Tensor approximation of the self-diffusion matrix of tagged particle processes

Jad Dabaghi, Virginie Ehrlacher, Christoph Strössner

Tensor approximation of the self-diffusion matrix of tagged particle processes

Jad Dabaghi[1], Virginie Ehrlacher[2,3], and Christoph Strössner[4]

[1]Léonard de Vinci Pôle Universitaire, Research Center, 92916 Paris La Défense, France

[2]Ecole des Ponts ParisTech, Marne-la-Vallée, France

[3]INRIA Paris, France, MATHERIALS team-project

[4]École Polytechnique Fédérale de Lausanne (EPFL), Institute of Mathematics, CH-1015 Lausanne, Switzerland

jad.dabaghi@devinci.fr, virginie.ehrlacher@enpc.fr, christoph.stroessner@epfl.ch

October 26, 2022

Abstract

The objective of this paper is to investigate a new numerical method for the approximation of the self-diffusion matrix of a tagged particle process defined on a grid. While standard numerical methods make use of long-time averages of empirical means of deviations of some stochastic processes, and are thus subject to statistical noise, we propose here a tensor method in order to compute an approximation of the solution of a high-dimensional quadratic optimization problem, which enables to obtain a numerical approximation of the self-diffusion matrix. The tensor method we use here relies on an iterative scheme which builds low-rank approximations of the quantity of interest and on a carefully tuned variance reduction method so as to evaluate the various terms arising in the functional to minimize. In particular, we numerically observe here that it is much less subject to statistical noise than classical approaches.

1 Introduction

The aim of this paper is to propose a new numerical method for the approximation of the self-diffusion matrix of a tagged particle process on a grid [20] which is less subject to statistical noise than standard Monte-Carlo strategies.

The coefficients of the self-diffusion matrix of this process are defined as long-time averages of the mean-square displacement of the tagged particle [3, 7, 17, 29]. The knowledge of the latter enables to identify the hydrodynamic limits of multi-species symmetric exclusion processes [29] and is thus of tremendous importance for applications. However, computing a numerical approximation of these coefficients is not an easy. Indeed, classical approaches consist in truncating the computational domain to a finite-size supercell with periodic boundary conditions and sampling a large number of realisations of the tagged particle trajectories in order to approximate the mean-square displacement by an empirical average computed with a standard Monte-Carlo approach. Moreover, the value of a finite final time has to be chosen beforehand so as to approximate the long-time limit. In [21], it has been proved that the error linked to the truncation of the computational domain decays exponentially with the size of the supercell. In contrast, the statistical error of the approximation linked to the use of a finite number of random samples of the trajectories of the tagged particle decays as the inverse of the square root of the number of samples. As a consequence, the

main source of error in the practical computation of approximations of the self-diffusion matrix is due to statistical noise [8, 9, 13, 32, 34, 35].

In this paper, we propose an alternative numerical method based on the fact that the self-diffusion coefficients can be equivalently reformulated using the solution of a deterministic high-dimensional optimization problem [3, 21, 29]. The main mathematical ingredient of the numerical method we investigate here is the use of low-rank tensor decompositions to obtain a numerical approximation of this solution, which enables to bypass the curse of dimensionality. This approach leads to an iterative scheme, each elementary step of which boils down to a minimization problem over the set of low-rank tensors, which is solved using a classical alternating linear scheme [2, 11, 30]. Our numerical experiments demonstrate that this low-rank approach leads to very accurate approximations of the self-diffusion coefficients on finite-size grids.

As an illustration of the method, we use the obtained numerical approximations in order to compute the self-diffusion matrix of a two-dimensional tagged particle process defined on a Cartesian grid. We see this work as a preliminary step to computing diffusion coefficients out of Kinetic Monte Carlo simulations for practical applications.

This article is organized as follows. In Section 2, we introduce the lattice-based stochastic hopping model, its hydrodynamic limit and the two equivalent definitions of the self-diffusion matrix of the tagged particle process our work is based upon. The numerical approach we propose here in order to compute a numerical approximation is presented in Section 3. Finally, the efficiency of our approach is illustrated through several numerical experiments presented in Section 4.

2 Self-diffusion matrix

2.1 Infinite-dimensional definition

Let $d = 1, 2, 3$ denote the dimension of the problem. We consider a symmetric tagged particle process defined on the infinite grid $\mathbb{Z}^d$. Let $K \in \mathbb{N}^*$ denote the number of possible jump directions for the particles and $(\mathbf{v}_k)_{1\leq k\leq K} \subset \mathbb{Z}^d \setminus \{0\}$ the set of possible jump directions. For all $1 \leq k \leq K$, the probability of jumping in the direction $\mathbf{v}_k$ is denoted by $p_k \in]0, 1]$. This jumping scheme is assumed to be symmetric in the sense that if the jump in the direction $\mathbf{v}_k$ occurs with probability p_k, then the jump in the direction $-\mathbf{v}_k$ occurs with the same probability.

The self-diffusion matrix application

$$\mathbb{D}_s : \left\{ \begin{array}{ccc} [0,1] & \to & \mathbb{R}^{d\times d} \\ \rho & \mapsto & \mathbb{D}_s(\rho) := (\mathbb{D}_{s,ij}(\rho))_{1\leq i,j\leq d} \end{array} \right.$$

can be defined in the following two ways.

Definition as optimization problem. Let us first introduce some notation. Let $S := \mathbb{Z}^d \setminus \{\mathbf{0}\}$. For all $\boldsymbol{\eta} := (\eta_\mathbf{s})_{\mathbf{s}\in S} \in \{0, 1\}^S$ and all $\mathbf{y} \neq \mathbf{z} \in S$, we define by $\boldsymbol{\eta}^{\mathbf{y},\mathbf{z}} := (\eta^{\mathbf{y},\mathbf{z}}_\mathbf{s})_{\mathbf{s}\in S}$ the element of $\{0, 1\}^S$ such that

$$\eta^{\mathbf{y},\mathbf{z}}_\mathbf{s} := \left\{ \begin{array}{ll} \eta_\mathbf{s} & \text{if } \mathbf{s} \neq \mathbf{y}, \mathbf{z}, \\ \eta_\mathbf{y} & \text{if } \mathbf{s} = \mathbf{z}, \\ \eta_\mathbf{z} & \text{if } \mathbf{s} = \mathbf{y}. \end{array} \right.$$

Furthermore, for all $\mathbf{w} \in S$, we define by $\boldsymbol{\eta}^{\mathbf{0},\mathbf{w}} := (\eta^{\mathbf{0},\mathbf{w}}_\mathbf{s})_{\mathbf{s}\in S}$ the element of $\{0, 1\}^S$ such that

$$\eta^{\mathbf{0},\mathbf{w}}_\mathbf{s} := \left\{ \begin{array}{ll} \eta_{\mathbf{s}+\mathbf{w}} & \text{if } \mathbf{s} \neq -\mathbf{w}, \\ 0 & \text{if } \mathbf{s} = -\mathbf{w}. \end{array} \right.$$

For a mean particle density $\rho \in [0, 1]$, we denote by $\rho^\otimes$ the Bernoulli product measure on $\{0, 1\}^S$ with marginals given by

$$\rho^\otimes \left(\{\boldsymbol{\eta} := (\eta_\mathbf{s})_{\mathbf{s}\in S} : \eta_{\hat{\mathbf{s}}} = 1\} \right) = \rho,$$

for all $\hat{\mathbf{s}} \in S$. Let us also define the set of functions $H_\rho := L^2_{\rho^\otimes}\left(\{0,1\}^S\right)$, where $L^2_{\rho^\otimes}$ denotes the L^2 Lebesgue space with measure $\rho^\otimes$.

For a drift direction $\mathbf{u} \in \mathbb{R}^d$ and mean particle density $\rho \in [0,1]$, the self-diffusion coefficient $\mathbf{u}^T \mathbb{D}_s(\rho)\,\mathbf{u}$ is given by:

$$\mathbf{u}^T \mathbb{D}_s(\rho)\mathbf{u} := 2 \inf_{\Psi \in H_\rho} \mathbb{E}_{\rho^\otimes}\left[\sum_{k=1}^{K} p_k \left((1-\eta_{\mathbf{v}_k})\left(\mathbf{u}\cdot\mathbf{v}_k + \Psi(\boldsymbol{\eta}^{\mathbf{0},\mathbf{v}_k}) - \Psi(\boldsymbol{\eta})\right)^2 + \frac{1}{2}\sum_{\substack{\mathbf{y}\in S \\ \mathbf{y}+\mathbf{v}_k \neq \mathbf{0}}} \left(\Psi(\boldsymbol{\eta}^{\mathbf{y}+\mathbf{v}_k,\mathbf{y}}) - \Psi(\boldsymbol{\eta})\right)^2 \right)\right], \tag{1}$$

where the notation $\mathbb{E}_{\rho^\otimes}$ refers to the fact that the expectation is taken over the product measure $\rho^\otimes$ [20, 29]. Problem (1) thus reads as an infinite-dimensional optimization problem over the set H_ρ.

Remark 2.1. Naturally, for all $\rho \in [0,1]$, since $\mathbb{D}_s(\rho)$ is a symmetric matrix, one can easily deduce the full matrix $\mathbb{D}_s(\rho)$ from the knowledge of $\mathbf{u}^T\mathbb{D}_s(\rho)\mathbf{u}$ for a few vectors $\mathbf{u} \in \mathbb{R}^d$.

Definition as long time mean square deviation. The quantity $\mathbf{u}^T\mathbb{D}_s(\rho)\mathbf{u}$ can be equivalently expressed as the long time limit of the following expectation [29, Theorem 2.3]. Let us assume that, at time $t = 0$, the tagged particle is located at position $\mathbf{0}$ and all other sites are occupied with probability ρ following a Bernoulli distribution. The duration between two consecutive jumping events follow an exponential law with parameter 1. The jumping directions (respectively rates) are given by $\{\mathbf{v}_1, \ldots, \mathbf{v}_K\}$ (respectively $p_1, \ldots, p_K$). Jumps are not allowed if the final site of the jumping particle is already occupied. Let $\mathbf{w}(t)$ denote the position of the tagged particle at time $t \geq 0$. It then holds that, for a drift direction $\mathbf{u} \in \mathbb{R}^d$ and mean particle density $\rho \in [0,1]$, the quantity $\mathbf{u}^T\mathbb{D}_s(\rho)\mathbf{u}$ can be equivalently formulated as the long-time limit mean square deviation of the tagged particle in the direction $\mathbf{u} \in \mathbb{R}^d$, i.e.

$$\mathbf{u}^T \mathbb{D}_s(\rho)\mathbf{u} = 2 \lim_{t\to\infty} \frac{\mathbb{E}_{\rho^\otimes}[\langle \mathbf{u}, \mathbf{w}(t)\rangle^2]}{t}, \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the euclidean scalar product of $\mathbb{R}^d$.

Remark 2.2. Starting from a Bernoulli-product measure and symmetric transition rates, it is a classical problem in probability theory to study the motion of a tagged particle on $\mathbb{Z}^d$ [18, 22, 31], see also [5, 10] for recent work on tagged particles on trees. For $d \geq 2$ and symmetric nearest neighbor transition rates, the tagged particle is known to satisfy a central limit theorem with non-degenerate limiting variance. However, to our best knowledge, a general closed formula for the limiting variance is not available.

Notice that when starting from a Bernoulli-product measure with a tagged particle in the origin, the resulting environment process is stationary with respect to the Bernoulli-product measure conditioned to contain a particle in the origin, usually called the Palm measure. This suggests that the limiting variance for the tagged particle can be described in terms of the sum of separable functions, i.e. in terms of a low-rank function. In turn, this indicates that the equivalent characterization of the limiting variance as in Equation (1) is also related to low-rank functions.

Let us point out that the quantity $\mathbf{u}^T\mathbb{D}_s(\rho)\mathbf{u}$ cannot be computed exactly in practice, neither using expression (1) nor expression (2). On the one hand, (1) reads as an infinite-dimensional optimization problem and has to be approximated by a finite-dimensional optimization problem in practice [15, 21]. On the other hand, (2) requires the computation of the long-time average of the stochastic process defined on the infinite lattice grid $\mathbb{Z}^d$. Equivalently, it has to be approximated in a finite-dimensional setting, as long-time average of the mean-square deviation of the tagged particle associated to a stochastic process defined on a finite-size grid with periodic boundary conditions. Both finite-dimensional approximations are presented in detail in the next section.

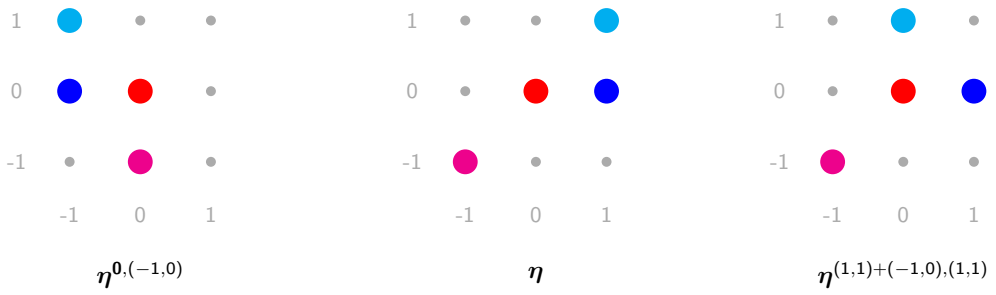

Figure 1: Middle: Visualization of one particular $\boldsymbol{\eta} \in S_1$ for $d = 2$ with three occupied sites marked in blue, cyan and magenta. Additionally, we mark the tagged particle $\mathbf{0}$ in red. Left: Visualization of the occupied sites of $\boldsymbol{\eta}^{\mathbf{0},(-1,0)}$. This can be seen as a jump of the imaginary red particle one step to the left, followed by immediately relabeling of the sites such that red particle remains at $\mathbf{0}$. By exploiting the periodicity, we obtain $\boldsymbol{\eta}^{\mathbf{0},(-1,0)} \in S_1$. Right: Visualization of the occupied sites of $\boldsymbol{\eta}^{(1,1)+(-1,0),(1,1)}$. This can be seen as jump of the cyan particle one step to the left.

2.2 Finite-dimensional approximation

Discretized minimization problems. Let $M \in \mathbb{N}^*$ denote a discretization parameter and introduce the finite grid $S_M := \{-M, \cdots, M\}^d \setminus \{\mathbf{0}\}$. For the sake of simplicity, we assume that $M \geq \max_{k\in\{1,\dots,K\}} ||v_k||_1$, where $||\cdot||_1$ denotes the taxicab norm. For any $\boldsymbol{\eta} \in \{0,1\}^{S_M}$, we can construct by periodicity an extension $\widetilde{\boldsymbol{\eta}} := (\widetilde{\eta}_{\mathbf{s}})_{\mathbf{s}\in S} \in \{0,1\}^S$ by assuming with a slight abuse of notation that the site $\mathbf{0}$ is occupied, i.e. $\widetilde{\eta}_{\mathbf{s}} = 1$ for $\mathbf{s} \in (2M+1)\cdot\mathbb{Z}^d \setminus \{\mathbf{0}\}$. Using this notation, for all $\boldsymbol{\eta} \in \{0,1\}^{S_M}$ and all $\mathbf{y}, \mathbf{z}, \mathbf{w} \in S$, we define $\boldsymbol{\eta}^{\mathbf{y},\mathbf{z}} \in \{0,1\}^{S_M}$ and $\boldsymbol{\eta}^{\mathbf{0},\mathbf{w}} \in \{0,1\}^{S_M}$ as

$$\boldsymbol{\eta}^{\mathbf{y},\mathbf{z}} := (\widetilde{\eta}^{\mathbf{y},\mathbf{z}}_{\mathbf{s}})_{\mathbf{s}\in S_M} \quad \text{and} \quad \boldsymbol{\eta}^{\mathbf{0},\mathbf{w}} := \left(\widetilde{\eta}^{\mathbf{0},\mathbf{w}}_{\mathbf{s}}\right)_{\mathbf{s}\in S_M}.$$

Figure 1 shows an illustration of $\boldsymbol{\eta}^{\mathbf{y},\mathbf{z}}$ and $\boldsymbol{\eta}^{\mathbf{0},\mathbf{w}}$.

Let $N := (2M+1)^d - 1 = \mathrm{Card}(S_M)$. For all $\ell \in \{0,\dots,N\}$, let $C_{M,\ell} := \{\boldsymbol{\eta} \in \{0,1\}^{S_M} \,|\, \sum_{\mathbf{s}\in S_M} \eta_{\mathbf{s}} = \ell\}$ denote the set of all possible configurations of the particles on S_M so that the total number of occupied sites is equal to ℓ.

Let us define $H_M := \left\{\Psi : \{0,1\}^{S_M} \to \mathbb{R}\right\}$. For every $\mathbf{u} \in \mathbb{R}^d$ and $l \in \{0,\dots,N\}$, we introduce the quadratic functional $A^{\mathbf{u}}_{M,\ell} : H_M \to \mathbb{R}$ defined by

$$A^{\mathbf{u}}_{M,\ell}(\Psi) := \frac{1}{|C_{M,\ell}|} \sum_{\boldsymbol{\eta}\in C_{M,\ell}} \sum_{k=1}^{K} p_k \left((1-\eta_{\mathbf{v}_k}) \left(\mathbf{u}\cdot\mathbf{v}_k + \Psi(\boldsymbol{\eta}^{\mathbf{0},\mathbf{v}_k}) - \Psi(\boldsymbol{\eta})\right)^2 + \frac{1}{2} \sum_{\substack{\mathbf{y}\in S_M \\ \mathbf{y}+\mathbf{v}_k\neq\mathbf{0}}} \left(\Psi(\boldsymbol{\eta}^{\mathbf{y}+\mathbf{v}_k,\mathbf{y}}) - \Psi(\boldsymbol{\eta})\right)^2 \right).$$

Then, assuming that $\overline{\rho} = \frac{\ell}{N}$ for some $0 \leq \ell \leq N$, one can define [3] for all $\mathbf{u} \in \mathbb{R}^d$,

$$\mathbf{u}^T \mathbb{D}^M_s\left(\frac{\ell}{N}\right)\mathbf{u} := 2 \min_{\Psi\in H_M} A^{\mathbf{u}}_{M,\ell}(\Psi). \tag{3}$$

It is proved in [21] that $\lim_{\substack{M\to+\infty \\ \frac{\ell}{N}\to\overline{\rho}}} \mathbf{u}^T \mathbb{D}^M_s\left(\frac{\ell}{N}\right)\mathbf{u} = \mathbf{u}^T \mathbb{D}_s(\overline{\rho})\mathbf{u}$.

Combined minimization problem. The collection of sets $C_{M,0},\dots,C_{M,N}$ form a partition of the set $\{0,1\}^{S_M}$. Observe for a given $\Psi \in H_M$, $A^{\mathbf{u}}_{M,\ell}(\Psi)$ only depends on the values of $\Psi(\boldsymbol{\eta})$ for $\boldsymbol{\eta} \in C_{M,\ell}$, since

$\boldsymbol{\eta} \in C_{M,\ell}$ implies that also $\boldsymbol{\eta}^{\mathbf{0},\mathbf{v}_k} \in C_{M,\ell}$ and $\boldsymbol{\eta}^{\mathbf{y}+\mathbf{v}_k,\mathbf{y}} \in C_{M,\ell}$ for all $1 \leq k \leq K$. As a consequence, if $\Psi_{\mathsf{opt}}^{M,\mathbf{u}} \in H_M$ is a minimizer of

$$\min_{\Psi \in H_M} A_M^{\mathbf{u}}(\Psi), \tag{4}$$

where

$$A_M^{\mathbf{u}}(\Psi) := \sum_{\boldsymbol{\eta} \in \{0,1\}^{S_M}} \sum_{k=1}^{K} p_k \left((1-\eta_{\mathbf{v}_k}) \left(\mathbf{u} \cdot \mathbf{v}_k + \Psi(\boldsymbol{\eta}^{\mathbf{0},\mathbf{v}_k}) - \Psi(\boldsymbol{\eta}) \right)^2 + \frac{1}{2} \sum_{\substack{\mathbf{y} \in S_M \\ \mathbf{y}+\mathbf{v}_k \neq \mathbf{0}}} \left(\Psi(\boldsymbol{\eta}^{\mathbf{y}+\mathbf{v}_k,\mathbf{y}}) - \Psi(\boldsymbol{\eta}) \right)^2 \right), \tag{5}$$

then it holds that $A_{M,\ell}^{\mathbf{u}}(\Psi_{\mathsf{opt}}^{M,\mathbf{u}}) = \min_{\Psi \in H_M} A_{M,\ell}^{\mathbf{u}}(\Psi)$ for all $\ell \in \{0, \dots, N\}$. The knowledge of $\Psi_{\mathsf{opt}}^{M,\mathbf{u}}$ then allows us to compute $\mathbf{u}^T \mathbb{D}_s^M \left(\frac{\ell}{N}\right) \mathbf{u}$ for all $0 \leq \ell \leq N$ as $2A_{M,\ell}^{\mathbf{u}}(\Psi_{\mathsf{opt}}^{M,\mathbf{u}})$. Note that the minimization problem (4) is then independent of ℓ, in contrast to (3). However, the minimization problem (4) has 2^N degrees of freedom and is thus intractable for large values of N.

The main contribution of our paper is to propose a low-rank tensor approximation algorithm to compute an approximation of $\Psi_{\mathrm{opt}}^{M,u}$ in order to mitigate this issue. Our proposed method is presented in Section 3.

Estimation of long-time mean square deviation. The quantity $\mathbf{u}^T \mathbb{D}_s^M \left(\frac{\ell}{N}\right) \mathbf{u}$ can be equivalently expressed as the long-time mean square deviation of a tagged particle evolving in a periodic environment [21]. The Monte-Carlo algorithm is the standard method of choice in practice to compute an approximation of $\mathbf{u}^T \mathbb{D}_s^M \left(\frac{\ell}{N}\right) \mathbf{u}$ by means of empirical averages. The quality of the obtained approximations then depends on the choice of two numerical parameters, namely the number of Monte-Carlo samples (i.e. stochastic realizations of the periodized tagged particle process) and the value of the chosen final time.

We aim to compare the low-rank tensor approximation algorithm we propose in this work with such Monte-Carlo methods. To this aim, we detail below the algorithm used to compute such empirical averages, and which we will use as a comparison with the method we develop here and which is presented in Section 3.

Let $1 \leq \ell \in \leq N$. An initial environment $\boldsymbol{\eta}$ is obtained by sampling uniformly from $C_{M,\ell}$. We sample from exponential distributions to determine the next time a particle in the environment or the tagged particle performs a jump. Whenever a jump is performed the environment $\boldsymbol{\eta}$ is updated accordingly. Let $\mathbf{w} = \mathbf{0} \in \mathbb{Z}^d$ denote the initial position of the tagged particle. Throughout the simulation, we track the position of the tagged particle in $\mathbb{Z}^d$, i.e. whenever the tagged particle successfully jumps in direction $\mathbf{v}_k$, we set $\mathbf{w} = \mathbf{w} + \mathbf{v}_k$. To approximate the expectation of $\mathbf{w}$, we repeat this simulation N_s times with different samples. Each of these simulations is stopped at the same stopping time T at which we compute the approximation of $\mathbf{u}^T \mathbb{D}_s(\rho)\mathbf{u}$. This approach is formalized in Algorithm 1.

Remark 2.3. In Algorithm 1, we set $N_s = \lceil \hat{N}_s/(\ell+1) \rceil$, where $\hat{N}_s$ is a given input parameter. This ensures that the expected runtime is approximately equal for all choices of ℓ. Additionally Figure 5 shows that the variance of the output is comparable independently of ℓ. If we were to use the same N_s for all ℓ, we would observe a much larger variance for smaller ℓ and the runtime would increase for larger ℓ.

3 Low-rank solutions for the optimization problem

The aim of this section is to present the numerical method based on low-rank tensor approximations we propose for the resolution of the high-dimensional optimization problem (4). We first define low-rank functions in H_M and introduce a fast and stable algorithm for the evaluation of $A_M^{\mathbf{u}}$. We then develop a successive minimization scheme to compute low-rank solutions of (4). Each minimization step is performed using an alternating linear scheme, which relies on the fast and stable evaluations of $A_M^{\mathbf{u}}$. Lastly, we discuss the evaluation of $A_{M,\ell}^{\mathbf{u}}$ for the computation of the self-diffusion coefficient (3).

Algorithm 1 Long-term Monte Carlo estimation

Input: $M \in \mathbb{N}^*$, $\mathbf{u} \in \mathbb{R}^d$, number of occupied sites $1 \leq \ell \leq N$ with $N = (2M+1)^d - 1$, final time $T > 0$, $\hat{N}_s \in \mathbb{N}^*$
Output: approximation $\alpha \approx \frac{1}{2}\mathbf{u}^T \mathbb{D}_s^M \left(\frac{\ell}{N}\right) \mathbf{u} = A_{M,\ell}^{\mathbf{u}}(\Psi_{\mathsf{opt}}^{M,\mathbf{u}})$
$\mathbf{w} = \mathbf{0}$, $\alpha = 0$, $N_s = \lceil \hat{N}_s/(\ell+1) \rceil$
for $i = 1, 2, 3, \ldots, N_s$
    randomly initialize $\boldsymbol{\eta} \in C_{M,\ell}$ and set $t_{\text{TOTAL}} = 0$,
    while true
        sample t_{NEW} from an exponential distribution with mean $\frac{1}{\ell+1}$ and set $t_{\text{TOTAL}} = t_{\text{TOTAL}} + t_{\text{NEW}}$
        if $t_{\text{TOTAL}} > T$
            break
        uniformly select either one occupied site $\mathbf{y} \in S_M$ with $\boldsymbol{\eta}_{\mathbf{y}} = 1$ or the tagged particle $\mathbf{y} = \mathbf{0}$
        select a jump direction $\mathbf{v}_k$ with probability p_k
        if $\tilde{\boldsymbol{\eta}}_{\mathbf{y}+\mathbf{v}_k} = \mathbf{0}$, i.e. the target location is not occupied
            when $\mathbf{y} \neq \mathbf{0} \Rightarrow$ set $\boldsymbol{\eta} = \boldsymbol{\eta}^{\mathbf{y}+\mathbf{v}_k,\mathbf{y}}$
            when $\mathbf{y} = \mathbf{0} \Rightarrow$ set $\boldsymbol{\eta} = \boldsymbol{\eta}^{\mathbf{0},\mathbf{v}_k}$ and update $\mathbf{w} = \mathbf{w} + \mathbf{v}_k$
    update $\alpha = \alpha + \langle \mathbf{u}, \mathbf{w} \rangle^2$
$\alpha = \frac{\alpha}{TN_s}$

3.1 Low-rank functions

A function $R \in H_M$ is called a rank-1 or pure tensor product function when it can be written as

$$R(\boldsymbol{\eta}) = \Pi_{\mathbf{s} \in S_M} R_{\mathbf{s}}(\eta_{\mathbf{s}}), \quad \forall \boldsymbol{\eta} = (\eta_{\mathbf{s}})_{\mathbf{s} \in S_M} \in \{0,1\}^{S_M},$$

for some $R_{\mathbf{s}} : \{0,1\} \to \mathbb{R}$ for $\mathbf{s} \in S_M$. Let $H_M^1 \subset H_M$ denote the set of pure tensor product functions of H_M. Let $r \in \mathbb{N}$. A function $\Phi \in H_M$ is called a rank-r function when it can be written as $\Phi = R^{(1)} + \cdots + R^{(r)}$, where $R^{(k)} \in H_M^1$ for $1 \leq k \leq r$. We denote by $H_M^r \subset H_M$ the set of all rank-r functions. For all $r \in \mathbb{N}^*$, it holds

$$\min_{\Psi \in H_M} A_M^{\mathbf{u}}(\Psi) = \min_{\Phi \in H_M^{2^N}} A_M^{\mathbf{u}}(\Phi) \leq \min_{\Phi \in H_M^{r+1}} A_M^{\mathbf{u}}(\Phi) \leq \min_{\Phi \in H_M^r} A_M^{\mathbf{u}}(\Phi) \leq \min_{R \in H_M^1} A_M^{\mathbf{u}}(R).$$

In the next sections, we derive an algorithm to compute an approximation of $\min_{\Phi \in H_M^r} A_M^{\mathbf{u}}(\Phi)$.

3.2 Fast and stable evaluation for low-rank functions

In this section, we introduce a fast and stable method to evaluate $A_M^{\mathbf{u}}(\Psi)$ for $\Psi \in H_M^r$. A naive evaluation would require to sum over 2^N terms, which is intractable for large values of N. In principle, the order of summation can be exchanged leading to terms of the form $\sum_{\boldsymbol{\eta} \in \{0,1\}^{S_M}} \Psi(\boldsymbol{\eta})^2$, which can be evaluated individually. However, subtracting such terms leads to a lot of numerical cancellation for $M > 1$. We circumvent these issues by treating the evaluation of $A_M^{\mathbf{u}}(\Psi)$ as the computation of the Frobenius norms of certain tensors. These Frobenius norms can then be evaluated in a fast and stable way.

3.2.1 Reformulation as tensor norm

For $\Psi \in H$, we consider the tensor $\mathcal{T}^{(\Psi)} \in \mathbb{R}^{2^N}$ with entries $\mathcal{T}^{(\Psi)}_{i_1,\ldots,i_N} = \Psi(\boldsymbol{\eta})$, where $\boldsymbol{\eta} = (\eta_{\mathbf{s}_1}, \ldots, \eta_{\mathbf{s}_N})$ for some enumeration $\mathbf{s}_1, \ldots, \mathbf{s}_N$ of the sites in S_M and $\eta_{\mathbf{s}_j} = i_j - 1$ for $1 \leq j \leq N$. Note that $\mathcal{T}^{(\Psi)}$ is a tensor of at most rank-r when $\Psi \in H_M^r$. In particular, when Ψ is given in term of the functions $R_{\mathbf{s}}^{(k)}$ for $1 \leq j \leq N$, $1 \leq k \leq r$, we can write $\mathcal{T}^{(\Psi)}$ in the so called CP-format [16, 19] as

$$\mathcal{T}^{(\Psi)}_{i_1,\ldots,i_N} = \sum_{k=1}^{r} \prod_{j=1}^{N} \mathbf{a}_{i_j}^{(j,k)}, \tag{6}$$

where $\mathbf{a}^{(j,k)} \in \mathbb{R}^2$ is defined as $\mathbf{a}^{(j,k)} = (R^{(k)}_{\mathbf{s}_j}(0), R^{(k)}_{\mathbf{s}_j}(1))$.

For $\mathbf{v}, \mathbf{y} \in S_m$, we analogously define the tensor $\mathcal{T}^{(\Psi^{(\mathbf{0},\mathbf{v})})} \in \mathbb{R}^{2^N}$ with entries $\mathcal{T}^{(\Psi^{(\mathbf{0},\mathbf{v})})}_{i_1,\dots,i_N} = \Psi(\boldsymbol{\eta}^{\mathbf{0},\mathbf{v}})$. When $\mathbf{y}+\mathbf{v} \neq \mathbf{0}$ we additionally define the tensor $\mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v},\mathbf{y})})} \in \mathbb{R}^{2^N}$ with entries $\mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v},\mathbf{y})})}_{i_1,\dots,i_N} = \Psi(\boldsymbol{\eta}^{\mathbf{y}+\mathbf{v},\mathbf{y}})$. We observe that these are again are rank-r tensors when $\Psi \in H^r_M$. Lastly, we define the rank-1 tensor $\mathcal{T}^{(\mathbf{u}\cdot\mathbf{v})} \in \mathbb{R}^{2^N}$ with entries $\mathcal{T}^{(\mathbf{u}\cdot\mathbf{v})}_{i_1,\dots,i_N} = \mathbf{u}\cdot\mathbf{v}$.

For a tensor $\mathcal{T} \in \mathbb{R}^{2^N}$ and a site $\mathbf{s}$ we define the projection operators $\mathcal{P}_\mathbf{s} : \mathbb{R}^{2^N} \to \mathbb{R}^{2^N}$ as

$$(\mathcal{P}_\mathbf{s}(\mathcal{T}))_{i_1,\dots,i_N} := \begin{cases} \mathcal{T}_{i_1,\dots,i_N} & i_\mathbf{s} = 1 \\ 0 & \text{otherwise} \end{cases}, \quad \text{for } 1 \le i_1,\dots,i_N \le 2,$$

where $i_\mathbf{s}$ denotes to the index assigned to site $\mathbf{s}$. We can now rewrite (5) as

$$A^\mathbf{u}_M(\Psi) = \sum_{k=1}^K p_k \left(\|P_{\mathbf{v}_k}\big(\mathcal{T}^{(\mathbf{u}\cdot\mathbf{v}_k)} + \mathcal{T}^{(\Psi^{(\mathbf{0},\mathbf{v}_k)})} - \mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v}_k,\mathbf{y})})}\big)\|_F^2 + \frac{1}{2} \sum_{\substack{\mathbf{y}\in S_M \\ \mathbf{y}+\mathbf{v}_k \neq \mathbf{0}}} \|\mathcal{T}^{(\Psi^{(\mathbf{0},\mathbf{v}_k)})} - \mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v}_k,\mathbf{y})})}\|_F^2 \right), \quad (7)$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

3.2.2 Efficient evaluation of Frobenius norms for sums of low-rank tensors

Let $k \in \{1,\dots,K\}$ and $\mathbf{y} \in S_M$ such that $\mathbf{y}+\mathbf{v}_k \neq \mathbf{0}$ be fixed. We derive an algorithm inspired by tensor train (TT) orthogonalization [26] to evaluate $\|\mathcal{T}^{(\Psi^{(\mathbf{0},\mathbf{v}_k)})} - \mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v}_k,\mathbf{y})})}\|_F^2$.

We start by rewriting Equation (6) as

$$T^{(\Psi)}_{i_1,\dots,i_N} = \sum_{k_1=1}^r \cdots \sum_{k_{N-1}=1}^r \mathcal{C}^1_{1,i_1,k_1} \mathcal{C}^2_{k_1,i_2,k_2} \mathcal{C}^3_{k_2,i_3,k_3} \cdots \mathcal{C}^N_{k_{N-1},i_N,1}, \quad (8)$$

with tensors $\mathcal{C}^1 \in \mathbb{R}^{1\times 2\times r}$, $\mathcal{C}^N \in \mathbb{R}^{r\times 2\times 1}$ and $\mathcal{C}^j \in \mathbb{R}^{r\times 2\times r}$ for $1 < j < N$, whose entries are given by

$$\begin{aligned}
\mathcal{C}^j_{k_1,i,k_2} &= \begin{cases} \mathbf{a}_i^{j,k_1} & k_1 = k_2 \\ 0 & \text{otherwise} \end{cases}, && \text{for } 1 < j < N,\ 1 \le i \le 2,\ 1 \le k_1, k_2 \le N, \\
\mathcal{C}^1_{1,i,k} &= \mathbf{a}_i^{1,k} && \text{for } 1 \le i \le 2,\ 1 \le k \le N, \\
\mathcal{C}^N_{k,i,1} &= \mathbf{a}_i^{N,k} && \text{for } 1 \le i \le 2,\ 1 \le k \le N.
\end{aligned}$$

The representation format (8) can be generalized to other low-rank tensors. Let $\mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v}_k,\mathbf{y})})}$ be analogously represented in the format (8) by tensors $\mathcal{D}^j$. The tensor $\mathcal{T}^{(\Psi^{(\mathbf{0},\mathbf{v}_k)})} - \mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v}_k,\mathbf{y})})}$ is at most rank-$2r$. It's representation in the form of (8) with tensors $\mathcal{E}^j$ can be constructed from the tensors $\mathcal{C}^j, \mathcal{D}^j$, i.e. the tensors $\mathcal{E}^1 \in \mathbb{R}^{1\times 2\times 2r}$, $\mathcal{E}^N \in \mathbb{R}^{2r\times 2\times 1}$ and $\mathcal{E}^j \in \mathbb{R}^{2r\times 2\times 2r}$ for $1 < j < N$ are defined as $\mathcal{E}^1_{1,:,1:r} = \mathcal{C}^1$, $\mathcal{E}^1_{1,:,r+1:2r} = -\mathcal{D}^1$, $\mathcal{E}^N_{1:r,:,1} = \mathcal{C}^N$, $\mathcal{E}^N_{r+1:2r,:,1} = \mathcal{D}^N$ and $\mathcal{E}^j_{1:r,:,1:r} = \mathcal{C}^j$, $\mathcal{E}^j_{r+1:2r,:,r+1:2r} = \mathcal{D}^j$. Note that for $1 < j < N$ these tensors are sparse by construction.

We can compute the norm $\|\mathcal{T}^{(\Psi^{(\mathbf{0},\mathbf{v}_k)})} - \mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v}_k,\mathbf{y})})}\|_F^2$ directly form the tensors $\mathcal{E}^j$ in a fast and stable way using using TT orthogonalization [26] as defined in Algorithm 2. We want to emphasize that the sparsity structure of $\mathcal{E}^j$ is preserved in Algorithm 2. The evaluation of $\|\mathcal{E}^N\|_F^2$ in line 13 requires summing up $4r$ terms, whereas a naive evaluation of $\|\mathcal{T}^{(\Psi^{(\mathbf{0},\mathbf{v}_k)})} - \mathcal{T}^{(\Psi^{(\mathbf{y}+\mathbf{v}_k,\mathbf{y})})}\|_F^2$ involves 2^N terms.

Remark 3.1. The sum of several low-rank tensors can be represented in the format (8). The representation of the sum can again be constructed from the representations of the individual low-rank tensors. This allows us to apply TT orthogonalization to evaluate all Frobenius-norms in Equation (7) in a fast and stable way.

Algorithm 2 Frobenius norm evaluation

Input: tensors $\mathcal{E}^1 \in \mathbb{R}^{1\times 2\times 2r}$, $\mathcal{E}^N \in \mathbb{R}^{2r\times 2\times 1}$ and $\mathcal{E}^j \in \mathbb{R}^{2r\times 2\times 2r}$ for $1 < j < N$
Output: $\|\mathcal{T}\|_F^2$, where $\mathcal{T} \in \mathbb{R}^{2^N}$ has entries $T_{i_1,\dots,i_N} = \sum_{k_1=1}^{r} \cdots \sum_{k_{N-1}=1}^{r} \mathcal{E}^1_{1,i_1,k_1} \mathcal{E}^2_{k_1,i_2,k_2} \cdots \mathcal{E}^N_{k_{N-1},i_N,1}$
Compute QR decomposition of $\mathcal{E}^1$ reshaped as element in $\mathbb{R}^{2\times 2r}$.
Set $\mathcal{E}^1$ to Q reshaped as element in $\mathbb{R}^{1\times 2\times 2r}$.
Update $\mathcal{E}^2$: reshape to $\mathbb{R}^{2r\times 2\cdot 2r}$, multiply with R from the left, reshape back to $\mathbb{R}^{2r\times 2\times 2r}$.
for $j = 2, \dots, N-1$
    Compute QR decomposition of $\mathcal{E}^j$ reshaped as element in $\mathbb{R}^{2r\cdot 2\times 2r}$.
    Set $\mathcal{E}^j$ to Q reshaped as element in $\mathbb{R}^{2r\times 2\times 2r}$.
    if $j < N-1$
        Update $\mathcal{E}^{j+1}$: reshape to $\mathbb{R}^{2r\times 2\cdot 2r}$, multiply with R from the left, reshape back to $\mathbb{R}^{2r\times 2\times 2r}$.
    else
        Update $\mathcal{E}^N$: reshape to $\mathbb{R}^{2r\times 2}$, multiply with R from the left, reshape back to $\mathbb{R}^{2r\times 2\times 1}$.
$\|\mathcal{T}\|_F^2 = \|\mathcal{E}^N\|_F^2$

3.3 Successive minimization

Let $r \in \mathbb{N}$. In order to approximate the solution of the minimization problem $\min_{\Phi\in\mathcal{T}_M^r} A_M^{\mathbf{u}}(\Phi)$, we decompose the problem into a sequence of successive minimization problems [1, 11, 25]. Let $\Phi \in H_M^r$ be represented as $\Phi = R^{(1)} + \cdots + R^{(r)}$ by rank-1 functions $R^{(k)} \in H_M^1$. The main idea of successive minimization is to first determine $R^{(1)}$ as solution of $\min_{R^{(1)}\in\mathcal{T}_M^1} A_M^{\mathbf{u}}(R^{(1)})$. In a successive step $R^{(2)}$ is determined as solution of $\min_{R^{(2)}\in\mathcal{T}_M^1} A_M^{\mathbf{u}}(R^{(1)} + R^{(2)})$. This is continued successively until Φ is determined. This idea is formalized in Algorithm 3.

Algorithm 3 Successive minimization

Input: rank r
Output: approximation $\Phi \approx \operatorname{argmin}_{\Phi\in\mathcal{T}_M^r} A_M^{\mathbf{u}}(\Phi)$
$\Phi = 0$
for $k = 1, \dots, r$
    $R^{(k)} = \operatorname{argmin}_{R\in\mathcal{T}_M^1} A_M^{\mathbf{u}}(\Phi + R)$
    $\Phi = \sum_{i=1}^{k} R^{(i)}$

3.4 Alternating least squares

In the successive minimization algorithm 3, we need to solve minimization problems of the form

$$\min_{R\in\mathcal{T}_M^1} A_M^{\mathbf{u}}(\Phi + R) \tag{9}$$

for given $\Phi \in H_M^r$. In the following, we introduce an alternating least squares algorithm [4, 6, 19, 27] to solve such minimization problems. The main idea is to approximate the solution of (9) by an iterative scheme which amounts to solving a sequence of small-dimensional linear problems. We start from an initial $R(\boldsymbol{\eta}) := \Pi_{\mathbf{s}\in S_M} R_{\mathbf{s}}(\eta_{\mathbf{s}})$. We first minimize $A^{\mathbf{u}}(\Phi + R)$ only with respect to a selected $R_{\mathbf{s}_0} : \{0,1\} \to \mathbb{R}$ for some $\mathbf{s}_0 \in S_M$ leaving the other $R_{\mathbf{s}}$, $\mathbf{s} \neq \mathbf{s}_0$ fixed. By partially evaluating $A^{\mathbf{u}}(\Phi + R)$ for all terms not depending on $R_{\mathbf{s}_0}$, we obtain that $\min_{R_{\mathbf{s}_0}\in\{\{0,1\}\to\mathbb{R}\}} A^{\mathbf{u}}(\Phi + R)$ with $R(\boldsymbol{\eta}) := R_{\mathbf{s}_0}(\eta_{\mathbf{s}_0})\Pi_{\mathbf{s}\in S_M\setminus\{\mathbf{s}_0\}} R_{\mathbf{s}}(\eta_{\mathbf{s}})$ is equivalent to a quadratic optimization problem

$$\min_{R_{\mathbf{s}_0}\in\{\{0,1\}\to\mathbb{R}\}} \alpha_1 R_{\mathbf{s}_0}(1)^2 + \alpha_2 R_{\mathbf{s}_0}(0)^2 + \alpha_3 R_{\mathbf{s}_0}(1)R_{\mathbf{s}_0}(0) + \alpha_4 R_{\mathbf{s}_0}(1) + \alpha_5 R_{\mathbf{s}_0}(0) + \alpha_6, \tag{10}$$

with constants $\alpha_1, \dots, \alpha_6 \in \mathbb{R}$ depending on the fixed $R_{\mathbf{s}}$, $\mathbf{s} \neq \mathbf{s}_0$ and Φ. This quadratic optimization problem always admits a unique optimal R_{s_0}, which is given by $R_{\mathbf{s}_0}(1) = a$ and $R_{\mathbf{s}_0}(0) = b$, where $a, b \in R$ are the solution of the linear system

$$\begin{pmatrix} 2\alpha_1 & \alpha_3 \\ \alpha_3 & 2\alpha_2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} -\alpha_4 \\ -\alpha_5 \end{pmatrix}. \tag{11}$$

This allows us to optimize $A_M^{\mathbf{u}}(\Phi + R)$ with respect to individual $R_{\mathbf{s}_0}$. By alternating the the selected $\mathbf{s}_0 \in S_M$, we obtain the alternating least squares algorithm, which is formalized in Algorithm 4.

Algorithm 4 Alternating least squares

Input: initial functions $R_{\mathbf{s}}^0 : \{0,1\} \to \mathbb{R}$ for $\mathbf{s} \in S_M$, function $\Phi \in H_M^r$, vector $\mathbf{u}$, tolerance ε
Output: approximation $R_{\text{opt}}(\boldsymbol{\eta}) = \Pi_{\mathbf{s} \in S_M} R_{\mathbf{s}}(\eta_{\mathbf{s}})$ of $\operatorname{argmin}_{R \in \mathcal{T}_M^1} A_M^u(\Phi + R)$
$v_{\mathsf{old}} = \infty$, $v_{\mathsf{new}} = A_M^{\mathbf{u}}(\Phi + R^0)$ with $R^0(\boldsymbol{\eta}) := \Pi_{\mathbf{s} \in S_M} R_{\mathbf{s}}^0(\eta_{\mathbf{s}})$
$\forall \mathbf{s} \in S_M,\ R_{\mathbf{s}} := R_{\mathbf{s}}^0$.
while $|v_{\mathsf{old}} - v_{\mathsf{new}}| > \varepsilon\, |v_{\mathsf{new}}|$.
    $v_{\mathsf{old}} = v_{\mathsf{new}}$
    for $\mathbf{s}_0 \in S_M$
        $R_{\mathbf{s}_0} = \operatorname{argmin}_{\widetilde{R}_{\mathbf{s}_0} : \{0,1\} \to \mathbb{R}} A^{\mathbf{u}}(\Phi + \widetilde{R})$ where $\widetilde{R}(\boldsymbol{\eta}) = \widetilde{R}_{\mathbf{s}_0}(\eta_{\mathbf{s}_0}) \Pi_{\mathbf{s} \in S_M \setminus \{\mathbf{s}_0\}} R_{\mathbf{s}}(\eta_{\mathbf{s}})$ for all $\boldsymbol{\eta} = (\eta_{\mathbf{s}})_{\mathbf{s} \in S_M}$
    $v_{\mathsf{new}} = A_M^{\mathbf{u}}(\Phi + R)$

Remark 3.2. To compute the constants α_i, we can either explicitly implement the partial evaluations of $A_M^{\mathbf{u}}(\Phi + R)$. Alternatively, we can treat $A_M^{\mathbf{u}}(\Phi + R)$ as a function in $\mathbb{R}^2 \to \mathbb{R}$ depending on the values $R_{\mathbf{s}_0}(0)$ and $R_{\mathbf{s}_0}(1)$. We know that this function is a multivariate-polynomial of the form $\alpha_1 R_{\mathbf{s}_0}(1)^2 + \alpha_2 R_{\mathbf{s}_0}(0)^2 + \alpha_3 R_{\mathbf{s}_0}(1) R_j(0) + \alpha_4 R_{\mathbf{s}_0}(1) + \alpha_5 R_{\mathbf{s}_0}(0) + \alpha_6$. The constants can be computed using multivariate-polynomial interpolation in six points. This interpolation has the advantages that it is non-intrusive and that the evaluations of $A_M^{\mathbf{u}}(\Phi + R)$ can be performed efficiently using the ideas of Section 3.2.

Remark 3.3. Throughout this work, we use approximations in the CP-format. The matrix product state representation [28] also known as tensor train format [26] would present an alternative low-rank format, for which minimization problems can also be solved using alternating algorithms [11, 14, 33].

3.5 Monte Carlo methods

Let $\Phi \in H_M^r$ denote an approximation of the solution of (4). In the following, we discuss how to evaluate $A_{M,\ell}^{\mathbf{u}}(\Phi)$. In the definition of $A_{M,\ell}^{\mathbf{u}}$ the function

$$f(\boldsymbol{\eta}) := \sum_{k=1}^{K} p_k \left((1 - \eta_{\mathbf{v}_k}) \left(\mathbf{u} \cdot \mathbf{v}_k + \Psi(\boldsymbol{\eta}^{0,\mathbf{v}_k}) - \Psi(\boldsymbol{\eta}) \right)^2 + \frac{1}{2} \sum_{\substack{\mathbf{y} \in S \setminus \{\mathbf{0}\} \\ \mathbf{y} + \mathbf{v}_k \neq \mathbf{0}}} \left(\Psi(\boldsymbol{\eta}^{\mathbf{y}+\mathbf{v}_k,\mathbf{y}}) - \Psi(\boldsymbol{\eta}) \right)^2 \right) \tag{12}$$

is evaluated for all $\boldsymbol{\eta} \in C_{M,\ell}$. For larger N and most choices of ℓ evaluating $|C_{M,\ell}| = \binom{N}{\ell}$ terms is intractable. We thus propose to use a Monte Carlo method [23] to approximate $A_{M,\ell}^{\mathbf{u}}(\Phi)$.

In a naive Monte Carlo method we compute N_s samples $\boldsymbol{\eta}^{(i)} \in C_{M,\ell}$ for $1 \leq i \leq N_s$ and replace $\frac{1}{|C_{M,\ell}|} \sum_{\boldsymbol{\eta} \in C_{M,\ell}}$ by $\frac{1}{N_s} \sum_{\boldsymbol{\eta}^{(1)}, \dots, \boldsymbol{\eta}^{(N_s)}}$. In Algorithm 5, we define a Monte Carlo method with additional variance reduction [12], which as demonstrated in Section 4 reduces the number of samples needed to obtain a given approximation accuracy. The main idea is to observe that the sites $\mathbf{v}_1, \dots, \mathbf{v}_K$ play a special role since they are the most relevant for jumps of the tagged particle. Instead of sampling the $\boldsymbol{\eta}$ uniformly in $C_{M,\ell}$, we now ensure that all possible states of the sites $\mathbf{v}_1, \dots, \mathbf{v}_K$ are occurring equally often in the set of sample points.

Algorithm 5 Monte Carlo method with variance reduction

1: **Input**: function $f : C_{M,\ell} \to \mathbb{R}$, parameter $\widetilde{N}_s$
2: **Output**: approximation S of $\frac{1}{|C_{M,\ell}|} \sum_{\boldsymbol{\eta} \in C_{M,\ell}} f(\boldsymbol{\eta})$
3: $S = 0$
4: **for** $\vec{\mathbf{w}}^{(1)} \in \{0,1\}^K$
5: $\quad n_1 := ||\vec{\mathbf{w}}||_1$, $n_2 := \ell - n_1$
6: $\quad$ **if** $0 \leq n_2 \leq N - \ell$
7: $\quad\quad \tilde{S} = 0$
8: $\quad\quad$ **for** $i = 1, \ldots, \widetilde{N}_s$
9: $\quad\quad\quad$ Sample $\vec{\mathbf{w}}^{(2)} \in \{0,1\}^{S_M \setminus \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}}$ such that $||\vec{\mathbf{w}}^{(2)}||_1 = n_2$.
10: $\quad\quad\quad$ Construct $\boldsymbol{\eta} \in C_{M,\ell}$ such that $\boldsymbol{\eta}(\mathbf{s}) = \begin{cases} \vec{\mathbf{w}}_k^{(1)} & \mathbf{s} = \mathbf{v}_k \\ \vec{\mathbf{w}}_{\mathbf{s}}^{(2)} & \text{otherwise} \end{cases}$.
11: $\quad\quad\quad \tilde{S} = \tilde{S} + f(\boldsymbol{\eta})$
12: $\quad\quad S = S + \binom{N-K}{n_2} \cdot \tilde{S} / \widetilde{N}_s$
13: $S = \frac{S}{\binom{N}{\ell}}$

Remark 3.4. In line 9 of Algorithm 5, we sample form the set set $\{\vec{\mathbf{w}}^{(2)} \in \{0,1\}^{S_M \setminus \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}} | ||\vec{\mathbf{w}}^{(2)}||_1 = n_2\}$ which contains $\binom{N-K}{n_2}$ elements. When the number of elements in this set is smaller than $\widetilde{N}_s$, we can compute $\tilde{S}$ based on all possible $\vec{\mathbf{w}}^{(2)}$ instead of sampling $\widetilde{N}_s$ times. This decreases the variance of the approximation further and reduces the number of required evaluations of f.

Remark 3.5. Lines 12 and 13 of Algorithm 5 potentially lead to stability issues in floating point arithmetic when $\binom{N-K}{n_2}$ or $\binom{N}{\ell}$ are large. This problem can be mitigated using the log-sum-exp trick.

3.6 Limitations of the approach

In this section, we briefly discuss the limitations of the proposed algorithm. These are primarily related to the following observation. Since $A^{\mathbf{u}}_{M,\ell}(\Psi^{M,\mathbf{u}}_{\mathsf{opt}})$ defines the entries of the self-diffusion coefficient (3), we know that $A^{\mathbf{u}}_{M,\ell}(\Psi^{M,\mathbf{u}}_{\mathsf{opt}}) \in [0,1]$. Note that $A^{\mathbf{u}}_{M,\ell}$ contains the normalization constant $|C_{M,\ell}|^{-1}$. The objective function A^u_M (5) does not include a normalization factor. This implies that the $A^{\mathbf{u}}_M(\Psi^{M,\mathbf{u}}_{\mathsf{opt}})$ is of order 2^N. Trying to minimize this objective function numerically using floating point arithmetic leads to issues caused by rounding errors for large N.

For larger values of the objective function, it becomes increasingly challenging to solve the individual minimization problems in line 8 of Algorithm 4. In particular, our approach of using polynomial interpolation to find the location of the minimum might encounter numerical rounding issues. We find that the positive eigenvalues of the 2×2 matrix in (11) tend to be many orders of magnitude smaller than the norm of the right hand side. This implies that small rounding errors in the constants $\alpha_1, \ldots, \alpha_6$ might lead to vastly different solutions. For $M > 2$, we even find that rounding errors can lead to negative eigenvalues in the system, i.e. we can not find the minimum of the polynomial (10) at all. It might be necessary to use a different approach to solve the individual ALS minimization problems for larger M. Alternatively, one could develop an algorithm to minimize the function $\log(A^{\mathbf{u}}_M)$.

Moreover, the number of ALS iterations needed to find a good rank-1 minimizer tends to increase when increasing the size of the domain. This is especially true when a random initialization is used in Algorithm 4. Note that it might be possible to circumvent this issue by initializing Algorithm 4 based on the solution computed for a smaller value of M. Overcoming these limitations will then require further investigation which we intend to carry out in a following work. A possible path could be to combine domain decomposition approaches together with the tensor-based optimization algorithm we propose here, in order to obtain a

global optimization procedure where only independent parallel local optimization problems on medium-sized cells are solved.

4 Numerical Experiments

In the following numerical experiments [1], we consider the a jumping model with $K = 4$ displacement vectors $\mathbf{v}_1 = (1, 0)$, $\mathbf{v}_2 = (-1, 0)$, $\mathbf{v}_3 = (0, 1)$, and $\mathbf{v}_4 = (0, -1)$ and with associated probability $p_k = 1/4$ for $k = 1, 2, 3, 4$. All computing times are measured without parallelization in MATLAB R2018b on a Lenovo Thinkpad T480s with Intel Core i7-8650U CPU and 15.4 GiB RAM. Note that both the sampling Algorithm 1 and Algorithm 5 as well as the evaluation of the different Frobenius-norms in Equation (7) can be parallelized to speed up computations.

4.1 Solving the optimization problem

In the following, we analyze the numerical approximation of the self-diffusion coefficient by solving the optimization problem (3).

Low-rank approximation error First, we study how the rank r affects $\min_{\Phi \in H_M^r} A_M^{\mathbf{u}}(\Phi)$ for $\mathbf{u} = (1, 0)$. Let $\Phi_{\mathsf{ALS}}^{r,\mathbf{u}} \in H_M^r$ denote the function obtained by using Algorithm 3 with rank r, where the minimization problem in each iteration is solved using Algorithm 4. The initial functions $R_{\mathbf{s}}^0$ in Algorithm 4 are selected randomly by assigning random values drawn form the uniform distribution on $[0, 1]$ to $R_{\mathbf{s}}^0(0), R_{\mathbf{s}}^0(1)$. In Algorithm 4, we set $\varepsilon = 10^{-12}$ and additionally stop after line 8 has been executed 420 times.

We depict the error of $\Phi_{\mathsf{ALS}}^{r,\mathbf{u}}$ for various r in Figure 2. The computation with $r = 10$ takes 4 minutes for $M = 1$ and 82 minutes for $M = 2$. We observe that the error decays quickly with increasing rank. In particular, for $M = 1$ an approximation with $r = 1$ is less than 0.1% away from a direct least squares solution of the minimization problem (4). Such a direct solution is intractable for $M = 2$.

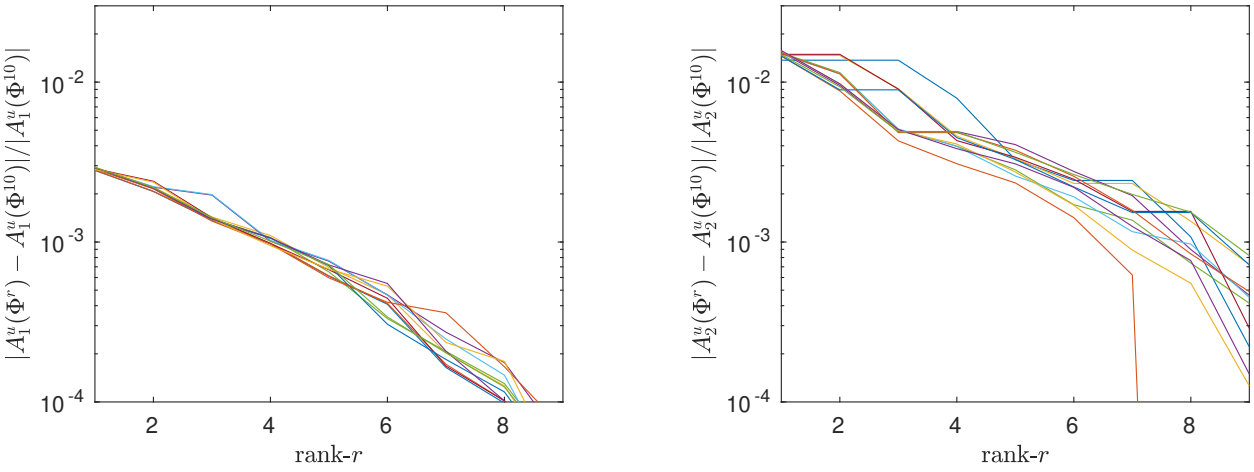

Figure 2: Algorithm 3 yields successive approximations $\Phi_{\mathsf{ALS}}^{r,\mathbf{u}} \in H_M^r$. For $r \in \{1, \ldots, 9\}$, we plot the relative error of $\Phi_{\mathsf{ALS}}^{k,\mathbf{u}}$ compared to $\Phi_{\mathsf{ALS}}^{10,\mathbf{u}}$. We repeat this experiment 12 times with different random initial functions in Algorithm 4. The resulting relative errors are displayed in different colors. Left: $M = 1$. Right: $M = 2$.

[1] The code to reproduce all experiments is available on `https://github.com/cstroessner/SelfDiffusion`

Sampling-based evaluation Let $M = 2$ and $\mathbf{u} = (1, 0)$. For a given approximate solution $\Phi^{3,\mathbf{u}}_{\mathsf{ALS}}$, we want to evaluate $A^{\mathbf{u}}_{2,\ell}(\Phi^{3,\mathbf{u}}_{\mathsf{ALS}})$. A single evaluation of (12) with Ψ replaced by $\Phi^{3,\mathbf{u}}_{\mathsf{ALS}}$ requires on average around $5.1 \cdot 10^{-4}$ seconds. Computing $A^{\mathbf{u}}_{2,\ell}(\Phi^{3,\mathbf{u}}_{\mathsf{ALS}})$ for $0 \leq \ell \leq N$ directly would require $2^{24} \approx 1.6 \cdot 10^7$ evaluations of (12), i.e. around 2.4 hours. In Section 3.5, we proposed two Monte Carlo algorithms to obtain approximations of $A^{\mathbf{u}}_{2,\ell}(\Phi^{3,\mathbf{u}}_{\mathsf{ALS}})$. We visualize the variance in the approximation obtained by these algorithms in Figure 3. The studied quantity $\frac{2}{N+1}\sum_{\ell=0}^{N} \mathbf{u}^T \mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$ approximates $\int_0^1 \mathrm{Tr}(\mathbb{D}_s(\overline{\rho}))d\overline{\rho}$, where Tr denotes the trace operator. Algorithm 5 clearly leads to a variance reduction. With only 10^5 samples, which can be evaluated in about one minute, we can already reach a variance of 10^{-6}.

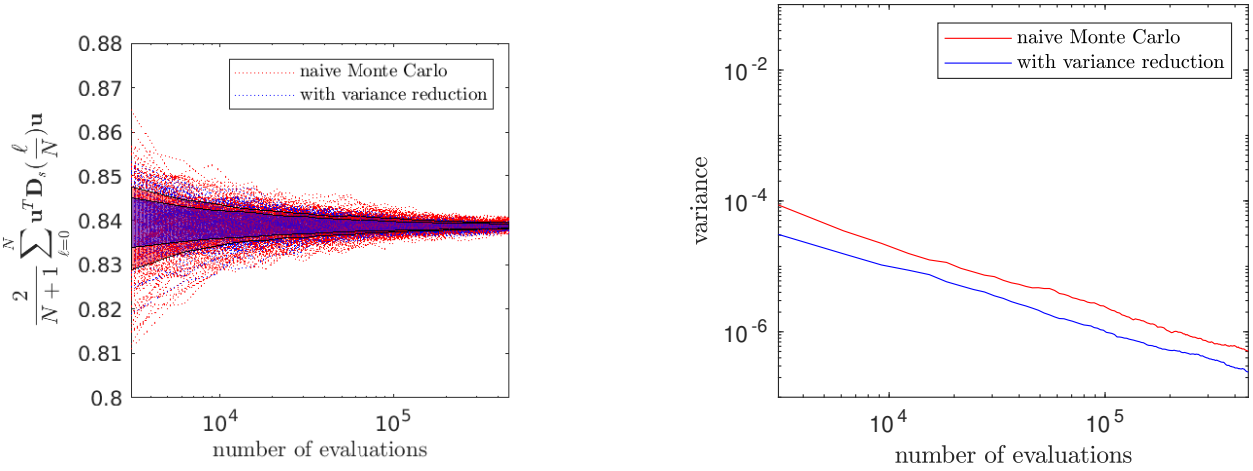

Figure 3: For $M = 2$ and $\mathbf{u} = (1, 0)$ we compute $\Phi^{3,\mathbf{u}}_{\mathsf{ALS}}$ using Algorithm 3. We then approximate $\mathbf{u}^T \mathbb{D}_s(\frac{\ell}{N})\mathbf{u} = 2A^{\mathbf{u}}_{2,\ell}(\Phi^{3,\mathbf{u}}_{\mathsf{ALS}})$ for $0 \leq \ell \leq N$ using sampling as in Section 3.5. We sample $\boldsymbol{\eta} \in C_{M,\ell}$ using two different approaches; from the uniform distribution (naive Monte Carlo) and using Algorithm 5 (with variance reduction). We use a total of 460 800 evaluations of (12) for both sampling methods. After every 3 072 evaluations, we evaluate $\frac{2}{N+1}\sum_{\ell=0}^{N} \mathbf{u}^T \mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$ based on the current approximation. The whole experiment is repeated 250 times. Left: Evolution of $\frac{2}{N+1}\sum_{\ell=0}^{N} \mathbf{u}^T \mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$ with increasing number of samples. The shaded areas mark one standard deviation from the mean for the respective algorithm. Right: Evolution of the variance of $\frac{2}{N+1}\sum_{\ell=0}^{N} \mathbf{u}^T \mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$ with increasing number of samples.

4.2 Estimation of long-time mean square deviation

In the following, we study estimating the long-term limit using Algorithm 1.

Figure 4 motivates our choice of $T = 300$ and $\hat{N}_s = 30\,000$ for the following numerical experiments. The error caused by stopping with $T = 300$ is negligible compared to the stochastic variance when stopping with $\hat{N}_s = 30\,000$.

In Figure 5 we analyze how the parameter $\hat{N}_s$ affects an approximation of $\frac{2}{N+1}\sum_{\ell=0}^{N} \mathbf{u}^T \mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$. We observe that the quantity of interest converges to the same value around 0.84 as in Figure 3. Note that a single run with $\hat{N}_s = 30\,000$ requires 33 minutes and reaches a variance of 10^{-4}. Compared to the results in Figure 3 the computational time needed to to achieve the same variance with Algorithm 1 compared to Algorithm 5 is much higher. This is visualized in Figure 6.

4.3 Comparison of algorithms

We now compare the two approaches of approximating the self-diffusion coefficient. We evaluate $\mathbf{u}^T \mathbb{D}_s(\ell/N)\mathbf{u}$ for $\mathbf{u} \in \{(1, 0), (0, 1), (1, 1)\}$ to recover all entries of the symmetric matrix $\mathbb{D}_s(\ell/N)$. Interpolation of the

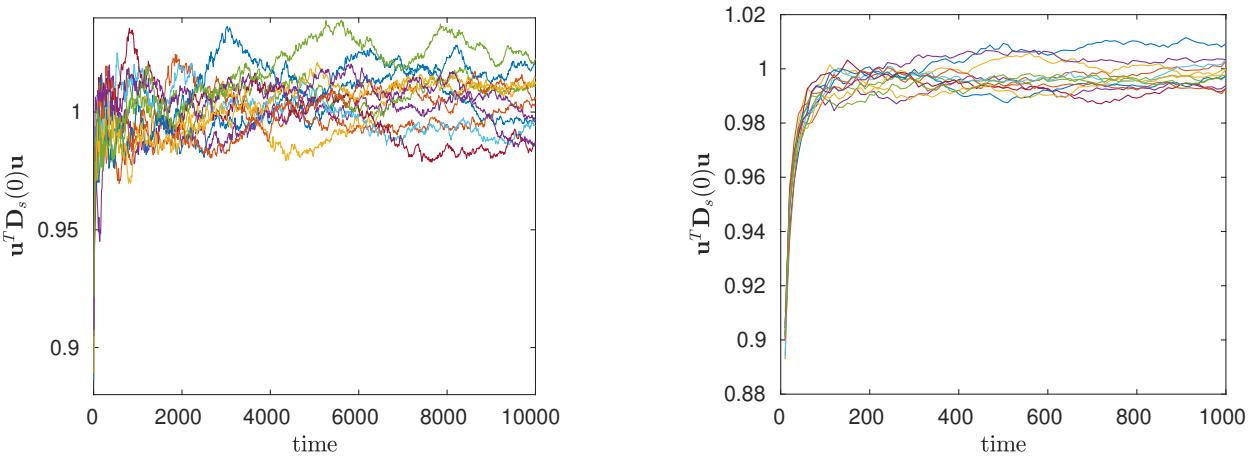

Figure 4: We run Algorithm 1 with $M = 2$, $\mathbf{u} = (1, 0)$, $\ell = 0$ and different values for T and $\hat{N}_s$. For each setting, we plot how the approximation of the value $\mathbf{u}^T\mathbb{D}_s(0)\mathbf{u}$, which is known to be equal to 1, evolves over time for 12 random initializations visualized in different colors. Left: $\hat{N}_s = 10\,000$, $T = 10\,0000$. Right: $\hat{N}_s = 100\,000$, $T = 1\,000$.

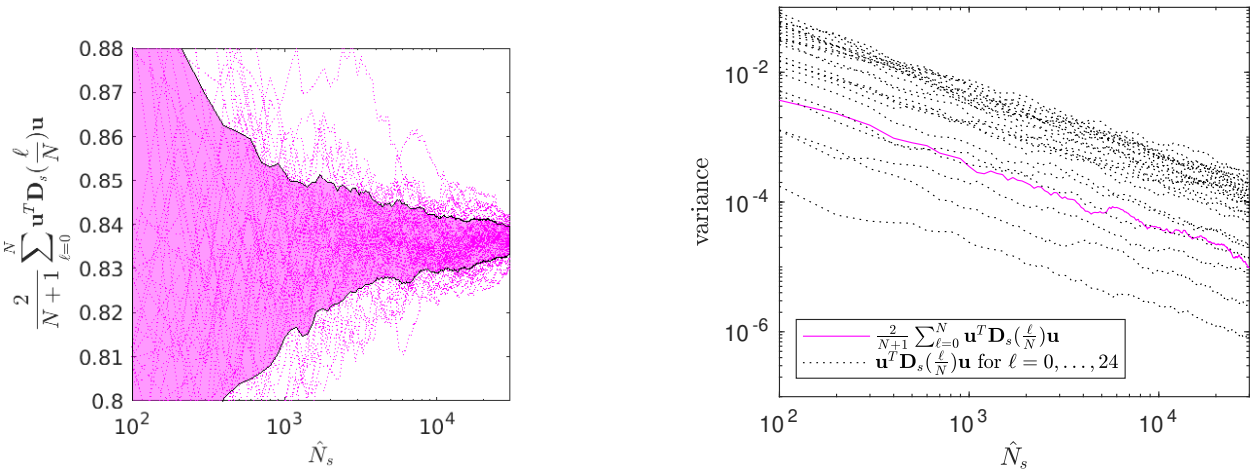

Figure 5: We run Algorithm 1 with $T = 300$, $\mathbf{u} = (1, 0)$ for $\ell = 0, \ldots, 24$ to approximate $\mathbb{D}_s(\frac{\ell}{N})$. We use $\hat{N}_s = 30\,000$ and store intermediate values for $\hat{N}_s = 100, 200, 300, \ldots$. Based on these approximations we compute $\frac{2}{N+1}\sum_{\ell=0}^{N}\mathbf{u}^T\mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$. This procedure is repeated 50 times with different random initializations. Left: Evolution of $\frac{2}{N+1}\sum_{\ell=0}^{N}\mathbf{u}^T\mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$ with increasing values of $\hat{N}_s$. The shaded area marks one standard deviation from the mean. Right: Evolution of the variance of $\frac{2}{N+1}\sum_{\ell=0}^{N}\mathbf{u}^T\mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$ with increasing values of $\hat{N}_s$. We additionally display the evolution of the variance of $\mathbf{u}^T\mathbb{D}_s(\frac{\ell}{N})\mathbf{u}$ for individual values of ℓ.

entries yields an approximation of $\mathbb{D}_s(\rho)$ for $\rho \in [0, 1]$. Alternatively, interpolation in the space of symmetric positive definite matrices could be used [24].

For the following experiments, we use Algorithm 1 with a slight modification to approximate $\mathbf{u}^T\mathbb{D}_s(\overline{\rho})\mathbf{u}$ for different values of $\mathbf{u}$ simultaneously. This is achieved by keeping track of different α for different $\mathbf{u}$ in lines 15 and 16. The optimization problem (4) needs to be solved for every $\mathbf{u}$ separately, but the solution can be evaluated for different ℓ. In contrast, the sampling in Algorithm 1 needs to be redone completely for every ℓ.

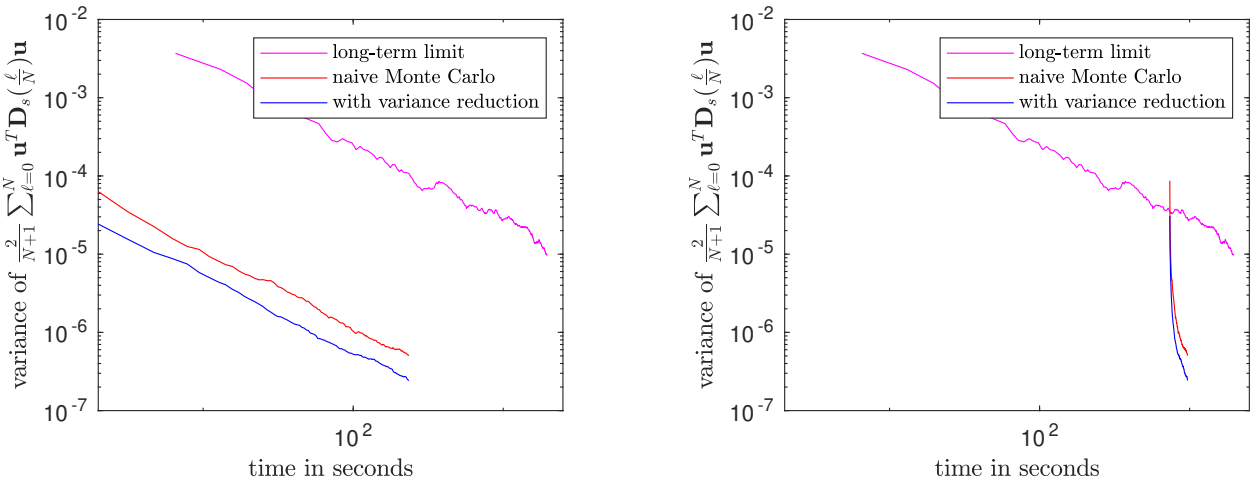

Figure 6: We plot the variances displayed on the right of Figure 3 and Figure 5. Instead of the number of samples, we display the computational time for evaluating the corresponding number of samples on the x-axis. Left: Computational times only for Algorithm 1, the naive Monte Carlo method and Algorithm 5. Right: We additionally include the computational time of 736 seconds for computing $\Phi_{\mathsf{ALS}}^{3,\mathbf{u}}$ using Algorithm 3.

In Figure 7, we display the trace of $\mathbb{D}_s(\rho)$ computed using different approaches. Computing the whole matrix $\mathbb{D}_s(\ell/N)$ for all $0 \leq \ell \leq N$ took 3 minutes (37 minutes) for the approach based on solving the minimization problem, whereas using Algorithm 1 took 15 minutes (45 minutes) for $M = 1$ ($M = 2$). The much higher variance of Algorithm 1 for approximating the long-term limit leads to clearly visible changes in the graph of trace at the values $\rho = \frac{\ell}{N}$. The figure also contains results using a 6×6 grid ($N = 35$) for which the minimization based approach took 5 hours and 40 and the sampling of the long-term limit took 9 hours and 24 minutes. The sampled solution of the minimization problem changes less due to a smaller sampling variance in Algorithm 5. The variance for both approaches is depicted in Figure 8 and listed in Table1. We want to emphasize that the approach based on solving the minimization problem is faster and simultaneously yields a lower variance in the entries of the self-diffusion compared to the estimation of the long-term limit.

		$\max_{\ell\in\{0,\dots,N\}} \mathrm{Var}(\mathrm{Tr}(D_s(\ell/N)))$	$\frac{1}{N+1}\sum_{\ell=0}^{N} \mathrm{Var}(\mathrm{Tr}(D_s(\ell/N)))$
N = 8	long-term limit	$2.313 \cdot 10^{-4}$	$1.072 \cdot 10^{-4}$
	minimization	$8.791 \cdot 10^{-9}$	$4.231 \cdot 10^{-9}$
N = 15	long-term limit	$3.433 \cdot 10^{-4}$	$1.410 \cdot 10^{-4}$
	minimization	$2.548 \cdot 10^{-4}$	$6.835 \cdot 10^{-5}$
N=24	long-term limit	$4.377 \cdot 10^{-4}$	$1.989 \cdot 10^{-4}$
	minimization	$3.262 \cdot 10^{-4}$	$8.001 \cdot 10^{-5}$

Table 1: We display the mean and maximum of the variance for the 100 approximations of the self-diffusion coefficient computed in Figure 8. Note that we do not need to sample to evaluate $A_{M,\ell}^{\mathbf{u}}$ for $N = 8$. The variance for $N = 8$ is solely caused by the random initializations of $R_{\mathbf{s}}^0$ in Algorithm4.

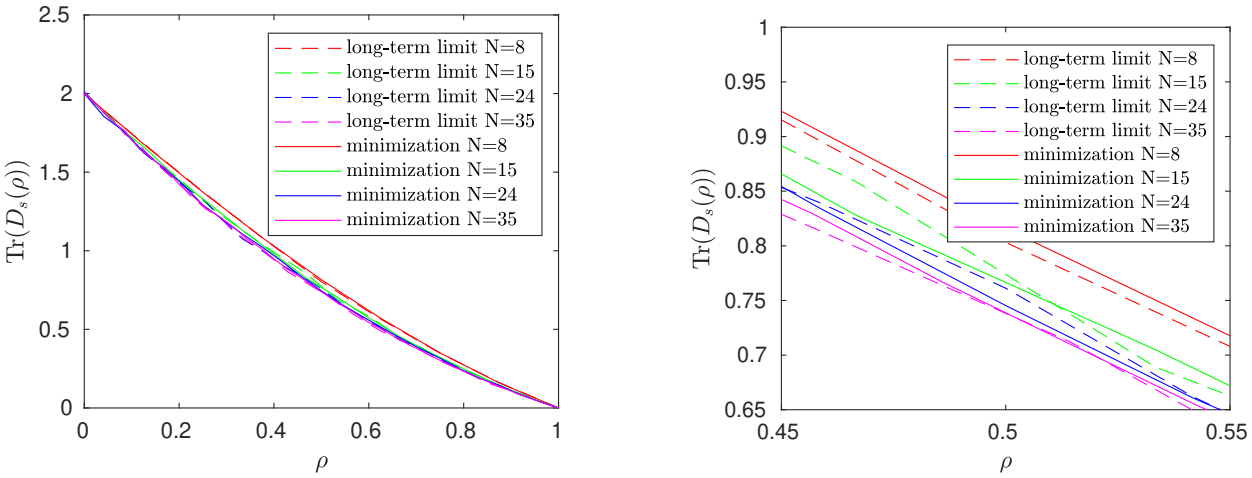

Figure 7: For $M = 1$ ($N = 8$) and $M = 2$ ($N = 24$) we solve the minimization problem (4) using Algorithm 3 with $r = 3$. We evaluate $A^{\mathbf{u}}_{M,\ell}$ directly for $M = 1$, whereas for $M = 2$ we use Algorithm 5 with $\widetilde{N}_s = 50$. We compare to Algorithm 1 to estimate the long-term limit (2) with $\hat{N}_s = 30\,000$ and $T = 300$ for $M = 1$ and $M = 2$ to compute $\mathbb{D}_s(\ell/N)$. Repeating this for $\mathbf{u} \in \{(1,0),(1,1),(0,1)\}$ yields $\mathbb{D}_s(\ell/N)$. Element-wise linear interpolation allows us to evaluate $\mathbb{D}_s(\rho)$ for $\rho \in [0,1]$. We plot the trace of $\mathbb{D}_s(\rho)$. In addition, we run both algorithms on a 4×4 and 6×6 periodic grid ($N = 15$ and $N = 35$). For $N = 15$ we use the same parameters as for $N = 24$. For $N = 35$ we use rank 10, $\widetilde{N}_s = 150$ and $\hat{N}_s = 80\,000$, $T = 600$. Note that the 4×4 grid lies in between the 3×3 grid for $M = 1$ and the 5×5 grid for $M = 2$. Right: Zoom on part of the graphs.

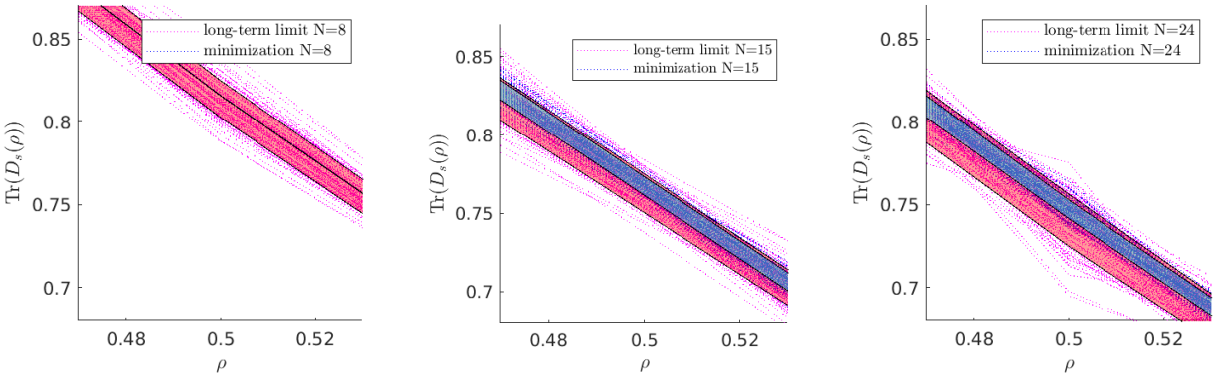

Figure 8: We repeat the experiment presented in Figure 7 100 times for $N = 8$, $N = 15$ and $N = 24$. We plot a zoom on a part of the graphs. The shaded areas mark one standard deviation from the mean for each of the methods.

5 Conclusion

Classical computational methods used for the approximation of the self-diffusion coefficient of a tagged particle process consist in exploiting the fact that the coefficient can be expressed as the long-time limit of the mean-square deviation of the tagged particle process (2). These standard numerical methods consist in truncating the computational domain and approximate the long-time limit of the mean square deviation by an empirical average computed with a standard Monte Carlo approach from a large number of realizations of trajectories of the tagged particle. However, this classical approach suffers from two drawbacks: first,

the amount of statistical noise due to the Monte-Carlo approach is very significant, as illustrated in our experiment, and yields larger errors than the error linked to the truncation of the computational domain. Indeed, it has been proved in [21], that the error in the approximation of the self-diffusion matrix decays exponentially with respect to the size of the finite computational domain. In contrast, the error due the statistical noise in Monte-Carlo averages only decays as the inverse of the square root of the number of random samples of trajectories. In addition, this traditional Monte Carlo approach requires to choose a priori a finite value of the time horizon, which induces additional errors which are difficult to estimate.

In this work, we have proposed a new approach to compute the self-diffusion matrix: we exploit the fact that the latter quantity can be expressed as the solution of a high-dimensional deterministic minimization problem (1) and we use tensor methods in order to build a numerical approximation of the solution of this problem. Here, the low-rank solutions are computed by means of an alternating optimization scheme.

Our numerical experiments demonstrate that our approach is very advantageous compared to the classical Monte Carlo method. Indeed, with the same amount of computational resources, the variance in the obtained self-diffusion matrix is much smaller when using our new approach. Since the statistical noise is the leading source of errors in the approximation of this self-diffusion matrix, the low-rank approach proposed here provides a very interesting alternative to standard Monte-Carlo methods.

This preliminary study yields several interesting remaining open questions, which we wish to investigate in future works. First, given the current numerical implementation of the low-rank algorithm proposed here, we are limited in terms of size of computational domains for which the present algorithm can be run because of round-off errors. We still believe that the approach described here is promising since, again, the errors linked to the truncation of the computational domain are small in comparison to statistical errors. However, we are not aware of any theoretical results on how to select the approximation rank, the number of ALS iterations and the number of samples depending on the size of the domain and the desired approximation error. These parameters are crucial for the computational time required to approximate the self-diffusion matrix on larger domains. In order to extend our approach to three-dimensional systems, it might be possible to combine the present method together with adapted domain decomposition approaches. We plan to study this in the future.

Another very interesting question is the extension of the proposed approach to continuous-state diffusion processes, like Langevin dynamics in molecular dynamics for instance. We see the present work as a preliminary step towards the computation of diffusion coefficients out of Kinetic Monte-Carlo simulations, which can be fed with continuous space Molecular Dynamics simulations. An example of a possible track one could follow in this directions could be to introduce a discretization grid of the continuous space and compute approximate jump probability rates and occupancy probabilities of each cell of the discretization grid together with continuous state stochastic dynamics. This will require however, as a first step, the extension of the proposed approach to tagged particle systems where each grid site has the possibility to be occupied by more than one particle. We intend to investigate this issue in a future work.

Acknowledgments

This work was supported by the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement number 614492 and under the European Union's Horizon 2020 Research and Innovation Programme, ERC Grant Agreement number 810367, project EMC2. The work was initiated during the CEMRACS 2021 summer school at CIRM, Luminy, Marseille. The authors also acknowledge funding by the ANR project COMODO (ANR-19-CE46-0002), the Center on Energy and Climate Change (E4C) and the I-Site FUTURE.

The authors would like to thank Mi-Song Dupuy, Guillaume Enchéry, Daniel Kressner and Dominik Schmid for stimulating discussions on this work.

References

[1] A. Ammar, F. Chinesta, E. Cueto, and M. Doblaré, *Proper generalized decomposition of time-multiscale models*, Internat. J. Numer. Methods Engrg., 90 (2012), pp. 569–596.

[2] G. Beylkin and M. J. Mohlenkamp, *Algorithms for numerical analysis in high dimensions*, SIAM J. Sci. Comput., 26 (2005), pp. 2133–2159.

[3] O. Blondel and C. Toninelli, *Kinetically constrained lattice gases: tagged particle diffusion*, Ann. Inst. Henri Poincaré Probab. Stat., 54 (2018), pp. 2335–2348.

[4] J. D. Carroll and J.-J. Chang, *Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition*, Psychometrika, 35 (1970), pp. 283–319.

[5] D. Chen, P. Chen, N. Gantert, and D. Schmid, *Limit theorems for the tagged particle in exclusion processes on regular trees*, Electron. Commun. Probab., 24 (2019), pp. 1–10.

[6] L. De Lathauwer and D. Nion, *Decompositions of a higher-order tensor in block terms. III. Alternating least squares algorithms*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1067–1083.

[7] A. De Masi and E. Presutti, *Mathematical methods for hydrodynamic limits*, vol. 1501 of Lecture Notes in Mathematics, Springer-Verlag, Berlin, 1991.

[8] H. P. Deutsch and K. Binder, *Interdiffusion and self-diffusion in polymer mixtures: A Monte Carlo study*, J. Chem. Phys., 94 (1991), pp. 2294–2304.

[9] R. Ferrando and E. Scalas, *Self-diffusion in a 2D lattice gas with lateral interactions*, Surf. Sci., 281 (1993), pp. 178–190.

[10] N. Gantert and D. Schmid, *The speed of the tagged particle in the exclusion process on Galton-Watson trees*, Electron. J. Probab., 25 (2020), pp. Paper No. 71, 27.

[11] L. Grasedyck, D. Kressner, and C. Tobler, *A literature survey of low-rank tensor approximation techniques*, GAMM-Mitt., 36 (2013), pp. 53–78.

[12] J. M. Hammersley and D. C. Handscomb, *Monte Carlo methods*, Methuen & Co., Ltd., London; Barnes & Noble, Inc., New York, 1965.

[13] S. Hao and D. S. Sholl, *Self-diffusion and macroscopic diffusion of hydrogen in amorphous metals from first-principles calculations*, J. Chem. Phys., 130 (2009), p. 244705.

[14] S. Holtz, T. Rohwedder, and R. Schneider, *The alternating linear scheme for tensor optimization in the tensor train format*, SIAM J. Sci. Comput., 34 (2012), pp. A683–A713.

[15] M. Jara, *Finite-dimensional approximation for the diffusion coefficient in the simple exclusion process*, Ann. Probab., 34 (2006), pp. 2365–2381.

[16] H. A. L. Kiers, *Towards a standardized notation and terminology in multiway analysis*, J. Chemom., 14 (2000), pp. 105–122.

[17] C. Kipnis, C. Landim, and S. Olla, *Hydrodynamical limit for a nongradient system: the generalized symmetric exclusion process*, Commun. Pure Appl. Math., 47 (1994), pp. 1475–1545.

[18] C. Kipnis and S. R. S. Varadhan, *Central limit theorem for additive functionals of reversible Markov processes and applications to simple exclusions*, Comm. Math. Phys., 104 (1986), pp. 1–19.

[19] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 51 (2009), pp. 455–500.

[20] T. Komorowski, C. Landim, and S. Olla, *Fluctuations in Markov processes*, vol. 345 of Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Springer, 2012.

[21] C. Landim, S. Olla, and S. R. S. Varadhan, *Finite-dimensional approximation of the self-diffusion coefficient for the exclusion process*, Ann. Probab., 30 (2002), pp. 483–508.

[22] T. M. Liggett, *Stochastic interacting systems: contact, voter and exclusion processes*, vol. 324 of Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Springer-Verlag, Berlin, 1999.

[23] N. Metropolis and S. Ulam, *The Monte Carlo method*, J. Amer. Statist. Assoc., 44 (1949), pp. 335–341.

[24] M. Moakher and P. G. Batchelor, *Symmetric positive-definite matrices: from geometry to applications and visualization*, in Visualization and processing of tensor fields, Math. Vis., Springer, Berlin, 2006, pp. 285–298, 452.

[25] A. Nouy, *Low-rank methods for high-dimensional approximation and model order reduction*, in Model reduction and approximation, vol. 15 of Comput. Sci. Eng., SIAM, Philadelphia, PA, 2017, pp. 171–226.

[26] I. V. Oseledets, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.

[27] I. V. Oseledets, M. V. Rakhuba, and A. Uschmajew, *Alternating least squares as moving subspace correction*, SIAM J. Numer. Anal., 56 (2018), pp. 3459–3479.

[28] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, *Matrix product state representations*, Quantum Inf. Comput., 7 (2007), pp. 401–430.

[29] J. Quastel, *Diffusion of color in the simple exclusion process*, Comm. Pure Appl. Math., 45 (1992), pp. 623–679.

[30] T. Rohwedder and A. Uschmajew, *On local convergence of alternating schemes for optimization of convex problems in the tensor train format*, SIAM J. Numer. Anal., 51 (2013), pp. 1134–1162.

[31] E. Saada, *A limit theorem for the position of a tagged particle in a simple exclusion process*, Ann. Probab., 15 (1987), pp. 375–381.

[32] E. Sanz and D. Marenduzzo, *Dynamic Monte Carlo versus Brownian dynamics: A comparison for self-diffusion and crystallization in colloidal fluids*, J. Chem. Phys., 132 (2010), p. 194102.

[33] S. Szalay, M. Pfeffer, V. Murg, G. Barcza, F. Verstraete, R. Schneider, and O. Legeza, *Tensor product methods and entanglement optimization for ab initio quantum chemistry*, Int. J. Quantum Chem., 115 (2015), pp. 1342–1391.

[34] A. L. Thorneywork, R. E. Rozas, R. P. A. Dullens, and J. Horbach, *Effect of hydrodynamic interactions on self-diffusion of quasi-two-dimensional colloidal hard spheres*, Phys. Rev. Lett., 115 (2015), p. 268301.

[35] C. Toninelli, G. Biroli, and D. S. Fisher, *Spatial structures and dynamics of kinetically constrained models of glasses*, Phys. Rev. Lett., 92 (2004), p. 185504.