Monte Carlo Testing and Verification of Numerical Algorithm Implementations

David D. Pokrajac[1], Abdullah-Al-Zubaer Imran[1], Predrag R. Bakic[2]

Abstract – **We develop a statistical test to assess correctness of a numerical algorithm implementation. We propose a Monte Carlo method to estimate the accuracy of an approximation algorithm without knowing a true value to be computed. The methodology is illustrated on computation of partial volumes in breast tissue simulation.**

Keywords – **Software testing, Monte Carlo, Partial volume.**

I. Introduction

Validation of numerical software plays an important role in software development cycle. For the purpose of this study, we consider validation to consist of: 1) software testing; and 2) software verification. Software testing [1] is a set of methods utilized to determine whether the algorithm in quest is correctly implemented. Software verification comprises techniques that can determine the adequacy of the developed algorithm to a task in quest. Software testing is frequently performed by providing a limited set of test cases with known outputs to the implementation. However, obvious drawback of this approach is difficulty to examine a variety of potential inputs to software and need to evaluate the test cases either manually or using an existing implementation of another algorithm for the same task. In software verification, it is often of interest to determine the accuracy of an approximation algorithm; here, techniques of numerical analysis may provide error bounds, but the bounds may apply only to a limited class of inputs, or the bounds may be loose or only of theoretical value [2]. Alternatively, validation can be performed empirically. However, when using this approach, an issue is that the accurate solution to the approximated problem is not available.

In this study, we propose the application of Monte Carlo approach [3] for validation of a class of numerical software. The method is developed for a class of multiple integral computation problems and demonstrated on a related problem of partial volume computation [4]. After the statement of the problem and preliminary considerations in Section II, in Section III we develop a statistic that has standard normal distribution asymptotically when an algorithm implementation is correct.

[1]David Pokrajac and Abdullah-Al-Zubaer Imran are with Delaware State University, 1200 N DuPont Hwy, Dover, DE, US, Email: {dpokrajac@desu.edu, aimran14@students.desu.edu}.

[2]Predrag Bakic is with University of Pennsylvania, Department of Radiology, 3400 Spruce Street, Philadelphia, PA, US, Email: predrag.bakic@uphs.upenn.edu

In Section IV, we demonstrate the estimate of the approximation error and the bounds for the standard deviation of the estimate. In Section V we discuss the application of the proposed method on validation of partial volume computation. Section VI contains discussion and conclusive remarks.

II. Preliminaries

Consider a function $\boldsymbol{f(x_1, x_2, \dots, x_{k-1})}$ that depends on a tuple p of random parameters, and let $\boldsymbol{0 \le f(x_1, x_2, \dots, x_{k-1}) \le 1, (x_1, x_2, \dots, x_{k-1}) \in D, D = [0,1]^{k-1}}$. For T realizations of random parameters p_i, $i=1,\dots,T$ and corresponding functions, $\boldsymbol{f_i(x_1, x_2, \dots, x_{k-1})}$, our goal is to calculate integrals:

$$\boldsymbol{I_i = \iint_D f_i(x_1, x_2, \dots, x_{k-1})\, dx_1 dx_2 \dots dx_{k-1}, i = 1, \dots, T.} \tag{1}$$

Note that the integrals, Eq. (1), can be treated as random values described by a probability density function $\boldsymbol{pdf(I_i)}$.

Assume that integrals, Eq. (1), can be approximated as:

$$\boldsymbol{I_{a,i} = \iint_D g_i(x_1, x_2, \dots, x_{k-1})\, dx_1 dx_2 \dots dx_{k-1}, i = 1, \dots, T,} \tag{2}$$

Where $\boldsymbol{g_i(x_1, x_2, \dots, x_{k-1}) = F(f_i(x_1, x_2, \dots, x_{k-1}))}$ are suitable chosen functions such that the exact computation of Eq. (2) is feasible.

Note that an integral $\boldsymbol{I_i}$ from Eq. (1) can also be approximately computed using the Monte Carlo approach as follows: a) Uniformly sample N_{MC} independent points $\mathbf{x_j} = \boldsymbol{(x_{j,1}, x_{j,2}, \dots, x_{j,k}) \in [0,1]^k, j = 1, N_{MC}}$; b) Determine:

$$\boldsymbol{N_i = \left|\{\mathbf{x}_j | x_{j,k} \le f_i(x_{j,1}, x_{j,2}, \dots, x_{j,k-1}), j = 1, \dots, N_{MC}\}\right|}; \tag{3}$$

c) Compute an approximation as:

$$\boldsymbol{I_{MC,i} = \frac{N_i}{N_{MC}}}. \tag{4}$$

Note that, for a randomly chosen $\mathbf{x_j} \in \boldsymbol{[0,1]^k}$, the probability that $\boldsymbol{x_{j,k} \le f_i(x_{j,1}, x_{j,2}, \dots, x_{j,k-1})}$ is equal to $\boldsymbol{I_i}$. Hence, a random variable $\boldsymbol{N_i}$ follows a Binomial distribution with expectation $\boldsymbol{N_{MC} I_i}$ and variance $\boldsymbol{N_{MC} I_i (1 - I_i)}$ [5]. If we define:

$$\boldsymbol{\varepsilon_{MC,i} = I_{MC,i} - I_i}, \tag{5}$$

the random variables $\varepsilon_{MC,i}$ have the following conditional moments:

$$E(\varepsilon_{MC,i}|I_i) = 0, \tag{6}$$

$$E(\varepsilon^2_{MC,i}|I_i) = \frac{I_i(1-I_i)}{N_{MC}}. \tag{7}$$

Let's further define:

$$\varepsilon_i = I_{MC,i} - I_{a,i}, \tag{8}$$

$$\varepsilon_{A,i} = I_i - I_{a,i}, \tag{9}$$

and consider an ensemble of functions f_i. It is obvious that $\varepsilon_{A,i} = \varepsilon_A(I_i)$. Hence, we can write:

$$E(\varepsilon_{MC}\varepsilon_A) = \int E(\varepsilon_{MC,i}|I_i) \cdot \varepsilon_A(I_i) pdf(I_i) dI_i = 0, \tag{10}$$

where the expectation is taken through random realizations of $\varepsilon_{MC,i}$ and the ensemble of functions, and $pdf(I_i)$ is a probability density function of the true value of integral. Observe that, from Eq. (5), (8) and (9) follows:

$$\varepsilon_{A,i} = \varepsilon_i - \varepsilon_{MC,i}, \tag{11}$$

and, due to Eq. (10):

$$E(\varepsilon^2_A) = E(\varepsilon^2) - E(\varepsilon^2_{MC}). \tag{12}$$

Note also that due to Eq. (7),

$$E(\varepsilon^2_{MC}) = \int \frac{I_i(1-I_i)}{N_{MC}} pdf(I_i) dI_i = \frac{E(I) - E(I^2)}{N_{MC}}. \tag{13}$$

Following the procedure from [4] we can obtain:

$$E(\varepsilon^2_{MC}) = \frac{1}{N_{MC}-1}\left(E(I_{MC}) - E(I^2_{MC})\right). \tag{14}$$

III. Software Testing

Consider now an important case when $f_i = g_i$, i.e., when f_i belong to a class of functions for which the approximation error of Eq. (2) is zero. Hence,

$$I_i = I_{a,i}, \tag{15}$$

$E(\varepsilon^2_A) = 0$ and (see Eq. 12):

$$E(\varepsilon^2) = E(\varepsilon^2_{MC}). \tag{16}$$

Further, assume that realizations I_i are independent. In this case, due to Eq. (15), from Eq. (13),

$$E(\varepsilon^2_{MC}) = \frac{E(I_a) - E(I_a^2)}{N_{MC}}. \tag{17}$$

Consider random variables *X* and *Y* defined as follows:

$$X = \frac{1}{T}\sum_{i=1}^{T} \varepsilon_i^2, \tag{18}$$

$$Y = \frac{1}{N_{MC}}\left(\frac{1}{T}\sum_{i=1}^{T} I_{a,i}(1 - I_{a,i})\right). \tag{19}$$

Due to Lindberg-Levy Central Limit Theorem [6], for large enough T, $X - \mathrm{E}(\varepsilon^2)$ has approximately normal distribution with zero mean and variance:

$$\sigma_X^2 = \frac{\sigma^2(\varepsilon^2)}{T}. \tag{20}$$

Similarly, for large enough T, $Y - E(\varepsilon^2_{MC})$ has approximately normal distribution with zero mean and variance:

$$\sigma_Y^2 = \frac{\sigma^2(I_a(1-I_a))}{T N_{MC}{}^2}. \tag{21}$$

Therefore, a random variable

$$Z = X - Y = \frac{1}{T}\sum_{i=1}^{T} \varepsilon_i^2 - \frac{1}{N_{MC}}\left(\frac{1}{T}\sum_{i=1}^{T} I_{a,i}(1 - I_{a,i})\right), \tag{22}$$

is asymptotically Gaussian, with zero mean and variance

$$\sigma_Z^2 = \sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}, \tag{23}$$

where σ_{XY} is a covariance defined [7] as:

$$\sigma_{XY} = E(X)E(Y) - E(XY). \tag{24}$$

Note that due to Eq. (16) and Eq. (19) we can write:

$$E(X) = E(Y) = \frac{1}{N_{MC}T} E(U), \tag{25}$$

where

$$U \equiv \sum_{i=1}^{T} (I_{a,i}(1 - I_{a,i})). \tag{26}$$

On the other hand, a conditional expectation of product *XY*, *given* values of $I_{a,i}, i = 1, \dots T$ is:

$$E(XY|I_{a,i}, i = 1, \dots T) =$$

$$\frac{1}{N_{MC}T^2}\left(\sum_{i=1}^{T} I_{a,i}(1 - I_{a,i})\right) E\left(\sum_{i=1}^{T} \varepsilon_i^2 \,|I_{a,i}, i = 1, \dots T\right) =$$

$$\frac{1}{N_{MC}T^2}\left(\sum_{i=1}^{T} I_{a,i}(1 - I_{a,i})\right)\left(\sum_{i=1}^{T} E(\varepsilon_i^2|I_{a,i})\right). \tag{27}$$

Due to Eq. (16), (15), (7) and (26) we finally obtain:

$$E(XY|I_{a,i}, i = 1, \dots T) = \frac{1}{N_{MC}^2 T^2} U^2. \tag{28}$$

From Eq. (28) directly follows:

$$E(XY) = \frac{1}{N_{MC}^2 T^2} E(U^2). \tag{29}$$

By combining Eq. (24), (25) and (29) we obtain:

$$\sigma_{XY} = \frac{1}{N_{MC}^2 T^2}[E^2(U) - E(U^2)] = -\frac{1}{N_{MC}^2 T^2}\sigma^2(U). \tag{30}$$

Note, however that $I_{a,i}$ are independent and identically distributed. Hence:

$$\sigma^2(U) = \sum_{i=1}^{T} \sigma^2\left(I_{a,i}(1 - I_{a,i})\right) = T\sigma^2\left(I_a(1 - I_a)\right), \tag{31}$$

and, from Eq. (30):

$$\sigma_{XY} = -\frac{1}{N_{MC}^2 T}\sigma^2\left(I_a(1 - I_a)\right). \tag{32}$$

By combining Eqs. (20), (21), (23) and (32) we obtain:

$$\boldsymbol{\sigma_Z^2} = \frac{\sigma^2(\varepsilon^2)}{T} + \frac{3\sigma^2(I_a(1-I_a))}{T{N_{MC}}^2}. \quad (33)$$

Assume that a numerical algorithm performing integration Eq. (2) is implemented. Consider a set of functions f_i , $i,\ldots,T$ where the algorithm provides an exact solution. Under H_0 that the algorithm is *correctly* implemented, for large enough T, random variable Z defined by Eq. (22) has approximately Gaussian distribution with variance defined by Eq. (33). Hence, we can compute test statistic Z^* as:

$$\boldsymbol{Z^*} = \frac{Z}{s_Z}, \quad (34)$$

where s_Z is an estimate of σ_Z :

$$s_Z = \sqrt{\frac{s^2(\varepsilon^2)}{T} + \frac{3s^2(I_a(1-I_a))}{T{N_{MC}}^2}}, \quad (35)$$

and s denotes sample standard deviation (e.g, $s(\varepsilon^2) = \sqrt{\frac{1}{T-1}\sum_{i=1}^{T}\left(\varepsilon_i^2 - \overline{\varepsilon^2}\right)^2}$ where $\overline{\varepsilon^2} = \frac{1}{T}\sum_{i=1}^{T}\varepsilon_i^2$).

We use two-sided test with the p-value calculated as:

$$p - value = \frac{2}{\sqrt{2\pi}}\int_{|Z^*|}^{\infty} e^{-\frac{t^2}{2}dt}, \quad (36)$$

and H_0 is rejected if the $p - value$ is smaller than a preset threshold (e.g., 0.005).

III. Software Verification

Consider random variables X (see Eq. (18)) and Y' defined as follows:

$$Y' = \frac{1}{N_{MC}-1}\left(\frac{1}{T}\sum_{i=1}^{T} I_{MC,i}\left(1 - I_{MC,i}\right)\right). \quad (37)$$

It is obvious that X and Y' represent unbiased estimates for $\boldsymbol{E}(\boldsymbol{\varepsilon^2})$ and $\boldsymbol{E}(\boldsymbol{\varepsilon_{MC}^2})$ based on observed T realizations of functions $\boldsymbol{f_i}$.

Due to Lindberg-Levy Central Limit Theorem [6], $Y' - E(\varepsilon_{MC}^2)$ has approximately normal distribution with zero mean and variance $\sigma_{Y'}^2 = \frac{\sigma^2(I_{MC}(1-I_{MC}))}{T(N_{MC}-1)^2}$. (Note that σ^2 here denotes a variance of a random variable.) Therefore, and due to Eq. (12), a random variable Z′ defined as:

$$\mathrm{Z'} = \mathrm{X} - \mathrm{Y'}$$
$$= \frac{1}{T}\sum_{i=1}^{T}\varepsilon_i^2 - \frac{1}{N_{MC}-1}\left(\frac{1}{T}\sum_{i=1}^{T} I_{MC,i}\left(1 - I_{MC,i}\right)\right), \quad (38)$$

is asymptotically Gaussian, with mean equal to $\boldsymbol{E}(\boldsymbol{\varepsilon_A^2})$ and variance $\boldsymbol{\sigma_Z^2}$, which square root is bounded as:

$$\frac{1}{\sqrt{T}}\left|\boldsymbol{\sigma}(\boldsymbol{\varepsilon^2}) - \frac{\boldsymbol{\sigma}(\boldsymbol{I_{MC}}(\mathbf{1}-\boldsymbol{I_{MC}}))}{\boldsymbol{N_{MC}}-\mathbf{1}}\right| \le \boldsymbol{\sigma_{Z'}} \le \frac{1}{\sqrt{T}}\left(\boldsymbol{\sigma}(\boldsymbol{\varepsilon^2}) + \frac{\boldsymbol{\sigma}(\boldsymbol{I_{MC}}(\mathbf{1}-\boldsymbol{I_{MC}}))}{\boldsymbol{N_{MC}}-\mathbf{1}}\right). \quad (39)$$

Hence, Z', as defined by Eq. (38) is a consistent estimate of $\boldsymbol{E}(\boldsymbol{\varepsilon_A^2})$. The boundaries for standard deviation of the estimate can be obtained from Eq. (39) when a squared root $\boldsymbol{\sigma}$ of variance is estimated using a sample standard deviation s [8] as:

$$\boldsymbol{s_{Z',min}} \le \boldsymbol{s_{Z'}} \le \boldsymbol{s_{Z',max}}, \quad (40)$$

where:

$$\boldsymbol{s_{Z',min}} = \frac{1}{\sqrt{T}}\left|\boldsymbol{s}(\boldsymbol{\varepsilon^2}) - \frac{\boldsymbol{s}(\boldsymbol{I_{MC}}(\mathbf{1}-\boldsymbol{I_{MC}}))}{\boldsymbol{N_{MC}}-\mathbf{1}}\right|, \quad (41)$$

$$\boldsymbol{s_{Z',max}} = \frac{1}{\sqrt{T}}\left(\boldsymbol{s}(\boldsymbol{\varepsilon^2}) + \frac{\boldsymbol{s}(\boldsymbol{I_{MC}}(\mathbf{1}-\boldsymbol{I_{MC}}))}{\boldsymbol{N_{MC}}-\mathbf{1}}\right). \quad (42)$$

IV. Practical Application

The methods discussed in Sections III and IV are applicable whenever a set of values $I_i \in [0,1], i = 1,\ldots,T$ can be approximated using the Monte Carlo approach (Section II and Eq. (4)) where N_i follows a Binomial distribution with expectation $N_{MC}I_i$ and variance $N_{MC}I_i(1 - I_i)$. In this section, we discuss their use in partial volume computation [4;9;10].

The problem of partial volume computation can be described as follows: Given a three-dimensional unit voxel $[0, 1]^3$ and surfaces $S_{i,l}$, $l=1,\ldots,s$, find a measure I_i of volume bounded by the surfaces and the voxel boundaries (a partial volume). In [4], cases when s=1 or s=2 are discussed. Surfaces $S_{i,l}$ are approximated using planes $P_{i,l}$, l=1,2 and values I_i are approximated by measures $I_{a,i}$ of volumes bounded by the planes and the voxel boundaries. Note also that I_i can be estimated using the Monte Carlo approach by randomly placing N_{MC} points inside the unit voxel and counting the fraction, Eq. (4), of the number of points N_i fitting into a partial volume. The surfaces S_i are depend on parameters which can be considered random. It is easy to observe that the partial volume computation as defined here satisfies the assumptions from Sections II-IV.

Following the approach from Section III, we tested implementations of Algorithm A.3 from [4]. We randomly generated T pairs of planes $P_{i,l}$, l=1,2 and calculated measures $I_{a,i}$ using the implementation. We also estimated $I_{MC,i}$ using N_{MC} and subsequently utilized Eqs. (22), (34)—(36) to test correctness of the implementation. The initial implementation was tested using T=10,000, $N_{MC} = 10{,}000$. The test resulted in $Z = 2.847e-04, s_Z = 2.4383e-05$ and *p-value*=1.6325e-31. Hence H_0 (that this implementation was correct) was rejected. Examination of the histogram of obtained values ε_i (Eq. (8)) indicated cases when the implementation did not work correctly. The subsequent (debugged) implementation was tested with a range of combinations of T and N_{MC}. The results, Table 1, consistently indicate that H_0 cannot be rejected (*p-value*>0.2) which suggests the correctness of this implementation.

With the approach from Section IV, we validated Algorithm A.2 from [4] using $N_{MC} = 63$, for voxels that contain simulated skin and ligaments/compartmental tissue, see Table 2. We recalculated Z' (Eq. (38)), $s_{Z',min}$ (Eq. (41)) and $s_{Z',max}$ (Eq. (42)) for simulated breast phantom data from Table IV [4]. The comparison of Z' with the corresponding values of sample means MSE_A [4] shows that | Z'- MSE_A|<$s_{Z',min}$ which indicates that $s_{Z',min}$ is accurately computed. Further, the difference between the approximate quantization error MSE_q=2.09e-5 (as calculated in [4]) and Z'=1.952e-05 is smaller than corresponding $s_{Z',min}$ (1.622e-06). This justifies a hypothesis from [4] that the discrepancy between MSE_q and Z' can be explained by statistical fluctuations related to the Monte Carlo method.

TABLE I
COMPUTED *P-VALUES* FOR STATISTICAL TEST OF ALGORITHM FOR CORRECT IMPLEMENTATION OF ALGORITHM A.3 [4] OBTAINED FOR DIFFERENT COMBINATIONS OF T AND $\boldsymbol{N_{MC}}$

T	1e6	1e5	1e4	1e5	1e4	5e6	1e6	1e5
N_{MC}	1e5	1e5	1e5	1e4	1e4	10	10	10
p-value	0.971	0.831	0.242	0.966	0.987	0.615	0.574	0.846

TABLE II
ESTIMATED APPROXIMATION ERROR (MEAN AND STANDARD DEVIATION BOUNDARIES) FOR ALGORITHM A.2 [4]: $\boldsymbol{N_{MC}} = \boldsymbol{63}$. MSE_A FROM [4] CORRESPONDING TO Z' IS INCLUDED FOR COMPARISON

Voxels containing	T	Z'	$s_{Z',min}$	$s_{Z',max}$	MSE_A
Skin	1,597,042	1.952e-05	1.622e-06	3.927e-06	2.01e-05
Ligaments and compartmental tissue	6,435,881	4.330e-04	1.191e-06	2.368e-06	4.32e-04

V. DISCUSSION AND CONCLUSIONS

We propose to utilize Monte Carlo method for software verification. We use Monte Carlo not to calculate *per se*, but to validate the calculation's performance using another method. Hence, the accuracy of the Monte Carlo approximation (that can be estimated using Eq. (14)) is of secondary importance.

We develop a statistic (Eq. (34)) that has a standard normal distribution under the hypothesis that an algorithm is implemented correctly. We demonstrated that the approach can be applied to a practical problem of testing numerical software for computation of partial volume. In this case, manual evaluation of test cases needed to test a complex algorithm is not feasible.

We utilize an important property that Monte Carlo and approximation errors are orthogonal (Eq. (10)) which results in the estimate of approximation error $\boldsymbol{E}(\boldsymbol{\varepsilon}_{\boldsymbol{A}}^{\boldsymbol{2}})$ (Eq. (38)). Assuming large enough number T of evaluations, we also demonstrated upper and lower bound for the standard deviation of the estimate (Eqs. (41), (42)). This distinguishes the proposed approach from other approaches that may provide only the point estimate of the approximation error. Note also that the proposed methods do not assume knowledge of correct values of the estimated variables I_i: instead, the knowledge of observable values $\boldsymbol{\varepsilon_i}$ and $I_{a,i}$ is sufficient. We demonstrated this approach on verification of software for partial volume computation [4]. We showed that the discrepancy between the theoretically minimal approximation error (due to quantization) and the approximation error estimated there can be explained by the standard deviation of the estimate.

The proposed methodology is developed for a relatively narrow class of multiple integration problems. However, as demonstrated on computation of partial volumes, the approach can be easily extended whenever the estimation using an analog of Eq. (4) is possible and where $\boldsymbol{N_i}$ follows binomial distribution (see Section II).

In Table I we demonstrated that the choice of N_{MC} does not seem to be of predominant importance for software testing. Work in progress includes quantitative investigation of influence of N_{MC} on estimation of $\boldsymbol{E}(\boldsymbol{\varepsilon}_{\boldsymbol{A}}^{\boldsymbol{2}})$.

ACKNOWLEDGEMENT

This project was supported by a grant from the National Institute of General Medical Sciences (P20 GM103446) from the National Institutes of Health. Also, the work was supported in part by the US National Institutes of Health (R01 grant #CA154444), the US Department of Defence Breast Cancer Research Program (HBCU Partnership Training Award #BC083639), the US National Science Foundation (CREST grant #HRD-0630388 and III grant # 0916690), and the US Department of Defence/Department of Army (45395-MA-ISP, #54412-CI-ISP, W911NF-11-2-0046).

REFERENCES

[1] G. J. Myers, *The Art of Software Testing*, New Jersey, John Wiley & Sons Inc, 2004.

[2] R. L. Burden, J. Douglas Faires, *Numerical Analysis*, Massachusetts, Brooks/Cole Cengage Learning, 2010.

[3] N. Metropolis, "The Beginning of the Monte Carlo Method", *Los Alamos Science*, Vol. 15, No. 12, pp. 125–130, 1987.

[4] F. Chen, P. R. Bakic, A. D.A. Maidment, S. T. Jensen, X. Shi, D. D. Pokrajac, "Description and Characterization of a Novel Method for Partial Volume Simulation in Software Breast Phantoms," *IEEE TMI, 2015*, in press, doi: 10.1109/TMI.2015.2424854.

[5] R. Y. Rubinstein, D. P. Kroese, *Simulation and The Monte Carlo Method*, New Jersey, Wiley Interscience, 2007.

[6] J. W. Lindeberg, "Eine Neue Herleitung des Exponentialgesetzes in der Wahrscheinlichkeitsrechnung", *Math. Z.*, Vol. 15, No. 1, pp. 211–225, 1922 (German). MR 1544569, http://dx.doi.org/10.1007/BF01494395.

[7] T.W. Anderson, *An Introduction to Multivariate Statistical Analysis*, New York, John Wiley & Sons, 1984.

[8] J. Devore, *Probability and Statistics for Engineering and the Sciences*, Massachusetts, Brooks/Cole Cengage Learning, 2012.

[9] F. Chen, D. Pokrajac, X. Shi, F. Liu, A.D.A. Maidment, P. Bakic, "Simulation of Three Materials Partial Volume Averaging in a Software Breast Phantom," *Proc. IWDM, pp. 149-156*, 2012.