Piecewise Convex-Concave Approximation in the $\ell_\infty$ Norm

M. P. CULLINAN
Maryvale Institute, Birmingham, B44 9AG, UK
mpcullinan@gmail.com
26 July 2010

Abstract
Suppose that $f \in \mathbb{R}^n$ is a vector of $n$ error-contaminated measurements of $n$ smooth values measured at distinct and strictly ascending abscissae. The following projective technique is proposed for obtaining a vector of smooth approximations to these values. Find $y$ minimizing $\|y - f\|_\infty$ subject to the constraints that the second order consecutive divided differences of the components of $y$ change sign at most $q$ times. This optimization problem (which is also of general geometrical interest) does not suffer from the disadvantage of the existence of purely local minima and allows a solution to be constructed in $O(nq)$ operations. A new algorithm for doing this is developed and its effectiveness is proved. Some of the results of applying it to undulating and peaky data are presented, showing that it is economical and can give very good results, particularly for large densely-packed data, even when the errors are quite large.

1. Introduction
Given a set of observations $f_j$, at strictly ascending abscissae $x_j$, for $1 \leq j \leq n$, it may be known that the observations represent measurements of smooth quantities contaminated by errors. A method is then needed for obtaining a smooth set of points while respecting the observations as much as possible. One method is to make the least change to the observations, measured by a suitable norm, in order to achieve a prescribed definition of smoothness. The data smoothing method of Cullinan & Powell (1982) proposes defining smoothness as the consecutive divided differences of the
1. INTRODUCTION

points of a prescribed order \( r \) having at most a prescribed number \( q \) of sign changes. There are many good reasons for using the number of sign changes in the divided differences of data as a criterion of smoothness. Normally data values of a smooth function will have very few sign changes, whereas if even one error is introduced, it will typically cause \( k \) sign changes in the \( k \)th order divided differences of the contaminated data. Thus constructing a table of divided differences is a cheap and sensitive test for smoothness (see, for example, Hildebrand, 1956). If the observations \( f_j \), for \( 1 \leq j \leq n \), are regarded as the components of a vector \( f \in \mathbb{R}^n \) and the function \( F : \mathbb{R}^n \to \mathbb{R} \) is defined through the chosen norm by \( F(\mathbf{v}) = \|\mathbf{v} - f\| \), then the data smoothing problem becomes the constrained minimization of \( F \). This approach has several advantages. There is no need to choose (more or less arbitrarily) a set of approximating functions, indeed the data are treated as the set of finite points which they are rather than as coming from any underlying function. The method is projective or invariant in the sense that it leaves smoothed points unaltered. It depends on two integer parameters which will usually take only a small range of possible values, rather than requiring too arbitrary a choice of parameters. It may be possible to choose likely values of \( q \) and \( r \) by inspection of the data. The choice of norm can sometimes be suggested by the kind of errors expected, if this is known. For example the \( \ell_1 \) norm is a good choice if a few very large errors are expected, whereas the \( \ell_\infty \) norm might be expected to deal well with a large number of small errors. There is also the possibility that the algorithms to implement the method may be very fast.

The main difficulty in implementing this method is that when \( q \geq 1 \), the possible existence of purely local minima of \( F \) makes the construction of an efficient algorithm very difficult. This has been done for the \( \ell_2 \) norm for \( r \leq 2 \)—see for example Demetriou (2002). The author dealt with the case \( q = 0 \) and arbitrary \( r \) for the \( \ell_2 \) norm (Cullinan, 1990).

It was claimed in Cullinan & Powell (1982) that when the \( \ell_\infty \) norm is chosen and \( r = 2 \), all the local minima of \( F \) are global and a best approximation can be constructed in \( O(nq) \) operations; and an algorithm for doing this was outlined. These claims were proved by Cullinan (1986) which also considered the case \( r = 1 \). It was shown that in these cases the minimum value of \( F \) is determined by \( q + r + 1 \) of the data, and a modified algorithm for the case \( r = 2 \) was developed which is believed to be better than that outlined in Cullinan & Powell (1982). This new algorithm will now be presented in Section 2 and its effectiveness will be proved. Section 3 will then describe the results of some tests of this method which show that it is a very cheap way of filtering noise but can be prone to end errors.
2. The algorithm

This section will construct a best $\ell_\infty$ approximation to a vector $f \in \mathbb{R}^n$ with not more than $q$ sign changes in its second divided differences. More precisely let $v \in \mathbb{R}^n$ with $x_1 < x_2 < \ldots < x_n$ and

$$F(v) = \|v - f\|_\infty$$

(2.1)

$$c_{ijk}(v) = \frac{1}{x_k - x_i} \left( \frac{v_k - v_j}{x_k - x_j} - \frac{v_j - v_i}{x_j - x_i} \right),$$

(2.2)

$$c_i(v) = c_{i,i+1,i+2}(v).$$

(2.3)

The feasible set of points $Y_q \subset \mathbb{R}^n$ is defined as the set of all vectors $v \in \mathbb{R}^n$ for which the signs of the successive elements of the sequence $1, c_1(v), \ldots, c_{n-2}(v)$ change at most $q$ times, and the problem is then to develop an algorithm to minimize $F$ over $Y_q$.

The solution depends on the fact that the value $h_q$ of the best approximation is determined by $q + 3$ of the data. Since a best $\ell_\infty$ approximation is not unique, there is some choice of which one to construct. The one chosen, $y$, has the following property: $y_1 = f_1$, $y_n = f_n$, and for any $j$ with $2 \leq j \leq n - 1$, if $\pm c_{j-1}(y) > 0$ then $y_j = f_j \pm h_q$.

The vector $y$ is then determined from $h_q$, from the set of indices $i$ where $c_{i-1}(y) \neq 0$, and from the ranges where the divided differences do not change sign.

The method by which the best approximation is constructed and the proof of the effectiveness of the algorithm that constructs it are best understood by first considering the cases $q = 0, 1, 2$ in detail and giving algorithms for the construction of a best approximation in each case. Once this has been done it is easy to understand the general algorithm.

When $q = 0$, this best approximation is formed from the ordinates of the points on the lower part of the boundary of the convex hull of the points $(x_j, f_j)$, for $1 \leq j \leq n$, (the graph of the data in the plane) by increasing these ordinates by an amount $h$.

When $q = 1$, there exist integers $s$ and $t$ such that $1 \leq s \leq t \leq n$, and $y_1, \ldots, y_s$ are the ordinates on the lower part of the boundary of the convex hull of the data $f_1, \ldots, f_s$ increased an amount $h$; $y_t, \ldots, y_n$ are ordinates on the upper part of the boundary of the convex hull of the data $f_t, \ldots, f_n$ decreased an amount $h$; and if $s < j < t$, $y_j$ lies on the straight line joining
2. THE ALGORITHM

\((x_s, y_s)\) to \((x_t, y_t)\). The best approximation therefore consists of a convex piece and a concave piece joined where necessary by a straight line.

The best approximation over \(Y_q\) consists of \(q + 1\) alternately raised pieces of lower boundaries of convex hulls and lowered pieces of upper boundaries of concave hulls joined where necessary by straight lines. These pieces are built up recursively from those of the best approximation over \(Y_{q-2}\).

The points on the upper or lower part of the boundary of the convex hull of the graph of a range of the data each lie on a convex polygon and are determined from its vertices. The algorithms to be described construct sets of the indices of these vertices and the value of a best approximation. The best approximation vector is then constructed by linear interpolation.

Before considering the cases \(q = 0, 1, \) and \(2\), an important preliminary result will be established. It is a tool that helps to show that the vectors constructed by the algorithms are optimal. The value \(h_q\) of the best approximation over \(Y_q\) will be found by the algorithm, together with a vector \(y \in Y_q\) such that \(F(y) = h_q\). To show that \(y\) is optimal, a set \(K\) of \(q + 3\) indices will be constructed such that if \(v\) is any vector in \(\mathbb{R}^n\) such that \(F(v) < h_q\), then the consecutive second divided differences of the components \(v_k\), for \(k \in K\), change sign \(q\) times starting with a negative one. It will then be inferred that \(v \not\in Y_q\). In order to make this inference it must be shown that the consecutive divided differences of all the components of \(v\) have at least as many sign changes as those of the components with indices in \(K\). This result will now be proved.

**Theorem 1** Let \(K \subseteq \{1, \ldots, n\}\) and let \(v \in \mathbb{R}^n\) be any vector such that the second divided differences of the \(v_k\), for \(k \in K\), change sign \(q\) times. Then the divided differences of all the components of \(v\) change sign at least \(q\) times.

**Proof**  Firstly, suppose that \(K\) is formed by deleting one element \(j\) from \(\{1, \ldots, n\}\) and that \(3 \leq j \leq n - 2\). Let \(c_{j-2}, c_{j-1}, c_j\) be defined from \(v\) by (2.2) and (2.3), and let \(c'_{j-2} = c_{j-2j-1j+1}(v)\) and \(c'_{j-1} = c_{j-1j+1j+2}(v)\) denote the new divided differences that result from deleting \(j\). Manipulation of (2.2) yields the equations

\[
\begin{align*}
    c'_{j-2} &= \frac{x_j - x_{j-2}}{x_{j+1} - x_{j-2}} c_{j-2} + \frac{x_{j+1} - x_j}{x_{j+1} - x_{j-2}} c_{j-1} \\
    c'_{j-1} &= \frac{x_{j+2} - x_j}{x_{j+2} - x_{j-1}} c_j + \frac{x_j - x_{j-1}}{x_{j+2} - x_{j-1}} c_{j-1},
\end{align*}
\]

and
2. THE ALGORITHM

so that \( c'_{j-2} \) lies between \( c_{j-2} \) and \( c_{j-1} \) and \( c'_{j-1} \) lies between \( c_{j-1} \) and \( c_j \). It follows that the number of sign changes in the sequence

\[
\cdots, c_{j-3}, c_{j-2}, c'_{j-2}, c_{j-1}, c'_{j-1}, c_j, c_{j+1}, \cdots
\]

is the same as that in the sequence

\[
\cdots, c_{j-3}, c_{j-2}, c_{j-1}, c_j, c_{j+1}, \cdots
\]

and hence that deleting \( c_{j-2}, c_{j-1}, c_j \) from (2.4) cannot increase the number of sign changes. The same argument covers the cases \( j = 2 \) and \( j = n - 1 \), and when \( j = 1 \) or \( j = n \) this result is immediate.

Repeated application of this result as elements \( j \) of the set \( \{1, \ldots, n\} \setminus K \) are successively deleted from \( \{1, \ldots, n\} \) then proves the theorem. □

**Corollary 1** If \( i \leq k - 2 \) and all the divided differences \( c_j(v), i < j < k \), are non-negative (or non-positive), and if \( i \leq r < s < t \leq k \), then \( c_{rst}(v) \) is also non-negative (or non-positive).

This Theorem is crucial to the effectiveness of the algorithm. In Cullinan (1986) it was proved that the set of best approximations is connected, so that purely local minima are ruled out, but that this is not the case for higher orders of divided differences. This Theorem allows the explicit construction of global minima determined by \( q + 3 \) of the data (and so it does not seem necessary to prove connectedness). There is no analogous result for higher order divided differences, and so no ready generalization of the methods of this paper to such cases.

2.1 The case \( q = 0 \)

When \( q = 0 \), the required solution is a best convex approximation to \( f \). The particular one, \( y^0 \), that will be constructed here was first produced by Ubhaya (1979). It will also be convenient to construct a best *concave* approximation to data.

The convex approximation is determined from the vertices of the lower part of the boundary of the convex hull of the points, and the concave approximation from the vertices of the upper part. Each of these sets of indices can be constructed in \( O(n) \) operations by the following algorithm. It is convenient to apply the construction to a general range of the data and to describe it in terms of sets of indices. Accordingly, define a *range* \([r, s]\) = \( \{j : r \leq j \leq s\} \), and a *vertex set* of \([r, s]\) to be any set \( I \) such that \( \{r, s\} \subseteq I \subseteq [r, s] \). Given a vertex set \( I \) of a range \([r, s]\) and also quantities \( v_i, i \in I \), define the *gradients*

\[
g_{ik}(v) = \frac{v_k - v_i}{x_k - x_i} \quad \text{for } i \neq k.
\]
2. THE ALGORITHM

Given any index \( j : 1 \leq j \leq n \), define the neighbours of \( j \) in \( I \) by

\[
j^+(I) = \min\{i > j\} \quad \text{when} \quad j < s
\]
\[
j^-(I) = \max\{i < j\} \quad \text{when} \quad j > r,
\]

and (for purposes of extrapolation)

\[
j^+(I) = s^-(I) \quad \text{when} \quad j \geq s
\]
\[
j^-(I) = r^+(I) \quad \text{when} \quad j \leq r.
\]

The interpolant \( v(I) \) can now be defined by

\[
v_j(I) = v_j \quad \text{for} \quad j \in I \tag{2.5}
\]
\[
v_j(I) = v_{j^-}(I) + g_{j^-}(I)(f)(x_j - x_{j^-}) \quad \text{for} \quad j \notin I. \tag{2.6}
\]

When \( I = \{p, q\} \) it is convenient to write \( v_j(I) \) as \( v_j(p, q) \) etc.

The two cases of convex and concave approximations are handled using the sign variable \( \sigma \), where \( \sigma = + \) for the convex case and \( \sigma = - \) for the concave case. The convex and concave optimal vertex sets \( I^+(r, s) \) and \( I^-(r, s) \) are then constructed by systematic deletion as follows.

**Algorithm 1** To find \( I^\sigma(r, s) \) when \( r \leq s - 2 \).

\begin{itemize}
    \item Step 1. Set \( I := [r, s] \) and \( i := r, j := r + 1, k := r + 2 \).
    \item Step 2. Evaluate \( c := c_{ijk}(f) \). If \( \sigma c > 0 \) go to Step 5.
    \item Step 3. Delete \( j \) from \( I \). If \( i = s \) go to Step 5.
    \item Step 4. Set \( k := j, j := i \), and \( i := i^-(I) \). Go to Step 2.
    \item Step 5. If \( k = t \) set \( I^\sigma(r, s) = I \) and **stop**.
        Otherwise: set \( i := j, j := k \), and \( k := k^+(I) \). Go to Step 2.
\end{itemize}

The price of making a convex/concave approximation in the range \([r, s]\) is given by

\[
h^\sigma(r, s) = \frac{1}{2} \max_{j \in [r, s]} \sigma(f_j - f_j(I^\sigma(r, s)) \tag{2.7}
\]

and the required best approximation \( y^0 \) is then defined by

\[
y^0_j = f_j(I) + h \tag{2.8}
\]

where \( I = I^+(1, n) \) and \( h = h^+(1, n) \).

The construction of \( y^0 \) is illustrated in Figure 1. Fig 101

**Theorem 2** Let \( n \geq 3 \), and let \( h_0 = h^+(1, n) \) and \( y^0 \) be defined from \( (2.7) \) and \( (2.8) \) using Algorithm 1, \( (2.5) \) and \( (2.6) \). Then \( F(y^0) = h_0 = \inf F(Y_0) \) and \( y^0 \in Y_0 \).
Proof} The first remark is that Algorithm 1 produces a well-defined vertex set $I$ of $[r, s]$ from which the quantities $f_j(I)$ are also well-defined for all $j$, so that $h_0$ and $y_0$ are also well-defined.

The proof that the points $(x_j, f_j(I))$, $1 \leq j \leq n$, lie on the lower part of the convex hull of the data is in Ubhaya. The components $f_j(I)$ can be defined as those that are maximal subject to the inequalities

$$f_j(I) \leq f_j, \quad \text{for } 1 \leq j \leq n, \quad (2.9)$$

and

$$c_i(f(I)) \geq 0, \quad \text{for } 1 \leq i \leq n - 2, \quad (2.10)$$

i.e. $f(I) \in Y_0$. It follows that $y^0 \in Y_0$ and from (2.7) and (2.9) that $F(y^0) = h$.

It remains to prove that $y^0$ is optimal. If $h = 0$, $y^0$ must be optimal. If $h > 0$, there will be a lowest integer $j^* \notin I$ such that equality is attained in (2.7), and since 1 and $n$ can never be deleted from $I$, it must be the case that $2 \leq j^* \leq n - 1$. Then

$$y^0_j = f_{j^*} - h_0,$$

and $k = j^-(I)$ and $k^+ = j^+(I)$ are consecutive elements of $I$ such that $k < j < k^+$ and

$$y^0_k = f_k + h_0,$$
2. THE ALGORITHM

\[ y_{k^+}^0 = f_{k^+} + h_0. \]

Since \( j^* \not\in I \), \( c_{k,j^*,k^+}(y^0) = c_{k,j^*,k^+}(f(I)) = 0 \). Now if \( v \in \mathbb{R}^n \) and \( F(v) < h \), then

\[ v_k < f_k + h = y_k^0, \]
\[ v_{j^*} > f_{j^*} - h = y_{j^*}^0, \]
\[ v_{k^+} < f_{k^+} + h = y_{k^+}^0. \]

Now the constraint function \( c_{k,j^*,k^+} \) is an increasing function of \( v_k \) and \( v_{k^+} \) and a decreasing function of \( v_{j^*} \), so \( c_{k,j^*,k^+}(v) < c_{k,j^*,k^+}(y^0) = 0 \). It now follows from Theorem 1 that \( v \not\in Y_0 \), and thus that \( h = h_0 \) and \( y^0 \) is optimal. □

**Corollary 2** The components of a best concave approximation to data \( f_j \), \( r \leq j \leq s \), are given from Algorithm 1 with \( \sigma = -1 \) by \( y_j = f_j(I) - h \), where in this case \( I = I^-(r,s) \) and \( h = h^-(r,s) \).

The algorithms in the next subsections will join optimal vertex sets produced by Algorithm 1 of consecutive ranges of the data, and they will also cut such optimal vertex sets in two. It is convenient to prove here that the resulting sets remain optimal vertex sets. The proof requires one further property of the optimal vertex sets produced by Ubhaya’s algorithm.

This algorithm is a specialization of an algorithm by Graham (1972) for finding the convex hull of an unordered set of points in the plane. It is logically equivalent to the algorithm of Kruskal (1964) for monotonic approximation, which is more efficient than the algorithm of Miles (1959) for monotonic approximation, but produces the same results. Two conclusions follow from this. The first, which will be needed later, is that if \( i \) and \( k \) are consecutive elements of \( F^\sigma(r,s) \) then

\[ \sigma g_{ik}(f(I)) = \min \{ \sigma g_{ij}(f(I)) : i \leq j \leq s \} \quad \text{and} \quad (2.11) \]
\[ \sigma g_{ik}(f(I)) = \min \{ \sigma g_{jk}(f(I)) : r \leq j \leq i \}. \quad (2.12) \]

The second, which is of some theoretical interest, is that the gradients \( g_{j+1}(f(I)) \), for \( 1 \leq j \leq n - 1 \), are the best least squares approximations to the numbers \( g_{j+1}(f) \), for \( 1 \leq j \leq n - 1 \), subject to the constraint that they monotonically increase, i.e. to convexity of the points of which they are gradients. The proof of this interesting equivalence is a straightforward consequence of the equivalence of Kruskal’s and Miles’s algorithms for monotonic approximation. It is given in Cullinan (1986).
2. THE ALGORITHM

The results for the joining and splitting of optimal vertex sets require definition of trivial optimal vertex sets by

\[ I^\sigma(r, s) = [r, s] \quad \text{when } s - 1 \leq r \leq s. \]  

(2.13)

It is also convenient to define \( h^\sigma(r, s) = 0 \) when \( r \geq s - 1 \). The following lemmas then hold.

**Lemma 1** A subset of one or more consecutive elements of an optimal vertex set is itself an optimal vertex set.

**Proof** The proof is trivial unless the subset has at least four elements, when it follows from the nature of extreme points of convex sets using (2.11) and (2.12). □

The second lemma gives conditions for the amalgamation of optimal vertex sets.

**Lemma 2** Given vertex sets \( I^\sigma(r, s) \) and \( I^\sigma(s, t) \) necessary and sufficient conditions for

\[ I^\sigma(r, t) = I^\sigma(r, s) \cup I^\sigma(s, t) \]  

(2.14)

are that there exist \( r' \leq r, r' \in I^\sigma(r, s) \), and \( t' \geq t, t' \in I^\sigma(s, t) \), such that

\[ s \in I^\sigma(r', t'). \]  

(2.15)

**Proof** The proof is trivial except for the sufficiency of (2.15) when \( r < s < t \). Let \( s^- \) be the left neighbour of \( s \) in \( I^\sigma(r, s) \) and \( s^+ \) its right neighbour in \( I^\sigma(s, t) \). It follows from (2.11) and (2.12) that (2.14) holds provided that

\[ \sigma_{c_{s^-, s^+}}(f) > 0. \]

However, it follows from (2.15) that \( \sigma_{c_{ij}}(f) > 0 \) for all \( i \) and \( j \) in the range \( r' \leq i < s < j \leq t' \), and this range contains \( s^- \) and \( s^+ \). □

2.2 The case \( q = 1 \)

The algorithm to be presented constructs ranges \([1, s] \) and \([t, n] \), where \( s \leq t \), a price \( h \geq 0 \), and a vertex set \( I \) of \([1, n] \) such that

\[ I = I^+(1, s) \cup I^-(t, n). \]
2. THE ALGORITHM

The best approximation $\mathbf{y}^1$ is then given by the final value of the vector $\mathbf{y}$ defined by

$$y_i = f_i + h \quad \text{when } i \in I^+(1,s) \quad (2.16)$$

$$y_i = f_i - h \quad \text{when } i \in I^-(t,n) \quad (2.17)$$

$$y_j = y_j(I) \quad 1 \leq j \leq n \quad (2.18)$$

It will be shown that $s = t$ only if $h = 0$, so that $\mathbf{y}^1$ is well defined. This construction is illustrated in Figure 2. Note that when $s = t = n$, $\mathbf{y} = \mathbf{y}^0$.

The algorithm to be given is believed to be more efficient than that in Cullinan & Powell (1982). For example if the data are in $Y_0$, the new algorithm will require only iteration, whereas the former one only has this property if the data lie on a straight line.

The algorithm builds up $I$ by looking alternately at the left and right ranges. Beginning with $I = \{1, n\}$, $s = 1$, and $t = n$, it adds an index $k$ of $I^+(s,t)$ to $I$ if the least possible final value of $F$ consistent with doing this is not greater than the least possible value of $F$ consistent with ending the calculation with the existing value of $s$. After adding one index in $I^+(s,t)$ to $I$ and increasing $s$ to the value of this index, it then examines the next. When it is not worth adding any more indices from $I^+(s,t)$ to $I$, it tries to add indices in $I^-(s,t)$ to $I$ working backwards from $t$, adding $k$ to $I$ if it is not necessarily more expensive to finish with $t$ reduced to $k$ than with $t$ as

![Figure 2: Construction of best convex-concave approximation](image-url)
2. **THE ALGORITHM**

It is and decreasing $t$. After indices have been added to $I$ from $I^-(s, t)$ it may then be possible to add more to $I$ from the new $I^+(s, t)$, so the process alternates between $I^+(s, t)$ and $I^-(s, t)$ until $s$ equals $t$ or $t - 1$ or until the algorithm fails twice running to add any indices.

**Algorithm 2** To find a best convex-concave approximation when $n \geq 2$.

Step 1. Set $s := 1$, $t := n$, $h := 0$, and $I := \{1, n\}$.

Step 2. If $s \leq t - 1$: stop. Otherwise: set $u = t - s$ and $I' := I^+(s, t)$.

Step 3. Let $s^+ = s^+(I')$. Find $h^+(s, s^+)$ and set $h' := \max(h, h^+(s, s^+))$.

If $f_t(s, s^+) > f_t - 2h'$: go to Step 5.

Step 4. Add $s^+$ to $I$ and delete $s$ from $I'$. Set $h := h'$ and $s := s^+$.

If $s < t - 1$: go back to Step 3.

Step 5. If $s = t$: stop. Otherwise: set $I' := I^-(s, t)$.

Step 6. Let $t^- = t^-(I')$. Find $h^-(t^-, t)$ and set $h' := \max(h, h^-(t^-, t))$.

If $f_s(t^-, t) < f_s + 2h'$: go to Step 8.

Step 7. Add $t^-$ to $I$ and delete $t$ from $I'$. Set $h := h'$ and $t := t^-$.

If $s < t$: go back to Step 6.

Step 8. If $t - s = u$: stop. Otherwise: go back to Step 2.

The optimal vertex sets $I'$ are found using Algorithm 1 or Corollary 2.1. The prices $h^\sigma(s, t)$ are found from (2.7) and $f_t(s, s^+)$ and $f_s(t^-, t)$ from (2.6).

The first remark is that $s$ is non-decreasing, $t$ is non-increasing, and $s \leq t$ with $s = t$ only when $h = 0$. The vector $y^1$ is therefore well-defined by (2.16)–(2.18).

The possibility that Algorithm 1 can end with $s = t$ will involve slightly more complexity when this Algorithm is used later on when $q \geq 3$. It might be prevented by relaxing either of the inequalities in Steps 3 and 6. However if both these inequalities are relaxed then the Algorithm will fail. For example with equally spaced data and $f = (0, 1, 0, 1)$, relaxing both inequalities yields $s = 1$, $t = 4$, and $F(y) = \frac{2}{3}$, whereas the Algorithm as it stands correctly calculates $h = \frac{1}{2}$ and $y = (\frac{1}{4}, \frac{1}{4}, \frac{1}{2}, 1)$. It seemed best to let both inequalities be strict for reasons of symmetry.

The next result concerns the conditions under which the Algorithm increases $s$ and decreases $t$. 
2. THE ALGORITHM

Lemma 3  At any entry to Step 2 of Algorithm 2, let the vector $y$ be defined by (2.16)–(2.18). Then the Algorithm strictly decreases $t - s$ if and only if

$$\text{there exists } j : s < j < t \quad \text{and} \quad |f_j - y_j| \geq h. \quad (2.19)$$

(This somewhat cumbersome statement is needed to include the case where $h = 0$.)

Since the points $(x_j, y_j)$, $s \leq j \leq t$, are collinear, (2.19) is equivalent to the statement that there exists a point of the graph of the data between $x_s$ and $x_t$ lying on or outside the parallelogram $\Pi(h)$ with vertices $(x_s, f_s)$, $(x_s, f_s + 2h)$, $(x_t, f_t)$, and $(x_t, f_t - 2h)$. This parallelogram $\Pi(h)$ is illustrated in Figure 3.

**Proof**  In the trivial case when $s \geq t - 1$, (2.19) is false and the Algorithm stops without altering $s$ or $t$.

Otherwise, there are one or more data points $f_j$ with $s < j < t$. Suppose first that (2.19) does not hold. Then $h > 0$. (If $h = 0$ then (2.19) holds trivially!) It will be shown that in this case both $s$ and $t$ are left unchanged. Step 3 will calculate an index $s^+ : s < s^+ \leq t$.

If $s^+ = t$, then because $h' \geq h > 0$, it immediately follows that $f_t(s, s^+) = f_t > f_t - 2h'$, so that Step 3 will lead immediately to Step 5 and $s$ will not be increased.

If $s^+ < t$, let $y_{s^+}$ be defined by (2.16)–(2.18) and let $g = g_{st}(y)$. Then by hypothesis, $f_{s^+} > y_{s^+} - h$, which implies that $g_{s,s^+}(f) > g$. Thus,

$$f_t(s, s^+) = f_s + g_{s,s^+}(f)(x_t - x_s)$$
2. THE ALGORITHM

\[ f_s + g(x_t - x_s) = y_t - h = f_t - 2h \]
\[ \geq f_t - 2h'. \]

Therefore Step 3 will again lead immediately to Step 5.

The same arguments applied to \( t^- \) calculated by Step 6 show that Step 6 will branch immediately to Step 8 and so \( t \) will also be left unchanged.

Now suppose conversely that (2.19) does hold, so that there is an index \( j \) with data point lying on or outside \( \Pi \). It will be shown that in this case if the algorithm does not increase \( s \), it must then decrease \( t \).

Consider first the case where \( f_j \leq y_j - h \). Step 3 will calculate an index \( s' \) in the range \( s < s' \leq t \), and, from (2.11), with \( g_{ss'}(f) \leq g_{sj}(f) \).

\[
f_t(s, s^+) = f_s + g_{ss'}(f)(x_t - x_s)
\leq f_s + g_{sj}(f)(x_t - x_s)
= f_s + ((f_j - f_s)/(x_j - x_s))(x_t - x_s)
\leq f_s + ((y_j - h - f_s)/(x_j - x_s))(x_t - x_s) \text{ by hypothesis (2.23)}
= y_s - h + ((y_j - y_s)/(x_j - x_s))(x_t - x_s)
= y_t - h, \text{ by the definition of } y_j.
\]

It now follows immediately that when \( h' = h \), Step 4 will be entered and \( s \) increased to \( s^+ \), as required.

Now suppose that \( h' = h^+(s, s^+) > h \) and that \( s \) is not increased, i.e. that the test in Step 3 leads to Step 5. Then \( f_t(s, s^+) > f_t - 2h' = f_t - 2h^+(s, s^+) \).

Let \( h^+ = h^+(s, s^+) \). By definition of \( h^+ \), there must be an index \( i \) such that \( f_i - f_i(s, s^+) = 2h^+ \). This case is illustrated in Fig. 4. It demonstrates the heart of the principle behind the algorithm, because the four data points with indices \( s, i, s^+, \) and \( t \) determine a lower bound on \( \inf F(Y_1) \).

Define

\[
2d = f_t - f_t(s, s^+).
\]

Then by the hypothesis that Step 3 led to Step 5,

\[
h^+ > d.
\]

Step 5 will be entered with \( h < h^+ \) and will calculate \( I^-(s, t) \). Step 6 will then find an index \( t^- : s \leq t^- < t \). It follows from (2.11) that \( f_i \leq f_i(t^-, t) \), and so

\[
2h^+ = f_i - f_i(s, s^+) \leq f_i(t^-, t) - f_i(s, s^+).
\]

This inequality along with (2.26) and (2.27) then show that the function \( l \mapsto f_l(t^-, t) - f_l(s, s^+) \) is strictly decreasing.
2. THE ALGORITHM

Figure 4: When $s$ does not increase, $t$ must decrease

If $h$ does not increase in Step 7, it follows immediately, from the assumption that $h < h^+$, that $f_s(t^-, t) - f_s(s, s^+) = f_s(t^-, t) - f_s > 2h^+ > 2h$, so that $t$ will be reduced to $t^-$ by the test in Step 6.

If, on the other hand, $h$ does increase in Step 7, its new value $h'$ is given by $2h' = f_k(t^-, t) - f_k$ for some $k$ in the range $s < k < t$. But the definition of $s^+$ implies that $f_k \geq f_k(s, s^+)$, and so $2h' \leq f_k(t^-, t) - f_k(s, s^+) < f_s(t^-, t) - f_s(s, s^+)$, so that again $t$ must be reduced to $t^-$ by Step 7. In fact it is easy to show that in this case, $t^- \geq i$, $h' < h^+$, and that $\inf F(Y_1) \geq \frac{1}{2}(f_{s^+}(i, t) - f_{s^+})$.

Thus $t$ will be reduced to $t^-$ in all cases.

In the case where there exists $j$ such that $f_j \geq y_j + h$, if $s$ is not increased immediately after the next entry to Step 3, the same argument shows that either $t$ must be reduced in the next operation of Steps 6 and 7, or $s$ must be increased immediately thereafter.  

**Lemma 4** At any exit from Step 4 or Step 7, let $y$ be defined by (2.16)--(2.18). Then

\[ \text{when } s > 1, \quad c_{s-1}(y) \geq 0, \quad (2.29) \]

and

\[ \text{when } t < n, \quad c_{t-1}(y) \leq 0. \quad (2.30) \]

**Proof** Suppose first that Step 3 is entered. Then (2.29) is equivalent to

\[ c_{s+1}(y) \geq 0, \quad (2.31) \]
which is in turn equivalent to the identity
\[ f_t - f_t(s, s^+) \geq 2h'. \tag{2.32} \]

But this is simply the test leading to Step 4.

If \( t < n \), there will exist an index \( t^+ = t^+(I) \), and \( f(s^+)(t, t^+) \geq 2h' \).

Suppose firstly that \( h' > h \), so that there exists \( i \) such that \( s < i < s^+ \) and
\[ f_i - f_i(s, s^+) = 2h'. \tag{2.34} \]

The earlier definition of \( t \) from \( t^+ \) implies that \( f_i \leq f_i(t, t^+) \) and it follows from \( 2.34 \) and \( 2.32 \) that the monotonic function \( \phi : l \mapsto f_l(t, t^+) - f_l(s, s^+) \) satisfies \( \phi(i) \geq 2h' \) and \( \phi(t) \geq 2h' \). It follows that \( \phi(s^+) \geq 2h' \), which is equivalent to \( 2.33 \). It follows that whenever \( h \) increases, both the join constraints \( c_{s-1} \) and \( c_{t-1} \) (when defined) are feasible.

When Step 4 is entered and \( h \) does not increase, if \( c_{t-1} \) is initially non-positive, which is equivalent to the inequality
\[ f_s(t, t^+) - f_s \geq 2h, \tag{2.35} \]
it follows from this and \( 2.32 \) with \( h' = h \) that \( \phi(s) \geq 2h \) and \( \phi(t) \geq 2h \) so that \( \phi(s^+) \geq 2h \) as required.

Similarly when Step 7 is entered and \( h \) does not increase, if \( c_{s-1} \) is initially non-negative, it will remain so when \( s \) is increased.

The result then follows by induction. \( \square \)

**Corollary 3** If steps 2 to 8 of Algorithm 2 are executed with \( h \) initially set to any number \( h > 0 \), then \( 2.29 \) and \( 2.30 \) remain satisfied.

The effectiveness of Algorithm 2 will now be established.

**Theorem 3** Algorithm 2 produces integers \( s \) and \( t \) with \( s \leq t \), an index set \( I \) such that
\[ I = I^+(1, s) \cup I^-(t, n), \tag{2.36} \]
and a real number \( h \) such that \( h = 0 \) if and only if \( s = t \). If \( y^1 \) is then defined by \( 2.16 \)–\( 2.18 \), then \( F(y^1) = h = \inf F(Y_1) \) and \( y^1 \in Y_1 \).
Proof  The first stage is to establish (2.36). Assume inductively that it holds before a series of sections is added, and without loss of generality that a series of convex sections with indices \(\{s, s_1, \ldots, s_\alpha\}\) are added by successive entries to Step 4 for the same value of \(t\). Each time an index is deleted from \(I'\) it follows immediately from Lemma 1 that the new value of \(I'\) is also an optimal index set, so it always holds that \(I' = I^+(s, t)\). It also follows from Lemma 1 that \(\{s, s_1, \ldots, s_\alpha\}\) = \(I' + (s, s_\alpha)\). Let \(I_1 = \{i \in I : i \leq s\}\). After all the sections are added, \(I_1 = I^+(1, s) \cup I^+(s, s_\alpha)\). If \(s = 1\), it follows at once from Lemma 2 that \(I_1 = I^+(1, s_\alpha)\) as required. Otherwise, let \(t'\) be the value that \(t\) had when \(s\) was increased from \(s - 1\). Then \(s_\alpha \leq t \leq t'\). Immediately after \(s\) was increased from \(s - 1\), \(s \in I^+(s - 1, t')\). The conditions of Lemma 2 are therefore satisfied and \(I_1 = I^+(1, s_\alpha)\). The same argument applied to concave sections then establishes (2.36).

The algorithm clearly produces a number \(h\) such that \(s = t\) only if \(h = 0\). It is a consequence of Lemma 3 that if \(s < t - 1\) and \(h = 0\), then the algorithm will reduce \(t - s\). If \(s = t - 1\) and \(h = 0\), Step 4 will increase \(s\) to \(t\). Thus \(h = 0\) if and only if \(s = t\). Thus \(y^1\) is well defined by (2.16)–(2.18). The number \(h\) is given by \(h = \max(h(1), h(2))\) where

\[
h(1) = h^+(s, t), \quad (2.37)
\]

\[
h(2) = h^-(t, n). \quad (2.38)
\]

It follows from (2.36) and Lemma 3 that when the algorithm terminates,

\[
F(y^1) = h. \quad (2.39)
\]

It follows directly from (2.36), (2.16)–(2.18) and Lemma 4 that \(y^1 \in Y_1\).

It remains to prove that \(y^1\) is optimal. The method of proof chosen to do this can be simplified in this case, but generalizes more directly to the case of \(q > 1\). If \(h = 0\), optimality follows immediately from (2.39). Otherwise suppose that there is a vector \(v\) such that \(F(v) < h\). The price \(h\) will be defined from (2.7) with particular values of \(\sigma, r,\) and \(s\). Let \(j^*\) be the lowest value of \(j\) in this equation that defines the final value of \(h\). Then \(j^*\) lies strictly between two neighbouring elements \(k\) and \(k^+\) of \(I\). Assume firstly that \(j^* < s\). Define the set \(K = \{k, j^*, k^+, s, t\}\). Since \(h > 0\), \(s < t\), so \(K\) has at least four elements (it is possible that \(k^+ = s\)). Now

\[
y_k^1 = f_k + h, \\
y_j^1 = f_j^* - h, \\
y_k^1 = f_k^+ + h, \\
y_s^1 = f_s + h, \\
y_t^1 = f_t - h.
\]
Since $F(v) < h$,

\[
\begin{align*}
  v_k &< f_k + h, \\
  v_j &> f_j + h, \\
  v_{k+} &< f_{k+} + h, \\
  v_s &< f_s + h, \\
  v_t &> f_t - h.
\end{align*}
\]

By definition of $y_{j^*}^1$,

\[
c_{kj^*k}(y^1) = 0,
\]

and so $c_{kj^*k}(v) < c_{kj^*k}(y^1) = 0$. From Lemma 4, $c_{i-1}(y^1) \geq 0$ for all $i$ in the range $j^* \leq i \leq t - 1$. It follows from the corollary to Theorem 1 that

\[
c_{j^*st}(y^1) \geq 0.
\]

Then $c_{j^*st}(v) > c_{j^*st}(y^1) \geq 0$. If $k^+ = s$, Theorem 1 can be immediately applied to $K$ to show that $v \notin Y_1$. If $k^+ < s$, it follows from the corollary to Theorem 1 that because $c_{j^*st}(v) > 0$, at least one of the consecutive divided differences $c_{j^*k+s}(v)$ and $c_{k+s}(v)$ must be positive, so that again $v \notin Y_1$.

If $j^* > s$, let $K = \{s, t, k, j^*, k^+\}$. Then the same argument shows that $v$ cannot be feasible. Therefore $y^1$ is optimal. □

### 2.3 The Case $q = 2$

The best $Y_1$ approximation constructed in Section 2.2 was defined by equations (2.16)–(2.18) from the pieces $[1, s]$ and $[t, n]$, the index set $I$, and the price $h$. The best $Y_2$ approximation will in general be constructed from three pieces $P_1 = [1, s_1]$, $P_2 = [t_1, s_2]$, $P_3 = [t_2, n]$, a price $h \geq 0$ with $h > 0$ only when $s_1 < t_1$ and $s_2 < t_2$, and an index set $I$ of $[1, n]$ such that

\[
I = I^+(P_1) \cup I^-(P_2) \cup I^+(P_3),
\]

as the ultimate value $y^2$ of the vector $y$ defined by the equations

\[
\begin{align*}
  y_i &= f_i + (-)^{\alpha-1} h & \text{when } i \in I \cap P_{\alpha}, \quad 1 \leq \alpha \leq 3, \\
  y_j &= y_j(I) & 1 \leq j \leq n.
\end{align*}
\]

The construction of $y$ is illustrated in Figure 5.

The Algorithm will construct this best $Y_2$ approximation from the quantities $h_0 = h^+(1, n)$ and $I^+(1, n)$ constructed by Algorithm 1. If the value $h_0$ of this approximation is zero, the best approximation over $Y_0$ is also a
2. THE ALGORITHM

Figure 5: Construction of $Y_2$ approximation

best approximation over $Y_2$. Otherwise, $h_0 > 0$ is determined by three data $f_k$, $f_{j^*}$, and $f_{k^+}$ such that $k$ and $k^+$ are consecutive elements of $I^+(1, n)$ and $k < j^* < k^+$. The discussion in Section 2.1 shows that unless the divided differences of $y$ change sign at least once in the range $[x_k, x_{k^+}]$, then $F(y) \geq h_0$. The set $I^-(t_1, s_2)$ is therefore put in this range. The Algorithm begins with $s_1 = k$, $t_1 = s_2 = j^*$, and $t_2 = k^+$. It then sets $h = \max(h^+(1, s_1), h^+(t_2, n))$. Next, it uses Algorithm 2 modified to calculate a best convex-concave approximation to the data with indices in $[1, j^*]$ consistent with paying this minimum price $h$. It is an important feature of the problem that this can be done by starting Algorithm 2 with $s = k$ and $t = j^*$, i.e. the existing elements of $I^+(1, j^*)$ below $k$ can be kept in place.

This process can increase $s_1$ beyond $k$, reduce $t_1$ below $j^*$, and increase $h$. Let $h^{(1)}$ be its new value. A best concave-convex approximation to $f_{j^*}, \ldots, f_n$ starting with $h = h^{(1)}$ is next identified by applying a modified version of Algorithm 2 with $s = j^*$ and $t = k^+$, in general increasing $s_2$ beyond $j^*$ and reducing $t_2$ below $k^+$. If this second calculation does not increase $h$ above $h^{(1)}$, the best approximation $y$ can be constructed immediately from (2.41)–(2.42). If, however, $h > h^{(1)}$, there is the complication that the lower value of $h$ when the first calculation took place may have joined too many sections for the first join constraints $c_{s_1-1}(y)$ and $c_{t_1-1}(y)$ determined by the new higher value of $h$ to have the right signs. In this case the remedy proposed is to repeat the first calculation starting with the new value of $h$.

The following algorithms will therefore require a modified version of Algorithm 2 to carry out a best convex-concave approximation or a best concave-convex approximation on a range of data, starting with a prescribed value of $h$. This task is best carried out by modifying Algorithm 2 in detail, but
the following description is equivalent and simpler. The following procedure calculates a best approximation to the data in the range \([s_\alpha, t_\alpha]\) compatible with an existing price \(h\), the approximation being over \(Y_1\) or \(Y_{-1}\) according as \(\alpha\) is odd or even. It can be seen as trying to close the join \([s_\alpha, t_\alpha]\) as much as possible by constructing the best approximation to this range of data compatible with the given starting value of \(h\).

The notation for the pieces is based on the observation that when \(h > 0\), \(t_\alpha = s_\alpha^+(I)\) and when \(h = 0\), \(t_\alpha = s_\alpha\), so that the location of each of the two joins \([s_1, t_1]\) and \([s_2, t_2]\) can be specified simply by consecutive elements of \(I\). Thus the location of the pieces and joins can be specified simply through the quantities \(s_\alpha, 1 \leq \alpha \leq 2\). It is convenient to regard these as members of an ordered subset \(S\) of \(I\), and to include \(n\) in \(S\). Given an index set \(I\), define a piece set \(S\) of \(I\) to be an ordered subset of \(I\) such that \(n \in S\). Given an index set \(I\), define a piece set \(S\) of \(I\) to be an ordered subset of \(I\) such that \(n \in S\).

Algorithm 3 \(\text{closejoin}(\alpha)\)

Step 1. If \(\alpha\) is even: replace \(f\) by \(-f\).

\[s = s_\alpha \text{ and } t = s_\alpha^+(I).\]

Step 2. Carry out Steps 2 to 8 of Algorithm 2.

Step 3. Set \(s_\alpha = s\).

If \(\alpha\) is even: replace \(f\) by \(-f\).

Note that this procedure modifies \(h, I,\) and \(S\).

The algorithm constructs \(y^2\) by calculating the appropriate pieces and a price \(h(S)\) from which \(y^2\) is constructed. The notation used for keeping track of the pieces needs to cover the case of pieces that consist of only one point, and to generalize easily when \(q > 2\). It also has to cope with the trivial cases where \(f \in Y_0\) or \(f \in Y_1\) and there are, therefore, only one or two pieces instead of three.

Given a piece set \(S\), define its price \(h(S)\) as follows. Let \(q' = |S| - 1, t_0 = 1,\) and when \(q' \geq 1\) define \(t_\alpha = s_\alpha^+(I)\) when \(1 \leq \alpha \leq q'\). Then

\[
h(S) = \max\{ h^{(\alpha - 1)}(t_{\alpha - 1}, s_\alpha) : 1 \leq \alpha \leq q' + 1 \}. \tag{2.43}
\]

Recall that \(h^*(t, s) = 0\) whenever \(s \geq t - 1\).

It is also worth recording the location of the data points determining the optimal value of \(h\). Given a piece set \(S\), when \(h(S) > 0\) it follows from (2.43) and (2.7) that there exists a lowest index \(j^*(S)\) such that

\[
h(S) = \frac{1}{2} \sigma(f_{j^*} - f_{j^*}(k, k^+)), \tag{2.44}
\]
2. THE ALGORITHM

where \( \sigma = (-)^{\beta-1}, 1 \leq \beta \leq q' + 1, \) and

\[
t_{\beta-1} \leq k < j^* < k^+ \leq s_{\beta}. \tag{2.45}
\]

Once \( j^* \) is known, the quantities \( k, k^+, \) and \( \beta \) are uniquely determined by (2.44) and (2.45).

The following algorithm then constructs \( h, I \) and \( S \) determining \( y^2 \).

**Algorithm 4** To find a best approximation over \( Y_2 \).

Step 1. Set \( S = \{n\} \). Use Algorithm 1 to calculate \( I = I^+(1,n) \) and \( h = h^+(1,n) \).

If \( h = 0 \): stop.

Otherwise: find \( j^*, k, k^+ \) determined from \( S \) by (2.44) and proceed to Step 2.

Step 2. Insert \( j^* \) into \( I \) and \( k, j^* \) into \( S \). Calculate \( h = h(S) \).

Step 3. Apply \( closejoin(1) \). Set \( h^{(1)} = h \).

Step 4. Apply \( closejoin(2) \). If \( h = h^{(1)} \): stop.

Otherwise proceed to Step 5.

Step 5. If \( h^{(1)} = 0 \): go to Step 6.

If \( s_1 = k \) and \( s_1^+(I) = j^* \): stop.

Otherwise set \( s = s_1 \) and \( t = s_1^+ \) and calculate \( g^{(1)} = (f_t - f_s - 2h)/(x_t - x_s) \).

If \( s > k \) and \( g_s < g_{s^+(I)} > g \): go to Step 6.

If \( t < j^* \) and \( g_t < g_{j^+(I)} > g \): go to Step 6.

Otherwise: stop.

Step 6. Set \( s_1 = k \). Delete all elements of \( I \) lying strictly between \( k \) and \( j^* \) and then apply \( closejoin(1) \).

Now define the vector \( y(S) \) from \( S \) and \( h \) as follows. Let \( q' = |S| - 1 \) and \( t_0 = 1 \). When \( q' \geq 1 \) define \( t_{\alpha} = s_{\alpha}^+(I) \) when \( h > 0 \) and \( t_{\alpha} = s_{\alpha} \) when \( h = 0 \), for \( 1 \leq \alpha \leq q' \). Then let

\[
P_{\alpha} = [t_{\alpha-1}, s_{\alpha}] \quad \text{when } 1 \leq \alpha \leq q' + 1, \tag{2.46}
\]

and define \( y(S) \) by (2.41)–(2.42). Set \( y^2 \) to the value of \( y(S) \) on exit from Algorithm 4.

Step 5 is designed to avoid calculating gradients unnecessarily. The following lemma will be used to justify this economy and also to show that \( y^2 \) is
feasible. Recall the parallelogram $\Pi(h)$ defined in the proof of Lemma 3, and given $I$ and $S$, let $s = s_\alpha$ and $t = s_+^\alpha(I)$ and define $\Pi_S^\alpha(h)$ as the parallelogram with vertices $(x_s, f_s)$, $(x_s, f_s + 2(-1)^{\alpha-1}h)$, $(x_t, f_t)$, and $(x_t, f_t - 2(-1)^{\alpha-1}h)$. Each such parallelogram then defines a join gradient
\[ g^{(\alpha)}(S, h) = \frac{f_t - f_s - 2(-1)^{\alpha-1}h}{x_t - x_s}. \]  

(2.47)

**Lemma 5** Let closejoin $(\alpha)$ be called, modifying $h'$, $I'$, and $S'$ to $h$, $I$, and $S$ respectively. Suppose that there exists $h \geq 0$ such that
\[(x_j, f_j) \in \Pi_{s_\alpha}^\alpha(h) \quad \text{for} \quad s_\alpha' \leq j \leq s_\alpha^+(I'). \]
and that
\[ h' \leq h. \]  

(2.48)

Then
\[ h \leq h. \]  

(2.49)

Further, if $\bar{y} = g^{(\alpha)}(S', h)$, $g^+ = g_{s_\alpha'}(y(S))$ and $g^- = g_{s_\alpha^+(I')-1}(y(S))$, then
\[ (-1)^{\alpha-1} \min(g^+, g^-) \geq (-1)^{\alpha-1} \bar{y}. \]  

(2.51)

**Proof** Assume for simplicity that $\alpha$ is odd and write $s' = s_\alpha'$ and $t' = s_\alpha^+(I')$.

First consider (2.50). The proof is trivial unless closejoin increases $h$. In this case, $h > 0$ and so there exist $j^*$, $k$, $k^+ \in I$ such that
\[ 2h = f_{j^*} - f_{j^*}(k, k^+). \]  

(2.52)

where $s' \leq k < j^* < k^+ \leq t'$. Let $t_\alpha = s_\alpha^+(I)$. It follows from Lemma 3 that either $j^* \leq s_\alpha$ or $j^* \geq t_\alpha$. The two cases are entirely similar: it suffices to consider the first. Define $\bar{y}_j$, $s' \leq j \leq t'$, by $\bar{y}_{s'} = f_{s'} + h$, $\bar{y}_{t'} = f_{t'} - h$, and $\bar{y}_j = \bar{y}_j(s', t')$, $s' < j < t'$. Then the $\bar{y}_j$, $s' \leq j \leq t'$, are collinear and it follows from (2.48) that $f_k \geq \bar{y}_k - h$ and $f_{k^+} \geq \bar{y}_{k^+} - h$. Therefore, since $\bar{y}_k$, $\bar{y}_{j^*}$, and $\bar{y}_{k^+}$ are collinear,
\[ \bar{y}_{j^*} - h \leq f_{j^*}(k, k^+). \]  

(2.53)

It also follows from (2.48) that
\[ f_{j^*} \leq \bar{y}_{j^*} + h. \]  

(2.54)

Addition of (2.53) and (2.54) and use of (2.52) then gives (2.50).
Now consider (2.51). The easiest case is when $s_\alpha > s'$. Then $g^+ = g_{s'}$, where $i = s'^+(I)$. Let the $\bar{g}_j$, $s' \leq j \leq t'$, be defined as above. Then $f_{s'} = \bar{g}_{s'} - \bar{h}$ and it follows from (2.48) that $f_i \geq \bar{g}_i - \bar{h}$. Then $g_{s'} \geq g_{s'}(y) = \bar{g}$. Similarly, when $t_\alpha < t'$, then $\bar{g} \leq g^-$. The next easiest case is when $s' = s_\alpha$ and $t_\alpha = t'$. In this case,

$$g^+ = g^- = \frac{f_{t'} - f_{s'} - 2h}{x_{t'} - x_{s'}},$$

and it follows from (2.50) that this quantity is not less than $\bar{g}$, as required to establish (2.51).

It remains to resolve the two similar cases where $s' < s_\alpha$ and $t_\alpha = t'$, and where $s' = s_\alpha$ and $t_\alpha < t'$. It suffices to consider the former case and establish that $\bar{g} \leq g^-$. The procedure closejoin will add one or more indices $s'^+, \ldots, i, s$ to $I$. (The gradient $g^-$ is now the new join gradient, and the result required is analogous to Lemma 4, but this lemma cannot be used directly unless $h > h'$, which may not be the case.) The gradients defined by the successive elements of $I$ from $s'^+$ to $s$ will be monotonically increasing from $g_{s'}$, and it has already been shown that $\bar{g} \leq g_{s'}$, so it follows that

$$\bar{g} \leq g_{s'}. \quad (2.56)$$

Since $t_\alpha = t'$, the test that allows Step 4 of Algorithm 2 to adjoin $s$ to $I$ will be the inequality

$$f_{t'}(i, s) \leq f_{t'} - 2h, \quad (2.57)$$

and this inequality is equivalent to

$$g_{s'} \leq g, \quad (2.58)$$

where $g$ is the new join gradient given by $(x_{t'} - x_s)g = f_{t'} - f_s - 2h$. But $g = g^-$, so from (2.56), $\bar{g} \leq g$ and (2.51) follows. \hfill \square

The proof of the effectiveness of Algorithm 4 and also of its generalization in the next section will require an important corollary of Lemma 5. When new sections are added on either side of an existing section, the convexity at the point where they are joined increases away from zero. Thus when Steps 2 to 6 of Algorithm 4 are applied, the constraints $c_{k-1}$, $c_{j',-1}$, and $c_{k+1}$ increase away from zero, so that no more than two sign changes can be created in the second divided differences. More precisely, when defined, these constraints satisfy

$$c_{k-1}(y^2) \geq c_{k-1}(y^0) \geq 0, \quad (2.59)$$
$$c_{j',-1}(y^2) \leq c_{j',-1}(y^0) \leq 0, \quad (2.60)$$
$$c_{k+1}(y^2) \geq c_{k+1}(y^0) \geq 0. \quad (2.61)$$

The effectiveness of Algorithm 4 can now be established.
2. THE ALGORITHM

**Theorem 4** Algorithm 4 produces a real number \( h \geq 0 \), an index set \( I \), a piece set \( S \) of \( I \), and a vector \( y^2 = y(S) \) well defined by (2.46), (2.47) and (2.43) such that \( h = h(S) = F(y^2) = \inf F(Y_2) \) and \( y^2 \in Y_2 \).

**Proof** If the Algorithm stops in Step 1, then \( h = h^+(1,n) = 0 \), \( I = I^+(1,n) \), and \( S = \{ n \} \). Then \( h(S) = 0 \). It follows from Theorem 2 that \( f \in Y_0^+ \subset Y_2 \). Equation (2.46) will set \( P_1 = [1,n] \) and (2.41) and (2.42) will set \( y^2 = f \). The theorem is then immediately established.

Otherwise the quantities \( j^*, k \), and \( k^+ \) are well-defined and satisfy the inequalities \( 1 \leq k < j^* < k^+ \leq n \). (Each equality is possible, for example when \( f \in Y_1^+ \).) It follows from Lemma 1 that at this point,

\[
I = I^+(1,k) \cup I^+(k^+,n). \tag{2.62}
\]

Step 2 is then entered. It inserts \( j^* \) into \( I \), increases \( S \) to \( \hat{S} = \{ k, j^*, n \} \), and calculates \( h = \max(h^+(1,k), h^+(k^+,n)) \). Note that \( j^*, k \), and \( k^+ \) are now consecutive elements of \( I \).

Steps 3 and 4 are then executed, calling closejoin in the ranges \([k, j^*]\) and \([j^*, k^+]\), in general increasing \( s_1 \), \( s_2 \), and \( h \), and adding new elements to \( I \). The new elements of \( S \) always satisfy the inequalities \( k \leq s_1 \leq j^* \leq s_2 \leq k^+ \), and \( h \) cannot decrease, so \( \hat{h} \leq h \). Note that \( s_2 \) can be increased to \( n \), for example when \( f \in Y_1^+ \).

Now consider the situation when the Algorithm stops. If \( h^{(1)} = 0 \), the Algorithm either stops in Step 4 when \( h = 0 \), or alternatively jumps straight from Step 5 to Step 6 re-calling closejoin(1) with \( h > 0 \). It follows from the properties of Algorithm 3 that when closejoin is called with \( h > 0 \) it cannot increase \( s \) to \( t \). Therefore when the algorithm stops, if \( h > 0 \) then \( s_1 < j^* \) and \( s_2 < k^+ \). Now define \( t_1 \) and \( t_2 \) as for (2.46). Then \( y^2 \) is well-defined in all cases. It must now be shown that it is feasible and optimal.

The first main step is to establish (2.40), i.e.

\[
I = I^+(1,s_1) \cup I^-(t_1,s_2) \cup I^+(t_2,n). \tag{2.63}
\]

First consider the range \([1,s_1]\). Since \( k \leq s_1 \), \([1,s_1] = [1,k] \cup [k,s_1] \). Let \( I_1 = I \cap [1,s_1] \). It follows from (2.62) and Theorem 3 that \( I_1 = I^+(1,k) \cup I^+(k,s_1) \). It is trivial that \( I_1 = I^+(1,s_1) \) unless \( 1 < k < s_1 \). In this case there will exist neighbouring indices \( k^- \) and \( i \) of \( k \) in \( I \) such that \( 1 \leq k^- < k < i \leq s_1 \leq j^* \). Let \( h_0 = h^+(1,n) \) and \( g_0 = g^{(1)}(\hat{S},h_0) \). Since \( k^- \), \( k \), and \( k^+ \) are neighbours in \( I^+(1,n) \), then \( g_{k^-k} \leq g_{kk^+} = g_0 \). Note that \( g^{(2)}(\hat{S},h_0) = g_0 \). It is now possible to apply Lemma 5 successively with \( \bar{h} = h_0 \). The definition of \( j^* \) allows a first application of the lemma with \( h' = \hat{h} \) \( S' = \hat{S} \) in the range \([k,j^*]\) (i.e. with \( \alpha = 1 \)) to infer that at entry to Step 4, \( h^{(1)} \leq h_0 \). This inequality and
2. THE ALGORITHM

the definition of \( j^* \) then allow a second application of the lemma in the range \([j^*, k^+]\), where also \( \gamma = g_0 \), to infer that \( h \leq h_0 \). If Step 6 is not entered, it follows from the first application of the lemma that \( g_{ki} \geq g_0 \). If closejoin is called again in Step 6, a third application of the lemma may be made, in the range \([k, j^*]\), to yield that in this case also, \( g_{ki} \geq g_0 \). Then \( g_{k-k} \leq g_{ki} \).

Lemma 2 can now be applied to prove that \( I^+(1, k) \cup I^+(k, s_1) = I^+(1, s_1) \).

Thus in all cases

\[
I_1 = I^+(1, s_1). \tag{2.64}
\]

In the same way, if \( I_3 = I \cap [t_2, n] \), then \( I_3 = I^+(t_2, n) \), trivially when \( t_2 = n \) and otherwise by a single application of Lemma 5.

Now consider the range \([t_1, s_2] \). Let \( I_2 = I \cap [t_1, s_2] \). Since \( j^* \in I \), \( t_1 \) can never exceed \( j^* \) and so \([t_1, s_2] = [t_1, j^*] \cup [j^*, s_2] \). In all cases \( I_2 = I^-(t_1, j^*) \cup I^-(j^*, s_2) \), and it is trivial that \( I_2 = I^-(t_1, s_2) \) unless \( t_1 < j^* < s_2 \).

In this case there will exist left and right neighbours of \( j^* \) in \( I \). Denote them by \( j^- \) and \( j^+ \). Then the successive applications of Lemma 5 made above establish that \( g_{j^- j^*} \geq g_0 \) and that \( g_{j^* j^+} \leq g_0 \) so that Lemma 2 again applies to give that \( I_2 = I^-(t_1, s_2) \). Equation \( \boxed{2.63} \) is then established.

The feasibility of \( y^2 \) can now be proved. It is only necessary to examine the four (or possibly fewer) join constraints \( c_{s_2-1}, c_{t_1-1}, c_{s_2-1}, \) and \( c_{t_2-1} \) when \( s_1 < t_1 \) and \( s_2 < t_2 \). First consider the last two constraints. Since \( s_2 < k^+ \), Lemma 4 shows that \( c_{s_2-1}(y^2) \leq 0 \) whenever \( s_2 \) exceeds \( j^* \). When \( s_2 = j^* \), the corollary to Lemma 5 applies to show that \( c_{s_2-1}(y^2) \leq 0 \). Similarly, Lemma 4 when \( t_2 < k^+ \) and the corollary to Lemma 5 when \( t_2 = k^+ \) show that \( c_{t_2-1}(y^2) \geq 0 \) whenever \( t_2 < n \). When \( t_2 = n \) the feasibility of \( y^2 \) will follow automatically from the signs of the other constraints. Now consider the constraints \( c_{s_1-1}, c_{t_1-1}. \) When Step 6 is entered or \( h^{(1)} = h \), the same reasoning applies to show that \( c_{s_1-1}(y^2) \geq 0 \) when \( s_1 > 1 \) and \( c_{t_1-1}(y^2) \geq 0 \) when \( t_1 > 1 \). (If \( f \in Y_0^-, \) the Algorithm will reduce \( t_1 \) to 1.) When Step 6 is not entered and \( h > h^{(1)} \), Lemma 4 cannot be applied, but Lemma 5 and its corollary then show that feasibility is assured unless \( s_1 > k \) or \( s_2^* < j^* \). In these cases all the gradients calculated are well-defined and the test in Step 5 allows the algorithm to stop only when \( y^2 \) is feasible.

It is now necessary to show that \( F(y^2) = h \).

The construction of \( \hat{h} \) and \( h \), the inequality \( \hat{h} \leq h \), and \( \boxed{2.63} \) show that \( h = h(S) \) and \( |y_j - f_j| \leq h \) when \( j \in [s_1] \cup [t_1, s_2] \cup [t_2, n] \), with equality when \( j \in I \). For any other value of \( j \) in the range \([1, n] \), Lemma 3 and the inequality \( h^{(1)} \leq h \) show that \( |y_j - f_j| < h \) whether or not Step 6 is entered. Then \( F(y^2) = h \).

If \( h = 0 \), there is nothing further to prove. Otherwise \( s_\alpha < t_\alpha \), for \( \alpha = 1, 2 \). Then it is possible to redefine \( j^* = j^*(S) \) and to let \( k, k^+ \), and \( \beta \) be uniquely
2. \textit{THE ALGORITHM}

Figure 6: Construction of $Y_5$ approximation

redefined from $j^*$. Then $j^* \in P_3$. Let $K = \{k, j^*, k^+, s_1, t_1, s_2, t_2\}$. Then $K$ cannot have fewer than five elements, even if $t_1 = s_2$. Assume first that $\beta = 1$. It follows by the same argument used in the proof of Theorem 3 that $c_{kj^*k^+}(y^2) = 0$, $c_{j^*s_1t_1}(y^2) \geq 0$, and $c_{s_1t_1t_2}(y^2) \leq 0$. Then if $F(v) \leq h$, $c_{kj^*k^+}(v) < 0$, $c_{j^*s_1t_1}(v) > 0$, and $c_{s_1t_1t_2}(v) < 0$, and therefore the consecutive divided differences of $v$ with indices in the subset $K$ must change sign twice starting with a negative sign, so that by Theorem 1, $v \notin Y_2$. If $\beta = 2$ or $\beta = 3$, the argument is similar. \hfill \Box

2.4 The general case

The algorithm to be described constructs a best approximation to $f$ over $Y_q$ from $q+1$ alternately convex and concave pieces joined where necessary by up to $q$ straight line joins. These pieces are built up recursively from the pieces of a similar best approximation over $Y_{q-2}$ by essentially the same method used in the previous section when $q = 2$. All the sections of the $Y_{q-2}$ approximation remain in place except one determining the value of $\inf F(Y_{q-2})$ which is deleted and replaced with a new piece of opposite convexity to the piece containing this section, and two new joins. Starting with the value $h$ of a best approximation over $Y_q$ determined by the remaining sections, the procedure \textit{closejoin} is called in each join. After this has been done, if $h$ has increased, the resulting join constraints are checked and if necessary the calculation in each join is repeated with the new value of $h$. The construction of $y$ is illustrated in Figure 6.

This method has the disadvantage that calculations sometimes have to be performed twice in each join, but each of the two sets of join calculations can be performed in parallel. The procedure \textit{closejoin} is a generalization of Algorithm 2 which is a modification of the method presented in Cullinan & Powell (1982). That method avoids having to repeat itself by having an upper bound on $h$ available throughout, but because of this does not admit of as much of its calculations being performed in parallel. It is therefore
believed that the algorithm to be presented will often be more efficient.

Most of the notation needed for this case has already been developed. In particular, given a piece set \( S \), (2.43) defines the corresponding price \( h(S) \) and when this price is non-zero, (2.44) and (2.45) define the index \( j^* \) giving rise to it and the index \( \beta \) of the piece within which it lies. For any \( h \) the piece \( P_\alpha \) is defined as \([t_{\alpha-1}, s_\alpha]\), where \( t_\alpha = s_\alpha^+ (I) \) when \( h > 0 \) and \( t_\alpha = s_\alpha \) when \( h = 0 \). Since every element of \( I \) lies in \( P_\alpha \), for some \( \alpha \), the vector \( y(S) \) can be defined by

\[
y_i = f_i + (-)^{\alpha-1} h \quad \text{when } i \in I \cap P_\alpha, \quad 1 \leq \alpha \leq q' + 1, \quad (2.65)
y_j = y_j(I) \quad 1 \leq j \leq n. \quad (2.66)
\]

The following algorithm then calculates \( S \), \( I \), and \( h \) from which \( y^q \) is defined by these equations as the final value of \( y(S) \). Each time \( h \) is increased, the new values of \( j^* \) and \( \beta \) are calculated and recorded. To identify joins where a second call of closejoin will always be needed when \( h \) increases from zero, the index \( \gamma \) will denote the lowest index for which \( h > 0 \) after the first call of closejoin(\( \gamma \)). When \( h > 0 \), the join gradients \( g^{(\alpha)}(S, h) \) are defined by (2.47).

**Algorithm 5** To find a best \( Y_q \) approximation.

1. Set \( \overline{q} = q \) modulo 2.
   If \( \overline{q} = 0 \): set \( S := \{n\} \) and use Algorithm 1 to calculate \( I := I^+(1, n) \) and \( h = h^+(1, n) \).
   Otherwise set \( S := \{1, n\} \), \( I := \{1, n\} \), \( h := 0 \) and call closejoin(1).
2. If \( \overline{q} = q \) or \( h = 0 \): stop.
   Otherwise: set \( j := j^*(S) \), \( k := j^-(I) \), insert \( j \) into \( I \), and insert \( j \) and \( k \) into \( S \), increase \( \overline{q} \) by 2 and calculate \( h := h(S) \).
   Set \( h' := h \), \( I' := I \), \( S' := S \) and \( \gamma := 1. \)
3. For \( \alpha = 1 \) to \( q \): apply closejoin(\( \alpha \)), and if \( h \) has increased set \( \overline{\alpha} := \alpha \), and if \( h = 0 \) set \( \gamma := \alpha + 1 \).
   If \( h = h' \): return to Step 2.
   Otherwise: set \( \alpha = 1 \) and go on to Step 4.
4. If \( \alpha < \gamma \) go to Step 5.
   If \( \alpha = \overline{\alpha} \) return to Step 2.
   Calculate \( g = g^{(\alpha)}(S, h) \).
If $s_\alpha > s'_\alpha$ and $(-1)^{\alpha-1}g_{s_\alpha(I)}s_\alpha > (-1)^{\alpha-1}g$: go to Step 5.
Set $t := s^+_\alpha(I')$ and $t^+ := t^+(I')$.
If $t < s'^+_\alpha(I')$ and $(-1)^{\alpha-1}gt_{t^+} > (-1)^{\alpha-1}g$: go to Step 5.
Increase $\alpha$ by 1 and repeat this step.

Step 5. Set $s_\alpha := s'_\alpha$, delete all elements of $I$ lying strictly between $s_\alpha$ and $s^+_\alpha(I')$, and call closejoin($\alpha$).
Increase $\alpha$ by 1 and return to Step 4.

In practice, in order to anticipate rounding errors, the tests whether $h = 0$ should be whether $h \leq 0$. The applications of closejoin in each step could be performed simultaneously. In this case the largest of the ensuing values of the parameter $h$ in Step 3 should be the value of $h$ at entry to Step 4 and $\overline{\pi}$ should be set to $n$. It will be shown in the proof of the following theorem that $h$ is constant during Step 4.

**Theorem 5** Algorithm 5 produces a real number $h \geq 0$, an index set $I$, a piece set $S$ of $I$, and a vector $y^q = y(S)$ well defined by (2.65) and (2.66) such that $h = F(y^q) = \inf F(Y_q)$ and $y^q \in Y_q$.

**Proof** The proof that $y \in Y_q$ and $F(y) = h$ is by induction on $\overline{q}$. Assume that at any entry to Step 2 leading to Step 3, i.e. with $\overline{q} < q$ and $h > 0$, that

\[ I = \bigcup_{\alpha=1}^{\overline{q}+1} I(-)^{\alpha}(t_{\alpha-1}, s_\alpha), \]  

(2.67)

where $t_\alpha = s^+_\alpha(I)$, and that

\[ h = h(S). \]  

(2.68)

The vector $y(S)$ is then well defined. Assume that

\[ F(y(S)) = h \]  

(2.69)

and that

\[ y(S) \in Y_\overline{q}. \]  

(2.70)

It will be deduced from these equations that when the algorithm terminates $y(S) \in Y_q$ and that $h = \inf Y_q$.

It has been shown in Sections 2.1 and 2.2 that (2.67)–(2.69) hold at first entry to Step 2.

Most of the work needed for the proof has already been done in Lemmas 3 to 5. The main task is to examine the way $h$ changes, so as to be able to apply these lemmas. Suppose that Step 2 begins with $h = \overline{h}$, $I = \overline{I}$
and $S = \overline{S}$, and that it increases $\bar{q}$ to $q'$. Step 2 also modifies $S$ from $\overline{S}$ to $S' = \overline{S} \cup \{j,k\}$ and $I$ from $I$ to $I' = I \cup \{j\}$, and recalculates $h$ as $h' = h(S')$ using (2.43). Now by (2.68) and the definition of $j$ in Step 2, $\overline{h} = h^\sigma(t_{\beta-1},s_\beta)$, where $\beta$ is defined in Step 2 by (2.45), $\sigma = (-)^\beta$, and $t_{\beta-1} = s_{\beta-1}^\pm(I)$. By Lemma 1, $I^\sigma(t_{\beta-1},s_\beta) = I^\sigma(t_{\beta-1},k) \cup I^\sigma(k^+,s_\beta)$, and by definition, $h^\sigma(t_{\beta-1},s_\beta) \geq \max(h^\sigma(t_{\beta-1},k),h^\sigma(k^+,s_\beta))$. It follows that

$$h' \leq \overline{h}.$$  \hfill (2.71)

Furthermore, it follows from the definitions of $\overline{h}$ and $\beta$ when $\alpha = \beta,\beta + 1$ and otherwise from (2.67)--(2.69) that

$$(x_j,f_j) \in \Pi^\alpha_S(\overline{h}) \quad \text{for} \quad s^-_\alpha \leq j \leq s^+_\alpha(I').$$  \hfill (2.72)

Step 3 then applies closejoin($\alpha$) in each join. Let $h^{(\alpha)}$ be the value of $h$ after closejoin($\alpha$) is called. Clearly the $h^{(\alpha)}$ are monotonically non-decreasing. Lemma 5 can now be applied successively beginning with (2.71) to show that

$$h^{(\alpha)} \leq \overline{h} \quad \text{for} \quad 1 \leq \alpha \leq q'.$$

Let $t'_{\alpha-1} = s^+_{\alpha-1}(I')$. Then Lemma 5 and its corollary also show that whenever $t'_{\alpha-1} < s'_\alpha < s_\alpha$, the gradients on either side of $f_{s_\alpha'}$ have the correct monotonicity for Lemma 2 to yield that $I^\sigma(t'_{\alpha-1},s'_\alpha) \cup I^\sigma(s'_\alpha,s_\alpha) = I^\sigma(t'_{\alpha-1},s_\alpha)$, and that when $t_{\alpha-1} < t'_{\alpha-1} < s'_\alpha$ that $I^\sigma(t_{\alpha-1},t'_{\alpha-1}) \cup I^\sigma(t'_{\alpha-1},s'_\alpha) = I^\sigma(t_{\alpha-1},s'_\alpha)$, where $\sigma = (-)^{\alpha-1}$. It follows either from this or otherwise trivially that in all cases after closejoin($\alpha$) is called,

$$I \cap P_\alpha = I^\sigma(t_{\alpha-1},s_\alpha).$$

Because $h$ cannot increase further after Step 3, this holds true whether closejoin($\alpha$) is called once only or again in Step 5, so that when Step 2 is next entered,

$$I = \bigcup_{\alpha=1}^{q'+1} I^{(-)^\alpha}(t_{\alpha-1},s_\alpha).$$  \hfill (2.73)

The next step is to establish when Step 2 is next entered with $h > 0$ then $h = h(S)$. Since $\overline{h} > 0$ and $k < j < k^+$ in Step 2, the quantity $h(S)$ to which $h$ is set by Step 2 is well defined by (2.43) with $t_\alpha = s^+_\alpha(I)$. Thus

$$h' = h(S').$$

If Step 5 recalls closejoin($\alpha$) for any $\alpha$, it will not increase $h$ further, so $h$ always attains its final value by the end of Step 3. It must be shown that
the quantity \( h(S) \) is always well defined when Step 2 is next entered. This can only fail to be the case if there is an \( \alpha \) for which \( h^{(\alpha)} = 0 \). In such a case it must hold that \( h' = 0 \) and that \( h \) increased during Step 3. Then \( \pi \) will be set to the index of the call of closejoin in which \( h \) achieved its final positive value, and the parameter \( \gamma \) will be set to the lowest index for which \( h^{(\gamma)} > 0 \). Clearly, then, \( \gamma \leq \pi \) and \( \alpha < \gamma \leq \pi \), so that Step 4 will branch to Step 5, recalling closejoin(\( \alpha \)) with a positive \( h \) so that afterwards \( t_\alpha = s_\alpha^+(I) \). The quantity \( h(S) \) will then be well defined when Step 2 is entered. It follows from the form of Algorithm 2 and [2.73] that whether \( h \) increases from \( h' \) or not, then when Step 2 is next entered,

\[
h = h(S),
\]

(2.74)

The vector \( y = y(S) \) will then be well defined at the next entry to Step 2. The next argument will establish that [2.69] then holds. It follows from [2.73] and [2.74] that it is sufficient to show that for any indices \( j \) such that \( s_\alpha < j < t_\alpha, |y_j(S) - f_j| < h \). This is equivalent to the statement that the graph point \((x_j, y_j(S))\) lies inside the parallelogram \( \Pi^S_\alpha(h) \). It follows from Lemma 3 that immediately after closejoin(\( \alpha \)) is last applied, \((x_j, y_j(S))\) lies strictly within \( \Pi^S_\alpha(h) \). If this call of closejoin comes in Step 5, \( h \) has already attained its value at next entry to Step 2 and the result is immediately established. If, on the other hand, this call of closejoin comes in Step 3, then \((x_j, y_j(S))\) lies inside \( \Pi^S_\alpha(h^{(\alpha)}) \), and \( h^{(\alpha)} \leq h \). A simple calculation shows that if \( a < b \) then \( \Pi^S_\alpha(a) \subset \Pi^S_\alpha(b) \). The result then follows. Thus at next entry to Step 2,

\[
F(y(S)) = h.
\]

(2.75)

The next argument will re-establish [2.70] at that point. It follows from [2.67] that it is only necessary to establish that the join constraints \( c_{s_\alpha - 1} \) and \( c_{t_\alpha - 1} \) of \( y \) have the correct signs when \( 2 \leq s_\alpha < t_\alpha \leq n - 1 \), i.e. when \( h > 0 \).

When \( 1 < s_\alpha = s'_\alpha \), [2.70] and the corollary to Lemma 5 imply that \( c_{s'_\alpha - 1}(y) \) had the correct sign at the last exit from Step 2 and has not now moved closer to zero. The same is true of \( c_{t_\alpha - 1} \) when \( t_\alpha = t'_\alpha < n \). When \( s_\alpha > s'_\alpha \), Lemma 4 applies provided that \( h \) has not increased after the last call of closejoin(\( \alpha \)) to yield that \((-1)^{\alpha-1}c_{s_\alpha - 1} \geq 0 \) and that when \( 1 < t_\alpha < t'_\alpha \) that \((-1)^{\alpha-1}c_{t_\alpha - 1} \leq 0 \). If closejoin(\( \alpha \)) is only called once and \( h \) increases subsequently, it must be shown that the test in Step 4 is adequate to ensure feasibility. In this case Step 4 will certainly be entered (because \( h > h' \)) and Step 3 will set \( \pi \) and \( \gamma \) so that \( \gamma \leq \alpha < \pi \), so that \( h^{(\alpha)} > 0 \). It follows that \( s_\alpha < t_\alpha \) and so \( g \) is well defined and the correct sign of \( c_{s_\alpha - 1} \) is assured by the test in Step 5. The case \( t_\alpha < t'_\alpha \) is similar. Thus at next entry to Step 2, even when \( h = 0 \),

\[
y \in Y_{q'}. \]

(2.76)
Thus (2.67)–(2.70) are established by induction. It follows that when the algorithm terminates with a number \( h \) and the pieces from which the vector \( y \) is constructed, \( y \in Y_q \subset Y_q \) and \( F(y) = h \).

The proof of optimality is similar to that in Theorem 4. If \( h = 0 \) there is nothing to prove. Otherwise, after the algorithm has terminated, let \( s_{\alpha}, t_{\alpha}, 1 \leq \alpha \leq q + 2 \), be defined as the join points that would next have been constructed in Step 2 if the algorithm had not terminated (i.e. by adding \( k, j^* \), and \( k^+ \) to the existing set of join points and reindexing.) It follows as in the proof of Theorem 3 that because \( h > 0 \), \( s_{\alpha} < t_{\alpha} \) for all \( \alpha \).

Now define \( K = \{ s_{\alpha}, t_{\alpha} : 1 \leq \alpha \leq q + 2 \} \). Because the piece \([t_{\beta}, s_{\beta+1}]\) has only one element and any of the other pieces can have only one element, this set \( K \) can have from \( q + 3 \) to \( 2q + 3 \) elements. It has to be shown that the consecutive divided differences with indices in \( K \) of any vector \( v \in \mathbb{R}^n \) giving a lower value of \( F \) than \( h \) change sign \( q \) times starting with a negative sign, so that by Theorem 1, \( v \notin Y_q \).

It follows from the construction of \( y \) and (2.70) that \((-1)^{a-1}c_{i-1}y \leq 0 \) for \( s_{\alpha} \leq i \leq t_{\alpha+1} \) and so, from the corollary to Theorem 1, that for any \( \alpha \) in the range \( 1 \leq \alpha \leq q + 2 \), \((-1)^{a-1}c_{s(\alpha),t(\alpha),s(\alpha+1)}(y) \leq 0 \) and \((-1)^{a-1}c_{s(\alpha),t(\alpha+1),s(\alpha+1)}(y) \leq 0 \). If \( t_\alpha = s_{\alpha+1} \), it follows immediately from (2.65) that if \( F(v) < h \)

\[
(-1)^{a-1}c_{s(\alpha),t(\alpha),t(\alpha+1)}(v) < 0.
\]

Otherwise when \( t_\alpha < s_{\alpha+1} \), it follows from (2.65) that if \( F(v) < h \), then \((-1)^{a-1}c_{s(\alpha),t(\alpha),t(\alpha+1)}(v) < 0 \) and \((-1)^{a-1}c_{s(\alpha),t(\alpha+1),t(\alpha+1)}(v) < 0 \) and hence, from the corollary to Theorem 1, that the consecutive divided differences with indices in \( K \) satisfy

\[
(-1)^{a-1}c_{s(\alpha),t(\alpha),s(\alpha+1)}(v) < 0 \quad \text{or} \quad (-1)^{a-1}c_{t(\alpha),s(\alpha+1),t(\alpha+1)}(v) < 0.
\]

Therefore the divided differences of \( v \) with indices in \( K \) have at least \( q \) sign changes starting with a negative sign, and so if \( v \) is any vector for which \( F(v) < h \), then \( v \notin Y_q \).

3. Conclusions

This section will describe the results of some tests of the data smoothing method developed in the previous section. These tests were conducted by contaminating sets of values of a known function with errors, applying the method, and then comparing the results with the exact function values. If \( g \)
is the vector of exact function values, a simple measure of the effectiveness of the method can be obtained from the quantity

$$P_p = \left(1 - \frac{\| y - g \|_p}{\| f - g \|_p}\right),$$

obtained from the $\ell_p$ norm.

Algorithm 5 was trivially extended to find a best approximation over $Y_{\leq q}$. It was coded in PASCAL and run on a SUN computer. The errors added to exact function values were either truncation or rounding errors, or uniformly distributed random errors in the interval $[-\epsilon, \epsilon]$. For each set of data the values of $P_\infty$ and $P_2$ were calculated.

As a preliminary test, equally spaced values of the zero function on $[-5, 5]$ were contaminated with uniformly distributed errors with $\epsilon = 0.1$. The results are shown in Table 1.

| $n$  | $P_\infty$ | $P_2$ |
|------|-----------|-------|
| 5001 | -19.37    | 94.51 |
| 501  | -68.97    | 76.25 |

Table 1: The zero function

The difference between $y_j$ and zero was of the order of $10^{-4}$ for most of the range, but near the ends it rose to $10^{-1}$, accounting for the relatively high value $\| y - g \|_2 = 0.3098$ when $n = 501$. High end errors were frequent and so the value of $P_\infty$ was not usually a reliable indicator of the efficiency of the method.

Two main types of data were then considered: undulating data and peaky data, because it is these types of data that can be hard to smooth using divided differences unless sign changes are allowed. The first main category of data were obtained from equally spaced values of the function $\sin \pi x$ on the interval $[-2, 2]$, and the second from equally spaced values of the normal distribution function $N_s(x) = (2\pi x)^{-\frac{1}{2}} \exp(-x^2/2s^2)$ with $s = 0.8$, on the interval $[-5, 5]$. These same functions were previously used to test the $\ell_2$ data smoothing method of Cullinan(1990) which did not allow sign changes, where it was shown that the sine data were possible to treat well but the peaked data did not give very good results.

Many of the results looked very acceptable when graphed, even when $P_2$ was only moderate. The results of Table 1 have already indicated that it is quite difficult for $P_2$ to approach 100. This is often because of end errors, as there, but there is also another phenomenon that can move smoothed points
further away from the underlying function values. The method raises convex pieces and lowers concave pieces, and this often results in points near an extremum of the underlying function being pulled away from it, for example if there is a large error on the low side of a maximum.

Different sets of random errors with the same $\epsilon$ can give very different values of both $P_\infty$ and $P_2$, particularly when the spacing between points is not very small, and in fact it was found that reducing the spacing beyond a certain point can make a great difference.

The method coped quite well with the sine data and was very good with the peaked data, and a great improvement on the method in Cullinan(1990), managing to model both the peak and the flat tail well. The results of one run are shown in Figure 7.

![Figure 7: Peaked Data with $n = 1$, $q = 2$, $\epsilon = 0.01$, $P_2 = 45.8$](image)

The method is very economical, and can give very good results, particularly when the data are close together. Thus it seems a good candidate for large densely packed data, even when the errors are relatively large. BIBLIOGRAPHY

References

[1] Cullinan, M. P. 1986 Data smoothing by adjustment of divided differences. Ph.D. dissertation, University of Cambridge.
List of Figures

1 Consecutive elements $k^-, k, k^+$ of $I = I^+(1, n)$ and the construction of $f_j(I)$ .................................................. 7
2 Construction of best convex-concave approximation .......... 10
3 The parallelogram $\Pi(h)$ ..................................................... 12
4 When $s$ does not increase, $t$ must decrease ................. 14
5 Construction of $Y_2$ approximation ................................. 18
6 Construction of $Y_3$ approximation ................................. 25
7 Peaked Data with $n = 1, q = 2, \epsilon = 0.01, P_2 = 45.8 ....... 32