The Ordinary Line Problem Revisited

Asish Mukhopadhyay *† Eugene Greene *‡

Abstract

Let $\mathcal{P}$ be a set of n points in the plane. A *connecting* line of $\mathcal{P}$ is a line that passes through at least two of its points. A connecting line is called *ordinary* if it is incident on exactly two points of $\mathcal{P}$. If the points of $\mathcal{P}$ are not collinear then such a line exists. In fact, there are $\Omega(n)$ such lines [8]. In this note, we present a very simple algorithm for finding an ordinary line, assuming that the points of $\mathcal{P}$ are not collinear.

1 Introduction

Let $\mathcal{P}$ be a set of n points in the plane. A *connecting* line of $\mathcal{P}$ is a line that passes through at least two of its points. Let $\mathcal{S}$ be the set of all connecting lines. A line $l \in \mathcal{S}$ is said to be an *ordinary line* if it passes through exactly two points of $\mathcal{P}$.

The problem of establishing the existence of such a line originated with Sylvester [12], who proposed the following problem in 1893:

> If n points in the plane are such that a line passing through any two of them passes through a third point, then are the points collinear?

No solution came forth during the next forty years. In 1943, a positive version of the same problem was proposed by Erdos [5], and was solved by Gallai in the following year [6].

Subsequently other proofs also appeared, notable among which were the proofs by Steinberg [11] and Kelly [2]. These results show that the answer is in the affirmative for plane projective geometry. Therefore if the points of $\mathcal{P}$ are not collinear then there is at least one ordinary line. In fact, Kelly and Moser [8] showed that there are at least $3n/7$ ordinary lines.

In [10] two different algorithms are reported for computing an ordinary line. One is based on parametric search, while the other uses dualization. Here we present a very simple algorithm that works in the primal plane.

The paper is organized as follows: In section 2, we discuss the existence of an ordinary line. In the following section we present our algorithm.

2 Existence of an ordinary line

We will henceforth assume that the given set of points is non-collinear. One (constructive) proof of existence, due to Kelly, can be found in [4]. The algorithm that it implies is in $O(n^3)$. Other existence proofs can be found in [9], [7]. The proof that we discuss below leads to an efficient algorithm. It is described in [3] in the setting of ordered geometry, in a strictly axiomatic way [1]. Our treatment on the other hand is more intuitive.

Let $[ABC]$ denote a configuration of three distinct, collinear points A, B, C with B lying between A and C. Let l_0 be a line incident with exactly one point of $\mathcal{P}$, say P_1. We can find this line as follows. Let l' be any line that does not contain P_1. The lines joining P_1 to all the other points of $\mathcal{P}$, intersect l' in at most $n-1$ points. Let Q be any *other* point on l'. We let l_0 be the line through P_1 and Q. Let A be the intersection of a connecting line with l_0 such that no other connecting line intersects the segment P_1A. If the connecting line through A is incident with exactly two points, say P_2 and P_3 (we reindex the points, if necessary), then we are done. Otherwise, let P_4 be a third point on this connecting line (Figure 1). Of the three points, at least two are on the same side of A. Let us assume that these are P_2 and P_3 so that we have the configurations $[P_2P_3A]$ and either $[P_4P_2P_3]$ or $[P_3AP_4]$ (Figure 1).

We make the following claim.

Claim 2.1 *The connecting line l_1 through P_1 and P_2 is ordinary.*

Proof. If not, let P_5 be a third point on l_1. We have then three different configurations to consider:

- $[P_5P_1P_2]$: In this case the connecting line through P_5 and P_3 intersects P_1A (Figure 2, left-hand configuration).
- $[P_1P_5P_2]$: In this case the connecting line through P_5 and P_4 intersects the segment P_1A (Figure 2, right-hand configuration).

*School of Computer Science, University of Windsor, Windsor, Canada

†asishm@cs.uwindsor.ca, Partially supported by an NSERC operating grant

‡greene6@uwindsor.ca, Supported by an NSERC USRA

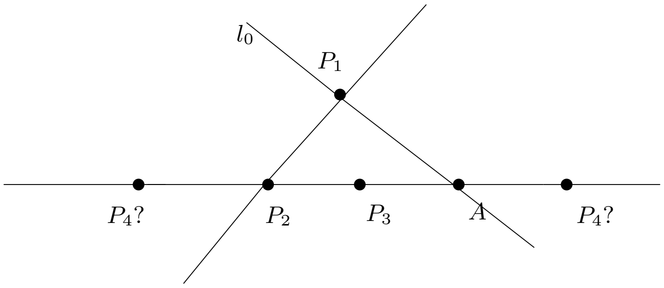

Figure 1: Points on the connecting line through A

- $[P_5P_2P_1]$: Similar to Case 2.

The conclusion in all three cases contradicts that P_1A is intersection-free. Hence the line l_1 is ordinary. □

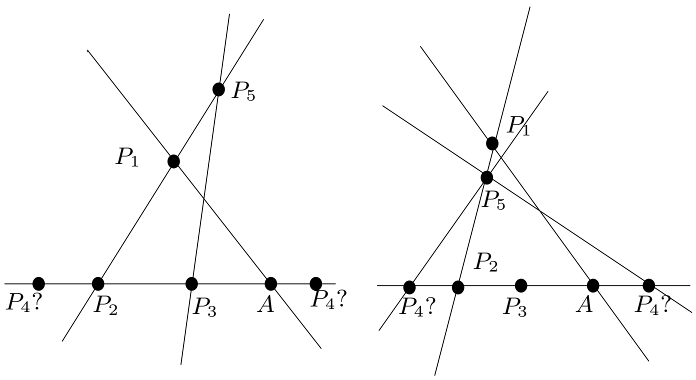

Figure 2: Illustration of Cases 1 and 2

The following observation by Kelly explains how the above construction might have been found. First, a small definition: Let P be any point of $\mathcal{P}$. The connecting lines of the set $\mathcal{P} - \{P\}$ dissect the plane into different regions. The connecting lines that bound the region in which P lies are said to be its *neighbours* (see Figure 3).

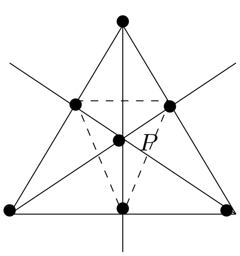

Figure 3: Neighbours of P

Observation 1 *[2] If there are no ordinary lines through a point P, then every neighbour of P is an ordinary line.*

Thus an efficient method for computing A would yield a fast algorithm for finding an ordinary line.

3 The algorithm

First, we want to choose a point P_1 and a line l_0 through P_1, such that all points of $\mathcal{P}' = \mathcal{P} - \{P_1\}$ are on the same side of l_0. If there is a unique left-most (or bottom-most, etc.) point of $\mathcal{P}$, then P_1 can be chosen as one of these points, and l_0 can be chosen as the vertical (or horizontal) line through P_1. If there is no "easy" choice for P_1, then we can construct the convex hull, C, of $\mathcal{P}$. Choose P_1 as a vertex of C, and choose l_0 as a tangent to C, through P_1, upon which no edge of C is incident.

Consider the points of intersection of the connecting lines of $\mathcal{P}'$ with l_0. We want to find the intersection point, A, closest to P_1, but not identical with P_1.

Note: It might be that all the points of $\mathcal{P}'$ are collinear, and that the supporting line of $\mathcal{P}'$ is parallel to the l_0 that we have chosen. In this case, any line defined by P_1 and some point of $\mathcal{P}'$ will be ordinary. This is true whenever the points of $\mathcal{P}'$ are collinear, not just when the supporting line of $\mathcal{P}'$ is parallel to l_0.

Assuming the points of $\mathcal{P}'$ are not collinear: We sort the points of $\mathcal{P}'$ twice. First, we sort the points in angular order around P_1, relative to l_0. Then we sort within each equivalence class (resulting from the previous sort) according to distance from P_1. Let $\theta(Q)$ be the counterclockwise angle that $\overline{P_1Q}$ makes with l_0. We end up with a two dimensional array L such that:

- For all i and j, if $i < j$ then $\theta(L[i,k]) < \theta(L[j,l])$ for all k and l; and
- For all i and j, if $i < j$ then $distance(P_1, L[k,i]) < distance(P_1, L[k,j])$ for all k.

We do not want the intersection point to be on P_1, so we do not try pairs of points in the same sub-array of L.

Claim 3.1 *The intersection point closest to P_1, but not identical with P_1, will be made by a connecting line that joins two points of $\mathcal{P}'$ that are in adjacent sub-arrays in L.*

Proof. Let Q and R be the points whose supporting line $\overline{QR}$ intersects l_0 at A, such that A is the closest intersection point to P_1, that is not identical with P_1. Assume that Q and R are in non-adjacent sub-arrays of L, with Q being closer to l_0 than R is. So, there is some point S between the rays $\overrightarrow{P_1Q}$ and $\overrightarrow{P_1R}$ (see Figure 4). If S is on the same side of $\overline{QR}$ as P_1, then the line $\overline{RS}$

will intersect l_0 at a point closer to P_1 than A is. If S is on the side of $\overline{QR}$ not containing P_1, then the line $\overline{QS}$ will intersect l_0 at a point closer to P_1 than A is. Either way, there is a contradiction. □

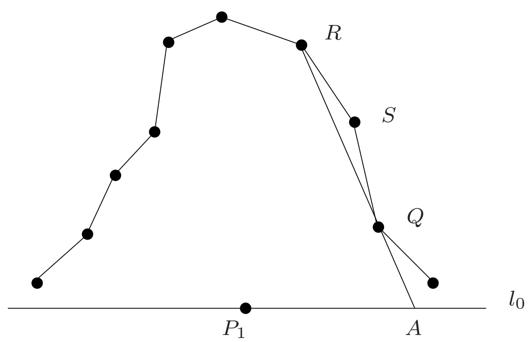

Figure 4: Intersection with l_0 of the join of non-adjacent points, Q and R

We do not want to try all combinations of points in adjacent sub-arrays to find the pair that generates the closest intersection point. We would like to test only a constant number of combinations for each pair of adjacent sub-arrays.

Claim 3.2 *Of all pairs of points from two adjacent sub-arrays, the first point from one sub-array and the last point from the other will generate the closest intersection point.*

Proof. Let Q and R be the points from two adjacent sub-arrays of L, whose supporting line $\overline{QR}$ intersects l_0 at A, such that A is the closest intersection point to P_1, that is not identical with P_1. Assume that Q is closer to l_0 than R is. If there is a point, S, in Q's sub-array that is closer to P_1 than Q is, then $\overline{RS}$ will intersect l_0 at a point closer to P_1 than A is. If there is a point, T, in R's sub-array that is farther from P_1 than R is, then $\overline{TQ}$ will intersect l_0 at a point closer to P_1 than A is. □

So we can find a pair of points that generate a connecting line l_1, for which the intersection point with l_0, at A, is closest to P_1 without being identical with P_1. If l_1 is an ordinary line, then we are done.

If l_1 is not an ordinary line, there must be at least three points of $\mathcal{P}'$ on l_1. All of the points of $\mathcal{P}'$ are on one side of l_0, so if we imagine A cutting l_1 into two rays, then the points of $\mathcal{P}'$ on l_1 are all on the same ray. Of the points of $\mathcal{P}'$ on l_1, let P_3 be the closest to A, P_2 the second-closest, and P_4 the third-closest.

Claim 3.3 *The line $l_2 = \overline{P_1P_2}$ is an ordinary line.*

Proof. This follows from the discussion in the previous section. □

A formal description of the algorithm is given in Figure 5.

Algorithm OrdinaryLine ($\mathcal{P}$)

Step 1. Compute the convex hull, C, of $\mathcal{P}$.

Step 2. Let P_1 be some vertex of C, and let l_0 be a tangent to C, through P_1, upon which no edge of C is incident.

Step 3. Create L as defined above.

Step 4. For each pair, (Q, R), of extreme (first or last in their sub-array) points in adjacent sub-arrays of L:

Step 4.1. Let $l'_1 = \overline{QR}$, and let A' be the intersection of l'_1 with l_0.

Step 4.2. If A' is closer to P_1 than the current closest intersection point, A, then let $A = A'$, and $l_1 = l'_1$.

Step 5. If l_1 is ordinary, then report l_1 and stop.

Step 6. Find the second-closest point, P_2, to A, on l_1.

Step 7. Report $l_2 = \overline{P_1P_2}$.

Figure 5: Our ordinary line algorithm

3.1 Analysis of the algorithm

Steps 1 and 3 will take $O(n \log n)$ time. Steps 2 and 7 can be done in constant time, and Steps 4, 5, and 6 in linear time. So the complexity of the algorithm is in $O(n \log n)$.

4 Conclusion

Here we have presented a simple $O(n \log n)$ time algorithm for finding an ordinary line from a set of n points. An interesting question is that of finding a non-trivial lower bound for this problem. The authors currently do not know of any such bound.

References

[1] E. Artin. *Geometric Algebra*. Interscience, New York, 1957.

[2] H. Coxeter. A problem of collinear points. *American Mathematical Monthly*, 55:26–28, 1948.

[3] H. Coxeter. *Introduction to Geometry*. J. Wiley and Sons, 1969.

[4] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag, 1987.

[5] P. Erdos. Problem 4065. *American Mathematical Monthly*, 50:65, 1943.

[6] T. Gallai. Solution to problem 4065. *American Mathematical Monthly*, 51:169–171, 1944.

[7] H. Hadwiger, H. Debrunner, and V. Klee. *Combinatorial Geometry in the plane*. Holt, Rinehart and Winston Inc., New York, 1964.

[8] L. M. Kelly and W. O. J. Moser. On the number of ordinary lines determined by n points. *Canad. J. of Math.*, 10:210–219, 1958.

[9] T. Motzkin. The lines and planes connecting the points of a finite set. *Trans. Amer. Math. Soc.*, 70:451–464, 1951.

[10] A. Mukhopadhyay, A. Agrawal, and R. M. Hosabettu. On the ordinary line problem in computational geometry. *Nordic J. of Computing*, 4(4):330–341, 1997.

[11] R. Steinberg. Three point collinearity. *American Mathematical Monthly*, 51:169–171, 1944.

[12] J. J. Sylvester. Mathematical questions 1851. *Educational Times*, 59:98, 1893.