Reprinted from:
ALGORITHMS AND COMPLEXITY
New Directions and Recent Results

ACADEMIC PRESS, INC.
New York San Francisco London

# GEOMETRY AND STATISTICS: PROBLEMS AT THE INTERFACE

Michael Ian Shamos
*Departments of Computer Science and Mathematics*
*Carnegie-Mellon University*
*Pittsburgh, PA 15213*

## Abstract

In this paper we approach the analysis of statistics algorithms from a geometric viewpoint and use techniques from computational geometry to develop new, fast algorithms for computing familiar statistical quantities. Such fundamental procedures as sorting and selection play an important role in nonparametric estimation as well as in correlation and regression and we use known results to obtain lower bounds on the time required to perform various statistical tests. For some problems, computing the test statistic is NP-hard. While geometric insight is helpful in understanding statistical calculations, the reverse is also true -- we employ statistical methods to analyze the average case of geometric algorithms.

## 1. Introduction

From the viewpoint of applied computational complexity, statistics is a gold mine, for it provides a rich and extensive source of unanalyzed algorithms and computational procedures. For the statistician, however, the search for efficient algorithms has not been of prime concern for several reasons: First, the design of fast algorithms is a new and developing art. Second, until recently, the cost of obtaining data has been far greater than the cost of analyzing it. Now, however, speech and image processing provide information to statistical analysis programs rapidly and cheaply, so that fast analysis is of considerable importance. Third, statisticians are properly concerned with the significance and effectiveness of the tests they perform, rather than with their cost. The result has been that the analysis of statistical algorithms remains largely ignored. We will undertake in this paper to begin the systematic study of the algorithms of statistics.

But where to begin? A reasonable approach is to lay bare the algorithmic elements that underpin statistical recipes and apply known results in order to analyze them. A large class

of statistical procedures, the nonparametric tests, are based on ranks, so it is natural to expect sorting and selection algorithms to come into play. The geometric flavor of statistics becomes apparent when a sample is regarded as a set of points in Euclidean space. For example, linear regression asks for a hyperplane which obeys a certain relation to the points of the sample. In this sense it can be looked on as a purely geometric problem. Our objective is to explore the connection between combinatorial, geometric, and statistical problems in order to obtain useful insight into the complexity of statistical algorithms. An important benefit in the reverse direction is that statistical methods can be applied to the average-case analysis of geometric algorithms.

The field of computational statistics is quite extensive, encompassing random-number generation [Knuth 76], monte-carlo methods [Hammersley 64], design of experiments, numerical analysis, interactive data analysis, time-series analysis, and other "classical" studies. We are not concerned with these here, but will concentrate primarily on the discrete algorithms of nonparametric statistics, many of which are detailed in [Conover (71), Hollander (73), and Siegel (56)]. Background material on geometric algorithms can be obtained from [Shamos (75a) and Shamos (75b)]. For a treatment of the spirit and methods of analysis of algorithms, [Aho (74)] is unsurpassed.

The model of computation assumed throughout, unless otherwise specified, is a random-access machine (RAM) similar to that described in [Aho (74)] but in which each storage location holds a single real number and infinite-precision real arithmetic is performed.

## 2. Central Tendency

Even such simple statistical quantities as the measures of central tendency present interesting features. Let X denote a sample of N real numbers, so $X = \{x_i \mid i=1,\ldots,N\}$. The arithmetic mean of X, as is well-known, can be computed in O(N) time and constant space via its algebraic definition:

$$\text{mean}(X) = \frac{1}{N} \sum_{i=1}^{N} x_i \quad . \tag{2.1}$$

An on-line algorithm is one which does not have the advantage of being able to inspect all its data at once, but must process each input in turn and provide the correct partial answer for all data read thus far. Clearly, (2.1) also provides an on-line algorithm for computing the mean. It is

known that the mean (and standard deviation) can be computed on-line in linear time without sacrificing numerical stability [Hanson (75), Cotton (75)].

The median of X is the "middle" element of the ordered $x_i$. If we denote the $i^{th}$ smallest element of X by $x_{(i)}$, then for N odd, median(X) = $x_{(\lfloor N/2 \rfloor)}$. It was long believed that the most efficient method of determining the median was to sort the $x_i$ and thus obtain the middle element. However, devilishly clever linear-time algorithms for the median (or any other order statistic) have been given by [Blum (72)] and [Floyd (73)]. It was shown in [Pohl (72)] that O(N) storage locations are required to compute the median under the assumption that each number saved is stored in a different location (in order to avoid encoding tricks). The possibility of an on-line linear time median algorithm is ruled out by

Theorem 2.1 *Any on-line median algorithm must take at least O(N log N) time in the worst case.*

Proof: Let M be a number larger than any of the $x_i$. Such an M can be found in linear time. Assume the existence of some algorithm A which will find the median on-line in less than O(N log N) time and consider the sequence of 3N numbers consisting of the $x_i$ followed by 2N M's. After the first N numbers are read (the $x_i$), the median of X will be output by A. For each successive pair of M's read by A, the next largest x must be output. Thus algorithm A will produce a list of the largest N/2 $x_i$ in sorted order. □

The O(N log N) bound can be achieved by a number of sorting algorithms.

The median can be computed purely by making comparisons among the $x_i$; the numerical values are irrelevant, which makes the median a useful estimator when only ordinal data are available. The discrete character of the median is disguised, however, by the following formulation, which makes it appear continuous:

$$\text{median}(X) = \min_x \sum_{i=1}^{N} | x_i - x | \quad . \tag{2.2}$$

A useful generalization is the weighted median, defined by

$$\text{wmedian}(X) = \min_x \sum_{i=1}^{N} w_i | x_i - x | \quad , \tag{2.3}$$

where the $w_i$ are given positive real weights. A linear-time algorithm for the weighted median, which makes recursive use

of a linear unweighted median algorithm, has been given by [Bentley (76)]. While (2.2) and (2.3) are unimodal, it is unprofitable to apply function minimization methods to them (the so-called median regression) because each function evaluation requires linear time, enough to solve the whole problem by other methods.

The mode of X is the value that occurs most frequently. Specifically, consider X to be a multiset in which element $x_i$ occurs $n_i$ times. Then $\text{mode}(X) = \{ i \mid n_i \text{ a maximum} > 1 \}$. Computing the mode is strictly more difficult than computing either the mean or median:

Theorem 2.2 *Finding the mode of N numbers requires at least O(N log N) time in the worst case, if comparisons are allowed only between linear functions of the inputs.*

Proof: Consider in particular a set whose elements are all distinct. This set has no mode. Thus any algorithm which finds the mode is able to determine whether all elements of a set are distinct, but this requires O(N log N) comparisons by a result of Dobkin and Lipton [Dobkin (75)]. □

If on-line algorithms are considered, the restriction on the comparisons can be removed. It is easy to see that any on-line algorithm for the mode must take O(N log N) time by again considering a set all of whose elements are distinct. As each number is read, the algorithm must report whether or not the number has been seen before, which must take O(log i) time after i numbers have been read. The same example shows that linear space is required, even by off-line algorithms, for there is no element that can be deleted before the entire set is read. Curiously, if the $x_i$ are integers, then their mode can be found in O(N) time. [Gonzalez (75a)]. The price we pay for this convenience is that the number of storage locations required is the magnitude of the largest $x_i$, although not all of these locations need be initialized.

While the results of this section are quite trivial, they serve to point out some of the complexity considerations that arise in computing the most elementary statistics.

## 3. Discrete Tools for Statistical Algorithms

In this section we will develop the techniques necessary to build optimal algorithms for more advanced statistical problems. A powerful principle of algorithm design is the recursive use of one optimal algorithm to yield another, an idea that we will exploit repeatedly in the following pages.

(All quantiles) While any particular order statistic can be found in linear time, we are often interested in determining all of the k-1 points that partition an ordered set into k equal pieces. A common example is that of computing the 99 percentile points of a discrete distribution. This can be done, for any k, in O(Nk) time by finding each quantile separately in O(N) time. It thus seems that sorting is a superior method for k>log N. The following theorem shows that it is never advantageous to sort.

Theorem 3.1 [Yao (75), Gibbons (76)] *All k-1 k-tiles of a set can be found in O(N log k) time.*

Proof: Of the k-1 quantile points, half lie above the median and half lie below. Likewise, N/2 elements lie above the median and N/2 lie below. Thus finding the median divides the problem into two subproblems of equal size. If T(N,k) denotes the time sufficient to find all k-tiles in a N-set, then we have the recurrence $T(N,k) \leq 2T(N/2,k/2) + O(N)$, whose solution is $T(N,k) = O(N \log k)$. The O(N) term in the recurrence includes the time necessary to find the median recursively and perform bookkeeping tasks. Whether N and k are even or odd at any stage of the recurrence does not affect the order of the solution. □

That Theorem 3.1 is best possible follows from the fact that any algorithm which has better asymptotic behavior could be used to sort in less than O(N log N) time.

The rank of an element is its position in a list of all the elements in ascending order. That is, $\text{rank}(x_{(j)}) = j$. While the success of Theorem 3.1 depends on the fact that the quantiles are evenly spaced, we have

Theorem 3.2 *The ranks of any k elements of a set can be found in O(N log k) time.*

Proof: Sort the k elements in O(k log k) time. These define k+1 buckets into which we will insert the remaining N-k elements one at a time by binary search. It is not necessary to perform the insertion, but merely to maintain a count of how many elements fall into each bucket. This can be done in $O((N-k) \log k)$ time after which a single linear pass through the bucket counts will determine the ranks of the k selected elements. □

Note that the above algorithm succeeds regardless of the relative position of the elements whose ranks are being found,

and that it is always faster than sorting the set.

Let $a_1\ a_2\ \ldots\ a_N$ be a permutation of $\{1,2,\ldots,N\}$. A pair of elements $(a_i,a_j)$ forms an inversion if $i<j$ but $a_i>a_j$. We will call the number of inversions in which a particular element appears its inversion index. The collection of all inversion indices is the inversion table. [Knuth (73)] contains two algorithms for computing the inversion table of a permutation in O(N log N) time and attributes the proof of their optimality to A.J.Smith. The importance of counting inversions will become apparent in the next section.

The number of elements in a set is a trivial lower bound on the number of operation required to process it, if each element must be examined. In many cases, however, it is possible to perform computations on a set without accessing all of its members. An example is provided by the following

Theorem 3.3 [Jefferson (76)] *The common median of two sorted sets, each containing N elements, can be found in O(log N) time.*

Proof: The fact that the sets are ordered enables us to access the $i$th largest element of either set in constant time. Let $m_1$, $m_2$ denote the separate medians of the sets. In one comparison we can determine which is larger; suppose without loss of generality that $m_1 > m_2$. No element of set 1 that is greater than $m_1$ can be the common median, since we can exhibit more than N (of the 2N) elements which lie below it. Similarly, no element of set 2 that is less than $m_2$ can be the common median. Since half of the elements in each set have been eliminated, we are left with a problem of the same form, but on two sets each containing N/2 elements. Thus $T(N) = T(N/2) + O(1) = O(\log N)$. □

We now generalize this result to allow N sets instead of two.

The Cartesian sum X + Y of two sets of N real numbers is the multiset whose $N^2$ elements are sums of pairs of elements taken from the sets X and Y. Thus $X+Y = \{x_i+y_j \mid 1\le i,j\le N\}$. Even though X+Y contains $N^2$ elements, these are completely determined by the 2N elements of X and Y. Can questions about X+Y be answered substantially faster than for sets that are not of this special form? The answer varies, depending on the problem. Sorting X+Y has been investigated in [Fredman (75) and Harper (75)] and, in a reasonable model of computation, $O(N^2 \log N)$ is a lower bound on the time sufficient to sort. The situation is considerably brighter for medians -- any given order statistic in X+Y can be found in O(N log N) time and any element can be ranked in linear time.

Theorem 3.4 (Jefferson, Shamos, Tarjan) *The median of X + Y can be found in O(N log N) time.*

Proof: We give an algorithm which successively discards elements of X + Y. Sort the sets X and Y initially, so that $x_i = x_{(i)}$ and $y_i = y_{(i)}$, i=1,...,N . This can be done in O(N log N) operations. We view the elements of X + Y as the entries of a matrix and maintain, for each row i, two pointers u(i) and $\ell(i)$, delimiting those elements of the row that are still under consideration. At any stage, the candidate set of medians is $C = \{x_i + y_j \mid 1 \le i \le N, \ell(i) \le j \le u(i)\}$. If no element of row i can be the median, then we will have $u(i) < \ell(i)$ . Initially, C = X + Y. However, at later stages in the computation, we will not necessarily have median(X + Y) equal to median(C). It is therefore important to keep track of p, the position of the global median within the subset C. At the start, $p = N^2/2$ and we take p = 1 to designate the largest element of C.

We now carry out a series of iterations, each of which eliminates from consideration a fixed fraction of the elements of C. At the beginning of each iteration, test whether $p \ge |C|/2$ . If so, select, for each i, the element $x_i + y_{j(i)}$, where $j(i) = \ell(i) + \frac{(u(i) - \ell(i))}{\sqrt{2}}$ . Assign to element $x_i + y_{j(i)}$ the weight $\frac{1}{\sqrt{2}}(u(i) - \ell(i))$. Now compute the weighted $\left(1 - \frac{1}{\sqrt{2}}\right)$-point among the $x_i + y_{j(i)}$ , using the linear weighted median algorithm of [Bentley (76)], and let the result be $x_{i_0} + y_{j(i_0)}$ . For each i such that $x_i + y_{j(i)} > x_{i_0} + y_{j(i_0)}$ , set $u(i) = \ell(i) + \frac{1}{\sqrt{2}}(u(i) - \ell(i)) - 1$. For each i such that $x_i + y_{j(i)} = x_{i_0} + y_{j(i_0)}$ , we set $u(i) = \ell(i) + \frac{1}{\sqrt{2}}(u(i) - \ell(i))$ . Since the pointers u(i) and $\ell(i)$ have been moved closer together, certain elements have been implicitly discarded. Each of these is greater than at least half of the elements of C and cannot be the global median. Now subtract from p the number of elements removed from C. The case for $p \le |C|/2$ is similar.

Each iteration requires O(N) steps and eliminates at least 1/16 of the elements of C. Thus O(N log N) operations suffice to find the median. □

With appropriate modification, the algorithm can find any percentile in X + Y, not just the median. The above procedure has been improved by [Mizoguchi (76)], who has found a way to discard more points at each iteration.

Theorem 3.5 (Tarjan) *Any algorithm which finds the median of X + Y, using only additions and binary comparisons, must require at least O(N log N) comparisons in the worst case.*

Proof: We assume that, once the median is found, no further comparisons are required to determine whether any given value $x_i + y_j$ is greater than or less than the median. (A proof which relaxes this assumption has been given by [Johnson 76]). Let (a,b) denote the number $a + b\varepsilon$, with $\varepsilon$ positive and suitably small. Let X be a permutation of the set $S_1 = \{(i,0) \mid 1 \le i \le N\}$ and let Y be a permutation of the set $S_2 = \{(i,n-i) \mid 1 \le i \le N\}$, where $N = 2k + 1$. The median of X + Y is then (k+1,0) + (k+1,k+1). Further, the number of values of j such that $x_i + y_j < x_{k+1} + y_{k+1}$ is n-i+1 if $i < k+1$, and n-i if $i > k+1$. It follows that, once the median is found, we have enough information to determine which permutation of $S_1$ constituted X, and O(N log N) binary comparisons must have been made. □

Problems on X + Y are sufficiently complex to justify geometric visualization. Working in the Cartesian plane, we may regard X to be a set of points on the x-axis and Y to be a set of points on the y-axis. The set X + Y can be viewed as the intersections of the 2N lines that are determined by these points (Figure 3.1). In the rectilinear, or $L_1$-metric, the distance between points $p_1 = (x_1,y_1)$ and $p_2 = (x_2,y_2)$ is given by

$$(3.1) \qquad d_1(p_1,p_2) = |x_1 - x_2| + |y_1 - y_2| \quad .$$

Thus the quantity $x_i + y_j$ is the rectilinear distance from the origin to the point $(x_i,y_j)$. The locus of points at some constant distance from the origin in the rectilinear metric is a straight line of slope -1. Any point below such a line is closer to the origin than any point above that line. Finding the median of X + Y is equivalent to finding a line of slope -1 such that half of the points of the set lie below the line. The rank of an element $x_i + y_j$ in X + Y is the number of points lying below a line of slope -1 that passes through $(x_i,y_j)$. The equation of this line has the simple form $y = -x + x_i + y_j$.

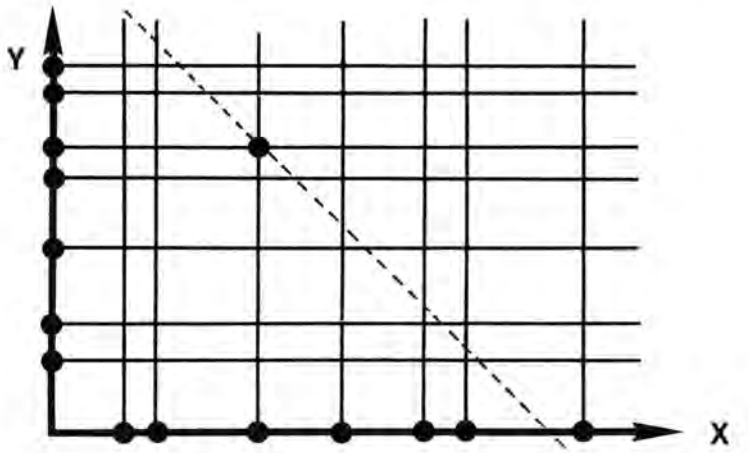



Figure 3.1. The median of $X + Y$ .

If the elements of X and Y are already sorted, the array of sums $x_i + y_j$ forms a Young tableau because it is monotonic in each row and column. Given a number c, whether there exist an $x_i$ and a $y_j$ satisfying $x_i + y_j = c$ can be determined in linear time. (See [Knuth (73) for material on searching Young tableaux.) Furthermore, the rank of $(x_i,y_j)$ can be found in linear time.

While the statistical applications of problems on $X + Y$ will be discussed in section five, it is appropriate to examine a geometric example here. In studying aggregates of points in the plane, the notion of the size or spread of a cluster arises [Meisel (72)]. The N points of a set determine $O(N^2)$ interpoint distances. Table 3.1 gives upper and lower bounds on computing various functions of these distances in the Euclidean metric in one and two dimensions.

| Interpoint Distance | Upper Bound (1-dim) | Upper Bound (2-dim) | Reference |
|---|---|---|---|
| MAXIMUM | O(N)* | O(N log N) | [Shamos (75a)] |
| MINIMUM | O(N log N)* | O(N log N)* | [Shamos (75c)] |
| MEAN | O(N log N)* | $O(N^2)$* | [Shamos (76)] |
| MEDIAN | O(N log N)* | $O(N^2)$ | see below |

*upper bounds marked with an asterisk are also known to be lower bounds.

Table 3.1. Bounds on Computing Various Functions of Interpoint Distance

Theorem 3.6 *The median interpoint distance determined by a set of N points in one dimension can be found in O(N log N) time.*

Proof: We note that, in one dimension, Euclidean and rectilinear distance are identical. The median interpoint distance is the median of the quantities $\{abs(x_i - x_j) \mid i \neq j\}$. But this is just the median of those elements of the Cartesian sum $X + (-X)$ which are positive. This can be found in O(N log N) time by the algorithm of Theorem 3.4 if the pointers u(i) and $\ell$(i) are initialized such that only positive elements are examined. □

We are now equipped to begin the analysis of some statistical algorithms.

## 4. Tests of Independence

In order to illustrate the close connection between geometric and statistical algorithms we will begin by examining some simple tests of correlation between two variables. Suppose that observations of two characteristics, X and Y, are made on each of N individuals. We wish to examine the hypothesis that the variables are independent; that is, large values of X are not associated with large values of Y and small values of X are not associated with small values of Y. Strictly speaking, X and Y are uncorrelated (independent) if

$$(4.1) \qquad \text{Prob}\{X \leq a \text{ and } Y \leq b\} = \text{Prob}\{X \leq a\} \cdot \text{Prob}\{Y \leq b\} \quad .$$

Since each observation $(x_i, y_i)$ of the variables X and Y can be viewed as a point in the X-Y plane, it is natural to expect that questions regarding the distribution of X and Y can be posed as geometric problems. The simple act of identifying an observation with a coordinate point establishes the connection between geometry and statistics. We will see that, in many statistical procedures, computing the test statistic is, in fact, a purely geometric matter.

The Olmstead-Tukey test [Olmstead (47)] judges the independence of X and Y by examining outlying points in the scatter diagram of the $(x_i, y_i)$. (See Figure 4.1) The test procedure is as follows: Let $x_m$ and $y_m$ be the median values of the $x_i$ and $y_i$, respectively, and draw the lines $x = x_m$ and $y = y_m$. These lines divide the plane into four quadrants, of which the upper right and lower left are regarded as positive, the others negative. Beginning at the point with largest x-value, proceed to the left, counting the

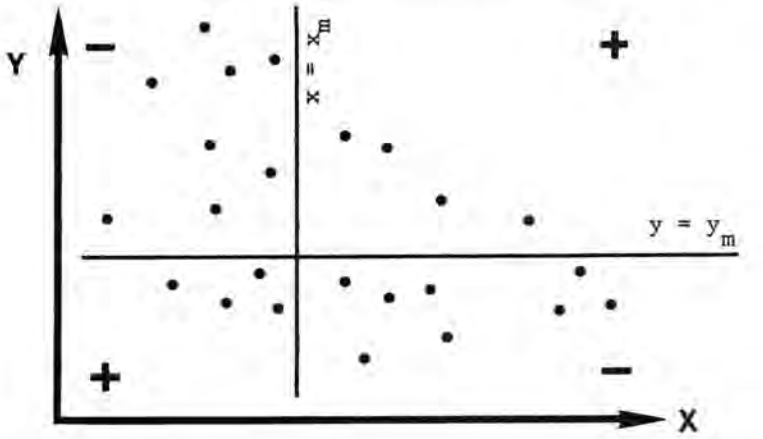



Figure 4.1. The Olmstead-Tukey Test

number of points passed over before one is forced to cross the horizontal median line $y = y_m$ . This count is given a positive sign if the points passed over lie in a positive quadrant, otherwise it is given a negative sign. In Figure 4.1, the signed count is -3 . The counting process is repeated from left to right, down from the top, and up from the bottom. The Olmstead-Tukey test statistic $T_4$ is the absolute value of the algebraic sum of the four values thus obtained. In our example, $T_4 = 11$ . One can readily see that large values of the test statistic indicate a correlation between X and Y, either positive or negative, while values near zero suggest independence. Quantitatively, as applied in practice, the null hypothesis of independence is tested by comparing the computed value of $T_4$ against a table of its theoretical null distribution. As with all of the tests to be discussed in this paper, we are not concerned with assembling statistical tables, which may be done once for all Time, but with the complexity of computing the test statistic, a process that is repeated for every sample.

It can be readily seen how to compute $T_4$ in O(N log N) time by simply following the instructions given in the description of the test above. That such a simple procedure needs this much time is surprising, particularly in view of the fact that it was originally developed as a quick test, one that engineers could easily perform at the bench. Actually, for reasonable amounts of data, it <u>can</u> be performed quickly -- in linear time, by sorting the points visually. Once the points are plotted, they can be read off by eye in sorted order in linear time. On a computer, however, this

perceptual trick is not available, an O(N log N) time is needed to sort. If we choose not to follow the original instructions, though, a fast algorithm can be obtained:

Theorem 4.1 *The Olmstead-Tukey statistic can be computed in linear time.*

Proof: It is unnecessary to examine the data in abscissa order. The following procedure suffices: The median lines $x_m$ and $y_m$ can be found in linear time. Find $p_{max}$, the index of the point having largest x-coordinate. Without loss of generality, we may assume that its y-coordinate exceeds $y_m$. Now determine, in a single pass through the points, that point having largest x-coordinate whose y-coordinate is less than $y_m$. The x-rank (with 1 indicating largest) of this point in the collection of all x-values is the magnitude of the contribution to $T_4$ in the right-left direction. A similar procedure can be applied in the left-right, up-down, and down-up directions. □

Thus the Olmstead-Tukey "quick" test, which is fast on paper, appears initially to suffer when implemented on a computer but turns out to be efficient after all.

To illustrate tests based wholly on ranks, we will study the Spearman, Kendall, and Hoeffding statistics. The principle underlying these tests is quite elementary. Suppose that the points have been sorted by x-coordinate, so that $x_i < x_j$ iff $i < j$. If the y-coordinates are now ordered, their indices will induce a permutation of $1,\ldots,N$. If X and Y are independent, then all permutations are equally likely. If, however, X and Y are correlated, the $y_i$ will tend to be monotonic, and the induced permutation will not be random. The tests we are considering assign numerical values to this departure from randomness.

Let $r(x_i)$ denote the rank of $x_i$ among all the $x_i$ and similarly for $r(y_i)$. The approach of Spearman was to work with the differences in rank $d_i = r(y_i) - r(x_i)$. The Spearman rank correlation coefficient, which varies between -1 and +1, is given by

$$r_S = 1 - \frac{6 \sum_{i=1}^{N} d_i^2}{N^3 - N} \quad . \tag{4.2}$$

Since all of the ranks $r(x_i)$ and $r(y_i)$ can be found by sorting, the Spearman rank correlation coefficient can be computed in O(N log N) time.

Kendall's rank correlation appears similar, but it is not

immediately clear how to compute it quickly. Following [Kendall (55)] we order the points by X and define $I$ to be the number of times, in the resulting sequence of y's, that a Y-observation is preceded by a larger Y-observation. Likewise, let $T$ be the number of times a Y-observation is followed by a larger Y-observation. The Kendall statistic S is given by $S = T - I$. It is clear that, by definition, $I$ is the number of inversions in the induced Y-permutation and, as we saw in section three, can be calculated in O(N log N) time. It is also easy to show [Bradley (68)] that an alternative expression for S is

$$S = \binom{N}{2} - 2I \quad , \tag{4.3}$$

and thus S can be found in O(N log N) time. The Kendall rank correlation coefficient is defined by

$$\tau = \frac{2S}{N(N-1)} \tag{4.4}$$

and we have

Theorem 4.2 *The Kendall rank correlation coefficient can be computed in O(N log N) time and this is optimal.*

Proof: Optimality follows from the fact that counting inversions requires O(N log N) operations. □

The Hoeffding test for independence [Hoeffding (48)] is based on the ranks $r(x_i)$ and $r(y_i)$ and on a quantity $c_i$ which is defined to be the number of points $(x_j,y_j)$ for which both $x_j < x_i$ and $y_j < y_i$. Given the $r(x_i)$, $r(y_i)$, and the $c_i$, the Hoeffding statistic, which is an algebraic expression in these quantities, can be computed in linear time. We already know that the ranks require O(N log N) time. The remaining question is whether the $c_i$ can be found that fast. At this point, a geometric view clarifies things considerably. (See Figure 4.2) Each point $p_i = (x_i,y_i)$ determines a rectangle whose vertices are $(0,0)$, $(x_i,0)$, $(0,y_i)$ and $(x_i,y_i)$. (Since the statistic is based only on ranks, it is translation-invariant and we may assume that all points lie in the first quadrant.) Then $c_i$ is the number of other points that lie in the rectangle determined by $p_i$. There are N values of $c_i$ to be determined.

Assume that the points have been ordered by x-coordinate, and label them $1,\ldots,N$ according to their x-ranks $r(x_i)$. The projections of these points on the y-axis induces a permutation $a_1,\ldots,a_N$ of $1,\ldots,N$. Given a point $(x_i,y_i)$, a

new point $(x_j,y_j)$ lies in the rectangle determined by $p_i$ iff $x_j < x_i$ and $y_j < y_i$. Since the $x_i$ are already sorted, this means that we require $j < i$ and $a_j < a_i$. The number of points satisfying these conditions, $c_i$ is just N-1 minus the inversion index of $a_i$. (See Figure 4.3) And we have

Theorem 4.3 *The Hoeffding statistic can be computed in O(N log N) time.*

Proof: Sorting the $x_i$ originally can be done in O(N log N) time. The y-permutation $a_1,\ldots,a_N$ can be found in O(N log N) time and all of its inversion indices can also be found in O(N log N) time. □

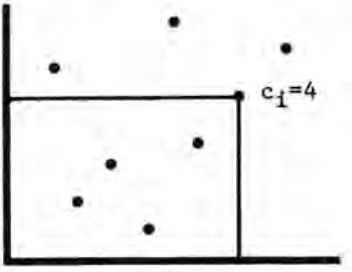



Figure 4.2.
The Hoeffding Test

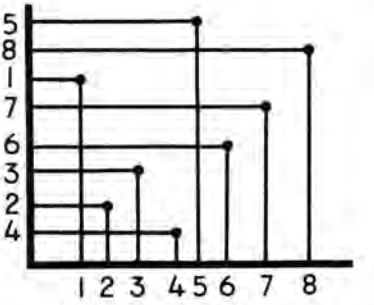



Figure 4.3
Counting Inversions

In one dimension, the empirical cumulative distribution function can be viewed as a count, for each $x_i$, of the number of points that lie to its left on the real line. These counts can all be computed in O(N log N) time by sorting the $x_i$. In the plane, the empirical CDF consists precisely of the quantities $c_i$ defined above, so Theorem 4.3 states that the two-dimensional CDF can also be computed in O(N log N) time.

There is a connection between the Hoeffding test, the empirical CDF, and finding the minima and maxima of a set of vectors. Given two k-vectors $(u_1,\ldots,u_k)$ and $(v_1,\ldots,v_k)$, we say that $(u_1,\ldots,u_k) \leq (v_1,\ldots,v_k)$ iff $u_i \leq v_i$, $1 \leq i \leq k$. A vector V is minimal if there is no vector U in the set under consideration satisfying $U \leq V$. The problem of find-finding the maxima and minima of a set of N k-vectors has been studied in [Kung (75)]. We note here that, in the plane, minimal vectors are those for which $c_i = 0$. Thus finding maxima and minima is reducible to computing the empirical CDF.

Of a somewhat different flavor is the chi-square test in a contingency table, described in [Conover (71)]. Here we do not examine inversions or relative ranks, but concentrate instead on the distribution of the sample points. Suppose we construct $r-1$ horizontal lines with the property that $N/r$ of the points lie between each pair of consecutive lines and similarly construct $c-1$ vertical lines to divide the set into slices each containing $N/c$ points. The lines partition the plane into $rc$ rectangles. (In Figure 4.4, $c = 4$ and $r = 3$.) If the $x_i$ and $y_i$ are independent, then the expected number of points falling in each rectangle is $N/rc$. The chi-square statistic $\chi^2$ measures departure from independence. Let $p_{ij}$ be the observed number of points that lie in the rectangle in row i and column j. The test statistic is then defined by

$$(4.5) \qquad \chi^2 = \frac{rc}{N} \sum_{i=1}^{r} \sum_{j=1}^{c} \left( p_{ij} - \frac{N}{rc} \right)^2 .$$

Theorem 4.4 *The chi-square statistic for independence in an $r \times c$ contingency table can be computed in $O(N \log rc)$ time.*

Proof: The horizontal lines are just the r-tiles of the y-values; the vertical lines are c-tiles of the x-values. By Theorem 3.1, these can be found in $O(N \log r + N \log c) = O(N \log rc)$ time. Once the partitioning into rectangles has been performed, examine each point individually. The row in which a point lies can be found in $O(\log r)$ time by binary search. Likewise, the column can be found in $O(\log c)$ time and the appropriate count $p_{ij}$ can be incremented. For N points, the total time required is $O(N \log rc)$. □

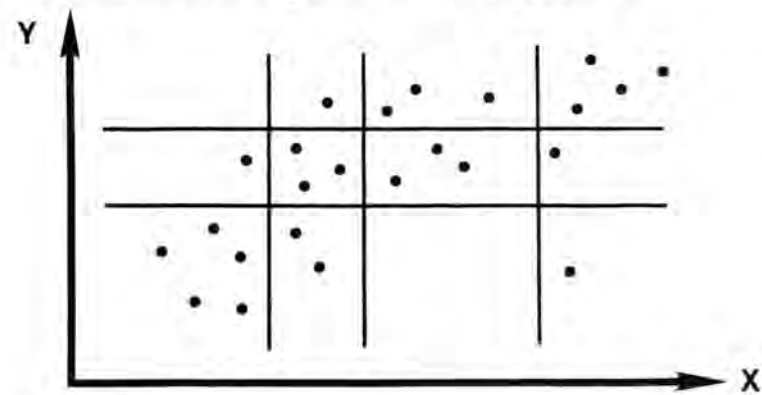



Figure 4.4. An $r \times c$ contingency table.

It appears, from these results, that efficient implementation of even the simplest statistical tests requires moderately sophisticated methods of algorithm design. But the challenge of computational statistics is not to squeeze the last microsecond from a data analysis routine; it is to understand what makes some statistical quantities more difficult to compute than others and to relate this difference to relevant measures of statistical efficacy. The computer scientist will be interested in the amount of time and storage space used in computing a statistic; the statistician is concerned with other indicators of performance: The efficiency of a test reflects how large a sample must be taken in order to achieve a given level of significance. The power of a test is the probability that it will reject a hypothesis that is, in fact, false. The robustness of a test is the quality of remaining insensitive to the underlying distribution from which the sample is chosen. One has the feeling that a statistic which is hard to compute is somehow "better" than one which is easy. We will examine in the next section some preliminary evidence that tends to support this notion.

## 5. Estimates of Location

One drawback of tests of hypothesis is that the hypothesis is often so unreasonable that very little statistical justification is needed to reject it. The statistician is usually interested in a quantitative appraisal of the effect of one variable upon another and is not satisfied with the qualitative statement that there is or is not some dependence. A great deal of statistical thought and ingenuity has gone into the development of procedures for obtaining these estimates and we will use a simple example to study their complexity.

Suppose that each of N patients with fevers are given a new brand of aspirin whose effectiveness is to be tested. Let the temperature of patient i before the administration of aspirin be $x_i$ and his temperature after be $y_i$ . We take as our model of aspirin action that it reduces the temperature of each patient by a constant amount, subject to random fluctuations. We wish to estimate the value of this constant from the given data. This problem is referred to as the problem of paired observations or "one-sample location". [Hollander (73)]. In short, we have N pairs $(x_i,y_i)$ and we assume that the observed shifts $z_i = y_i - x_i$ obey the law

$$(5.1) \qquad z_i = \theta + e_i \ ,$$

where the constant $\theta$ is the effect due to treatment and the $e_i$ are mutually independent random variables each with a continuous distribution that is symmetric about zero. The problem is to estimate $\theta$ from the $z_i$ .

Enough estimates have been proposed to fill an entire book [Andrews (72)]. Below we analyze the complexity of a number of the more familiar ones, roughly in order of increasing robustness. As before, we let $z_{(i)}$ denote the ith smallest of the $z_i$ .

a. The Arithmetic Mean. An obvious estimator of $\theta$ is the arithmetic mean of the $z_i$ . This can be computed in linear time but has the disadvantage that it is severely affected by outliers (extremes of the data).

b. Trimmed Mean. One proposal that alleviates some of the difficulties of the mean is to discard the highest and lowest $\alpha$-fractions of the data and take the mean of the remainder. This is the $\alpha$-trimmed mean, given by

$$\hat{\theta} = \frac{1}{(1-2\alpha)N} \sum_{i=\alpha N}^{(1-\alpha)N} z_{(i)} \quad . \tag{5.2}$$

Since the $\alpha$ and $1-\alpha$ order statistics can be found in linear time [Blum (72)], the trimmed mean can be computed in linear time. (The median is the special case $\alpha = 1/2$.)

In higher dimensions, the analog of trimming has been called "peeling" by Tukey and consists of successively stripping away extreme points of the convex hull of the data until a certain fraction of the points remains. In the plane, an $O(N^2)$ algorithm for peeling appears in [Shamos (75b)]. $O(N \log N)$ is a lower bound in more than one dimension but no algorithm that achieves this bound is known.

c. Gastwirth Estimators [Gastwirth (66)]. These are a class of estimators based on k-tiles, in which weights are attached to the various order statistics, depending on their distance from the median.

$$\hat{\theta} = \sum_{i=1}^{k-1} w_i \, z_{(iN/k)} \quad . \tag{5.3}$$

By Theorem 3.1, the order k Gastwirth estimator can be computed in $O(N \log k)$ time. This improves the naive method of finding the quantiles separately, which would require $O(Nk)$ time.

d. Bickel-Hodges Estimator [Bickel (67)]. This is the median of pairwise means of symmetric order statistics:

$$(5.4) \qquad \hat{\theta} = \frac{1}{2}\,\text{median}\{(z_{(1)}+z_{(N)}),\ldots,(z_{(k)}+z_{(N-k+1)})\},$$

where k runs from 1 to N/2 . (5.4) can be computed in linear time once the data have been sorted, so O(N log N) is an upper bound. It is not known whether improvement is possible.

e. Hodges-Lehmann Estimator [Hodges (63)]. This is the median of all pairwise averages of the $z_i$ :

$$(5.5) \qquad \hat{\theta} = \frac{1}{2}\,\text{median}\ \{\ z_i + z_j \mid 1 \le i,j \le N\ \}\ .$$

While this estimator has robust properties, it is regarded as being too difficult to compute, as only $O(N^2)$ algorithms are known. This has prompted attempts to develop estimators with similar characteristics, but amenable to fast calculation [Antille (74)]. However, we have

Theorem 5.1 *The Hodges-Lehmann estimator can be computed in O(N log N) time.*

Proof: We note that (5.5) is just one-half the median of the set $Z + Z$ , where $Z = \{\ z_i \mid 1 \le i \le N\ \}$ , and Theorem 3.4 applies directly. □

Comprehensive treatments of the modern estimators can be found in [Andrews (72)] and [Huber (72)].

Beyond the crude observation that the "good" (robust) estimators cost more to compute than the "bad" ones, can anything quantitative be said? It is conceivable that there exists a point above which no additional computational work will improve the performance of an estimator; possibly this is even a polynomial quantity. How are the above results to affect the practical and mathematical life of the statistician? The utility of fast algorithms seems obvious, but a more exciting possibility is that new statistical procedures may be based on what can be computed quickly, or that certain tests will be discarded because they can be performed too quickly (and hence might exhibit poor statistical behavior). As of now, however, no relation is known between computational complexity and any statistical measure.

## 6. The Complexity of Regression

Regression is a technique used to model dependencies among variables. One hypothesizes a functional form of the dependence, which normally involves several undetermined coefficients. The problem of regression is to find values for these parameters such that the resulting function most closely represents the observed data. In this light, regression can be viewed as best approximation in a subspace, the geometry of which is well-studied [Rice (64)]. We will concentrate on the geometric aspects of regression in order to develop fast algorithms.

Before performing regression, though, we are obliged to choose (1) the subspace of allowable approximating functions, and (2) the error norm that is to be minimized. For simplicity we will deal chiefly with linear regression in two variables, but under a variety of error measures. That is, we are trying to fit the parameters m and b to the linear model $y = mx + b$, given N observations $(x_i, y_i)$.

a. Least-squares (Gaussian) Regression.
This is best approximation in the Euclidean norm :

$$\min_{m,b} \sum_{i=1}^{N} (y_i - mx_i - b)^2 \tag{6.1}$$

The constants defined by (6.1) can be computed in O(N) time from the expressions

$$b = \frac{(\Sigma y_i)(\Sigma x_i^2) - (\Sigma x_i)(\Sigma x_i y_i)}{N(\Sigma x_i^2) - (\Sigma x_i)^2}$$

$$m = \frac{N(\Sigma x_i y_i) - (\Sigma x_i)(\Sigma y_i)}{N(\Sigma x_i^2) - (\Sigma x_i)^2} \tag{6.2}$$

In fact, the computation can be performed on-line in O(N) time such that the slope and intercept of the new regression line are available as each point is read. Its speed notwithstanding, however, least-squares regression places great weight on outlying points, precisely the ones that may be suspect. It is interesting to note that, despite the continuous aspect of (6.1), no one is tempted to use gradient methods to find its minimum, since the discrete form (6.2) is available. This contrasts with the case of $L_1$-regression.

b. $L_1$-Regression

While least-squares regression minimizes the sum of the squares of the residuals, $L_1$-regression minimizes the sum of the magnitudes of the residuals:

$$(6.3) \qquad \min_{m,b} \sum_{i=1}^{N} | y_i - mx_i - b | \quad .$$

Unfortunately, no efficient methods for minimizing (6.3) are known. [Barrodale (66)] employs a linear programming approach, while an $O(N^2 \log N)$ algorithm appears in [Rivlin (69)].

Theorem 6.1 *A best $L_1$ linear approximation to N points in the plane can be found in $O(N^2)$ time and $O(N)$ space.*

Proof: It is known that a best $L_1$ approximation in the plane passes through at least two points of the set [Rice (64)]. Trying all pairs of points in order to find the best line would require $O(N^3)$ time, but we can reduce this by solving N instances of a restricted problem. Consider the best $L_1$-line that is constrained to pass through the origin. Then b=0 and (6.3) becomes

$$(6.4) \qquad \min_{m} \sum_{i=1}^{N} | y_i - mx_i | \quad .$$

Define $m_i = y_i/x_i$ and this becomes

$$(6.5) \qquad \min_{m} \Sigma \, | m_i x_i - m x_i | \;=\; \min_{m} \Sigma \, |x_i| \, |m_i - m| \quad .$$

Notice that the second form of (6.5) is an instance of (2.3) with weights $w_i = |x_i|$, so it can be minimized in $O(N)$ time. The general problem can now be solved in $O(N^2)$ time by taking each point in turn as the origin of coordinates. The best of the N lines so obtained is a global optimum. □

Even though the variables m and b in (6.3) are continuous, an optimum can be found by discrete methods. The algorithm of Theorem 6.1 is clumsy, however, because no information gained in the solution of one subproblem is used in solving any of the others. Until a better algorithm is developed, iterative methods will remain preferable. Furthermore, no non-trivial lower bound is known.

c. $L_\infty$ Regression.
The Chebyshev, $L_\infty$, or minimax approximation minimizes the magnitude of the largest residual:

$$(6.6) \qquad \min_{m,b} \max_{i} \; | \, y_i - mx_i - b \, | \quad .$$

While regression in this norm does not appear to have good statistical properties, we mention it here because the best known algorithm for this problem requires $O(N^3)$ time. [Rice (64)] shows that a best Chebyshev approximation on N points is also a best Chebyshev approximation on some three of the points, so it suffices to examine all triples. Such a rote method cries out for improvement.

d. The Theil Estimator.
Given N points $(x_i,y_i)$, define $s_{ij} = (y_j - y_i)/(x_j - x_i)$, the slopes of the lines determined by pairs of points, and we require $i \neq j$. The expression (6.2) for the slope $m$ of the least-squares regression line is actually a weighted average of the $s_{ij}$. [Theil (50)] recommends the median of the $s_{ij}$ as a distribution-free estimator of slope because it is less sensitive to outliers. No efficient method is known for computing this median slope and straightforward application of linear selection requires $O(N^2)$ time and space. [Yuval (76)] has shown that the number of slopes $s_{ij}$ lying in any given interval can be found in $O(N \log N)$ time (by counting inversions!) and this appears to lead to a better algorithm. The median slope is one of a growing class of problems whose natural formulations involve a large number of intermediate quantities (the slopes), but for which there are only $O(N)$ inputs (the coordinates of the points) and a constant number of outputs (the median). Very little is known about lower bounds on these problems.

e. Isotonic Regression.
One way to relax the requirement that the regression function be linear is to restrict it merely to be monotone. This is realistic in a number of statistical problems [Barlow (72)]. The question the arises of how to perform such a regression in the various norms. Least-squares isotonic regression turns out to reduce to the problem of finding the convex hull of a set of points in the plane, which can be done in $O(N \log N)$ time in general, or in $O(N)$ time if the points are already ordered by abscissae. [Shamos (75b)] contains a fuller discussion.

## 7. Difficult Problems

Not all statistical quantities can be computed rapidly. To show this we will prove that two "simple" problems are NP-hard. For background material the reader is referred to [Karp (72)]. Informally, a problem is NP-hard if its membership in P implies that P = NP . A problem is NP-complete if it is NP-hard and also in NP. Problems that are not in P, the class of problems solvable in polynomial time, are regarded as computationally intractable, and there is a mountain of evidence to indicate that P ≠ NP [Garey (76)]. Thus, showing that a problem is NP-hard is tantamount to showing that it cannot be solved exactly in any reasonable time. On the brighter side, polynomial-time approximate solutions to these problems can sometimes be obtained.

Consider a set of integers and define the weight of a subset to be the sum of its elements. The partition problem is to determine, given a set of N integers, whether it can be partitioned into two subsets of equal weight.

Theorem 7.1 [Karp (72)] *The partition problem is NP-complete.* □

We will show other problems to be NP-hard by proving that the partition problem is reducible to them.

A k-clustering of a set is a partition of it into k disjoint subsets. The clustering problem is to determine which partition minimizes a given function over k-clusters. This function is called the error function f(P). Thus we want

$$\min_{P \text{ a partition}} f(P) \quad . \tag{7.1}$$

Theorem 7.2 *The clustering problem is NP-hard even for some polynomial-time computable error function and just two clusters.*

Proof: Let the set to be clustered consist of N integers, and call the two clusters of a partition $c_1$ and $c_2$ . Let $f(P) = |\Sigma c_1 - \Sigma c_2|$, that is, the difference of the sums of the elements in the two clusters. Suppose there exists a P such that f(P) = 0 . Since f(P) is non-negative, this P must be found by the clustering algorithm. But then P is a solution to the partition problem above. Similarly, if no such P exists, then the non-existence of a solution to the partition problem is confirmed. Hence the partition problem is reducible to clustering. □

In many practical examples, clustering is not hopeless because the error function has a special form. Theorem 7.2 puts us on notice, however, that we must not always expect to be able to perform clustering quickly.

The randomization test for matched pairs [Siegel (56)] is designed to test the hypothesis that $\theta = 0$ in the model of (5.1). In general, when we observe the $z_i = y_i - x_i$, some will be positive and some negative. Define T to be the sum of the positive $z_i$, with $T^*$ indicating the value of T that is actually observed in the sample. Now consider the $2^N$ possible assignments of + and - signs to the $z_i$. The test statistic is the number of such assignments that give a value for T that is less than $T^*$, the value actually observed.

Theorem 7.3 *The randomization test for matched pairs is NP-hard.*

Proof: Let I be a set of positive integers whose sum is S. Perform the randomization test on these numbers with $T^* = S/2$. If there is no partition of I into equal-weight subsets, then the number of assignments giving $T < T^*$ is exactly $2^{N-1}$, where $N = |I|$. If some partition into equal-weight subsets exists, then at least two assignments of sign will give $T = T^*$ and the test statistic will be less than $2^{N-1}$. In this way, the randomization test can be used to solve the partition problem. □

Note that Theorem 7.3 does not imply that testing the hypothesis $\theta = 0$ is NP-hard, but that the randomization test (one of many) is NP-hard.

## 8. Average-case Analysis of Geometric Algorithms

Having seen how geometric algorithms can be turned into statistical ones, we will now reverse direction and use results from probability theory to assist in analyzing their expected-time behavior. Geometrical probability, which deals with the properties of random geometric objects, is a subject that is difficult and fraught with paradoxes [Kendall (63)]. To illustrate some of the troubles that appear, we will examine a problem that arose in the average-case analysis of an algorithm to find the smallest circle enclosing a set of points in the plane [Shamos (75b)].

Two points determine a circle if they are the endpoints of a diameter. Given three points in the plane, what is the probability the circle determined by the two that are farthest apart encloses the third? In Figure 8.1, the two farthest

points are labeled P and Q, while the third point is X. We want to know the probability that X lies within the circle whose diameter is PQ, subject to the constraint that $XP \le PQ$ and $XQ \le PQ$ (or P and Q would not be the two farthest points). Because of these constraints, X must lie in the circle centered at P with radius PQ. It also must lie in the circle centered at Q with radius PQ. Thus X must lie in the lune that is the intersection of the two circles. (See Figure 8.2) The required probability is thus the ratio of the area of the PQ circle to the area of the lune. These areas are given in Figure 8.2.

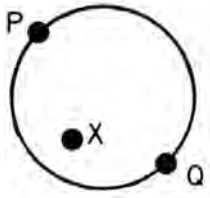
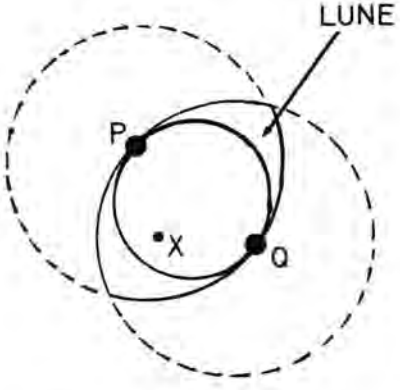



Figure 8.1
What is the probability that X lies in the circle?




$$\text{AREA(circle PQ)} = \pi$$

$$\text{AREA(lune)} = \frac{8\pi}{3} + 2\sqrt{3}$$

Figure 8.2. The probability as a ratio of areas.

Thus the required probability is given by

$$\text{(8.1)} \qquad \text{PROB} = \frac{\pi}{\frac{8\pi}{3} + 2\sqrt{3}} \approx 0.639$$

The only difficulty with the above analysis is that it is completely erroneous! We have tacitly assumed that X is uniformly distributed in the lune, but points can be chosen uniformly in the plane only within some bounded set. The probability calculation is correct only if, for every choice

of P and Q, the lune of Figure 8.2 lies entirely within this set. Unfortunately, there is no plane set having this property. For the unit square, the probability that X falls in the PQ circle is approximately 0.47 (obtained by computer simulation). It is to no avail to allow the side of the square to go to infinity, for the problem is scale-invariant and edge effects do not disappear in the limit. It is difficult to imagine a more simply-stated problem in geometrical probability; even so, great care is required in order to obtain correct and meaningful results. The papers of [Karp (76)] and [Rabin (76)] demonstrate the power of stochastic reasoning applied to geometric problems.

The problem of finding the convex hull of N points in the plane has received a good deal of attention recently, and a variety of algorithms are known, including one due to [Graham (72)] which requires O(N log N) time. A different algorithm [Jarvis (73)] uses time O(hN), where h is the number of extreme points of the final convex hull. Since all of the points of the set may lie on the hull (for example, the vertices of a convex polygon), Jarvis' algorithm has a worst-case performance of $O(N^2)$. If, however, $h < \log N$, then it will be faster than Graham's. Normally we do not known h in advance, but if information is available concerning the distribution from which the points have been obtained, the expected value of h can be computed.

Theorem 8.1 [Rényi (63)] *If N points are chosen independently and uniformly in a bounded convex plane figure F, the expected value of h, the number of vertices of their convex hull, is given by*

$$E(h) = \frac{2r}{3} \ln N + O(r), \quad \textit{if F is a convex r-gon,} \tag{8.2}$$

$$E(h) = O(N^{1/3}), \quad \textit{if F has a continuously turning tangent.} \ \square \tag{8.3}$$

Theorem 8.2 [Rényi (63)] *If N points are chosen from a normal distribution in the plane, the expected number of points on their convex hull is given by*

$$E(h) = O(\sqrt{\ln N}) \ . \ \square \tag{8.4}$$

Thus, for points drawn from a normal distribution, the algorithm of Jarvis is asymptotically faster than that of Graham. For points chosen uniformly in a circle, however, its performnce is $O(N^{4/3})$, while Graham's algorithm is always O(N log N).

Note that the expected number of points on the hull of the set depends on the shape of the figure from which the points are drawn, but not on its area.

A different sort of probabilistic analysis has proven useful in designing approximation algorithms for NP-complete geometric problems [Karp (76)]. Even though finding the length of a traveling salesman path through N points in the plane is NP-hard, we have a very good idea how long such a path will be:

Theorem 8.3 [Beardwood (59)] *Given N points chosen independently and uniformly in a bounded plane region of area A,*

1. *There exists a constant $c_1$ such that the expected length of the minimum spanning tree on the N points is asymptotic to $c_1\sqrt{AN}$ as $N \to \infty$.*

2. *There exists a constant $c_2$ such that the expected length of a traveling salesman path on the N points is asymptotic to $c_2\sqrt{AN}$ as $N \to \infty$, and $c_2 \approx 0.75$.* □

The constant $c_1$ in Theorem 8.3 has been determined to be approximately 0.68 [Gilbert (65)]. Note that the expected lengths depend on the area of the region but not on its shape.

A large amount of work has been done on geometrical probability, but little has found its way into the analysis of algorithms [Moran (66), Moran (69), Little(74)]. We anticipate that a good many average-case results will be obtained once these techniques become more widely known.

## 9. Summary

The interplay between statistics, geometry, and computer science is very close. Each discipline contributes problems to the others and provides methods for their solution. Many statistical problems raise new questions in the domain of discrete algorithms, while recently-discovered fast algorithm techniques apply directly to the computation of classical statistical quantities. With its complement of techniques, tricks, lower bounds, and NP-hard problems, computational statistics appears to be a rich new research area that will produce both practical results and theoretical insight. The interdisciplinary nature of the subject is enhanced by the use of both geometry and probability in the expected-time analysis of algorithms.

10. Acknowledgements

We wish to thank Dave Jefferson and Larry Rafsky for numerous engaging and useful conversations. Robert Tarjan provided a correction to the algorithm of Theorem 3.4, as well as the statement and proof of Theorem 3.5.

References

Aho (74) Aho, A., Hopcroft, J., and Ullman, J. *The Design and Analysis of Computer Algorithms*. Addison-Wesley (1974).

Andrews (72) Andrews, D.F., Bickel, P.J., Hampel, F.R., Rogers, W.H., and Tukey, J.W. *Robust Estimation of Location: Survey and Advances*. Princeton University Press (1972).

Antille (74) Antille, A. *A Linearized Version of the Hodges-Lehmann Estimator*. Ann. Math. Stat. 2:6 (197 1308-1313.

Barlow (72) Barlow, R., Bartholomew, D., Bremner, J., and Brunk, H. *Statistical Inference Under Order Restrictions*. Wiley (1972).

Barrowdale (66) Barrowdale, I. and Young, A. *Algorithms for Best $L_1$ and $L_\infty$ Linear Approximations on a Discrete Set*. Numer. Math. 8(1966), 295-306.

Beardwood (59) Beardwood, J., Halton, J.H., and Hammersley, J.M. *The Shortest Path Through Many Points*. Proc. Camb. Phil. Soc. 55(1959), 299-327.

Bentley (76) Bentley, J.L. and Shamos, M.I. *A Linear-Time Weighted Median Algorithm and Applications*. In preparation, (1976).

Bickel (67) Bickel, P.J. and Hodges, J. *The Asymptotic Theory of Galton's Test and a Related Simple Estimate of Location*. Ann. Math. Stat. 38(1967), 73-89.

Blum (72) Blum, M., Floyd, R.W., Pratt, V., Rivest, R., and Tarjan, R.E. *Time Bounds for Selection*. JCSS 7:4 (1972), 448-461.

Bradley (68) Bradley, J.V. *Distribution-Free Statistical Tests*. Prentice-Hall (1968).

Conover (71) Conover, W. *Practical Nonparametric Statistics*. Wiley (1971).

Cotton (75) Cotton, I.W. *Remark on Stably Updating Mean and Standard Deviation of Data*. CACM 18:8(1975), 458.

Dobkin (75) Dobkin, D. and Lipton, R. *On the Complexity of Computations Under Varying Sets of Primitives*. Yale Dept. of Comp. Sci. Research Report #42 (1975).

Floyd (73) Floyd, R.W. and Rivest, R. *Expected Time Bounds for Selection.* CACM 18:3(1973), 165-172.

Fredman (75) Fredman, M. *Two Applications of a Probabilistic Search Technique: Sorting X + Y and Building Balanced Search Trees.* Proc. Seventh ACM Symposium on Theory of Computing (1975), 240-244.

Garey (76) Garey, M.R. and Johnson, D.S. *Performance Guarantees for Heuristic Algorithms: An Annotated Bibliography.* Proc. Symp. on New Directions and Recent Results in Algorithms and Complexity. Academic Press (1976).

Gastwirth (66) Gastwirth, J. *On Robust Procedures.* J. Amer. Stat. Assn. 65(1966), 929-948.

Gibbons (76) Gibbons, W. *private communication.*

Gilbert (65) Gilbert, E.N. *Random Minimal Trees.* J. SIAM 13:2(1965), 376-387.

Gonzalez (75) Gonzalez, T. *Algorithms on Sets and Related Problems.* Dept. of Information and Computing Sciences Technical Report #75-15 (1975).

Graham (72) Graham, R.L. *An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set.* Info. Proc. Letters 1(1972), 132-133.

Hammersley (64) Hammersley, J.M. and Handscomb, D.C. *Monte-Carlo Methods.* Methuen & Co. and Wiley (1964).

Hanson (75) Hanson, R.J. *Stably Updating Mean and Standard Deviation of Data.* CACM 18:1(1975).

Harper (75) Harper, L., Payne, T., Savage, J., and Straus, E. *Sorting X + Y.* CACM 18:6(1975), 347-349.

Hodges (63) Hodges, J.L. and Lehmann, E.L. *Estimates of Location Based on Rank Tests.* Ann. Math. Stat. 34 (1963), 598-611.

Hoeffding (48) Hoeffding, W. *A Nonparametric Test of Independence.* Ann. Math. Stat. 19(1948), 546-557.

Hollander (73) Hollander, M. and Wolfe, D. *Nonparametric Statistical Methods.* Wiley (1973).

Huber (72) Huber, P. *Robust Statistics: A Review.* Ann. Math. Stat. 43:4(1972), 1041-1067.

Jarvis (73) Jarvis, R.A. *On the Identification of the Convex Hull of a Finite Set of Points in the Plane.* Info. Proc. Letters 2(1973), 18-21.

Jefferson (76) Jefferson, D. *private communication.*

Johnson (75) Johnson, D.B. *Bounds on the Complexity of Kth Best Subset Problems.* Penn. State Univ. Comp. Sci. Dept. Technical Report #176, November, 1975.

Karp (72) Karp, R.M. *Reducibility Among Combinatorial Problems.* In *Complexity of Computer Computations.* Miller and Thatcher, eds. Plenum Press (1972), 85-104.

Karp (76) Karp, R.M. *The Probabilistic Analysis of Some Combinatorial Search Algorithms.* Proc. Symposium on New Directions and Recent Results in Algorithms and Complexity. Academic Press (1976).

Kendall (55) Kendall, M.G. *Rank Correlation Methods.* Hafner Publishing Co. (1955).

Kendall (63) Kendall, M.G. and Moran, P.A.P. *Geometrical Probability.* Griffin's Statistical Monographs #10. Griffin (1963).

Knuth (73) Knuth, D.E. *The Art of Computer Programming. Volume III: Sorting and Searching.* Addison-Wesley (1973).

Knuth (76) Knuth, D.E. and Yao, A.C.-C. *The Complexity of Nonuniform Random Number Generation.* Proc. Symposium on New Directions and Recent Results in Algorithms and Complexity. Academic Press (1976).

Kung (75) Kung, H.T., Luccio, F., and Preparata, F.P. *On Finding the Maxima of a Set of Vectors.* JACM 22(1975), 469-476.

Little (74) Little, D.V. *A Third Note on Recent Research in Geometrical Probability.* Advances in Applied Probability 6(1974), 103-130.

Meisel (72) Meisel, W.S. *Computer-Oriented Approaches to Pattern Recognition.* Academic Press (1972).

Mizoguchi (76) Mizoguchi, T. *Finding the Kth Pair.* Unpublished manuscript.

Moran (66) Moran, P.A.P. *A Note on Recent Research in Geometrical Probability.* J. Appl. Prob. 3(1966), 453-63.

Moran (69) Moran, P.A.P. *A Second Note on Recent Research in Geometrical Probability.* Advances in Applied Probability 1(1969), 73-89.

Olmstead (47) Olmstead, P. and Tukey, J.W. *A Corner Test for Association.* Ann. Math. Stat. 18(1947), 495-513.

Pohl (72) Pohl, I. *A Sorting Problem and Its Complexity.* CACM 15:6(1972), 462-464.

Rabin (76) Rabin, M.O. *Short Expected Time Algorithm for the Nearest Pair.* Proc. Symposium on New Directions and Recent Results in Algorithms and Complexity. Academic Press (1976).

Rényi (63) Rényi, A. and Sulanke, R. *Über die konvexe Hülle von n zufällig gewählten Punkten. I.* Z. Wahrscheinlichkeitstheorie 2(1963), 75-84.

Rice (64) Rice, J. *The Approximation of Functions. Volume I: The Linear Theory.* Addison-Wesley (1964).

Rivlin (69) Rivlin, T. *An Introduction to the Approximation of Functions.* Blaisdell (1969).

Shamos (75a) Shamos, M.I. *Geometric Complexity.* Proc. Seventh ACM Symposium on Theory of Computing (1975), 224-233.

Shamos (75b) Shamos, M.I. *Problems in Computational Geometry.* Lecture Notes in Computer Science. Springer-Verlag (1976).

Shamos (75c) Shamos, M.I. and Hoey, D. *Closest-Point Problems.* Proc. 16th Annual IEEE Symposium on Foundations of Computer Science. IEEE (1975), 151-162

Shamos (76) Shamos, M.I. and Yuval, G. *Lower Bounds from Complex Function Theory.* extended abstract (1976).

Siegel (56) Siegel, S. *Nonparametric Statistics for the Behavioral Sciences.* McGraw-Hill (1956).

Theil (50) Theil, H. *A Rank-Invariant Method of Linear and Polynomial Regression Analysis, I.* Proc. Kon. Ned. Akad. v. Wetensch. A. 53(1950), 386-392.

Yao (75) Yao, A.C.-C. *An $O(|E| \log \log |V|)$ Algorithm for Finding Minimum Spanning Trees.* Info. Proc. Letters 4(1975), 21-23.

Yuval (76) Yuval, G. *private communication.*