Interpretation of a Monte Carlo approach of a finite difference scheme by a game method for modelling

Ivan Dimov\textsuperscript{1} and Krassimir Atanassov\textsuperscript{2}

\textsuperscript{1} Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Acad. G.Bouchev Str., bl.25 A, 1113 Sofia, BULGARIA

\textsuperscript{2}Department of Bioinformatics, Bulgarian Academy of Sciences, Acad. G.Bouchev Str., bl. 105, 1113 Sofia, BULGARIA

Abstract

A Monte Carlo technique of a finite difference scheme is interpreted by a game method for modelling (GMM). The simple GMM model is approximated by a second order difference scheme. To estimate the number of moves (jumps) from a given point of the domain to the boundary we approximate the finite difference scheme by a boundary value problem for an elliptic partial differential equation. Then we use the fundamental solution of the problem as an approximation to the average number of moves needed to reach the boundary from an arbitrary point inside the domain.

Keywords: Finite difference scheme, Game method for modelling, Modelling, Monte Carlo approach

AMS Classification: 11C20

1 Introduction

A simple model is approximated by a second order difference scheme following the Monte Carlo technique developed in [2]. To estimate the number of moves (jumps) from a given point of the domain to the boundary we approximate the finite difference scheme by a boundary value problem for an elliptic partial differential equation. Then we use the fundamental solution of the problem as an approximation to the average number of moves needed to reach the boundary from an arbitrary point inside the domain.

Here, using the notations from [1], we give GMM interpretation of the finite difference scheme. Namely, the 2-dimensional GMM uses a grid covered by a (finite or infinite) square lattice. Here, the grid will be finite, but the GMM-objects will go not from cell to a neighbouring cell, but from a
2 Grid Monte Carlo Algorithm

Consider a regular mesh (lattice) with step-size $h$ in $\mathbb{R}^d$. Let $\Omega_h$ be the set of all inner mesh points ($\gamma \in \Omega_h$ if and only if $\gamma \in \Omega$); $\partial \Omega_h$ be the set of all boundary mesh points ($\gamma \in \partial \Omega_h$ if there exists a neighboring mesh point $\gamma^*$ which does not belong to $\mathbb{R}^d \setminus \Omega$) and $u_h$ be a function defined on a set of mesh points (a mesh function).

The differential operator $L$ at the mesh point $x_i \in \Omega_h$ is approximated by a difference operator $L_h$ as follows:

$$ (L_h u_h)_i = \sum_{x_j \in P_h(x_i)} a_h(x_i; x_j) u_h(x_j) , \quad (1) $$

where $a_h(x_i; x_j)$ are coefficients; and $P_h(x_i)$ is a set of mesh points with center in $x_i \in \Omega_h$ called scheme.

Since $L$ is a linear differential operator, after the discretization of (1), the following system of linear equations arises

$$ Au = b , \quad (2) $$
where \( b = (b_1, \ldots, b_n)\) is an \( n\)-dimensional vector and \( A \in \mathbb{R}^{n \times n} \) is an \( n \times n\)-dimensional matrix. For solving the system (2) one can use MC methods Monte Carlo method described in [2].

3 Monte Carlo Algorithms

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

Let the differential operator \( L \) be the Laplacian:

\[
L = \Delta.
\]

Using a regular discretisation with a step-size \( h \) one can have the following approximation of the differential equation by a difference equation

\[
\Delta_h^{(d)} u = -f_h. \tag{3}
\]

Assume that (3) 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 (3) becomes

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

The matrix form of equation (4) 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 (4) consists in simulating a Markov chain with initial density \( p_0 \) which is permissible to the vector \( h \) (see, [2]). The probability \( p_{\alpha \beta} \) for the transition from the point \( \alpha \) to the next point \( \beta \) in our case is equal to \( \frac{1}{2d} \) if the point is inside of the domain, that is \( ((x_{(1)}, x_{(2)}) \in \Omega_h, \text{ and } p_{\alpha \beta} = 0 \) 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:

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

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 4):
Fig. 4. Grid Monte Carlo algorithm (N trajectories from the initial point \((x_{10}, x_{20})\) to the boundary are constructed and the mean value of the encountered boundary values is computed).

**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 \(\theta\) and approximate the solution by their mean as described above.

### 4 General Description

Here we consider the grid Monte Carlo algorithm described above. Using a regular discretization with step \(h\), the Game model under consideration is approximated by the difference equation

\[ \Delta_h^{(d)} u = -f_h, \]  

or solved for the \(i\)th point \(i = (i_1, \ldots, i_n)\):

\[ u_i = L_h u + \frac{1}{2d} f_i, \]
where $\Delta_h^{(d)}$ is the Laplace difference operator, and $L_h$ is an averaging operator.

The approximation error of (5), (6) is

$$|u_i - u(x_i)| = O(h^2).$$  \hspace{1cm} (7)

We need $h = \sqrt{\epsilon}$ to ensure consistency of the approximation error and the probability error.

Every transition in the Markov chain is done following the algorithm

(i) generation of a random number (it is usually done in arithmetic operations, where $= 2$ or 3);

(ii) determination of the next point which includes a random number of logical operations \(^1\) with expectation equal to $d$, and maximum $2d$ ($d$ is the space dimension) and one algebraic operation to calculate the coordinates of the point.

Let $m_i(h, d)$ be the mean number of steps required to reach the boundary for a process (or a chain) starting at the point $i$ and $m(h, d)$ is the vector with coordinates $m$. Then $m(h, d)$ is the solution of the following finite difference problem

$$\begin{cases} m_i = L_h m + \varphi_i, & i \in \Omega_h, \quad \varphi_i = \frac{2d}{h^2}, \\ m_i = 0, & i \in \partial \Omega_h. \end{cases} \hspace{1cm} (8)$$

If $\{m^\prime_i(h, d)\}$ is the solution of problem (8) in the unit cube $E^d$, then the following inequality holds:

$$m^\prime_i(h, d) \geq m_i(h, d), \quad i = 1, 2, \ldots.$$  

The difference problem for $m^\prime_i(h, d)$ could be approximated by the partial differential problem

$$\begin{cases} \Delta m^\prime(x) = -\frac{d^2}{h^2}, & x \in E^d, \\ m^\prime(x) = \frac{1}{h^2} \sum_{i=1}^{d} (x_i - x_i^2), & x \in \partial E^d, \end{cases}$$

where the error is of order $O(1)$. Note that to estimate the computational complexity we approximate the finite difference problem by a differential problem which has an exact solution.

Let $M$ be the maximum of $m^\prime(x)$, i.e.,

$$M = \max_{E^d} m^\prime(x) = \frac{1}{4h^2} = \frac{1}{4\epsilon}. \hspace{1cm} (9)$$

This value is independent of the problem dimension and could be reached when $x_i = \frac{1}{2}, i = 1, 2, \ldots, d.$

The result in (9) is natural because the inverse operator of $\Delta_h^{(d)}$ is uniformly bounded over $h$, and so it follows that

$$m_i(h, d) = \frac{c(d)}{h^2} + O\left(\frac{1}{h^2}\right).$$

\(^1\)Here logical operation means testing the inequality "$a < b"."
Thus, $M$ is an upper bound for the expectation of the number of steps in a Markov chain, which is a realization of the r.v. $\theta$. To achieve a probable error $\varepsilon$, it is necessary to average $N$ realizations of the r.v. $\theta$, where

\[ N = \frac{1}{\varepsilon^d} \frac{\sigma^2(\theta)}{\varepsilon^2}. \]

The expectation of the number of all transitions $R$ in the Markov chain will then be

\[ R \leq MN = \frac{1}{d} \frac{1}{\varepsilon^d} \frac{1}{\varepsilon^2} \sigma^2(\theta) = t \sigma^2(\theta). \]  

(10)

5 GMM-model

The so described Monte Carlo process can obtain the following GMM-interpretation (see [1]). One GMM-object stays in some initially fixed vertex "i" and having as an initial characteristic value $f_i$. The rules for the GMM-functioning are the following.

1. A random number $r$ is generated. The object will go from its vertex "a" in one of the four neighbouring vertices:

\[ a \rightarrow \begin{cases} b, & \text{if } r \in [0, \frac{1}{4}] \\ c, & \text{if } r \in (\frac{1}{4}, \frac{1}{2}] \\ d, & \text{if } r \in (\frac{1}{2}, \frac{3}{4}] \\ e, & \text{if } r \in (\frac{3}{4}, 1] \end{cases} \]

2. The object obtains in its new vertex the value $\frac{b^2}{2d} f_i$, where $i \in \{b, c, d, e\}$.

3. The process stops when the object arrives to the boundary of the GMM-grid, where it obtains as a (final) characteristic the sum of its previous characteristics, using formula $(4^*)$.

6 Conclusion

The GMM-model presented in this paper corresponds to the simplest form the problem under consideration.

In a next research the authors plan to discuss essentially more complex processes. The first of them is related to the possibility for generation of a series of objects in one (initially fixed) vertex. These objects will go in the GMM-grid simultaneously, following the rules from Section 4. In this case it is possible, two or more objects to enter one vertex.

The second extension is related to use simultaneously different vertices from where objects to start their movement.
Interpretation of a Monte Carlo approach of a finite difference scheme

The third extension is related to use of probabilities with different values for object transfer from one vertex to another (cf. point 1 of the above mentioned list of rules).

The fourth extension of the model includes possibility to construct intervals on the boundaries, in each of which cells objects are entered.

7 Acknowledgement

The authors are grateful for the support provided by the project DID-02-29 "Modelling processes with fixed development rules", as well as project DTK 02-44 "Efficient Monte Carlo methods for large-scale scientific problems" funded by the National Science Fund, Bulgarian Ministry of Education, Youth and Science.

References
