International Journal of Computational Geometry & Applications

APPROXIMATING SMALLEST ENCLOSING BALLS WITH APPLICATIONS TO MACHINE LEARNING*

FRANK NIELSEN

Sony Computer Science Laboratories Inc.
3-14-13 Higashi Gotanda
141-0022 Shinagawa-Ku, Tokyo, Japan
and
École Polytechnique, LIX
Route de Saclay
F-91128 Palaiseau Cedex, France
Frank.Nielsen@acm.org

RICHARD NOCK

University of Antille-Guyane, CEREGMIA
Campus de Schoelcher, B.P. 7209
97275 Schoelcher Cedex, Martinique, France
Richard.Nock@martinique.univ-ag.fr

Received June 2007
Revised December 2007, Accepted February 2008
Communicated by Deok-Soo Kim

In this paper, we first survey prior work for computing exactly or approximately the smallest enclosing balls of point or ball sets in Euclidean spaces. We classify previous work into three categories: (1) purely combinatorial, (2) purely numerical, and (3) recent mixed hybrid algorithms based on coresets. We then describe two novel tailored algorithms for computing arbitrary close approximations of the smallest enclosing Euclidean ball of balls. These deterministic heuristics are based on solving relaxed decision problems using a primal-dual method. The primal-dual method is interpreted geometrically as solving for a minimum covering set, or dually as seeking for a minimum piercing set. Finally, we present some applications in machine learning of the exact and approximate smallest enclosing ball procedure, and discuss about its extension to non-Euclidean information-theoretic spaces.

Keywords: Smallest Euclidean enclosing ball, duality piercing/covering, coresets.

*This article revises and extends in light of the recent research results the paper[1] that first appeared at the Computational Geometry and Applications (CGA) workshop of the 2004 International Conference on Computational Science and Its Applications (ICCSA), Lecture Notes in Computer Science series, Volume 3045, pp. 147–157, DOI 10.1007/b98053 (Springer-Verlag).

1. Introduction

As far as one surveys the scientific literature, the *smallest enclosing disk* (SED for short) problem is traditionally reported to date back to 1857 when J. J. Sylvester[2,3,4] first asked for the smallest radius disk enclosing a given set of n points on the *plane*. More formally, let $x_i(S) = s_i$ denote the i-th coordinate of point $S = (s_1, ..., s_d)$ $(1 \leq i \leq d)$ of d-dimensional Euclidean space $\mathbb{E}^d$, and denote by $\text{Ball}(S, r)$ the Euclidean ball of center S and radius r: $\text{Ball}(S, r) = \{X \in \mathbb{E}^d \mid ||SX|| \leq r\}$, where $||\cdot||$ denotes the Euclidean distance (L_2-norm) $||PQ|| = \sqrt{\sum_{i=1}^{d}(x_i(P) - x_i(Q))^2}$ of $\mathbb{E}^d$. Further, let $\mathcal{B} = \{B_1, ..., B_n\}$ be a set of n d-dimensional balls, such that $B_i = \text{Ball}(S_i, r_i)$ for $i \in \{1, ..., n\}$. Let $\mathcal{S}$ be the set of ball centers $\mathcal{S} = \{S_1, ..., S_n\}$. Given a ball B, denote by $r(B)$ its radius and $C(B)$ its center so that $C(B_i) = S_i$ and $r(B_i) = r_i$ for all $i \in \{1, ..., n\}$. The smallest enclosing ball (SEB) of $\mathcal{B}$ is the *unique ball*[5,6], $B^* = \text{SEB}(\mathcal{B}) = \text{Ball}(C^*, r^*)$, fully enclosing $\mathcal{B}$ ($\mathcal{B} \subseteq \text{Ball}(C^*, r^*)$) of minimum radius r^*. Figure 1 depicts an example of a SEB of a planar ball set. A point set can be viewed as a degenerated ball set by fixing all radii to zero. Finding the center of the SEB of a ball set $\mathcal{B} = \{B_1, ..., B_n\}$ can be written mathematically into the following *minimax* optimization problem:

$$r^* = \min_{C \in \mathbb{E}^d} \max_{i=1}^{n}(||CS_i|| + r_i), \tag{1}$$

$$C^* = \text{argmin}_{C \in \mathbb{E}^d} \max_{i=1}^{n}(||CS_i|| + r_i). \tag{2}$$

Note that the result of this optimization problem generalizes to *any* strictly monotonous function applied on $||CS_i|| + r_i$. That is, for a given strictly monotonous function f, $f(r^*)$ is the optimal solution of $\min_{C \in \mathbb{E}^d} \max_{i=1}^{n} f(||CS_i|| + r_i)$. In practice, when dealing with point sets (all r_i's set to zero), we purposely choose the squared function $||CS_i||^2$ to get rid off the unnecessary square root operations.

The smallest enclosing ball problem belongs to the family of *minimum containment problems* and is also referred in the literature under various terms such as the minimum enclosing ball, minimum spanning ball, minimum covering sphere, Euclidean 1-center, d-outer radius, minimum bounding sphere, or minimax problem in facility locations, etc. The smallest enclosing ball is a fundamental primitive that finds many applications in computer graphics (collision detection, visibility culling, ...), machine learning (support vector clustering, similarity search, ...), metrology (roundness measurements, ...), facility locations (base station locations, ...), and so on. Notice that in the aforementioned applications, *approximate* solutions are often enough. A $(1 + \epsilon)$-approximation of *the* SEB is *a ball* $\text{Ball}(S', r')$ such that $\mathcal{B} \subset \text{Ball}(S', r')$ with $r' \leq (1 + \epsilon)r^*$ for a prescribed threshold value $\epsilon > 0$.

The structure of the paper is as follows: we survey in the next section the *main* algorithms for computing either *exactly* or *approximately* the smallest enclosing balls.

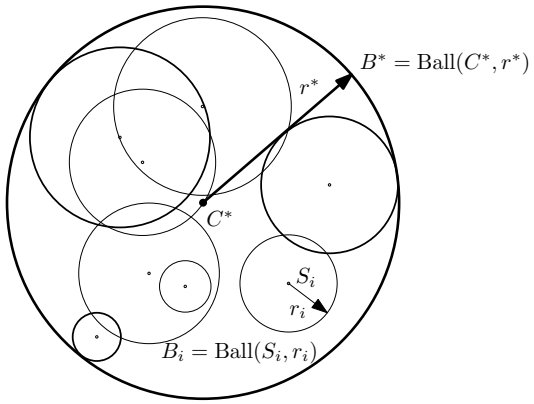

Fig. 1. The smallest enclosing ball of a set of balls is unique and combinatorially induced by at most $d + 1$ balls tangent to its boundary.

We classify the rich literature of previous work according to three *algorithmic paradigms*:

(1) Exact combinatorial algorithms,
(2) Approximate numerical algorithms,
(3) Approximate hybrid (semi-numerical) algorithms.

Section 3 describes a computational efficient filtering mechanism for calculating the maximum distance of a point to a given point set. This general technique is then used in section 4 to speed-up an implementation of the Bădoiu -Clarkson and Panigrahy core-set approximation algorithms[7,8]. Section 5 presents a novel core-set primal-dual tailored method based on solving relaxed decision problems geometrically interpreted either as covering and piercing problems. Section 6 gives an alternative approach better suited for small dimensions and discusses on the algebraic degree of predicates used in branching tests. Section 8 introduces the use of the SEB primitive in computational machine learning. Finally, section 9 concludes the paper by presenting a few selected open problems and venues for future research.

2. Previous Work

Although the SEB problem was first tackled in the 19th Century (as mentioned previously in the Introduction), it is still nowadays surprisingly a *very active* and *challenging* mainstream problem of computational geometry that attested major developments during the past few years. It is beyond the scope of this paper to extensively list historically and summarize all the papers dealing with the SEB. We rather present below a *classification* of current SEB algorithms. The classification is organized into three categories: combinatorial (section 2.1), numerical (section 2.2), and hybrid algorithms (section 2.3).

2.1. *Combinatorial algorithms*

The algorithmic complexity of SEB was only settled in 1984 by N. Megiddo's first *linear-time* prune-and-search algorithm[9] for solving linear programs in *fixed* dimension. Later, the method was extended to the case of balls[10,11]. Since the smallest enclosing ball is unique and defined by at most $d+1$ support points[a] (or tangent balls as depicted in Figure 1) in strictly convex position (implying being affinely independent as well), a brute-force naïve combinatorial algorithm requires $O_d(n^{d+2})$ time[b] (with linear memory). Combinatorial SEB algorithms provide not only the explicit SEB Ball(C^*, r^*) but also the support points which lead implicitly to it. A major breakthrough was obtained by E. Welzl[5] who described an elegant *randomized* almost tight expected $\lfloor (e-1)(d+1)! \rfloor n$ time[c] algorithm. The number of *expected* basis computations of size j is shown to be $\tilde{O}(\log^j n)$ $(2 \leq j \leq d+1)$, so that most of the time of Welzl's algorithm is spent by checking whether points/balls are inside some candidate ball or not.[d] For point sets being vertices of a regular simplex, the algorithm exhibits unfortunately the *curse of dimensionality* as it provably requires $\Omega(2^d)$ recursive calls, thus limiting its tractability up to a few dozen dimensions in practice (say, $d \simeq 30$). Recently, Chernoff-type tail bound has been given for *nondegenerate* input data sets by B. Gärtner and E. Welzl[12]. Although the Chernoff-type bound gives a better understanding of the power of randomization, tight worst-case bound is still unknown[e] as is also the tail estimate in the case of cospherical point sets. Subexponential running time was obtained by B. Gärtner[13] who described a general randomized algorithm for the class of so-called *abstract optimization problems* (AOP). Focusing on small instances (*i.e.*, $n = O(d)$), B. Gärtner and E. Welzl[14] presented a practical randomized approach for affinely independent points using $\tilde{O}(1.5^n)$ basis computations. T. Szabo and E. Welzl[15] further improve the bound to $\tilde{O}(1.47^n)$ using the framework of unique sink orientations of hypercubes. So far, B. Chazelle and J. Matoušek gave the current best $O(d^{O(d)}n)$ deterministic time algorithm[16].

From the pratical viewpoint, B. Gärtner[17] updated the move-to-front heuristic of E. Welzl[5] by introducing a pivot mechanism and improving the numerical robustness of basis computations. Furthermore, K. Fischer *et al.*[18] describe a simplex-like pivoting combinatorial algorithm with a Bland-type rule that guarantees termination based on the seminal idea of T. Hopp et C. Reeve[19] of deflating an enclosing sphere: They devise a dynamic data-structure for maintaining intermediate candidate balls and a robust floating-point implementation is tested with point sets up

[a] The circumcenter falls necessarily inside the convex hull of the support points.[11]
[b] Notation $O_d(\cdot)$ means that we hide all d terms in the multiplicative constants of functions of n. For example, $O(2^d n^{d+2}) = O_d(n^{d+2})$.
[c] $e \simeq 2.71828182846...$ is the irrational number such that $\log e = 1$.
[d] This may explain why descriptions of computing the primitives and their time complexity were omitted in[5] since $\sum_{i=2}^{d+1}(2+\ln n)^i = o_d(n)$.
[e] That is, to know the worst-case geometric configuration that implies a worst number of recursive calls (i.e., geometric realization of permutations).

to dimension[f] $d = 10000$. Overall complexity is $O(d^3 + d^2 l)$, where $l \leq \binom{n}{d+1}$ is a finite number of iterations; In practice, although the algorithm requires algebraic degree two on the rationals $\mathbb{Q}$, they observe *good experimental* floating-point errors of at most 10^4 times the machine precision. For ball sets, K. Fischer and B. Gärtner show[11] that E. Welzl's MiniBall algorithm[5] extends to balls provided that the ball centers are *affinely* independent. Furthermore, they provide a linear programming type (LP-type) algorithm which runs in expected $\tilde{O}(2^{O(d)}n)$-time. The combinatorial algorithms described so far compute the exact smallest enclosing ball (*i.e.*, $\epsilon = 0$), report a support point/tangent ball set and look similar to those handling linear programming. Notice that the smallest enclosing ball problem, as LP, is not known to be *strongly* polynomial (see P. Gritzmann and V. Klee[20] for a weakly polynomial algorithm).

2.2. *Numerical algorithms*

Let $d_2(\mathcal{A}, \mathcal{B})$ denote the maximum distance between all pairs of point (A, B), with $A \in \mathcal{A}$ and $B \in \mathcal{B}$. Consider without loss of generality point sets $\mathcal{B}$. Observe that picking any point $P \in \mathcal{B}$ gives a 2-approximate ball $\text{Ball}(P, d_2(P, \mathcal{B}))$ (*i.e.*, $\epsilon = 1$). This allows to easily *convert* from *relative* to *absolute* $(1+\epsilon)$-approximation values. Motivated by computer graphics applications, J. Ritter[21] proposed a simple and fast constant approximation of the smallest enclosing ball that can be extended straightforward for points/balls in arbitrary dimension. Tight worst-case approximation ratio was unknown until very recently[22]. It is easy to check that J. Ritter's heuristic can be as bad as 18.3 percents.[h] H. Zarrabi-Zadeh and T. Chan actually proved that it yields in fact a $\frac{3}{2}$-approximation in *arbitrary* dimension.

It is quite natural to state the smallest enclosing ball problem as a *mathematical program*. In facility locations, the smallest enclosing ball of a ball set $\mathcal{B}$ is often written as $\min_{C \in \mathbb{E}^d} F_{\mathcal{B}}(C)$ where $F_{\mathcal{B}}(X) = d_2(X, \mathcal{B}) = \max_{i \in \{1,\ldots,n\}} d_2(X, B_i)$ is the cost function. For ball sets, we have $d_2(X, B_i) = d_2(X, S_i) + r_i$. Since the minimum is unique, we obtain the circumcenter as $C^* = \text{argmin}_{C \in \mathbb{E}^d} F_{\mathcal{B}}(C)$. Observe that the function $F_B(X)$ is not differentiable everywhere. Namely, the function is differentiable everywhere *except* at the faces of the furthest Voronoi diagram of the point/ball set, where the notion of furthest point is not uniquely defined. Figure 2 displays the discretization of the maximum distance field of a planar point set. Observe that one can *perceive* the *furthest Voronoi diagram* inside this rasterized distance field.

Using the ellipsoid method for solving approximately convex programs (CP), we

[f] In fact, T. Hopp and C. Reeve[19] reported experimentally a time complexity[g] of $\bar{O}(d^{2.3}n)$ for *uniform* spherical data sets.

[h] *E.g.*, considering a regular simplex in dimension 2. In the paper[21], J. Ritter evaluates it to "around" 10 percents (*dixit*). X. Wu.[23] suggests a variant based on finding principal axis as a preprocessing stage of J. Ritter's greedy algorithm. It requires roughly twice more time and do not guarantee to perform better. (Actually, we found it experimentally worse sometimes.)

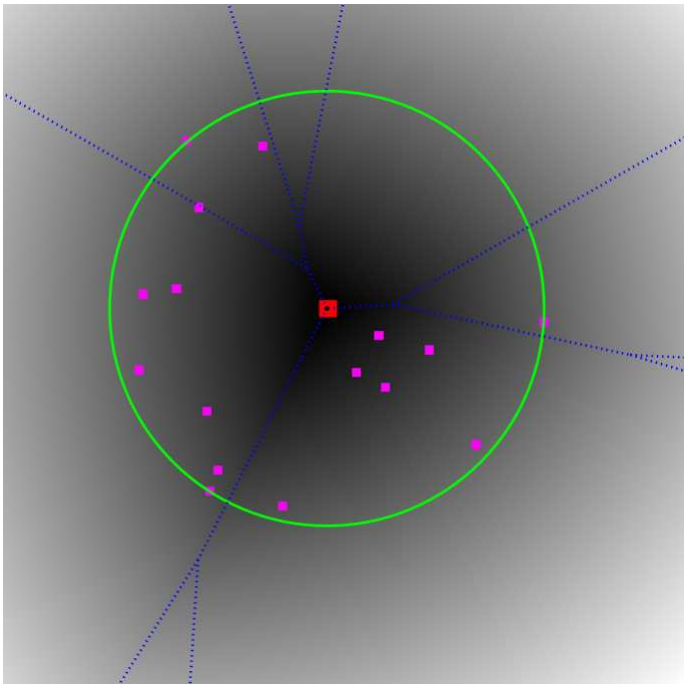

Fig. 2. Discretization of the maximum distance field for a set of sixteen (16) points on the plane. The circumcenter of the smallest enclosing disk minimizes this maximum distance field function and is highlighted using a large square lying on the furthest Voronoi diagram (plotted in dashed style). In this case, the smallest enclosing circle passes through exactly three (3) points whose intersection of furthest Voronoi bisectors yields the circumcenter.

get a $(1+\epsilon)$-approximation in $O(d^3 n \log \frac{1}{\epsilon})$ time[24]. B. Gärtner and S. Schönherr[25] describe a generic quadratic programming (QP) solver tuned up for dense problems with few variables, as it is the case for solving basic instances. The solver *behaves polynomially* but requires arbitrary-precision linear algebra that limits its use to a few hundred dimensions. Recently, another method which turns out to perform so far best in practice, is the second-order cone programming[26] (SOCP). SOCP requires $O(\sqrt{n} \log \frac{1}{\epsilon})$ iterations[27] using interior-point methods. Each iteration can be performed in $O(d^2(n+d))$ time for the smallest enclosing ball. G. Zhou *et al.*[26] present another algorithm, based on providing a smooth approximation of the *nondifferentiable* minimax function $F_{\mathcal{B}}(\cdot)$ using so-called log-exponential *aggregation* functions, that scale well with both terms: dn and $\frac{1}{\epsilon}$. For coarse ϵ values, say $\epsilon \in [0.001, 0.01]$, subgradient steepest-descent methods can be used as it first converges fast before slowly zigzagging towards the optimum. These numerical techniques rely on *off-the-shelves* optimization procedures that have benefited from extensive code optimization along the years but seem not particularly tuned up for the specific smallest enclosing ball problem.

2.3. *Hybrid algorithms*

An ϵ-core set (also written as *core-set*) of $\mathcal{S}$ is a subset $\mathcal{C} \subseteq \mathcal{S}$ such that the smallest enclosing ball of $\mathcal{C}$ expanded by a factor of $1+\epsilon$ fully covers set $\mathcal{S}$. Surprisingly, it was shown by M. Bădoiu *et al.*[28] that for any $\epsilon > 0$ there is a core set of size at most $\frac{1}{\epsilon^2}$, *independent* of dimension d. The bound was later improved to the tight $\frac{1}{\epsilon}$ value[7]. Note that since the smallest enclosing ball is defined by at most $d+1$ points/balls, the result is combinatorically meaningful for $\frac{1}{\epsilon} \leq d+1$. Besides, they also give a simple iterative $O(\frac{dn}{\epsilon^2})$-time algorithm (see procedure *SimpleIterativeBall* that we describe next) to compute a $(1+\epsilon)$-approximation of the smallest enclosing ball, for any $\epsilon > 0$. Combining the ellipsoid numerical approximation method with the combinatorial core-set approach yields a $O(\frac{dn}{\epsilon} + \frac{d}{\epsilon^4})$-time hybrid algorithm. P. Kumar *et al.*[29] relies on the work of G. Zhou et al.[26] to obtain a better $O(\frac{dn}{\epsilon} + \frac{1}{\epsilon^{\frac{9}{2}}} \log \frac{1}{\epsilon})$-time[i] bound. S. Har-Peled mentioned an unpublished[30] $O(\frac{dn}{\epsilon} + \frac{1}{\epsilon^2} \log^2 \frac{1}{\epsilon})$-time algorithm, so that the hybrid algorithm runs in $O(\frac{dn}{\epsilon} + \frac{1}{\epsilon^4} \log^2 n)$-time. Although not explicitly stated in the pioneer work of M. Bădoiu *et al.*[28], the algorithms/bounds are still valid for ball sets (also noticed by P. Kumar et al.[29]).

2.4. *Summary of contributions*

Although combinatorial algorithms are available nowadays for solving the SEB problem of points in very large dimensions (say, $d \simeq 10000$) that prove efficient in practice but lacks deterministic bound (i.e, tight worst-case analysis that guarantees termination), we would like to emphasize on the merits of computing approximate solutions:

- Guaranteed worst-case time dependent on $\frac{1}{\epsilon}$. That is, the less demanding, the faster,
- Very short code: no code is required for computing the basis of at most $d+1$ points/balls,
- No special care are required for handling degeneracies (*i.e.*, cospherical or affinely dependent subsets of points),
- Stable: use predicates of lower degrees (see section 6) for robust computations.

The contributions of our paper are summarized as follows:

- We show an efficient implementation of approximate enclosing balls of core-sets ($d \simeq 15000$ and $\epsilon \simeq 1\%$) based on distance filtering in large dimensions,
- We describe a new tailored core-set algorithm for dual covering/piercing decision problems,
- We propose an alternative effective algorithm for small dimensions,
- We review algorithm performances according to experiments obtained on a common platform.

[i] More precisely, $O(\frac{dn}{\epsilon} + \frac{d^2}{\epsilon^{\frac{3}{2}}}(\frac{1}{\epsilon} + d) \log \frac{1}{\epsilon})$-time.

The next section describes a simple and efficient approach to speed-up point-to-point set distance queries. In the remainder, we use the notations summarized in the following table.

Notations:

S	:	An arbitary point
$\mathcal{S} = \{S_1, ..., S_n\}$	:	A point set
$\text{Ball}(S, r)$	:	A ball with circumcenter S and radius $r \geq 0$
$\mathcal{B} = \{B_1, ..., B_n\}$	:	A ball set with $B_i = \text{Ball}(S_i, r_i)$
$B^* = \text{Ball}(C^*, r^*)$	:	Smallest enclosing ball (SEB) with center C^* and radius r^*
	:	B^* is the SEB of either $\mathcal{S}$ or $\mathcal{B}$, depending on the context
C_i	:	Circumcenter approximation of C^*, at the i^{th} iteration
$\mathcal{C} \subseteq \mathcal{S}$	:	A core-set (a subset of $\mathcal{S}$ or $\mathcal{B}$)
$B_i(r)$	:	A ball centered at $C(B_i) = S_i$ with radius $r - r_i$
K_i	:	Smallest enclosing ball of core-set $\mathcal{C}_i$

3. Farthest Point/Set Distance Queries

SEB algorithms often need to compute the distance, $d_2(P, \mathcal{B})$, from a *query point* P to a given point/ball set $\mathcal{B}$ (with $|\mathcal{B}| = n$). A naive algorithm consists in computing distance pairs $d_2(P, B_i)$ iteratively for $i = 1, ..., n$. This procedure requires $O(dn)$ time per query so that q *farthest queries* $d_2(\cdot, \mathcal{B})$ cost overall $O(qdn)$ time. When dimension d is large, say $d \geq 100$, computing distances of query point/set using the naive technique becomes in itself an expensive operation. We present below a simple yet effective *filtering* technique to *potentially* skip computing a large proportion of explicit distances. Observe that $d_2(X, Y) = ||X - Y|| = \sqrt{\sum_{i=1}^{d}(X_i - Y_i)^2}$ can be written as $||X - Y||^2 = ||X||^2 + ||Y||^2 - 2 < X, Y >$, where $<, >$ denotes the vector inner product (dot product): $< X, Y >= \sum_{i=1}^{d} X_i Y_i = X^T Y$. Using Cauchy-Schwarz inequality, we have $| < X, Y > | \leq ||X||\ ||Y||$. Therefore, the distance of X to Y is upper bounded by

$$\sqrt{||X||^2 + ||Y||^2 + 2\sqrt{||X||^2||Y||^2}} \geq ||X - Y||. \tag{3}$$

Thus when answering q farthest queries, we can first build lookup tables of $||P_i||^2$ ($P_i \in \mathcal{B}$) in a preprocessing stage in $O(dn)$ time and then use a simple distance filtering mechanism. That is, when iteratively seeking for the maximum distance given a query point X and set $\mathcal{B}$, we skip in $O(1)$ time evaluating distance $d_2(X, P_i)$ if the so far maximum distance is above the upper bound given by the Cauchy-Schwarz inequality of Eq. 3. For sets drawn randomly from *statistical distributions*, let $\bar{\alpha}$ be the *expected* number of skipped distances, we answer q queries in $O(dn + q + q(1 - \bar{\alpha})dn)$ time. For uniform d-dimensional cube distributions or

normal distributions we observe experimentally $\bar{\alpha} \stackrel{n}{\rightarrow} 1$ (thus for $n \geq \frac{1}{\epsilon^2}$, the algorithm converges towards optimal linear $O(dn)$ time), for uniform distributions on the d-dimensional sphere, we conversely observe $\bar{\alpha} \stackrel{n}{\rightarrow} 0$. This filtering technique is further described with a C++ code snippet in the book[31] (Chapter 8, pp. 461-464). The methodology extends to ball sets as well but requires extra square-root operations in order to handle ball radii. Interestingly, approximating efficiently the Euclidean distance between *two* given points in *ultra high-dimensional* spaces can be done using logarithmic space[32] only. Sketching these distance computations is a key primitive for data stream algorithms processing gigantic datasets not fitting local memory (see section 9).

4. Approximating Smallest Enclosing Balls of Core-Sets

Although M. Bădoiu and K. Clarkson's algorithm[7] (procedure Bădoiu -Clarkson below) extends to ball sets as well, for ease of description, we consider here point sets $\mathcal{S} = \{S_1, ..., S_n\}$. Given a current circumcenter C, the procedure finds a farthest point S_m of $\mathcal{B}$ and moves C towards S_m in $O(dn)$ time per iteration, thus bypassing the costly $O(d^2n)$ time Jacobian matrix computation necessarily required in a typical steepest-descent numerical optimization scheme. The overall cost is $O(\frac{dn}{\epsilon^2})$ time as we need to perform $\lfloor \frac{1}{\epsilon^2} \rfloor$ iterations to ensure a $(1+\epsilon)$-approximation algorithm.

```
Bădoiu -Clarkson(S, ε);
◁ Compute a (1+ε)-approximation of the smallest enclosing ball ▷
◁ Return the circumcenter of a small enclosing ball in O(dn/ε²) time ▷
C = S_1 ;
for i = 1 to ⌈1/ε²⌉ do
    ◁ The core-set is the collection of furthest points ▷
    ◁ Furthest point is F_i = S_j ▷
    j = argmax_{i=1}^{n} ||CS_i||;
    C = C + 1/(i+1) CS_j;
return C;
```

Using this simple and elegant algorithm (renamed as procedure SimpleIterativeBall below) and coupling it with approximations of smallest enclosing balls of core-sets (see paper[28]), we obtain a $O(\frac{dn}{\epsilon} + \frac{d}{\epsilon^4})$-time algorithm (procedure *ApproximateCoreSet*). For $\frac{1}{\epsilon} = O(\sqrt[3]{n})$, the bottleneck of the algorithm is finding the core-set itself rather than the overall cost of simple loops.

Lemma 1 (Bădoiu et al. [28]). *The approximation algorithm reported in procedure ApproximateCoreSet*$(\mathcal{S}, \epsilon)$ *delivers a* $(1+\epsilon)$*-approximation of the smallest enclosing ball of point set* $\mathcal{S}$ *(for* $0 < \epsilon < 1$*).*

The proof is found in Bădoiu et al. [28] (Lemma 2.3 and Theorem 2.5), pages 3

and 4. Further, additional breakthrough results on k-center and k-median clustering problems with or without outliers are also reported. The smallest enclosing ball problem is a particular case, namely the 1-center problem.

SimpleIterativeBall$(\mathcal{S}, \epsilon)$;
◁ $\mathcal{S} = \{S_1, ..., S_n\}$ is a point set ▷
◁ The algorithm generalizes straightforwardly to arbitrary radii ball sets $\mathcal{B}$ ▷
Pick arbitrary $C_1 \in \mathcal{S}$;
$i \leftarrow 1$;
$a = \lfloor \frac{1}{\epsilon^2} \rfloor$;
while $i \leq a$ **do**
    $m = \text{argmax}_j ||C_i S_j||$;
    ◁ Distance filtering ▷
    $C_{i+1} = C_i + \frac{1}{i+1}(S_m - C_i)$;
    $i \leftarrow i + 1$;
$r_a = d_2(C_a, \mathcal{S})$;
return Ball(C_a, r_a);

ApproximateCoreSet$(\mathcal{S}, \epsilon)$;
◁ $\mathcal{S} = \{S_1, ..., S_n\}$ is a point set ▷
◁ The $\mathcal{C}_i$'s are the collection of greedy core-sets ▷
◁ Overall time complexity is $O(\frac{dn}{\epsilon} + \frac{d}{\epsilon^4})$ ▷
$\gamma = \frac{\epsilon}{3}$;
$\delta = \frac{\epsilon}{3}$;
◁ Guarantee $(1 + \delta)(1 + \gamma) \leq 1 + \epsilon$ for any $\epsilon \leq 1$▷
$\mathcal{C}_1 \leftarrow \{B_1\}$;
$r_1 = 0$;
$i \leftarrow 1$;
while $d_2(C_i, \mathcal{B}) \geq (1 + \delta) r_i$ **do**
    ◁ Distance filtering ▷
    $k = \text{argmax}_i \; d_2(C_i, \mathcal{B})$;
    $\mathcal{C}_{i+1} \leftarrow \mathcal{C}_i \cup \{B_k\}$;
    $K_{i+1} \leftarrow SimpleIterativeBall(\mathcal{C}_{i+1}, \gamma)$;
    $C_{i+1} \leftarrow C(K_{i+1})$;
    $r_{i+1} \leftarrow r(K_{i+1})$;
    $i \leftarrow i + 1$;
return Ball(C_i, r_i);

Plugging the distance filtering mechanism of section 3, for uniform distribution of ball sets with $d \simeq 10000, n = d + 1, \epsilon \simeq 0.01$, the algorithm requires experimen-

tally a mere few seconds on current commodity PCs for a short 30-line C code.[j] It performs better in practice than the steepest-descent method. The algorithm is adaptive according to the core-set size, bounded by $\frac{6}{\epsilon}$, but not in the iteration process of Bădoiu and Clarkson[7] as we need to loop exactly $\lfloor \frac{9}{\epsilon^2} \rfloor$ time.[k] Theoretically speaking, this algorithm is only slightly outperformed by a SOCP solver, but its extreme simplicity coupled with the distance filtering trick make it attractive for machine learning applications as discussed in section 8. The filtering technique also applies to the optimal core-set algorithm of Panigrahy[8] that computes a $(1+\epsilon)$-approximation of the SEB in $\Theta(\frac{dn}{\epsilon})$ time using a two-level procedure that guesses stepwise the range of the optimal radius r^* (procedure *PanigrahyCoreSet* below). Panigrahy's approximation SEB algorithm successively increments the ball radius starting from an initial lower bound radius guess r of the optimal radius r^*. The heuristic always guarantees that $r \leq r^* \leq r + \delta$ and therefore terminates as soon as $\delta \leq \epsilon$, in at most $O(\frac{1}{\epsilon})$ iterations (see Figure 3). Core-sets have proven to be an important concept in many geometric approximate optimization tasks. We refer to the survey[30] of P. Agarwal et al. to see how geometric optimization problems have been revisited under the auspices of core-sets.

The core-set algorithms generalize to ball sets easily as follows: For a current circumcenter position C, we seek the farthest point F in the ball set $\mathcal{B} = \{B_1 = \text{Ball}(S_1, r_1), ..., B_n = \text{Ball}(S_n, r_n)\}$ by maximizing $\max_{i\in\{1,...,n\}} d_2(C, B_i)$. Since $\max_{X_i \in B_i} d_2(C, X_i) = d_2(C, S_i) + r_i$, we deduce that $\max_{i\in\{1,...,n\}} d_2(C, B_i) = \max_{i\in\{1,...,n\}}(d_2(C, S_i) + r_i)$, and that $F = (1 + \frac{r_f}{d_f})\overrightarrow{CS_f}$, where $f = \arg\max \max_{i\in\{1,...,n\}} d_2(C, F_i)$.

5. Core-sets for decision problems

Let $B^* = \text{Ball}(C^*, r^*) = \text{SEB}(\mathcal{B})$ denote the smallest enclosing ball of a ball set $\mathcal{B} = \{B_1, ..., B_n\}$ with $B_i = \text{Ball}(S_i, r_i)$ for all $1 \leq i \leq n$ (see Figure 1 for such an example). Our novel approximation algorithms proceed by solving *dual* piercing/covering *decision problems* as depicted in Figure 4 for finding dichotomically a $(1+\epsilon)$-approximation of the SEB by testing whether $r \geq r^*$ or not (within a tolerance precision factor ϵ). The dual piercing problem is related to a geometric covering/duality property of balls that we explain for the general case of balls with arbitrary different radii. (Point sets are considered as degenerated ball sets with all radii set to zero.) The covering/piercing duality relationship is stated as follows:
$\mathcal{B}$ can be covered by a ball of radius r *if and only if* $\cap\mathcal{B}(r) = \cap_{i\in\{1,...,n\}} B_i(r) \neq \emptyset$, where $B_i(r) = \text{Ball}(S_i, r - r_i)$ for $1 \leq i \leq n$. It follows a lower bound on r: $r \geq \max_i r_i \geq 0$. Note that we can also subtract to both the query r and ball radii the minimum radius $\min_i r_i$ without changing the equivalence covering↔piercing

[j]The code is available online from the book[31] at `http://www.charlesriver.com/visualcomputing/programs/SmallEnclosingBall.cpp` There is also a Java™ applet at `http://www.sonycsl.co.jp/person/nielsen/BregmanBall/BBC/`

[k]It is of practical interest to find a better stopping adaptive criterion.

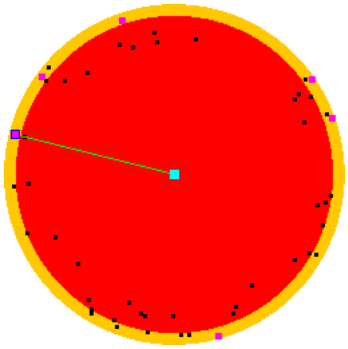

Fig. 3. Illustration of Panigrahy's core-set algorithm on a point set randomly distributed in a small-width annulus. Approximate Ball(C, r) and enclosing Ball$(c, r + \delta)$ balls are displayed respectively in red and orange. The approximate center and core-set are drawn in blue and magenta, respectively. The farthest point from the current center position is indicated by the green segment.

```
1  PanigrahyCoreSet(S, ε);
2  ◁ S = {S_1, ..., S_n} is the input set of points ▷
3  Pick arbitrary C_1 ∈ S;
4  r = (1/2) max_i d_2(C_1, S);
5  δ = (1/2) max_i d_2(C_1, S);
6  repeat
7      for O(1/δ) iterations do
8          m = argmax_i d_2(C, S);
9          Move Ball(C, r) until it touches S_m;
10     s = max_i d_2(C, S_i) − r;
11     if s ≤ 3δ/4 then
12         δ = 3δ/4
13     else
14         r = r + δ/4;
15         δ = 3δ/4;
16 until δ ≤ ε ;
```

property. We prove the following stronger property for Euclidean balls by showing that for $r \geq r^*$ there exists a unique ball of radius $r - r^*$ fully contained into the common intersection:

Lemma 2. *For $r \geq r^*$, there exists a ball B of radius $r(B) = r - r^*$ centered at $C(B) = C^*$ fully contained inside the intersection $\cap\mathcal{B}(r)$.*

Proof. In order to ensure that C^* is in each $B_i(r)$, a sufficient condition is to have

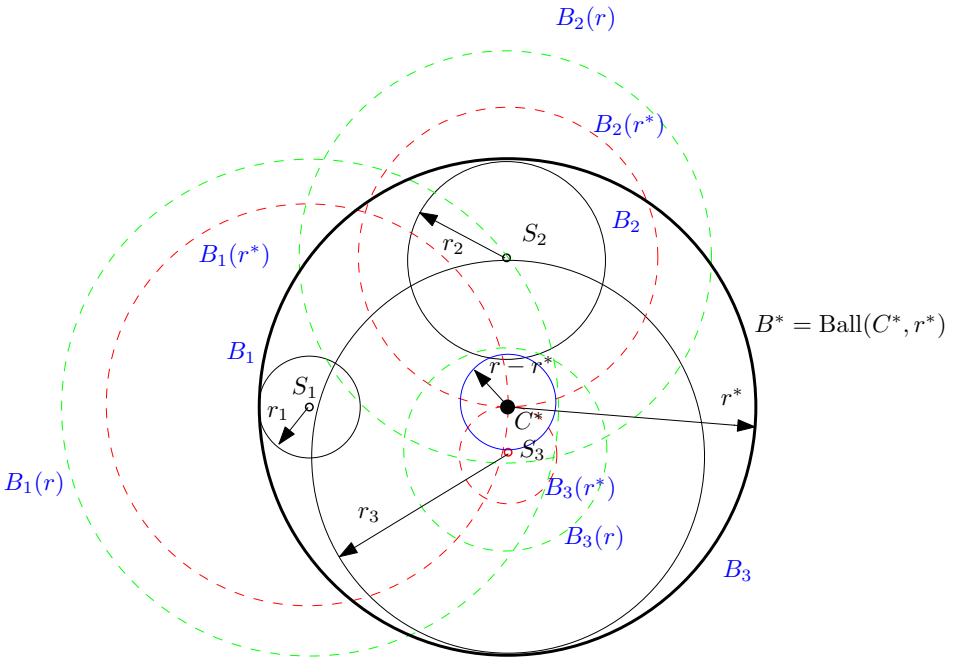

Fig. 4. Geometric covering/piercing duality. Given a prescribed $r \geq \max_i r_i$ value, balls B_1, B_2, B_3 are associated to corresponding dashed balls $B_1(r), B_2(r), B_3(r)$ such that $C(B_i(r)) = S_i$ and $r(B_i(r)) = r - r_i$ for $i \in \{1, 2, 3\}$. We have $B_1(r^*) \cap B_2(r^*) \cap B_3(r^*) = \{C^*\}$. For $r \geq r^*$, there exists a unique ball of radius $r - r^*$ fully contained in (i.e., piercing) the intersection $B_1(r) \cap B_2(r) \cap B_3(r)$.

$r \geq \max_i\{r_i + d_2(S_i, C^*)\}$. Since $B_i \subseteq \text{Ball}(C^*, r^*), \forall i \in \{1, 2, ..., n\}$, we have

$$\max_i\{r_i + d_2(S_i, C^*)\} \leq r^*. \tag{4}$$

Thus, provided $r \geq r^*$, we have $C^* \in \cap\mathcal{B}(r)$. Now, notice that for all $i \in \{1, 2, ..., n\}$ and for any $0 \leq r' \leq (r - r_i) - d_2(S_i, C^*)$, we have the property that $\text{Ball}(C^*, r') \subseteq B_i(r)$. Thus, if we ensure that $r' \leq r - \max_i(r_i + d_2(S_i, C^*))$, then $\text{Ball}(C^*, r') \subseteq \cap\mathcal{B}(r)$. From Ineq. 4, we choose $r' = r - r^*$ and obtain the lemma (see Figure 4). For $r = r^*$, the common intersection $\cap\mathcal{B}(r)$ reduces to a single point, namely the circumcenter of the SEB: $C^* = \cap\mathcal{B}(r^*)$. □

For practical algorithmic considerations to be detailed in the complexity analysis later on, we *relax* this *1-piercing* point problem (a typical 0-transversal problem in combinatorial geometry) to that of a common *piercing* ϵr^**-ball* (*i.e.*, a ball of radius ϵr^*): Namely, report whether there exists a ball $B = \text{Ball}(C, \epsilon r^*)$ such that $B \subseteq \cap\mathcal{B}(r)$ or not (see Figure 4).

The algorithm for answering whether such a ball set $\mathcal{B}$ can be covered by a ball of radius r or not within *tolerance factor* ϵr^* (ie., the corresponding set of

balls $B_i(r)$ with radii $r - r_i$ can be pierced by an ϵr^* ball) is reported in details in procedure *DecisionProblem*. The algorithm builds a core-set (sets $\mathcal{C}_i$'s) iteratively for the decision problem by *narrowing* the feasible domain for circumcenter C^*. It is a primal-dual method since solving the dual piercing problem requires to solve for primal smallest enclosing balls of balls (a covering problem). The algorithm proceeds by considering the set of 1D intervals obtained as the intersection of the input balls with the vertical line passing through the circumcenter of the smallest enclosing ball of the core-set at the current iteration. If that set of intervals is complete (i.e., meaning all balls intersect the line) and admits a common piercing point[l] then obviously this piercing point also stabs the set of d-dimensional balls. Otherwise, there are two cases: Either there is a single ball that is not intersected by the current vertical line, or we know from Helly's theorem[m] that there exists two disjoint intervals that do not intersect. We add the at most two balls to the current core-set, update the circumcenter of the SEB of the core-set, check whether the radius of the core-set SEB is less than r (otherwise we can already conclude that $r^* > r$) and repeat the process until we find a piercing point or the absolute difference of the core-set radius with the query radius falls below the prescrived threshold ϵ. The algorithm is summarized in procedure $DecisionProblem(\mathcal{B}, r, \epsilon)$.

Let k denote the maximum number of iterations of the while loop. Clearly, we have the size of the core-set at the i-th iteration that is upper bounded by $2i$: $|\mathcal{C}_i| \leq 2i$. Moreover, since $\cap\mathcal{C}_{i+1} \subset \cap\mathcal{C}_i$ and because the smallest enclosing ball is unique, we have $r_{i+1} > r_i$.

Lemma 3. *For any ball B already chosen in some core-set $\mathcal{C}_i$, the ball is necessarily pierced (i.e., contained) by circumcenter points $C(K_j)$ of core-sets, for $j \geq i + 1$.*

Proof. Since $C(K_i)$ is the center of the smallest enclosing ball of the center points of balls of radius r of $\mathcal{C}_i$, and $r_i = r(K_i) \leq r$, we have $d_2(C(K_i), C(B)) \leq r$ for all $B \in \mathcal{C}_i$. □

Because the algorithm is greedy and add one or two new balls in the core-set at each iteration, it produces always terminate and produce the correct result. We now bound the *maximum number* of iterations of this primal-dual decision problem.

Theorem 1. *Procedure DecisionProblem correctly reports whether a set of ball can be pierced by an ϵr^*-ball or not. The number of iterations, k, required by algorithm DecisionProblem is a function depending only on d and ϵ, and is independent of n. We have $k = O(\frac{1}{\epsilon})^d$.*

Proof. Let $v_d(r)$ denote the volume of a d-dimensional ball of radius r, and $\mathrm{vol}(\mathcal{A})$

[l]A simple task requiring to sort the $2n$ interval extremities and check that the maximum left interval bracket is below the minimum right interval bracket. This requires $O(n \log n)$ time.
[m]In convex geometry, Helly's theorem states that a set of convex objects has a common intersection if and only if every subset of $d + 1$ members do.

Algorithm: $DecisionProblem(\mathcal{B}, r, \epsilon)$

◁ Solve test $r \geq r^*$ or not (with precision ϵ) using ϵr^*-ball piercing decision problem ▷
◁ $\mathcal{B} = \{B_1, ..., B_n\}$ is the ball set with $B_i = \text{Ball}(S_i, r_i)$ and $B_i(r) = \text{Ball}(S_i, r - r_i)$ ▷
◁ $0 < \epsilon < 1$ is the approximation factor ▷
Let $rr \leq 2r^*$ be the radius obtained from the naïve 2-approx. algorithm;
Choose arbitrary $B_1 \in \mathcal{B}$;
$\mathcal{C}_1 \leftarrow \{B_1\}$;
$r_1 \leftarrow r(B_1)$;
$i \leftarrow 1$;
while $r - r_i \geq \epsilon \frac{rr}{2}$ **do**
  ◁ Consider intersections with the "vertical" line passing through S_i ▷
  ◁ x_d denote the unit vector of the d-th coordinate axis ▷
  Let $L_i : S_i + \lambda x_d$;
  ◁ $\mathcal{B}_{L_i}$ denote the 1D intervals of the subset of balls intersecting line L_i ▷
  $\mathcal{B}_{L_i}(r) = \{B(r) \cap L_i \mid B \in \mathcal{B}\}$;
  if $\cap \mathcal{B}_{L_i}(r) \neq \emptyset$ **then**
    ◁ We found a piercing point, i.e. $r \geq r^*$ ▷
    return **Yes**
  else
    if $\exists B(r) | B(r) \cap L_i = \emptyset$ **then**
      ◁ Add a single ball to the core-set ▷
      $\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{B\}$;
    else
      ◁ Add two balls to the core-set ▷
      ◁ At most two cases to consider from Helly theorem on 1D intervals ▷
      Let $B_k(r)$ and $B_l(r)$ such that $(B_k(r) \cap L_i) \cap (B_l(r) \cap L_i) = \emptyset$;
      ◁ Increase core-set greedily ▷
      $\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{B_k, B_l\}$;
  ◁ Primal-dual piercing/covering method on core-sets ▷
  $i \leftarrow i + 1$;
  $K_i = \text{SEB}(\mathcal{C}_i)$;
  if $r(K_i) > r$ **then**
    ◁ We have obviously $r^* > r$ ▷
    return **No**
  $S_i = C(K_i)$;
◁ We have $r - r^* \leq \epsilon r^*$ ▷
◁ We stop looking for a piercing point (radius falls within precision range) ▷
◁ Otherwise, we may iterate too many times ▷
return **MayBe** ;

the volume of an object $\mathcal{A} \in \mathbb{E}^d$. Clearly, we have $\cap\mathcal{C}_{i+1} \subset \cap\mathcal{C}_i$ for all $i \geq 1$. Let K_i be the unique maximal ball contained in $\cap\mathcal{C}_i$ (obtained from the smallest enclosing ball of the centers of balls contained in $\mathcal{C}_i$). If $C(K_i)$, the center of ball K_i, does not fully pierce $\mathcal{B}$, then there exists either one ball M_i or two balls M_i and N_i such that their intersection A_i (either $A_i = M_i$ or $A_i = M_i \cap N_i$) does not contain $C(K_i)$. Since A_i is convex, this means that there exists an hyperplane H_i separating A_i from $C(K_i)$. Let H_i' be an hyperplane parallel to H_i and passing through $C(K_i)$, $H_i'^+$ be the halfspace not containing A_i. Since $\cap\mathcal{C}_{i+1} \subset \cap\mathcal{C}_i$, we have $\text{vol}(\mathcal{C}_{i+1}) \leq \text{vol}(\mathcal{C}_i) - \frac{1}{2}v_d(r(K_i))$. Since $r(K_i) \geq \epsilon r^*$ and $\text{vol}(C_1) \leq v_d(2r^*)$, we get a sloppy upperbound $k = O(\frac{1}{\epsilon})^d$ (equivalently, $k = O(1)^d\frac{1}{\epsilon^d}$). In a good scenario, where we split in half the volume of $\cap\mathcal{C}_i$, we get $k = O(d\log_2\frac{1}{\epsilon})$. □

By using these relaxed decision problems, we build an effective dichotomic algorithm for finding a $(1+\epsilon)$-approximation of the SEB (section 6 describes this stage). Overall, this yields an $O(d^2 n\log_2\frac{1}{\epsilon}) + O_{d,\epsilon}(1)$ time[n] algorithm (improve by a factor $O(d)$ over the ellipsoid method). We observe *experimentally* that k tends indeed to behave as in the good scenario (i.e., $O_d(\log\frac{1}{\epsilon})$) and that the core-set sizes are similar to the ones obtained by M. Bădoiu and K. Clarkson's algorithm. By solving $O(\log\frac{1}{\epsilon})$ decision problems, we thus obtain a $(1+\epsilon)$-approximation of the smallest enclosing ball.

6. Small dimensions revisited

For small dimensions, we propose another efficient approach for solving planar decision problems that relies on the same covering/piercing geometry property of Lemma 2. We consider without loss of generality the planar case. Let $x(P) = x_1(P) = x_P$ denote the x-abscissa of a 2D point $P = (x_P, y_P)$. Let $[n] = \{1, ..., n\}$ and $[x_m, x_M]$ be an interval on the x-axis where an ϵr^*-disk center might be located if it exists. (That is $x(C) \in [x_m, x_M]$ if it exists.) We initialize x_m, x_M as the x-abscissae extrema: $x_m = \max_{i\in[n]}(x_i) - r,\ x_M = \min_{i\in[n]}(x_i) + r$. If $x_M < x_m$ then clearly vertical line $L : x = \frac{x_m+x_M}{2}$ separates two extremum disks (those whose corresponding centers give rise to x_m and x_M) and therefore $\mathcal{B}(r)$ is not 1-pierceable (therefore not ϵr^*-ball pierceable). Otherwise, the algorithm proceeds by dichotomy. Let $e = \frac{x_m+x_M}{2}$ and let L denotes the vertical line $L : x = e$. Denote by $\mathcal{B}_L = \{B_i \cap L | i \in [n]\}$ the set of n y-intervals obtained as the intersection of the disks of $\mathcal{B}$ with line L (see Figure 5). We check whether $\mathcal{B}_L = \{B_i \cap L = [a_i, b_i] | i \in [n]\}$ is 1-pierceable or not. Since $\mathcal{B}_L$ is a set of n y-intervals, we just need to check whether $\min_{i\in[n]} b_i \geq \max_{i\in[n]} a_i$ or not. If $\cap\mathcal{B}_L \neq \emptyset$, then we have found a point $(e, \min_{i\in[n]} b_i)$ in the intersection of all balls of $\mathcal{B}$ and we stop recursing. (In fact we found a $(x = e, y = [y_m = \max_i a_i, y_M = \min_i b_i])$ vertical piercing segment.) Otherwise, we have $\cap\mathcal{B}_L = \emptyset$ and need to choose on which side of L to recurse. W.l.o.g.,

[n]The $O_{d,\epsilon}(1)$ term denotes a function of d and ϵ that is independent of n.

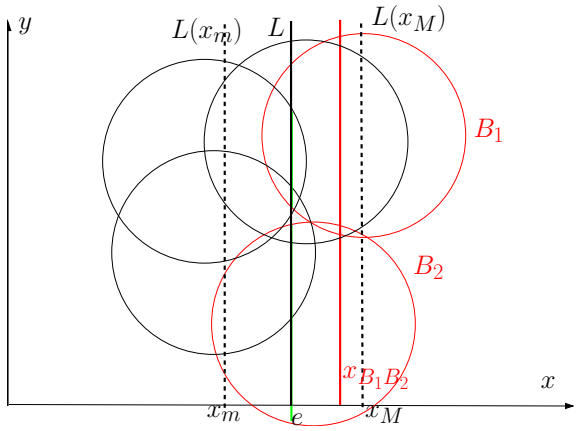

Fig. 5. A recursion step: Median line $L : x = e$ intersecs all disks. However, two y-intervals do not intersect on L. Recursion is performed on x-range $[e, x_M]$.

let B_1 and B_2 denote the two disks whose corresponding y-intervals on L are disjoint. We choose to recurse on the side where $B_1 \cap B_2$ is located (if the intersection is empty then we stop by reporting the two non intersecting balls B_1 and B_2). Otherwise, $B_1 \cap B_2 \neq \emptyset$ and we branch on the side where $x_{B_1B_2} = \frac{x(C(B_1))+x(C(B_2))}{2}$ lies in.

At each stage of the dichotomic process, we halve the x-axis range where the solution is to be located (if it exists). We stop the recursion as soon as $x_M - x_m < \epsilon\frac{r}{2}$. Indeed, if $x_M - x_m < \epsilon\frac{r}{2}$ then we know that *no center of a ball* of radius ϵr is contained in $\cap\mathcal{B}$. (Indeed if such a ball exists then *both* $\cap\mathcal{B}_{L(x_m)} \neq \emptyset$ and $\cap\mathcal{B}_{L(x_M)} \neq \emptyset$.) Overall, we recurse at most $3 + \lceil \log_2 \frac{1}{\epsilon} \rceil$ times since the initial interval width $x_M - x_m$ is less than $2r^*$ and we consider $r \geq \frac{r^*}{2}$.

Thus, by solving $O(\log_2 \frac{1}{\epsilon})$ decision problems (dichotomy search), we obtain a $O(n \log_2^2 \frac{1}{\epsilon})$-time deterministic $(1+\epsilon)$-approximation algorithm. We bootstrap this algorithm in order to get a $O(n \log_2 \frac{1}{\epsilon})$-time algorithm. The key idea is to shrink potential range $[a, b]$ of r^* by selecting iteratively different approximation ratios ϵ_i until we ensure that, at kth stage, $\epsilon_k \leq \epsilon$. Let $\mathrm{Ball}(C, r)$ be a $(1+\epsilon)$-approximation enclosing ball. Observe that $|x(C) - x(C^*)| \leq \epsilon r^*$. We update the x-range $[x_m, x_M]$ according to the current piercing point abcissae $x(C)$ and current approximation factor. We start by solving the approximation of the smallest enclosing ball for $\epsilon_1 = \frac{1}{2}$. It costs $O(n \log_2 \frac{1}{\epsilon_1}) = O(n)$. Using the final output range $[a, b]$, we now have $b - a \leq \epsilon_1 r^*$. Consider $\epsilon_2 = \frac{\epsilon_1}{2}$ and reiterate until $\epsilon_l \leq \epsilon$. The overall cost of the procedure is $\sum_{i=0}^{\lceil \log_2 \frac{1}{\epsilon} \rceil} O(n \log_2 2) = O(n \log_2 \frac{1}{\epsilon})$. This dichotomic approximation algorithm is summarized in the pseudo-code procedure *DichotomicApproximationSED* below. Note that the method extends to disks as

Method/Distribution	□ max	⊙ max	□ avg	⊙ avg
D. E. Eberly ($\epsilon = 10^{-5}$)	0.7056	0.6374	0.1955	0.2767
J. Ritter ($\epsilon \leq 0.5$)	0.0070	0.0069	0.0049	0.0049
2nd Method ($\epsilon = 10^{-2}$)	0.0343	0.0338	0.0205	0.0286
2nd Method ($\epsilon = 10^{-3}$)	0.0515	0.0444	0.0284	0.0405
2nd Method ($\epsilon = 10^{-5}$)	0.0719	0.0726	0.0473	0.0527

Table 1. Benchmarking our approximation algorithms with respect to J. Ritter [21] and D. E. Eberly's implementation of MiniBall [5] for point sets drawn either from the uniform square distribution (□) or inside a small-width annulus (⊙).

well. We report on timings obtained from experiments[o] done on 1000 trials for uniformly distributed 100000-point sets in a unit ring of width 2ϵ (⊙) or unit square (□). See Table 1. Maximum (max.) and average (avg.) running times are in fractions of a second obtained by a 30-line C code on an Intel 1.6 GHz processor. See the public code of D. E. Eberly[p] for a randomized implementation.

7. Predicate degree

Predicates are the basic computational atoms of algorithms that are related to their numerical stabilities. D. E. Eberly uses the *InCircle* containment predicate of algebraic degree 4 on integers ($d + 2$ in dimension d for integer arithmetic. The degree drops to 2 if we consider rational arithmetic[11]). We show how to replace the predicates of algebraic degree 4 by predicates of degree 2 for integers: "Given a disk center (x_i, y_i) and a radius r_i, determine whether a point (x, y) is inside, on or outside the disk". It boils down to compute the sign of $(x - x_i)^2 + (y - y_i)^2 - r_i^2$. This can be achieved using another dichotomy search on line $L : x = l$. We need to ensure that if $y_m > y_M$, then there do exist two disjoint disks B_m and B_M. We regularly sample line L such that if $y_m > y_M$, then there exists a sampling point in $[y_M, y_m]$ that does not belong to both disks B_m and B_M. In order to guarantee that setting, we need to ensure some *fatness* of the intersection of $\cap\mathcal{B}(r) \cap L$ by recursing on the x-axis until we have $x_M - x_m \leq \frac{\epsilon}{\sqrt{2}}$. In that case, we know that if there was a common ϵr^*-ball intersection, then its center x-coordinate is inside $[x_m, x_M]$: this means that on L, the width of the intersection is at least $\frac{\epsilon}{\sqrt{2}}$. Therefore, a regular sampling on vertical line L with step width $\frac{\epsilon}{\sqrt{2}}$ guarantees to find a common piercing point if it exists. A straightforward implementation would yield a time complexity $O(\frac{n}{\epsilon} \log_2 \frac{1}{\epsilon})$. However, it is sufficient for each of the n disks, to find the upper most and bottom most lattice point in $O(\log_2 \frac{1}{\epsilon})$-time using the floor function. Using the bootstrapping method, we obtain a $O(n \log_2 \frac{1}{\epsilon})$ time using integer arithmetic with algebraic predicates InCircle of degree 2. In dimension 3 and higher, the dimension

[o]Source code is available at `http://www.sonycsl.co.jp/person/nielsen/PT/seb/sebdisk.html`
[p]`http://www.magic-software.com`

```
1  DichotomicApproximationSED(S = {(x_i, y_i)}_i, ε);
2  xmin = min_{i∈{1,...,n}} x_i; xmax = max_{i∈{1,...,n}} x_i;
3  d_1 = max_{i∈{1,...,n}} ||S_i − S_1||;
4  b = d_1; a = d_1/2; ε ← (1/4)(b − a)ε;
5  pierceable = false; qdisjoint = false;
6  while b − a > ε do
7      r = (a+b)/2;
8      x_M = xmin + r;
9      x_m = xmax − r;
10     pierceable = false;
11     while x_M − x_m ≥ ε and ¬pierceable and ¬qdisjoint do
12         l = (x_M + x_m)/2;
13         y_m = max_{i∈{1,...,n}} y_i − sqrt(r^2 − (l − x_i)^2);
14         m = argmax_{i∈{1,...,n}} y_i − sqrt(r^2 − (l − x_i)^2);
15         y_M = min_{i∈{1,...,n}} y_i + sqrt(r^2 − (l − x_i)^2);
16         M = argmin_{i∈{1,...,n}} y_i + sqrt(r^2 − (l − x_i)^2);
17         if y_M ≥ y_m then
18             x = l;
19             y = (y_m + y_M)/2;
20             pierceable = true;
           else
21             // m and M are arg indices of y_m and y_M;
22             if ||S_m − S_M|| > 2(r − ε) then
                   qdisjoint = true;
               else
23                 if (x_m + x_M)/2 > l then
24                     x_m = l;
                   else
25                     x_M = l;
26     if pierceable then
27         b = r;
       else
28         if qdisjoint then
29             a = ||S_m − S_M||/2 + ε ;
           else
30             a = r;
```

reduction algorithm extends with a running time $O_d(n \log_2 \frac{1}{\epsilon})$. As a side-effect, we improve the result of D. Avis and M. E. Houle[33] for the following problem: Given a set $\mathcal{B}$ of n d-dimensional balls of $\mathbb{E}^d$, we can find whether $\cap\mathcal{B} = \emptyset$ or report a common intersection point in $\cap\mathcal{B}$ in deterministic $O_d(n^d \log n)$ time and $O_d(n^d)$

space.

8. Applications of SEBs in Machine Learning

The computational machine learning community investigates algorithms that improve their performances by their *experience*. Classification generally follows three paradigms: it is *unsupervised* when the goal is to discover the structure of data sets, it is *supervised* when the goal is to infer a link between this structure and the prediction of a class, and it is *partially supervised* when both aspects are mixed (equivalently, class information is available only for a subset of the data). We refer to the textbook of T. Hastie et al.[34] for an overview of the statistical approach[q] of machine learning. Statistical learning is the branch of machine learning (mostly classification) that burgeons in statistical issues, such as consistency and generalization.

We describe concisely below how the use of approximate SEB algorithms was recently used to extend a popular classification technique, called *support vector machines*[35,36] (SVMs), for state-of-the-art performance in real-world applications. It is actually not surprising to find applications of computational geometry in machine learning as data sets can be geometrically interpreted as point sets lying in some information space and classifiers visualized as geometric separators.

8.1. *From support vector machines to ball vector machines*

The basic underlying principle of SVMs for 2-class classification task was first reported by V. Vapnik and A. Lerner in their 1963 paper.[37] Let $\mathcal{D} = \{(x_1, y_1), ..., (x_n, y_n)\}$ be a *training set* of data, where $x_i \in \mathbb{R}^d$ denotes the d-dimensional feature vector and $y_i \in \{-1, +1\}$ its boolean[r] class. Suppose the two classes $+1$ and -1 are linearly separable. This means that there exists a hyperplane $H :< W, X > +b = 0$ that splits the training vectors so that all the $+1$ features are located, say, on the upper half-space $H^+ :< W, X > +b > 0$ (and all -1 features are in $H^- :< W, X > +b < 0$). The (linear) support vector machine simply choose the *unique* separating hyperplane H^* that maximizes the *margin*, i.e. the minimum distance of $+1/-1$ features to the hyperplane. The optimal separating hyperplane H^* and the bounding feature vectors are found by solving a simple quadratic minimization program[38]: $\min_W \frac{1}{2} < W, W >$ (provided the hyperplane equation is written in the usual *canonical* way[36]). Since most data sets in practice are unfortunately not linearly separable, we need do adapt the linear SVM technique by introducing a trade-off between correctly and incorrectly classified vectors. Typically, a penalty function is introduced in the quadratic program to determine the *soft margin*. Soft margin linear SVM yet does not allow to consider 2-class sets

[q]Statistical learning assumes that data sets are randomly drawn from *any kind* but *fixed* statistical distributions.

[r]Extension to multi-class[34,36] is easy from the 2-class setting.

of concentric data. This was tackled later on, in 1992, by B. Boser et al.[35] by introducing kernel SVM. The *kernel* "trick" that works for any non-linearly separable data sets consists in mapping the feature vectors x_i into higher-dimensional vectors $k(x_i)$ so that the *lifted* training data becomes linearly separable. It is always possible to find such a kernel $k : \mathbb{R}^d \to \mathbb{R}^D$ by augmenting the dimension D. The beauty of kernel SVMs is that we do not manipulate explicitly higher-dimensional feature vectors but rather uses some inner product property induced by the kernel to find the optimal separating hyperplane of $\mathbb{R}^D$ that yields a (non-linear) separator in $\mathbb{R}^d$ with potentially several connected components. Kernel SVMs have obtained outstanding practical performance and is acclaimed as one of the crowns of machine learning. The computational bottleneck of (kernel) SVM is the quadratic programming (QP) optimization stage. A naive QP solver requires $O(n^3)$ time using $O(n^2)$ space. Tsang et al.[39] proposed in 2005 the Core Vector Machine (CVM) which combines the principle of SVM training with the SEB and allows one thus to scale up kernel methods. Informally speaking, the QP solver in SVM is replaced by a SEB problem for which an efficient $(1+\epsilon)$-approximation is obtained (see section 4) yielding to close-to-optimal solution. Note that hyperplanes can be interpreted as infinite-radius balls, and thus CVMs nicely generalizes the principle of geometric containment classifier. An important trick of CVM is to map the SVM problem into a SEB problem in which the class information is melted into the structure of data. Although the CVMs have experimentally proved efficient for classification, regression and semi-supervised learning, it was further made simple by Tsang et al.[40] in their recent paper on Ball Vector Machines (BVMs) by considering fixed radius balls in the core-set optimization process (see Panigrahy's core-set[8] algorithm described in section 4).

8.2. *From Euclidean balls to information-theoretic balls*

Feature vectors F in machine learning are conveniently interpreted as geometric points in an underlying feature space $\mathbb{F}$. In practice, real-world data sets to process are highly heterogeneous. That is, feature vectors are assembled from homogeneous vectors by stacking successively their components on top of each other. Thus feature space $\mathbb{F}$ is rather seen as a *heterogeneous* feature space obtained as the Cartesian product of k elementary spaces: $\mathbb{F} = \mathbb{F}_1 \times ... \times \mathbb{F}_k$. Homogeneous spaces can either be quantitative (values obtained from measurements), qualitative (descriptive tags) or ordered categorical (such as small<medium<large). Therefore the distance between any two feature points is likely not to be the usual Euclidean distance, and special care need to be made for defining appropriately the equivalent meaning of "maximum" margin or balls. This is all the more important as distances measuring the (dis)similarity may be asymmetric and not any more satisfying the triangle inequal-

ity. We have shown[s] in subsequent papers[6,41,42] that both Welzl's exact randomized linear time SEB algorithm and Bădoiu and Clarkson's core-set approximation SEB algorithm generalize[t] nicely to a broader class of information-theoretic divergences called Bregman divergences. Bregman divergences are motivated by an axiomatic characteristic of a least squares "projection" optimization problem.[43] They are useful from the practitioner standpoint as they ease the construction of appropriate distances for heterogeneous spaces (modularity) and encapsulate conveniently the traditional (squared) Euclidean distance with various entropic functions like the relative entropy (also known as Kullback-Leibler or Information divergence). Moreover, it was proved[44] that the Vapnik-Chervonenkis dimension of Bregman balls[u] of $\mathbb{R}^d$ is $d+1$, namely identical to the case of Euclidean balls. This invariant property of the VC-dimension is all the more important for characterizing the capacity and accuracy[34] of learning methods that should not overfit training data sets. Further applications of information-theoretic SEBs are therefore expected in machine learning. Note that the geometric duality of covering and piercing problems described in Lemma 2 does not apply anymore "as is" in those information-theoretic spaces. (Lemma 2 only holds for symmetric Bregman divergences: namely, generalized quadratic distances.[44])

9. Conclusion

In this paper, we have first concisely surveyed and categorized major methods for computing exactly or approximately the smallest enclosing ball of a given set of points/balls in arbitrary dimension. We have then described a novel primal-dual core-set approximation algorithm for fitting the smallest enclosing ball that relies on the geometric duality of piercing/covering balls by balls. We have refined the approximation algorithm for small dimensions, and reported on implementation experiments. Finally, we have presented state-of-the-art machine learning algorithms for classification problems that rely on effective computations of exact or approximate SEBs, and discussed about the generalizations of the SEB in information-theoretic spaces: information-theoretic Bregman SEBs.

We conclude by mentioning a few research avenues for future work on the SEB problem:

- Consider the SEB problem under the *data stream* model of computation that assumes only sublinear memory (typically polylogarithmic space $O(\log^{O(1)} n)$). Data sets are potentially allowed to be streamed several times for computing the solution using multi-pass algorithms. H. Zarrabi-Zadeh and T. Chan[22]

[s] Applets online at `http://www.sonycsl.co.jp/person/nielsen/BregmanBall/BBC/` and `http://www.sonycsl.co.jp/person/nielsen/BregmanBall/MINIBALL/`

[t] Panigrahy's core-set algorithm[8] also generalizes to the case of Bregman divergences by sliding the ball circumcenter on the geodesic linking it to the farthest point F until it touches F.

[u] Bregman balls may not be necessarily convex. That makes them even more attractive to use as geometric classifiers.

proved that the simple heuristic of J. Ritter[21] yields a 3/2-approximation algorithm in arbitrary dimension using minimum space and a single pass. The query point/set distance filtering mechanism presented in section 3 can further be adapted into that setting using the interpoint distance approximation algorithm of N. Alon et al.[32] for ultra high dimensions. See also extensions and negative results concerning information-theoretic distances.[45]

- We have assumed (uniform constant) Euclidean distance function for writing down the circumcenter as a minimax optimization problem (see Equation 1). Namely, the SEB in Euclidean space. Consider now a *tensor metric* $\mathbf{G}$ that allows to compute the distance between any two points as the integral of the Riemannian metric along the geodesic path $\Gamma(P,Q)$ linking these points ($d\mathrm{t} = \mathbf{G}d\mathrm{x}$):

$$d_{\mathbf{G}}(P,Q) = \int_{\Gamma(P,Q)} \sqrt{\mathbf{G}_{ij} \frac{d\mathrm{x}_i}{d\mathrm{t}} \frac{d\mathrm{x}_j}{d\mathrm{t}}} d\mathrm{t}. \quad (5)$$

  The Riemannian SEB generalizes the Euclidean SEB by taking $\mathbf{G} = \mathbf{I}$, the identity matrix. The hyperbolic geometric (and hyperbolic SEB) is characterized by choosing tensor $d\mathrm{t}^2 = 4\frac{\sum_i d\mathrm{x}_i^2}{1-\sum_i x_i^2}^2$. Preliminary result for the special case where $d_{\mathbf{G}}(P,Q) = d_P(Q)$ is described in the thesis of Zürcher[46]. See also the notions of bags of Bregman divergences[44] and farthest Bregman Voronoi diagrams from which the circumcenter can be efficiently retrieved[47] from.
- The SEB problem is one of the simplest form of geometric containment parametric optimization problems. Consider more challenging shapes like the smallest enclosing ellipsoid of ellipsoids[48], smallest enclosing annulus of balls[49] with SVM-like applications in machine learning. See the recent work of Brandenberg and Roth presenting[50] a cutting plane algorithm for homothetic containment problems and that of K. Clarkson[51] reinterpreting and strenghtening core-set bound results of computational geometry under the framework of the Frank-Wolfe algorithm[38] for convex programming.
- Consider several statistical distributions (eg., uniform, normal, Poisson, etc.) and analyze the expected combinatorial size complexity of the support point/ball basis of the SEB. Devise efficient SEB algorithms according to those statistical properties.

Acknowledgements

The figures have been prepared with Ipe extensible drawing editor of Prof. Otfried Cheong. We are indebted to the reviewers who helped us improved the readibility of the manuscript. We thank Jason Hughes of Insomniac Games R&D for his appreciated feedback concerning the 2D/3D approximation algorithm. Part of this study was financially supported by ANR grant GAIA.

References

1. F. Nielsen and R. Nock. Approximating smallest enclosing balls. In *Proceedings International Conference on Computational Science and Its Applications (ICCSA)*, volume 3045 of *LNCS*. Springer.
2. J. J. Sylvester. A question in the geometry of situation. *Quarterly Journal of Pure and Applied Mathematics*, 1:79, 1857.
3. J. J. Sylvester. On Poncelet's approximate linear valuation of Surd forms. *Philosophical Magazine*, 20:203–222, 1860.
4. *The Collected Mathematical Papers of James Joseph Sylvester*. American Mathematical Society/Chelsea, 2007.
5. E. Welzl. Smallest enclosing disks (balls and ellipsoids). In H. Maurer, editor, *New Results and New Trends in Computer Science*, LNCS. Springer, 1991.
6. R. Nock and F. Nielsen. Fitting the smallest enclosing Bregman ball. In *European Conference on Machine Learning*, pages 649–656, 2005.
7. M. Bădoiu and K. L. Clarkson. Smaller core-sets for balls. In *Proceedings 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 801–802, 2003.
8. R. Panigrahy. Minimum enclosing polytope in high dimensions. *CoRR*, cs.CG/0407020, 2004.
9. N. Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of the ACM*, 31(1):114–127, 1984.
10. N. Megiddo. On the ball spanned by balls. *Discrete Computational Geometry*, 4(6):605–610, 1989.
11. K. Fischer and B. Gärtner. The smallest enclosing ball of balls: combinatorial structure and algorithms. *International Journal of Computational Geometry and Applications*, 14(4-5):341–378, 2004.
12. B. Gärtner and E. Welzl. On a simple sampling lemma. *Electronic Notes on Theoretical Computer Science*, 31, 2000.
13. B. Gärtner. A subexponential algorithm for abstract optimization problems. *SIAM Journal of Computing*, 24(5):1018–1035, 1995.
14. B. Gärtner and E. Welzl. Explicit and implicit enforcing: Randomized optimization. In *Computational Discrete Mathematics*, pages 25–46, 2001.
15. T. Szabos;. Unique sink orientations of cubes. In *FOCS '01: Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*, page 547, Washington, DC, USA, 2001. IEEE Computer Society.
16. B. Chazelle and J. Matoušek. On linear-time deterministic algorithms for optimization problems in fixed dimension. *Journal of Algorithms*, 21(3):579–597, 1996.
17. B. Gärtner. Fast and robust smallest enclosing balls. In *Proceedings 7th European Symposium on Algorithms*, pages 325–338, 1999.
18. K. Fischer, B. Gärtner, and M. Kutz. Fast smallest-enclosing-ball computation in high dimensions. In *11th Annual European Symposium*, pages 630–641, 2003.
19. T. Hopp and C. Reeve. An algorithm for computing the minimum covering sphere in any dimension, 1996. NISTIR 5831.
20. P. Gritzmann and V. Klee. Computational complexity of inner and outer j-radii of polytopes in finite-dimensional normed spaces. *Mathematical Programming*, 59(2):163–213, 1993.
21. J. Ritter. An efficient bounding sphere. In *Graphics gems*, pages 301–303. Academic Press Professional, Inc., 1990.
22. H. Zarrabi-Zadeh and T. Chan. A simple streaming algorithm for minimum enclosing balls. In *Proceedings 18th Canadian Conference on Computational Geometry*, pages 139–142, 2006.

23. X. Wu. A linear-time simple bounding volume algorithms. In *Graphics gems III*, pages 301–306. Academic Press, 1992.
24. M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2. Springer, second corrected edition edition, 1993.
25. B. Gärtner and S. Schonherr. An efficient, exact, and generic quadratic programming solver for geometric optimization. In *Proceedings 16th Symposium on Computational Geometry*, pages 110–118, 2000.
26. G. Zhou, J. Sun, and K.C. Toh. Efficient algorithms for the smallest enclosing ball problem in high dimensional space. Technical report, AMS Fields Institute Communications 37, 2003.
27. Y. E. Nesterov and M. J. Todd. Primal-dual interior-point methods for self-scaled cones. *SIAM Journal on Optimization*, 8(2):324–364, 1998.
28. M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings 34th ACM Symposium on Theory of Computing*, pages 250–257, 2002.
29. P. Kumar, J. S. B. Mitchell, and A. Yıldırım. Computing core-sets and approximate smallest enclosing hyperspheres in high dimensions. In *Algorithm Engineering and Experimentation*, LNCS, pages 45–55. Springer-Verlag, 2003.
30. P. K. Agarwal, S. Har-Peled, and K. Varadarajan. Geometric approximation via coresets. In J. E. Goodman, J. Pach, and E. Welzl, editors, *Combinatorial and Computational Geometry*. Math. Sci. Research Inst. Pub., Cambridge.
31. F. Nielsen. *Visual Computing: Geometry, Graphics, and Vision*. Charles River Media, 2005.
32. N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer Systems Science*, 58(1):137–147, 1999.
33. D. Avis and M. E. Houle. Computational aspects of Helly's theorem and its relatives. *International Journal of Computational Geometry and Applications*, 5(4):357–367, 1995.
34. T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, August 2001.
35. B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Conference on Learning Theory*, pages 144–152, 1992.
36. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, March 2000.
37. V. Vapnik and A. Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24, 1963.
38. M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.
39. I. W. Tsang, J. T. Kwok, and P.-M. Cheung. Core vector machines: Fast SVM training on very large data sets. *Journal of Machine Learning Research*, 6:363–392, 2005.
40. I. W. Tsang, A. Kocsor, and James T. Kwok. Simpler core vector machines with enclosing balls. In *International Conference on Machine Learning*, 2007.
41. F. Nielsen and R. Nock. On approximating the smallest enclosing Bregman balls. In *Proceedings 22nd ACM Symposium on Computational Geometry*, pages 485–486, 2006.
42. F. Nielsen and R. Nock. On the smallest enclosing information disk. In *Proceedings 18th Canadian Conference on Computational Geometry*, pages 131–134, 2006.
43. I. Csiszár. Why least squares and maximum entropy? an axiomatic approach to inference for linear inverse problems. *The Annals of Statistics*, 19(4):2032–2066, 1991.
44. F. Nielsen, J.-D. Boissonnat, and R. Nock. Bregman Voronoi diagrams: Properties, algorithms and applications. Technical Report RR-6154, INRIA, 2007. Extend ACM-SIAM Symposium on Discrete Algorithms, pp. 746–755, 2007.

45. S. Guha, P. Indyk, and A. McGregor. Sketching information divergences. In Nader H. Bshouty and Claudio Gentile, editors, *ACM Computational Learning Theory*, volume 4539, pages 424–438. Springer, 2007.
46. S. Zürcher. Smallest enclosing ball for a point set with strictly convex level sets, March 2007. Master's thesis.
47. S. Skyum. A simple algorithm for computing the smallest enclosing circle. *Information Processing Letters*, 37(3):121–125, 1991.
48. E. A. Yildirim. On the minimum volume covering ellipsoid of ellipsoids. *SIAM Journal on Optimization*, 17(3):621–641, 2006.
49. S. Schonherr. *Quadratic Programming in Geometric Optimization: Theory, Implementation and Applications*. PhD thesis, 2002.
50. R. Brandenberg and L. Roth. Minimal containment under homothetics - A practical approach. Technical report, 2006. preprint.
51. K. L. Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. Technical report, IBM Almaden Research Center, 2007. preprint, To appear in ACM-SIAM Symposium on Discrete Algorithms 2008.