Statistical tolerance analysis based on good point set and homogeneous transform matrix

Huiwen Yan\textsuperscript{a}, Yanlong Cao\textsuperscript{a,b,*}, Jiangxin Yang\textsuperscript{a}

\textsuperscript{a} Key Laboratory of Advanced Manufacturing Technology of Zhejiang Province, college of Mechanical Engineering, Zhejiang University, Hangzhou, 310027, China
\textsuperscript{b} State Key Laboratory of Fluid Power and Mechatronic Systems, college of Mechanical Engineering, Zhejiang University, Hangzhou, 310027, China
* Corresponding author. Tel.: +86-571-87953198; fax: +86-571-87951145. E-mail address: sdcaoyl@zju.edu.cn

Abstract

Tolerance analysis is increasingly becoming an important tool for mechanical design, process planning, manufacturing and inspection. It provides a quantitative analysis tool for evaluating the effects of manufacturing variations on performance and overall cost of the final assembly. This paper develops a method for statistical tolerance analysis. Firstly, \(n\) samples are generated based on the good point set according to the distribution of each dimension. Then, for every sample, one sample of functional requirement can be computed according to the assembly constraint equations. In this paper, the homogeneous transform matrix is introduced to describe the assembly function and Newton-Raphson iterative procedure is adopted to solve the assembly constraint equations after the iterative formula is deduced. Finally, the first four order moments of function requirements can be computed after \(n\) samples of function requirements are gotten. A case study is carried out to validate the proposed method at the end of this paper.

1. Introduction

Tolerance analysis aims at not only improving the performance of product, but also reducing the cost of product. Many well-known models or methods are introduced in several review literatures for tolerance analysis \[1, 2\]. Jiang et al.\[3\] describe the use of tolerance maps (T-Maps) (patent no. 6963824) and manufacturing maps(M-maps) to establish analytical relationship among all relevant design and machining tolerances for transfer of cylindrical datum. Giordano et al. \[4\] apply Deviation Domains to axi-symmetric cases and thus reduce the space to three dimensions at the maximum instead of six in the general case. Ghie et al. \[5\] describe how one can use the same set of interval-based deterministic equations in a statistical context. Clément et al. \[6\] introduce a SDT (Small Displacement Torsor) model using six small displacements to represent the position and orientation of an ideal surface in relation to another ideal surface in a kinematic way. Desrochers et al. \[7\] put forward a unified Jacobian - Torsor model which combines the advantages of the torsor model and the Jacobian matrix. Anwer et al. \[8\] investigate the fundamentals of the skin model at a conceptual, geometric and computational level and present representation and simulation issues for product design. Gao et al. \[9\] introduce a DLM based on the first order Taylor’s series expansion of vector-loop-based assembly models which use vectors to represent either component dimensions or assembly dimensions. Statistical tolerance analysis takes statistical behaviors of manufacturing variations into account. Statistical tolerancing is a more practical and economical way of looking at tolerances and works on setting the tolerances in order to ensure a desired yield. For Statistical tolerancing, the methods or approaches include RSS method\[10\], system moments \[11\], reliability index \[12\], and Monte Carlo simulations\[13\]. Monte Carlo simulation is the simplest and the most popular method for nonlinear statistical tolerance analysis. Monte Carlo simulation offers a powerful analytical method for predicting the effects of manufacturing variations on design performance and production cost. However, the main
drawbacks of this method are that it is necessary to generate very large samples to assure calculation accuracy, and that the results of analysis contain errors of probability. To solve this problem, the so-called good point set is used to generate the samples of dimensions. Besides, the homogeneous transform matrix is introduced to describe the assembly function and Newton’s method is adopted to solve the assembly constraint equations after the iterative formula is deduced. The paper is organized as five parts. In the following part, dimension sample generation method is illustrated. The representation and solution techniques of assembly function are discussed in Section 3. In the Section 4, a case of one-way clutch assembly verifies the validity and accuracy of the method. The conclusions are summarized in the Section 5.

2. Dimension sample generation method

Usually, statistical tolerance analysis uses a relationship of the form $Y = f(X)$, where $Y$, the output variable, is the function requirement (characteristics such as gap or functional characteristics) of the assembly and $X = (X_1, X_2, ..., X_s)$ is the random vector representing the values of some characteristics (such as situation deviations or/and intrinsic deviations) of the individual parts or subassemblies making up the assembly. The random variables $X_i$ should have known probability density functions or known lower order moments.

The relationship can exist in any form for which it is possible to compute a value for $Y$, given set of values of $X_i$. It could be an analytic expression, explicit or implicit, or could involve complex engineering calculations, or conducting experiments, or running simulations. The input variables $X_i$ are continuous random variables which have cumulative distribution functions $F_i(x_i)$. In general, they could be mutually dependent.

Let $F(x)$ be the cumulative distribution function (cdf) of the random vector $X$. In generally, the form of $F(x)$ is very complicated so that it is difficult or impossible to calculate the cumulative distribution function of $Y$. Hence, we try to generate rep-points for $F(x)$.

2.1. Good point set

Let $\mathbf{y} = (y_1, y_s) \in \mathbb{C}^s$. If $\mathbf{P}_i$ forms the first $n$ points of

$\{(\lfloor y_i \rfloor, \ldots, \lfloor y_i \rfloor), k = 1, 2, \ldots\}$

(1)

where $\lfloor y_i \rfloor$ denotes the fractional part of $y_i$, with discrepancy $D(n, P_i) = o(n^{1+\epsilon})$ as $n \to \infty$; then the set $\mathbf{P}_i$ is called a good point set (gp set) and $\mathbf{y}$ a good point. For simplicity, the method using this set is called gp method \cite{14}. The following are some useful good points:

(a) Let $p_1, \ldots, p_s$ be the first $s$ primes. Take

$\mathbf{y} = (p_1, \ldots, p_s)$

(2)

(b) Let $p$ be a prime and $q = p^{1/(s+1)}$. Take

$\mathbf{y} = (q, q^2, \ldots, q^s)$

(3)

(c) Let $p$ be a prime and $p \geq 2s + 3$. Take

$\mathbf{y} = \{2\cos \frac{2\pi}{p}, 2\cos \frac{4\pi}{p}, \ldots, 2\cos \frac{2\pi s}{p}\}$

(4)

2.2. Inverse transformation method

After the good point set is gotten, various techniques, such as the inverse transformation method, the compositional method, the acceptance-rejection method and the conditional distribution method can be used for generating an observation from a given distribution.

Inverse transformation method, also known as inversion sampling, is a basic method for pseudo-random number sampling, i.e., for generating sample numbers at random from any probability distribution given its cumulative distribution function (cdf). The basic idea is to uniformly sample a number $\mu$ between 0 and 1, interpreted as a probability, and then return the largest number $x$ from the domain of the distribution $p(X)$ such that $p(-\infty < X < x) \leq \mu$. That is to say, if the cumulative distribution function of random variable $X$ is $F(x)$, then the largest number $x = F^{-1}(\mu)$.

The natural idea for generating rep-points $P = \{x_k, k = 1, \ldots, n\}$ for $F(x)$ is as follows:

(a) Generate a NT-net $\{e_k = (e_{k1}, \ldots, e_{kn}), k = 1, \ldots, n\}$ on $C^n$ employing gp method.

(b) Let $x_k = (F_1^{-1}(e_{k1}), \ldots, F_n^{-1}(e_{kn})), k = 1, \ldots, n$, where $F_i^{-1}(x_i)$ is the inverse function of $F_i(x_i)$.

Then we can use the rep-points $P = \{x_k, k = 1, \ldots, n\}$ to generate a sample of $Y$ according to the relationship $Y = f(X)$ and thus compute mean value and the tolerance range.

3. The representation and solution techniques of assembly function

For 2-D assemblies, each closed loop can result in three independent scalar equations, respectively representing the sum of the $x$ and $y$ projections of the dimensions in global coordinates and the sum of the relative rotations. The system of equations may be nonlinear and it is necessary to use nonlinear equations solving algorithm, including Newton’s Method. The key problem of this method is to get the iterative formula. The Jacobian matrix can easily help get the iterative formula, but it needs many calculations of partial derivatives. In order to avoid these calculations, the iterative formula is deduced through the first order Taylor expansion of the assembly function expressed in matrix form below.

For a closed loop, the transformation from joint $i-1$ to $i$ consists of a combination of one rotation and one translation matrix. It is assumed and reasonable that the translation is always along the local $x$ axis. For 2-D transformations, we have

$R(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

(5)
It is mathematically convenient to describe the operation of taking the derivative as a matrix product. Define the derivative of one rotation or one translation matrix as follows:

\[
\frac{dR(t)}{dt} = R(t)Q_1
\]  

(7) \[
\frac{dT(t)}{dt} = T(t)Q_2
\]  

(8)

where \(Q_1\) and \(Q_2\) are two derivative operators. In order to obtain the derivative operator, we form the derivative of the matrix by replacing each element in the matrix by its derivative with respect to the one variable of interest. This results in:

\[
\begin{bmatrix}
-\sin \theta & -\cos \theta & 0 \\
\cos \theta & -\sin \theta & 0 \\
0 & 0 & 0
\end{bmatrix}
\]  

(9) \[
\begin{bmatrix}
0 & 0 & 1 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]  

(10)

Substituting Eqs. (5) and (9) into Eq. (7), we get

\[
\begin{bmatrix}
0 & 1 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]  

(11)

Substituting Eqs. (6) and (10) into Eq. (8), we get

\[
\begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 1 \\
1 & 0 & 0
\end{bmatrix}
\]  

(12)

Now that taking derivatives of both rotation transformation matrix and translation matrix has been defined as matrix product, we are ready to take the derivative of the loop-closure equation. It is important to note that \(\theta\) is always relative to the prior vector. It is a positive angle when the rotational direction is anticlockwise. It is a negative angle, otherwise. With this convention, the assembly constraints can be expressed as a concatenation of homogeneous transformation matrices and the product of all the transformation matrices is equal to the identity matrix for a closed loop:

\[
R_i T_i R_{i-1} T_{i-1} \ldots R_2 T_2 R_1 T_1 I = I
\]  

(13)

Where \(R_i\) is the rotational transformation matrix at node \(i\), \(T_i\) is the translational matrix at node \(i\), \(R_f\) is the final rotational transformation matrix for bringing the loop to closure, \(I\) is the identity matrix.

Eq. (13) describes a series of rotations and translations to transform the local coordinates from vector-to-vector, until it has traversed the entire vector loop and returned to the starting point.

For convenience, we define:

\[
R_i = S_{r_{(2-2i+2i-1)}}
\]  

(14) \[
T_i = S_{r_{(2i-1)}}
\]  

(15) \[
R_f = S_{r_{(2n0)}}, s_{r_{0}}
\]  

(16)

According to the definitions above, Eq. (13) can be rewritten as:

\[
S_{s_{r_{0}}}, S_{r_{2}}, \ldots S_{r_{(2i-1)}}, S_{s_{r_{2}}}, \ldots S_{s_{r_{(2n-2i+2i-1)}}}, S_{r_{(2i-1)}} \ldots S_{s_{r_{(2i-1)}}}, S_{s_{r_{(2n-2i+2i-1)}}}, S_{s_{r_{0}}}, s_{r_{0}} = I
\]  

(17)

Through observation, Eq. (17) can be further simplified as:

\[
S_{s_{r_{0}}}, S_{r_{2}}, \ldots S_{r_{(2i-1)}}, S_{s_{r_{2}}}, \ldots S_{s_{r_{(2n-2i+2i-1)}}}, S_{s_{r_{0}}}, s_{r_{0}} = I
\]  

(18)

It is important to note that the right of the Eq. (18) contains only three unknown or assembly variables and that every unknown or assembly variable is contained in only one of the matrices making up the product. Define the right of the Eq. (18) as a function of the three unknown or assembly variables:

\[
S(q) = S_{s_{r_{0}}}, S_{r_{2}}, \ldots S_{r_{(2i-1)}}, S_{s_{r_{2}}}, \ldots S_{s_{r_{(2n-2i+2i-1)}}}, S_{s_{r_{0}}}, s_{r_{0}}
\]  

(19)

Where \(q = (q_1, q_2, q_3)\), \(q_i\) represents the unknown or assembly variable.

Because one variable is wholly contained in only one matrix, in the example shown, the variable denoted by \(q_i\) is contained in matrix \(S_{s_{r_{i-1}}}\) only. The partial derivative of Eq. (19) with respect to \(q_i\) can be derived:

\[
\frac{\partial S(q)}{\partial q_i} = S_{s_{r_{0}}}, S_{r_{2}}, \ldots S_{r_{(2i-1)}}, S_{s_{r_{2}}}, \ldots S_{s_{r_{(2n-2i+2i-1)}}}, S_{s_{r_{0}}}, s_{r_{0}}
\]  

(20)

According to Eqs. (14), (15) and (16), the matrix \(S_{s_{r_{i-1}}}(q_i)\) has the form of either rotation transformation matrix or translation matrix. Replacing the derivative in the Eq. (20) by its equivalent matrix product:

\[
\frac{\partial S(q)}{\partial q_i} = S_{s_{r_{0}}}, S_{r_{2}}, \ldots S_{r_{(i-1)}}, S_{s_{r_{2}}}, \ldots S_{s_{r_{(i-1)}}}, S_{s_{r_{0}}}, s_{r_{0}}
\]  

(21)

Where \(Q\) is the proper type of derivative operator matrix for the matrix \(S(q_i)\).

Define the identity matrix as follows:
The derivative of the projection of the derivative of the projection of the projection of Eq. (25) is defined as

\[ \frac{d^2 S(q)}{dq^2} = S_{0i} - S_{1i} \cdot \left[ S_{0j} - S_{1j} \cdot \left( S_{0k} - S_{1k} \cdot (q_k) \right) \right] \cdot \left[ S_{0l} - S_{1l} \cdot \left( S_{0m} - S_{1m} \cdot (q_m) \right) \right] \cdot \left( q_i \right) \cdot \left( q_j \right) \cdot \left( q_k \right) \cdot \left( q_l \right) \cdot \left( q_m \right) \cdot \left( q_n \right) \cdot \left( q_o \right) \cdot \left( q_p \right) \cdot \left( q_q \right) \cdot \left( q_r \right) \cdot \left( q_s \right) \cdot \left( q_t \right) \cdot \left( q_u \right) \cdot \left( q_v \right) \cdot \left( q_w \right) \cdot \left( q_x \right) \cdot \left( q_y \right) \cdot \left( q_z \right) \]

When this is done, the two following results are achieved:

1. Expressing the product shown in Eq. (25) for one kind of variable contained within it. Furthermore, this is done by premultiplying the matrix product at the end of Eq. (23) by a derivative operator matrix as shown in Eq. (24).

\[ \frac{d^2 S}{ dq^2 } = D_1 \cdot S \]

In this case, the following is obtained:

\[ D_1 = \left[ \begin{array}{cccc} 0 & -1 & \Delta y & \Delta t \\ 1 & 0 & -\Delta t & \Delta y \\ 0 & 0 & 1 & 0 \end{array} \right] \]

If \( Q = Q_2 \),

\[ D_2 = \left[ \begin{array}{cccc} 0 & 0 & r_{i1} & 0 \\ 0 & 0 & -r_{i2} & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] \]

In summary, we have obtained a procedure for taking the derivative of the matrix product at the front of Eq. (25) for one kind of variable contained within it. Furthermore, this is done by premultiplying the matrix product at the front of Eq. (25) by a derivative operator matrix as shown in Eq. (24).

In order to form this matrix product, the derivative operator matrix must be formulated as stated in Eq. (28) or (29). Consequently, the location of the variable and its type must be known. We see next how this technique is used in deducing the iterative formula for Newton-Raphson iterative procedure. According to Eq. (18) and (19), the matrix equation can be rewritten as:

\[ S(q)^{-1} \cdot I = 0 \]

It is correct only when the unknown or assembly variables have their correct values. Consequently, we must find the “roots” or “zeros” of this matrix equation. A Newton-Raphson iterative procedure is well suited for this type of problem. Beginning with an initial estimate of the unknown or assembly variables ( \( q_0 \) ), a small correction \( \Delta q = (\Delta q_1, \Delta q_2, \Delta q_3) \) can be found such that when it is added to the estimate, a value closer to the correct value is obtained.

This iterative procedure is derived below. It begins by performing a first-order linear expansion of Eq. (30) about \( q_0 \).

\[ S(q) = I + \sum_{i=1}^{n} \frac{\partial S(q_i)}{\partial q} \cdot \Delta q_i = 0 \]

Using the derivative operator matrix \( D_i \) for differentiating the matrix equation, Eq. (30) becomes

\[ S(q) \cdot I + \sum_{i=1}^{n} (D_i \cdot S(q_i) \cdot \Delta q_i) = 0 \]

Postmultiplying by \( S^{-1}(q_0) \) yields

\[ \sum_{i=1}^{n} (D_i \cdot \Delta q_i) = S^{-1}(q_0) \cdot I \]

This is a matrix equation. From it, 9 scalar equations can be formed by equating the two sides, element by element. However, all 9 equations are not meaningful and independent. According to Eq. (28) and (29), a general form of the derivative operator matrix for the \( i \)th variable can be gotten:

\[ D_i = \left[ \begin{array}{cccc} 0 & -d_{i1} & d_{i2} & 0 \\ d_{i1} & 0 & -d_{i2} & 0 \\ 0 & 0 & 0 & 0 \end{array} \right] \]

The bottom row and diagonal elements are all zeros and therefore cannot be used. Furthermore, the second element in the first row of the derivative operator matrix equals the negative of the second element in the first column of the derivative operator matrix. As a result, there are a total of three elements from \( D_i \) that can be used to formulate independent equations. They are \( d_{121}, d_{123} \) and \( d_{123} \).

Next consider the term on the right side of Eq. (33). \( S(q_0) \) can be defined as:

\[ S(q_0) = I + \sum_{i=1}^{n} (D_i \cdot S(q_i) \cdot \Delta q_i) = 0 \]
Summarizing, the scalar equations for the iteration procedure can be written in matrix form as

\[
S(q_{el}) = \begin{bmatrix}
    c_{11} & c_{12} & c_{13} \\
    -c_{12} & c_{11} & c_{23} \\
    0 & 0 & 1
\end{bmatrix}
\]

So,

\[
S^{-1}(q_{el}) = \begin{bmatrix}
    c_{11} - 1 & -c_{12} & -c_{13}c_{11} \\
    -c_{11} & c_{11} & c_{12}c_{13} \\
    0 & 0 & 1
\end{bmatrix}
\]

\[
S^{-1}(q_{el}) - I = \begin{bmatrix}
    c_{11} - 1 & -c_{12} & -c_{13}c_{11} \\
    -c_{11} & c_{11} & c_{12}c_{13} \\
    0 & 0 & 0
\end{bmatrix}
\]

The main purpose of performing a tolerance analysis on the clutch is to determine the variable range of the angle \(\varphi_1\) due to manufacturing variations in the clutch component dimensions. The independent manufacturing variables are the hub dimension \(a\), the roller radius \(c\) and the ring radius \(e\). The distance \(b\) and angle \(\varphi_2\) are not dimensioned because they are assembly variables determined by the sizes of \(a\), \(c\) and \(e\) when the parts are assembled. Table 1 shows the detailed dimensions for the assembly.

| Dimensions | Nominal value | Tolerance(±) |
|------------|---------------|--------------|
| \(a\)      | 27.645        | 0.0125       |
| \(c\)      | 11.43         | 0.01         |
| \(e\)      | 50.8          | 0.05         |
| \(b\)      | 4.81          | ?            |
| \(\varphi_1\) | -7.01838°   | ?            |
| \(\varphi_2\) | 97.01838°    | ?            |
dimension sample generation method described in the Section 2. Here $s = 3$, so set $p = 11$. According to the Eq. (4), the good point $\gamma$ can be gotten:

$$\gamma = (0.6825, 0.8308, 0.7154)$$

(43)

The assembly function expressed by homogeneous transformation matrices is

$$R(\frac{\pi}{2})T(a) R(\frac{-\pi}{2}) T(b) R(\frac{\pi}{2}) T(c) R(-\Phi_1) T(c) R(\Phi_1) R(x) R(\Phi_1) = I$$

(44)

For every sample of $a$, $c$, and $\epsilon$, a sample of $b$, $\Phi_1$, and $\Phi_2$ can be obtained through the Newton-Raphson iterative procedure. Generally, the initial estimates of $b$, $\Phi_1$, and $\Phi_2$ are relevant nominal value. Firstly, compute $S_{ij}$ of every variable and $S(q_1)$. For the variable $b$, $\Phi_1$, and $\Phi_2$, corresponding $S_{b}, S_{\Phi_1}$ and $S_{\Phi_2}$ are respectively obtained by taking the product of first four, seven and eleven matrices of Eq. (44) after substituting the sample values of $a$, $c$ and $\epsilon$, and the initial estimates of $b$, $\Phi_1$ and $\Phi_2$ into Eq. (44). Here, $S(q_1)$ is equal to $S_{b}$. Secondly, obtain the corresponding $D_i$. For the variable $b$, $D_b$ is gotten according to Eqs. (26) and (29), because $b$ is contained in the translation matrix. For the variables $\Phi_1$ and $\Phi_2$, corresponding $D_\Phi$ and $D_\Phi$ are gotten according to Eqs. (26) and (28), because $\Phi_1$ and $\Phi_2$ are contained in the rotation matrix. Thirdly, $G$ is computed according to Eqs. (34) and (39) and $H$ is obtained according to Eqs. (35) and (40). A small correction $\Delta q$ can be gotten through Eq. (42). When it is added to the estimate, a value closer to the correct value is obtained. Finally, repeat steps 1-3 until the length of small correction $\Delta q$ is smaller than specified accuracy (0.001 mm). The components of the final $q_0$ are set to sample values of $b$, $\Phi_1$, and $\Phi_2$ for the given sample values of $a$, $c$ and $\epsilon$.

After all the samples of $b$, $\Phi_1$, and $\Phi_2$ are generated, the samples of $q_1$ can be used to calculate the mean and standard deviation through statistical formulas. With the help of Matlab, a program for tolerance analysis is developed. The sample size is set to 2000. With a desired quality level of $\leq 0.0001$ mm, the components of the final $q_0$ are set to sample values of $b$, $\Phi_1$, and $\Phi_2$ for the given sample values of $a$, $c$ and $\epsilon$.

5. Conclusion

In this paper, a method for statistical tolerance analysis is developed. The proposed method is useful when it is difficult or impossible to get the explicit assembly function. It is a statistical approach and can give results that are reasonable relatively. The good point set is used to generate the samples of dimensions. Besides, the homogeneous transform matrix is introduced to describe the assembly function and Newton’s method is adopted to solve the assembly equations after the iterative formula is deduced. By introducing the homogeneous transform matrix and relevant derivation, a vast number of computations of partial derivatives can be avoided, especially when the sample number is large.

The proposed method is applied to 2D chain, but it can be expanded to 3D chain. The main difference between 2D and 3D chain is about the representation and solution techniques of assembly function. In the future, we will study the representation and solution techniques of assembly function in the 3D case.

Acknowledgements

This research was supported by the National Nature Science Foundation of China (No. 51275464 and 51575484) and Science Fund for Creative Research Groups of National Natural Science Foundation of China (No.51221004).

References

[1] Marziale M, Polini W. A review of two models for tolerance analysis of an assembly: Jacobian and torsor. International Journal of Computer Integrated Manufacturing, 2011;24(1):74-86.
[2] Chen H, Jin S, Li Z, Lai X. A comprehensive study of three dimensional tolerance analysis methods. Computer-Aided Design, 2014;53(2):1-13.
[3] Jiang K, Davidson JK, Liu J, Shah JJ. Using tolerance maps to validate machining tolerances for transfer of cylindrical datum in manufacturing process. The International Journal of Advanced Manufacturing Technology, 2014;73(1-4):465-78.
[4] Giordano M, Samper S, Petit J-P. Tolerance analysis and synthesis by means of deviation domains, axi-symmetric cases. Models for computer aided tolerancing in design and manufacturing. Springer; 2007. p. 85-94.
[5] Ghie W, Laperrière L, Desrochers A. Statistical tolerance analysis using the unified Jacobian–Torsor model. International Journal of Production Research, 2010;48(15):4609-30.
[6] Clément A, Desrochers A, Rivière A. Theory and practice of 3-D tolerancing for assembly. CIRP International Working Seminar on Computer-Aided Tolerancing: 16–17 May 1991; Penn State University 1991. p. 25-55.
[7] Desrochers A, Ghie W, Laperrière L. Application of a Unified Jacobian—Torsor Model for Tolerance Analysis. Journal of Computing and Information Science in Engineering. 2003;3(1):2-14.
[8] Anwer N, Schleibich B, Mathieu L, Wartack S. From solid modelling to skin model shapes: Shifting paradigms in computer-aided tolerancing. CIRP Annals-Manufacturing Technology, 2014;63(1):137-40.
[9] Gao J, Chase KW, Magley SP. Generalized 3-D tolerance analysis of mechanical assemblies with small kinematic adjustments. IIE Trans. 1998;30(4):367-77.
[10] Bender A. Statistical Tolerancing as it Relates to Quality Control and the Designer (6 times 2.5= 15). SAE Technical Paper. 1968.
[11] Evans DH. Statistical tolerancing. The state of the art. III- Shifts and drifts. Journal of Quality Technology. 1975;7:72-6.
[12] Lee W-J, Woo T. Tolerances: their analysis and synthesis. Journal of Engineering for Industry. 1990;112(2):113-21.
[13] Qureshi AJ, Dantan J-Y, Sabri V, Beaucarre P, Gayton N. A statistical tolerance analysis approach for over-constrained mechanism based on optimization and Monte Carlo simulation. Computer-Aided Design, 2012;44(2):132-42.
[14] Hua LK, Wang Y. Application of number theory to numerical analysis. Berlin: Springer; 1981.