

Sequential Monte Carlo domain decomposition for the Poisson equation

E. Gobet

CMAP, Ecole Polytechnique
91128 Palaiseau cedex, France
E-mail: emmanuel.gobet@polytechnique.fr

S. Maire

ISITV, Université du sud Toulon Var
Avenue G. Pompidou, BP 56
83362 La Valette du Var cedex, France
E-mail: maire@univ-tln.fr

Abstract - We describe a sequential Monte Carlo version of the domain decomposition method for the Poisson equation. Numerical examples are studied in dimension one and two using piecewise polynomial interpolations for the approximations in each subdomain and on the interface. The pointwise solutions are computed using the Feynman-Kac formula. A geometric reduction of the variance and of the bias of the simulations up to a threshold is observed simultaneously in the subdomains and on the interface.

Keywords— Domain decomposition, sequential Monte Carlo, Poisson equation, variance reduction, Feynman-Kac formula.

I. INTRODUCTION

Our goal is to introduce and study a new domain decomposition Monte Carlo algorithm to solve the Poisson equation in a domain D with Dirichlet boundary conditions. We only focus in this paper on a decomposition in two subdomains. The standard way [PEI 03] to combine a deterministic method and the Monte Carlo method for domain decomposition is the following. The domain D is divided into two subdomains D_1 and D_2 separated by an artificial boundary $\Gamma_{1,2}$. One first computes a Monte Carlo approximation of the solution of the partial differential equation

$$-\frac{1}{2}\Delta u = f$$

in D , with Dirichlet boundary conditions

$$u = g$$

on ∂D , at some points x_i of $\Gamma_{1,2}$ using the Feynman-Kac formula [FRI 76] [FRE 85][LAP 98]

$$u(x) = \mathbf{E}_x(g(B_{\tau_D}) + \int_0^{\tau_D} f(B_s)ds)$$

where B_s is a d -dimensional Brownian motion starting from x . The Monte Carlo computation of this pointwise solution leads to two kinds of numerical errors. The first one

is the discretization error in the simulation of the Brownian motion B_t . This discretization error is $O(\Delta t)$ of the discretization step Δt of the Euler scheme if using a refined approach based on a Brownian bridge [GOB 00][GOB 01]. The walk on spheres method [SAB 91][HWA 03] can also be used. Its accuracy depends on the absorption layer thickness ε which can be taken very small without increasing too much the CPU times. The second kind of error is the standard Monte Carlo error using M simulations that is σ/\sqrt{M} where σ^2 is the variance of $g(B_{\tau_D}) + \int_0^{\tau_D} f(B_s)ds$. Using the approximate values $u^1(x_i)$ of the solution at the points x_i , a global and regular approximation $u_{1,2}^1(x)$ of the solution is built on the boundary $\Gamma_{1,2}$. This can be achieved for example by a simple polynomial interpolation. This approximation $u_{1,2}^1(x)$ is then taken as the new boundary condition on $\Gamma_{1,2}$ for each of the two new problems on D_1 and D_2 . Then one can now solve the equation

$$-\frac{1}{2}\Delta u = f$$

on the domains D_1 and D_2 using a deterministic method like for instance finite differences or finite elements. The problem of this approach is that the error on the interface $\Gamma_{1,2}$ is usually large except if using a lot of trajectories with a small stepsize in the discretization scheme. It has been observed in [PEI 03] that, except in very complex situations, the numerical cost of an accurate approximation on $\Gamma_{1,2}$ is too large compared to a global resolution by means of a deterministic method. In order to compete with standard deterministic methods, we introduce a more sophisticated Monte Carlo algorithm. We first make the description of this algorithm in Section II in quite general situations. Then we study numerical examples in dimension one and two to check the efficiency of our method. We finally make some conclusions and prospects.

II. DESCRIPTION OF THE ALGORITHM

We have introduced in [GOB 04][GOB 05] a sequential Monte Carlo method based on control variates (see also

[MAI 03]) which computes a global spectral approximation of the solution u . This method reduces drastically at a geometric speed of convergence both the Monte Carlo and the discretization error. Nevertheless spectral approximations are less adapted to less smooth solutions and the complexity of our algorithm increases with the degree of the spectral approximation. Piecewise approximations can be more efficient in various situations. Adapting our previous method to piecewise approximations in fact leads to a new domain decomposition method. The algorithm provides a better approximation on the interface $\Gamma_{1,2}$ and in the subdomains as well.

For sake of clarity, we describe our algorithm in the case $D =]-1, 1[\times D'$ where D' is a bounded domain in \mathfrak{R}^{d-1} . The subdomains are $D_1 =]-1, 0[\times D'$ and $D_2 =]0, 1[\times D'$, the interface is $\Gamma_{1,2} = \{0\} \times D'$. The first step is the same than with the standard Monte Carlo method. At some points of the interface we evaluate u using M simulated Feynman-Kac functionals; then a global solution $u_{1,2}^1(x)$ on the interface is obtained through a Tchebychev polynomial interpolation (in dimension $d-1$). Once this approximation has been obtained, we then compute approximations $u_1^1(x)$ and $u_2^1(x)$ on respectively D_1 and D_2 (with $u_{1,2}^1$ as a new boundary condition on $\Gamma_{1,2}$). Actually, pointwise Monte Carlo approximations are computed in each subdomain at Tchebychev grids, from which we derive two approximations $u_1^1(x)$ and $u_2^1(x)$ of the solution on Tchebychev interpolation polynomials in dimension d .

As the global solution u should be regular, the approximations on D_1 and D_2 must satisfy some interface conditions. The deterministic Schur complement method [FAR 94][LET 94] gives a global solution which is continuous and has continuous normal derivatives at the boundary $\Gamma_{1,2}$. The approximations $u_1^1(x)$ and $u_2^1(x)$ do not verify these crucial conditions. To ensure these conditions, we transform these approximations on each subdomain thanks to the error $h(x) = u_1^1(x) - u_2^1(x)$ on $\Gamma_{1,2}$. The new approximations become respectively on D_1 and D_2

$$u_1^1(x) - \frac{g(x_1)\partial_{x_1}h(x)|_{x_1=0} + h(x)|_{x_1=0}}{2},$$

$$u_2^1(x) + \frac{g(x_1)\partial_{x_1}h(x)|_{x_1=0} + h(x)|_{x_1=0}}{2},$$

where the function g verifies $g(0) = 0$ and $g'(0) = 1$. Different functions will be tested in the numerical experiments. We define u^1 by u_1^1 on D_1 and u_2^1 on D_2 . This ends the initialization of the algorithm.

At some points of the interface $\Gamma_{1,2}$, we then compute the correction, which is solution of the new Poisson equation

$$-\frac{1}{2}\Delta y_{1,2}^1 = -\frac{1}{2}\Delta(u - u^1) = f + \frac{1}{2}\Delta u^1 \quad \text{in } D$$

with the boundary conditions

$$y_{1,2}^1 = g - u^1 \quad \text{on } \partial D$$

and let $y_{1,2}^1(x)$ be the relative approximation of this correction using the interpolation process. The next step is to solve on each subdomain the equations

$$-\frac{1}{2}\Delta v_i^1 = -\frac{1}{2}\Delta(u - u_i^1) = f + \frac{1}{2}\Delta u_i^1$$

with the boundary conditions

$$v_i^1 = y_{1,2}^1$$

on the interface $\Gamma_{1,2}$ and

$$v_i^1 = g - u_i^1$$

on the other boundaries of D_i . The new approximation $u^2(x)$ is the regularization of the function

$$u^1(x) + v^1(x)$$

by means of the previous process. One iterates the algorithm until convergence.

III. THE ONE DIMENSIONAL CASE

In this section, we describe the application of our algorithm for the Poisson equation on the domain $D =]-1, 1[$ divided in the subdomains $D_1 =]-1, 0[$ and $D_2 =]0, 1[$. The pointwise solutions are computed by means of an integral representation which avoids discretization errors. The approximations are done using piecewise Tchebychev interpolation polynomials. Finally, a numerical example is studied.

A. Integral representation

The solution of the Poisson equation

$$-\frac{1}{2}u''(x) = f(x), \quad x \in]a, b[$$

with boundary conditions

$$u(a) = c, \quad u(b) = d$$

is

$$u(x) = c \mathbf{P}_x(B_{\tau_D} = a) + d (1 - \mathbf{P}_x(B_{\tau_D} = a)) + \mathbf{E}_x \left(\int_0^{\tau_D} f(B_s) ds \right).$$

As

$$\mathbf{P}_x(B_{\tau_D} = a) = \frac{b-x}{b-a},$$

the contribution of the boundary conditions to the solution can be easily simulated. The contribution to the source term writes

$$\mathbf{E}_x\left(\int_0^{\tau_D} f(B_s) ds\right) = (b-x)(x-a)\mathbf{E}(f(Y_x))$$

where Y_x is a random variable with density

$$\frac{2}{b-a}\left(\frac{r-a}{x-a}1_{a \leq r \leq x} + \frac{b-r}{b-x}1_{x \leq r \leq b}\right).$$

To simulate Y_x , we first define the Bernoulli variable Z_x such that

$$\mathbf{P}(Z_x = 0) = \frac{2}{b-a} \int_a^x \frac{r-a}{x-a} dr = \frac{x-a}{b-a}.$$

Then, we define the random variables A_x and B_x which have for density respectively

$$\frac{2(r-a)}{(x-a)^2}1_{a \leq r \leq x}, \quad \frac{2(b-r)}{(b-x)^2}1_{x \leq r \leq b}.$$

They can be simulated for example on $]a, b[$ respectively by $-1 + (x+1)\sqrt{U}$ and $1 + (x-1)\sqrt{U}$ where U is uniformly distributed on $[0,1]$. We finally set

$$Y_x = (1 - Z_x)A_x + Z_x B_x.$$

B. Tchebychef interpolation on an interval

Tchebychef polynomials $T_n(x) = \cos(n \arccos(x))$ are the orthogonal polynomials on $[-1, 1]$ with respect to the

inner product $\langle P, Q \rangle = \int_{-1}^1 \frac{P(x)Q(x)}{\sqrt{1-x^2}} dx$. The interpola-

tion polynomial $P_N(u)$ of a function u on an interval $]a, b[$ writes

$$P_N(u)(x) = \sum_{k=0}^N \alpha_k T_k\left(\frac{2x-a-b}{b-a}\right)$$

where

$$\alpha_k = \frac{\pi}{\|T_k\|^2 (N+1)} \sum_{i=0}^N u\left(\frac{b-a}{2}x_i + \frac{a+b}{2}\right) T_k(x_i)$$

using the Tchebychef abscissae

$$x_k = \cos\left(\frac{2k+1}{N+1} \frac{\pi}{2}\right), \quad k = 0, 1, \dots, N.$$

If we assume for example that $u \in C^{2m}([-1, 1])$, the coefficients α_k decrease very quickly as $\frac{C}{k^{2m}}$ [BER 97]. We also need to compute the first and the second derivative of $P_N(u)$ for respectively the regularisation at $x = 0$ and the new source terms on each subdomain. We have [CAN 88]

$$(P_N(u))'(x) = \sum_{k=0}^{N-1} \gamma_k T_k\left(\frac{2x-a-b}{b-a}\right)$$

with

$$\gamma_k = \frac{4}{(b-a)} \sum_{p=k+1, p+k \text{ odd}}^N p \alpha_p$$

and

$$(P_N(u))''(x) = \sum_{k=0}^{N-2} \beta_k T_k\left(\frac{2x-a-b}{b-a}\right)$$

with

$$\beta_k = \frac{4}{(b-a)} \sum_{p=k+2, p+k \text{ even}}^N p(p^2 - k^2) \alpha_p.$$

C. Numerical results

We study the Poisson equation

$$-\frac{1}{2}u''(x) = -\frac{\exp(x)}{2}, \quad x \in]-1, 1[$$

with boundary conditions

$$u(-1) = \frac{1}{e}, \quad u(1) = e$$

so that the solution of this equation is $u(x) = \exp(x)$. The interpolation points are denoted by $(y_k)_{0 \leq k \leq N}$ on $]-1, 0[$ and $(z_k)_{0 \leq k \leq N}$ on $]0, 1[$. We denote by

$$e(j) = \max_{0 \leq k \leq N} \left(\max_{0 \leq k \leq N} |u(y_k) - u^j(y_k)|, \max_{0 \leq k \leq N} |u(z_k) - u^j(z_k)| \right)$$

the error in sup-norm as a function of the number j of steps and of the number M of sample values to compute the pointwise approximations. The accuracy of the crude Monte Carlo method with M sample values is given by $e(0)$.

We first study the impact of the regularization on the solution which is the main difference compared to our previous works. As mentioned in Section II, the function g must verify $g(0) = 0$ and $g'(0) = 1$. The most natural choice is obviously $g_1(x) = x$, but the use of this function for regularization perturbs the solutions u_1^j and u_2^j on each subdomains away from the origin. To attenuate this problem, we can use for example the functions

$$g_{2,n}(x) = x(1+x)^n 1_{[-1,0[}(x) + x(1-x)^n 1_{]0,1]}(x)$$

which are close to zero near -1 and 1. If $n+2 \leq N$, these functions also have nice properties in terms of approximation as their restrictions on D_1 and D_2 belong to the same polynomial approximation spaces than u_1^j and u_2^j . In particular the function $(h'(0)g_{2,n}(x) + h(0))/2$ is equal to its interpolation on the grid of each subdomain which makes programming easier. This property is also valid in higher dimensions.

In the following table, we compare the errors $e_1(K)$ and $e_2(K)$ after K steps of the algorithm using respectively the functions g_1 and $g_{2,4}$. We fix for the moment $N = 7$.

| M | $e(0)$ | $e_1(10)$ | $e_1(20)$ | $e_2(10)$ | $e_2(20)$ |
|-----|--------|-----------|-----------|-----------|-----------|
| 300 | 0.05 | 1.2 | 1.5 | 0.03 | 0.002 |
| 500 | 0.07 | 0.2 | 0.05 | 0.01 | 0.001 |
| 700 | 0.07 | 0.01 | 0.0001 | 0.002 | 0.00001 |

We observe as in [GOB 04][GOB 05] a geometric reduction of the error if the number of sample values M is large enough. Convergence is achieved using less sample values with $g_{2,4}$ and is faster as well. Hence the function $g_{2,4}$ is preferable for regularization certainly because it perturbs only slightly the solution away from the interface. We now make some more tests for different values of N using this function for the regularization. We denote by L the number of steps until convergence for a given value of M . We give the CPU time until convergence in seconds.

| N | M | L | $e_2(0)$ | $e_2(L)$ | CPU |
|-----|------|-----|--------------------|--------------------|------|
| 5 | 100 | 23 | 0.1 | 2×10^{-6} | 0.05 |
| 7 | 500 | 47 | 0.1 | 2×10^{-9} | 0.5 |
| 9 | 1000 | 60 | 2×10^{-2} | 10^{-12} | 1.4 |

The error at convergence on the solution at $x = 0$ is of the same order than $e(L)$. The error at convergence $e(L)$ corresponds exactly to the interpolation error of the piecewise interpolation of the exact solution at the interpolation points. We can also make a comparison between these results and results obtained by a global spectral approxima-

tion [GOB 04]. We have for example obtained an accuracy of 6×10^{-10} with $N = 10$ in 0.9 seconds. This is comparable to what we obtain in 0.5 seconds when $N = 7$. This means that even in a situation which is favourable to a global polynomial approximation it is worth considering piecewise polynomial approximations instead. Indeed as observed in [9], the complexity of the algorithm increases quickly with the degree N of the spectral approximation.

IV. THE BIDIMENSIONAL CASE

In this section, we describe the application of our algorithm for the Poisson equation on the domain $D =]-1, 1[\times]0, 1[$ divided in the square subdomains $D_1 =]-1, 0[\times]0, 1[$ and $D_2 =]0, 1[\times]0, 1[$. The pointwise solutions are computed using the modified walk on spheres method. The approximations are done using piecewise Tchebychef tensor product interpolations. We finally give some numerical results to study the impact of the different parameters on the algorithm.

A. Description of the modified walk on spheres

We describe briefly the computation of

$$u(x) = \mathbf{E}_x(g(B_{\tau_D})) + \int_0^{\tau_D} f(B_s) ds,$$

by means of the modified walk on spheres method [HWA 03] which can take into account the source term f . In the original walk on spheres method [SAB 91], the walk goes from x to the boundary ∂D from a sphere to another until the motion reaches the ε -absorption layer. The spheres are built so that the jumps are as large as possible. The radius r_j of the next sphere from a starting point x_j is $d(x_j, \partial D)$. The next point is chosen uniformly on this sphere because of the isotropy of the Brownian motion. The contribution of the boundary conditions to walk i is $g(B_{\tau_D}^i)$. The contribution of the source term at step j of the walk i is simulated by $[r_j^i]^2 f(Y_j^i)/2$ where the explicit law of Y_j^i is obtained using a conditional Green function [HWA 03]. The Monte Carlo approximation of $u(x)$ using M trajectories writes

$$u(x) \simeq \frac{1}{M} \sum_{i=1}^M \sum_{j=1}^{n_i} \frac{[r_j^i]^2}{2} f(Y_j^i) + g(B_{\tau_D}^i).$$

B. Description of the interpolation

The Tchebychef approximation of a function $u \in L^2([a, b] \times [c, d])$ is built using the same process than in dimension one. We give directly the interpolation polynomial $P_N(u)$ at the points of the Tchebychef grid $y_i =$

$\frac{b-a}{2}x_i + \frac{a+b}{2}$ and $z_i = \frac{d-c}{2}x_i + \frac{d+c}{2}$ of this function by

$$P_N(u)(x, y) = \sum_{n=0}^N \sum_{m=0}^N \alpha_{n,m} T_n\left(\frac{2x-a-b}{b-a}\right) T_m\left(\frac{2y-c-d}{d-c}\right)$$

where the $\alpha_{n,m}$ are defined by

$$\frac{\pi^2}{\|T_n\|^2 \|T_m\|^2 (N+1)^2} \sum_{k=0}^N \sum_{j=0}^N u(y_k, z_j) T_n(x_k) T_m(x_j).$$

The quality of this approximation is studied very precisely in [BER 97]. As in dimension one, we also need to compute derivatives of $P_N(u)$ to perform the algorithm. We have

$$\frac{\partial P_N(u)}{\partial x}(x, y) = \sum_{n=0}^N \sum_{m=0}^N \gamma_{n,m} T_n\left(\frac{2x-a-b}{b-a}\right) T_m\left(\frac{2y-c-d}{d-c}\right)$$

with

$$\gamma_{n,m} = \frac{4}{(b-a)} \sum_{p=n+1, p+n \text{ odd}}^N p \alpha_{p,m}$$

and also

$$\Delta P_N(u)(x, y) = \sum_{n=0}^N \sum_{m=0}^N \alpha_{n,m}^{(2)} T_n\left(\frac{2x-a-b}{b-a}\right) T_m\left(\frac{2y-c-d}{d-c}\right)$$

with

$$\alpha_{n,m}^{(2)} = \sum_{p=n+2, p+n \text{ even}}^N \frac{4p(p^2 - n^2)}{(b-a)^2} \alpha_{p,m} + \sum_{p=m+2, p+m \text{ even}}^N \frac{4p(p^2 - m^2)}{(b-a)^2} \alpha_{n,p}.$$

C. Numerical results

We still study a Poisson equation where the solution is very regular. The complexity of the method increases compared to the one-dimensional case. The pointwise solution will have to be computed at $2(N+1)^2$ points instead of $2(N+1)$ and the source terms on each subdomains will be constituted of a $O(N^2)$ terms instead of a $O(N)$. The walk on spheres method is obviously more consuming than the integral representation but it really emphasizes the advantage of dividing the domain in terms of duration of the trajectories. Indeed the computations of the approximations on each subdomain are done using more local trajectories. Only the pointwise solutions on the interface are computed by means of trajectories which move in the whole domain. We consider the equation

$$-\frac{1}{2}\Delta u(x, y) = -\exp(x + y)$$

with Dirichlet boundary conditions chosen so that the solution of this equation is $u(x, y) = \exp(x + y)$. We first take $\varepsilon = 10^{-6}$ to focus on the Monte Carlo error. We use the function $g_{2,4}$ for the regularization process when $N = 5, 7$ and the function $g_{2,6}$ when $N = 9$ which is really more efficient in this case.

| N | M | L | $e(0)$ | $e(L)$ | CPU |
|-----|------|-----|--------|--------------------|-----|
| 5 | 100 | 21 | 0.2 | 2×10^{-6} | 6.1 |
| 7 | 400 | 39 | 0.18 | 2×10^{-9} | 94 |
| 9 | 1000 | 52 | 0.1 | 10^{-12} | 488 |

We achieve an accuracy on the solution of this equation corresponding exactly to the interpolation error. Moreover, we observe that we can obtain an accuracy of 2×10^{-9} when $N = 7$ which is below the discretization error of the walk on spheres. This phenomenon has already been observed and justified in our previous works. In order to reduce the CPU times, we have now diminished ε until this property drops. Taking $\varepsilon = 5 \times 10^{-3}$, we have the following results.

| N | M | L | $e(0)$ | $e(L)$ | CPU |
|-----|------|-----|--------|---------------------|-----|
| 5 | 100 | 21 | 0.18 | 3×10^{-6} | 1.9 |
| 7 | 400 | 39 | 0.05 | 2×10^{-9} | 32 |
| 9 | 1000 | 52 | 0.07 | 2×10^{-12} | 148 |

The numerical accuracy on the solution is similar to the one obtained with $\varepsilon = 10^{-6}$. The CPU times have been divided by about three. The accuracies obtained here are of course out of reach of a crude Monte Carlo method.

V. CONCLUSION AND FUTURE WORK

We have introduced and studied a sequential Monte Carlo version of the domain decomposition method for the Poisson equation. Different numerical tests in dimension one and two have showed that as in [GOB 04] both the variance and the bias reduce geometrically at each step of the algorithm up to a threshold. The main difficulty compared to our previous global spectral method was to handle with the interface conditions. This has been done by a computation of residuals on this interface combined with an appropriate regularization process. There are two main advantages compared to our previous method. First most of the simulations of the Brownian motion are done in smaller domains than the original one which diminishes the simulation times. Second the use of lower degree polynomial interpolations enables to reduce the complexity of the method and hence the number of simulations required for convergence.

Proofs of convergence are in progress using a similar fixed point approach as the one developed in [GOB 05]. Our goal is to extend this method to multidomain decomposition on more complex domain and in higher dimensions. As a consequence different strategies need to be developed for the passages from a domain to an other and for the approximations in each domain as well. Furthermore, as many Monte Carlo methods our algorithm can also take a huge benefit from massive parallel computing. We think that a combination of all these tools can make these sequential methods really competitive compared to deterministic numerical methods for partial differential equations even in low dimensions.

REFERENCES

- [BER 97] BERNARDI C., MADAY Y., Spectral methods, *Handbook of numerical analysis, Vol. V*, Handb. Numer. Anal., V, p. 209–485, North-Holland, Amsterdam, 1997.
- [CAN 88] CANUTO C., HUSSAINI M., QUARTERONI A., ZANG T., *Spectral methods in fluid dynamics*, Springer Series in Computational Physics, Springer-Verlag, New York, 1988.
- [FAR 94] FARHAT C., ROUX F., *Implicit parallel processing in structural mechanics*, *Comput. Mech. Adv.*, vol. 2, n1, p. 124, 1994.
- [FRE 85] FREIDLIN M., *Functional integration and partial differential equations*, Annals of Mathematics Studies - Princeton University Press, 1985.
- [FRI 76] FRIEDMAN A., *Stochastic differential equations and applications. Vol. 2*, New York - San Francisco - London: Academic Press, a subsidiary of Harcourt Brace Jovanovich, Publishers. XIII, 1976.
- [GOB 00] GOBET E., *Euler schemes for the weak approximation of killed diffusion*, *Stochastic Processes and their Applications*, vol. 87, p. 167–197, 2000.
- [GOB 01] GOBET E., *Euler schemes and half-space approximation for the simulation of diffusions in a domain*, *ESAIM: Probability and Statistics*, vol. 5, p. 261–297, 2001.
- [GOB 04] GOBET E., MAIRE S., *A spectral Monte Carlo method for the Poisson equation.*, *Monte Carlo Methods Appl.*, vol. 10, n3-4, p. 275–285, 2004, IVth IMACS Seminar on Monte Carlo Methods MCM-2003 (15-19 September 2003, Berlin).

[GOB 05] GOBET E., MAIRE S., *Sequential control variates for functionals of Markov processes*, *To appear in SIAM Journal on Numerical Analysis*, 2005.

[HWA 03] HWANG C., MASCAGNI M., GIVEN J., *A Feynman-Kac path-integral implementation for Poisson's equation using an h-conditioned Green's function*, *Math. Comput. Simulation*, vol. 62, n3-6, p. 347–355, 2003, 3rd IMACS Seminar on Monte Carlo Methods—MCM 2001 (Salzburg).

[LAP 98] LAPEYRE B., PARDOUX E., SENTIS R., *Méthodes de Monte Carlo pour les processus de transport et de diffusion*, Collection Mathématiques et Applications 29 - Springer Verlag, 1998.

[LET 94] LE TALLEC P., *Domain decomposition methods in computational mechanics*, *Comput. Mech. Adv.*, vol. 1, n2, p. 121–220, 1994.

[MAI 03] MAIRE S., *Reducing variance using iterated control variates*, *The Journal of Statistical Computation and Simulation*, vol. 73, n1, p. 1–29, 2003.

[PEI 03] PEIRANO E., TALAY D., *Domain decomposition by stochastic methods*, *Domain decomposition methods in science and engineering*, p. 131–147 (electronic), Natl. Auton. Univ. Mex., México, 2003.

[SAB 91] SABELFELD K., *Monte Carlo methods in boundary value problems*, Springer Series in Computational Physics, Springer-Verlag, Berlin, 1991, Translated from the Russian.