Minimum Zone Evaluation of the Form Errors of Quadric Surfaces

Xiangchao Zhang*, Xiangqian Jiang and Paul J. Scott

Centre for Precision Technologies, University of Huddersfield, Huddersfield, HD1 3DH, UK

Abstract

Quadric surfaces commonly exist in natural objects and artificial components. It is widely needed to evaluate the form quality of a measured data set, but the mostly used least squares method will lead to over-estimation and its results are not consistent with the definitions in ISO standards. In this paper a shape recognition approach is presented to determine the surface type and shape parameters from the general implicit quadratic function. Then a self-adaptive differential evolution algorithm is utilized to perform minimum zone evaluation for generic quadrics. The maximal orthogonal distance from the data points to the associated surface is taken as the target to be optimized. Finally experimental examples are presented to verify the developed algorithm.

Keywords: coordinate metrology, quadric surfaces, minimum zone, form error, self-adaptive differential evolution
2000 MSC: 90C47, 90C31, 65K10

1. Introduction

Quadric surfaces are used very extensively in engineering. It has been reported that approximately 85% of manufactured objects can be well-modelled with quadric surfaces, such as sphere, cylinder, cone and paraboloid [1].

The evaluation of the form errors of quadric surfaces is among the most important problems in computational metrology. Most current commercial

*Corresponding author. Tel.: +44-1484-473949
Email address: x.zhang@hud.ac.uk (Xiangchao Zhang)

software applies the least squares method for this purpose due to its ease of implementation and the unbiasedness of its solution for uncorrelated Gaussian distributed noise [2]. However it is likely to overestimate the form tolerance and lead to unnecessary rejections, therefore its solution is only an approximate one, not the optimum. According to ISO 1101 [3], a geometrical tolerance applied to a feature defines the tolerance zone within which that feature shall be contained. This means, taking the cylinder as an example, the cylindricity tolerance is a permissible deviation zone bounded by two coaxial cylinders within which the measured data must lie in between. Here we only consider the symmetric tolerances, i.e. the allowable deviations on both sides of the nominal surface (datum) are equal. Defining a sphere whose centre travels on the nominal surface, the space covered by this moving sphere is regarded as the tolerance zone. Consequently the width of the zone is defined as the diameter of the sphere, as shown in Fig. 1,

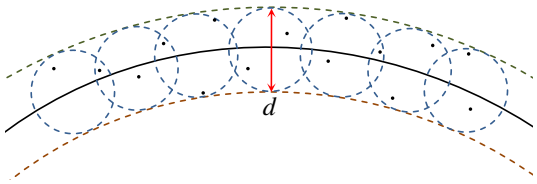


Figure 1: Minimum zone form error

Obviously, the evaluation of minimum zone form error is equivalent to minimize the maximum orthogonal distance from the data points to the nominal surface, as suggested by ISO 5459-3 [4],

$$\min \max_i \|\mathbf{p}_i - \mathbf{q}_i\| \tag{1}$$

where $\mathbf{q}_i$ is the projection point of an arbitrary measured point $\mathbf{p}_i$ onto the nominal surface.

If the nominal function is moved to a non-standard position, the representation will become rather complicated. It is proved that moving the measurement data is equivalent to moving the datum, and their evaluation results are the same [5]. As a consequence transformations are always performed on the measurement data.

This minimax problem is not continuously differentiable, thus very difficult to be solved. This paper presents a heuristic optimization algorithm,

called *self-adaptive differential evolution* (SADE), to conduct minimum zone evaluation of general quadric surfaces. This method shows great superiorities on stability and accuracy, and makes a good balance between exploration and exploitation. Some previous work is surveyed in section 2. To supply an approximate solution for the optimization program, Section 3 describes the shape recognition to initialize the shape parameters and move the data to a standard position. Section 4 discusses to calculate the orthogonal distance from a point to quadric surfaces. Then optimization using SADE follows, as illustrated in Section 5. Experimental verification and validation are given in Section 6 and this paper is summarized in Section 7.

2. Review of related work

Minimum zone evaluations of straightness, flatness, roundness and sphericity are linear or quadratic programming problems, and various algorithms have been well developed in literature, e.g. exchange algorithm [6, 7], support vector machine [8], simplex method [9] and computational geometry methods [10, 11, 12]. But the minimum zone evaluation of generic quadric surfaces is a highly nonlinear programming problem and these algorithms cannot be applied directly. Some researchers attempted to approximate the cylinder/cone iteratively and then solve a sequence of linear programs [13, 14], but these methods prone to sub-optimal solutions. Lai and Chen [15] converted a cylinder into a plane and obtained appropriate control points. The cylindricity is calculated by implementing a series of inverse transformations.

In recent years heuristic optimizers have been employed to solve the minimum zone problems. Lai et al. [16] and Liu et al. [17] used genetic algorithms to evaluate the cylindricity and conicity errors, respectively, while Wen and Song adopted an immune evolutionary algorithm for the sphericity error [18]. The particle swarm optimization technique was used by [19, 20]. The particle swarm optimization method is prone to stagnating at sub-optimal solutions. On the contrary, evolutionary algorithms are able to search the variable space more amply, but normally have a slower convergence rate. Recently, Huang and Lee evaluated the conicity error using minimum potential energy algorithms[21], which are very elaborate.

As for the least squares fitting of quadric surfaces, a lot of research has been carried out, e.g. [22, 23, 24]. So far, little work has been done to systematically study the minimum zone evaluation of generic quadrics.

3. Shape recognition

The general function of a quadric surface is given as,

$$\begin{aligned} & Q(x,y,z) \\ =& ax^2+by^2+cz^2+dxy+exz+fyz+gx+hy+iz+j \\ =& 0 \end{aligned} \tag{2}$$

with $\{a,b,c,d,e,f,g,h,i,j\}$ denoting the coefficients.

The most straightforward way to fit data $\mathbf{P}=\{\mathbf{p}_i\}$ by quadric functions is to minimize $\sum_{i=1}^{N} Q^2(\mathbf{p}_i)$ in the sense of linear least squares. In order to avoid trivial solutions, an eigen-decomposition method can be applied [25]. Then the specific surface type and rough position are restored as follows.

The general function of quadrics in Eq. (2) can be rewritten as,

$$Q(\mathbf{x}) = \mathbf{x}^T \begin{pmatrix} a & d/2 & e/2 \\ d/2 & b & f/2 \\ e/2 & f/2 & c \end{pmatrix} \mathbf{x} + \begin{pmatrix} g & h & i \end{pmatrix} \mathbf{x} + j \tag{3}$$

with $\mathbf{x}=[x,y,z]^T$.

Transform Eq. (3) into a standard form so that the cross terms can be eliminated. According to *the spectral theorem*, the eigenvectors of a real symmetric matrix compose an orthogonal space and all its eigenvalues are real [26]. So that we implement eigen-decomposition onto the quadric form,

$$\begin{pmatrix} a & d/2 & e/2 \\ d/2 & b & f/2 \\ e/2 & f/2 & c \end{pmatrix} = \mathbf{U}\mathbf{S}\mathbf{U}^T$$

In the equation, $\mathbf{S}$ is a diagonal matrix $\mathbf{S}=diag\{\sigma_1,\sigma_2,\sigma_3\}$ with its diagonal entries $\sigma_1 \geq \sigma_2 \geq \sigma_3$ being the eigenvalues. $\mathbf{U}$ is a 3×3 orthogonal matrix. We enforce its determinant be positive, so that $\mathbf{U}$ can be regarded as a rotation matrix in the 3-D Euclidean space and the coordinate system will not be reflected from right-handed to left-handed. Assuming $\widehat{\mathbf{x}}=\mathbf{U}^T\mathbf{x}$, then

$$Q(\mathbf{x}) = \widehat{\mathbf{x}}^T\mathbf{S}\widehat{\mathbf{x}} + \begin{pmatrix} g & h & i \end{pmatrix} \mathbf{U}\widehat{\mathbf{x}} + j$$

It is rewritten as,

$$Q(\mathbf{x}) = \sigma_1\hat{x}^2+\sigma_2\hat{y}^2+\sigma_3\hat{z}^2+G\hat{x}+H\hat{y}+I\hat{z}+j$$

(1) The following standard form emerges when $\sigma_1\sigma_2\sigma_3 \neq 0$,

$$Q(\mathbf{x}) = \sigma_1(\hat{x} - a_1)^2 + \sigma_2(\hat{y} - a_2)^2 + \sigma_3(\hat{z} - a_3)^2 + a_4 \tag{4}$$

The parameters σ_1, σ_2 and σ_3 are used to recognize the surface shape. If $\sigma_2 < 0$, we can change the signs of all the coefficients in Eq. (2) to make $\sigma_2 > 0$ and the surface shape will not be affected.

(2) If any one of the three shape parameters vanishes, say $\sigma_3 = 0$, Eq. (4) will be in the form of,

$$Q(\mathbf{x}) = \sigma_1(\hat{x} - a_1)^2 + \sigma_2(\hat{y} - a_2)^2 + I\hat{z} + a_4 \tag{5}$$

If $\sigma_2 = 0$, the function can be processed in the same manner.

(3) If only one of the three semi-axis lengths is nonzero, it could only be $\sigma_1 \neq 0$,

$$Q(\mathbf{x}) = \sigma_1(\hat{x} - a_1)^2 + H\hat{y} + I\hat{z} + a_4$$

The data will be rotated further about the x axis with a matrix

$$\mathbf{V} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{H}{\sqrt{H^2+I^2}} & \frac{I}{\sqrt{H^2+I^2}} \\ 0 & -\frac{I}{\sqrt{H^2+I^2}} & \frac{H}{\sqrt{H^2+I^2}} \end{pmatrix}$$

So that the new data is $\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix} = \mathbf{V} \begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{pmatrix}$ and the standard function becomes,

$$Q(\mathbf{x}) = \sigma_1(\tilde{x} - a_1)^2 + \sqrt{H^2 + I^2}\tilde{y} + a_4 \tag{6}$$

To make the representations unique, the coefficients in Eqs. (4,5,6) are scaled by appropriate positive factors to convert them into standard functions of the corresponding quadric shapes.

The relationships between surface types and shape parameters in different cases are summarized in Table 1 [27].

Due to the existence of measurement noise in the data, tolerances should be allowed in the determination of the surface types and shape parameters. If the surface type is known before, this procedure can also be used to obtain the approximate shape parameters and move the data to a standard position.

Table 1 has another two columns for the degrees of freedom in rotation and translation, respectively. If a surface is rotationally symmetric about an

Table 1: Determining the shapes of quadratic functions

σ_2	σ_3 & other		shape	R	T
$\sigma_2 = \sigma_1$	$\sigma_2 = \sigma_3$		sphere	3	0
	$\sigma_2 > \sigma_3 > 0$		oblate spheroid	1	0
	$\sigma_3 = 0$	$I = 0$	cylinder	1	1
		$I \neq 0$	circular paraboloid	1	0
	$\sigma_3 < 0$	$a_4 > 0$	two-sheet circular hyperboloid	1	0
		$a_4 = 0$	cone	1	0
		$a_4 < 0$	one-sheet circular hyperboloid	1	0
$\sigma_1 > \sigma_2 > 0$	$\sigma_2 = \sigma_3$		prolate spheroid	1	0
	$\sigma_2 > \sigma_3 > 0$		ellipsoid	0	0
	$\sigma_3 = 0$	$I = 0$	elliptic cylinder	0	1
		$I \neq 0$	elliptic paraboloid	0	0
	$\sigma_3 < 0$	$a_4 > 0$	two-sheet hyperboloid	0	0
		$a_4 = 0$	elliptic cone	0	0
		$a_4 < 0$	one-sheet hyperboloid	0	0
$\sigma_2 = 0$	$\sigma_3 = 0$	$H \neq 0$	parabolic cylinder	0	1
		$H = 0$, $a_4 > 0$	two parallel planes	1	2
		$H = 0$, $a_4 = 0$	plane	1	2
	$\sigma_3 < 0$	$H \neq 0$	hyperbolic paraboloid	0	0
		$H = 0$, $a_4 = 0$	two intersecting planes	0	1
		$H = 0$, $a_4 \neq 0$	hyperbolic cylinder	0	1

axis or translationally symmetric along a direction, this motion variable will be discarded during optimization.

In the standard handbook [28] real quadric surfaces are classified into 12 categories. Here we undertake classification more meticulously. Not only the signs, but also the relationship between the values of the shape parameters σ_1, σ_2 and σ_3 are considered, because this is directly related with the number of variables (degrees of freedom) in the optimization program.

4. Calculating the orthogonal distance from a point to quadric surfaces

The orthogonal distance from a point $\mathbf{p}$ to the quadric surface $Q(\mathbf{x}) = 0$ is defined as $e = \|\mathbf{p} - \mathbf{q}\|$, with $\mathbf{q}$ denoting the projection point of $\mathbf{p}$ on the

surface. In practice, the distance is regarded to be signed, and its sign is the same with $Q(\mathbf{p})$. For standard geometries like planes, spheres, cylinders and cones, e and $\mathbf{q}$ are quite straightforward to obtain. But for a generic quadric surface, this is not so easy. Obviously, the vector $\overline{\mathbf{pq}}$ is parallel with the gradient $\nabla Q(\mathbf{q})$ and both of them are perpendicular to the quadric surface. Hence $\mathbf{q}$ is solved from

$$\begin{cases} Q(\mathbf{q}) = 0 \\ \nabla Q(\mathbf{q}) \times (\mathbf{p} - \mathbf{q}) = 0 \end{cases} \tag{7}$$

using the Gauss-Newton or Levenberg-Marquardt algorithm.

But this method is time consuming, especially when there are many data points. Consequently at the first tens of iterations of the optimization program, the distance is approximated with [25]

$$e_i \approx \frac{Q(\mathbf{p})}{\|\nabla Q(\mathbf{p})\|} \tag{8}$$

As the motion and shape parameters have been approximately identified using linear least squares, the distance e_i will not be very large, i.e. $\mathbf{p}$ is reasonably near to the associated surface. Thus this approximation is acceptable. At the final iterations, the orthogonal distance is in turn calculated from Eq. (7).

As a consequence the minimum zone error is,

$$E = 2 \max_i |e_i| \geq \max_i e_i - \min_i e_i \tag{9}$$

5. A self-adaptive differential evolution algorithm

Inspired by the natural evolution of biological species, Holland proposed a popular algorithm called the *genetic algorithm* (GA) [29]. In 1995, Price and Storn replaced the classical crossover and mutation operators in genetic algorithms by a differential operator, which leads to an algorithm called *differential evolution* (DE) [30]. Compared to GA, DE is more simple but performs better on many numerical optimization problems [31].

At each generation, a Donor vector $\mathbf{v}_i$ is generated for each individual of the population (called *genome* or *chromosome*) $\{\mathbf{y}_i | i = 1, \cdots, N\}$. It is the method of creating this Donor vector that demarcates between various DE schemes. Price et al. suggested 10 mutation schemes [31]. Two mutation

schemes 'DE/rand/1/bin' and 'DE/current to best/2/bin' are applied in this paper [32],

$$\mathbf{v}_i = \begin{cases} \mathbf{y}_r + F(\mathbf{y}_s - \mathbf{y}_t), & \text{rand}[0,1] < p \\ \mathbf{y}_i + F(\mathbf{p}_g - \mathbf{y}_i) + F(\mathbf{y}_r - \mathbf{y}_s), & \text{otherwise} \end{cases} \tag{10}$$

where r, s and t are integers randomly selected from the range $[1, N]$ (excluding i). $F \in [0, 2]$ is used to scale the differential vector.

These two strategies are used very commonly in literature and perform well on problems with distinct characteristics. 'DE/rand/1/bin' demonstrates good diversity while 'DE/current to best/2/bin' shows good convergence property. Here $p \in [0, 1)$ is a user-set parameter.

After the mutation phase, a 'binomial' crossover operation is applied,

$$u_{ij} = \begin{cases} v_{ij} & \text{if rand}_j[0,1] \leq CR \text{ or } j = j_{rand} \\ y_{ij} & \text{otherwise} \end{cases} \tag{11}$$

where $CR \in [0, 1)$ is a user specified crossover constant and j_{rand} is a randomly chosen integer in $[1, N]$ to ensure that the trial vector $\mathbf{u}_i$ will differ from $\mathbf{y}_i$ by at least one parameter. The subscript j refers to the jth dimension. rand$[0, 1]$ is a random number uniformly generated in $[0, 1]$.

Then a selection operation follows,

$$\mathbf{y}_i^{k+1} = \begin{cases} \mathbf{u}_i^k & \text{if } g(\mathbf{u}_i^k) < g(\mathbf{y}_i^k) \\ \mathbf{y}_i^k & \text{otherwise} \end{cases} \tag{12}$$

with k and $k+1$ denoting the individuals in the kth and $(k+1)$th generations, respectively and g representing the objective function to be minimized.

The optimal configuration, i.e. the values of F, CR and p, is very problem-dependant. To obtain relatively good performance in different situations, a *self-adaptive differential evolution* (SADE) is employed here [33].

This technique assigns different crossover constants $\{CR_i, i = 1, 2, \cdots, N\}$ and scale factors $\{F_i, i = 1, 2, \cdots, N\}$ for the individuals and updates them according to the fitness information (objective functions),

$$F_i \Leftarrow \begin{cases} 0.1 + (F_i - 0.1)\frac{f(\mathbf{y}_i) - f_m}{f_a - f_m}, & \text{rand}[0,1] < \tau_1 \text{ and } f(\mathbf{y}_i) < f_a \\ \text{rand}[0.1, 1], & \text{rand}[0,1] < \tau_1 \text{ and } f(\mathbf{y}_i) \geq f_a \\ F_i, & \text{otherwise} \end{cases} \tag{13}$$

$$CR_i \Leftarrow \begin{cases} CR_i\frac{f(\mathbf{y}_i) - f_m}{f_a - f_m}, & \text{rand}[0,1] < \tau_2 \text{ and } f(\mathbf{y}_i) < f_a \\ \text{rand}[1, 1], & \text{rand}[0,1] < \tau_2 \text{ and } f(\mathbf{y}_i) \geq f_a \\ CR_i, & \text{otherwise} \end{cases} \tag{14}$$

Traditionally the individuals are initialized using uniform pseudo-random numbers within the variable space. The random number sequences have a discrepancy of order $[\log(\log N]^{1/2}$ and do not achieve the lowest possible discrepancy, thus the random points cannot evenly cover the whole space, as illustrated in Fig. 2(a). The discrepancy is used to measure the sample point equidistribution, i.e. how uniformly distributed the point set is [34]. Various low discrepancy sequences have been proposed. Among them, the Hammersley sequence shows remarkable superiority on ease of implementation [35]. Thus it is adopted to generate initial population.

Each nonnegative integer i can be expanded by a prime base p,

$$i = a_0 + a_1 p + a_2 p^2 + \cdots + a_r p^r$$

with $0 \leq a_j < p, j = 1, 2, \cdots, r$. A function Φ_p is defined for i as,

$$\Phi_p(i) = \sum_{j=0}^{r} \frac{a_j}{p^{j+1}}$$

It can be proved that for any $i \geq 0$ and $p \geq 2$, $0 \leq \Phi_p(i) < 1$ always holds true.

A series of prime numbers $p_1 < p_2 < \cdots < p_{D-1}$ determine a sequence of functions $\{\Phi_{p_1}, \Phi_{p_2}, \cdots, \Phi_{p_{D-1}}\}$. Then a $D-$dimensional Hammersley point is defined as,

$$\left(\frac{i}{N}, \Phi_{p_1}(i), \cdots, \Phi_{p_{D-1}}(i)\right), i = 1, 2, \cdots, N$$

With p increasing, the point distribution becomes more and more regular [34]. To make the prime numbers as small as possible, we set $p_j = j + 1, j = 1, \cdots, D - 1$.

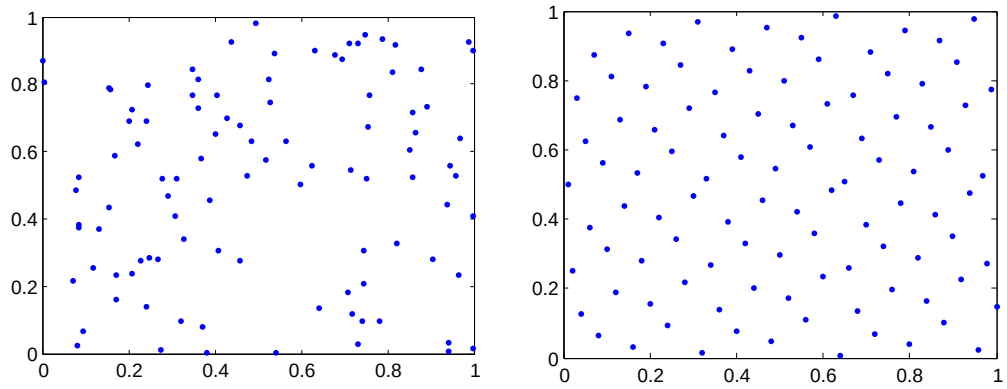
The pseudocode for the SADE program is shown in Algorithm 1.

6. Experimental validation

The validity of the developed algorithm is verified with the cone data given by [36], as listed in Table A.3. The signed distances $\{e_i\}$ from the data points to the associated cone surface are also presented. In the program, the control parameters τ_1 and τ_2 in Eqs. (13,14) are set to be 0.1 [33]. Actually, the performance of the program is not sensitive to them. The

Input: $\mathbf{X}$, $\mathbf{y}_0$
// $\mathbf{X}$:data points, $\mathbf{y}_0$:rough guess of solution
Initialize population $\mathbf{Y}$ and parameters p, $\{F_i\}$ and $\{CR_i\}$;
Evaluate fitness $\{f(\mathbf{y}_i)\}$;
$t = 0$; // generation number for SADE
while $t < t^{max}$ **do**
    $t++$;
    Determine f_m and f_a;
    for $i = 1$ *to* N **do**
        Mutation of $\mathbf{y}_i$ using Eq (10);
        Crossover and selection of $\mathbf{y}_i$ using Eqs (11) and (12);
        Update global optimum $\mathbf{p}_g$;
        Adapt F_i and CR using Eqs (13) and (14);
    end
    if *termination condition satisfied* **then**
        Break;
    end
end
Output: $\mathbf{p}_g$

Algorithm 1: A SADE algorithm



(a) Random numbers (b) Hammersley sequence

Figure 2: Random points generated by random numbers and Hammersley sequence

parameter p in Eq. (10) is updated as $p = 0.382 + 0.618k/K_{max}$. Here k and K_{max} are the current generation and the maximal allowable generation number, respectively. We set $K_{max} = 400$. At the beginning the individuals have more opportunities to use 'DE/current to best/2/bin', and in the later generations tend to 'DE/rand/1/bin'. The maximum projection distance at each generation is stored. When the change of the form error during 20 generations is smaller than a user-set threshold, i.e. the program converges, the optimization process is terminated. Here we set the threshold to be 1e-7. Fig. 3 shows the variation of the objective function. It can be found that the program stabilizes after 170 generations and has a relatively fast convergence rate compared to other heuristic optimization methods. In fact, no heuristic optimization methods can guarantee global convergence. To examine the reliability of the associated results, the program was run 500 times and the standard deviation of the minimum zone form error is 5.44e-6, which is much lower than the uncertainties of CMMs and sufficient for most practical applications. If we alter the termination condition, the uncertainty of the optimization result can be reduced further.

The calculated form error is $E = 2 \max_i |e_i| = 2 \times 0.0016 = 0.0032$. It is consistent with the results of [36] and [21].

To demonstrate the usability of this algorithm for generic quadrics, three shapes are employed: ellipsoid $ax^2 + by^2 + cz^2 = 1$, hyperbolic paraboloid $ax^2 - bz^2 - y = 0$ and parabolic cylinder $x^2 + ay = 0$. The corresponding data sets are listed in Tables A.4, A.5 and A.6.

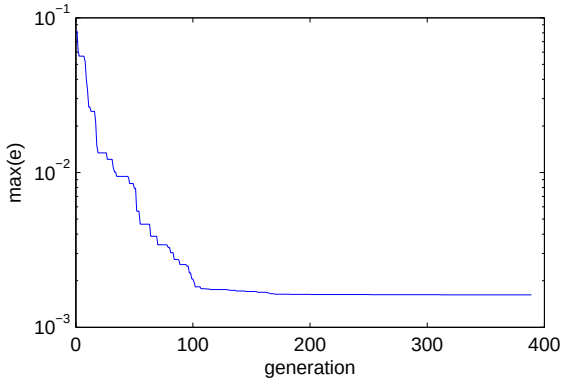


Figure 3: Variation of the objective function

After optimization, the obtained form errors and shape parameters are given in Table 2. For the purpose of comparison, the evaluation results of the orthogonal distance least squares fitting algorithm [24] are also presented (denoted with E_{LS}).

Table 2: Optimization results of generic quadrics

data sets	E_{LS}	E_{SADE}	a	b	c
Ellipsoid	0.086813	0.061639	9.758967e-4	3.572573e-4	2.430297e-4
Hyperbolic paraboloid	0.022467	0.017796	0.024528	0.018628	-
Parabolic cylinder	0.042439	0.035572	100.384759	-	-

This table clearly shows that the form errors of SADE are much smaller than the least squares method.

7. Conclusions

This paper presents a powerful method to evaluate the minimum zone form errors of quadrics, whose optimization target is consistent with the definitions in ISO 1101. Firstly the surface type and shape parameters are identified, so that a good initial guess can be supplied. In this way the number of variables and the range of the variable space can be effectively decreased, which in turn improves the stability and efficiency of the subsequent optimization program. A self-adaptive differential evolution algorithm is employed and its control parameters are updated according to the objective function of each individual during optimization. In this way the program

can be made very flexible and behave very well in different situations. Experimental examples show that this algorithm can achieve very high accuracy and stability for different generic quadrics. In fact its utilization can be extended to more complex surfaces. Therefore this method could be used for online inspection in high precision manufacturing or characterization on coordinate measuring machines.

Acknowledgements

The authors gratefully acknowledge the European Research Council for its programme ERC-2008-AdG 228117-Surfund.

Appendix A. Data sets

References

[1] Chivate PN, Jablokow AG. Solid-model generation form measured point data. Computer Aided Design 1993;25(9):587–600.

[2] Björck Å. Numerical Methods for Least Squares Problems. Philadelphia, PA: SIAM; 1996.

[3] ISO 1101 Geometrical Product Specifications-Geometrical Tolerancing-Tolerances of Form, Orientation, Location and Run-out; 2004.

[4] ISO 5459-3 Geometrical Product Specifications- Datums and Datum System for Geometrical Tolerancing-Part 3: Association Methods; 2000.

[5] Atieg A, Watson GA. A class of methods for fitting a curve or surface to data by minimizing the sum of squares of orthogonal distances. J Computat Appl Math 2003;158(2):277–96.

[6] Chetwynd DG. Applications of linear programming to engineering metrology. Proc Instn Mech Engrs 2010;199(B2):93–100.

[7] Anthony GT, Anthony HM, Bittner B, Butler BP, Cox MG, Drieschner R, et al. Chebyshev best-fit geometric elements. Tech. Rep. NM-R9317; NPL; Teddington, UK; 1993.

Table A.3: Cone data

x	y	z	e	x	y	z	e
9.3702	-12.2214	2.5824	-1.4e-3	9.9508	-11.3232	3.3759	-1.5e-3
8.4808	-12.2215	2.5125	4e-4	9.0621	-11.3232	2.7185	2e-4
7.6404	-12.2215	3.0342	1.6e-3	8.0209	-11.3233	2.9657	1.0e-3
7.4024	-12.2215	4.5240	1.5e-3	7.7587	-11.3233	4.7314	1e-4
8.2881	-12.2215	5.4016	1e-4	9.1982	-11.3232	5.2191	-7e-4
9.1902	-12.2215	5.4477	-1.6e-3	9.7140	-11.3232	4.9157	1.6e-3
9.4425	-12.0327	2.6639	-1.2e-3	9.9019	-11.0681	3.4158	-8e-4
8.2705	-12.0327	2.6256	9e-4	9.2775	-11.0681	2.8480	-1e-4
7.5454	-12.0327	3.2533	1.5e-3	7.5976	-11.0682	4.1781	1.5e-3
7.6772	-12.0328	4.9090	6e-4	8.7349	-11.0681	5.2127	-1.0e-3
8.6765	-12.0327	5.4435	-6e-4	9.6199	-11.0681	4.9130	-1.6e-3
9.5224	-12.0327	2.7053	-1.6e-3	9.7062	-11.0681	4.8302	-1.6e-3
9.5976	-11.8500	2.8031	-1.6e-3	9.8261	-10.8747	3.3763	-1e-4
8.9539	-11.8500	2.5731	-2e-4	9.3116	-10.8746	2.9138	5e-4
7.8444	-11.8500	2.9473	1.6e-3	8.4050	-10.8747	2.8762	8e-4
7.4206	-11.8501	4.2612	1.2e-3	7.6307	-10.8747	3.9913	3e-4
8.2758	-11.8501	5.2998	-1e-4	9.0108	-10.8747	5.1517	1.0e-3
9.5187	-11.8500	2.7540	-1.2e-3	9.6785	-10.8747	4.7913	-1.2e-3
9.5517	-11.6938	2.8186	-1.3e-3	9.7693	-10.7562	3.3406	3e-4
8.9852	-11.6938	2.6151	-3e-4	9.3945	-10.7561	2.9894	8e-4
8.2879	-11.6938	2.7076	9e-4	8.7182	-10.7562	2.8363	9e-4
7.4332	-11.6938	3.9255	9e-4	7.7690	-10.7562	3.4949	8e-4
7.8702	-11.6938	4.9951	-3e-5	7.7813	-10.7562	4.5021	-1e-4
9.0904	-11.6938	5.3394	-1.1e-3	9.1342	-10.7562	5.0941	-8e-4
9.7813	-11.5212	3.0609	-1.6e-3	9.6969	-10.6925	3.2686	7e-4
9.2427	-11.5212	2.7173	-6e-4	9.1131	-10.6925	2.8878	1.2e-3
8.4733	-11.5213	2.6899	6e-4	8.2903	-10.6926	2.9747	1.2e-3
7.6114	-11.5213	3.3940	1.0e-3	7.7149	-10.6926	4.2855	1e-4
7.5303	-11.5213	4.3710	5e-4	8.6423	-10.6926	5.1115	-6e-4
8.6271	-11.5213	5.3149	1.6e-3	9.2927	-10.6926	5.0182	-8e-4

Table A.4: Ellipsoid data

x	y	z	e
-23.648533	-24.790447	-41.620889	-3.0311e-2
-26.413254	-22.372674	-36.539642	-9.925e-3
-28.555111	-19.474662	-30.620247	-2.2093e-2
-30.086129	-16.086731	-24.027960	-1.4342e-2
-30.945717	-12.307547	-16.852787	-8.380e-3
-31.095689	-8.318525	-9.310700	-1.2613e-2
-30.552193	-4.103098	-1.566913	-3.0637e-2
-23.119474	-28.120481	-39.894006	-9.628e-3
-25.757061	-26.469707	-34.381904	-7.681e-3
-27.835619	-24.255718	-28.069052	2.0007e-2
-29.243912	-21.502108	-21.149385	1.3803e-2
-30.016145	-18.268189	-13.744467	2.6994e-2
-30.089499	-14.592153	-6.009070	7.352e-3
-29.492614	-10.614737	1.838134	-1.6649e-2
-22.302453	-31.296941	-38.274510	3.161e-3
-24.730736	-30.460174	-32.362555	1.0042e-2
-26.582538	-28.922445	-25.702240	-1.0102e-2
-27.851292	-26.791446	-18.463853	9.816e-3
-28.498920	-24.039229	-10.817148	3.082e-2
-28.453578	-20.691495	-2.929675	-1.6011e-2
-27.814795	-16.933629	5.016116	-1.7546e-2
-21.125267	-34.360565	-36.785092	-2.7819e-2
-23.317449	-34.274818	-30.513895	1.2221e-2
-24.935899	-33.457291	-23.575743	3.0717e-2
-25.941877	-31.848131	-16.028041	-1.6441e-2
-26.430091	-29.533953	-8.132825	1.3400e-2
-26.319365	-26.548492	-0.108042	3.0816e-2
-25.572494	-23.017772	7.943992	1.3791e-2
-19.763923	-37.176891	-35.509929	4.543e-3
-21.514780	-37.838490	-28.896430	-1.5053e-2
-22.835961	-37.645952	-21.625817	8.340e-3
-23.630836	-36.576291	-13.889779	1.4181e-2
-23.844664	-34.652530	-5.814731	-3.0819e-2
-23.610766	-32.002665	2.387010	2.8995e-2
-22.793811	-28.605127	10.477573	5.139e-3
-18.101870	-39.737572	-34.393094	-1.572e-2

-19.437957	-41.055660	-27.537360	-1.3943e-2
-20.417773	-41.424342	-19.994365	2.1954e-2
-20.897788	-40.793462	-12.107662	-2.435e-3
-20.915387	-39.316127	-3.829656	3.0806e-2
-20.400109	-36.927822	4.482945	-7.252e-3
-19.485123	-33.708671	12.663577	-1.7547e-2
-16.208397	-42.023970	-33.554509	-5.969e-3
-17.184607	-43.831729	-26.469240	2.961e-2
-17.687212	-44.674288	-18.769628	6.816e-3
-17.809832	-44.516035	-10.649327	5.375e-3
-17.537674	-43.334128	-22.70169	1.265e-3
-16.869671	-41.177835	6.112108	-1.4046e-2
-15.842285	-38.115659	14.415989	7.96e-4
-14.234488	-43.858484	-32.919065	1.4096e-2
-14.619931	-46.106687	-25.698337	-2.3536e-2
-14.668137	-47.384015	-17.849113	-2.878e-2
-14.461581	-47.531341	-9.615832	-3.0805e-2
-13.890572	-46.673500	-1.160881	-4.44e-3
-12.983858	-44.738583	7.290986	-2.8483e-2
-11.843975	-41.782971	15.624923	-3.0688e-2

Table A.5: Hyperbolic paraboloid data

x	y	z	e
-17.180514	-8.941395	-15.934777	5.209e-3
-17.888038	-9.250145	-12.424115	8.897e-3
-18.567234	-9.156198	-9.125688	-7.565e-3
-19.181538	-8.681030	-6.027885	-4.571e-3
-19.780469	-7.826162	-3.138222	-8.898e-3
-14.041601	-7.100608	-16.315518	-8.898e-3
-14.750149	-7.415890	-12.811013	-2.04e-4
-15.417520	-7.324092	-9.505146	-8.029e-3
-16.047698	-6.863480	-6.414453	-3.121e-3
-16.621625	-6.008672	-3.513023	8.898e-3
-10.920090	-5.576396	-16.527081	1.44e-4
-11.638712	-5.892685	-13.030751	6.750e-3
-12.312275	-5.807779	-9.736958	5.10e-4
-12.946284	-5.331997	-6.627745	-4.976e-3
-13.530002	-4.479890	-3.748005	-2.552e-3
-7.866135	-4.343016	-16.608771	-7.852e-3
-8.571244	-4.651149	-13.108714	-4.635e-3
-9.241885	-4.573633	-9.787423	2.17e-4
-9.855084	-4.099786	-6.686523	3.01e-3
-10.447764	-3.242443	-3.792832	4.188e-3
-4.814334	-3.409300	-16.506702	-2.585e-3
-5.528704	-3.727481	-13.008645	8.898e-3
-6.196199	-3.631721	-9.695432	-3.2221e-3
-6.819656	-3.165223	-6.594226	2.830e-3
-7.395878	-2.295751	-3.716504	-8.898e-3
-1.802739	-2.763101	-16.262164	-6.508e-3
-2.511148	-3.068671	-12.746089	-6.107e-3
-3.182373	-2.986035	-9.454554	-7.407e-3
-3.811217	-2.531782	-6.354236	8.067e-3
-4.394263	-1.656030	-3.451739	-3.712e-3
1.181182	-2.427385	-15.852554	4.320e-3
0.461094	-2.725873	-12.336681	-3.010e-3
-0.211293	-2.641995	-9.025874	-2.923e-3
-0.835700	-2.176388	-5.933672	2.935e-3
-1.416592	-1.306802	-3.051176	-8.764e-3
4.126449	-2370581	-15.277921	-2.489e-3
3.402740	-2677078	-11.771713	0.485e-3
2.742441	-2584500	-8.470833	-8.898e-3
2.107198	-2.108780	-5.353348	-8.395e-3
1.520080	-1.268693	-2.475818	8.898e-3
7.035365	-2.619172	-14.561025	4.103e-3
6.325983	-2.928963	-11.038203	8.898e-3

Table A.6: Parabolic cylinder data

x	y	z	e
-26.771888	-17.477312	-18.943336	6.941e-3
-27.475917	-15.590256	-15.490258	-1.7786e-2
-28.161683	-13.657796	-12.064897	-2.083e-3
-28.858903	-11.740690	-8.615446	-5.246e-3
-29.563864	-9.826127	-5.181030	5.39e-4
-30.279765	-7.900064	-1.743005	1.7786e-2
-21.608659	-15.058192	-19.242051	9.843e-3
-22.291450	-13.156084	-15.795498	-8.643e-3
-22.994590	-11.225723	-12.349388	2.466e-3
-23.717973	-9.327257	-8.911928	8.6e-5
-24.400351	-7.402518	-5.459769	-3.632e-3
-25.103776	-5.49272	-2.032506	1.020e-3
-16.505591	-13.092972	-19.310873	8.262e-3
-17.205084	-11.183651	-15.855243	-3.592e-3
-17.887247	-9.258635	-12.414269	3.8e-4
-18.591823	-7.339053	-8.971460	4.481e-3
-19.284367	-5.428816	-5.529270	-1.694e-3
-19.985999	-3.501439	-2.081130	5.575e-3
-11.432980	-11.539979	-19.130100	1.6609e-2
-12.122015	-9.640123	-15.681656	-1.799e-3
-12.815837	-7.727785	-12.244039	-3.314e-3
-13.499393	-5.800834	-8.809352	6.993e-3
-14.231483	-3.891704	-5.341829	-5.093e-3
-14.925531	-1.985631	-1.925411	-1.094e-3
-6.398560	-10.461238	-18.732706	4.581e-3
-7.110231	-8.548441	-15.260553	-1.1727e-2
-7.817541	-6.633451	-11.834641	-3.492e-3
-8.497968	-4.700222	-8.408729	1.7786e-2
-9.182986	-2.776651	-4.950978	1.5315e-2
-9.888834	-0.882808	-1.504127	-5.256e-3
-1.417596	-9.806872	-18.086627	1.2e-5
-2.129519	-7.898281	-14.635957	-1.0477e-2
-2.813315	-5.965698	-11.198342	6.271e-3
-3.532753	-4.078446	-7.768228	-1.2831e-2
-4.238148	-2.127197	-4.3044	7.521e-3
-4.926256	-0.217207	-0.876034	9.045e-3
3.498877	-9.585894	-17.196425	-5.055e-3

2.778418	-7.658674	-13.768594	9.494e-3
2.098538	-5.755853	-10.327628	3.04e-4
1.388776	-3.836736	-6.877571	-1.679e-3
0.702907	-1.930407	-3.439289	-7.096e-3
-0.010998	-0.016636	0.018482	-1.7786e-2
8.364356	-9.814942	-16.093724	-1.3343e-2
7.646055	-7.887395	-12.628490	-1.7348e-2
7.646055	-7.887395	-12.628490	-1.7348e-2
6.963775	-5.977914	-9.195173	-1.5962e-2
6.275932	-4.068135	-5.789352	-3.078e-3
5.570706	-2.126371	-2.346349	1.7786e-2
4.888887	-0.226516	1.105079	2.750e-3
13.186081	-10.466206	-14.747083	-1.4769e-2
12.471746	-8.548399	-11.304778	-1.7786e-2
11.785287	-6.633183	-7.856140	-1.7786e-2
11.110765	-4.709021	-4.431375	3.379e-3
10.401681	-2.797925	-0.991010	-3.110e-3
9.709211	-0.894223	2.454550	-1.3431e-2

[8] Malyscheff AM, Trafalis TB, Raman S. From support vector machine learning to the determination of the minimum enclosing zone. Comput Indus Eng 2002;42:59–74.

[9] Kanada T. Evaluation of spherical form errors: computation of sphericity by means of minimum zone method and some examinations with using simulated data. Precision Eng 1995;17:281–9.

[10] Roy UR, Zhang X. Establishment of a pair of concentric circles with the minimum radial separation for assessing roundness error. Computer Aided Des 1992;24:161–8.

[11] Huang J. An exact minimum zone solution for sphericity evaluation. Computer Aided Design 1999;31:845–53.

[12] Samuel CL, Shunmugam MS. Evaluation of sphericity error from form data using computational geometric techniques. Int J Mach Tools Manuf 2002;42:405–16.

[13] Carr K, Ferreira P. Verification of form tolerances part ii: Cylindricity and straightness of a median line. Precision Eng 1995;17:144–56.

[14] Devillers O, Preparata FP. Evaluating the cylindricity of a nominally cylindrical point set. In: Proc. 11th ACM-SIAM Sympos. Discrete Algorithms. San Francisco, CA, USA; 2000, p. 518–27.

[15] Lai J, Chen J. Minimum zone evaluation of circles and cylinders. Int J Machine Tools Manuf 1996;36:435–51.

[16] Lai HY, Jywe WY, Chen CK, Liu CH. Precision modeling of form errors for cylindricity evaluation using genetic algorithms. Precision Eng 2000;24:310–9.

[17] Liu CH, Jywe WY, Chend CK. Quality assessment on a conical taper part based on the minimum zone definition using genetic algorithms. Int J Mach Tools Manuf 2004;44:183–90.

[18] Wen X, Song A. An immune evolutionary algorithm for sphericity error evaluation. Int J Mach Tools Manuf 2004;44:1077–84.

[19] Kovvur Y, Ramaswami H, Annad RB, Anand S. Minimum-zone form tolerance evaluation using particle swarm optimisation. Int J Intellig Sys Tech Appl 2008;4:79–96.

[20] Wen XL, Huang JC, Sheng DH, Wang FL. Conicity and cylindricity error evaluation using particle swarm optimization. Precision Eng 2010;34:338–44.

[21] Huang PH, Lee JC. Minimum zone evaluation of conicity error using minimum potential energy algorithms. Precision Eng 2010;34:709–17.

[22] Forbes AB. Least squares best fit geometric elements. In: Mason JC, Cox MG, editors. Algorithms for Approximation II. London: Chapman and Hall; 1990, p. 311–9.

[23] Petitjean S. A survey of methods for recovering quadrics in triangle meshes. ACM Comput Surv 2002;34(2):211–62.

[24] Zhang X, Jiang X, Scott PJ. Shape recognition and form error evaluation of quadric surfaces. In: Measurement Systems and Process Improvement 2010. NPL, Teddington, UK; 2010,.

[25] Taubin G. Estimation of planar curves, surfaces and nonplanar spaces curves defined by implicit equations with applications to edge and range image segmentation. IEEE Trans Patt Anal Mach Intell 1991;13(11):1115–38.

[26] Halmos P. What does the spectral theorem say? Am Math Monthly 1963;70(3):241–7.

[27] Zhang X. Freeform surface fitting for precision coordinate metrology. Ph.D. thesis; University of Huddersfield, Huddersfield, UK; 2009.

[28] Zwillinger D. CRC Standard Mathematical Tables and Formulae. Chapman and Hall/CRC; 31 ed.; 2003.

[29] Holland JH. Adaptation in Natural and Articicial Systems. Ann Arbor: University of Michigan Press; 1975.

[30] Price K, Storn R. Differential evolution− a simple and efficient adaptive scheme for global optimization over continuous spaces. Tech. Rep.; International Computer Science Institute; Berkley; 1995.

[31] Price KV, Storn RM, Lampinen JA. Differential Evolution−A Practical Approach to Global Optimization. Natural Computing Series; Berlin: Springer; 2005.

[32] Qin AK, Suganthan PN. Self-adaptive differential evolution algorithm for numerical optimization. In: The 2005 IEEE Congress on Evolutionary Computation; vol. 2. 2005, p. 1785–91.

[33] Jia L, Gong W, Wu H. Computational intelligence and intelligent systems. Springer; 2009, p. 215–24.

[34] Wong TT, Luk WS, Heng PA. Sampling with hammersley and halton points. J Graphics Tools 1997;2(2):9–24.

[35] Niederreiter H. Random Number Generation and Quasi-Monte Carlo Methods. Philadelphia: SIAM; 1992.

[36] Chatterjee G, Roth B. Chebyshev approximation methods for evaluating conicity. Measurement 1998;23:63–76.