbf{x}_{-n}-\mathbf{v}_{-n}-\mathbf{H}_{-n}(\mathbf{x}_n+\boldsymbol{\phi}_n))^{\mathrm{T}}\boldsymbol{\Sigma}_{-n}^{-1}(\mathbf{x}_{-n}-\mathbf{v}_{-n}-\mathbf{H}_{-n}(\mathbf{x}_n+\boldsymbol{\phi}_n))\Big]\Big) \\
&\quad= \exp\Big(-\tfrac{1}{2}\Big[(\mathbf{x}_{-n}-\mathbf{v}_{-n})^{\mathrm{T}}\boldsymbol{\Sigma}_{-n}^{-1}(\mathbf{x}_{-n}-\mathbf{v}_{-n}) \\
&\qquad\qquad + (\mathbf{x}_n+\boldsymbol{\phi}_n)^{\mathrm{T}}\mathbf{H}_{-n}^{\mathrm{T}}\boldsymbol{\Sigma}_{-n}^{-1}\mathbf{H}_{-n}(\mathbf{x}_n+\boldsymbol{\phi}_n) \\
&\qquad\qquad - 2(\mathbf{x}_{-n}-\mathbf{v}_{-n})^{\mathrm{T}}\boldsymbol{\Sigma}_{-n}^{-1}\mathbf{H}_{-n}(\mathbf{x}_n+\boldsymbol{\phi}_n)\Big]\Big) \\
&\quad= \exp\Big(-\tfrac{1}{2}\Big[(\mathbf{x}-\mathbf{v})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{v}) \\
&\qquad\qquad - (\mathbf{x}_n-\mathbf{v}_n)^{\mathrm{T}}(\mathbf{D}^{-1}-\mathbf{G})(\mathbf{x}_n-\mathbf{v}_n) \\
&\qquad\qquad + 2(\mathbf{x}_{-n}-\mathbf{v}_{-n})^{\mathrm{T}}[\mathbf{1}_{N\times 1}\otimes\mathbf{G}](\mathbf{x}_n-\mathbf{v}_n) \\
&\qquad\qquad + N(\mathbf{x}_n+\boldsymbol{\phi}_n)^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}(\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}(\mathbf{x}_n+\boldsymbol{\phi}_n) \\
&\qquad\qquad - 2(\mathbf{x}_{-n}-\mathbf{v}_{-n})^{\mathrm{T}}\boldsymbol{\Sigma}_{-n}^{-1}\mathbf{H}_{-n}(\mathbf{x}_n+\boldsymbol{\phi}_n)\Big]\Big) \\
&\quad= \exp\Big(-\tfrac{1}{2}\Big[(\mathbf{x}-\mathbf{v})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{v}) \\
&\qquad\qquad - (\mathbf{x}_n-\mathbf{v}_n)^{\mathrm{T}}(\mathbf{D}^{-1}-\mathbf{G})(\mathbf{x}_n-\mathbf{v}_n) \\
&\qquad\qquad + N(\mathbf{x}_n+\boldsymbol{\phi}_n)^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}(\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}(\mathbf{x}_n+\boldsymbol{\phi}_n) \\
&\qquad\qquad + 2(\mathbf{x}-\mathbf{v})^{\mathrm{T}}\left\{\mathbf{1}_{(N+1)\times 1}\otimes[\mathbf{G}(\mathbf{x}_n-\mathbf{v}_n)-(\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}(\mathbf{x}_n+\boldsymbol{\phi}_n)]\right\} \\
&\qquad\qquad - 2(\mathbf{x}_n-\mathbf{v}_n)^{\mathrm{T}}[\mathbf{G}(\mathbf{x}_n-\mathbf{v}_n)-(\mathbf{D}^{-1}-N\mathbf{G})\mathbf{H}(\mathbf{x}_n+\boldsymbol{\phi}_n)]\Big]\Big) \\
&\quad= H_{\boldsymbol{\phi}_n}(\mathbf{x}_n,\mathbf{v}_n,\bar{\mathbf{x}},\bar{\mathbf{v}})I((\mathbf{x}_m-\mathbf{v}_m)_{m=0}^{N}),
\end{aligned}
$$

with

$$I((\mathbf{x}_m-\mathbf{v}_m)_{m=0}^{N}) \propto \exp(-\tfrac{1}{2}(\mathbf{x}-\mathbf{v})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{v})).$$

This completes the proof. □

D Generic algorithms and proof of Propositions 1–13

In this section, we prove that the algorithms proposed in this work leave π_T invariant. To this end, we first prove the validity of two generic algorithms.

- **Generic auxiliary algorithm.** The first generic algorithm includes auxiliary variables $\mathbf{u}_t$ in the space and admits the 'auxiliary-variable' based algorithms: Particle-aMALA, Particle-aGRAD, Particle-aPCNL as well as their smoothing-gradient ('+') and twisted versions, as special cases. Its proof extends the auxiliary-variable interpretation of the Particle-RWM algorithm which was given in Corenflos and Särkkä (2023).

- **Generic marginal algorithm.** The second generic algorithm integrates out the auxiliary variables and admits the 'marginal' algorithms from the main manuscript (Particle-MALA, Particle-mGRAD, Particle-PCNL). Its proof relies on an argument previously given in Finke et al. (2016).

D.1 Generic auxiliary algorithm

Define an extended target distribution

$$\pi'_T(\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \coloneqq \pi_T(\mathbf{x}_{1:T}) \prod_{t=1}^{T} \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t + \Phi_t(\mathbf{x}_{1:T}), \mathbf{E}_t(\mathbf{x}_{t-1:t})), \tag{32}$$

where, for any $t \in [T]$, $\Phi_t \colon \mathcal{X}^T \to \mathcal{X}$ is a function satisfying

$$\Phi_t(\mathbf{x}_{1:T}) = \begin{cases} \boldsymbol{\phi}_T(\mathbf{x}_{t-T:T}), & \text{if } t = T, \\ \boldsymbol{\phi}_t(\mathbf{x}_{t-1:t}) + \boldsymbol{\psi}_t(\mathbf{x}_{t:t+1}), & \text{otherwise,} \end{cases}$$

and $\mathbf{E}_t(\mathbf{x}_{t-1:t}) \in \mathbb{R}^{D \times D}$ is some positive-definite symmetric matrix. Additionally, let

$$Q'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{1:T}) = M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{1:T}) G'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{1:T}),$$

for some mutation kernel $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{1:T})$ and some potential function $G'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{1:T})$ (both of which may depend on some or all of $\mathbf{u}_1, \dots, \mathbf{u}_t$) such that

$$\pi'_T(\mathbf{x}_{1:T}|\mathbf{u}_{1:T}) \propto \prod_{t=1}^{T} Q'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{1:T}).$$

Algorithm 14 (generic auxiliary algorithm). Given $\mathbf{x}_{1:T} \in \mathcal{X}^T$, sample

$$\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \Phi_t(\mathbf{x}_{1:T}), \mathbf{E}_t(\mathbf{x}_{t-1:t})),$$

for any $t = 1, \dots, T$ and then

```
1. for $t = 1, \dots, T$,
    a) sample $k_t$ from a uniform distribution on $[N]_0$ and set $\mathbf{x}_t^{k_t} \coloneqq \mathbf{x}_t$,
    b) if $t > 1$, set $a_{t-1}^{k_t} \coloneqq k_{t-1}$ and sample $a_{t-1}^n = i$ w.p. $W_{t-1}^i$, for $n \in [N]_0 \setminus \{k_t\}$,
    c) sample $\mathbf{x}_t^n \sim M'_t(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n}; \mathbf{u}_{1:T})$ for $n \in [N]_0 \setminus \{k_t\}$,
    d) for $n \in [N]_0$, set $w_t^n \propto G'_t(\mathbf{x}_{t-2:t}^{(n)}; \mathbf{u}_{1:T})$,
    e) for $n \in [N]_0$, set $W_t^n \coloneqq w_t^n / \sum_{m=0}^N w_t^m$;
2. sample $i \in [N]_0 \setminus \{k_T\}$ w.p. $\frac{W_T^i}{1 - W_T^{k_T}}$; set $l_T \coloneqq i$ w.p. $1 \wedge \frac{1 - W_T^{k_T}}{1 - W_T^i}$; otherwise, set $l_T \coloneqq k_T$;
3. for $t = T-1, \dots, 1$, sample $l_t = i \in [N]_0$ w.p.
    $$\frac{W_t^i Q'_{t+1}((\mathbf{x}_{t-1:t}^{(i)}, \mathbf{x}_{t+1}^{l_{t+1}}); \mathbf{u}_{1:T}) Q'_{t+2}((\mathbf{x}_t^i, \mathbf{x}_{t+1}^{l_{t+1}}, \mathbf{x}_{t+2}^{l_{t+2}}); \mathbf{u}_{1:T})}{\sum_{n=0}^N W_t^n Q'_{t+1}((\mathbf{x}_{t-1:t}^{(n)}, \mathbf{x}_{t+1}^{l_{t+1}}); \mathbf{u}_{1:T}) Q'_{t+2}((\mathbf{x}_t^n, \mathbf{x}_{t+1}^{l_{t+1}}, \mathbf{x}_{t+2}^{l_{t+2}}); \mathbf{u}_{1:T})};$$
4. return $\tilde{\mathbf{x}}_{1:T} \coloneqq (\mathbf{x}_1^{l_1}, \dots, \mathbf{x}_t^{l_T})$.
```

Proposition 14 (validity of the generic auxiliary algorithm). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 14 induces a Markov kernel $P(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

Proof (of Proposition 14). The extended distribution from (D.1) admits $\pi_T(\mathbf{x}_{1:T})$ as a marginal. Therefore, a valid MCMC update for sampling from this extended distribution is given by alternating the following two steps. Given $\mathbf{x}_{1:T} \in \mathcal{X}^T$,

1. sample $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \Phi_t(\mathbf{x}_{1:T}), \mathbf{E}_t(\mathbf{x}_{t-1:t}))$, for $t = 1, \dots, T$;

2. run a standard CSMC algorithm with backward sampling (as in Algorithm 1) targeting $\pi'_T(\mathbf{x}_{1:T}|\mathbf{u}_{1:T})$ but with $M_t(\mathbf{x}_t|\mathbf{x}_{t-1})$, $G_t(\mathbf{x}_{t-1:t})$, and $Q_t(\mathbf{x}_{t-1:t})$ replaced by $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{1:T})$, $G'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{1:T})$ and $Q'_t(\mathbf{x}_{t-2:t}; \mathbf{u}_{1:T})$, and with appropriate adjustments (e.g., of the backward kernels) to account for the possibility that the model may only be second-order Markov.

These to steps are equivalent to Algorithm 14. $\square$

D.2 Generic marginal algorithm

Consider the same setting as above but now assume that for any $t \in [T]$, $\boldsymbol{\psi}_t \equiv \mathbf{0}$, so that $\Phi_t(\mathbf{x}_{1:T}) = \boldsymbol{\phi}_t(\mathbf{x}_{t-1:t})$ as well as that $\mathbf{E}_t(\mathbf{x}_{t-1:t}) = \mathbf{E}_t$ is independent of $\mathbf{x}_{t-1:t}$.

Furthermore, assume that $M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{1:T}) = M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t)$ only depends on the tth auxiliary variable $\mathbf{u}_t$ and, specifically, is a Gaussian distribution of the following form:

$$M'_t(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_t) \coloneqq \mathrm{N}(\mathbf{x}_t; \mathbf{v}_t(\mathbf{x}_{t-1}) + \mathbf{H}_t\mathbf{u}_t, \mathbf{D}_t),$$

where $\mathbf{v}_t(\mathbf{x}) \in \mathcal{X}$ whilst $\mathbf{H}_t, \mathbf{D}_t \in \mathbb{R}^{D\times D}$ do not depend on $\mathbf{x}_{t-1}$ and define

$$q_t^{-n}(\mathbf{x}_t^{-n}, \mathbf{u}_t|\mathbf{x}_t^n; \mathcal{H}_{t-1}) \coloneqq \mathrm{N}(\mathbf{u}_t; \mathbf{x}_t^n + \boldsymbol{\phi}_t(\mathbf{x}_{t-1:t}^{(n)}), \mathbf{E}_t) \prod_{\substack{m=0\\ m\neq n}}^{N} M'_t(\mathbf{x}_t^m|\mathbf{x}_{t-1}^{a_{t-1}^m}; \mathbf{u}_t),$$

where $\mathcal{H}_{t-1}$ is the history of the particle system up to time $t-1$, i.e., all particles and ancestor indices with 'time' subscript $s \leq t-1$. By Lemma 3 from Appendix C, we obtain a closed-form expression for

$$q_t^{-n}(\mathbf{x}_t^{-n}|\mathbf{x}_t^n; \mathcal{H}_{t-1}) \coloneqq \int_{\mathcal{X}} q_t^{-n}(\mathbf{x}_t^{-n}, \mathbf{u}_t|\mathbf{x}_t^n; \mathcal{H}_{t-1}) \, \mathrm{d}\mathbf{u}_t.$$

Algorithm 15 (generic marginal algorithm). Given $\mathbf{x}_{1:T} \in \mathcal{X}^T$:

```
1. for $t = 1, \dots, T$,
    a) sample $k_t$ from a uniform distribution on $[N]_0$ and set $\mathbf{x}_t^{k_t} := \mathbf{x}_t$,
    b) if $t > 1$, set $a_{t-1}^{k_t} := k_{t-1}$ and sample $a_{t-1}^n = i$ w.p. $W_{t-1}^i$, for $n \in [N]_0 \setminus \{k_t\}$,
    c) sample $\mathbf{x}_t^{-k_t} \sim q_t^{-k_t}(\mathbf{x}_t^{-k_t}|\mathbf{x}_t^{k_t}; \mathcal{H}_{t-1})$
       (e.g. by sampling $\mathbf{u}_t \sim \mathrm{N}(\mathbf{x}_t + \boldsymbol{\phi}_t(\mathbf{x}_{t-1:t}), \mathbf{E}_t)$ and then $\mathbf{x}_t^n \sim M_t'(\,\cdot\,|\mathbf{x}_{t-1}^{a_{t-1}^n}; \mathbf{u}_t)$ for $n \in [N]_0 \setminus \{k_t\}$),
    d) for $n \in [N]_0$, set $w_t^n \propto Q_t(\mathbf{x}_{t-1:t}^{(n)}) q_t^{-n}(\mathbf{x}_t^{-n}|\mathbf{x}_t^n; \mathcal{H}_{t-1})$,
    e) for $n \in [N]_0$, set $W_t^n := w_t^n / \sum_{m=0}^N w_t^m$;
2. sample $i \in [N]_0 \setminus \{k_T\}$ w.p. $\dfrac{W_T^i}{1 - W_T^{k_T}}$; set $l_T := i$ w.p. $1 \wedge \dfrac{1 - W_T^{k_T}}{1 - W_T^i}$; otherwise, set $l_T := k_T$;
3. for $t = T-1, \dots, 1$, sample $l_t = i \in [N]_0$ w.p. $\dfrac{W_t^i Q_{t+1}(\mathbf{x}_t^i, \mathbf{x}_{t+1}^{l_{t+1}})}{\sum_{n=0}^N W_t^n Q_{t+1}(\mathbf{x}_t^n, \mathbf{x}_{t+1}^{l_{t+1}})}$;
4. return $\tilde{\mathbf{x}}_{1:T} := (\mathbf{x}_1^{l_1}, \dots, \mathbf{x}_t^{l_T})$.
```

Algorithm 15 can be implemented in $\mathrm{O}(N)$ operations because Lemma 3 from Appendix C allows us to write the weight in Step 1d as

$$
\begin{aligned}
w_t^n &\propto Q_t(\mathbf{x}_{t-1:t}^{(n)}) q_t^{-n}(\mathbf{x}_t^{-n}|\mathbf{x}_t^n; \mathcal{H}_{t-1})\\
&\propto Q_t(\mathbf{x}_{t-1:t}^{(n)}) H_{t, \boldsymbol{\phi}_t(\mathbf{x}_{t-1:t}^{(n)})}(\mathbf{x}_t^n, \mathbf{v}_t^n, \bar{\mathbf{x}}_t, \bar{\mathbf{v}}_t),
\end{aligned}
$$

where $\mathbf{v}_t^n := \mathbf{v}_t(\mathbf{x}_{t-1}^{a_{t-1}^n})$, $\bar{\mathbf{x}}_t := \frac{1}{N+1}\sum_{m=0}^N \mathbf{x}_t^m$, $\bar{\mathbf{v}}_t := \frac{1}{N+1}\sum_{m=0}^N \mathbf{v}_t^m$ and

$$
\begin{aligned}
\log H_{t,\boldsymbol{\phi}}(\mathbf{x}, \mathbf{v}, \bar{\mathbf{x}}, \bar{\mathbf{v}}) = \tfrac{1}{2}(\mathbf{x} - \mathbf{v})^{\mathrm{T}}(\mathbf{D}_t^{-1} + \mathbf{G}_t)(\mathbf{x} - \mathbf{v})&\\
- \tfrac{1}{2}N(\mathbf{x} + \boldsymbol{\phi})^{\mathrm{T}}\mathbf{H}_t^{\mathrm{T}}(\mathbf{D}_t^{-1} - N\mathbf{G}_t)\mathbf{H}_t(\mathbf{x} + \boldsymbol{\phi})&\\
- (N+1)(\bar{\mathbf{x}} - \bar{\mathbf{v}})^{\mathrm{T}}[\mathbf{G}_t(\mathbf{x} - \mathbf{v}) - (\mathbf{D}_t^{-1} - N\mathbf{G}_t)\mathbf{H}_t(\mathbf{x} + \boldsymbol{\phi})]&\\
- (\mathbf{x} - \mathbf{v})^{\mathrm{T}}(\mathbf{D}_t^{-1} - N\mathbf{G}_t)\mathbf{H}_t(\mathbf{x} + \boldsymbol{\phi}),&
\end{aligned}
$$

with $\mathbf{G}_t := (\mathbf{D} + N\mathbf{H}_t\mathbf{E}_t\mathbf{H}_t^{\mathrm{T}})^{-1}\mathbf{H}_t\mathbf{E}_t\mathbf{H}_t^{\mathrm{T}}\mathbf{D}_t^{-1}$ (see (2) for an alternative expression).

Proposition 15 (validity of the generic marginal algorithm). *Sampling $\tilde{\mathbf{x}}_{1:T}$ given $\mathbf{x}_{1:T}$ via Algorithm 15 induces a Markov kernel $P(\tilde{\mathbf{x}}_{1:T}|\mathbf{x}_{1:T})$ which leaves π_T invariant.*

Proof (of Proposition 15). We begin with a few observations.

1. Since the unnormalised weights satisfy

$$
w_t^n \propto Q_t(\mathbf{x}_{t-1:t}^{(n)}) q_t^{-n}(\mathbf{x}_t^{-n}|\mathbf{x}_t^n; \mathcal{H}_{t-1}), \tag{33}
$$

we have that

$$
q_t^{-k_t}(\mathbf{x}_t^{-k_t}|\mathbf{x}_t^{k_t}; \mathcal{H}_{t-1}) = \frac{w_t^{k_t}}{w_t^{l_t}} \frac{Q_t(\mathbf{x}_{t-1:t}^{(l_t)})}{Q_t(\mathbf{x}_{t-1:t}^{(k_t)})} q_t^{-l_t}(\mathbf{x}_t^{-l_t}|\mathbf{x}_t^{l_t}; \mathcal{H}_{t-1}).
$$

2. For a given set of final-time weights $\{W_T^n\}_{n\in[N]_0}$, let $R_T(\,\cdot\,|\,\cdot\,;\mathcal{H}_T)$ be the $\sum_{n=0}^N W_T^n \delta_n$-invariant Markov kernel used in Step 2 of Algorithm 15. That is, sampling $l_T \sim R_T(\,\cdot\,|k_T;\mathcal{H}_T)$ could be the forced-move update; or, in the more common specification of CSMC algorithms (Andrieu et al., 2010), i.e. without the forced-move update, we would simply have $R_T(l_T|k_T;\mathcal{H}_T) = W_T^{l_T}$. In either case, it can then be verified that
$$W_T^{k_T} R_T(l_T|k_T;\mathcal{H}_T) = W_T^{l_T} R_T(k_T|l_T;\mathcal{H}_T),$$
for any $k_T, l_T \in [N]_0$.

3. Under Algorithm 15, we have the following identities (with probability 1): $\mathbf{x}_t = \mathbf{x}_t^{k_t}$ and $\mathbf{x}_t' = \mathbf{x}_t^{l_t}$, for $1 \le t \le T$, as well as $a_{t-1}^{k_t} = k_{t-1}$, for any $1 < t \le T$.

Putting these observations together then shows that the Algorithm 15 targets the following extended distribution (i.e., this is the distribution of all random variables obtained if we first sampled $\mathbf{x}_{1:T} \sim \pi_T$ and then ran Algorithm 15):
$$\begin{aligned}
&\frac{\pi_T(\mathbf{x}_{1:T})}{(N+1)^T}\delta_{\mathbf{x}_{1:T}}(\mathbf{x}_{1:T}^{k_{1:T}})\left[\prod_{t=1}^T q_t^{-k_t}(\mathbf{x}_t^{-k_t}|\mathbf{x}_t^{k_t};\mathcal{H}_{t-1})\right]\left[\prod_{t=2}^T \delta_{k_{t-1}}(a_{t-1}^{k_t})\prod_{\substack{n=0\\n\neq k_t}}^N W_{t-1}^{a_{t-1}^n}\right]\\
&\times R_T(l_T|k_T;\mathcal{H}_T)\left[\prod_{t=1}^{T-1}\frac{w_t^{l_t}Q_{t+1}(\mathbf{x}_t^{l_t},\mathbf{x}_{t+1}^{l_{t+1}})}{\sum_{m=0}^N w_t^m Q_{t+1}(\mathbf{x}_t^m,\mathbf{x}_{t+1}^{l_{t+1}})}\right]\delta_{\mathbf{x}_{1:T}^{l_{1:T}}}(\tilde{\mathbf{x}}_{1:T})\\
=&\frac{\pi_T(\tilde{\mathbf{x}}_{1:T})}{(N+1)^T}\delta_{\tilde{\mathbf{x}}_{1:T}}(\mathbf{x}_{1:T}^{l_{1:T}})\left[\prod_{t=1}^T q_t^{-l_t}(\mathbf{x}_t^{-l_t}|\mathbf{x}_t^{l_t};\mathcal{H}_{t-1})\right]\\
&\times\left[\prod_{t=2}^T\frac{w_{t-1}^{a_{t-1}^{l_t}}Q_t(\mathbf{x}_{t-1}^{a_{t-1}^{l_t}},\mathbf{x}_t^{l_{t+1}})}{\sum_{m=0}^N w_{t-1}^m Q_t(\mathbf{x}_{t-1}^m,\mathbf{x}_t^{l_t})}\prod_{\substack{n=0\\n\neq l_t}}^N W_{t-1}^{a_{t-1}^n}\right]\\
&\times R_T(k_T|l_T;\mathcal{H}_T)\left[\prod_{t=1}^{T-1}\delta_{a_{t-1}^{k_t}}(k_{t-1})\right]\delta_{\mathbf{x}_{1:T}^{k_{1:T}}}(\mathbf{x}_{1:T}),
\end{aligned}$$
where the r.h.s. is the distribution obtained if we first sampled $\tilde{\mathbf{x}}_{1:T} \sim \pi_T$ and then ran Algorithm 15 algorithm but with ancestor sampling (Lindsten et al., 2012) instead of backward sampling. This is a modified version of the proof technique from Finke et al. (2016). In other words, if $\mathbf{x}_{1:T} \sim \pi_T$ and if $\tilde{\mathbf{x}}_{1:T}$ is sampled via Algorithm 15, then $\tilde{\mathbf{x}}_{1:T} \sim \pi_T$. This completes the proof. □

D.3 Invariance of the algorithms

We can now easily verify the validity of the 'auxiliary' algorithms (Particle-aMALA, Particle-aMALA+, Particle-aGRAD, Particle-aGRAD+, Particle-aPCNL, Particle-aPCNL+, and twisted Particle-aGRAD/Particle-aGRAD+/Particle-aPCNL/Particle-aPCNL+) by noting that these are special cases of Algorithm 14, and the validity of the 'marginal' algorithms (Particle-MALA, Particle-mGRAD, Particle-PCNL) by noting that these are special cases of Algorithm 15.

Proof (of Proposition 1). This follows by taking $\boldsymbol{\phi}_t(\mathbf{x}_{t-1:t}) := \kappa\frac{\delta_t}{2}\nabla_{\mathbf{x}_t}\log Q_t(\mathbf{x}_{t-1:t})$, $\boldsymbol{\psi}_t \equiv \mathbf{0}$, $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1};\mathbf{u}_{1:T}) = \mathrm{N}(\mathbf{x}_t;\mathbf{u}_t,\frac{\delta_t}{2}\mathbf{I})$ and $\mathbf{E}_t \equiv \frac{\delta_t}{2}\mathbf{I}$ in Proposition 14. □

Proof (of Proposition 2). This follows from Proposition 15 with the same setting as in Proposition 1. In particular, in this case, $\mathbf{v}_t \equiv \mathbf{0}$, $\mathbf{H}_t = \mathbf{I}$, $\mathbf{D}_t = \mathbf{E}_t = \frac{\delta_t}{2}\mathbf{I}$. Consequently, (1) then simplifies to (3.2), where we have used that $\mathbf{G}_t = [\frac{\delta_t}{2}(N+1)]^{-1}\mathbf{I} = \mathbf{D}_t^{-1}/(N+1)$ and $\mathbf{D}_t^{-1} - N\mathbf{G}_t = \mathbf{G}_t$. □

Proof (of Proposition 3). This follows in the same way as the proof of Proposition 1 except that now $\boldsymbol{\psi}_t(\mathbf{x}_{t:t+1}) = \kappa \frac{\delta_t}{2} \nabla_{\mathbf{x}_t} \log Q_{t+1}(\mathbf{x}_{t:t+1})$. □

Proof (of Proposition 4). This follows in the same way as the proof of Proposition 1 except that now $\boldsymbol{\phi}_t(\mathbf{x}_{t-1:t}) \coloneqq \kappa \frac{\delta_t}{2} \nabla_{\mathbf{x}_t} \log G_t(\mathbf{x}_{t-1:t})$, and $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{1:T}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t'(\mathbf{x}_{t-1}, \mathbf{u}_t), \mathbf{C}_t'(\mathbf{x}_{t-1}))$, where $\mathbf{m}_t'(\mathbf{x}_{t-1}, \mathbf{u}_t)$ and $\mathbf{C}_t'(\mathbf{x}_{t-1})$ are defined in (4.1) and (4.1). □

Proof (of Proposition 5). This follows from Proposition 15 with the same setting as in Proposition 4. In particular, in this case, $\mathbf{H}_t = \mathbf{A}_t \coloneqq (\mathbf{C}_t + \frac{2}{\delta_t}\mathbf{I})^{-1}\mathbf{C}_t$, $\mathbf{D}_t = \frac{\delta_t}{2}\mathbf{A}_t$ and $\mathbf{E}_t = \frac{\delta_t}{2}\mathbf{I}$. Consequently, (1) then simplifies to (1d), where we have used that $\mathbf{A}_t$ is symmetric, that $\mathbf{H}_t^{\mathrm{T}}\mathbf{D}_t^{-1} = \mathbf{D}_t^{-1}\mathbf{H}_t = \frac{2}{\delta_t}\mathbf{I}$ and hence

$$\mathbf{D}_t^{-1} - N\mathbf{G}_t = \mathbf{A}_t^{-1}\mathbf{G}_t = \mathbf{G}_t\mathbf{A}_t^{-1}.$$

This completes the proof. □

Proof (of Proposition 6). This follows in the same way as the proof of Proposition 4 except that now $\boldsymbol{\psi}_t(\mathbf{x}_{t:t+1}) = \kappa \frac{\delta_t}{2} \nabla_{\mathbf{x}_t} \log G_{t+1}(\mathbf{x}_{t:t+1})$. □

Proof (of Proposition 7). This follows in the same way as the proof of Propositions 4 and 6, respectively, but with $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{1:T}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}_t'\mathbf{x}_{t-1} + \mathbf{b}_t', \mathbf{C}_t')$. □

Proof (of Proposition 10). This follows in the same way as the proof of Proposition 1 except that now $\boldsymbol{\phi}_t(\mathbf{x}_{t-1:t}) \coloneqq \kappa \frac{\delta_t}{2} \widetilde{\mathbf{C}}_t(\mathbf{x}_{t-1}) \nabla_{\mathbf{x}_t} \log G_t(\mathbf{x}_{t-1:t})$, $\mathbf{E}_t(\mathbf{x}_{t-1:t}) \coloneqq \frac{\delta_t}{2}\mathbf{C}_t(\mathbf{x}_{t-1})$ and $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{1:T}) = \mathrm{N}(\mathbf{x}_t; \mathbf{m}_t'(\mathbf{x}_{t-1}, \mathbf{u}_t), \mathbf{C}_t'(\mathbf{x}_{t-1}))$, where $\mathbf{m}_t'(\mathbf{x}_{t-1}, \mathbf{u}_t)$ and $\mathbf{C}_t'(\mathbf{x}_{t-1})$ are defined in (A.1) and (A.1). □

Proof (of Proposition 11). This follows from Proposition 15 with the same setting as in Proposition 10. In particular, in this case, $\mathbf{H}_t = \beta_t\mathbf{I}$, $\mathbf{D}_t = (1-\beta_t)\mathbf{C}_t$ and $\mathbf{E}_t = \frac{\delta_t}{2}\mathbf{C}_t$. Consequently, (1) then simplifies to (A.2), where we have used that $\mathbf{E}\mathbf{H}_t^{\mathrm{T}}\mathbf{D}_t^{-1} = \mathbf{D}_t^{-1}\mathbf{H}_t\mathbf{E} = \mathbf{I}$ and hence

$$\begin{aligned}\mathbf{D}_t^{-1} + \mathbf{G}_t &= (\beta_t^{-1} + N + 1)\mathbf{G}_t,\\ \mathbf{D}_t^{-1} - N\mathbf{G}_t &= \beta_t^{-1}\mathbf{G}_t.\end{aligned}$$

This completes the proof. □

Proof (of Proposition 12). This follows in the same way as the proof of Proposition 10 except that now $\boldsymbol{\psi}_t(\mathbf{x}_{t:t+1}) = \kappa \frac{\delta_t}{2} \widetilde{\mathbf{C}}_t(\mathbf{x}_t) \nabla_{\mathbf{x}_t} \log G_{t+1}(\mathbf{x}_{t:t+1})$. □

Proof (of Proposition 13). This follows in the same way as the proof of Propositions 10 and 12, respectively, but with $M_t'(\mathbf{x}_t|\mathbf{x}_{t-1}; \mathbf{u}_{1:T}) = \mathrm{N}(\mathbf{x}_t; \mathbf{F}_t'\mathbf{x}_{t-1} + \mathbf{b}_t', \mathbf{C}_t')$. □

E Proof of Propositions 8 and 9

E.1 Preliminaries

For some given $N \in \mathbb{N}$, let Ψ^n denote either the *Boltzmann selection function* (with the convention $h^0 \coloneqq 0$):

$$\Psi^n(h^{1:N}) \coloneqq \frac{\exp(h^n)}{1 + \sum_{m=0}^{N} \exp(h^m)},$$

or the *Rosenbluth–Teller selection function*:

$$\Psi^n(h^{1:N}) := \begin{cases} \dfrac{\exp(h^n)}{1 + \sum_{m=1}^N \exp(h^m) - 1 \wedge \exp(h^n)}, & \text{if } n > 0, \\ 1 - \displaystyle\sum_{l=1}^N \Psi^l(h^{1:N}), & \text{if } n = 0. \end{cases}$$

In either case, Ψ^n is Lipschitz continuous with constant denoted $[\Psi^n]_{\text{LIP}}$.

E.2 Marginal MCMC kernels in the special case: $T = 1$

For the moment, we assume that $T = 1$. To simplify the notation, we drop the 'time' subscripts $t = 1$. With this convention, for some bounded and differentiable $G : \mathbb{R}^D \to (0, \infty)$, define

$$\pi(\mathbf{x}) \propto \mathrm{N}(\mathbf{x}; \mathbf{m}, \mathbf{C})G(\mathbf{x}).$$

The π-invariant Markov kernels induced by the (non-auxiliary variable based) algorithms discussed in this work can then be written as

$$P_a(\mathrm{d}\tilde{\mathbf{x}}|\mathbf{x}) = \sum_{l=0}^N \int_{\mathcal{X}^{N+2}} \delta_{\mathbf{x}}(\,\mathrm{d}\mathbf{x}^0) q_a^{-0}(\mathrm{d}\mathbf{x}^{-0}|\mathbf{x}^0)\Psi^l(\{h_a^n(\mathbf{x}^{0:N})\}_{n=1}^N)\delta_{\mathbf{x}^l}(\mathrm{d}\tilde{\mathbf{x}}),$$

where have appealed to symmetry to always place the reference 'path' in position 0, and with

$$\begin{aligned} h_a^n(\mathbf{x}^{0:N}) &:= \log q_a^{-n}(\mathbf{x}^{-n}|\mathbf{x}^n) - \log q_a^{-0}(\mathbf{x}^{-0}|\mathbf{x}^0), \\ q_a^{-n}(\mathbf{x}^{-n}|\mathbf{x}^n) &= \mathrm{N}(\mathbf{x}^{-n}; \mathbf{m}_a(\mathbf{x}^n), \mathbf{C}_a), \end{aligned}$$

where $\mathbf{m}_a(\mathbf{x}^n) \in \mathbb{R}^{ND}$ is a suitable mean vector (which may depend on $\mathbf{x}^n \in \mathbb{R}^D$), $\mathbf{C}_a \in \mathbb{R}^{(ND)\times(ND)}$ a suitable variance, and where we again slightly abuse notation to let $\mathbf{x}^{-n}$ represent both the tuple $(\mathbf{x}^0, \dots, \mathbf{x}^{n-1}, \mathbf{x}^{n+1}, \dots, \mathbf{x}^N)$ and its vectorised form

$$\mathbf{x}^{-n} := \mathrm{vec}(\mathbf{x}^{-n}) = \begin{bmatrix} \mathbf{x}^0 \\ \vdots \\ \mathbf{x}^{n-1} \\ \mathbf{x}^{n+1} \\ \vdots \\ \mathbf{x}^N \end{bmatrix} \in \mathbb{R}^{ND}.$$

Additionally, 'a' is a placeholder for 'CSMC', 'Particle-MALA', or 'Particle-mGRAD'. Specifically, by the developments from Section C (Lemma 3 and its proof), and recalling that the block matrix operator $\mathcal{M}_N$ was defined in (C.1),

$$\begin{aligned} \mathbf{m}_{\text{Particle-mGRAD}}(\mathbf{x}^n) &= \mathbf{1}_{N\times 1} \otimes [\mathbf{m} + \mathbf{A}(\mathbf{x}^n + \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m})], \\ \mathbf{C}_{\text{Particle-mGRAD}} &= \tfrac{\delta}{2}\mathcal{M}_N(\mathbf{A}, \mathbf{A}^2) = \tfrac{\delta}{2}[\mathbf{I}_N \otimes \mathbf{A} + \mathbf{1}_{N\times N} \otimes \mathbf{A}^2], \\ \mathbf{m}_{\text{CSMC}}(\mathbf{x}^n) &= \mathbf{1}_{N\times 1} \otimes \mathbf{m}, \\ \mathbf{C}_{\text{CSMC}} &= \mathcal{M}_N(\mathbf{C}, \mathbf{0}_{D\times D}) = \mathbf{I}_N \otimes \mathbf{C}, \\ \mathbf{m}_{\text{Particle-MALA}}(\mathbf{x}^n) &= \mathbf{1}_{N\times 1} \otimes [\mathbf{x}^n + \boldsymbol{\phi}(\mathbf{x}^n) + \boldsymbol{\varphi}(\mathbf{x}^n)], \\ \mathbf{C}_{\text{Particle-MALA}} &= \tfrac{\delta}{2}\mathcal{M}_N(\mathbf{I}, \mathbf{I}) = \tfrac{\delta}{2}[\mathbf{I}_{ND} + \mathbf{1}_{N\times N} \otimes \mathbf{I}], \end{aligned}$$

where $\boldsymbol{\phi}(\mathbf{x}) \coloneqq \kappa\frac{\delta}{2}\nabla \log G(\mathbf{x})$ and $\boldsymbol{\varphi}(\mathbf{x}) \coloneqq \kappa\frac{\delta}{2}\nabla \log M(\mathbf{x})$ and with $\mathbf{A} \coloneqq (\frac{\delta}{2}\mathbf{C}^{-1} + \mathbf{I})^{-1} = \mathbf{C}(\mathbf{C} + \frac{\delta}{2}\mathbf{I})^{-1} = (\mathbf{C} + \frac{\delta}{2}\mathbf{I})^{-1}\mathbf{C}$.

Key to our proofs will be the following bound which follows from the triangle inequality and a telescoping-sum decomposition (here: a and b are again placeholders which take values in $\{$CSMC, Particle-MALA, Particle-mGRAD$\}$):

$$\begin{aligned}
&\|P_a(\,\cdot\,|\mathbf{x}) - P_b(\,\cdot\,|\mathbf{x})\|_{\text{TV}} \\
&\quad\leq \|q_a^{-0}(\,\cdot\,|\mathbf{x}) - q_b^{-0}(\,\cdot\,|\mathbf{x})\|_{\text{TV}} \\
&\qquad + \sum_{l=0}^{N} \int_{\mathcal{X}^{N+1}} \delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}^0) q_a^{-0}(\mathrm{d}\mathbf{x}^{-0}|\mathbf{x}^0) |\Psi^l(\{h_a^n(\mathbf{x}^{0:N})\}_{n=1}^N) - \Psi^l(\{h_b^n(\mathbf{x}^{0:N})\}_{n=1}^N)| \\
&\quad\leq \sqrt{\mathrm{KL}(q_a^{-0}(\,\cdot\,|\mathbf{x}) \| q_b^{-0}(\,\cdot\,|\mathbf{x}))} \\
&\qquad + \sum_{l=0}^{N} [\Psi^l]_{\text{LIP}} \int_{\mathcal{X}^{N+1}} \delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}^0) q_a^{-0}(\mathrm{d}\mathbf{x}^{-0}|\mathbf{x}^0) \sum_{n=1}^{N} |h_a^n(\mathbf{x}^{0:N}) - h_b^n(\mathbf{x}^{0:N})| \\
&\quad\leq C\biggl[\sqrt{D_{a,b}^0(\mathbf{x})} + \sum_{n=0}^{N} D_{a,b}^n(\mathbf{x})\biggr].
\end{aligned} \tag{34}$$

Here, the penultimate line follows from Pinsker's inequality and the Lipschitz continuity of the selection function; $C \geq 0$ is some constant which may depend on these Lipschitz constants and N and D; for the last inequality, we have defined

$$D_{a,b}^n(\mathbf{x}) \coloneqq \int_{\mathcal{X}^{N+1}} \delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}^0) q_a^{-0}(\mathrm{d}\mathbf{x}^{-0}|\mathbf{x}^0) |\log q_a^{-n}(\mathbf{x}^{-n}|\mathbf{x}^n) - \log q_b^{-n}(\mathbf{x}^{-n}|\mathbf{x}^n)|. \tag{35}$$

E.3 Proofs of Part 1

Proof (of Part 1 of Proposition 8). By Assumption **A1**, the model factorises over time and so do the CSMC and Particle-mGRAD algorithms. Hence, without loss of generality, we prove the result in the case that $T = 1$ (and we drop the 'time' subscript $t = 1$ hereafter). Throughout the proof, we will also make repeated use of the fact that the eigenvalues of $\mathbf{A}_k$ are given by $(2\lambda_{k,d})/(2\lambda_{k,d} + \delta)$, for $d \in [D]$.

For $\varepsilon \geq 1$ the result is trivially true but meaningless. Fix $\varepsilon \in (0, 1)$.

$$F_k \coloneqq \Bigl\{\mathbf{x} \in \mathbb{R}^D \Bigm| \|\mathbf{x} - \mathbf{m}\|_2 \leq \lambda_k^{(1-\varepsilon)/2}\Bigr\},$$

denote a ball of radius $\lambda_k^{(1-\varepsilon)/2}$ around $\mathbf{m}$, for any $k \geq 1$. We then have $\pi_k(F_k) = (1 + H_k)^{-1}$, where, letting $F_k^{\mathrm{c}} \coloneqq \mathcal{X} \setminus F_k$:

$$\begin{aligned}
H_k &\coloneqq \frac{\int_{F_k^{\mathrm{c}}} G(\mathbf{x})\, \mathrm{N}(\mathrm{d}\mathbf{x}; \mathbf{m}, \mathbf{C}_k)}{\int_{F_k} G(\mathbf{x})\, \mathrm{N}(\mathrm{d}\mathbf{x}; \mathbf{m}, \mathbf{C}_k)} \\
&\leq \frac{\sup_{x \in \mathcal{X}} G(\mathbf{x})}{\inf_{\mathbf{x} \in F_k} G(\mathbf{x})} \frac{\int_{F_k^{\mathrm{c}}} \mathrm{N}(\mathrm{d}\mathbf{x}; \mathbf{m}, \mathbf{C}_k)}{\int_{F_k} \mathrm{N}(\mathrm{d}\mathbf{x}; \mathbf{m}, \mathbf{C}_k)} \\
&\leq \frac{\sup_{x \in \mathcal{X}} G(\mathbf{x})}{\inf_{\mathbf{x} \in F_k} G(\mathbf{x})} \frac{\int_{\mathcal{X}} \mathrm{N}(\mathrm{d}\mathbf{x}; \mathbf{0}, \mathbf{I})\, \mathbb{I}\{\|\mathbf{x}\|_2 > \lambda_k^{-\varepsilon/2}\}}{\int_{\mathcal{X}} \mathrm{N}(\mathrm{d}\mathbf{x}; \mathbf{0}, \mathbf{I})\, \mathbb{I}\{\|\mathbf{x}\|_2 \leq \lambda_k^{-\varepsilon/2}\}} \\
&\to 0,
\end{aligned}$$

as $k \to \infty$, where we have used that G is bounded and that $(\inf_{\mathbf{x} \in F_k} G(\mathbf{x}))_{k \geq 1}$ is an increasing sequence in $(0, \infty)$ (since $(F_k)_{k \geq 1}$ is decreasing and F_1 is compact).

By the decomposition from (E.2), all that remains is to control the terms

$$\sup_{\mathbf{x}^0 \in F_k} D^n_{\text{CSMC,Particle-mGRAD},k}(\mathbf{x}^0),$$

for arbitrary $n \in [N]_0$.

Firstly, by Lemma 1 from Appendix C, letting $\lambda(\mathbf{C}_k) = \{\lambda_{k,1}, \dots, \lambda_{k,D}\}$ denote the eigenvalues of $\mathbf{C}_k$ and noting that $\mathbf{A}_k$ is simultaneously diagonalisable with $\mathbf{A}_k^2$:

$$\begin{aligned}
&|\log(\det(\mathbf{C}_{\text{CSMC},k})) - \log(\det(\mathbf{C}_{\text{Particle-mGRAD},k}))| \\
&\quad= \left|\sum_{d=1}^{D} N \log \lambda_{k,d} - (N-1)\log\left(\frac{\delta\lambda_{k,d}}{2\lambda_{k,d}+\delta}\right) - \log\left(\frac{\delta\lambda_{k,d}}{2\lambda_{k,d}+\delta} + \frac{2\delta N \lambda_{k,d}^2}{(2\lambda_{k,d}+\delta)^2}\right)\right| \\
&\quad= \left|\sum_{d=1}^{D} N \log\left(\frac{2\lambda_{k,d}+\delta}{\delta}\right) + \log\left(\frac{\delta\lambda_{k,d}}{2\lambda_{k,d}+\delta}\right) - \log\left(\frac{\delta\lambda_{k,d}}{2\lambda_{k,d}+\delta} + \frac{2\delta N \lambda_{k,d}^2}{(2\lambda_{k,d}+\delta)^2}\right)\right| \\
&\quad= \left|\sum_{d=1}^{D} N \log\left(\frac{2\lambda_{k,d}+\delta}{\delta}\right) + \log\left(\frac{2\lambda_{k,d}+\delta}{2\lambda_{k,d}(N+1)+\delta}\right)\right| \in \mathrm{O}(\lambda_k).
\end{aligned} \tag{36}$$

Secondly, by Lemma 2 from Appendix C,

$$\mathbf{C}^{-1}_{\text{Particle-mGRAD},k} = \frac{2}{\delta}\mathcal{M}_N(\mathbf{A}_k^{-1}, -(\mathbf{I} + N\mathbf{A}_k)^{-1}),$$

and with the conventions that the sum symbol $\sum_i$ is shorthand for $\sum_{i \in [N]_0 \setminus \{n\}}$, that $\sum_j$ is shorthand for $\sum_{j \in [N]_0 \setminus \{n\}}$, that $\sum_{i \neq j}$ is shorthand for $\sum_{j \in [N]_0 \setminus \{n,i\}}$, and again writing $\boldsymbol{\phi}(\mathbf{x}) =$

$\kappa \frac{\delta}{2} \nabla \log G(\mathbf{x})$ we obtain:

$$\begin{aligned}
&\Big|(\mathbf{x}^{-n} - \mathbf{m}_{\text{CSMC}}(\mathbf{x}^n))^{\mathrm{T}} \mathbf{C}_{\text{CSMC},k}^{-1} (\mathbf{x}^{-n} - \mathbf{m}_{\text{CSMC}}(\mathbf{x}^n)) \\
&\quad - (\mathbf{x}^{-n} - \mathbf{m}_{\text{Particle-mGRAD},k}(\mathbf{x}^n))^{\mathrm{T}} \mathbf{C}_{\text{Particle-mGRAD},k}^{-1} (\mathbf{x}^{-n} - \mathbf{m}_{\text{Particle-mGRAD},k}(\mathbf{x}^n))\Big| \\
&\qquad = \Big| \sum_i (\mathbf{x}^i - \mathbf{m})^{\mathrm{T}} \mathbf{C}_k^{-1} (\mathbf{x}^i - \mathbf{m}) \\
&\qquad\quad - \frac{2}{\delta} \sum_i (\mathbf{x}^i - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m})^{\mathrm{T}} \mathbf{A}_k^{-1} (\mathbf{x}^i - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m}) \\
&\qquad\quad + \frac{2}{\delta} \sum_i \sum_j (\mathbf{x}^i - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m})^{\mathrm{T}} (\mathbf{I} + N\mathbf{A}_k)^{-1} (\mathbf{x}^j - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m}) \Big| \\
&\qquad = \Big| \frac{2}{\delta} \sum_i \sum_j (\mathbf{x}^i - \mathbf{m})^{\mathrm{T}} (\mathbf{I} + N\mathbf{A}_k)^{-1} (\mathbf{x}^j - \mathbf{m}) \\
&\qquad\quad + \frac{2}{\delta} \sum_i (\mathbf{x}^i - \mathbf{m})^{\mathrm{T}} \Big(\frac{\delta}{2} \mathbf{C}_k^{-1} - \mathbf{A}_k^{-1}\Big) (\mathbf{x}^i - \mathbf{m}) \\
&\qquad\quad + \frac{2}{\delta} \sum_i (\mathbf{x}^i - \mathbf{m})^{\mathrm{T}} \Big[\mathbf{I} + \Big(\frac{4(N-1)}{\delta} - 1\Big)(\mathbf{I} + N\mathbf{A}_k)^{-1} \mathbf{A}_k\Big] (\mathbf{x}^n - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m}) \\
&\qquad\quad + N(\mathbf{x}^n - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m})^{\mathrm{T}} \Big[\mathbf{A}_k + \Big(\frac{2(N-1)}{\delta} - 1\Big) \mathbf{A}_k (\mathbf{I} + N\mathbf{A}_k)^{-1} \mathbf{A}_k\Big] (\mathbf{x}^n - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m}) \Big| \\
&\qquad \leq \frac{2}{\delta} \sum_i \sum_j \|\mathbf{x}^i - \mathbf{m}\|_2 \|\mathbf{x}^j - \mathbf{m}\|_2 \|(\mathbf{I} + N\mathbf{A}_k)^{-1}\|_{2,2} \\
&\qquad\quad + \frac{2}{\delta} \sum_i \|\mathbf{x}^i - \mathbf{m}\|_2 \|\mathbf{x}^i - \mathbf{m}\|_2 \Big\|\frac{\delta}{2} \mathbf{C}_k^{-1} - \mathbf{A}_k^{-1}\Big\|_{2,2} \\
&\qquad\quad + \frac{2}{\delta} \sum_i \|\mathbf{x}^i - \mathbf{m}\|_2 \|\mathbf{x}^n - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m}\|_2 \Big\|\mathbf{I} + \Big(\frac{4(N-1)}{\delta} - 1\Big)(\mathbf{I} + N\mathbf{A}_k)^{-1} \mathbf{A}_k\Big\|_{2,2} \\
&\qquad\quad + N \|\mathbf{x}^n - \boldsymbol{\phi}(\mathbf{x}^n) - \mathbf{m}\|_2^2 \Big\|\mathbf{A}_k + \Big(\frac{2(N-1)}{\delta} - 1\Big) \mathbf{A}_k (\mathbf{I} + N\mathbf{A}_k)^{-1} \mathbf{A}_k\Big\|_{2,2} \\
&\qquad \leq C \Big[\sum_i \sum_j \|\mathbf{x}^i - \mathbf{m}\|_2 \|\mathbf{x}^j - \mathbf{m}\|_2 \\
&\qquad\qquad + (1 + \|\mathbf{x}^n - \mathbf{m}\|_2) \sum_i \|\mathbf{x}^i - \mathbf{m}\|_2 \\
&\qquad\qquad + \lambda_k (1 + \|\mathbf{x}^n - \mathbf{m}\|_2)^2\Big],
\end{aligned} \tag{37}$$

for some constant $C \geq 0$ which only depends on N, δ and $\mathbf{m}$. Here, we have used that all the matrices inside the operator norms are simultaneously diagonalisable with $\mathbf{C}_k$ (so that the operator norms can be bounded above by some function of λ_k):

$$\begin{aligned}
\|(\mathbf{I} + N\mathbf{A}_k)^{-1}\|_{2,2} &\leq 1, \\
\Big\|\frac{\delta}{2} \mathbf{C}_k^{-1} - \mathbf{A}_k^{-1}\Big\|_{2,2} &= 1, \\
\Big\|\mathbf{A}_k + \Big(\frac{2(N-1)}{\delta} - 1\Big) \mathbf{A}_k (\mathbf{I} + N\mathbf{A}_k)^{-1} \mathbf{A}_k\Big\|_{2,2} &\leq C' \frac{2\lambda_k}{2\lambda_k + \delta} \leq C' \frac{2}{\delta} \lambda_k, \\
\Big\|\mathbf{I} + \Big(\frac{4(N-1)}{\delta} - 1\Big)(\mathbf{I} + N\mathbf{A}_k)^{-1} \mathbf{A}_k\Big\|_{2,2} &\leq C'',
\end{aligned}$$

for other constants $C', C'' \geq 0$.

Furthermore, by definition of $(F_k)_{k \geq 1}$,

$$\sup_{\mathbf{x} \in F_k} \|\mathbf{x} - \mathbf{m}\|_2 \in \mathrm{O}(\lambda_k^{(1-\varepsilon)/2}).$$

Consequently, for $i, j \in [N]_0$:

$$\begin{aligned} &\sup_{\mathbf{x}^0 \in F_k} \int_{\mathcal{X}^N} \mathrm{N}(\mathrm{d}\mathbf{x}^{-0}; \mathbf{m}_{\mathrm{CSMC}}, \mathbf{C}_{\mathrm{CSMC},k}) \|\mathbf{x}^i - \mathbf{m}\|_2 \|\mathbf{x}^j - \mathbf{m}\|_2 \\ &\in \begin{cases} \mathrm{O}(\lambda_k^{(1-\varepsilon)}), & \text{if } i = j = 0, \\ \mathrm{O}(\lambda_k^{(2-\varepsilon)/2}), & \text{if either } i = 0 \text{ or } j = 0, \\ \mathrm{O}(\lambda_k), & \text{if neither } i = 0 \text{ nor } j = 0, \end{cases} \end{aligned} \tag{38}$$

as $\lambda_k \to 0$, and where the last two cases follow from the Cauchy–Schwarz inequality. Similarly, for $i \in [N]_0$,

$$\sup_{\mathbf{x}^0 \in F_k} \int_{\mathcal{X}^N} \mathrm{N}(\mathrm{d}\mathbf{x}^{-0}; \mathbf{m}_{\mathrm{CSMC}}, \mathbf{C}_{\mathrm{CSMC},k}) \|\mathbf{x}^i - \mathbf{m}\|_2 \in \begin{cases} \mathrm{O}(\lambda_k^{(1-\varepsilon)/2}), & \text{if } i = 0, \\ \mathrm{O}(\lambda_k^{1/2}), & \text{if } i \neq 0, \end{cases} \tag{39}$$

as $\lambda_k \to 0$. Combining the bounds from (18)–(18) then shows that

$$\sup_{\mathbf{x}^0 \in F_k} D^n_{\mathrm{CSMC,Particle\text{-}mGRAD},k}(\mathbf{x}^0) \in \mathrm{O}(\lambda_k^{(1-\varepsilon)/2}),$$

for any $n \in [N]_0$. Plugging these bounds into (E.2) completes the proof. □

Proof (of Part 1 of Proposition 9). By Assumption **A1**, the model factorises over time and so do the Particle-MALA and Particle-mGRAD algorithms. Hence, without loss of generality, we again only prove the result in the case that $T = 1$ (and we again drop the 'time' subscript $t = 1$ hereafter).

For $\varepsilon \geq 1$ the result is trivially true but meaningless. Fix $\varepsilon \in (0, 1)$. Since G is integrable (by Assumption **A3**) and since π_k is invariant to scaling of G by a positive constant factor, we assume that $\int_{\mathcal{X}} G(\mathbf{x})\, \mathrm{d}\mathbf{x} = 1$, without loss of generality, so that G can be viewed as a density (and we will also use the symbol G to denote the corresponding distribution). Let $\mathbf{m}_G$ and $\mathbf{C}_G$ be mean and variance of G (which exist by Assumption **A3**) and define

$$F_k \coloneqq \left\{ \mathbf{x} \in \mathcal{X} \,\middle|\, \|\mathbf{x} - \mathbf{m}\|_2 \vee \sqrt{(\mathbf{x} - \mathbf{m}_G)^{\mathrm{T}} \mathbf{C}_G^{-1} (\mathbf{x} - \mathbf{m}_G)} < \lambda_k^{\varepsilon/2} \right\}.$$

We then have $\pi_k(F_k) = (1 + H_k)^{-1}$, where, letting $\mathbf{Y} \sim G$ and letting $F_k^{\mathrm{c}} := \mathcal{X} \setminus F_k$:

$$\begin{aligned}
H_k &:= \frac{\int_{F_k^{\mathrm{c}}} G(\mathbf{x})\, \mathrm{N}(\mathrm{d}\mathbf{x}; \mathbf{m}, \mathbf{C}_k)}{\int_{F_k} G(\mathbf{x})\, \mathrm{N}(\mathrm{d}\mathbf{x}; \mathbf{m}, \mathbf{C}_k)} \\
&\leq \frac{\int_{F_k^{\mathrm{c}}} G(\mathbf{x})\, \mathrm{d}\mathbf{x}}{\inf_{\mathbf{x} \in F_k} \exp(-\frac{1}{2}\|\mathbf{x} - \mathbf{m}\|_2^2 \lambda_k^{-1}) \int_{F_k} G(\mathbf{x})\, \mathrm{d}\mathbf{x}} \\
&\leq \frac{\int_{F_k^{\mathrm{c}}} G(\mathbf{x})\, \mathrm{d}\mathbf{x}}{\int_{F_k} G(\mathbf{x})\, \mathrm{d}\mathbf{x}} \exp(\tfrac{1}{2}\lambda_k^{\varepsilon - 1}) \\
&= \mathbb{P}(\mathbf{Y} \in F_k^{\mathrm{c}}) \frac{\exp(\frac{1}{2}\lambda_k^{\varepsilon - 1})}{\int_{F_k} G(\mathbf{x})\, \mathrm{d}\mathbf{x}} \\
&\leq \mathbb{P}\Big(\sqrt{(\mathbf{Y} - \mathbf{m}_G)^{\mathrm{T}} \mathbf{C}_G^{-1} (\mathbf{Y} - \mathbf{m}_G)} \geq \lambda_k^{\varepsilon/2}\Big) \frac{\exp(\frac{1}{2}\lambda_k^{\varepsilon - 1})}{\int_{F_k} G(\mathbf{x})\, \mathrm{d}\mathbf{x}} \\
&\leq \frac{D}{\lambda_k^{\varepsilon}} \frac{\exp(\frac{1}{2}\lambda_k^{\varepsilon - 1})}{\int_{F_k} G(\mathbf{x})\, \mathrm{d}\mathbf{x}} \\
&\to 0.
\end{aligned}$$

The penultimate line follows from the (multidimensional) Chebyshev's inequality and the last line uses that $F_k \to \mathcal{X}$ as $k \to \infty$.

By the decomposition from (E.2), all that remains is to control the terms

$$\sup_{\mathbf{x}^0 \in F_k} D^n_{\text{Particle-MALA},\text{Particle-mGRAD},k}(\mathbf{x}^0),$$

for arbitrary $n \in [N]_0$

Firstly, by Lemma 1 from Appendix C, letting $\lambda(\mathbf{C}_k) = \{\lambda_{k,1}, \dots, \lambda_{k,D}\}$ denote the eigenvalues of $\mathbf{C}_k$ and noting that $\mathbf{A}_k$ is simultaneously diagonalisable with $\mathbf{A}_k^2$:

$$\begin{aligned}
&|\log(\det(\mathbf{C}_{\text{Particle-MALA}})) - \log(\det(\mathbf{C}_{\text{Particle-mGRAD},k}))| \\
&\quad = \left|\sum_{d=1}^{D} N \log\Big(\frac{2\lambda_{k,d} + \delta}{2\lambda_{k,d}}\Big) + \log\Big(\frac{2\lambda_{k,d} + \delta}{2\lambda_{k,d} + \delta/(N+1)}\Big)\right| \in \mathrm{O}(\lambda_k^{-1}).
\end{aligned} \tag{40}$$

Secondly, by Lemma 2 from Appendix C, and again with the conventions that $\sum_i$ is shorthand for $\sum_{i \in [N]_0 \setminus \{n\}}$, that $\sum_j$ is shorthand for $\sum_{j \in [N]_0 \setminus \{n\}}$, that $\sum_{i \neq j}$ is shorthand for $\sum_{j \in [N]_0 \setminus \{n,i\}}$, and writing $\boldsymbol{\phi}(\mathbf{x}) := \kappa \frac{\delta}{2} \nabla \log G(\mathbf{x})$ as well as $\boldsymbol{\varphi}_k(\mathbf{x}) := \kappa \frac{\delta}{2} \nabla \log M_k(\mathbf{x}) = \kappa \frac{\delta}{2} \mathbf{C}_k^{-1}(\mathbf{m} - \mathbf{x})$, so that $\boldsymbol{\phi}(\mathbf{x}) + \boldsymbol{\varphi}_k(\mathbf{x}) = \kappa \frac{\delta}{2} \nabla \log \pi_k(\mathbf{x})$:

$$\begin{aligned}
&\Big|(\mathbf{x}^{-n} - \mathbf{m}_{\text{Particle-MALA},k}(\mathbf{x}^n))^{\mathrm{T}} \mathbf{C}_{\text{Particle-MALA}}^{-1} (\mathbf{x}^{-n} - \mathbf{m}_{\text{Particle-MALA},k}(\mathbf{x}^n)) \\
&\quad - (\mathbf{x}^{-n} - \mathbf{m}_{\text{Particle-mGRAD},k}(\mathbf{x}^n))^{\mathrm{T}} \mathbf{C}_{\text{Particle-mGRAD},k}^{-1} (\mathbf{x}^{-n} - \mathbf{m}_{\text{Particle-mGRAD},k}(\mathbf{x}^n))\Big| \\
&\qquad \leq C\lambda_k^{-1}\Big[\sum_i \sum_j \|\mathbf{x}^i - \mathbf{m}\|_2 \|\mathbf{x}^j - \mathbf{m}\|_2 + (1 + \|\mathbf{x}^n - \mathbf{m}\|_2) \sum_{i=0}^{N} \|\mathbf{x}^i - \mathbf{m}\|_2\Big],
\end{aligned} \tag{41}$$

for some constant $C \geq 0$ which only depends on N, δ and $\mathbf{m}$. Here, we have followed the same steps as for (18) and used that all the matrices inside the operator norms are simultaneously diagonalisable with $\mathbf{C}_k$ (so that the operator norms can be bounded above by some function of λ_k^{-1}).

Furthermore, by definition of F_k, we have

$$\sup_{\mathbf{x}\in F_k} \|\mathbf{x}-\mathbf{m}\|_2 \in \mathrm{O}(\lambda_k^{\varepsilon/2}),$$

as $\lambda_k \to \infty$. Consequently, for $i, j \in [N]_0$, by the Cauchy–Schwarz inequality:

$$\begin{aligned}
&\sup_{\mathbf{x}^0\in F_k} \int_{\mathcal{X}^N} \mathrm{N}(\mathrm{d}\mathbf{x}^{-0}; \mathbf{m}_{\text{Particle-MALA},k}, \mathbf{C}_{\text{Particle-MALA}}) \|\mathbf{x}^i-\mathbf{m}\|_2 \|\mathbf{x}^j-\mathbf{m}\|_2 \\
&\qquad \in \begin{cases} \mathrm{O}(\lambda_k^{\varepsilon}), & \text{if } i=j=0,\\ \mathrm{O}(\lambda_k^{\varepsilon/2}), & \text{if either } i=0 \text{ or } j=0,\\ \mathrm{O}(1), & \text{if neither } i=0 \text{ nor } j=0,\end{cases}
\end{aligned} \tag{42}$$

as $\lambda_k \to \infty$. Similarly, for $i \in [N]_0$,

$$\sup_{\mathbf{x}^0\in F_k} \int_{\mathcal{X}^N} \mathrm{N}(\mathrm{d}\mathbf{x}^{-0}; \mathbf{m}_{\text{Particle-MALA},k}, \mathbf{C}_{\text{Particle-MALA}}) \|\mathbf{x}^i-\mathbf{m}\|_2 \in \begin{cases} \mathrm{O}(\lambda_k^{\varepsilon/2}), & \text{if } i=0,\\ \mathrm{O}(1), & \text{if } i\neq 0,\end{cases} \tag{43}$$

as $\lambda_k \to \infty$. Combining the bounds from (19)–(19) then shows that

$$\sup_{\mathbf{x}^0\in F_k} D^n_{\text{Particle-MALA},\text{Particle-mGRAD},k}(\mathbf{x}^0) \in \mathrm{O}(\lambda_k^{-(1-\varepsilon)/2}),$$

for any $n \in [N]_0$. Plugging these bounds into (E.2) completes the proof. □

E.4 Auxiliary MCMC kernels in the special case: $T = 1$

The π-invariant Markov kernels induced by the auxiliary-variable based algorithms discussed in this work can then be written as

$$P_a(\mathrm{d}\tilde{\mathbf{x}}|\mathbf{x}) = \sum_{l=0}^{N} \int_{\mathcal{X}^{N+3}} \delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}^0) q_a^{-0}(\mathrm{d}\mathbf{x}^{-0} \times \mathrm{d}\mathbf{u}|\mathbf{x}^{-0}, \mathbf{x}^0) \Psi^l(\{h_a^n(\mathbf{x}^{0:N}, \mathbf{u})\}_{n=1}^N) \delta_{\mathbf{x}^l}(\mathrm{d}\tilde{\mathbf{x}}),$$

where have appealed to symmetry to always place the reference 'path' in position 0, where 'a' is now a placeholder for 'Particle-aGRAD', 'Particle-aMALA', or 'CSMC' and with

$$\begin{aligned}
q_a^{-n}(\mathbf{x}^{-n}, \mathbf{u}|\mathbf{x}^n) &= q_a^{-n}(\mathbf{x}^{-n}|\mathbf{x}^n) q_a^{-n}(\mathbf{u}|\mathbf{x}^{-n}, \mathbf{x}^n),\\
h_a^n(\mathbf{x}^{0:N}, \mathbf{u}) &:= \log q_a^{-n}(\mathbf{x}^{-n}|\mathbf{x}^n) - \log q_a^{-0}(\mathbf{x}^{-0}|\mathbf{x}^0)\\
&\qquad + \mathbb{I}\{a \neq \text{CSMC}\}[\log q_a^{-n}(\mathbf{u}|\mathbf{x}^{-n}, \mathbf{x}^n) - \log q_a^{-0}(\mathbf{u}|\mathbf{x}^{-0}, \mathbf{x}^0)],\\
q_a^{-n}(\mathbf{x}^{-n}|\mathbf{x}^n) &= \mathrm{N}(\mathbf{x}^{-n}; \mathbf{m}_a(\mathbf{x}^n), \mathbf{C}_a),\\
q_a^{-n}(\mathbf{u}|\mathbf{x}^{-n}, \mathbf{x}^n) &= \mathrm{N}(\mathbf{u}; \boldsymbol{\nu}_a(\mathbf{x}^{-n}, \mathbf{x}^n), \boldsymbol{\Upsilon}_a),
\end{aligned}$$

where again $\mathbf{m}_a(\mathbf{x}^n) \in \mathbb{R}^{ND}$ and $\boldsymbol{\nu}_a(\mathbf{x}^{-n}, \mathbf{x}^n) \in \mathbb{R}^D$ are suitable mean vector, and $\mathbf{C}_a \in \mathbb{R}^{(ND)\times(ND)}$, $\boldsymbol{\Upsilon}_a \in \mathbb{R}^{D\times D}$ are suitable covariance variance matrices, and we again write $\mathbf{x}^{-n} := \mathrm{vec}(\mathbf{x}^{-n})$. Specifically,

$$\begin{aligned}
\mathbf{m}_{\text{Particle-aGRAD}}(\mathbf{x}^n) &= \mathbf{m}_{\text{Particle-mGRAD}}(\mathbf{x}^n)\\
\mathbf{C}_{\text{Particle-aGRAD}} &= \mathbf{C}_{\text{Particle-mGRAD}},\\
\boldsymbol{\nu}_{\text{Particle-aGRAD}}(\mathbf{x}^{-n}, \mathbf{x}^n) &= (\mathbf{I} + N\mathbf{A})^{-1}[(N+1)\bar{\mathbf{x}} + \boldsymbol{\phi}(\mathbf{x}^n) + N(\mathbf{A} - \mathbf{I})\mathbf{m}],\\
\boldsymbol{\Upsilon}_{\text{Particle-aGRAD}} &= \tfrac{\delta}{2}(\mathbf{I} + N\mathbf{A})^{-1},\\
\mathbf{m}_{\text{Particle-aMALA}}(\mathbf{x}^n) &= \mathbf{m}_{\text{Particle-aMALA}}(\mathbf{x}^n),\\
\mathbf{C}_{\text{Particle-MALA}} &= \mathbf{C}_{\text{Particle-MALA}},\\
\boldsymbol{\nu}_{\text{Particle-aMALA}}(\mathbf{x}^{-n}, \mathbf{x}^n) &= \bar{\mathbf{x}} + \tfrac{1}{N+1}\boldsymbol{\phi}(\mathbf{x}^n),\\
\boldsymbol{\Upsilon}_{\text{Particle-aMALA}} &= \tfrac{\delta}{2(N+1)}\mathbf{I},
\end{aligned}$$

by Part 2 of Lemma 3 and Lemma 2 from Appendix C. Of course, the standard CSMC algorithm does not make use of the auxiliary variable $\mathbf{u}$, so we extend the space to include $\mathbf{u}$ with

$$\boldsymbol{\nu}_{\text{CSMC}}(\mathbf{x}^{-n}, \mathbf{x}^n) = \boldsymbol{\nu}_{\text{Particle-aGRAD}}(\mathbf{x}^{-n}, \mathbf{x}^n), \tag{44}$$

$$\boldsymbol{\Upsilon}_{\text{CSMC}} = \boldsymbol{\Upsilon}_{\text{Particle-aGRAD}}. \tag{45}$$

Key to our proofs will be the following bound which follows by the triangle inequality and a telescoping-sum decomposition (here: a is again a placeholder which takes values in $\{\text{CSMC}, \text{Particle-MALA}\}$ whilst we will always set $b = \text{Particle-aGRAD}$ and $q_a^{-m}(\,\cdot\,|\mathbf{x})$; and, unless otherwise stated, $q_b^{-m}(\,\cdot\,|\mathbf{x})$ denote the joint distributions on the space that includes the auxiliary variable $\mathbf{u}$):

$$\begin{aligned}
&\|P_a(\,\cdot\,|\mathbf{x}) - P_b(\,\cdot\,|\mathbf{x})\|_{\text{TV}} \\
&\quad\leq \|q_a^{-0}(\,\cdot\,|\mathbf{x}) - q_b^{-0}(\,\cdot\,|\mathbf{x})\|_{\text{TV}} \\
&\qquad + \sum_{l=0}^{N} \int_{\mathcal{X}^{N+2}} \delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}^0) q_a^{-0}(\mathrm{d}\mathbf{x}^{-0} \times \mathrm{d}\mathbf{u}|\mathbf{x}^0) \\
&\qquad\qquad \times |\Psi^l(\{h_a^n(\mathbf{x}^{0:N}, \mathbf{u})\}_{n=1}^N) - \Psi^l(\{h_b^n(\mathbf{x}^{0:N}, \mathbf{u})\}_{n=1}^N)| \\
&\quad\leq \sqrt{\text{KL}(q_a^{-0}(\,\cdot\,|\mathbf{x})\|q_b^{-0}(\,\cdot\,|\mathbf{x}))} \\
&\qquad + \sum_{l=0}^{N} [\Psi^l]_{\text{LIP}} \int_{\mathcal{X}^{N+2}} \delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}^0) q_a^{-0}(\mathrm{d}\mathbf{x}^{-0} \times \mathrm{d}\mathbf{u}|\mathbf{x}^0) \sum_{n=1}^{N} |h_a^n(\mathbf{x}^{0:N}, \mathbf{u}) - h_b^n(\mathbf{x}^{0:N}, \mathbf{u})| \\
&\quad\leq C\Bigl[\sqrt{D_{a,b}^0(\mathbf{x}) + E_{a,b}^0(\mathbf{x})} + \sum_{n=0}^{N} D_{a,b}^n(\mathbf{x}) + \widetilde{E}_{a,b}^n(\mathbf{x})\Bigr].
\end{aligned}$$

Here, the penultimate line follows from Pinsker's inequality and the Lipschitz continuity of the selection function; $C \geq 0$ is some constant which may depend on these Lipschitz constants and on N and D; $D_{a,b}^n(\mathbf{x})$ is defined exactly as in the marginal case (E.2). Furthermore, we have defined

$$E_{a,b}^n(\mathbf{x}) \coloneqq \int_{\mathcal{X}^{N+1}} \delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}^0) q_a^{-0}(\mathrm{d}\mathbf{x}^{-0} \times \mathrm{d}\mathbf{u}|\mathbf{x}^0) |\log q_a^{-n}(\mathbf{u}|\mathbf{x}^{-n}, \mathbf{x}^n) - \log q_b^{-n}(\mathbf{u}|\mathbf{x}^{-n}, \mathbf{x}^n)|. \tag{46}$$

Finally, if $a \neq \text{CSMC}$ and $b \neq \text{CSMC}$, we we have defined

$$\widetilde{E}_{\text{Particle-aMALA,Particle-aGRAD}}^n(\mathbf{x}) \coloneqq E_{\text{Particle-aMALA,Particle-aGRAD}}^n(\mathbf{x}),$$

whilst

$$\begin{aligned}
&\widetilde{E}_{\text{CSMC,Particle-aGRAD}}^n(\mathbf{x}) \\
&\quad\coloneqq \int_{\mathcal{X}^{N+2}} \delta_{\mathbf{x}}(\mathrm{d}\mathbf{x}^0) q_{\text{CSMC}}^{-0}(\mathrm{d}\mathbf{x}^{-0} \times \mathrm{d}\mathbf{u}|\mathbf{x}^0) \\
&\qquad \times |\log q_{\text{Particle-aGRAD}}^{-n}(\mathbf{u}|\mathbf{x}^{-n}, \mathbf{x}^n) - \log q_{\text{Particle-aGRAD}}^{-0}(\mathbf{u}|\mathbf{x}^{-0}, \mathbf{x}^0)|.
\end{aligned}$$

E.5 Proofs of Part 2

Proof (of Part 2 of Proposition 8). Assume the same setting as in the proof of Part 1 of Proposition 8 with the same definition of F_k.

We proceed by controlling the terms in (E.4). Note that $D_{\text{CSMC,Particle-aGRAD},k}^n = D_{\text{CSMC,Particle-mGRAD},k}^n$. Hence, by the arguments from the proof of Part 1 of Proposition 8,

$$\sup_{\mathbf{x} \in F_k} D_{\text{CSMC,Particle-aGRAD},k}^n(\mathbf{x}) \in \mathrm{O}(\lambda_k^{(1-\varepsilon)/2}).$$

Additionally, due to (E.4)–(E.4), $E^n_{\text{CSMC,Particle-aGRAD}}(\mathbf{x}) = 0$. Finally, using similar arguments as in the proofs for the 'marginal' algorithm, we can verify that

$$\sup_{\mathbf{x}\in F_k} \widetilde{E}^n_{\text{CSMC,Particle-aGRAD},k}(\mathbf{x}) \in \mathrm{O}(\lambda_k^{(1-\varepsilon)/2}).$$

This completes the proof. □

Proof (of Part 2 of Proposition 9). Assume the same setting as in the proof ofPart 1 of Proposition 9 with the same definition of F_k.

We proceed by controlling the terms in (E.4). Note that $D^n_{\text{Particle-aMALA,Particle-aGRAD},k} = D^n_{\text{Particle-MALA,Particle-mGRAD},k}$. Hence, by the arguments from the proof Part 1 of Proposition 9,

$$\sup_{\mathbf{x}\in F_k} D^n_{\text{Particle-aMALA,Particle-aGRAD},k}(\mathbf{x}) \in \mathrm{O}(\lambda_k^{-(1-\varepsilon)/2}).$$

Additionally, using similar arguments as in the proofs for the 'marginal' algorithm, we can verify that

$$\sup_{\mathbf{x}\in F_k} E^n_{\text{Particle-aMALA,Particle-aGRAD},k}(\mathbf{x}) \in \mathrm{O}(\lambda_k^{-(1-\varepsilon)}).$$

This completes the proof. □

F Step-size adaptation

All our algorithms involve the calibration of several step sizes δ_t, one for each time step. To calibrate these, we implement a routine that recursively increases or decreases δ_t if the running average of the acceptance rate α_t (i.e., the relative frequency with which $\mathbf{x}_t$ is updated) is respectively above or below a pre-specified target acceptance rate (in our experiments, we picked this to be $\alpha^* = 75\,\%$). The only exception to this lies in the twisted algorithms of Section 4.4 which we calibrate using a single step-size δ (so that $\delta = \delta_1 = \ldots = \delta_T$), and for which the target relates to the overall acceptance rate averaged across time steps. The reason for this difference stems from the fact that the twisting causes the acceptance rate at time s additionally depend on *future* auxiliary variables $\mathbf{u}_t$, and therefore the future step-size parameters δ_t (for $t > s$), thereby making the behaviour per time-step harder to control. In practice, our calibration of the twisted Particle-aGRAD is therefore more similar to that of aGRAD than that of our other algorithms. The adaptation procedure is summarised in the following algorithm.

Algorithm 16 (step-size adaptation).

```
1. Initialise the trajectory $\mathbf{x}_{1:T}[0]$, the initial step sizes $\delta_t[0]$ (for $t \in [T]$), the initial learning rate $\rho[0] = \frac{1}{2}$.
2. Initialise the history of accepted time steps $A \coloneqq (A_{w,t}) \in \{0,1\}^{W \times T}$, with 0 everywhere.
3. For $k = 1, \dots, K$,
    a) sample $\mathbf{x}_{1:T}[k] \sim P(\,\cdot\,|\mathbf{x}_{1:T}[k-1])$, where $P$ denotes the Markov kernel induced by one of the algorithms discussed in this work with step sizes $\delta_{1:T}$ set equal to $\delta_{1:T}[k-1]$,
    b) roll the array $A$ by one: set $A_{2:\min\{W,k\},t} \coloneqq A_{1:\min\{W-1,k-1\},t}$, and $A_{1,t} = \mathbb{I}\{\mathbf{x}_t[k] = \mathbf{x}_t[k-1]\}$, for $t \in [T]$,
    c) compute $\alpha_t = \frac{1}{\min\{W,k\}} \sum_{w=1}^{\min\{W,k\}} A_{w,t}$, for $t \in [T]$,
    d) if $|\alpha_t - \alpha^*| < \sigma$ then keep $\delta_t[k] = \delta_t[k-1]$ unchanged;
       otherwise, set
           $$\delta_t[k] \coloneqq \delta_t[k-1] + \max\{k^\gamma \rho, \rho_{\min}\}(\alpha_t - \alpha)/\alpha^*.$$
```

In our experiments, we took $\sigma = 5\,\%$, $K = 10\,000$, $\delta_t[0] = 10^{-2}$, $W = 100$, $\rho = \frac{1}{2}$, $\rho_{\min} = 10^{-3}$, $\gamma = -\frac{1}{2}$.

G Additional experimental results

In this section, we provide additional simulation results for the multivariate stochastic volatility model experiments from Section 5.

G.1 Calibrated step sizes and acceptance rates

Recall that the step sizes δ_t were calibrated to achieve an acceptance rate of 75 %. Here, the 'acceptance rate' at time t refers to the relative frequency with which the state $\mathbf{x}_t$ is updated. The calibrated step sizes are shown in Figure 6; and the corresponding acceptance rates are shown in Figure 7.

The results are averaged over the four chains and five simulated data sets. We do not report CSMC as it does not require calibration. All methods consistently resulted in acceptance rates close to the target 75 %. Only the twisted Particle-aGRAD algorithm showed more instability as the informativeness of the prior decreased: this is because, contrary to the methods, only a single step-size is used for all time steps, so calibrating for the informativeness of individual observations is not feasible. This seems to hint to the fact that the twisted Particle-aGRAD, *under our proposed calibration,* is less robust than alternatives to heterogeneous levels of informativeness.

G.2 Breakdown of CSMC, aMALA and MALA

In this section, we illustrate the breakdown of CSMC, aMALA and MALA.

Firstly, Figure 8 illustrates that the estimates of the marginal posterior means of $x_{t,15}$ (the 15th component of the state at time t) produced by CSMC, aMALA and MALA differ substan-

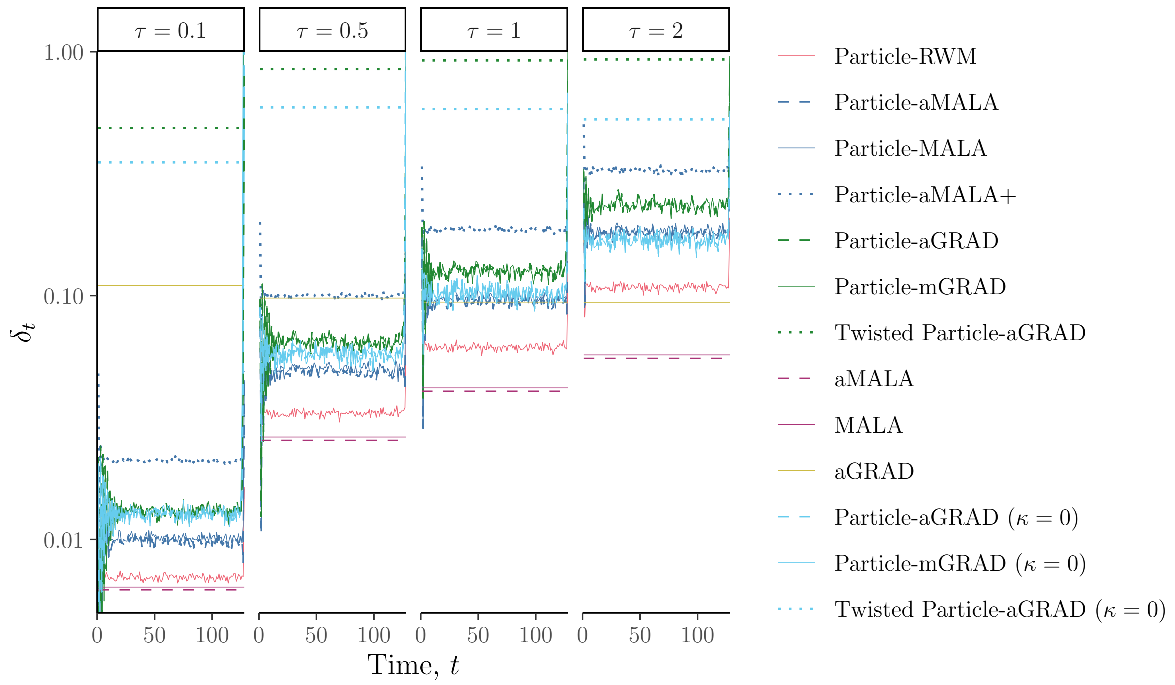


Figure 6: Adaptation of the step-size parameters δ_t, averaged across all four chains and all five simulated data sets (per value of τ) in the multivariate stochastic volatility model.

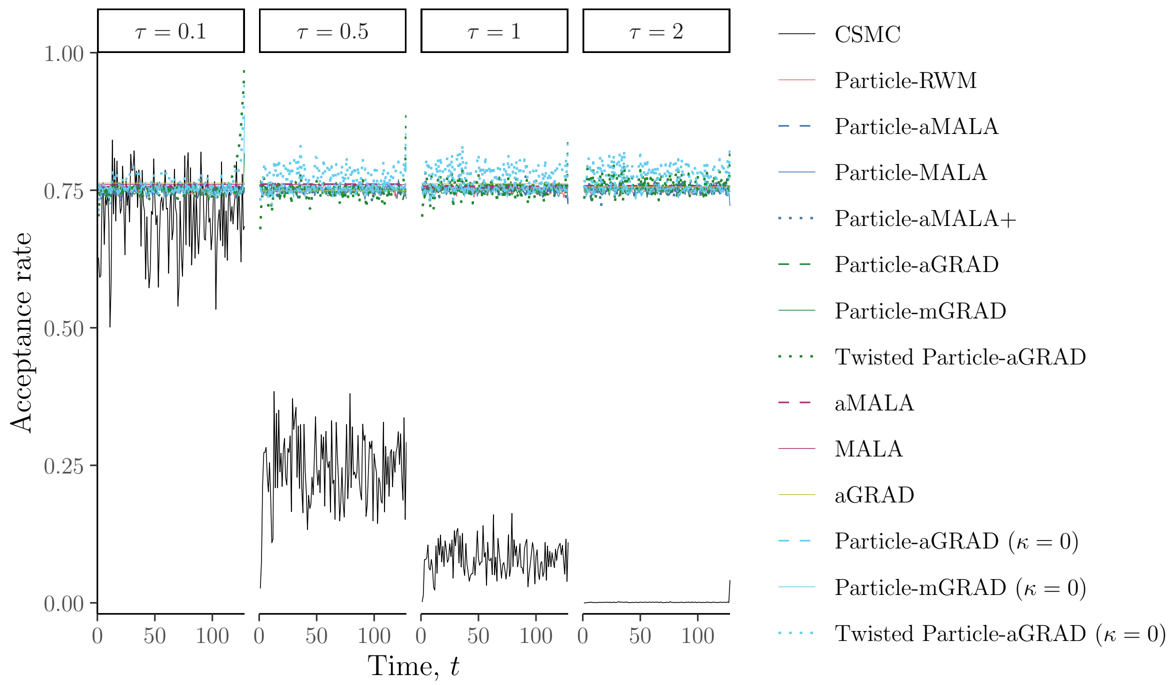


Figure 7: Acceptance rates (i.e., relative frequencies with which states are updated), averaged across all four chains and all five simulated data sets (per value of τ) in the multivariate stochastic volatility model.

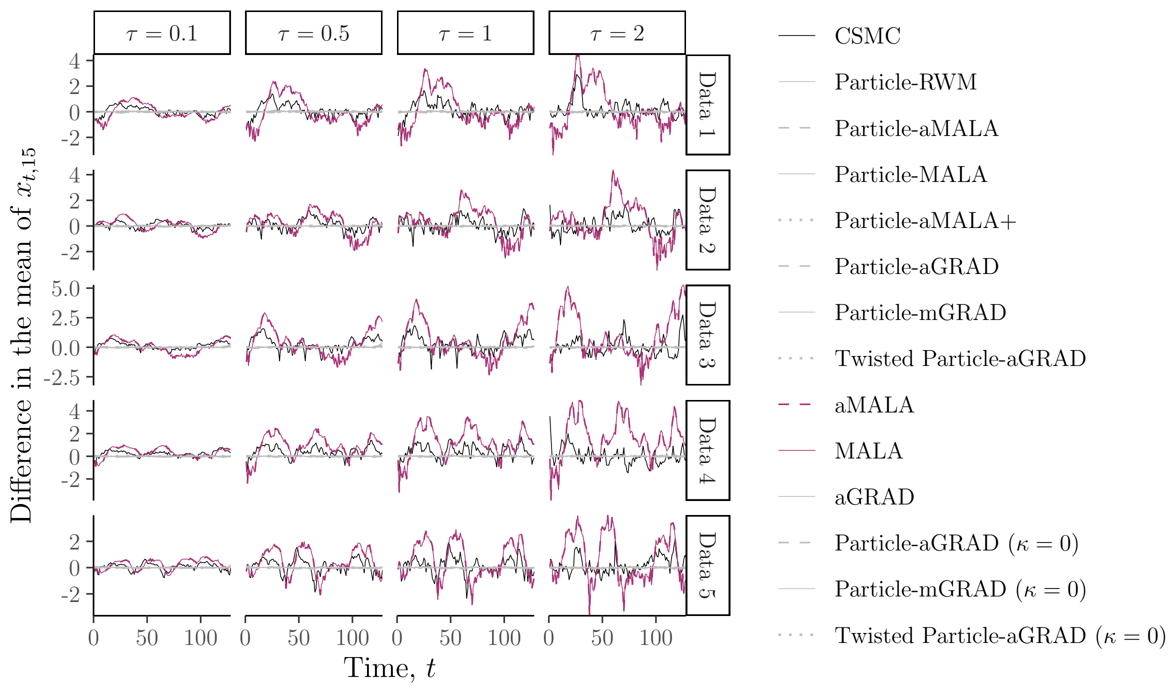


Figure 8: Estimated posterior mean of $x_{t,15}$ minus the estimated posterior mean of $x_{t,15}$ under the aGRAD algorithm, averaged across all four chains for each of the five simulated data sets (per value of τ) in the multivariate stochastic volatility model. The figure shows that the estimated posterior means of CSMC, aMALA and MALA differ substantially from those of all the other algorithms.

tially from those produced by all the other algorithms. We emphasise that the 15th component was arbitrarily chosen as an example and is representative of the other components.

Secondly, Figure 9 illustrates that the energy traces of CSMC, aMALA and MALA differ substantially from those of all the other algorithms. Here, the energy is defined as $\log \pi_T(\mathbf{x}_{1:T}[i])$, where $\mathbf{x}_{1:T}[i]$ is the sample from the ith iteration after burn-in. Such energy traces serve as a visual illustration of both stationarity and mixing speed: if the energy trace of a sampler differs too much from the others, or is not consistent across the independent Markov chains we used, the sampler is unlikely to perform correctly.

G.3 Effective sample sizes

In this section, in Figures 10–12 report the minimum, median and maximum ESS and ESS per second (averaged across all four chains and all five simulated data sets) individually for each time step $t = 1, \dots, T$.

G.4 Autocorrelation

Figure 13 shows the autocorrelation (corrected using Vehtari et al., 2021) of the energy from Figure 9. This serves as a visual confirmation of the statistical performance of the different algorithms considered under several prior dispersion regimes: as expected, the twisted Particle-aGRAD dominates all other alternatives, while Particle-aMALA+ dominates other alternatives, including aGRAD as soon as the prior variance is large enough, followed by Particle-aGRAD/Particle-aGRAD, and then by Particle-aMALA/Particle-MALA, with Particle-RWM being the least efficient.

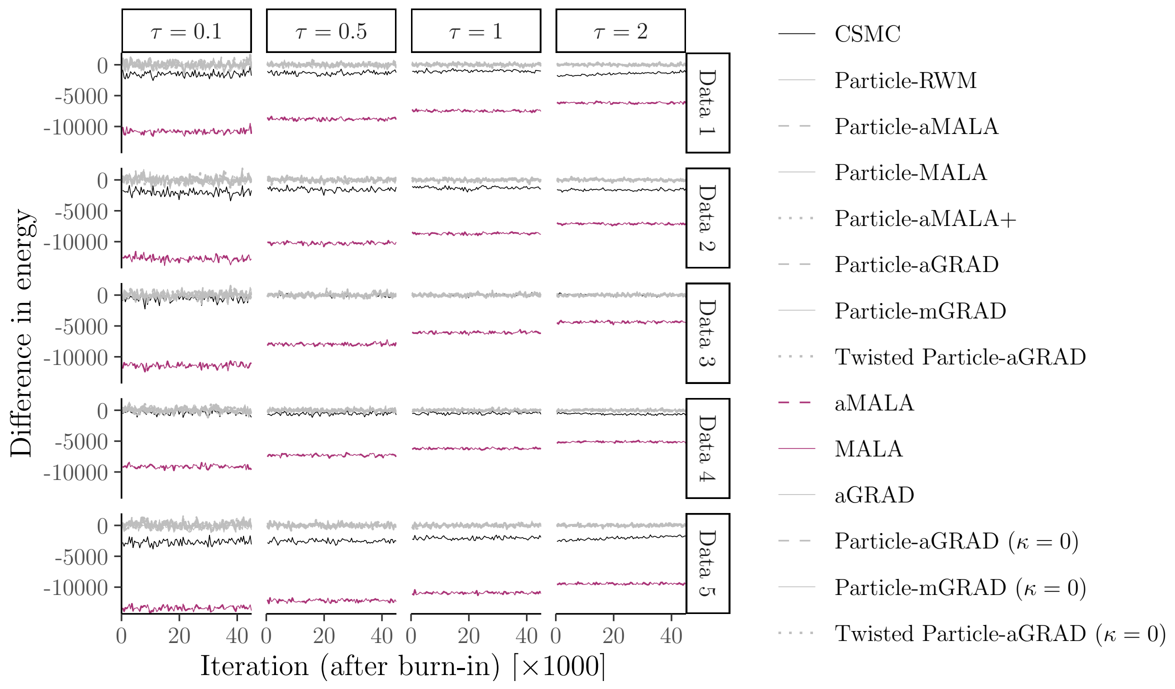


Figure 9: Energy (i.e., $\log \pi_T(\mathbf{x}_{1:T}[i]) + \text{const}$, where $\mathbf{x}_{1:T}[i]$ is the sample from the ith iteration after burn-in) minus the energy under the aGRAD algorithm, averaged across all four chains for each of the five simulated data sets (per value of τ) in the multivariate stochastic volatility model. The figure shows that the energy traces of CSMC, aMALA and MALA differ substantially from those of all the other algorithms.

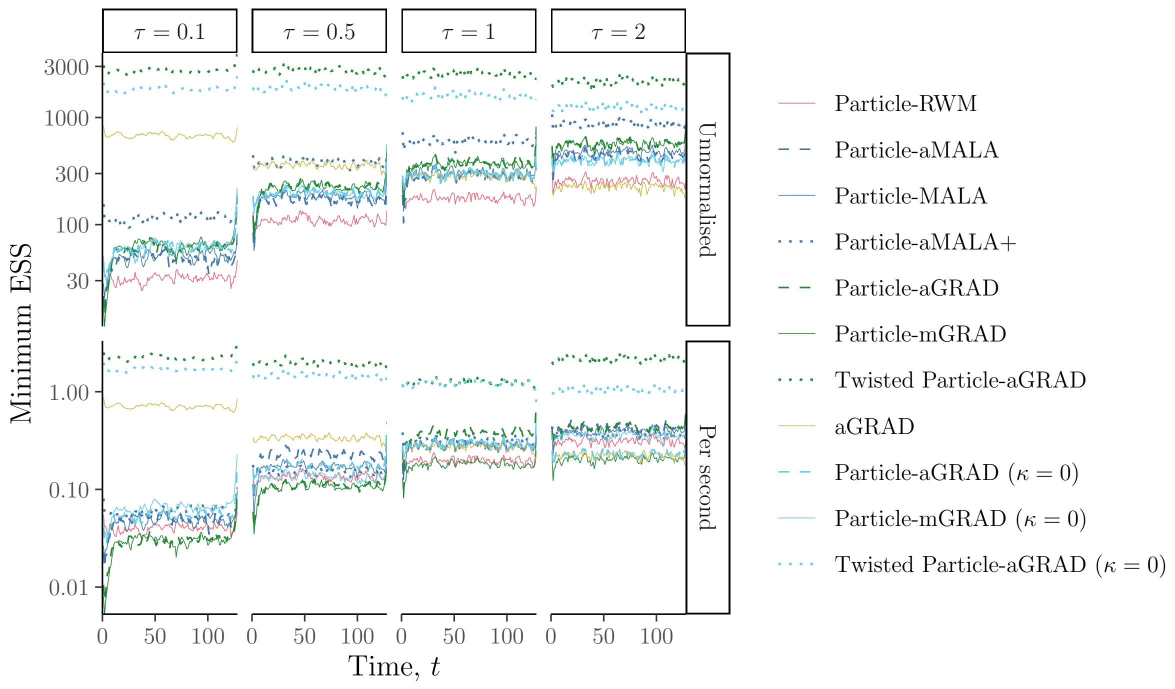


Figure 10: Minimum ESS and ESS per second averaged across all four chains and all five simulated data sets (per value of τ) in the multivariate stochastic volatility model.

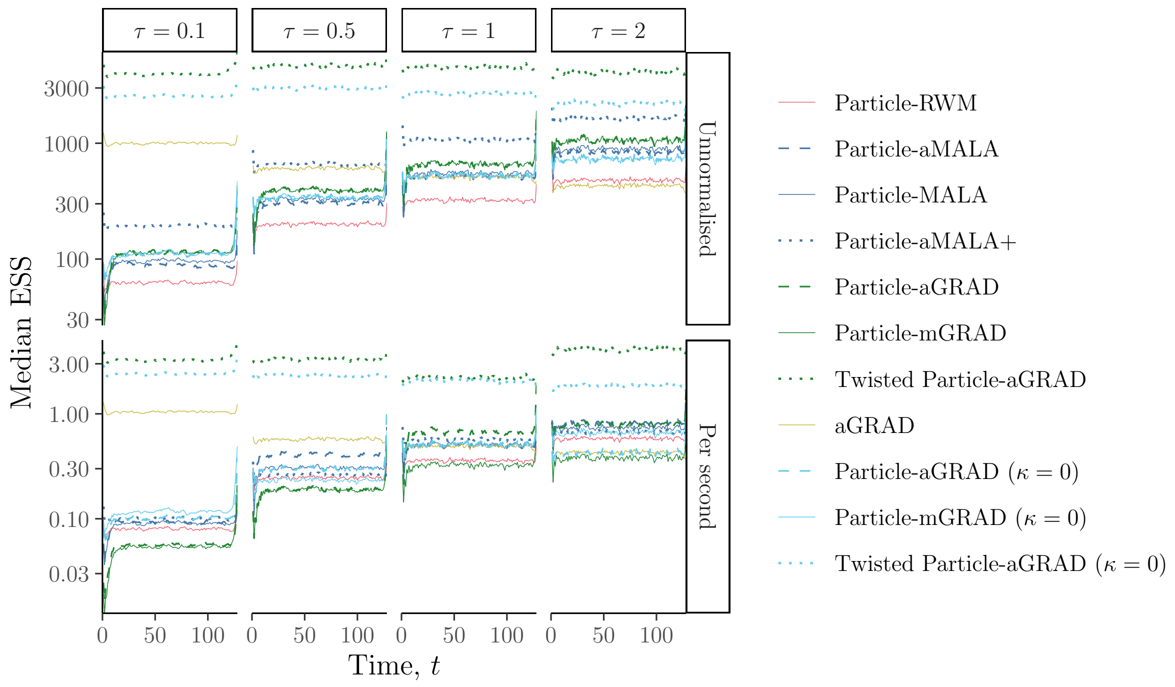


Figure 11: Medium ESS and ESS per second averaged across all four chains and all five simulated data sets (per value of τ) in the multivariate stochastic volatility model.

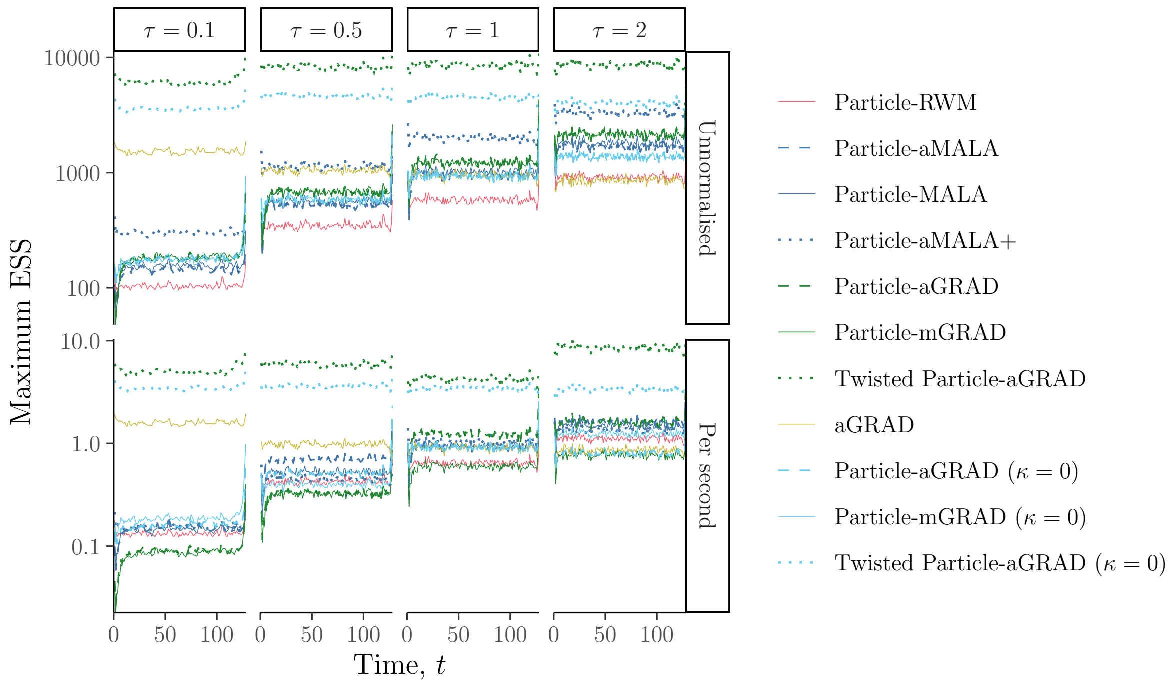


Figure 12: Maximum ESS and ESS per second averaged across all four chains and all five simulated data sets (per value of τ) in the multivariate stochastic volatility model.

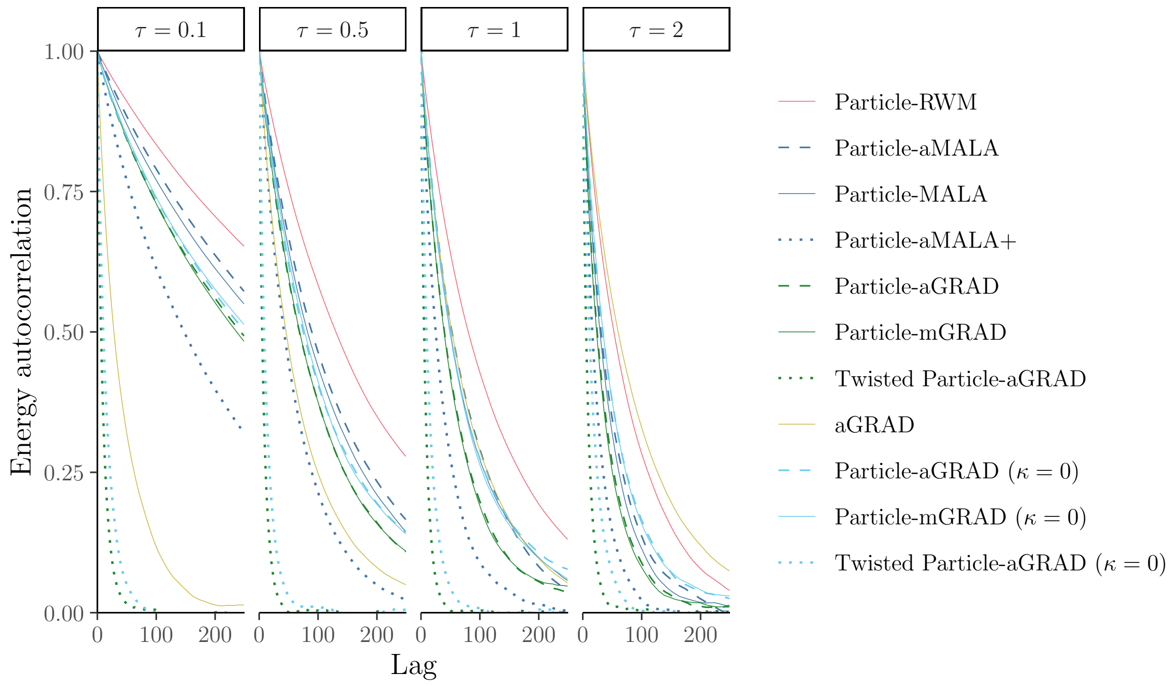


Figure 13: Autocorrelation of the energy from Figure 9 in the multivariate stochastic volatility model.