Reference Algorithms for Chebyshev and One-Sided Data Fitting for Coordinate Metrology

C.M. Shakarji[1]
[1]Manufacturing Engineering Laboratory, National Institute of Standards and Technology, Gaithersburg, MD, USA
Submitted by A. Clement (1), Suresnes, France

Abstract
This paper describes reference algorithms developed at the National Institute of Standards and Technology that fit geometric shapes to data sets according to Chebyshev, maximum-inscribed, and minimum-circumscribed criteria. Using an improved approach, we have developed more reliable reference algorithms for Chebyshev fitting for lines, planes, circles, spheres, cylinders, and cones. In the cases of circles, spheres, and cylinders, we also include maximum-inscribed and minimum-circumscribed fitting. In every case, we obtain the fit through an iteration that begins by using a (relatively easy) least-squares fit and then refine to the desired Chebyshev, maximum-inscribed, or minimum-circumscribed fit. We discuss why computing these fits is substantially more difficult than computing a least-squares fit, as the topography of the objective function prevents certain naïve algorithms from working. We describe our choice of simulated annealing as a method that is general enough to be used for all the geometric shapes considered, requiring minimal customization for each shape. We outline steps taken for each geometric shape to reduce the number of fit parameters, thus improving the performance of the algorithms. We describe a suitable temperature reduction schedule that allows these algorithms to converge. We note cases of nonuniqueness related to maximum-inscribed fits. Finally we document test results showing the effectiveness of these algorithms against a battery of data sets with known solutions, against a limited number of exhaustive search results, against intercomparisons with other algorithms that provide for some of these fits, and against themselves by means of a repeatability study. We note that during intercomparisons, we found significant differences between our well-researched reference results and results obtained from algorithms that can be found in industrial use today.

Keywords:
Algorithm, Coordinate measuring machine (CMM), Optimisation

1 INTRODUCTION

Coordinate Measuring Machines (CMMs) critically rely on mathematical software, particularly for curve and surface fitting. Over the last 15 years, serious problems with the performance of such least-squares fitting software have been identified [1,2]. In response, the National Institute of Standards and Technology (NIST) developed a software package, the NIST Algorithm Testing System (ATS), which can help in assessing the performance of least-squares fitting routines of various geometric shapes.

As helpful as that work has been, its scope extends to include only least-squares algorithms. While these are common, *there are other fit objectives also used in* industry, specifically one-sided fits (here restricted to maximum-inscribed and minimum-circumscribed) and minimum-zone (i.e., Chebyshev) fits. These criteria naturally match some of the language used in internationally utilized standards such as Y14.5 [3].

We sought to extend the original work at NIST that provided reference algorithms for least-squares fitting to now include these other fit objectives. Table 1 indicates which fit objectives are applicable to which geometric shapes that are considered in this paper.

This paper, then, documents the reference algorithms developed at NIST for these fits and is organized as follows: We begin with observing the maximum-inscribed circle problem as an example case to see the requirements for the solution strategy and then show how simulated annealing meets these requirements. We then describe the manner by which the single simulated annealing algorithm can be used to solve every fit type considered in this paper, by means of appropriate parameter and objective function selections. We then go on to describe a fourfold testing method that was applied to the reference algorithms and the summarized results. The testing includes results from an intercomparison that shows how these reference algorithms were used to uncover significant deviations in the fit results of some algorithms.

	Least-squares	Min-zone	Max-inscribed	Min-circumscribed
Line	X	X		
Plane	X	X		
Circle	X	X	X	X
Sphere	X	X	X	X
Cylinder	X	X	X	X
Cone	X	X		

Table 1: Applicability of fit objectives to geometric shapes. Reference algorithms for the least-squares fit objective were completed in previous research [4] and are not developed in this paper.

2 CHOICE OF OPTIMIZATION STRATEGY

In seeking an appropriate algorithm to compute fits according to these objectives, we begin by understanding the nature of these problems, using the example case of a maximum-inscribed circle.

2.1 Definition – maximum-inscribed circle

The following definition will suit our needs for the purposes of this paper: Given a set of data points, we define a maximum inscribed circle as a circle satisfying the following three conditions:

1. No data points lie inside the circle.
2. The circle touches three data points which form an acute or right triangle. (The circle may touch other data points.)
3. No circle of greater radius satisfies conditions (1) and (2).

Note: This definition implies that not every set of data points has a maximum inscribed circle associated with it. For instance, a set of data points that sweep 90° of arc on a circle do not, according to the definition of this paper, have a maximum-inscribed circle. We will not be considering partially sampled geometries in this paper.

Now one might extend the definition above to be meaningful for this type situation, but for the purposes of this paper, we will not be considering such data sets.

Also, the definition reads *a* maximum inscribed circle instead of *the* maximum inscribed circle, since a data set might have more than one maximum inscribed circle as the following illustration shows:

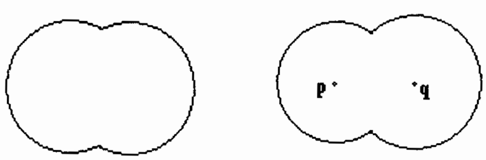

Figure 1: Points selected around the left hand figure might have more than one maximum inscribed circle. The figure on the right shows two inscribed circles, one slightly larger than the other.

When such a case arises, we are content (in this paper) to find one maximum inscribed circle. For this paper we do not consider partially sampled geometries, like points that sweep only an arc of a circle.

2.2 Choice of optimization algorithm

Since a data set might have more than one maximum inscribed circle, we can slightly perturb this situation to see that there can be an inscribed circle centered at **p** (figure 1) which is almost as good (i.e., having nearly the same radius) as one centered at **q**.

In other words, the inscribed circle having the greatest radius (globally) may be hidden among other candidates having locally maximum radii. Many iterative optimization algorithms simply start with an initial guess and search naively (in a strictly downhill fashion) to find a local minimum. If the circle initially guessed in such an algorithm were centered near **p** in the above illustration, the method's search would likely move to find the circle centered at **p** and report that local solution as the optimal. To avoid this, we must require that the algorithm we choose will search out a global minimum among several local candidates. This requirement alone significantly reduces the number of algorithms we have to choose from.

We also note that the objective function is difficult to manipulate mathematically. Some optimization algorithms require that the objective function's first derivatives be provided, and some need the second derivatives as well. We would prefer a method that didn't require these, though appropriate measures could be taken if there were no other choices (e.g., solving numerically for needed derivative values.)

One algorithm noted for solving optimization problems when the global minimum is hidden among several local minima is called simulated annealing [5].

Annealing is the process by which a substance cools slowly enough that the molecules position themselves into a frozen crystalline form. That the crystalline form is their lowest energy state means the molecules have found this global minimum for their arrangement even though there are several arrangements yielding local minima. Simulated annealing borrows from this physical phenomenon and incorporates analogous properties into a numerical algorithm.

Naïve minimization algorithms simply seek in the downhill direction of the objective function and thus will never escape from a local minimum. Simulated annealing avoids this pitfall by searching in the downhill direction most of the time but allowing an uphill move some of the time. The uphill decision is made using the Metropolis criteria in which the probability of accepting an uphill move depends on the "temperature" (the parameter that mirrors the physical cooling process.) As the temperature parameter decreases to zero slowly enough, the algorithm converges on the global minimum.

3 IMPLEMENTATION OF SIMULATED ANNEALING

3.1 Solution procedure

In all cases identified in Table 1, we solve the fitting problem using the same basic strategy:

1. Fit the feature to the data in a least-squares sense using the method described in [4].
2. Rotate and translate the data based on the computed least-squares fit. For circles and spheres, the data is translated so the least-squares center is at the origin. For the other geometries the translation and rotation is such that the line, axis, or normal of the geometry coincides with the *z*-axis.
3. The transformation allows for the fitting geometry and fit objective function to be defined using fewer variables than the general case, as given in table 2.
4. Search for the minimum (or maximum) using the simulated annealing technique. The parameters of the search are given in table 2. The transformed least-squares solution is used as the initial guess for the optimization search.
5. Derive any additional parameters that define the geometry according to table 2.

3.2 Notation

$\mathbf{x} = (x, y, z)$	A point in 3-dimensional space.
$\lvert\cdot\rvert$	The Euclidean (L_2) norm. E. g., $\lvert\mathbf{x}\rvert = \sqrt{x^2 + y^2 + z^2}$.
$\mathbf{x_i} = (x_i, y_i, z_i)$	The i^{th} data point.
$\mathbf{A} = (A, B, C)$	Direction numbers that specify an orientation, $\mathbf{A} \neq \mathbf{0}$.
$\mathbf{a} = (a, b, c)$	Direction cosines that specify an orientation. Note: $\lvert\mathbf{a}\rvert = 1$. An orientation's direction numbers can be converted into direction cosines by $\mathbf{a} = \mathbf{A}/\lvert\mathbf{A}\rvert$.

We define $g_i = g(\mathbf{x_i}, \mathbf{x}, \mathbf{A})$ as the distance from the point, $\mathbf{x_i}$, to the plane containing $\mathbf{x}$ and having normal direction, $\mathbf{a}$. Specifically,

$$g_i = g(\mathbf{x_i}, \mathbf{x}, \mathbf{A}) = \mathbf{a} \cdot (\mathbf{x_i} - \mathbf{x}). \qquad (1)$$

We also define

$$f_i = f(\mathbf{x_i}, \mathbf{x}, \mathbf{A}) = |\mathbf{a} \times (\mathbf{x_i} - \mathbf{x})| \qquad (2)$$

as the distance from the point, $\mathbf{x_i}$, to the line containing $\mathbf{x}$ and having direction $\mathbf{a}$. We let h_i represent the distance from the point, $\mathbf{x_i}$, to $\mathbf{x}$. That is,

$$h_i = |\mathbf{x_i} - \mathbf{x}|. \qquad (3)$$

Finally, it will be useful for the case of cones to denote

$$d_i = f_i \cos\psi + g_i \sin\psi\ , \qquad (4)$$

Where ψ represents the apex semi-angle of the cone. The variable, r, stands for the radius, and, in the case of cones, s is the orthogonal distance from the specified point on the cone's axis to the surface of the cone.

	Location	Direction	Parameters used in optimization	Objective function	Derived parameter after optimization
Min-zone line	$(x,y,0)$	$(A,B,1)$	(x,y,A,B)	$\max(f_i)$	
Min-zone plane	$(0,0,z)$	$(A,B,1)$	(A,B)	$\max(g_i)-\min(g_i)$	$z=[\max(g_i)+\min(g_i)]/(2c)$
Min-zone circle	$(x,y,0)$		(x,y)	$\max(h_i)-\min(h_i)$	$r=[\max(h_i)+\min(h_i)]/2$
Min-circ circle	$(x,y,0)$		(x,y)	$\max(h_i)$	$r=\max(h_i)$
Max-ins circle	$(x,y,0)$		(x,y)	$\min(h_i)$	$r=\min(h_i)$
Min-zone sphere	(x,y,z)		(x,y,z)	$\max(h_i)-\min(h_i)$	$r=[\max(h_i)+\min(h_i)]/2$
Min-circ sphere	(x,y,z)		(x,y,z)	$\max(h_i)$	$r=\max(h_i)$
Max-ins sphere	(x,y,z)		(x,y,z)	$\min(h_i)$	$r=\min(h_i)$
Min-zone cylinder	$(x,y,0)$	$(A,B,1)$	(x,y,A,B)	$\max(f_i)-\min(f_i)$	$r=[\max(f_i)+\min(f_i)]/2$
Min-circ cylinder	$(x,y,0)$	$(A,B,1)$	(x,y,A,B)	$\max(f_i)$	$r=\max(f_i)$
Max-ins cylinder	$(x,y,0)$	$(A,B,1)$	(x,y,A,B)	$\max(f_i)$	$r=\min(f_i)$
Min-zone cone	$(x,y,0)$	$(A,B,1)$	(x,y,A,B,ψ)	$\max(d_i)-\min(d_i)$	$s=[\max(d_i)+\min(d_i)]/2$

Table 2. This table shows the parameterization of each geometry along with the parameters used in the optimization, the appropriate objective function, and any further derived parameters. In the maximum-inscribed cases, the objective function is to be maximized, rather than minimized as in the other cases.

3.3 An illustrative example

To illustrate the procedure and the use of the information in table 2, consider the case of finding the minimum-zone cylinder to a set of data points.

1. We first compute the least-squares cylinder to the data, using the method described in [4].
2. The least-squares solution can be represented by a point on the axis, the axis direction, and the radius. Using that information, we translate and rotate the data points such that the axis of the least-squares solution of the transformed data is, in fact, the *z*-axis.
3. Table 2 indicates that we can identify a nearby cylinder by knowing the location it pierces the *xy*-plane and its direction. The least squares cylinder is located at $(0,0,0)$ and has direction $(0,0,1)$.
4. Starting with an initial guess of $(0,0,0,0)$, we search over other values of (x,y,A,B) to find the minimum of the objective function. For any fixed values of (x,y,A,B), the objective function is given in table 2 as $\max(f_i)-\min(f_i)$.
5. Once the minimum is found, we know the minimum-zone cylinder's location and direction. But in order to find its radius, we simply compute it based on the last column of table 2. Thus the minimum-zone cylinder is obtained.

3.4 Implementation details of temperature

Simulated annealing is an iterative method that requires at each step a reduction in a key parameter (called "temperature," mimicking the physical annealing process). A suitable temperature reduction schedule is needed that allows these algorithms to converge. Finding a working schedule through iterations in the search can be more of an art than a science. In our implementation, we began with a temperature of 10^{-4} and reduced it by a factor of 0.9 at each iteration. Faster decreases (factors of 0.7 and lower) seemed to work equally well, but since these are purely reference algorithms, we can afford to be overly conservative.

4 TESTING THE REFERENCE ALGORITHMS

We implemented the simulated annealing algorithm and tested them four different ways, which we document here. By performance, we speak only of the accuracy of the results, not the time required to obtain them. As these are reference algorithms, speed is a secondary consideration. The testing documented here involves data sets of up to several hundred points. We have not yet conducted any testing with large data sets of over 10,000 points.

4.1 Testing versus the exhaustive search solution

For some geometric cases (lines, planes, circles), and for relatively small data sets, the solution of the fitting problem can easily and reliably (albeit not quickly) be obtained through an exhaustive search [6], and these can be compared with results from these new algorithms.

We created a data set with two superimposed lobed form errors designed to yield a data set having two maximum inscribed circles. Each algorithm found a different maximum inscribed circle as table 3 shows. The 100 data points were selected using equispaced angular intervals on a perturbed unit circle given by

$$r(\theta) = 1 + (.02)\sin 4\theta + (.01)\sin 2\theta \,. \qquad (5)$$

	Exhaustive search	Simulated annealing
x	-.00369371351261293	.00369371351260858
y	-.00784954077495501	.00784954077494546
r	.9726878093314897	.9726878093314895

Table 3. Results from two algorithms using a data set having two maximum-inscribed circles.

We observe that the methods found different maximum inscribed circles, but each has the same radius (to within computational limits). Also, the effect of roundoff errors is evident in the fact that the reported centers are not exactly symmetric with respect to the origin.

The close agreement between the results of the two methods (up to nearly the computational limits) is representative of hundreds of other comparisons made between the methods. The computed centers were the same in all cases except this contrived case, designed to have a nonunique solution.

4.2 Testing versus data sets with known solutions

Since the fit types considered in the paper are determined by a small set of critical points within a data set, it is possible to construct data sets with known solution information by prescribing the critical points and adding additional data points. If the computed solution has a higher objective function than that in the data construction, then the algorithm has not found the optimal fit.

We performed tests on about 100 data sets in this fashion and never encountered the case where the objective function value returned by the algorithm was worse than the value designed (aside from negligible differences due to computational limits of precision).

4.3 Testing versus industrial results

We ran comparisons with industrial partners for all the algorithms described in this paper. We found the results encouraging with respect to our algorithms, but alarming with regard to the outside algorithms used in the comparison. For each discrepancy, we computed the objective functions in order to determine the better fit. In every case of diifference, the algorithms presented in this paper gave the better objective function values. This means that serious problems exist in some algorithms that can be found in commercial use today.

Details were presented at [7], but, in summary, out of about 200 data sets, there were very large differences in about 20% of the fits. Of these, the algorithms presented in this paper invariably returned the better fits, based on objective function values.

4.4 Testing by observing repeatability

Repeatability can be seen by making several runs on the same data set while varying the initial guess. When the sizes of the initial guess perturbations were of the order of the form error of the data, the algorithms consistently found the true fit up to negligible roundoff effects due to computational limits. Figure 2 shows the repeatability of computing the minimum-zone cylinders for several data sets. This figure is representative of repeatability results of other fits described in this paper. Each of the 10 data sets was fit 10 times using different initial guesses. Shown are the ranges of the diameters returned for each data set. These data sets contained 100 points each and had nominal diameters between 150 and 180 mm.

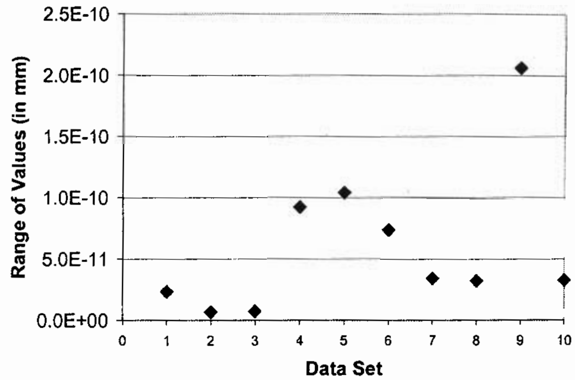

Figure 2. The repeatability of the diameters of the maximum-inscribed cylinder fits.

5 CONCLUSION

We successfully implemented simulated annealing optimization techniques to create 12 reference fitting algorithms of various combinations of geometric shape and fit objective. We used a fourfold method of testing the results of these algorithms to find that the reference algorithms, though not meant to be fast, are very reliable in their fit results. We also found alarming results when comparing results with other algorithms that can be found in industrial use today. At NIST we are using these algorithms to develop data sets and reference results that can be made available to industry to allow for testing and provide means for improvements in industrial software.

6 REFERENCES

[1] Walker, R., 1988, CMM Form Tolerance Algorithm Testing, GIDEP Alert X1-A-88-01, Government-Industry Data Exchange Program, DOD, Washington, D.C.

[2] Hopp, T. H., 1993, Computational Metrology, Manufacturing Review, Vol. 6, No. 4, pp. 295–304, American Society of Mechanical Engineers, New York, NY.

[3] ASME, 1994, Dimensioning and Tolerancing, ANSI/ASME Y14.5, American Society of Mechanical Engineers and American National Standards Institute, New York, NY.

[4] Shakarji, C.M., 1998, Least Squares Fitting Algorithms of the NIST Algorithm Testing System, Journal of Research of the National Institute of Standards and Technology 103 (6), 633-641.

[5] Press, W.H. et al, 1995, Numerical Recipes in C, Cambridge University Press, 451-455.

[6] Shakarji, C.M., 2004, Investigating Maximum-Inscribed and Minimum-Circumscribed Fitting Algorithms for the NIST Algorithm Testing System, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD, to appear.

[7] Shakarji, C. M., 2002, Evaluation of One and two Sided Geometric Fitting Algorithms In Industrial Software, Proceedings, American Society for Precision Engineering, October 20-25, 100-105.