SERIE B — INFORMATIK

Covering Shapes by Ellipses for the Computer Analysis of Protein Patterns

Alon Efrat**
Frank Hoffmann*
Klaus Kriegel*
Christof Schultz*

B 00-11
July 2000

Abstract

We study the problem of how to cover a polygonal region by a small number of axis–parallel ellipses. This question is well motivated by a special pattern recognition task where one has to identify ellipse shaped protein spots in 2–dimensional electrophoresis images. We present and discuss two algorithmic approaches solving this problem: a greedy brute force method and a linear programming formulation. Furthermore we discuss related theoretical questions.

*Institut für Informatik, Freie Universität Berlin, Takustr. 9, D-14195 Berlin
E-mail: *name*@inf.fu-berlin.de
`http://www.inf.fu-berlin.de/inst/theo/index.html`
**Computer Science Department, Stanford University
E-mail: *alon@Graphics.Stanford.EDU*

1 Detecting Spots in 2–dimensional Gel Electrophoresis Images

1.1 Gel Electrophoresis: The Application Background

Proteomics is a rapidly growing field within computational molecular biology. In proteomics 2–dimensional gel electrophoresis (2DE) is the best known and widely used technique to separate proteins. A 2DE gel is the product of two separations performed sequentially in acrylamide gel media: isoelectric focusing as the first dimension and a separation by molecular size as the second dimension. A two-dimensional pattern of spots each representing a protein is the result of that process. Eventually, spots are made visible by staining or radiographic methods. By analyzing series of such 2DE images one hopes to identify those proteins that change their expression (size, intensity) and reflect/cause certain biochemical and biomedical conditions of an organism, see [17]. Ideally, in an gel

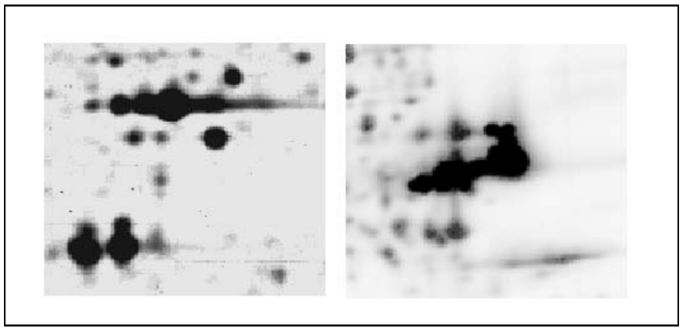
Figure 1: Twin spots, streaks and complex region

image each spot has the shape of an axis–parallel ellipse, which is a widely accepted modeling assumption, see e.g. [3] or [12]. However, spots that are very close to each other can partially merge (modeled by overlapping ellipses) and form rather complicated regions as depicted in Figure 1.

At Freie Universität Berlin we have started a few years ago to develop the software system CAROL (see [6]) that answers local and global matching queries for gel images (given in GIF format). The novelty of its matching tool was that setting landmarks by hand, a necessary preprocessing step in previous algorithms, could be avoided. Instead, the matching between a source and a target image uses the history of the incremental Delaunay triangulation, [13], [11], of the target spots. The matching tool starts from the assumption that images are already given as spot lists with each spot represented by point coordinates of its center and a real value describing its intensity. However, since the accessible spot detection algorithms did not supply results precise enough for our approach we developed and included a new detection algorithm into the CAROL system, see [15] for details.

The most difficult task for the algorithm is how to interpret twin spots, streaks (left side in Fig. 1) and so called *complex regions* (right side in Fig. 1) as unions of ellipses. In fact, in [15] the latter case was left open and in the implementation the user had to edit these complex regions manually. To present an algorithmic solution to this question is the subject of this paper and to our knowledge it is the first algorithmic approach that deals

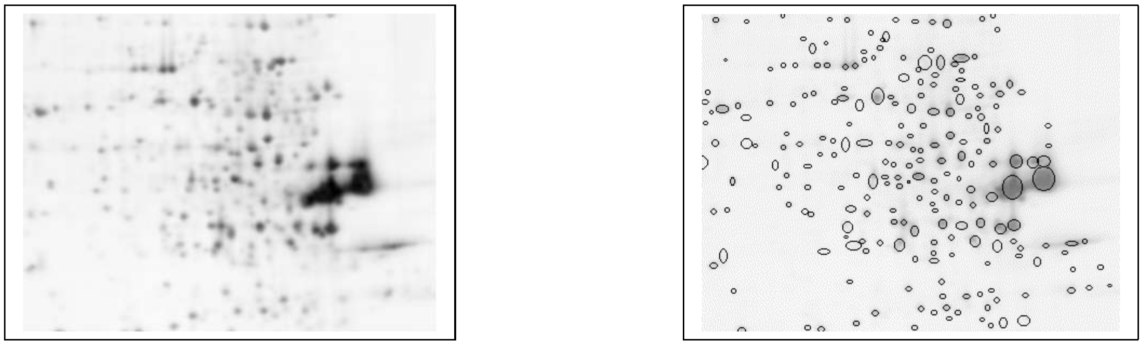

Figure 2: Part of a gel image and spots computed

with complex regions in 2DE images.
In fact, just by looking at the images it is evident that there is an inherent uncertainty in the 2DE gel images. This is due to the electrophoresis process itself which is highly susceptible to faults and geometric distortions, so there is no hope to come up with a perfect spot detection algorithm. Consequently, in the upcoming CAROL version we try to cope with that problem by maintaining a list of proposals how such an ambiguous region could be covered instead of computing only the 'best' covering which could be erroneous. In the matching algorithm part we then have the possibility to accept the matching of two ambiguous regions if there is a pair of proposed coverings that match.

1.2 Modeling the Covering Problem

The following formalization is a compromise stemming from discussions with practitioners who solve these covering instances by peer review. We assume that a connected pixel pattern R is given, which is fat in the sense that there are no short horizontal/vertical cuts consisting of two pixels only. In the application R is usually a 1–connected subpattern of a 100×100–pixel square. We identify a pixel with its center and denote by ∂R the rectilinear polygonal curve traversing the boundary pixels. Then we shrink and expand this boundary to the inside and the outside as follows by defining the two pixel sets $\partial R_- = \{p \in R | d_\infty(p, \partial R) = 1\}$ and, analogously, $\partial R_+ = \{p \notin R | d_\infty(p, \partial R) = 1\}$.

Now we can formulate the approximative covering problem we are interested in from the application point of view. For numbers $0 \leq p, q \leq 1$ and a natural number $\rho \geq 1$ find a smallest possible set $\mathcal{E}$ of axis–parallel ellipses fulfilling the following conditions.

1. **(Shape)** For each $E \in \mathcal{E}$ the ratio of its halfaxes is in the interval $[1/\rho, \rho]$.
2. **(Fitting)** Each ellipse $E \in \mathcal{E}$ *respects* ∂R_+, i.e., it does not intersect ∂R_+.
3. **(Intersection)** The boundaries of any pair of ellipses $E, E' \in \mathcal{E}$ intersects in at most 2 points, and $\cdot\text{area}(E \cap E') \leq q \cdot \min\{\text{area}(E), \text{area}(E')\}$.
4. **(Covering)** $\bigcup_{E \in \mathcal{E}} E$ covers at least a $1 - p$ portion of pixels in ∂R_-.

The aim to find minimal coverings is in accordance with Occam's razor principle, since the smallest cardinality set is, in a sense, the simplest hypothesis to explain how a complex region could have been evolved. Moreover, we especially emphasize that the somehow strange intersection condition 3 is justified by the application because two spots (ellipses) can only partially merge and do not form a cross.

2 Brute Force Solution vs. Linear Programming

In this section we present two algorithmic solutions to our covering problem that have been implemented and tested. In the implementation we assumed that the regions R are simply connected and rectilinear, however it is straightforward how to extend the algorithms to arbitrary polygonal regions. Aiming at better running times the sets ∂R_- and ∂R_+ are sampled by subsets S_- and S_+. Since a convex boundary segment is likely to be part of a single ellipse, S_- and S_+ are chosen in such a way that convex segments have denser samples.

2.1 Computing a Brute Force Solution

The basic idea behind the naive brute force solution is to generate all ellipses that fulfill condition 1 and 2. A voting scheme is then set up to select the covering.
In a first step we discretize the parameter space of possible ellipses. Recall that an axis–parallel ellipse is formed by all points (x, y) fulfilling the equation

$$\frac{(x-c)^2}{a^2} + \frac{(y-d)^2}{b^2} - 1 = 0 \qquad (1)$$

with parameters $a, b, c, d \in R$.

To this end we restrict the ellipses to have centers which are pixel centers and one of the halfaxes, say a, has to have multiple pixel side length. Finally, to model condition 1, for a fixed a we restrict the second halfaxis to values from the set $\{as^i | -k \leq i \leq k\}$, with $s = \sqrt[k]{\rho}$
Now, the algorithm simply computes for each center (c, d) and for each i the maximal halfaxis a such that the ellipse with parameters (a, as^i, c, d) does not intersect ∂R_+.
For each ellipse we store which points from S_- it covers and their number. Eventually, in a greedy fashion we choose the covering. Assume a partial covering is already chosen. The next ellipse E is selected among all ellipses satisfying condition 3 (with respect to already chosen ellipses) according to the following criteria, ordered as ranked:

1. E covers a maximal number of previously uncovered points from S_-
2. E maximizes the length of longest chain of consecutive covered points from S_-
3. E minimizes maximal intersection area with an already chosen ellipse.

After selecting a best E (ties are broken randomly) we update the scores of all other remaining ellipses by deleting the points covered by E. We stop augmenting ellipses when condition 4 is met.
The drawback of the brute force approach is obvious, too many ellipses are tested. Moreover, by discretizing ellipse parameters we may miss interesting ellipses like the big one in the right hand solution in Figure 3.
Remark: Let $\mathcal{F}$ be the family of ellipses (a, as^i, c, d) as described above, and let k_{opt} be the minimal number of ellipses of $\mathcal{F}$, needed to cover all points of S_- and avoiding the

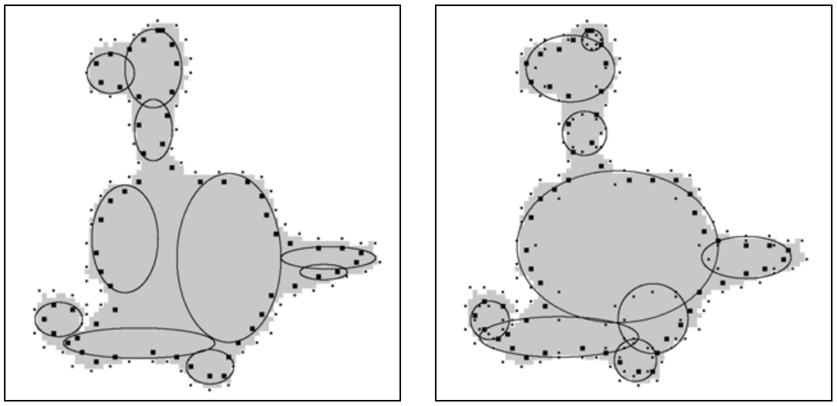

Figure 3: Ellipse covering computed by brute force method (left) and by LP approach (right)

ones of ∂R_+. Then the standard $\log|S_-|$–approximation (see [7]) of the optimal solution by the greedy approach cannot be guaranteed, at least for arbitrary sample sets S_- in polygons with holes. This is due to the restrictive intersection property. An example that illustrates this observation consists of a set of rather thin ellipses arranged in a grid like fashion. Besides the very recent paper [2] we are not aware of approximation results for set covers with restrictive intersection properties.

2.2 Using an LP Approach to Generate Ellipses

Compared to the brute force approach we do not want to restrict the set of ellipses under consideration for covering a region by an apriori parameter discretization and we want to avoid to generate to many explicite ellipses.
Observe that each element in the pixel set ∂R_+, now sampled by S_+, forms a constraint for each ellipse in the cover, since any ellipse must not contain such pixels. On the other hand, we want at least a few points (say, at least three) from S_- to be included in a ellipse of the covering.
Therefore, we start from a randomly chosen triplet of mutually visible points from S_- and ask whether there is an axis–parallel ellipse containing these points such that it does not violate an outer constraint. Once having the information that for a given triplet there is a feasible solution one can efficiently extend the covered sample subset by an LP–based approach.
We start from the observation that the parameters of an ellipse E can be transformed into variables of an LP in such a way that each of the three points which have to be covered by E adds a linear constraint to the outer constraints. The constraints are derived by putting a point $p = (p_x, p_y)$ into the ellipse formula $\frac{(p_x-c)^2}{a^2} + \frac{(p_y-d)^2}{b^2} - 1$ which has to be zero (negative, positive) for points on (inside, outside) E. Since this is not a linear constraint we start with one of the form

$$p_x x_1 - p_y^2 x_2 + p_y x_3 - x_4 = p_x^2 \qquad \text{(resp. } <, > \text{)} \qquad (2)$$

By elementary calculation, setting $t = \frac{x_1^2}{4} + \frac{x_3^2}{4x_2} - x_4$, we derive as ellipse parameters

$$a = \sqrt{t} \qquad b = \sqrt{\frac{t}{x_2}}$$

$$c = \frac{x_1}{2} \qquad d = \frac{x_3}{2x_2}$$

Once having the existence of a feasible solution (not an explicite ellipse yet!) one wants to extend the point set that can be covered. Clearly, it makes sense to add and check first the neighbors of the already covered sample points. This is also done in a random way. However, it is clear that one can easily get stuck, when there is a very restrictive partial solution, in the sense that we have still a feasible system but the actual solution set of ellipses is very small and does not allow to add a new sample point. To implement the necessary backtracking step, we have adapted a simplified version of the so called Metropolis methodology, see for example [14], where it has been used for the generation of large cliques in graphs, a similar situation to ours.
This random Markov-chain like process is organized in rounds. With probability q we have a round 'extend', that is we choose and try to add a random neighbor; with probability $1-q$ we have a round 'backtrack' and we delete a randomly chosen point from the already covered point set. After a fixed number of rounds we actually compute an explicite candidate ellipse to be included next into the partial covering. Among all possible ellipses we choose that one yielding a halfaxes ratio closest to 1. This is repeated a constant number of times and we augment the candidate that is optimal according to the criteria above.
The only thing that remains to explain is how to incorporate condition 3 into the LP–approach. As before we delete already covered points from S_-. But now for each selected ellipse E we add new 'outer' constraints to S_+ by sampling the ellipse interior. These new points prevent later chosen ellipses from having large intersections with E. Again, this scheme is run until condition 4 is met. In the example depicted in Figure 3 the black pixels represent S_-. The dots outside the region describe the initial outer constraints, within the chosen ellipses the additional outer constraints are also indicated. Together they form the final set S_+.

2.3 Some Implementation Details and Discussion

We implemented the Brute Force greedy approach with ellipses that have a halfaxes ratio from the set $\{s^i | -10 \leq i \leq 10, s = \sqrt[10]{5}\}$, cover the inner sample by 80% and do not pairwise intersect in more than 30% of the area. For the region in Figure 3 (subset of a 76× 80 array,with 59 inner sample points drawn in black) this yields 13000 ellipses that fit into it, to compute the 10 ellipse covering indicated took 24 seconds on a SUN Ultra Sparc 300MHz.

In the LP based solution we used the CPLEX software ([8]). Again, for Fig. 2 (initially 76 points defining outer constraints) it took 13s to compute the indicated (random) solution. For each ellipse to be included into the covering we computed 10 candidate ellipses, and for each candidate we run 30 rounds of the Metropolis process, with a $4:1$ expected ratio of extend to backtrack rounds.

Both the running times and the solutions computed on all the test regions we studied clearly indicate that the LP based solutions outdo the brute force solutions. To refine the latter by denser sampling the ellipses quickly ended in running time and memory problems.

A clear advantage of the LP solution is its randomness which allows to compute several alternative coverings that are then input to the matching procedures.
The evaluation of the programs on original gel data is rather hard because even experts have problems to agree on optimal solutions. Thus, in addition to tests on original gel data, the LP based program has been validated by a series of runs on randomly generated input samples. Each sample consists of ten randomly generated ellipses which overlap in such a way that the requirements of our modeling are fulfilled. This way there was always a unique optimal solution the algorithm's result could be compared with. The regions were approximated by rectilinear polygons of average size 140×120 what lead to sets of about 150 inner and 175 outer sample points. The average running time was 13 seconds. For each run the following three ellipse types were counted:

1. Correctly detected ellipses, i.e., ellipses from the sample such that all inner points defined by their boundary are covered by a single ellipse of the solution;

2. Partly detected ellipses, i.e., ellipses from the sample such that at least 60% (but not all) of their inner points are covered by a single ellipse in the solution;

3. Missing ellipses, i.e., ellipses from the sample that are neither correctly nor partly detected.

Missing ellipses and the uncovered part of partly detected ellipses are usually covered by small ellipses in the solution, so called dummy ellipses. Since the algorithm is randomized we evaluated ten runs for each input sample. The following table shows the average numbers for the worst and the best sample and the total average values over all test samples.

	correctly detected	partly detected	missing ellipses	dummy ellipses
worst case	9.2	0.6	0.2	1.1
best case	9.8	0.2	0.0	0.0
average case	9.5	0.4	0.1	0.9

The computation of the complete spot detection for a full gel image (1200×1500 pixels, 4500 spots) takes about half a minute without complex regions, in such large images we observed between 0 and 20 complex regions. Finally we remark that usually a spot detection is only computed once before storing the image in a data base.

3 Further Theoretical Aspects

We discuss the ellipses coverage problem from a more theoretical point of view. Assume $P = \{p_1, \ldots, p_m\}$ is an input polygon, now neither necessarily simple nor rectilinear, and let $\varepsilon > 0$ be a given fault parameter. Again we assume that the polygon has no short cuts of length $\leq 2\varepsilon$. Moreover, we drop the condition that two ellipses in the covering can intersect in at most 2 points. Let S_- (resp. S_+) denote the vertices of polygonal chains P_-

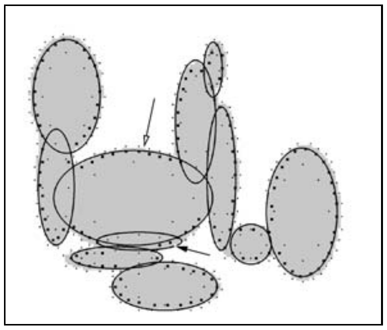

Figure 4: A randomly generated input sample. The solid (transparent) arrow marks a dummy (partly detected) ellipse.

(resp. P_+) which approximate the boundaries of the ε-neighborhood of P, have Hausdorff distance ε from P, and fulfill the property that the distance between any two consecutive vertices is at most ε.
The *ellipses-covering problem* is to find a collection $\mathcal{E} = \{E_1, \ldots, E_k\}$ of minimal cardinality of axis-aligned ellipses such that the union $U = \bigcup E_i$ contains all the points of S_- and none of the points of S_+. Let n denote the number of points in S_- and S_+ together.

In this context, ε is of course a bound on the error of our estimation of the boundary of the original collection of spots.

Note that we do not require that each point of ∂U has a point of P within distance $\leq \varepsilon$. Thus, we are willing to accept holes in the solution. We will discuss later the problem of finding a set of ellipses that avoids S_+ and approximately covers the whole interior of P.

The problem of (approximate) covering a shape with ellipses is strictly related to the problem of exact covering a shape with rectangles, which was shown to be an NP-hard. It also related to the problem of covering a shape with strips [1], and to the problem range covering in a hypergraph [5]. Thus, in the general setting there is no much hope for finding a polynomial-time algorithm. On the other hand, there is no much difficulty to obtain a polynomial time algorithm that approximates the set up to a $\log k_{opt}$ factor, where k_{opt} is the cardinality of the optimal-size solution, using the algorithm of [5]. The running time of a straightforward implementation of their algorithm is $O(n^5 k_{opt} \log n)$.

3.1 Efficient algorithms for the greedy approach

In the "traditional" greedy approach we perform a sequence of iterations, in each we find an ellipse that respects S_+ (that is, contains none of its points), and covers the maximal number of points of S_- not covered by previously chosen ellipses. We show how to perform each such iteration in time $O(n^3\alpha(n)\log n)$, where $n = |S_-| + |S_+|$. This, using standard greedy arguments, yields a running time of $O(n^3 k_{opt}\alpha(n)\log^2 n)$ where k_{opt} is the cardinality of an optimal output, and $\alpha(n)$ is the inverse Ackermann function.
Note that for each ellipse with parameters a, b, c, d equation (2) yields a dual representation by a point $(x_1, x_2, x_3, x_4) \in R^4$ where $x_1 = 2c$, $x_2 = a^2/b^2$, $x_3 = 2a^2d/b^2$ and $x_4 = c^2 - a^2 + a^2d^2/b^2$. Consequently, the set of ellipses containing (not containing) a given point (p_x, p_y) is described by the hyperplane $p_x x_1 - p_y^2 x_2 + p_y x_3 - x_4 > p_x^2$ (resp. $<$).

Let t_1, t_2 be two points of S_-, and consider the two-dimensional plane $\mathcal{D}$ in R^4 representing all axis-parallel ellipses that pass through t_1, t_2. Then $\mathcal{D}$ is the intersection of the two hyper-planes that corresponds to t_1, t_2. Each point p of $S_- \cup S_+$ determines a line $\tilde{p}$ in $\mathcal{D}$, and an ellipse E contains p if and only if $\tilde{p}$ is below the point $\tilde{E}$ which is the dual point in $\mathcal{D}$ of E. Accordingly, E respects all points of S_+ if and only if $\tilde{E}$ is below all lines corresponding to point of S_+. Let $\tilde{S}_-$ (resp. $\tilde{S}_+$) denote the collections of lines corresponding to points of S_- (resp. S_+). Let Γ denote the lower envelope of the lines of $\tilde{S}_+$. We seek a point $\tilde{E}$ which is below Γ, and above a maximum number of the lines of $\tilde{S}_-$. Consider the arrangement $\mathcal{A}_-$ formed by the lines of $\tilde{S}_-$. Clearly $\tilde{E}$ lies on an edge of one of the cells of $\mathcal{A}_-$ intersected by Γ. The collections of these edges (and the vertices they define) is known in the Computational Geometry literature as the *zone* of Γ in $\mathcal{A}_-$. See [4] for further reference.

Using the recent algorithm of Har-Peled [10], the zone of Γ in $\mathcal{A}_-$ can be constructed in time $O(n\alpha(n)\log n)$. Once the zone is created, finding a point in this zone that lies above the maximal number of lines of $\mathcal{A}_-$ is therefore doable by scanning this zone (in linear time). Since t_1 and t_2 are not known in advance, we need to iterate over all pairs of points of S_-, and repeat this process for every new ellipse that we find. Using standard greedy-algorithms arguments, this number of ellipses we find is larger by at most $O(\log n)$ than the optimum number, thus the running time of $O(n^3 k_{opt}\alpha(n)\log^2 n)$ where k_{opt} is the cardinality of an optimal output, as asserted.

Remark: We can use a similar algorithm also for finding a collection of ellipses which respect S_+, and approximately cover the interior of P. The necessary simple modification is as follows: We replace S_- by the point set D_- which are the vertices of a uniform grid of edge length $\varepsilon/2$ which are inside P (i.e., all points of the form $(i\varepsilon/2, j\varepsilon/2)$ for i, j integers). Additionally, S_+ is replaced by a point set S'_+ lying in between P and S_+. Using either the greedy approach or the algorithm of [5], we find a small collection of ellipses $\mathcal{E}' = \{E'_1 \dots E'_k\}$ whose union avoids S'_+ and covers D_-. Next we replace each ellipse E'_i of $\mathcal{E}'$ by an ellipse E_i slightly larger. Formally, E_i and E'_i share the same centers, but each of the axes of E_i are larger by $\sqrt{2}\varepsilon/4$ than the corresponding axes of E'_i. Clearly, the collection $\mathcal{E} = \{E_i, \dots, E_k\}$ avoids the points of S_+, given S'_+ was chosen accordingly. Since each interior point is at distance at most $\sqrt{2}\varepsilon/4$ from a point of D_- the polygon is completely covered. One can also show that the cardinality of $\mathcal{E}$ is within $O(\log n)$ factor of the optimum. Note that the number of grid-points in D_- is at most $O(n^2)$.

3.2 Polygons with long edges.

If the input polygon P consists of long edges (i.e. the assumption that the length of each edge is $O(\varepsilon)$ is not valid), we can use one of several variants of this algorithm. The decision which variant to use depends on the assumption of the model we use. More precisely, whether an edge of P needs to be covered by a single ellipse, or can it be covered by an arbitrary number of ellipses. In the later case, we currently see no better alternative than artificially replace this edge by a sequence of short edges, each of length $O(\varepsilon)$, and treat this case as this was the original input. We strongly suspect that one can do better by implicitly creating these edges.

However, if we assume that each edge of P is covered by a unique ellipse, (but a single ellipse can cover several edges) a better alternative exists. For each boundary edge e_i of P we create the edges e_i^+ (e_i^-) of the circumscribed (inscribed) polygon with distance ε from

P. Instead of defining S_- to be an ε-dense sample it is sufficient to choose the vertices of the inscribed polygon.

Fixing two points of S_-, and using the same duality transform as above, we have to consider a two-dimensional plane $\mathcal{D}$ in R^4. We need to cope with a few modification of the algorithms.

First, we seek for ellipses in $\mathcal{D}$ respecting all edges e_i^+. The dual $\tilde{e}_i^+$ of this edge is a quadratic curve. The lower envelope Γ now consists of pieces of such curves. Its complexity is still $O(n\alpha(n))$. The second modification applies to the points in the zone of Γ we are looking for. We require that they are located above a maximal number of line pairs corresponding to edges e_i^- of the inscribed polygon. It is not hard to show that the running time remains $O(n^3 k_{opt}\alpha(n)\log^2 n)$ in this case as well.

3.3 Partitioning into Essential Parts

Next we sketch how to amplify algorithmically the observation that an almost convex boundary segment of a region is typically covered by one ellipse. That is, we assume that the region to be covered is indeed the result of merging a small number of axis-parallel ellipses. Recall that this idea was already used in the design of the LP based solution.
We assume therefore subsequently that an input polygon P, with edge length ε, is well–behaved in the following way:

1. We assume that there exists a set of ellipses $\mathcal{E}' = \{E'_1, \ldots, E'_k\}$ whose union boundary is within Hausdorff distance $\leq \varepsilon$ from P, and
2. Each vertex v of $\partial \bigcup E'_i$ is "significant" in the following sense: Let D be a disk of radius $K\varepsilon$ (for an appropriate constant K) centered at v, and let E'_1 and E'_2 be the ellipses of $\mathcal{E}'$ determining v. Then for every pair of points $p_1 \in \partial E'_1$ and $p_2 \in \partial E'_2$ inside D, the orientation of the tangent to E'_1 at p_1 differs by at least a constant κ from the orientation of the tangent to E'_2 at p_2.

Given the polygon P obeying this condition (for an unknown set $\mathcal{E}'$), we propose the following algorithm.

Approximating tangents. For every vertex $p \in P$ we compute p^r, the *approximate tangent to the right*, defined as the line that passes through p and the orientation of which is the average of all lines passing through p and one of the vertices of P *clockwise* from p and within distance $\leq K\varepsilon$. The approximated tangent to the left p^l is defined analogously, when points counterclockwise to p are considered.
Approximated vertices. We compute the set V of all vertices p of P for which the orientation of p^l is different by at least $\kappa/2$ from the orientation of p^r. Note that by property (2) above, each vertex of $\bigcup \mathcal{E}'$ yields a vertex of V. However, V might contain other points as well.
Partition into parts. We partition ∂P into *parts*, where each part is the portion of P between two consecutive approximated vertices of V. A part is called *essential* if the total angular span (difference of approximated tangents in the approximated vertices) is at least some constant α. All other parts are called *non-significant.*
Covering significant parts. We represent again each possible ellipse as a point in the 4D parameter space. Let γ be a significant part, and let γ^* denote the region in the dual

space of all ellipses that respect P and approximate γ. As one easily observes, γ^* is "fat" in the sense that for each axis in the parameter space, the diameter of γ^* along this axis is roughly ε. We replace γ^* by a box $\hat{\gamma}$ of size ε, which is easier to handle, yet each point $q^* \in \hat{\gamma}$ is dual to an ellipse that is within distance $c\varepsilon$ from γ, for a constant c.
Now, the main observation is that a set of ellipses that covers all the significant parts $\gamma_1, \ldots, \gamma_l$ corresponds to a stabbing set S for the regions $\hat{\gamma}_1, \ldots, \hat{\gamma}_l$ in the parameter space. Using very similar ideas to the ones in [9], one can find such a set S in time $O(l\text{polylog}l)$ where l is the number of significant parts, and the size of S is within a constant factor off the optimum.

Eventually, we are left with the task of covering the non-significant parts. Here we use the standard greedy approach which is doable in polynomial time and gives a solution with a logarithmic factor far from the optimum. Thus the size of the output is $O(l_1^* + l_2^* \log l_2^*)$ where l_1^* (resp. l_2^*) is the size of a minimal covering of the significant (resp. non-significant) parts.

3.4 Using the Brönnimann & Goodrich paradigm

The purpose of this subsection is to show an efficient implementation of the Brönnimann & Goodrich paradigm for finding a collection $\mathcal{E}$ of ellipses that approximately covers that interior of P. For technical reasons, we use the following strategy: Let D_- be the grid defined in Remark 3.1. Following the discussion of Remark 3.1, it is enough to find a collection $\mathcal{E}$ of ellipses that avoids S_+, whose union covers D_-, and whose centers lie on a grid point of D_-. Let k_{opt} be the minimal cardinality of such a collection.

Theorem 1 *Let P and S_+ be as above. Let $n = |S_+|$. Then in expected time $O(n^3\alpha(n) + n^2 k_{opt} \log^2 n)$ one can find a set $\mathcal{E}$ of ellipses that cover the interior of P, and its boundary $\partial \cup \mathcal{E}$ is of distance $\leq 2\varepsilon$ from the boundary of P. Moreover, the cardinality of $\mathcal{E}$ is of size $O(k_{opt} \log k_{opt})$.*

Proof: We use for this the paradigm of Brönnimann & Goodrich, which consists in this case of the following stages. It assumes that k_{opt} is known (otherwise it is found using unbounded search), and is much less than n. It maintains a set $\mathbf{E}$ of ellipses whose union covers D_-. The solution we seek is a subset of $\mathbf{E}$. It gives a weight $w(E)$ to each ellipse of $\mathbf{E}$, initially all set to be 1. The algorithm consist of *phases.* At each phase, the algorithm picks a random sample R of size $ck_{\text{opt}} \log k_{\text{opt}}$, of the ellipses of $\mathbf{E}$, where the probability of an ellipse to be picked is proportional to its weight, and c is a constant as computed in [5]. Next the algorithm checks if R covers D_-. Assuming this is not the case. Let p be a grid point which is not covered, (arbitrarily chosen if there are more then one) and let $\mathbf{E}_p \subseteq \mathbf{E}$ be the set of ellipses containing p. The algorithm doubles the weight of each ellipse of $\mathbf{E}_p$, provided that $\sum_{E \in \mathbf{E}_p} w(E) \leq w(\mathbf{E})/2k_{\text{opt}}$. The algorithm stops when a cover is found. Since the Vapnik-Chervonenk is dimension of the problem is clearly finite, it is shown in [5] that the algorithm stops after $O(k_{\text{opt}} \log n)$ phases.
Preliminaries As the first step toward an efficient implementation of the algorithm, we restrict our search to *maximal ellipses.* An ellipse E is *y-maximal* if (1) its center is in a grid point of D_-, it (2) respects S_+, and in addition (3), among all ellipses that has the same center and x-axis, E has the largest y-axis. We define an x-maximal ellipse analogously. An ellipse is *maximal* if it is either x-maximal or y-maximal. Clearly, we can assume that the optimal cover consists of maximal ellipses, and thus assume that $\mathbf{E}$

consists only of maximal ellipses. We claim the number of such ellipses is only $O(n^3)$. This results from that fact that there are $O(n^2)$ points x in D_-, and each x gives raise to $O(n)$ maximal ellipses centered at x.

In a somehow similar fashion to [1], we construct a data structure that enables us to (implicitly) modify the weight of all ellipses in $\mathbf{E}_p$. We describe the process for y-maximal ellipses. Analogous process is applied to the x-maximal ellipses. Let p be a fixed point of D_-, and let r be a row of a grid-points of D_-. Let $\Psi(w)$ denote the family of all axis-parallel ellipses (not necessarily maximal) whose width is w. Let $e(x)$ (for $x \in r$) be the y-maximal ellipse in $\Psi(w)$ whose center is in x. We paint a grid-point $x \in r$ *white* if $p \in e(x)$. Otherwise x is *black*.

Lemma 1 *There is at most one connected sub-interval consisting of white points.*

Proof Let $q \in S_+$ be a point.
Claim There is a at most a single $x = x_q$ for which $e(x_q)$ contains both p and q on its boundary.

The proof of the claim is based on the fact that if $x_1, x_2 \in r$, then the boundaries of $e(x_1)$ and of $e(x_2)$ intersect in at most a single point. Details are omitted from this extended abstract.

Now assume by contradiction that there are at least two white subintervals of r, and let x_1 be the right endpoint of the leftmost sub-intervals. Let $f_q(x)$ (for $x \in r, q \in S_+$) be the unique ellipse whose center is in x, whose width is w, and whose boundary passes through q. Clearly $\partial e(x_1)$ contains both p and some point $q \in S_+$ on its boundary, that is $e(x_1) = f_q(x_1)$. For any point $x_2 \in r$ which is very close and to the right of x_1, we have that $p \notin f_q(x_2)$, and thus for any $x_3 \in r, x_3 > x_1$, we have that $p \notin f_q(x_3)$, (otherwise there must have been $x' \in r, x' \neq x_1$ for which $p \in \partial f_q(x')$, which contradicts the claim). Since $f_q(x)$ (if exists) contains $e(x)$ for every x, we deduce that $p \notin e(x)$ for every $x > x_1$. This concludes the proof of Lemma 1.

Let us denote by $I_{w,r}(p)$ the subinterval on r of all points x for which a maximal ellipse of width w centered at x contains p.
Computing the heights of the the maximal ellipses. Let r be a fixed row and w a fixed width, as before. We compute for each $q \in S_+$ the function $\phi_q(\cdot)$, which is defined for each point $x \in c$, and denote the height of the (unique) axis-parallel ellipse in $\Psi(w)$ whose center is in x, and its boundary contains q. Clearly the height of $e(x)$, the maximal ellipse in $\Psi(w)$, equals $\min_{q \in S_+} \phi_q(x)$. Thus, in order to compute the height of these ellipses for each $x \in r$, we merely need to compute the lower envelope of the functions $\{\phi_q(x) | q \in S_+\}$. The complexity of this envelope is known to be $\lambda_s(n)$ (for an appropriate constant s) and can be efficiently computed in time $\lambda_s(s) \log n$, see [16]. Multiplying this number by the number $O(n)$ of possible values of w, and by the number $O(n)$ of points in S_+, we obtained a running time of $O(n^3 \alpha(n) + n^2 k_{opt} \log k_{opt})$ for the algorithm.
Finding an uncovered point. After picking a random sample R as described above, we need to see if D_- is contained in $\cup R$. For this, we compute $\cup R$ explicitly, (using for example a line sweep technique). The complexity of this union is $O({k_{\text{opt}}}^2 \log^2 k_{\text{opt}}) = O(n^2)$, as the size of R is $O(k_{\text{opt}} \log k_{\text{opt}})$, and it can be computed in time $O({k_{\text{opt}}}^2 \log^3 k_{\text{opt}})$. This bound is bounded by $O(n^2)$, according to our assumption. Next we scan the plan to find the first point point $p \in D_- \setminus \cup R$.

Computing $I_{w,r}(p)$. Letting r and w be fixed as above, once a point $p \in D_- \setminus \cup R$ is found, we perform a binary search along r in order to find a point of $I_{w,r}(p)$, and then to determine its endpoints. The binary search is done as follows. With each point of r we maintain the height of $e(x)$ and which point(s) of S_+ bounds it. If $x \in I_{w,r}(p)$ (i.e. $p \in e(x)$) we skip to the second part of the search - determining the endpoints of $I_{w,r}(p)$. Otherwise ($p \notin e(x)$) we can deduce from the configuration of x, p, and the point $q \in S_+$ that bounds $e(x)$ whether $I_{w,r}(q)$, if not empty, lie to the left or to the right of x. Again, details are omitted from this extended abstract. Thus the time needed to compute $I_{w,r}(p)$ for all rows r and all values of $1 \leq w \leq n$ is $O(n^2 \log n)$.

Maintaining the weights. In order to maintain the weight of the ellipses efficiently, to [1], we do not construct the weights explicitly. Instead, for each row r and each possible fixed width w, $(1 \leq w \leq n)$ of the ellipses, we construct a binary balanced search tree $\mathcal{T} = \mathcal{T}_{r,w}$ on the points of r, sorted by their x-coordinate. Each leaf $v \in \mathcal{T}$ is associated with the maximal ellipse of width w whose center is (at the point) v. Each node $\mu \in \mathcal{T}$ maintains a weight ω_μ, a subinterval $I_\mu \subseteq r$ of all points which are descendants of μ, and the sum σ_μ of all the weights of all its descendants leaves. In order to double the weights of all maximal ellipses of width w whose center lie on $I_{w,r}(p)$, we double the field ω_μ for each node μ for which $I_\mu \subseteq I_{w,r}(p)$, but $I_{father(\mu)}$ is not contained in $I_{w,r}(p)$. Analogous to standard segment trees, we see that we need to update only $O(\log n)$ fields ω_μ, and that computing the weight of a specific maximal ellipse is doable in $O(\log n)$. Thus the running time of each phase is $O(n^2 \log n)$, since we need to repeat this process for every value of w (the width of the ellipse) and every row r, we obtain that the updating the weights is doable in $O(n^2 \log n)$. Note that in order to compute the weight of an ellipse E associated with a leaf $v \in \mathcal{T}$, we multiply the weights ω_μ along the path from v to the root of $\mathcal{T}$.

Picking the random subset is done as follows. We repeatedly pick a tree $\mathcal{T}$ at random, according to the sum of weights of leaves of the tree, which is stored in the field $\sigma_{root(\mathcal{T})}$. Next we compute a random path in the tree (from the root to a leaf), where once visiting a node μ, the probability of branching to the left node $left(\mu)$ or to the right node $right(\mu)$. depends on ratio of the sum of weights $\sigma_{left(\mu)}$ and $\sigma_{right(\mu)}$. The ellipse we pick is the one assosiated with the leaf that ends the path. This concludes the proof of Theorem 1.

4 Conclusions

We have studied a well motivated covering problem for polygonal regions. The randomized LP based algorithm gives satisfying solutions for instance sizes relevant in the application. More advanced algorithmic ideas and data structures like those discussed in Section 3, although interesting from a theoretical point of view, are at present unlikely to be of use in efficient practicable implementations. Nevertheless, both approximate and exact set covering problems with additional intersection restrictions (geometric or combinatorial) deserve to be studied also in the theoretical context (hardness results, approximations,etc.).

Acknowledgments

We would like to that Helmut Alt, Sariel Har-Peled and Günter Rote for fruitful discussions.

References

[1] P.K. Agarwal and C.M. Procopiuc, Covering Points by Strips in the Plane, *Proc. 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2000, 538–547.

[2] V. Anil Kumar, S. Arya and H. Ramesh, Hardness of Set Cover with Intersection 1, to appear in Proc. ICALP'2000

[3] R. Appel, J. Vargas, P. Palagi, D. Walther and D. Hochstrasser, Melanie II, a third–generationsoftware package for analysis of two–dimensional electrophoresis images: II. Algorithms, *Electrophoresis* 18 (1997), 2735–2748

[4] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry, Algorithms and Applications, Springer-Verlag, 1997.

[5] H. Brönnimann and M. T. Goodrich, Almost Optimal Set Covers in Finite VC-Dimension *Discrete Comput. Geom.* 14 (1995) 463–479.

[6] CAROL, Software system for Matching 2DE Gel Images, *http://gelmatching.inf.fu–berlin.de*

[7] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms* MIT Press, 1990.

[8] CPLEX software, *http://www.cplex.com*

[9] A. Efrat, M.J. Katz, F. Nielsen and M. Sharir, Dynamic Data Structures for Fat Objects and Their Applications, *Proc. 5th Workshop Algorithms Data Struct.* 1997, 297–306.

[10] S. Har-Peled, Taking a Walk in a Planar Arrangement, *Proceedings 40th Annual IEEE Symposium on Foundations of Computer Science* 1999.

[11] F. Hoffmann, K. Kriegel and C. Wenk, An applied pattern matching problem: comparing 2D patterns of protein spots, Discrete Applied Mathematics 93 (1999), 75–88

[12] J. Garrels, The QUEST System for quantitative analysis of 2D gels, J. Biological Chemistry, 264 (1989), 5269–5282

[13] F. Hoffmann, K. Kriegel and C. Wenk, Matching 2D Patterns of Protein Spots, in Proceedings 14th Annual ACM Symposium on Computational Geometry, June 7–10 1998, 231–239

[14] M. Jerrum, Large Cliques Elude the Metropolis Process, Random Structures and Algorithms 3 (4), 1992, 347–359

[15] K.–P. Pleißner, F. Hoffmann, K. Kriegel, C. Wenk, S. Wegner, A. Sahlström, H. Oswald, H. Alt, E. Fleck, New algorithmic approaches to protein spot detection and pattern matching in two–dimensional electrophoresis gel databases, Electrophoresis 20 (1999), 755–765

[16] M. Sharir and P.K. Agarwal, *Davenport-Schinzel Sequences and Their Geometric Applications*, Cambridge University Press, New York, 1995.

[17] M. R. Wilkins, K. L. Williams, R. D. Appel, D. F. Hochstrasser (Eds.) *Proteome Research: New Frontiers in Functional Genomics*, Springer-Verlag Berlin Heidelberg New York, 1997