Noname manuscript No.
(will be inserted by the editor)

The PDD method for solving linear, nonlinear, and fractional PDEs problems

Ángel Rodríguez-Rozas · Juan A. Acebrón · Renato Spigler

Received: date / Accepted: date

Abstract We review the Probabilistic Domain Decomposition (PDD) method for the numerical solution of linear and nonlinear Partial Differential Equation (PDE) problems. This Domain Decomposition (DD) method is based on a suitable probabilistic representation of the solution given in the form of an expectation which, in turns, involves the solution of a Stochastic Differential Equation (SDE). While the structure of the SDE depends only upon the corresponding PDE, the expectation also depends upon the boundary data of the problem.

The method consists of three stages: (i) only few values of the sought solution are solved by Monte Carlo or Quasi-Monte Carlo at some interfaces; (ii) a continuous approximation of the solution over these interfaces is obtained via interpolation; and (iii) prescribing the previous (partial) solutions as additional Dirichlet boundary conditions, a fully decoupled set of sub-problems is finally solved in parallel.

For linear parabolic problems, this is based on the celebrated Feynman-Kac formula, while for semilinear parabolic equations requires a suitable generalization based on branching diffusion processes. In case of semilinear transport equations and the Vlasov-Poisson system, a generalization of the probabilistic representation was also obtained in terms of the Method of Characteristics (characteristic curves). Finally, we present the latest progress towards the extension of the PDD method for nonlocal fractional operators.

The algorithm notably improves the scalability of classical algorithms and is suited to massively parallel implementation, enjoying arbitrary scalability and fault tolerance properties. Numerical examples conducted in 1D and 2D, including some for the KPP equation and Plasma Physics, are given.

Ángel Rodríguez-Rozas
Banco Santander, S.A.
E-mail: angel.rodriguez.rozas@gmail.com

Juan A. Acebrón
ISCTE – Instituto Universitário de Lisboa, Lisbon, Portugal
E-mail: juan.acebron@iscte-iul.pt

Renato Spigler
Università degli Studi Roma Tre, Rome, Italy
E-mail: spigler@mat.uniroma3.it

Keywords Probabilistic Domain Decomposition · Domain Decomposition Methods · Partial Differential Equations · Monte Carlo · Quasi-Monte Carlo · Elliptic Operators · Transport Equations · Vlasov-Poisson system · Nonlocal and Fractional Operators

1 Introduction

Since its introduction in 2005 for numerically solving boundary-value elliptic problems [1], the *PDD (Probabilistic Domain Decomposition) method* has been successfully extended by the authors of this article for solving a wide range of problems described through partial differential equations (PDEs) (see [8,2–7,14]).

In this article, we review the PDD method highlighting its main developments ([8,14]) and its latest investigations regarding fractional operators. The class of equations for which method has been applied include linear elliptic and parabolic equations, the KPP-equation, general semilinear parabolic equations, linear purely advection-dominated equations, the non-linear Vlasov-Poison system of equations governing plasma physics, and the Telegraph equation. Applications include all kind of diffusion and advection problems, finance (Black-Scholes and similar equations), plasma physics, and electrical transmission lines (see [8,2–7]).

For linear parabolic and elliptic problems defined in $\Omega \subseteq \mathbb{R}^d$, this method is based on the celebrated Feynman-Kac formula, that establishes a connection between the solution of a PDE and a suitable expectation over a corresponding stochastic process driven by Brownian motion, referred to as the stochastic solution. It exploits such solution to be approximated by the Monte Carlo method only at a few points along certain $\mathbb{R}^{d-1}$ interfaces, such that the original domain problem Ω is decomposed into as many independent subdomain problems as convenient. The result is a domain decomposition technique based on a probabilistic method that is suited for massively parallel computers, enjoying full scalability and fault tolerance.

For semilinear problems, the Feynman-Kac formula is generalized to solutions by means of branching stochastic processes in the real space. For linear and semi-linear hyperbolic problems, the extension of the Feynam-Kac formula is based on the *method of characteristics*, where the characteristic curves play a similar role in the corresponding solution as the stochastic process does for the parabolic problems. In the case of the Vlasov-Poison system, the stochastic solution is found in the Fourier domain after coupling the equations. Finally, for fractional PDEs, the method is extended to deal with space-fractional diffusion equations.

The structure of the article is as follows: First, we give a general introduction of the method when applied to linear parabolic problems; second, we describe an important extension of the method for solving general semi-linear parabolic problems; then, we present the extension of the method for transport problems and the Vlasov-Poisson system of equations; subsequently, we present the latest progress towards the extension of the PDD method for non-local fractional operators; finally, we conclude the paper with some remarks and future work.

2 Linear Parabolic Problems

2.1 Introduction

The purpose of this section is to introduce the PDD method (see [8]) to solve initial- as well as initial-boundary value problems for linear parabolic differential equations. The linear case is merely considered here to illustrate in the simplest way the essencial features of the PDD method. Rather unexpectedly, however, it turned out that even in such case important computational advantages were observed with respect to some existing more traditional parallel schemes. To assess the computational feasibility of our algorithm, we compare our results with those obtained using competitive (freely available) parallel numerical codes, which are widely used by the high-performance scientific community.

The plan of the section is as follows. First, some necessary mathematical generalities are provided. Then, the algorithm is described and different sources of parallelization are discussed. Later, numerical examples considering one-dimensional problems are given, where the efficiency of the PDD algorithm is illustrated. In a short final section, we summarize the salient points of the method.

2.2 Mathematical preliminaries

A variety of phenomena pertaining to Engineering, Physics, and other Sciences, are governed by diffusion equations. The relations between macroscopic diffusion and the mean statistical effect of the microscopic random (Brownian) motion of molecules goes back, among the others, to A. Einstein and M. Smoluchowski. A connection between "stochastic differential equations", that can be thought of ordinary differential equations driven by a certain kind of random noise (Langevin equations), and partial differential equations, was established.

Inspired by the work of R. Feynman on "path integrals" in quantum physics, M. Kac realized that a similar formulation could be applied to obtain a representation of the solution to the heat equation and to other diffusive (parabolic) linear partial differential equations. This lead to the so-called Feynman-Kac formula. Let $u(\mathbf{x},t)$ be a bounded function satisfying the Cauchy problem for the linear parabolic partial differential equation,

$$\frac{\partial u}{\partial t} = Lu - c(\mathbf{x},t)u, \quad u(\mathbf{x},0) = f(\mathbf{x}), \tag{1}$$

where $\mathbf{x} \in \mathbf{R}^n$, L is a linear elliptic operator, say $L := a_{ij}(\mathbf{x},t)\partial_i\partial_j + b_i(\mathbf{x},t)\partial_i$ (using the summation convention), with continuous bounded coefficients, $c(\mathbf{x},t) \geq 0$ and continuous bounded initial condition, $f(\mathbf{x})$. The probabilistic representation of the solution u to Eq. (1) is given through the Feynman-Kac formula

$$u(\mathbf{x},t) = E\left[f(\beta(t))\, e^{-\int_0^t c(\beta(s),t-s)ds}\right], \tag{2}$$

see [30,34], e.g., where $\beta(\cdot)$ is the n-dimensional stochastic process starting at $(\mathbf{x},0)$, associated to the operator L, and the expected values are taken with respect to the corresponding measure. When L is the n-dimensional Laplace operator, $\beta(\cdot)$ reduces to the standard n-dimensional Brownian motion, and the measure reduces to the Gaussian measure. In general, the stochastic process $\beta(\cdot)$ is the solution of a system of stochastic differential equations (SDEs) of the Itô type, related to the elliptic operator in (1),

$$d\beta = \mathbf{b}(\mathbf{x},t)\, dt + \sigma(\mathbf{x},t)\, d\mathbf{W}(t). \tag{3}$$

Here $\mathbf{W}(t)$ represents the n-dimensional standard Brownian motion (or Wiener process); see [34,10], e.g., for generalities, and [37] for related numerical treatments. As is known, the solution to (3) is a n-dimensional stochastic process, $\beta(t,\omega)$, where ω, usually not indicated explicitly in probability theory, denotes the "chance variable", which ranges on an underlying abstract probability space. The drift vector, $\mathbf{b}$, and the diffusion matrix, σ, in (3), are related to the coefficients of the elliptic operator in (1) by $\mathbf{b} = (b_1, \ldots, b_n)^T$, and $\sigma\sigma^T = \mathbf{a}$, with $\sigma = \{\sigma_{ij}\}_{i,j=1,\ldots,n}$, $\mathbf{a} = \{a_{ij}\}_{i,j=1,\ldots,n}$.

The representation in Eq. (58) can be generalized to deal with problems on *bounded* domains, say $\Omega \subset \mathbf{R}^n$, where given boundary data $u(\mathbf{x},t)|_{\mathbf{x}\in\partial\Omega} = g(\mathbf{x},t)$ of the Dirichlet type are prescribed. Thus, the following representation holds, for the solution of the problem, being now continuous and bounded on $\overline{\Omega} \times [0,T]$,

$$\begin{aligned} u(\mathbf{x},t) = E\left[f(\beta(t))\, e^{-\int_0^t c(\beta(s),t-s)ds}\, \mathbf{1}_{[\tau_{\partial\Omega}>t]}\right] \\ +E\left[g(\beta(\tau_{\partial\Omega}),\tau_{\partial\Omega})\, e^{-\int_0^{\tau_{\partial\Omega}} c(\beta(s),t-s)ds}\, \mathbf{1}_{[\tau_{\partial\Omega}<t]}\right]. \end{aligned} \tag{4}$$

Here $\tau_{\partial\Omega}$ denotes the first exit (or hitting) time of the path $\beta(\cdot)$, started at $(\mathbf{x},t)$, when $\partial\Omega$ is crossed, and $\mathbf{1}_{[\tau>t]}$ is the characteristic function, which takes the value 1 or 0, depending whether $\tau_{\partial\Omega}$ is or is not greater than t.

The solution to the linear inhomogeneous problem

$$\frac{\partial u}{\partial t} = Lu - c(\mathbf{x},t)u + F(\mathbf{x},t), \tag{5}$$

where $F(\mathbf{x},t)$ is a bounded continuous function of $\mathbf{x}$ and t, can also be represented probabilistically, using the related Green function, which, in turn, can be represented as above, being the solution to the associated homogeneous problem (e.g., see [3,4]).

2.3 The numerical method

The algorithm consists of three steps. To illustrate how it works, in Fig. 1 a sketchy diagram is plotted where such steps are shown for a two-dimensional problem. The first step consist in computing the solution at a few points by a probabilistic Monte Carlo-type method, based on averaging over certain random paths. This is done on some chosen *interfaces*, located inside the space-time domain $D := \Omega \times [0,T]$, where $\Omega \subset \mathbf{R}^n$. In the following, such interfaces are obtained, for simplicity, partitioning the domain into subdomains as $D_i := [x_{i-1}, x_i] \times \Omega_0 \times [0,T]$, being $\Omega_0 \subset \mathbf{R}^{n-1}$. For instance, in $\mathbf{R}^2$ this corresponds to divide the domain in slices where the interfaces are parallel to y-axis. For complex domains, a proper partitioning algorithm may be employed to define such interfaces. Once the solution has been computed, the second step is interpolating on such points, considered as interpolation nodes, thus obtaining continuous approximations of interfacial values of the solution. The third step, finally, consists in computing the solution inside each subdomain, which task can be assigned to separate processors. This can be realized resorting to local solvers, which may use classical numerical methods, such as finite differences or finite elements methods.

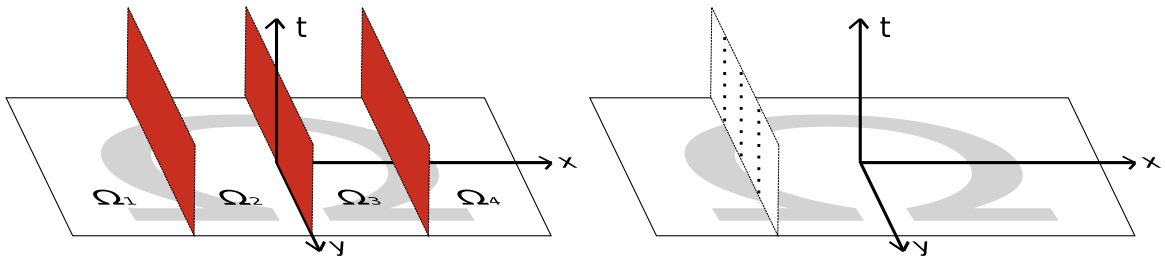

Fig. 1 A sketchy diagram illustrating the main steps of the algorithm in 2D: The figure on the left shows how the domain decomposition is done in practice. The figure on the right shows the points where the solution is computed probabilistically; these are used afterwards as nodal points for interpolation.

2.3.1 Probabilistic part

The purpose of this step is to compute the sought solution at a few single points, inside the space-time domain. Computing the solution at a high number of points so as to cover a full computational domain is also possible but is exceedingly expensive, even though this approach could be pursued when the number of the available processors is extremely high. This can be done assigning the task of computing the solution at a set of points to different processors. The Monte Carlo method is, in fact, capable of fully exploiting massively parallel architectures. Moreover, it is scalable to an arbitrary number of processors as well as naturally fault tolerant.

When the parabolic equations are linear, a given number of random paths have to be generated, which obey the SDE in (3), tracking them until they either touch the boundary for the first time or else reach a prescribed final time, t. The former case occurs in initial-boundary value problems (e.g., with Dirichlet boundary conditions), while the latter case occurs in both a purely initial value problem, and a initial-boundary value problems. The solution to the equation at a given point, $(\mathbf{x}, t)$, can then be obtained by means of the Feymann-Kac formula in (4) or (58). In practice, the expected value is replaced by an arithmetic mean, since we must deal with a finite sample size, N. An alternative strategy to evaluate the solution was proposed in [43] for initial-boundary problems, which requires generating a random exponential time, S, obeying the probability density $P(S) = c \exp(-cS)$ for every random path. Then, depending on whether $S < t$ or not, the given path $\beta(t)$, contributes or not to the solution. Therefore, the solution is computed as

$$u(\mathbf{x}, t) = E[f(\beta(t))]. \tag{6}$$

In practice, the expected value above must be replaced necessarily by a finite sum, and moreover the stochastic paths are actually simulated resorting to suitable numerical schemes. Thus, approximately,

$$u(\mathbf{x}, t) = \frac{1}{N} \sum_{j=1}^{N} f(\beta_j^*(t)), \tag{7}$$

where N is the sample size, and β^* is the stochastic path with discretized time. Such a discretization procedure unavoidably introduces two sources of numerical error. The first one is the pure Monte Carlo

statistical error, which it is known to be of order $O(1/\sqrt{N})$ when N goes to infinity. The second error is due to the fact that the ideal stochastic path, $\beta_j(\cdot)$, has to be approximated, discretizing time, by some numerical scheme yielding the paths $\beta_j^*(\cdot)$. The truncation error made in solving numerically the stochastic differential equation (3), obviously depends on the specific scheme chosen, see [37], e.g. Among these are the Euler scheme, which was used here to simulate numerically Eq. (3). Such scheme is well known to have a truncation error of order $O(\Delta t^\alpha)$, where $\alpha = 1/2$ or $\alpha = 1$, understood in the "strong" or "weak" sense, respectively [37].

For the case of a boundary-value problems, a new source of numerical error should be taken into account. In fact, for the purpose of illustration let us consider the Dirichlet problem for the one-dimensional heat equation, in presence of a constant sink term, $c > 0$,

$$\begin{aligned} &\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} - cu, \quad a < x < b,\, t > 0 \\ &u(a,t) = 0,\, u(b,t) = 0 \\ &u(x,0) = f(x). \end{aligned} \tag{8}$$

The solution can be computed as

$$u(\mathbf{x},t) = \frac{1}{N}\sum_{j=1}^{N} f(\beta_j^*(t))\mathbf{1}_{[S_j > \tau_\Omega]}. \tag{9}$$

In Fig. 2, we sketched the three possible scenarios the random paths $\beta_j^*(t)$ can undergo. Note that for the random paths of the type labelled with (3) in Fig. 2, it is required to evaluate precisely the first exit time out of the boundary. Such a task is however by far nontrivial, since $\tau_{\partial\Omega}$, in general, will be estimated numerically, and hence will be affected by numerical errors. Indeed, numerical experiments show that the error in estimating it may dominate over the other sources of numerical errors, and is therefore of paramount importance to assess accurately such a quantity.

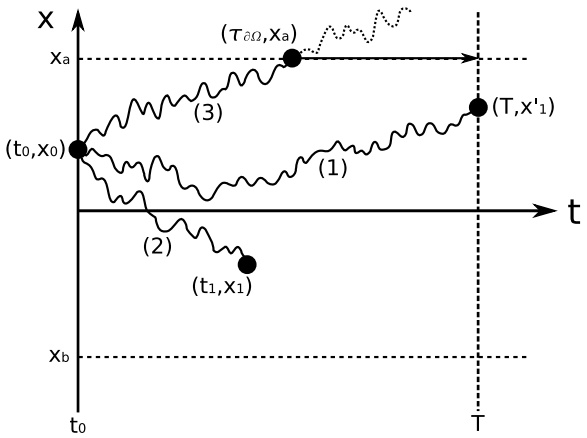

Fig. 2 The three possible scenarios for a random path, for the one-dimensional problem in (8). In (1), the random time is greater than the final time, T; in (2), the random time turns out to be smaller than T; in (3), the first exit time is smaller than both the random and the final time.

In practice, the probability that a given approximate path exits the boundary between two consecutive time steps, is nonzero, and then it is possible that the true exit time might be systematically overestimated. This circumstance has been pointed out in several occasions, see, e.g., [48,12,42].

In [31], it was estimated that, due to the presence of boundaries, the weak convergence of the naive Euler scheme in evaluating (9) is reduced to $O(\Delta t^{1/2})$, being Δt the time step used in solving numerically the SDE (3). To reduce such an error (ideally, to recover the convergence order achieved in the absence of boundaries), it becomes crucial to evaluate accurately the first exit time, adopting suitable numerical strategies.

Among the various possibilities considered in the literature, we chose to implement that proposed in [39] for one-dimensional problems, which is based on a theoretical approximation of the exit probability. To solve two-dimensional problems on the square, the value of the exit probability on Ω has been taken as the maximum among the four hitting probabilities that a trajectory first exits the four possible boundary-sides. This consists on an approximation of the true two-dimensional exit probability, but it suffices in order to achieve a numerical error now well below the statistical error.

In fact, for general $n-$dimensional diffusion processes, there exist other interesting alternatives to approximate the first exit time. In [18,42], Buchmann and Petersen presented an algorithm to simulate stopping diffusion processes to obtain again weak order one using the Euler scheme. More recently, Gobet and Menozzi proposed in [32] a new simpler and computationally more efficient approach, of weak order $o(\Delta t^{1/2})$. The idea consists of stopping the simulation of discrete paths generated by means of the Euler scheme, when such paths exit through a conveniently modified domain, shrinking (or shifting) the boundary of Ω in the direction of the inward normal. The amplitude of such shrinking (or shifting) depends on, among other values, the diffusion coefficient of the process and the square root of the time step used in the numerical scheme. Given the general applicability and simplicity of this approach, it is specially convenient when dealing with more complex geometries.

As mentioned before, for the linear case, in order to evaluate the probabilistic representation we resort to numerical simulations of the Monte Carlo type, considering a finite size sample, N. In practice, we replace the expected value with an arithmetic mean, which is known to provide the best unbiased estimator [33]. The error made in doing so is statistical in nature and of the order of $N^{-1/2}$.

Finally, a carefully analysis of the computational cost associated to this part of the algorithm is provided in the next section, when dealing with general semilinear problems.

2.3.2 Interpolation in space-time

Let assume that we have already computed the values of the sought solution at some points on the interfaces $x = x_i$, by the previous Monte Carlo approach. These are the points $(x_i, \mathbf{y}_j, t_k)$, where $\mathbf{y}_j \in \Omega_0 \subset \mathbf{R}^{n-1}$, for every fixed i, and very few j's and k's. A number of numerical schemes can be adopted to interpolate in the $n-1$ dimensional space Ω_0. The simplest method of obtaining multivariate interpolation is to consider a univariate method and derive from it a multivariate method by tensor product. In practice, given $n-1$ set of points, the tensor product interpolation finds the corresponding interpolation coefficients solving repeatedly univariate interpolation problems as described in [25]. For the one-dimensional examples given in this section we used the Chebyshev polynomials, while for two-

dimensional examples, a tensor product interpolation based on cubic splines was adopted [9]. Here the nodal points are uniformly distributed on Ω_0, and a not-a-knot condition has been imposed, which means imposing continuity of the third derivative at the boundary. When the number of nodal points, n, is the same along each dimension, interpolating at a single point (y_j, t_k) requires $n+1$ spline calculations to obtain the spline coefficients, and then evaluating the spline value at $n+1$ points. The computational cost for calculating the spline coefficients is known to be of order $O(n)$, while for evaluating the spline value it is $O(\log n)$. The interpolation error when the interpolating function is sufficiently smooth (C^8 at least) is of order of $O(h^4 + l^4)$ [46], where h and l are the widths of the interpolating grid in the y and t axes, respectively.

2.3.3 Local solver

Once that continuous interfacial approximations of the solution have been obtained upon interpolation on the previously computed nodes (by Monte Carlo), we can solve the original problem *on each* subdomain, D_i, *independently* of each other, since a full decoupling has been realized. Hence, the numerical treatment on each subdomain can be accomplished by a local solver, which can also be different from all the others. In the numerical examples below, we used a solver based on the LU factorization.

2.3.4 Sources of parallelization

We stress that in practice there are three sources of parallelization, namely (1) the Monte Carlo generation of internal node functional values (even each single sample can be ran on independent processors), (2) the interpolation part (the interpolation on each interface can be accomplished independently), and (3) the domain decomposition solution (that can be assigned to independent local solver). Moreover, each of such three stages enjoyed a natural fault tolerant property: (1) if a number of processors fail in the Monte Carlo simulations, it will be enough to ignore the result from them using the remaining samples. Hence, at a price of a small additional errors, the algorithm will still provide meaningful results. (2) Failure of processors computing interpolated values of the solution on some interfaces may only imply to neglect, temporarily, the solution on those subdomains having such interfaces as part of their boundary. (3) Failure of processors responsible for the numerical solution on some sub-domains can also be temporarily neglected, while the solution computed by the local solvers on the remaining sub-domains will be computed correctly. Note that on the interfaces and on the sub-domains where the processors failed, the solution can be computed restarting again the algorithm.

For numerical examples, see [3].

2.4 Summary

We have described the PDD algorithm for solving linear parabolic partial differential equations in any space dimension, where a domain decomposition approach is used to split the given space-time domain into as many subdomains as the number of available processors. The solution at the interfaces that separates the subdomains are computed after interpolating on the nodal points for which the solution is

previously obtained probabilistically via Monte Carlo. Such probabilistic computation consists of evaluating averages on suitably generated random paths, without the need of deploying a computational mesh. Moreover, every available processor is devoted to compute the solution at one of such interfaces, without introducing communication nor synchronization among other processors. This fact is of paramount importance, because once the solution on the interfaces has been independently computed, fully in parallel, the remaining task of evaluating the solution inside each subdomain turns out to be totally independent as well, resulting in a complete, communication- and synchronization-free, fully scalable algorithm.

3 Semilinear Parabolic Problems

3.1 Introduction

Probabilistic representations do exist for some elementary *semilinear* parabolic equations. Indeed, in [38] H.P. McKean derived the representation formula

$$u(x,t) = E[\prod_{i=1}^{k_t(\omega)} f(x_i(\omega,t))] \tag{10}$$

for the KPP equation

$$u_t = u_{xx} + u(u-1), \quad x \in \mathbf{R},\ t > 0, \tag{11}$$

subject to the initial value $u(x,0) = f(x)$ (see also [30,40,44]), where $k_t(\omega)$ represents a time-dependent random variable that accounts for the number of branches of the underlying branching diffusion process. Later, we have found a similar representation (see also [3,4]) for the solution of a more general semilinear parabolic problem, given by

$$\frac{\partial u}{\partial t} = Lu - cu + \sum_{j=2}^{m} \alpha_j u^j, \tag{12}$$

where L is a general linear elliptic operator, say $L := a_{ij}(\mathbf{x},t)\partial_i\partial_j + b_i(\mathbf{x},t)\partial_i$, with continuous bounded coefficients, $m \geq 2$ is an integer, $\alpha_j \geq 0$, $\sum_{j=2}^{m} \alpha_j = 1$, and c is a positive constant. Such a representation is based on generating *branching* diffusion processes, associated with the elliptic operator in Eq. (12), and governed by an exponential random time, S, with probability density $p(S) = c\,\exp(-cS)$.

In this section, we explain how the probabilistic representation was extended to deal with a wider class of semilinear parabolic problems (see also [5]), whose general form is given by

$$\begin{aligned} \frac{\partial u}{\partial t} = Lu + f(u,x,t),\ x \in \mathbf{R}^n,\ t > 0 \\ u(x,0) = g(x), \end{aligned} \tag{13}$$

where

$$f(u,x,t) = \sum_{j=2}^{m} c_j(x,t) u^j,$$

where the $c_j(x,t)$ are continuous given functions. It is worth to observe that this generalizes further the previous representation in (12), explained in [3,4], since it accounts for the following aspects: A constant potential term such as $-cu$ is not required anymore; the coefficients multiplying the nonlinear terms, $c_j(x,t)$, can be chosen arbitrarily, hence overcoming the constraint imposed in the previous representation

consisting in $\sum_{j=2}^{m} c_j(x,t) = 1$, and finally the initial data $g(x)$ may now be chosen negative, or greater than 1.

Here it is the outline of the section. First, the generalized probabilistic representation is presented. Then, a qualitative study of the numerical errors is accomplished analyzing a few relevant test examples. Later, some numerical examples are shown, where the high efficiency of the PDD method comparing with classical methods is illustrated. Finally, we summarize the more relevant findings to close the applicability of the PDD method to parabolic problems.

3.2 A generalized probabilistic representation for semilinear parabolic problems

In order to generalize the class of parabolic problems amenable to a probabilistic representation in terms of branching diffusion processes, it becomes more convenient to rewrite Eq.(13) in an integral form. This can be done readily resorting to the Duhamel principle [26] for inhomogeneous initial-value parabolic problems, and yields

$$u(x,t) = \int_{\mathbf{R}^n} dy\, g(y)\, p(x,t,y,0) + \int_0^t \int_{\mathbf{R}^n} ds\, dy\, f(u(y,s),y,s)\, p(x,t,y,s), \tag{14}$$

where $p(x,t,y,\tau)$ is the associated Green's function, satisfying the equation

$$\begin{aligned} \frac{\partial p}{\partial t} &= Lp, \quad x \in \mathbf{R}^n, \quad t > \tau \\ p(x,\tau,y,\tau) &= \delta(x-y). \end{aligned} \tag{15}$$

The main difference with the previous representation obtained in (12) rests on the absence of the constant potential term $-cu(x,t)$. Such a term was crucial, since it allowed to obtain a probabilistic representation based on generating *branching* diffusion processes governed by an exponential random time, S, with density probability $p(S) = c\exp(-cS)$ (see previous sections, and [3,4]). In the following we describe the main strategy derived in [5] (referred to as *Strategy B*), capable to overcome such a constraint generalizing further the aforementioned representation.

3.2.1 Probabilistic representation

A way to obtaining a probabilistic representation for the problem in Eq. (13) consists in sampling both terms of the integral equation (14), by introducing a two-point discrete random variable ξ taking the values 0, and 1 with probability $P(0) = q$, $P(1) = 1-q$. Therefore, the integral equation (14) can be rewritten as follows,

$$\begin{aligned} u(x,t) = {} & q \int_\Omega dy\, \tilde{g}(y)\, p(x,t,y,0) \\ & + (1-q) \int_0^t \int_\Omega ds\, dy \sum_{j=2}^{m} \tilde{c}_j(y,t-s) u^j(y,t-s)\, p(x,s,y,0), \end{aligned} \tag{16}$$

where $\tilde{g}(x) = g(x)/q$, and $\tilde{c}_j(x,t) = c_j(x,t)/(1-q)$. The probabilistic representation can be readily found and has the form

$$\begin{aligned} u(x,t) = {} & E\left[\tilde{g}(\beta(t))\delta(\xi)\right] \\ & + E\left[\eta(t)\tilde{c}'_\alpha(\beta(tS),t(1-S))\, u^\alpha(\beta(tS),t(1-S))\delta(\xi-1)\right], \end{aligned} \tag{17}$$

where the expectation is given by the measure generated by the following random variables: the diffusion processes $\beta_i(\cdot)$, the time S, a random time uniformly distributed; and α, a discrete random variable taking on the values between 2 and m with equal probability $p = 1/(m-1)$, $\tilde{c}'_\alpha = (m-1)\tilde{c}_\alpha$, and $\eta(t) = t$. The equation above is not in a closed-form but it can be recursively expanded, to yield

$$\begin{aligned}
&u(x,t) = E\left[\tilde{g}(x_0(t))\,\delta(\xi_0)\right] \\
&+E\left[\eta(t)\tilde{c}'_{\alpha_1}(y_1(tS_0), t(1-S_0))\prod_{i=1}^{\alpha_1}\tilde{g}(x_i(t(1-S_0))\,\delta(\xi_i)\delta(\xi_0-1)\right] \\
&+E\left[\eta(t)\eta(t(1-S_0))\tilde{c}'_{\alpha_1}(y_1(tS_0), t(1-S_0))\right. \\
&\times\tilde{c}'_{\alpha_2}(y_2(t(1-S_0)S_1), t(1-S_0)(1-S_1))\prod_{i=2}^{\alpha_1}g(x_i(t(1-S_0))))\,\delta(\xi_i) \\
&\left.\times\prod_{j=\alpha_1+1}^{\alpha_1+\alpha_2+1}g(x_j(t(1-S_0)(1-S_1)))\delta(\xi_j)\delta(\xi_1-1)\delta(\xi_0-1)\right]+\cdots,
\end{aligned} \tag{18}$$

where x_i and y_j corresponds to the position of the i-th branch at the final time t and the position of the j-th splitting event, respectively.

While in Eq. (17) the expectation is taken with respect to the measures generated by β, ξ and α (which would be enough if consisted of a closed-form), the expectation in Eq. (18) is rather taken from the underlying measure corresponding to β (i.e., by x_i and y_i), again, and the infinite sequence of random variables $\xi_0, \xi_1, ...$, and $\alpha_1, \alpha_2, ...$, that the determine the final configuration of every random tree generated.

Therefore, as we illustrate next, this solution can be obtained as the expectation over suitable random trees of a given multiplicative functional of the initial condition, being given now as follows:

$$u(x,t) = E\left[\prod_{i=1}^{Ne(\omega)}\eta(t\bar{S}_i(\omega))c_{\alpha_i(\omega)}(y_i(\omega), t\bar{S}_i(\omega))\prod_{l=1}^{k(\omega)}g(x_l)\right]. \tag{19}$$

Here $k(\omega)$ and $Ne(\omega)$ are random variables that represents the number of branches at final time t, and the number of splitting events obtained when generating a particular random tree, respectively. Note that $P\{Ne(w) = i\} = (1/2)^i$. In [4], it is shown that $P(k,m)$, the probability of finding a random tree with k branches, being m the number of children, is given by

$$P(k,m) = q^k(1-q)^{Ne}\frac{1}{m\,Ne+1}\begin{pmatrix} m\,Ne+1 \\ Ne \end{pmatrix}. \tag{20}$$

By $\bar{S}$ we denote the corresponding global random time obtained by summing conveniently the random times S_i according to the specific structure of the generated random tree. It is worth to observe that such trees are used as a tool to construct the structure representing a given partial contribution to the solution, allowing afterward to follow easily how the arguments of the functions are exchanged when solving recursively Eq.(17).

To illustrate how this representation can be implemented in practice for solving a particular problem, let consider the following equation,

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + u^2, \quad x \in \mathbf{R}, t > 0 \tag{21}$$

$$u(x,0) = f(x). \tag{22}$$

From Eq. (17), the probabilistic representation is given by

$$u(x,t) = E\left[\tilde{g}(\beta(t))\delta(\xi)\right] + E\left[\eta(t)\, u^2(\beta(tS), t(1-S))\delta(\xi-1)\right], \tag{23}$$

or in a more compact format, using Eq.(19) for the expectation value over random trees of a given multiplicative functional in Eq.(23), by

$$u(x,t) = E\left[\prod_{i=1}^{Ne(\omega)} \eta(t\bar{S}_i(\omega)) \prod_{l=1}^{k(\omega)} g(x_l)\right]. \tag{24}$$

Every random tree is built generating a sequence of interconnected binary random variables, ξ_i, branching off from the previous one as follows: Let ξ_1 the random variable associated to the root of the tree. Only when ξ_1 takes value 1 with probability $P(1) = 1 - q$, two new random variables denoted by $\xi_{2,3}$ (child nodes of the root), are created. These new variables proceed further creating other nodes governed by the same rule, until no random number ξ_i takes anymore the value 1. At this point the procedure is concluded, giving rise to a random tree characterized by k branches or leaves, and Ne splitting events.

The nodes of the tree are labeled in binary format according to their ancestors as follows: A given node with label $[a_0a_1a_2...a_N]$, where $a_i = 0, 1$, is connected to the set of nodes $\{[a_0], [a_0a_1], [a_0a_1a_2], \ldots, [a_0a_1a_2\cdots a_{N-1}]\}$. The global time random variable $\bar{S}$ associated to a given tree with k branches is given by

$$\bar{S} = \prod_{i=1}^{2^{k-1}-1} S_j^{\gamma_j}, \quad \gamma_l = \sum_{j=l+1}^{2^{k-1}-1} \nu_j \langle j|l\rangle, \; l = 1, \ldots, 2^{k-1}-1 \tag{25}$$

where ν_l is 0, or 1 depending on whether the tree contains or not the node l. The function $\langle\cdot|\cdot\rangle$ is defined as follows,

$$\langle j|l\rangle := \begin{cases} 1 \text{ if } T_j^{[l]} = l \\ 0 \text{ otherwise.} \end{cases}, \tag{26}$$

where both, j and l are numbers written in binary format, and $T_j^{[l]}$ is an operator that truncates the number j to their most significant $[l]$ digits, where $[l]$ is the number of digits of l. By example, let $j = [a_0a_1a_2...a_N]$, then $T_j^{[l]} = [a_0a_1...a_{[l]-1}]$.

Figure 3 shows the different random trees obtained with $k = 4$, and $Ne = 3$, and their corresponding labels according to the rule defined above.

Finally, note that in practice, a series arises when evaluating the solution of Eq. (18) by Monte Carlo, when attempting to summing up the partial contribution of trees of different branches. This series is infinite but in practice, we always end up with a finite series because the probability of obtaining trees with an increasing number of branches is increasingly smaller and therefore, their contribution is negligible up to defining a tolerance for the numerical error. In the case of asymptotic divergent series that may appear, we resort to approximation techniques such as the Padé approach [13,11], to approximate conveniently the sum of the aymptotic series. In this method, called Padé approximation, the idea is to replace the asymptotic divergent power series by a sequence of rational functions converging towards the solution u, as follows: Each rational function, P_M^N, given as a ratio of two polynomials of degree N and M, is constructed such that the first $N + M + 1$ terms of its series expansion match the first $M + N + 1$ terms of the divergent power series. The hope is that $P_M^N \to u$ as $N, M \to \infty$.

For the computational complexity and numerical results, we refer to the reader to [4].

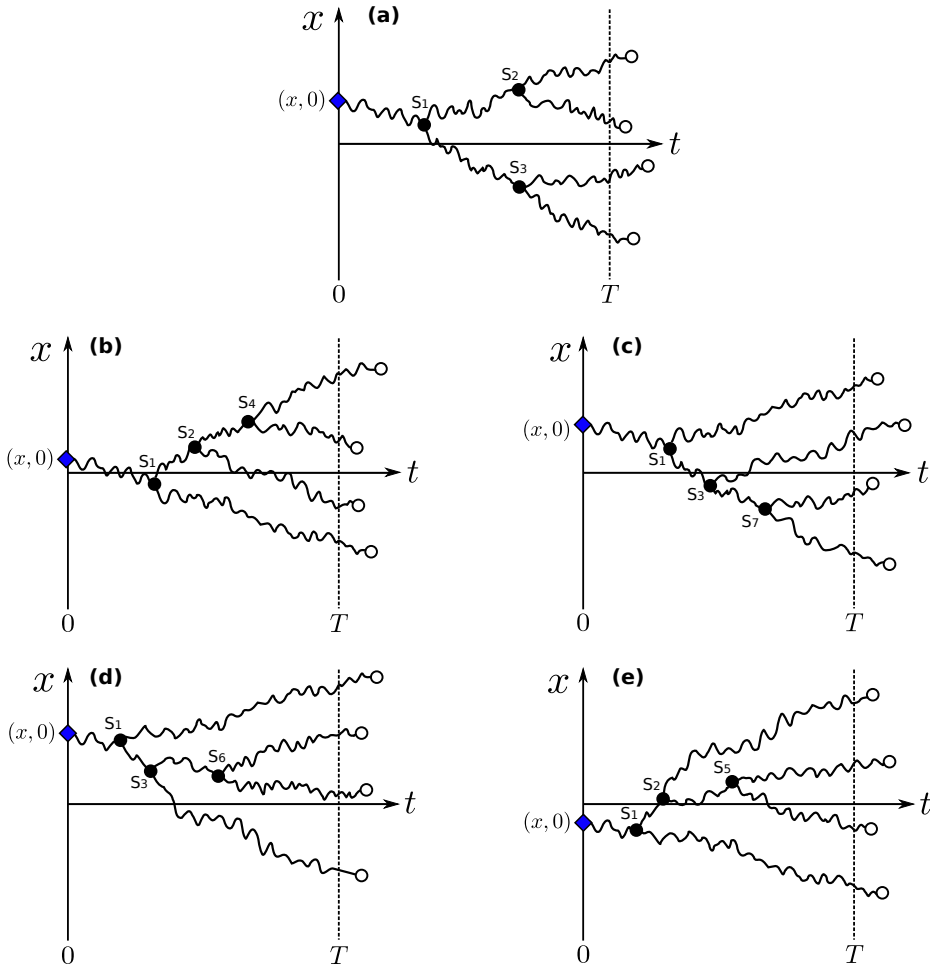

Fig. 3 Configuration diagram for the case of 4 branches and 3 splitting events. Here S_i is a random time uniformly distributed between the previous generated time, and the final time, T. The corresponding labels i of the random time S_i are defined according to the rule explained in the text.

3.3 Summary

The class of semilinear parabolic problems amenable to a probabilistic solution was expanded by introducing suitable generalized random trees. The probabilistic computation consists of evaluating averages on the generated random tree, which plays a role similar to that of a random path in linear problems. The new representation allows treatment of semilinear problems without a potential term, with arbitrary coefficients multiplying the nonlinear term, and arbitrary initial data, including negative definite and greater than one. The implementation requires computing the solution through a series where the coefficients represent the partial contribution to the solution coming from generated random trees with any number of branches. Summation of divergent asymptotic series expansions cannot be summed simply by a sequence of partial sums. Nevertheless, numerical experiments show that (see [4], in many cases, the asymptotic series can be approximated quite accurately by the Padé approximant ([13,11]).

The new probabilistic representation has been used successfully as a crucial element for implementing a suitable probabilistic domain decomposition method. In fact, at the time when these simulations were

conducted, numerical examples showed the excellent scalability properties of the PDD algorithm in large-scale simulations, where up to 512 processors were used on a high performance supercomputer.

4 The Vlasov-Poisson system for Plasma Physics

4.1 Introduction

To illustrate the potentiality behind these probabilistic techniques for solving numerically transport equations, in this section the method is particularized to the Vlasov-Poisson system of equations. We have considered merely periodic boundary conditions in space. This has been done for simplifying as much as possible the probabilistic representation of the solution, and thus to help the reader to understand easily such a representation. It is theoretically well-known [34, 40], that dealing with general boundary conditions requires estimating various stochastic quantities, which in turn introduces new sources of numerical errors, that we tried to avoid at this stage. The ultimate goal of this section is to show the feasibility of this method as a complementary method capable to speed up the Vlasov-Poisson simulations in a parallel environment, and this was done simplifying the nature of the boundary data as much as possible. The generalization of the method to situations where more complicated geometries and boundary conditions are imposed, is left for a future work.

Being the boundary conditions periodic in space, it becomes natural to solve numerically the problem in Fourier space for the spatial coordinates. Moreover, it turns out that for dealing accurately with the filamentation phenomenon, it is convenient to analyze also the problem in Fourier space for velocities [28]. That is why in practice the numerical method to be presented in this section is fully analyzed in Fourier space. Apart from such a mathematical reason, in some practical experimental situations one could be interested not to know the distribution function, but rather the spectrum energy or any other related quantities, being therefore natural to investigate the problem in Fourier space. Furthermore, while the probabilistic representation introduced in this section was obtained in Fourier space, there already exists representations in configuration space [50], which may potentially be used to generalize further what has been done in this section.

This section is organized as follows. Sec. 4.2 concerns the probabilistic representation for the Vlasov-Poisson system of equations. Here such a representation is derived in the Fourier space for arbitrary dimensions, and the validation of the representation is done analyzing the classical linear Landau damping. In Sec. 4.3, it is explained how the probabilistic representation can be used in practice, and which are the associated numerical errors. First the algorithm is described, and analyzed the numerical error, then the computational cost is estimated, and finally several numerical tests consisting in typical problems considered often in the literature are given, focusing in both, the accuracy and performance of the method. To conclude we summarize the main results and discuss potential directions for future research.

4.2 Probabilistic representation for the Vlasov-Poisson system

The Vlasov-Poisson system describes the temporal evolution of charged particles subject to the self-generated electric field created by the charged particles inside the plasma. It is actually a system of

equations, consisting of a kinetic equation, the Vlasov equation which describes the transport of charged particles, along with the classical Poisson equation for electrostatic potential. The solution of the equation is the distribution function of particles in the phase space, where the independent variables are space, x, velocity, v, and time t. Consider the two-species Vlasov-Poisson equation in d dimensions conveniently adimensionalized,

$$\begin{aligned}\frac{\partial f^{(1)}}{\partial t}+\bar{v}\cdot\nabla_{\bar{x}}f^{(1)}-\nabla\phi\cdot\nabla_{\bar{v}}f^{(1)}&=0,\\ \frac{\partial f^{(2)}}{\partial t}+\bar{v}\cdot\nabla_{\bar{x}}f^{(2)}+\frac{1}{m_2}\nabla\phi\cdot\nabla_{\bar{v}}f^{(2)}&=0,\\ \triangle_{\bar{x}}\phi=-\left[\int f^{(1)}d\bar{v}-\int f^{(2)}d\bar{v}\right],&\end{aligned}\tag{27}$$

along with a 2π-periodic boundary condition for the space variables, $f^{(i)}(\bar{x},\bar{v},t)=f^{(i)}(\bar{x}+2\pi,\bar{v},t)$, decay to zero as $|\bar{v}|\to\infty$ with sufficiently high rate for velocity variables, and suitable initial conditions for both, $f^{(i)}$, and the space average over a period of the electric field $\bar{E}=-\nabla\phi$. In [36] it has been proved that in order the Vlasov-Poisson equations provide a complete description of the plasma, such a quantity should be kept fixed to zero for all time. Finally, being $f^{(i)}$ a density probability, it holds that $\int_0^{2\pi}\int_{-\infty}^{\infty}d\bar{x}\,d\bar{v}\,f^{(i)}=1$.

Since the prescribed boundary condition for space variables are periodic, it is more natural to analyze the problem in Fourier space. Moreover, it turns out to be more convenient to transform as well to the Fourier space for velocities, in order to mitigate the well known filamentation effect in velocity space observed in the solution for sufficiently long times [28]. Because of the periodicity in the space variables, the transformation in space is discrete, while for velocities is continuous. Then, transforming Eq. (27), yields

$$\begin{aligned}\frac{\partial F^{(1)}_{\bar{k}}}{\partial t}-\bar{k}\cdot\nabla_{\bar{\xi}}F^{(1)}_{\bar{k}}-\bar{k}\cdot\bar{\xi}\,\hat{\phi}_{\bar{k}}*F^{(1)}_{\bar{k}}&=0,\\ \frac{\partial F^{(2)}_{\bar{k}}}{\partial t}-\bar{k}\cdot\nabla_{\bar{\xi}}F^{(2)}_{\bar{k}}+\frac{1}{m_2}\bar{k}\cdot\bar{\xi}\hat{\phi}_{\bar{k}}*F^{(2)}_{\bar{k}}&=0,\\ -|\bar{k}|^2\hat{\phi}=-\left[F^{(1)}_{\bar{k}}(0,t)-F^{(2)}_{\bar{k}}(0,t)\right],\quad|\bar{k}|\neq 0,&\end{aligned}\tag{28}$$

where

$$F^{(i)}_{\bar{k}}(\bar{\xi},t)=\int_{\mathbb{R}^d}d\bar{v}\int_0^{2\pi}d\bar{x}\,e^{-i\bar{\xi}\cdot\bar{v}}e^{-i\bar{k}\cdot\bar{x}}f^{(i)}(\bar{x},\bar{v},t),\quad i=1,2,\tag{29}$$

$$\hat{\phi}_{\bar{k}}(t)=\int_0^{2\pi}d\bar{x}\,e^{-i\bar{k}\cdot\bar{x}}\phi(\bar{x},t).\tag{30}$$

Here $\bar{k}$, and ξ corresponds to the conjugate variables of $\bar{x}$, and $\bar{v}$, respectively, being $\bar{k}$ a discrete variable, while ξ is continuous, and $*$ denotes the convolution operator for $\bar{k}$. Note that passing to the Fourier space becomes crucial to be able to reduce the system of equations into a single one. Moreover, this is mandatory for the purpose of finding a probabilistic representation for the solution of Eqs. (27), applying directly the strategy described in [6] for the semilinear transport equation. The first step towards the probabilistic representation requires rewriting the system of equations (28) in integral form, and is given

by

$$F_{\bar{k}}^{(i)}(\bar{\xi},t) = F_{\bar{k}}^{(i)}(\bar{\xi}+t\bar{k},0) + \rho_i \int_0^t ds \sum_{\substack{\bar{k}'=-\infty \\ \bar{k}'\neq 0}}^{\infty} \frac{\bar{k}'\cdot(\bar{\xi}+s\bar{k})}{|\bar{k}'|^2}$$
$$\times[F_{\bar{k}'}^{(1)}(0,t-s) - F_{\bar{k}'}^{(2)}(0,t-s)]F_{\bar{k}-\bar{k}'}^{(i)}(\bar{\xi}+s\bar{k},t-s), \tag{31}$$

where $\rho_1 = 1$, and $\rho_2 = -1/m_2$. Both, the static and dynamic probabilistic representation can be derived similarly to the case of the semilinear transport equation. Here we describe the dynamic representation, since the static one is straightforward. Eq. (31) can be written probabilistically as follows,

$$\begin{aligned} F_{\bar{k}}^{(i)}(\bar{\xi},t) &= E\left[\tilde{F}_{\bar{k}}^{(i)}(\bar{\xi}+t\bar{k},0)\delta(\zeta)\right] \\ &+E\left[\eta(t)\, g^{(i)}(\bar{k},\bar{k}',\bar{\xi},S)\, F_{\bar{k}'}^{(1)}(0,t-S)\, F_{\bar{k}-\bar{k}'}^{(i)}(\bar{\xi}+S\bar{k},t-S)\delta(\rho-1)\delta(\zeta-1)\right] \\ &+E\left[\eta(t) g^{(i)}(\bar{k},\bar{k}',\bar{\xi},S)\, F_{\bar{k}'}^{(2)}(0,t-S)\, F_{\bar{k}-\bar{k}'}^{(i)}(\bar{\xi}+S\bar{k},t-S)\delta(\rho-2)\delta(\zeta-1)\right], \end{aligned} \tag{32}$$

where

$$\begin{aligned} g^{(1)}(\bar{k},\bar{k}',\bar{\xi},S) &= 2\rho_1 \frac{\bar{k}'\cdot(\bar{\xi}+S\bar{k})}{(1-q)p(\bar{k})|\bar{k}'|^2}, \\ g^{(2)}(\bar{k},\bar{k}',\bar{\xi},S) &= -2\rho_1 \frac{\bar{k}'\cdot(\bar{\xi}+S\bar{k})}{(1-q)p(\bar{k})|\bar{k}'|^2}, \end{aligned} \tag{33}$$

and $\tilde{F}_{\bar{k}}^{(i)} = F_{\bar{k}}^{(i)}/q$. Here four random variables have been introduced, those are: ρ is a two-point, $\rho = 1, 2$, discrete random variable equally distributed with probability $1/2$; S a continuous random variable uniformly distributed between 0 and t, and therefore $\eta(t) = t$; $\bar{k}'$ is a discrete random variable with density probability $p(\bar{k}')$, and finally ζ, which takes the values 0, and 1, with probability $P(0) = q$, $P(1) = 1-q$, respectively. E denotes the expected value taken with respect to the density probabilities corresponding to all those four random variables.

In [29], the authors proposed a probabilistic representation of the solution of the system in (27). However, such a representation is rather stringent for practical purposes, since it requires to fulfill strong constraints in terms of the allowed initial data and time. Moreover, the density probability $p(\bar{k}')$ should be carefully chosen, hindering the task of finding a valid density for any dimension. This is related to the problem of finding admissible majorizing kernels, see [15]. Indeed, it can be readily proved that the majorizing kernel chosen in [29] is no longer valid in one dimensional problems. However, this does not mean that no solution can be found for the system of equations (27), but rather that the numerical method based on such a probabilistic representation gives rise to a divergent series, which requires further numerical treatment. In fact, in this section we show that relaxing the requirements concerning the initial condition and time, the solution is still smooth, and as explained in the previous section, we resort to Padé approximant [13,11] for approximating the asymptotic expansion of the solution given as divergent series. Since the density probability $p(\bar{k}')$ can now be chosen freely among a suitable class of functions, this can be used to reduce the statistical error done computing numerically the solution. Typically, this corresponds to the well known variance reduction techniques often used in Monte Carlo simulations.

Note that Eq.(32) is indeed a probabilistic representation of the Vlasov-Poisson system of equations (in the sense defined previously for the transport equations), and therefore, it can be used for computing the solution at a single point $(\bar{k},\bar{\xi},t)$, without the need of any computational mesh. This can be done

generating suitable random trees governed by the densities probabilities mentioned above, and taking the expected value of a corresponding multiplicative functional. For a numerical purpose, the associated numerical algorithm turns out to be specially costly for computing the solution in the whole computational domain, since a large sample size is typically required to avoid a large statistical error. However, an alternative does exist, and this will be addressed in Sec. 4.3.

4.2.1 Validation of the representation for the linear theory

In order to validate analytically the probabilistic representation derived above, we consider the classical linear Landau damping. This will be done linearizing conveniently the system (27) around the equilibrium solution. Let look for a density function according to the Ansatz

$$\begin{aligned} f^{(i)}(\bar{x},\bar{v},t) &= f_0^{(i)}(\bar{v}) + \varepsilon\, f_1^{(i)}(\bar{x},\bar{v},t) + O(\varepsilon^2), \\ \phi(\bar{x},t) &= \phi_0 + \varepsilon\,\phi_1(\bar{x},t) + O(\varepsilon^2), \end{aligned} \tag{34}$$

where $\varepsilon \ll 1$. Note that ϕ_0 is intentionally set to be constant to satisfy the constraint mentioned above concerning the space average over a period of the electric field. Inserting (34) into (27), we obtain to order ε

$$\begin{aligned} \frac{\partial f_1^{(1)}}{\partial t} + \bar{v}\cdot\nabla_{\bar{x}} f_1^{(1)} - \nabla\phi_1\cdot\nabla_{\bar{v}} f_0^{(1)} &= 0, \\ \frac{\partial f_1^{(2)}}{\partial t} + \bar{v}\cdot\nabla_{\bar{x}} f_1^{(2)} + \frac{1}{m_2}\nabla\phi_1\cdot\nabla_{\bar{v}} f_0^{(2)} &= 0, \\ \triangle_{\bar{x}}\phi_1 = -\left[\int f_1^{(1)} d\bar{v} - \int f_1^{(2)} d\bar{v}\right], & \end{aligned} \tag{35}$$

with initial data $f_1^{(i)}(\bar{x},\bar{v},0) = A\, g_i(\bar{v})\cos(k_1 x), i = 1,2$, 2π-periodic boundary conditions in $\bar{x}$, and $g_i(v)$ decaying to zero as $|\bar{v}|\to\infty$ with sufficiently high rate. Applying identical mathematical treatment as done previously for the fully Vlasov-Poisson system of equations, the following integral equation for the Fourier transform $F_{\bar{k}}^{(i)}(\bar{\xi},t)$ of $f_1^{(i)}$ is obtained,

$$\begin{aligned} F_{\bar{k}}^{(i)}(\bar{\xi},t) &= F_{\bar{k}}^{(i)}(\bar{\xi}-t\bar{k},0) \\ +\rho_i\int_0^t ds\,\frac{\bar{k}\cdot(\bar{\xi}-s\bar{k})}{|\bar{k}|^2}&[F_{\bar{k}}^{(1)}(0,t-s) - F_{\bar{k}}^{(2)}(0,t-s)]\hat{g}_i(\bar{\xi}_2 - s\bar{k}). \end{aligned} \tag{36}$$

Here $\hat{g}_i$ is the corresponding Fourier transform of $g_i(\bar{v})$. In the following let consider $\bar{\xi}=0$, and for simplicity we assume $\bar{k}=(k_1,0,0)$. The Fourier transform of the x-component of the electric field is given by, $\hat{E_x}(k_1,t) = -\frac{i}{k_1}[F_{k_1}^{(1)}(0,t) - F_{k_1}^{(2)}(0,t)]$. A recursive solution can be obtained for $\hat{E_x}(k_1,t)$ using Eq.(36), and yields,

$$\hat{E_x}(k_1,t) = -i\frac{1}{k_1}\Phi(k_1,t) + \sum_{j=1}^{\infty}(-1)^j\,\eta_j(\bar{k},t). \tag{37}$$

Here η_j is given by

$$\begin{aligned} \eta_j(k_1,t) = \frac{(-i)^j}{k_1^j}\int_0^t ds_1\, s_1 \int_0^{t-s_1} ds_2\, s_2 \cdots \int_0^{t-\sum_{l=1}^{j} s_l} ds_j\, s_j \\ \times\Phi(k_1, t-\sum_{l=1}^{j} s_l)\hat{h}(-s_1\, k_1)\hat{h}(-s_2\bar{k})\cdots\hat{h}(-s_j\bar{k}), \end{aligned} \tag{38}$$

where $\Phi(k_1,t) = F^{(1)}_{k_1}(-t\,k_1,0) - F^{(2)}_{k_1}(-t\,k_1,0)$, and $\hat{h}(-t\,k_1) = [\hat{g}_1(-t\,k_1) + \frac{1}{m_2}\hat{g}_2(-t\,k_1)]$. Note that η_j can be obtained recursively from η_{j-1} as follows

$$\eta_j(k_1,t) = \int_0^t ds\, s\, \eta_{j-1}(k_1,t-s)\hat{h}(-s\,k_1) \tag{39}$$

Multiplying Eq.(37) by $\hat{h}(-s\,k_1)$, and integrating with respect to s, it holds that

$$\int_0^t ds\, s\, \hat{E_x}(k_1,t-s)\hat{h}(-s\,k_1) = -\hat{E_x}(k_1,t) - i\frac{1}{k_1}\Phi(k_1,t), \tag{40}$$

The integral equation above turns out to be a Volterra equation of the second kind, whose solution can be obtained readily by means of the Laplace transform. In fact, Laplace transforming Eq. (40), we obtain

$$\tilde{\hat{E}}_x(k_1,p) = -i\frac{1}{k_1}\frac{\tilde{\Phi}(k_1,p)}{D(k_1,p)}, \tag{41}$$

where $D(k_1,p)$ is given by

$$D(k_1,p) = 1 - \frac{1}{k_1}\frac{d\tilde{\hat{h}}}{dp} \tag{42}$$

The solution can be obtained taking the inverse Laplace transform, and is given formally by

$$\hat{E_x}(k_1,t) = \frac{1}{2\pi i}\int_{\sigma-i\infty}^{\sigma+i\infty} \tilde{\hat{E}}_x(k_1,p)e^{p\,t}\,dp, \tag{43}$$

where integration is taken along a line parallel to the imaginary p-axis and to the right of all singularities of the integral. Then, it holds

$$\hat{E_x}(k_1,t) = \sum_j R_j e^{p_j\,t}, \tag{44}$$

where p_j are simple poles where the function $D(k_1,p)$ vanishes, and R_j is the residue of $\tilde{\hat{E}}_x$ at p_j. Since the poles are in general complex, Eq. (44) can be rewritten as

$$\hat{E_x}(k_1,t) = \sum_j R_j e^{\gamma_j\,t + i\omega_j t}, \tag{45}$$

being $p_j = \gamma_j + i\omega_j$. Note that if $\gamma_j < 0$, all terms are exponentially damped, and the electric field behaves as a damped oscillator, where γ_j, and ω_j denote the damping rate, and the frequency of the oscillation, respectively. In the following some specific examples are given:

(a) *Landau damping with homogeneous background.* Let consider only electron motion, assuming that the ions form a static, homogeneous, neutralizing background, that is $m_2 = \infty$, and the initial condition $f_1^{(2)}$ does not depend on $\bar{x}$. Suppose that the initial condition for the electrons is the maxwellian distribution, $g_1(\bar{v}) = (\alpha/\pi)^{d/2}\,\exp(-\alpha\bar{v}^2)$, that is

$$f_1^{(1)}(\bar{x},\bar{v},0) = A\,g_1(\bar{v})\,\cos(k_1 x). \tag{46}$$

Then, the function $D(k_1,p)$ takes the form

$$D(k_1,p) = 1 - \frac{\alpha}{k_1^2}Z'(\zeta), \tag{47}$$

where $\zeta = i\sqrt{\alpha}p/k_1$, and $Z(\zeta)$ the plasma dispersion function. Note that this reproduces exactly the classical result for the dispersion relation using the well-known linear Landau theory [24].

(b) *Landau damping with heterogeneous background*. Now we take into account also the ion dynamics. For simplicity, let assume that both initial conditions for ions and electrons are maxwellian distributions for velocities, given by

$$f_1^{(i)}(\bar{x},\bar{v},0) = A\left(\frac{\alpha_i}{\pi}\right)^{d/2} e^{-\alpha_1\bar{v}^2}\cos(k_1 x), \quad i=1,2 \tag{48}$$

The function $D(k_1,p)$ reduces to

$$D(k_1,p) = 1 - \frac{\alpha_1}{k_1^2}Z'(\zeta_1) - \frac{1}{m_2}\frac{\alpha_2}{k_1^2}Z'(\zeta_2), \tag{49}$$

where $\zeta_i = i\sqrt{\alpha_i}p/k_1\, i=1,2$. Again, this coincides exactly with the results obtained using the classical linear Landau theory [24].

4.3 Evaluating numerically the probabilistic representation of Vlasov-Poisson

Here we explain in detail how the probabilistic representation (32) can be used in practice to compute numerically the solution of the Fourier-transformed Vlasov-Poisson system, and which are the numerical errors done. For simplicity, in the following only the 1-dimensional case and one specie of charged particles (electrons) moving in a neutralizing homogeneous background charge (ions), has been considered. Eq.(31) can then further simplified by setting $i=1$ and $P(\rho=1)=1$. Note that the probabilistic representation for the Vlasov-Poisson system in (32) is not given in a closed-form. Following the same strategy as explained previously for the case of the transport equations, such an implicit equation can be solved recursively resorting to the aforementioned hybrid probabilistic representation approach, which in practice requires generating prescribed random trees governed by two random variables S, and k'.

Regarding numerical errors, recall that in general the convergence of the Padé approximant can be affected by artificial poles present in the denominator of the approximant, but not being own by the function to be approximated, see e.g. [11,13].

Concerning the apparent robustness of the Padé approximant against the statistical error affecting the coefficients of the series expansion, a main reason could be that the solution of the test examples seems to be apparently locally Lipschitz. Thus, the error made in computing the coefficients of the Padé approximant should be bounded. In fact, in [51] it has been proved the following related theorem

$$\parallel P_f - P_{f'} \parallel \leq K \parallel c - c' \parallel, \tag{50}$$

provided that $\parallel c - c' \parallel \leq d$. Here P_f, and $P_{f'}$ are the Padé approximants of order (m,n) in $[a,b]$ of a given power series f and f' with coefficients c_j, and c'_j respectively, being $\parallel c \parallel = max_{i\leq i\leq n+m}|c_i|$, f locally Lipschitz, and K and d constants depending only on c_i and $[a,b]$.

In closing, it is worth to observe that all errors described above may be alleviated in any case by increasing conveniently the sample size N, and considering more coefficients in the expansion in order to compute the Padé approximant.

4.3.1 Numerical test examples

Before analyzing the performance of our algorithm when ran in parallel, we give in the following some numerical examples. These are chosen from the classical repertoire of possible initial conditions traditionally used for testing numerical methods developed for Vlasov-Poisson system, and which describe certain phenomena well known in Plasma Physics. The ultimate goal is to characterize the accuracy of the algorithm in realistic cases.

Landau damping. Let consider the following initial condition

$$f(x,v,0) = \left(\frac{\alpha}{\pi}\right)^{d/2} e^{-\alpha v^2} \left[1 + A\cos\left(k_1\, x\right)\right], \tag{51}$$

which in Fourier space, reads,

$$F_k(\xi,0) = e^{-\frac{\xi^2}{4\alpha}} [A/2\,\delta(|k| - k_1) + \delta(k)] \tag{52}$$

The first numerical test deals with the so-called *weak* Landau damping, being the perturbation parameter A chosen sufficiently small. Here the damping rate and the oscillation frequency obtained numerically with our algorithm has been compared with the results obtained by the linear theory theoretically derived in Sec. 4.2.1. In Fig. 4 and 5, the damping rate and the oscillation frequency are shown for different values of α, which is related so far to different values of the plasma temperature β, being $\beta = 1/4\alpha$. Here A has been kept fixed to 0.01. The remarkable agreement between the analytical linear theory and the numerical results allows us to safely analyze more complicated situations such as the *strong* Landau damping, where the aforementioned filamentation phenomenon is significantly more severe. Let consider

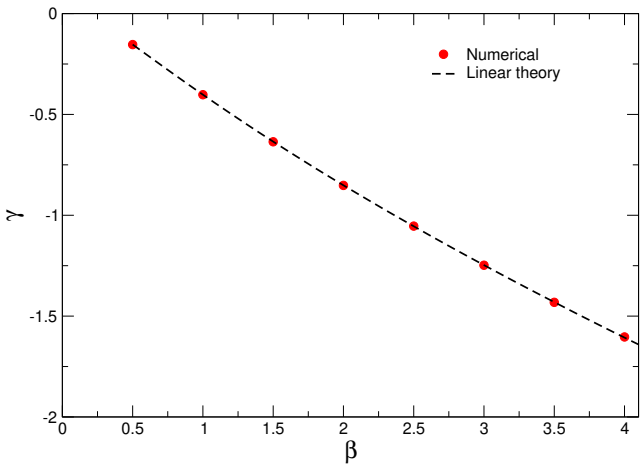

Fig. 4 Weak Landau damping: damping rate for different values of β. Parameters are: $A = 0.01$ and $M = 10^3$.

the same initial condition, but now choosing larger values of A, that is $A = 0.5$ for exploring the strong Landau damping regime. The numerical solution obtained by the PDD method has been compared with

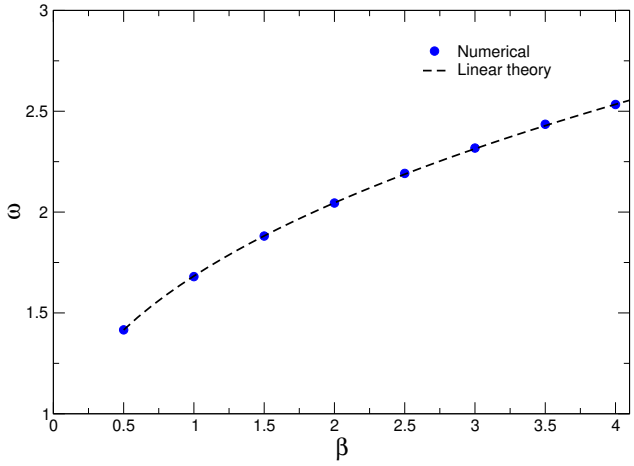

Fig. 5 Weak Landau damping: oscillation frequency for different values of β. Parameters are: $A = 0.01$ and $M = 10^3$.

the solution obtained when using an implicit upwind finite-difference scheme with a very fine mesh. The initial evolution of the mode $k = 1$ is shown in Fig. 6, showing the well-known filamentation phenomenon: an initial profile smooth in velocities, and peaked around $\xi = 0$ in Fourier space, evolves in time along the corresponding characteristics at constant velocity given by k, that is $F_k(\xi, t) = F_k(\xi - kt, 0)$. Thus, the solution propagates toward higher values of $|\xi|$, proportionally fast to the value of $|k|$, therefore faster for shorter wavelengths. Eventually this give rise to the development of small structures in the velocity distribution. So it is observed that the filamentation and mixing of modes appears strongly for long times in the nonlinear regime, and in the Fourier space, specifically for large values of ξ. Therefore, for this case, it has been considered a computational domain large enough in the $\xi-$dimension, despite in the figure it is only shown a part of it. To see more clearly that the solution is closely in agreement with the results obtained using other classical methods, the time evolution of the first harmonic of the electric field obtained by the PDD method is shown in Fig. 7, being qualitatively similar to the typical plots found in the literature [35, 22]. The solution obtained by the PDD method was satisfactorily compared in [6] with that obtained with an upwind implicit scheme with a very fine mesh. Here, $\Delta\xi$ has been kept fixed to 10^{-2} for the local solver. Note again the perfect agreement between the solution obtained by an upwind implicit scheme with a very fine mesh and our PDD method. The absolute numerical error has been numerically computed and it turns out to be or order of 10^{-2} in all simulations done.

Two streaming instability. Let consider the following initial condition, chosen for analyzing the two streaming instability phenomenon,

$$f(x, v, 0) = \left(\frac{\alpha}{\pi}\right)^{d/2} 2\alpha\ v^2 e^{-\alpha v^2} \left[1 + A\cos\left(k_1 \cdot x\right)\right], \tag{53}$$

which in Fourier space reads

$$F_k(\xi, 0) = (1 - \frac{\xi^2}{2\alpha}) e^{-\frac{\xi^2}{4\alpha}} \left[A/2\,\delta(|k| - k_1) + \delta(k)\right] \tag{54}$$

In Fig. 8 it is shown the time evolution of the predominant modes, where an initial, small perturbation leads to a final state characterized by a rapid modes grow and saturation, approximately at the time

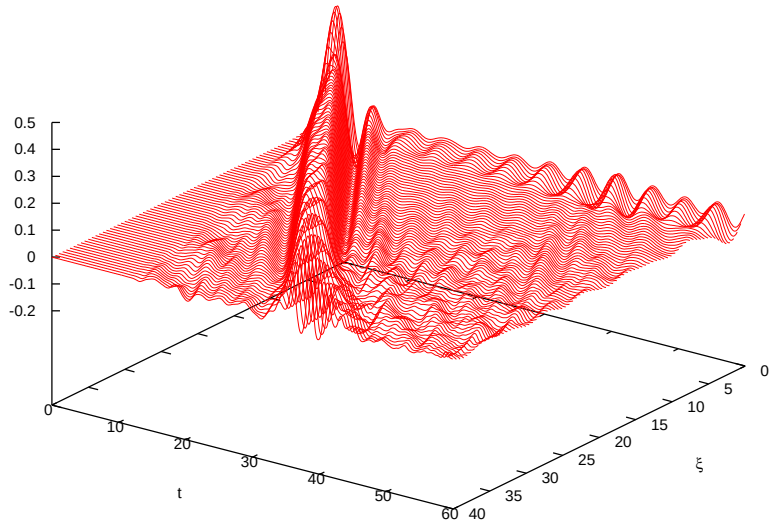

Fig. 6 Strong Landau damping: Time evolution of the numerical solution for $k = 1$. The contour lines correspond to $F_1(\xi, t) = 0$. Parameters are: $A = 0.5$, $\alpha = 2$ and $M = 10^3$.

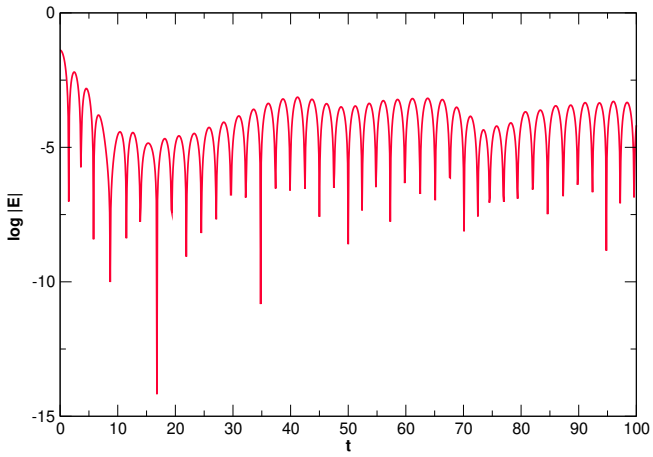

Fig. 7 Strong Landau damping: Logarithm of the absolute value of the first harmonic of the electric field. Parameters are: $A = 0.5$, and $\alpha = 2$.

$t = 20$. As already reported in literature by other authors (e.g., see [35]), the mode-one is the dominant due to its initial excitation and reaches its maximum amplitude at $t = 18$. Once again, the numerical solution has been compared with the solution obtained with an implicit upwind scheme with a very fine mesh, and the numerical error computed as in the previous example, obtaining a similar result.

For performance results of the PDD method for large scale simulations, see [6].

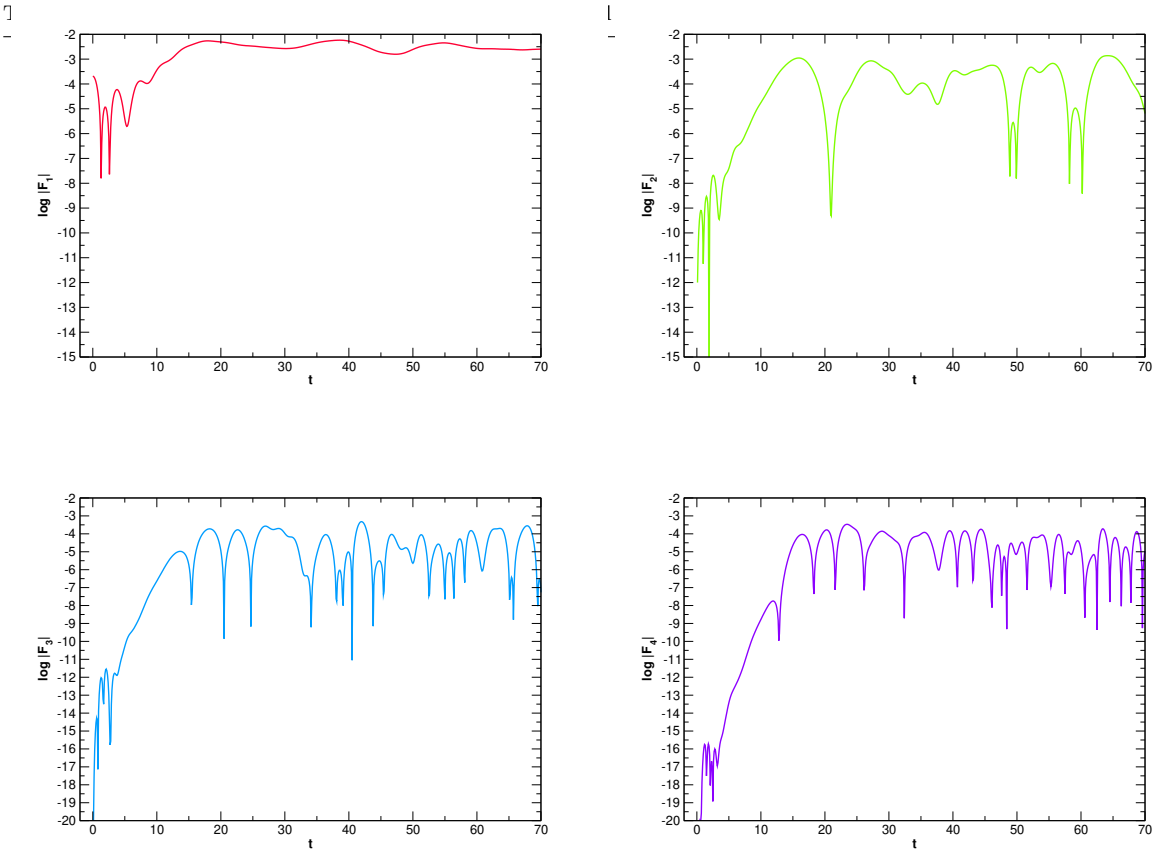

Fig. 8 Two streaming instability: Logarithm of the absolute values of the first four Fourier modes.

4.4 Summary

The PDD method presented shows that, when combined with classical methods, is capable of accelerating the Vlasov-Poisson simulations, thus improving dramatically the overall scalability of classical algorithms. Such method is based on the probabilistic representation of the Vlasov-Poisson system of equations, obtained in Fourier space and generalized to deal with any realistic initial condition. The probabilistic representation allows to compute the solution at single points within the computational domain, and is obtained as the expected value of a multiplicative functional over suitable random trees. Such a feature can be exploited to decouple the original problem into independent subproblems, previously obtaining the required boundary conditions at given interfaces dividing the domain. The probabilistic method was used to compute the solution at a few points, to be used as interpolation nodes to obtain the sought boundary conditions at the interfaces. It consists therefore of a straightforward application of the Probabilistic Domain Decomposition(PDD) method for the numerical solution of the Vlasov-Poisson system.

Moreover, the probabilistic representation was validated successfully in the linear regime comparing with the classical results of the linear Landau damping theory. Regarding the numerical implementation of such representation, this requires evaluating in practice some series with terms including definite integrals, corresponding to the partial contribution to the solution of random trees with a given number of branches. Typically, the higher terms are high dimensional, and were calculated by quasi-Monte Carlo methods.

Rather than classical Monte Carlo method, the quasi-Monte Carlo offers a better convergence rate, of order of $O(1/N)$ compared with $O(N^{-1/2})$, speeding up notably the simulations. When dealing with arbitrary initial conditions, such a series might be divergent, and was approximated by the Padé approximant. To study the error made, several test problems were analyzed so far, and it was shown that considering a few coefficients of the series suffices to obtain a reasonable accuracy for sufficiently small times. Since the approximation degrades fast when the time grows, and to avoid including higher terms in the series expansion with the computational cost that this entails, a restarting procedure in time has been proposed. The solution is computed globally in the full domain from time to time, and it is reused as the new initial condition restarting the numerical procedure again. Being now the new initial condition numerically obtained, a suitable interpolation procedure was implemented, which in practice degrades the theoretical expected performance of the algorithm. However, some theoretical considerations were given and applied to the algorithm to improve its overall performance, reducing the computational cost associated to such a global interpolation.

To conclude, several examples were run in parallel and the results compared with those obtained with classical algorithms. The examples were chosen from the typical repertoire of initial conditions traditionally used for testing numerical methods developed for solving the Vlasov-Poisson system of equations. The results shows the excellent scalability properties of the algorithm proposed when run in large-scale simulations.

It is worth to remark that the method can be further generalized to deal with the Vlasov-Poisson system in configuration space, since the needed probabilistic representation does already exist [50]. Moreover, the probabilistic representation can be combined with any classical existing numerical method according to the procedure described in this section, improving notably the performance of the resulting algorithm when run in parallel supercomputers.

5 Fractional Partial Differential boundary-value problems

In these days, there is a renewed interest in Fractional partial differential equations (fPDEs). Relevant aaplications in Science and Engineering include, for instance, control, biological tissues, materials for civil engineering, neurosciences, complex (heterogeneous and random) media, plasma physics, seismology and earthquakes modeling.

While solving purely initial-value problems seems to be to some extent tractable, the case of boundary-value problems on a smooth bounded domain $\Omega \subset \mathbb{R}^n$ is quite different, and it was observed that the results depends strongly on the definition of the fractional derivative used so far [27,49]. One of the most important differences among these variety of fractional derivatives are in the type of boundary data. Essentially there are of two types: Those nonlocal boundary conditions (also called extended boundary conditions) which are imposed on the complement Ω^c of the domain, and the local boundary conditions which are given only on $\partial\Omega$. The latter coincides with the type of boundary conditions typically imposed for classical partial differential equations, and moreover, under computational point of view, it has been found to be the more advantageous for dealing with large scale problems. In fact, note that the nonlocal boundary conditions require in practice to be able to tackle the unbounded region Ω^c, which can be

computationally very costly for solving numerically those large scale problems. Therefore, in the following we focus exclusively on the case of local boundary conditions.

A promising numerical method for solving fPDEs in bounded domains was recently proposed in [23] by using the so-called spectral fractional derivative. The spectral fractional derivative is a nonlocal operator, which is defined mathematically as the spectral decomposition of the standard Laplace operator, and is given by

$$(-\Delta)^{\beta/2} v(x) = -\frac{1}{\Gamma(-\beta/2)} \int_0^\infty (e^{t\Delta} v(x) - v(x)) \frac{dt}{t^{1+\beta/2}}. \tag{55}$$

Here Δ denotes the classical Laplacian operator, and the exponential operator e^{Δ} is formally defined, as usual, through its expansion in Taylor series,

$$e^{\Delta} = \sum_{k=1}^{\infty} \frac{(\Delta)^k}{k!}. \tag{56}$$

The main idea of the numerical method consists in exploiting the integral formulation of the fractional operator using the classical heat-semigroup formalism. One of the main advantage of this formalism rests on the fact that classical methods, such as the well-known finite element method, could be adopted to solve as well fPDEs by adapting it conveniently to this different mathematical framework. This in practice will allow to potentially solve fPDEs in arbitrary complex geometries and boundary conditions, which is of paramount importance in order to be able to analyze natural phenomena modeled by fPDEs in realistic environments.

An important problem, however, remains open, which is the possibility of using this numerical method for solving large scale problems. Since the numerical method is based on the finite element method, and therefore it requires a computational mesh to be solved, it inherits all of its disadvantages. In fact an important disadvantage of the method is the strong intercommunication overhead of the algorithm for solving large scale problems in distributed memory parallel computers. The major problem is the routine use of computational meshes when solving numerically a given problem. Since the mesh is a numerical tool connecting globally the discretized domain, any classical domain decomposition techniques induce an unavoidable communication among the processors involved when the numerical method is parallelized. It is worth pointing out that such a communication overhead acts always negatively degrading the performance of the algorithms, being even worse when using a large number of cores. Note that for the case of fPDEs this may be even more dramatic since the fractional operators are by definition nonlocal.

An alternative to the aforementioned domain decomposition method does exist, and consists in probabilistic methods based on Monte Carlo simulations. The main advantage of the probabilistic methods are mainly due to its special computational features, such as simplicity to code and parallelization. This in practice allows to develop parallel codes with extremely low communication overhead among processors, having a positive impact in parallel features such as scalability and fault-tolerance. Furthermore, there is also another distinguishing aspect of the method, which is the capability of computing the solution of the problem at specific chosen points, without the need of solving the entire problem. This remarkable feature has been explored for efficiently solving continuous problems such as boundary-value problems for classical PDEs offering important advantages in dealing with some specific applications found in Science and Engineering.

A feasible alternative, therefore, consists in generalizing the PDD method for solving now fPDE boundary-value problems. In principle, the PDD method could be applied to any problem, provided a probabilistic representation of the solution can be found. For the specific case of the spectral fractional Laplacian, it is known that the spectral operator in Eq. (55) with Dirichlet boundary conditions is the generator of a suitable subordinate stopped Brownian motion, i.e., stopped Brownian motion that is then *subordinated* by the standard *stable subordinator*. Therefore, it can be derived an analogous Feynman-Kac formula, where the corresponding Brownian motion is replaced now by a subordinate stopped Brownian motion [49]. Thus, the solution of the homogeneous Dirichlet boundary-value problem

$$\begin{aligned} \frac{\partial u}{\partial t} = \Delta^{\beta/2} u, \quad a < x < b,\, t > 0 \\ u(x,0) = f(x). \\ u(a,t) = f(a) = 0,\, u(b,t) = f(b) = 0, \end{aligned} \tag{57}$$

can be represented probabilistically as follows

$$u(x,t) = E\left[f(X_{S_t})\, \mathbf{1}_{[\tau_{\partial\Omega} > S_t]}\right] \tag{58}$$

where S_t is the subordinate process, that is an increasing *stable Lévy process* with index β, and X_t the corresponding Brownian motion. Here $\tau_{\partial\Omega}$ denotes the first exit time of the path X_t, started at $X_0 = x$, when $\partial\Omega$ is crossed, and $\mathbf{1}_{[\tau_{\partial\Omega} > t]}$ is the indicator (or characteristic) function. Note that in practice the subordination process consists merely of replacing the time t by the operational time given by the subordinator S_t.

Generating the stopped Brownian motion can be done simply by using the standard numerical techniques already available for solving probabilistically classical partial differential equations. A special care should be paid, however, when computing the first exit time and point out of the domain, as it was already mentioned in previous sections. Concerning the subordinated process, there are already several procedures available in the literature to generate numerically such a process, see [45] e.g. Since the solution is computed through an expected value of a given finite sample N, whose elements are independent from each other, it becomes straightforward to be parallelized.

In the following, to illustrate the probabilistic method we solved a simple example consisting in a Dirichlet boundary value problem for a 1D space-fractional diffusion equation in Eq. (57) using the probabilistic representation of the solution as described above. For this simple example, there exists an analytical solution [27] and is given by

$$u(x,t) = \frac{2}{\pi} \sum_{k=1}^{\infty} \frac{\cos(2k\pi/5) - \cos(3k\pi/5)}{k} e^{-(k\pi/2)^{\beta} t} \sin(k\pi/2(1+x)). \tag{59}$$

for the problem in the domain $x \in [-1, 1]$, and initial condition

$$f(x) = \begin{cases} 1 & \text{if } x \in (-0.2, 0.2) \\ 0 & \text{otherwise} \end{cases} \tag{60}$$

In Fig. 5 we compare the results corresponding to the numerical solution obtained using the probabilistic representation at different spatial points inside the domain, and the analytical solution plotted in

solid line. The results correspond to two different times, $t = 1$, and $t = 2$. Note the excellent agreement between both solutions.

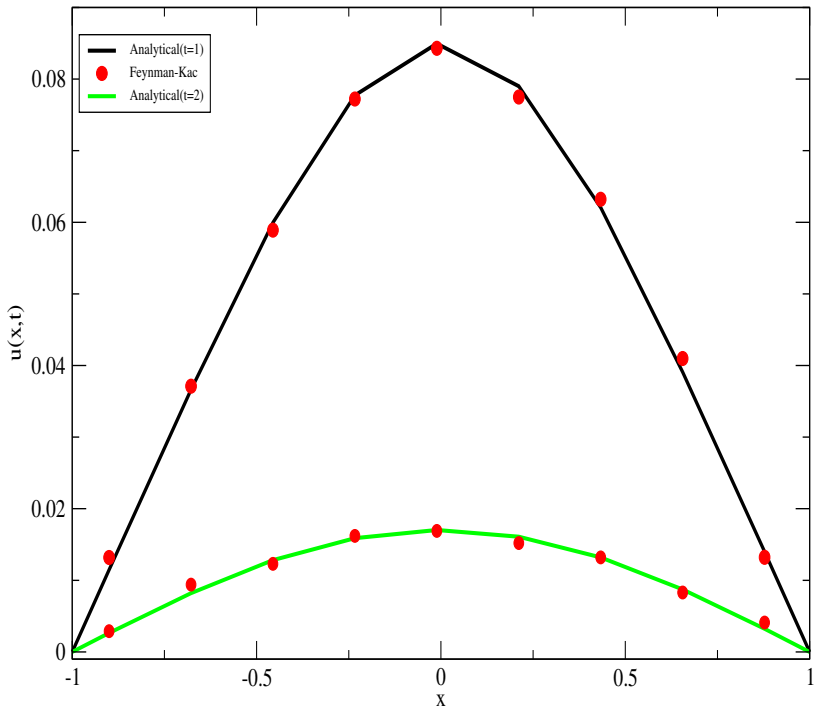

Fig. 9 Comparison between the probabilistic solution and the analytical solution of a Dirichlet fPDE boundary value problem. The fractional index has been kept fixed to $\beta = 1.5$, the sample size $N = 10^4$, and the time step $\Delta t = 10^{-2}$.

6 Conclusions

We have reviewed the PDD (probabilistic domain decomposition) method for numerically solving a wide range of linear and nonlinear partial differential equations of parabolic and hyperbolic type, as well as for fractional equations. This method was originally introduced for solving linear elliptic problems. It is based on a novel hybrid approach where an efficient Domain Decomposition is effectively accomplished by means of the probabilistic representation of the solution of the problem.

After an introduction detailing how this method works for linear parabolic problems, we have first showed the probabilistic representation for certain nonlinear parabolic problems, generalizing the results derived by McKean for the KPP equation. Later, a further extension of the method for dealing with semilinear parabolic partial differential equations has been presented. It is important to emphasize the major drawback faced in the attempt of such generalization, consisting in the numerical evaluation of a

truncated, alternating divergent series. This issue was sorted out by relying on Padé-type approximants to obtain numerically asymptotic approximations, always proven to be robust and reliable.

On the other hand, for semilinear transport equations, a probabilistic representations has been proposed in terms of the characteristic curves. It is important to remark that for hyperbolic problems in general, the characteristic curves play a similar role in such a representation as the stochastic process does for the parabolic problems. As an important application, the PDD method has been extended to deal with the Vlasov-Poisson system of equations in Fourier Space, which represents a very important and challenging problem in the field of Plasma Physics. Here, an existing probabilistic representation in Fourier space has been conveniently reformulated for computational purposes, validated successfully in the linear regime comparing with the classical results of the linear Landau damping theory. This theory has also been used to validate the probabilistic method numerically, which has had particular importance given the lack of exact analytical solutions.

In practice, for semi-linear problems a series with terms composed of definite integrals have to be evaluated, corresponding to the partial contribution to the solution of random trees with a given number of branches. Typically, the higher terms are of high dimensionality, and are calculated by the quasi-Monte Carlo method. Rather than classical Monte Carlo method, the quasi-Monte Carlo offers a better convergence rate, speeding up notably the simulations. When dealing with arbitrary initial conditions, such a series might be divergent, and was again approximated by the Padé approximant. The PDD method has shown theoretically and in practice to accelerate the numerical simulations, improving dramatically the overall scalability of classical algorithms.

Finally, we have shown the latest progress of the PDD method for dealing with fractional PDEs. We have provided a case example for the 1D space-fractional diffusion equation, showing promising results. As future work, the PDD method should be generalize further to deal with different kind of fractional operators (in time as well), an also for high dimensions.

As a final remark, when applicable, the PDD method enables for sustained high performance computing when solving challenging scientific problems formulated via partial differential equations. As observed through several test examples when compared with classical domain decomposition techniques, the method achieves superior performance and scalability results on supercomputing environments. Based on the fact that it is naturally fault-tolerant, the method also provides robustness and reliability by construction. Given that the subproblems are fully decoupled, a system failure is no longer dramatic, as it is simply required to run the simulation again, possibly asynchronously, only for the set of subproblems that have not finished successfully.

References

1. J. A. Acebrón, M. P. Busico, P. Lanucara, and R. Spigler. Domain decomposition solution of elliptic boundary-value problems via Monte Carlo and Quasi-Monte Carlo methods, SIAM Journal on Scientific Computing. 27(2):440–457, 2005.
2. J. A. Acebrón, M. P. Busico, P. Lanucara, and R. Spigler. Probabilistically induced domain decomposition methods for elliptic boundary-value problems. Journal of Computational Physics, 210(2):421–438, 2005.
3. J. A. Acebrón, A. Rodríguez-Rozas, and R. Spigler. Domain decomposition solution of nonlinear two-dimensional parabolic problems by random trees. Journal of Computational Physics, 228(15):5574–5591, 2009.

4. J. A. Acebrón, A. Rodríguez-Rozas, and R. Spigler. Efficient Parallel Solution of Nonlinear Parabolic Partial Differential Equations by a Probabilistic Domain Decomposition. Journal of Scientific Computing, 43(2):135–157, 2010.
5. J. A. Acebrón and A. Rodríguez-Rozas. A new parallel solver suited for arbitrary semilinear parabolic partial differential equations based on generalized random trees. Journal of Computational Physics, 230(21):7891–7909, 2011.
6. J. A. Acebrón and A. Rodríguez-Rozas. Highly efficient numerical algorithm based on random trees for accelerating parallel Vlasov-Poisson simulations. Journal of Computational Physics, 250:224–245, 2013.
7. J. A. Acebrón and M.A. Ribeiro, A Monte Carlo method for solving the one-dimensional telegraph equations with boundary conditions, Journal of Computational Physics, 305:29–43, 2016,
8. A. Rodríguez-Rozas, Highly Efficient Probabilistic-Based Numerical Algorithms for Solving Partial Differential Equations on Massively Parallel Computers. PhD Thesis in Computational Engineering, Instituto Superior Técnico, Universidade Técnica de Lisboa, 2012.
9. Antia, H.M., *Numerical methods for scientists and engineers*, Tata McGraw-Hill, New Delhi, 1995.
10. Arnold, L., *Stochastic Differential Equations: Theory and Applications*, Wiley, New York, 1974.
11. G. A. Baker, and P. Graves-Morris, Padé Approximants. New York: Cambridge University Press, 1996.
12. Baldi, P., *Exact asymptotics for the probability of exit from a domain and applications to simulation*, Ann. Prob., **23** (1995), pp. 1644–1670.
13. C. Bender, and S. A. Orszag, Advanced Mathematical Methods for Scientists and Engineers, McGraw Hill, New York (1978)
14. F. Bernal, G. dos Reis, and G. Smith, "Hybrid PDE solver for data-driven problems and modern branching", European J. Appl. Math., 28, pp. 949-972 (2017)
15. R. Bhattacharya, L. Chen, S. Dobson, R. B. Guenther, and C. Orum, Majorizing Kernels and Stochastic Cascades with Applications to Incompressible Navier-Stokes Equations. M. Ossiander, E. Thomann, and E. C. Waymire. Reviewed work (s): Source : Transactions of the American Mathematical Society , Vol . 355 , No. Transactions of the American Mathematical Society, 355:5003–5040, 2003.
16. A. Bihlo, R.D. Haynes and E.J. Walsh, Stochastic domain decomposition for time dependent adaptive mesh generation. J. Math. Study 48 (2), 106–124, 2015.
17. A. Bihlo and R.D. Haynes. Parallel stochastic methods for PDE based grid generation. Comput. Math. Appl. 68 (8), 804–820, 2014.
18. F.M. Buchmann, Computing exit times with the Euler scheme, Research report no. 2003-02, ETH (2003).
19. R. E. Caflisch. Monte Carlo and quasi-Monte Carlo methods, Acta Numerica, 7, 1-49, 1998.
20. W. Morokoff, and R. E. Caflisch, Quasi-Monte Carlo integration, Journal of Computational Physics, 122, 218-231, 1995.
21. B. Moskowitz, R. E. Caflisch, Smoothness and dimension reduction in Quasi-Monte Carlo methods, Mathematical and Computer Modelling, Volume 23, Issues 8-9, April-May 1996, Pages 37-54.
22. N. Crouseilles, M. Mehrenberger, and E. Sonnendrücker. Conservative semi-Lagrangian schemes for Vlasov equations. Journal of Computational Physics, 229(6):1927–1953, 2010.
23. N. Cusimano, F. del Teso, L. Gerardo-Giorda, and G. Pagnini, Discretizations of the spectral fractional Laplacian on general domains with Dirichlet, Neumann, and Robin boundary conditions, SIAM J. Numer. Anal., 56: 1243–1272, 2018.
24. R. C. Davidson, Kinetic Waves and Instabilities in Uniform Plasma, in Handbook of Plasma Physics by A. A. Galeev and R. N. Sudan, Volume 1, pp.521–585. North-Holland Publishing Company (1983).
25. DeBoor, C., *A practical guide to splines*, Springer, 1994.
26. DuChateau, P. and Zachmann, D., *Applied Partial Differential Equations*. Dover Publications (2002).
27. S. Duo, H. Wang, and Y. Zhang, A comparative study on nonlocal diffusion operators related to the fractional Laplacian,Discrete and Continuous Dynamical Systems B 24 231–256, 2019.
28. B. Eliasson. Outflow Boundary Conditions for the Fourier Transformed One-Dimensional Vlasov Poisson System. Journal of Scientific Computing, 16(1):1–28, 2001.
29. E. Floriani, R. Lima, and R. Vilela Mendes. Poisson-Vlasov: stochastic representation and numerical codes. The European Physical Journal D, 46(2):295–302, 2007.

30. M. Freidlin, Functional Integration and Partial Differential Equations. Annals of Mathematics Studies no. 109, Princeton Univ. Press, Princeton (1985)
31. Gobet, E., *Weak approximation of killed diffusion using Euler schemes*, Stochastic Processes and their Applications, **87**, (2000), 167–197.
32. Gobet, E. and Menozzi, S. Stopped diffusion processes: boundary corrections and overshoot. Stoch. Process. Appl. 120, 130-162, 2010.
33. Kalos, M.H., and Withlock, P.A., *Monte Carlo Methods, Vol. I: Basics*, Wiley, New York (1986).
34. I. Karatzas, and S. E. Shreve, Brownian Motion and Stochastic Calculus. 2nd ed., Springer, Berlin (1991)
35. A. J. Klimas. A numerical method based on the Fourier-Fourier transform approach for modeling 1-D electron plasma evolution. Journal of Computational Mathematics, 50:270–306, 1983.
36. A. J. Klimas. Vlasov-Maxwell and Vlasov-Poisson equations as models of a one-dimensional electron plasma. Physics of Fluids, 26(2):478, 1983.
37. Kloeden, P.E., and Platen, E., *Numerical Solution of Stochastic Differential Equations*, Springer, Berlin, 1992.
38. H. P. McKean, Application of brownian motion to the equation of Kolmogorov-Petrovskii-Piskunov., Communications on Pure and Applied Mathematics, 28 (1975), 323-331.
39. Mannella, R., *Absorbing boundaries and optimal stopping in a stochastic differential equation*, Phys. Lett. A, **254** (1999), pp. 257–262.
40. G.N. Milstein, and M.V. Tretyakov, Stochastic Numerics for Mathematical Physics. Springer (2004)
41. H. Niederreiter, Random Number Generation and Quasi-Monte Carlo Methods, Society for Industrial and Applied Mathematics, Philadelphia, (1992).
42. F. M. Buchmann, W. P. Petersen., An Exit Probability Approach to Solving High Dimensional Dirichlet Problems, SIAM J. Scientific Computing, 28(3):1153-1166, 2006.
43. J.M. Ramirez, Multiplicative cascades applied to PDEs (two numerical examples), Journal of Computational Physics, 214 (2006), 122-136.
44. H. Regnier, and D. Talay, Special Edition of the Proceedings of the Royal Society on Stochastic Analysis A(460), 199-220, 2004.
45. D. Fulger, E. Scalas, and G. Germano, Monte Carlo simulation of uncoupled continuous-time random walks yielding a stochastic solution of the space-time fractional diffusion equation, Phys. Rev. E 77, 1–7, 2008.
46. Shikin, E.V., and Plis, A.I., *Handbook on splines for the user*, CRC-Press, 1995.
47. I. H. Sloan and H. Wózniakowski. When are quasi-Monte Carlo algorithms efficient for high dimensional integrals?, Journal of Complexity, 14:1-33, 1998. (refs: pp. 28, 29, 30)
48. Strittmatter, W., *Numerical simulation of the mean first passage time*, University Freiburg Report No. THEP 87/12 (unpublished).
49. A. Lischke, G. Pang, M. Gulian, F. Song, C. Glusa, X. Zheng, Z. Mao, W. Cai, M. Meerschaert, M. Ainsworth, and G. Karniadakis,What Is the Fractional Laplacian? A comparative review with new results, J. Comput. Phys., 404, 2020.
50. R. Vilela Mendes. Poisson-Vlasov in a strong magnetic field: A stochastic solution approach. Journal of Mathematical Physics, 51(4):043101, 2010.
51. L. Wuytack, On the conditioning of the Pade approximant problem, Lect. Notes Math. 888 (1981), 78-89.