Monte Carlo Algorithms

Some simple numerical examples for performing *grid* and *grid-free* Monte Carlo algorithms will now be considered.

Let the operator L in the equation (86) be the Laplacian:

$$L = \Delta.$$

Using a regular discretisation with a step–size h equation (86) is approximated by the following difference equation

$$\Delta_h^{(d)} u = -f_h. \tag{111}$$

Assume that (111) is solved for the i^{th} point $i = (i_1, \ldots, i_d)$:

$$u_i = L_h u + \frac{h^2}{2d} f_i,$$

where $\Delta_h^{(d)}$ is the Laplace difference operator, and L_h is an averaging operator. For example, the operator L_h in $\mathbb{R}^2$ is

$$L_h u = \frac{1}{4}[u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}] = \frac{1}{4}\Lambda_1(i,j)$$

and then (111) becomes

$$u_{ij} = \frac{1}{4}\Lambda_1(i,j) + \frac{h^2}{4} f_{i,j}. \tag{112}$$

The matrix form of equation (112) has only $2d$ non-zero elements in each row and they all are equal to $\frac{1}{2d}$.

The *grid* Monte Carlo algorithm for solving (112) consists in simulating a Markov chain with initial density p_0 which is *permissible* to the vector h. The probability $p_{\alpha\beta}$ for the transition from the point α to the next point β in our case is equal to $\frac{1}{2d}$ if the point is inside of the domain, that is $((x_{(1)})_\alpha, (x_{(2)})_\beta) \in \Omega_h$, and $p_{\alpha\beta} = 0$ for boundary points (the boundary is an absorbing barrier for the process). Then the random variable whose mathematical expectation coincides with the solution of the problem is:

$$\theta = \frac{h^2}{2d} \sum_{i=1}^{i^*-1} f_i + \varphi_{i^*},$$

where f_i are values of the function f in the points of the Markov chain, and i^* is the point where Markov chain reaches the boundary $\partial\Omega_h$

The well known *grid* algorithm (can be described in pseudo-code notation (see Figure **??**):

Algorithm 4

```
Start at the grid point $(x_{10}, x_{20})$, i.e. $(x_{(1)}, x_{(2)}) := (x_{10}, x_{20})$
While $(x_{(1)}, x_{(2)})$ is not at the boundary
  Move to a neighboring point $(x'_{(1)}, x'_{(2)}) \in \{(x_{(1)} - h, x_{(2)}),$
  $(x_{(1)} + h, x_{(2)}), (x_{(1)}, x_{(2)} - h), (x_{(1)}, x_{(2)} + h)\}$
  (i.e. $(x_{(1)}, x_{(2)}) := (x'_{(1)}, x'_{(2)})$)
  such that each neighboring is selected with
  the same probability $p = 1/4$
```

Let $(x^*_{(1)}, x^*_{(2)})$ be the final point at the boundary. Then, the searched random variable is:

$$\theta := u(x^*_{(1)}, x^*_{(2)}).$$

In order to compute $E\theta$, we start N Markov processes of the above kind, delivering N realizations $\theta_1, \ldots, \theta_N$ of the random variable θ and approximate the solution by their mean as described above.

Now consider the *grid-free Monte Carlo algorithm* based on the local integral representation of the problem.

First, let us describe the *selection algorithm* (due to John von Neumann) in general case. This approach is also commonly called the *acceptance-rejection* method or *accept-reject algorithm*. Suppose $v_1(x)$ and $v_2(x)$ are given functions, $0 \leq v_1(x) \leq v_2(x)$ and

$$\int_\Omega v_1(x)dx = V_1 < \infty, \int_\Omega v_2(x)dx = V_2 < \infty,$$

where $\Omega \subset \mathbb{R}^3$.

Consider an algorithm for simulation of the random variable with density function $v_2(x)/V_2$ and simulate other random variable with the density function $v_1(x)/V_1$. It is necessary to give a realization ξ of the random variable with density $v_2(x)/V_2$ and an independent realization γ of the random variable uniformly distributed in $(0, 1)$, as well as to check the inequality $\gamma v_2(x) \leq v_1(x)$. If the last inequality holds, ξ is the needed realization. Otherwise, the process have to be repeated. This means that, with enough replicates, the algorithm generates a sample from the desired distribution $v_2(x)/V_2$. There are a number of extensions to this algorithm, such as the *Metropolis (or Metropolis-Hastings)* algorithm. The efficiency of the *selection algorithm* is measured by $E = V_1/V_2$.

A local integral representation (110) for the boundary value problem (93, 94) is obtained. Comparing (95) with (110) one can get

$$k(x,y) = \begin{cases} M_y^* L_p(y,x), & \text{when} \quad x \in \Omega \setminus \partial\Omega, \\ 0, & \text{when} \quad x \in \partial\Omega, \end{cases}$$

and

$$f(x) = \begin{cases} \int_{B(x)} L_p(y,x)\phi(y)dy & \text{when} \quad x \in \Omega \setminus \partial\Omega, \\ \psi(x), & \text{when} \quad x \in \partial\Omega. \end{cases}$$

The Monte Carlo procedure for solving this problem can be defined as a *ball process*. To ensure the convergence of the process, we introduce the ε-strip of the boundary, i.e.

$$\partial\Omega_\varepsilon = \{x \in \Omega : B(x) = B_\varepsilon(x)\}, \text{ where } B_\varepsilon(x) = \{y : r =\mid y - x \mid \leq \varepsilon\}.$$

Consider a transition density function

$$p(x,y) = k(x,y) = M_y^* L_p(y,x) \geq 0. \qquad (113)$$

This transition density function defines a Markov chain $\xi_1, \xi_2, \dots, \xi_i$ such that every point $\xi_j,\ j = 1, \dots, i-1$ is chosen on the maximal ball $B(x_{j-1})$, lying in Ω in accordance with the density (113). The Markov chain stops when it reaches $\partial\Omega_\varepsilon$. So, $\xi_i \in \partial\Omega_\varepsilon$.

Let us consider the random variable

$$\theta[\xi_0] = \sum_{j=0}^{i} Q_j \int_{B(\xi_j)} L_p(y, \xi_j) f(y) dy + \varphi(\xi_i),$$

where

$$Q_0 = 1 \tag{114}$$

$$Q_j = Q_{j-1} M_y^* L_p(\xi_j, \xi_{j-1}) / p(\xi_{j-1}, \xi_j), j = 1, 2, \dots, i, \tag{115}$$

$\varphi(\xi_i)$ is the value of the boundary function at the last point of the Markov chain ξ_i.

It is easy to see that the solution of the problem at the point ξ_0 can be presented as

$$u(\xi_0) = E\theta[\xi_0]. \tag{116}$$

To ensure the convergence of the process we consider the next estimate:

$$\int\int k(x,y)k(y,z)dydz < \int \delta(y-z) \int \delta(z-y)dzdy$$

$$= \int \delta(y-x)dy < 1 - \frac{\varepsilon^2}{4R_m^2},$$

where $k(x,y) \geq 0$ is defined by (113) and R_m is the supremum of all radii of the spheres lying in Ω.

The above estimate ensures the convergence of the Neumann series and, therefore of the process (115) as well.

Obviously, all non-zero values of Q_j are equal to 1 and the problem consists in simulating a Markov chain with a transition density function $p(x,y)$ in the form (113). Thus, the problem of calculating $u(\xi_0)$ is reduced to estimating the expectation (116). As the approximate value of $u(\xi_0)$ is set up

$$\theta_N = 1/N \sum_{s=1}^{N} \{\theta[\xi_0]\}_s,$$

where $\{\theta[\xi_0]\}_s$ is the s^{th} realization of the random variable $\theta[\xi_0]$ on a Markov chain with initial point ξ_0 and transition density function (113).

As it was shown in Section **??**, the probable error for this type of random processes is $r_N = c\sigma(\theta[\xi_0])N^{-1/2}$, where $c \approx 0.6745$ and $\sigma(\theta[\xi_0])$ is the standard deviation.

The direct simulation of a random variable with the stationary density function $p(x,y)$ is unsuitable since the complexity of the expression for $M_y^* L(y,x)$ would sharply increase the algorithm's computational complexity. In this case it is advisable to use the *rejection sampling algorithm*.

Denote by $p_0(x,y)$ the transition density function of the Markov chain $M_y^* L_p$ with $c(x) \equiv 0$.

It is easy to see, that

$$p(x,y) \le p_0(x,y).$$

The function $p_0(x,y)$ satisfies the condition for a transition density function of the Markov chain.

$$\int_{B(x)} p_0(x,y)dy = 1. \tag{117}$$

Indeed,

$$\int_{B(x)} p_0(x,y)dy = \int_{B(x)} M_y^* L_p(y,x)\Big|_{c(y)\equiv 0} dy$$

$$= \int_{B(x)} \sum_{i=1}^{3} \frac{\partial^2 L_p(y,x)}{\partial y_{(i)}^2} dy - \int_{B(x)} \sum_{i=1}^{3} \left(b_i(y) \frac{\partial L_p(y,x)}{\partial y_{(i)}} \right) dy.$$

Apply Green's formula (98) to the second integral:

$$\int_{B(x)} \sum_{i=1}^{3} \left(b_i(y) \frac{\partial L_p(y,x)}{\partial y_{(i)}} \right) dy$$

$$= \int_{\partial B(x)} \sum_{i=1}^{3} n_i b_i(y) L_p(y,x) d_y S - \int_{B(x)} L_p(y,x) \text{div } \mathbf{b}(y) dy = 0,$$

because div $\mathbf{b}(y) = 0$ and $L_p(y,x)|_{y \in \partial B(x)} = 0$, where $\mathbf{n} \equiv (n_1, n_2, n_3)$ is the exterior normal to the boundary $\partial B(x)$.

Calculate the first integral using spherical coordinates:

$$
\begin{aligned}
&\int_{B(x)} \sum_{i=1}^{3} \frac{\partial^2 L_p(y,x)}{\partial y_{(i)}^2} dy \\
&= \int_0^R \int_0^\pi \int_0^{2\pi} \frac{r^2 \sin\theta}{4\pi q_p(R)} \frac{1}{r^2} \frac{\partial}{\partial r} r^2 \frac{\partial}{\partial r} \left(\int_r^R (1/r - 1/\rho) p(\rho) d\rho \right) dr d\theta d\varphi \\
&= \frac{1}{q_p(R)} \int_0^R \frac{\partial}{\partial r} r^2 \frac{\partial}{\partial r} \left(\int_r^R (1/r - 1/\rho) p(\rho) d\rho \right) dr \\
&= \frac{1}{q_p(R)} \left(r^2 \frac{\partial}{\partial r} \int_r^R (1/r - 1/\rho) p(\rho) d\rho \right) \Bigg|_{r=0}^{r=R} \\
&= \frac{1}{q_p(R)} (-1) \int_r^R p(\rho) d\rho \, \Bigg|_{r=0}^{r=R} = \frac{q_p(R)}{q_p(R)} = 1.
\end{aligned}
$$

Thus, we proved (117).

The function $p_0(x,y)$ can be expressed in Cartesian coordinates as

$$p_0(x,y) = \frac{\mu_p(R)}{r^2}\left[p(r) + \sum_{i=1}^{3} b_i(y)\frac{y_{(i)} - x_{(i)}}{r}\right]\int_r^R p(\rho)d\rho.$$

Taking into consideration that

$$dy_1dy_2dy_3 = r^2\sin\theta drd\theta d\varphi \text{ and } y_{(i)} - x_{(i)} = rw_i, \quad i = 1,2,3$$

one can write:

$$p_0(r,\mathbf{w}) = \mu_p(R)\sin\theta\left[p(r) + \sum_{i=1}^{3} b_i(x + r\mathbf{w})w_i\right]\int_r^R p(\rho)d\rho,$$

or

$$p_0(r,\mathbf{w}) = \frac{\sin\theta}{4\pi q_p(R)}\left[p(r) + \sum_{i=1}^{3} b_i(x + r\mathbf{w})w_i\right]\int_r^R p(\rho)d\rho.$$

Here **w**$\equiv$ (w_1, w_2, w_3) is an unique isotropic vector in $\mathbb{R}^3$, where $w_1 = \sin\theta\cos\varphi$, $w_2 = \sin\theta\sin\varphi$ and $w_3 = \cos\theta$.

Now one can write $p_0(r, \mathbf{w})$ in the following form:

$$p_0(r, \mathbf{w}) = p_0(r) p_0(\mathbf{w}/r),$$

where

$$p_0(r) = \frac{p(r)}{q_p(R)} = \frac{k e^{-kr}}{1 - e^{-kR}}$$

is a density function and

$$p_0(\mathbf{w}/r) = \frac{\sin\theta}{4\pi}\left[1 + \frac{\mid \mathbf{b}(x + r\mathbf{w}) \mid \cos(\mathbf{b}, \mathbf{w})}{p(r)} \int_r^R p(\rho) d\rho\right]$$

is a conditional density function.

In Ermakov, Nekrutkin, Sipin, 1992 it is proved that $E \geq \frac{1}{2}$ for the same density function and for the boundary value problem in $\mathbb{R}^d (d \geq 2)$.

A majorant function $h_r(\mathbf{w})$ for $p_0(\mathbf{w}/r)$ was found and the following theoretical result for the algorithm efficiency of the rejection sampling *grid-free* Monte Carlo algorithm was proved:

$$E \geq \frac{1+\alpha}{2+\alpha}, \tag{118}$$

where

$$\alpha = \frac{\max_{x\in\Omega} \mid c(x) \mid R}{\max_{x\in\Omega} \mid \mathbf{b}(x) \mid},$$

and R is the radius of the maximal sphere lying inside Ω.

The following result holds:

Theorem 15. *For the efficiency of the rejection sampling grid-free Monte*

Carlo algorithm the inequality:

$$E \geq \frac{1+\alpha}{2+\alpha-\varepsilon_R}, \quad 0 < \varepsilon_R = \frac{1}{e^{kR}} < 1,$$

holds, when the majorant function

$$h_r(\mathbf{w}) = \frac{\sin\theta}{4\pi}\left[1 + \frac{\max_{x\in\Omega} \mid \mathbf{b}(x) \mid}{p(r)} \int_r^R p(\rho)d\rho\right]$$

is used.

Proof 12. *Estimate the conditional density function $p_0(\mathbf{w}/r)$:*

$$p_0(\mathbf{w}/r) = \frac{\sin\theta}{4\pi}\left[1 + \frac{\mid \mathbf{b}(x + r\mathbf{w}) \mid \cos(\mathbf{b}, \mathbf{w})}{p(r)} \int_r^R p(\rho) d\rho\right]$$

$$\leq \frac{\sin\theta}{4\pi}\left[1 + \frac{B}{p(r)} \int_r^R p(\rho) d\rho\right] = h_r(\mathbf{w})$$

where $B = \max_{x\in\Omega} \mid \mathbf{b}(x) \mid$*.*

On the other hand

$$
\begin{aligned}
h_r(\mathbf{w}) &= \frac{\sin\theta}{4\pi}\left[1+\frac{B}{p(r)}\int_r^R p(\rho)d\rho\right] = \frac{\sin\theta}{4\pi}\left[1+\frac{B}{k}\left(1-e^{-k(R-r)}\right)\right] \\
&= \frac{\sin\theta}{4\pi}\left[1+\frac{B}{k}\left(1-\frac{e^{kr}}{e^{kR}}\right)\right] \\
&\le \frac{\sin\theta}{4\pi}\left[1+\frac{B}{k}\left(1-\frac{1}{e^{kR}}\right)\right] = H(\mathbf{w}). \qquad (119)
\end{aligned}
$$

The functions $h_r(\mathbf{w})$ and $H(\mathbf{w})$ are majorants for the $p_0(\mathbf{w}/r)$. For the efficiency of the rejection sampling Monte Carlo algorithm in the case when $c(y) \equiv 0$ one can obtain:

$$
\begin{aligned}
E &= \frac{\int_0^{2\pi}\int_0^{\pi} p_0(\mathbf{w}/r)d\theta d\varphi}{\int_0^{2\pi}\int_0^{\pi} H(\mathbf{w})d\theta d\varphi} = \frac{1}{1+\frac{B}{k}\left(1-\frac{1}{e^{kR}}\right)} \\
&= \frac{k}{k+B(1-\frac{1}{e^{kR}})} = \frac{B+R\max_{x\in\Omega}\mid c(x)\mid}{2B+R\max_{x\in\Omega}\mid c(x)\mid-\frac{B}{e^{kR}}} = \frac{1+\alpha}{2+\alpha-\varepsilon_R},
\end{aligned}
$$

where

$$k = B + R\max_{x\in\Omega}\mid c(x)\mid, \quad \text{(see (103))},$$

$$\alpha = \frac{\max_{x\in\Omega}\mid c(x)\mid R}{B} \quad \text{and } \varepsilon_R = \frac{1}{e^{kR}}.$$

Taking into consideration (119), one can get

$$E \geq \frac{1+\alpha}{2+\alpha-\varepsilon_R}, \tag{120}$$

when the majorant function $h_r(\mathbf{w})$ is used. This completes the proof.

It is clear that if $\varepsilon_R \to 0$ then the result (118) follows.

Denote by $\bar{p}(x, y)$ the following function:

$$\bar{p}(x, y) = \frac{p(x, y)}{V}, \text{ where } \int_{B(x)} p(x, y) dy = V < 1,$$

This is a density function in the case when $c(y) \neq 0$.

The function $p(x, y)$ can be expressed in spherical coordinates as:

$$\begin{aligned} p(r,\mathbf{w}) &= \frac{\sin\theta}{4\pi q_p(R)} \\ &\times \left[p(r) + \left(\sum_{i=1}^{3} b_i(x + r\mathbf{w}) w_i + c(x + r\mathbf{w}) r \right) \right. \\ &\times \left. \int_r^R p(\rho) d\rho - c(x + r\mathbf{w}) r^2 \int_r^R \frac{p(\rho)}{\rho} d\rho \right]. \end{aligned}$$

The following inequalities hold:

$$\begin{aligned} p(r,\mathbf{w}) &\le p_0(r,\mathbf{w}) \le \frac{p(r)}{q_p(R)} h_r(\mathbf{w}) \\ &= \frac{\sin\theta\, p(r)}{4\pi q_p(R)} \left[1 + \frac{\max_{x\in\Omega} \mid \mathbf{b}(x) \mid}{p(r)} \int_r^R p(\rho) d\rho \right] \equiv h(r,\mathbf{w}). \end{aligned} \tag{121}$$

It is easy to prove that in case when $h(r,\mathbf{w})$ is a majorant of the function $p(r,\mathbf{w})$ the efficiency of the rejection sampling algorithm is

$$E \geq V \frac{1+\alpha}{2+\alpha-\varepsilon_R}.$$

This estimation follows from Theorem 15 and (121). Clearly, the efficiency E depends on the norm of the kernel $k(x,y)$, because $p(x,y)=k(x,y)$.

In the rejection sampling algorithm it is necessary to simulate a random variable η with a density

$$\bar{p}_r(\mathbf{w}) = 1 + \left[\frac{|\,\mathbf{b}(x+r\mathbf{w})\,|\cos(\mathbf{b},\mathbf{w}) + c(x+r\mathbf{w})r}{p(r)}\right]\times$$

$$\times \int_r^R p(\rho)d\rho - \frac{c(x+r\mathbf{w})r^2}{p(r)}\int_r^R \frac{p(\rho)}{\rho}d\rho.$$

Since

$$\bar{p}_r(\mathbf{w}) \leq 1 + \frac{B}{p(r)} \int_r^R p(\rho)d\rho = h(r)$$

the function $h(r)$ is a majorant for the rejection sampling algorithm.

Here a Monte Carlo algorithm for the selection algorithm is described:

Consider a point $x \in \Omega$ with the initial density $p(x)$. Suppose that $p(x)$ is tolerant to $g(x)$.

Algorithm 5. Grid-free Monte Carlo Algorithm

1. **Calculate** *the radius $R(x)$ of the maximal sphere lying inside Ω and having center x.*

2. **Calculate** *a realization r of the random variable τ with the density*

$$\frac{p(r)}{q_p(R)} = \frac{ke^{-kr}}{1 - e^{-kR}}. \tag{122}$$

3. **Calculate** *the function*

$$h(r) = 1 + \frac{B}{p(r)} \int_r^R p(\rho) d\rho = 1 + \frac{B}{k}\left(1 - e^{-k(R-r)}\right).$$

4. **Simulate** *independent realizations* $\mathbf{w}_j$ *of a unique isotropic vector in* $\mathbb{R}^3$*.*

5. **Simulate** *independent realizations* γ_j *of a uniformly distributed random variable in the interval* $[0, 1]$*.*

6. **Calculate** *the parameter* j_0*, given by*

$$j_0 = \min\{j : h(r)\gamma_j \leq \bar{p}_r(\mathbf{w}_j)\},$$

and **stop** *the execution of the steps 4 and 5. The random vector* $\mathbf{w}_{j_0}$ *has the density* $\bar{p}_r(\mathbf{w})$*.*

7. **Calculate** *the random point y, with a density $\bar{p}_r(\mathbf{w})$, using the following formula:*

$$y = x + r\mathbf{w}_{j_0}.$$

The value $r = \mid y - x \mid$ is the radius of the sphere lying inside Ω and having center at x.

8. **Stop** *the random trajectory when the random process reaches the ε-strip $\partial\Omega_\varepsilon$, i.e. $y \in \partial\Omega_\varepsilon$. The random variable is calculated.* **If** *$y \bar{\in} \partial\Omega_\varepsilon$* **then** *the algorithm has to be repeated for $x = y$.*

9. **Perform** *N random trajectories repeating steps 2 to 8.*

An illustration of the considered *grid-free* algorithm is given on Figure **??**.

It is clear that the algorithmic efficiency depends on the expectation of the position of the point y. The location of y depends of the random variable τ with a density (122). When the random point is near to the boundary of the ball, the process goes to the boundary of the domain $\partial\Omega_\varepsilon$ quickly. So, it will be important to have an estimate of the mathematical expectation of τ.

One can calculate the value of $E\tau$:

$$E\tau = \int_0^R r \frac{p(r)}{q_p(R)} dr = \int_0^R \frac{rke^{-kr}}{1 - e^{-kR}} dr = \frac{1}{k} + \frac{R}{1 - e^{kR}}.$$

Obviously, the sequential algorithmic efficiency depends of the product (kR) (where R is the radius of the maximal ball, lying inside of the domain Ω for the starting point of the random process). Therefore, the computational results given in the next section are performed for different values of the product (kR).

Parallel Implementation of the Grid-Free Algorithm and Numerical Results

It is well known that Monte Carlo algorithms are well suited for parallel architectures. In fact, if we consider the calculation of a trajectory as a single computational process, it is straightforward to regard the Monte Carlo algorithm as a collection of asynchronous processes evolving in parallel. Clearly, MIMD (multiple instruction, multiple data) - machines are the $natural$ hardware platform for implementing such algorithms; it seems to be interesting to investigate the feasibility of a parallel implementation on such type of machines. There are two main reasons:

- Monte Carlo algorithms are frequently used, within or in conjunction with more complex and large existing codes (usually written in FORTRAN or C), the easiness in programming makes the use of these machines very attractive;
- the peak performance of each processor of these machines is usually not very high, but when a large number of processors is efficiently used a high general computational performance can be reached.

The MIMD computer used for our tests is a IBM SP1 with 32 processors. The aim of our implementation was to demonstrate a high efficiency of a MIMD system in which each processing element is relatively slow. The algorithm has a very good scalability so that it can be easily implemented on modern and future MIMD systems. The environment for parallel programming is ATHAPASCAN which is developed by the research group on Parallel algorithms in LMC/IMAG, Grenoble. ATHAPASCAN environment is developed using C-language and a special library for message passing which is similar to well-known MPI-Message Passing Interface and PVM-Parallel Virtual Machine. ATHAPASCAN allows to distribute the computational problem on different type of processors or/and computers. This environment provides use of dynamic distribution of common resources and has a high level of parallel efficiency if the numerical algorithm is well parallelized.

In the previous section a general description of the Monte Carlo algorithm for the selection algorithm has been provided. Note that, in the case of an implementation on a sequential computer, all the steps of the algorithm and all the trajectories are executed iteratively, whereas on a parallel computer the trajectories can be carried concurrently.

Example. A numerical example is considered. The example deals with the following problem

$$\sum_{i=1}^{3}\left(\frac{\partial^2 u}{\partial x_{(i)}^2}+b_i(x)\frac{\partial u}{\partial x_{(i)}}\right)+c(x)u=0, \text{ in } \Omega=\mathbf{E}^3\equiv[0,1]^3.$$

Note that the cube $\mathbf{E}^3$ does not belong to the $\mathbf{A}^{(1,\lambda)}$, but this restriction is not important for our algorithm since an ε-strip of the domain Ω is considered. In fact now we consider another domain Ω_ε which belongs to the class $\mathcal{A}^{(1,\lambda)}$.

The boundary conditions for the example are:

$$u(x_{(1)},x_{(2)},x_{(3)})=e^{a_1x_{(1)}+a_2x_{(2)}+a_3x_{(3)}},\ (x_{(1)},x_{(2)},x_{(3)})\in\partial\Omega.$$

In our tests

$$b_1(x)=a_2a_3(x_{(2)}-x_{(3)}),\ b_2(x)=a_3a_1(x_{(3)}-x_{(1)}),\ b_3(x)=a_{(1)}a_2(x_{(1)}-x_{(2)})$$

(thus, the condition $div\ \mathbf{b}(x) = 0$ is valid) and

$$c(x) = -(a_1^2 + a_2^2 + a_3^2),$$

where $a_1, a_2,\ a_3$ are parameters.

The problem is solved using selection *grid-free* Monte Carlo algorithm.

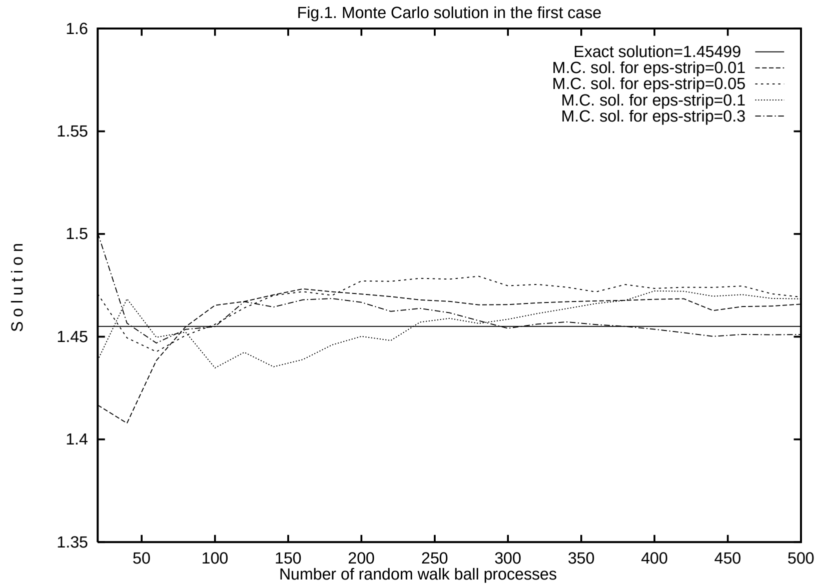

Figure 9: Monte Carlo solution for the first case.

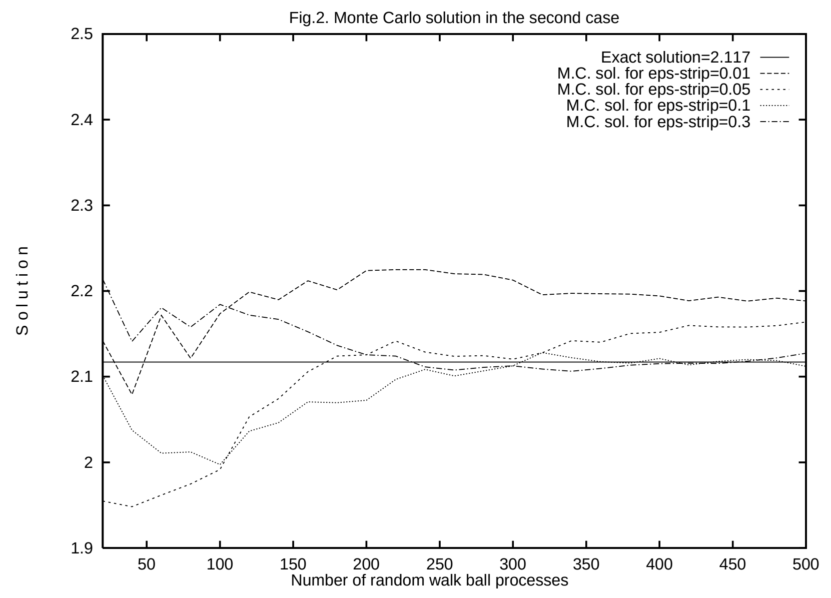

Figure 10: Monte Carlo solution for the second case.

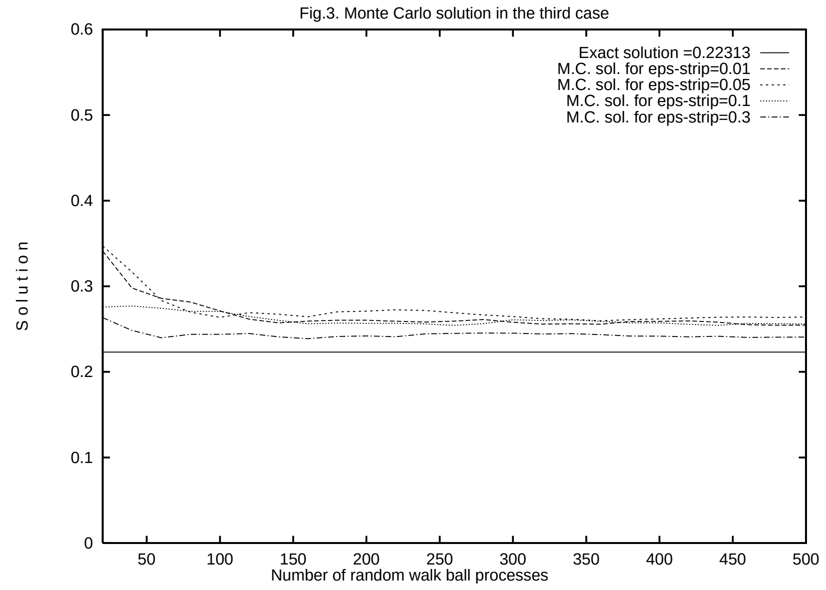

Figure 11: Monte Carlo solution in the third case.

We consider three cases for the coefficients:

- the first case when

$$a_1 = 0.25\,,\ a_2 = 0.25\,,\ a_3 = 0.25 \text{ and } k * R = 0.101;$$

- the second case when

$$a_1 = 0.5\,,\ a_2 = 0.5\,,\ a_3 = 0.5 \text{ and } k * R = 0.40401;$$

- the third case

$$a_1 = -1\,,\ a_2 = -1\,,\ a_3 = -1 \text{ and } k * R = 1.61603.$$

Four different ε-strip are used:

$$\varepsilon = 0.01,\ 0.05,\ 0.1,\ 0.3.$$

The results of evaluating the linear functional (45) for the above mentioned parameters and functions are presented in Figures 9 - 11, the case when

$$h(x) = \delta[(x_{(1)} - 1/2), (x_{(2)} - 1/2), (x_{(3)} - 1/2)].$$

The efficiency of the selection *grid-free* Monte Carlo does not depend on the number of trajectories (see, Table 8). The result of selection efficiency confirms our corresponding theoretical result.

Table 8: Selection efficiency and number of the steps to the boundary domain. The number of realizations of the random ball process is 600.

Epsilon strip	k * R	No of steps	Selection efficiency
0.01	0.101	36 - 37	0.99
0.01	0.40401	35 - 36	0.97123
0.01	1.61603	43 - 44	0.91071
0.05	0.101	17 - 18	0.99
0.05	0.40401	17 - 18	0.9596
0.05	1.61603	20 - 21	0.85829
0.10	0.101	8 - 9	0.9887
0.10	0.40401	9 - 10	0.95371
0.10	1.61603	12 - 13	0.83596
0.30	0.101	1 - 2	0.97
0.30	0.40401	2 - 3	0.92583
0.30	1.61603	2 - 3	0.75561

The efficiency of the presented *grid-free* algorithm is studied in the case when

$$h(x) = \delta[(x_{(1)} - 1/4), (x_{(2)} - 1/4), (x_{(3)} - 1/4)],$$

$$h(x) = \delta[(x_{(1)} - 1/2), (x_{(2)} - 1/2), (x_{(3)} - 1/2)],$$

respectively.

We investigate the parallel efficiency for the following values of the coefficients:

$$a_1 = 1\,,\ a_2 = -0.5\,,\ a_3 = -0.5 \ \text{ and } \ \varepsilon - \text{strip} = 0.01\,,\ 0.05\,,\ 0.15.$$

For the definition of parallel efficiency and other parallel properties we refer to Chapter **??**. The results of parallel implementation showed a very high scalability of the algorithm with increasing the size of the computational job (by increasing the number of random trajectories). The measured parallel efficiency increases with increasing the number of random trajectories and is close to 1.

Concluding Remarks

- An iterative Monte Carlo algorithm using Green's function is presented and studied. It is proved that the integral transformation kernel in local integral presentation can be used as transition density function in the Markov chain. An algorithm called $ball\ process$ is presented. This algorithm is a $grid\text{-}free$ Monte Carlo algorithm and uses the so-called selection.

- One of the advantages of the $grid\text{-}free$ Monte Carlo algorithm is that it has the rate of convergence $(|log\ r_N|/r_N^2)$ (where r_N is the statistical error) which is better than the rate r_N^{-3} of the $grid$ algorithm. This means that the same error can be reached with a smaller number of trajectories.

- It is preferable to use the selection algorithm when it is difficult to calculate the realizations of the random variable directly.

- The studied algorithm has high parallel efficiency. It is easily programmable and parallelizable.

- The tests performed show that Monte Carlo algorithms under consideration can be efficiently implemented on MIMD-machines. The algorithms under consideration are highly scalable.