JOURNAL OF APPROXIMATION THEORY **42**, 52–63 (1984)

The Complexity of Interpolating Given Data in Three Space with a Convex Function of Two Variables

DAVID S. SCOTT

Computer Sciences Department, University of Texas at Austin, Austin, Texas 78712, U.S.A.

Communicated by Oved Shisha

Received September 26, 1983

Three questions concerning the interpolation of a set (x_i, y_i, z_i), $i = 1,..., N$, in $\mathbf{R}^3$ by a convex function of two variables ($z = f(x, y)$) are examined. A given set of N points can be interpolated by a convex function if and only if it can be interpolated by a convex piecewise planar function. Every possible piecewise planar interpolation of the data is determined by a triangulation of the (x, y) points in the plane. An algorithm is presented which builds up an acceptable triangulation by sequentially adding points. The algorithm either terminates as soon as it discovers that the interpolation is impossible or terminates with the desired triangulation. Numerical experiments are presented which indicate the effectiveness of the algorithm. Suppose that the points in the plane, (x_i, y_i), $i = 1,..., N$, are fixed and it is desired to minimize a function $f(z_1, z_2,..., z_N)$ subject to the constraint that the triples (x_i, y_i, z_i) can be interpolated by a convex function. Convexity can be represented by a set of linear inequality constraints among the z's but many of these constraints may be redundant. For efficiency it is important to reduce the list to the independant constraints and some minimization algorithms actually require independent constraints. An efficient algorithm for generating the set of independent linear inequalities is given. Finally it is shown that the number of independent constraints depends on the location of the (x, y) points and varies from zero to $O(N^3)$. It is conjectured that the expected number of independent constraints for N points chosen randomly from a uniform distribution on a square is $O(N^2 \log N)$. Both theoretical and numerical justification for the conjecture are given. Finally it is shown that there are $O(N^2)$ independent constraints when the points are arranged in a square (or triangular) lattice.

1. INTRODUCTION

This paper examines three problems concerning the interpolation of a finite set of points (x_i, y_i, z_i) in $\mathbf{R}^3$ with a convex function $z = f(x, y)$. The first problem is to decide whether a convex interpolation is possible for a given set of data. The second problem is to determine the minimal set of

0021-9045/84 $3.00

independent, linear inequalities that represents the convexity constraint when the (x, y) values are fixed but the z values are variable. Finally, the number of such linear inequalities as a function of the location of the (x, y) points is investigated.

For comparison, the corresponding problems for points in the plane $(z = f(x))$ are also described.

2. Fixed Data

Given fixed data with one independent variable (i.e., (x_i, z_i), $i = 1, 2, \ldots, N$) then a convex interpolating function (**Interpolant**) $z = f(x)$ exists if and only if all the points lie on the lower boundary of their convex hull. This lower boundary is a piecewise linear function with the line segments defined between data points, so a convex interpolant exists if and only if a convex piecewise linear interpolant exists. Since there is only one piecewise linear interpolant whose line segments are defined between pairs of data points, it is only necessary to determine whether this function is convex. In particular it is only necessary to verify that when two line segments meet at a point they satisfy the condition that the slope of the left line segment is less than the slope of the right line segment.

If there are two independent variables the situation is more complicated. As before, a convex interpolant exists if and only if all the points lie on the lower boundary of their convex hull. This lower boundary is a piecewise planar function whose faces are polygons (usually triangles) with data points for vertices. Thus if there is any convex interpolating function there must be a convex piecewise planar one with vertices at the data. Since the convex hull of a set of points is unique the corresponding convex piecewise planar function is also unique. Unfortunately there may be many piecewise planar interpolating functions with vertices at the data.

Removing the z values from the given data yields a set of points in the plane. Associated with each triangulation of these (x, y) points is a unique piecewise planar function with vertices at the (x, y, z) data. (On each triangle the function is defined as the piecewise planar interpolant of the data values associated with the vertices of the triangle.) Provided all the faces of the piecewise planar function are triangles, then the triangulation associated with the function is also unique. In the special case of nontriangular faces, more than one triangulation is associated with the same function. Thus the problem reduces to determining whether an appropriate triangulation of the (x, y) data exists. We will now describe an algorithm for finding the convex piecewise planar interpolant (when it exists) by finding a corresponding triangulation.

DEFINITION 1. An edge of a triangulation of the (x, y) points is called concave (convex, flat) with respect to the (x, y, z) data, if the edge is an interior edge and the angle between the two planes of the associated piecewise planar interpolant which meet along the edge is concave (convex, flat).

Every interior edge of the desired triangulation (if it exists) is convex (or flat). The basic idea of the algorithm is to start with any triangulation of the (x, y) data and repeatedly try to replace concave edges until the desired triangulation is found. Associated with any concave edge in the current triangulation is a quadrilateral made up of the two triangles based on the edge. The given edge is one of the diagonals of the quadrilateral. If this quadrilateral is convex then the concave edge can be replaced in the triangulation by the other diagonal. The new edge will always be convex so the new triangulation has fewer concave edges than the old one. If the quadrilateral itself contains reflex or straight angle then the desired triangulation does not exist.

For efficiency grounds it is important to impose some order on the concave edges. The easiest way to do this is to order the (x, y) points in some manner and then recursively build up an acceptable triangulation as the points are sequentially included. This is the same approach used by Renka [2] to compute the Thiessen triangulation of a set of points in the plane. As each point is added it is included in the triangulation by adding some edges. If any of these edges is concave then a convex interpolation is impossible (since the quadrilaterals associated with added edges always have a reflex angle). Otherwise all the quadrilaterals which include the new point are checked to see if their diagonals are concave or convex. Concave edges are replaced (if possible) and this process is iterated untill all the edges are convex or the triangulation is known to be impossible. If all the edges are convex then the next point is added to the triangulation.

Only a few modifications were needed to make Renka's software solve the new problem. It was necessary to check the edges added to the triangulation as a new point was added to see if they were convex. It was also necessary to change the edge swap test so that it checked whether the edge was convex or concave. As was shown by Renka, the cost of computing the desired triangulation depends strongly on the order in which the points are added to the triangulation. For Renka's problem it was best to order the points by increasing x value. For this problem it seems to be better to order them by increasing z value. Tables I–IV show the number of edge swaps needed to find the desired triangulation. The $N(x, y)$ points were generated randomly from a uniform distribution on the unit square. The z values were generated as function values of the (x, y) points. All the functions used were convex so that the desired triangulation always existed. (Nonconvex data would simply

TABLE I

$f(x, y) = (x - 0.3)^2 + (y - 0.4)^2$

	Unsorted		Sorted	
N	Swaps	σ	Swaps	σ
10	6.0	.81	2.7	.47
20	22.0	2.82	17.3	1.69
50	98.3	2.49	56.3	2.49
100	211.7	8.01	159.0	3.27
200	489.3	12.50	387.3	6.13
500	1363.3	14.26	1213.3	20.29

TABLE II

$f(x, y) = (x - 0.3)^4 + (y - 0.4)^4$

	Unsorted		Sorted	
N	Swaps	σ	Swaps	σ
10	6.7	.94	2.7	1.70
20	23.0	1.63	11.3	4.19
50	101.3	2.35	39.0	2.45
100	227.0	9.93	106.0	7.07
200	506.0	8.29	281.7	8.58
500	1363.3	23.44	918.7	16.13

TABLE III

$f(x, y) = (x - 0.3)^2$

	Unsorted		Sorted	
N	Swaps	σ	Swaps	σ
10	7.0	2.94	1.0	.82
20	21.7	8.06	12.3	4.03
50	91.3	7.85	30.7	3.68
100	235.0	16.31	55.3	6.13
200	495.3	16.03	122.7	4.99
500	1355.3	55.53	326.0	6.53

TABLE IV

$f(x, y) = (x - 0.3)^4$

	Unsorted		Sorted	
N	Swaps	σ	Swaps	σ
10	7.3	3.40	1.0	0.00
20	24.7	8.73	9.7	2.87
50	88.3	4.50	24.0	3.74
100	235.3	16.82	48.0	5.35
200	498.7	15.28	113.7	4.78
500	1351.7	57.97	308.3	5.44

terminate early.) The effect of presorting the (x, y) points by increasing z value is also displayed. For each function and each value of N three repetitions were performed and the average number of edge swaps and the standard deviation of the three values were computed.

The percentage savings in edge swaps achieved by presorting the points ranged from 11% to a factor of 7 (86%). The number of edge swaps needed to find the desired triangulation is an increasing function of N. The number of edge swaps needed per added node is either a constant or a slowly growing function of N for the cases tested. The total cost is no worse than $O(N \log N)$, which is the cost of sorting the data. In the worst case it is possible to arrange the data so that every possible edge is included in the triangulation at some time during the algorithm even when the data are presorted. Thus the worst case has $O(N^2)$ edge swaps.

3. Fixed x's and y's

Suppose it is desired to minimize a function $f(z_1, z_2, \ldots, z_N)$ subject to the constraint that the triples (x_i, y_i, z_i), $i = 1, 2, \ldots, N$, can be interpolated by a convex function, where the (x, y) values are fixed. Convexity is equivalent to a set of linear inequalities among the z's (with coefficients in the x's and y's). In the one dimensional case ($z = f(x)$) every pair of x's which contains a third x between them represents a convexity constraint on the z values. If

$$x_i < x_k < x_j$$

then

$$(x_j - x_i)^*(z_k - z_i) \leqslant (z_j - z_i)^*(x_k - x_i).$$

There are $O(N^3)$ constraints, one for each triple of x's. Many of these

constraints are not independent. A minimal constraint is represented by a pair of x's which contains exactly one x in between (i.e., three consecutive x's). The minimal constraints form a complete and independent set of constraints in that all z values which satisfy the minimal constraints can be interpolated by a convex function and for each minimal constraint there is a set of z's which violates only that constraint. There are always $O(N)$ minimal constraints. On efficiency grounds alone it is important to restrict consideration to the minimal constraints and some constrained minimization algorithms require independent constraints.

In two dimensions ($z = f(x, y)$) any three (x, y) points which contain a fourth point in their convex hull represents a linear inequality constraint. There may be as many as $O(N^4)$ constraints. Most of these are redundant too. The minimal inequalities are represented as either triples which contain exactly one point inside (but not on the edge) of their convex hull (which will be called **minimal triangles**) or pairs of points which contain exactly one point in between (which will be called **minimal line segments**). Unlike the one dimensional case it is not easy to determine the minimal inequalities represented by a given set of (x, y) points. The following algorithm generates all the minimal constraints. It sequentially checks through all pairs of points first to see if they are the endpoints of a minimal line segment. If not, it then finds all the minimal triangles which are based on the points. To prevent counting triangles three times the third point of the triangle must have an x value between the x values of the given pair of points. At one point the algorithm uses a special kind of sorting technique called **insertion sort**. In an insertion sort a sequence of records is sorted by sequentially inserting the new record in the proper spot so that the records that have already been processed are correctly sorted.

Assume that the (x, y) points are sorted by increasing x value. If two points have the same x value then they are sorted by increasing y value. Define p_i to be the point (x_i, y_i).

Algorithm to Determine All Minimal Constraints

For $i = 1, N - 2$ do

  For $j = i + 2, N$ do

    Let L be the line segment joining p_i and p_j.

    Separate the points $p_k, k = i + 1, j - 1$ into three subsets:

      A the set of points above the line segment L.

      O the set of points on L.

      B the set of points below L.

    If there is more than one point in O then there are no minimal constraints based on p_i and p_j.

If there is exactly one point in O then p_i, p_j,
and that point form a minimal line segment and there
are no minimal triangles based on these points.

If there are no points in O then

For each point p in A and B define $l(p)$ to be the acute
angle between the line segments $p_i p$ and L and
similarly define $r(p)$ to be the acute angle between
the line segments $p_j p$ and L.

For the sets A and B separately

Sort the points by increasing l value. Two points with
the same l value should be sorted by increasing r value.

Do an insertion sort on the sorted points to resort the
points by increasing r value. Two points with the same
r value should be sorted by increasing l value.

Any point inserted in the second slot during the insertion
sort forms a triangle with p_i and p_j which contains
only the point currently in the first slot. Unless this
fourth point is on an edge, these points form a minimal
triangle.

As a point is inserted during the sort, points earlier in the list must have both a smaller l value and a smaller r value than the point being inserted and no other points can have both a smaller l and r value. A given point p_k is inside the triangle $p_i p_j p$ precisely if both $l(p_k)$ and $r(p_k)$ are smaller than $l(p)$ and $r(p)$ (respectively). This establishes the correctness of the algorithm. Of course in practice it is not necessary to perform the entire insertion sort. It is only necessary to keep track of the first two slots.

4. Variable (x, y) Points

The number of minimal convexity constraints for a fixed set of $N(x, y)$ points can be determined by the algorithm given in the previous section. The actual number of such constraints can vary enormously depending on the location of the (x, y) points. The minimum possible number is zero, which is obtained by any set of (x, y) points all of whose points lie outside the convex hull of the remaining points. This happens, for example, if all the points lie on a circle. The maximum possible number is $O(N^3)$ since that is the order of the set of all possible triangles. $O(N^3)$ is actually obtained when $N-1$ points are spread uniformly on a circle and the Nth point is placed at the center. One fourth of all the triangles with vertices on the circle contain the center point.

What is the expected number of minimal constraints if the (x, y) points are chosen randomly from a uniform distribution on the unit square? The probability of three randomly chosen points lying on a line is zero and so the expected number of minimal line segments is zero. The conjectured number of minimal triangles is $O(N^2 \log N)$. This comes from analyzing how many minimal triangles would be found by the algorithm given in the previous section. If the number of points in the upper set A is m and if each possible ordering of the r values before the insertion sort is equally likely then the expected number of insertions into the second element of array is

$$1/2 + 1/3 + 1/4 + \cdots + 1/m = \log m + d$$

where d is less than 1.

For a given pair of points p_i and p_j the number of points in between is $q = j - i - 1$, some of which are in the upper set and some of which are in the lower set. If a is the number of points in A and b is the number of points in the set B, then $a + b = q$ and

$$c = 1/2 + 1/3 + \cdots + 1/a + 1/2 + 1/3 + \cdots + 1/b$$

is the expected number of minimal triangles (the contribution from a (or b) is taken to be zero when $a < 2$ (or $b < 2$)). It can be easily be verified that c is minimized (with value slightly less than $\log q$) when $a = q$ (or $b = q$). c is maximized (with value less than $2 \log q$) when $a = b$. So the expected number of minimal constraints is bounded by

$$\sum_{i=1,N-1} 2i \log (N - i).$$

TABLE V

Number of Minimal Constraints as a Function of N

NREP	*N*	*AVG*	$AVG/N^2 \log N$
20	15	97.10	0.159360
20	20	230.35	0.192232
15	25	409.27	0.203433
15	30	670.73	0.219117
10	35	994.00	0.228228
10	40	1364.40	0.231168
5	45	1852.00	0.240255
5	50	2360.80	0.241389
5	55	2970.80	0.245071
5	70	5104.60	0.245206
4	100	11486.00	0.249415

This in turn can be bounded by an integral which leads to the result that the expected number of minimal constraints is $O(N^2 \log N)$.

The entire analysis is rigorous except the assumption that the orderings generated by the l values and the r values are independent. This is already false in the case of two points in the A set. However, it seems that the expected correlation is not strong enough to change more than the coefficient of the bound.

Some numerical experiments using the above algorithm were performed to measure the dependence of the number of minimal triangles as a function of N the number of points. The points were chosen randomly from a uniform distribution on the unit square. The experiment was repeated several times with each value of N. *NREP* is the number of repetitions and *AVG* is the average number of minimal triangles (Table V). The results clearly support the claim that the expected number of minimal constraints is $O(N^2 \log N)$.

5. Square Lattice

A popular choice of points for interpolation is a square lattice of $n^2 = N$ points. Square lattices contain both minimal line segments (three consecutive points on a line) and minimal triangles (triangles with exactly one point inside–not on an edge). In this section it will be shown that the number of minimal lines segments and the number of minimal triangles are both $O(N^2)$. The following definitions and lemma will be useful in proving the theorem.

Definition 2. A line segment connecting two lattice points is **simple** if there are no other lattice points on the line segment.

Definition 3. The **central sublattice** of an n by n square lattice is the central $n/3$ by $n/3$ sublattice.

Lemma 4. *Given a square lattice which contains $N = n^2$ points, the number of simple line segments is $O(N^2)$.*

To prove this lemma we will identify the square lattice with the points in $\mathbf{R}^2$ which have integer coordinates between 0 and $n - 1$.

Proof. A point (x, y) with integer coordinates forms a simple line segment with the origin, $(0, 0)$, if and only if x and y are relatively prime. For a given x the number of y's which are relatively prime to x is proportional to $\phi(x)/x$, where ϕ is the Euler phi function. The proportionality constant is n, the number of possible y values. For a certain sequence of x's this ratio becomes arbitrarily small. However, the number m of simple line segments based at the origin is proportional to the average value of $\phi(x)/x$

for x's between 1 and $n-1$. This is known by number theorists to be $O(1)$ (with coefficient $6/\pi^2$, see [1, p. 62]). The proportionality constant is N, the number of lattice points. Thus the origin is one end of $O(N)$ simple line segments. Every lattice point is at one corner of a square sublattice of size at least $n/2$. Thus every lattice point is at one end of $O(N)$ simple line segments. Since there are $O(N)$ lattice points, there are $O(N^2)$ simple line segments in the lattice.

THEOREM 5. *Given a square lattice with $N = n^2$ points then both the number of minimal line segments and the number of minimal triangles is $O(N^2)$.*

Proof. The maximum possible number of minimal line segments is the same as the number of pairs of lattice points, which is $O(N^2)$. So to prove the first part of the theorem it is only necessary to show that there are at least $O(N^2)$ minimal line segments. Applying the lemma to the central sublattice shows that there are $O(N^2)$ simple line segments in the central sublattice. The line containing any of these simple line segments contains at least four lattice points and hence at least two minimal line segments. This proves the first half of the theorem.

The second half of the theorem will be proved by associating every minimal triangle with a simple line segment, by showing that every simple line segment is associated with at most two minimal triangles, and by showing that there are two minimal triangles associated with most simple line segments in the central sublattice.

Three simple line segments can be associated with every minimal triangle, namely the line segments between the vertices of the triangle and the interior point. Associate the triangle with the longest of these simple line segments. (It is actually impossible for a minimal triangle not to have a unique longest line segment but even if it were possible it would only effect the count by a constant factor.) It will now be shown that for any given simple line segment there are at most two minimal triangles associated with it.

Let AD be a simple line segment and let ABC be a triangle which contains only the lattice point D. In particular the triangle ABD has no lattice points

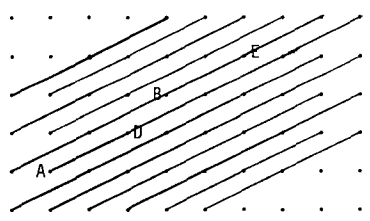

FIG. 1. Points A, B, D, and E with parallel lines.

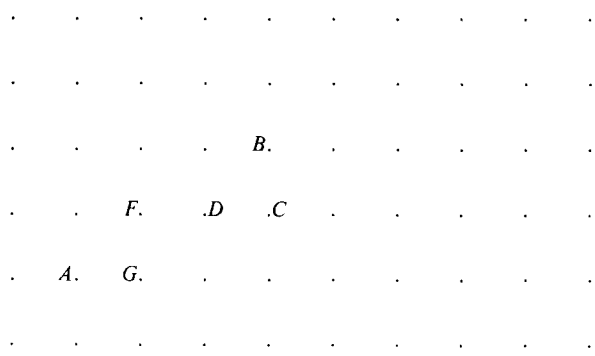

FIG. 2. Points A, B, F, C, G, and D.

in it. Let E be the lattice point which make $ABED$ a parallellogram. By rotational symmetry $ABED$ has no lattice points inside. All lattice points can be thought of as lying on lines parallel to AD. The distance between A and D is also the distance between adjacent lattice points on each of the other lines. Since $ABED$ has no other points in it (including on the edges), the points B and E must lie on the line closest to the line which contains A and D (see Fig. 1). By symmetry, C must also lie on the other line closest to AD. There are at most four points on these two lines which are at least as close to D as A (labelled F, B, G, and C in Fig. 2). Only the two furthest from A (B and C) make an acceptable triangle. If the line segment AD happens to be parallel to one of the axes then only two points on the nearby lines are close to D as A (labelled B and C in Fig. 3) and they do not make an acceptable triangle. For all other simple segments AD there is a minimal triangle containing D. There is a second possible triangle associated with AD which has A as the interior point. This shows that there are at most $O(N^2)$ minimal triangles.

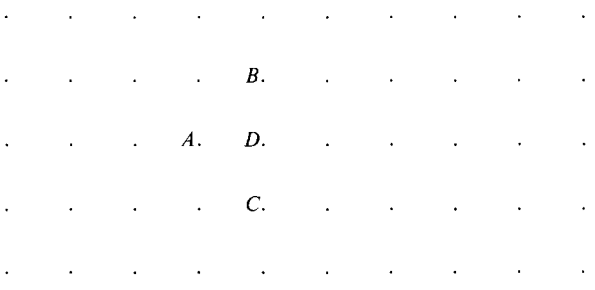

FIG. 3. The failing case.

Finally the triangles associated with the simple line segments in the central sublattice do not extend beyond the boundary of the lattice. Of the $O(N^2)$ simple line segments in the central sublattice, there are only $O(N)$ of the failing kind and so that there are at least $O(N^2)$ minimal triangle.

The theorem is also true for equilateral triangular lattices since a triangular lattice with N points in it contains a diamond with $N/2$ points in it which is projectively equivalent to a square.

6. Conclusions

This paper shows that a convexity constraint in two independent variables is rather more complex than the corresponding constraint in one dimension. The cost of determining whether a given set of data has a convex interpolation is $O(N \log N)$ in most cases. For variable z values the cost of determining the minimal linear inequalities could be $O(N^3)$ in the worst case and appears to be $O(N^2 \log N)$ in the average case. The number of independent linear constraints can vary from zero to $O(N^3)$. Unfortunately the expected number of constraints seems to be $O(N^2 \log N)$. The points have to be arranged very carefully to obtain a significant reduction in the number of independent linear constraints. Even a square lattice has $O(N^2)$ constraints. This would indicate that convexity constraints in two independent variables are computationally intractable for large data sets.

References

1. T. M. Apostol, "Introduction to Analytic Number Theory," Springer-Verlag, Berlin/New York/Heidelberg, 1976.
2. R. J. Renka, "Triangulation and Bivariate Interpolation for Irregularly Distributed Data," Ph.D. thesis, University of Texas at Austin, May, 1981.