Almost-Delaunay Simplices: Nearest Neighbor Relations for Imprecise Points

Deepak Bandyopadhyay * Jack Snoeyink†

Abstract

Delaunay tessellations and Voronoi diagrams capture proximity relationships among sets of points in any dimension. When point coordinates are not known exactly, as in the case of 3D points representing protein atom coordinates, the Delaunay tessellation may not be robust; small perturbations in the coordinates may cause the Delaunay simplices to change. In this paper, we define the *almost-Delaunay simplices*, derive some of their properties, and give algorithms for computing them, especially for neighbor analysis in three dimensions. We sketch applications in proteins that will be described more fully in a companion paper in biology. http://www.cs.unc.edu/~debug/papers/AlmDel.

1 Introduction

The Voronoi diagram and Delaunay tessellation, which are geometric structures defined for sets of points, have found use in many areas of science and engineering [5, 13, 9, 34].

These diagrams are defined by exact geometric criteria. In many applications, however, point coordinates are known only imprecisely, so it is natural to ask whether the uses of Voronoi and Delaunay are stable and robust under changes to the input coordinates. In this paper, we consider the possible structures that could be defined by nearby inputs. Specifically, we focus on the Delaunay tessellation, and consider the *almost-Delaunay simplices*, which are additional sets of sites that could become Delaunay simplices if all sites are perturbed by a minimum amount $\epsilon \geq 0$.

In the next section, we list several problems from protein structure analysis that motivate the definition of almost-Delaunay simplices. In Section 3 we relate almost-Delaunay simplices to a variant of the minimum-width annulus problem, which allows us to determine several properties of almost-Delaunay simplices and sketch a straightforward, but slow, algorithm to compute them. In Section 4 we consider the properties of our applications and derive efficient algorithms to compute almost-Delaunay edges, triangles and tetrahedra in two and three dimensions.

*Department of Computer Science, University of North Carolina at Chapel Hill. {debug,snoeyink}@cs.unc.edu

†Portions of this research were supported by NSF grant 0076984.

2 Motivation & Definitions

We first need the following definitions for a finite set of point *sites* $\mathcal{S} \in \Re^d$. A k-simplex is the convex hull of $k+1$ affinely independent sites; the simplices in $\Re^3$ are points, edges, triangles and tetrahedra formed by sites of $\mathcal{S}$. The *Voronoi diagram of* $\mathcal{S}$ is the decomposition of space into regions with the same set of closest neighbor sites [44]. The dual *Delaunay tessellation* is a decomposition of the same space based on an "empty sphere property:" [14] if a subset of sites, $S \subset \mathcal{S}$, lie on the boundary of a sphere that is otherwise empty of sites, then the convex hull of S is a region of the Delaunay tessellation. It is common to assume, or to simulate, that the sites $\mathcal{S}$ are in general position—no $d+1$ sites may be co-planar or $d+2$ sites co-spherical in $\Re^d$—so that the Delaunay regions are simplices.

Proteins are long chains of amino acids that reliably fold into 3-d structures that perform the functions essential for life [25, 24]. The genomics revolution has produced powerful tools to manipulate the sequence of amino acids, but it is the structure of a protein that determines its function. Crystallography and NMR give us information about the structures of many molecules; the Protein Data Bank (PDB) contains the coordinates of atoms for over 20,000 proteins [1]. One of the most important open problems in science, the "protein folding problem," is to derive 3-d structure directly from the amino acid sequence.

Many researchers have found Delaunay tessellations and Voronoi diagrams to be useful tools for problems of protein structure analysis, such as

> **Protein geometry definition** Given a set of labeled points representing atoms or residues of a protein, what is the protein's volume, surface, and density? What are its cavities and pockets?
>
> Richards [36] pioneered the use of Voronoi diagrams to compute protein volumes. This has been an active research area, with more detailed empirical analysis of parameters [42, 43], refinements on the definition of the surface [29, 30], and analysis of differential packing in the core and surface regions [26]. The Delaunay tessellation has been used to define and detect pockets and cavities [6, 27, 28].
>
> **Decoy discrimination** Given a collection of "decoy" structures for the same protein, (often generated by

structure prediction programs), determine which is the most native-like.

Both Voronoi and Delaunay have been used to score residue interactions in folded proteins and decoys. The Voronoi diagram naturally assigns a region to each atom or representative point, and the contact area between residues has been incorporated into "two-body" potentials [48, 4]. The Delaunay tessellation collects sets of four "neighboring" representative points into tetrahedra. Researchers have analyzed the frequency of occurrence of different amino acid types in tetrahedra to develop empirical four-body potentials [11, 33, 40, 46]. These four-body potentials complement fragment-based methods [39] and pairwise potentials [32] to capture favorable or unfavorable packing interactions, which are difficult to efficiently incorporate into structure prediction codes.

Motif detection What shapes or structure fragments occur frequently in proteins?

The Voronoi diagram has been used to partition proteins into structural domains with minimal interaction between them [47]. Wako and Yamato use the Delaunay tessellation of C_α carbons and find patterns of the backbone sequence among neighboring tetrahedra to identify local motifs for helices and sheets [45].

Small changes in the coordinates chosen to represent atoms or residues can produce different sets of Delaunay simplices. We would like to identify the simplices that are *almost Delaunay* for a finite set of sites $\mathcal{S}$.

DEFINITION 2.1. (ALMOST-DELAUNAY) *A k-tuple $Q \subset \mathcal{S}^k$ is in the set of* almost Delaunay simplices $AD(\delta)$ *iff, by perturbing each site of $\mathcal{S}$ by at most $\delta \geq 0$, the perturbed Q becomes a Delaunay k-simplex—the perturbed Q has an empty circumscribing sphere. We say that Q is* almost Delaunay with threshold ϵ *iff* $\epsilon = \liminf\{\delta \mid Q \in AD(\delta)\}$.

Every k-tuple for $1 \leq k \leq d+1$ is an almost Delaunay simplex for some finite threshold; k-tuples with threshold 0 are Delaunay. In general, the sets of $AD(\delta)$ do not form a simplicial complex—$AD(\delta)$ includes the faces of its simplices, but not necessarily the intersections of simplices.

Abellanas *et al.* [2] explored the robustness of a Delaunay triangulation in the plane, giving a complementary definition of *tolerance* as the supremum of all $\epsilon > 0$ such that $AD(\epsilon)$ contains only the Delaunay triangles. They used a *minimum-width annulus problem*, defined below, to find value of ϵ over the whole triangulation and for individual Delaunay edges. We generalize to ϵs for all simplices, and to higher dimensions.

Other generalizations of Voronoi and Delaunay diagrams have been studied extensively in computational geometry. Order-k Voronoi diagrams [9, 13, 12] are a generalization that uses the regions with the same set of k-closest neighbors. Gudmundsson *et al.*[21] have studied the related order-k Delaunay triangulations, which include the tetrahedra whose circumspheres can become empty if $k-4$ points are deleted. Although this definition is easier to analyze and implement, it is less natural in our analysis of perturbations. Points that require a large perturbation to remove from a circumsphere can be deleted all too easily.

Perturbation is most often studied to put points in general position, so that implementations of algorithms have fewer special cases to handle. For example, the regions of the Delaunay tessellation are simplices only if the points are in general position (e.g. no five points are cospherical). Thus, codes for computing the Delaunay usually enforce general position by a random perturbation [8] or symbolic perturbation [19], but otherwise take the input data as exact.

Papers that do consider computation from inexact input usually search for only one output that is consistent with the input [22]. When one consistent output exists, then there may be exponentially many—even in a 2D Delaunay triangulation, two rows of grid points can be perturbed to form $2^{(n-1)/2}$ triangulations by selecting which diagonals of the grid squares to draw. This is why we define almost-Delaunay simplices instead of almost-Delaunay tessellations—we know that the number of simplices is less than n^{d+1}, which helps keep the computation feasible.

3 Almost-Delaunay as an annulus problem

The computation of almost-Delaunay thresholds can be related to a minimum-width annulus problem from computational metrology. In the plane, a set of points is tested for roundness by the smallest width between two concentric circles that form an annulus enclosing all the points. This *roundness problem* has been studied extensively [18, 37, 38]. Solutions are known for two and higher dimensions, and for various special cases [3, 15, 31, 35].

We now review the definition of an annulus in d dimensions, and define some useful notation.

DEFINITION 3.1. (d-ANNULUS) *A d-annulus is a set $\{p \in \Re^d : r \leq dist(p,c) \leq R\}$, defined by its* center c, *its* inner *and* outer *radii, r and R, and its* width, $w = R - r$. *We allow the center to be a point at infinity, in which case the inner and outer radii are infinite and the annulus is actually a slab of width w defined by two parallel hyperplanes [41].*

Finding the almost-Delaunay threshold ϵ for a k-simplex can be formulated as a variant of the problem of finding an annulus of minimum width:

THEOREM 3.1. (ALMOST-DELAUNAY) *Given finite sets of points $\mathcal{Q} \subset \mathcal{S} \subset \Re^d$, let $\mathcal{A}$ denote the d-annulus of minimum width that contains $\mathcal{Q}$ and whose inner hypersphere is empty of points from $\mathcal{S}$. Then $width(\mathcal{A}) = 2\epsilon$ if and only if $\mathcal{Q}$ is almost-Delaunay with threshold ϵ.*

Proof. Suppose that $\mathcal{Q}$ can be enclosed by an annulus $\mathcal{A}$ with center c, inner radius r, and width 2ϵ so that the inner (hyper)sphere is empty of $\mathcal{S}$. As in Figure 1, every point of $\mathcal{A}$ is within ϵ of the *medial (hyper)sphere*, M, having center c and radius $r + \epsilon$. Thus, there is a perturbation of $\mathcal{Q}$ onto M and $\mathcal{S}$ outside of M showing that $\mathcal{Q} \in AD(\epsilon)$.

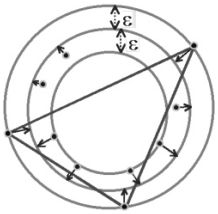

Figure 1: an $AD(\epsilon)$ triangle

If simplex $\mathcal{Q} \in AD(\epsilon)$, then we can bound the annulus width as follows. By definition, we can perturb the points $\mathcal{S}$ to $\mathcal{S}'$ so that the perturbed simplex $\mathcal{Q}'$ lies on an empty (hyper)sphere M. Note that each point of $\mathcal{S}$ is within distance ϵ of its corresponding point $\mathcal{S}'$, so if we construct the annulus $\mathcal{A}$ with the same center as M and outer and inner spheres offset by $\pm\epsilon$, then the inner sphere of $\mathcal{A}$ is empty of $\mathcal{S}$ because M was empty of $\mathcal{S}'$, and outer sphere of $\mathcal{A}$ contains $\mathcal{Q}$ because M contained $\mathcal{Q}'$.

Thus, $\mathcal{Q}$ is in $AD(\epsilon)$ iff there is an annulus of width 2ϵ, and the *minimum* annulus width equals the threshold of $\mathcal{Q}$. □

3.1 Configurations that determine ϵ

The threshold ϵ for an almost-Delaunay simplex $\mathcal{Q} \subseteq \mathcal{S}$ is determined by properties of its associated minimum-width annulus. These properties, and their derivations, are similar to those for minimum-width annuli enclosing all points [20, 41].

Recall that the *Voronoi diagram* is the partition of space into maximally connected regions with the same set of closest neighbor sites. The *furthest-point Voronoi diagram* is the partition of space into maximally connected regions with the same set of furthest neighbor sites [5, 9, 13, 34].

Modifying notation of [20], any *candidate center* point c has a set $CN(c)$ of closest neighbors in the Voronoi of $\mathcal{S}$, and a set $FN_{\mathcal{Q}}(c) \subset \mathcal{Q}$ of furthest neighbors in the furthest-point Voronoi of $\mathcal{Q}$. Note that c is in the intersection of closest- and furthest-point Voronoi regions corresponding to sets CN and $FN_{\mathcal{Q}}$.

An easy technical lemma establishes a geometric condition on the directions of motion that decrease the difference in distances to two points.

LEMMA 3.1. *Consider two points p and q, and the difference in their distances to the origin, $d(p,0) - d(q,0)$. Let v be a unit vector, and α and β be its angles to p and q respectively. Suppose that we move the center c of a minimum-width annulus by λv for some infinitesimal λ. The directional derivative*

$$\frac{d}{d\lambda}\big(d(p,\lambda v) - d(q,\lambda v)\big) = \cos(\beta) - \cos(\alpha),$$

which is negative if angle $0 \leq \alpha < \beta \leq \pi$.

As a corollary, we have a variant of the "cone condition" of [20] to further characterize the center c.

THEOREM 3.2. (CONE CONDITION) *From a finite center c of a minimum-width annulus associated with simplex $\mathcal{Q} \subset \mathcal{S}$, no cone separates $CN(c)$ from $FN_{\mathcal{Q}}(c)$.*
From an infinite center c, no plane separates $CN(c)$ from $FN_{\mathcal{Q}}(c)$ and any cone (that is, cylinder) that separates them has $CN(c)$ inside.

Proof. For a finite point c, if there is a separating cone, then one of the directions along the axis of the cone has angles to sites of $FN_{\mathcal{Q}}(c)$ smaller than those of $CN(c)$. Lemma 3.1 says that the directional derivative is negative, so that c cannot be the center of a minimum-width annulus.

For an infinite point c, if there is a separating cylinder with $CN(c)$ outside, then moving c to a finite point produces an annulus with smaller width. If there is a separating plane, then moving c on the plane at infinity decreases the width between the parallel planes. □

We can now count the points needed to define a minimum-width annulus.

THEOREM 3.3. (NUMBER OF POINTS) *Suppose that the k-simplex $\mathcal{Q} \subset \Re^d$ has a minimum-width annulus $\mathcal{A}$ with center c. Then $2 \leq |FN_{\mathcal{Q}}(c)| \leq \min(d, |Q|)$, and $|CN(c)| + |FN_{\mathcal{Q}}(c)| \geq d + 2$ for finite c, and $\geq d + 1$ for c at ∞, with equality holding in general position.*

Proof. A single closest or furthest point can always be trivially separated from the other points by a cone with vertex at c enclosing just that point, so $n_F = |FN_{\mathcal{Q}}(c)| > 1$. Also, $n_C = |CN(c)| \geq 1$, with equality only when center c is at infinity.

To bound the sum $n_F + n_C$, consider the relatively open Voronoi regions for $FN_{\mathcal{Q}}(c)$ and $CN(c)$, which are regions of dimensions $\geq d + 1 - n_F$ and $\geq d + 1 - n_C$, equality holding in general position. Lemma 3.1 shows that the minimum width occurs for a point on the boundary of the intersection—if the intersection contains more than a single finite point c, then the minimum must occur at infinity, because moving to a finite boundary will take the point out of $CN(c)$ or $FN_{\mathcal{Q}}(c)$—one of these sets will gain an additional site. The dimension of the intersection is at least $d - (d+1-n_F) - (d+1-n_C) = n_F + n_C - d - 2$. Thus, when c is finite, we have $0 \leq n_F + n_C - d - 2$, or $n_F + n_C \geq d + 2$, and when c is infinite, we have $1 \leq n_F + n_C - d - 2$, or $n_F + n_C \geq d + 1$. □

Remark: The case with $c = \infty$ and $|CN(c)| = 1$ occurs in our variant on the annulus problem; Smid and Janardan [41] had observed that it never defines the width of a minimum-width annulus that encloses all points.

Finally, we prove a fact asserted in the previous section.

LEMMA 3.2. (SIMPLEX THRESHOLD) *The almost-Delaunay threshold for a k-simplex in $\Re^d$ is at least as high as that for each of the m-simplices that constitute it, $m < k$.*

Proof. A perturbation ϵ that makes the k-simplex Delaunay creates an empty hypersphere containing all of the points of the simplex after perturbation. □

The above properties immediately suggest a simplistic algorithm: For each of the $\binom{n}{d+1}$ simplices, maintain a minimum threshold seen, which is initially infinite. For all sets of $d+2$ points and all $2 \leq j \leq d+1$, consider the annuli defined by j points on the outer hypersphere and $d-j+2$ on the inner hypersphere. If the inner hypersphere of an annulus is empty of all other points, update the minimum threshold seen for each k-simplex from among the $d+2$ points. This algorithm would take $O(\binom{n+d}{d+2} \cdot n) = O(n^{d+3})$ time. Since the points on the inner hypersphere form a Delaunay simplex, we can reduce this to $O(n^{d+2})$ time for the d-simplices, and less for the lower-dimensional simplices.

If we do want thresholds for all $\binom{n}{d+1}$ simplices, then there is not much room to improve this algorithm; we'd expect to spend between n^4 and n^5 time in 3D. In our applications, however, we are interested in a subset of simplices, either those with small thresholds or those with short edges, or both. In the next section, therefore, we will sketch worst-case analyses primarily to help make the algorithm clear, but will concentrate on algorithms that have good practical performance, as demonstrated by experiments and by analysis under assumptions such as even or random distribution of points.

4 Algorithms for relevant thresholds

In this section, we identify "relevant" parameters that are monotone increasing in simplex dimension, so that if the parameter for a simplex σ is too large, then the parameter for any simplex that has σ as a face will also be too large. Thus, we begin by computing almost-Delaunay edges, as in Figure 2. Through these and other geometric observations, we make practical improvements, and attempt to explain these improvements by analysis and experiments.

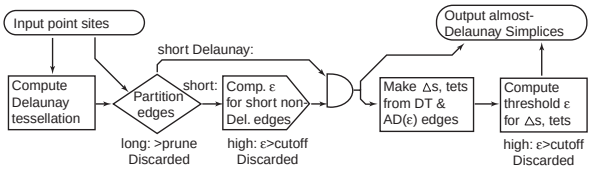

Figure 2: Processing AD edges then AD simplices

4.1 Parameters

Threshold cutoff: For analyzing robustness, we are primarily interested in small perturbations, relative to interpoint distance. We set a *threshold cutoff* (or ϵ_{cutoff}) and consider only the simplices in $AD(\epsilon_{cutoff})$. Equivalently, we discard any edge with AD threshold $\epsilon > \epsilon_{cutoff}$, and, by Lemma 3.2, also the simplices that use these edges. Unless otherwise specified, $\epsilon_{cutoff} = 2.0$ Å in our protein experiments, where the typical distance between C_α carbons is 5–6Å.

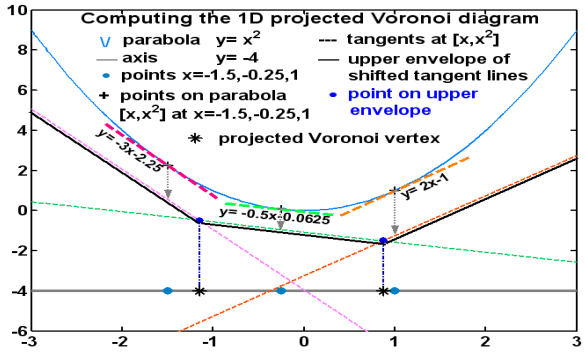

Figure 3: The lifting technique, illustrated on a 1D axis.

Edge length prune: For analyzing neighbor interactions in proteins, only pairs of points within about 10 Å of each other are relevant. We set an *edge length prune* parameter to restrict the maximum length of any edge of the simplices whose AD thresholds we evaluate. Edge pruning also eliminates long Delaunay edges near the convex hull that are not really between neighbors. We use a 10 Å prune by default.

As illustrated in Figure 2, we find the threshold for each k-simplex that satisfies the prune requirement, starting with the 1-simplices (edges), and discard the simplices with threshold higher than the cutoff before proceeding to the next stage. For each remaining simplex, we use a *lifting map* to find the candidate centers. This makes the time and space dependent on the number of relevant simplices, rather than the total.

4.2 Computation of candidate centers by a lifting map

We modify Brown's *lifting* technique [10, 13] to compute the Voronoi diagram of the points projected onto the furthest-point Voronoi region of k furthest neighbors, but with distances based on the original points. We sketch the technique in d dimensions, and illustrate it in Figure 3 for points projected on a 1D *axis*.

The proof of Theorem 3.3 observes that k furthest neighbors define a region of dimension $d-k+1$ in the furthest-point Voronoi diagram in $\Re^d$, which is a subspace if we choose as origin the centroid of the furthest neighbors. The distance from the origin to a point $\mathbf{p}$ can be factored into components $\mathbf{x} = \{x_1, \ldots, x_{(d-k+1)}\}$ within this subspace and h which is the Euclidean norm of all orthogonal components, in an orthogonal dimension y. We construct tangents to the unit paraboloid $y = {x_1}^2 + \ldots + {x_{d-k+1}}^2$ at values $\mathbf{x}$ corresponding to each point $\mathbf{p}$, then shift each tangent "down" in the negative y direction by h^2. Now at any point $\mathbf{x}'$ within the subspace, the distance in the y direction between the shifted tangent at point $\mathbf{x}$ and the paraboloid at $\mathbf{x}'$ equals the square of the distance between $\mathbf{x}'$ and $\mathbf{p}$.

Using the above property, the intersection points of two shifted tangents along the upper envelope of these tangents

give the vertices of the Voronoi diagram, which are the candidate centers. The intersection points can be computed in dual space by finding the lower convex hull of the dual points to these tangent planes. We use the Quickhull[8] implementation which runs in expected $O(n \log n)$ time in 2 and 3 dimensions. Lifting avoids higher-dimensional Voronoi diagrams that are hard to compute robustly, and the lifting implementation is easy and efficient for the dimension (3) and problem sizes (n=100–10,000) we are interested in.

4.3 Computing almost-Delaunay edges

The pseudo-code for the algorithm is outlined below.

- Enumerate all non-Delaunay edges, (i, j).
- For each non-Delaunay edge (i, j) that is shorter than the edge length prune, do:
  1. Compute the equation of P, the furthest-point Voronoi region corresponding to i and j. For example, in 3D this is the bisector plane of $\overline{ij}$.
  2. Find the intersections of the Voronoi diagram with P using a lifting map. These are candidate centers with i and j as furthest neighbors and the points of a Delaunay simplex as closest neighbors, and include Voronoi vertices at infinity.
  3. Evaluate annulus width function at all candidate centers, and record the minimum annulus width δ.
  4. Record the value of $\delta/2$ as the AD threshold for edge (i, j), if it is less than ϵ_{cutoff}.
- Report the edges with threshold values between 0 and ϵ_{cutoff} as almost-Delaunay edges.

4.4 Computing almost-Delaunay k-simplices

We can easily modify the algorithm for an AD edge in any dimension to find the minimum threshold (ϵ) when that edge is part of a k-simplex. The only constraint is that each candidate annulus should contain (not necessarily as closest or furthest neighbors) all the other points of the simplex. The AD threshold for the simplex is then the minimum constrained edge threshold over all its edges. Now we describe efficient ways of computing the AD threshold for a few special cases.

4.4.1 Almost-Delaunay triangles in 2D

For AD edge (p, q) of $\triangle pqr$, we will consider all candidate centers c for which the distance to p or q is greater than the distance to point r. The candidate centers are computed using the 1D lifting map of the points as discussed in Section 4.2. The distance to point r is given by the tangent line to the parabola $y = x^2$ at r_x, shifted down by ${r_y}^2$, where r_x and r_y are the on-axis and off-axis components of r. At the point where the shifted tangent line at r intersects the shifted horizontal tangent line at points p and q, all three points are equidistant. Thus there is a 1D *half-line constraint* on the candidate centers generated by the condition that point r be

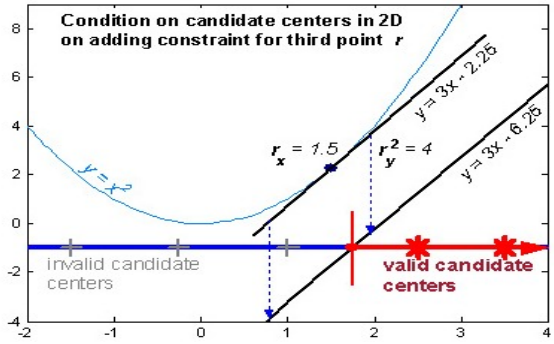

Figure 4: Finding the AD threshold for $\triangle pqr$ in 2D by adding a 1D half-line constraint generated by r.

closer to the center than p and q. If r_x is positive, and hence the slope of the tangent line is positive, this constraint is satisfied by all candidate centers to the right of the equidistance point of p,q and r, which we call a *left cutoff*. This is illustrated in Figure 4. Similarly if r_x is negative, the slope of the tangent is negative and the constraint is satisfied by all candidate centers on the left of the equidistance point, which we call a *right cutoff*.

4.4.2 Almost-Delaunay triangles/tetrahedra in 3D

Given a tetrahedron $pqrs$, assume that its associated minimum-width annulus has points p and q as furthest neighbors. From all the candidate annuli for edge $\overline{pq}$, we need to consider only those which contain both the points r and s. The annulus centers are computed using a 2D lifting map constructed on the bisector plane of $\overline{pq}$ as described in Section 4.2. The condition that a point r be closer than p or q generates a 3D *half-plane constraint* on the annulus center, 2D when projected onto the bisector plane. Thus finding the minimum annulus containing p, q and r is equivalent to finding the minimum annulus for edge $\overline{pq}$ from among the candidate centers (including centers at infinity) that satisfy the half-plane constraint generated by r, and symmetrically for edges $\overline{pr}$ and $\overline{qr}$. For tetrahedron $pqrs$, the two half-planes for r and s form a wedge on the bisector plane of $\overline{pq}$, which gives us a region in which to search for centers of annuli containing the points r and s.

The center of the minimum-width annulus for an edge in 3D always has 2 furthest neighbors, and either 3 closest neighbors for a finite center, or 2 closest neighghbors when the center is at ∞ (Theorem 3.3). There is a third possibility for finding the minimum threshold of an AD triangle, that the center lies on a Voronoi edge between two candidate centers, at its intersection with a half-plane constraint. This corresponds to three furthest neighbors (two for the bisector plane and one for the constraint) and two closest neighbors. Also the points on each constraint at $\pm\infty$ correspond to slabs with three furthest neighbors and one closest neighbor, the one whose Voronoi region the point is in.

For an AD tetrahedron, candidate centers of edge $\overline{pq}$ and Voronoi edges between them may contribute to the minimum-width annulus if they satisfy both half-plane constraints for points r and s. For efficiency, we separate the edges into two groups. Each edge that intersects only one constraint is completely on one side of the other constraint, so it suffices to test any one of its points against

the other constraint, a test whose result is already known from the AD triangles. Edges that intersect both constraints are valid if the intersection point of each constraint satisfies the other constraint, a more expensive but relatively infrequent test.

Recall that we computed the AD threshold only for non-Delaunay edges, since for Delaunay edges it is zero by definition. In 3D there are AD triangles and tetrahedra with all edges Delaunay, and we enumerate and handle them separately for efficiency.

4.5 Acceleration for applications

The algorithm as given above for 3D takes $O(n^3 \log n)$ time to run in the worst case for just the AD edge computation, with $\Theta(n^2)$ for computing the Delaunay, $O(n^2)$ for enumerating short edges, and then $O(n \log n)$ for finding the candidate centers for each short edge. Note that centers of some minimum width annuli lie arbitrarily far from the midpoint of the short edge, and thus we have to compute $O(n \log n)$ candidate centers and may not search within a region near the edge. Addition of a third and fourth point to the annuli and computation of the constrained minimum with one and two half-plane constraints takes $O(n)$ and $O(n^2)$ time for each candidate center. Thus with n sites and $\binom{n}{4}$ potential AD tetrahedra, testing all of them takes $O(n^5 \log n)$ time.

We can improve the algorithm in 3D to take $O(n^2 \log n)$ time by exploiting properties of minimum-width annuli and assumptions made about the input data. We concentrate on practical improvements for the number of points found in proteins; further asymptotic improvement may be possible.

4.5.1 Effect of edge length pruning

Pruning long edges speeds up the algorithm by considering only AD edges both of whose points lie in a bounded spherical region, a bucket. Pruning needs to be done only once, and then the complexity of the rest of the algorithm is determined by the size of the set of pruned (short) edges. This step reduces the search space of AD edges considerably — from $\binom{n}{2}$ total edges, we get $O(n^2)$ short edges in the worst case, but $O(n)$ edges under the assumption that points have nearly uniform packing density.

LEMMA 4.1. (NUMBER OF SHORT EDGES) *The expected number of edges shorter than an edge length prune p in n points closely packed such that the closest distance between 2 points is $2r$, is $O(n \cdot p^3)$ for constant r, or $O(n)$ for constant p and r.*

Proof. We use a volume packing argument: Represent each residue by a ball of radius $\sim r$; balls pack within the volume of the protein without overlap. Denote the sphere of radius p around residue A as its *prune volume*; all other residues B_i whose centers lie within this sphere would form short edges $\overline{AB_i}$. There can be a maximum of $0.78(p+r)^3/r^3$ spheres within the prune volume, since spheres in close packing can fill up at most 78% of the volume they occupy. Thus the total number of short edges is $O(n \cdot p^3)$ for constant r, or $O(n)$ for constant p and r. □

This lemma is clearly not true for arbitrary data, as we can get $O(n^2)$ short edges if we take uniformly distributed data in a fixed volume, and arbitrarily increase n and the packing density. In the limit, each newly added point adds a number of short edges proportional to the number of points already existing, and this adds up to $O(n^2)$ edges. However, the lemma holds for close-packed protein data where two C_αs are a minimum of 3 Å apart, and two sidechain centroids are even further, say 5 Å.

Increasing the edge length prune parameter allows longer edges to exist on the outer sphere of the annulus, and in turn allows tetrahedra with larger thresholds, from Lemma 3.2. As shown in Figure 5 for a typical protein with $\sim$ 180 residues, the maximum AD edge threshold at any edge length increases linearly with the edge length, which justifies picking a value for ϵ_{cutoff} based on the value of edge length prune.

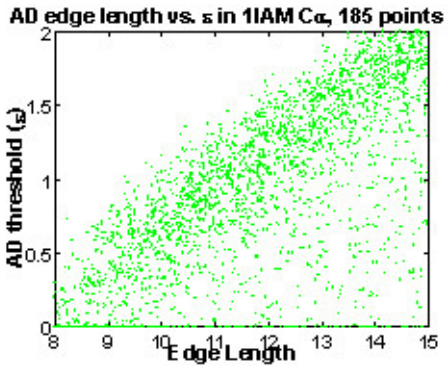

Figure 5: Scatter plot showing correlation of AD edge threshold with edge length for a protein with 185 points.

4.5.2 Angle pruning

Using the properties from Section 3.1, we get conditions for valid Voronoi edges and valid configurations of points in 2D and 3D, which can be used to prune away candidate centers that cannot give minimum-width annuli.

In 2D, if the furthest neighbors are p and q for a center c, exactly one of the two closest neighbors is inside the triangle $\triangle cpq$, or within the cylinder from c at infinity bounded by p and q. In 3D, the cone condition can be checked just like 2D once we fix the axis of the cone as the angle bisector of two furthest or closest neighbors, by rotating all other points into the same plane.

In experiments on 3D protein data, we found that between 80 and 85% of candidate centers get eliminated by angle pruning. But we have to test every candidate center, and the pruning computation for a center needs more work than just evaluating the annulus width. Thus, angle pruning is useful only if we can use its results to prune the candidate centers and edges between centers for computation of AD triangles and tetrahedra. This is easy to see for the centers;

for the edges we keep them if either endpoint satisfies the angle prune conditions, or is closer to the origin than an empirically determined distance of $prune + cutoff^2$. For both centers and edges, we can apply angle pruning in conjunction with threshold cutoff, and results are shown in Figure 6 for a cutoff of 1.0 for a test dataset of 440 proteins, and tabulated later in Section 4.7.

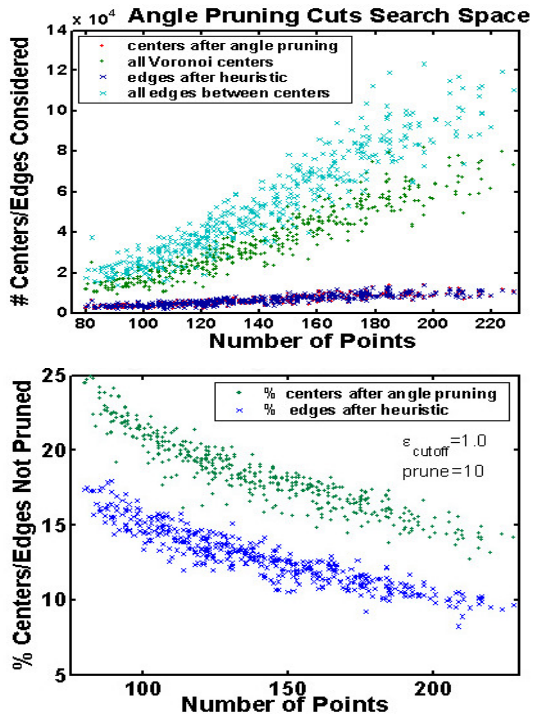

Figure 6: Angle pruning of candidate centers and our edge pruning heuristic greatly reduce the number of centers and edges considered for each AD triangle and tetrahedron.

4.6 Time complexity with assumptions about data

Edge-length pruning speeds up the AD edge algorithm by a factor of n to run in $O(n^2 \log n)$ expected time on typical protein data, as proved in Lemma 4.1. We shall now argue that the AD triangle and tetrahedron algorithms run in expected $O(n^2)$ time after the edges have been processed, for an algorithm that is $O(n^2 \log n)$ overall, and scales very well from edges to higher dimensional simplices.

LEMMA 4.2. *The expected time taken to check all the candidates for AD triangles and tetrahedra, once the AD edges are known, in n points closely packed such that the closest distance between 2 points is $2r$, and the edge length prune is p, is $O(n^2)$ for constant p and r.*

Proof. Since each point has $O(1)$ short edges to its neighbors (Lemma 4.1), each short edge forms a constant number of triangles and tetrahedra with all short edges. For each triangle there is one half-plane constraint and for each tetrahedron there are two half-plane constraints induced on the candidate centers and edges. Previous work by Dwyer [16, 17] shows the expected number of k-faces in the Voronoi diagram of n independent, uniformly distributed sites in $\Re^d$ to be $O(n)$. We can apply this result to our computation of the lifting map if we assume the projections of our data points to be uniformly distributed, and thus can bound the number of candidate centers per short edge and the number of edges connecting these centers by $O(n)$.

After angle pruning and applying the threshold cutoff, the number of candidate centers and edges is still $O(n)$, though it may be sublinear as we shall see in Section 4.7. Now finding all the AD triangle and tetrahedron thresholds involves checking the threshold of $O(n)$ valid candidate centers and edges against $O(1)$ half-plane constraints, and doing this for $O(n)$ short edges. Thus we get the time complexity as $O(n^2)$. □

4.7 Practical performance

To show that the practical behavior is not worst-case, but fits the theoretical bounds based on assumptions about our data, we estimated the performance of the algorithm in terms of the number of simplices checked at each stage and the output size, fitted using regression as shown in Table 1. Our dataset comprised the 440 protein chains, represented by C_α carbons, with chain length (n) between 80 and 256, from our training set of 1100 proteins. This training set was selected as mentioned in [23], to span a variety of different folds and families.

From these results we may determine the practical time complexity – the number of short edges, $O(n)$, times the $O(n \log n)$ lifting map computation gives the complexity for AD edges. For AD triangles and tetrahedra, performance measures are more naturally expressed in terms of s (the number of short non-Delaunay edges that pass the cutoff) which is $O(n)$.

Angle pruning is instrumental in making the practical case more efficient than the theoretical upper bound; the number of candidates per short edge is almost constant ($\sim s^{\frac{1}{6}}$) after angle pruning, which leads to $O(s^{\frac{7}{6}})$ time taken. The number of AD triangles and tetrahedra tested and output is $O(s)$, which corresponds to $O(1)$ simplices per short edge.

Thus the algorithm has output size and memory requirements linear in the number of points for typical edge length prune and ϵ_{cutoff} values. Our MATLAB implementation computes the AD tetrahedra for protein data in 3D with 100–1000 points in a few seconds to two minutes on a 2.0GHz computer.

5 Delaunay probability

In the previous sections we discussed one new method for estimating nearest neighbors in sets of imprecise points. Here we introduce another notion that is useful in describing random perturbations of points. The almost-Delaunay simplices

Quantity	Regression Fit $\pm$ Residual	
	ϵ_{cutoff}=1.0 Å	ϵ_{cutoff}=0.1 Å
Delaunay edges	$7.1n - 92 \pm 30$	
Short D edges	$5.7n - 60 \pm 60$	
Short non-D edges	$2.9n - 73 \pm 100$	
% that pass cutoff (# that pass = s)	$98 \pm 2\%$	$12 \pm 3\%$
Candidate centers, pre-angle pruning	$1.87n^{0.78} + 14 \pm 5$	
Edges between ctrs, pre-angle pruning	$2.12n^{0.83} + 25 \pm 5$	
Candidate centers, post-angle pruning	$7.1s^{0.16}+0.5 \pm 1$	$8.9s^{0.16} \pm 1.5$
Edges between ctrs, post-angle pruning	$6.7s^{0.16}+2.7 \pm 2$	$8.9s^{0.16}+0.5 \pm 2$
AD triangles tested	$8s - 260 \pm 300$	$4.4s - 5.9 \pm 20$
AD tetrahedra tested	$21s-1300 \pm 1500$	$6.2s - 18 \pm 60$
AD triangles output	$5s - 39 \pm 100$	$3.7s + 1.3 \pm 20$
AD tetrahedra output	$8.5s - 180 \pm 300$	$4.6s + 0.65 \pm 40$

Table 1: Performance measures of the AD algorithm over 440 proteins as functions of *number of points* (n) or *number of short non-Delaunay edges that pass cutoff* (s) obtained from regression, along with 99^{th}-percentile residuals. Candidate centers and edges are per short edge, while other quantities are per point set. Measurements were made for two typical values of ϵ_{cutoff}, 1.0 and 0.1 Å, and edges longer than 10 Å were pruned.

$AD(\epsilon)$ capture in a sense the worst case change in the Delaunay triangulation due to some bounded and carefully orchestrated perturbation. The actual perturbations that occur are random, and will sometimes cause a simplex to become Delaunay, and at other times will not. Here we characterize random perturbations; the analysis and algorithm are given for tetrahedra, though the concept holds in arbitrary dimensions.

If we assume a particular distribution of error in the coordinates, it is possible to calculate the probability that a particular tetrahedron is Delaunay. Assume that the probability density function (pdf) has radial symmetry so that it can be expressed in spherical coordinates as a function of the radius, $f(r)$. Such a pdf must satisfy $\int_0^\infty 4\pi x^2 f(x)dx = 1$; examples include the Gaussian distributions with expected radius a given by $f_a(x) = 8/(a\pi)^3 \exp\bigl(-(2x/a)^2/\pi\bigr)$.

A sphere C of radius r at distance d from the origin will contain total density

$$C(d,r) = \int_{d-r}^{d+r} \pi x \frac{(d+r-x)(x-d+r)}{d} f(x)dx.$$

The probability that a point is outside sphere C is $1 - C(d,r)$, so if we have a fixed circle, we can compute the probability that it is empty. For the distribution f_a, we can even give an expression in terms of the normal error function $\mathrm{erf}(x) = \int_0^x \exp(-u^2)du$:

$$C(d,r) = \frac{1}{2}\left(\mathrm{erf}\Bigl(\frac{2(d+r)}{a\sqrt{\pi}}\Bigr) - \mathrm{erf}\Bigl(\frac{2(d-r)}{a\sqrt{\pi}}\Bigr)\right) + \frac{a}{4d}\left(e^{\frac{-4(d+r)^2}{a^2\pi}} - e^{\frac{-4(d-r)^2}{a^2\pi}}\right). \quad (5.1)$$

Given positions of all sites, $s_1, s_2, \ldots, s_n$, we could obtain the probability that $\{s_1, s_2, s_3, s_4\}$ form a Delaunay tetrahedron if we could integrate, over each sphere that these sites can define, the probability of the sphere times the probability that it is empty. This integral cannot be calculated in closed form, but we can perform a Monte Carlo integration by generating sites s_1 through s_4 from the Gaussian distribution and using (5.1) to compute the probability that their circumsphere is empty.

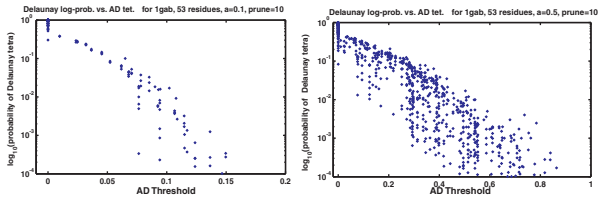

Figure 7: Tetrahedron probability vs. AD ϵ with perturbations of average radius $a = 0.1$ and 0.5.

Computation of the Delaunay probabilities suffers from the same problem as the naïve almost-Delaunay computation: there are roughly n^4 tetrahedra to check, and each one requires significant computation. Fortunately, Figure 5 illustrates that the probability of a tetrahedron is inversely correlated with AD threshold. With the Gaussian distribution f_a, the probability falls below 10^{-6} for $AD(2a)$ tetrahedra. Thus, our efficient enumeration of almost-Delaunay tetrahedra makes it feasible to estimate probabilities for tetrahedra in proteins.

Two confirmations of the accuracy of the Delaunay probability calculation: First, in a configuration whose convex hull contains the origin, the sum of the probabilities of all simplices containing the origin numerically converges to 1 as we increased the number of Monte Carlo trials. Second, in a large training set of 1100 proteins selected as detailed in [23], the sum of the Delaunay probabilities of all AD(0.3) tetrahedra converged to within 99–101% of the number of Delaunay tetrahedra at any value of edge length prune.

6 Results from analysis of protein structure

We studied the robustness of the Delaunay tessellation for analysis of protein packing interactions and finding motifs of secondary structure. We briefly report a few of our results, which are detailed in a companion paper [7].

1. An α-helix has a striking distribution of positive almost-Delaunay thresholds: three sharp peaks at $\epsilon =$

0.3,0.7 and 1.2, which arise from the regular geometric pattern. Individual C_α histograms also reveal the same peaks, which characterize residues in α-helices.

2. Proteins represented by side-chain centroids produce fewer AD tetrahedra for a given cutoff and prune than those represented by C_αs.

3. As point sets become more structured (i.e. protein-like), the number of AD tetrahedra decreases, particularly at low threshold values. This indicates the relative stability of the Delaunay for protein data.

4. Simplicial Neighborhood Analysis of Protein Packing (SNAPP) scores protein structures using the likelihood of neighboring four-tuples of residues from the Delaunay tessellation of their side-chain centroids. We evaluated the sensitivity of the SNAPP scores to a change in the Delaunay tessellation using AD tetrahedra weighted by Delaunay probability and unweighted. We found the modified scores equally effective at distinguishing proteins from decoys, which indicates the robustness of Delaunay tessellation.

5. We characterized the better inter-chain packing of the native state than decoys (artificial structures) in terms of AD tetrahedra.

6. Using the characteristic AD thresholds of the α-helix, we filtered out the tetrahedra containing residues in helical conformation, and were able to visualize and quantify the α-helical content of a protein to high accuracy. We also identified β-sheets and β-turns in a similar way.

7. We identified hinge regions in a protein undergoing conformational change by comparing AD tetrahedra across many copies of the protein chain, grouping them based on number of Delaunay tetrahedra and maximum threshold across all the chains.

7 Future work

We have introduced the tools of almost-Delaunay simplices and Delaunay probability to give worst-case and average-case analysis of the Delaunay tessellation under perturbation of the input sites. We have proved properties of the AD simplices that lead to an efficient 3D implementation, and applied them to analyze protein data. We hope to use these and other tools in the future for a clearer understanding of nearest neighbors and packing interactions in imprecise data, and investigate Delaunay probability to improve results in GIS and meshing applications of Delaunay tessellation.

Acknowledgements

We thank the biogeometry research group at UNC, Alexander Tropsha and members of his lab for discussions, David O'Brien and Bala Krishnamoorthy for code to run experiments on protein data, and Robert-Paul Berretty for work on the annulus problem.

References

[1] Brookhaven protein data bank. http://www.rcsb.org.

[2] M. Abellanas, F. Hurtado, and P. A. Ramos. Structural tolerance and Delaunay triangulation. *Information Processing Letters*, 71(5–6):221–227, 1999.

[3] P. K. Agarwal, B. Aronov, S. Har-Peled, and M. Sharir. Approximation and exact algorithms for minimum-width annuli and shells. In *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pages 380–389, 1999.

[4] B. Angelov, J. Sadoc, R. Jullien, A. Soyer, J. Mornon, and J. Chomilier. Nonatomic solvent-driven Voronoi tessellation of proteins: an open tool to analyze protein folds. *Proteins*, 49(4):446–456, 2002.

[5] F. Aurenhammer. Voronoi diagrams: A survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, 1991.

[6] D. Bakowies and W. F. van Gunsteren. Water in protein cavities: A procedure to identify internal water and exchange pathways and application to fatty acid-binding protein. *Proteins*, 47(4):534–545, 2002.

[7] D. Bandyopadhyay, A. Tropsha, and J. Snoeyink. Almost-delaunay tetrahedra for analyzing protein structure, 2003. submitted. http://www.cs.unc.edu/~debug/papers/AlmDel.

[8] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The Quickhull algorithm for convex hulls. *ACM Trans. Math. Softw.*, 22(4):469–483, 1996.

[9] J.-D. Boissonnat and M. Yvinec. *Algorithmic Geometry*. Cambridge University Press, UK, 1998.

[10] K. Q. Brown. *Geometric transforms for fast geometric algorithms*. PhD thesis, Carnegie–Mellon University, Pittsburgh, Penn., 1980.

[11] C. W. Carter, B. C. LeFebvre, S. Cammer, A. Tropsha, and M. H. Edgell. Four-body potentials reveal protein-specific correlations to stability changes caused by hydrophobic core mutations. *Journal of Molecular Biology*, 311(4):625–638, 2001.

[12] B. Chazelle and H. Edelsbrunner. An improved algorithm for constructing kth-order Voronoi diagrams. *IEEE Trans. Comput.*, C-36:1349–1354, 1987.

[13] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, Berlin, Germany, 2nd edition, 2000.

[14] B. Delaunay. Sur la sphère vide. A la memoire de Georges Voronoi. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskih i Estestvennyh Nauk*, 7:793–800, 1934.

[15] C. A. Duncan, M. T. Goodrich, and E. A. Ramos. Efficient approximation and optimization algorithms for computational metrology. In *Proc. 8th ACM-SIAM Sympos. Discrete Algorithms*, pages 121–130, 1997.

[16] R. A. Dwyer. Higher-dimensional voronoi diagrams in linear expected time. *Discrete Comput. Geom.*, 6(4):343–367, 1991.

[17] R. A. Dwyer. The expected number of k-faces of a voronoi diagram. *Computers Math Applications*, 26(5):13–19, 1993.

[18] H. Ebara, N. Fukuyama, H. Nakano, and Y. Nakanishi. Roundness algorithms using the Voronoi diagrams. In *Canadian Conference on Computational Geometry*, page 41, 1989.

[19] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graph.*, 9(1):66–104, 1990.

[20] J. Garcia-Lopez, P. Ramos, and J. Snoeyink. Fitting a set of points by a circle. *Discrete and Computational Geometry*, 20:389–402, 1998.

[21] J. Gudmundsson, M. Hammar, and M. J. van Kreveld. Higher order Delaunay triangulations. *Comput. Geom.: Theory and Appl.*, 23(1):85–98, 2002.

[22] L. J. Guibas, D. Salesin, and J. Stolfi. Epsilon geometry: building robust algorithms from imprecise computations. In *Proc. 5th ACM Symp. Comp. Geom.*, pages 208–217, 1989.

[23] B. Krishnamoorthy and A. Tropsha. Development of a four-body statistical pseudo-potential to discriminate native from non-native protein conformations. *Bioinformatics*, 19(12), 2003.

[24] A. R. Leach. *Molecular Modelling: Principles and Applications, second edition*. Pearson Education EMA, UK, 2001.

[25] A. M. Lesk. *Introduction to Protein Architecture - The Structural Biology of Proteins*. Oxford University Press, UK, 2000.

[26] J. Liang and K. A. Dill. Are proteins well-packed? *Biophys J.*, 81(2):751–766, 2001.

[27] J. Liang, H. Edelsbrunner, P. Fu, P. Sudhakar, and S. Subramaniam. Analytical shape computing of macromolecules II: identification and computation of inaccessible cavities inside proteins. *Proteins*, 33:18–29, 1998.

[28] J. Liang, H. Edelsbrunner, and C. Woodward. Anatomy of protein pockets and cavities: Measurement of binding site geometry and implications for ligand design. *Protein Science*, 7:1884–1897, 1998.

[29] Y. C. Liao, D. J. Lee, and B.-H. Chen. Description of multi-particle systems using Voronoi polyhedra. *Powder Technology*, 119(2–3):81–88, 2001.

[30] B. McConkey, V. Sobolev, and M. Edelman. Quantification of protein surfaces, volumes and atom-atom contacts using a constrained Voronoi procedure. *Bioinformatics*, 18(10):1365–1373, 2002.

[31] K. Mehlhorn, T. C. Shermer, and C.-K. Yap. A complete roundness classification procedure. In *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pages 129–138, 1997.

[32] S. Miyazawa and R. L. Jernigan. Residue-residue potentials with a favorable contact pair term and an unfavorable high packing density term, for simulation and threading. *Journal of Molecular Biology*, 256:623–644, 1996.

[33] P. J. Munson and R. K. Singh. Statistical significance of hierarchical multi-body potentials based on Delaunay tessellation and their application in sequence-structure alignment. *Protein Sci*, 6(7):1467–1481, 1997.

[34] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, Chichester, UK, 1992.

[35] P. A. Ramos. Computing roundness is easy if the set is almost round. In *Proc. 15th Annu. ACM Sympos. Comput. Geom.*, pages 307–315, 1999.

[36] F. M. Richards. The interpretation of protein structures: total volume, group volume distributions, and packing density. *J. Molecular Biology*, 82:1–14, 1974.

[37] T. Rivlin. Approximation by circles. *Computing*, 21:93–104, 1979.

[38] U. Roy and X. Zhang. Establishment of a pair of concentric circles with the minimum radial separation for assessing roundness errors. *Computer Aided Design*, 24(3):161–168, 1992.

[39] K. T. Simons, C. Kooperberg, E. Huang, and D. Baker. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. *Journal of Molecular Biology*, 268:209–225, 1997.

[40] R. Singh, A. Tropsha, and I. Vaisman. Delaunay tessellation of proteins. *J. Comput. Biol.*, 3:213–222, 1996.

[41] M. Smid and R. Janardan. On the width and roundness of a set of points in the plane. In *Canadian Conference on Computational Geometry*, pages 193–198, Aug. 1995.

[42] J. Tsai and M. Gerstein. Calculations of protein volumes: sensitivity analysis and parameter database. *Bioinformatics*, 18:985–995, 2002.

[43] J. Tsai, R. Taylor, C. Chothia, and M. Gerstein. The packing density in proteins: Standard radii and volumes. *Journal of Molecular Biology*, 290(1):253–266, 1999.

[44] G. M. Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième Mémoire: Recherches sur les parallélloèdres primitifs. *J. Reine Angew. Math.*, 134:198–287, 1908.

[45] H. Wako and T. Yamato. Novel method to detect a motif of local structures in different protein conformations. *Protein Engineering*, 11:981–990, 1998.

[46] G. Weberndorfer, I. L. Hofacker, and P. F. Stadler. An efficient potential for protein sequence design. In *Computer Science in Biology. Univ. Bielefeld, D. GCB'99 Proceedings*, pages 107–112, Hannover, Germany, 1999. http://www.tbi.univie.ac.at/papers/Abstracts/GCB026.ps.gz.

[47] L. Wernisch, M. Hunting, and S. Wodak. Identification of structural domains in proteins by a graph heuristic. *Proteins*, 35(3):338–352, 1999.

[48] R. Zimmer, M. Wöhler, and R. Thiele. New scoring schemes for protein fold recognition based on Voronoi contacts. *Bioinformatics*, 14(3):295–308, 1998.