Distribution of Number of Roots of Random Polynomial Equations in Small Intervals

E. Shmerling

***Abstract*—** The problem of finding the probability distribution of the number of zeros in some real interval of a random polynomial whose coefficients have a given continuous joint density function is considered. A new simulation algorithm for solving this problem is presented. The effectiveness of the presented algorithm for the case where the real interval is small is proved.

***Keywords*—**Random Polynomial, Monte Carlo Algorithm, Matlab, Simulation.

I. Introduction

ANALYSIS of the behavior of random polynomials and their zeros has been a subject of active research for several decades. Motivation for these studies and most of the early results in this field were collected and summarized in the monograph by Bharucha-Reid and Sambandham [1], where one can find applications in such varied fields as spectral analysis, statistics, filtering theory and economics. More recent results appear, for example, in [2],[3],[7]-[11] among others. Consider a random polynomial

$$F_n^*(\vec{a},\omega)=a_0(\omega)x^n+a_1(\omega)x^{n-1}+\ldots+a_n(\omega) \qquad (1)$$

of degree *n*, whose coefficients are real-valued random variables with given continuous joint probability density function $p(a_0,a_1,\ldots,a_n)$. An important problem is the study of the behavior of zeros of the random polynomial (1). A natural first step in the study of the behavior of such zeros is to estimate the distribution of the random variable $N(B,\omega)$, which is the number of zeros of the polynomial $F^*(\vec{a},\omega)$ that belong to some arbitrary interval $B=[l(B);r(B)]$ on the Euclidean space R. This problem has been widely studied and has been solved for some specific cases in which the coefficients $a_0(\omega),\ldots,a_n(\omega)$ are distributed according to some given law or satisfy certain specific conditions. For quadratic random polynomials the following result has been obtained and described in [9].

Theorem 1

The probability $P_2(l,g)$ that two zeros of $F_2(x,\omega)=a(\omega)x^2+b(\omega)x+c(\omega)$ belong to a real interval (*l*,*d*) equals

$$\int_0^{\infty}\int_{-a(l+d)}^{-2al}\int_{-al^2-bl}^{b^2/4a}p(a,b,c)dcdbda$$
$$+\int_0^{\infty}\int_{-2ad}^{-a(l+d)}\int_{-ad^2-bd}^{b^2/4}p(a,b,c)dcdbda$$
$$+\int_0^{\infty}\int_{-a(l+d)}^{2ad}\int_{b^2/4a}^{-ad^2-bd}p(a,b,c)dcdbda$$
$$+\int_{-\infty}^{0}\int_{-2al}^{-a(l+d)}\int_{b^2/4a}^{-al^2-bl}p(a,b,c)dcdbda$$

The probability $P_l(l,d)$ that exactly one zero of $F_2(x,\omega)$ belongs to an interval *(l,d)* equals

$$\int_{-\infty}^{\infty}\int_{-\infty}^{-al-ad}\int_{-al^2-bd}^{-ad^2-bd}p(a,b,c)dcdbda$$
$$+\int_{-\infty}^{\infty}\int_{-al-ad}^{\infty}\int_{-ad^2-bd}^{-al^2-bl}p(a,b,c)dcdbda$$

Analogous formulas have been obtained for random polynomials of degree *n>2*, all the coefficients of which equal zero except $a_0(\omega),a_1(\omega)$ and $a_n(\omega)$. It is shown that for this type of random polynomials the problem of defining the distribution of $N(B,\omega)$ can be reduced to calculating multiple integrals. In order to calculate these intervals recently developed highly efficient multiple integration algorithms and software routines based on Monte Carlo simulation can be utilized (some of these algorithms and routines are described in [4],[5],[6]). The classical method for estimating the probability that exactly *k* zeroes of $F(\vec{a},\omega)$ belong to *B*, that has been actively used during the last decades, includes the following steps:

- generating a set of random points $(a_0,a_1,\ldots,a_n)$, distributed according to $p(a_0,a_1,\ldots,a_n)$, the coordinates of which are the coefficients of the polynomial
- for each generated point, calculating the zeros of the corresponding polynomial and checking whether exactly *k* zeros belong to *B*.
- calculating the proportion of generated polynomials which have *k* zeros in *B*.

If a set is big enough, the proportion of generated polynomials which have *k* zeros in *B* is a sufficiently good estimate for the desired probability.

The algorithm requires generation of a big number of points in order to achieve the desired rate of accuracy of the estimate and is, therefore, time-expensive. On the other

Manuscript received February 21.02, 2012.

E. Shmerling, Department of Computer Science and Mathematics, Ariel University Center of Samaria, Science Park ,Ariel 44837, Israel. (e-mail: efraimsh@ariel.ac.il; efraimsh@yahoo.com).

hand, the classical method requires generating random vectors distributed according to the given joint probability density function of the coefficients, and such generators may not be available in some specific cases.

The above mentioned disadvantages of the classical method make the development of more effective algorithms highly desirable. This motivated us to develop a new simulation algorithm, which is more effective for the case where the interval *B* is small. The algorithm is presented in this article.

II. Description of the Algorithm

Let $N(B,\omega)$ designate the number of real zeros of a random polynomial of *n*-th order

$$F_n(\vec{a},\omega) = x^n + a_1(\omega)x^{n-1} + \ldots + a_n(\omega)$$

The problem of estimating *P(N(B,ω)=i)* with a given rate of accuracy *ε* can be reduced to calculating the probabilities *P(i,m,B)=P(N(B,ω)=i∩N(R,ω)=m)* that the polynomial has exactly *m* real zeros, from which exactly *i* belong to *B*.

Let N_1 designate a constant which satisfies the following condition: the probability ε_1 that $\max_{1\le i\le n}\{1+|a_i(\omega)|>N_1\}$ is negligible with respect to ε. Obviously, the probability that real and imaginary parts of all complex zeros of are smaller than N_1 in absolute value and the moduli of all real zeros of $F_n(\vec{a},\omega)$ are smaller than N_1 is greater than $1-\varepsilon_1$.

Let *Q(i,m,B)* denote the set consisting of all points $\vec{a}\in R^n$ such that the corresponding polynomial $F(\vec{a})$ has exactly *n-m* distinct complex zeros and exactly *m* distinct real zeros, from which exactly *i* belong to *B*. It can be easily shown that *Q(i,m,B)* consists of a finite number of non-intersecting semi-algebraic sets which can be properly defined, therefore

$$P(i,m,B) = \int Q(i,m,B)p\vec{a}d\vec{a} \tag{2}$$

All the polynomials $F(\vec{a}), \vec{a}\in Q(i,m,B)$ have *k* complex zeros $z_l, l=\overline{1,k}$ with positive imaginary parts $ip_l > 0$ and real parts $rp_l, l=\overline{1,k}$ such that $rp_1 < rp_2 < \ldots < rp_k$, and *m* real zeros $rr_1 < rr_2 \ldots < rr_n$ such that $rr_1 < rr_2 \ldots < rr_m$, *m=n-2k.*

We utilize the expression

$$a_j = (-1)^k \cdot S_j,\ j=\overline{1,n}, \tag{3}$$

where the polynomials S_j are elementary symmetric functions of the roots $x_1, x_2, \ldots, x_n$ of the equation $F(\vec{a})=0$. S_j designates the sum of C_n^j products of *j* factors x_i with distinct indices:

$$S_1 = x_1 + x_2 + \ldots + x_n,$$
$$S_2 = x_1x_2 + x_1x_3 + \ldots,$$
$$S_3 = x_1x_2x_3 + x_1x_2x_4 + \ldots,$$
$$\ldots$$
$$S_n = x_1x_2\ldots x_n.$$

Substituting $rp_{\frac{j-1}{2}} + i\cdot ip_{\frac{j-1}{2}}$ instead of x_j, where j is an odd integer less or equal to k, $rp_{\frac{j}{2}} + i\cdot ip_{\frac{j}{2}}$ instead of x_j, where *j* is an even integer less or equal to $k/2$, rr_j instead of x_j, where $2k<j<n$, in (3), we obtain expressions for a_j, $j=1$, n as polynomial functions $f_j(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))$ of variables $rp_1,\ldots,rp_k$, $ip_1,\ldots,ip_k$, $rr_1,\ldots,rr_m$. In order to calculate the integral (2), we utilize the following change of variables from the initial set of variables $\vec{a}=(a_1,a_2,\ldots,a_n)$ to a new set of variables

$$(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m)) = (rp_1, rp_2,\ldots,rp_k, ip_1,\ldots,ip_k, rr_1,\ldots,rr_m).$$

The integral (2) equals

$$\begin{aligned}&\int_D p(f_1(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))),\ldots,\\ &f_n(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m)).\\ &J(\vec{a},(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))).\\ &d(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m)),\end{aligned} \tag{4}$$

where $J(\vec{a},(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m)))$ designates the determinant of the matrix

$$\begin{pmatrix}
\frac{df_1(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(rp_1)} & \cdots & \frac{df_n(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(rp_1)}\\
\vdots & & \vdots\\
\frac{df_1(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(rp_k)} & \cdots & \frac{df_n(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(rp_k)}\\
\frac{df_1(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(ip_1)} & \cdots & \frac{df_n(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(ip_1)}\\
\vdots & & \vdots\\
\frac{df_1(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(ip_k)} & \cdots & \frac{df_n(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(ip_k)}\\
\frac{df_1(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(rr_1)} & & \frac{df_n(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(rr_1)}\\
\vdots & & \vdots\\
\frac{df_1(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(rr_m)} & & \frac{df_n(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m))}{d(rr_m)}
\end{pmatrix}$$

$d(\vec{a},(r\vec{p}(k), i\vec{p}(k), r\vec{r}(m)))$ designates

$$d(rr_m)\cdot d(rr_{m-1})\cdot\ldots\cdot d(rr_1)\cdot\ldots$$
$$\cdot d(ip_k)\cdot d(ip_{k-1})\cdot\ldots$$
$$\cdot d(ip_1)\cdot d(rp_m)\cdot d(rp_{m-1})\cdot\ldots\cdot d(ip_1),$$

the integration domain D is a union of $m-i+1$ sets $D_s \subset R^n$, $s=\overline{0,m-i}$,defined as

$$(-\infty;\infty)\times[rp_1;\infty)\times\ldots\times[rp_k;\infty)\times$$
$$\underbrace{[0;\infty)\times\ldots\times[0;\infty]}_{k\ \ times}\times[-\infty;l(B)]\times$$
$$[rr_1;l(B)]\times\ldots\times[rr_2;l(B)]\times\ldots\times[rr_{s-1};l(B)]$$
$$\times[l(B);r(B)]\times[rr_{s+1};r(B)]\times[rr_{s+2};r(B)]\times\ldots$$
$$\times[rr_{s+i-1};r(B)]\times[r(B);\infty)$$
$$\times[r_{s+i+1};\infty)\ldots\times[rr_{n-1};\infty).$$

Integral

$$\int_{D_s} p\left(\vec{f}\left(\vec{rp}(k),\vec{ip}(k),\vec{rr}(m)\right)\right)\cdot$$
$$J\left(\vec{rp}(k),\vec{ip}(k),\vec{rr}(m)\right)\cdot d\left(\vec{rp}(k),\vec{ip}(k),\vec{rr}(m)\right)$$
$$=\int_{D_s} Integrand\left(\vec{rp}(k),\vec{ip}(k),\vec{rr}(m)\right)\cdot$$
$$d\left(\vec{rp}(k),\vec{ip}(k),\vec{rr}(m)\right)$$

gives us the probability that $F_n(\vec{a},\omega)$ has exactly *m* real zeros, from which exactly *i* belong to *B* and exactly *s* belong to $(-\infty;l(B)]$.

Let's define bounded domains

$$D_s^* = [-N_1;N_1]\times[rp_1;N_1]\times\ldots$$
$$\times[rp_{k-1};N_1]\times\underbrace{[0;N_1]\times\ldots\times[0;N_1]}_{k\ \ times}\times$$
$$\times[-N_1;l(B)]\times[rr_1;l(B)]\times$$
$$\times[rr_2;l(B)]\times\ldots\times[rr_{s-1};l(B)]\times$$
$$\times[l(B);r(B)]\times[rr_{s+1};r(B)]\times$$
$$[rr_{s+2};r(B)]\times\ldots\times[rr_{s+i-1};r(B)]\times$$
$$[r(B);N_1]\times[rr_{s+i+1};N_1]\times\ldots\times$$
$$\times[rr_{n-1};N_1],$$
$$s=\overline{0,m-i},\ D^*=\bigcup_{s=0}^{m-i} D_s^*$$

Obviously, in view of the definition of N_1, the difference between the integral (4) over D and the integral over D^* is negligible with respect to ε, which enables one to calculate the integral over D^* instead of (4) in order to obtain an approximation with the desired rate of accuracy ε.

Since D^* is an extremely convenient domain from the point of view of generating random tuples uniformly distributed in it (such tuples can be generated utilizing standard uniform random number generator), classical Monte Carlo algorithm can be used for numerical calculation of the integral

$$\int_{D^*} Integrand\left(\vec{rp}(k),\vec{ip}(k),\vec{rr}(m)\right)\cdot$$
$$d\left(\vec{rp}(k),\vec{ip}(k),\vec{rr}(m)\right).$$

Let us formulate a theorem that proves the superior efficiency of the presented method for defining the distribution of the number of real zeros of random polynomials that belong to sufficiently small intervals.

Let $F_n(\vec{a},\omega)$ be a random polynomial with uniformly bounded continuous joint probability density function of its coefficients. Let $P(m,B,N_1)$ designate the probability that all the zeros of $F_n(\vec{a},\omega)$ are less than N_1 in absolute value and exactly *m* of them belong to a real interval $B=[l;r], -N_1 < l < r < N_1$.

Let $std\_pres(m,B,N_1,N^*)$ designate the standard deviation of the estimate of $P(m,B,N_1)$ based on the presented method, and let $std\_class(m,B,N_1,N^*)$ designate the standard deviation of the estimate of $P(m,B,N_1)$ based on the classical method. Both estimates are obtained by generating N^* random tuples. The following theorem is true.

Theorem 2 *For any positive number* g, $0<g<1$, *there exists a positive number* $l(g)$ *such that the following condition is satisfied: for any real interval in* $[-N_1;N_1]$, *the length of which* $len(B)$ *is less than* $l(g)$, *the ratio*

$$std\_pres(m,B,N_1,N^*)/\,std\_class(m,B,N_1,N^*)$$

is less than g.

Proof: The estimate *est_pres* of $P(m,B,N_1)$ based on the presented method is $\frac{1}{N^*}\sum_{i=1}^{N^*} Integrand\left(tupD^*(i)\right)$, where $tupD^*(i)$ designates a random tuple uniformly distributed in D^*, and the standard deviation of *est_pres* is calculated via formula

$$std\_pres(m,B,N_1,N^*)=$$
$$=\sqrt{\frac{1}{N^*}\sum_{i=1}^{N^*}\left(Integrand\left(tupD^*(i)\right)-est\_pres\right)^2}\cdot$$
$$\cdot V\left(D^*\right),$$

where $V\left(D^*\right)$ designates the volume of D^*. Assume that numbers $G_1>0$ and $G_2>0$ exist such that $G_1 < Integrand(d) < G_2$ for any $d\in D^*$.

We have the following inequality:

$$std\_pres(m,B,N_1,N^*)<\sqrt{\frac{1}{N^*}}\cdot G_2\cdot V\left(D^*\right).$$

On the other hand, we have the following formula for the standard deviation of the estimate *est_class* of $P(m,B,N_1)$ based on the classical method

$$std\_class(m,B,N_1,N^*)\approx$$
$$\approx\sqrt{\frac{1}{N^*}\cdot P(m,B,N_1)\cdot(1-P(m,B,N_1))}$$
$$>\sqrt{\frac{1}{N^*}\cdot G_1\cdot V(D^*)\cdot(1-P(m,B,N_1))},$$

which proves the theorem since

$$V(D^*)<(len(B))^m(2N_1^2)^k.$$

At $n=3$, we have a random polynomial $P_3(\vec{a},\omega)$ $x^3+a_1(\omega)x^2+a_2(\omega)x+a_3(\omega)$. Let's calculate the probability that the polynomial has a unique real zero belonging to the interval *B*=[l;r]. We have

$$f_1(rp,ip,rr)\equiv-(rr+2rp)$$
$$f_2(rp,ip,rr)\equiv(2rr\cdot rp+rp^2+ip^2)$$
$$f_3(rp,ip,rr)\equiv-rr(rp^2+ip^2)$$

The Jacobian of transformation has the form

$$J(\vec{a},(rp,ip,rr))=$$
$$=\det\begin{pmatrix}-2 & 2(rr+rp) & -2rr\cdot rp\\ 0 & 2ip & -2rr\cdot ip\\ -1 & 2rp & -(rp^2+ip^2)\end{pmatrix}=$$
$$=4rp^2\cdot ip+4ip^3+4ip\cdot rr^2-8rp\cdot ip\cdot rr$$
$$P_c=P(N(B,\omega)=1\cap N(R,\omega)=1)\approx$$
$$\int_{-N_1}^{N_1}\int_{l}^{N_1}\int_{l}^{r}p(-(2rp+rr),2rp\cdot rr+rp^2+$$
$$+ip^2,-rp^2\cdot rr-ip^2rr))\cdot(4rp^2\cdot ip+4ip^3$$
$$+4ip\cdot rr^2-8rp\cdot ip\cdot rr)d(rr)d(ip)d(rp)$$

For the case where we need to calculate the probability that random polynomial has 3 real zeros satisfying certain conditions, we have

$$f_1(rr_1,rr_2,rr_3)=-(rr_1+rr_2+rr_3)$$
$$f_2(rr_1,rr_2,rr_3)=rr_1rr_2+rr_1rr_3+rr_2rr_3$$
$$f_3(rr_1,rr_2,rr_3)=-rr_1rr_2rr_3$$

and the Jacobian of transformation $J(\vec{a},rr_1,rr_2,rr_3)$ has the form $(rr_3-rr_1)(rr_3-rr_2)(rr_2-rr_1)$. In order to calculate the probability that $P_3(\vec{a},\omega)$ has 3 real roots, one of which belongs to $[l;r]$, we have to summarize 3 probabilities: the probability P_0 that exactly one zero of $P_3(\vec{a},\omega)$ belongs to $[l;r]$ and 2 zeros are greater than *r*, the probability P_1 that one real zero of $P_3(\vec{a},\omega)$ is less than *l*, another is between *l* and *r*, and the third is greater than *r*, and the probability P_2 that $P_3(\vec{a},\omega)$ has 3 real zeros , one of which belongs to $[l;r]$ and two others are greater than *r*.

The probabilities P_0, P_1 and P_2 can be calculated via formulas

$$P_0=\int_{l}^{r}\int_{r}^{N_1}\int_{rr_1}^{N_1}p(-(rr_1+rr_2+rr_3),$$
$$rr_1\cdot rr_2+rr_1\cdot rr_3+rr_2\cdot rr_3, \tag{5}$$
$$-rr_1\cdot rr_2\cdot rr_3)\cdot(rr_3-rr_1)\cdot(rr_3-rr_2)$$
$$\cdot(rr_2-rr_1)d(rr_3)d(rr_2)d(rr_1)$$

$$P_1=\int_{-N_1}^{l}\int_{l}^{r}\int_{r}^{N_1}p(-(rr_1+rr_2+rr_3),$$
$$rr_1\cdot rr_2+rr_1\cdot rr_3+rr_2\cdot rr_3, \tag{6}$$
$$-rr_1\cdot rr_2\cdot rr_3)\cdot(rr_3-rr_1)\cdot(rr_3-rr_2)$$
$$\cdot(rr_2-rr_1)d(rr_3)d(rr_2)d(rr_1)$$

$$P_2=\int_{l}^{r}\int_{r}^{N_1}\int_{rr_2}^{N_1}p(-(rr_1+rr_2+rr_3),$$
$$rr_1\cdot rr_2+rr_1\cdot rr_3+rr_2\cdot rr_3, \tag{7}$$
$$-rr_1\cdot rr_2\cdot rr_3)\cdot(rr_3-rr_1)\cdot(rr_3-rr_2)$$
$$\cdot(rr_2-rr_1)d(rr_3)d(rr_2)d(rr_1)$$

In order to calculate the probability that $P_3(\vec{a},\omega)$ has 3 real roots, two of them belonging to $[l;r]$, we need to calculate the probability that 2 zeros belong to $[l;r]$ and one zero is less than *l* which equals

$$\int_{-N_1}^{l}\int_{l}^{N_r}\int_{rr_2}^{N_1}p(-(rr_1+rr_2+rr_3),$$
$$rr_1\cdot rr_2+rr_1\cdot rr_3+rr_2\cdot rr_3, \tag{8}$$
$$-rr_1\cdot rr_2\cdot rr_3)\cdot(rr_3-rr_1)\cdot(rr_3-rr_2)$$
$$\cdot(rr_2-rr_1)d(rr_3)d(rr_2)d(rr_1)$$

and the probability that $P_3(\vec{a},\omega)$ has 2 zeros in $[l;r]$ and one zero greater than *r* which equals

$$\int_{0}^{1}\int_{rr_1}^{1}\int_{1}^{N_1}p(-(rr_1+rr_2+rr_3),$$
$$rr_1\cdot rr_2+rr_1\cdot rr_3+rr_2\cdot rr_3, \tag{9}$$
$$-rr_1\cdot rr_2\cdot rr_3)\cdot(rr_3-rr_1)\cdot(rr_3-rr_2)$$
$$\cdot(rr_2-rr_1)d(rr_3)d(rr_2)d(rr_1)$$

The probability that $P_3(\vec{a},\omega)$ has 3 zeros belonging to $[l;r]$, can be found by calculating the following integral

$$\int_{l}^{r}\int_{rr_1}^{r}\int_{rr_2}^{r_1} p(-(rr_1+rr_2+rr_3),$$
$$rr_1\cdot rr_2+rr_1\cdot rr_3+rr_2\cdot rr_3, \qquad (10)$$
$$-rr_1\cdot rr_2\cdot rr_3)\cdot(rr_3-rr_1)\cdot(rr_3-rr_2)$$
$$\cdot(rr_2-rr_1)d(rr_3)d(rr_2)d(rr_1)$$

Thus the problem of defining the distribution of $N_3(B,\omega)$ is reduced to calculating the 6 integrals mentioned above. In order to calculate each of these integrals numerically via classical Monte Carlo method we need to generate a sequence of random tuples uniformly distributed in its integration domain. For P_2 the integration domain is integral defined by the system of double inequalities

$$\begin{cases} l < rr_1 < r \\ r < rr_2 < N_1 \\ rr_2 < rr_3 < N_1 \end{cases}$$

a sequence of random tuples

$$\vec{rr}(i) = (rr_1(i), rr_2(i), rr_r(i), \ \overline{i=1,N}$$

uniformly distributed in the domain can be sampled in the following way

$$\begin{cases} rr_1(i) = l + unif1(i)\cdot(r-l) \\ rr_2(i) = r + (1-\sqrt{unif2(i)}\cdot(N_1-r) \\ rr_3(i) = rr_2 + unif3(i)\cdot(N_1-rr_2) \end{cases},$$

where $unif1(i)$, $unif2(i)$, $unif3(i)$ $(i=1,n)$ designate numbers in [0;1] sampled by the standard uniform random number generator. The volume V of the integration domain equals $(N-r)^2\cdot(r-l)/2$. The Monte Carlo estimate for P_2 obtained via generating N^* tuples is given by

$$\frac{1}{N^*}\sum_{1}^{N^*} p(-(rr_1(i)+(rr_2(i)+$$
$$+(rr_3(i),(rr_1(i)\cdot rr_2+rr_1(i)\cdot rr_3+$$
$$+rr_2(i)\cdot rr_3(i),-rr_1(i)$$
$$\cdot rr_2(i)\cdot rr_3(i))\cdot(rr_3(i)-rr_1(i))\cdot$$
$$(rr_3(i)-rr_2(i))\cdot(rr_2(i)-rr_1(i))\cdot V.$$

Random tuples uniformly distributed in the domains defined by systems of inequalities

$$\begin{cases} l < rr_1 < r \\ rr_1 < rr_2 < r \\ r < rr_3 < N_1 \end{cases} \text{ and } \begin{cases} N_1 < rr_1 < l \\ rr_1 < rr_2 < l \\ l < rr_3 < r \end{cases}$$

can be sampled in an analogous way, integration domains defined by systems of inequalities

$$\begin{cases} -N_1 < rp < N_1 \\ l < ip < N_1 \\ l < rr < r \end{cases} \text{ and } \begin{cases} -N_1 < rr_1 < l \\ l < rr_2 < r \\ r < rr_3 < N_1 \end{cases}$$

are three-dimensional cubes, generating random tuples in which is trivial. The integration domain defined by the system

$$\begin{cases} l < rr_1 < r \\ rr_1 < rr_2 < r \\ rr_2 < rr_3 < r \end{cases}$$

is a three-dimensional simplex with vertices (*l*;*l*;*l*), (*l*;*l*;*r*), (*l*;*r*;*r*) and (*r*;*r*;*r*). It is well-known that generation of random tuples uniformly distributed in a simplex is simple and fast (see, for example [6]). In order to generate random tuple in a simplex with (*n*+1) vertices V_0, $V_1,\dots,V_n$ one has to sample uniformly *n* random numbers in (0,1), sort them in ascending order, thus obtaining *n* numbers $u_1,\dots,u_n$, and set $u_0=0, u_{n+1}=1$. The random tuple is given by

$$\sum_{i=0}^{n} V_i\cdot(u_{i+1}-u_1).$$

III. Results of Experimental Calculations

In order to illustrate the effectiveness of the presented method we obtained two estimates of the probability $P_1(l,r)$ that random polynomial of order 3 whose coefficients $a_1(\omega)$, $a_2(\omega)$, $a_2(\omega)$,are independent standard Gaussian random variables, has exactly one zero in certain intervals. One estimate was obtained by the developed software based on the presented method (function *est_pres_meth*), another was obtained by the developed software based on the classical one (function *est_class_meth*). The developed software enables one to calculate standard deviations of both estimates which makes it possible to compare the effectiveness of the classical method and the presented one. The function *est_pres_meth* calls four auxiliary functions:

- *unique_real_zero*

  which gives an estimate for P_c calculated via presented method and the standard deviation of the estimate
- *two_zeros_greater_r*

  which gives an estimate for P_0 calculated via presented method and the standard deviation of the estimate
- *two_zeros_less_l*

  which gives an estimate for P_2 calculated via presented method and the standard deviation of the estimate
- *one_zero_less_l*

  which gives an estimate for P_1 calculated via presented method and the standard deviation of the estimate

The MATLAB code of the programs with comments is presented in Appendix.

The results of experimental calculations are given in the following table.

Table 1

Interval B=[l;r]	Std_pres	Std_class	Ratio $\frac{Std\_pres}{Std\_class}$
[0; 0.5]	0.001904	0.00120	1.5866
[0; 0.25]	0.000857	0.000864	0.9918
[0; 0.1]	0.000332	0.000553	0.6003
[0; 0.05]	0.000168	0.001173	0.14358

The estimates were based on the same number of generated tuples (10^5). Results of experimental calculations presented in Table 1 illustrate that the ratio of standard deviations of estimates obtained via presented method and via classical one decreases as the length of interval *B* decreases, therefore the presented method is relatively more efficient than the classical one for small intervals and much more efficient for extremely small intervals.

IV. Conclusive Remarks

Let us stress the two main advantages of the presented method. The first advantage is its superior efficiency for small intervals proved in previous sections. The second advantage of the presented method is that it enables one to develop the software which can be implemented for counting the zeros of random polynomials with arbitrary joint probability density function of its coefficients, while software routines based on the classical method can't be universal since they have to incorporate a specific random number generator for each type of random polynomials. Program 2 described above with minor modifications can be utilized for defining the distribution of the number of real zeros in a given interval for any random polynomial of order 3 with continuous joint probability density function of its coefficients. Software based on the presented method for polynomials of higher order has to be developed, which is one of the directions of further work.

In order to utilize the presented method for defining the distribution of the number of real zeros of a random polynomial $F_n^*(\vec{a},\omega)$, all the coefficients of which are random variables, the algorithm should be slightly modified. The change of variables from $(a_0,...,a_n)$ to $(a_0,\vec{rp}(k),\vec{ip}(k),\vec{rr}(m))$ should be utilized for calculating integrals (2). The Jacobian of transformation equals the corresponding Jacobian for the case where the first coefficient equals one, multiplied by a_0^n. For example, we have the following formula for calculating the probability that $F_3^*(\vec{a},\omega)$ has one real zero belonging to $[l;r]$ and 2 complex zeros.

$$P(N(B,\omega)=1\bigcap N(R,\omega)\equiv 1)\approx$$
$$\int_{N(l)}^{N(r)}\int_{-N_1}^{N_1}\int_{0}^{N_1}\int_{l}^{r} p(a_0,-a_0(2rp+rr),$$
$$a_0(2rp\cdot rr+rp^2+ip^2),$$
$$-a_0\cdot rp^2\cdot rr-a_0\cdot ip^2\cdot rr)\cdot a_0^3(4rp^2\cdot ip+4ip^3$$
$$+4ip\cdot rr^2-8rp\cdot ip\cdot rr)d(rr)d(ip)d(rp)d(a_0),$$

where $N(l)$, $N(r)$, N_1 are constants which satisfy the condition: the probability ε_1 that at least one of the inequalities $N(l)<a_0(\omega)<N(r)$ and $\max\limits_{1\le i\le n}\{1+|a_i(\omega)|/|a_0(\omega)|\}<N_1$ is not satisfied is negligible with respect to ε.

Appendix

Function est_class_meth

```
function [p,std]=...
est_class_meth(l,r,n)
%input :
%l,r - left and right boundaries of the
%interval n - number of generated tuples
%output : p- estimate of proportion
% std - standard deviation of the estimate
counter=0;
for i=1:n
%generating the coefficients of the polynomial
pol=[1 [randn(1,3)]];
%finding the zeros of the polynomial
rt=roots(pol);
%finding the real zeros of the polynomial
rr= rt(imag(rt)==0);
%checking whether exactly one zero
%belongs to interval [l,r]
if sum(rr>l &rr< r)==1
counter=counter+1;
end
end
%calculating the proportion of generated
%polynomials having
%exactly one zero in [l,r]
p=counter/n;
%calculating the standard
%deviation of the estimate
std=sqrt(p*(1-p)/n);

function [m,std]=est_pres_meth(n,n1,l,r)
%input parameters :
%   n- number of tuples;
% n1-upper bound for the moduli of all the zeros
%   l,r- the left and right  bounds
%   of the interval
%output : m  - estimate of P1([l;r]),
%          std - standard deviation
 [m1,s1]=unique_real_zero(n,n1,l,r);

 [m2,s2]=two_zeros_greater_r(n,n1,l,r);
 [m3,s3]=two_zeros_less_l(n,n1,l,r);
 [m4,s4]=one_zero_less_l(n,n1,l,r);
% calculating estimate of P1([l;r])
 m=m1+m2+m3+m4;
%calculating standard deviation
 std=sqrt(s1^2+s2^2+s3^2+s4^2);
```

Function unique_real_zero

```
function [est_prob,est_std]=...
  unique_real_zero(n,n1,l,r)
%function for calculating unique real zero
%input: n-number of generated tuples,
% n1-upper bound for the moduli of all the zeros
% l,r-left and right bounds of interval
%output: est_prob,est_std - estimate of
% the probability
% and standard deviation of the estimate
 integrand1=[];

 for i=1:n
 %generating the coefficients of the polynomial
  rp=-n1+rand*(2*n1);
  ip=rand*n1;
  rr=l+rand*(r-l);
 %calculating the integrand
  integrand1=...
   [integrand1 (1/sqrt(2*pi))^3*...
  exp(-((rr+2*rp)^2+(2*rr*rp+rp^2+ip^2)^2...
  +(rr*rp^2+rr*ip^2)^2)/2)*...
  (4*rp^2*ip+4*ip^3+4*ip*rr^2-8*rp*ip*rr)];
 end
%calculating the volume
 v=2*n1^2*(r-l);
%calculating the estimate for Pc
 est_prob=mean(integrand1)*v;
%calculating sample standard deviation
 est_std=std(integrand1)*v/sqrt(n);
```

Function one_zero_less_l

```
function [est_prob,est_std]=...
        one_zero_less_l(n,n1,l,r)
%input: n-number of generated tuples,
% n1-upper bound for the moduli of all the zeros
% l,r-left and right bounds of interval
%output: est_prob,est_std - estimate of
% the probability
% and standard deviation of the estimate
 integrand2=[];
 for i=1:n
  %generating the coefficients of the polynomial
  rr1=-n1+rand*(l+n1);
  rr2=l+rand*(r-l);
  rr3=r+rand*(n1-r);
% calculating the integrand
  integrand2=...
   [integrand2 (1/sqrt(2*pi))^3*...
   exp(-((rr1*rr2*rr3)^2+...
   (rr1*rr2+rr1*rr3+rr2*rr3)^2+...
   (rr1+rr2+rr3)^2)/2)*(rr3-rr1)*...
   (rr3-rr2)*(rr2-rr1)];
 end
%calculating the volume
  v=(l+n1)*(r-l)*(n1-r);
%calculating the estimate for P1
  est_prob=mean(integrand2)*v;
%calculating standard deviation
  est_std=std(integrand2)*v/sqrt(n);
```

Function two_zeros_greater_r

```
function [est_prob,est_std]=...
    two_zeros_greater_r(n,n1,l,r)
%function for finding two real zeros
% greater than the right bound
%input: n-number of tuples,
% n1-upper bound for the moduli of all the zeros
% l,r-left and right bounds of interval
%output: est_prob,est_std - estimate of
% the probability
% and standard deviation of the estimate
 integrand2=[];
 for i=1:n
%generating the coefficients of the polynomial
   rr1=l+rand*(r-l);
   rr2=r+(1-sqrt(rand))*(n1-r);
   rr3=rr2+rand*(n1-rr2);
%calculating the integrand
   integrand2=...
   [integrand2 (1/sqrt(2*pi))^3*...
   exp(-((rr1*rr2*rr3)^2+(rr1*rr2+...
   rr1*rr3+rr2*rr3)^2+(rr1+rr2+rr3)^2)/2)*...
   (rr3-rr1)*(rr3-rr2)*(rr2-rr1)];
 end
%calculating the volume
 v=(n1-r)^2*(r-l)/2;
%calculating the estimate for P1
 est_prob=mean(integrand2)*v;
%calculating standard deviation
 est_std=std(integrand2)*v/sqrt(n);
```

Function two_zeros_less_l

```
function [est_prob,est_std]=...
       two_zeros_less_l(n,n1,l,r)
%input: n-number of generated tuples,
% n1-upper bound for the moduli of all the zeros
%    l,r-left and right bounds of interval
%output: est_prob,est_std - estimate of
% the probability
% and standard deviation of the estimate
integrand2=[]; for i=1:n
%generating the coefficients of the polynomial
 rr1=-n1+(1-sqrt(rand))*(l+n1);
 rr2=rr1+rand*(l-rr1);
 rr3=l+rand*(r-l);
%calculating the integrand
 integrand2=...
 [integrand2 (1/sqrt(2*pi))^3*...
 exp(-((rr1*rr2*rr3)^2+...
 (rr1*rr2+rr1*rr3+rr2*rr3)^2+...
 (rr1+rr2+rr3)^2)/2)*(rr3-rr1)*...
 (rr3-rr2)*(rr2-rr1)];
end
```

```
%calculating the volume
 v=(l+n1)^2*(r-l)/2;
%calculating the estimate for P2
 est_prob=mean(integrand2)*v;
%calculating standard deviation
 est_std=std(integrand2)*v/sqrt(n);
```

REFERENCES

[1] A.T. Bharucha-Reid and M. Sambandham, *Random Polynomials*, Academic Press, Orlando/London, 1986.

[2] A. Edelman and E. Kostlan, "How many zeros of a random polynomial are real?", *Bull. Amer. Math. Soc. (N.S)*, Vol 32,1995,pp. 1-37.

[3] Y. Castin, Z. Hadzibabic, S. Stock, J. Dalibard and S. Stringari, Quantized Vortices in the Ideal Bose Gas: A Physical Realization of Random Polynomials. *Phys. Rev. Lett.* 96,04005,2006.

[4] Elise de Doncker, Shujun Li and Karlis Kaugars. Error Distribution for Iterated Integrals, *Proceedings of the 10th WSEAS International Confenrence on Applied Mathematics*, Dallas, Texas, USA, November 2006, pp. 371-375.

[5] J. Kulhanec, R. Farana. Distributed simulation with SIPRO program. *Proceedings of the 7th WSEAS International Conference on Automatic Control, Modeling and Simulation, ACMOS'05*, Prague, Czech Republic, 2005, pp. 6-9.

[6] R.L Rubinshtein, D.P. Kroese, *Simulation and the Monte Carlo Method*, Wiley-Interscience, 2007.

[7] B. Shiffman and S. Zelditch, Equilibrium distribution of zeros of random polynomials, *International Math. Res. Notes*, Vol. 2003, pp. 25-49, 2003

[8] E. Shmerling and K.J. Hochberg, Asymptotic behavior of roots of random polynomial equations, *Proc. Amer. Math. Soc.*, Vol.130, 2002, pp. 2761--2770.

[9] E. Shmerling and K.J. Hochberg, Sturm's Method in Counting Roots of Random Polynomial Equations. *Methodology and Computing in Applied Probability*, No.6, 2004, pp. 203--218.

[10] E. Shmerling. Algorithm for Defining the Distribution of Zeros of Random Polynomials, *Proceedings of the 11-th WSEAS International Conference of Computers*, Vol. 4,2007, pp. 657--660.

[11] M. Sodin, Zeros of Gaussian analytic functions, *Math. Res. Lett.*, Vol.7, 2000, pp. 371-381.

[12] C. Stocker, S. Vey, A. Voigt. AMDiS-Adaptive multidimensional simulations: adaptive finite elements for complex domain. *Proceedings of the 7th WSEAS International Conference on Automatic Control, Modeling and Simulation, ACMOS'05*, Prague, Czech Republic, 2005, pp. 385-388