Parallel Importance Separation and Adaptive Monte Carlo Algorithms for Multiple Integrals*

Ivan Dimov, Aneta Karaivanova, Rayna Georgieva, and Sofiya Ivanovska

CLPP - Bulgarian Academy of Sciences
Acad. G. Bonchev St., Bl.25A, 1113 Sofia, Bulgaria
ivdimov@bas.bg, anet@copern.bas.bg,
rayna@copern.bas.bg, sofia@copern.bas.bg

Abstract. Monte Carlo Method (MCM) is the only viable method for many high-dimensional problems since its convergence is independent of the dimension. In this paper we develop an adaptive Monte Carlo method based on the ideas and results of the *importance separation*, a method that combines the idea of separation of the domain into uniformly small subdomains with the Kahn approach of importance sampling. We analyze the error and compare the results with crude Monte Carlo and *importance sampling* which is the most widely used variance reduction Monte Carlo method. We also propose efficient parallelizations of the importance separation method and the studied adaptive Monte Carlo method. Numerical tests implemented on PowerPC cluster using MPI are provided.

1 Introduction

Multidimensional numerical quadratures are of great importance in many practical areas, ranging from atomic physics to finance. The crude Monte Carlo method has rate of convergence $O(N^{-1/2})$ which is independent of the dimension of the integral, and that is why Monte Carlo integration is the only practical method for many high-dimensional problems. Much of the efforts to improve Monte Carlo are in construction of variance reduction methods which speed up the computation.

Importance sampling is probably the most widely used Monte Carlo variance reduction method, [5]. One use of importance sampling is to emphasize rare but important events, i.e., small regions of space in which the integrand is large. One of the difficulties in this method is that sampling from the importance density is required, but this can be performed using acceptance-rejection.

It is also known that importance sampling can greatly increase the variance in some cases, [11]. In Hesterberg (1995, [8]) a method of defensive importance sampling is presented; when combined with suitable control variates, defensive importance sampling produces a variance that is never worse than the crude

* Supported by Center of Excellence BIS-21 Grant ICA1-2000-70016 and by the Ministry of Education and Science of Bulgaria under Grants I-1201/02 and MM-902/99

I. Dimov et al. (Eds.): NMA 2002, LNCS 2542, pp. 99–107, 2003.

Monte Carlo variance, providing some insurance against the worst effects of importance sampling. Defensive importance sampling can however be much worse than the original importance sampling.

Owen and Zhow (1999) recommend an importance sampling from a mixture of m sampling densities with m control variates, one for each mixture component. In [11] it is shown that this method is never much worse than pure importance sampling from any single component of the mixture.

Another method, multiple importance sampling, similar to defensive importance sampling, is presented in Veach&Guibas (1995, [13]) and Veach (1997, [14]). It is standard practice to weight observations in inverse proportion to their sampling probability. Multiple importance sampling can break that rule, and do so in a way that still results in an unbiased estimate of the integral. The idea is that in some parts of the sample space, the integrand may be roughly proportional to one of sampling densities while other densities are appropriate to other parts of the space. The goal is to place greater weight on those locally most appropriate densities.

In [9,10] a method called importance separation that combines ideas from importance sampling and stratification is presented and studied. This method has the best possible rate of convergence for certain class of functions but its disadvantage is the increased computational complexity.

Another group of algorithms, widely used for numerical calculation of multidimensional integrals, are the adaptive algorithms. Most of the adaptive algorithms use a sequence of increasingly finer subdivisions of the original region, chosen to concentrate integrand evaluations on subregions with difficulties. Two main types of subdivision strategies are in common use: local and global subdivision. The main disadvantage of local subdivision strategy is that it needs a local absolute accuracy requirement which will be met after the achievement of the global accuracy requirement. The main advantage of the local subdivision strategy is that it allows a very simple subregion management (there is no need to store inactive subregions). Globally adaptive algorithms usually require more working storage than locally adaptive routines, and accessing the region collection is slower. These algorithms try to minimize the global error as fast as possible, independent of the specified accuracy requirement. For example, see [2], where an improved adaptive algorithm for the approximate calculation of multiple integrals is presented - this algorithm is similar to a globally adaptive algorithm for single integrands first described by van Dooren and de Ridder [6]. The modifications are imposed by that the new algorithm applies to a vector of integrands.

The adaptive algorithms proved to be very efficient but they do not have the inherent parallel properties of crude Monte Carlo. In recent years, two approaches to parallel adaptive integration have emerged, for comparison see Bull&Freeman (1998, [4]). One is based on adapting the ideas of sequential globally adaptive algorithms to the parallel context by selecting a number of subdomains of the integration domain according to the associated error estimate, see, for example, Bull&Freeman (1994, [3]). The other approach proceeds by imposing an

initial static partitioning of the domain and treats the resulting problems as independent. This approach needs a mechanism for detecting load imbalance and for redistributing work to other processors, see, for example, [7]. Let us mention that a central feature of the parallel adaptive algorithms is the list containing the subintervals and corresponding error estimates. Fundamentally different parallel algorithms result depending on whether the list is maintained as a single shared data structure accessible to all processors, or else as the union of nonoverlapping sublists, each private to a processor.

In this paper, we use the ideas of importance separation to create an adaptive algorithm for integration. We describe parallelization of these algorithms, study their parallel properties and compare them with importance sampling.

2 Importance Separation and Adaptive Monte Carlo Method

Consider the problem of approximate calculation of the multiple integral

$$I = \int_G f(x)p(x)\,dx, \quad G \equiv [0;1]^d \tag{1}$$

where $f(x)$ is an integrable function for any $x \in G \subset \mathbb{R}^d$ and $p(x) \geq 0$ is a probability density function, such that $\int_G p(x)\,dx = 1$.

The Monte Carlo quadrature formula is based on the probabilistic interpretation of an integral. If $\{x_n\}$ is a sequence in G sampled with density $p(x)$, then the Monte Carlo approximation to the integral is, [12],

$$I \approx I_N = \frac{1}{N}\sum_{n=1}^{N} f(x_n)$$

with the integration error $\varepsilon_N = |I - I_N| \approx \sqrt{\frac{Var(f)}{N}}$.

2.1 Importance Separation

Here we briefly present a Monte Carlo method called importance separation first described and studied in [9,10]. This method combines the ideas of stratification and importance sampling and has the best rate of convergence (see [1]) for the class of functions with bounded derivatives.

The method of importance separation uses a special partion of the domain and computes the given integral as a sum of the integrals on the subdomains. First, let us describe the method in the one-dimensional case - when the domain is an interval, say $[0,1]$ and $f(x) \in \mathbb{C}_{[0,1]}$. Partition [0,1] into M subintervals in the following way: $x_0 = 0$; $x_M = 1$; $G_i \equiv [x_{i-1}, x_i]$;

$$x_i = \frac{C_i}{f(x_{i-1})(M-i+1)}, \quad i = 1, ..., M-1 \tag{2}$$

where

$$C_i = 1/2[f(x_{i-1}) + f(1)](1 - x_{i-1}),\ i = 1, \ldots, M-1.$$

Obviously, $I = \int_0^1 f(x)p(x)\,dx = \sum_{i=1}^{M} \int_{x_{i-1}}^{x_i} f(x)p(x)dx$. If $f(x) \in H(1, L)_{[0,1]}$, there exist constants $L_i \left(L \geq \max\limits_i L_i\right)$, such that

$$L_i \geq \left|\frac{\partial f}{\partial x}\right| \quad \text{for any } \ x \in G_i. \tag{3}$$

Moreover, for the above scheme there exist constants c_{1_i} and c_{2_i} such that

$$p_i = \int_{G_i} p(x)\,dx \leq c_{1_i}/M,\ i = 1, \ldots, M \tag{4}$$

$$\sup_{x_{1_i}, x_{2_i} \in G_i} |x_{1_i} - x_{2_i}| \leq c_{2_i}/M,\ i = 1, \ldots, M. \tag{5}$$

The following theorem, proved in [9], gives the rate of convergence:

Theorem Let $f(x) \in H(1, L)_{[0,1]}$ and $M = N$. Then using the importance separation (3)-(5) of G we have the following Monte Carlo integration error:

$$\varepsilon_N \approx \sqrt{2}[1/N \sum_{j=1}^{N} (L_j c_{1_j} c_{2_j})^2]^{1/2} N^{-3/2}.$$

Now consider the multidimensional case. For an importance separation with analogous properties (for each coordinate we apply the already described one-dimensional scheme (2) in the same manner), we have the following integration error ($M = N$):

$$\varepsilon_N \approx \sqrt{2}d \left[\frac{1}{N} \sum_{i=1}^{N} (L_i c_{1_i} c_{2_i})^2\right]^{1/2} N^{-1/2-1/d}.$$

The disadvantage of the above described methods is the increased computational complexity. The accuracy is improved (in fact, importance separation gives the theoretically optimal accuracy, [10]) but the price is increased number of additional computations which makes these methods impractical for large d.

2.2 Adaptive Monte Carlo Method

Based on advantages and disadvantages of importance separation we develop an adaptive approach for calculation of the desired scalar variable I. Our adaptive method does not use any a priori information about the smoothness of the integrand, but it uses a posteriori information for the variance. The idea of the method consists in the following: the domain of integration G is separated into subdomains with identical volume. The interval [0;1] on every dimension coordinate is partitioned into M subintervals, i.e.

$$G = \sum_j G_j, \quad j = 1,\ M^d.$$

Denote by p_j and I_{G_j} the following expressions:

$$p_j = \int_{G_j} p(x)\,dx \quad \text{and} \quad I_{G_j} = \int_{G_j} f(x)p(x)\,dx.$$

Consider now a random point $\xi^{(j)} \in G_j$ with a density function $p(x)/p_j$ and in this case

$$I_{G_j} = \mathrm{E}\left[\frac{p_j}{N}\sum_{i=1}^{N} f(\xi_i^{(j)})\right] = \mathrm{E}\theta_N.$$

The algorithm starts with a relatively small number M which is given as input data. For every subdomain the integral I_{G_j} and the variance are evaluated. Then the variance is compared with a preliminary given value. The obtained information is used for the next refinement of the domain and for increasing the density of the random points. In order to choose the first and the next dimension coordinate on which an additive division is made, we use random numbers. To avoid the irregular separation on different coordinates a given coordinate recurs only even all other coordinates have been already chosen. In the end an approximation for the integral $I = \sum_j I_{G_j}$ is obtained. The algorithm is described below.

Algorithm

1. **Input data**: *number of points N, number of subintervals M on every dimension coordinate, constant ε (estimation for the variance), constant δ (stop criterion; estimation for the length of subintervals on every coordinate).*
2. **For** $j = 1, M^d$
   - **2.1 Calculate** *the approximation of I_{G_j} and the variance D_{G_j} in subdomain G_j based on N independent realizations of random variable θ_N*
   - **2.2 If** $(\mathrm{D}_{G_j} \geq \varepsilon)$ **then**
     - **2.2.1 Choose** *the axis direction on which the partition will perform*
     - **2.2.2 Divide** *the current domain into two (G_{j_1}, G_{j_2}) along the chosen direction*
     - **2.2.3 If** *the length of obtained subinterval is less than δ* **then go to** *step 2.2.1* **else** *$j = j_1$ (G_{j_1} is the current domain) and* **go to** *step 2.1*
   - **2.3 Elseif** $(\mathrm{D}_{G_j} < \varepsilon)$, *but an approximation of $I_{G_{j_2}}$ has not been calculated yet* **then** *$j = j_2$ (G_{j_2} is the current domain along the corresponding direction) and* **go to** *step 2.1*
   - **2.4 Elseif** $(\mathrm{D}_{G_j} < \varepsilon)$, *but there are subdomains along the other axis directions* **then go to** *step 2.3*
   - **2.5 Else** *Accumulation in the approximation I_N of I*

3 Parallel Implementation

In this section we present the parallel importance separation and parallel adaptive Monte Carlo algorithms for evaluation of multiple integrals. The crude

Monte Carlo possesses inherent parallelism which is based on the possibility to calculate simultaneously realizations of the random variable on different processors. For our two algorithms (importance separation and simple adaptive) we have additional work: partitioning of the domain and assigning the correspondent portions of work to the available processors. This has to be done very carefully in order to have good load balancing. We consider a multiprocessor configuration with p nodes.

N uniformly distributed random points $x_i \in [0;1]^d$, $i = 1, \ldots, N$ are used to obtain an approximation with given accuracy of the integral (1). For generation of $d-$dimensional random point we need d random numbers.To estimate the performance of the parallel algorithms we use:

$\mathbf{ET_p(A)}$ mathematical expectation of time, required for a set of p processing elements to solve the problem using algorithm A

$\mathbf{S_p(A)} = \frac{\mathbf{ET_1(A)}}{\mathbf{ET_p(A)}}$ speed-up

$\mathbf{E_p(A)} = \frac{\mathbf{S_p(A)}}{\mathbf{p}}$ parallel efficiency.

In parallel version of the adaptive algorithm the processors are separated as "master" and "slaves". The "master" calculates the variance in every subdomain G_j, analyzes it and sends information about the new partition — the limits of new subdomains - to the "slaves".

4 Numerical Experiments

We now present the numerical results for the accuracy and the convergence of crude, importance sampling, adaptive, importance separation MCM for numerical integration. The numerical tests are implemented on a cluster of 4 two-processor computers Power Macintosh using MPI. The results are presented as a function of the sample size, N. For each case the error is computed with respect to the exact solution. A lot of numerical tests were performed. Here we present the results of evaluating the 5-dimensional integral over $I^5 = [0;1]^5$ of the function $f(x) = \exp\left(\sum_{i=1}^5 a_i x_i^2 \frac{2+\sin(\sum_{j=1, j\neq i}^5 x_j)}{2}\right)$ using the positive definite importance function (density) $h(x) = \frac{1}{\eta}\exp\left(\sum_{i=1}^5 a_i x_i^2\right)$ where $\mathbf{a} = (1, 0.5, 0.2, 0.2, 0.2)$ and $\eta = \int_{I^5} exp(\sum_{i=1}^5 a_i x_i^2)\, dx$ so that h is normalised. We denote by I_N an estimation of the integral using one sample of N random points.

We compare the results (accuracy and CPU time in seconds) for four methods: crude Monte Carlo, importance sampling, importance separation (IMS) and the proposed adaptive Monte Carlo (AMC). The results shown in the Table 1 and Figure 1 illustrate the superior behaviour of the considered in this paper methods - for twice more computational time (6 seconds), IMS gives approximately 10 times better accuracy than crude Monte Carlo and importance sampling. Let us mention also that the results obtained with the adaptive method and with importance separation are very similar when the sample size increases. A

Table 1. Comparison between Crude MCM, Importance sampling, Importance separation and Adaptive MCM. Input data: $M = 6, \varepsilon = 0.6, \delta = 0.1\text{e}-06$ (calculations are implemented on one processor).

N	Crude MCM		Imp. sampling		Imp. separation		Adaptive MCM	
	$\lvert I - I_N \rvert$	T_1	$\lvert I - I_N \rvert$	T_1	$\lvert I - I_N \rvert$	T_1	$\lvert I - I_N \rvert$	T_1
100	0.009532	0.001	0.081854	0.008	0.000316	6	0.001102	20
500	0.092960	0.004	0.007102	0.036	0.000003	31	0.000246	121
2500	0.009027	0.020	0.006381	0.175	0.000068	152	0.000131	587
10000	0.006611	0.076	0.004673	0.697	0.000061	610	0.000036	2390
50000	0.008443	0.386	0.003212	3.489	0.000021	3047	0.000009	12093

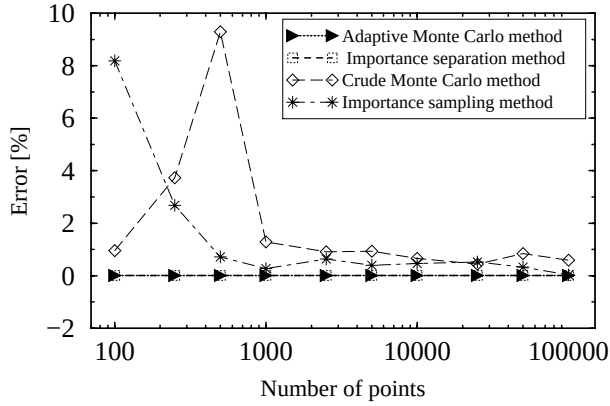

Fig. 1. Comparison of the accuracy of Crude MCM, Importance sampling, Importance separation and Adaptive MCM.

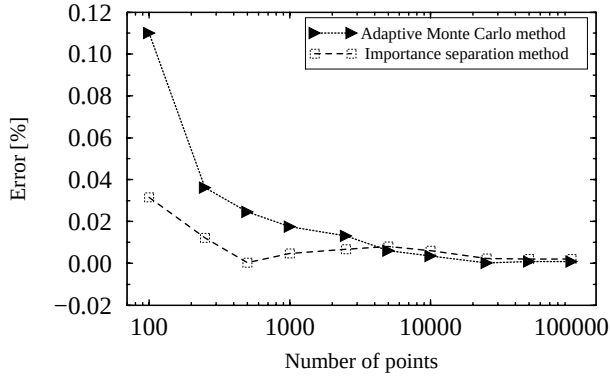

Fig. 2. Comparison of the accuracy of Importance separation and Adaptive MCM.

comparison of these two methods is given in Figure 2. It can be seen that importance separation method needs less points than adaptive method to achieve the desired accuracy. The reason for these results is the additional partitioning of the domain for adaptive method using only a posteriori information about the variance. The Table 2 presents the achieved efficiency of the parallel imple-

Table 2. Implementation of Adaptive MCM and Importance separation using MPI ($I = 2.923651$).

Adaptive MCM						Importance separation					
$N = 1000$			$N = 10000$			$N = 1000$			$N = 10000$		
p	I_N	E_p	p	I_N	E_p	p	I_N	E_p	p	I_N	E_p
1	2.923476	1	1	2.923648	1	1	2.923604	1	1	2.923590	1
2	2.923631	0.970	2	2.923611	0.974	2	2.923603	0.979	2	2.923573	0.985
3	2.920495	0.722	3	2.923380	0.948	3	2.920636	0.967	3	2.923336	0.983
4	2.923608	0.610	4	2.923628	0.847	4	2.923804	0.941	4	2.923638	0.980
5	2.923557	0.523	5	2.923644	0.909	5	2.923463	0.934	5	2.923602	0.979
6	2.912064	0.204	6	2.922495	0.841	6	2.911825	0.925	6	2.922537	0.977

mentation (using MPI) for both methods, importance separation and adaptive Monte Carlo. The speed-up of the same two methods with respect to the number of processors is shown on Figure 3. The achieved speed-up is almost linear and

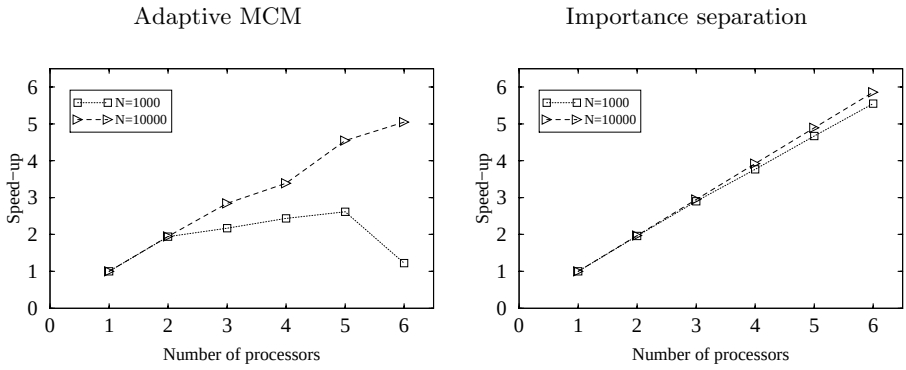

Fig. 3. Speed-up with respect to the number of processors.

the efficiency grows with the increase number of points. When the number of processors is large, but CPU time (even on one processor) is small, the efficiency is not very good because most of CPU time is used for communications between processors. An additional explanation of the efficiency of the adaptive method is

that the "master" carries out some preliminary calculations (computing of the limits of two new subregions obtained after division).

5 Conclusion and Future Directions

In this paper we present and study the parallel properties of importance separation and an adaptive Monte Carlo algorithms. We compare the results with crude Monte Carlo and with importance sampling. Our algorithms show some advantages.

While the results of applying our methods for approximate calculation of integrals are very satisfying, we would like to extend these methods for solving integral equations. We already started working on these algorithms.

References

1. Bahvalov, N. S.: On the optimal estimations of convergence of the quadrature processes and integration methods. Numerical Methods for Solving Differential and Integral Equations, Nauka, Moscow (1964) 5–63 (in Russian).
2. Berntsen, J., Espelid, T. O., and Genz, A.: An adaptive algorithm for the approximate calculation of multiple integrals. ACM Trans. Math. Softw., 17 (1991) 437–451.
3. Bull, J. M. and Freeman, T. L.: Parallel globally adaptive quadrature on the KSR-1. Advances in Comp. Mathematics, 2 (1994) 357–373.
4. Bull, J. M. and Freeman, T. L.: Parallel algorithms for multi-dimensional integration. Parallel and Distributed Computing Practices, 1(1) (1998) 89–102.
5. Caflisch, R. E.: Monte Carlo and quasi-Monte Carlo methods. Acta Numerica, 7 (1998) 1–49.
6. van Dooren, P. and de Ridder, L.: An adaptive algorithm for numerical integration over an N-dimensional cube. Journal of Computational and Applied Mathematics, 2 (1976) 207–217.
7. Freeman, T. L. and Bull, J. M.: A comparison of parallel adaptive algorithms for multi-dimensional integration. Proceedings of 8th SIAM Conference on Parallel Processing for Scientific Computing (1997)
8. Hesterberg, T.: Weighted average importance sampling and defensive mixture distributions. Technometrics, 37(2) (1995) 185–194.
9. Karaivanova, A.: Adaptive Monte Carlo methods for numerical integration. Mathematica Balkanica, 11 (1997) 391–406.
10. Karaivanova, A. and Dimov, I.: Error analysis of an adaptive Monte Carlo method for numerical integration. Mathematics and Computers in Simulation, 47 (1998) 201–213.
11. Owen, A. and Zhou, Y.: Safe and effective importance sampling. Technical report, Stanford University, Statistics Department (1999)
12. Sobol', I. M.: Monte Carlo Numerical Methods. Nauka, Moscow (1973) (in Russian).
13. Veach, E. and Guibas, L. J.: Optimally combining sampling techniques for Monte Carlo rendering. Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH '95 (1995) 419–428.
14. Veach, E.: Robust Monte Carlo Methods for Light Transport Simulation. Ph.D. dissertation, Stanford Universty (1997)