アルゴリズム 84−11
（2002．5．23）

半空間の和集合の例からの推定

阿久津 達也[1,2]　　オット ザーシャ[3]

[1] 京都大学 化学研究所 バイオインフォマティクスセンター

[2] 京都大学大学院 情報学研究科 知能情報学専攻

[3] 東京大学 医科学研究所 ヒトゲノム解析センター

半空間を利用して分類を行う研究はサポートベクタマシンなどを始めとして盛んに行われている。本稿では、ユークリッド空間上の正例と負例が与えられた時、半空間の和集合を用いて分離する問題を考える。なお、和集合は正例のみをカバーするものとする。本稿では、分離するために必要な半空間の個数を最小化する問題について主に考察し、一般にはグラフ彩色問題と少なくとも同等以上に近似困難であることを示す。一方、２次元においては多項式時間で最適解が計算できることも示す。さらに、近似的に分類する場合などの関連する結果についても示す。

Inferring a Union of Halfspaces from Examples

Tatsuya Akutsu[1,2]　　Sascha Ott[3]

[1]Bioinformatics Center, Institute for Chemical Research, Kyoto University,
Gokasho, Uji, Kyoto-Fu 611-0011, Japan.

[2]Dept. Intelligence Science and Technology, Graduate School of Informatics, Kyoto University,
Yoshidahoncho, Sakyo-ku, Kyoto 606-8501, Japan.

[3]Human Genome Center, Institute of Medical Science, University of Tokyo,
4-6-1 Shirokanedai, Minato-ku, Tokyo 108-8639, Japan.

takutsu@kuicr.kyoto-u.ac.jp　ott@ims.u-tokyo.ac.jp

We consider the following problem which is motivated by applications in Bioinformatics: given positive and negative points in d-dimensions, find a minimum cardinality set of halfspaces whose union covers all positive points and no negative points. We prove that approximation of this problem is at least as hard as approximation of graph coloring. On the other hand, we show that the two-dimensional case of the problem can be solved in polynomial time. Other related results are shown, too.

1 Introduction

Separation of *positive examples* from *negative examples* using a *hyperplane* is an important problem in both *machine learning* and *statistics* [2, 3, 6, 13]. Though it is a classic problem, extensive studies are still being done motivated by the invention of the *support vector machine* [6]. Recent studies focus on finding large margin classifiers [6].

Learning a *mixture of models* is another important problem in machine learning and statistics, because a single model is not always enough to characterize the given data (the given examples). Using multiple sets of parameters is also important in Bioinformatics, because parameter sets (for example, amino acid scores) sometimes depend on environments [7, 15]. For example, *Dirichlet mixture* was effectively applied to sequence analysis [7]. Many techniques have been proposed for deriving a mixture of probability distributions such as the EM algorithm and local search heuristics [4, 7]. Recently, Arora and Kannan proposed approximation algorithms for deriving a mixture of Gaussian distributions [4].

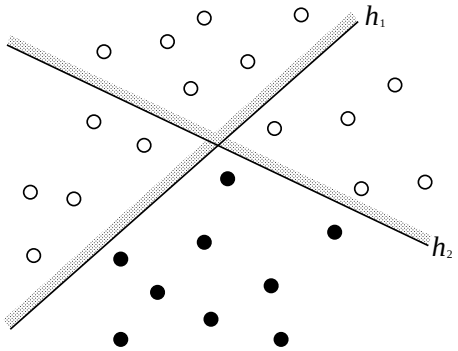

Figure 1: Separation of positive points from negative points using two halfspaces. White circles and black circles denote positive points and negative points, respectively.

However, to our knowledge, there had been no algorithmic study on deriving a union or a mixture of halfspaces from examples. Therefore, we study the computational complexity of this problem and derive inapproximability as well as approximability results. Furthermore, we show that this problem has a close relationship to the problem of deriving a PSSM (*Position Specific Score Matrix*) from examples [1], where PSSMs are widely used in Bioinformatics [7]. We define the problem in the following way, where a halfspace means a *closed halfspace*.

Problem 1.
Given point sets POS and NEG in d-dimensional Euclidean space, find a minimum cardinality set of halfspaces $\{h_1, \ldots, h_k\}$ such that $POS \subseteq \bigcup_{i=1,\ldots,k} h_i$ and $NEG \cap \left(\bigcup_{i=1,\ldots,k} h_i \right) = \emptyset$.

Along with the above, we consider the following decision problem.

Problem 2.
Given point sets POS and NEG in d-dimensional Euclidean space and an integer K, find a set of halfspaces $\{h_1, \ldots, h_K\}$ such that $POS \subseteq \bigcup_{i=1,\ldots,K} h_i$ and $NEG \cap \left(\bigcup_{i=1,\ldots,K} h_i \right) = \emptyset$.

It should be noted that there is a solution for K in Problem 2 if there is a solution for $K-1$. Hereafter, we let $n = |POS|$ and $m = |NEG|$.

In this paper, we prove that approximation of Problem 1 is at least as hard as approximation of graph coloring. We also prove that Problem 2 is NP-hard even for $K = 2$. This result is interesting because Problem 2 can be trivially solved in polynomial time using *linear programming* if $K = 1$. On the other hand, we show that the two-dimensional case of Problem 1 can be solved in polynomial time. We present approximation algorithms for a special case of Problem 1 and variants (maximization of the number of correctly classified examples) of Problem 2. Though the *Boosting* technique [10] might be applied to these problems, it would not solve the problems optimally or near optimally because we derive strong hardness results.

2 Hardness Results

We briefly introduce the problem of deriving a mixture of PSSMs [1]. Let POS_{pssm} and NEG_{pssm} be sets of strings of length l over an alphabet Σ. For a string S, $S[i]$ denotes the i-th letter of S. A PSSM is a function $f_i(a)$ from $[1, \ldots, l] \times \Sigma$ to the set of real numbers, where

$i \in [1, \ldots, l]$ and $a \in \Sigma$. For a string S and a PSSM $f_i(a)$, we define $f(S)$ (the score of S) by $f(S) = \sum_{i=1,\ldots,l} f_i(S[i])$.

Problem 3. (Derivation of a Mixture of PSSMs)
Given Σ, POS_{pssm}, NEG_{pssm} and a positive integer K, find a set of K PSSMs and a threshold Θ which satisfy the following conditions:

- For all $S \in POS_{pssm}$, $f^k(S) \geq \Theta$ for some $k \in [1, \ldots, K]$,
- For all $S \in NEG_{pssm}$ and for all $k \in [1, \ldots, K]$, $f^k(S) < \Theta$,

where f^k denotes the k-th PSSM.

It was proven in [1] that Problem 3 is NP-hard even for $K = 2$ and $\Sigma = \{0, 1\}$.

Theorem 1.
Problem 2 is NP-hard even for $K = 2$.
(Proof)
We use a polynomial time reduction from Problem 3.

We identify Σ with $\{1, 2, \ldots, |\Sigma|\}$. For a point $\boldsymbol{p}$ in $(l|\Sigma|)$-dimensional Euclidean space, $\boldsymbol{p}[i]$ denotes the i-th coordinate value of $\boldsymbol{p}$. From each string S of length l in $POS_{pssm} \cup NEG_{pssm}$, we create a point $\boldsymbol{s}$ by $\boldsymbol{s}[|\Sigma|(i-1) + S[i]] = 1$ for $i = 1, \ldots, l$, $\boldsymbol{s}[j] = 0$ for the other j.

We identify a PSSM $f_i(a)$ with an $(l|\Sigma|)$-dimensional vector $\boldsymbol{a}$ by $\boldsymbol{a}[|\Sigma|(i-1) + a] = f_i(a)$ for $i = 1, \ldots, l$ and $a = 1, \ldots, |\Sigma|$.

Then, we have the following relation:

$$f(S) \geq \Theta \text{ iff. } \boldsymbol{a} \cdot \boldsymbol{s} \geq \Theta,$$

where $\boldsymbol{a} \cdot \boldsymbol{s}$ is the *inner product* between $\boldsymbol{a}$ and $\boldsymbol{s}$. That is, each PSSM corresponds to a halfspace. It should be noted that each halfspace $\boldsymbol{a} \cdot \boldsymbol{s} \geq \theta$ can be normalized so that the righthand side value takes Θ.

From this relation, it is straight-forward to see the following:

there exists a solution $(\{f^1, \ldots, f^K\}, \Theta)$ for Problem 3 iff.
there exists a solution $\{ \boldsymbol{a}^1 \cdot \boldsymbol{x} \geq \Theta, \ \ldots, \ \boldsymbol{a}^K \cdot \boldsymbol{x} \geq \Theta \}$ for Problem 2.

Therefore, the theorem follows from the NP-hardness result on Problem 3 [1]. □

Theorem 2.
Problem 3 can not be approximated within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$ in polynomial time unless $ZPP = NP$, where n denotes the number of positive strings. [1]
(Proof)
We construct an *approximation preserving reduction* from Minimum Graph Coloring (GT4 in [11]) to Problem 3. Let $G = (V, E)$ be an undirected graph. We define POS_G as

$$POS_G = \{0^{i-1}10^{n-i} \mid x_i \in V\},$$

where we choose an arbitrary ordering of the vertices in V and $n = |V|$. Furthermore, we define NEG_G as

$$NEG_G = \{0^{i-1}10^{j-i-1}10^{n-j} \mid (x_i, x_j) \in E, \ i < j\} \cup \{0^n\}.$$

[1] We say that a minimization problem (resp. a maximization problem) can be approximated within a factor of $f(n)$ if there is an algorithm for which $\max(\frac{APR}{OPT}, \frac{OPT}{APR}) \leq f(n)$ holds where APR and OPT are the scores of an approximate solution and an optimal solution, respectively.

$I = (POS_G, NEG_G)$ forms an input for Problem 3. We denote strings of POS_G corresponding to vertex x_i as w_{x_i} and strings of NEG_G corresponding to $e \in E$ as w_e.

It suffices to show, that there is a solution for I with K matrices, iff G can be colored with K colors. Let $g : V \to \{1, \ldots, K\}$ be a coloring for G. For $k \in \{1, \ldots, K\}$, we define a PSSM f^k. Let $i \in \{1, \ldots, n\}$ and $a \in \{0, 1\}$.

$$f_i^k(a) = \begin{cases} 0 & \text{if } a = 0 \\ 1 & \text{if } a = 1,\ g(x_i) = k \\ \text{-1} & \text{if } a = 1,\ g(x_i) \neq k \end{cases}$$

With $\Theta = 1$, f^k accepts all strings of POS_G, which correspond to a vertex colored with color k. Since every string in NEG_G corresponds to an edge of G and all edges of G connect vertices with different colors, all negative strings are rejected by all PSSMs. Therefore, $f^1, \ldots, f^K$ is a solution for I with K PSSMs.

For the proof of the opposite direction, suppose there is a solution for I with K PSSMs. For every vertex x_i, we choose a PSSM f^k accepting w_{x_i} and color x_i with color k. If there were an edge $(x_i, x_j) \in E$ with both x_i and x_j colored with the same color k, we could conclude

$$f^k(w_{(x_i,x_j)}) + f^k(0^n) = f^k(w_{x_i}) + f^k(w_{x_j}) \geq 2\Theta,$$

which is a contradiction to f^k rejecting both $w_{(x_i,x_j)}$ and 0^n.

Thus, the above construction of an input for Problem 3 yields an approximation preserving reduction from Minimum Graph Coloring to Problem 3. Since Minimum Graph Coloring can not be approximated within $O(|V|^{1-\epsilon})$ unless $ZPP = NP$ ([9]), we have the theorem. □

Using the reduction in the proof of Theorem 1, we have:

Corollary 1.
Problem 1 can not be approximated within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$ in polynomial time unless $ZPP = NP$.

In [1], it was shown that a set of $|POS|$ PSSMs can be computed by linear programming. Though it is a trivial solution to compute one PSSM for each positive string, this simple algorithm is surprisingly nearly optimal, since Theorem 2 implies that the minimal number of PSSMs can not be approximated within a factor of $O(|POS|^{1-\epsilon})$.

3 A Polynomial Time Algorithm in Two-Dimensions

We assume without loss of generality that all points in $POS \cup NEG$ are in general positions (i.e., no three points are collinear), where we can modify the algorithm for a general case without increasing the order of the time complexity.

For sets X and Y, $X - Y$ denotes the subset of X obtained by removing $X \cap Y$ from X. Let Ω denote the *convex hull* [8] of NEG, where the boundary is included in Ω. Let $\delta\Omega$ denote the boundary of Ω and $\Omega^- = \Omega - \delta\Omega$. Similarly, δh denotes the boundary of a halfplane h.

Proposition 1. (see Fig. 2)
For each halfplane h such that $h \cap NEG = \emptyset$ and $h \cap POS \neq \emptyset$, there exists a halfplane h' satisfying the following conditions: (i) $h' \cap \Omega^- = \emptyset$, (ii) $(h \cap POS) \subseteq (h' \cap POS)$, (iii) $|h' \cap NEG| = 1$, (iv) $|\delta h' \cap POS| = 1$.
(Proof)
First, simply translate h so that the intersection with NEG consists of exactly one point. Next, rotate the halfplane around the point so that the intersection of the boundary with POS consists of exactly one point. □

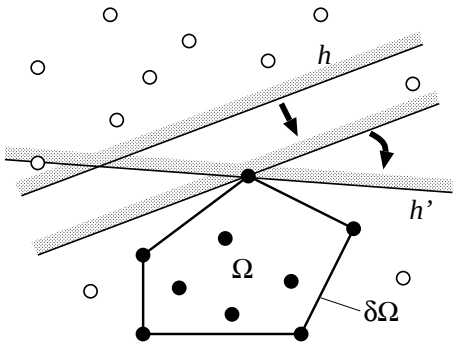

Figure 2: Explanation of Proposition 1.

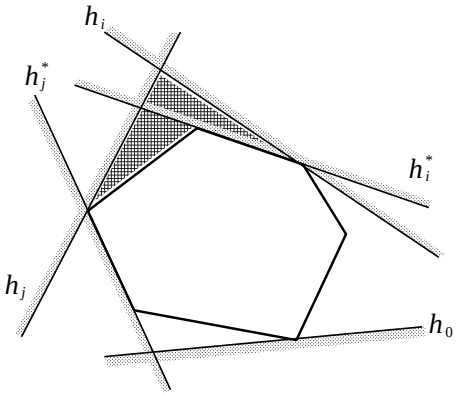

Figure 3: Explanation of Algorithm 1. $T[i]$ is updated if the halftone region $(h^*_{j,i} - h_j - h_i)$ does not contain a point of POS.

It follows that there exists a solution for Problem 1 and Problem 2 in which each halfplane is obtained by slightly perturbing a halfplane satisfying the conditions of Proposition 1.

Let H be the set of halfplanes satisfying the conditions of Proposition 1, where the number of such halfplanes is $O(n)$. We fix an arbitrary element h_0 in H. For each halfplane h, we define $angle(h)$ (the angle of h to h_0) as follows. Let $\boldsymbol{q}$ be the intersection point of δh and δh_0. Assume that h_0 coincides with h if h_0 is rotated clockwise around $\boldsymbol{q}$ by A radian ($0 \leq A < 2\pi$). Then, we define $angle(h) = A$.

We define the order on halfplanes by $h \prec h'$ iff. $angle(h) < angle(h')$. For example, $h_0 \prec h_j^* \prec h_j \prec h_i^* \prec h_i$ holds for halfplanes in Fig. 3. It should be noted that $angle(h)$ is a bijection from H to $[0, 2\pi)$ if the domain of $angle(h)$ is restricted to H. Since each $h \in H$ is tangent to Ω, the ordering gives the *total order* on H. This ordering allows us to use *dynamic programming*.

Let $< \boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_h >$ be the points in $\delta\Omega \cap NEG$, where these points are arranged in the clockwise order. For each $h \in H$, h^* is the halfplane such that $h^* \supset \{\boldsymbol{q}_i, \boldsymbol{q}_{i-1}\}$ and $h^* \cap \Omega^- = \emptyset$ hold, where $\boldsymbol{q}_i = h \cap NEG$ and $\boldsymbol{q}_0 = \boldsymbol{q}_h$. Let $< h_0, h_1, \ldots, h_N >$ be the sorted list of H. For each pair of halfplanes (h_j, h_i) such that $j < i$, we define $h^*_{j,i}$ by $h^*_{j,i} = \left(\bigcup_{k=j,\ldots,i} h_k^*\right) - h_j^*$, where $h^*_{j,i} = \emptyset$ if $h_j^* = h_i^*$.

Algorithm 1 shown below (see also Fig. 3) outputs the minimum number of halfplanes, where

it can be easily modified for computing the halfplanes. Though we assume that $h_0^* \neq h_{i_k}^*$ holds for the optimal solution $< h_{i_1} = h_0, h_{i_2}, \ldots, h_{i_k} >$, Algorithm 1 can be modified for the case of $h_0^* = h_{i_k}^*$ without increasing the order of the time complexity.

ALGORITHM 1
  Construct the convex hull Ω of NEG;
  if $POS \cap \Omega \neq \emptyset$ **then** output "no solution" and **halt**;
  $k \leftarrow +\infty$;
  for all $h_0 \in H$ **do** $k \leftarrow \min(k, \text{SUBPROC}(h_0))$;
  Output k;

SUBPROC(h_0)
  Let $< h_0, h_1, \ldots, h_N >$ be the sorted list of H;
  if $h_0 \cap POS = POS$ **return** 1;
  $T[0] \leftarrow 1$;
  for $i = 1$ **to** N **do** $T[i] \leftarrow +\infty$;
  for $i = 1$ **to** N **do**
    for $j = 0$ **to** $i - 1$ **do**
      if $\left(h_{j,i}^* \bigcap POS\right) \subseteq h_j \cup h_i$ **then** $T[i] \leftarrow \min(T[i],\ T[j] + 1)$;
  for $i = 1$ **to** N **do**
    if $\left(POS - h_{0,i}^*\right) - (h_0 \cup h_i) \neq \emptyset$ **then** $T[i] \leftarrow +\infty$;
  return $\min\{T[i] \mid i = 1, \ldots, N, h_i^* \neq h_0^*\}$;

Algorithm 1.

Theorem 3.
The two-dimensional case of Problem 1 can be solved in polynomial time.
(Proof)
First, we show the correctness of Algorithm 1. Clearly, Problem 1 has a solution if and only if $POS \cap \Omega = \emptyset$.

From Proposition 1, it suffices to find a minimum cardinality set of halfplanes satisfying $POS \subseteq \cup h_i$ and the conditions of Proposition 1. Let H_{opt} be such a set.

Then, we can put the total order on H_{opt} by choosing an arbitrary element h_0 in H_{opt}. Let $< h_0 = h_{i_1}, h_{i_2}, \ldots, h_{i_k} >$ be the ordered list of H_{opt}. Since all the elements in POS must be covered by $\cup h_{i_k}$, $\left(h_{i_j, i_{j+1}}^* \cap POS\right) \subseteq h_{i_j} \cup h_{i_{j+1}}$ must hold for $j = 1, \ldots, k-1$ and $\left(POS - h_{0,i_k}^*\right) \subseteq (h_0 \cup h_{i_k})$ must hold. Algorithm 1 finds a smallest H_{opt} satisfying this condition.

Next we analyze the time complexity. The convex hull of NEG can be constructed in $O(m \log m)$ time [8]. Before applying the dynamic programming, we compute $h_{i,j}^* \cap POS$ for all i, j. It takes $O((n+m)^2 n)$ time using an incremental procedure, where we omit details. Since both $|H|$ and $|POS|$ are $O(n)$, the dynamic programming procedure takes $O(n^3)$ time per h_0.

Therefore, Algorithm 1 takes $O(n^4 + m^2 n)$ time in total. □

4 Approximation Algorithms in d-Dimensions

In this section, we assume that all points in $POS \cup NEG$ are in general positions. It is easy to see that Problem 2 can be solved in polynomial time if both d and K are fixed.

Proposition 2.
Problem 2 can be solved in $O((n+m)^{dK+1})$ time for constants d and K.

Using a reduction to the set cover problem [11, 12], we can also have:

Proposition 3.
Problem 1 can be approximated within a factor of $O(\log n)$ in $O((n+m)^{d+1})$ time if d is a constant.

From a practical viewpoint, it is more important to develop algorithms for minimizing the number of misclassified examples or maximizing the number of correctly classified examples, where the number of halfspaces is bounded by a constant K. We consider the latter case in this paper, because it seems difficult to develop a good approximation algorithm for the former problem [3]. Let H be a set of halfspaces. $\boldsymbol{p} \in POS$ (resp. $\boldsymbol{p} \in NEG$) is called a *true positive* (resp. a *false positive*) if $\boldsymbol{p} \in h$ for some $h \in H$. $\boldsymbol{p} \in NEG$ (resp. $\boldsymbol{p} \in POS$) is called a *true negative* (resp. *false negative*) if $\boldsymbol{p} \notin h$ for any $h \in H$. Let $\#TP$ and $\#FP$ denote the numbers of true positives and false positives, respectively. Let $\#TN$ and $\#FN$ denote the numbers of true negatives and false negatives, respectively. The following proposition is almost trivial (or directly follows from the result of Amaldi and Kann [2]).

Proposition 4.
The maximization problem of $\#TP + \#TN$ can be approximated within a factor of 2 in polynomial time.

We are also interested in maximizing $\#TP$ under the condition that $\#FP = 0$ since it is sometimes important not to output false positives. We call it Problem 4. Clearly, Problem 4 is NP-hard even for $K = 2$. We propose a simple randomized approximation algorithm (Algorithm 2) for the case of $K = 2$.

ALGORITHM 2
Let R be a set of r points randomly selected from POS;
$H_{max} \leftarrow \emptyset$;
for all partition $R_1 \cup R_2$ of R **do**
$H \leftarrow \emptyset$;
if there is a halfspace h_1 such that $h_1 \cap R_1 = R_1$ and $h_1 \cap NEG = \emptyset$
then $H \leftarrow H \cup h_1$;
if there is a halfspace h_2 such that $h_2 \cap R_2 = R_2$ and $h_2 \cap NEG = \emptyset$
then $H \leftarrow H \cup h_2$;
if $|H \cap POS| > |H_{max} \cap POS|$ **then** $H_{max} \leftarrow H$;
Output H_{max};

Algorithm 2.

The performance of Algorithm 2 is poor if $\#TP$ in the optimal solution is small (e.g., $\#TP$ is $O(\log n)$). However, such a case is meaningless since we need a classifier with small errors. Therefore, we are interested in the case where $\#TP$ is large enough in the optimal solution. In order to analyze Algorithm 2 for such a case, we use β-center points. A point $c \in \mathcal{R}^d$ is called a *β-center point* of a point set P if every closed halfspace containing c contains at least βn points of P [5]. Clarkson *et al.* showed that $\left(\frac{1}{d+1} - \epsilon\right)$-center point can be found in $O((d/\epsilon)^2 \log(d/\epsilon))^{d+O(1)} \log(1/\delta))$ time with probability $1-\delta$ by using $O((d/\epsilon)\cdot\log(d/\epsilon)\cdot\log(1/\delta))$ elements randomly sampled from P.

Theorem 4.
Let $K = 2$. Suppose that $\#TP > n/2$ holds in the optimal solution for Problem 4. For any constants d, $\delta > 0$ and $\epsilon > 0$, Algorithm 2 outputs a pair of hyperplanes such that $\#FP = 0$ and $\#TP \geq \frac{n}{4} \cdot \left(\frac{1}{(d+1)} - \epsilon\right)$ in $O(n+m)$ time with probability $1 - \delta$.

(Sketch of Proof)
Let (h_a, h_b) be an optimal solution, where we assume without loss of generality that $|h_a \cap POS| \geq |h_b \cap POS|$. Using a sufficiently large constant r, there exists R_1 with high probability which can be considered as a random sample of size $O((d/\epsilon) \cdot \log(d/\epsilon) \cdot \log(1/\delta))$ from $h_a \cap POS$. Since h_1 contains R_1 and $|h_a \cap POS| \geq \frac{n}{4}$ holds, h_1 contains $\frac{n}{4} \cdot \left(\frac{1}{d+1} - \epsilon\right)$ points of POS with high probability.

Since r is a constant and linear programming can be done in linear time for fixed d [14], Algorithm 2 works in linear time. □

It should be noted that Problem 4 can be solved exactly in $O((n+m)^{dK+1})$ time for fixed d and K as in Proposition 2.

References

[1] Akutsu, T., Bannai, H., Miyano, S. and Ott, S., On the complexity of deriving position specific score matrices from examples, *IPSJ Technical Report*, AL-82-5, 27–33, 2002.

[2] Amaldi, E. and Kann, V., The complexity and approximability of finding maximum feasible subsystems of linear relations, *Theoretical Computer Science*, 147, 181–210, 1995.

[3] Amaldi, E. and Kann, V., On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems, *Theoretical Computer Science*, 209, 237–260, 1998.

[4] Arora, S. and Kannan, R., Learning mixtures of arbitrary gaussians, *Proc. 33rd ACM Symp. Theory of Computing*, 247–257, 2001.

[5] Clarkson, K. L., Eppstein, D., Miller, G. L., Sturtivant, C. and Teng S-H., Approximating center points with iterated radon points, *Proc. 9th ACM Symp. Computational Geometry*, 91–98, 1993.

[6] Cortes, C. and Vapnik, V., Support-vector networks, *Machine Learning*, 20, 273–297, 1995.

[7] Durbin, R., Eddy, S., Krogh, A. and Mitchison, G., *Biological Sequence Analysis. Probabilistic Models of Proteins and Nucleic Acids*, Cambridge University Press, 1998.

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

[9] Feige, U. and Kilian, J., Zero knowledge and the chromatic number, *Journal of Computer and System Sciences*, 57, 187-199, 1998.

[10] Freund, Y. and Schapire, R. E., A decision-theoretic generalization of on-line learning and an application to boosting, *Journal of Computer and System Sciences*, 55, 119-139, 1997.

[11] Garey, M. R. and Johnson, D. S., *Computers and Intractability*, Freeman, 1979.

[12] Johnson, D. S., Approximation algorithms for combinatorial problems, *J. Computer and System Sciences*, 9, 256–278, 1974.

[13] Kearns, M. J. and Vazirani, U.V., *An Introduction to Computational Learning Theory*, The MIT Press, 1994.

[14] Megiddo, N., Linear programming in linear time when the dimension is fixed, *SIAM J. Computing*, 12, 759–776, 1983.

[15] Radivojac, R., Obradoviv, Z., Brown, C. J. and Dunker, A. K., Improving sequence alignments for intrinsically disordered proteins, *Proc. Pacific Symp. Biocomputing*, 2002.