Finding the minimum-width V-shape with few outliers

Boris Aronov[1] **John Iacono**
Özgür Özkan **Mark Yagnatinsky**

Polytechnic Institute of New York University

Abstract

A V-shape is an infinite polygonal region bounded by two pairs of parallel rays emanating from two vertices (see Figure 1). We describe a randomized algorithm that, given n points and a number k, finds the minimum-width V-shape enclosing all but k of the points with probability at least $1 - 1/n^c$ for any c, requiring $O(n^2)$ space with expected running time $O(cn^2(k+1)^4 \log n(\log n \log\log n + k))$.

1 Introduction

Motivation. The motivation for this problem comes from curve reconstruction: given a set of points sampled from a curve in the plane, find a shape close to the original curve. It has been suggested in [AD-V] that in an area where the curve makes a sharp turn, it makes sense to model the curve by a *V-shape*. The authors remark that it would be natural to investigate a variant that can handle a small number of outliers. We investigate that variant here. The problem is an instance of a large class of problems known as *geometric optimization* or *fitting* questions, (see [GeomOpt] for a survey).

Definitions. Consider two rays with a common vertex. Call these the *outer rays*. Make a copy of the outer rays superimposed on the original and call the copy the *inner rays*. Translate the inner rays while keeping their common vertex within the convex hull of the outer rays. The region between the inner and outer rays is called a *V-shape* (see Figure 1).

A *strip* is the region bounded by two parallel lines. Observe that a V-shape is contained in the union of two strips. The *width* of a strip is the distance between its two lines. The *width* of a V-shape is the width of its wider strip. An *outlier* of a V-shape is a point not contained in that V-shape.

[1]Work by B.A. on this paper has been supported by NSF Grants CCF-08-30691, CCF-11-17336, and CCF-12-18791, and by NSA MSP Grant H98230-10-1-0210.

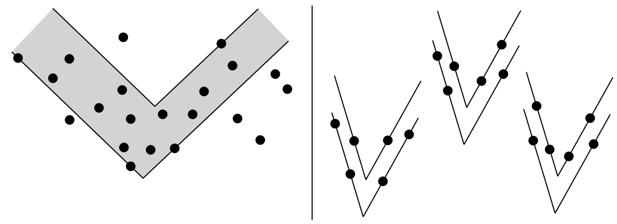

Figure 1: Left: a V-shape with six outliers. Right: a both-outer V-shape, an inner-outer V-shape, and a both-inner V-shape (in left-to-right-order).

Previous work. In [AD-V], the authors develop an algorithm for covering a point set with a V-shape of minimum width that runs in $O(n^2 \log n)$ time and uses $O(n^2)$ space. They also find a constant-factor approximation algorithm with running time $O(n \log n)$, and a $(1+\varepsilon)$-approximation algorithm with a running time of $O((n/\varepsilon)\log n + n/(\varepsilon^{3/2})\log^2(1/\varepsilon))$, which is $O(n \log n)$ for a constant ε.

Result. Given a set of n points in the plane and an integer k, we show how to find the minimum-width V-shape enclosing all but k of the points.

2 The algorithm

Theorem 1. *There is a randomized algorithm that, given n points and a number k, finds the minimum-width V-shape enclosing all but k of the points with probability at least $1 - 1/n^c$ for any c, requiring $O(n^2)$ space with expected running time $O(cn^2(k+1)^4 \log n(\log n \log\log n + k))$.*

Proof. A V-shape is *locally minimal* with respect to a point set P if there is no way to decrease the width of either of the strips by a slight translation of one of the rays or by a slight simultaneous rotation of two parallel rays, without increasing the number of outliers. (Intuitively, both of its strips should hug the part of the point set they cover.) Since there exists a locally minimal V-shape that achieves the smallest possible width of all V-shapes, it is safe to focus only on locally minimal V-shapes.

We divide locally minimal V-shapes into the same three classes as [AD-V] (see Figure 1). A *both-outer* V-shape is a locally minimal V-shape where both outer rays have two points on them. A *both-inner* V-shape is a locally minimal V-shape where both inner rays have two points on them. An *inner-outer* V-shape is a locally minimal V-shape where one of the outer rays and one of the inner rays has two points on it. The algorithm works by finding the minimum-width V-shape of each class, and returning the one that has the smallest width of all three.

Our approach for the both-outer case and the inner-outer case was inspired by the approach of [AD-V] for the inner-outer case, except we use a binary search for one step where they use total enumeration. When there are zero outliers, our algorithm for the both-outer and inner-outer cases would be easier to implement than theirs, at the cost of a logarithmic factor in the running time. However, most of the complexity of their solution was in the both-outer case, and we use their both-outer algorithm as a black box in our both-outer algorithm, by running it on random subsets of the point set (or the entire point set when there are zero outliers).

We handle both-inner V-shapes and inner-outer V-shapes in almost the same way (see Figure 2). We begin by enumerating the edges at levels 0 through k of the point set. An *edge at level* k of a point set P is a directed edge connecting two points in the set such that exactly k of the points lie to the left of the directed line through the edge (so in general position there are $n-k-2$ points to the right). For example, an edge at level 0 is a directed edge of the convex hull. Each enumerated edge e is considered as a candidate for one of the outer rays to go through. The points to the left of e are considered outliers already accounted for. For each e, we do a binary search among points not yet considered outliers. The order for the search is by perpendicular distance from e, which represents the width of the first candidate strip. For each point of the search we find the second strip that has the smallest possible width and still covers the remaining points, except the outliers. If the second strip is wider than the first, the binary search moves farther out from e so that the second strip has fewer points, otherwise it moves closer. To find the second strip, we again enumerate the edges at levels 0 through k of the remaining points. The precise definition of "remaining" here is the key difference between the both-outer and the inner-outer algorithm; we will gloss over this subtle point for now. By now we have chosen three rays, and have no freedom for the fourth: it is dictated by how many more outliers we need. The running time is $O(n^2(k+1)^2 \log^2 n)$.

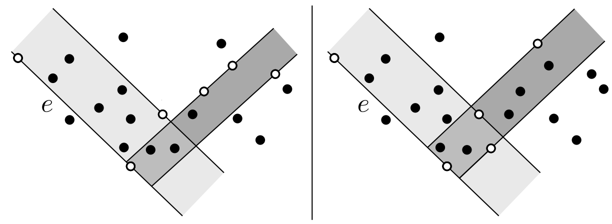

Figure 2: Snapshot of inner-outer algorithm (left) and both-outer algorithm (right).

The best deterministic algorithm we have for finding the minimum-width both-inner V-shape runs in $O(n^3k^2 \log n)$ time. Instead, we use a randomized algorithm that simply takes many random samples of the given point set. For each sample, it enumerates *all* both-inner V-shapes (with no outliers) using the algorithm from [AD-V]. We show that with probability $1-n^c$, the minimum-width both-inner V-shape with k outliers will be one of the V-shapes enumerated. The V-shapes we enumerate might have more than k outliers, so we use a range searching data structure from [Range-Search, pages 2–3] to check that, and discard the V-shapes that have too many. The running time of the both-inner case is $O(cn^2(k+1)^4 \log n(\log n \log\log n + k))$, which dominates the run time of the other two cases. □

Acknowledgments

We would like to thank Muriel Dulieu for participating in discussions of a simpler version of the problem, and Sariel Har-Pelad for the randomized algorithm.

References

[AD-V] B. Aronov and M. Dulieu, How to cover a point set with a V-shape of minimum width, *WADS* 2011, pages 61–72.

[GeomOpt] P. K. Agarwal and M. Sharir, Efficient Algorithms for Geometric Optimization, *ACM Computing Surveys*, Vol. 30 Issue 4, 1998, pages 412–458.

[Range-Search] R. Cole and C. K. Yap, Geometric retrieval problems, *Information and Control*, Vol. 63, Issues 1–2, Oct.–Nov. 1984, pages 39–57.