Computing the Distance between Piecewise-Linear Bivariate Functions

Guillaume Moroz, Boris Aronov

Computing the Distance between Piecewise-Linear Bivariate Functions

Guillaume Moroz*Boris Aronov†

February 3, 2015

Abstract

We consider the problem of computing the distance between two piecewise-linear bivariate functions f and g defined over a common domain M, induced by the L_2 norm, that is $\|f-g\|_2 = \sqrt{\int_M (f-g)^2}$. If f is defined by linear interpolation over a triangulation of M with n triangles, while g is defined over another such triangulation, the obvious naïve algorithm requires $\Theta(n^2)$ arithmetic operations to compute this distance. We show that it is possible to compute it in $\mathcal{O}(n \log^4 n \log\log n)$ arithmetic operations, by reducing the problem to multi-point evaluation of a certain type of polynomials.

We also present several generalizations and an application to terrain matching.

[0]Work on this paper was initiated at the 2011 International INRIA-McGill-Victoria Workshop on Problems in Computational Geometry, held at the Bellairs Research Institute of McGill University in Barbados, West Indies. Part of the work was done at the 2011 Bellairs Workshop and at the Workshop on Discrete and Algebraic Geometry in September 2010, at Val d'Ajol, France. A preliminary version of this work appeared in the *Proceedings of Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)* Moroz and Aronov (2012).

Work by B.A. on this paper has been supported by grant No. 2006/194 from the U.S.-Israel Binational Science Foundation, by NSF Grants CCF-08-30691, CCF-11-17336, and CCF-12-18791, and by NSA MSP Grant H98230-10-1-0210.

*INRIA Nancy - Grand Est, 615 rue du Jardin Botanique, 54600 Villers-lès-Nancy, France; *guillaume.moroz@inria.fr*.

†Department of Computer Science and Engineering, Polytechnic Institute of NYU, Brooklyn, NY 11201-3840, USA; *aronov@poly.edu*.

1 Introduction and problem statement

In this paper we use a novel combination of tools from computational geometry and computer algebra to speed up the computation of the L_2-norm distance between two bivariate piecewise-linear functions. Algebraic tools have already been used in other recent work in computational geometry, to seemingly defy "obvious" lower bounds: for example, Ajwani, Ray, Seidel, and Tiwary Ajwani et al. (2007) use algebraic tools to compute the centroid of all vertices in an arrangement of n lines in the plane in $\mathcal{O}(n \log^2 n)$ time, without explicitly computing the $\Theta(n^2)$ vertices.

We feel that working on geometry problems using a combination of traditional and algebraic tools expands the repertoire of questions that can be approached and answered satisfactorily. Indeed, in a significant recent development, several breakthrough results have been obtained by applying algebraic methods to problems of combinatorial geometry: Guth and Katz's recent work on the joints problem Guth and Katz (2010) and on the Erdős distinct distance problem Guth and Katz (2011) has triggered an avalanche of activity; see, for example, Elekes et al. (2011); Elekes and Sharir (2010); Kaplan et al. (2012); Matoušek (2011); Quilodrán (2010); Solymosi and Tao (2012); Kaplan et al. (2010). It appears that the use of algebraic tools in geometry (both combinatorial and computational) allows one to approach problems inaccessible by more traditional methods. The present work is just one small step in that direction.

Background and previous work In Agarwal et al. (2010, 2013), Aronov et al. considered quite a common object in computational geometry and geographical information systems: that of a "terrain." A *terrain* is (the graph of) a bivariate function over some planar domain, say, a rectangle or a square. It is sometimes used to model actual geographic terrains, i.e., elevation in a mountainous locale, but can also be applied to storing any two-dimensional data set, such as precipitation or snow cover data. A common (though by no means the only) way of interpolating and representing discrete two-dimensional data is a *triangulated irregular network*: the values of a bivariate function are given at discrete points in, say, the unit square. The square is triangulated, using data points as vertices. The function is then linearly interpolated over each triangle. This produces a piecewise-linear approximation of the "real" (and unknown) function.

The problem raised in Agarwal et al. (2010, 2013) was that of comparing

two terrains over the same domain, say, the unit square, but given over two unrelated triangulations. One could imagine comparing the outcome of two different ways of measuring the same data, or finding correlation between, say, the elevation and the snow cover over the same geographic region. The focus of that work was on identifying linear dependence between the two functions or terrains. Three natural distance measures between the two functions were considered in Agarwal et al. (2010) and several algorithms presented for computing such a distance and optimizing it, subject to vertical translation and scaling. The only observation made for the L_2 norm (see the definitions below) in Agarwal et al. (2010) is that, if the two terrains share a triangulation of size n, both the distance computation and the optimization problem can be solved easily in linear time in n, and investigating the existence of a subquadratic running time for the general case was left as an open problem. The substance of the current work is describing near-linear-time algorithms for both the distance computation and the optimization problem.

Problem statement and results Given bivariate functions $f, g\colon M \to \mathbb{R}$, one can naturally define a distance between them as

$$\|f - g\|_2 = \Big(\int_M (f(x,y) - g(x,y))^2 dydx\Big)^{1/2}.$$

Expanding the expression under the integral, we obtain $\int (f-g)^2 = \int f^2 - 2\int fg + \int g^2$. If the two functions are piecewise linear, defined over different triangulations of M, only the middle term presents a problem for efficient computation. Thus, in the bulk of the paper we will focus on the computation of $\int fg$, showing the following:

Theorem 1. *Given two piecewise-linear functions f and g defined over possibly different triangulations of the same domain M, each with at most n triangles, the integral $\int_M f(x,y)g(x,y)dydx$ can be computed using $\mathcal{O}(n \log^4 n \log\log n)$ arithmetic operations.*

In particular, the two triangulations need not share vertices (except those of the boundary of M). Armed with this result, as already mentioned, we can quickly compute $\|f - g\|_2$:

Theorem 2. *Given piecewise-linear functions f and g defined over different triangulations of the same domain M, with at most n triangles each, $\|f - g\|_2$ can be computed using $\mathcal{O}(n \log^4 n \log\log n)$ arithmetic operations.*

Naïvely, the integral in Theorem 1 can be expressed as a sum of integrals over each cell appearing in the overlay of the two triangulations of f and g. Unfortunately, this overlay has a quadratic number of cells, in the worst case. The main idea of our algorithm is to reduce the computation of the integral to double sums of algebraic functions over (irregular) grids, which allows us to use fast multi-point evaluation algorithms. Several algorithms in the recent literature improved the complexity of multi-point evaluation for multivariate polynomials Nüken and Ziegler (2004); Kedlaya and Umans (2008). For the purposes of the problem considered in this paper, we will only need a variant of univariate multi-point evaluation Gathen and Gerhard (2003).

The paper is organized as follows. In section 2, we demonstrate how the integral of a bivariate function over a convex polygon can be expressed as a sum of integrals of the function over certain triangles, one per polygon vertex. Applying the observation to a convex decomposition of a region M expresses the value of $\int_M fg$ as a summation over the vertices of the decomposition. Then, in section 3, we show how the integral over the overlay of the two triangulations can be reduced to a sum of elementary functions over pairs of edges, plus some additional terms computable in near-linear time. In section 4, we use the bipartite clique decomposition to arrange the pairs of edges in complete irregular grids of a fairly specific form. Finally, in section 5, we use a fast multi-point evaluation algorithm to compute the sums over each grid, completing the description of our method. Section 6 describes an application of this method to the computation of the distance between two terrains and of the correlation between two terrains in the sense of Agarwal et al. (2010, 2013), and section 7 discusses the difficulties of extending these methods to three bivariate functions and two trivariate functions.

In the remainder of the paper, a *polynomial* refers to a polynomial with real coefficients in one or more variables, stored in explicit expanded representation, so that any coefficient can be accessed in constant time.

2 How to integrate over a convex subdivision

Consider a bivariate function $h\colon \mathbb{R}^2 \to \mathbb{R}$. We are interested in expressing its integral over a convex polygon C as a sum of terms, one for each vertex of C. To simplify our presentation and without loss of generality, we assume that all vertices of C lie to the right of the y-axis and no edge of C is vertical; in the application below one can apply an appropriate rotation to eliminate vertical

edges, if necessary. For a vertex p of C we refer to the lines supporting the edges of C incident to p as $L_{p,C}$ (the one with higher slope) and $U_{p,C}$ (the one with lower slope); to the left of p, $L_{p,C}$ is below $U_{p,C}$. Let

$$L_{p,C}\colon y = y_l(p,C) + s_l(p,C)x, \text{ and}$$
$$U_{p,C}\colon y = y_u(p,C) + s_u(p,C)x.$$

We omit the explicit dependence on C and/or p whenever it causes no confusion. Define $\delta(p,C)$ to be -1 if C is above both $L_{p,C}$ and $U_{p,C}$, or below both of them, and $+1$ if C is below $U_{p,C}$ and above $L_{p,C}$, or vice versa. Finally, denoting by x_p the abscissa of p, put

$$\mathcal{T}(p,C,h) := \delta(p,C) \int_{x=0}^{x_p} \int_{y=y_l(p,C)+s_l(p,C)x}^{y_u(p,C)+s_u(p,C)x} h(x,y)dydx.$$

In words, $\mathcal{T}$ is the signed integral of h over the triangle T_p delimited by the y-axis, $L_{p,C}$, and $U_{p,C}$.

With the above notation, we express the integral of h over C in a convenient way as a sum of terms associated with its vertices,

Lemma 3. *Let C be a convex polygon with vertices $p_1, ..., p_k$, and h a bivariate function. Then*

$$\int_C h(x,y)dydx = \sum_{j=1}^{k} \mathcal{T}(p_j, C, h). \tag{1}$$

Proof. It is sufficient to show that characteristic functions of C and the triangles T_{p_j} satisfy

$$1_C = \sum_{j=1}^{k} \delta(p_j, C) 1_{T_{p_j}}. \tag{2}$$

Partition the vertices of C into four subsets V_L, V_R, V_T, and V_B as follows. $V_L := \{p_L\}$ (resp., $V_R = \{p_R\}$) consists of the unique leftmost (resp., rightmost) vertex of C. $V_T := \{t_1, \ldots\}$ (resp., $V_B := \{b_1, \ldots\}$) is the sequence of vertices of $\mathcal{C}$ from p_R to p_L in the counterclockwise (resp., clockwise) direction; refer to Fig. 1.

In this case, we have:

$$\delta(p,C) = \begin{cases} +1 & \text{if } p \in V_L \cup V_R, \\ -1 & \text{if } p \in V_T \cup V_B. \end{cases}$$

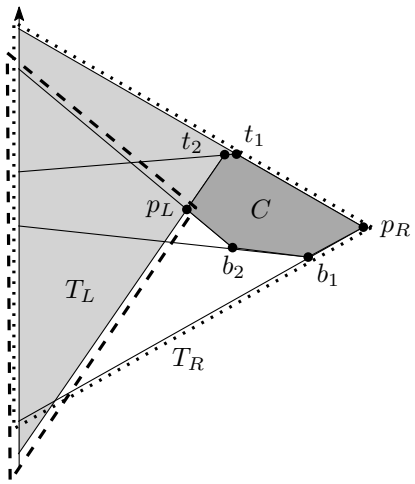

Figure 1: An illustration of the proof of Lemma 3. T_R is dotted; T_L is dashed; A_T is lightly shaded.

Put $T_L := T_{p_L}$ and $T_R := T_{p_R}$. Since C is convex, for all i, the line $t_i t_{i+1}$ (resp., $b_i b_{i+1}$) is below the line $t_{i-1}t_i$ (resp., above the line $b_{i-1}b_i$) left of C. Thus, the triangles T_t, $t \in V_T$, (resp., T_b, for $b \in V_B$) do not overlap. Let $A_T = \bigcup_{t \in V_T} T_p$ and $A_B = \bigcup_{b \in V_B} T_b$. By construction we have

$$A_T \cup A_B = (T_L \cup T_R) \setminus C \quad \text{and} \quad A_T \cap A_B = T_L \cap T_R.$$

Since $C \subset T_R \subset T_L \cup T_R$, the first equality, written in terms of characteristic functions, gives

$$1_{A_T} + 1_{A_B} - 1_{A_T \cap A_B} = 1_{T_L} + 1_{T_R} - 1_{T_L \cap T_R} - 1_C,$$

which can be simplified, using the second equality, to yield

$$1_{A_T} + 1_{A_B} = 1_{T_L} + 1_{T_R} - 1_C.$$

Expanding 1_{A_T} and 1_{A_B}, we obtain

$$\sum_{p \in V_T} 1_{T_p} + \sum_{p \in V_B} 1_{T_p} = 1_{T_L} + 1_{T_R} - 1_C.$$

This implies eq. (2) and proves the lemma. □

We now consider a convex subdivision $\mathcal{C}$ of some bounded region M in the plane, with each cell C of $\mathcal{C}$ associated with its own bivariate function h_C, thereby defining a function h over all of M (as it does not affect the value of the integral, the functions h_C need not agree along the common boundaries of adjacent cells). By summing eq. (1) over the cells C of $\mathcal{C}$, we can compute $\int_M h(x,y)dydx$:

Corollary 4.

$$\int_M h(x,y)dydx = \sum_{cell\ C \in \mathcal{C}} \int_C h_C(x,y)dydx = \sum_{\substack{cell\ C \in \mathcal{C} \\ p\ vertex\ of\ C}} \sum \mathcal{T}(p, C, h_C). \quad (3)$$

One of the advantages of the formulation in eq. (3) for our application is that an individual integral under the sum can be expressed as a rational function when h_C is a bivariate polynomial.

Lemma 5. *Let $i, j \geq 0$ be integers and p be an intersection point of $y = y_l + s_l x$ and $y = y_u + s_u x$ as above. Then*

$$x_p = -\frac{y_u - y_l}{s_u - s_l}$$

and

$$\int_{x=0}^{x_p} \int_{y=y_l+s_l x}^{y_u+s_u x} x^i y^j dydx = \frac{P_{i,j}(y_l, y_u, s_l, s_u)}{(s_u - s_l)^{i+j+1}},$$

where $P_{i,j}$ is a polynomial of total degree $i + 2j + 2$, degree j in s_l, s_u and degree $i + j + 2$ in y_l, y_u.

Proof. Let $Q_{i,j}(u, v, x)$ be the only polynomial such that $\frac{\partial Q_{i,j}}{\partial x} = x^i(u + vx)^j$ and $Q_{i,j}(u, v, 0) = 0$ for all $u, v \in \mathbb{R}$. In particular, $Q_{i,j}$ has total degree $i+2j+1$. The exponents of x range from $i+1$ to $i+j+1$, and the coefficient of x^{i+j+1} in $Q_{i,j}$ is $\frac{1}{i+j+1}v^j$. Now

$$\begin{aligned} \int_{x=0}^{x_p} \int_{y=y_l+s_l x}^{y_u+s_u x} x^i y^j dydx &= \int_{x=0}^{x_p} \frac{1}{j+1}(x^i(y_u + s_u x)^{j+1} - x^i(y_l + s_l x)^{j+1})dx \\ &= \frac{1}{j+1}(Q_{i,j+1}(y_u, s_u, x_p) - Q_{i,j+1}(y_l, s_l, x_p)). \end{aligned}$$

Therefore the value of the integral can be expressed as a polynomial in y_l, y_u, s_l, s_u, x_p of total degree $i + 2j + 3$ of the form

$$\frac{(s_u^{j+1} - s_l^{j+1})x_p^{i+j+2}}{(i+j+2)(j+1)} + \sum_{k=0}^{j} q_k(y_l, y_u, s_l, s_u)x_p^{i+1+k}$$

where q_k are polynomials of degree k in s_l, s_u and $j + 1 - k$ in y_l, y_u. After substituting $-\frac{y_u - y_l}{s_u - s_l}$ for x_p and bringing to a common denominator, we conclude that the expression can be rewritten in the form

$$\frac{(s_u - s_l)P(y_l, y_u, s_l, s_u)}{(s_u - s_l)^{i+j+2}},$$

as claimed, with P of total degree $i + 2j + 2$, degree j in s_l, s_u and degree $i + j + 2$ in y_l, y_u. □

3 How to integrate over an overlay

In our problem, we are interested in computing the integral $\int f(x, y)g(x, y)dydx$, where f is defined over a triangulation $\mathbf{T}_f$ of $M \subset \mathbb{R}^2$, with a separate linear function $f_\Delta(x, y)$ determining f over each triangle $\Delta \in \mathbf{T}_f$; g is defined similarly over a possibly different triangulation $\mathbf{T}_g$ of M. The product $h(x, y) := f(x, y)g(x, y)$ is thus naturally defined over the convex decomposition $\mathcal{C}$ of M that is the overlay of $\mathbf{T}_f$ and $\mathbf{T}_g$. By Corollary 4, it is sufficient to evaluate a sum over all vertices of $\mathcal{C}$. The vertices of $\mathcal{C}$ come in two flavors: the original vertices of $\mathbf{T}_f$ and of $\mathbf{T}_g$, and the (proper) intersections of edges of $\mathbf{T}_f$ and $\mathbf{T}_g$. Therefore,

$$\begin{aligned}\int_M f(x, y)g(x, y)dydx &= \sum_{\substack{p \text{ vertex of } \mathcal{C} \\ C \text{ adjacent to } p}} \sum \mathcal{T}(p, C, h_C) \\ &= \underbrace{\sum_{\substack{p \text{ vertex} \\ \text{of } \mathbf{T}_f \cup \mathbf{T}_g}} \sum_{\substack{C \text{ adjacent} \\ \text{to } p}} \mathcal{T}(p, C, h_C)}_{\Sigma_v} + \underbrace{\sum_{\substack{p = e_1 \cap e_2, \\ (e_1, e_2) \in \mathbf{T}_f \times \mathbf{T}_g}} \sum_{\substack{C \text{ adjacent} \\ \text{to } p}} \mathcal{T}(p, C, h_C)}_{\Sigma_e}.\end{aligned}$$

The sum Σ_v involves $\mathcal{O}(n)$ integrals. We preprocess each of $\mathbf{T}_f$ and $\mathbf{T}_g$ for logarithmic-time point location queries, in $\mathcal{O}(n)$ time (see, for example,

Kirkpatrick (1983)). For each vertex p of $\mathbf{T}_f$, we locate the k triangles $\Delta_g^j \in \mathbf{T}_g$ containing it; $k = 1$ if p lies in the interior of a triangle of $\mathbf{T}_g$, $k = 2$ if it lies on an edge, and $k > 2$ if p is a shared vertex of the two triangulations. We denote by $\Delta_f^i \in \mathbf{T}_f$ the l triangles incident to p. The overlay of these triangle restricted to the cells adjacent to p contains at most $k + l$ cells and can be computed in $O(k + l)$. Then for each cell such that $\Delta_f^i \cap \Delta_g^j \neq \emptyset$, we can compute $\mathcal{T}(p, \Delta_f^i \cap \Delta_g^j, f_{\Delta_f^i} g_{\Delta_g^j})$ in constant time. The treatment of vertices of $\mathbf{T}_g$ is symmetric. We spend $\mathcal{O}(n \log n)$ total time for point location. The remaining work is proportional to the sum of vertex degrees in both triangulations, which is $\mathcal{O}(n)$. Hence Σ_v can be computed in $\mathcal{O}(n \log n)$ arithmetic operations. We devote the rest of the discussion to computing Σ_e efficiently.

4 Bipartite clique decomposition

Let E_f be the set of edges of $\mathbf{T}_f$ and E_g the set of edges of $\mathbf{T}_g$. As in section 2, we assume that no edge is vertical so that we can associate naturally a slope to each edge. Furthemore, the number of edges in E_f and E_g is $O(n)$, if each triangulation has at most n triangles.

In Chazelle et al. (1994), it is shown that it is possible to compute a family $\mathcal{F} = \{(R_1, B_1), \ldots, (R_u, B_u)\}$ where $R_k \subset E_f$ and $B_k \subset E_g$, such that

(i) every segment in R_k intersects every segment in B_k;
(ii) every segment of R_k has lower slope than every segment of B_k, or vice versa;
(iii) for every intersecting pair $(e_1, e_2) \in E_f \times E_g$ there is exactly one k such that $e_1 \in R_k$, $e_2 \in B_k$;
(iv) $\sum_k (|R_k| + |B_k|) = O(n \log^2 n)$.

This family can be computed in $\mathcal{O}(n \log^2 n)$ time.

In other words, each pair (B_k, R_k) forms an irregular grid, by properties (i) and (ii). We use this decomposition to rewrite Σ_e, which is a sum over all proper intersections of an edge of $\mathbf{T}_f$ with an edge of $\mathbf{T}_g$, as a sum of (disjoint, by property (iii)) sums over individual subgrids $B_k \times R_k$. Property (iv) allows us to control the total size of the subproblems thus created. Below we focus on a single grid sum.

5 Multi-point evaluation

In this section, we explain how to efficiently compute a grid sum

$$\sum_{\substack{p=e_1\cap e_2,\\(e_1,e_2)\in R\times B}} \sum_{\substack{C \text{ adjacent}\\ \text{to } p}} \mathcal{T}(p, C, h_C) = \sum_{e_1\in R}\sum_{e_2\in B} \sum_{\substack{C \text{ adjacent}\\ \text{to } p = e_1\cap e_2}} \mathcal{T}(p, C, h_C), \quad (4)$$

where $(R, B) := (R_k, B_k)$ is a pair of sets of triangulation edges produced by the bipartite clique decomposition. In particular, all segments of R intersect all segments of B and, moreover, without loss of generality, the slopes of segments of R are greater than those of segments of B.

5.1 Reduction to sums of rational functions

Each triangle of $\mathbf{T}_f$ and $\mathbf{T}_g$ is associated with a bivariate linear function. If e is an edge of $\mathbf{T}_f \cup \mathbf{T}_g$, it is not vertical by assumption, and we let $f_u(e)$ be the linear function associated to the upper triangle (i.e., the one lying immediately above e in the y direction in the triangulation to which e belongs) adjacent to e; $f_l(e)$ is the corresponding function for the lower triangle; we can define the function to be identically zero for regions outside M, but it will never be used by the algorithm.

A vertex $p = e_1 \cap e_2$, with $e_1 \in R$ and $e_2 \in B$, lies on the boundary of four cells of $\mathcal{C}$. We focus on the cell $C_{\text{left}} = C_{\text{left}}(p)$ lying above e_1 and below e_2, for which p is the rightmost point. Thus $h_{C_{\text{left}}} = f_u(e_1)f_l(e_2)$ and $\delta(p, C_{\text{left}}) = +1$. Suppose $f_u(e_1)\colon (x, y) \mapsto a_f(e_1) + b_f(e_1)x + c_f(e_1)y$ and $f_l(e_2)\colon (x, y) \mapsto a_g(e_2) + b_g(e_2)x + c_g(e_2)y$. We compute the contribution of such cells to the sum (4), over all choices of p. (The remaining three types of cells adjacent to p are treated by an entirely symmetric argument.) Given an edge e of $\mathbf{T}_f \cup \mathbf{T}_g$, let $y = y(e) + s(e)x$ be the equation of the line supporting it, so we can write

$$\mathcal{T}(p, C_{\text{left}}, f_u(e_1)f_l(e_2)) = \int_{x=0}^{x_p} \int_{y=y(e_1)+s(e_1)x}^{y(e_2)+s(e_2)x} \begin{pmatrix} (a_f(e_1) + b_f(e_1)x + c_f(e_1)y)\cdot \\ (a_g(e_2) + b_g(e_2)x + c_g(e_2)y) \end{pmatrix} dydx.$$

The above integral can be expressed as the sum of nine integrals of a function of the form $v_k(e_1)w_k(e_2)x^{i_k}y^{j_k}$, $1 \le k \le 9$, where $0 \le i_k + j_k \le 2$ and v_k (resp., w_k) is a linear function in a_f, b_f, c_f (resp., a_g, b_g, c_g).

Example 1. Suppose the supporting lines of e_1 and e_2 have respective equations $y = 1 + 3x$ and $y = 3 - x$, so $x_p = 1/2$. If the linear function on the triangle above e_1 (resp., below e_2) is $f_u(e_1)\colon z = 1 + 10x + 100y$ (resp., $f_l(e_2)\colon z = -5+7x-2y$), then $x_p = 1/2$ and the integral we need to compute is

$$\mathcal{T}(p, C_{\text{left}}, f_u(e_1)f_l(e_2)) = \int_{x=0}^{x_p}\int_{y=1+3x}^{3-x}(1+10x+100y)\cdot(-5+7x-2y)dydx$$

$$= \int_{x=0}^{x_p}\int_{y=1+3x}^{3-x}\left(-5+7x-2y-50x+70x^2-20xy-500y+700xy-200y^2\right)dydx.$$

Note that we don't collect the terms under the integral, and keep the fully expanded form, so that each term on the right-hand side of the equation can be easily factored subsequently (notably in eqs. (7) and (8)).

Gathering all the terms and recalling that $p = e_1 \cap e_2$, eq. (4) can be rewritten, for each of the four types of cell $C_{\text{left}}, C_{\text{right}}, C_{\text{top}}, C_{\text{bottom}}$, as the sum of nine expressions of the form

$$\sum_{e_1\in R}\sum_{e_2\in B}\int_{x=0}^{x_{e_1\cap e_2}}\int_{y=y(e_1)+s(e_1)x}^{y(e_2)+s(e_2)x} v_k(e_1)w_k(e_2)x^{i_k}y^{j_k}dydx$$

$$= \sum_{e_1\in R}\sum_{e_2\in B}\frac{v(e_1)w(e_2)P_{i,j}(y(e_1),y(e_2),s(e_1),s(e_2))}{(s(e_2)-s(e_1))^{i_k+j_k+1}}, \quad (5)$$

by Lemma 5. Note that for different types, the integration is done over the same area, but the constants $v_k(e_1)$ and $w_k(e_2)$ come from coefficients of different piecewise linear functions.

5.2 Fast multi-point evaluation

Now we will use multi-point evaluation to speed up the computation of eq. (5). To accomplish this, we will replace the values associated to the edges of R by symbolic variables, while using the actual numerical values for the edges of B. Then we will compute the corresponding symbolic rational function using a divide-and-conquer strategy (Lemma 6). Finally, we will use multi-point evaluation on the resulting polynomials (Lemma 7).

We denote by $\mathcal{M}(q) = \mathcal{O}(q \log q \log\log q)$ the number of arithmetic operations needed to multiply two univariate polynomials of degree at most q (see (Gathen and Gerhard, 2003, Theorem 8.23)).

Lemma 6. *Let u and v be two functions from B to $\mathbb{R}$ and $|B| \leq \mu$. Then*

$$\sum_{e \in B} \frac{u(e)}{(X - v(e))^d} \tag{6}$$

can be expressed in the form

$$\frac{N(X)}{D(X)},$$

where $N(X)$ and $D(X)$ are polynomials of degree at most $(\mu - 1)d$ and μd respectively; their coefficients can be computed explicitly in $\mathcal{O}(\mathcal{M}(\mu d) \log \mu)$ arithmetic operations.

Proof. For simplicity of presentation and without loss of generality, assume that $|B|$ is a power of two. We bring eq. (6) to a common denominator by combining the fractions in pairs, reducing their number to $|B|/2$, and repeating the process $\log |B| = \mathcal{O}(\log \mu)$ times. The bounds on the degree of the final numerator and denominator are immediate from examining the original fractions.

We now explain how to bring

$$\frac{N_1(X)}{D_1(X)} + \frac{N_2(X)}{D_2(X)},$$

with N_1, N_2, D_1, D_2 of degree at most kd, to a common denominator in time $3\mathcal{M}(kd) + \mathcal{O}(kd)$. Indeed, the above fraction is equal to

$$\frac{N_1(X)D_2(X) + N_2(X)D_1(X)}{D_1(X)D_2(X)},$$

so it can be computed by three calls to fast polynomial multiplication plus a linear number of additional operations, as claimed.

This completes the proof of the lemma, as the cost of one round of combining fractions with denominators and numerators of degree at most kd is $\mu/k \cdot (3\mathcal{M}(kd) + \mathcal{O}(kd)) = \mathcal{O}(\mathcal{M}(\mu d))$, since $\mathcal{M}$ is superlinear. □

Remark 1. Although not directly needed for our algorithm, the above lemma also holds with $u(e)$ replaced by a polynomial in X of degree $\mathcal{O}(d)$ with coefficients depending on e, with the same proof and running time.

The second lemma handles summing the values of a polynomial of a special type.

Lemma 7. *Let $P(X_0, X_1, \ldots, X_r)$ be a polynomial of degree at most μ in X_0 and at most d in each of $X_1, \ldots, X_r$. Let E be a set of at most μ points of $\mathbb{R}^{r+1}$. The values of P at the points of E can be simultaneously computed in $\mathcal{O}(\binom{d+r}{r}\mathcal{M}(\mu)\log\mu)$ time.*

Proof. P can be expanded with respect to the variables $X_1, \ldots, X_r$ as follows

$$P = \sum_{1 \le i \le \binom{d+r}{r}} C_i(X_0) M_i(X_1, \ldots, X_r);$$

it has at most $\binom{d+r}{r}$ monomials M_i and each coefficient C_i is a univariate polynomial in X_0 of degree at most μ. Let $0 \le i \le \binom{d+r}{r}$. Using the standard multi-point evaluation algorithm for univariate polynomials (Gathen and Gerhard, 2003, Chapter 10), we can compute the μ values of C_i at every point of E in $\mathcal{O}(\mathcal{M}(\mu)\log\mu)$ arithmetic operations. For the evaluation of the M_i's, we order the monomials in a manner compatible with the division, i.e., so that, if M_i is of the form $X_k M_j$, for some $1 \le k \le r$, then $j < i$. One can evaluate M_i at a point of E in one multiplication, provided the value of M_j on this point has already been computed; thus the values of all the $\binom{d+r}{r}$ monomials M_i at the μ points of E can be constructed in $\binom{d+r}{r}\mu$ multiplications. In total, computing, multiplying, and adding these values costs $\mathcal{O}(\binom{d+r}{r}\mathcal{M}(\mu)\log\mu)$ arithmetic operations, as claimed. □

Now we are ready to efficiently evaluate eq. (5). Let μ be an upper bound on the size of A and B. Let $F_k(X, Y, V)$ be the polynomial

$$F_k(X, Y, V) := \sum_{e_2 \in B} \frac{V w_k(e_2) P_{i_k, j_k}(Y, y(e_2), X, s(e_2))}{(s(e_2) - X)^{i_k + j_k + 1}}.$$

After expanding the numerator of this fraction, we note that F_k has the form

$$\sum_{\substack{0 \le l \le j_k \\ 0 \le m \le i_k + j_k + 2}} \sum \left(\sum_{e_2 \in B} \frac{c_{l,m}(w_k(e_2), y(e_2), s(e_2))}{(s(e_2) - X)^{i_k + j_k + 1}} \right) X^l Y^m V. \tag{7}$$

In our case, $i_k + j_k + 1 \le 3$, and, using Lemma 6, we can compute each coefficient of $X^l Y^m V$ and express F_k in the following form, in $\mathcal{O}(\mu \log^2 \mu \log\log\mu)$ time:

$$F_k(X, Y, V) = \frac{N_k(X, Y) V}{D_k(X)}, \tag{8}$$

with N_k a polynomial of degree $(\mu-1)(i_k+j_k+1)+j_k$ in X, i_k+j_k+2 in Y, and D_k a univariate polynomial of degree $\mu(i_k+j_k+1)$.

Finally, eq. (5) can be rewritten as

$$\sum_{e_1\in R}\sum_{e_2\in B}\frac{v_k(e_1)w_k(e_2)P_{i_k,j_k}(y(e_1),y(e_2),s(e_1),s(e_2))}{(s(e_2)-s(e_1))^{i_k+j_k+1}}=\sum_{e_1\in R}\frac{N_k(s(e_1),y(e_1))v(e_1)}{D_k(s(e_1))}.$$

As $|R|\le\mu$ and degrees of N_k and D_k are $O(\mu)$ in the first argument and $O(1)$ in the others, using Lemma 7 we can simultaneously evaluate the numerators and denominators of all fractions in the summation, and thus compute the value of the expression in $\mathcal{O}(\mu\log^2\mu\log\log\mu)$ arithmetic operations.

5.3 Putting it together

To summarize, we can evaluate eq. (5) in $\mathcal{O}(\mu_k\log^2\mu_k\log\log\mu_k)$ operations, for each pair (R_k,B_k), where $\mu_k=\max\{|B_k|,|R_k|\}\le|B_k|+|R_k|$. We also saw in section 4 that $\sum_k(|R_k|+|B_k|)=\mathcal{O}(n\log^2 n)$, where n is the maximum number of triangles appearing in $\mathbf{T}_f$, $\mathbf{T}_g$. Therefore

$$\begin{aligned}\sum_k\mu_k\log^2\mu_k\log\log\mu_k&\le\sum_k(|R_k|+|B_k|)\log^2 n\log\log n\\&=\mathcal{O}(n\log^4 n\log\log n),\end{aligned}$$

which allows us to conclude that we can compute Σ_e in $\mathcal{O}(n\log^4 n\log\log n)$ time. This completes the proof of Theorem 1.

6 An application and a generalization

In Agarwal et al. (2010, 2013), the following optimization problem was considered: given two functions f and g and a distance measure $\|f-g\|$ between them (the paperAgarwal et al. (2010) discusses $\|\cdot\|_1$, $\|\cdot\|_2$, and $\|\cdot\|_\infty$, but we only consider $\|\cdot\|_2$ here), find the values of real parameters s and t that minimize $\|f-(sg+t)\|$. If f and g are interpreted as geometric "terrains," we are looking for the scaling and translation of the vertical coordinate of the terrain g to best match terrain f. Since $\|f-(sg+t)\|_2^2=\int(f-(sg+t))^2$ is a degree-two polynomial in s and t with coefficients easily expressible in terms of $\int f$, $\int g$, $\int f^2$, $\int g^2$, $\int fg$, and $\int 1$, being able to compute $\int fg$ efficiently immediately yields

Theorem 8. *Given piecewise-linear functions f and g defined over possibly different triangulations of the same domain M, with at most n triangles each, the real values s and t minimizing $\|f - (sg + t)\|_2$ can be computed*[1] *using $\mathcal{O}(n \log^4 n \log\log n)$ arithmetic operations.*

Theorems 1, 2, and 8 extend to piecewise-polynomial functions of constant maximum degree k with essentially no modifications. In this case, we need to compute a double sum of $\mathcal{O}(k^2 \cdot k^2)$ rational functions of the form (5), and each of them is further split into k^2 terms of the form (6), with $d = \mathcal{O}(k)$. Naïvely applying Lemmas 6 and 7 k^6 times results in $\mathcal{O}(k^7 n \log^4 n \log\log n)$ arithmetic operations, assuming $k < n$; notice that the variable V always occurs with exponent 1 in eq. (7). The dependence on k can be slightly improved by a more careful analysis. We omit the details.

In particular, Theorems 1 and 2 generalize to L_p norms, for even integer $p > 2$, at the cost of roughly a factor of p^4 in the running time; when p is odd, the function under the integral is no longer polynomial.

What other classes of functions can be handled using similar methods?

7 Extensions

We now consider two natural extensions of the problem studied in this paper. The first is the computation of the integral of the product of *three* functions, each defined over a potentially different triangulation of the same domain. The second involves integrating the product of two piecewise-linear *trivariate* functions defined over potentially different triangulations of the same three-dimensional domain, such as the unit cube.

7.1 Three bivariate functions

Unfortunately, the following theorem shows that this problem is as hard as solving 3-SUM, even for piecewise-constant functions Gajentaan and Overmars (1995).

Theorem 9. *Let A, B, and C be three sets of distinct positive integers, each of size n. Then in $\mathcal{O}(n \log n)$ time one can construct three piecewise-constant*

[1] By $sg + t$ we mean the function defined by $(x, y) \mapsto s \cdot g(x, y) + t$.

functions f, g, and h defined on the unit square, such that

$$\int f(x,y)g(x,y)h(x,y)dydx \neq 0 \Leftrightarrow \exists (a,b,c) \in A \times B \times C \colon a+b=c.$$

Proof. We are given sets $A = \{a_1, \ldots, a_n\}$, $B = \{b_1, \ldots, b_n\}$, and $C = \{c_1, \ldots, c_n\}$, of n distinct positive integers each. Sort them in $\mathcal{O}(n \log n)$ time. Compute the integer $Z := 1 + \max\{x \in A \cup B \cup C\}$.

Now define piecewise-constant functions f, g, and h on the unit square, as follows: f is 1 on the vertical strips $[(a_i - 1/10)/Z, (a_i + 1/10)/Z] \times [0,1]$, for each $i = 1, \ldots, n$, and 0 between these strips; g is 1 on the horizontal strips $[0,1] \times [(b_i - 1/10)/Z, (b_i + 1/10)/Z]$ and 0 between them. Function h defined on the unit square is 1 on the slanted strips delimited by the lines $x + y = (c_i \pm 1/10)/Z$, for each $i = 1, \ldots, n$, and 0 between the strips.

Notice that an explicit description of f, g, and h as piecewise-constant functions defined over suitable triangulations of the unit square can be constructed in linear time from the sorted sets A, B, and C.

The three functions are defined in such a way that their product is non-zero only at points where a vertical, a horizontal, and a diagonal strip intersect simultaneously. It is easily checked that such an intersection is possible if and only if $a_i + b_j = c_k$ for some indices i, j, k. Moreover, such an intersection, if it exists, has positive area. Therefore the integral of the product of the functions is indeed non-zero if and only if the original instance of the 3-SUM problem has a solution. This completes the description of an $\mathcal{O}(n \log n)$-time reduction of 3-SUM to our problem. □

It is not difficult to modify the above construction so that the functions f, g, and h are piecewise-linear and continuous.

7.2 Two trivariate functions

Given two trivariate function $f(x,y,z)$ and $g(x,y,z)$ defined over different triangulations of, say, the unit cube, we want to compute the integral of their product in subquadratic time; here again the naïve algorithm runs in quadratic time. To apply the approach used for bivariate functions, we need to solve the following subproblems:

(i) What is the natural generalization of the inclusion-exclusion formula of Lemma 3 for a convex polytope in three dimensions? What polyhedra should appear under the sum?

(ii) Can we ensure that the integrals of the monomials over the chosen polyhedra have the same form as in Lemma 5? In particular, can we ensure that the denominator contains only one variable depending on each three-dimensional triangulation?

(iii) Finally, does there exist a compact bipartite clique decomposition of the intersections in three dimensions and can it be computed in subquadratic time?

We leave resolving these issues for future work.

Acknowledgments

B.A. would like to thank Raimund Seidel for a spark of inspiration and some gentle encouragement. The authors would also like to thank Pankaj K. Agarwal, Christian Knauer, Sylvain Lazard, and Marc Mezzarobba for helpful discussions.

References

P. K. Agarwal, B. Aronov, M. van Kreveld, M. Löffler, and R. Silveira. 2010. Computing Similarity Between Piecewise-linear Functions. In *Proc. 26th Ann. Sympos. Comput. Geometry*. 375–383.

P. K. Agarwal, B. Aronov, M. van Kreveld, M. Löffler, and R. Silveira. 2013. Computing Correlation Between Piecewise-linear Functions. *SIAM J. Computing* 42, 5 (2013), 1867–1887.

D. Ajwani, S. Ray, R. Seidel, and H. Tiwary. 2007. On Computing the Centroid of the Vertices of an Arrangement and Related Problems. In *Proceedings Symp. Algo. Data Structures (WADS'07)*, Vol. LNCS 4619. 519–528.

B. Chazelle, H. Edelsbrunner, L. J. Guibas, and M. Sharir. 1994. Algorithms for Bichromatic Line Segment Problems and Polyhedral Terrains. *Algorithmica* 11 (1994), 116–132.

G. Elekes, H. Kaplan, and M. Sharir. 2011. On Lines, Joints, and Incidences in Three Dimensions. *J. Combinat. Theory, Ser. A* 118 (2011), 962–977.

G. Elekes and M. Sharir. 2010. Incidences in Three Dimensions and Distinct Distances in the Plane. In *Proc. 26th Annu. ACM Sympos. Comput. Geom.* 413–422.

A. Gajentaan and M. H. Overmars. 1995. On a Class of $O(n^2)$ Problems in Computational Geometry. *Computational Geometry* 5, 3 (1995), 165–185.

J. von zur Gathen and J. Gerhard. 2003. *Modern Computer Algebra* (2nd ed.). Cambridge University Press.

L. Guth and N. H. Katz. 2010. Algebraic Methods in Discrete Analogs of the Kakeya Problem. *Advances in Mathematics* 225, 5 (2010), 2828–2839.

L. Guth and N. H. Katz. 2011. On the ErdŐs Distinct Distance Problem in the Plane. (2011). `arXiv:1011.4105v3 [math.CO]`.

H. Kaplan, J. Matoušek, and M. Sharir. 2012. Simple Proofs of Classical Theorems in Discrete Geometry Via the Guth–Katz Polynomial Partitioning Technique. *Discrete Comput. Geometry* 48, 3 (2012), 499–517.

H. Kaplan, M. Sharir, and E. Shustin. 2010. On lines and joints. *Discrete Comput. Geometry* 44, 4 (2010), 838–843.

K. S. Kedlaya and C. Umans. 2008. Fast Modular Composition in Any Characteristic. In *Proc. 49th Annu. IEEE Symp. Found. Comput. Science (FOCS'08)*. 146–155.

D. Kirkpatrick. 1983. Optimal Search in Planar Subdivisions. *SIAM J. Comput.* 12, 1 (1983), 28–35.

J. Matoušek. 2011. The Dawn of an Algebraic Era in Discrete Geometry? *27th European Workshop on Computational Geometry (EuroCG 2011)* (2011). extended abstract.

G. Moroz and B. Aronov. 2012. Computing the Distance Between Piecewise-linear Bivariate Functions. In *Proceedings of Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2012)*. 288–293.

M. Nüken and M. Ziegler. 2004. Fast Multipoint Evaluation of Bivariate Polynomials. In *Proc. European Symp. Algo. (ESA 2004)*. 544–555.

R. Quilodrán. 2010. The Joints Problem in $\mathbb{R}^n$. *SIAM J. Discrete Math.* 23, 4 (2010), 2211–2213.

J. Solymosi and T. Tao. 2012. An Incidence Theorem in Higher Dimensions. *Discrete Comput. Geometry* 48, 2 (2012), 255–280.