PLISKA
STUDIA MATHEMATICA
BULGARICA

ПЛИСКА
БЪЛГАРСКИ
МАТЕМАТИЧЕСКИ
СТУДИИ

Pliska Stud. Math. Bulgar. 13 (2000), 57-77

PLISKA
STUDIA MATHEMATICA
BULGARICA

MONTE CARLO ALGORITHMS FOR LINEAR PROBLEMS

Ivan Dimov

Keywords: Monte Carlo algorithms, linear problems, boundary value problem, efficiency estimator, Markov chain, parallel algorithms.

MSC subject classification: 65 C 05, 65 U 05 .

1 Introduction

Monte Carlo methods are a powerful tool in many fields of mathematics, physics and engineering. It is known, that these methods give statistical estimates for the functional of the solution by performing random sampling of a certain chance variable whose mathematical expectation is the desired functional.

Monte Carlo methods are methods for solving problems using random variables. In the book [16] edited by Yu.A.Shreider one can find the following definition of the Monte Carlo method.

Definition 1.1 *The Monte Carlo method consists of solving various problems of computational mathematics by means of the construction of some random process for each such problem, with the parameters of the process equal to the required quantities of the problem.*

Usually Monte Carlo methods reduce problems to the approximate calculation of their mathematical expectations.

The year 1949 in generally regarded as the official birthday of the Monte Carlo methods when the paper of Metropolis and Ulam [13] was published, although some authors point to earlier dates. Ermakov [7], for example, notes that a solution of a problem by the Monte Carlo method is contained in the Old Testament. The development of the method is connected with the names of John von Neumann, E. Fermi and G. Kahn, who worked at Los Alamos (USA) for forty years. The development of modern computers and particularly parallel computing systems, provided fast and specialized generators of random numbers and gave a new momentum to the development of Monte Carlo methods.

There are two main directions in development and studying Monte Carlo algorithms. The first direction is *Monte Carlo simulation* algorithms. These algorithms are used for *simulation* of real-life processes and phenomena. In this case, the algorithm follows the corresponding physical, chemical or biological processes under consideration. In such simulations Monte Carlo is used as a tool for choosing different possibilities. Monte Carlo simulation could be considered as a method for solving *probabilistic* problems using some kind of simulations of *random variables* or *random fields.*

The second direction is *Monte Carlo numerical* algorithms. The Monte Carlo numerical algorithms are usually used for solving *deterministic* problem by modeling random variables or random fields. The main idea here is to construct some *artificial* random process and to prove that the mathematical expectation of the process is equal to the unknown solution of the problem or to some functional of the solution. Usually, there are more than one possibility to create such an artificial process. After finding the process one needs to define an algorithm for computing realizations of the random variable. Usually, the random variable can be considered as a weight of a random process (usually, a Markov process). Then, The Monte Carlo algorithm for solving the problem under consideration consists in simulation the Markov process and computing the realizations of the random variables.

In this paper algorithms of the second direction, i.e. *Monte Carlo numerical* algorithms will be considered. The general scheme of our consideration is the following

- We define the problem under consideration and show the conditions which have to be satisfied for obtaining the unique solution.
- Construct a random process and prove that such a process can be used for obtaining the approximate solution of the problem.
- Estimate the probability error of the method.
- Try to find the optimal (in some sense) algorithm, i.e. choose the random process for which the statistical error is minimal.
- Choose the parameters of the algorithm (such that the number of the realizations of the random variable, the length of the Markov chain and, so on) in order to have a good balance between the *statistical* and the *systematic* errors.
- Obtain a priori estimates for the *speed-up* and the *parallel efficiency* of the algorithm when *parallel or vector machines* are used.

An important advantage of the Monte Carlo algorithms is that they allow to find directly the unknown functional of the problem solution with a number of operations, necessary to solve the problem in one point of the domain [17], [4]. Often, one do not need to know the solution in the whole domain in which the problem is defined. Usually, it is necessary to know the value of some functional of the solution. Problems of this kind could be found in many field of the applied sciences. For example, in statistical physics, one is interested in computing linear functional of the solution of the equations for density distribution function (such as Boltzmann , Wigner or Schroedinger equation), i.e., probability to find the particle in a given place of the space and in a given time (integral of the solution), mean value of the velocity of the particles (the first integral moment of the velocity) or the energy (the second integral moment of the velocity) and, so on.

It is well known that, Monte Carlo methods are very efficient when parallel processors or parallel computers are available [5], [6]. This is because these methods are inherently parallel and have loose dependencies. In addition, they are also well vectorizable when powerful vector machines are used [5]. Nevertheless, the problem of parallelization of Monte Carlo methods is not a trivial problem. To find the most efficient parallelization in order to obtain a high value of the speed-up of the algorithm is very important practical problem.

2 Iterative Monte Carlo algorithms

In general, Monte Carlo numerical algorithms may be divided into two classes – *direct* algorithms and *iterative* algorithms. Define an iteration of *order i* as a function of the following form

$$u^{(k+1)} = F_k(A, b, u^{(k)}, u^{(k-1)}, \dots, u^{(k-i+1)}),$$

where $u^{(k)}$ is obtained from the k-th iteration. It is desired that

$$u^{(k)} \to u = A^{-1}b \ \text{ as } \ k \to \infty.$$

The method is called *stationary* if $F_k = F$ for all k, that is, F_k is independent of k. The iterative process is called *linear* if F_k is a linear function of $u^{(k)}, \dots, u^{(k-i+1)}$. In this paper we study *stationary linear iterative Monte Carlo* algorithms.

Consider a general description of the iterative Monte Carlo algorithms. Let $\mathbf{X}$ be a Banach space of real-valued functions. Let $f = f(x) \in \mathbf{X}$ and $u_k = u(x_k) \in \mathbf{X}$ be defined in $\mathbf{R}^d$ and $L = L(u)$ be a linear operator defined on $\mathbf{X}$.

Consider the sequence $u_1, u_2, \dots$, defined by the recursion formula

$$u_k = L(u_{k-1}) + f, \quad k = 1, 2, \dots \tag{1}$$

The formal solution of (1) is the truncated Neumann series

$$u_k = f + L(f) + \dots + L^{k-1}(f) + L^k(u_0), \qquad k > 0, \tag{2}$$

where L^k means the k-th iterate of L.

As an example consider the integral iterations. Let $u(x) \in \mathbf{X}$, $x \in G \subset \mathbf{R}^d$ and $l(x, x')$ be a function defined for $x \in G, x' \in G$. The integral transformation $Lu(x) = \int_G l(x, x')u(x')dx'$ maps the function $u(x)$ into the function $Lu(x)$, and is called an *iteration of* $u(x)$ *by the integral transformation kernel* $l(x, x')$. The second integral iteration of $u(x)$ is denoted by $LLu(x) = L^2u(x)$. Obviously, $L^2u(x) = \int_G \int_G l(x, x')l(x', x'')dx'dx''$. In this way $L^3u(x), \dots, L^iu(x), \dots$ can be defined.

When the infinite series converges, the sum is an element u from the space $\mathbf{X}$ which satisfies the equation

$$u = L(u) + f. \tag{3}$$

The truncation error of (2) is $u_k - u = L^k(u_0 - u)$. Let $J(u_k)$ be a functional that is to be calculated. Consider the spaces

$$\mathbf{T}_{i+1} = \underbrace{\mathbf{R}^d \times \mathbf{R}^d \times \ldots \times \mathbf{R}^d}_{i \text{ times}}, \qquad i = 1, 2, \ldots, k, \tag{4}$$

where "$\times$" denotes the Cartesian product of spaces.

Random variables θ_i, $i = 0, 1, \ldots, k$ are defined on the respective product spaces $\mathbf{T}_{i+1}$ and have conditional mathematical expectations:

$$E\theta_0 = J(u_0), \; E(\theta_1/\theta_0) = J(u_1), \ldots, E(\theta_k/\theta_0) = J(u_k),$$

where $J(u)$ is a functional of u.

The computational problem then becomes one of calculating repeated realizations of θ_k and combining them into an appropriate statistical estimator of $J(u_k)$.

As an approximate value of the linear functional $J(u_k)$ is set up

$$J(u_k) \approx \frac{1}{n} \sum_{s=1}^{n} \{\theta_k\}_s, \tag{5}$$

where $\{\theta_k\}_s$ is the s-th realization of the random variable θ_k.

The probable error is defined as a value r_n (see, for example [17], [8]) for which the following condition

$$P\{|J(u_k) - E\theta_k| < r_n\} \approx \frac{1}{2} \approx P\{|J(u_k) - E\theta_k| > r_n\}$$

is fulfilled.

One can show (following [17]) that for algorithms under consideration

$$r_n \approx 0.6745\sqrt{Var\theta_k/n},$$

where

$$Var\theta_k = E((\theta_k)^2) - (E\theta_k)^2$$

Note that the nature of the every process realization of θ is a Markov process. We will consider only *discrete Markov processes with a finite set of states*, the so called *finite discrete Markov chains* $S = \{s_1, s_2, \ldots, s_m\}$. At each of a discrete sequence of times $t = 0, 1, \ldots, k$ the system S is in one of the states $s_{t_0}, s_{t_1}, \ldots, s_{t_k}, \ldots$, which satisfies the Markov property

$$P(s_{t_q} = \alpha_q | s_{t_{q-1}}, s_{t_{q-2}}, \ldots, s_{t_0}) = P(s_{t_q} = \alpha_q | s_{t_{q-1}}), \; q = 0, 1, \ldots, \; \alpha_q \in \{s_1, \ldots, s_m\}.$$

·Any state s_i is associated with a set of conditional probabilities p_{ij}, such that p_{ij} is the probability that the system, which at the t-th time, is in the state s_i, will be in the state s_j at the $(t+1)$-th time, i.e. $P(s_{t_{k+1}} = s_j | s_{t_k} = s_i) = p_{ij}$.

Definition 2.1 *The state is called absorbing if the chain terminates in this state with probability* 1.

In the general case, iterative Monte Carlo algorithms can be defined as *terminated Markov chains*:

$$S_k = s_{t_0} \to s_{t_1} \to s_{t_2} \to \ldots \to s_{t_k}, \tag{6}$$

where s_{t_q}, $(q = 1, \ldots, k)$ is one of the absorbing states. This determines the value of some function $F(S) = \theta_k$, which depends on the sequence (6). The function $F(S)$ is a random variable. After the value of $F(S)$ has been calculated, the system is restarted to its initial state s_{t_0} and the transitions are begun anew. A number of n independent runs are made through the Markov chain starting from the state s_{t_0} to any of the absorbing states. The average $\frac{1}{n}\sum_T F(S)$ is taken over all actual sequences of transitions. This value approximates $E\{F(S)\}$, which is the required functional.

We also will be interested in *computational complexity.*

Definition 2.2 *The computational complexity is defined by $nE(k)t_0$, where $E(k)$ is the mathematical expectation of the number of transitions in the sequence (6) and t_0 is the mean time needed for realization of one transition.*

There are two approaches which correspond to two special cases of the operator L :
(i) L is a matrix and u and f are vectors;
(ii) L is an ordinary integral transform

$$L(u) = \int_G l(x,y)u(y)dy.$$

First consider the second case. Equation (3) becomes

$$u(x) = \int_G l(x,y)u(y)dy + f(x) \text{ or } u = Lu + f. \tag{7}$$

Monte Carlo algorithms frequently involve the evaluation of linear functionals of the solution of the following type

$$J(u) = \int_G h(x)u(x)dx = (u,h). \tag{8}$$

In fact, the equation (8) defines an inner product of a given function $h(x) \in \mathbf{X}$ with the solution of the integral equation (7).

Sometimes, the adjoint equation

$$v = L^* v + h \tag{9}$$

will be used.

In (9) $v, h \in \mathbf{X}^*$, $L^* \in [\mathbf{X}^* \to \mathbf{X}^*], \mathbf{X}^*$ is the adjoint functional space to $\mathbf{X}$ and L^* is an adjoint operator.

For some important applications $\mathbf{X} = \mathbf{L}_1$ and

$$\| f \|_{\mathbf{L}_1} = \int_G \mid f(x) \mid dx; \quad \| L \|_{\mathbf{L}_1} \leq \sup_x \int_G \mid l(x, x') \mid dx'.$$

In this case $h(x) \in \mathbf{L}_\infty$, hence $\mathbf{L}_1{}^* \equiv \mathbf{L}_\infty$ and $\| h \|_{\mathbf{L}_\infty} = \sup |h(x)|$, $x \in G$. For many applications $\mathbf{X} = \mathbf{X}^* = \mathbf{L}_2$. Note also, that if $h(x), u(x) \in \mathbf{L}_2$ then the inner product (8) is finite.

If it is assumed that $\| L^m \| < 1$, where m is any natural number, then the Neumann series $u = \sum_{i=0}^{\infty} L^i f$ converges. The condition $\| L^m \| < 1$ is not very strong, since it is possible to construct a Monte Carlo method for which the Neumann series does not converge. Analytically extending the resolvent by a change of the spectral parameter gives a possibility to obtain a convergent method when Neumann series for the original problem does not converge or to accelerate the rate of convergence when it converges slowly.

Consider the Monte Carlo method for evaluation the functional (8). It can be seen that when $l(x, x') \equiv 0$ evaluation of the integrals can pose a problem. Consider a random point $\xi \in G$ with a density $p(x)$ and let there be n realizations of the random point $\xi_i (i = 1, 2, ..., n)$. Let a random variable $\theta(\xi)$ be defined in G, such that $E\theta(\xi) = J$.

An approximate value of the functional J, defined by (8) is

$$J \approx \frac{1}{n} \sum_{s=1}^{n} (\theta)_s = \hat{\theta}_n,$$

where $(\theta)_s$ is the s-th realization of the random variable θ.

The random variable whose mathematical expectation is equal to $J(u)$ is given by the following expression

$$\theta[h] = \frac{h(\xi_0)}{p(\xi_0)} \sum_{j=0}^{\infty} Q_j f(\xi_j),$$

where $Q_0 = 1$; $Q_j = Q_{j-1} \dfrac{l(\xi_{j-1}, \xi_j)}{p(\xi_{j-1}, \xi_j)}, j = 1, 2, \ldots$, and $\xi_0, \xi_1, \ldots$ is a Markov chain in G with initial density function $p(x)$ and transition density function $p(x, y)$.

For the first case, when the linear operator L is a matrix, the equation (2) can be written in the following form :

$$u_k = L^k u_0 + L^{k-1} f + \ldots + Lf + f = (I - L^k)(I - L)^{-1} f + L^k u_0, \tag{10}$$

where I is the unit (identity) matrix;

$L = (l_{ij})_{i,j=1}^{m}$; $u_0 = (u_1^0, \ldots, u_m^0)$ and matrix $I - L$ is supposed to be non-singular.

It is well known that if all eigenvalues of the matrix L lie within the unit circle of the complex plane there exists a vector u such that $u = \lim_{k\to\infty} u_k$, which satisfies the equation

$$u = Lu + f \tag{11}$$

(see, for example, [11]).

Now consider the problem of evaluating the inner product

$$J(u) = (h, u) = \sum_{i=1}^{m} h_i u_i , \tag{12}$$

where $h \in \mathbf{R}^m$ is a given vector .

To construct a random variable whose mathematical expectation coincides with (12). Consider the integral equation (7) for which $G = [0, m)$ is an one-dimensional interval divided into equal subintervals $G_i = [i-1, i)$, $i = 1, 2, \dots m$ such that

$$\begin{cases} l(x,y) = l_{ij} & , x \in G_i,\ y \in G_j \\ f(x) = f_i & , x \in G_i \end{cases}$$

Then the integral equation (7) becomes $u_i = \sum_j \int_{G_j} l_{ij} u(y)dy + f_i$ for $u_i \in G_i$. Denote $u_j = \int_{G_j} u(y)dy$ so that one obtains, for $u(x) \in G_i$, $u(x) = \sum_{j=1}^{m} l_{ij} u_j + f_i$. From the last equation it follows that $u(x) = u_i$ and so, $u_i = \sum_{j=1}^{m} l_{ij} u_j + f_i$, or in a matrix form $u = Lu + f$, where $L = \{l_{ij}\}_{i,j=1}^{m}$.

The above permits the construction of the following random variable

$$\theta[h] = \frac{h_{k_0}}{p_0} \sum_{\nu=0}^{\infty} Q_\nu f_{k_\nu} , \tag{13}$$

where

$$Q_0 = 1; \qquad Q_\nu = Q_{\nu-1} \frac{l_{k_{\nu-1},k_\nu}}{p_{k_{\nu-1},k_\nu}}, \qquad \nu = 1, 2, \dots \tag{14}$$

and $k_0, k_1, \dots$ is Markov chain on elements of the matrix L constructed by using an initial probability p_0 and a transition probability $p_{k_{\nu-1},k_\nu}$ for choosing the element $l_{k_{\nu-1},k_\nu}$ of the matrix L.

Now consider the following system of linear equations

$$Au = b , \tag{15}$$

where $A = [a_{ij}] \in \mathbf{R}^{m\times m}$ is a matrix, $b = (b_1, \dots, b_m)^T \in \mathbf{R}^{m\times 1}$ is an m-dimensional vector.

It is possible to choose a non singular matrix $M \in \mathbf{R}^{m\times m}$ such that $MA = I - L$ ($I \in \mathbf{R}^{m\times m}$ is the identity matrix), and $Mb = f$, $f \in \mathbf{R}^{m\times 1}$ Then (11) becomes $MAu = Mb$, i.e, (15). The last equation is equivalent to (12). If matrices M and A are both non-singular and L has its eigenvalues all inside the unit circle, then (13) becomes a *stationary linear iterative Monte Carlo algorithm.* As a result the convergence of the Monte Carlo method depends on truncation error of (10).

3 Monte Carlo Algorithms for Boundary Value Problems

Let $G \subset \mathbf{R}^d$ be a bounded domain with a boundary ∂G. The following notations are used:

$x = (x_1, x_2, \dots, x_d)$ is a point in $\mathbf{R}^d$;

$D^\alpha = D_1^{\alpha_1} D_2^{\alpha_2} \dots D_d^{\alpha_d}$ is an $|\alpha| = \alpha_1 + \alpha_2 + \dots + \alpha_d$ derivative, where $D_i = \partial/\partial x_i$, $i = 1, \dots, d$ and $C^k(\overline{G})$ is a space of functions $u(x)$ continuous on $\overline{G}$ such that $D^\alpha u$ exists in G and admits a continuous extension on $\overline{G}$ for every $\alpha : |\alpha| \leq k$.

We consider the linear boundary value problem

$$Lu \equiv \sum_{|\alpha| \leq 2m} a_\alpha(x) D^\alpha u(x) = -f(x), \qquad x \in G \tag{16}$$

$$u(x) = \varphi(x), \qquad x \in \partial G, \tag{17}$$

where L is an arbitrary linear elliptic operator in $\mathbf{R}^d$ of order $2m$, $a_\alpha(x) \in C^\infty(\mathbf{R}^n)$ and the function $f(x)$ belongs to the Banach space $\mathbf{X}(G)$.

Assume that $f(x)$, $\varphi(x)$, and the boundary ∂G satisfy conditions ensuring that the solution of the problem (16), (17) exists and is unique [14], [15].

We shall study Monte Carlo algorithms for calculating linear functionals (8) of the solution of the problem (16), (17).

There are two approaches for calculating (8). The first approach uses a discretisation of the problem (16, 17) on a mesh and solves the resulting linear algebraic system, which approximates the original problem (16, 17). This is the so-called *grid Monte Carlo algorithm*, or *grid walk algorithm*. The second approach (grid-free approach) uses an integral representation for the problem (16, 17).

3.1 Grid Monte Carlo Algorithm

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

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

$$(L_h u_h)_i = \sum_{x_j \in P_k(x_i)} a_h(x_i, x_j) u_h(x_j) \ , \tag{18}$$

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

Since L is a linear differential operator, after the discretisation of (18), the following system of linear equation arises: $Au = b$, where $b = (b_1, \dots, b_m) \in \mathbf{R}^{m \times 1}$ is an m-dimensional vector and $A \in \mathbf{R}^{m \times m}$ is an $m \times m$-dimensional matrix.

Parallel versions of different grid Monte Carlo algorithm are studied in [4], [6].

3.2 Grid-free Monte Carlo Algorithm

The grid-free Monte Carlo approach under consideration is based on use of local integral representation of the solution. In this case the Green's function for standard domains, lying inside the domain G (for example - ball, sphere, ellipsoid) is used.

Consider the elliptic boundary value problem:

$$Mu = -\phi(x),\ x \in G,\ G \subset \mathbf{R}^3 \tag{19}$$

$$u = \psi(x),\ x \in \partial G, \tag{20}$$

where $M = \sum_{i=1}^{3} \left(\frac{\partial^2}{\partial x_i^2} + b_i(x) \frac{\partial}{\partial x_i} \right) + c(x)$.

Define the class of domains $\mathbf{A}^{(k,\lambda)}$.

Definition 3.1 *The domain G belongs to the class $\mathbf{A}^{(k,\lambda)}$ if for any point $x \in \partial G$ the boundary ∂G can be presented as a function $z_3 = \sigma(z_1, z_2)$ in the neighborhood of x for which $\sigma^{(k)}(z_1, z_2) \in \mathbf{C}^{(0,\lambda)}$, i.e.*

$$|\sigma^{(k)}(y) - \sigma^{(k)}(y')| \leq N|y - y'|^{\lambda},$$

where the vectors $y \equiv (z_1, z_2)$ and $y' \equiv (z_1', z_2')$ are 2-dimensional vectors, N is constant and $\lambda \in (0, 1]$.

If in the closed domain $\overline{G} \in \mathbf{A}^{(1,\lambda)}$ the coefficients of the operator M satisfy the conditions b_j, $c(x) \in \mathbf{C}^{(0,\lambda)}(\overline{G})$, $c(x) \leq 0$ and $\phi \in \mathbf{C}^{(0,\lambda)}(G) \cap \mathbf{C}(\overline{G})$, $\psi \in \mathbf{C}(\partial G)$, the problem (19), (20) has an unique solution $u(x)$ in $\mathbf{C}^2(G) \cap \mathbf{C}(\overline{G})$. The conditions for uniqueness of a solution can be found in ([14], p. 179, [1], p. 79).

We obtain an integral representation of the solution $u(x)$. This representation allows to use the random variable for calculation the functional (8). We have to estimate the functional (8) by means of grid-free Monte Carlo approach. This approach is based on the use of a local integral representation of the solution $u(x)$ in the problem (19), (20). The representation uses the Green's function approach for standard domains, lying inside the domain G. The initial step in studying the grid-free Monte Carlo approach is obtaining of an integral representation of the solution in the form:

$$u(x) = \int_{B(x)} l(x, y) u(y) dy + f(x) \tag{21}$$

assuming that a representation exists.

For the existence of the integral representation, (21) might be obtained using the result [15] taking into consideration that the domain $B(x)$ belongs to the space $\mathbf{A}^{(1,\lambda)}$ and that the operator M is of elliptic type.

We seek a representation of the integral kernel $k(x, y)$ using Levy's function and the adjoint operator M^* for the initial differential operator M.

The Levy's function for the problem (19), (20) is

$$L_p(y,x) = \mu_p(R)\int_r^R (1/r - 1/\rho)p(\rho)d\rho, \;\; r \le R, \tag{22}$$

where the following notations are used:

$p(\rho)$ is a density function; $r =\mid x-y \mid = \left(\sum_{i=1}^{3}(x_i - y_i)^2\right)^{1/2}$, $\mu_p(R) = [4\pi q_p(R)]^{-1}$, $q_p(R) = \int_0^R p(\rho)d\rho$.

It is clear that the Levy's function $L_p(y,x)$, and the parameters $q_p(R)$ and $\mu_p(R)$ depend on the choice of the density function $p(\rho)$. In fact, the equality (22) defines a family of functions. We seek a choice of $p(\rho)$ which leads to a representation of type (21). Moreover, the kernel of the integral transform should be a transition density function, i.e. $l(x,y) \ge 0$.

From an algorithmic point of view the domain $B(x)$ must be chosen in such a way that the coordinates of the boundary points $y \in \partial B(x)$ could be easily calculated.

Denote by $B(x)$ the ball:

$$B(x) = B_R(x) = \{y : r =\mid y - x \mid \le R(x)\}, \tag{23}$$

where $R(x)$ is the radius of the ball.

It is easy to prove that the conditions

$$M_y^* L_p(y,x) \ge 0 \;\; for\ any \;\; y \in B(x)$$

and

$$L_p(y,x) = \partial L_p(y,x)/\partial y_i = 0, \;\; for\ any \;\; y \in \partial B(x), \;\; i = 1,2,3$$

are satisfied for $p(r) = e^{-kr}$, where

$$k \ge \max_{x \in G} \mid \mathbf{b}(x) \mid + R \max_{x \in G} \mid c(x) \mid$$

and R is the radius of the maximal ball $B(x) \subset \overline{G}$.

This statement shows that it is possible to construct the Levy's function choosing the density $p(\rho)$ such that $M_y^* L_p(y,x)$ is non-negative in $B(x)$ and such that $L_p(y,x)$ and its derivatives vanish on $\partial B(x)$.

It follows that the representation (21) can be written in the form:

$$u(x) = \int_{B(x)} M_y^* L_p(y,x)u(y)dy + \int_{B(x)} L_p(y,x)\phi(y)dy.$$

The last representation allows to construct an unbiased estimate for the solution of our problem.

Consider a transition density function

$$p(x,y) = k(x,y) = M_y^* L_p(y,x) \ge 0. \tag{24}$$

This transition density function defines a Markov chain $\xi_1, \xi_2, \dots, \xi_i$ such that every point ξ_j, $j = 1, \dots, i-1$ is chosen on the maximal ball $B(x_{j-1})$, lying in G in accordance with the density (24). The Markov chain stops when it reaches ∂G_ε. So, $\xi_i \in \partial G_\varepsilon$.

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

$$Q_0 = 1; \;\; Q_j = Q_{j-1} M_y^* L_p(\xi_j, \xi_{j-1}) / p(\xi_{j-1}, \xi_j), j = 1, 2, \dots, i,$$

$\varphi(\xi_i)$ 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 in the point ξ_0 can be presented as

$$u(\xi_0) = E\theta[\xi_0]. \tag{25}$$

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 (24). Thus, the problem of calculating $u(\xi_0)$ is reduced to estimation the expectation (25).

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 *selection algorithm*. Let us describe the *selection algorithm*, which we use here. Suppose that $v_1(x)$ and $v_2(x)$ are given functions, $0 \le v_1(x) \le v_2(x)$ and

$$\int_G v_1(x) dx = V_1 < \infty, \;\; \int_G v_2(x) dx = V_2 < \infty,$$

where $G \subset \mathbf{R}^3$.

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 $\gamma v_2(x) \le v_1(x)$. If the last inequality holds, ξ is the needed realization. Otherwise, the process have to be repeated. The efficiency of the selection algorithm is measured by $E = V_1/V_2$.

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

In [9] it is proved that $E \ge \frac{1}{2}$ for the same density function and for the boundary value problem in $\mathbf{R}^d (d \ge 2)$.

In [2] a majorant function $h_r(\mathrm{w})$ for $p_0(\mathrm{w}/r)$ was found and the following theoretical result for the algorithm efficiency of the selection grid-free Monte Carlo algorithm was proved:

$$E \ge \frac{1+\alpha}{2+\alpha},$$

where $\alpha = \frac{\max_{x\in G}|c(x)|R}{\max_{x\in G}|\mathbf{b}(x)|}$, and R is the radius of the maximal sphere lying inside G.

Now, we can prove the following result [3]:

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

when the majorant function $h_r(\mathbf{w}) = \frac{\sin\theta}{4\pi}\left[1 + \frac{\max_{x\in G}|\mathbf{b}(x)|}{p(r)}\int_r^R p(\rho)d\rho\right]$ is used.

Algorithm 3.1 *:*

Grid-free Monte Carlo Algorithm

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

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

$$\frac{p(r)}{q_p(R)} = \frac{ke^{-kr}}{1-e^{-kR}}. \tag{26}$$

3. **Calculate** *the function*

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

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

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

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

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

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

7. **Calculate** *the random point y, with a density $\overline{p}_r(\mathbf{w})$, 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 G and having center at x.

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

3.3 Parallel Implementation of the Grid-free Algorithm and Numerical Results

In the previous subsection a description of the Monte Carlo algorithm for the selection algorithm has been presented. Note that, in the case of implementation on a sequential computer, all the steps of the algorithm and all the trajectory are executed iteratively, whereas on a parallel computer each trajectory can be carried concurrently.

Example. Numerical examples for the following problem

$$\sum_{i=1}^{3}\left(\frac{\partial^2 u}{\partial x_i^2} + b_i(x)\frac{\partial u}{\partial x_i}\right) + c(x)u = 0, \text{ in } G = [0,1]^3$$

are considered.

Note that the cube $G = [0,1]^3$ 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 G is considered. In fact now we consider another domain G_ε which belongs to the class $\mathbf{A}^{(1,\lambda)}$.

The boundary conditions for our examples are:

$$u(x_1,x_2,x_3) = e^{a_1x_1+a_2x_2+a_3x_3}\ , \ (x_1,x_2,x_3) \in \partial G.$$

In our tests $b_1(x) = a_2a_3(x_2 - x_3)$, $b_2(x) = a_3a_1(x_3 - x_1)$, $b_3(x) = a_1a_2(x_1 - x_2)$ (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 problems are solved using selection grid-free Monte Carlo algorithm.

We consider two cases for the coefficients:

- **(i)**: $a_1 = 0.25$, $a_2 = 0.25$, $a_3 = 0.25$ and $k * R = 0.101$;
- **(ii)**: $a_1 = 0.5$, $a_2 = 0.5$, $a_3 = 0.5$ and $k * R = 0.40401$.

Three different ε-strip are used: $\varepsilon = 0.01,\ 0.05,\ 0.1,$.

The function $h(x) \in \mathbf{L}_1$ used in the functional (8) has the form:

$$h(x) = \delta[(x_1 - 1/2),(x_2 - 1/2),(x_1 - 1/2)].$$

The efficiency of the selection grid-free Monte Carlo does not depend on the number of trajectories (see, Table 1). The result of selection efficiency confirms our corresponding theoretical result.

Table 1: **Selection efficiency and number of the steps to the boundary domain.**

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
0.05	0.101	17 – 18	0.99
0.05	0.40401	17 – 18	0.9596
0.10	0.101	8 – 9	0.9887
0.10	0.40401	9 – 10	0.95371

The parallel efficiency is studied for the following values of the coefficients: $a_1 = 1$, $a_2 = -0.5$, $a_3 = -0.5$.

One of the advantages of the grid-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 for 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 have high parallel efficiency. It is easily programmable and parallelizable.

The tests performed show also that Monte Carlo algorithms can be efficiency implemented on MIMD-machines.

4 Applications: Sensitivity Analysis in Environmental Mathematics

Here a special Monte Carlo sensitivity technique to study air-pollution transport over Europe is presented. First, the developed technique is applied on the box-model in order to study the sensitivity of the concentrations of some important pollutants (like NO_2 and O_3). It is shown that the most important parameter is the ratė constant of the reaction producing NO_2 from O_3 and NO. The results are sensitive to small variances of the rate constant.

Second, the developed Monte Carlo simulation technique is apply to the **Danish Eulerian Model.** For running the model to get realistic results for a real-live scenario of air-pollution transport the vector machine CRAY Y-MP C92A is used.

It is shown that the results of the real-live modeling of air-pollution transport are not equal sensitive to different parameters used in the model as input parameters. There are some parameters (like the rate constant of the reaction producing NO_2 from O_3 and NO), which are very important since the results are sensitive to the small changes of the values of these parameters. In this sense the Monte Carlo sensitivity simulation could by used as a special "advisor" to physicists, because this simulation permits to find how accurate have to be measured the input parameters, as well as how strong theory is needed to describe some of the processes of air-pollution transport.

High pollution levels may lead to the destruction of eco-systems and may cause damages on plants, animals and humans. The treatment of the mathematical models will lead to very large computational tasks. Indeed, the application of discretization and splitting procedures leads to several systems of ordinary differential equations. Every system may contain several millions of equations and has to be treated during many time-steps (as a rule several thousand time-steps). It is clear that these computational problems will cause difficulties even when big modern computers are used. This is why it is often necessary to perform some simplifications in the model. Such simplifications must be made so that the output results are still reliable. In order to satisfy the last requirement, one has to investigate how the changes of some parameters or some physical and chemical mechanisms will influence the output results. If the output results are not very sensitive to the variations of certain parameter or mechanism, then this means that the model can be simplified by choosing a simpler algorithm to calculate the parameter under consideration or to describe the mechanism under consideration by a simpler algorithm. If the output results are sensitive to changes in a given parameter of algorithm, then one must be more careful: the parameter must be calculated in a more accurate way (by a more complicated algorithm), the mechanism must be described by a more advanced (and, again, more complicated) algorithm. This short discussion shows that it is useful to perform some sensitivity analysis in order to understand better the relationships between parameters and/or mechanisms used in the model and the output results.

4.1 The Danish Eulerian Model

The Danish Eulerian Model, see [18] is described mathematically by the following system of partial differential equations:

$$\begin{aligned} (27) \qquad \frac{\partial c_s}{\partial t} &= -\frac{\partial(uc_s)}{\partial x} - \frac{\partial(vc_s)}{\partial y} - \frac{\partial(wc_s)}{\partial z} \\ &\quad + \frac{\partial}{\partial x}\left(K_x \frac{\partial c_s}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial c_s}{\partial y}\right) + \frac{\partial}{\partial z}\left(K_z \frac{\partial c_s}{\partial z}\right) \\ &\quad + E_s + Q_s(c_1, c_2, \dots c_q) - (k_{1s} + k_{2s})c_s \,, \quad s = 1, 2, \dots q \,. \end{aligned}$$

The number q of equations in this system is equal to the number of chemical species that are studied by the model. This number varies from 10 to 168 in the experiments described in [18]. The other quantities involved in the model can be described as follows:

- the unknowns c_s are concentrations of the chemical species,
- u, v and w are the components of the wind along the coordinate axes,
- K_x, K_y and K_z are diffusion coefficients,
- the emissions in the space domain are represented by the functions E_s,
- k_{1s} and k_{2s} are coefficients of dry and wet deposition respectively ($s = 1, \dots q$),
- the chemical reactions between species are described by the non-linear functions $Q_s(c_1, c_2, \dots c_q)$, where $s = 1, 2, \dots, q$ (the condensed CBM IV scheme that was proposed by [10], see also [18], is the particular chemical scheme which will be used in this paper).

It is very difficult to treat directly the above system of equations. Therefore, some kind of splitting is to be used. Splitting according to the major physical processes is very popular; see, for example, [19]. Such splitting procedures lead often to five sub-models which are to be treated cyclicly at every time-step ([18]). Five large systems of ordinary differential equations can be obtained from the sub-models by applying any space discretization method:

(28) $dg^{(i)}/dt = f^{(i)}(t, g^{(i)})\ , \quad g^{(i)} \in \mathcal{R}^{N_x \times N_y \times N_z \times N_s}\ , \quad f^{(i)} \in \mathcal{R}^{N_x \times N_y \times N_z \times N_s}\ ,$

where N_x, N_y and N_z are the numbers of grid-points on the grid-lines parallel to the coordinate axes and $N_s = q$ is the number of chemical species involved in the model. The function $f^{(i)}$ is a vector-function whose components are approximations to the concentrations at the grid-points, while $f^{(i)}$ depends on the particular discretization method applied to the corresponding sub-model.

The number of equations in any of the five ODE systems in (28) is equal to the product of the number of grid-points and the number of species. Thus, if $N_x = 96$, $N_y = 96$, $N_z = 10$ and $N_s = 35$, then every ODE system contains 3225600 equations. Furthermore, the five ODE systems are to be treated numerically during many time-steps (typically several thousand time-steps are needed).

It is important to emphasize the fact that the chemical sub-model consists of $N_x \times N_y \times N_z$ *independent* ODE systems. Each of these systems contains $N_s = q$ equations and can be rewritten as

$$dg/dt = f(t, g), \qquad g \in \mathcal{R}^{N_s}, \qquad f \in \mathcal{R}^{N_s}, \tag{29}$$

where g is a vector whose components are approximations to the concentrations at a given grid-point, while the right-hand-side vector f depends on the chemical mechanism which is used in the model. It is clear now that (29) can sometimes be considered (instead of the whole chemical sub-model) in studies of some phenomena which are directly connected to the chemical scheme. The much simpler model (29) will be called the box-model.

4.2 Finding the reaction that has greatest influence on the nitrogen di-oxide and ozone concentrations

The box model is defined by the system of ordinary differential equations (spdisl). This model has been used to find the reaction that has greatest influence on the concentrations of nitrogen di-oxide and ozone. The Monte Carlo algorithm is used in the experiments. In each experiment a sequence of 100 normally distributed random values of the chemical rate constant of one of the chemical reactions was produced by using a random number generator. After that the box model was run for these 100 random values. This procedure has been carried out for all 70 chemical reactions involved in the chemical scheme used in the Danish Eulerian Model; the condensed CBM IV scheme (see [10] and [18]). This means that 100 runs were perform per each chemical reaction with normally distributed random values. The standard deviations of the nitrogen di-oxide and ozone concentrations produced when the chemical rate constants were varied as described above were compared. It has been found in this way that the most important chemical reaction for the nitrogen di-oxide and ozone concentrations is:

$$O_3 + NO \Longrightarrow NO_2\ . \tag{30}$$

This means that small changes of the rate constant of this reaction lead to considerably large changes in the concentrations of nitrogen di-oxide and ozone and, moreover, these changes are larger than the changes, for the same two chemical species, observed when the rate constants of the other chemical reactions were varied by using the same procedure.

4.3 Sensitivity tests with the Danish Eulerian Model

Consider the reaction (30). It is interesting to see what will happen if the experiment with a sequence of 100 normally distributed random values of the constant rate of reaction (30) is performed by using the two-dimensional version of the Danish Eulerian Model (instead of the box model used in the previous section). Several such experiments, each of them consisting of 100 runs with the Danish Eulerian Model, have been carried out. The standard deviation used in the calculating the sequence of normally distributed random values was varied in these experiments. Results obtained when the sequences of 100 normally distributed random values of the constant rate of reaction (30) are produced by using standard deviations $\sigma = 0.5$ and $\sigma = 0.25$ will be presented here, but some other values of the standard deviations were also used.

The results, obtained after performing 100 runs with normally distributed random values of the constant rate of reaction (30), were used to calculate the standard deviations and the skewness of the nitrogen dioxide and ozone concentrations for every value of the standard deviation σ used in the experiments. The results are presented in Figure 3 - Figure 8 (where by X and $Y = \sigma$ the mean value and the standard deviation of the randomly generated normalized rate constants are denoted). The following major conclusions can be drawn from this experiment:

- The standard deviations of the ozone concentrations given in Figure 3 for $\sigma = 0.5$ and Figure 5 for $\sigma = 0.25$ are greatest in the areas where the European emissions are biggest (compare Figure 3 and Figure 5 with Figure 1) and where the nitrogen di-oxide concentrations are highest (compare Figure 3 and Figure 5 with Figure 2).

- The patterns of the distributions of the standard deviations of the nitrogen di-oxide concentrations are not so pronounced (see Figure 4 and Figure 6, where the results obtained, respectively, with $\sigma = 0.5$ and $\sigma = 0.25$ are given). Nevertheless, it is clear that the effect is opposite to the effect observed when the standard deviations of the ozone concentrations are studied. The standard deviations in the most polluted with nitrogen species areas are smaller than the standard deviation in the areas which are far away from the highly polluted areas; compare Figure 4 and Figure 6 with Figure 1 and Figure 2).

- If the standard deviation by which the sequences of normally distributed random values of the rate constant of reaction (30) is reduced, then the standard deviations of the ozone concentrations are also reduced, but the pattern of the distribution of the highest standard deviations remains the same; compare Figure 3 with Figure 5. The same is also true for the pattern of distribution of the highest standard deviations of the nitrogen di-oxide concentrations; compare Figure 4 with Figure 6.

- Results concerning the distribution of the skewness (for the sequence obtained with standard deviation $\sigma = 0.5$) are given in Figure 7 for the skewness of the ozone concentrations and in Figure 8 for the skewness of the nitrogen di-oxide concentrations. The two plots indicate that also here is the effect opposite: in the areas where the skewness of the ozone concentrations is greatest, the skewness of the nitrogen di-oxide concentrations is smallest.

The main result is that the influence of the rate constant of reaction (30) on the ozone concentrations seems to be great in the highly polluted with nitrogen pollutants areas. Therefore an accurate value of this rate constant is needed if the model is to be used on a space domain in which are highly polluted with nitrogen species areas. On the other hand, if in the space domain of the model there are not areas which are highly polluted with nitrogen species, then the accuracy with which this rate coefficient is determined becomes less important.

The author thanks T. Gurov for providing the parallel implementation of the grid-free algorithm in Section 3.3.

REFERENCES

[1] A. V. Bitzadze. Equations of the Mathematical Physics, Nauka, Moscow, 1982.

[2] I. T. Dimov. Efficiency estimator for the Monte Carlo algorithms. In: Numerical Methods and Applications, Proc. II Intern. Conf. on Numerical Methods and Appl., Sofia, Publ. House of the Bulg. Acad. Sci., Sofia, 1989, 111-115.

[3] I. T. Dimov, T. V. Gurov. Estimates of the computational complexity of iterative Monte Carlo algorithm based on Green's function approach. In: IMACS Seminar on Monte Carlo methods and Applications, Brussels, April, 1997, to be published, 1997.

[4] I. Dimov, O. Tonev. Random walk on distant mesh points Monte Carlo methods. *J. of Statistical Physics* **70**(5/6) (1993), 1333-1342.

[5] I. Dimov, O. Tonev. Monte Carlo algorithms: performance analysis for some computer architectures. *J. of Computational and Applied Mathematics* **48** (1993), 253-277.

[6] I. Dimov, A. Karaivanova, H. Kuchen, H. Stoltze. Monte Carlo Algorithms for Elliptic Differential Equations. Data Parallel Functional Approach. *Journal of Parallel Algorithms and Applications* **9** (1996), 39-65.

[7] S. M. Ermakov. On summation of series connected with integral equation. *Vestnik Leningrad Univ. Math.* **16** (1984), 57-63.

[8] S. M. Ermakov, G. A. Mikhailov. Statistical Modeling. Nauka, Moscow, 1982.

[9] S. M. Ermakov, V. V. Nekrutkin, A. S. Sipin. Random processes for solving classical equations of mathematical physics, Nauka, Moscow, 1984.

[10] M. W. Gery, G. Z. Whitten, J. P. Killus, M. C. Dodge. A photochemical kinetics mechanism for urban and regional modeling, *J. Geophys. Res.* **94** (1989), 12925-12956.

[11] G. H. GOLUB, C. F. VAN LOON. Matrix computations. The Johns Hopkins Univ. Press, Baltimore, 1983.

[12] J. M. HAMMERSLEY, D. C. HANDSCOMB. Monte Carlo methods. John Wiley & Sons, inc., New York, London, Sydney, Methuen, 1964.

[13] N. METROPOLIS, S. ULAM. The Monte Carlo Method. *J. of Amer. Statistical Assoc.* **44**, (1949), No. 247, 335-341.

[14] V. P. MIKHAILOV. Partial differential equations. Moskow, Nauka, 1983.

[15] C. MIRANDA. Equasioni alle dirivate parziali di tipo ellittico. Springer-Verlag, Berlin, 1955.

[16] YU. A. SHREIDER. The Monte Carlo method. Method of Statistical Testing. Pergamon Press Ltd., Ozford, London, Edinburg, New York, Paris, Frankfurt, 1966.

[17] I. M. SOBOL. Monte Carlo numerical methods. Nauka, Moscow, 1973.

[18] Z. ZLATEV. Computer treatment of large air pollution models. Kluwer Academic Publishers, Dordrecht-Boston-London, 1995.

[19] Z. ZLATEV, I. DIMOV, K. GEORGIEV. Three-dimensional version of the Danish Eulerian Model. *Zeitschrift für Angewandte Mathematik und Mechanik*, **76**, (1996) S4, 473-476.

Central Laboratory for Parallel Processing
Bulgarian Academy of Sciences
Acad. G. Bonchev St.,bl. 25 A, Sofia, 1113, Bulgaria
e-mail: `dimov@copern.acad.bg`
Web site: http://www.acad.bg/BulRTD/math/dimov2.html

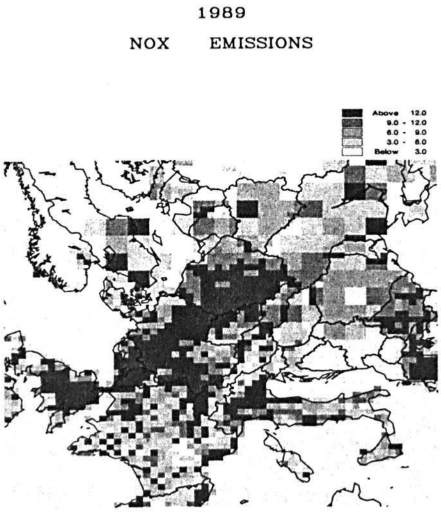

Figure 1
European nitrogen oxides emissions

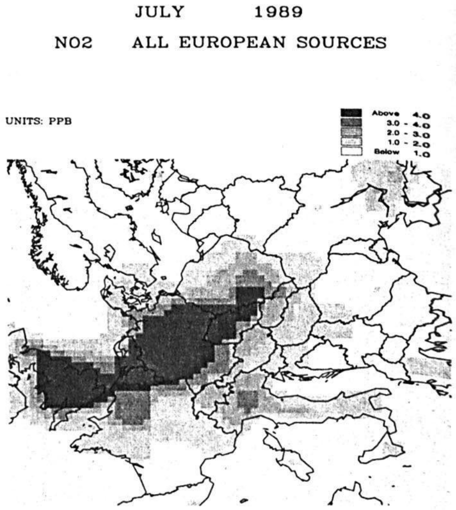

Figure 2
Nitrogen di-oxide concentrations in Europe

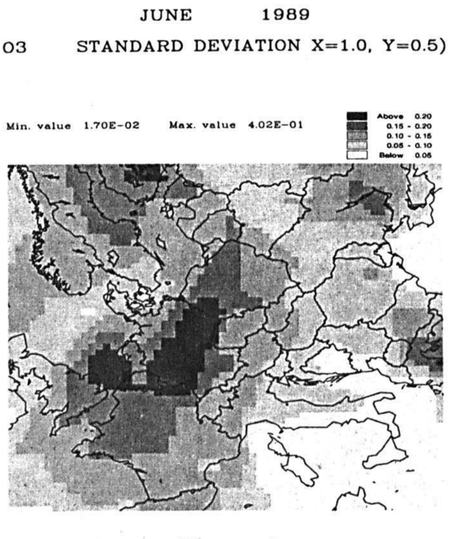

Figure 3
Standard deviations of the ozone concentrations (variance 0.50)

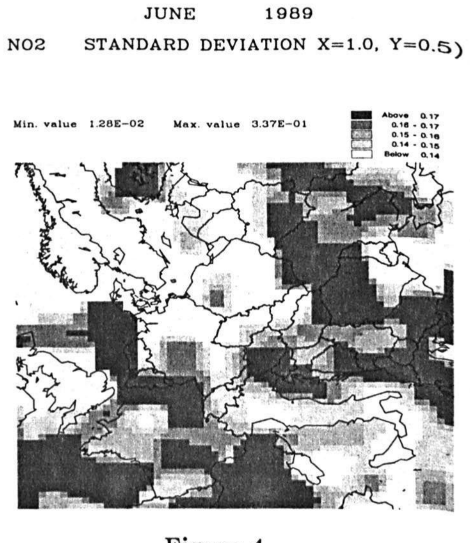

Figure 4
Standard deviations of the nitrogen di-oxide concentrations (variance 0.50)

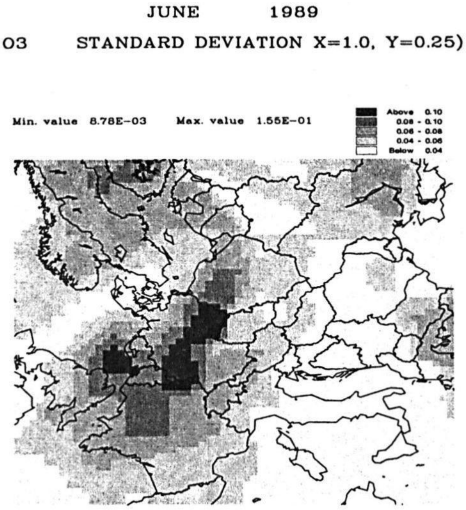

Figure 5
Standard deviations of the ozone concentrations (variance 0.25)

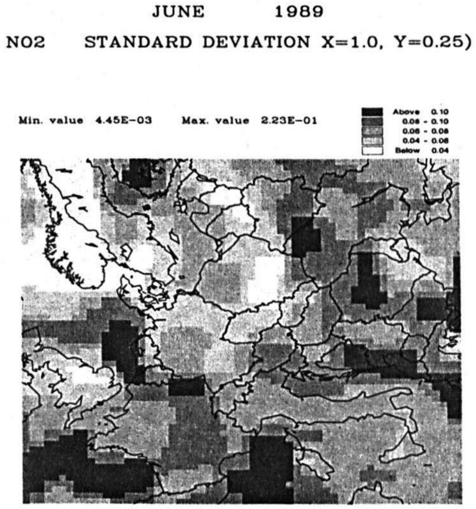

Figure 6
Standard deviations of the nitrogen di-oxide concentrations (variance 0.25)

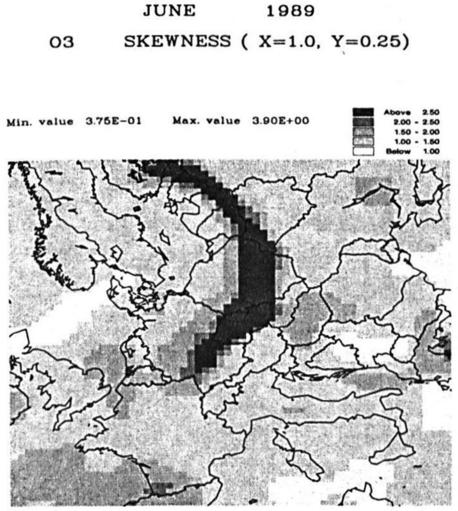

Figure 7
Skewness of the ozone concentrations (variance 0.50)

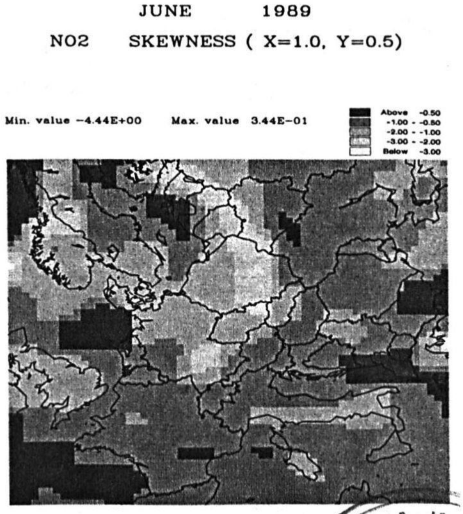

Figure 8
Skewness of the nitrogen di-oxide concentrations (variance 0.50)