Fourteenth International Conference on Domain Decomposition Methods
Editors: Ismael Herrera , David E. Keyes, Olof B. Widlund, Robert Yates ©2003 DDM.org

11. Domain Decomposition by Stochastic Methods

Éric PEIRANO and Denis TALAY [1]

1. Introduction: Monte Carlo methods for domain decomposition. As shown by P-L. Lions in [19], stochastic representations of solutions of linear and nonlinear partial differential equations are useful to analyze the convergence of the Schwarz alternating method and related decomposition methods. The aim of this note is to show that one can also deduce simulation algorithms from stochastic representations in view of decomposing domains. To summarize, the Monte Carlo method allows one to compute approximate Dirichlet conditions on the boundaries of the subdomains of the decomposition without approximating the solution in the whole domain. One can thus easily localize problems in bounded domains, or compute the solution outside sub–domains where the solution has strong variations or the viscosity is small or the coefficients are discontinuous, etc. An advantage of the Monte Carlo method is that it is extremely easy to program and the simulations can be made in parallel.

We start our discussion by recalling the use of a Monte Carlo method to approximate π. The method is a variant of Buffon's needle method. Draw a vertical line in the 2-D space. Attach the extremity of a needle with one unit length to the line and choose the angle θ between the needle and an horizontal line at random according to the uniform distribution on $\left[0, \frac{\pi}{2}\right]$.

One has

$$\mathbb{E}\cos(\theta) = \frac{2}{\pi}\int_0^{\frac{\pi}{2}} \cos(z)\, dz = \frac{2}{\pi}.$$

The Strong Law of Large Numbers implies that

$$\frac{\cos(\theta_1) + \ldots + \cos(\theta_N)}{N} \xrightarrow[N\to+\infty]{} \frac{2}{\pi} \text{ a.e.},$$

where the θ_k's are independent copies of θ, that is, they are independent random variables and have the same probability distribution as θ (in practice one throws the needle at random N times, or simulates that game on a computer by using random number generators).

Observe that, in order to construct our Monte Carlo method, we have written the quantity under consideration, namely $\frac{2}{\pi}$, as the expectation of a random variable whose law is explicitly known and easy to simulate. To solve partial differential equations the situation is usually much more complex. We start by considering a case where the probabilistic representation is simple, namely the heat equation in the whole space: consider a bounded continuous function f and

$$\begin{cases} \dfrac{\partial u}{\partial t}(t,x) = \dfrac{1}{2}\Delta u(t,x),\ 0 < t,\ x \in \mathbb{R}^d, \\ u(0,x) = f(x). \end{cases}$$

Thus

$$u(t,x) = \mathcal{G}_t * f(x),$$

where $\mathcal{G}_t$ is the heat kernel

$$\mathcal{G}_t(y) := \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{|y|^2}{2t}\right).$$

Observe that $\mathcal{G}_t$ is the density function of the random vector $\sqrt{t}\, G$ where G is a d dimensional Gaussian vector with zero mean and unit variance. We therefore have

$$u(t,x) = \mathbb{E} f(x + \sqrt{t}\, G). \tag{1.1}$$

[1]INRIA, 2004 Route des Lucioles, B.P. 93, 06902 Sophia-Antipolis (France)

In view of the Strong Law of Large Numbers it comes

$$u(t,x) \simeq \frac{1}{N}\sum_{k=1}^{N} f(x+\sqrt{t}\ G_k),$$

where the G_k's are independent copies of G. The error corresponding to N trials is

$$u(t,x) - \frac{1}{N}\sum_{k=1}^{N} f(x+\sqrt{t}\ G_k),$$

and therefore is random. It may be large but with small probabilities only when N is large, as shown by the Central Limit Theorem or more precise results such as the Berry–Esseen theorem:

Theorem 1.1 (Berry–Esseen) *Let $(X_k)_{k\geq 1}$ be a sequence of independent and identically distributed random variables with zero mean. Denote by σ the common standard deviation. Suppose that*

$$\mathbb{E}\,|X_1|^3 < +\infty.$$

Then

$$\begin{aligned}\epsilon_N := \sup_{x\in\mathbb{R}} &\left| \mathbb{P}\left(\frac{X_1+\cdots+X_N}{\sigma\sqrt{N}} \leq x\right) - \int_{-\infty}^{x} e^{-u^2/2}\frac{du}{\sqrt{2\pi}}\right| \\ &\leq \frac{C\ \mathbb{E}\,|X_1|^3}{\sigma^3\sqrt{N}}.\end{aligned}$$

In addition, one has $0.398 \leq C \leq 0.8$.

For a proof see, e.g., Shiryayev [24]. Using the preceding theorem one can easily estimate the minimal number N of simulations which allows one to get a prescribed accuracy ϵ with a probability larger than a prescribed confidence threshold $1-\delta$.

In order to approximate the solution of a general parabolic or elliptic equation by a Monte Carlo method we have to extend the probabilistic representation (1.1) to cases where the partial differential equation involves a differential operator different from the Laplace operator. Then the fundamental solution cannot be related to the law of random variables so simple as Gaussian laws, and we need to consider the class of stochastic processes which are solutions to stochastic differential equations. We now shortly introduce that difficult and widely studied subject.

2. Probabilistic representation of parabolic and elliptic equations.

Let $b:\mathbb{R}^d \to \mathbb{R}^d$ and $\sigma_j:\mathbb{R}^d \to \mathbb{R}^d$ $(1\leq j\leq r)$ be smooth vector fields. Denote by $\sigma(x)$ the matrix whose column vectors are the $\sigma_j(x)$'s. Consider the elliptic operator

$$L\psi(x) := \sum_{i=1}^{d} b^i(x)\ \partial_i\psi(x) + \frac{1}{2}\sum_{i,j=1}^{d} a_j^i(x)\ \partial_{ij}\psi(x),$$

where

$$a(x) := \sigma(x)\ \sigma(x)^t,$$

and the evolution problem

$$\begin{cases} \dfrac{\partial u}{\partial t}(t,x) &= Lu(t,x),\ t>0,\ x\in\mathbb{R}^d, \\ u(0,x) &= f(x),\ x\in\mathbb{R}^d. \end{cases} \tag{2.1}$$

Suppose that (2.1) a smooth solution with bounded derivatives on $[0,T]\times\mathbb{R}^d$. We aim to construct a probabilistic representation of that solution. To this end we introduce stochastic processes. A stochastic process is a family of random variables indexed by time. The time index may be (subsets of) $\mathbb{R}_+$ or $\mathbb{N}$.

Our basic stochastic process will be the one dimensional Brownian motion (W_t) which satisfies: for all integer $n>1$ and all times $0\leq t_1<\ldots<t_n$, the random vector $(W_{t_1}-W_{t_0},,\ldots,W_{t_n}-W_{t_{n-1}})$ is Gaussian with zero mean and diagonal covariance matrix; the diagonal terms of the covariance matrix are

$$\mathbb{E}\,(W_{t_j}-W_{t_{j-1}})^2=t_j-t_{j-1}.$$

By definition a d dimensional Brownian motion is a process $(W_t^1,\ldots,W_t^d)$ whose components are independent one dimensional Brownian motions. Observe that we can rewrite (1.1) as

$$u(t,x)=\mathbb{E}\,f(W_t(x)),$$

where W is a d dimensional Brownian motion starting from x at time 0. When the differential operator L is not the Laplace operator, one needs to consider more complex processes, namely the solutions of stochastic differential equations. Unfortunately these objects cannot be rigorously introduced without the heavy machinery of stochastic calculus (see, e.g., the textbooks by Friedman [13] and Karatzas and Shreve [17]). To avoid too many complexities we limit ourselves to introduce discrete time processes which approximate the solutions of stochastic differential equations and, owing to elementary calculations, we establish their link with the smooth solutions of equations of the type (2.1). Let us thus onsider the Euler scheme defined as

$$\begin{cases} X_0^h(x) &= x,\\ X_{(p+1)h}^h(x) &= X_{ph}^h+b(X_{ph}^h(x))\,h+\sum_{j=1}^r\sigma_j(X_{ph}^h(x))\,\sqrt{h}G_{p+1}^j,\end{cases}\tag{2.2}$$

where $h:=\frac{T}{M}$ is a discretization step of the time interval $[0,T]$ and (G_p^j) is a family of real valued independent Gaussian random variables with zero mean and unit variance. As the function $u(t,x)$ is supposed smooth with bounded derivatives and as $u(0,x)=f(x)$ for all x a Taylor expansion leads to

$$\begin{aligned}\mathbb{E}\,f(X_T^h(x))-u(T,x)&=\sum_{p=0}^{M-1}\mathbb{E}\left[u(T-(p+1)h,X_{(p+1)h}^h(x))-u(T-ph,X_{ph}^h(x))\right]\\ &=\mathbb{E}\left[u(T-(p+1)h,X_{ph}^h(x))-u(T-ph,X_{ph}^h(x))\right]\\ &\quad+h\sum_{p=0}^{M-1}\mathbb{E}\left[Lu(T-(p+1)h,X_{ph}^h(x))\right]+\sum_{p=0}^{M-1}\mathcal{O}(h^2)\\ &=h\sum_{p=0}^{M-1}\mathbb{E}\left[Lu(T-ph,X_{ph}^h(x))-\frac{\partial u}{\partial t}(T-ph,X_{ph}^h(x))\right]\\ &\quad+\sum_{p=0}^{M-1}\mathcal{O}(h^2)\\ &=\sum_{p=0}^{M-1}\mathcal{O}(h^2)\\ &=\mathcal{O}(h).\end{aligned}\tag{2.3}$$

Thus

$$u(T,x)=\mathbb{E}\,f(X_T^h(x))+\mathcal{O}(h).$$

Remark 2.1 *The Euler scheme is easy to simulate since it requires Gaussian simulations only. Noticing that $\sqrt{h}\ G_p^j$ has the same Gaussian distribution function as $W^j_{(p+1)h} - W^j_{ph}$, one can think the Euler scheme as a time discretization of the stochastic differential equation*

$$X_t(x) = x + \int_0^t b(X_s(x))\ ds\ +\ \sum_{j=1}^d \int_0^t \sigma_j(X_s(x))\ dW_s^j,\ 0 \leq t \leq T, \tag{2.4}$$

where $\int_0^t \sigma_j(X_s(x))\ dW_s^j$ denotes the 'stochastic integral of the process $(\sigma_j(X_s(x)))$ with respect to the Brownian motion (W_s^j)' whose construction requires long developments. Equation (2.4) is shown to have a unique solution (in the appropriate space of stochastic processes) when the vector fields b and σ_j are Lipschitz. One can prove that the exact probabilistic representation of (2.1) is

$$u(T,x) = \mathbb{E}\, f(X_T(x)). \tag{2.5}$$

The key point of the proof is the Itô's formula which one needs to use instead of the above Taylor expansion: for all real valued function ϕ of class $\mathcal{C}^{1,2}([0,T]\times\mathbb{R}^d)$ it holds that

$$\phi(t,X_t(x)) = \phi(0,x) + \int_0^t L\phi(s,X_s(x))\ ds + \sum_{i=1}^d \sum_{j=1}^r \int_0^t \partial_i\phi(s,X_s)\ \sigma_j^i(s,X_s)\ dW_s^j.$$

Using that formula and deep notions of stochastic calculus, for a very large class of parabolic problems it can be shown that, if a smooth solution exists, then it verifies the equality (2.5). Stochastic calculus techniques may also be useful to prove the existence of smooth solutions: for example, when the coefficients b_i and σ_j^i are smooth, then (2.4) defines a smooth stochastic flow of diffeomorphisms, so that the mapping $x \mapsto X_t(x)$ is almost surely differentiable; therefore, if the function f itself is smooth and its derivatives satisfy appropriate growth at infinity conditions, the mapping $x \mapsto \mathbb{E}\, f(X_t(x))$ also is differentiable: see, e.g., Kunita [18].

From the preceding consideration one deduces that

$$u(T,x) \simeq \frac{1}{N}\sum_{k=1}^N \mathbb{E}\, f(X_T^{h,k}(x)), \tag{2.6}$$

where

$$(X_{ph}^{h,k}(x),\ p = 0,\dots,M,\ k = 0,\dots,N)$$

is a family of independent trajectories of the Euler scheme. Such trajectories can be simulated as follows: in view of (2.2), owing to $d\times M$ calls to the generator of Gaussian random variables one obtains $(X_{ph}^{h,1}(x),\ p = 0,\dots,M)$. Then time is reset to 0, and $d\times M$ new calls to the generator allow one to obtain the second trajectory, and so on. One finally computes the right hand side of (2.6) by averaging the end points of the N trajectories. Observe that the simulations can be done in parallel instead of sequentially if one can distribute the computations on a set of processors. The number of communications between the processors is extremely weak. In counterpart one has to ensure that the processors run independent sequences of random numbers, which may require a clever programming.

The global error of the Monte Carlo method (2.6) is

$$u(T,x) - \frac{1}{N}\sum_{k=1}^N \mathbb{E}\, f(X_T^{h,k}) = \underbrace{u(T,x) - \mathbb{E}\, f(X_T^h(x))}_{=:\epsilon_d(h)}$$
$$+ \underbrace{\mathbb{E}\, f(X_T^h(x)) - \frac{1}{N}\sum_{k=1}^N \mathbb{E}\, f(X_T^{h,k})}_{=:\epsilon_s(h,N)}.$$

The *discretization error* is described by the inequality (2.3) and even more accurate estimates. Indeed, under various sets of hypotheses including cases where f is supposed measurable only, one has (Talay and Tubaro [27], Bally and Talay [2])

$$e_d(h) = C_f(T,x)\ h + Q_h(f,T,x)\ h^2$$

and

$$|C_f(T,x)| + sup_h|Q_h(f,T,x)| \leq K(T)\|f\|_\infty \frac{1+\|x\|^Q}{T^q}$$

for some real number $C_f(T,x)$ which does not depend on the discretization step h. Thus Romberg extrapolation techniques are available: for example, simulations with the discretization steps h and $\frac{h}{2}$ lead to a second order accuracy in view of

$$u(T,x) - \Big(2\ \mathbb{E}\, f(X_T^{h/2}(x)) - \mathbb{E}\, f(X_T^h(x))\Big) = 2\ e_d(h/2) - e_d(h) = \mathcal{O}(h^2).$$

The *statistical error* $s(h,N)$ is described by the Berry–Esseen theorem 1.1 or its variants. Notice that (2.3) ensures that the standard deviation of $X_T^h(x)$ and $\mathbb{E}\,|X_T^h(x)|^3$ can be bounded from above by constants which do not depend on h, so that the time discretization step plays no role in the choice of the number of simulations corresponding to a desired accuracy and a prescribed confidence interval.

For parabolic and elliptic equations with Dirichlet boundary conditions (Neumann boundary conditions respectively) probabilistic interpretations, and therefore Monte Carlo methods, are also available (for various probabilistic interpretations we again refer to, e.g., the textbooks by Friedman [13] and Karatzas and Shreve [17]). We here give an example of a parabolic problem with Dirichlet boundary conditions.

Let D be a domain in $\mathbb{R}^d$, and consider

$$\begin{cases} \dfrac{\partial u}{\partial t}(t,x) & = Lu(t,x),\ t>0,\ x\in D, \\ u(0,x) & = f(x),\ x\in D, \\ u(t,x) & = g(x),\ t>0,\ x\in\partial D, \end{cases}$$

where $f(x) = g(x)$ on ∂D. Under various sets of hypotheses one has

$$u(T,x) = \mathbb{E}\,[f(X_T(x))\ \mathbb{I}_{T<\tau}] + \mathbb{E}\,[g(X_\tau(x))\ \mathbb{I}_{T\geq\tau}],$$

where τ is the 'first exit time of the domain', that is,

$$\tau := \inf\{0\leq t,\ X_t(x)\in\partial D\}. \tag{2.7}$$

In view of that formula it is natural to numerically approximate $u(T,x)$ by using the Euler scheme stopped at its 'first exit time of the domain' τ^h, that is,

$$\tau^h := \inf\{0\leq p\leq M,\ X_{ph}^h\in\partial D\}\times h. \tag{2.8}$$

If the boundary condition were of Neumann type then the solution $u(T,x)$ would have been expressed in terms of a process whose trajectories are reflected at the boundary of the domain and the Monte Carlo method would have involved the 'reflected Euler scheme'. For the error analysis of the stopped or reflected Euler scheme we refer to Gobet [14] and [15], Costantini, Pacchiarotti and Sartoretto [9]. It is worthy to notice that Gobet shows that, to preserve a rate of convergence of order $\mathcal{O}(h)$ one has to define the first exit time of the domain of the Euler scheme in a more clever way than (2.8), and to add the simulation of random times to the preceding algorithm; that additional simulation often has a low cost.

3. Application to Domain Decomposition. In view of the preceding stochastic representations it seems interesting to study the following domain decomposition technique: localize the problem by building artificial boundaries, and compute the solution along these boundaries by Monte Carlo simulations. More precisely, given points x_i on the artificial boundaries, one simulate N independent trajectories issued from each x_i and average the values of the $X_T^{h,k}(x_i)$'s. If the original problem is posed in a bounded domain with Dirichlet (Neumann respectively) boundary conditions, then the simulation needs to involve the stopped (reflected respectively) Euler scheme.

An important issue consists in estimating the error induced by the 'stochastic approximations along the artificial boundaries of the decomposed domain'. At the time being this question is widely open. For parabolic problems corresponding to European options, Crépey [10] has done a pionneering work. Berthelot [4] is studying the case of the variational inequalities corresponding to American options. To our knowledge no precise result is available on the convergence rate of the global error corresponding to the combination of the Monte Carlo method (along the artificial boundaries) and a classical deterministic method for the numerical resolution of the original problem in each one of the sub–domains with approximate Dirichlet conditions obtained from simulation. The OMEGA research group at Inria Sophia Antipolis has obtained only very preliminary results in that direction.

4. Stochastic particle methods for nonlinear equations. Stochastic numerical methods have been developed for nonlinear equations. The structure of such methods is much more complex than for linear problems: for variational inequalities (particularly those which describe American option prices in finance) one has to consider backward stochastic differential equations (see, e.g., the review papers by El Karoui, Quenez and Pardoux [11], Pardoux [21]); for Burgers equation, McKean–Vlasov–Fokker–Planck and Boltzman equations one has to consider interacting stochastic particle systems and their limits in the 'propagation of chaos' sense (see, e.g., Sznitman [25], Bossy and Talay [7], Jourdain [16], Méléard [20], Fournier and Méléard [12]). For estimates on the numerical methods deduced from such stochastic representations, see, e.g., Chevance [8], Bally and Pagès [1] for backward stochastic differential equations; Bossy [5], Bossy and Jourdain [6], Talay [26] for interacting stochastic particle methods (the reference [6] being a pionneering work in the analysis of the convergence rate of stochastic particle methods for problems with boundary conditions).

We now give the example of an extension of the Sherman and Peskin [23] method for convection-reaction–diffusion equations

$$\left.\begin{aligned} &\frac{\partial V}{\partial t}(t,x) = LV(t,x) + f\circ V(t,x),\\ &V(0,x) = V_0(x), \end{aligned}\right\} \tag{4.1}$$

where L is defined as

$$L\psi(x) := b(x)\ \psi'(x) + \frac{1}{2}\sigma(x)^2\ \psi''(x),$$

V_0 is a distribution function, and f is a smooth function such that $V(t,\cdot)$ is a distribution function for all $t>0$. Set $u(t,x) := \dfrac{\partial V}{\partial x}(t,x)$. It solves

$$\left.\begin{aligned} \frac{\partial u}{\partial t}(t,x) = \frac{1}{2}\,\sigma^2(x)\,\frac{\partial^2 u}{\partial x^2}(t,x) + [b(x)+\sigma(x)\sigma'(x)]\frac{\partial u}{\partial x}(t,x) + b'(x)u(t,x)&\\ + f'\left(\int_{-\infty}^{x} u(t,y)\ dy\right)u(t,x),&\\ u(0,x) = V_0'(x),\qquad\qquad\qquad\qquad\qquad\qquad& \end{aligned}\right\} \tag{4.2}$$

with, for example (Fisher equation), $f(u) := u(u-1)$. The numerical method described below is based on the representation of the measure $u(T,x)\ dx$ in terms of the limit of the empirical distribution of the living particles at time T of a branching interacting particles system.

The algorithm is as follows.

(i) At time 0, N particles with mass $1/N$ are located at points $V_0^{-1}\big(\frac{i}{N}\big)$, $i = 1, \dots, N-1$, and at $V_0^{-1}\big(1 - \frac{1}{2N}\big)$.

(ii) Let h be the time discretization step; between times kh and $(k+1)h$ each particle living at time kh moves independently of the other particles; its position at time $(k+1)h$ is

$$\begin{aligned}\overline{Y}_{(k+1)h} &= \overline{Y}_{kh} + \left\{\sigma\left(\overline{Y}_{kh}\right)\sigma'\left(\overline{Y}_{kh}\right) - b\left(\overline{Y}_{kh}\right)\right\}h + \sigma\left(\overline{Y}_{kh}\right)\left(W_{(k+1)h} - W_h\right)\\ &\quad + \frac{1}{2}\sigma\left(\overline{Y}_{kh}\right)\sigma'\left(\overline{Y}_{kh}\right)\left\{\left(W_{(k+1)h} - W_h\right)^2 - h\right\}.\end{aligned}$$

(iii) At each time step one creates and deletes particles according to the following rule. Let $\overline{\mathcal{N}}^N_{(k+1)h}$ denote the number of particles living at time $(k+1)h$, and

$$\overline{V}^N((k+1)h, x) := \frac{1}{N}\sum_{j=1}^{\overline{\mathcal{N}}^N_{(k+1)h}} H(x - \overline{y}^j_{(k+1)h}),$$

where $\{\overline{y}^j_{(k+1)h}\}$ is the set of the simulated particles which are living at time $(k+1)h$ and H is the Heaviside function. The particle numbered j dies with probability $h\big|f' \circ \overline{V}^N((k+1)h, \overline{y}^j_{(k+1)h})\big|$. If $f' \circ \overline{V}^N((k+1)h, \overline{y}^j_{(k+1)h}) \geq 0$, it gives birth to two particles.

Finally, the function $\overline{V}^N$ is our approximation of $V(T,x)$. The corresponding statistical error is $\mathcal{O}\frac{1}{\sqrt{N}}$ and discretization error is $\mathcal{O}\sqrt{h} + \mathcal{O}\frac{1}{\sqrt{N}}$: see Régnier and Talay [22].

As the Monte Carlo methods considered in the preceding section for linear problems, the stochastic particle methods may be used to approximate the solutions to nonlinear problems on artificial boundaries. However a lot of work still needs to be done in that area, either to construct probabilistic interpretations or to develop numerical methods based on the probabilistic interpretations. To illustrate that point, consider the 2D Navier-Stokes equation

$$\begin{cases}\dfrac{\partial u}{\partial t}(t,x) = \nu\Delta u(t,x) - (u(t,x)\cdot\nabla)u(t,x) - \nabla p(t,x),\ 0 < t,\ x \in \partial D,\\ \text{div } u(t,x) = 0,\ 0 < t,\ x \in \partial D,\\ u(t,x) = 0,\ 0 < t,\ x \in \partial D,\\ u(0,x) = f(x),\ x \in D.\end{cases}$$

Set $\omega := \text{rot } u$. Benachour, Roynette and Vallois [3] have shown that

$$\int_{\bar{D}} \omega(t,x)\ f(x)\ \text{Leb}(dx) = \mathbb{E}_{\omega_0}\int_{\bar{D}} f(x)\ dY_t \text{ for all } f \in \mathcal{C}^\infty(D),$$

where (Y_t) is a measure valued branching process with reflected paths. Unfortunately the law of the branching times is quite complex (much more complex than in the case of the Sherman and Peskin algorithm discussed above) and, at the time being, there is no efficient way of simulating the process (Y_t).

5. A first numerical illustration: an elliptic problem.

One seeks the numerical approximation of $u(x,y)$ which satisfies

$$\begin{cases} L\,u &= f \quad \text{in} \quad D, \quad \text{with} \quad D = [0,2]\times[0,1], \\ u &= g \quad \text{on} \quad \partial D. \end{cases} \tag{5.1}$$

The elliptic operator is defined as

$$L = B^x(x,y)\frac{\partial}{\partial x} + B^y(x,y)\frac{\partial}{\partial x} + \frac{1}{2}a(x,y)\frac{\partial^2}{\partial x^2} + c(x,y)\frac{\partial^2}{\partial x\partial y} + \frac{1}{2}b(x,y)\frac{\partial^2}{\partial y^2}. \tag{5.2}$$

5.1. Problem definition.

The diffusion matrix is defined by the following coefficients: $a(x,y)$, $b(x,y)$ and $c(x,y)$. For $b(x,y)$ one has

$$b(x,y) = (C_B+1) + (x-1)^2,$$

and for $a(x,y)$, if $x \le 1$,

$$a(x,y) = \left\{ \frac{C_B - C_A}{\pi}\arctan[-C(x-0.8)] + \frac{C_B + C_A}{2} \right\} b(x,y), \tag{5.3}$$

else,

$$a(x,y) = \left\{ \frac{C_B - C_A}{\pi}\arctan[C(x-1.2)] + \frac{C_B + C_A}{2} \right\} b(x,y), \tag{5.4}$$

with $C_B = (K/(k\,\pi))^2$ and $C_A = (K/(l\,\pi))^2$. The last coefficient, $c(x,y)$, is defined as

$$c(x,y) = 0.1\,(x-1)^2 + 0.005. \tag{5.5}$$

The values of the different constants k, l, K and C are listed in Table 5.1.

Table 5.1: Numerical values for the constants in Eqs (5.3) and (5.4).

k	l	K	C
5	20	1	1

The drift vector is defined by its two components $B^x(x,y)$ and $B^y(x,y)$, that is,

$$\begin{cases} B^x(x,y) &= K\,c(x,y), \\ B^y(x,y) &= 0.1. \end{cases} \tag{5.6}$$

The function $f(x,y)$ reads

$$\begin{aligned} f(x,y) = {} & C_E(x)\sin(C_E(x)\pi x)\exp(-Ky)\,[c(x,y)K - B^x(x,y)] \\ & + \cos(C_E(x)\pi x)\exp(-Ky) \\ & \left[-KB^y(x,y) - \frac{1}{2}(C_E(x)\pi)^2 a(x,y) + \frac{1}{2}K^2 b(x,y) \right], \end{aligned}$$

with $C_E(x) = k$ for $x \in [0;0.8] \cup [1.2;2]$ and $C_E(x) = l$ for $x \in]0.8;1.2[$. The analytical solution to system (5.1) is

$$u(x,y) = 2 + \cos(C_E(x)\pi x)\exp(-Ky). \tag{5.7}$$

Figure 5.1 shows the shape of both the function $f(x,y)$ and the analytical solution $u(x,y)$ for the numerical values indicated in Table 5.1.

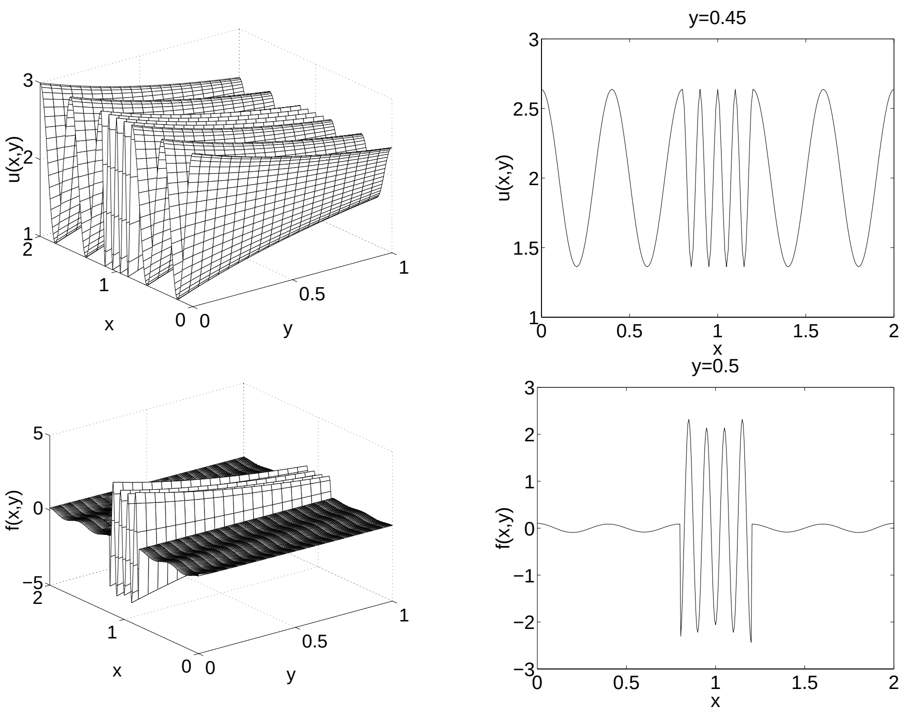


Figure 5.1: Top: shape of the analytical solution $u(x, y)$. Bottom: shape of $f(x, y)$. The results are given for the numerical values indicated in Table 5.1.

5.2. Deterministic method. Here, the elliptic problem is solved by classical discretisation techniques such as finite difference methods. For example, the second order derivative is approximated by (second order scheme)

$$\frac{\partial^2 u}{\partial x^2} = \frac{1}{(\Delta x)^2}[u(x+\Delta x, y) - 2u(x,y) + u(x-\Delta x, y)] + \mathrm{o}(\Delta x)^2. \tag{5.8}$$

All other derivatives are also computed with centered schemes. The computational domain is discretized with a uniform Cartesian mesh, that is

$$\begin{cases} x_i &= x_m + (i-1)(x_M - x_m)/(N_x - 1), \\ y_j &= y_m + (j-1)(y_M - y_m)/(N_y - 1). \end{cases} \tag{5.9}$$

The domain is defined by $[x_m, x_M] \times [y_m, y_M]$ and N_x and N_y represent the total number of discrete points in the x and y directions, respectively.

Eq. (5.1) can be written in its discretized form as $(u(x_i, y_j) = u_{i,j})$

$$\begin{aligned} f_{i,j} = {} & P_{imjp}\, u_{i-1,j+1} + P_{ijp}\, u_{i,j+1} + P_{ipjp}\, u_{i+1,j+1} \\ & + P_{imj}\, u_{i-1,j} + P_{ij}\, u_{i,j} + P_{ipj}\, u_{i+1,j} \\ & + P_{imjm}\, u_{i-1,j-1} + P_{ijm}\, u_{i,j-1} + P_{ipjm}\, u_{i+1,j-1}, \end{aligned} \tag{5.10}$$

where

$$\begin{aligned} &P_{imjm} = \frac{c_{i,j}}{4\,\Delta x\,\Delta y}, \quad P_{ijm} = \frac{b_{i,j}}{2\,\Delta x\,\Delta y} - \frac{B^y_{i,j}}{2\,\Delta y}, \quad P_{ipjm} = -P_{imjm}, \\ &P_{imj} = \frac{a_{i,j}}{2(\Delta x)^2} - \frac{B^x_{i,j}}{2\,\Delta x}, \quad P_{ij} = -\frac{a_{i,j}}{(\Delta x)^2} - \frac{b_{i,j}}{(\Delta y)^2}, \quad P_{ipj} = \frac{a_{i,j}}{2(\Delta x)^2} + \frac{B^x_{i,j}}{2\Delta x}, \\ &P_{imjp} = -P_{imjm}, \quad P_{ijp} = \frac{b_{i,j}}{2\,\Delta x\,\Delta y} + \frac{B^y_{i,j}}{2\Delta y}, \quad P_{ipjp} = P_{imjm}. \end{aligned}$$

The linear non-symmetric $(N_x - 2) \times (N_y - 2)$ system is solved with a preconditioned conjugate gradient method (NAG library, routine name: F11DEF).

The numerical error (relative error) is computed as follows (where $\mathcal{O}u(x,y)$ is the approximated value of $u(x,y)$)

$$e(x,y) = |u(x,y) - \mathcal{O}u(x,y)|, \tag{5.11}$$

which gives for the maximum error

$$e_{max} = \sup_{(x,y) \in D} e(x,y). \tag{5.12}$$

The maximum error is given, for different resolutions, in Table 5.2. The results show that (i) only the resolution along the x axis is important, (ii) the numerical scheme is, in reality, of first order in space. The CPU time is given for further comparison in execution time with alternative numerical methods (Monte-Carlo). The computer used to perform the simulations is a SUN Ultra 5/10 with a 440 MHz sparcv8plus+vis processor.

The shape of the numerical error along the x axis is displayed in Figure 5.2 for two different resolutions. It is seen, as expected, that the maximum error is obtained in the regions of steepest gradients.

Table 5.2: Numerical parameters and results for the deterministic method: spatial resolution, maximum error and CPU time (in seconds).

N_x	41	401	401	4001
N_y	21	201	21	201
Δx	0.05	0.005	0.005	0.0005
Δy	0.05	0.005	0.05	0.005
e_{max}	0.4164	0.0519	0.0519	0.0052
CPU (s)	0.1	33	0.8	744

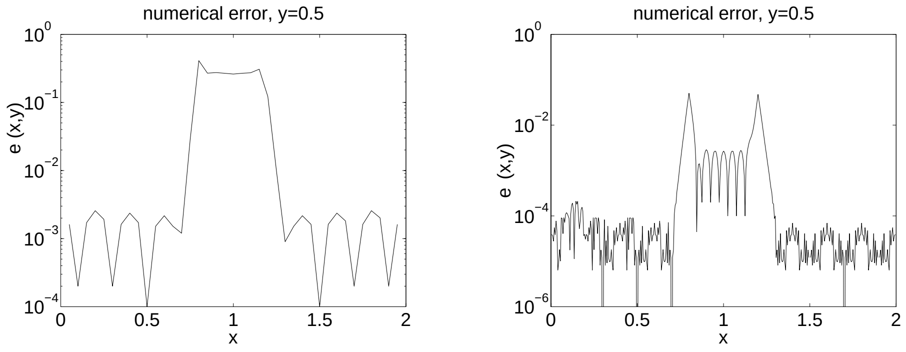


Figure 5.2: Numerical error along the x axis for $y = 0.5$. Two different resolutions are displayed: $(N_x, N_y) = (41, 21)$ (left) and $(N_x, N_y) = (401, 201)$ (right).

5.3. Probabilistic method. The probabilistic interpretation of the solution of (5.1) is

$$u(x,y) = -\mathbb{E}\left[\int_0^\tau f(X_t(x,y))\,dt\right] + \mathbb{E}\left[g(X_\tau(x,y))\right],$$

where the underlying stochastic process solves

$$\begin{cases} X_t^1(x,y) &= x + \int_0^t B^x(X_s(x,y))\,ds + \int_0^t \sigma_1^1(X_s(x,y))\,dW_s^1 + \int_0^t \sigma_2^1(X_s(x,y))\,dW_s^2, \\ X_t^2(x,y) &= y + \int_0^t B^y(X_s(x,y))\,dt + \int_0^t \sigma_1^2(X_s(x,y))\,dW_s^1 + \int_0^t \sigma_2^2(X_s(x,y))\,dW_s^2. \end{cases}$$

Here (W_t^1, W_t^2) are two independent Wiener processes, σ is a matrix valued function such that

$$\sigma(x,y)\sigma(x,y)^t = \begin{pmatrix} a(x,y) & c(x,y) \\ c(x,y) & b(x,y) \end{pmatrix},$$

and τ is the first exit time of the domain D as defined in Section 2.

The Euler scheme reads

$$\begin{aligned} {X^h}^1_{(p+1)h}(x,y) &= {X^h}^1_{ph}(x,y) + B^x(X^h_{ph}(x,y))h + \sigma_1^1(X^h_{ph}(x,y))\,(W^1_{(p+1)h} - W^1_{ph}) \\ &\quad + \sigma_2^1(X^h_{ph}(x,y))\,(W^2_{(p+1)h} - W^2_{ph}) \\ {X^h}^2_{(p+1)h}(x,y) &= {X^h}^2_{ph}(x,y) + B^y(X^h(x,y))h + \sigma_1^2(X^h_{ph}(x,y))\,(W^1_{(p+1)h} - W^1_{ph}) \\ &\quad + \sigma_2^2(X^h_{ph}(x,y))\,(W^2_{(p+1)h} - W^2_{ph}). \end{aligned}$$

The corresponding Monte Carlo approximation is defined as

$$\mathcal{O}u(x,y) = \frac{1}{N}\sum_{k=1}^{N}\left[-h\sum_{p=0}^{\tau^h-1} f(X_{ph}^{h,k}(x,y)) + g(X_{\tau^h}^{h,k}(x,y))\right].$$

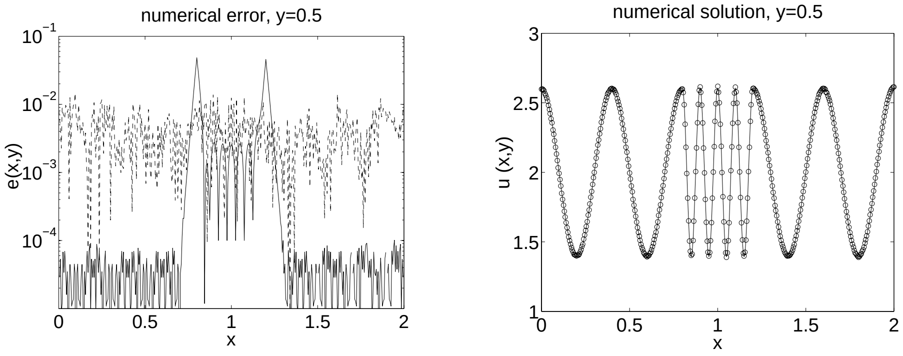


Figure 5.3: Numerical results for a Monte-Carlo simulation with $N = 10^4$ and $h = 10^{-4}$ s. The resolution is given by $(N_x, N_y) = (401, 21)$ but the results are presented along the x axis for $y = 0.5$. Left: numerical error (dashed line for the Monte-Carlo simulation and continuous line for the deterministic method). Right: numerical solution ($\circ$ for the Monte-Carlo simulation and continuous line for the exact solution).

Different numerical parameters (time steps and number of trajectories) and numerical procedures (Romberg extrapolation and treatment of the killed diffusion with Brownian bridges as proposed by Gobet [14] and [15]) were employed. In the computations presented here, $h = 10^{-4}$ and $N = 10^4$. The Euler scheme has been used with no specific treatment (the simulation is stopped at the time step where the point leaves the domain). The numerical error and the numerical solution are displayed in Figure 5.3. It can be observed that the numerical error is not too sensitive to the gradients. However, each Monte-Carlo point takes approximately 120 s of computer time.

Domain decomposition can now be performed. For example, one can consider the following domain: $D_1 = [0, 0.8] \times [0, 1]$. The computations can be done with the deterministic method by using the results of the Monte-Carlo simulation as boundary conditions (for $x = 0.8$). Figure 5.4 displays the results of such a computation.

The computation is performed for $(Nx, Ny) = (81, 21)$ on D_1. The Monte-Carlo points used as boundary conditions are obtained from the previous results presented in Figure 5.3.

It can be concluded that, even though the domain decomposition is technically feasible, there is no improvement for the CPU time. Indeed, for a similar precision in the domain of steep gradients ($x \in [0.8, 1.2]$), the CPU time for one Monte-Carlo point is roughly equal to the whole computation with the deterministic method and this for roughly 10^3 points.

6. A second numerical illustration: a parabolic problem. One seeks the numerical approximation of $u(x, y)$ which satisfies

$$\begin{cases} \dfrac{\partial u}{\partial t} + L\,u = 0 \quad \text{in} \quad D := [0,T] \times \mathbb{R}, \\ u(T,x) = f(x), \end{cases} \tag{6.1}$$

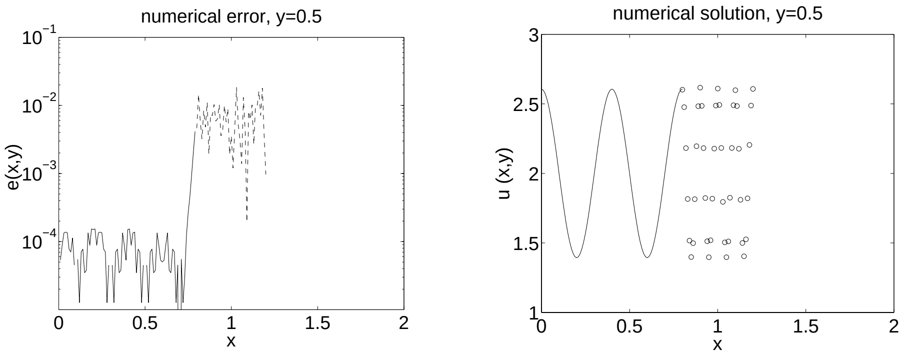


Figure 5.4: Numerical results for a deterministic/Monte-Carlo simulation. For the deterministic computation in D_1, one has $(N_x, N_y) = (81, 21)$. For the Monte-Carlo simulation $N = 10^4$ and $h = 10^{-4}$ s. The results are presented along the x axis for $y = 0.5$. Left: numerical error (dashed line for the Monte-Carlo simulation and continuous line for the deterministic method). Right: numerical solution ($\circ$ for the Monte-Carlo simulation and continuous line for the deterministic method).

where the elliptic operator L is defined as

$$L = D(t,x)\frac{\partial}{\partial x} + \frac{1}{2}B(t,x)\frac{\partial^2}{\partial x^2}. \tag{6.2}$$

The functions $D(t,x)$ and $B(t,x)$ are given by

$$D(t,x) = \cos(x)\,\sin(x)\left\{\lambda\cos(\lambda t) - [\cos(x)\exp(a(t))]^2\right\}, \tag{6.3}$$

and

$$B(t,x) = \left[\cos^2(x)\exp[a(t)]\right]^2, \tag{6.4}$$

respectively. The function $a(t)$ is defined as $a(t) = \sin(\lambda t)$ where λ is a constant (a real positive number). The final condition, $u(T,x) = f(x)$ is defined as

$$f(x) = \exp\left[\cos\left(\frac{1}{0.1+\lambda x^2}\right)\right] + \exp\left[\sin\left(\frac{1}{0.1+\mu x^2}\right)\right], \tag{6.5}$$

where $\mu \in \mathbb{R}^+$.

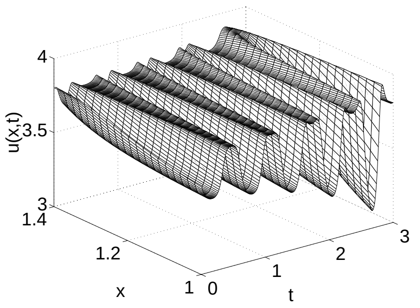
It can be shown that the analytical solution to system (6.1) is

$$u(t,x) = \int_{-\infty}^{+\infty} f\left(\arctan\left\{\exp[a(T)-a(t)]\tan(x) + y\,\exp[a(T)]\sqrt{T-t}\right\}\right) p(y)\,dy, \tag{6.6}$$

where $p(y)$ is the normal centered Gaussian law,

$$p(y) = \frac{1}{\sqrt{2\pi}}\exp(-y^2/2). \tag{6.7}$$

The solution can be computed numerically by resorting to a proper numerical procedure (here a NAG routine, D01AHF, is used). Figure 6.1 shows the shape of $u(t,x)$ for $[0,3]\times[1.0,1.4]$ ($f(x)$ is also shown).



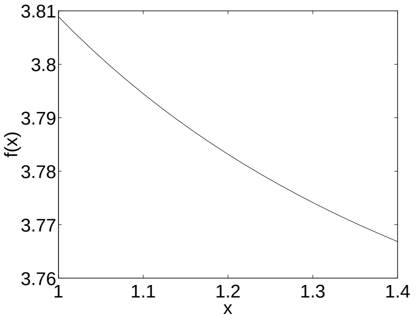


Figure 6.1: Left: approximated solution of system (6.1)on $[0,3]\times[1.0,1.4]$ by numerical integration of Eq. (6.6). Right: shape of the final condition, $f(x)$.

6.1. Deterministic method. System (6.1) can be solved by a simple deterministic method. A finite difference method is adopted where a first order approximation is used for the time derivative and a second order approximation for the space derivatives, that is, for example,

$$\frac{\partial u}{\partial x} = \frac{1}{2\Delta x}[u(x+h,x) - u(x-h,x)] + \mathrm{o}(\Delta x)^2,$$
$$\frac{\partial u}{\partial t} = \frac{1}{h}[u(t,x+h) - u(t,x)] + \mathrm{o}(h).$$

The scheme is explicit and it reads

$$u_i^{k+1} = -\frac{h}{2\Delta x}\left(D_i^k + \frac{B_i^k}{\Delta x}\right) u_{i+1}^k + \left(1 + \frac{B_i^k h}{2\,\Delta x^2}\right) u_i^k - \frac{\Delta t}{2\Delta x}\left(\frac{B_i^k}{\Delta x} - D_i^k\right) u_{i-1}^k,$$

where u_i^k is the approximation of $u(t,x)$ for $x = i\,\Delta x$ and $t = k\,h$ (Δx is the space resolution and h is the time step). Figure 6.2 shows the result of a computation on $[0,T]\times[A,B]$ with $\Delta x = 2.10^{-2}$ and $h = 10^{-3}$ (it has been checked by Von Neumann analysis that the scheme is stable for these values of Δx and h). The values of the different constants μ, λ, A, B and T are listed in Table 6.1. As in the previous examples, the maximum numerical error is obtained in the regions of steepest gradients, Figure 6.2.

Table 6.1: Numerical values for the numerical solution of system (6.1) on $[0,T]\times[A,B]$.

μ	λ	A	B	T
10	10	1.0	1.4	3

6.2. Probabilistic method. The preceding deterministic method requires to know exact values, or at least good approximations, of the solution $u(t,x)$ along the artificial boundaries $x = A$ and $x = B$. The Monte Carlo method allows one to get good approximations for all choice of the pair (A,B).

For all $0 \le t \le T$ the probabilistic interpretation of the solution to the system (6.1) is

$$u(t,x) = \mathbb{E}\left[f(X_T^{t,x})\right] \tag{6.8}$$

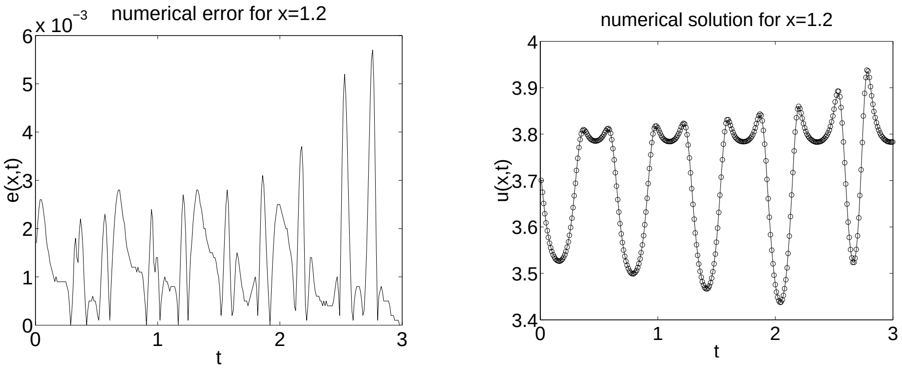


Figure 6.2: Numerical results for a deterministic simulation of system (6.1) on $[0, 3] \times [1.0, 1.4]$. The time step is $h = 10^{-3}$ and the space resolution is $\Delta x = 2.10^{-2}$. The results are presented along the t axis for $x = 1.2$. Left: numerical error. Right: numerical solution ($\circ$ for the simulation and continuous line for the 'exact' solution).

where the underlying stochastic process is the solution to

$$\begin{aligned} X_\theta^{t,x} = x &+ \int_t^\theta \cos(X_s^{t,x}) \sin(X_s^{t,x}) \left\{ a'(s) - \left[\cos(X_s^{t,x}) b(s)\right]^2 \right\} ds \\ &+ \int_t^\theta \cos^2(X_s^{t,x})\, b(s)\, dW(s), \ t \le \theta \le T. \end{aligned} \tag{6.9}$$

Here, $a'(t)$ is the time derivative of $a(t)$ and $b(t) = \exp[a(t)]$. Notice that the solution $u(t, x)$ is expressed in terms of a process starting at time t, whereas in (2.5) the solution $u(T, x)$ is expressed in terms of a process observed at time T: that difference is due to the fact that, in (6.1), the initial condition is fixed at time T instead of 0 and one integrates backward in time instead of forward in time, which leads to a more convenient probabilistic interpretation when the coefficients of L are time dependent.

The numerical solution is obtained by a Monte-Carlo simulation as done in Section 5.3. The trajectories of (X_t) are simulated by applying the Euler scheme to Eq. (6.9). Figure 6.3 shows the result of a Monte-Carlo computation for $N = 10^4$ and $h = 10^{-4}$, which gives a numerical precision of the same order of magnitude as the computation performed with the deterministic method, see Figure 6.2. However, for such a simulation, each Monte-Carlo point takes approximately 30 s of computer time whereas the deterministic method requires 2 s for 6000 nodes (the value of 30 s for a Monte-Carlo point is an expected value for several points in the domain since the computations are faster for points near the boundary), see Table 6.2.

Table 6.2: CPU time for the deterministic and the Monte-Carlo method.

	CPU time: 1 point	CPU time: 6000 nodes
Monte-Carlo method	30 s	
Deterministic method		2 s

7. Conclusion. It is possible to use Monte-Carlo methods for domain decomposition problems when solving PDEs with deterministic techniques. However, the CPU time required

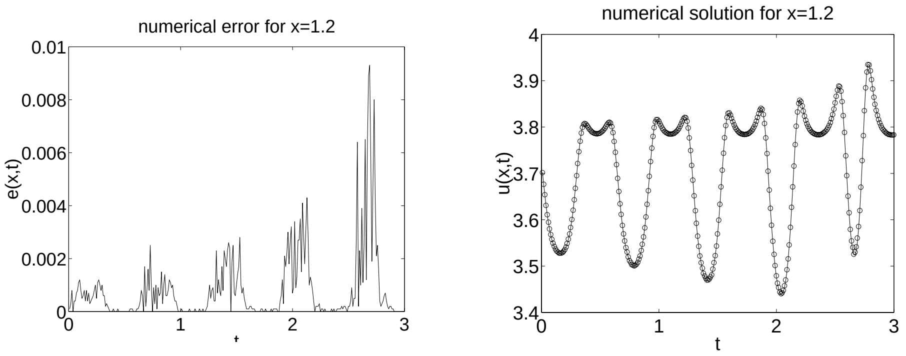


Figure 6.3: Numerical results for a Monte-Carlo simulation with $N = 10^4$ and $h = 10^{-4}$ s. The results are presented along the t axis for $x = 1.2$. Left: numerical error. Right: numerical solution ($\circ$ for the Monte-Carlo simulation and continuous line for the 'exact' solution).

by the Monte-Carlo methods does not make, in this sense, any improvement compared to full deterministic methods. Monte-Carlo methods are, however, interesting in cases where they are the only alternative (unknown boundary conditions or high dimensional problems, for example).

A widely still open problem is to find optimal estimates for the global error of algorithms combining deterministic methods in sub–domains and Monte Carlo methods to approximate the solutions along the artificial boundaries produced by a domain decomposition.

REFERENCES

[1] V. Bally and G. Pagès. A quantization algorithm for solving multi-dimensional optimal stopping problems. Submitted for publication, 2002.

[2] V. Bally and D. Talay. The law of the Euler scheme for stochastic differential equations (I) : convergence rate of the distribution function. *Probability Theory and Related Fields*, 104(1), 1996.

[3] S. Benachour, B. Roynette, and P. Vallois. Branching process associated with 2D Navier Stokes equation. *Revista Matematica Iberoamericana*, 2002 (to appear).

[4] C. Berthelot. PhD thesis in preparation.

[5] M. Bossy. Optimal rate of convergence of a stochastic particle method to solutions of 1D scalar conservation laws. Submitted for publication, 2002.

[6] M. Bossy and B. Jourdain. Rate of convergence of a stochastic particle method to solutions of 1D scalar conservation laws in a bounded interval. *Annals Prob.*, 2002. To appear.

[7] M. Bossy and D. Talay. A stochastic particle method for the McKean-Vlasov and the Burgers equation. *Math. Comp.*, 66(217):157–192, 1997.

[8] D. Chevance. Numerical methods for backward stochastic differential equations. In L. Rogers and D. Talay, editors, *Numerical Methods in Finance*, Publications of the Newton Institute. Cambridge University Press, 1997.

[9] C. Costantini, B. Pacchiarotti, and F. Sartoretto. Numerical approximation for functionals of reflecting diffusion processes. *SIAM J. Appl. Math.*, 58(1):73–102, 1998.

[10] S. Crépey. *Contribution des Méthodes Numériques Appliquées la Finance et aux Jeux*. PhD thesis, École Polytechnique, 2000.

[11] N. El Karoui, E. Pardoux, and M-C. Quenez. Reflected backward stochastic differential equations and American options. In L. Rogers and D. Talay, editors, *Numerical Methods in Finance*, Publications of the Newton Institute, pages 215–231. Cambridge University Press, 1997.

[12] N. Fournier and S. Méléard. A stochastic particle numerical method for 3D Boltzmann equations without cutoff. *Math. Comp.*, 71:583–604, 2002.

[13] A. Friedman. *Stochastic Differential Equations and Applications*, volume 1. Academic Press, New York, 1975.

[14] E. Gobet. Weak approximation of killed diffusion using Euler schemes. *Stoch. Proc. Appl.*, 87:167–197, 2000.

[15] E. Gobet. Euler schemes and half-space approximation for the simulation of diffusion in a domain. *ESAIM Probability and Statistics*, 5:261–297, 2001.

[16] B. Jourdain. Probabilistic characteristics method for a 1D scalar conservation law. *Annals Appl. Prob.*, 12(1):334–360, 2002.

[17] I. Karatzas and S. Shreve. *Brownian Motion and Stochastic Calculus.* Springer-Verlag, New York, 1988.

[18] H. Kunita. Stochastic differential equations and stochastic flows of diffeomorphisms. In *Ecole d'Été de Saint-Flour XII*, volume 1097 of *Lecture Notes in Mathematics.* Springer, 1984.

[19] P.-L. Lions. On the Schwarz alternating method. II. In T. Chan, R. Glowinski, J. Périaux, and O. Widlund, editors, *Domain Decomposition Methods*, pages 47–70, Philadelphia, PA, 1989. SIAM.

[20] S. Méléard. Asymptotic behaviour of some interacting particle systems; McKean–Vlasov and Boltzmann models. In D. Talay and L. Tubaro, editors, *Probabilistic Models for Nonlinear PDE's and Numerical Applications*, Lecture Notes in Math., Berlin, Heidelberg, New York, 1996. Springer Verlag.

[21] E. Pardoux. Backward stochastic differential equations and viscosity solutions of systems of semilinear parabolic and elliptic PDEs of second order. In L. Decreusefond, J. Gjerde, B. Oksendal, and A. Ustünel, editors, *Stochastic Analysis and Related Topics : The Geilo Workshop*, pages 79–127. Birkhäuser, 1998.

[22] H. Régnier and D. Talay. Vitesse de convergence d'une méthode particulaire stochastique avec branchements. *Note au C.R.A.S.*, t. 332(Série I):933–938, 2001.

[23] A. Sherman and C. Peskin. A Monte Carlo method for scalar reaction-diffusion equations. *SIAM J. Sci. Statist. Comput.*, 7(4):1360–1372, 1986.

[24] A. Shiryayev. *Probability.* Springer-Verlag New-York, 1984.

[25] A. Sznitman. Topics in propagation of chaos. In P. Hennequin, editor, *Ecole d'Eté de Probabilités de Saint-Flour XIX - 1989*, volume 1464 of *Lecture Notes in Math.*, pages 165–251, Berlin, Heidelberg, New York, 1991. Springer-Verlag.

[26] D. Talay. Probabilistic numerical methods for partial differential equations: elements of analysis. In D. Talay and L. Tubaro, editors, *Probabilistic Models for Nonlinear Partial Differential Equations*, volume 1627 of *Lecture Notes in Mathematics*, pages 148–196. Springer-Verlag, 1996.

[27] D. Talay and L. Tubaro. Expansion of the global error for numerical schemes solving stochastic differential equations. *Stoch. Anal. Appl.*, 8(4):94–120, 1990.