BIT 35 (1995), 001–003.

THINNING ALGORITHMS FOR SCATTERED DATA INTERPOLATION *

MICHAEL S. FLOATER[1] and ARMIN ISKE[2]

[1] *SINTEF Applied Mathematics, Post Box 124 Blindern, 0314 Oslo, NORWAY. email: Michael.Floater@math.sintef.no*

[2] *Zentrum Mathematik, Technische Universität München, Arcisstraße 21 D-80290 München, GERMANY. email: iske@mathematik.tu-muenchen.de*

Abstract.

Multistep interpolation of scattered data by compactly supported radial basis functions requires hierarchical subsets of the data. This paper analyzes thinning algorithms for generating evenly distributed subsets of scattered data in a given domain in $\mathbb{R}^d$.

AMS subject classification: 41A15, 65D05, 65D07.

Key words: Multistep interpolation, scattered data.

1 Introduction.

Let $\Omega \subset \mathbb{R}^d$ be a given, non-empty, bounded, connected open set and let $X = \{x_1, \dots, x_N\} \subset \Omega$ be a set of pairwise distinct scattered data points, $N \geq 1$. This paper concerns the construction of hierarchical sequences of the form

$$X_1 \subset X_2 \subset \cdots \subset X_N = X, \tag{1.1}$$

with $\#(X_i) = i$, where the subsets X_i are chosen to be as evenly distributed in Ω as possible. Such sequences will be generated by *thinning algorithms*, algorithms which recursively remove points from X according to some criterion.

Our motivation for studying thinning algorithms comes from multilevel interpolation of scattered data. Let us give a brief outline of how one applies compactly supported radial basis functions to multilevel interpolation. For a comprehensive treatment of radial basis function interpolation we refer the reader to [4], [6] and [7]. The use of radial functions for multilevel interpolation has been discussed in [1], [2] and [3].

We assume we are given function values $f(x_1), \dots, f(x_N)$ of an unknown function $f : \Omega \to \mathbb{R}$, sampled at $x_1, \dots, x_N$. For a fixed compactly supported continuous basis function $\phi : [0, \infty) \to \mathbb{R}$ and some $M \geq 2$, multilevel interpolation consists of matching f by interpolants $s_k : \mathbb{R}^d \to \mathbb{R}$ of the form

$$s_k \in \langle \phi_{\alpha_k}(\| \cdot - x\|) : x \in X_{N_k} \rangle, \quad k = 1, \dots, M$$

*Received November 1996. Revised July 1997.

on a sequence of M subsets

$$X_{N_1} \subset X_{N_2} \subset \ldots \subset X_{N_M} = X \tag{1.2}$$

according to the recursive scheme

$$s_k\big|_{X_{N_k}} = f - (s_1 + \ldots + s_{k-1})\big|_{X_{N_k}}, \quad k = 1, \ldots, M. \tag{1.3}$$

Here, the functions ϕ_{α_k} are given by $\phi_{\alpha_k}(\cdot) = \phi(\cdot/\alpha_k)$ for positive scaling parameters α_k, $k = 1, \ldots, M$, and the interpolation problems in (1.3) are well posed provided that ϕ is strictly positive definite on $\mathbb{R}^d$, i.e. the matrix $(\phi(\|y_j - y_k\|)_{1 \le j,k \le N}$ is strictly positive definite for all possible selections of pairwise distinct points $y_1, \ldots, y_N \in \mathbb{R}^d$.

As was observed in [1], a critical point about this interpolation method is the selection of the subsets in (1.2). According to general results from the theory of radial basis function interpolation, a reasonable balance between the stability and the reproduction quality of the interpolation method is achieved when the sets X_{N_k} are evenly distributed in Ω (see [7] and remarks in [1]). This then explains why we need a thinning algorithm which generates sequences (1.1) where the homogeneity or *uniformity* of the subsets is as high as possible. It only remains to choose suitable break points $N_1 < N_2 < \cdots < N_M$, in order to obtain a suitable subsequence (1.2) for the multistep method.

In [1] we presented a thinning algorithm for bivariate data ($d = 2$) based on successive Delaunay triangulations. Encouraging numerical results of both the algorithm and the subsequent multilevel interpolation were provided. Further numerical examples can be found in [2].

In the current paper we wish to concentrate purely on thinning in $\mathbb{R}^d$. After some basic definitions, we introduce in Section 2 a somewhat idealized thinning algorithm, Algorithm 1, which maximizes uniformity at each removal. In Section 3, we discuss a short way to locate points for removal in this algorithm and derive a consequence about the algorithmic complexity in one dimension. The global behaviour of the algorithm is analyzed in Section 4. In particular it is shown that after one removal, the uniformity never drops by more than one half in any space dimension. Moreover if the data set is large, uniformity does not drop by more than one half of the original after several removals.

In Section 5 we propose two simpler thinning algorithms (Algorithms 2 and 3) which are naturally suggested by the results of Section 3. For the special case $d = 1$, we compare the performance of all three algorithms via selected numerical examples in Section 6.

Finally in Section 7, we take the pragmatic view of implementing Algorithm 2, defined in Section 5, and we use three different methods to do this. Firstly, for general space dimensions we implement a method which requires $\mathcal{O}(N^3)$ steps. Secondly, in the planar case, we implement a faster method using successive Delaunay triangulations, usually requiring only $\mathcal{O}(N^2)$ steps, similar to the algorithm in [1]. Finally, by employing a priority queue, implemented using a heap structure, in addition to triangulations we obtain a method which usually requires merely $\mathcal{O}(N \log N)$ steps. This latter method represents a considerable improvement in computational cost over the $\mathcal{O}(N^2)$ algorithm described in [1] and [2].

2 Uniformity and the thinning algorithm.

In what follows, though the points $x_1, \dots, x_N$ in X are fixed, we will typically refer to a general element of X as x or y or sometimes y_1, y_2, etc. rather than x_i. For $x \in X$, we use the notation X^x to denote $X \setminus \{x\}$, the set X after the removal of the point x. Also for an indexed set X_i and some $x \in X_i$, the expression X_i^x will mean $X_i \setminus \{x\}$. For any closed set $S \subset \mathbb{R}^d$ and any $x \in \mathbb{R}^d$, we write $d(x, S)$ as shorthand for $\min_{y \in S} ||x - y||$, where $||.||$ is the Euclidean norm.

In order to make a definition of uniformity, we first consider two ways of measuring the *sparsity* (the reciprocal of density) of the points in X with respect to Ω. Let

$$(2.1) \qquad s(X) = \min_{x \in X} d(x, X^x \cup \partial\Omega), \qquad \ell(X) = 2 \max_{y \in \Omega} d(y, X \cup \partial\Omega),$$

where $\partial\Omega$ denotes the boundary of Ω. We have chosen to take into account the distance of points to the boundary in these two measures, unlike the corresponding measures q and Q in [1]. This will make it easier to define thinning algorithms later since there is no need to thin points at the boundary as in [1].

Note that the maximum in $\ell(X)$ in (2.1) is attained by some $y \in \Omega$ even though Ω is an open set. The quantity $\ell(X)$ can be interpreted geometrically by observing that it is the diameter of the largest open ball contained in $\Omega \setminus X$. One can also interpret $s(X)$, namely as the minimum of all distances between points in X and distances between points in X and points in $\partial\Omega$; see Figure 1. Since Ω is fixed throughout the paper, we do not indicate the dependency of s and ℓ on Ω in the notation.

In the case $d = 1$, Ω is an interval (a, b) and for convenience we then assume that the points (or values) in X satisfy $a < x_1 < x_2 < \cdots < x_N < b$. If one then defines $x_0 = a$, $x_{N+1} = b$, and $\Delta_j = x_{j+1} - x_j$ for $j = 0, \dots, N$, one sees that

$$s(X) = \min_{0 \le j \le N} \Delta_j, \qquad \ell(X) = \max_{0 \le j \le N} \Delta_j,$$

that is, s and ℓ are the lengths of the shortest and longest intervals in $\Omega \setminus X$ respectively.

One can regard $s(X)$ in (2.1) as a lower bound on the sparsity of X and conversely $\ell(X)$ as an upper bound. In fact one can easily show that $s(X) \le \ell(X)$. This suggests taking the ratio in order to define a measure of the *uniformity* of X:

$$\rho(X) = s(X)/\ell(X).$$

Bearing in mind the application to multistep interpolation, our goal is to construct from X a hierarchical sequence of N subsets in the form (1.1) such that $\rho(X_i)$ is as high as possible for each i. Indeed, let $\mathcal{X} = (X_1, \dots, X_N)$ denote

Fig. 1. The quantities $s(X)$, $\ell(X)$, and $\ell_X(x)$.

a hierarchical sequence of the form (1.1) and denote by $\mathrm{H}(X)$ the set of all such sequences. Further we define the vector $\rho(\mathcal{X}) = (\rho(X_1), \ldots, \rho(X_N)) \in \mathbb{R}^N$. We regard $\mathcal{X} \in \mathrm{H}(X)$ as a 'good' hierarchical sequence if it has a large l_1 norm:

$$||\rho(\mathcal{X})||_1 = \sum_{i=1}^{N} \rho(X_i). \tag{2.2}$$

Though one could consider other norms and other measurements, we believe that (2.2) provides an intuitive and sufficiently sensitive indicator of uniformity when comparing two hierarchical sequences.

Since there are $N!$ hierarchical sequences in $\mathrm{H}(X)$, it would be very costly to compute $||\rho(\mathcal{X})||_1$ for all $\mathcal{X} \in \mathrm{H}(X)$ in order to find a sequence $\mathcal{X}^* \in \mathrm{H}(X)$ for which

$$||\rho(\mathcal{X}^*)||_1 = \max_{\mathcal{X} \in \mathrm{H}(X)} ||\rho(\mathcal{X})||_1.$$

Instead we want to design a thinning algorithm to find a sequence $\mathcal{X}$ which has a high value of $||\rho(\mathcal{X})||_1$. To this end, let $Y \subset X$, $Y \neq \emptyset$. Let us say that $x \in Y$ is a *removable point* if

$$\rho(Y^x) = \max_{y \in Y} \rho(Y^y). \tag{2.3}$$

This now leads us to the basic thinning algorithm.

ALGORITHM 1.

(1) Let $X_N = X$.

(2) For decreasing $i = N, \ldots, 2$,

  (2a) locate a removable point $x \in X_i$,

  (2b) let $X_{i-1} = X_i^x$.

(3) Output $\mathcal{X}^* = (X_1, \ldots, X_N)$.

Algorithm 1 is *greedy* for it maximizes ρ at each iteration. It will be useful in subsequent sections to make the general observation that if $Y \subset X$ and $x \in Y$, then from (2.1),

$$s(Y^x) \geq s(Y), \qquad \ell(Y^x) \geq \ell(Y). \tag{2.4}$$

Thus if $(X_1, \ldots, X_N) \in \mathrm{H}(X)$ then the two sequences $s(X_1), \ldots, s(X_N)$ and $\ell(X_1), \ldots, \ell(X_N)$ are monotonically decreasing. However the same will not in general be true of the sequence $\rho(X_1), \ldots, \rho(X_N)$.

3 Location of a removable point.

In this section we wish to investigate how one locates a removable point. To find a removable point in X, it appears from (2.3) and (2.1) that one needs to compute $s(X^y)$ and $\ell(X^y)$ for all $y \in X$. We will show that this involves unnecessary calculation.

In addition to s and ℓ there is a third quantity which plays an important role in this discussion. Let us first define for each $x \in X$ the set

$$V_X(x) = \{y \in \Omega : \|y - x\| \leq \|y - z\| \ \forall z \in X^x \cup \partial\Omega\}.$$

We call $V_X(x)$ the *generalized Voronoi tile* of x with respect to X and Ω. The set $V_X(x)$ contains all those points in Ω which are no further from x than any other point in X or $\partial\Omega$. Now we define for each $x \in X$ the quantity

$$\ell_X(x) = 2 \max_{y \in V_X(x)} d(y, X^x \cup \partial\Omega). \tag{3.1}$$

One can think of $\ell_X(x)$ as a measure of the *local* sparsity of the data set, in contrast to $s(X)$ and $\ell(X)$. We can interpret $\ell_X(x)$ geometrically by noticing that it is the diameter of the largest open ball in Ω which contains no point of X^x and whose closure contains x; see Figure 1.

We now derive a useful identity concerning the value of ℓ after the removal of a point x from X in terms of $\ell_X(x)$.

LEMMA 3.1. *For all $x \in X$,*

$$\ell(X^x) = \max\{\ell(X), \ell_X(x)\}. \tag{3.2}$$

PROOF. Let $S = X \cup \partial\Omega$. By the definition of $\ell_X(x)$ we see that

$$\ell(X^x) = 2 \max_{y \in \Omega} d(y, S^x) \geq 2 \max_{y \in V_X(x)} d(y, S^x) = \ell_X(x).$$

Since also $\ell(X^x) \geq \ell(X)$ we therefore have

$$\ell(X^x) \geq \max\{\ell(X), \ell_X(x)\}. \tag{3.3}$$

To demonstrate the opposite inequality consider that for each $y \in \Omega$ there is at least one $z \in S$ for which the least distance from y to S is attained, that is $d(y, S) = ||y - z||$. By the definition of $V_X(x)$, if $y \in \Omega \setminus V_X(x)$, such a point z cannot equal x and so $z \in S^x$. Therefore

$$d(y, S) = d(y, S^x), \qquad \text{for } y \in \Omega \setminus V_X(x).$$

In consequence

$$\begin{aligned} \ell(X^x) &= \max\left\{2 \max_{y \in \Omega \setminus V_X(x)} d(y, S^x),\ 2 \max_{y \in V_X(x)} d(y, S^x)\right\} \\ &= \max\left\{2 \max_{y \in \Omega \setminus V_X(x)} d(y, S),\ \ell_X(x)\right\} \\ &\leq \max\left\{2 \max_{y \in \Omega} d(y, S),\ \ell_X(x)\right\} \\ &= \max\{\ell(X), \ell_X(x)\}. \end{aligned}$$

□

Using Lemma 3.1, we now find that a removable point can be found very quickly once three special points with simpler characteristics have been located.

PROPOSITION 3.2. *Let $y_1 \in X$ and $y_2 \in X \cup \partial\Omega$, $y_1 \neq y_2$, be such that $s(X) = \|y_1 - y_2\|$ and let $y_3 \in X$ be such that $\ell_X(y_3) = \min_{x \in X} \ell_X(x)$. Define $S = \{y_1, y_2, y_3\} \cap X$, so that S contains 1, 2, or 3 elements. If $x \in S$ satisfies*

$$\rho(X^x) = \max_{y \in S} \rho(X^y), \tag{3.4}$$

then x is a removable point.

PROOF. If $S = X$ then x is a removable point by definition. Otherwise let $z \in X \setminus S$. Then $z \neq y_1$ and $z \neq y_2$ and so

$$s(X^z) = \|y_1 - y_2\| = s(X) \leq s(X^{y_3}).$$

Further, using Lemma 3.1, we have from the definition of y_3 that

$$\ell(X^z) = \max\{\ell(X), \ell_X(z)\} \geq \max\{\ell(X), \ell_X(y_3)\} = \ell(X^{y_3}).$$

It follows that

$$\max_{y \in S} \rho(X^y) \geq \rho(X^{y_3}) = s(X^{y_3})/\ell(X^{y_3}) \geq s(X^z)/\ell(X^z) = \rho(X^z).$$

Therefore $\max_{y\in S}\rho(X^y) = \max_{y\in X}\rho(X^y)$ and so from (2.3), any $x \in S$ satisfying (3.4) is a removable point. □

Proposition 3.2 will help to locate a removable point. Indeed, the computation of $\ell_X(x)$ for any $x \in X$ can be performed locally by merely considering the Voronoi tile $V_X(x)$.

In the case $d = 1$, we write ℓ_j for $\ell_X(x_j)$, and we find that $\ell_j = x_{j+1} - x_{j-1}$ for $j = 1, \ldots, N$. Then suppose $i_1 \in \{0, \ldots, N\}$ is any index such that $s(X) = \Delta_{i_1}$ and $i_2 \in \{1, \ldots, N\}$ is any index such that $\ell_{i_2} = \min_{1\le k\le N} \ell_k$. If we define

$$I = \{i_1, i_1 + 1, i_2\} \cap \{1, \ldots, N\}, \tag{3.5}$$

then Proposition 3.2 says that if

$$\rho(X^{x_k}) = \max_{l\in I} \rho(X^{x_l}),$$

then x_k is a removable point. This has the following consequence.

COROLLARY 3.3. *When $d = 1$, the thinning algorithm can be performed in $\mathcal{O}(N^2)$ steps.*

PROOF. Since both i_1 and i_2 in (3.5) can be located in $\mathcal{O}(N)$ steps, it follows that a removable point in X_N can be located in $\mathcal{O}(N)$ steps. Since for all $i \in \{1, \ldots, N\}$, we have $X_i \subset X_N$ it follows that a removable point in X_i can also be located in $\mathcal{O}(N)$ steps. We deduce that a hierarchical sequence can be obtained from the thinning algorithm in $\mathcal{O}(N^2)$ operations. □

4 Global behaviour.

Now let $\mathcal{X}^* = (X_1, \ldots, X_N)$ be any hierarchical sequence generated by the thinning algorithm and consider the sequence of values $\rho(X_1), \ldots, \rho(X_N)$. It was pointed out in Section 2 that $\rho(X_1), \ldots, \rho(X_N)$ is not necessarily monotonically decreasing. Despite this, we show it tends to decrease rather than increase, confirming numerical observations made in [1]. We begin with an observation about the deletion of an arbitrary point from X. First we require an estimate for $\ell_X(x)$.

LEMMA 4.1. *For all $x \in X$,*

$$\ell_X(x) \le 2\ell(X). \tag{4.1}$$

PROOF. Let $x \in X$ and for convenience set $S = X \cup \partial\Omega$. By the definition of $\ell_X(x)$ in (3.1), there exists $y \in V_X(x)$ such that

$$\ell_X(x) = 2d(y, S^x).$$

If we let $r = d(y, S^x)$ we then see that the open ball $B(y, r)$ contains no points of S^x. Further, since $y \in V_X(x)$, we have that $||y - x|| \le r$, and so x belongs to the closure of $B(y, r)$.

Fig. 2. The positions of x and z relative to y.

Suppose that $y = x$. If we let z be any point whose distance from y is $r/2$ then the open ball $B(z, r/2)$ contains no points in S. Otherwise $x \neq y$; see Figure 2. Then let

$$z = y + \frac{r}{2} \frac{(y - x)}{||y - x||}$$

and again the open ball $B(z, r/2)$ contains no points in S. Therefore in both cases we have by the definition of $\ell(X)$ that

$$\ell(X) \geq 2d(z, S) \geq r = \ell_X(x)/2.$$

□

We remark that inequality (4.1) is the best possible, independent of the space dimension d. To see this let X contain only one point x and suppose that Ω is any open ball $B(x, r)$, $r > 0$. Then $X^x = \emptyset$ and $V_X(x)$ is the closure of $B(x, r/2)$ and therefore

$$\ell_X(x) = 2 \max_{y \in V_X(x)} d(y, \partial\Omega) = 2d(x, \partial\Omega) = 2r.$$

On the other hand, if y_0 is any point such that $||y_0 - x|| = r/2$ we have

$$\ell(X) = 2 \max_{y \in \Omega} d(y, X \cup \partial\Omega) = 2d(y_0, X \cup \partial\Omega) = r$$

and so $\ell_X(x) = 2\ell(X)$. Lemma 4.1 leads to the following estimate for the uniformity after removing one point from X.

Proposition 4.2. *For any $x \in X$,*

$$\rho(X^x) \geq \rho(X)/2.$$

PROOF. From (2.4), $s(X^x) \geq s(X)$. On the other hand, from (3.2) and Lemma 4.1,

$$\ell(X^x) = \max\{\ell(X), \ell_X(x)\} \leq 2\ell(X),$$

and therefore $\rho(X^x) \geq \rho(X)/2$. □

Due to Proposition 4.2, we see that

$$\rho(X \setminus \{y_1, y_2\}) \geq \rho(X \setminus \{y_1\})/2 \geq \rho(X)/4$$

for any $y_1, y_2 \in X$, $y_1 \neq y_2$. However by choosing y_1 and y_2 to be successive removable points according to (2.3), we can improve on this inequality. The following proposition shows that the uniformity of the subsets in the thinning algorithm can never fall below a half of the original until a subset X_i of X is reached for which the value of $\ell_{X_i}(x)$ is strictly greater than the original $\ell_X(x)$ for all x in X_i. Since $\ell_{X_j}(x)$ depends on only a few points in X_j close by (three in Figure 1), there will usually be many removals before such an X_i is reached.

PROPOSITION 4.3. *Let $\mathcal{X}^* = (X_1, \ldots, X_N)$ be a hierarchical sequence generated by the thinning algorithm. Let m be any integer in $\{1, \ldots, N\}$ such that there exists some $x \in X_{N-m}$ such that $\ell_{X_{N-m}}(x) = \ell_X(x)$. Then for all $k = 1, \ldots, m+1$,*

$$\rho(X_{N-k}) \geq \rho(X_N)/2. \tag{4.2}$$

PROOF. The proof is by induction on k. Inequality (4.2) holds when $k = 1$ since Proposition 4.2 then applies. Now let $k \in \{2, \ldots, m+1\}$ and suppose that $\rho(X_{N-k+1}) \geq \rho(X_N)/2$. Since $\ell_{X_{N-i}}(x)$ is monotonically increasing in i, and $k \leq m+1$, we have $\ell_{X_{N-k+1}}(x) = \ell_X(x)$. Therefore from (3.2),

$$\ell(X^x_{N-k+1}) = \max\{\ell(X_{N-k+1}), \ell_{X_{N-k+1}}(x)\} \leq \max\{\ell(X_{N-k+1}), 2\ell(X_N)\}.$$

Then using (2.3) and the fact that $s(X_{N-i})$ is monotonically increasing in i, we find that

$$\begin{aligned}
\rho(X_{N-k}) &\geq \rho(X^x_{N-k+1}) \\
&= s(X^x_{N-k+1})/\ell(X^x_{N-k+1}) \\
&\geq \min\{s(X^x_{N-k+1})/\ell(X_{N-k+1}), s(X^x_{N-k+1})/2\ell(X_N)\} \\
&\geq \min\{s(X_{N-k+1})/\ell(X_{N-k+1}), s(X_N)/2\ell(X_N)\} \\
&= \min\{\rho(X_{N-k+1}), \rho(X_N)/2\} \\
&\geq \rho(X_N)/2.
\end{aligned}$$

□

We illustrate the use of Proposition 4.3 in the case $d = 1$. For $t \in \mathbb{R}$, $t \geq 0$, let $[t]$ denote the integer part of t.

COROLLARY 4.4. *Let $d = 1$ and let $\mathcal{X}^* = (X_1, \ldots, X_N)$ be any hierarchical sequence generated by the thinning algorithm. Then for $k = 0, \ldots, [(N+2)/3]$,*

$$\rho(X_{N-k}) \geq \rho(X_N)/2. \tag{4.3}$$

PROOF. Let $m = [(N+2)/3] - 1$. If we can show that there is an index $i \in \{1, \dots, N\}$ such that $\{x_{i-1}, x_i, x_{i+1}\} \subset X_{N-m} \cup \partial\Omega$ then since $\ell_{X_{N-m}}(x_i) = \ell_{X_N}(x_i) = x_{i+1} - x_{i-1}$, (4.3) immediately follows from Proposition 4.3.

To this end, suppose for the purpose of contradiction that there is no index $i \in \{1, \dots, N\}$ such that $\{x_{i-1}, x_i, x_{i+1}\} \subset X_{N-m} \cup \partial\Omega$. Then, noting that

$$3m + 2 = 3[(N+2)/3] - 1 \leq N + 1,$$

we have that for all $j = 0, \dots, m$,

$$\{x_{3j}, x_{3j+1}, x_{3j+2}\} \not\subset X_{N-m} \cup \partial\Omega.$$

This means that at least $m+1$ points have been removed in the thinning algorithm from X_N to obtain X_{N-m} which is a contradiction. ◻

Since all arguments in Proposition 4.3 and Corollary 4.4 apply also when X_N is replaced by X_n, for any $n = 1, \dots, N$, we have more generally that when $d = 1$,

$$\rho(X_{n-k}) \geq \rho(X_n)/2,$$

for all $n \in \{1, \dots, N\}$ and $k = 0, \dots, [(n+2)/3]$.

EXAMPLE 4.1. *Let $d = 1$, $\Omega = (0,1)$ and $x_i = 2^{i-N-1}$, $i = 1, \dots, N$, for some $N \geq 1$. In this case the hierarchical sequence $\mathcal{X}^* = (X_1, \dots, X_N)$ obtained from the thinning algorithm is unique with $X_i = \{x_{N-i+1}, \dots, x_N\}$ for $i = 1, \dots, N$. Further, $s(X_i) = 2^{-i}$, $\ell(X_i) = 1/2$ and thus $\rho(X_i) = s(X_i)/\ell(X_i) = 2^{1-i}$. So $\rho(\mathcal{X}^*) = (1, 1/2, \dots, 1/2^{N-1})$ which is monotonically decreasing.*

EXAMPLE 4.2. *Let $d = 1$, $\Omega = (0,1)$, $N = 3$, and $x_i = i/4$ for $i = 1, 2, 3$ and let $\mathcal{X}^*$ be a hierarchical sequence generated by the thinning algorithm. Now the thinning algorithm may remove x_1 or x_3 first, in which case $\rho(\mathcal{X}^*) = (1, 1/2, 1)$ and $||\rho(\mathcal{X}^*)||_1 = 5/2$. However the algorithm may instead remove x_2 first and then $\rho(\mathcal{X}^*) = (1/3, 1/2, 1)$ and $||\rho(\mathcal{X}^*)||_1 = 11/6$. The latter case shows that (4.3) need not hold for $k > [(N+2)/3]$.*

5 Simplified and modified algorithms.

Let $Y \subset X$, $Y \neq \emptyset$. It was shown in Proposition 3.2 that there is a removable point in Y among any three points $y_1, y_3 \in Y$ and $y_2 \in Y \cup \partial\Omega$ satisfying

$$s(Y) = \|y_1 - y_2\| \quad \text{and} \quad \ell_Y(y_3) = \min_{x \in Y} \ell_Y(x). \tag{5.1}$$

This suggests two simpler algorithms: in the first we only remove points of type y_1, y_2 and in the second points of type y_3. To be precise, let us define Algorithm 2 to be identical to Algorithm 1 except that we say $x \in Y$ is removable if

$$d(x, Y^x \cup \partial\Omega) = s(Y).$$

We define Algorithm 3 in a similar way but where $x \in Y$ is removable if

$$\ell_Y(x) = \min_{y \in Y} \ell_X(y).$$

The advantage of Algorithms 2 and 3 is that they obviously require less computation than Algorithm 1.

In addition to the two simpler algorithms we propose modifications to Algorithms 1 and 2 by using a second criterion for point removal in the event that there are several removable points. Indeed, removable points as defined in (2.3) are not necessarily unique. In some cases one will find that y_2 in (5.1) belongs to Y and moreover

$$\rho(Y^{y_1}) = \rho(Y^{y_2}) > \rho(Y^{y_3}).$$

A natural choice of a side condition is to choose that removable point x in Y which minimizes $\ell_Y(x)$. Thus we define Algorithm $1'$ by saying that x is removable if

$$\rho(Y^x) = \max_{y \in Y} \rho(Y^y) \quad \text{and} \quad \ell_Y(x) = \min_{y \in Y} \{\ell_Y(y) : \rho(Y^x) = \rho(Y^y)\}.$$

Similarly we define Algorithm $2'$ by saying that $x \in Y$ is removable if

$$d(x, Y^x \cup \partial\Omega) = s(Y)$$

$$\text{and} \qquad \ell_Y(x) = \min_{y \in Y} \{\ell_Y(y) : d(x, Y^x \cup \partial\Omega) = d(y, Y^y \cup \partial\Omega)\}.$$

The thinning algorithm presented in [1] which is based on Delaunay triangulations for bivariate data, $d = 2$, is similar to Algorithm $2'$. A point is a *candidate* for removal in the algorithm of [1] if it is an endpoint of one of the shortest edges in the triangulation. A point is *removable* if it is one of these candidates and in addition its longest incident edge is least among them.

6 Numerical examples in one dimension.

For one-dimensional scattered data in an interval, we have implemented all the five thinning algorithms of Section 5 for the sake of comparison. For the numerical examples, two data sets were chosen with $d = 1$, $\Omega = (0, 1)$ and $N = 200$. In Data Set 1 the points $x_1, \ldots, x_N \in X$ were chosen randomly while in Data Set 2, they were chosen uniformly, that is $x_i = i/(N + 1)$, $i = 1, \ldots, N$. For both data sets, Algorithms 1, $1'$, 2, $2'$, and 3 were applied and the sequence $\rho(X_1), \ldots, \rho(X_{200})$ recorded where $\mathcal{X}^* = (X_1, \ldots, X_{200})$ was the output of the algorithm. Table 1 shows the norms $||\rho(\mathcal{X}^*)||_1$ in each of the ten cases.

Table 1. Uniformity norms

Algorithm	1	$1'$	2	$2'$	3
Data Set 1	41.6	43.8	32.7	43.3	35.6
Data Set 2	91.8	91.3	83.6	82.4	88.9

The table suggests that generally Algorithms 1 and $1'$ yield the highest uniformity. In the case of random data, Algorithms 2 and 3 fall somewhat behind the others, indicating that when only one measure of sparsity is taken into account the result is significantly worsened. Figures 3, 4, 5, 6, and 7 show the graphs of the sequences $\rho(X_1), \ldots, \rho(X_{200})$ for Data Sets 1 and 2 respectively.

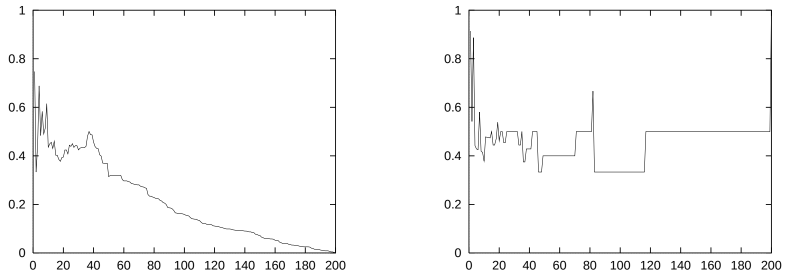

Fig. 3. Uniformity for Algorithm 1.

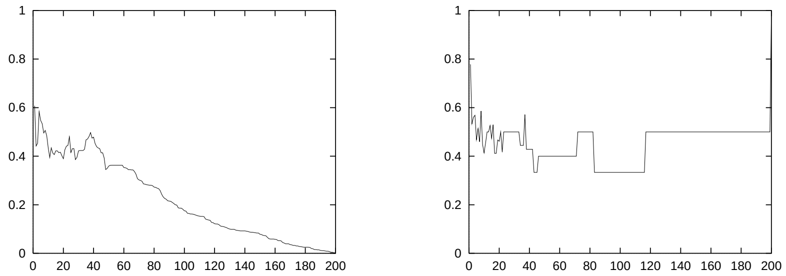

Fig. 4. Uniformity for Algorithm 1′.

7 Numerical examples in higher dimensions.

For practical purposes, Algorithm 1 seems rather difficult to deal with and may require a generalized Voronoi diagram to be implemented. Instead we take the pragmatic approach in this section of implementing Algorithm 2 (removing a point whose least distance to the other points and the boundary is minimal). We describe three methods for implementing Algorithm 2 and we note that all of them can easily be adapted to disregard the distance of points to the boundary if this is desirable.

7.1 Comparing all pairs.

For general dimension, a very simple implementation of Algorithm 2 is the following. Suppose that the domain Ω is the unit hypercube $[0,1]^d$. To find a removable point from the current subset, we calculate the distance between every pair of current data points ($\mathcal{O}(N^2)$ steps) and take the minimum. Then we calculate the minimum distance of the current data points to the boundary of the hypercube ($\mathcal{O}(N)$ steps). Any data point which achieves the over all minimum is then removed. The complexity of the whole thinning algorithm is therefore $\mathcal{O}(N^3)$ in any fixed space dimension d. The results of Algorithm 2 in two and three dimensions are displayed in Figures 8 and 9 where $N = 1000$ data points were chosen randomly from the unit square and unit cube respectively. In each dimension, the subsets X_{500}, X_{250}, and X_{100} generated by the algorithm

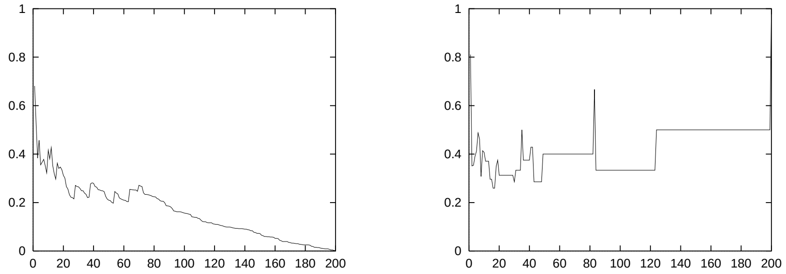

Fig. 5. Uniformity for Algorithm 2.

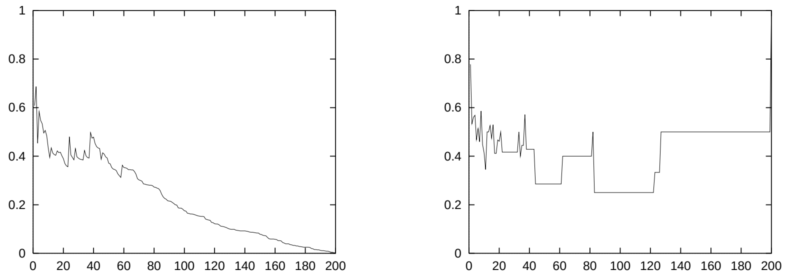

Fig. 6. Uniformity for Algorithm 2′.

are shown. The column 'All Pairs' of Table 2 shows the CPU time in seconds used by Algorithm 2 using this method on points taken randomly from the unit square.

7.2 Delaunay triangulations.

In the planar case ($d = 2$) an improvement on the above simple algorithm is to first set up a Delaunay triangulation of the initial data set which can be achieved in $\mathcal{O}(N \log N)$ operations; see [5]. Then we use the fact that the closest pair of points always form an edge in the triangulation [5]. Thus a removable point in Algorithm 2 can be located in $\mathcal{O}(N)$ operations since there are $\mathcal{O}(N)$ edges to check and the distance from each point to the boundary of the unit square requires only four comparisons. After removing the point, the remaining points are again triangulated with a Delaunay triangulation. Assuming the number of neighbours of each point is $\mathcal{O}(1)$, the retriangulation requires only $\mathcal{O}(1)$ steps. Iterating this procedure, the hierarchy of subsets of Algorithm 2 can thus be generated in $\mathcal{O}(N^2)$ steps. We implemented this algorithm and the column 'Triangulation' of Table 2 shows the CPU time in seconds. This was also the general philosophy of the algorithm described in [1].

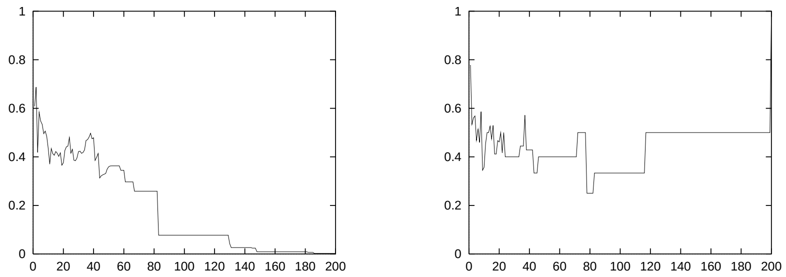

Fig. 7. Uniformity for Algorithm 3.

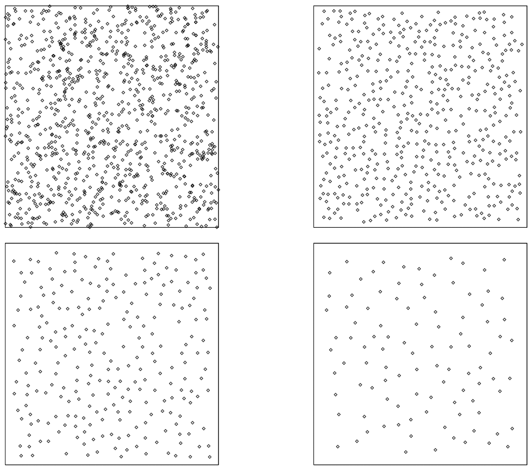

Fig. 8. The set X and subsets X_{500}, X_{250}, and X_{100} in Algorithm 2.

7.3 Delaunay triangulations and priority queue.

Finally, we suggested in [2] that an algorithm of complexity $O(N \log N)$ might be possible using techniques from computational geometry. We have now made an implementation of Algorithm 2, based on a *priority queue* in addition to Delaunay triangulations, whose performance is approximately $O(N \log N)$ in our numerical examples. The basic idea of our approach is to assign a *priority* for removal to each point in the current subset. Specifically we take the priority of the point to be the reciprocal of its minimum distance to the other points and to the boundary of the unit square. The point with the highest priority is the point to be removed.

We first build a Delaunay triangulation of the point set. We then compute successively the priorities of the points and insert them into a *heap*, a binary tree in which the priorities of the two children are never greater than the priority of the parent (the heap condition); see Sedgewick [8], Chapter 11. Due to standard properties of the Delaunay triangulation, the minimum distance of a point to

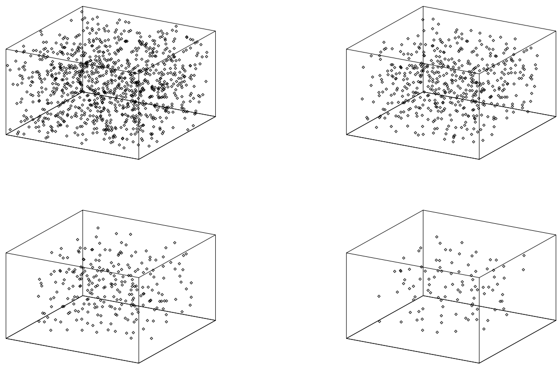

Fig. 9. The set X and subsets X_{500}, X_{250}, and X_{100} in Algorithm 2.

the other points is the length of one of the edges incident on it [5]. So, assuming the number of neighbours in the triangulation of each point is $\mathcal{O}(1)$ and because every insertion in the heap takes $\mathcal{O}(\log N)$ steps, the construction of the initial heap takes $\mathcal{O}(N \log N)$ steps.

Once the heap contains all data points, we begin the thinning sequence. We begin by deleting from the heap the point with highest priority (which will be the root of the heap) and update the heap so that it again satisfies the heap criterion. This requires $\mathcal{O}(\log N)$ operations. We then remove the point also from the triangulation, retriangulate, and calculate the new priorities of the points that were neighbours of the removed point in the previous triangulation. These priorities are updated in the heap, each requiring $\mathcal{O}(\log N)$ steps. The priorities of all other points are the same as before. We now remove the point with the current highest priority and so on. Since the total time required to remove one point is $\mathcal{O}(\log N)$ we deduce that under the assumption of a bounded number of neighbours in every triangulation, the total complexity of the thinning (including the initial triangulation and heap construction) is $\mathcal{O}(N \log N)$. We implemented this method and the column 'Priority Queue' of Table 2 shows the CPU time in seconds. It was found in these examples that approximately 75% of the CPU time was spent on the thinning sequence while the remainder was spent on the initial triangulation and construction of the initial heap.

We would expect that an analogous algorithm, using Delaunay tetrahedralizations together with a priority queue would yield an implementation of Algorithm 2 in three dimensions with low order of algorithmic complexity.

Table 2. CPU times in seconds of three implementations of Algorithm 2

N	All pairs	Triangulation	Priority Queue
50	0.01	0.05	0.03
100	0.10	0.17	0.08
200	0.83	0.63	0.16
400	6.61	2.48	0.33
800	51.96	9.65	0.71
1600	416.60	39.48	1.53
3200		172.61	3.33
5000			5.47
10000			12.36
20000			28.22
40000			73.93
100000			232.35

REFERENCES

1. M. S. Floater and A. Iske, *Multistep Scattered Data Interpolation using Compactly Supported Radial Basis Functions*, J. Comp. Appl. Math., 73 (1996), pp. 65–78.
2. M. S. Floater and A. Iske. *Thinning and Approximation of Large Sets of Scattered Data.* In Advanced Topics in Multivariate Approximation, F. Fontanella, K. Jetter, P.-J. Laurent (eds.), World Scientific, Singapore, 1996, pp. 87–96.
3. F. J. Narcowich, R. Schaback, and J. D. Ward. *Multilevel Interpolation and Approximation*, preprint, 1996.
4. M. J. D. Powell *The Theory of Radial Basis Function Approximation in 1990.* In Advances in Numerical Analysis II: Wavelets, Subdivision, Algorithms, and Radial Basis Functions, W. A. Light (ed.), Oxford University Press, 1992, pp. 105–210.
5. F. P. Preparata and M. I. Shamos, *Computational Geometry*, Springer, New York, 1985.
6. R. Schaback *Creating Surfaces from Scattered Data using Radial Basis Functions.* In Mathematical Methods for Curves and Surfaces, M. Dæhlen, T. Lyche, and L. L. Schumaker (eds.), Vanderbilt University Press, Nashville, 1995, pp. 477–496.
7. R. Schaback *Multivariate Interpolation and Approximation by Translates of a Basis Function.* In Approximation Theory VIII, C. K. Chui and L. L. Schumaker (eds.), World Scientific, Singapore, 1995, pp. 491–514.
8. R. Sedgewick, *Algorithms.* Addison-Wesley, U. S. A., 1983.