DAMTP/1997/NA21

An optimal way of moving a sequence of points onto a curve in two dimensions[1]

M.J.D. Powell

Abstract: Let $\underline{s}(t)$, $0 \le t \le T$, be a smooth curve and let $\underline{x}_i$, $i=1,2,\ldots,n$, be a sequence of points in two dimensions. An algorithm is given that calculates the parameters t_i, $i=1,2,\ldots,n$, that minimize the function $\max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 : i=1,2,\ldots,n\}$ subject to the constraints $0 \le t_1 \le t_2 \le \cdots \le t_n \le T$. Further, the final value of the objective function is best lexicographically, when the distances $\|\underline{x}_i - \underline{s}(t_i)\|_2$, $i=1,2,\ldots,n$, are sorted into decreasing order. The algorithm finds the global solution to this calculation. Usually the magnitude of the total work is only about n when the number of data points is large. The efficiency comes from techniques that use bounds on the final values of the parameters to split the original problem into calculations that have fewer variables. The splitting techniques are analysed, the algorithm is described, and some numerical results are presented and discussed.

Department of Applied Mathematics and Theoretical Physics,
University of Cambridge,
Silver Street,
Cambridge CB3 9EW,
England.

December, 1997 (revised May, 1998).

[1]This paper is dedicated to Olvi Mangasarian in celebration of his 65th birthday. The author is one of many people in optimization who appreciate greatly the quality of his academic work, his friendship, and his willingness to discuss research.

1. Introduction

If two pictures of a scene are taken at two different times, then differences may occur, not only because of changes in the scene, but also because of the way in which the pictures are taken. A highly useful technique that compensates for the external changes requires the identification of several fixed points of the scene that occur in both pictures. Then one seeks the smoothest function that maps the fixed points of the first picture into the fixed points of the second picture. This mapping is applied to all the data in the first picture. A comparison of the resultant image with the second picture is often far more revealing than a comparison of the original two pictures. This method is called "image registration". It has applications in medicine, in the analysis of data from satellites, and in mine detection, for example. Further information can be found in Brown (1992), Flusser (1992) and Barrodale, Kuwahara, Poeckert and Skea (1993).

Sometimes there are not enough fixed points to identify a suitable mapping function, but one may be able to make use of one or more fixed curves in the scene, such as part of a rib-cage in medical imaging. Therefore the author has developed a procedure for mapping curves into curves in two dimensions (Powell, 1996). Each curve in the first picture is replaced by a sequence of points on the curve, but the corresponding curve in the second picture is approximated by pieces of circular arcs and straight line segments that are joined to provide first derivative continuity. Let the sequence and the approximation be $\{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_n\} \subset \mathcal{R}^2$ and $\mathcal{S} = \{\underline{s}(t) : 0 \leq t \leq T\} \subset \mathcal{R}^2$, respectively, where the parameter t can be regarded as the distance along $\mathcal{S}$. Then the mapping function is required to have the property that, for $i=1,2,\ldots,n$, the image of $\underline{x}_i$ is $\underline{s}(t_i)$ for some t_i in $[0,T]$. Further, the ordering of the sequence of points is preserved by imposing the constraints

$$0 \leq t_1 \leq t_2 \leq \cdots \leq t_n \leq T. \tag{1.1}$$

For any choice of the parameters subject to the conditions (1.1), one can regard $\underline{s}(t_i)$ as the required image of $\underline{x}_i$, $i = 1,2,\ldots,n$, in order to apply the standard image registration method that is the subject of the first paragraph. The smoothness of the resultant mapping function depends on the choice of t_i, $i = 1,2,\ldots,n$. Powell (1996) adjusts the parameters to the values that make the mapping function as smooth as possible, which is an interesting optimization calculation in n variables that has a differentiable objective function and the linear constraints (1.1). The initial values of the variables are chosen by an algorithm that solves the subproblem that is stated in the next paragraph. The author was delighted to find that the solution of the subproblem requires very little computation, which may be useful to other applications. Therefore we are going to consider some of the details, theoretical properties and numerical results of the method that generates the initial values of t_i, $i=1,2,\ldots,n$.

The given sequence $\{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_n\} \subset \mathcal{R}^2$ is moved onto the given smooth curve $\mathcal{S} = \{\underline{s}(t) : 0 \leq t \leq T\} \subset \mathcal{R}^2$ in a way that preserves the ordering and that minimizes the largest of the changes to the positions of the individual points. In other words, the algorithm calculates the real numbers t_i, $i = 1, 2, \ldots, n$, that minimize the function

$$f(\underline{t}) = \max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 : i = 1, 2, \ldots, n\}, \qquad \underline{t} \in \mathcal{R}^n, \tag{1.2}$$

subject to the constraints (1.1). Further, if there is not a unique solution, then the freedom is taken up by the well-known lexicographic method of Chebyshev. Specifically, for any feasible $\underline{t}$, we let $\{d_1(\underline{t}), d_2(\underline{t}), \ldots, d_n(\underline{t})\}$ be the distances $\|\underline{x}_i - \underline{s}(t_i)\|_2$, $i = 1, 2, \ldots, n$, arranged in descending order, so the definition (1.2) gives $f(\underline{t}) = d_1(\underline{t})$. Then two feasible vectors, $\hat{\underline{t}}$ and $\check{\underline{t}}$ say, are equally good only if all the equations $d_i(\hat{\underline{t}}) = d_i(\check{\underline{t}})$, $i = 1, 2, \ldots, n$, are satisfied. Otherwise, letting j be the least integer such that $d_j(\hat{\underline{t}}) \neq d_j(\check{\underline{t}})$, we assume that $\hat{\underline{t}}$ is better than $\check{\underline{t}}$ if and only if $d_j(\hat{\underline{t}})$ is less than $d_j(\check{\underline{t}})$. The algorithm generates a feasible vector $\underline{t} \in \mathcal{R}^n$ that is best in this sense. Local minima can occur, because $\mathcal{S}$ can be any curve with a continuous direction that is constructed from the circular arcs and straight line segments that have been mentioned already. Therefore $\mathcal{S}$ is allowed to have many fluctuations and to intersect itself several times. Fortunately, such situations are handled routinely by the algorithm in a way that always provides a global solution to the calculation of this paragraph.

The method depends on the remark that the objective function (1.2) is an upper bound on the distances $\|\underline{x}_i - \underline{s}(t_i)\|_2$, $i = 1, 2, \ldots, n$, that is achieved for at least one value of i. Therefore it is very different from any method that would allow expression (1.2) to be replaced by the sum of squares $\sum_{i=1}^{n} \|\underline{x}_i - \underline{s}(t_i)\|_2^2$.

The main ideas and lemmas that are important to the algorithm are given in Section 2. Details of the algorithm are addressed in Section 3, and some numerical results are presented in Section 4. We find that it is usual for the total amount of computation to be only of magnitude n, but this work can be $\mathcal{O}(n^2)$, when typical spacings between adjacent points in the sequence $\underline{x}_i$, $i =, 2 \ldots, n$, are much less than the distances of most of the points from $\mathcal{S}$. A picture of a calculation is also given in Section 4.

2. Some properties of the calculation

Let h^* be the least value of the objective function (1.2) subject to the constraints (1.1). The algorithm picks several estimates of h^*, and for about half of them it generates the numbers $\alpha_i(h)$, $i = 0, 1, \ldots, n$, sequentially, where h is the estimate. Specifically, $\alpha_0(h)$ is zero, and, for each $i \geq 1$, we let $\alpha_i(h)$ be the least number in the closed interval $[\alpha_{i-1}(h), T]$ that satisfies the inequality

$$\|\underline{x}_i - \underline{s}(\alpha_i(h))\|_2 \leq h, \tag{2.1}$$

except that $\alpha_i(h)$ is given the value $+\infty$ if $\alpha_{i-1}(h)$ is already $+\infty$ or if the distance from $\underline{x}_i$ to all the points $\{\underline{s}(t) : \alpha_{i-1}(h) \leq t \leq T\}$ is greater than h. It follows that

$\alpha_n(h) \leq T$ or $\alpha_n = +\infty$ occurs in the cases $h \geq h^*$ or $h < h^*$, respectively. This remark is so important to the algorithm that it is presented as a lemma.

Lemma 1 If $\alpha_n(h) \leq T$ is achieved, then h has the property $h \geq h^*$. Further, $\alpha_i(h)$ is a lower bound on the parameter t_i^* for every integer i in $[1, n]$, where $\underline{t}^* \in \mathcal{R}^n$ is any solution of our optimization calculation. Alternatively, if $\alpha_n(h) = +\infty$ occurs, then h is strictly less than h^*.

Proof It is sufficient to prove that $h \geq h^*$ and $h < h^*$ imply $\alpha_n(h) \leq T$ and $\alpha_n(h) = +\infty$, respectively. In the case $h \geq h^*$, we let $\underline{t}^*$ be any optimal vector of parameters. Therefore the components of $\underline{t}^*$ satisfy the inequalities

$$0 \leq t_1^* \leq t_2^* \leq \cdots \leq t_n^* \leq T \quad \text{and} \quad \|\underline{x}_i - \underline{s}(t_i^*)\|_2 \leq h, \quad i = 1, 2, \ldots, n. \tag{2.2}$$

Hence condition (2.1) and $\alpha_{i-1}(h) \leq \alpha_i(h) \leq T$ are achieved for $i = 1$ if we set $\alpha_1(h) = t_1^*$. Thus the definition of $\alpha_1(h)$ implies $\alpha_1(h) \leq t_1^*$. It now follows from expression (2.2) that condition (2.1) and $\alpha_{i-1}(h) \leq \alpha_i(h) \leq T$ are achieved for $i = 2$ if we set $\alpha_2(h) = t_2^*$, so the definition of $\alpha_2(h)$ provides $\alpha_2(h) \leq t_2^*$. By continuing this argument inductively, we deduce the required bounds $\alpha_i(h) \leq t_i^*$, $i = 1, 2, \ldots, n$, which imply $\alpha_n(h) \leq T$. Thus the lemma is true in the case $h \geq h^*$.

Alternatively, if $h < h^*$, it follows from the definition of h^* that it is not possible for inequality (2.1) to hold for every i with the ordering condition $0 \leq \alpha_1(h) \leq \alpha_2(h) \leq \cdots \leq \alpha_n(h) \leq T$. Therefore $\alpha_n(h)$ must be infinite, which completes the proof. □

The lemma suggests a procedure for calculating h^* to arbitrarily high accuracy, because, by generating $\alpha_n(h)$, we can discover whether any positive number h satisfies $h \geq h^*$ or $h < h^*$. The procedure obtains a bracket on h^* that is refined by a bisection method, and, for each h, one works forwards through the points $\underline{x}_i$, $i = 1, 2, \ldots, n$, and along the curve $\mathcal{S}$, in order to calculate the sequence $\alpha_i(h)$, $i = 1, 2, \ldots, n$. There is an analogous way of working backwards through the data points and along $\mathcal{S}$. It is the subject of the following corollary of Lemma 1, because we can improve on the preliminary procedure by combining forwards and backwards directions.

Corollary 2 Let h be any nonnegative number and let $\beta_{n+1}(h)$ be T. For $i = n, n-1, \ldots, 1$, we let $\beta_i(h)$ be the greatest number in the interval $[0, \beta_{i+1}(h)]$ that satisfies the inequality

$$\|\underline{x}_i - \underline{s}(\beta_i(h))\|_2 \leq h, \tag{2.3}$$

except that we set $\beta_i(h) = -\infty$ if $\beta_{i+1}(h)$ has this value already or if all of the distances $\{\|\underline{x}_i - \underline{s}(t)\|_2 : 0 \leq t \leq \beta_{i+1}(h)\}$ exceed h. If $\beta_1(h)$ is finite, then $h \geq h^*$ and $t_i^* \leq \beta_i(h)$, $i = 1, 2, \ldots, n$, hold, where $\underline{t}^* \in \mathcal{R}^n$ is any optimal vector of parameters. Alternatively, the condition $\beta_1(h) = -\infty$ implies that h is strictly less than h^*. □

The proof of the corollary is omitted because it is analogous to the justification of Lemma 1.

We see that a bisection procedure for calculating h^* can apply either Lemma 1 or Corollary 2. Further, if the trial number h satisfies $h \geq h^*$, then the procedure provides either upper or lower bounds on the optimal parameters t_i^*, $i=1,2,\ldots,n$. The best available bounds on t_i^* for each i can be recorded and revised as the calculation proceeds. This task is straightforward, because the method of proof of Lemma 1 gives the relations

$$\alpha_i(h_1) \leq \alpha_i(h_2) \leq t_i^* \leq \beta_i(h_2) \leq \beta_i(h_1), \qquad i=1,2,\ldots,n, \tag{2.4}$$

for all numbers h_1 and h_2 such that $h^* \leq h_2 \leq h_1$. We reserve α_i and β_i for the best available bounds on t_i^*, $i=1,2,\ldots,n$. They are stored and updated explicitly by the computer program that produced the numerical results of Section 4.

The computer program employs both the forwards and backwards directions that have been mentioned, in order that the lengths of the intervals $[\alpha_i, \beta_i]$, $i=1,2,\ldots,n$, can become small. Thus it may happen during the calculation that the condition $\beta_j \leq \alpha_{j+1}$ is achieved for some integer j in $[1, n-1]$. Then the original optimization problem in n variables is split into two similar problems, where one of the new problems has j variables and the other one has n–j variables. Specifically, they are the minimization of the functions

$$\left.\begin{array}{l} \max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 : i=1,2,\ldots,j;\ 0 \leq t_1 \leq t_2 \leq \cdots \leq t_j \leq \beta_j\}, \ \text{ and} \\ \max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 : i=j+1, j+2,\ldots,n;\ \alpha_{j+1} \leq t_{j+1} \leq \cdots \leq t_n \leq T\} \end{array}\right\}. \tag{2.5}$$

We see that these calculations provide the optimal moves of the points $\{\underline{x}_i : i=1,2,\ldots,j\}$ and $\{\underline{x}_i : i=j+1, j+2,\ldots,n\}$ onto the curves $\{\underline{s}(t) : 0 \leq t \leq \beta_j\}$ and $\{\underline{s}(t) : \alpha_{j+1} \leq t \leq T\}$, respectively. Although it seems obvious that this splitting is valid, we present a formal proof.

Lemma 3 Let τ_i^*, $i=1,2,\ldots,j$, and τ_i^*, $i=j+1, j+2,\ldots,n$, be parameters that minimize the two expressions (2.5) in the lexicographic sense that is described in the penultimate paragraph of Section 1, and let β_j and α_{j+1} satisfy $\beta_j \leq \alpha_{j+1}$. Then the parameter values τ_i^*, $i=1,2,\ldots,n$, solve the original calculation.

Proof Let t_i^*, $i=1,2,\ldots,n$, be any solution of the original calculation. The bounds $t_j^* \leq \beta_j$ and $\alpha_{j+1} \leq t_{j+1}^*$ are satisfied and we are assuming $\beta_j \leq \alpha_{j+1}$. Therefore, if the first j components of $\underline{t}^*$ are replaced by the numbers τ_i^*, $i=1,2,\ldots,j$, that occur in the statement of the lemma, then the constraints (1.1) are preserved. Further, the choice of τ_i^*, $i=1,2,\ldots,j$, implies that the new vector $\underline{t}^*$ is also a solution of the original calculation, so we restrict attention to optimal vectors $\underline{t}^*$ that have the property $t_i^* = \tau_i^*$, $i=1,2,\ldots,j$. It follows similarly that $\underline{t}^*$ remains optimal if we replace its last $n-j$ components by τ_i^*, $i=j+1, j+2,\ldots,n$, which gives the required result. □

Our work so far suggests an algorithm that combines splitting with bracketing and bisection, where splitting occurs whenever the relation $\beta_j \leq \alpha_{j+1}$ is found for some integer j in $[1, n-1]$. It is very useful that, due to Lemma 3, all the current bounds of the form $\alpha_i \leq t_i^* \leq \beta_i$, $i=1,2,\ldots,n$, can be included in the new calculations that are introduced by the splitting technique. Some "one-sided" splitting is also possible, as indicated in the next lemma.

It assumes that α_i and β_i are always available, because $\alpha_i = 0$ and $\beta_i = T$, $i=1,2,\ldots,n$, can be set initially. Further, we introduce the name "h-problem" for a relaxed form of the original calculation, where h is any nonnegative number. Specifically, in this problem the function (1.2) is minimized subject to the constraints (1.1), and we take up some freedom in the variables by applying the lexicographic method only to the distances $\|\underline{x}_i - \underline{s}(t_i)\|_2$, $i=1,2,\ldots,n$, that are greater than h. Therefore any solution to the h_2-problem is also a solution to the h_1-problem if $0 \leq h_2 \leq h_1$, and the 0-problem is the original calculation.

Lemma 4 Let a value of h provide $\alpha_n(h) = +\infty$. If j is any integer in $[1, n-1]$ such that $\alpha_j(h) \leq \alpha_{j+1}$, then, for $i=1,2,\ldots,j$, the current lower bound α_i can be increased to $\max[\alpha_i, \alpha_i(h)]$. Moreover, let the parameters τ_i^*, $i=j+1, j+2, \ldots, n$, minimize the second part of expression (2.5) lexicographically as before, and let the other components of $\underline{\tau}^* \in \mathcal{R}^n$ be $\tau_i^* = \alpha_i(h)$, $i=1,2,\ldots,j$. Then $\underline{\tau}^*$ is a solution of the h-problem.

Proof Let $\underline{t}^*$ be any optimal vector of parameters for the original calculation. Then the largest of the distances $\|\underline{x}_i - \underline{s}(t_i^*)\|_2$, $i=1,2,\ldots,j$, cannot be reduced by adjusting t_i^*, $i=1,2,\ldots,j$, subject to $0 \leq t_1^* \leq \cdots \leq t_j^* \leq t_{j+1}^*$. Now the numbers $\alpha_i(h)$, $i=1,2,\ldots,j$, have the properties

$$\left.\begin{array}{l} 0 \leq \alpha_1(h) \leq \alpha_2(h) \leq \cdots \leq \alpha_j(h) \leq \alpha_{j+1} \leq t_{j+1}^* \\ \text{and} \qquad \|\underline{x}_i - \underline{s}(\alpha_i(h))\|_2 \leq h, \qquad i=1,2,\ldots,j \end{array}\right\}. \tag{2.6}$$

It follows from the possibility $t_i^* = \alpha_i(h)$, $i=1,2,\ldots,j$, that the first j components of $\underline{t}^*$ satisfy $\|\underline{x}_i - \underline{s}(t_i^*)\|_2 \leq h$, $i=1,2,\ldots,j$. Hence the definition of $\alpha_i(h)$ and the method of proof of Lemma 1 give $\alpha_i(h) \leq t_i^*$, $i=1,2,\ldots,j$, which justifies the assertion that the current lower bound on t_i^* can be increased from α_i to $\max[\alpha_i, \alpha_i(h)]$ for each integer i in $[1, j]$.

For the remainder of the proof, we let $\underline{t}^*$ be any solution of the h-problem. The conditions (2.6) imply that $\underline{t}^*$ remains a solution of the h-problem if we replace its first j components by $\tau_i^* = \alpha_i(h)$, $i=1,2,\ldots,j$. Therefore we restrict attention to vectors $\underline{t}^*$ that have the property $t_i^* = \tau_i^*$, $i=1,2,\ldots,j$. The proof is completed by applying the last part of the proof of Lemma 3. □

If the conditions of Lemma 4 are achieved by the algorithm, there is an immediate switch from the original problem to the minimization of the second function of expression (2.5), using the largest integer j in $[1, n–1]$ that satisfies $\alpha_j(h) \leq \alpha_{j+1}$.

Lemma 4 shows that, if t_i^* is an optimal parameter for the new calculation and if $\|\underline{x}_i - \underline{s}(t_i^*)\|_2 > h$ occurs, then t_i^* is also an optimal parameter for the original calculation. We treat the new problem, which has only $n-j$ variables, as though it were the original one, while seeking the value of h^*. Therefore further splittings are likely. If they reduce the number of variables to only one, which happens frequently, partly because of another splitting technique that will be described later, then we require the point on a known section of $\mathcal{S}$ that is closest to a particular data point, $\underline{x}_p$ say. This calculation is done analytically, using the fact that $\mathcal{S}$ is composed of straight line segments and circular arcs. Then t_p^* is set to the parameter value of the point that is found on $\mathcal{S}$. Another analytic calculation that determines h^* and t_p^* directly for some p is mentioned in Section 3. Otherwise, the bracket on h^* is refined until its width is less than a prescribed tolerance. Let the parameters that remain in the calculation after all the splittings be t_i, $i=p,p+1,\ldots,q$. Further, among all the values of h for which the numbers $\alpha_i(h)$, $i=p,p+1,\ldots,q$, are generated, let h_1 be the least value and h_2 be the greatest value that satisfy $h_1 \geq h^*$ and $h_2 < h^*$, respectively, which are identified using Lemma 1. The algorithm sets $t_p^* = \alpha_p(h_1)$. Therefore we consider the suitability of this choice.

We should avoid an unnecessarily large distance between $\underline{x}_p$ and $\underline{s}(t_p^*)$, but often the required condition $\|\underline{x}_i - \underline{s}(\alpha_i(h))\|_2 \leq h$ is achieved by satisfying the inequality as an equation, so the choice $t_p^* = \alpha_p(h_1)$ is likely to give $\|\underline{x}_p - \underline{s}(t_p^*)\|_2 = h_1$. Further, by constructing pathological examples, it can be shown that occasionally this choice can make the distance $\|\underline{x}_p - \underline{s}(t_p^*)\|_2$ much greater than necessary, even in the case $h_1 = h^*$. These examples contain degeneracies, however, due to nonuniqueness of the shifts of some of the data points that have to move the full distance h^*, and the achievement of optimality in difficult degenerate situations can require much computation. Therefore we justify $t_p^* = \alpha_p(h_1)$ by showing that it is optimal for a perturbed version of the current calculation.

The calculation after all the splittings is the minimization of the function

$$\max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 : i=p,p+1,\ldots,q;\ \alpha \leq t_p \leq t_{p+1} \leq \cdots \leq t_q \leq \beta\}, \tag{2.7}$$

where α and β are constants. We let the perturbed problem be the minimization of the expression

$$\max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 + \varepsilon\, \sigma(t_i - \alpha_i(h_1)) : i=p,p+1,\ldots,q;\ \alpha \leq t_p \leq \cdots \leq t_q \leq \beta\}, \tag{2.8}$$

in the case $\varepsilon = h_1 - h_2$, where σ is the piecewise constant function

$$\sigma(\theta)=0, \quad \theta \leq 0, \qquad \text{and} \qquad \sigma(\theta)=1, \quad \theta > 0. \tag{2.9}$$

It is not expensive in practice to refine the bracket on h^* until the magnitude of ε is comparable to computer rounding errors. Our analysis of the new problem depends on the assumption that the available values of $\alpha_i(h_1)$ and $\alpha_i(h_2)$, $i=p, p+1, \ldots, q$, do not allow a Lemma 4 splitting, which is the condition

$$\alpha_j(h_2) > \alpha_{j+1}(h_1), \qquad j=p, p+1, \ldots, q-1. \tag{2.10}$$

Lemma 5 Let the conditions of the previous paragraph be satisfied, and let t_i^*, $i=p, p+1, \ldots, q$, be any parameters that provide the least value of expression (2.8) subject to $\alpha \leq t_p^* \leq \cdots \leq t_q^* \leq \beta$. Then t_p^* has the value $\alpha_p(h_1)$.

Proof First we deduce that, if the parameters t_i^*, $i=p, p+1, \ldots, q$, are optimal for the perturbed problem, then they have the lower bounds

$$t_i^* \geq \alpha_i(h_1), \qquad i=p, p+1, \ldots, q, \tag{2.11}$$

and one or more of them holds as an equation. We require the remark that the possibility $t_i^*=\alpha_i(h_1)$, $i=p, p+1, \ldots, q$, shows that the least value of expression (2.8) is no greater than h_1. Further, the definition of $\alpha_i(h_1)$ implies that, if expression (2.11) fails for any i, then at least one of the distances $\|\underline{x}_j - \underline{s}(t_j^*)\|_2$, $j=p, p+1, \ldots, i$, is greater than h_1. It follows that all the conditions (2.11) are necessary for optimality. Next we suppose that they all hold as strict inequalities. Then expression (2.8) is the sum of expression (2.7) and the constant $\varepsilon = h_1 - h_2$, but this sum is bounded below by $h^* + h_1 - h_2 > h_1$, which also contradicts optimality. Hence $t_k^* = \alpha_k(h_1)$ is satisfied for one or more integers k in $[p, q]$.

We let k be as small as possible subject to this equation, which gives the required result if $k=p$. Otherwise, conditions (2.11), (1.1) and (2.10) provide the bounds

$$\alpha_{k-1}(h_1) < t_{k-1}^* \leq t_k^* = \alpha_k(h_1) < \alpha_{k-1}(h_2). \tag{2.12}$$

Therefore we can let ℓ be the least integer in $[p, k-1]$ that satisfies $\alpha_\ell(h_1) < t_\ell^* < \alpha_\ell(h_2)$. It follows from the optimality of t_ℓ^* and $\alpha_\ell(h_1) < t_\ell^*$ that we have the relation

$$h_1 \geq \|\underline{x}_\ell - \underline{s}(t_\ell^*)\|_2 + \varepsilon\, \sigma(t_\ell^* - \alpha_\ell(h_1)) = \|\underline{x}_\ell - \underline{s}(t_\ell^*)\|_2 + h_1 - h_2. \tag{2.13}$$

We are going to make use of the fact that this inequality is $\|\underline{x}_\ell - \underline{s}(t_\ell^*)\|_2 \leq h_2$. Now, by definition, $\alpha_\ell(h_2)$ is the least value of t in $[\alpha_{\ell-1}(h_2), \beta]$ that satisfies $\|\underline{x}_\ell - \underline{s}(t)\|_2 \leq h_2$, the value of $\alpha_{p-1}(h_2)$ being α. Therefore $t_\ell^* < \alpha_\ell(h_2)$ and $\|\underline{x}_\ell - \underline{s}(t_\ell^*)\|_2 \leq h_2$ imply $t_\ell^* < \alpha_{\ell-1}(h_2)$, which excludes $\ell=p$. In the other case $\ell > p$, the choice of k provides $\alpha_{\ell-1}(h_1) < t_{\ell-1}^*$, so we find $\alpha_{\ell-1}(h_1) < t_{\ell-1}^* \leq t_\ell^* < \alpha_{\ell-1}(h_2)$, but this conclusion contradicts the definition of ℓ. Thus all values of ℓ are excluded. The only surviving possibility is $t_p^* = \alpha_p(h_1)$, so the lemma is true. □

In addition to convenience and the optimality of a nearby problem, there are two more advantages of the choice $t_p^* = \alpha_p(h_1)$. One is that, because it is the leftmost reasonable candidate for selection in the interval $[\alpha, \beta]$, where we are using the notation of expression (2.7), it leaves as much freedom as possible for the selection of the final values of t_i, $i = p+1, p+2, \ldots, q$. The other advantage is that we expect $\alpha_p(h_1)$ to be very close to the optimal value $\alpha_p(h^*)$, because $\alpha_p(h)$ converges to $\alpha_p(h^*)$ as h tends to h^* from above. This assertion is proved as follows. By definition, $\alpha_p(h)$ increases monotonically as h decreases. Therefore $\alpha_p(h)$ tends to a limit, α_p^* say, that satisfies $\alpha_p^* \leq \alpha_p(h^*)$. Furthermore, the conditions $\|\underline{x}_p - \underline{s}(\alpha_p(h))\|_2 \leq h$ and $h \to h^*$ imply $\|\underline{x}_p - \underline{s}(\alpha_p^*)\|_2 \leq h^*$. Hence the definition of $\alpha_p(h^*)$ gives $\alpha_p(h^*) \leq \alpha_p^*$. It follows that α_p^* is equal to $\alpha_p(h^*)$ as claimed.

The analogue of Lemma 4 for the upper bounds β_j on t_j^*, $j = 1, 2, \ldots, n$, is that, if a value of h provides $\beta_1(h) = -\infty$, and if $\beta_j \leq \beta_{j+1}(h)$ occurs for some integer j in $[1, n-1]$, then β_i can be reduced to $\beta_i(h)$ for $i = j+1, j+2, \ldots, n$. Further, the value of h^* is the least value of the function in the first line of expression (2.5). The algorithm makes a one-sided splitting of this kind whenever it can do so. It follows from symmetry that, when t_p^* is set to $\alpha_p(h_1)$, as suggested before Lemma 5, then it is usually suitable to set $t_q^* = \beta_q(h_3)$ too, where h_3 is the least h satisfying $h \geq h^*$ for which the numbers $\beta_i(h)$, $i = p, p+1, \ldots, q$, have been generated. Further, after making the choices $t_p^* = \alpha_p(h_1)$ and $t_q^* = \beta_q(h_3)$, the remaining components of $\underline{t}$ are derived from up to three independent problems whose objective functions have the usual form. Specifically, if $p \geq 2$ there is a problem for the first $p-1$ components of $\underline{t}$, if $p \leq q-2$ there is a problem for the components t_i, $i = p+1, p+2, \ldots, q-1$, and if $q \leq n-1$ there is a problem for the last $n-q$ components of $\underline{t}$. Some of these calculations can be assisted by information from any previous one-sided splittings that have been made, which will be explained in the next section. Otherwise, each of these problems is treated as a new calculation, except that all bounds of the form $\alpha_i \leq t_i^* \leq \beta_i$ are retained. They are valid because α_i is increased to $\alpha_i(h)$ or β_i is decreased to $\beta_i(h)$ only when it is known that the current h is an upper bound on the final value of $\|\underline{x}_i - \underline{s}(t_i)\|_2$. The next section also includes a procedure that keeps track of the splittings and subproblems that occur.

If the solution of the original calculation has to satisfy $\|\underline{x}_1 - \underline{s}(t_1^*)\|_2 = h^*$, then, unfortunately, the conditions of Lemma 4 fail for every $h < h^*$. It is also possible that $\|\underline{x}_n - \underline{s}(t_n^*)\|_2 = h^*$ has to hold too, which would rule out the splitting that is mentioned at the beginning of the previous paragraph. Therefore we include yet another one-sided splitting technique in the algorithm of Section 3, which we introduce by considering the following simple example. Let the current lower and upper bounds on the parameters satisfy $\alpha_{j+1} < \beta_j \leq \alpha_{j+2} < \beta_{j+1}$ for an integer j in $[1, n-2]$, and let some investigations with a trial value of h provide $h < h^*$ and a value of t_{j+1} in $[\beta_j, \alpha_{j+2}]$ that satisfies $\|\underline{x}_{j+1} - \underline{s}(t_{j+1})\|_2 \leq h$. This choice of t_{j+1} is always permissible, because the parameters satisfy $t_1 \leq \cdots \leq t_j \leq \beta_j$ and $\alpha_{j+2} \leq t_{j+2} \leq \cdots \leq t_n$. Thus h^* is independent of t_{j+1}, although the conditions $\alpha_{j+1} < \beta_j$ and $\alpha_{j+2} < \beta_{j+1}$ do not allow t_{j+1} to be isolated by Lemma 3 splittings. Further, when seeking h^*, the adjustment of the parameters t_i, $i = 1, 2, \ldots, j$, can be separated from the adjustment of the parameters t_i, $i = j+2, j+3, \ldots, n$.

Our use of a generalization of these remarks requires the calculation of some numbers $\hat{\alpha}_i(h)$, $i=1,2,\ldots,n$, that are a variation on $\alpha_i(h)$, $i=1,2,\ldots,n$. Specifically, for any $h\geq 0$, we set $\hat{\alpha}_0(h)=\beta_0=0$. Then, for $i=1,2,\ldots,n$, we let $\hat{\alpha}_i(h)$ be the least value of t that satisfies the conditions

$$\|\underline{x}_i-\underline{s}(t)\|_2 \leq h \qquad \text{and} \qquad \min[\hat{\alpha}_{i-1}(h),\beta_{i-1}] \leq t \leq \beta_i, \tag{2.14}$$

except that we define $\hat{\alpha}_i(h)=+\infty$ if these inequalities cannot be achieved. Thus $\hat{\alpha}_i(h)$ is the same as $\alpha_i(h)$ for every i if $h\geq h^*$, because $\hat{\alpha}_{i-1}(h)\leq\beta_{i-1}$ always occurs in this case. On the other hand, if $h<h^*$, then the presence of β_{i-1} in expression (2.14) may allow $\hat{\alpha}_i(h)$ to be finite when $\hat{\alpha}_{i-1}(h)=+\infty$. The new splitting method will be derived from the following lemma.

Lemma 6 Let α_i and β_i be the current lower and upper bounds on t_i^* for each integer i in $[1,n]$, and let α_{n+1} be T. Further, let the numbers $\alpha_i(h)$ and $\hat{\alpha}_i(h)$, $i=1,2,\ldots,n$, be calculated with $h<h^*$, which implies $\alpha_n(h)=+\infty$. If k is an integer in $[1,n-1]$ with the properties $\alpha_k(h)=+\infty$ and $\hat{\alpha}_{k+1}(h)\leq\alpha_{k+2}$, then we form a vector $\underline{\tau}^*\in\mathcal{R}^n$. Specifically, the first j components of $\underline{\tau}^*$ minimize the first half of expression (2.5) with respect to the lexicographic ordering, where j is the greatest integer in $[1,k]$ such that $\hat{\alpha}_j(h)=+\infty$. We set $\tau_i^*=\hat{\alpha}_i(h)$, $i=j+1,j+2,\ldots,k+1$. Further, in the case $k+2\leq n$, we let the last $n-k-1$ components of $\underline{\tau}^*$ minimize the function

$$\max\{\|\underline{x}_i-\underline{s}(t_i)\|_2 : i=k+2,k+3,\ldots,n;\ \alpha_{k+2}\leq t_{k+2}\leq\cdots\leq t_n\leq T\}, \tag{2.15}$$

with respect to the lexicographic ordering. Then $\underline{\tau}^*$ is a solution of the h-problem that is introduced before the statement of Lemma 4.

Proof Let $\underline{t}^*$ be any solution of the h-problem. We are going to show that $\underline{t}^*$ remains a solution if t_i^* is replaced by $\tau_i^*=\hat{\alpha}_i(h)$ for $i=j+1,j+2,\ldots,k+1$. The equation $\beta_j=\min[\hat{\alpha}_j(h),\beta_j]$ is satisfied due to the choice of j, the condition $\hat{\alpha}_{k+1}(h)\leq\alpha_{k+2}$ is given, and all of the numbers $\hat{\alpha}_i(h)$, $i=j+1,j+2,\ldots,k+1$, are finite. Therefore the definition of $\hat{\alpha}_i(h)$ and the choice $\tau_i^*=\hat{\alpha}_i(h)$ for each of these values of i provide the bounds

$$\beta_j \leq \tau_{j+1}^* \leq \tau_{j+2}^* \leq \cdots \leq \tau_{k+1}^* \leq \alpha_{k+2}. \tag{2.16}$$

It follows that the replacement preserves the conditions (1.1) on the components of $\underline{t}^*$. Furthermore, the first part of expression (2.14) gives $\|\underline{x}_i-\underline{s}(\tau_i^*)\|_2\leq h$, $i=j+1,j+2,\ldots,k+1$, so the replacement also preserves the required lexicographic optimality of $\underline{t}^*$. Therefore we restrict attention to vectors $\underline{t}^*$ that satisfy $t_i^*=\tau_i^*$, $i=j+1,j+2,\ldots,k+1$.

Now τ_j^* has the property $\tau_j^*\leq\beta_j$ and the first of the inequalities (2.16) states $\beta_j\leq\tau_{j+1}^*$. Therefore the first part of the proof of Lemma 3 is applicable. Thus optimality is retained if we also overwrite the first j components of $\underline{t}^*$ by the first j components of $\underline{\tau}^*$. Finally, if $k+2 \le n$, then we deduce from the bounds $\tau_{k+1}^* \le \alpha_{k+2} \le \tau_{k+2}^*$ and the choice of τ_i^*, $i = k+2, k+3, \ldots, n$, that the last $n-k-1$ components of $\underline{t}^*$ can be treated similarly. Therefore $\underline{\tau}^*$ is a solution of the h-problem as required. $\square$

Our algorithm makes a one-sided splitting whenever the conditions of Lemma 6 are satisfied. Indeed, it switches to the optimization of the first function of expression (2.5), because it is known that the least value of this function is greater than h. Thus the algorithm takes advantage of the following corollary of the lemma. If t_i^*, $i \in [1, j]$, is an optimal value of t_i in the new calculation that satisfies $\|\underline{x} - \underline{s}(t_i^*)\|_2 > h$, then t_i^* is also an optimal value of t_i in the original calculation. A difference between this splitting and the previous ones is that now the optimal value of the new objective function may be less than h^*, because h^* may be determined by the function (2.15). Another difference is that the conditions of Lemma 6 do not allow any of the bounds $\alpha_i \le t_i^* \le \beta_i$, $i = 1, 2, \ldots, n$, to be revised. Of course the algorithm tries to reduce the number of variables in the new problem by applying further splittings. Moreover, there is a useful analogue of Lemma 6 when $\beta_1(h) = -\infty$ occurs, which follows from symmetry, and which is also applied by the algorithm whenever possible.

Further details of the splitting techniques are given in the next section. It may be very helpful to future work that the analysis so far applies not only to data points and curves in $\mathcal{R}^2$, but also to data points and curves in higher dimensions.

3. Some details of the algorithm

It is helpful if the first trial value of h in the main calculation, h_0 say, satisfies $h_0 \ge h^*$, but is not much larger than h^*, in order that the numbers $\alpha_i(h_0)$ and $\beta_i(h_0)$, $i = 1, 2, \ldots, n$, are useful lower and upper bounds on t_i^*, $i = 1, 2, \ldots, n$. Therefore h_0 is generated by a preliminary calculation that requires $\mathcal{O}(n)$ operations when n is large, but that is relatively inexpensive, because no searches are made along the continuous curve $\mathcal{S}$. Instead, $\mathcal{S}$ is replaced by the discrete point set $\{\underline{s}_j : j = 1, 2, \ldots, m\}$ for the moment, where $\underline{s}_1 = \underline{s}(0)$ and $\underline{s}_m = \underline{s}(T)$ are the initial and final points of $\mathcal{S}$, and where $\underline{s}_j$, $j = 2, 3, \ldots, m-1$, are the internal joins of the pieces of $\mathcal{S}$, the joins being in sequence, and each piece being a straight line segment or a circular arc. The following crude form of the main calculation provides h_0.

If h is a trial value of h_0, we ask whether there exist integers $1 \le j_1(h) \le j_2(h) \le \cdots \le j_n(h) \le m$, that satisfy the conditions

$$\|\underline{x}_i - \underline{s}_{j_i(h)}\|_2 \le h, \qquad i = 1, 2, \ldots, n. \tag{3.1}$$

In other words, we ask whether the data points can be mapped onto the discrete form of $\mathcal{S}$, so that the ordering of the data is preserved, and so that the length of each move is at most h. An affirmative answer implies $h \ge h^*$, because the points

$\underline{s}_{j_i(h)}$, $i=1,2,\ldots,n$, are in order on the original curve $\mathcal{S}$. Therefore we let h_0 be the least trial h that gives an affirmative answer. It is straightforward to seek the integers $j_i(h)$, $i=1,2,\ldots,n$, in sequence, by letting each one be as small as possible. We begin these trials by picking the value

$$h = \max\{\|\underline{s}_j - \underline{s}_{j+1}\|_2 : j=1,2,\ldots,m-1\} = h_{00}, \tag{3.2}$$

say, and if necessary we overwrite h by $2h+h_{00}$ recursively, until it becomes an upper bound on h_0. The greatest available lower bound on h_0 is also noted, even if it is zero. Thus, when the upper bound is found, the difference between the two bounds is $2^{k-1}h_{00}$, where k is the number of values of h that have been tried. Now the discrete approximation of $\mathcal{S}$ makes it appropriate to refine the difference between the bounds by bisection until it becomes $h_{00}/2$. Therefore k bisections are made, and we choose h_0 to be the final upper bound that occurs.

The other preliminary work is as follows. Let η be the prescribed tolerance on the width of the final bracket on h^* that is mentioned after the proof of Lemma 4. The algorithm ensures that η is at least h_0 times a pessimistic estimate of the relative accuracy of the computer arithmetic. Further, the initial values

$$\alpha_1=0, \quad \beta_n=T \quad \text{and} \quad \gamma_i=h_0, \qquad i=1,2,\ldots,n, \tag{3.3}$$

are chosen, in order that α_1 and β_n are lower and upper bounds on t_1^* and t_n^*, and where γ_i is reserved for an upper bound on $\|\underline{x}_i - \underline{s}(t_i^*)\|_2$, $i=1,2,\ldots,n$, except that γ_i will be altered to $\gamma_i=-1$ when t_i is given its final value t_i^*. The numbers $p=1$ and $q=n$ are also set, because the objective function of the main part of the calculation has the form

$$\max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 : i=p,p+1,\ldots,q;\ \alpha_p \le t_p \le t_{p+1} \le \cdots \le t_q \le \beta_q\}. \tag{3.4}$$

We retain the notation h^* for the least value of this function, which is unknown. Further, h^- and h^+ are lower and upper bounds on h^* that may be refined by bisection as suggested in Section 2. The algorithm picks the initial values $h^-=0$ and $h^+=h_0$.

We are now ready to address the recursive part of the calculation. The first task is the preliminary work for the minimization of the function (3.4), given p, q, h^- and h^+. If p equals q, there is an immediate branch to the part of the algorithm that treats this case analytically. Otherwise, the possibility of obtaining an improvement to h^+ from the bound

$$\begin{aligned} h^* \ \le\ \min\,[\,&\max\{\|\underline{x}_i - \underline{s}(\alpha_p)\|_2 : i=p,p+1,\ldots,q\}, \\ &\max\{\|\underline{x}_i - \underline{s}(\beta_q)\|_2 : i=p,p+1,\ldots,q\}\,] \ =\ \rho_0, \end{aligned} \tag{3.5}$$

say, is considered, the bound being valid because the constraints of expression (3.4) allow the choices $t_p = t_{p+1} = \cdots = t_q = \alpha_p$ and $t_p = t_{p+1} = \cdots = t_q = \beta_q$. A useful trick is employed in the case $h^+ \geq \rho_0$, which is to reduce h^+ to the value $\max[\rho_0 - \eta, 0]$. Then $\alpha_i(h^+)$ and $\beta_i(h^+)$, $i = p, p+1, \ldots, q$, are calculated by the methods of Lemma 1 and Corollary 2, respectively. We find $\alpha_q(h^+) = +\infty$ and $\beta_p(h^+) = -\infty$ if the trick provides $h^+ < h^*$, which is very welcome, because one or more of the final values t_i^*, $i = p, p+1, \ldots, q$, can be assigned. Specifically, if $h^+ < h^*$ and if the first maximum value in expression (3.5) does not exceed the second one, then the algorithm sets $t_i^* = \alpha_p$, $i = p, p+1, \ldots, j$, where j is the least integer in $[p, q]$ that satisfies $\alpha_j(h^+) > \alpha_p$. Similarly, if $h^+ < h^*$ and if the second maximum value in expression (3.5) is less than the first one, then the algorithm sets $t_i^* = \beta_q$, $i = k, k+1, \ldots, q$, where k is the greatest integer in $[p, q]$ such that $\beta_k(h^+) < \beta_q$. These choices of t_i^* can be justified when $t_i^* = \alpha_p$, for instance, by considering the function

$$\max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 : i = j, j+1, \ldots, q;\ \alpha_p \leq t_j \leq t_{j+1} \leq \cdots \leq t_q \leq \beta_q\}. \tag{3.6}$$

Its least value is at most h^* but is greater than h^+, because the choice of j implies that its numbers $\alpha_i(h^+)$, $i = j, j+1, \ldots, q$, are the same as before, including $\alpha_q(h^+) = +\infty$. Thus the suitability of $t_j^* = \alpha_p$ follows from Lemma 5, which implies $\alpha_p \leq t_i^* \leq \alpha_p$ for every integer i in $[p, j]$. After employing these advantages of $h^+ < h^*$, there is a branch to the part of the algorithm that picks the next values of p and q. Alternatively, in the usual case $h^+ \geq h^*$, the bounds $\alpha_i = \alpha_i(h^+)$ and $\beta_i = \beta_i(h^+)$, $i = p, p+1, \ldots, q$, are assigned, and then the main recursive procedure is begun.

This procedure applies the methods of Lemma 1 and Corollary 2 for several values of h. Thus it updates the bounds h^- and h^+ until the tolerance condition

$$h^+ - h^- \leq \eta \tag{3.7}$$

holds, or until there is a switch to an analytical calculation of h^*. The efficiency of the algorithm depends crucially on the splittings that are studied in Section 2, but those techniques will be addressed later. Indeed, this paragraph presents some other techniques by describing the operations of the algorithm when there are no splittings and when condition (3.7) is achieved eventually. If it is found that the current h satisfies $h < h^*$, then h^- is always replaced by $\max[h, h^-]$. Further, h^+ is replaced by $\min[h, h^+]$ whenever $h \geq h^*$ is revealed. Therefore only the value of each new h and the choice between the methods of Lemma 1 and Corollary 2 remain to be specified. The first h is set to $\frac{7}{8}h^+$ or to $\frac{1}{2}(h^- + h^+)$ in the case $h^- = 0$ or $h^- > 0$, respectively, the factor of 7/8 being helpful, because $h^- = 0$ is a default value and $h^* \geq \frac{7}{8}h^+$ occurs frequently. Each later choice of h is either $\frac{1}{2}(h^- + h^+)$ or h^+, the smaller value being preferred more often. Thus inequality (3.7) is attained eventually. The reason for letting the new h be h^+ occasionally is due to the importance of good values of α_i and β_i, $i = p, p+1, \ldots, q$, to successful splittings. For example, if $h^* = 2 - 10^{-10}$, and if $h^- = 1$ and $h^+ = 3$

occur initially, then h^+ becomes $h^+=2$, and it stays there until a trial h satisfies $h^+-2^{-10} \le h < h^+$. Therefore, if every new h is $h=\frac{1}{2}(h^-+h^+)$, then $\alpha_i=\alpha_i(2)$ and $\beta_i=\beta_i(3)$, $i=p,p+1,\ldots,q$, are retained until the difference h^+-h^- is reduced to $\max[\eta, 2\times 10^{-10}]$. Thus the β_i's can be useless for most of the calculation. The algorithm avoids such inefficiencies in the following way. When the values

$$\alpha_i=\alpha_i(h^+) \quad \text{and} \quad \beta_i=\beta_i(h^+), \qquad i=p,p+1,\ldots,q, \tag{3.8}$$

have been set for the current h^+, then the new h is always $\frac{1}{2}(h^-+h^+)$. On the other hand, when either the α_i's or the β_i's do not satisfy expression (3.8), we replace the formula $h=\frac{1}{2}(h^-+h^+)$ by $h=h^+$ if and only if at least three values of h have been tried and the last three consecutive ones have the property $h<h^*$. The choice $h=h^+$ leads to the completion of the equations (3.8), because of the following switching between the methods of Lemma 1 and Corollary 2. First the method of Lemma 1 is applied until $h \ge h^*$ occurs. Then Corollary 2 is preferred until the next occurrence of $h \ge h^*$. This alternation continues throughout the procedure of this paragraph. Therefore, letting ℓ be the integer such that the condition $h \ge h^*$ has been satisfied ℓ times already, the algorithm applies Lemma 1 instead of Corollary 2 to the next value of h if and only if ℓ is even.

The Fortran program that calculated the numerical results of Section 4 includes several subroutines, but we use the term "subroutine" for the one that applies either Lemma 1 or Corollary 2, the choice between these alternatives being controlled by an argument that is set by the calling program. This paragraph describes the output from the subroutine only in the case of Lemma 1, because the Corollary 2 output can be deduced from symmetry. An integer variable, namely `INFO`, is set to -1, 0 or 1. The value `INFO` $=-1$ indicates that the trial value of h satisfies $h<h^*$, and that none of the splittings of Section 2 occurs. The value `INFO` $=1$ indicates $h \ge h^*$, and that α_i and γ_i have been updated to $\alpha_i(h)$ and h, respectively, for $i=p,p+1,\ldots,q$. In this case the possibility of a splitting has to be investigated by the calling program. Therefore the remaining value `INFO` $=0$ indicates $h<h^*$, and that at least one Lemma 4 or Lemma 6 splitting is available. The variables that are allowed to be deleted from the current calculation by a splitting can be found by the calling program, because, if i is any integer in $[p,q]$ such that t_i can be dropped, then the subroutine gives $\gamma_i=h$, but the other values of γ_i remain greater than h. Furthermore, if a Lemma 4 splitting is possible, then the subroutine makes the changes to the lower bounds that are stated in Lemma 4, but the other values of α_i are not altered.

When the subroutine provides `INFO` $=1$ during the main recursive procedure, a search is made for a Lemma 3 splitting. We know that the main procedure will continue if there is no splitting, so we suppose that there exists an integer j in $[p,q-1]$ that satisfies $\beta_j \le \alpha_{j+1}$. Then h and q are reduced temporarily to h^- and j, respectively, and the subroutine is called again. Thus we determine whether the least value of the function

$$\max\{\|\underline{x}_i - \underline{s}(t_i)\|_2 : i = p, p+1, \ldots, j;\ \alpha_p \le t_p \le t_{p+1} \le \cdots \le t_j \le \beta_j\} \tag{3.9}$$

is at most h^-. This is the case if $\texttt{INFO}=1$ occurs again, and then the subroutine provides $\alpha_i = \alpha_i(h^-)$ and $\gamma_i = h^-$, $i = p, p+1, \ldots, j$, automatically, which will be useful later, because the algorithm makes the splitting that removes t_i, $i = p, p+1, \ldots, j$, from the current calculation. This is done by restoring h and q to their previous values and by increasing p to $j+1$. Now Lemma 3 shows that the value of h^* for the new calculation is the same as before, so the current h^- and h^+ still provide the bounds $h^- < h^* \le h^+$. Therefore, except for a branch to the part of the algorithm that has been noted already if $p = q$, one can go back to the beginning of this paragraph to find out what happens next, forgetting about the increase in p. Alternatively, if the least value of the function (3.9) is greater than h^-, then the current bounds $h^- < h^* \le h^+$ apply not only to the objective function (3.4) but also to expression (3.9), although the new h^* may be less than before. Therefore the temporary value $q = j$ is accepted and h is restored to its previous value, in order that the minimization of the function (3.9) becomes the current calculation. Again a $p = q$ branch is possible. Otherwise, the work of the main procedure is resumed, forgetting that q has decreased. The description of Lemma 3 splittings is complete.

The one-sided splitting when $\texttt{INFO}=0$ is easy to describe, because the variable t_i can be removed by a splitting if and only if the integer $i \in [p, q]$ has the property $\gamma_i = h$. Therefore p is increased if necessary to the least integer k in $[p, q]$ that satisfies $\gamma_k \ne h$. Then q is reduced if necessary to the greatest integer ℓ in $[p, q]$ that is allowed by the conditions

$$\gamma_i > h, \qquad i = k, k+1, \ldots, \ell. \tag{3.10}$$

The minimum value of the new objective function (3.4) is greater than h, because otherwise all the numbers γ_i, $i = k, k+1, \ldots, \ell$, would have been set to h. Thus the bounds $h^- < h^* \le h^+$ are inherited by the new calculation, even if h^* is smaller than before, and the next h^- is the current value of h. Then, apart from the usual treatment of the case $p = q$, the algorithm returns to the main procedure, regardless of any changes to p and q.

We complete the description of the operations that may be relevant to the initial choice of a component of $\underline{t}^* \in \mathcal{R}^n$ by addressing the analytic calculations that are mentioned in Section 2. When $p = q$, the algorithm determines the final value of t_p, namely t_p^*, by minimizing the distance $\|\underline{x}_p - \underline{s}(t_p)\|_2$, $\alpha_p \le t_p \le \beta_p$. The other analytic calculation that is employed instead of bisection is the subject of this paragraph. Imagine that $\mathcal{S}$ is fairly straight and that one is going along it, looking for $\underline{x}_1$ initially, but that one sees $\underline{x}_2$ nearby before $\underline{x}_1$ comes into sight, although $\underline{x}_1$ is also close to the curve. Then, if there is no interference from the points $\underline{x}_i$, $i=3,4,\ldots,n$, the final values of t_1 and t_2 have the properties

$$t_1^* = t_2^* \qquad \text{and} \qquad \|\underline{x}_1 - \underline{s}(t_1^*)\|_2 = \|\underline{x}_2 - \underline{s}(t_2^*)\|_2. \tag{3.11}$$

Now $\underline{s}(\alpha_1(h))$ comes before $\underline{s}(t_1^*)$ on $\mathcal{S}$ if and only if $\alpha_1(h)$ is a strict lower bound on t_1^*, and then expression (3.11) and the scenario provide $\|\underline{x}_2 - \underline{s}(\alpha_1(h))\|_2 < \|\underline{x}_1 - \underline{s}(\alpha_1(h))\|_2$. Further, this inequality and the definition of $\alpha_2(h)$ imply $\alpha_2(h) = \alpha_1(h)$, and one can argue similarly that $\beta_1(h) = \beta_2(h)$ is likely to hold. Thus, if $\alpha_1 = \alpha_2$ and $\beta_1 = \beta_2$ occur during the calculation, they suggest strongly that the equations (3.11) are going to be satisfied. This kind of situation happens frequently, perhaps 100 times in a problem with $n = 1000$. Therefore the algorithm includes the following device. If both the conditions

$$\alpha_p = \alpha_q \quad \text{and} \quad \beta_p = \beta_q \tag{3.12}$$

are found during the minimization of the function (3.4), where $p < q$, then the algorithm tests the possibility of the given scenario by seeking answers to some questions analytically, except that the sequence of questions is abandoned if an answer is negative. Firstly, we ask if the inequalities

$$\|\underline{x}_p - \underline{s}(\alpha_p)\|_2 \geq \|\underline{x}_q - \underline{s}(\alpha_q)\|_2 \quad \text{and} \quad \|\underline{x}_p - \underline{s}(\beta_p)\|_2 \leq \|\underline{x}_q - \underline{s}(\beta_q)\|_2 \tag{3.13}$$

hold, at least one of them being strict. Secondly, we ask if the distance $\|\underline{x}_p - \underline{s}(t_p)\|_2$, $\alpha_p \leq t_p \leq \beta_p$, decreases strictly monotonically. Thirdly, we ask if $\|\underline{x}_q - \underline{s}(t_q)\|_2$, $\alpha_q \leq t_q \leq \beta_q$, increases strictly monotonically. When all the answers so far are favourable, there is a unique number $t^* \in [\alpha_p, \beta_p]$ that satisfies $\|\underline{x}_p - \underline{s}(t^*)\|_2 = \|\underline{x}_q - \underline{s}(t^*)\|_2 = d^*$, say, and it is calculated analytically. Finally, if there are integers between p and q, we ask whether all the distances $\|\underline{x}_i - \underline{s}(t^*)\|_2$, $p+1 \leq i \leq q-1$, are bounded above by d^*. Affirmative answers imply that the choices $t_i = t^*$, $i = p, p+1, \ldots, q$, give the objective function (3.4) the value d^*. Further, the strict monotonicity requirements of the second and third questions, and the constraints $\alpha_p \leq t_p \leq t_{p+1} \leq \cdots \leq t_q \leq \beta_q$, imply that $t_i = t^*$ is the only optimal choice of t_i for each integer i in $[p, q]$. Thus the final values of all the variables of the current optimization calculation are often assigned before the tolerance condition (3.7) is achieved. The successful completion of either of the analytic techniques of this paragraph is always followed by the operations of the algorithm that pick the next values of p and q.

Access to the analytic method that has just been described is a branch from the main recursive procedure, immediately before the choice of the next value of h. The branch is made if the equations (3.12) hold, and if the analytic technique has not been tried already for the current values of p, q, α_p and β_q. A convenient way of ensuring the last condition is to set $\rho_1 = 2T$ when expression (3.5) is considered, and to reduce ρ_1 to $\beta_q - \alpha_p$ whenever the analytic technique is unsuccessful, as then it is sufficient to add the constraint $\beta_q - \alpha_p < \rho_1$ to the equations (3.12). An unfavourable answer to any of the questions of the previous paragraph is followed by a branch back to the recursive procedure, in order to refine the bracket on h^* if inequality (3.7) is not yet satisfied.

If the recursive procedure finds that condition (3.7) is achieved, then, instead of picking a new h, it assigns the final values of t_p and t_q, the indices p and q being different, because of the action that is taken analytically in the case $p=q$. These final values are $t_p^*=\alpha_p(h^+)$ and $t_q^*=\beta_q(h^+)$, as recommended in Section 2. Now the most recent call of the subroutine with $h \geq h^*$ employed $h=h^+$. Further, if that call applied the method of Lemma 1, then α_p is the required value of $\alpha_p(h^+)$, and, if Corollary 2 was applied instead, then β_q is the required value of $\beta_q(h^+)$. We also recall that occasionally both $\alpha_p(h^+)$ and $\beta_q(h^+)$ may be available, because the subroutine may have been invoked twice with $h=h^+$. Otherwise, the call that completes the equations (3.8) is made, and it is repeated with a slightly larger value of h if rounding errors cause an `INFO` $=-1$ or `INFO` $=0$ return. Then, after setting $t_p^*=\alpha_p$ and $t_q^*=\beta_q$, the flow of the algorithm goes to the operations of the next paragraph.

New integers p and q are chosen whenever the final values of one or more of the variables t_i, $i=1,2,\ldots,n$, are assigned. If t_j^* is any of the new final values, then the bounds are updated by applying the formula $\alpha_j=\beta_j=t_j^*$, but all other revisions of bounds are made later. Further, γ_j is set to -1 as mentioned soon after equation (3.3), and we assume $\gamma_0=\gamma_{n+1}=-1$. The techniques that pick the new p and q depend on the integer, r say, that is the least j for which t_j^* has just been determined, and on a property of the algorithm that we call the "staircase" condition. This condition is that, for every p throughout the calculation, the sequence γ_i, $i=1,2,\ldots,p$, increases monotonically. It is trivial initially due to $p=1$. The following remarks prove that the condition is achieved by the method of calculation that has been described so far. Any changes to γ_i are made by the subroutine, each new γ_i being the current h, and each h being bounded below by h^-. Therefore it is sufficient if $h^- \geq \gamma_{p-1}$ holds at all times, and if the splittings preserve the staircase structure. Neither the recursive procedure nor a splitting reduces h^-, so it remains to consider splittings that increase p, from $\check{p}$ to $\hat{p}$ say. A Lemma 3 splitting is acceptable, because it sets $\gamma_i=h^-$, $i=\check{p},\check{p}+1,\ldots,\hat{p}-1$, if it increases p. A one-sided splitting, however, provides $\gamma_i=h$, $i=\check{p},\check{p}+1,\ldots,\hat{p}-1$, for the current h, which is acceptable too, because this h is the next h^-. Therefore the staircase condition can be used in the choice of p and q that follows the selection of t_r^*.

First we address the case when not all of the final values t_i^*, $i=1,2,\ldots,r$, have been chosen. According to the staircase structure, it is characterized by $\gamma_{r-1}\geq 0$. Then a provisional choice of p and q is made that corresponds to the top step of the staircase just before t_r^*, so q and p are set to $r-1$ and the least positive integer that satisfies $\gamma_p=\gamma_q$, respectively. During the search for p, the bound β_p is reduced if necessary so that $\beta_p \leq \beta_{p+1}$ holds for each trial p. Next we look for

Lemma 3 splittings of the variables t_i, $i=p,p+1,\ldots,q$, and, if there is one, q is decreased to the least integer that is allowed by the inequalities

$$q \ge p \quad \text{and} \quad \beta_q \le \alpha_{q+1}. \tag{3.14}$$

These choices of p and q are the final ones if $\gamma_{p-1}=-1$, and then h^- is set to zero. Alternatively, in the case $\gamma_{p-1}\ge 0$, the algorithm finds out if the top step of the staircase can be lowered to the level of the previous step, by calling the Algorithm 1 version of the subroutine with $h=\gamma_{p-1}$, for the current p and q. An $\texttt{INFO}=1$ return is received if and only if the lowering is valid, and then the subroutine will have overwritten γ_i by γ_{p-1} for all integers i in $[p,q]$. The successful broadening of the top step demands a decrease in p, so there is a branch back to the operation of this paragraph that chooses the provisional p. The other two returns from the subroutine, however, indicate that γ_{p-1} is strictly less than the minimum value of the function (3.4). Therefore h^- is set to γ_{p-1}, and the chosen values of p and q are going to be from the current interval $[p,q]$. In the case of an $\texttt{INFO}=0$ return, the technique that gives expression (3.10) is applied, because the subroutine has reduced one or more of the numbers γ_i, $i=p,p+1,\ldots,q$, to γ_{p-1}. Specifically, letting k be the least integer in $[p,q]$ such that $\gamma_k>\gamma_{p-1}$ holds, and then letting ℓ be the greatest integer in $[p,q]$ that has the property

$$\gamma_i > \gamma_{p-1}, \qquad i=k,k+1,\ldots,\ell, \tag{3.15}$$

the indices p and q are reset to k and ℓ, respectively. Thus the $\texttt{INFO}=0$ and $\texttt{INFO}=-1$ cases become equivalent. Again there is a search for Lemma 3 splittings, so q is decreased if possible to the least integer that satisfies the inequalities (3.14), as before. Fortunately, $h^-=\gamma_{p-1}$ remains a strict lower bound on the function (3.4) if q is reduced, because any splitting that decreases q was not given as a Lemma 4 splitting by the recent call of the subroutine. The description of the choice of the new indices p and q when $\gamma_{r-1}\ge 0$ occurs is complete. Therefore, after setting $h^+=\gamma_p$, the flow of the algorithm goes to the beginning of the paragraph that includes expression (3.5), in order to begin the search for the least value of the new objective function (3.4).

In the case $\gamma_{r-1}=-1$, the algorithm increases r by 1 if the conditions $r\le n-1$ and $\gamma_{r+1}=-1$ are satisfied, which is done recursively. Thus the value $r=n$ is achieved if and only if all the final values of the parameters have been assigned, so the calculation is terminated in this case. Otherwise, the search for the next indices p and q gives priority to Lemma 3 splittings, because, if n is large, it is important to efficiency to avoid many values of $q-r$ that are of magnitude n. Therefore the first provisional value of q is the least integer in $[r+1,n]$ that satisfies one or both of the inequalities

$$\beta_q \le \alpha_{q+1} \quad \text{and} \quad \gamma_q > \gamma_{q+1}. \tag{3.16}$$

Further, during the search for q, the bound α_{q+1} is increased if necessary so that $\alpha_{q+1} \geq \alpha_q$ holds for each trial q. Then the numbers γ_i, $i=1,2,\ldots,q$, have the staircase structure, so again p is set provisionally to the least positive integer that has the property $\gamma_p=\gamma_q$. We employ the notation γ^*_{q+1} for the number γ_{q+1} or -1 in the case $\beta_q > \alpha_{q+1}$ or $\beta_q \leq \alpha_{q+1}$, respectively, in order that γ_{q+1} becomes irrelevant when it is known that t^*_q is independent of t^*_{q+1}. It follows that the required values of p and q have been found already if the conditions

$$\gamma_p = \gamma_{p+1} = \cdots = \gamma_q \geq 0 \quad \text{and} \quad \gamma_{p-1} = \gamma^*_{q+1} = -1 \tag{3.17}$$

are achieved, the choice of q having excluded Lemma 3 splittings of the variables t_i, $i=p,p+1,\ldots,q$. Then the algorithm sets $h^-=0$ and $h^+=\gamma_p$, and branches to the instructions of the paragraph that includes expression (3.5). Otherwise, we investigate if the top step of the staircase can be lowered by calling the Lemma 1 version of the subroutine for the current p and q, with h set to $\max[\gamma_{p-1},\gamma^*_{q+1}]$. Again the answer is negative if an $\texttt{INFO}=0$ or $\texttt{INFO}=-1$ return occurs, which is analogous to the corresponding situation in the previous paragraph. Therefore h^- is set to the current h, and there is a branch to the operations of the previous paragraph that select the final p and q from the current interval $[p,q]$ for these values of INFO. On the other hand, after an $\texttt{INFO}=1$ return, the subroutine has reduced γ_i to the current h for every integer i in $[p,q]$, which allows the step under consideration to be widened, but it may not be the top step any more if h is γ^*_{q+1}. Therefore we pick new provisional values of p and q. Because the subroutine has not altered α_i, β_i and γ_i, $i=1,2,\ldots,p-1$, but it may have changed α_p, the new q satisfies $q \geq \max[r+1,p-1]$. Specifically, it is the least q subject to this lower bound that achieves one or both of the conditions (3.16). Then p is revised and the subsequent operations are reached by going back to the provisional choice of p that is made earlier in this paragraph. The description of the updating of p and q is complete. We note that the given recursive procedure cannot cycle indefinitely. Indeed, each cycle with an $\texttt{INFO}=1$ return makes changes to the bounds γ_i, $i=1,2,\ldots,n$, that provide a strict reduction in the cardinality of the set $\{i \in [r+1,r^*-1] : \gamma_i \neq \gamma_{i+1}\}$, where r^* is the least integer in $[r+2,n]$ that has the property $\beta_{r^*} \leq \alpha_{r^*+1}$, the value of α_{n+1} being T. We note also that the staircase condition is preserved by the techniques that select the new p and q.

The final subject of this section is a few brief remarks on the representation of $\mathcal{S}$ by the algorithm and on the analytic calculations that have been mentioned. Therefore we recall the notation $\underline{s}_1$ and $\underline{s}_m$ for the initial and final points of $\mathcal{S}$ and $\underline{s}_j$, $j=2,3,\ldots,m$–1, for the internal joins of the pieces of $\mathcal{S}$. In every application of the algorithm so far, m is an even integer, and the part of $\mathcal{S}$ between $\underline{s}_j$ and $\underline{s}_{j+1}$ is a straight line segment or a circular arc when j is odd or even, respectively. Thus $\mathcal{S}$ is defined uniquely by the positions of $\underline{s}_j$, $j=1,2,\ldots,m$. Further, these positions have to satisfy the constraint that, if j is any even integer in $[2,m-2]$, then three lines are concurrent, namely the straight line extension from $\underline{s}_{j-1}$ through $\underline{s}_j$, the straight line extension from $\underline{s}_{j+2}$ through $\underline{s}_{j+1}$, and the perpendicular bisector of the line segment from $\underline{s}_j$ to $\underline{s}_{j+1}$. The constraint is necessary for the continuity of the direction of $\mathcal{S}$ between $\underline{x}_{j-1}$ and $\underline{x}_{j+2}$. The algorithm is provided with a sequence $\underline{s}_j$, $j=1,2,\ldots,m$, that satisfies these conditions.

Then the parametric form $\mathcal{S}=\{\underline{s}(t) : 0\leq t\leq T\}$ is defined by induction in the following way, beginning with $\underline{s}_1=\underline{s}(\theta_1)$, where θ_1 is zero. For $j=1,2,\ldots,m-1$, the parameter θ_{j+1} of the identity $\underline{s}_{j+1}=\underline{s}(\theta_{j+1})$ is given the value

$$\theta_{j+1} = \begin{cases} \theta_j + \|\underline{s}_{j+1}-\underline{s}_j\|_2 \quad \text{or} \\ \theta_j + \text{ distance along arc from } \underline{s}_j \text{ to } \underline{s}_{j+1}, \end{cases} \tag{3.18}$$

the latter alternative being preferred when j is even and the arc from $\underline{s}_j$ to $\underline{s}_{j+1}$ turns through more than $\pi/2$ radians. This construction provides $T=\theta_m$. Further, when j is odd or when the second line of expression (3.18) applies, we let $\underline{s}(t)$, $\theta_j\leq t\leq\theta_{j+1}$, be the point on $\mathcal{S}$ whose arc-length distance from $\underline{s}_j$ is $t-\theta_j$. In the remaining case, the parameterization

$$\underline{s}(t) = \underline{s}_j + \frac{\lambda\,\{\|\underline{u}\|^3 + 4\,\|\underline{v}\|^2\,(2\lambda-\|\underline{u}\|)\}\,\underline{u} + 4\lambda\,\|\underline{u}\|^2\,(\|\underline{u}\|-\lambda)\,\underline{v}}{\|\underline{u}\|^4 + 4\,\|\underline{v}\|^2\,(2\lambda-\|\underline{u}\|)^2} \tag{3.19}$$

is employed for every t in $[\theta_j,\theta_{j+1}]$. Here λ and $\underline{u}$ are the differences $t-\theta_j$ and $\underline{s}_{j+1}-\underline{s}_j$, respectively, while $\underline{v}$ is the difference between the midpoint of the circular arc from $\underline{s}_j$ to $\underline{s}_{j+1}$ and the midpoint of the chord from $\underline{s}_j$ to $\underline{s}_{j+1}$. Thus $\underline{v}$ is orthogonal to $\underline{u}$. Further, it can be verified that $\underline{s}(t)$ takes the values $\underline{s}_j$, $\underline{s}_{j+1}$ and $\frac{1}{2}(\underline{s}_j+\underline{s}_{j+1})+\underline{v}$ in the cases $\lambda=0$, $\lambda=\|\underline{u}\|$ and $\lambda=\frac{1}{2}\,\|\underline{u}\|$. Moreover, if $\underline{v}$ is zero, then the parameterization (3.19) reduces to $\underline{s}_j+\lambda\,\underline{u}/\|\underline{u}\|$, which is a usual expression for a point on the straight line segment between $\underline{s}_j$ and $\underline{s}_{j+1}$. One can also deduce the properties $\|\underline{s}'(\theta_j)\|=\|\underline{s}'(\theta_{j+1})\|=1$. It follows from the continuity of the direction of $\mathcal{S}$ that the gradient of the representation $\{\underline{s}(t) : 0\leq t\leq T\}$ is also continuous, which is important to the procedure for mapping curves into curves that is mentioned in Section 1. When $\underline{s}_j$ is joined to $\underline{s}_{j+1}$ by a circular arc whose angle of rotation is more than $\pi/2$, however, then the algorithm applies the formula

$$\underline{s}(t) = \underline{s}_j + r\sin(\lambda/r)\,\underline{\hat{u}} + r\{1-\cos(\lambda/r)\}\,\underline{\hat{v}}, \qquad \theta_j\leq t\leq\theta_{j+1}, \tag{3.20}$$

where λ is $t-\theta_j$ as before, where r is the radius of the arc, and where $\underline{\hat{u}}$ and $\underline{\hat{v}}$ are vectors of unit length that are parallel and orthogonal to the direction of the incoming tangent, namely $\underline{s}_j-\underline{s}_{j-1}$. The large angle of rotation ensures that r is bounded. On the other hand, expression (3.19) is suitable when the arc is close to a straight line.

In all of these cases, one can discover analytically if a data point, $\underline{x}_i$ say, is within distance h of the j-th section of $\mathcal{S}$, by addressing the distances from $\underline{x}_i$ to the end-points of the section, and then by considering the stationary points of the function $\|\underline{x}_i - \underline{s}(t)\|_2$, $\theta_j \le t \le \theta_{j+1}$, if necessary. Thus there is no need to solve a quadratic equation, but a quadratic is solved if a point on $\mathcal{S}$ is required that is distance h from $\underline{x}_i$. Another analytic calculation occurs soon after the third question in the paragraph of equations (3.11)–(3.13). There the number t^* that is defined by $\|\underline{x}_p - \underline{s}(t^*)\|_2 = \|\underline{x}_q - \underline{s}(t^*)\|_2$ may be derived from the fact that $\underline{s}(t^*)$ is on the perpendicular bisector of the line segment between $\underline{x}_p$ and $\underline{x}_q$. Other details of these operations of the algorithm include some formulae that are convenient for computation and that avoid unnecessary loss of accuracy, but their descriptions are omitted because they would be rather long and tedious.

4. Numerical results

Each curve $\mathcal{S}$ of the numerical experiments of this section has the following form. The end-points and joins of the pieces of $\mathcal{S}$ are $\underline{s}_j = \underline{s}(\theta_j)$, $j = 1, 2, \ldots, m$, as before, the parameters θ_j being defined in the way that includes expression (3.18), and m being an even integer. Further, for convenience, we pick the points

$$\underline{s}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \text{and} \quad \underline{s}_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \tag{4.1}$$

and we satisfy the equations

$$\|\underline{s}_{j+1} - \underline{s}_j\|_2 = 1, \ \ j \,\text{odd}, \quad \text{and} \quad \|\underline{s}_{j+1} - \underline{s}_j\|_2 = \tfrac{1}{2}, \ \ j \,\text{even}. \tag{4.2}$$

We also let the pieces of $\mathcal{S}$ be straight line sections and circular arcs alternately, as mentioned already. It follows from the conditions (4.1) and (4.2) that $\mathcal{S}$ is defined uniquely by the angles of rotation of the arcs, ψ_k, $k = 1, 2, \ldots, \hat{m}$, say, where $\hat{m}$ is the integer $\frac{1}{2}(m-2)$. We begin with the case when $\hat{m}$ is 6 and each ψ_k is $\pi/3$ radians. Thus $\mathcal{S}$ is a regular hexagon with rounded corners, except that the line segment from $\underline{s}_{m-1}$ to $\underline{s}_m$ is a repeat of the line segment between the points (4.1).

Many features of the splitting techniques can be tested by placing all the data points $\underline{x}_i$, $i = 1, 2, \ldots, n$, on $\mathcal{S}$, so we make the initial choice

$$\underline{x}_i = \underline{s}(\,[i-1]\,T/\,[n-1]\,) + \underline{\varepsilon}_i, \qquad i = 1, 2, \ldots, n, \tag{4.3}$$

with $\underline{\varepsilon}_i = 0$. In general, however, each $\underline{\varepsilon}_i$ is generated randomly from the uniform distribution on the disc $\{\underline{x} : \|\underline{x}\|_2 \le \delta\}$ for some parameter δ. Specifically, three values of δ were employed, namely 0, Δ and 10Δ, where Δ is the average distance

$$\Delta = (n-1)^{-1} \sum_{i=1}^{n-1} \|\underline{x}_{i+1} - \underline{x}_i\|_2 \tag{4.4}$$

between the adjacent points of expression (4.3) when every $\underline{\varepsilon}_i$ is zero. Moreover, throughout the experiments the parameter η of the tolerance condition (3.7) was

	$\mathcal{S}$ has one cycle			$\mathcal{S}$ has $n/125$ cycles		
n	$\delta=0$	$\delta=\Delta$	$\delta=10\Delta$	$\delta=0$	$\delta=\Delta$	$\delta=10\Delta$
125	0.02	0.03	0.03	0.02	0.03	0.03
250	0.04	0.07	0.07	0.03	0.07	0.07
500	0.08	0.14	0.16	0.07	0.14	0.14
1000	0.17	0.29	0.34	0.13	0.30	0.30
2000	0.37	0.61	0.68	0.27	0.61	0.61
4000	0.76	1.25	1.42	0.53	1.23	1.25
8000	1.67	2.62	2.87	1.08	2.57	2.70

Table 1: Hexagonal $\mathcal{S}$ with nearby data points

set to 10^{-10}. The calculations were run in double precision on a Sparc 5/170 workstation. The CPU times in seconds for this test problem, using several values of n, are given in the "$\mathcal{S}$ has one cycle" columns of Table 1. Next, the choice $\hat{m}=6$ was replaced by $\hat{m}=6n/125$, but the angles $\psi_k=\pi/3$, $k=1,2,\ldots,\hat{m}$, and formulae (4.3) and (4.4) were retained. Thus the number of cycles of $\mathcal{S}$ becomes $n/125$ and there is a corresponding change to T. The CPU times of the new experiment are shown in the last three columns of Table 1. The performance of the algorithm is highly satisfactory, because the amount of work is nearly proportional to n. Further, we see that it is not disadvantageous to increase the length of $\mathcal{S}$.

Some less regular choices of $\mathcal{S}$ were made by generating each of the angles ψ_k, $k=1,2,\ldots,\hat{m}$, randomly from the uniform distribution on $[-\pi/3, 2\pi/3]$. Thus the average value of ψ_k becomes $\pi/6$, so the complete curve $\mathcal{S}$ turns through about 2π radians if we pick $\hat{m}=12$ and $m=26$. A curve of this form is displayed in Figure 1, with the points (4.3) when $n=30$ and $\delta=2\Delta$, each $\underline{x}_i$ being shown as a circle that contains the value of i. The positions of a few of these points were modified, however, in order to make the picture clearer. They were shifted onto $\mathcal{S}$ by the algorithm, the new positions being given as bullets in the figure. The straight line segments between $\underline{x}_i$ and $\underline{s}(t_i^*)$ are drawn too for every i. Thus we have an illustration of the moves of the data points that are made by the algorithm. The test problem of this paragraph was also employed in some numerical experiments that are analogous to the calculations of Table 1. Indeed, the same values of n and δ were chosen, and the "one cycle" and "$n/125$ cycles" cases were obtained by picking $\hat{m}=12$ and $\hat{m}=12n/125$, respectively. The results are not tabulated, however, because the times are only a little greater than before. For example, the new measurements that correspond to the $n=8000$ row of Table 1 are 1.85, 2.89, 3.22, 1.19, 2.93 and 3.05 seconds.

Unfortunately, the total work of the algorithm is usually much greater than

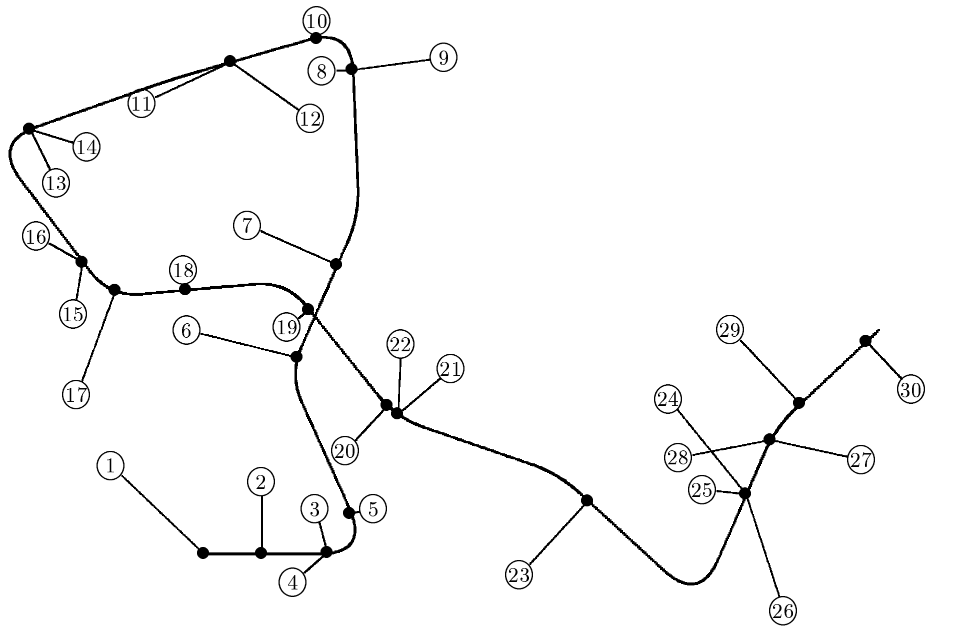

Figure 1: A random choice of the angles of $\mathcal{S}$

the results so far, if the points $\underline{x}_i$, $i=1,2,\ldots,n$, lie on a smooth curve that is different from $\mathcal{S}$, and if the distance from $\underline{x}_i$ to $\mathcal{S}$ is large in comparison with $\|\underline{x}_{i+1}-\underline{x}_i\|_2$ for most integers i in $[1,n-1]$. For example, we let $\mathcal{S}$ be the part of the x-axis between $(1,0)$ and $(n,0)$, and we let the coordinates of $\underline{x}_i$ be (i,i), $i=1,2,\ldots,n$. Then no Lemma 3 splittings can occur before a one-sided splitting, because the minimum value of the objective function (1.2) is $h^*=n$, and the inequality $\beta_j(h^*)\leq\alpha_{j+1}(h^*)$ takes the form

$$\min\left[n,\, j+\sqrt{n^2-j^2}\right] \leq \max\left[1,\, j+1-\sqrt{n^2-(j+1)^2}\right], \tag{4.5}$$

which fails for all integers j in $[1,n-1]$, the value of n^2-j^2 being greater than 1. On the other hand, some Lemma 4 splittings are possible, but they do not happen for $j\geq 2^{-1/2}n+1$, unless the h that provides $\alpha_n(h)=+\infty$ is close to h^*. Indeed, when the conditions of Lemma 4 hold for the first time for such a j, the current lower bound α_{j+1} satisfies the inequality

$$\alpha_{j+1} \leq \alpha_{j+1}(h^*) = \max\left[1,\, j+1-\sqrt{n^2-(j+1)^2}\right]. \tag{4.6}$$

Therefore, if h gives a splitting due to $\alpha_j(h)\leq\alpha_{j+1}$, then it has the property

$$\alpha_j(h) = \max\left[1,\, j-\sqrt{h^2-j^2}\right] \leq \max\left[1,\, j+1-\sqrt{n^2-(j+1)^2}\right]. \tag{4.7}$$

	$\mathcal{S}$ has one cycle			$\mathcal{S}$ has $n/125$ cycles		
n	$\delta=0$	$\delta=\Delta$	$\delta=10\Delta$	$\delta=0$	$\delta=\Delta$	$\delta=10\Delta$
125	0.14	0.08	0.04	0.14	0.08	0.04
250	0.57	0.25	0.11	0.31	0.19	0.09
500	2.46	0.82	0.30	0.67	0.38	0.19
1000	10.74	3.50	0.78	1.35	0.77	0.38
2000	47.47	14.46	3.08	2.76	1.58	0.90
4000	209.08	65.57	11.82	5.54	3.25	1.79
8000	908.41	268.41	52.04	11.16	6.60	3.93

Table 2: Hexagonal $\mathcal{S}$ with separate data points

It follows from $j \geq 2^{-1/2}n+1$ and $j \leq h < n$ that the term $(h^2-j^2)^{1/2}$ is at least $\{n^2-(j+1)^2\}^{1/2}-1$, so some straightforward algebra provides the bound

$$h^2 \geq n^2-2j-2\sqrt{n^2-(j+1)^2} \geq n^2 - 2\sqrt{2}\,n + 2, \tag{4.8}$$

the last inequality being elementary. Hence we require $h \geq n-2^{1/2}$ when n is large. These remarks suggest that, for some data, the efficiency of the algorithm may be very poor in comparison with the timings that are shown in Table 1.

We investigate this suggestion by changing the data points of the first test problem from expression (4.3) to the values

$$\underline{x}_i = \begin{pmatrix} \sin([i-1]/[n-1]) \\ 1-\cos([i-1]/[n-1]) \end{pmatrix} + \underline{\varepsilon}_i, \qquad i=1,2,\ldots,n, \tag{4.9}$$

but we do not alter $\mathcal{S}$. Thus, when every $\underline{\varepsilon}_i$ is zero, the new data points are equally spaced on a circle that is mainly well inside $\mathcal{S}$. Further, the definition (4.4) provides $\Delta = 2\sin(1/[2n-2])$, and then we pick δ and $\underline{\varepsilon}_i$, $i=1,2,\ldots,n$, as before. The times of the algorithm in this case are shown in the "$\mathcal{S}$ has one cycle" columns of Table 2. It seems that the amount of work has become of magnitude n^2 for each of the three choices of δ, and that the times are reduced greatly when the data points are perturbed by random errors, because useful splittings occur much earlier in the calculations. Finally, we employed the old $\mathcal{S}$ that has $n/125$ cycles, and we provided the data with the same number of cycles, by changing the denominators of equation (4.9) from $[n-1]$ to 125, which implies $\Delta = 2\sin(1/250)$. The CPU times that occurred are given in the last three columns of Table 2. We find that the approximate $\mathcal{O}(n)$ complexity has returned, which is probably due to a tendency in the algorithm to process the cycles sequentially.

The accuracy of the numerical experiments of this section was tested in the following way. We let $\underline{\hat{s}}(\hat{t}_i^*)$, $i=1,2,\ldots,n$, be the points on $\mathcal{S}$ that were computed by the algorithm. Further, both the direction of $\mathcal{S}$ and the order of the data $\underline{x}_i$, $i=1,2,\ldots,n$, were reversed, giving a new problem, which was also solved by the algorithm, the computed points on the new $\mathcal{S}$ being $\underline{\check{s}}(\check{t}_i^*)$, $i=1,2,\ldots,n$, say. Now, assuming that the lexicographic method provides uniqueness, the identities $\underline{\check{s}}(\check{t}_i^*) = \underline{\hat{s}}(\hat{t}_{n+1-i}^*)$, $i=1,2,\ldots,n$, would hold in exact arithmetic. Therefore the numbers

$$\zeta_i = \|\underline{\check{s}}(\check{t}_i^*) - \underline{\hat{s}}(\hat{t}_{n+1-i}^*)\|_2, \qquad i=1,2,\ldots,n, \tag{4.10}$$

were recorded, because they may show some serious consequences of computer rounding errors, and they may provide some useful information on the tolerance parameter η of condition (3.7). It is possible for ζ_i to be of magnitude $\eta^{1/2}$. Indeed, if $h^* = \|\underline{x}_i - \underline{s}(t_i^*)\|_2$ is the least distance from $\underline{x}_i$ to the curve $\mathcal{S}$, if h^* is positive, and if t_i^* is in the open interval $(0,T)$, then the difference $\underline{x}_i - \underline{s}(t_i^*)$ is orthogonal to the direction of $\mathcal{S}$ at $\underline{s}(t_i^*)$. Thus a bound of the form

$$\|\underline{x}_i - \underline{s}(t)\|_2 - h^* \le c\,|t - t_i^*|^2, \qquad 0 \le t \le T, \tag{4.11}$$

is satisfied for some number c that depends on $\underline{x}_i$ and $\mathcal{S}$. Therefore, because the algorithm may set $\hat{t}_i^*$ to a value of t such that the left hand side of this inequality is $\mathcal{O}(\eta)$, the discrepancy $|\hat{t}_i^* - t_i^*|$ is allowed to be of magnitude $\eta^{1/2}$, and then ζ_i is usually of this magnitude too.

Values of ζ_i that are slightly greater than 10^{-5} may be tolerable in the given experiments because of the choice $\eta = 10^{-10}$, but it was found that ζ_i hardly ever exceeded 10^{-10}. In particular, throughout the "$\mathcal{S}$ has $n/125$ cycles" tests, including the experiments on random choices of the angles of the circular arcs of $\mathcal{S}$, the fourth largest value of ζ_i was 3.7×10^{-11}. These very small values are incredible if the argument of the previous paragraph is relevant. It seems, therefore, that the splittings are so successful in practice that nearly all of the parameters t_i^*, $i=1,2,\ldots,n$, are determined by one of the analytic methods of Section 3. This hypothesis was checked by providing output from the Fortran program whenever the tolerance condition (3.7) is achieved, because then the analytic techniques are replaced by the formulae $t_p^* = \alpha_p(h^+)$ and $t_q^* = \beta_q(h^+)$. There were only four instances of this output throughout the "$\mathcal{S}$ has $n/125$ cycles" experiments, and they coincided with the three largest values of ζ_i, which were 3.5×10^{-5}, 1.2×10^{-5} and 1.4×10^{-6}. On the other hand, in the "$\mathcal{S}$ has one cycle" problems, the formulae $t_p^* = \alpha_p(h^+)$ and $t_q^* = \beta_q(h^+)$ were employed 10 times altogether, 58 of the problems having no applications, 2 of them having one, 2 of them having two, and one of them having four applications. Further, in the 5 experiments that satisfied the tolerance condition (3.7), all the values of $\max\{\zeta_i : i=1,2,\ldots,n\}$ were in the interval $[1.9\times10^{-8}, 1.6\times10^{-5}]$. Two values of ζ_i in the other problems were of magnitude 10^{-6}, and they deserve further attention, but the largest value otherwise in the "$\mathcal{S}$ has one cycle" experiments was $\zeta_i = 7.3 \times 10^{-13}$. Moreover, in all the occurrences of $t_p^* = \alpha_p(h^+)$ and $t_q^* = \beta_q(h^+)$, the indices had the property $q \leq p+2$. Therefore it may be possible to develop further analytic techniques that avoid the relatively low accuracy of the adjustment of h by the bisection method.

The author, however, does not know of any applications of the algorithm that require high accuracy. Further, his practical experience otherwise is just the calculation of the initial values of the variables of the optimization problem for image registration that is mentioned in Section 1. On the other hand, he has found in similar situations that the development of a new algorithm can create applications, because many computer users restrict attention to numerical computations that can be solved routinely by available software. Thus there is a possibility that the new algorithm will become very useful. In any case, the method generates the global solution of an optimization problem that is interesting from an approximation point of view. Further, typical CPU times are only a few seconds when there are thousands of variables and many local minima.

Acknowledgement

This work was done at the University of Canterbury, Christchurch, New Zealand. The author is very grateful for the support and hospitality that he received there, including the cooperation of members of the Mathematics Department. He is also grateful to two referees for several helpful suggestions.

References

I. Barrodale, R. Kuwahara, R. Poeckert and D. Skea (1993), "Side-scan sonar image processing using thin plate splines and control point matching", *Numerical Algorithms*, Vol. 5, pp. 85–98.

L.G. Brown (1992), "A survey of image registration techniques", *Computing Surveys*, Vol. 24, pp. 325–376.

J. Flusser (1992), "An adaptive method for image registration", *Pattern Recognition*, Vol. 25, pp. 45–54.

M.J.D. Powell (1996), "A thin plate spline method for mapping curves into curves in two dimensions", in *Computational Techniques and Applications: CTAC95*, editors R.L. May and A.K. Easton, World Scientific (Singapore), pp. 43–57.