NPL Report DITC 221/93

Chebyshev Best–Fit Geometric Elements

G T Anthony, H M Anthony, B Bittner, B P Butler,
M G Cox, R Drieschner, R Elligsen, A B Forbes, H Groß,
S A Hannaby, P M Harris and J Kok

September 1993

NPL Report DITC 221/93

September 1993

Chebyshev Best–Fit Geometric Elements

G T Anthony, H M Anthony, B Bittner*, B P Butler, M G Cox, R Drieschner*, R Elligsen*, A B Forbes, H Groß*, S A Hannaby, P M Harris and J Kok†

Division of Information Technology and Computing
National Physical Laboratory
Teddington
Middlesex
United Kingdom
TW11 0LW

ABSTRACT

This report summarises the work carried out under the European Communities' Bureau of Reference (BCR) contract 3327/1/0/158/89/9 – BCR – UK (30) Chebyshev Reference Software for the Assessment of Geometric Form [1]. This involved developing algorithms and software to determine best fits in the Chebyshev sense to coordinate data nominally representing a line, plane, circle, sphere, cylinder or cone. Chebyshev is here extended to include maximum inscribed and minimum circumscribed circle, sphere and cylinder.

*Physikalisch–Technische Bundesanstalt, Braunschweig, Germany.

†Centrum voor Wiskunde en Informatica, Amsterdam, The Netherlands.

ISSN 0262-5369

National Physical Laboratory
Teddington, Middlesex, United Kingdom, TW11 0LW

Approved on behalf of Chief Executive, NPL,
by Mr A J Marks, Head, Division of Information Technology and Computing

CONTENTS

TABLES

1 INTRODUCTION

This report summarises work carried out in a project supported by the European Communities' Bureau of Reference (BCR). The project partners were the National Physical Laboratory (NPL) in the UK, the Physikalisch–Technische Bundesanstalt (PTB) in Germany and the Centrum voor Wiskunde en Informatica (CWI) in the Netherlands.

The aim of the project was to produce reference software for the assessment, under a specified criterion, of various geometric forms from coordinate data. The geometric forms considered are line, plane, circle, sphere, cylinder and cone. The criterion for the assessments is, generally, minimum zone (MZ) and, where appropriate, minimum circumscribed (MC) and maximum inscribed (MI).

The report is organised as follows: in the next Section some background to the project and problems is given; Section 3 describes and classifies the problems; Section 4 gives geometric characterizations of solutions to some of the problems; Sections 5 and 6 outline, respectively, the algorithmic approaches adopted and their implementation; Section 7 details the extensive testing of the software and, finally, Section 8 reports on the extent to which the original aims have been achieved.

2 BACKGROUND

The assessment of the geometric form of manufactured components is an essential part of any quality control scheme. As such, it is arguably the commonest and most important procedure in engineering metrology. The forms assessed are often (parts of) simple geometric shapes, *viz* straight lines, planes, circles, spheres, cylinders and cones. These features can be assessed using specific gauges or dedicated measuring instruments; but, increasingly, the general purpose *coordinate measuring machine* (CMM) is used to measure all such shapes. The CMM is a powerful and versatile measuring device employing three movable components moving along mutually perpendicular guideways.[1] There are many different configurations of CMM but in all cases these three degrees of freedom permit a measuring probe to be located anywhere within the working volume of the machine. The measuring probe contains a ball-ended tip that contacts the surface of the workpiece at a number of points, and for each such point the CMM records the Cartesian coordinates of the centre of the ball. The measured coordinates do not give an assessment of form directly. It is necessary first to analyse the measured points to infer the continuous shape of which they are a discrete representation. Due to errors of form and of measurement, the points will not in general lie on the nominal form, or even on a similar form. The usually adopted analytical procedure is to find a "substitute element". That is, the geometric form of the same kind as the nominal form that best fits the measured points, under some criterion. The departures of the measured points from this substitute element are used to assess the manufactured form.

The fitting criterion is chosen according to circumstances. Four criteria are frequently referred to *viz* least squares (Gaussian), minimum zone, minimum circumscribed and maximum inscribed (*vide eg* BS3730). In practice, least squares, which is appropriate where measurement errors predominate, is almost universally used. Minimum zone is appropriate in the more usual case where measurement errors are small compared to form errors. The remaining two criteria, which can only sensibly be defined for a circle, sphere or cylinder, are suitable where mating of parts is required.

[1]There are other, less common, CMM types, *eg* with a rotary table. They are not considered here.

In 1983 BCR sponsored an intercomparison of software for the least squares assessment of geometric form using simulated data sets [16]. The results demonstrated considerable shortcomings in most such software then available. A further intercomparison in 1986 [4], with data sets that presented a more stringent test of the software, indicated that significant improvements had been effected, although much of the software was still unsatisfactory. These results imply that some of the providers of the software were unaware of the unsatisfactory performance of their products.

Numerical software that resides in the processors which form an integral part of a coordinate measuring machine is subject to two major constraints: it must produce results in a time commensurate with the time required to make the measurements, and it may not have a large amount of memory available to it. These constraints will possibly impose compromises on the algorithms that the software implements. Software designed for a large computer detached from the measurement process is relatively unaffected by these constraints and therefore is more likely to produce acceptable results in all cases. The availability of mainframe software made the intercomparison exercises possible, and these in turn led to improved production software.

With increasing accuracy of CMM's, the minimum zone criterion for determining a substitute element becomes more appropriate. However, the design of Chebyshev algorithms is a far more challenging task than the design of the corresponding least squares algorithms. So, software for the Chebyshev assessment of form is likely to be even less satisfactory than that originally available for Gaussian assessment. In fact, very little such software is currently commercially available. Consequently, a BCR committee of experts recommended that a project be sponsored to produce Chebyshev (and maximum inscribed and minimum circumscribed where appropriate) *reference* software for the assessment of geometric form. In the event, the project partners were aware that the methods developed were likely to be implemented as production software, so attempts were made to contain the time and storage requirements within reasonable bounds, but not at the expense of accuracy or reliability.

It is an unfortunate fact that although drawing office practice (and the Standards that inform it) is to quote tolerances in terms of the criteria considered here, the software assessment of a manufactured realisation of a design is usually by least squares. The cause (or consequence) of this is that very little has been published on Chebyshev fitting to geometric elements. However, the following are worthy of note: [17, 5, 8, 9].

3 SPECIFICATION OF PROBLEMS

The geometric elements we are interested in are the line (2D and 3D), circle (2D and 3D), plane, sphere, cylinder and cone. For each of these we define a Chebyshev best-fit problem where we wish to determine the geometric element that minimizes the maximum distance, measured orthogonally to the surface of the element, of a data point from the element. In mathematical terms, if the geometric element is described using the vector of parameters $\boldsymbol{u}$ and $d_i(\boldsymbol{u})$ denotes the distance from the i^{th} data point to the element defined by $\boldsymbol{u}$, we wish to find $\boldsymbol{u}^*$ that solves

$$\min_{\boldsymbol{u}} \max_{i} |d_i(\boldsymbol{u})|. \tag{1}$$

We call the element with parameters $\boldsymbol{u}^*$ the *Chebyshev best-fit* element to the data. We may write (1) as the following constrained minimization problem:

$$\min_{\boldsymbol{u},s} s \tag{2}$$

subject to

$$s - |d_i(\boldsymbol{u})| \geq 0, \quad \forall i \in I, \tag{3}$$

where I is the set of data point indices. The solution $(\boldsymbol{u}^*, s^*)$ to this problem satisfies

$$s^* = \max_i |d_i(\boldsymbol{u}^*)|,$$

ie it is the *minimax* (also called *Chebyshev*) solution.

For all the geometric elements except the line in 3D and circle in 3D, the distance function $d_i(\boldsymbol{u})$ is positive or negative depending on which side of the element the i^{th} point lies. For these problems, the parameters $(\boldsymbol{u}, s)$ may be used to define a *zone* element, and the problem (2) and (3) is called a *minimum zone* problem. The zone element $(\boldsymbol{u}, s)$ consists of the two geometric elements composed of points at distances s and $-s$ from the geometric element $\boldsymbol{u}$. The distance $2s$ between the two surfaces of the zone element is often called its *width*. In these terms the solution to the minimum zone problem is the zone element that, by virtue of (2), has minimum width and, because (3) holds, encloses all the data points.

For the line in 3D and circle in 3D the distance function $d_i(\boldsymbol{u})$ can take only one sign and the problem (2) and (3) is interpreted differently. For the line in 3D, the parameters $(\boldsymbol{u}, s)$ define a cylinder of radius s whose axis is the line with parameters $\boldsymbol{u}$. The solution to (2) and (3) is the cylinder of smallest radius that contains all the data: the *minimum circumscribed* cylinder. For the circle in 3D, the parameters $(\boldsymbol{u}, s)$ define a torus of "small radius" s, and (2) and (3) is interpreted as the *minimum circumscribed* torus problem. We emphasise that although we here find a cylinder and torus the distances $d_i(\boldsymbol{u})$ are measured from the line and circle, respectively, defined by the parameters $\boldsymbol{u}$. The line and circle are the *core* of the cylinder and torus, respectively (see below). Indeed, for each of the circle (2D), sphere and cylinder we may define a *minimum circumscribed* problem and, as above, these may be posed mathematically in the form of the problem (2) and (3). Similarly, for these forms a *maximum inscribed* problem may be defined in the same way, with appropriate changes of sign in (2) and (3). Also, for the maximum inscribed problems to have finite solutions the core must be "within" the data, for example by being constrained to lie inside the convex hull of the data. Such a constraint will be tacitly assumed. For both maximum inscribed and minimum circumscribed problems the distances $d_i(\boldsymbol{u})$ are measured from the core $\boldsymbol{u}$, rather than from the geometric form.

For convenience, we describe the three problems of minimum zone, minimum circumscribed and maximum inscribed as Chebyshev problems, although the last two may not always be derived from the Chebyshev or minimax norm.

It is clear, therefore, that the geometric element fitting problems with which we are concerned correspond to a special case of the more general problem

$$\min_{\boldsymbol{u}} F(\boldsymbol{u})$$

subject to

$$c_i(\boldsymbol{u}) \geq 0, \quad \forall i \in I.$$

Here, the parameters $\boldsymbol{u} \in \mathcal{R}^n$ describe the element that we wish to find, $F(\boldsymbol{u})$ describes the property of that element that we wish to minimize, and the constraints $c_i(\boldsymbol{u})$ restrict the position of the element with respect to the measured data points.

Another way to classify the problems of interest is as follows. In place of the geometric elements line (2D and 3D), plane, circle (2D and 3D), sphere, cylinder and cone and the minimum zone, minimum circumscribed and maximum inscribed problems, consider the following elements

stripe	(*ie* the space between 2 parallel line segments)
disc	
annulus	
plate	(*ie* the space between 2 parallel plane segments)
sphere	
spherical shell	
cylinder	
cylindrical shell	
conical shell	
torus.	

Each of these can be thought of as all the points (of $\mathcal{R}^2$ or $\mathcal{R}^3$ as appropriate) lying within a fixed orthogonal distance from the element's *core*. The fixed distance is called the *width* (generally, half the width as defined above). The core is the mean of the bounding elements of the stripe, annulus, plate and the shells; the centre of the disc or sphere; the axis of the cylinder and the "large circle" of the torus. For each of these 10 elements a *minimum inclusion* problem can be defined, *viz* find the designated element of minimum width that encloses or contains all the data points. For the disc, sphere and cylinder a *maximum exclusion* problem can also be defined, *viz* the element of maximum width that encloses none of the data points.

4 CHARACTERIZATION OF SOLUTIONS

Two properties are required for an element to be a solution to any one of the inclusion problems:

1. it must be *feasible, ie* contain all the data points
2. it must be *minimal, ie* any small change to the position of the element, maintaining feasibility, increases its width.

This defines a *local* solution. A smaller solution may possibly be found by making a *large* change to the position of the element. The *global* solution to the problem is the smallest of the local solutions.

Similar conditions, replacing "contains" by "excludes", "minimal" by "maximal" and so on, apply to solutions to the exclusion problems. Hereafter, this is generally to be understood and we will not always make the distinction explicitly.

The solution to a problem for a given data set is an interpolating element to a subset of the data points, the *contacting* points on the boundary of the element. For some of the problems a geometric characterization of the contacting points to a local solution can be given:

MZ lines in 2D: 3 points with 1 on one line projecting orthogonally into the interior of the line segment defined by the other 2 points.

MZ planes: 4 points with 1 on one plane projecting orthogonally into the interior of the triangular plane segment formed by the other 3 points, *or*
with 2 on each plane defining line segments that intersect internally when projected orthogonally onto either plane.

MC circle: 2 points defining a diameter of the circle *or*
3 points at the vertices of a triangle containing the circle centre (an acute angled triangle).

MI circle: 3 points at the vertices of a triangle for which the circle centre lies within its interior.

MZ circles: 4 points, 2 on each circle, which when radially projected onto a common concentric circle define chords that intersect internally.

MC sphere: 2 points defining a diameter,
3 points at the vertices of a triangle that contains the centre *or*
4 points at the vertices of a tetrahedron that contains the centre.

MI sphere: 4 points at the vertices of a tetrahedron for which the sphere centre lies within its interior.

MZ spheres: 5 points, 3 on one sphere and 2 on the other which, when radially projected onto a common concentric sphere, define respectively the vertices of a triangle and the ends of a chord that intersect internally.

These characterizations can be unified by saying that the intersection of the convex hulls of the contacting points (projected onto the mean element) of the bounding elements for MZ or of the contacting points of the element and of the core (centre) for MC or MI, is not empty.

Strictly, the list above is incomplete. There are several special arrangements of more than the given number of contacting points defining a solution. For example, 4 contacting points at the vertices of a rectangle may define a MZ line, but no 3 of them satisfies the given condition. However, they do satisfy the hull intersection property – in fact the projected line segments coincide. The hull intersection property characterizes all local solutions for all the elements listed.

It should be noted that the elements included in this list are just those that correspond to linearly constrained optimization problems (*vide* Section 5.1). Indeed, the characterizations were determined by considering how the Kuhn–Tucker equations could be satisfied, *ie* by considering the stationary points of the Lagrangian. It should further be noted that for these elements we can determine all the interpolating elements to the quoted number of contacting points.

If we have a feasible solution for a set of data that happens to be global for the contacting points, then the solution is global also for the entire dataset. For, if there were a better solution for the set, it would also be a better solution for the subset containing just the contacting points. If a feasible solution is not global for the contacting points we cannot in general tell whether it is global for the data set, as it may be.

For certain of the elements it is possible to give geometric characterizations for the global solution to the quoted number of contacting points. For example, for the 2D line the solution is global if the line segment between the 2 points on one line is the largest side of the triangle defined by all 3 points. Moreover, for the minimum circumscribed circle and sphere there is just one interpolating element satisfying the hull intersection property for the given number of contacting points, so we know immediately that the corresponding feasible solution is global.

For the elements not in the list we have neither characterizations nor bounds on the number of necessary contacting points. Furthermore, we do not know how to determine all interpolating cylinders, cones or tori to small numbers of points. The significance of these comments will become apparent in the following Sections.

5 ALGORITHMIC APPROACHES

In this Section we broadly outline the algorithms implemented in the assessment software.

5.1 MATHEMATICAL PROGRAMMING METHODS

In Section 3 we demonstrated that the geometric element fitting problems correspond to a special case of the more general problem

$$\min_{\boldsymbol{u}} F(\boldsymbol{u}) \tag{4}$$

subject to

$$c_i(\boldsymbol{u}) \geq 0, \quad i \in I, \tag{5}$$

where the parameters $\boldsymbol{u} \in \mathcal{R}^n$ describe the element that we wish to find, $F(\boldsymbol{u})$ describes the property of that element that we wish to minimize, and the constraints (5) restrict the position of the element with respect to the measured data points. We also note that the particular formulation of each element fitting problem as an optimization problem provides an important subdivision of the elements. The problems for the line (2D), plane, circle (2D) and sphere may be posed as *linearly* constrained optimization problems, whereas those for the cylinder, cone and circle (3D) are *nonlinearly* constrained. This distinction has a bearing on the difficulty of the problems, as well as on methods for their solution.

We give below a brief overview of classical optimization theory which is appropriate to the problems of interest. Motivated by this theory, we present a "feasible-descent" algorithm for the linearly constrained problems which possesses known mathematical properties. We discuss the difficulties that arise when this algorithm is applied to the nonlinearly constrained problems, and we indicate how these difficulties may be overcome. We conclude with some remarks concerning how we cope with the problem of *degeneracy* within the algorithms presented.

5.1.1 Constrained optimization theory

We consider the general constrained minimization problem defined by (4) and (5). Then, at a local minimizer $\boldsymbol{u}^*$ the following conditions hold (*vide* [6, 7]).

Firstly, $\boldsymbol{u}^*$ is a *feasible* point, *ie* all the constraints (5) are satisfied and, for some (*a priori* unknown) subset I^* of I, the constraints are satisfied with equality:

$$c_i(\boldsymbol{u}^*) = 0, \quad i \in I^*.$$

The constraints corresponding to I^* (hereafter abbreviated to "the constraints I^*") are said to be *active* at the solution.

Secondly, $\boldsymbol{u}^*$ solves the equality constrained problem defined by the active constraints:

$$\min_{\boldsymbol{u}} F(\boldsymbol{u}) \tag{6}$$

subject to

$$c_i(\boldsymbol{u}) = 0, \quad i \in I^*. \tag{7}$$

Now, $\boldsymbol{u}^*$ is a local solution to this equality constrained problem if there exist *Lagrange multipliers* $\boldsymbol{\lambda}^*$ for which $(\boldsymbol{u}^*, \boldsymbol{\lambda}^*)$ is a stationary point of the *Lagrangian* function $\mathcal{L}(\boldsymbol{u}, \boldsymbol{\lambda})$ defined by

$$\mathcal{L}(\boldsymbol{u}, \boldsymbol{\lambda}) = F(\boldsymbol{u}) - \sum_{i \in I^*} \lambda_i c_i(\boldsymbol{u}).$$

By considering first order derivatives of $\mathcal{L}$, we deduce that $\boldsymbol{u}^*$ and $\boldsymbol{\lambda}^*$ must satisfy the *Kuhn-Tucker* equations

$$\begin{aligned} \boldsymbol{g}(\boldsymbol{u}^*) &= \sum_{i \in I^*} \lambda_i^* \boldsymbol{a}_i(\boldsymbol{u}^*), \\ c_i(\boldsymbol{u}^*) &= 0, \quad i \in I^*, \end{aligned} \tag{8}$$

where $\boldsymbol{g}(\boldsymbol{u}) = \nabla F(\boldsymbol{u})$ and $\boldsymbol{a}_i(\boldsymbol{u}) = \nabla c_i(\boldsymbol{u})$.

Finally, if the solution $(\boldsymbol{u}^*, \boldsymbol{\lambda}^*)$ to the Kuhn-Tucker equations corresponds to a local minimum for the inequality constrained problem, the Lagrange multipliers satisfy a non-negativity condition,

$$\lambda_i^* \geq 0, \quad i \in I^*, \tag{9}$$

and the matrix

$$Z(\boldsymbol{u}^*)^{\mathrm{T}} W(\boldsymbol{u}^*, \boldsymbol{\lambda}^*) Z(\boldsymbol{u}^*)$$

is positive semi-definite. Here, $W(\boldsymbol{u}, \boldsymbol{\lambda})$ is the Hessian of $\mathcal{L}(\boldsymbol{u}, \boldsymbol{\lambda})$ with respect to $\boldsymbol{u}$, and $Z(\boldsymbol{u})$ denotes a matrix whose columns form a basis for the set of vectors orthogonal to the vectors $\boldsymbol{a}_i(\boldsymbol{u})$, $i \in I^*$.

For the element fitting problems with which we are concerned, an active constraint defines a point that contacts the element determined by $\boldsymbol{u}$. An important consideration affecting methods for solving inequality constrained minimization problems is that the set I^* of constraints active at the solution is not known *a priori*. Mathematical programming methods, which are motivated by the optimality conditions given above, search for the active set, or equivalently the corresponding contacting points, in a systematic manner by examining the solution to the Kuhn-Tucker equations (8) for each iterate. In particular, a negative Lagrange multiplier indicates that the associated contacting point should be dropped from the active constraint set, and a new contacting point perhaps introduced.

Having updated an estimate I^a of I^* using the Kuhn-Tucker equations, a new estimate of the solution element may be determined by solving the equality constrained problem (6) and (7) defined by I^a. For the linearly constrained problems, this sub-problem may be solved directly (analytically) using simple geometric arguments. However, for the nonlinearly constrained problems, the sub-problem is computationally difficult and expensive to solve, and we may choose to solve it only approximately.

Finally, the optimality conditions given above are also useful for deriving *characterizations* for local solutions to the Chebyshev geometric element fitting problems. Given a feasible element, we may ask what geometrical distribution of the contacting points ensures that these conditions are satisfied. In this way, geometrical characterizations of local minima are derived in [13] for the minimum circumscribed, maximum inscribed and minimum zone circle problems, in [14] for the minimum zone line problem, and in [15] for the minimum circumscribed and maximum inscribed sphere problems. These results are as summarised in Section 4. Geometric interpretation of the optimality conditions for the line (3D), cylinder, cone and circle (3D) problems would appear to be much harder.

In the next two subsections we describe how the optimality conditions given above are used to construct algorithms for solving, respectively, the linearly and nonlinearly constrained problems.

5.1.2 An algorithm for the linearly constrained problems

We illustrate the algorithm using the minimum circumscribed circle problem, for m data points.

The algorithm employs a sequence of *exchanges* in order to generate a sequence of circles which "converge" to a solution. At any particular stage we have a circle which is feasible with respect to the data, *ie* circumscribes the data, and three identified contacting points. Based on the solution to the Kuhn-Tucker equations corresponding to these points, we *drop* one of the points from the active constraint set. It is easy to see that the solution to the equality constrained sub-problem defined by the two remaining contacting points is the circle for which these points define a diameter. However, this circle may violate some of the other constraints. Therefore, the centre of the new iterate is chosen to lie on the line joining the original centre to the centre defining this minimum in such a way that the new circle is feasible and has a smaller radius. An $O(m)$ search through the points tells us where the new centre should lie and whether a point should be added to the active set. We repeat the procedure until the optimality conditions described before are satisfied.

Since there is a finite number of possible choices of contacting points and provided the radius is *strictly decreased* at each iteration, the algorithm is finite. Occasionally, however, the radius may remain constant from one iteration to the next, so there is the possibility of cycling. This can occur, for example, when there are contacting points in addition to the three identified contacting points, and an iteration results in an exchange among these points with no change in the circle parameters. In active set methods for constrained optimization this situation is known as *degeneracy*. We cope with degeneracy in the following way. Whenever the radius is not strictly decreased we identify the *complete* set of active constraints and choose from those a subset for which either (a) the optimality conditions are satisfied, or (b) a strict decrease in the radius is guaranteed at the next iteration. The choice of an appropriate subset can be made using either geometrical arguments (*vide eg* [13]), or by interpreting suitably a constrained solution to the Kuhn-Tucker equations (*vide eg* [15]). The topic of degeneracy is explored further below within the context of nonlinearly constrained problems.

We present below a statement of a generic algorithm applicable to the linearly constrained problems. So as not to complicate the statement of the algorithm, a test for degeneracy and the consequent action required to cope with degeneracy are not included.

I Given parameters $\boldsymbol{u}$ defining a feasible element, determine the active constraints I_0^a.

II Solve the Kuhn-Tucker equations defined by $\boldsymbol{u}$ and I_0^a to give Lagrange multipliers $\boldsymbol{\lambda}$.

III If $\lambda_j < 0$ for some $j \in I_0^a$, drop the corresponding constraint from the active set: $I^a = I_0^a \backslash \{j\}$.[2]

IV Find $\boldsymbol{p}$ such that $\boldsymbol{u} + \boldsymbol{p}$ solves

$$\min_{\boldsymbol{v}} F(\boldsymbol{v})$$

[2]If the number of constraints with a negative Lagrange multiplier is greater than one, j is chosen to be that constraint with the most negative Lagrange multiplier.

subject to

$$c_i(\boldsymbol{v}) = 0, \quad i \in I^a.$$

V Find the largest α satisfying $0 < \alpha \leq 1$ for which

$$F(\boldsymbol{u} + \alpha \boldsymbol{p}) < F(\boldsymbol{u})$$

and

$$c_i(\boldsymbol{u} + \alpha \boldsymbol{p}) \geq 0, \quad i \in I,$$

adjusting I^a as necessary.

VI Set $\boldsymbol{u} = \boldsymbol{u} + \alpha \boldsymbol{p}$, and return to Step I if the optimality conditions do not hold.

The algorithm presented above has the following properties. Firstly, the estimate of the solution generated at each iteration is *feasible* and each iteration is a *descent* step. Secondly, the algorithm is *finite*, and has complexity $O(km)$ where k is the number of iterations. Moreover, based on considerable computational experience and the use of a sensible (automatically determined) starting estimate, we expect $k \ll m$. Finally, the algorithm converges to a *local* solution which is dependent on the starting estimate that is used. However, for the minimum circumscribed circle and sphere problems, the algorithm can be shown to converge to the *global* solution independently of the starting estimate. This follows from the observation that these particular problems possess a unique local minimum that is therefore the global solution.

5.1.3 An algorithm for the nonlinearly constrained problems

In order to develop an algorithm for the nonlinearly constrained problems, it is informative to consider the difficulties that can arise when the algorithm described above is applied to such a problem.

Firstly, because of the nonlinearity of the constraints, the equality constrained problem considered at Step IV is computationally more difficult and more expensive to solve. Consequently, we choose to solve the sub-problem only *approximately*: it is only solved exactly when we believe we have identified the correct set I^* of active constraints.

Secondly, since we solve the sub-problem at Step IV approximately, and because the step taken at Step V cannot follow the curvature of the constraints I^a, it is not possible to ensure that contact with the active constraints is maintained. It is therefore useful to consider a set of *working* constraints I^w which is used to predict those constraints that are active at the solution. I^w is constructed to contain I^a and those working constraints from the previous iteration for which the constraint value is sufficiently small.

We present below a statement of a generic algorithm applicable to the nonlinearly constrained problems: see, for example, [11, 7, 6]. Once again, we omit details of how degeneracy, which is discussed later, is handled.

I Given parameters $\boldsymbol{u}$ defining a feasible element, determine the set of active constraints I^a and a working set $I_0^w \supseteq I^a$.

II Solve the Kuhn-Tucker equations defined by $\boldsymbol{u}$ and I_0^w to give Lagrange multipliers estimates $\boldsymbol{\lambda}$.

III If $\lambda_j < 0$ for some $j \in I_0^w$, drop the corresponding constraint from the working set: $I^w = I_0^w \backslash \{j\}$.[3]

IV Find $\boldsymbol{p}$ such that $\boldsymbol{u} + \boldsymbol{p}$ approximately solves

$$\min_{\boldsymbol{v}} F(\boldsymbol{v})$$

subject to

$$c_i(\boldsymbol{v}) = 0, \quad i \in I^w.$$

If $\boldsymbol{p}$ is a feasible descent direction, proceed to Step V. Otherwise, calculate any descent direction that does not decrease the active constraint values, *ie* solve the linear problem

$$\boldsymbol{g}^T\boldsymbol{p} < 0$$

subject to

$$\boldsymbol{a}_i^T\boldsymbol{p} \geq 0, \quad i \in I^a.$$

V Find α satisfying $0 < \alpha \leq 1$ for which

$$F(\boldsymbol{u} + \alpha\boldsymbol{p}) < F(\boldsymbol{u})$$

and

$$c_i(\boldsymbol{u} + \alpha\boldsymbol{p}) \geq 0, \quad i \in I,$$

adjusting I^w as necessary.

VI Set $\boldsymbol{u} = \boldsymbol{u} + \alpha\boldsymbol{p}$, and return to Step I if the optimality conditions are not satisfied.

The algorithm has the following properties. Firstly, the estimate of the solution generated at each iteration is *feasible*, and each iteration is a *descent* step. Secondly, the algorithm is *finite*. Moreover, the equality constrained problem considered at Step IV is solved in such a fashion that once the constraints I^* active at a solution have been identified ($I^w = I^*$), convergence to a solution is quadratic. The behaviour of the algorithm away from the solution depends on the strategy for determining I^w. If I^w is chosen to be I^a, convergence is likely to be slow. Conversely, if I^w is too large, the step $\boldsymbol{p}$ defined at Step IV is unlikely to give a descent. Finally, the algorithm converges to a *local* solution which is dependent on the starting estimate that is used.

We conclude with some remarks concerning *degeneracy* which, as for the linearly constrained problems, requires explicit attention within the algorithm presented above. We say that the current iterate $\boldsymbol{u} \in \mathcal{R}^n$ is degenerate if the constraint gradient vectors $\nabla c_i(\boldsymbol{u})$, $i \in I^w$, are linearly dependent (or, equivalently, if the matrix of such vectors is column rank deficient). Clearly, one cause of degeneracy is the existence of more than n active constraints, where n is the number of parameters for which we are solving. In this situation it is possible for the "feasible-descent" algorithms that we have described to cycle. Such behaviour has already been noted in the context of the minimum circumscribed circle problem. However, even for the nonlinearly constrained problems, it is a simple matter to construct examples for which the constraint gradient vectors associated with n *or fewer* active constraints are linearly dependent. This form of degeneracy is algorithmically much harder to deal with. Moreover, examples exist where local solutions to the problems are characterized by fewer than n active constraints that

[3] As before, if the number of constraints with a negative Lagrange multiplier is greater than one, j is chosen to be that constraint with the most negative Lagrange multiplier.

are linearly dependent. We must ensure that the algorithm is able to converge onto, and indeed recognise, such degenerate solutions.

The effects of degeneracy on the algorithm presented are as follows. Firstly, because the Kuhn-Tucker equations are column rank deficient, it is not possible to determine at Step II unique Lagrange multiplier estimates, and consequently it is not possible to perform Step IV which relies on the availability of such estimates. Secondly, the convergence of the algorithm to a degenerate solution defined by at most n active constraints can be very poor.

In order to overcome these difficulties we systematically control the column rank of the constraint gradient matrix in order that Lagrange multiplier estimates are always available. Moreover, when estimates are not uniquely defined, we aim to find a solution to the Kuhn-Tucker equations which satisfies the non-negativity conditions (9) if this can be achieved. The strategy for constructing the working set of constraints I^w is also used to influence the numerical properties of the algorithm with respect to degeneracy. In these ways we are able to cope with degeneracy without degrading the performance of the algorithm.

5.1.4 A projection method

We now describe an algorithm that is applicable to the three cylinder problems. First, consider a set of data points nominally representative of a cylinder. Project these points orthogonally onto a given plane. It is clear that the minimum circumscribed circle to this projected data is the projection of the minimum circumscribed cylinder *with axis orthogonal to the given plane* to the original data. For, if there were a smaller minimum circumscribed cylinder with axis in this direction, then its projection would be a smaller circle circumscribing the projected data. Similar remarks apply to the minimum zone and maximum inscribed cylinders. So, if the cylinder axis direction can be found, its radius can be found by solving a corresponding circle problem. Thus, the algorithm is to choose an axis direction, project the data in this direction, find the radius to solve the corresponding circle problem for the projected data and optimise this radius (with respect to the axis direction). This optimisation can be performed by, for example, the Nelder and Mead polytope algorithm [12].

5.1.5 A "linear" method

If the minimax problem for a line in 2D is posed in terms of deviations in a fixed direction (parallel to the y–axis, say) rather than orthogonal to the line itself, then the objective function is linear in the line parameters. This "linear" Chebyshev line is found by minimizing a linear function subject to linear constraints, using, for example, the algorithm of Barrodale and Phillips [3]. This is computationally less expensive than determining the true Chebyshev line. If the data is rotated so that the line is approximately horizontal, then the linear and true Chebyshev lines are often characterized by the same three points. A condition for this equivalence can be expressed in terms of the minimum separation in the x–direction of the data and the (estimated) Chebyshev separation.

5.2 COMBINATORIAL METHODS

Because there is an essentially discrete aspect to the problems considered here, it follows that to some extent all solution methods are combinatorial. Thus, in the exchange method described

above, we are looking for the set of contacting points for a particular element. However, the means employed to determine these points involve the analysis of functions of continuous variables. In this section we describe methods where the emphasis is more on the discrete processes.

5.2.1 An essential subset method

Consider again the problem of finding the minimum circumscribed circle, or equivalently the minimum covering disc. We know that this is defined either by two points forming the ends of a diameter or by three points on the circumference forming a triangle that encloses the centre (*ie* an acute angled triangle). Clearly, a possible algorithm is to examine such discs for each pair and triple of data points and to choose the smallest that covers all the remaining data points. Equally clearly, although the calculation of any one circle is not computationally expensive, the number of discs to be examined becomes prohibitively large as the size of the data set increases. Consequently, this process of complete enumeration, although it guarantees finding the global solution, is unacceptable even in the context of *reference* software. However, if the number of pairs and triples to be examined can be substantially reduced then the process becomes more attractive. Such a reduction is possible using the notion of *essential subsets.* In Chebyshev fitting not every data point necessarily affects the solution, *eg* points near the centre of a minimum covering disc have no effect on the solution. It follows that there may exist subsets of a data set that have the same solution, for a particular element, as the entire set. If such a subset is minimal with respect to this property (*ie* no proper subset of it has the same solution) then it is termed an essential subset, for that element. An essential subset as defined is not necessarily unique.

In the following description of the resulting algorithm, an element means the particular element we wish to fit. Let D be a finite data set representative of a particular element. First, we select an initial subset $D_0 \subset D$ and determine G_0, where G_0 is a set of elements of width zero covering D_0. If some point $q \in D$ is not covered by every $g \in G_0$, we form $D_1 = D_0 \cup \{q\}$ and determine, by complete enumeration as described above, the set G_1 of elements of globally minimum width covering D_1. Then we determine an essential subset E_1 of D_1 as follows. First set $E_1 = D_1$ and then omit points one at a time and check, by enumeration as before, whether each successive omission changes the global solution. If it does not the point is excluded from the essential subset. If some point $q \in D$ is not covered by G_1 we form $D_2 = E_1 \cup \{q\}$ and determine G_2 and an essential E_2 of D_2. This procedure is repeated until G_j covers D, *ie* E_j is an essential subset of D. The index j indicates the number of repetitions of the covering procedure. G_j consists of all elements covering D as well as E_j with minimum width. Hence, G_j delivers all global solutions of the minimum inclusion problem for D by the particular element. E_j gives further information suitable for an analysis of the result.

This algorithm can be applied to any element for which a sufficiently small upper bound, N, can be set on the number of contacting points that need be considered (*eg* $N = 3$ for the minimum covering disc) and for which all local minima can be determined for up to n data points. For the present, because of the observations made at the end of Section 4, it can be seen that the elements that can be fitted by this algorithm are stripe, plate, disc, annulus, sphere and spherical shell, *ie* those expressible as linearly constrained problems.

Distributions of data can be devised for which the essential subsets contain all the data points. In these cases the method can take longer than complete enumeration of the data. This is because the method looks at every choice of some small number of points from the current essential subset, and the subset is built up a point at a time. However, for practical data the

method can be very efficient.

5.2.2 Convex hull

For the circumscribed elements, the minimum zone 2D line and the minimum zone plane, the contacting points of a feasible solution lie on the convex hull of the data. So it might appear advantageous to compute the convex hull initially and apply a chosen method only to this reduced set of points. This course has not been taken in the project, principally because the engineering components and data that we are concerned with are very accurate, so in many cases a large proportion of the data points will be vertices of the hull. Consequently, the reduction in the number of points to be dealt with is not sufficiently great to compensate for the computational cost of forming the hull. In any case, we are unaware of readily available, suitable software to form a 3D convex hull.

The hybrid algorithm for 2D lines uses, as a last resort, complete enumeration of the edges of the convex hull: the edges are taken one at a time as candidates for one of the pair of zone lines and the greatest orthogonal distance to a hull vertex is found. The line–point pair with smallest separation is the desired global minimum zone solution.

6 SOFTWARE IMPLEMENTATION

In order to avoid giving preferential advantage to some commercial producers of metrology software, it was originally intended that the algorithms should be implemented in a language not normally available on coordinate measuring machines. The chosen language was Matlab [4] [10]. This provides ready access to large amounts of well tried numerical software, in particular for linear algebra. It has the further advantage as a means of disseminating the methods developed in that Matlab statements look sufficiently similar to mathematical notation. However it was thought to be less suitable for the essential subset algorithm for which TurboPascal [5] was preferred. This language was then used for the other two pieces of software produced at PTB.

Table 1 (page 17) indicates the language and method implemented for each of the elements. We give here some details of the implementations.

MZ line (2D): the hybrid algorithm proceeds as follows. First the "linear" problem is solved (Section 5.1.5). The solution is checked to see whether it is that of the required Chebyshev problem. If this is not the case, use is made of the characterization in the exchange algorithm (Section 5.1.2) to yield a local solution. Finally, if this is not sufficient, a convex hull finder is employed to reduce the data and provide connectivity information, in order to guide an efficient search which is guaranteed to return the global solution (Section 5.2.2).

MI circle (2D): the greatest distance from a data point to the initial estimate of the centre is calculated. If at any iteration of the exchange algorithm the radius of the inscribed circle exceeds this value, the software reports failure to find a solution. This is because it is judged that the centre of the circle has "escaped" from the data and the size of the radius will diverge. This constraint is an easily implemented alternative to requiring the centre of the circle to lie within the convex hull of the data.

[4] Matlab is actually an interactive program, but looks to the user very much like a language.

[5] A version of Pascal, copyright Borland International, USA.

A similar constraint is applied to the

MI sphere and

MI cylinder.

Parametrization. In all fitting processes it is essential that appropriate, well-conditioned parametrizations are used. In this project the parametrizations recommended for complete elements in [2] were used, with the following exceptions:

1. for the cone the parametrization for large apex angle was not used. This means that the software is suitable for cones with apex angle not exceeding about 0.9π radians.
2. for the minimum circumscribed and minimum zone cylinders the point on the axis is defined as the intersection of the axis and the projection plane containing the origin of the coordinate axes. In this case it is advisable to shift the origin so that it lies near the centroid of the data.

Tolerances. Because of the finite precision of computers some tolerance must be allowed in deciding whether, for example, a point contacts an element. In most of the software such tolerances are determined relative to the size (*eg* radius) of the element, the precision of the computer and the accuracy of the data (assumed to be of the order of micrometres). However, in the essential subset and projection methods the tolerances are not relative to the size, so the data should ideally be scaled so that the size is approximately unity.

Initial estimates. All the methods used, except that based on essential subsets, require an initial estimate of the element sought in order to start the iterative process. Most of the implementations allow the user to supply an estimate and a few require him to do so. The estimates are in terms of the parameters rather than contacting points. Not all the parameters defining an element are always required; for example, for circles and spheres only the centre coordinates are required and for cylinders only the direction of the axis. For the cone, besides this direction an estimate of the apex angle is needed. For data realistically representative of the desired geometric form the choice of initial estimate is not likely to be critical. Methods implemented include: centroid of the data (for centres), join of extreme points (axis of cylinder) and least squares fits. The core of the torus is determined by first fitting a least squares plane to the data and then a least squares circle to the data's projection into that plane.

7 INDEPENDENT TESTING AND EVALUATION

The software and documentation developed during this project were produced at NPL and PTB. Apart from "in-house" testing by the producers, an "independent" testing and evaluation was carried out at CWI for all modules. This was part of the project's "quality regime" to ensure that the produced reference software is of the highest possible quality.

In accord with the project contract, all software is available (through the BCR) to independent assessors. Our testing of the software developed in the project (and, in relation to this, testing the improvements that resulted from earlier testing experience) should, therefore, minimize the possibility that, in an independent assessment, unexpected problems would arise. We have used the experience of the testing for all modules to improve

- details of the method implemented
- portability, reliability and usability of the software

- the documentation.

In the next subsections we give details of the testing strategy in general and of the results of testing software for specific geometric elements.

7.1 GENERAL ASPECTS OF TESTING

The testing and evaluation of modules developed in the project consisted of several activities, from which we mention here:

algorithm study: characteristics of the solution method were identified that enabled special data sets to be constructed with possible difficulties for the employed method. Branches were analysed in order to design suitable test data.

input testing: calls were made to ensure that input satisfying the specifications was always accepted and that illegal input (not defining an element-fitting problem) would lead to a comprehensible reaction.

output testing: the usability of the output was investigated.

performance testing: for a large number of data sets, partly well-chosen and partly randomly generated departing from geometric elements with or without perturbations: it was verified that the solution was found and that the output contained all information that the software claims to compute.

appropriateness of the tests: as far as economically possible the tests devised sufficiently explored all software paths of the implemented method.

portability: the testing at different sites was a means for discovering and removing non-portable elements of the coding.

Different ways have been followed to ensure that the output indeed represents a solution. In Table 1 the main approach is indicated, but usually more methods were applied for evaluating the results of test runs.

These were:

feasibility checking: for all modules, it was verified by additional computations that the returned parameters representing a geometric element actually defined a feasible solution.

verification of mathematical properties: such properties, that were verified in the output, are the *convex hull property* for zone line and zone plane, and the acute triangle formed by the contacting points of a circumscribed circle.

comparison: for test data derived by departing from a given geometric element, the solution is the original element or a better one (a smaller minimum). Otherwise, a solution is not always known in advance. In this case, often a comparison was possible of the results returned by different solution methods.

reference solutions: for particularly difficult geometric elements, additional software was developed (in Matlab) for computing (usually at very high cost) guaranteed best solutions through an exhaustive search, and this software was then applied for a selection of data sets (chosen or generated).

7.2 TEST DATA

Matlab modules were developed for generating large collections of test data. The modules have been designed (per geometric element) to produce sets of different size and different positioning in 2D or 3D space. With these modules both sets with data points lying exactly on a (randomly positioned) geometric element and points obtained by randomly perturbing their positions were generated.

In addition to this, as has been mentioned, data sets were chosen to look for particular difficulties of the methods and in order to walk through different branches of the algorithms. Further, for assessing the robustness, tests were always made with data sets (usually non-realistic) that had been used for other geometric elements. Among these were the data sets developed for Phase II of the BCR intercomparison of least squares form assessment software [4].

7.3 TESTING SPECIFIC GEOMETRIC ELEMENTS

Table 1 gives a survey of the different geometric elements tested in the project.

For the zone line, circumscribed circle and zone circle in 2D and circumscribed sphere (3D), alternative software in TurboPascal, and for the zone plane and zone cylinder in 3D alternative software in Matlab, has also been tested. The alternatives were used in particular for verifying the different implementations by comparison.

7.4 CONCLUSION

Initial phases in the testing of the developed software have led to many improvements, which in itself has already demonstrated that the followed "quality regime" of the project was highly useful.

For all modules extensive testing ensured that the software met the requirements and that its performance was in agreement with the documentation. For acceptable data sets solutions (representing local minima) were returned without failure. Moreover, for realistic data sets an excellent rate of global minima were returned in a most efficient way.

8 EPILOGUE

The aim of the project to produce reference software for the 13 elements indicated in Section 3 has been achieved. We believe that the algorithms devised to this end could be readily implemented in formats suited to the processors that control coordinate measuring machines.

The constraints of working to a timetable for the delivery of algorithms to the funding body (BCR) has meant that knowledge gained later in the project could not always be used to improve earlier software. With hindsight, we would prefer to offer software implementing the essential subsets method, with its guarantee of global solutions, where it can be applied and the exchange method for the remaining elements, minimum inscribed circle and sphere, the 3 cylinders, the cone and the torus. This would represent a more uniform approach and the software could then be designed to be more compact.

Table 1 Algorithms implemented for the geometric elements.

space	*geometric element*	*method*	*implementation language*	*main testing approach*
$2D$	MZ line	hybrid	Matlab	mathematical property: hull intersection
	MC circle	exchange	Matlab	mathematical property: acute triangle
	MI circle	exchange	Matlab	reference solutions
	MZ circle	exchange	Matlab	mathematical property: crossing chords
$3D$	MZ plane	essential subset	TurboPascal	mathematical property: hull intersection
	MC sphere	exchange	Matlab	mathematical property: acute tetrahedron
	MI sphere	exchange	Matlab	reference solutions
	MZ sphere	essential subset	TurboPascal	mathematical property: contacting points
	MC cylinder	projection	TurboPascal	reference solutions
	MI cylinder	minimization	Matlab	comparisons
	MZ cylinder	projection	TurboPascal	reference solutions
	MZ cone	minimization	Matlab	comparisons
	MC torus	minimization	Matlab	comparisons

Key: MC: minimum circumscribed, MI: maximum inscribed, MZ: minimum zone.

All the software developed in this project is available within the EC from BCR, initially *via* the project partners but eventually probably from a metrology software repository currently being investigated under a further BCR project.

9 ACKNOWLEDGEMENT

We thank our colleague Professor G T Symm (NPL) for commenting on a draft of this report.

10 REFERENCES

[1] G T Anthony, H M Anthony, B Bittner, B P Butler, M G Cox, R Drieschner, R Elligsen, A B Forbes, H Groß, S A Hannaby, P M Harris, and J Kok. *Chebyshev Reference Software for the Evaluation of Coordinate Measuring Machine Data.* Commission of the European Communities, BCR Report EUR 15304 EN, Luxembourg, 1993.

[2] G. T. Anthony, H. M. Anthony, M. G. Cox, and A. B. Forbes. *The Parametrization of Geometric Form.* Commission of the European Communities, BCR Report EUR 13517 EN, Luxembourg, 1991.

[3] I. Barrodale and C. Phillips. Algorithm 495: Solution of an overdetermined system of linear equations in the Chebyshev norm. *ACM Trans. Math. Software*, 1:264–270, 1975.

[4] R. Drieschner, B. Bittner, R. Elligsen, and F. Wäldele. *Testing Coordinate Measuring Machine Algorithms: Phase II.* Commission of the European Communities, BCR Report EUR 13417 EN, Luxembourg, 1991.

[5] J. Elzinga and D. W. Hearn. The minimum covering sphere problem. *Mgmt. Sci*, 19:96–104, 1972.

[6] R. Fletcher. *Practical Methods of Optimization Volume 2: Constrained Optimization.* John Wiley and Sons, Ltd., New York, 1981.

[7] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press, London, 1981.

[8] D. W. Hearn and J. Vijay. A geometrical solution for the (weighted) minimum circle problem. Technical Report 81–2, Industrial and Systems Engineering Department, University of Florida, 1981.

[9] D. W. Hearn and J. Vijay. Efficient algorithms for the (weighted) minimum circle problem. *Operations Research*, 30:777–795, 1982.

[10] C. Moler, J. Little, and S. Bangert. *PRO-MATLAB, User's Guide.* The MathWorks, Inc., South Natick, MA, USA, 1988.

[11] W. Murray and M. L. Overton. A projected Lagrangian algorithm for nonlinear minimax optimization. *SIAM Journal for Scientific and Statistical Computing*, 1(3):345–370, 1980.

[12] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7:308–313, 1965.

[13] NPL. *Chebyshev Approximation to Data by Circles.* National Physical Laboratory, Teddington, UK, 1991.

[14] NPL. *Chebyshev Approximation to Data by Lines.* National Physical Laboratory, Teddington, UK, 1991.

[15] NPL. *Chebyshev Approximation to Data by Spheres.* National Physical Laboratory, Teddington, UK, 1992.

[16] C. Porta and F. Wäldele. *Testing of Three Coordinate Measuring Machine Evaluation Algorithms.* Commission of the European Communities, BCR Report EUR 10909 EN, Luxembourg, 1986.

[17] J. J. Sylvester. On Poncelet's approximate linear valuation of surd forms. *Philosophical Magazine*, 20:203–222, 1860.