The Exact Fitting Problem for Points

Leonidas Guibas[*] Mark Overmars[†] and Jean-Marc Robert[‡]

April 3, 1991

1 Introduction

A problem which arises in areas such statistical analysis, pattern analysis and computer graphics is that of approximating sets of points by lines where the "best" line is determined according to some criterion. For example, the line must minimizize the maximum orthogonal distances between any of the points and the line or minimize the sum of theses distances [HIIRY]. In this paper, we consider an another variation of this problem: the *exact fitting* problem for a set of N points in the plane consists of finding a line containing the maximum number of points of the set. This problem can be solved easily in $O(N^2)$ time by transforming the points into lines in the dual space [Bro80] and by using the topological line sweep algorithm of Edelsbrunner and Guibas [EG89]. A solution to the exact fitting problem corresponds to a vertex of the line arrangement incident to the maximum number of lines. Hence, by sweeping the line arrangement, it is possible to find such a vertex in $O(N^2)$ time. This is the best algorithm known so far to determine whether there exist three collinear points in a set of points; proving its optimality is a well known open problem (see [Ede88]). On the other hand, it is quite simple to determine in linear time if there is a line containing "almost" all the N points i.e. $N-c$ points for any fixed c. In this case, we simply have to take $2c+2$ different points and to pair them up. One of the $c+1$ couples should determine a line containing $N-c$ points.

In this paper, we present an algorithm to find all the lines containing at least M of the N points in $O\left(\min\left\{\frac{N^2}{M}\log\frac{N}{M}, N^2\right\}\right)$ time. The parameter M is given with the set of N points as input. We should notice that this solution gives an optimal linear time algorithm when M represents a fixed fraction of the points i.e. $M = \epsilon N$ for a constant $0 < \epsilon < 1$. We also show how to use this algorithm to find a line containing the maximum number of points of the set in $O\left(\min\left\{\frac{N^2}{M}\log\frac{N}{M}, N^2\right\}\right)$ time. In this case, only the set of N points is given as input and the parameter M corresponds to the number of points contained in the solution.

2 The Exact Fitting Algorithm

We begin by presenting a simple algorithm solving the exact fitting problem in $O(\frac{N^2}{M}\log\frac{N}{M})$ time when $M \in O(N/\log N)$.

Algorithm MinN1

Input: Set S of N points and an integer $3 \leq M \leq N$.
Output: All the lines containing at least M points.

1. If $M \leq 4$, dualize the points and use the topological line sweep algorithm to find all the vertices incident to M lines and stop.

[*]Laboratory of Computer Science, MIT, Cambridge
[†]Department of Computer Science, University of Utrecht, Utrecht
[‡]School of Computer Science, McGill University, Montréal

2. Split S into the subsets S_1 and S_2 of $\lfloor N/2 \rfloor$ and $\lceil N/2 \rceil$ points , respectively.
3. For each subset, find all the lines containing at least $\lfloor M/2 \rfloor$ points.
4. For each candidate line found in Step 2, determine how many points lie on it.

The analysis of this algorithm relies heavily on the following lemmas:

Lemma 1 [STJ83], [CEG*90] *Let $I(x,y)$ be the number of incidences between x lines and y points in the plane. Then, $I(x,y) \in O(x^{2/3}y^{2/3} + x + y)$.*

Lemma 2 [Aga90], [Mat90] *The incidences between x lines and y points in the plane can be computed in $O(x^{2/3}y^{2/3}\log^{2/3} x + (x+y)\log x)$ time.*

The running time of Algorithm MinN1 can be expressed by the recurrence

$$\begin{array}{lcll} T(N,M) & \in & O(N^2) & \text{if } M = 3,4, \\ T(N,M) & \leq & T(\lfloor N/2 \rfloor, \lfloor M/2 \rfloor) + T(\lceil N/2 \rceil, \lfloor M/2 \rfloor) + V(N,M) & \text{otherwise .} \end{array}$$

Here, $V(N,M)$ represents the time taken by Steps 2 and 3. This function depends on the number of candidate lines found in Step 2 and how fast they can be checked in Step 3 with the algorithm mentioned implicitly in Lemma 2 determining the incidences between a set of lines and a set of points. Let *#lines* be the number of candidate lines from each subproblem i.e. the number of lines containing at least $\lfloor M/2 \rfloor$ of the $\lfloor N/2 \rfloor$ points. Using Lemma 1, we can show that *#lines* $\in O\left(\max\left\{\frac{N^2}{M^3}, \frac{N}{M}\right\}\right)$. Hence, the number *#lines* depends whether M^2 is less than N or not. To simplify the analysis, we suppose that $N = 2^n$ and $M = 2^m$ and replace $T(2^n, 2^m)$ by $t(n,m)$. Furthermore, we split the analysis into two cases. When $2m \leq n$, the number of candidate lines is in $O(2^{2n-3m})$ and the recurrence becomes

$$\begin{array}{lcl} t(n,1) & \in & O(2^n), \\ t(n,m) & \leq & 2t(n-1,m-1) + O(2^{2n-2m}(2n-3m)^{2/3} + (2^{2n-3m} + 2^n)(2n-3m)). \end{array}$$

By solving this recurrence, we obtain that $t(n,m) \in O(2^{2n-m}(n-m))$. When $2m \geq n$, the number of candidate lines is in $O(2^{n-m})$ and the recurrence becomes

$$t(n,m) \leq 2t(n-1,m-1) + O((2^{2n-m}(n-m))^{2/3} + 2^n(n-m)).$$

In this case, the recurrence can be applied as long as $2m \geq n$ i.e. $2m - n$ times, since $2(m-i) \geq (n-i)$ if and only if $i \leq 2m - n$. After $2m - n$ iterations, the value of $t(2n-2m, n-m)$ is given by the first case of the analysis. Therefore, the solution is $t(n,m) \in O(2^{2n-m}(n-m) + 2^n(n-m)(2m-n))$.

Finally, a solution for the general case where N and M are not power of two can also be derived (see [BB88]). Let n and m be such that $2^{n-1} \leq N < 2^n$ and $2^{m-1} \leq M < 2^m$. Using the fact that $T(N,M)$ is eventually non-decreasing in N and non-increasing in M, it is easy to show that $T(N,M) \leq t(n, m-1)$. Therefore, we have $T(N,M) \in O\left(\frac{N^2}{M}\log\frac{N}{M} + N\log\frac{N}{M}\log\frac{M^2}{N}\right)$. The second term of this expression becomes dominant for $M \in \Omega(N/\log N)$. In order to obtain a better solution when M is "close" to N, we have to modify the previous algorithm.

Algorithm MinN2

Input: Set of N points S and an integer $M < N$.
Output: All the lines containing at least M points.

1. Split S into $\lfloor M^2/N \rfloor$ subsets of $\lceil N^2/M^2 \rceil$ points.
2. For each subset, find all the lines containing at least $\lfloor N/M \rfloor$ points.
3. For each candidate found in Step 2, determine how many points lie on it.

Algorithm MinN2 will be used only when $M^3 \geq N^2$. This corresponds to the case where each subset defined in Step 1 contains less than M points. The time complexity of this algorithm is given by the recurrence

$$T(N,M) \leq \lfloor M^2/N \rfloor T(\lceil N^2/M^2 \rceil, \lfloor N/M \rfloor) + V(N,M).$$

By using the same notation and the same assumptions for N and M as we did in the first case and by using Algorithm MinN1 to solve each subproblem in Step 2, the time complexity of Algorithm MinN2 can be given by

$$t(n,m) \leq O(2^{2n-m}(n-m)) + V(2^n, 2^m).$$

For each subproblem, the number of lines containing at least 2^{n-m} of the 2^{2n-2m} points is in $O(2^{n-m})$. Therefore, the number of lines found in Step 2 is in $O(2^m)$. Fortunately, all these lines do not correspond to candidate lines. Suppose there are 2^m collinear points. These collinear points define at least $\Omega(2^{4m-3n})$ times the line in Step 2. To obtain this lower bound, we should find the "worst" distribution of the 2^m collinear points among the subproblems to minimize the number of subproblems having 2^{n-m} of them. This distribution is obtained by putting $2^{n-m}-1$ of the points in each subproblem and, then, pack the remaining 2^{2m-n} points in fewest number of subproblems. Hence, the lower bound on the number of times that the line appeared corresponds to the lower bound on the number of subproblems with at least 2^{n-m} of the 2^m collinear points i.e. $\lfloor 2^{2m-n}/(2^{2n-2m}-2^{n-m}+1) \rfloor$.

Since, each set of at least 2^m collinear points determined at least $\Omega(2^{4m-3n})$ lines in Step 2, the maximum number of candidate lines is in $O(2^{3n-3m})$ i.e. $O(2^m)/\Omega(2^{4m-3n})$. These candidate lines can be determined efficiently by adapting the algorithm presented in [MG82] to find the repeated elements in a multiset. In their paper, Misra and Gries showed how to find the values that occur more than n/k times in an array of n elements in $O(n \log k)$ time. We simply have to find all the lines appearing at least $\Omega(2^{4m-3n})$ times in the list of $O(2^m)$ lines. This step can be done in $O((n-m)2^m)$ time. Therefore, the running time of Algorithm MinN2 is given by

$$t(n,m) \in O(2^{2n-m}(n-m)) + O(2^{8n/3-2m}(n-m)^{2/3} + (2^{3n-3m}+2^n)(n-m)).$$

Since $3m \geq 2n$, $t(n,m) \in O(2^{2n-m}(n-m))$. Furthermore, using the same argument as before, we obtain $T(N,M) \in O\left(\frac{N^2}{M}\log\frac{N}{M}\right)$.

By choosing the brute-force algorithm based on the topological line sweep algorithm when $M \leq \log N$, Algorithm MinN1 when $M^3 \leq N^2$ and Algorithm MinN2 otherwise, we obtain the following result:

Theorem 3 *Let S be a set of N points in the plane. It is possible to determine if there are M collinear points in S in $O\left(\min\left\{\frac{N^2}{M}\log\frac{N}{M}, N^2\right\}\right)$ time.*

This solution can be used to find in the same time complexity the maximum number of collinear points in a set of N points. Here, the value M represents the number of points lying on the solution line.

Algorithm EF

```
Input:   Set of N points S.
Output:  A line containing the maximum number of collinear points.

1. Set $i$ to 1.
2. Find all the lines containing at least $N/2^i$ points of $S$.
3. If there is no such line, increase $i$ by 1 and go to Step 2.
4. If there are such lines, output a line containing the maximum number of points.
```

Suppose there are M collinear points in S. The algorithm will stop when $M \geq N/2^i$ i.e. after $\lceil \log \frac{N}{M} \rceil$ iterations. Therefore, the running time is given by

$$T(N,M) \in O\left(\sum_{i=1}^{\lceil \log \frac{N}{M} \rceil} T(N, N/2^i)\right).$$

The following corollary summarizes the result:

Corollary 4 *Let M be the maximum number of collinear points in S. Therefore, these points can be found in* $O\left(\min\left\{\frac{N^2}{M}\log\frac{N}{M}, N^2\right\}\right)$.

3 Conclusion and Open problems

We have presented an $O\left(\min\left\{\frac{N^2}{M}\log\frac{N}{M}, N^2\right\}\right)$ time algorithm solving the exact fitting problem for sets of N of points in the plane. Can we have a better solution? Any improvement to the algorithm computing the incidences between a set of lines and a set of points will be reflected in the time complexity of our solution. Hence, an optimal $O(x^{2/3}y^{2/3} + x + y)$ time solution, in Lemma 2, would reduce the time complexity of our algorithm to $O\left(\frac{N^2}{M}\right)$.

An another interesting open problem is to extend the exact fitting problem to sets of N vertical line segments. In this case, we want to find a line *intersecting* the maximum number of line segments. This problem can also be solved in $O(N^2)$ time by sweeping the dual arrangement with a topological line. Edelsbrunner and Guibas [EG89] presented it as an application of their topological line sweep algorithm. Nevertheless, we can formulate the problem as we did for sets of points and ask to find a line intersecting at least M line segments, for a given M, in $o(N^2)$ time, when M is more than a constant.

References

[Aga90] P.K. Agarwal. Partitioning arrangements of lines II: applications. *Disc. Comp. Geom.*, 5:533-573, 1990.

[BB88] G. Brassard and P. Bratley. *Algorithmics: Theory and Practice.* Prentice-Hall, Englewood Cliffs, NJ, 1988.

[Bro80] K.Q. Brown. *Geometric transforms for fast geometric algorithms.* PhD thesis, Carnegie-Mellon University, Pittsburg, PA, 1980.

[CEG*90] K.L. Clarkson, H. Edelsbrunner, L.J. Guibas, M. Sharir, and E. Welzl. Combinatorial complexity bounds for arrangements of curves and spheres. *Disc. Comp. Geom.*, 5:99-160, 1990.

[Ede88] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer-Verlag, 1988.

[EG89] H. Edelsbrunner and L.J. Guibas. Topologically sweeping an arrangement. *J. Comp. Syst. Sc.*, 38:165-194, 1989.

[HIIRY] M.E. Houle, H. Imai, K. Imai, J.-M. Robert and P. Yamamoto. Orthogonal weighted linear L_1 and L_∞ approximation and applications. Submitted to *Disc. Appl. Math.*.

[Mat90] J. Matoušek. Cutting hyperplane arrangements. In *Proc. of the 6th Annual ACM Symp. on Comp. Geom.*, pages 1-9, 1990.

[MG82] J. Misra and D. Gries. Finding repeated elements. *Sc. Comp. Prog.*, 2:143-152, 1982.

[STJ83] E. Szemerédi and W.T. Trotter Jr. Extremal problems in discrete geometry. *Combinatorica*, 3:381-392, 1983.