# Some new simulations schemes for the evaluation of Feynman-Kac representations

Sylvain Maire, Etienne Tanré

HAL Id: inria-00182436

https://inria.hal.science/inria-00182436

Submitted on 26 Oct 2007

# Some new simulations schemes for the evaluation of Feynman-Kac representations

Sylvain Maire[1] and Etienne Tanré[2]

October 25, 2007

**Abstract**

We describe new variants of the Euler scheme and of the walk on spheres method for the Monte Carlo computation of Feynman-Kac representations. We optimize these variants using quantization for both source and boundary terms. Numerical tests are given on basic examples and on Monte Carlo versions of spectral methods for the Poisson equation. We especially introduce a new stochastic spectral formulation with very good properties in terms of conditioning.

**Keywords:** Feynman-Kac formula, Simulation schemes, Quantization, Poisson equation, Sequential Monte Carlo algorithms, Stochastic spectral formulation.

## 1 Introduction

The Feynman-Kac formula is a well-known tool to achieve stochastic representations of the pointwise solution of numerous partial differential equations like diffusion or transport equations [8, 9, 16]. If we consider for example the Dirichlet boundary value problem in a domain $D \subset \mathbb{R}^d$ with a sufficiently smooth boundary $\partial D$

$$-Au = f$$

and boundary conditions

$$u = g$$

on $\partial D$, we have $\forall x \in D$

$$u(x) = \mathbb{E}_x \left[ g(X_{\tau_D}) + \int_0^{\tau_D} f(X_s)ds \right],$$

where $(X_t)_{t \geq 0}$ is a stochastic process solution of the stochastic differential equation relative to the operator $A$ and where $\tau_D$ is the exit time of this process from the domain $D$. The Monte Carlo computation of this pointwise solution leads to two kinds of numerical errors. The first one comes from the discretization error of the stochastic differential equation, which is due to the simulation of the stochastic process $(X_t)_{t \geq 0}$. If we call $\Delta t$ the discretization step, the relative error on the approximate solution in the whole space using the Euler scheme is a $O(\Delta t)$ [2]. In [10, 11], it is proved that one can keep an error of a $O(\Delta t)$ for this Euler scheme in a bounded domain with either Dirichlet or Neumann boundary conditions by using a more sophisticated version based on a Brownian bridge. The second kind of error is the standard Monte Carlo error using $N$ simulations, that is $\frac{\sigma}{\sqrt{N}}$ , where $\sigma^2$ is the variance of

$$g(X^x_{\tau_D}) + \int_0^{\tau_D} f(X^x_s)ds.$$

[1]ISITV, Université de Toulon et du Var, avenue G. Pompidou, BP 56, 83262 La Valette du Var CEDEX, France, `maire@univ-tln.fr`

[2]INRIA, Equipe Projet TOSCA, 2004 route des Lucioles, BP93, 06902 Sophia-Antipolis, France, `Etienne.Tanre@sophia.inria.fr`

The numerical approximation of the source term $\int_0^{\tau_D} f(X_s^x)ds$ is usually done by the rectangle method so that it requires as many evaluations of the function $f$ as they are steps until the absorption by the boundary. On the contrary, the function $g$ is only evaluated once, so there is not a good balance between the evaluations of the source term and of the boundary one. This is even more true when $f$ has a complex expression, when the step size is small and when the variance of $g(X_{\tau_D}^x)$ is greater than the variance of $\int_0^{\tau_D} f(X_s^x)ds$. In the case of the Laplace operator, the process to simulate is the Brownian motion and the walk on spheres method [21] can be used instead of the Euler scheme in order to reduce the number of simulation steps and hence the number of evaluations of $f$. The modified walk on spheres method introduced in [15] for two dimensional problems requires only few evaluations of the function $f$ namely as many as the number of spheres until the process reaches an $\varepsilon$-boundary layer.

In Section 2, we introduce one random step schemes based on either the Euler scheme or on the walk on spheres method where the function $f$ is only evaluated once after the end of the trajectory. The idea is to store the discretization points of the trajectory and to approximate $\int_0^{\tau_D} f(X_s^x)ds$ by the product of its length $\tau_D$ and of the function $f$ at a random point of this trajectory. We can also use the same scheme for the approximation of $\int_0^T f(X_s^x)ds$ in the whole space. In a simple one dimensional case, we give some theoretical reasons to explain why this kind of approach can work. We make some numerical tests to compare these new schemes with the standard ones on different situations depending on the complexity of $f$ and $g$ and of the variance of the source and boundary terms.

In some specific problems like domain decomposition methods [1, 7] or spectral methods [4, 5], the domain and the points where the solution is computed are fixed. In such situations it is possible to optimize our schemes by using quantization techniques [18, 19, 20]. We will do this for the computation of both source and boundary terms. This enables to replace the simulation schemes by quadrature formulae respectively in the domain and on the boundary which have moreover an increased convergence rate than the Monte Carlo method. In Section 3, we describe how to build these formulae numerically for square domains using the Kohonen algorithm and we test them on the numerical examples of the previous sections.

We have introduced sequential Monte Carlo algorithms to compute the solution of the Poisson equation with a great accuracy using either spectral methods [12, 13] or domain decomposition methods [14]. In Section 4, we use the formulae of Section 3 to reduce the simulation times and also the number of steps until convergence of these algorithms on two dimensional problems. Finally in Section 5, we give a new interpretation of the algorithm which leads to direct spectral formulations. We study these new formulations theoretically and numerically and show they have very good properties in terms of conditioning.

# 2 Description of the global one random point method

## 2.1 Simulations based on the Euler scheme

The Euler scheme for the simulation of the Brownian motion starting at $x \in D(\subset \mathbb{R}^d)$ with discretization parameter $\triangle t$ writes

$$B_0 = x, \ B_{n+1} = B_n + \sqrt{\triangle t}Y_n$$

where the $Y_n$ are independent standard Gaussian random variables $\mathcal{N}(0, I_d)$. The crude version in a bounded domain $D$ makes the simulation stops once $B_{n+1} \in D^C$. For example, the approximation of $\tau_D$ is either $n \triangle t$ , $(n + \frac{1}{2}) \triangle t$ or a slightly refined approximation based on the distances $d_n = d(B_n, \partial D)$ and $d_{n+1} = d(B_{n+1}, \partial D)$. In any case, these approximations are of weak order $\sqrt{\triangle t}$ . The main simulation error comes from the possibility for the Brownian motion to leave the domain between step $n$ and $n+1$ and be back into it at time $(n+1) \triangle t$. It is possible to take into account this possibility to obtain a scheme of weak order $\triangle t$ using the half-space approximation [10, 11]. An additional random test is required based on $d_n$ and $d_{n+1}$. Taking a uniform random variable $U_n$, the motion stops if

$$\exp\left(-\frac{2d_n d_{n+1}}{\triangle t}\right) > U_n.$$

If the motion stops between steps $n$ and $n+1$, the approximation of $B_{\tau_D}$ and hence the one of $g(B_{\tau_D})$ is done using a projection based on $B_n$ and $B_{n+1}$. The standard approximation of $\int_0^{\tau_D} f(X_s^x)ds$ by the rectangle method is given by

$$A_1 = \triangle t \sum_{i=1}^{n} f(B_{i\triangle t}).$$

We replace this approximation by $A_2 = n\triangle t f(B_{J\triangle t})$ where $J$ is a discrete uniform random variable on the set $[1..n]$ and we obviously have

$$\mathbb{E}(A_2) = \triangle t \sum_{i=1}^{n} \mathbb{E}(f(B_{i\triangle t}))$$

which proves that the two approximations have same mean value. If we now look at the second order moment of the estimators, we have

$$\mathbb{E}(A_1^2) = (\triangle t)^2 \mathbb{E}(\sum_{i=1}^{n} f(B_{i\triangle t}))^2 \leq n(\triangle t)^2 \sum_{i=1}^{n} \mathbb{E}(f^2(B_{i\triangle t})) = \mathbb{E}(A_2^2)$$

and the same inequality for their variances.

## 2.2   Simulations based on the modified walk on spheres

In order to be faster in the computation of

$$\mathbb{E}_x \left[ g(B_{\tau_D}) + \int_0^{\tau_D} f(B_s)ds \right],$$

we can also use the modified walk on spheres method. In the original walk on spheres method [21], the walk goes from $x$ to the boundary $\partial D$ from a sphere to another until the motion reaches the $\varepsilon$-absorption layer. The spheres are built so that the jumps are as large as possible. The radius of the next sphere from a starting point $x_n$ is $d(x_n, \partial D)$. The next point is chosen uniformly on this sphere because of the isotropy of the Brownian motion. The number of steps until absorption is proportional to $|\log(\varepsilon)|$. To compute the contribution of the source term in this walk for a problem in dimension two, it is proposed in [15] to compute this contribution in each of the balls from the passage to their centers to the boundary, conditioned by the exit point. This is achieved using the Green function conditioned by the exit point $z$ writing

$$\mathbb{E}_{x_n} \left[ \int_0^{\tau_S} f(B_s)ds | B_{\tau_S} = z \right] = \int_S K(z,y)f(y)dy.$$

For the unit ball $B_1$, the cumulative radial distribution is

$$f_R(r) = r^2(1 - 2\log(r))\mathbb{1}_{0 \leq r \leq 1}$$

and the cumulative conditional angular distribution is

$$f_{\theta/R=r}(\theta) = \frac{1}{2} + \frac{1}{\pi} \arctan \left( \frac{1+r}{1-r} \frac{\tan(\theta - \theta_0)}{2} \right) \mathbb{1}_{-\pi \leq \theta \leq \pi}$$

where $\exp(i\theta_0) = z$. For a ball $B_j$ of radius $r_j$ centered at $(x_j, y_j)$, we write

$$\int_{B_j} K(z,y)f(y)dy = \frac{r_j^2}{2} E(f(Y_j))$$

with $Y_j = (r_j R\cos(\theta) + x_j, r_j R\sin(\theta) + y_j)$. The Monte Carlo computation of

$$\mathbb{E}_x \left[ g(B_{\tau_D}) + \int_0^{\tau_D} f(B_s)ds \right]$$

3

is then achieved as

$$\frac{1}{N}\sum_{i=1}^{N} Z_i$$

where

$$Z_i = \sum_{j=1}^{n_i} \frac{r_j^2}{2} f(Y_j^{(i)}) + g(B_{\tau_D}^{(i)}),$$

using the one random point method [6] to approximate $E(f(Y_j))$. This means that only one evaluation of the function $f$ is done for each sphere of the walk. As we did for the Euler scheme, we can reduce this number of evaluations to only one for the whole walk by using the approximation

$$f(Y_J^{(i)}) \sum_{j=1}^{n_i} \frac{r_j^2}{2}$$

of the source term of trajectory $i$ where $J$ is a discrete random variable such that

$$\mathbb{P}(J = j) = \frac{\frac{r_j^2}{2}}{\sum_{j=1}^{n_i} \frac{r_j^2}{2}}.$$

Both approximations of the source term have same obviously mean value since

$$\mathbb{E}\left[f(Y_J^{(i)}) \sum_{j=1}^{n_i} \frac{r_j^2}{2}\right] = \sum_{j=1}^{n_i} \frac{r_j^2}{2} \mathbb{E}\left[f(Y_j^{(i)})\right].$$

In practice, we store along with the walk the locations of the centers of the circles and their radius . By doing this, not only we avoid the evaluation of the function $f$ but also the Monte Carlo simulations of all but one interior point. Hence this method reduces also the simulation times of each walk.

## 2.3   Preliminary example

We intend to compare the variance of the new estimator and of the standard one based on the Euler scheme in the simple situation of the Brownian motion on $\mathbb{R}$. As many functions are easily approximated by polynomials, we look at the respective variances when $f(x) = x^k$ and $\triangle t \to 0$. We describe in detail how we can compute these quantities for $k$ odd even though similar computations are available for $k$ even. The second order moment of our estimator is

$$\mathbb{E}(A_2^2) = \frac{T^2}{n}\mathbb{E}(\sum_{i=1}^{n} B_{i\triangle t}^{2k}) = \frac{T^2}{n}\mathbb{E}(B_1^{2k})\sum_{i=1}^{n}(i\triangle t)^k$$

which converges to

$$T\mathbb{E}(B_1^{2k})\int_0^T s^k ds = \frac{T^{k+2}\mathbb{E}(B_1^{2k})}{k+1} = \frac{T^{k+2}(2k)!}{2^k(k+1)!}$$

as $n$ goes to infinity. The second order moment of the standard estimator is

$$(\Delta t)^2\mathbb{E}(\sum_{i=1}^{n}\sum_{j=1}^{n} B_{i\triangle t}^{k} B_{j\triangle t}^{k}) = (\Delta t)^2\sum_{i=1}^{n}\mathbb{E}(B_{i\triangle t}^{2k}) + 2(\Delta t)^2\sum_{i=1}^{n}\sum_{j=i+1}^{n}\mathbb{E}(B_{i\triangle t}^{k} B_{j\triangle t}^{k}).$$

The first term of the right handside is equal to

$$(\Delta t)^2\mathbb{E}(B_1^{2k})\sum_{i=1}^{n}(i\Delta t)^k$$

and the second one to

$$\frac{2(\Delta t)^{2+k}}{2^k} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{l=0}^{(k-1)/2} \left( \begin{array}{c} 2l+1 \\ k \end{array} \right) i^{k+l}(j-i)^{k-l} \frac{(k+2l+1)!(k-2l-1)!}{(\frac{k+2l+1}{2})!(\frac{k-2l-1}{2})!}.$$

We now study in the following table, the coefficients of the leading terms $(T^{k+2})$ of each of the variances of the two estimators as $\Delta t$ goes to 0.

| k | 1 | 2 | 3 | 5 | 10 | 20 |
|---|---|---|---|---|---|---|
| $V_1 = \lim \text{Var}(A_1^2)$ | 1/3 | 1/3 | 9/5 | 785/14 | $1.27 \times 10^7$ | $1.81 \times 10^{21}$ |
| $V_2 = \lim \text{Var}(A_2^2)$ | 1/2 | 3/4 | 15/4 | 315/2 | $5.95 \times 10^7$ | $1.52 \times 10^{22}$ |
| $V_2/V_1$ | 1.5 | 2.25 | 2.1 | 2.8 | 4,7 | 8.4 |

We observe that the ratio of the variances is increasing from 1.5 when $k = 1$ to 8.4 when $k = 20$. For small values of $k$, say less than 5, the variance of the new estimator is not more than 3 times bigger than the one of the standard one. Hence, this preliminary example shows that the new estimator does not increase too much the variance in the case of polynomial approximations of small degree.

## 2.4   Numerical results

We now compare the new estimators and the standard ones on the numerical resolution of the Poisson equation

$$-\frac{1}{2} \triangle u_m = f_m$$

in the domain $D = ]-1,1[^2$ with boundary conditions

$$u_m = g_m$$

on $\partial D$ where

$$g_m(x,y) = \sum_{i=1}^{m} \exp((1 + \frac{i}{m})(x+y))$$

and

$$f_m(x,y) = -\sum_{i=1}^{m} (1 + \frac{i}{m})^2 \exp((1 + \frac{i}{m})(x+y))$$

are chosen so that we have $\forall x \in D$

$$u_m(x,y) = \mathbb{E}_x \left[ g_m(X_{\tau_D}) + \int_0^{\tau_D} f_m(X_s)ds \right] = \sum_{i=1}^{m} \exp((1 + \frac{i}{m})(x+y)).$$

The parameter $m$ enables to build solutions with different complexities and variances. We focus for the moment on the source term and so we give the variance of this term and the global CPU times at two reference points, one away from the boundary and one close to it, as a function of $m$. All the computations are done on a standard laptop with a 1.66 GHz processor. We begin in the following table by the computation of the solution at the origin using 10000 trajectories of the Euler scheme with stepsize $10^{-3}$ and of the modified walk on spheres method with $\varepsilon$-absorption layer $10^{-4}$. The value of the exact solution at this point is $m$.

If we want to compare numerical methods, the most efficient is the one for which the product $\sigma^2 CPU$ is the smallest. If we look at the variances of the one random step methods compared to the standard ones, we observe that they are only about twice greater. This means that the one random step methods are more efficient than the standard ones if their CPU times are twice smaller. This is always the case when $m$ is

| m | Euler | | One Point Euler | | Wos | | One Point Wos | |
|---|---|---|---|---|---|---|---|---|
| | CPU | $\sigma^2$ | CPU | $\sigma^2$ | CPU | $\sigma^2$ | CPU | $\sigma^2$ |
| 1 | 6 | 22 | 5.4 | 58 | 0.4 | 24 | 0.18 | 51 |
| 10 | 16 | 486 | 5.6 | 1100 | 0.8 | 480 | 0.2 | 910 |
| 100 | 98 | $4.1 \times 10^4$ | 6 | $8.9 \times 10^4$ | 2.7 | $3.9 \times 10^4$ | 0.5 | $7.5 \times 10^4$ |
| 1000 | 940 | $4.3 \times 10^6$ | 9 | $8.8 \times 10^6$ | 23 | $3.8 \times 10^6$ | 3.4 | $7.1 \times 10^6$ |

Table 1: Comparison of methods $(x_0, y_0) = (0, 0)$

greater than 10 in all the numerical examples. The Table 1 shows that the one random step methods are getting more and more competitive when $m$ increases. For example, when $m = 1000$, the standard Euler scheme is about 100 times slower than the new one and the standard walk on spheres is about 7 times slower than the new one. This one random sphere method appears as the best one in all cases (even when $m = 1$). We can also add that the variance of the boundary term is about twice greater than the variance of the source term computed by the modified methods, which is also the case in the next example. We now compute the solution at the point $(-0.9, 0.9)$, which is also equal to $m$, to see what happens if the starting point is close to the boundary.

| m | Euler | | One Point Euler | | Wos | | One Point Wos | |
|---|---|---|---|---|---|---|---|---|
| | Cpu | $\sigma^2$ | Cpu | $\sigma^2$ | CPu | $\sigma^2$ | Cpu | $\sigma^2$ |
| 1 | 0.5 | 0.6 | 0.6 | 1.5 | 0.4 | 0.5 | 0.17 | 0.9 |
| 10 | 1.1 | 15 | 0.7 | 34 | 0.7 | 14 | 0.2 | 23 |
| 100 | 6 | $1.4 \times 10^3$ | 1 | $2.9 \times 10^3$ | 3 | $1.2 \times 10^3$ | 0.5 | $1.9 \times 10^3$ |
| 1000 | 65 | $1.3 \times 10^5$ | 3.7 | $2.8 \times 10^5$ | 17 | $1.2 \times 10^5$ | 2.8 | $2.1 \times 10^5$ |

Table 2: Comparison of methods $(x_0, y_0) = (-0.9, 0.9)$

We observe that the one random step versions of the schemes are not so efficient compared to the standard one in this case. Indeed the variances of these schemes are still twice greater but the number of steps until absorption are a lot smaller. This is especially true for the Euler scheme so that the new version is only really interesting for values of $m$ larger than 100. We can also notice that the variances of all the schemes are a lot smaller for this starting point. This could be used to make a good balance between the simulations if the solution has to be computed at different points ( far or close to the boundary) with the same accuracy. The one random sphere method is however more efficient than all the other methods for every value of $m$ and we can recommend it as it is furthermore less sensitive than the Euler scheme to the decay of its discretization parameter.

# 3 Quantization

## 3.1 General description

We have shown in the previous section that it is possible to reduce drastically the number of evaluations of the source term for only a small increase of the variance. In some situations like spectral methods [4, 5] or in the sequential Monte Carlo methods developed in earlier works [12, 13, 14], the points where the solution is computed are fixed. This means that we can simply replace the simulations by quadrature formulae at some random points of the boundary and of the interior of the domain. In those situations, we can furthermore optimize the locations of the points of evaluations of both $f$ and $g$ by using quantization techniques in order to increase the rate of convergence of these formulae.

Optimal quantization in the quadratic case consists in finding the $M$ points in a domain $D$ minimizing the functional

$$J(M) = \min(\int_D \inf_{1 \le i \le M} d^2(x - x_i)w(x)dx : \{x_1, x_2...x_M \in D\})$$

where $w(x)$ is probability density function on $D$ and $d$ a distance in $D$. This kind of problems is solved numerically using the competitive learning vector quantization algorithm (see [3]) which can be described as follows. First draw $M$ independent points $X_i$ according to the density $w$. Draw one more point $Y_1$ from the same density and find the closest point $X_{min}$ to $Y_1$ among the $X_i$. Move a little $X_{min}$ towards $Y_1$ such that its new location is defined by

$$X_{min} + \varepsilon_1(Y_1 - X_{min})$$

where $\varepsilon_1 > 0$ is a small parameter. The point $Y_1$ is then removed, another point $Y_2$ is drawn, $\varepsilon_2$ replace $\varepsilon_1$ and so on. The sequence $(\varepsilon_n)_n$ is decreasing. A very complete discussion on the numerical aspects of the algorithm is done in [19] in the case of multidimensional Gaussian densities. For problems in the whole space, it is also possible to use another approach based on functional quantization [20] to compute the mean value of similar quantities than the ones computed in this paper.

It was first proposed in [18] to use the points $x_i$ corresponding to the optimal quadratic quantization for numerical integration. For each point $x_i$ a tessel

$$C_i = \{u \in D/d(x_i - u) < d(x_k - u), k \ne i\}$$

is associated. Then the approximation of $\int_D f(x)w(x)dx$ is given by the quadrature formula

$$\sum_{i=1}^n (\int_{C_i} w(x)dx)f(x_i).$$

The weights can be computed for either using Monte Carlo simulations after the end of the algorithm or along with it. These quadrature formulae are more accurate than Monte Carlo integration in rather low dimensions.

## 3.2 Brownian trajectories in a square

We shall now describe how to quantify the points used in our new schemes for the source term and also for the boundary term for the evaluation of Feynman-Kac representations at a given point $(x, y)$ in a fixed bounded domain $D$ in dimension two. Then we will give numerical results on the square domain $[-1, 1]^2$. We first quantify the boundary term

$$\mathbb{E}_{x,y}(g(X_{\tau_D})) = \int_{\partial D} g(s)w^b_{x,y}(s)ds$$

using $q$ points where $w^b_{x,y}(s)$ is the the law of the exit position of the Brownian motion starting at the point $(x, y)$. For this, we only need to define a distance on $\partial D$ which is just in our case the geodesic distance on this set. For the source term, the quantization problem consists in the minimization of

$$J_{x,y,\beta}(p) = \min(\int_0^\infty \int_D \inf_{1 \le i \le M} d^2_\beta(z - z_i)w^s_{x,y}(z)dz : \{z_1, z_2...z_p \in D \times ]0, \infty[)$$

where $w^s_{x,y}(z)$ is the joint law of $(\tau^{x,y}_D, B_{U\tau^{x,y}_D})$ ($U$ is a uniform random variable on $[0, 1]$) and $d^2_\beta(z, z_i) = (x - x_i)^2 + (y - y_i)^2 + \beta(t - t_i)^2$. The parameter $\beta < 1$ is a weight that can be used to focus on the position of the interior points. To compute the quantization points, we have used the Euler scheme with a discretization parameter $\Delta t = 10^{-3}$ and the additional test for each of the two optimization problems. We have chosen in all the numerical examples presented here $\beta = 0.5$, $p = q$. The parameter $\varepsilon(n)$ is defined in two parts in order to allow the points to move from a side of the square to the closer one in the beginning of the

iterations. We first choose a very slow decay $\varepsilon(n) = \frac{C}{\log(\log(5+n))}$ for half of the iterations and then a faster one $\varepsilon(n) = \frac{C_1}{n}$ for the second half. After convergence, we obtain an approximation of the solution at a given point $(x, y)$

$$u(x,y) \simeq \sum_{i=1}^{q} \left( \int_{C_i} w_{x,y}^b(s)ds \right) g(x_i, y_i) + \sum_{i=1}^{p} \left( \int_{C_i} w_{x,y}^s(z)dz \right) t_i f(x_i, y_j)$$

which can be written as a quadrature formula of the form

$$u(x,y) \simeq \sum_{i=1}^{q} a_i g(x_i^b, y_i^b) + \sum_{j=1}^{p} b_j f(x_j^s, y_j^s).$$

We plot on the next figures the quantization points for the boundary and source terms for respectively $(x_0, y_0) = (0, 0)$ and $(-0.9, 0.9)$.
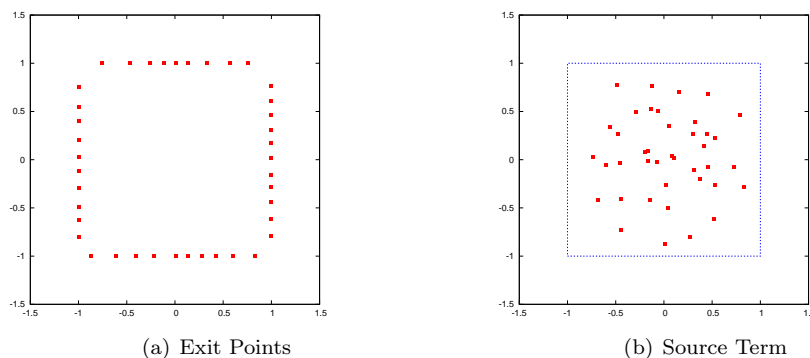


(a) Exit Points          (b) Source Term

Figure 1: Quantization points $(x_0, y_0) = (0, 0)$



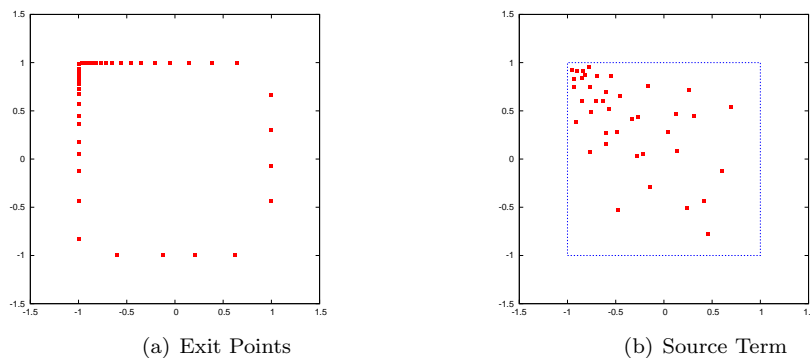(a) Exit Points          (b) Source Term

Figure 2: Quantization points $(x_0, y_0) = (-0.9, 0.9)$

We see on Figure 1 that we have not obtained the optimal quantization points: the symmetry of the problem is not kept by our points. However, the numerical results obtained with these points have a good accuracy.

## 3.3 Numerical results

The aim of this section is to compare these quadrature formulae with the one random sphere method on the previous example and on a slightly different one. We begin with the application of the one random sphere method with $N$ simulations and $\varepsilon = 10^{-6}$ at the two reference points of the example of Section 2. We obtain the Table 3.

| | $N$ | 100 | 400 | 1000 | 10000 | 100000 | 1000000 |
|---|---|---|---|---|---|---|---|
| $(x_0, y_0) = (0, 0)$ | $m = 1$ | 0.101 | -0.1 | 0.48 | 0.867 | 0.998 | 1.012 |
| | $m = 10$ | 6.11 | 4.89 | 7.55 | 9.29 | 9.996 | 10.046 |
| $(x_0, y_0) = (-0.9, 0.9)$ | $m = 1$ | 1.029 | 0.939 | 0.963 | 0.996 | 0.998 | 0.9996 |
| | $m = 10$ | 10.22 | 9.65 | 9.746 | 9.95 | 9.995 | 10.001 |

Table 3: Monte Carlo Procedure

We can see that the computation of the solution at the point $(0, 0)$ is very hard. At least 10000 simulations are required to obtain an acceptable relative error of about 10 percent and $10^6$ simulations lead to a relative error of about one percent. The computations are more accurate at the point (-0.9,0.9) and the relative error is 0.03 using 1000 simulations. We now compare with the quantization points (Table 4).

| | $N$ | 20 | 30 | 40 | 60 | 80 | 100 |
|---|---|---|---|---|---|---|---|
| $(x_0, y_0) = (0, 0)$ | $m = 1$ | 1.52 | 1.13 | 1.16 | 1.35 | 1.12 | 1.20 |
| | $m = 10$ | 11.95 | 10.47 | 10.57 | 11.28 | 10.51 | 10.78 |
| $(x_0, y_0) = (-0.9, 0.9)$ | $m = 1$ | 1.01 | 1.02 | 1.004 | 1.002 | 1.008 | 1.007 |
| | $m = 10$ | 10.04 | 10.08 | 10.01 | 10.005 | 10.04 | 10.03 |

Table 4: Quantization Method

We remark that we obtain with no more than 100 points the same accuracy than with 10000 random points on the computation at the point $(0, 0)$. For the point $(-0.9, 0.9)$, 30 or 40 points are sufficient to obtain a relative error of 0.02. This confirms the theoretical speed of convergence of the two methods. Nevertheless, we can notice that the accuracy obtained with 100 quantization points is not clearly better than the one obtained with fewer points (40 for example). This first example was quite hard to solve numerically because the variances were large. We would like to see the accuracy of the estimators on an example with a smaller variance. To do this, we just choose

$$v_m(x, y) = \sum_{i=1}^{m} \exp(\frac{i}{m}(x + y))$$

as the solution of our Poisson equation. We first use the random points (Table 5).

| | $N$ | 100 | 400 | 1000 | 10000 | 100000 | 1000000 |
|---|---|---|---|---|---|---|---|
| $(x_0, y_0) = (0, 0)$ | $m = 1$ | 0.899 | 0.863 | 0.939 | 0.977 | 1.0004 | 1.0004 |
| | $m = 10$ | 9.453 | 9.437 | 9.753 | 9.901 | 10.006 | 10.004 |
| $(x_0, y_0) = (-0.9, 0.9)$ | $m = 1$ | 1.016 | 0.986 | 0.987 | 0.996 | 1.0002 | 1.0003 |
| | $m = 10$ | 10.10 | 9.941 | 9.923 | 9.976 | 10.002 | 10.002 |

Table 5: Monte Carlo Procedure

We now obtain a relative error of 2 digits with 10000 simulations for the point $(0, 0)$ and with 1000 simulations for the point $(-0.9, 0, 9)$. If we compare to the quantization points of the table 6, we obtain the same accuracies with only 30 quantization points.

| | $N$ | 20 | 30 | 40 | 60 | 80 | 100 |
|---|---|---|---|---|---|---|---|
| $(x_0, y_0) = (0,0)$ | $m = 1$ | 1.02 | 1.001 | 1.002 | 1.01 | 1.007 | 1.008 |
| | $m = 10$ | 10.04 | 0.995 | 0.994 | 10.02 | 10.02 | 10.02 |
| $(x_0, y_0) = (-0.9, 0.9)$ | $m = 1$ | 0.9997 | 1.001 | 0.9996 | 0.9998 | 1.0006 | 1.0001 |
| | $m = 10$ | 9.998 | 10.003 | 9.998 | 9.999 | 10.002 | 9.9999 |

Table 6: Quantization Method

We can conclude that the quantization techniques are really efficient in the computation of the Feynman-Kac representations. This is especially true when the variance is small: we can obtain an accuracy of 4 digits using only 100 points in the last example. Nevertheless, the search of the optimal points is costly and not always reliable as it depends of many parameters. Quantization is hence recommandable when the points where the solution has to be computed are fixed and when there are not too many quantization points. We give in the next two sections examples of such situations.

# 4    Sequential Monte Carlo applications

## 4.1    Description of the algorithm

We describe briefly on the Poisson equation in a domain $D$ an iterative Monte Carlo method introduced in [12, 13] which can be used to compute a global approximate solution of many linear elliptic or parabolic partial differential equations. The idea is to use the approximation of the solution at a given step as a control variate for the next step. We first compute approximate values $u_i^{(1)}$ of the solution of this equation at $N$ points $x_i$ using a Monte Carlo method to approximate the Feynman-Kac representation. We denote by $M$ the number of drawings and by $\Delta t$ the discretization parameter of the simulation scheme of the Brownian motion. Using this information, we can then build a global and regular linear approximation $u^{(1)}(x)$ of the Poisson equation by for instance a simple interpolation. We now use the control variate method with $u^{(1)}$ as an approximation of $u$. We let $\forall x \in D$

$$r(x) = u(x) - u^{(1)}(x)$$

and we now have to solve $\forall x \in D$ the Poisson equation

$$-\frac{1}{2}\triangle r = -\frac{1}{2}\triangle(u - u^{(1)}) = f + \frac{1}{2}\triangle u^{(1)}$$

with boundary conditions

$$r = g - u^{(1)}.$$

A global solution $r^{(1)}$ is computed using the same method that we have used to compute $u^{(1)}$. Then, we approximate the solution of the initial equation by

$$u^{(2)}(x) = u^{(1)}(x) + r^{(1)}(x)$$

and we can expect that this solution is more accurate than the previous one. We iterate this method to achieve an approximation $u^{(n)}(x)$ at the $n$th step of the algorithm. We are really in a situation where it is important to have an efficient Monte Carlo scheme to approximate the Feynman-Kac representations: both boundary and source terms have a complex expression and the solution has to be computed many times at many points of the domain $D$. We now recall the main hypotheses and convergence results of the algorithm which are described with more details in [13]. We assume that the approximation of the solution $u$ can be written in a linear form

$$Pu(x) = \sum_{j=1}^{N} u(x_j)\Psi_j(x)$$

for some functions $\Psi_j(x)$. Some weak assumptions are also required on the simulation scheme of the Brownian motion with a discretization parameter $\Delta t$. The algorithm is stochastic and biased due to the simulation scheme. Hence we define the quantities

$$m_n = \max_{1 \leq i \leq N} \left| E(u^{(n)}(x_i) - u(x_i)) \right|, v_n = \max_{1 \leq i \leq N} \mathrm{Var}(u^{(n)}(x_i))$$

to study its convergence. To study the influence of the simulation scheme, we consider the difference of the solution of the Poisson equations, with $g$ as both source term and boundary condition, between respectively the discretized and the continuous one. Then $e(g, \Delta t, x)$ and $V(g, \Delta t, x)$ are respectively the mean value and the variance of the previous quantity. We first state the convergence result for the bias.

**Theorem 4.1.** *For any $n \geq 1$, we have*

$$m_n \leq \rho_m m_{n-1} + \max_{1 \leq i \leq N} |[P(u) - u](x_i) + P[e(u - Pu, \Delta t, .)](x_i)|$$

*where $\rho_m = \max_{1 \leq i \leq N} [\sum_{j=1}^{N} |P[e(\Psi_j(, \Delta t, .)](x_i)|]$. If $\Delta t$ is small enough, then $\rho_m < 1$ and $m_n$ converges at a geometric rate up to a threshold equal to*

$$\limsup m_n \leq \frac{1}{1 - \rho_m} \max_{1 \leq i \leq N} |[P(u) - u](x_i) + P[e(u - Pu, \Delta t, .)](x_i)|.$$

This theorem shows that even if the simulations are biased the upper limit on the bias depends mainly of the quality of the approximation $P(u) - u$. We now state the convergence of the variance $v_n$.

**Theorem 4.2.** *Setting*

$$C(\Delta t, N) = 2 \max_{1 \leq i \leq N} \sum_{j=1}^{N} \Psi_j^2(x_i) [\sum_{k=1}^{N} \sqrt{V(\Psi_k, \Delta t, x_j)}]^2,$$

$$\rho_v = \max_{1 \leq i \leq N} (\sum_{j=1}^{N} |P[e(\Psi_j(, \Delta t, .)](x_i)|)^2 + \frac{C(\Delta t, N)}{M},$$

*then one has for any $n \geq 1$*

$$v_n \leq \rho_v v_{n-1} + \frac{1}{M} 2 \max_{1 \leq i \leq N} \sum_{j=1}^{N} \Psi_j^2(x_i) V(u - Pu, \Delta t, x_j) + C(\triangle, N) m_{n-1}.$$

*If $\Delta t$ is small enough and $M$ large enough, then $\rho_v < 1$ and $v_n$ converges at a geometric rate up to a threshold equal to*

$$\limsup m_n \leq \frac{1}{(1 - \rho_v)M} (2 \max_{1 \leq i \leq N} \sum_{j=1}^{N} \Psi_j^2(x_i) V(u - Pu, \Delta t, x_j) + C(\Delta t, N) \limsup m_n^2).$$

Note that when $\rho_v < 1$ and $\rho_m < 1$ (which is always true for $\Delta t$ small enough and $M$ large enough) the convergence holds for both the bias and the variance. We now describe a practical example of this algorithm based on spectral approximations on a square domain.

## 4.2 The two dimensional test case

We consider the Poisson equation on the square domain $D = [-1, 1]^2$

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

11

with Dirichlet boundary conditions chosen so that the solution of this equation is $u(x,y) = \exp(x+y)$. To approximate the solution, we use the interpolation polynomial

$$P_N(u) = \sum_{n=0}^{N} \sum_{m=0}^{N} \alpha_{n,m} T_n(x) T_m(y)$$

of the function $u$ at the Tchebychef grid, where the $\alpha_{n,m}$ are defined by

$$\alpha_{n,m} = \frac{\pi^2}{\|T_n\|^2_{L^2_w} \|T_m\|^2_{L^2_w} (N+1)^2} \sum_{k=0}^{N} \sum_{j=0}^{N} u(x_k, y_j) T_n(x_k) T_m(y_j)$$

with

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

The quality of this approximation is studied very precisely in [4]. To build the source terms which appear in the algorithm, we need to compute $\triangle P_N(u)$. We have [5]

$$\triangle P_N(u) = \sum_{n=0}^{N} \sum_{m=0}^{N} \alpha_{n,m}^{(2)} T_n(x) T_m(y)$$

with

$$\alpha_{n,m}^{(2)} = \frac{1}{c_n} \sum_{p=n+2, p+n \; even}^{N} p(p^2 - n^2)\alpha_{p,m} + \frac{1}{c_m} \sum_{p=m+2, p+m \; even}^{N} p(p^2 - m^2)\alpha_{n,p}$$

where the normalization coefficient $c_k$ is such that $c_0 = 2$ and $c_k = 1$ if $k \neq 0$.

## 4.3 Numerical results

We first consider the modified walk on spheres method with an absorption boundary layer $\varepsilon = 10^{-2}$ which leads to the following results. As shown in [12], the numerical accuracy on the solution would be exactly the same with a smaller $\varepsilon$. The number $M$ is chosen large enough so that the convergence happens. We define $L$ as the number of steps until convergence and $err(L)$ as the maximum of the absolute errors at each of the grid points. The results are already very accurate and the method quite fast (see Table 7). We now use the

| $N$ | $M$ | $L$ | $err(L)$ | CPU |
|---|---|---|---|---|
| 5 | 200 | 8 | $5 \times 10^{-5}$ | 1.1 |
| 7 | 600 | 13 | $3 \times 10^{-7}$ | 9 |
| 9 | 1500 | 15 | $8 \times 10^{-10}$ | 47 |

Table 7: Modified Walk on Spheres

one random sphere method with the same boundary-layer.

| $N$ | $M$ | $L$ | $err(L)$ | CPU |
|---|---|---|---|---|
| 5 | 200 | 15 | $7 \times 10^{-5}$ | 0.7 |
| 7 | 600 | 27 | $7 \times 10^{-7}$ | 6 |
| 9 | 1500 | 41 | $2 \times 10^{-9}$ | 42 |

Table 8: One Random Sphere Method

We can notice that there is not a really significant improvement between the new method and the old one. In the examples of Section 2.4, the variance of the source term was lower than the one of the boundary term. In this problem, the source term is obtained by the computation of second derivatives of an interpolation polynomial with random coefficients. Hence the variance of the source term is a lot bigger than the one of the boundary term. We can nevertheless improve this new approach by replacing the simulations by random quadrature formulae. For each of the grid points, the idea is to store in a file $2M$ simulations of the boundary term, of the source term and of the exit time. Then for each grid point, we pick at random either the first $M$ elements or the last $M$ elements of the file to build the quadrature formulae.

| $N$ | $M$ | $L$ | $err(L)$ | CPU |
|---|---|---|---|---|
| 5 | 200 | 17 | $1 \times 10^{-4}$ | 0.2 |
| 7 | 600 | 29 | $8 \times 10^{-7}$ | 1.6 |
| 9 | 1500 | 38 | $2 \times 10^{-9}$ | 12 |

Table 9: Random Quadrature

The CPU times have been divided by 4 as now there are no more simulations. We test the same method with quantization points using two files containing respectively 40 and 41 points for $N = 5, 7$ and respectively 120 and 122 points for $N = 9$. The idea is to improve the accuracy of the corrections at each step of the algorithm by using a method with an higher convergence rate than the Monte Carlo method. It has been already tested successfully using quasi Monte Carlo sequences in [12] and also for numerical integration in [17].

| $N$ | $M$ | $L$ | $err(L)$ | CPU |
|---|---|---|---|---|
| 5 | 40 | 6 | $2 \times 10^{-4}$ | 0.02 |
| 7 | 40 | 27 | $1 \times 10^{-6}$ | 0.21 |
| 9 | 120 | 47 | $5 \times 10^{-9}$ | 1.8 |

Table 10: Quantified Quadrature

Very small numbers of quantization points are sufficient to ensure the convergence. We observe that the CPU times have been divided by 8 compared to the previous method and hence by a factor about 40 compared to the original one. The error at convergence is nevertheless slightly bigger than the ones of the previous methods because the threshold in Theorem 4.2 is also slightly bigger. This sequential quantified algorithm is hence a great improvement of the original sequential Monte Carlo one.

## 5 A Direct stochastic spectral formulation

### 5.1 The stochastic spectral formulation

Instead of using randomized quantization grids at each step of the algorithm, we have tried to use only one to see the impact on the algorithm, even though no more independence properties remain. We have noticed that there is still a decay of the error but at a lot more slow rate until a threshold which is also greater than the previous one. This version of the algorithm is clearly worse than the other one for practical computations but we can try to make its interpretation. The threshold does not change at convergence and this means that the value of the residual at each of the interpolation points is equal to zero. The approximation $u_N(x, y)$ of the solution of the Poisson equation writes

$$u_N(x, y) = \sum_{n=0}^{N} \sum_{m=0}^{N} \alpha_{n,m} T_n(x) T_m(y)$$

where the coefficients must satisfy the previous property at each point of the interpolation grid. The approximation at a grid point $(x_i, y_j)$ of the residual $r(x_i, y_j) = u(x_i, y_j) - u_N(x_i, y_j)$ writes

$$\sum_{k=1}^{p} \gamma_{i,j,k}(f(x_{i,j,k}^s, y_{i,j,k}^s) + \frac{1}{2}\sum_{n=0}^{N}\sum_{m=0}^{N}\alpha_{n,m}(T_n^{''}(x_{i,j,k}^s)T_m(y_{i,j,k}^s) + T_n(x_{i,j,k}^s)T_m^{''}(y_{i,j,k}^s))$$

$$+ \sum_{l=1}^{q}\beta_{i,j,l}(g(x_{i,j,l}^b, y_{i,j,l}^b) - \sum_{n=0}^{N}\sum_{m=0}^{N}\alpha_{n,m}T_n(x_{i,j,l}^b)T_m(y_{i,j,l}^b))$$

where the quadrature points and the relative weights are built either using the one random sphere method or the quantization method of the previous sections. The exit points of the Brownian motion starting at $(x_i, y_j)$ are for instance $(x_{i,j,1}^b, y_{i,j,1}^b), \cdots, (x_{i,j,p}^b, y_{i,j,p}^b)$. We are looking for the coefficients $\alpha_{n,m}$ such that this approximation of the residual is equal to zero at each of the grid points. This leads to the linear system

$$\sum_{n=0}^{N}\sum_{m=0}^{N}a_{n,m}\alpha_{n,m} = \sum_{l=1}^{q}\beta_{i,j,l}g(x_{i,j,l}^b, y_{i,j,l}^b) + \sum_{k=1}^{p}\gamma_{i,j,k}f(x_{i,j,k}^s, y_{i,j,k}^s)$$

where the coefficients $a_{n,m}$ are equal to

$$\frac{-1}{2}\sum_{k=1}^{p}\gamma_{i,j,k}(T_n^{''}(x_{i,j,k}^s)T_m(y_{i,j,k}^s) + T_n(x_{i,j,k}^s)T_m^{''}(y_{i,j,k}^s)) + \sum_{l=1}^{q}\beta_{i,j,l}T_n(x_{i,j,l}^b)T_m(y_{i,j,l}^b).$$

The quality of the resolution and the speed of convergence of iterative methods depend of the condition number of the linear system. We can first look at the asymptotic system that is when $p$ and $q$ go to $+\infty$. Each limit term is the solution at the grid point $(x_i, y_j)$ of the Poisson equation with source term

$$-\frac{1}{2}(T_n^{''}(x)T_m(y) + T_n(x)T_m^{''}(y))$$

and boundary term

$$T_n(x)T_m(y)$$

that is

$$T_n(x_i)T_m(y_j).$$

We have computed numerically the condition number of this asymptotic system which seems to be a $O(N^2)$. We will give some more numerical results in Section 5.3 on a more efficient formulation but we first need to compare our method to deterministic ones.

## 5.2 Optimization and comparison with standard spectral methods.

This formulation is quite similar to a deterministic collocation method described for example in [4] page 102. In this formulation, the bases functions are the Lagrange polynomials instead of the Tchebychef ones and the grid points are built using the Tchebychef Gauss-Lobatto points, that is the zeros $z_j = \cos(\frac{(N-j)\pi}{N})$, $0 \le j \le N$ of $(1-x^2)T_N^{'}(x)$. The solution is approximated by

$$V_N(x, y) = \sum_{n=0}^{N}\sum_{m=0}^{N}u_{n,m}l_n(x)l_m(y)$$

where $l_n$ is the Lagrange polynomial associated to $z_n$ and $u_{n,m}$ is the value of the solution at the point $(z_n, z_m)$. This formulation writes

$$-\frac{1}{2}\sum_{n=1}^{N-1}\sum_{m=1}^{N-1}b_{n,m}u_{n,m} = f(z_s, z_r) + \sum_{(z_n, z_m)\in B}g(z_n, z_m)(l_n^{''}(z_s)l_m(z_r) + l_n(z_s)l_m^{''}(z_r))$$

14

where $B$ is the set of the boundary points and where

$$b_{m,n} = l_n''(z_s)l_m(z_r) + l_n(z_s)l_m''(z_r).$$

It is proved in [4] that the condition number of the system is a $O(N^4)$ in the slightly different case of the Legendre Gauss-Lobatto points. Moreover, this system is not symmetric which may be an additional drawback with respect to the standard collocation system which is symmetric and whose condition number is a $O(N^3)$. The condition of our asymptotic system seems to be a $O(N^2)$ which is already better. But we can still diminish this asymptotic condition number by just replacing in our formulation the bases functions which are Tchebychef polynomials by Lagrange interpolation polynomials. Indeed, if we write that the approximation of the solution is

$$W_N(x, y) = \sum_{n=0}^{N} \sum_{m=0}^{N} v_{n,m} l_n(x) l_m(y),$$

then the coefficients $c_{m,n}$ of the relative linear system are equal to

$$-\frac{1}{2} \sum_{k=1}^{p} \gamma_{i,j,k}(l_n''(x_{i,j,k}^s)l_m(y_{i,j,k}^s) + l_n(x_{i,j,k}^s)l_m''(y_{i,j,k}^s)) + \sum_{l=1}^{q} \beta_{i,j,l} l_n(x_{i,j,l}^b) l_m(y_{i,j,l}^b)$$

which converge to

$$l_n(x_i)l_m(y_j) = \delta_{i,n}\delta_{j,m}$$

when $p$ and $q$ go to $+\infty$. This means that the matrix of the spectral formulation converges toward the identity matrix and hence its condition number to one. Furthermore, we can easily use for the numerical resolution of our non-symmetric system a very simple iterative method like the Jacobi method.

## 5.3   Numerical results

We have first studied the properties of this last formulation obtained respectively with the random points and the quantization points in the case $N = 3$. We compute the solution at the 16 points of the Tchebychef grid using either 1000 random simulations or 40 quantization points for both the source term and the boundary one. The corresponding condition number $\kappa(A)$ are respectively 1,31 and 1,11. Note that because of symmetries, we need to compute the quantization points at only few points of the Tchebychef grid. The spectral radius $\rho(J)$ of the iteration matrices of the Jacobi method are respectively 0,101 and 0,05. This confirms the expected properties of the iteration matrices and the efficiency of the quantization techniques compared to Monte Carlo simulations. We give some more detailed results on the example of the previous section in particular the error $err$ using only the quantization points.

| $N$ | $M$ | $\kappa(A)$ | $\rho(J)$ | $err$ |
|---|---|---|---|---|
| 5 | 40 | 1.8 | 0.23 | $2 \times 10^{-4}$ |
| 7 | 40 | 16.7 | 0.77 | $1 \times 10^{-6}$ |
| 9 | 120 | 15.4 | 0.5 | $6 \times 10^{-9}$ |

Table 11: Properties of the spectral matrix

These numerical results were obtained with Matlab. Most of the CPU time is spent building the matrix $A$. The accuracy on the solution is the same than the one obtained with the sequential method of the previous section. The condition numbers are very small compared to $N^4$. The only drawback compared to a deterministic method is that the source term and the boundary term need to be evaluate at more points (not only at the grid points). Hence one has to find a good balance in the choice of $M$ between this drawback and the big advantage of a small condition number.

15

# 6    Conclusion

We have developed new methods for the numerical computations of Feynman-Kac representations of the solution of the Poisson equation in a bounded domain. First we have introduced one random step schemes for the evaluation of the source term of these representations. The one random sphere method has appeared very efficient compared to the standard methods on all our numerical tests. We have then used quantizations techniques to optimize the numerical computations of both source term and boundary term. Even though this optimization was costly, the numerical results were really impressive compared to those obtained using Monte Carlo estimators. In the case where the domain and the points where the solution is computed are fixed, this optimization can be anyhow considered as preprocessing. We have finally given some examples on global resolution of Poisson equations using either sequential Monte Carlo methods or a new stochastic spectral formulation. The sequential Monte Carlo algorithm has been improved by a factor 40 in terms of CPU times by using the quantization points. The stochastic spectral formulation introduced has very good properties in terms of both its condition number and the spectral radius of its relative Jacobi iteration matrix. These two new hybrid methods are really promising to solve the Poisson equation compared to fully deterministic methods. We should now make really comparisons in terms of complexity between these methods and deterministic spectral methods and also make the extension of our method to problems in higher dimensions and to more complex domains or operators.

# References

[1] J. A. Acebrón, M. P. Busico, P. Lanucara, and Renato Spigler. Domain decomposition solution of elliptic boundary-value problems via Monte Carlo and quasi-Monte Carlo methods. *SIAM J. Sci. Comput.*, 27(2):440–457 (electronic), 2005.

[2] V. Bally and D. Talay. The law of the Euler scheme for stochastic differential equations. I. Convergence rate of the distribution function. *Probab. Theory Related Fields*, 104(1):43–60, 1996.

[3] A. Benveniste, M. Métivier, and P. Priouret. *Adaptive algorithms and stochastic approximations*, volume 22 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, 1990. Translated from the French by Stephen S. Wilson.

[4] C. Bernardi and Y. Maday. *Approximations spectrales de problèmes aux limites elliptiques*, volume 10 of *Mathématiques & Applications (Berlin) [Mathematics & Applications]*. Springer-Verlag, Paris, 1992.

[5] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral methods in fluid dynamics*. Springer Series in Computational Physics. Springer-Verlag, New York, 1988.

[6] J. M. DeLaurentis and L. A. Romero. A Monte Carlo method for Poisson's equation. *J. Comput. Phys.*, 90(1):123–140, 1990.

[7] C. Farhat and F.-X. Roux. Implicit parallel processing in structural mechanics. *Comput. Mech. Adv.*, 2(1):124, 1994.

[8] M. Freidlin. *Functional integration and partial differential equations*, volume 109 of *Annals of Mathematics Studies*. Princeton University Press, Princeton, NJ, 1985.

[9] A. Friedman. *Stochastic differential equations and applications. Vol. 2*. Academic Press [Harcourt Brace Jovanovich Publishers], New York, 1976. Probability and Mathematical Statistics, Vol. 28.

[10] E. Gobet. Weak approximation of killed diffusion using Euler schemes. *Stochastic Process. Appl.*, 87(2):167–197, 2000.

[11] E. Gobet. Euler schemes and half-space approximation for the simulation of diffusion in a domain. *ESAIM Probab. Statist.*, 5:261–297 (electronic), 2001.

[12] E. Gobet and S. Maire. A spectral Monte Carlo method for the Poisson equation. *Monte Carlo Methods Appl.*, 10(3-4):275–285, 2004.

[13] E. Gobet and S. Maire. Sequential control variates for functionals of Markov processes. *SIAM J. Numer. Anal.*, 43(3):1256–1275 (electronic), 2005.

[14] E. Gobet and S. Maire. Sequential monte carlo domain decomposition for the poisson equation. 2005. Proceedings of the 17th IMACS World Congress, Scientific Computation, Applied Mathematics and Simulation (11-15 July 2005, Paris).

[15] C.-O. Hwang, M. Mascagni, and J. A. Given. A Feynman-Kac path-integral implementation for Poisson's equation using an $h$-conditioned Green's function. *Math. Comput. Simulation*, 62(3-6):347–355, 2003. 3rd IMACS Seminar on Monte Carlo Methods—MCM 2001 (Salzburg).

[16] B. Lapeyre, É. Pardoux, and R. Sentis. *Méthodes de Monte-Carlo pour les équations de transport et de diffusion*, volume 29 of *Mathématiques & Applications (Berlin) [Mathematics & Applications]*. Springer-Verlag, Berlin, 1998.

[17] S. Maire. Polynomial approximations of multivariate smooth functions from quasi-random data. *Stat. Comput.*, 14(4):333–336, 2004.

[18] G. Pagès. A space vector quantization method for numerical integration. *J. Computational and Applied Mathematics*, 89:1–38, 1998.

[19] G. Pagès and J. Printems. Optimal quadratic quantization for numerics: the Gaussian case. *Monte Carlo Methods Appl.*, 9(2):135–165, 2003.

[20] G. Pagès and J. Printems. Functional quantization for numerics with an application to option pricing. *Monte Carlo Methods Appl.*, 11(4):407–446, 2005.

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