Coresets, Sparse Greedy Approximation, and the Frank-Wolfe Algorithm

Kenneth L. Clarkson*

July 22, 2008

Abstract

The problem of maximizing a concave function $f(x)$ in a simplex S can be solved approximately by a simple greedy algorithm. For given k, the algorithm can find a point $x_{(k)}$ on a k-dimensional face of S, such that $f(x_{(k)}) \geq f(x_*) - O(1/k)$. Here $f(x_*)$ is the maximum value of f in S. This algorithm and analysis were known before, and related to problems of statistics and machine learning, such as boosting, regression, and density mixture estimation. In other work, coming from computational geometry, the existence of ϵ-*coresets* was shown for the minimum enclosing ball problem, by means of a simple greedy algorithm. Similar greedy algorithms, that are special cases of the Frank-Wolfe algorithm, were described for other enclosure problems. Here these results are tied together, stronger convergence results are reviewed, and several coreset bounds are generalized or strengthened.

1 Introduction

Consider the optimization problem

$$\begin{aligned} &\max_{x \in \mathbb{R}^n} f(x) \\ &\text{subject to } x \in S, \end{aligned} \tag{1}$$

where the given function $f(x)$ is concave, and S is the simplex that is the convex hull of the unit basis vectors of $\mathbb{R}^n$. The vertices of S are the points $e(i)$, $i = 1 \ldots n$, where $e(i)$ has coordinate $e(i)_i = 1$, and all other coordinates zero.

Special cases of this problem include the problems of training support vector machines and other classifiers, approximating functions as convex combinations of other functions, finding D-optimal designs, estimating mixtures of probability densities, and finding the smallest balls, ellipsoids, or axis-aligned ellipsoids containing a given set of points.

*IBM Almaden Research Center, San Jose, CA. klclarks@us.ibm.com Some of this work was done at Bell Labs, Alcatel-Lucent.

Algorithm 1.1. *Pick as* $x_{(0)}$ *the vertex of* S *with largest* f *value, that is,* $x_{(0)} := \arg\max\{f(e(i)) \mid e(i)$ *a vertex of* $S\}$;

For $k = 0 \ldots \infty$, *find* $x_{(k+1)}$ *from* $x_{(k)}$ *as follows:*

$i' := \arg\max_i \nabla f(x_{(k)})_i$;

$\alpha' := \arg\max_{\alpha\in[0,1]} f(x_{(k)} + \alpha(e(i') - x_{(k)}))$;

$x_{(k+1)} := x_{(k)} + \alpha'(e(i') - x_{(k)})$; *that is,* $x_{(k+1)}$ *is the point on the line segment from* $x_{(k)}$ *to* $e(i')$ *that maximizes* f;

Figure 1. Maximizing the concave function $f(x)$ in the simplex S

Algorithm 1.1, shown in figure 1, generates a sequence $x_{(k)} \in S$, $k = 0, \ldots \infty$, with increasing $f(x_{(k)})$. In practice the loop would exit when $x_{(k)}$ is an adequate solution by some application-specific criterion.

The procedure follows a limited form of gradient ascent: rather than optimize in the direction of the gradient, only the largest component of the gradient is used, so that $x_{(k)}$ has at most $k + 1$ nonzero entries. Moreover, the search direction $e(i') - x_{(k)}$ is used, from $x_{(k)}$ toward $e(i')$, not the direction $e(i')$; this keeps the search within S.

The computational task of finding the maximizing α' can be done by solving a quadratic equation, if f is a quadratic (multivariate) function. In § 3 on page 16, provably good performance is shown for a variant in which the α' value used at step k is a pre-defined α_k. Otherwise, the determination of α' is left to be determined for a given problem class.

Instances of this procedure, and variations of it, have been proposed independently many times, but the oldest version seems to be due to Frank and Wolfe [FW56]; they also proved a fundamental approximation result for general concave functions, described below. Similar algorithms, and results, have appeared in the machine learning literature, as *sparse greedy approximation*, as discussed in § 1.3 on page 7. In the computational geometry literature, similar algorithms have been proposed for the purpose of finding (or simply proving the existence of) coresets, which are discussed in §1.2, §4, and §5.

One of the contributions of this paper is just to put together these lines of work[1]. However, there are some specific new results:

- A definition of coresets that applies in the general setting of Algorithm 1.1, with corresponding general existence results (§4), proven with algorithms that are variations of Algorithm 1.1. One construction is worst-case tight (§5), exactly;
- Coreset results for support vector machines (SVM) that are significantly tighter, in constant factor, than known before (§7.2);
- Sample complexity bounds for learning, based on the coreset results (§6); in the case of SVM, these are similar to those proven via perceptrons and Novikoff's mistake bound (§7.2);

[1] The Frank-Wolfe algorithm was applied as such in some of the cited applications (e.g., [AST06, KY05]).

- Improvements in generality and provable speed of a practical SVM training algorithm based on coresets [TKZ06], and a sharper analysis of a simple approximation algorithm for minimum enclosing balls[BC];
- A quantitative approximation bound for Anyboost.L1 [MBBF00], since it is an instance of Algorithm 1.1;
- An algorithm for sparse greedy approximation that is a bit simpler than the one known in the machine learning literature [Zha03];
- Coreset results for boosting, (§7.3) previously unknown.

A look at Figure 3 on page 10 gives some idea of the range of applications of Algorithm 1.1. (However, results here do not generally apply to the ellipsoid problems.)

After discussing some general properties of Algorithm 1.1 and its use for sparse approximation, this introduction describes its use for primal/dual approximation, and how some further refinements result in an algorithmic existence proof for coresets. Finally, the relation to the sparse greedy approximation technique of learning theory is given, and an outline of the remainder of the paper.

1.1 Sparse approximation

The original Frank-Wolfe algorithm applies in the general context of maximizing a concave function $f()$ within a feasible polytope F. At each iteration, the algorithm solves the linear programming problem of finding the optimum y' maximizing within F the local linear approximation $f(y) \approx f(x) + (y - x)^T \nabla f(x)$, where x is the current iterate $x_{(k)}$. The algorithm then solves the one-dimensional optimization problem of finding the largest value of f on the line segment $[x, y']$.

Here the feasible set F is the simplex S, and the solution of the maximization problem is the $y' \in S$ that maximizes $y^T \nabla f(x)$; that optimum y' is simply $e(i')$, as used in Algorithm 1.1. That is, $\max_{y \in S} y^T \nabla f(x) = \max_i \nabla f(x)_i$. The maximum value of the linear approximation is

$$(2) \qquad f(x) + ((e(i') - x)^T \nabla f(x) = \max_i \nabla f(x)_i + f(x) - x^T \nabla f(x).$$

This function of x is the *Wolfe dual*, as discussed in § 2.1 on page 9.

As reviewed in Theorem 2.2 on page 14, for a certain nonlinearity measure $C_f \geq 0$ of f, related to the second derivative of f, and for $x_* \in S$ maximizing f in S, this procedure satisfies

$$f(x_{(k)}) \geq f(x_*) - 4C_f/(k+3).$$

Such a relation was shown by Frank and Wolfe[FW56], and has also been shown for the related Algorithm 1.2 on page 8, as discussed in § 1.3.

This convergence rate is slow, compared to modern methods, but the simplicity of Algorithm 1.1 means that for many problems, the work per iteration is small; in particular, the solution of linear systems, as is often done in faster-converging methods, is not needed by Algorithm 1.1. For some large-scale applications, such a trade-off is favorable to algorithms like Algorithm 1.1, as discussed by Platt [Pla99] and by Tsang *et al.* [TKC05, TKL05] for the particular case of the training of support vector machines. Another example is semidefinite programming, where the number of variables can be quite large. Similar considerations also motivate some recent interest in more general gradient descent algorithms [Nem05, Haz06, TLJJ06].

Another motivation is that sometimes rough approximations are acceptable for an application. For example, when computing a statistical estimator involves an optimization problem, the noisiness of the data implies that the parameters given by the optimal solution, and the corresponding true (population) values of those parameters, are only related by error upper bounds. In such a situation, an approximation within a small factor of the error upper bounds is likely to be quite acceptable. Such a point was observed most recently perhaps by Altun and Smola [AS06]; they propose the use of Algorithm 1.2 on page 8 for some general problems of inference; it's worth noticing that Algorithm 1.1 is often applicable to these problems also.

Perhaps the main point of interest for Algorithm 1.1 is the sparsity of the solutions that it finds. The iterate $x_{(k)}$ has few (at most $k+1$) nonzero entries, and so is sparse in that sense. Thus the convergence result mentioned above shows that there are sparse solutions that are provably good approximations. In the extreme case $C_f = 0$, f is simply a linear function, and the optimum x_* is one of the vertices: a very sparse solution, with one nonzero entry. More generally, the smaller C_f is, the flatter f is, and the more effective a procedure based on local linear approximation by the gradient will be.

When $f(x)$ is a quadratic function $f(x) = a + x^T b + x^T M x$, for M negative semidefinite, the value of C_f is no more than the square of the diameter of the minimum enclosing ball of a set of points derived from M.

The Minimum Enclosing Ball (MEB, 1-center, smallest enclosing sphere) problem is as follows: given a set of points $P = \{p_1 \dots p_n\}$ in d dimensions, find the smallest ball containing all points of P. The dual of the MEB problem is of the form (1), with $f(x) = x^T b - x^T A^T A x$, where $b_i = p_i^2 = p_i^T p_i$, and A is the matrix whose columns are the p_i. Letting $c := Ax$, the gradient of f is $b - 2A^T Ax = b - 2A^T c$, and the i'th coordinate of $\nabla f(x)$ is $p_i^2 - 2p_i^T c = (p_i - c)^2 - c^2$. Thus the coordinate i' corresponds to the point $p_{i'}$ farthest from c. Picking $\alpha' = \alpha_k = 2/(k+3)$ at the k'th step, as discussed in § 3 on page 16, thus is very close to an algorithm proposed by Bădoiu and Clarkson ([BC03], the "simple" algorithm), and so by Theorem 2.2 on page 14, their algorithm needs $O(nd/\epsilon)$ time to return a ball with radius within $1+\epsilon$ of smallest. This sharpens their analysis, and matches the running time of Panigrahy's algorithm [Pan04], with an arguably simpler algorithm. Some other approximation algorithms for MEB [BC03, KMY03] have running time $O(nd/\epsilon + 1/\epsilon^{O(1)})$, and are somewhat more complicated.

Although the approximation error bound for Algorithm 1.1 is additive, in general, for the MEB problem the term C_f is proportional to the square R^2 of the MEB radius, and since the algorithm is used in a formulation of the MEB problem where the optimum of the objective function is also R^2, the error bound for MEB is relative. For SVM, where the objective function is the squared thickness G^2 of a certain separating slab, the error bound can be expressed in terms of R^2/G^2, a ratio which is familiar as the VC-dimension of a range space associated with SVM.

Similarly to MEB, the MEE (minimum enclosing ellipsoid) problem, to find the smallest ellipsoid enclosing a set of points, is dual to a concave maximization problem over S [Kha96, KY05, AST06]. That maximization problem is called the D-optimal design problem, and an algorithm similar to Algorithm 1.1 was proposed for it by Wynn and by Fedorov [Wyn70, Fed72]. Not all results given here apply in an interesting way to this problem, however, because the C_f bound is too large when considered over all of S.

Primal/Dual Approximation. The property $f(x_{(k)}) \geq f(x_*) - 4C_f/(k+3)$ mentioned above can be expressed as $h(x_{(k)}) \leq 1/(k+3)$, where $h(x)$ is the scaled measure

$$h(x) := (f(x_*) - f(x))/4C_f. \tag{3}$$

A key property satisfied by the algorithm, that implies this approximation bound, is that

$$h(x_{(k+1)}) \leq h(x_{(k)}) - h(x_{(k)})^2, \tag{4}$$

when $h(x_{(k)}) \leq 1/2$, as given in (18) on page 14. A similar relation was shown by Frank and Wolfe[FW56] for Algorithm 1.1, and so this implies that such a result holds for Algorithm 1.2 on page 8; the latter was shown also by Zhang [Zha03], as discussed in § 1.3.

The analysis reviewed here also implies this, but gives a stronger condition: let $w(x)$ denote the value at x of the dual optimization of (1); this problem is described at (8). Let

$$g(x) := (w(x) - f(x))/4C_f, \tag{5}$$

a scaled version of the gap between $w(x)$ and $f(x)$. Note that $w(x) \geq f(x)$ for all feasible x. Then as stated as Theorem 2.1 on page 13, the iterates of algorithms 1.1 and 1.2 satisfy

$$h(x_{(k+1)}) \leq h(x_{(k)}) - g(x_{(k)})^2, \tag{6}$$

when $g(x_{(k)}) \leq 1/2$.

This bound is plainly always as good as the prior one (4), since the dual gap measure $g(x_{(k)}) \geq h(x_{(k)})$ always, and when $g(x_{(k)}) - h(x_{(k)})$ is large, the bound will be much better. Of course, it could also be that $g(x_{(k)}) - h(x_{(k)})$ is small, so that the guaranteed improvement in f is no better than that previously known. However, as Theorem 2.3 on page 14 states, the implication of this stronger bound is that in $2/\epsilon$ iterations, an iterate $x_{(k)}$ will be seen that has both $h(x_{(k)}) \leq \epsilon$, as in prior analyses, and also $g(x_{(k)}) \leq \epsilon$. That is, $x_{(k)}$ will be "good" both primally and dually.

A special case of this observation was made by Khachian [Kha96], and proven in this generality by Ahipasaoglu *et al.*[AST06]; (In that work, the main focus was on the minimum enclosing ellipsoid problem.) This primal/dual approximation property is not far from the specification of a *coreset*, discussed next.

1.2 Coresets

Coresets were first explicitly described for the MEB problem described above. For $\epsilon > 0$, an *ϵ-coreset* $P' \subset P$ has the property that if the smallest ball containing P' is expanded by a factor of $1 + \epsilon$, then the resulting ball contains P. Also, since the smallest ball containing P also contains P' in particular, the smallest ball containing P must be at least as large as the smallest ball containing P'. That is, P' gives both a good approximate solution, and proves that no solution is much better.

A fundamental property of the MEB problem is that, as shown by Bădoiu *et al.*[BHPI02], there are ϵ-coresets whose size (number of points) depends only on ϵ, and not on the dimension d, or the number n of points. Algorithms are known for finding ϵ-coresets of size $O(1/\epsilon)$ [KMY03, BC03], and size $\lceil 1/\epsilon \rceil$ is worst-case optimal [BC]. Coresets for MEB have been applied to the k-center problem [BHPI02], to computational biology [BJGL$^+$03], and to machine learning [NN06], including support vector regression [TKL05].

The results of Ben-David *et al.* also imply the existence of coresets, for MEB and a few other problems [BDES02]. Their work relies on the existence result attributed to Maurey, as mentioned in § 1.3 on the next page; their also § 2.4 on page 15 gives an argument like Maurey's. Their application was the *densest-ball problem*, discussed below.

As mentioned, the Wolfe dual of the MEB problem is an instance of the optimization problem (1). Moreover, a known algorithm for finding MEB coresets [BC03] is similar to Algorithm 1.1. This is not a coincidence: § 4 on page 17 shows that ϵ-coresets exist for the dual problem of (1), with a size that depends only on ϵ and C_f. Here the idea of a coreset is generalized from the MEB problem to the more general setting.

As defined technically (Definition 4.1 on page 18), a coreset here is a subset $N \subset \{1, \ldots, n\}$, or equivalently the face S_N of S specified by N, where S_N is the convex hull of $\{e(i) \mid i \in N\}$. In particular, a coreset is such a subset (or face) with the property that $\arg\max_{x \in S_N} f(x)$ is a good approximate solution to the full problem (1), both primally and dually. In other words, N is a *combinatorial* specification of an approximate solution, which is also a certificate of the solution's near-optimality.

The technical definition includes a factor of 2 and of a variation C_f^* of C_f, so that as specialized to the MEB problem, the definition matches the previous one.

The coreset existence proof includes an algorithm, Algorithm 4.2 on page 18, that builds a coreset; the algorithm is very similar to Algorithm 1.1, but does a little more work: having chosen a coordinate to make non-zero, it finds the point x that maximizes $f(x)$ over all points with the same set of non-zero entries. That is, it solves some small optimization subproblems.

The coreset size, for a given quality of additive approximation ϵ, is at most $4C_f/\epsilon$, as shown in Theorem 4.3 on page 18. A simpler general algorithm, given in Theorem 4.4 on page 19, gives a poorer quality coreset, while a slower algorithm, Algorithm 5.1 on page 22, gives a coreset of roughly half the size obtainable via the faster algorithm, with the same approximation quality. Algorithm 5.1 uses an "away" step within each iteration, in which a nonzero coordinate is set to zero. Such a step was considered by Todd and Yildirim [TY05], as a heuristic improvement for optimization, and within an algorithm for optimal MEB coresets, by Bădoiu and Clarkson [BC]. The results here generalize the latter, and are asymptotically optimal for MEB coresets, as $\epsilon \to 0$.

The exact size ℓ of a coreset of given quality is significant, because some algorithms exploiting coresets involve enumerating, for an input set of n points, all $\binom{n}{\ell}$ subsets of size ℓ. One such application is the densest-ball problem: given a set of points, find the smallest ball containing half the points. The existence of a small ϵ-coreset for the points inside that smallest ball implies that enumeration of all subsets of that size will allow the densest-ball problem to be solved approximately. A similar application of coresets in the convex approximation setting is described in § 7.1 on page 24; for the densest-ball problem, such algorithms were proposed by Ben-David *et al.*, as mentioned [BDES02].

Several previous papers have shown the effectiveness of coreset techniques for support vector machine (SVM) training and regression [TKC05, CTK04, TKZ06], of which one [TKZ06] generalizes from the MEB problem to a more general quadratic objective function. The work here is inspired by, and would seem to include, that prior generalization. Har-Peled *et al.* [HPRZ07] give an algorithm for hard-margin SVM training that is not far from Algorithm 4.2 on page 18. However, their algorithm solves small subproblems optimally at each step, as in Algorithm 4.2, but unlike Algorithm 1.1, and is perhaps slower than Algorithm 1.1 as specialized to SVM training. (However, a variant of their algorithm avoids such subproblem computations [HP07].)

SVM training is discussed in § 7.2 on page 26. As applied to proving the existence of coresets for SVM, the coreset size proven here via Algorithm 5.1 on page 22 is smaller by a constant factor than that in [HPRZ07].

Coresets for other classes of problems have seen wide application in computational geometry: see [AHPV05] for a survey.

1.3 Sparse Greedy Approximation

The problem (1) is often given in the equivalent formulation of minimizing a convex function, and in that form is considered in statistics, approximation theory, and machine learning. (Here we minimize a convex function by maximizing its concave negation.) For example, for an appropriate $d \times n$ matrix A and point p, maximizing $f(x) := -\|Ax - p\|_2^2$ for $x \in S$ would correspond to the problem of finding the convex combination of the columns of A that is closest to p in Euclidean norm. Similarly, if a collection of functions $p_i(t)$ and a target function $p(t)$ is given, then $f(x) := -\|\sum_i x_i p_i(t) - p(t)\|_2^2$, corresponds to the problem of finding the closest (in L_2) convex combination of the $p_i()$ functions to match p [LB00].

Algorithm 1.1 on page 2 can be viewed generically as finding, at each step, a good coordinate in which to change x, and then adjusting that coordinate to maximize $f(x)$. One such algorithm in particular, described in figure 2 on the following page, has been analyzed before. Plainly this algorithm will find an iterate $x_{(k+1)}$ for which $f(x_{(k+1)})$ is at least as large as for Algorithm 1.1.

Algorithm 1.2. *Proceed as in Algorithm 1.1 on page 2, but rather than pick i' and α' as in that algorithm, find the i' and α' that maximize $f(x_{(k)} + \alpha(e(i) - x_{(k)}))$ over all i and α, and set $x_{(k+1)}$ to the corresponding point.*

Figure 2. Maximizing the concave function $f(x)$ in the simplex S

Sparsity results for Algorithm 1.2 were shown in generality by Zhang[Zha03], whose proof has the same general idea as analyses by Li and Barron [LB00] and Jones [Jon92]. Maurey proved the existence of a sparse x' with $f(x') \geq f(x_*) - O(1/(k+3))$, for a class of functions $f(x)$ arising in convex approximation, using a probabilistic argument ([Pis81], see also [Bar93]), and Jones showed that a greedy algorithm such as Algorithm 1.2 yields a similar output [Jon92].

Algorithm 1.2, and variations, has been applied to boosting, regression, convex approximation, and estimation of mixture models [Zha03] (applicability to SVM training is also mentioned by Zhang, but no specific results are given). Algorithm 1.1 can thus be similarly applied, for those $f(x)$ whose gradients can be computed; this includes all the specific applications considered by Zhang. When a single gradient computation is faster than n function evaluations, as may be true, Algorithm 1.1 and its variants will be faster than Algorithm 1.2 and comparable variants. (However, it's also true that the n evaluations of a function at closely related arguments may be sped up.) Also, since the problems discussed by Zhang involve optimization over convex hulls, the coreset results of § 4 on page 17 apply. The existence of coresets for these problems may have some useful applications; the case of convex approximation is discussed in § 7.1 on page 24.

Algorithm 1.1 is also closely related to Anyboost.L1[MBBF00], which can roughly be viewed as the specialization of Algorithm 1.1 to boosting. Although convergence results were shown for Anyboost, it doesn't seem that quantitative results have been, and so the results here are a new contribution in that respect.

1.4 Outline

The next section gives an analysis of Algorithm 1.1, after describing the dual optimization problem and defining the nonlinearity measure C_f. The probabilistic argument for sparse approximate solutions is reviewed in § 2.4 on page 15, with a particular example related to k-means clustering. Section 3 on page 16 gives some variations on Algorithm 1.1, that do less work per iteration by using a predefined series of α' values. Section 4 on page 17 considers the more general case of optimization within the convex hull of a set of points, and then gives a construction for coresets of functions $f(x)$ of the general form $\hat{f}(Ax)$; as noted, this includes all quadratic concave $f(x)$. Tighter bounds for coresets using an algorithm with "away" steps is discussed in § 5 on page 21. In § 6 on page 23, some tail estimates for random data are shown, and finally some specific applications are considered in § 7 on page 24: convex approximation, support vector machines (SVM), Adaboost, and approximation in L_v. The results for SVM include an error probability estimate based on the tail estimates of § 6.

Figure 3 lists some of the problems that can be described as instances of

(1). The conditions that x is an n-vector and A is $d \times n$ imply the dimensions of the remaining values. As elsewhere in this paper, the notation c^2 for a vector c denotes $c^T c$, and $\mathbf{e}$ is the n-vector with all coordinates equal to one. Also the i'th column of A is p_i, or instead $y_i p_i$, when a given n-vector y is associated with the problem. As usual in this paper, $c = Ax$, and $M := -A^T A$, a negative-semidefinite matrix. The notation $(A \circ A)_j$ denotes the n-vector whose i'th coordinate is p_{ij}^2, and $H \succeq 0$ denotes the condition that matrix H is positive semidefinite.

The three problems listed after the general quadratic are themselves quadratic, so that the duals of MEB, convex approximation, and L_2-SVM can be read off from the dual for the general quadratic. At the risk of some confusion, the "M" of the general quadratic is instantiated for L_2-SVM as the matrix with block structure

$$-[A^T \; y \; I/C][A^T \; y \; I/C]^T,$$

and so the "c" of the general quadratic is for L_2-SVM the $(d+1+n)$-vector $[c^T \; q \; w^T]^T$. Along similar lines, the "A" of the general quadratic is not quite the same as the "A" for the general expression for maximizing functions within a convex hull of points: see the beginning of § 4.2 on page 20.

2 Primal and Primal/Dual Approximation

Before analyzing Algorithm 1.1, some reminders. We let $\mathbf{e}$ be the column n-vector with all coordinates equal to one, so that the simplex S can be written as

$$S := \left\{ x \in \mathbb{R}^n \;\middle|\; x^T \mathbf{e} = 1, x \geq 0 \right\}. \tag{7}$$

2.1 The Wolfe Dual

When $f(x)$ is continuously differentiable, the *Wolfe dual* problem to the maximization problem (1) is

$$\begin{aligned} &\min_{z \in \mathbb{R}, x \in \mathbb{R}^n} \; z + f(x) - x^T \nabla f(x) \\ &\text{subject to } z \geq \max_i \nabla f(x)_i. \end{aligned} \tag{8}$$

Since plainly the smallest feasible z is $\max_i \nabla f(x)_i$, let

$$z(x) := \max_i \nabla f(x)_i,$$

and the dual problem becomes

$$\min_{x \in \mathbb{R}^n} w(x), \text{ where } w(x) := z(x) + f(x) - x^T \nabla f(x).$$

As mentioned in § 1, this function $w(x)$ is the maximum value within S of the local linear approximation to $f(x)$, corresponding to the tangent hyperplane to

Problem	$f(x)$ References	Dual problem Notes
General Function	$f(x)$ §2.1 for dual	$\min_{z,x}\quad z + f(x) - x^T\nabla f(x)$ $\text{s.t.}\quad z\mathbf{e} \geq \nabla f(x)$
Functions Within Convex Hull	$f(x) := \hat{f}(Ax)$ §4	$\min_{z,c}\quad z + \hat{f}(c) - c^T\nabla \hat{f}(c)$ $\text{s.t.}\quad z\mathbf{e} \geq A^T\nabla \hat{f}(c)$
General Quadratic	$a + x^T b + x^T Mx$ §4.2	$\min_{z,c}\quad a + z + c^2$ $\text{s.t.}\quad z\mathbf{e} \geq b - 2A^T c$ $c = Ax$ $C_f \leq \operatorname{diam}(AS)^2$
ME Ball, SVDD	$x^T b(A) + x^T Mx$ §1, §4.2	$\min_{z,c}\quad z + c^2$ $\text{s.t.}\quad z\mathbf{e} \geq b(A) - 2A^T c$ $b(A)_i := p_i^2$ $C_f^* \leq \operatorname{radius}(AS)^2$
Convex Approx., SVM	$-(p - Ax)^2$ §7.1,[TKC05],[HPRZ07]	$\min_{z,c}\quad -p^2 + z + c^2$ $\text{s.t.}\quad z\mathbf{e} \geq 2A^T p - 2A^T c$
L_2-SVM (two-class)	$-x^T[M + yy^T + I/C^2]x$ $= -([A^T\ y\ I/C]^T x)^2$ [TKZ06]	$\min_{z,c,q,w}\quad z + c^2 + q^2 + w^2$ $\text{s.t.}\quad z\mathbf{e} \geq -2(A^T c$ $+qy + w/C)$ $C_f \leq (\operatorname{diam}(AS) + 1 + 1/C)^2$; $c = Ax$, $q = y^T x$, $w = x/C$; $y_i = \pm 1$
L_2-SVR	(see [TKZ06])	
Adaboost	$-\log \sum_j \exp(-C(Ax)_j r_j)$ §7.3 [Zha03]	(not given here) $A_{ji} =$ predictor i, datapoint j; $r_j = \pm 1$; $C_f \leq 4C^2$;
L_v Regression, $1 < v < \infty$	$-\lVert p - Ax\rVert_v^{v'}$, $v' := \min\{v, 2\}$ §7.4,[Zha03]	(not given here) $C_f \leq 2(v-1)\operatorname{diam}(AS)^2$, $v \geq 2$;
ME Ellipsoid, D-Optimal Design	$\log\det AXA^T$ [AST06][Kha96]	$\min_{H \succeq 0}\quad -\log\det H$ $\text{s.t.}\quad p_i^T H p_i \leq d,$ $i = 1 \dots n$ diagonal X, $X_{ii} = x_i$;
ME Axis-Aligned Ellipsoid	$(1/2)[d\log d + \sum_{1\leq j\leq d} \log(x^T(A\circ A)_j - (Ax)_j^2)]$ [KY05]	$\min_{\gamma,\mu}\quad -\sum_j \log\gamma_j$ $\text{s.t.}\quad \sum_j(\gamma_j p_{ij} - \mu_j)^2 \leq 1$ $i = 1 \dots n$

Figure 3. Some problems within the framework.

the graph of f at x. By the concavity of f, every such tangent hyperplane is above (or touching) the graph of f, and so for all $x \in S$, $w(x) \geq f(x_*)$.

Thus the weak duality condition $w(x) \geq w(x_*) \geq f(x_*) \geq f(x)$ is immediate. This condition is all that is needed for many results here, but for the some purposes, in particular the "away" algorithm of § 5 on page 21, stronger duality conditions must be proven, specifically, that if x^* is primal optimum, then it is also dual optimum. Also, *complementary slackness* holds, that is, the ith coordinate of x_* and the quantity $z_* - \nabla f(x_*))_i$ cannot both be nonzero.

To obtain these conditions, and prove the weak duality conditions more formally, the Lagrangian relaxation can be used. A familiar form of this relaxation, for a convex function $-f(x)$ and for the equivalent optimization problem $-\min_{x\in S} -f(x)$, is:

$$-\max_{\substack{z\in\mathbb{R}\\ \lambda\in\mathbb{R}^n,\lambda\geq 0}} \min_{x\in\mathbb{R}^n} -f(x) - x^T\lambda + z(x^T\mathbf{e} - 1),$$

for which the minimum with respect to x is achieved when all derivatives with respect to x are zero, that is, $-\nabla f(x) - \lambda + z\mathbf{e} = 0$, or $\lambda = z\mathbf{e} - \nabla f(x)$. The condition $\lambda \geq 0$ implies $z\mathbf{e} \geq \nabla f(x)$, and plugging this expression for λ into the Lagrangian, the problem becomes

$$\begin{aligned}
-\max_{z\in\mathbb{R}} \min_{x\in\mathbb{R}^n} & -f(x) - x^T(z\mathbf{e} - \nabla f(x)) + z(x^T\mathbf{e} - 1) \\
&= -\max_{z\in\mathbb{R},x\in\mathbb{R}^n} -f(x) + x^T\nabla f(x) - z \\
&= \min_{z\in\mathbb{R},x\in\mathbb{R}^n} z + f(x) - x^T\nabla f(x),
\end{aligned}$$

subject to $z\mathbf{e} \geq \nabla f(x)$, as claimed.

The Slater condition, which is that a feasible point exists that satisfies all inequality constraints strictly, is easily seen to be satisfied, and so strong duality holds, by Slater's Theorem [BV04]. Thus an optimum point x_*, for which $f(x_*)$ is maximum, is also a dual optimal point. For any feasible point $x \in S$ we have

$$w(x) \geq w(x_*) = f(x_*) \geq f(x),$$

and also complementary slackness.

2.2 The measure C_f

The quantity C_f is defined as

$$C_f := \sup \frac{1}{\alpha^2}(f(x) + (y - x)^T\nabla f(x) - f(y)), \tag{9}$$

where the supremum is over all x and z in S, and over all α so that $y = x + \alpha(z - x)$ is also in S. The set of such α includes $[0, 1]$, but α can also be negative. The concavity of f implies that $f(y)$ is always no more than the local linear approximation $f(x)+(y-x)^T\nabla f(x)$. The quantity C_f bounds how much less $f(y)$ will be, measured as a quadratic function of the "distance" of y from x, where the "distance" is the α so that $y = x + \alpha(z - x)$ for some $z \in S$.

The expression in the supremum bears a clear relation to the *Bregman distance* from y to x induced by $-f$.

There are some expressions for C_f that hold for special cases.

When f is twice differentiable, the Taylor expansion at $\alpha = 0$ of $f(x+\alpha(z-x))$ as a function of α is:

$$f(x+\alpha(z-x)) = f(x) + \alpha(z-x)^T\nabla f(x) + \frac{1}{2}\alpha^2(z-x)^T\nabla^2 f(\tilde{x})(z-x).$$

Here the last term is the Lagrange remainder, where $\tilde{x}$ is a point on the line segment $[x, x+\alpha(z-x)]$. (Everywhere but in § 5 on page 21, the values of α considered are between 0 and 1, so that $\tilde{x}$ in on the line segment $[x, z]$.)

Since f is concave, $\nabla^2 f(\tilde{x})$ is negative semidefinite, and so the last term is always negative. Rearranging,

$$\begin{aligned} C_f &\leq \sup_{x,z\in S} -\frac{1}{2}(z-x)^T\nabla^2 f(\tilde{x})(z-x) \\ &\leq \sup_{x,z\in S,\tilde{x}} -\frac{1}{2}(z-x)^T\nabla^2 f(\tilde{x})(z-x), \end{aligned} \tag{10}$$

where $\tilde{x}$ is constrained to lie in S, and on the line through x and z.

When $f(x)$ has the form $f(x) = \hat{f}(Ax)$, where A is a matrix of conforming shape and $\hat{f}$ is a function taking a conforming argument, the bound (10) for C_f can be expressed as follows. With $a := Ax \in AS$, and similarly $b := Az, \tilde{a} := A\tilde{x}$, and observing that $\nabla^2 f(x) = A^T\nabla^2\hat{f}(Ax)A$,

$$\begin{aligned} C_f &\leq \sup_{x,z\in S} -\frac{1}{2}(z-x)^T\nabla^2 f(\tilde{x})(z-x) \\ &= \sup_{x,z\in S} -\frac{1}{2}(z-x)^T(A^T\nabla^2\hat{f}(A\tilde{x})A)(z-x) \\ &= \sup_{a,b\in AS} -\frac{1}{2}(b-a)^T\nabla^2\hat{f}(\tilde{a})(b-a) \end{aligned} \tag{11}$$

Finally, when $f(x) = x^Tb + x^TMx$, for some vector b and negative semidefinite matrix M, then $\nabla^2 f(x) = 2M$. Moreover, there is a factorization $-M = A^TA$ for a matrix A, and so

$$\begin{aligned} C_f &\leq \sup_{x,z\in S} -\frac{1}{2}(z-x)^T\nabla^2 f(x)(z-x) \\ &\leq \sup_{x,z\in S} (z-x)^TA^TA(z-x) \\ &= \sup_{a,b\in AS} \|b-a\|_2^2. \end{aligned} \tag{12}$$

That is, C_f is no more than the square of the diameter of the polytope AS, and the latter is bounded by the square of the diameter of the minimum enclosing ball of AS, and hence by four times the square of the radius of that ball.

A related quantity that has appeared in the functional analysis and machine learning literatures is the *modulus of convexity*, introduced by Clarkson (in particular, J. A. Clarkson). This quantity could be defined similarly to C_f, but using the infinum instead of the supremum: a lower bound on nonlinearity, instead of an upper bound.

Asymptotic C_f^*. It is of interest to bound a more restricted set of values, namely, where x is near the optimal point x_*. This models the use of C_f in analyzing the algorithms discussed here, as they converge to the optimum. Given region $S' \subset S$, consider $C_f(S')$, defined as in (9) on page 11, but where x is restricted to S'. For given $\gamma \geq 0$, consider S_γ to be the subset of S that contains all x such that $f(x) \geq f(x^*) - \gamma$. For large enough $k = \Omega(C_f/\gamma)$, as $\gamma \to 0$, all $x_{(k)}$ are in S_γ, and so for $C_f(S\gamma) < C_f = C_f(S)$, the additive bound of Theorem 2.2 on the following page can be tightened for large k. We will assume at times that $C_f(S_\gamma) = C_f^*(1 + o(1))$ as $\gamma \to 0$, where $C_f^* := C_f(\{x_*\})$. This is true for quadratic functions, for which C_f^* is the maximum squared distance of x_* to a point in AS. For MEB in particular, C_f^* is the square of the radius of the MEB. That is, the asymptotic C_f^* for MEB is one quarter the general bound. This distinction is not significant in general, but by showing results with respect to C_f^*, it will be possible to show that some constructions for MEB are asymptotically optimal, including the constant factor in the leading term.

2.3 Primal/Dual Bounds

The following theorem is essentially due to Ahipasaoglu *et al.*[AST06], and generalizes a lemma of Khachian [Kha96].

Theorem 2.1. *For simplex S and continuously differentiable concave function f, one iteration of Algorithm 1.1 on page 2 satisfies*

$$h(x_{(k+1)}) \leq h(x_{(k)}) - g(x_{(k)})^2.$$

As noted, this bound applies also for Algorithm 1.2 on page 8.

Proof. For simpler notation, let $x := x_{(k)}$, $y := x_{(k+1)}$.

We have, by definition,

$$\nabla f(x)_{i'} = z(x) = w(x) - f(x) + x^T \nabla f(x). \tag{13}$$

Also by definition (9),

$$f(x + \alpha(e(i) - x)) \geq f(x) + \alpha(e(i) - x)^T \nabla f(x) - \alpha^2 C_f, \tag{14}$$

for all i, including i'. But using (13),

$$\begin{aligned}(e(i') - x)^T \nabla f(x) &= \nabla f(x)_{i'} - x^T \nabla f(x)\\ &= w(x) - f(x),\end{aligned}$$

and with (14),

$$f(y) \geq f(x) + \alpha(w(x) - f(x)) - \alpha^2 C_f. \tag{15}$$

which implies, using (3), (5), and (17),

$$
\begin{aligned}
h(y) = (f(x_*) - f(y)/4C_f \\
&\le h(x) - \alpha(w(x) - f(x))/4C_f + \alpha^2/4 \\
&= h(x) - \alpha g(x) + \alpha^2/4 && (16)\\
&\le h(x) - g(x)^2. && (17)
\end{aligned}
$$

Here the last step assumes $g(x) \le 1/2$, so that the minimizing $\alpha = 2g(x)$ can be chosen. However, this assumption will always hold, by the choice of $x_{(0)}$: suppose $g(x) > 1/4$. Then by (16), $h(y) < h(x)$ even when $\alpha = 1$, that is, when y is a vertex of S. But since $f(x_{(0)}) \ge f(e(i))$ for all i, and f is concave, and $f(x_{(k)})$ increases as k increases, $h(x_{(k)})$ is less than $h(e(i))$ for any i. Hence $g(x) \le 1/4$, and the minimizing $\alpha = 2g(x)$. □

Theorem 2.2. *For simplex S and concave function f, Algorithm 1.1 (and Algorithm 1.2 on page 8) finds a point $x_{(k)}$ on a k-dimensional face of S such that*

$$h(x_{(k)}) = (f(x_*) - f(x_{(k)}))/4C_f \le 1/(k+3)$$

for $k > 0$.

Proof. The vertices of the k-face will be the vertices of S (the $e(i')$) associated with each $x_{(k)}$. It remains to bound the values $h(x_{(k)})$. Since $g(x) \ge h(x)$, we always have

$$h(x_{(k+1)}) \le h(x_{(k)}) - h(x_{(k)})^2. \tag{18}$$

From the proof of the last theorem, $g(x_{(0)}) \le 1/2$, and so $h(x_{(1)}) \le 1/2 - (1/2)^2 = 1/4$. More generally, noting that $1 - \gamma \le 1/(1+\gamma)$ for $\gamma > -1$, and letting $h_k := h(x_{(k)})$,

$$h(x_{(k+1)}) \le h_k(1 - h_k) \le \frac{h_k}{1 + h_k} = \frac{1}{1 + 1/h_k}$$

and so by induction, $h(x_{(k)}) \le 1/(k+3)$ for $k > 0$, and the theorem follows. □

Theorem 2.3. *For simplex S and continuously differentiable concave function f, and given $\epsilon > 0$, Algorithm 1.1 (and Algorithm 1.2) will have an iterate $x_{(\hat{k})}$ with $\hat{k} \le 2K$, where $K := \lceil 1/\epsilon \rceil$ so that $g(x_{(\hat{k})}) \le \epsilon$. That is, $w(x_{(\hat{k})}) - f(x_{(\hat{k})}) \le 4\epsilon C_f$.*

Of course, to determine which of the $x_{(k)}$ has $g(x_{(k)}) \le \epsilon$ seems to need the knowledge of C_f; for an existence proof this knowledge can be assumed. Since $w(x) - f(x) = z(x) - x^T \nabla f(x)$, however, the point $x_{(k)}$, for $k = K \dots 2K$ with minimum $z(x) - x^T \nabla f(x)$ will have both $h(x_{(k)}) \le \epsilon$ and $g(x_{(k)}) \le \epsilon$.

Proof. The previous theorem shows that $k \le K$ iterations suffice to obtain $x_{(k)}$ with $h(x_{(k)}) \le \epsilon$. During subsequent iterations, either iterate $x_{(j)}$, has $g(x_{(j)}) \le \epsilon$ or $h(x_{(j+1)}) \le h(x_{(j)}) - \epsilon^2$ by Theorem 2.1 on the previous page. Hence for some $\hat{k} \le k + K \le 2K$, $g(x_{(\hat{k})}) \le \epsilon$, since otherwise $h(x_{(2K)}) < 0$. □

2.4 Existence via a Probabilistic Argument

A probabilistic proof can be given for the existence of a sparse x, that has K nonzero entries, such that $f(x)$ is not far from $f(x_*)$. (Not a coreset, however.) For simplicity, consider here only the quadratic case $f(x) = a + x^T b + x^T M x$, with M negative semidefinite as usual. Let $Y \in \mathbb{R}^d$ be a random variable defined as follows: a coordinate i is chosen with probability x_i^* (here writing x^* for the optimum), and the i'th coordinate $Y_i := 1/K$, and all other coordinates are zero. Consider a collection Y^k, $k = 1 \ldots K$, of such random variables, independently distributed. Then for $Z := \sum_k Y^k \in S$, $EZ = x^*$, and at most K coordinates of Z are nonzero. Also, for $i \neq j$,

$$\begin{aligned}
E[Z_i Z_j] &= E[\sum_k Y_i^k \sum_{k'} Y_j^{k'}] \\
&= \sum_{k \neq k'} E[Y_i^k Y_j^{k'}] + \sum_k E[Y_i^k Y_j^k] \\
&= \sum_{k \neq k'} E[Y_i^k] E[Y_j^{k'}] + 0 \\
&= (K^2 - K) x_i^* x_j^* / K^2 = x_i^* x_j^* (1 - 1/K),
\end{aligned}$$

and

$$\begin{aligned}
E[Z_i^2] &= E[\sum_k Y_i^k \sum_{k'} Y_i^{k'}] \\
&= \sum_k x_i^* / K^2 + \sum_{k \neq k'} (x_i^*)^2 / K^2 \\
&= x_i^* / K + (x_i^*)^2 (1 - 1/K),
\end{aligned}$$

and so

$$E[Z^T M Z] = \sum_{i,j} m_{ij} E[Z_i Z_j] = (x^*)^T M x^* (1 - 1/K) + \sum_i m_{ii} x_i^* / K.$$

Using these observations, and with as usual $M = -A^T A$ where A has columns p_i,

$$\begin{aligned}
a + (x^*)^T b + (x^*)^T M x^* &- E[a + Z^T b + Z^T M Z] \\
&= (x^*)^T M x^* - E[Z^T M Z] \\
&= (x^*)^T M x^* - [(x^*)^T M x^* (1 - 1/K) + \sum_i m_{ii} x_i^* / K] \\
&= (x^*)^T M x^* / K + \sum_i p_i^2 x_i^* / K \\
&\leq \sum_i p_i^2 x_i^* / K \qquad (19)
\end{aligned}$$

Since $f(x^*) - Ef(Z)$ is bounded in this way, there must exist some z with K nonzero entries that meets this bound. The above satisfies

$$\sum_i p_i^2 x_i^*/K \leq [\max_i p_i^2] \sum_i x_i^*/K = \max_i p_i^2/K. \tag{20}$$

Any quadratic problem can be translated, without loss of generality for this construction, so that the origin is the center of the MEB of the p_i, implying $\max_i p_i^2 = \text{radius}(AS)^2$, and so $\text{radius}(AS)^2/K$ is a bound on the additive error. As mentioned, for the special case of convex approximation, such a bound was shown by Maurey[Pis81] in a similar way. The bound (20) is slightly sharper, with respect to the constant, than the bounds shown for Algorithm 1.1 and Algorithm 1.2. For some problems, it could be that (19) on the previous page, the average of the p_i^2 as weighted by x_i^*, gives a better bound than simply using $\max_i p_i^2$. A specific useful example of this is given below; heuristically, this implies that sparse solutions exist that are significantly better than the greedy constructions here can find. However, for MEB the x_i are nonzero only for the p_i with largest p_i^2, and so $\sum_i p_i^2 x_i^*$ for MEB will be $\text{radius}(AS)^2$, implying that the greedy algorithms do pretty well for MEB.

As an example of a problem where (19) on the preceding page can be helpful, consider the quadratic problem where the objective function is

$$\begin{aligned} -\sum_i (p_i - Ax)^2/n &= -\left[\sum_i p_i^2/n - 2x^T A^T p_i/n + x^T A^T Ax\right] \\ &= -\left[\left[\sum_i p_i^2/n\right] - 2x^T A^T A\mathbf{e}/n + x^T A^T Ax\right], \end{aligned}$$

for which the optimum x is $\mathbf{e}/n$. That is, the problem is to minimize the sum of squares of Euclidean distances to the p_i, and the solution is the center of mass, the coordinate-wise mean. The additive error, from (19) on the previous page, is $\sum_i p_i^2/nK$. We can assume, without loss of generality, that $\sum_i p_i = 0$, and so the optimum value of the objective function is $-\sum_i p_i^2/n$, and the *relative* error is $1/K$. This observation is due to Inaba *et al.* [IKI94], and has been applied to k-means clustering. The probabilistic choice in the above construction is for this problem simply a uniform random sample of the p_i.

Is there a deterministic construction that finds a sample of comparable quality?

3 Variations

A few variations of Algorithm 1.1 or Algorithm 1.2 suggest themselves. For one, since $h(x_{(k)}) \leq 1/(k+3)$ by Theorem 2.2 on page 14, and the optimal α minimizing (16) is $2g(x_{(k)}) \leq 2h(x_{(k)})$, we might avoid searching for α simply by using $\alpha_k := 2/(k+3)$ at step k.

With this multiplier,

$$h(x_{(k+1)}) \leq h(x_{(k)}) - \alpha_k h(x_{(k)}) + \alpha_k^2/4,$$

by (16) on page 14, and using $g(x) \geq h(x)$. Since for $\alpha_k < 1$ the right-hand side is increasing in $h(x_{(k)})$, it is maximized when $h(x_{(k)})$ is at its maximum, which inductively is no more than $1/(k+3)$. We have

$$\begin{aligned} h(x_{(k+1)}) &\leq \frac{1}{k+3} - \frac{2}{k+3}\frac{1}{k+3} + \frac{1}{4}\left(\frac{2}{k+3}\right)^2 \\ &= \frac{1}{k+4} - \frac{1}{(k+4)(k+3)^2} \\ &< 1/(k+4). \end{aligned}$$

Thus searching for α is not necessary for the bounds to hold. Note that with this approach, the function $f(x)$ need only be evaluated n times at the beginning, to determine $x_{(0)}$; only gradient evaluations are needed thereafter.

Such lazier algorithms have already been proposed for special cases, for example by Li and Barron [LB00], and by Bădoiu and Clarkson [BC03]. (It may only be coincidental that subgradient and stochastic gradient algorithms also often are described using a step size that decreases as $1/k$.)

The primal-dual approximation bounds of Theorem 2.3 on page 14 can also be attained using a fixed α_k sequence: choose $\alpha_k = 2/(k+3)$ as before, for $k \leq K := \lceil 1/\epsilon \rceil$, and then use $\alpha_k = 2\epsilon$ for $k > K$. If $g(x_{(k)} \geq \epsilon$, for all $k = K \ldots 2K$, then $h(x_{(2K)}) < 0$, since $h(x_{(k+1)}) \leq h(x_{(k)}) - \epsilon^2$ for $k > K$. Therefore some $k \leq 2K$ must have $g(x_{(k)}) < \epsilon$.

Another variation is to work harder within the current face: that is, optimize the coordinates that are currently nonzero, before making another coordinate nonzero. For example, the sparse greedy algorithm itself could be used for several "minor" steps, choosing among the current set of nonzero coordinates, obtaining a solution that is close to optimal, subject to the restriction of being on the current k-face. Heuristically, this suggests that more "bang for the buck" would be obtained at each major step, but improvements in provable bounds don't seem to have been obtained. Such a harder-working algorithm is helpful, however, for showing the existence of coresets, as discussed next. (This variant algorithm is not far from orthogonal matching pursuit, as discussed in § 8 on page 30.)

4 Coresets

The term "coreset" has many different technical meanings; the most natural definition for this paper encompasses some, but not all of them. To give that definition, first we need to give notation for some restricted versions of the primal and dual problems.

As in § 1.2 on page 6, for $N \subset \{1, 2, \ldots n\}$ let S_N denote the face of S that is the convex hull of $\{e(i) \mid i \in N\}$. Then the Wolfe dual of the restricted primal problem

$$\max\{f(x) \mid x \in S_N\}$$

is $\min_x w_N(x)$, where

$$w_N(x) := z_N(x) + f(x) - x^T \nabla f(x),$$

Algorithm 4.2. *Given concave function $f(x)$, $\epsilon > 0$:*

> *Let $i' := \arg\max_i f(e(i))$ and $N_0 := \{i'\}$;*
>
> *For $k = 0 \ldots \infty$, find N_{k+1} from N_k as follows:*
>
> > *if $g(x_{N_k}) < \epsilon$ return x_{N_k};*
> >
> > *$i' := \arg\max_i \nabla f(x_{N_k})_i$;*
> >
> > *$N_{k+1} := N_k \cup \{i'\}$;*

Figure 4. Finding a coreset for $f(x)$

and $z_N(x) := \max_{i\in N} \nabla f(x)_i$. We let x_N denote the optimum for the restricted primal; this is also the optimum for the restricted dual.

The dual of the restricted primal has fewer constraints than the dual to the full problem, and so $w_N(x) \le w(x)$, and in general $w_N(x) \le w_{N'}(x)$ for $N \subset N'$. This is consistent with an analogous relation for the primal problem: the more restricted the domain over which f is maximized, the smaller that maximum can be.

Theorem 2.3 on page 14 shows that Algorithm 1.1 can be used to find a point x' with few nonzero entries, and with primal and dual values close to optimal. That is, x' is a kind of sparse certificate regarding the optimum value of the optimization problem. A coreset is combinatorial version of such a certificate.

Definition 4.1. *Given a concave function $f(x)$, an ϵ-coreset for the problem $\max_{x\in S} f(x)$ is a subset N of the coordinate indices so that $w(x_N) - f(x_N) \le 2\epsilon C_f^*$.*

(Note that the asymptotic nonlinearity C_f^* is used, not the global nonlinearity C_f. This, with the factor of 2, implies that an ϵ-coreset for MEB by this definition is a $(1/(1+1/\epsilon))$-coreset by prior definitions [BC], that is, nearly the same. As discussed in § 2.2 on page 11, $C_f^* \le C_f \le 4C_f^*$ for quadratic problems, and for the MEB problem, C_f^* is equal to the square of the radius of the MEB.)

It might be thought that the set N of indices of the nonzero coordinates of a sparse primal/dual approximate solution x' is a coreset. This is not necessarily true, as it may be that $w(x_N) \gg w(x')$. A patch for this is to use a variant of Algorithm 1.1, for which each iterate $x_{(k)}$ is the optimum for its corresponding k-face of S. This ensures that x_N is a "canonical" approximate solution, determined by N.

The variant algorithm is shown in figure 4. Here is a theorem whose proof uses it.

Theorem 4.3. *Let $f(x)$ be a concave function. Algorithm 4.2 returns a set N_k indexing a $(2\epsilon C_f/C_f^*)$-coreset, for $k \le 2K$, where $K := \lceil 1/\epsilon \rceil$. This requires $O(ndK + KQ(f,K))$ time, assuming evaluation of $\nabla f(x)$ needs $O(nd)$ time, for some d, and where where $Q(f,K)$ is the time needed to find the restricted maxima x_{N_k} for $k \le 2K$. The returned set is a $\epsilon(2+o(1))$-coreset, as $\epsilon \to 0$, assuming $C_f(S_\gamma) = (1+o(1))C_f^*$ as $\gamma \to 0$.*

Proof. Algorithm 4.2 on the previous page computes an iterate x_{N_k} such that $f(x_{N_k})$ is at least as large as that for Algorithm 1.2 on page 8, and hence also for Algorithm 1.1. Since Algorithm 4.2 uses the same choice of new coordinate i' as Algorithm 1.1, it follows that Theorem 2.2 on page 14 and Theorem 2.3 on page 14 apply to it also, so that some N_k for $k \leq 2K$ has $g(x_{N_k}) \leq \epsilon$. Since each iterate x_{N_k} of Algorithm 4.2 is an optimum for N_k, it follows that the N_k with $g(x_{N_k}) \leq \epsilon$ is a $(2\epsilon C_f/C_f^*)$-coreset, using the definitions (5) on page 5 and Definition 4.1 on the preceding page of $g()$ and coresets. The time bound follows from Theorem 2.3 on page 14 and the structure of Algorithm 4.2. The last statement uses the convergence of x_{N_k} and the asymptotic relation discussed in § 2.2 on page 11. □

Algorithm 1.1 and Algorithm 1.2 can also be used to create coresets in the following more direct way, but the bounds are worse.

Theorem 4.4. *After K^2 iterations of Algorithm 1.1 (or Algorithm 1.2 on page 8), where $K := 1/\epsilon$, the coordinate set N_{K^2} indexing nonzero entries of $x_{(K^2)}$ corresponds to a $(2\epsilon C_f/C_f^*)$-coreset.*

Proof. From Theorem 2.1 on page 13, since $h(x) \geq 0$ for all x, it must hold that $g(x) \leq \sqrt{h(x)}$ for all $x \in S$. Moreover, by definition $f(x_{N_k}) \geq f(x_{(k)})$, and so $h(x_{N_k}) \leq h(x_{(k)})$. (Here again, x_{N_k} denotes the optimum in S_{N_k}.) Since by Theorem 2.2 on page 14, the value $h(x_{(K^2)} \leq 1/(K+3)^2$, it follows that

$$g(x_{N_{K^2}}) \leq \sqrt{h(x_{N_{K^2}})} \leq \sqrt{h(x_{(K^2)})} \leq 1/(K+3) < \epsilon,$$

and so N_{K^2} is an $(4\epsilon C_f/C_f^*)$-coreset, using the definitions (5) on page 5 and Definition 4.1 on the previous page of $g()$ and coresets. □

4.1 Approximation and Coresets Within Convex Hulls

The main case of interest for the above coreset results is for functions $f(x)$ that have the form $f(x) = \hat{f}(Ax)$, where A is a $d \times n$ matrix for some d, and $\hat{f} : \mathbb{R}^d \to \mathbb{R}$. Here a subset N of the indices corresponds to a subset of the columns of A; as discussed below, for the MEB problem, the columns of A correspond to the input points, so a coreset for MEB is a subset of the input points.

For functions of the form $f(x) = \hat{f}(Ax)$, the gradient

$$\nabla f(x) = A^T \nabla \hat{f}(Ax),$$

so that

$$\begin{aligned} f(x) - x^T \nabla f(x) &= \hat{f}(Ax) - x^T (A^T \nabla \hat{f}(Ax)) \\ &= \hat{f}(Ax) - x^T A^T \nabla \hat{f}(Ax) \\ &= \hat{f}(c) - c^T \nabla \hat{f}(c), \end{aligned}$$

for $c := Ax$. The dual problem (8) becomes

$$\begin{aligned} &\min_{z\in\mathbb{R}, c\in\mathbb{R}^d} \ z + \hat{f}(c) - c^T\nabla\hat{f}(c) \\ &\text{subject to } z\mathbf{e} \geq A^T\nabla\hat{f}(c)). \end{aligned} \tag{21}$$

We can write this as $\min_c \hat{w}(c)$, with $\hat{w}(c) := \hat{z}(c) + \hat{f}(c) - c^T\nabla\hat{f}(c)$, and $\hat{z}(c) := \max_i(A^T\nabla\hat{f}(c))_i$. Note that $\hat{w}(Ax) = w(x)$, just as $\hat{f}(Ax) = f(x)$.

The above results imply approximation results for $\hat{f}$, over $AS := \{Ax \mid x \in S\}$, that is, the convex hull of the columns of A.

The following is a corollary of Theorem 2.2 on page 14, and is the main part of Theorem 3.1 of [Zha03].

Theorem 4.5. *If $f(x) = \hat{f}(Ax)$, where $\hat{f} : \mathbb{R}^d \to \mathbb{R}$ is a twice differentiable concave function, then with $C_f = -\frac{1}{2}\sup_{\substack{a,b\in AS \\ \tilde{a}\in[a,b]}}(b-a)^T\nabla^2\hat{f}(\tilde{a})(b-a)$, there is a k-face S' of S and a point $a \in AS'$ such that*

$$\hat{f}(a) - \inf_{b\in AS}\hat{f}(b) \leq 4C_f/(k+3).$$

Proof. The function $f(x) := \hat{f}(Ax)$ is concave if $\hat{f}$ is, and the theorem follows from Theorem 2.2 on page 14 and the bound (11) on page 12 for C_f. $\square$

4.2 Quadratic Functions and Coresets for MEB

If $f(x)$ is a quadratic concave function, that is, has the form

$$f(x) = a + x^Tb + x^TMx, \tag{22}$$

where $a \in \mathbb{R}$, $b \in \mathbb{R}^n$, and M is a negative semidefinite $n \times n$ matrix, then $f(x) = \hat{f}(Ax)$, where:

$$\begin{aligned} \hat{f}(c) &:= a + c_d - \tilde{c}^T\tilde{c} \\ \tilde{c} &:= [c_1, \ldots, c_{d-1}, 0]^T \text{ (a column vector)} \\ A &:= \begin{bmatrix} \tilde{A} \\ b^T \end{bmatrix} \\ M &= -\tilde{A}^T\tilde{A} \\ \tilde{A} &\text{ is } (d-1)\times n, \text{ for some } d. \end{aligned}$$

Note that such a $\tilde{A}$ can always be found, and the given $\hat{f}()$ is concave.

Thus coresets for quadratic concave functions correspond to columns of A, and give bounds also for the dual problem

$$\begin{aligned} &\min_{z\in\mathbb{R}, x\in\mathbb{R}^d} \ z - x^TMx \\ &\text{subject to } z \geq \max_i(b + 2Mx)_i. \end{aligned} \tag{23}$$

As discussed in § 2.2 on page 11, the nonlinearity measure C_f is bounded by the squared diameter of AS, which is bounded by the squared diameter of the MEB of the columns of A, and when f corresponds to the MEB problem, C_f^* is the squared radius of the MEB. Thus Theorem 4.3 on page 18 implies that an ϵ-coreset exists for MEB, of size close to $4/\epsilon$.

It was claimed near Definition 4.1 on page 18 that the general definition of an ϵ-coreset specializes for MEB to something close to the standard definitions for MEB. For example, one definition ("alternate" in [BC]) is that $r(c_N) \le r_N(c_N)/(1-\epsilon)$, where $r(c_N) = \sqrt{\hat{w}(c_N)}$ is the distance of c_N to the farthest input point p_i, and $r_N(c_N) = \sqrt{\hat{w}_N(c_N)}$ is the distance of c_N to the farthest point indexed by N; that is, if the smallest ball containing the points indexed by N is expanded by a factor of $1/(1-\epsilon)$, the resulting ball contains all the points.

To show this relation: suppose N is an ϵ-coreset for MEB, by Definition 4.1 on page 18. Then $w(c_N) - w_N(c_N) = w(c_N) - f(c_N) \le 2\epsilon C_f^*$. Since here $C_f^* = w(c_*) = r(c_*)^2$, we have

$$r(c_N)^2 - r_N(c_N)^2 \le 2\epsilon r(c_*)^2 \le 2\epsilon r(c_N),$$

or

$$r(c_N)^2 \le r_N(c_N)^2/(1-2\epsilon),$$

or

$$r(c_N) \le r_N(c_N)(1+o(1))/(1-\epsilon),$$

as $\epsilon \to 0$, since

$$\begin{aligned}\frac{1}{1-2\epsilon} &= \frac{1}{(1-\epsilon)^2}\left(1+\frac{\epsilon^2}{1-2\epsilon}\right)\\ &\le \frac{1}{(1-\epsilon)^2}\left(1+\frac{\epsilon^2}{2(1-2\epsilon)}\right)^2.\end{aligned}$$

5 Coresets Via "Away" Steps

The algorithms presented so far have been monotone, in the sense that once a coordinate of x becomes nonzero, it remains so, or at least, is not specifically made to be zero again. However, a few known algorithms have been non-monotone: an algorithm for optimal MEB coreset construction [BC] preserves the number of non-zero entries: the number of non-zero entries is some K, and in a single step, a new coordinate is made non-zero, and then another coordinate is set to zero, preserving the number of non-zero coordinates. It is shown that progress can be made in this way, by showing that a quantity similar to $g(x)$ is large enough.

Another algorithm, by Todd and Yildirm [TY05], discusses a version of Algorithm 1.1 with an additional possible "away" step: it considers whether reducing a non-zero variable, not necessarily to zero, might improve the objective function. If so, the reduction is done. No provable improvement is shown, however, by including such a step.

Algorithm 5.1. *For concave $f(x)$ and $\epsilon > 0$:*

Let $i' := \arg\max_i f(e(i))$ *and* $N := \{i'\}$*;*

Repeatedly update N as follows:

if $g(x_N) \leq \epsilon/2$ *return* x_N*;*
$i' := \arg\max_i \nabla f(x_N)_i$*;*
$N := N^* := N \cup \{i'\}$*;*
if $|N^*| > \lceil 1/\epsilon \rceil$*:*
$x := x_{N^*}$*;*
$i'' := \arg\min_{i \in N^*} x_i$*;*
$N := N^{**} := N^* \setminus \{i''\}$*;*

Figure 5. A coreset algorithm with "away" steps

It is shown next that a constant-factor improvement in provable coreset size is obtainable, by including a step that reduces a non-zero variable to zero. The improvement is almost as sharp as the optimal MEB coreset result[BC], and gives an improvement over Theorem 4.3 on page 18 in the general setting.

The algorithm, shown in figure 5, is simply Algorithm 4.2 on page 18, with an additional possible "away" step, where a coordinate is set to zero. Since the points $x_{(k)}$ are always the optimum points x_N of some simplex S_N, with vertices $\{e(j) \mid j \in N\}$, the algorithm is given in terms of maintenance of the set N. The sets N^* and N^{**} are defined only to aid discussion of N at different points in the algorithm.

For $|N| \leq \lceil 1/\epsilon \rceil$, the algorithm proceeds as in Algorithm 4.2 on page 18. When $|N| > \lceil 1/\epsilon \rceil$, the "away" step finds an index i'' to remove from N. Note that before finding i'', the optimum for the current N is found; this is helpful in the analysis.

Theorem 5.2. *For $\epsilon > 0$ and concave $f(x)$, Algorithm 5.1 returns an $(\epsilon C_f / C_f^*)$-coreset. The returned set is an $\epsilon(1+o(1))$-coreset, as $\epsilon \to 0$, assuming $C_f(S_\gamma) = (1 + o(1))C_f^*$ as $\gamma \to 0$.*

Proof. Since Algorithm 5.1 exits with a similar termination condition as Algorithm 4.2 on page 18, similar conditions as in Theorem 4.3 on page 18 hold for the index set N that it returns. It remains to show that the algorithm terminates.

Theorem 2.1 on page 13 shows that addition of i' to N results in $x = x_{N^*}$ with

$$h(x) \leq h(x_N) - g(x_N)^2, \tag{24}$$

for $|N| \leq K$, so inductively $h(x_{N^*}) \leq 1/|N^*|$ for $|N^*| \leq K$.

When $|N^*| > K$, the "away" step is also done. Since $f(x_{N^{**}}) \geq f(y)$ for any $y \in S_{N^{**}}$, this inequality holds in particular when $y = x - \alpha''(e(i'') - x)$, where $\alpha'' := x_{i''}/(1 - x_{i''})$.

For $x = x_{N^*}$ as in the algorithm, since x has at least $K+1$ nonzero entries, $\sum_j x_j = 1$, and $x \geq 0$, it must hold that $x_{i''} \leq 1/(K+1)$, and so

$$\alpha'' = \frac{x_{i''}}{1 - x_{i''}} \leq \frac{1/(K+1)}{1 - 1/(K+1)} = 1/K. \tag{25}$$

Also, since $x = x_{N^*}$ is optimal for N^*, its duality gap is zero:

$$0 = w_{N^*}(x) - f(x) = z_{N^*}(x) - x^T \nabla f(x) = \max_{i \in N^*} \nabla f(x)_i - x^T \nabla f(x).$$

Since $x^T \nabla f(x)$ is a convex combination of the coordinates of $\nabla f(x)$, this can be true only if all coordinates $\nabla f(x)_i$ with $i \in N^*$ must be equal to $\max_{i \in N^*} \nabla f(x)_i = x^T \nabla f(x)$. In particular,

$$e(i'')^T \nabla f(x) = \nabla f(x)_{i''} = x^T \nabla f(x). \tag{26}$$

Thus

$$\begin{aligned}
f(x_{N^{**}}) &\geq f(x - \alpha''(e(i'') - x)) & & \\
&\geq f(x) - \alpha''(e(i'') - x)^T \nabla f(x) - (\alpha'')^2 C_f & & \text{def. of } C_f \\
&= f(x) - (\alpha'')^2 C_f & & \text{by (26)} \\
&\geq f(x) - C_f / K^2. & & \text{by (25)}
\end{aligned}$$

Putting together this relation and (24) on the previous page, and referring to N at the beginning of the loop:

$$\begin{aligned}
h(x_{N^{**}}) &\leq h(x) + 1/4K^2 & & \text{def. of } h\text{, and above} \\
&\leq h(x_N) - g(x_N)^2 + 1/4K^2 & & \text{by (24)} \\
&< h(x_N) - \epsilon^2/4 + 1/4K^2, & & \text{by test on } g(x_N). \\
&\leq h(x_N), & & K > 1/\epsilon.
\end{aligned}$$

and so an iteration reduces the value of $h()$. Since the value of $h(x_N)$ (at the beginning of each iteration) is decreasing, any given set N is seen only once, and so eventually the loop terminates with $g(x_N) \leq \epsilon/2$, and N of size at most K specifies a coreset with the properties as claimed. □

As remarked in § 1, as applied to MEB this result is best possible in the leading term [BC].

6 Tail Estimates for Random Data

In the case of optimizing within a convex hull, so $f(x) = \hat{f}(Ax)$ for a concave function $\hat{f}()$ and $d \times n$ matrix A, suppose the columns p_i, $i = 1 \ldots n$ of A are random variables, independently and identically distributed. Consider an ϵ-coreset N, so that

$$w(x_N) - f(x_N) = z(x_N) - x_N^T \nabla f(x_N) \leq 2\epsilon C_f^*.$$

Since $z(x_N) \geq \nabla f(x_N)_i = p_i^T \nabla \hat{f}(Ax_N)$ for all i, this condition implies that

$$p_i^T \nabla \hat{f}(Ax_N) \leq x_N^T \nabla f(x_N) + 2\epsilon C_f^*,$$

that is, all the p_i lie in a particular halfspace with normal vector $\nabla \hat{f}(Ax_N)$. Let

$$H(x) := \{p \in \mathbb{R}^d \mid p^T \nabla \hat{f}(Ax) \leq x^T \nabla f(x) + 2\epsilon C_f^*\},$$

so that all the p_i lie in $H(x_N)$. For any fixed $x \in \mathbb{R}^n$, the fact that all the $p_i \in H(x)$ would suggest that $H(x)$ contains most of the probability mass of the distribution of the p_i: if that mass is $1 - m$, then the probability is $(1 - m)^n \leq \exp(-mn)$ that all the points p_i appear in $H(x)$. So, unless the mass m in the complement of $H(x)$ is small, it is unlikely that all p_i will be in $H(x)$. Similarly, for any given choice of N, and letting $s := |N|$, the probability is $(1 - m)^{n-s} \leq \exp(-m(n - s))$ that all $p_i \in H(x_N)$, for all $i \notin N$, and when $H(x_N) \geq 1 - m$. Since there are only $\binom{n}{s}$ possible N, we almost have the following theorem.

Theorem 6.1. *For $f(x) = \hat{f}(Ax)$, where the columns of A are i.i.d., for a coreset N of size s, with probability $1-\delta$ the probability mass in the complement of $H(x_N)$ is no more than $(\log(1/\delta) + s\log(ne/s))/(n - s)$.*

Proof. We use the union bound, so that the probability of failure at most $\binom{n}{s}(1-m)^{n-s}$, together with the bounds $\binom{n}{s} \leq (ne/s)^s$ and as above $(1 - m)^{n-s} \leq \exp(-m(n - s))$. It is thus enough to pick a value of m large enough that $(ne/s)^s \exp(-m(n - s)) \leq \delta$, which on solving for m yields the bound of the theorem. □

A similar result, in a setting where regions are "defined" by a small number of objects, implies the existence of probabilistic algorithms for a variety of geometric problems [Cla88, Cla87], and is called the *compression lemma* in the learning theory literature[LW86, FW95].

7 Specific Cases

The following subsections discuss some specific applications; the discussions of convex approximation, Adaboost, and L_v approximation follow Zhang[Zha03], and serve simply to translate results of that paper for Algorithm 1.2 into results for Algorithm 1.1 in the notation of this paper. The discussion of kernel methods follows that of Tsang *et al.* [TKC05] to some degree.

7.1 Convex Approximation

For suitable matrix A and point p, consider the primal problem (1) where $f(x) := \hat{f}(c) := -(c - p)^2$ with $c = Ax$; that is, the problem is to find the convex combination of the columns of A of minimum distance to p. Via (21) on page 20, and using $\nabla \hat{f}(c) = -2(c - p)$, the dual objective function is

$$z + \hat{f}(c) - c^T \nabla \hat{f}(c) = z - (c - p)^2 + 2c^T(c - p) = z + c^2 - p^2$$

so the dual problem is

$$(27)\qquad \begin{aligned} &\min_{z\in\mathbb{R}, c\in\mathbb{R}^d} \; z + c^2 - p^2 \\ &\text{subject to } z \geq \max_i -2[A^T(c-p)]_i. \end{aligned}$$

Although the above considers only the finite-dimensional case, and not more general functional approximation, as discussed by Zhang [Zha03], the barriers to such a generalization are not enormous.

As shown by Zhang[Zha03], and also discussed in § 2.2 on page 11, the measure C_f here is the square of the diameter of the MEB of the columns of A, and the asymptotic version C_f^* is the square of the radius. It is clear that 1.1 as well as Algorithm 1.2 on page 8 can be used to obtain an approximate solution, and that Algorithm 5.1 on page 22 can be used to show that coresets exist.

In contrast to some other problems considered in this paper, here the main problem of interest is the primal problem, and the nature and significance of coresets for the dual problem is perhaps mysterious: what do they mean here, and what are they good for?

Here is a geometric interpretation. The corresponding optimum point c_N has a small duality gap: the gap is $\hat{z}(c_N) + c_N^2 - p^2 - (-(p-c_N)^2) \leq 4\epsilon \operatorname{diam}(AS)^2$, or

$$\begin{aligned} 4\epsilon \operatorname{diam}(AS)^2 &\geq (\max_i -2[A^T(c_N - p)]_i) + 2c_N^2 - 2c_N^T p \\ &= 2[c_N^T(c_N - p) + \max_i -p_i^T(c_N - p)] \\ &= 2[(c_N - p)^T(c_N - p) + \max_i -(p_i - p)^T(c_N - p))] \\ &= 2[(c_N - p)^2 - \min_i (p_i - p)^T(c_N - p))] \end{aligned}$$

or

$$(p_i - p)^T(c_N - p)) \geq (c_N - p)^2 - 2\epsilon \operatorname{diam}(AS)^2,$$

for all i, or $(p_i - c_N)^T(c_N - p) \geq -2\epsilon \operatorname{diam}(AS)^2$. The boundary of the halfspace $H(c_N, 0) := \{q \mid (q - c_N)^T(c_N - p) \geq 0\}$ is the hyperplane passing through c_N, and normal to $c_N - p$. The given halfspace is on the side away from p. The points p_i thus are all in a halfspace $H(c_N, \beta)$, bounded by a hyperplane parallel to $H(c_N, 0)$, but translated by $\beta(c_N - p)$, where

$$\beta := -2\epsilon \operatorname{diam}(AS)^2/(c_N - p)^2 = 2\epsilon \operatorname{diam}(AS)^2/\hat{f}(c_N).$$

That is, c_N provides an immediate proof that the p_i cannot have a convex combination much closer to p than c_N is. A use of MEB coresets is the detection of outliers, the densest-ball problem discussed in § 1.2 on page 6; here, they could be used to detect inliers. That is, suppose we want to find that 10% of the points p_i such that deleting them makes the solution as much worse as possible, that is, decreases $f(x)$ the most. We could simply try all $\binom{n}{n/10}$ subsets, checking f after removing each, but this is very slow. An approximate solution, for some given $\epsilon > 0$, would be to try all subsets N' of size $1/\epsilon$, checking if at least 10% of the points p_i are not in $H(c_{N'})$, choosing the $c_{N'}$ that has minimum distance to p among all such $c_{N'}$.

7.2 Kernel Methods

Kernel methods, and in particular *support vector machines* (SVM), are a popular approach to classification, regression, and outlier detection, and training them is an optimization problem that can be solved using the methods discussed here. For example, for *hard margin SVM*, a dataset comprising $p_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$, for $i = 1 \ldots n$, is given, and the training problem is to find

$$\tag{28} \begin{aligned} &\min_{c \in \mathbb{R}^d, \rho \in \mathbb{R}} \rho + c^T c/2 \\ &\text{subject to } \rho \geq -y_i c^T p_i, i = 1 \ldots n, \end{aligned}$$

if this problem is feasible. If $\rho < 0$ and vector c are feasible, then they specify a hyperplane $\{x \mid c_*^T x = 0\}$, that separates the p_i with $y_i = +1$ from those with $y_i = -1$. The problem is to make the margin, which is the minimum distance $G(c) := -\rho/\|c\|$ of any point p_i to that hyperplane, as large as possible. This problem is a scaled version of the convex approximation of § 7.1 on page 24, with $p = 0$.

While many formulations of this problem fix ρ and minimize $c^T c$, or fix $c^T c = 1$ and maximize ρ, the same margin is obtained in the formulation here: for any given feasible pair c, ρ, and value $\beta > 0$, the pair $\beta c, \beta\rho$ is also feasible, and the value of β that minimizes $\beta\rho + \beta^2 c^T c/2$ is $\tilde{\beta} = -\rho/c^T c$, yielding the β-scaling-invariant $\tilde{\beta}\rho + \tilde{\beta}^2 c^T c/2 = -\rho^2/2c^T c$, proportional to the negative square of the margin. That is, the optimal value of the training problem is the one half the negative square of the margin.

The dual to (28) is

$$\tag{29} \begin{aligned} &\max_{x \in \mathbb{R}^n} - x^T A^T A x/2 \\ &\text{subject to } x \in S, \end{aligned}$$

where A is the $d \times n$ matrix whose columns are the vectors $y_i p_i$. (This can be read off from (27) on the previous page, for example.) That is, training SVM, for this version, is a special case of convex approximation: finding $x \in S$ so that the vector Ax is as close to the origin as possible.

(Indeed, at some cost in n, it is well-known that a broader class of SVM training problems are dual to a problem of the same form: use a $d \times n'$ matrix A', whose columns are all those possible of the form $p_i - p_j$, where $y_i = 1$ and $p_j = -1$. The polytope $A'S$ is the Minkowski difference of the convex hulls of the "+" points and the "−" points, and the length of the shortest vector in $A'S$ is the minimum distance between those two polytopes. Although n' may be as large as $n^2/4$, this does not affect results regarding coreset sizes; also, other formulations do not involve such a blow-up in A.)

This training problem for SVM fits in the framework here, both for algorithms and for coresets. Of particular importance for SVM, the algorithms depend on d, and the data points p_i, only through the need to evaluate $A^T c$, or equivalently, the dot products $c^T p_i$. As is well-known, such dot products can sometimes be evaluated efficiently even when d is very large or infinite.

As for other quadratic problems, the quantity C_f is at most $\operatorname{diam}(AS)^2/2$ (with the division by two due to the scaling). By Theorem 2.3 on page 14, Algorithms 1.1 and 1.2 find $x_{(\hat{k})}$, and corresponding $c = Ax$ and $\rho = \max_i -y_i c^T p_i$, with $\hat{k} \leq 2/\epsilon$, such that, with $x := x_{(\hat{k})}$ and $G(x)$ the margin of separation,

$$-G(x)^2/2 \leq -G(x_*)^2/2 + 4\epsilon \operatorname{diam}(AS)^2/2,$$

or

$$G(x)^2 \geq G(x_*)^2(1 - 4\epsilon \operatorname{diam}(AS)^2/G(x_*)^2),$$

or

$$G(x) \geq G(x_*)(1 - 3\epsilon \operatorname{diam}(AS)^2/G(x_*)^2)$$

for $\epsilon \operatorname{diam}(AS)^2/G(x_*)^2 \leq 2/9$.

A similar argument using Theorem 5.2 on page 22 shows that there is a set of points indexed by N, of size $(1 + o(1))/\epsilon$, so that

$$G(x_N) \geq G(x_*)(1 - \epsilon \operatorname{diam}(AS)^2/G(x_*)^2)$$

as $\epsilon \to 0$. The previous bound on the size was $64/\epsilon$ [HPRZ07].

Putting this condition together with Theorem 6.1 on page 24, we can say the following: if there is coreset N with $G(x_N) > 0$ and if the points p_i are i.i.d., then with probability $1 - \delta$, the probability mass of the region $H(c_N) := \{p \in \mathbb{R}^d \mid \rho < -y_i c_N^T p\}$ is at most $\log(1/\delta) + \frac{s}{n-s} \log(\frac{ne}{s})$, where $s := |N|$ need be no more than $(1 + o(1)) \operatorname{diam}(AS)^2/G(x_*)^2$. Since for a classifier based on x_N, only the points in $H(c_N)$ can be misclassified, this gives an upper bound on the error probability of such a classifier. This result is similar to that derived by Graepel *et al.* for perceptrons [GHW00]. This is not surprising: the perceptron algorithm is a greedy procedure akin to Algorithm 1.1, and Novikoff's mistake bound (see [GHW00] for example) implies the existence of sparse solutions to an optimization problem.

Tsang *et al.*[TKC05] have shown that variations like hard-margin SVDD, one- and two-class L2-SVM, and L2-SVR can also be put into the framework here. As discussed in the introduction, the approximation algorithms given here do not require, as for coresets, the exact solution of small problem instances. It remains to be seen, however, whether such simplification is helpful in practice.

7.3 Adaboost

Boosting refers to the improvement of a collection of classifiers by using a linear combination of them. A collection of n classifiers is given, and d datapoints, such that classifier i gives a value $a_{ji} \in [-1, 1]$ for datapoint j, which has actual classification $r_j \in \{-1, 1\}$. The training problem is: for a given *loss function* $L(c, r)$, giving the cost making predictions c for classifications r, find x such that $L(Ax, r)$ is as small as possible. It is no loss of generality to assume that the data is symmetric about the origin, that is, for every prediction a_{ji} for r_j, there is a j^- with $a_{j^-i} = -a_{ji}$ and $r_{j^-} = -r_j$. This implies that it is enough to consider $x \geq 0$, putting the problem nearly into the framework of this paper.

The "ideal" loss function is perhaps $L(c,r) = \frac{1}{d}\sum_j I_{c_j r_j<0}$, where the indicator function $I_s = 1$ when s is true, and $I_s = 0$ otherwise. That is, there is no loss if c_j and r_j agree in sign, and a loss $1/d$ otherwise. This function is difficult to work with, however, so proxies are often used. A popular one, for *Adaboost*, is $\frac{1}{d}\sum_j \exp(-Cc_j r_j)$, where C is a tunable parameter.

The training problem is then to minimize the loss, or equivalently to maximize $f(x) := -\log\sum_j \exp(-C(Ax)_j r_j)/d$, over $x \in S$.

(Training an Adaboost classifier is an instance of *geometric programming* [BV04], often done with interior-point methods. However, it is also amenable to the simpler approach here.)

The restriction to nonnegative x is no loss of generality, as mentioned, and the restriction $x^T\mathbf{e} = 1$ simply amounts to an adjustment (or normalization) of the parameter C. With $c = Ax$, as usual, $f(x)$ can also be written as $f(x) = \hat{f}(c) = -\log\sum_j \exp(-Cc_j r_j)/d$.

The gradient of this $\hat{f}$ is, using $V_j := \exp(-Cc_j r_j)/d$,

$$\nabla\hat{f}(x)_j = \frac{V_j C r_j}{\sum_j V_j},$$

and $\nabla f(x) = A^T\nabla\hat{f}(Ax)$, as usual. The i' maximizing $\nabla f(x)_i$ is thus readily found.

To apply the results here, the value of C_f for this function $f()$ needs to be bounded. We will apply (11) on page 12, which requires a bound on $-(b-a)^T\nabla^2\hat{f}(\tilde{a})(b-a)/2$, for $a,b \in AS$ and $\tilde{a}$ on a line through a and b, and in AS.

The following lemma will be helpful.

Lemma 7.1. *Suppose function $\hat{f} : \mathbb{R}^d \to \mathbb{R}$ has the form $-\gamma(\tilde{f}(x))$, where $\tilde{f} : \mathbb{R}^d \to \mathbb{R}$, like $\hat{f}$, and $\gamma : \mathbb{R} \to \mathbb{R}$ with $\gamma''(x) \le 0$. Then for $c, \tilde{a} \in \mathbb{R}^d$,*

$$-c^T\nabla^2\hat{f}(\tilde{a})c \le \gamma'(\tilde{f}(\tilde{a}))c^T\nabla^2\tilde{f}(\tilde{a})c.$$

Proof. We have

$$\nabla\hat{f}(\tilde{a}) = \gamma'(\tilde{f}(\tilde{a}))\nabla\tilde{f}(\tilde{a}),$$

and

$$\nabla^2\hat{f}(\tilde{a}) = \gamma''(\tilde{f}(\tilde{a}))\nabla\tilde{f}(\tilde{a})\nabla\hat{f}(\tilde{a})^T + \gamma'(\tilde{f}(\tilde{a}))\nabla^2 f(\tilde{a}).$$

Thus here

$$\begin{aligned}-c^T\nabla^2\hat{f}(\tilde{a})c &= \gamma''(\tilde{f}(\tilde{a}))(c^T\nabla\tilde{f}(\tilde{a}))^2 + c^T\gamma'(\tilde{f}(\tilde{a}))\nabla^2\tilde{f}(\tilde{a})c \\ &\le \gamma'(\tilde{f}(\tilde{a}))c^T\nabla^2\tilde{f}(\tilde{a})c,\end{aligned}$$

as claimed, since the first term is always no more than zero. □

Applying this lemma to $\gamma(x) = \log x$, and $\tilde{f}(\tilde{a}) = \sum_j \exp(-C\tilde{a}_j r_j) = \sum_j V_j$, we have

$$
\begin{aligned}
-(b-a)^T \nabla^2 \hat{f}(\tilde{a})(b-a) &\leq \frac{(b-a)^T \nabla^2 \tilde{f}(\tilde{a})(b-a)}{\sum_j V_j} \\
&= \frac{(b-a)^T (C^2 \operatorname{diag}(V))(b-a)}{\sum_j V_j} \qquad \text{using } r_j^2 = 1 \\
&\leq 4C^2 \qquad \text{using } |a_{ji}| \leq 1
\end{aligned}
$$

Here $\operatorname{diag}(V)$ is the diagonal matrix with $\operatorname{diag}(V)_{jj} = V_j$.

7.4 Convex Approximation in L_v

When approximating a point p by a convex combination Ax, for $x \in S$, it may be of interest to use a different measure of distance than the Euclidean norm; a more general setting is to use L_v norms, $\|p\|_v := \left(\sum_j |p_j|^v\right)^{1/v}$. (Usually L_p norms are discussed, but this collides with our notation a bit, so here v is used for the numerical parameter instead.) The distance $\|Ax - p\|_v$ has the same minimum as the maximum of the function $f(x) := -\|Ax - p\|_v^{v'}$, where $v' := \min\{2, v\}$, and the latter functions are considered here. As usual, consider $f(x) = \hat{f}(c) = -\|c - p\|_v^{v'}$, where $c = Ax$.

For $v \geq 2$, apply Lemma 7.1 on the preceding page with $\gamma(x) = x^{2/v}$, and $\tilde{f}(\tilde{a}) = \|\tilde{a}\|_v^v$, obtaining

$$
\begin{aligned}
-(b-a)^T \nabla^2 \hat{f}(\tilde{a})(b-a) &\leq \gamma'(\tilde{f}(\tilde{a})) c^T \nabla^2 \tilde{f}(\tilde{a}) c \\
&= (2/v)(\|\tilde{a}\|_v^v)^{2/v-1} \sum_j (b-a)_j^2 [v(v-1)|c_j - p_j|^{v-2} \\
&= 2(v-1) \sum_j (b-a)_j^2 \frac{|c_j - p_j|^{v-2}}{\|\tilde{a}\|_v^{v-2}} \\
&= 2(v-1) \sum_j (b-a)_j^2 \left(\frac{|c_j - p_j|^v}{\sum_j |c_j - p_j|^v} \right)^{1-2/v} \\
&\leq 2(v-1) \operatorname{diam}(AS)^2.
\end{aligned}
$$

Thus Algorithm 1.1 can be used for L_v approximation, with similar bounds, for $\infty > v \geq 2$.

For $1 < v < 2$, unfortunately C_f cannot be found; however, an analog can be obtained, via an analog of (9) on page 11, with $1/\alpha^v$ instead of $1/\alpha^2$. This allows an analog of (18) on page 14 in which $h(x_{(k)})^2$ is replaced by $h(x_{(k)})^v$, which leads to an additive error of $O(1/\epsilon^{v-1})$ instead of $O(1/\epsilon^v)$. As shown by Donahue *et al.* [DGDS97], an incremental approach like Algorithm 1.1 or Algorithm 1.2 cannot work for the $v = 1$ case.

8 Conclusions

A related line of research is concerned with finding *best m-term approximations*, finding sparse x minimizing $(p - Ax)^2$, where $x \in \mathbb{R}^n$ need not be in S. The orthogonal matching pursuit algorithm, which is very close to Algorithm 1.1 (in particular, the "harder working" version described in § 3 on page 16), has been shown to yield x that are good *relative* approximations, compared to the best x with the same sparsity [Tro04], for suitable A. A sufficient condition for A to be suitable is that $A^T A = I + E$, where I is the identity matrix and E has entries that are all small in magnitude.

Acknowledgement. I am grateful to Kasturi Varadarajan for pointing out an error in the probabilistic proof of § 2.4, and other helpful remarks.

References

[AHPV05] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximation via coresets. Survey, 2005.

[AS06] Y. Altun and A. Smola. Unifying divergence minimization and statistical inference via convex duality. In *COLT '06: The Nineteenth Annual Conference on Learning Theory*, pages 139–153, 2006.

[AST06] D. Ahipasaoglu, P. Sun, and M. J. Todd. Linear convergence of a modified Frank-Wolfe algorithm for computing minimum volume ellipsoids. Technical Report 1452, Cornell University, 2006.

[Bar93] A. R. Barron. Univeral approximation bounds for superpositions of a sigmoidal function. *IEEE Trans. Inform. Theory*, 39(3):930–945, 1993.

[BC] M. Bădoiu and K. L. Clarkson. Optimal core-sets for balls. to appear, *Computational Geometry: Theory and Applications*.

[BC03] M. Bădoiu and K. L. Clarkson. Smaller core-sets for balls. In *SODA '03: Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2003.

[BDES02] S. Ben-David, N. Eiron, and H.-U. Simon. The computational complexity of densest region detection. *Journal of Computer and System Sciences*, 64:22–47, 2002.

[BHPI02] M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings of the 34th Symposium on Theory of Computing*, 2002.

[BJGL+03] Z. Bar-Joseph, G. K. Gerber, T. I. Lee, N. J. Rinaldi, J. Y. Yoo, F. Robert, D. B. Gordon, E. Fraenkel, T. S. Jaakkola, R. A. Young, and D. K. Gifford. Computational discovery of gene modules and regulatory networks. *Nature Biotechnology*, 21:1337–1342, 2003.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, 2004.

[Cla87] K. L. Clarkson. New applications of random sampling to computational geometry. *Discrete & Computational Geometry*, 2:195–222, 1987. Preliminary version: Further applications of random sampling to computational geometry, *STOC '86: Proceedings of the Eighteenth Annual SIGACT Symposium*, May 1986.

[Cla88] K. L. Clarkson. A randomized algorithm for closest-point queries. *SIAM Journal on Computing*, pages 830–847, 1988. Preliminary version: A probabilistic algorithm for the post office problem, *STOC '85: Proceedings of the Seventeenth Annual SIGACT Symposium*, 1985.

[CTK04] C. S. Chu, I. W. Tsang, and J. T. Kwok. Scaling up support vector data description by using core-sets. In *Proceedings of the International Joint Conference on Neural Networks*, pages 425–430, 2004.

[DGDS97] M. J. Donahue, L. Gurvits, C. Darken, and E. Sontag. Rates of convex approximation in non-Hilbert spaces. *Constructive Approximation*, 13:187–220, 1997.

[Fed72] V. V. Fedorov. *Theory of Optimal Experiments*. Academic Press, New York, 1972.

[FW56] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Res. Logist. Quart.*, 3:95110, 1956.

[FW95] S. Floyd and M. Warmuth. Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning*, 21(3):269–304, 1995.

[GHW00] T. Graepel, R. Herbrich, and R. C. Williamson. From margin to sparsity. In *NIPS*, pages 210–216, 2000.

[Haz06] E. Hazan. Approximate convex optimization by online game playing. 2006.

[HP07] S. Har-Peled. Personal communication., 2007.

[HPRZ07] S. Har-Peled, D. Roth, and D. Zimak. Maximum margin coresets for active and noise tolerant learning. In *IJCAI'07*, 2007.

[IKI94] M. Inaba, N. Katoh, and H. Imai. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering: (extended abstract). In *SCG '94: Proceedings of the tenth annual symposium on Computational geometry*, pages 332–339, New York, NY, USA, 1994. ACM Press.

[Jon92] L. K. Jones. A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training. *Annals of Statistics*, 20(1):608–613, 1992.

[Kha96] L. G. Khachiyan. Rounding of polytopes in the real number model of computation. *Math. Oper. Res.*, 21(2):307–320, 1996.

[KMY03] P. Kumar, J. Mitchell, and E. A. Yildirim. Approximate minimum enclosing balls in high dimensions using core sets, 2003.

[KY05] P. Kumar and A. Yildirim. Minimum volume enclosing ellipsoids and core sets. *Journal of Optimization Theory and Applications*, 126(1), 2005. To appear.

[LB00] J. Q. Li and A. R. Barron. Mixture density estimation. In *Advances in Neural Information Processing Systems*, volume 12, pages 279–285, 2000.

[LW86] N. Littlestone and M. Warmuth. Relating data compression and learnability, June 1986. Unpublished manuscript.

[MBBF00] L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting algorithms as gradient descent. In *NIPS 2000:Advances in Neural Information Processing System*, volume 12, 2000.

[Nem05] A. Nemirovski. Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM J. on Optimization*, 15(1):229–251, 2005.

[NN06] F. Nielsen and R. Nock. On approximating the smallest enclosing Bregman balls. In *SCG '06: Proceedings of the Twenty-Second Annual Symposium on Computational Geometry*, pages 485–486, New York, NY, USA, 2006. ACM Press.

[Pan04] R. Panigrahy. Minimum enclosing polytope in high dimensions, 2004.

[Pis81] G. Pisier. Remarques sur un resultat non publié de B. Maurey. *Séminaire d'Analyze Fonctionelle Palaiseau*, 1(12):1980–1981, 1981.

[Pla99] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods: support vector learning*, pages 185–208, 1999.

[TKC05] 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.

[TKL05] I. W. Tsang, J. T. Kwok, and K. T. Lai. Core vector regression for very large regression problems. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 912–919, New York, NY, USA, 2005. ACM Press.

[TKZ06] I. W. Tsang, J. T. Kwok, and J. M. Zurada. Generalized core vector machines. *IEEE Transactions on Neural Networks*, 17(5):1126–1140, 2006.

[TLJJ06] B. Taskar, S. Lacoste-Julien, and M. I. Jordan. Structured prediction, dual extragradient and Bregman projections. *Journal of Machine Learning Research*, 7:1627–1653, 2006.

[Tro04] J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory*, 50(10):2231–2242, 2004.

[TY05] M. J. Todd and E. A. Yildirim. On Khachiyan's algorithm for the computation of minimum volume enclosing ellipsoids. Technical report, Cornell University, 2005.

[Wyn70] H. P. Wynn. The sequential generation of D-optimum experimental design. *Annals of Mathematical Statistics*, 41:1655–1664, 1970.

[Zha03] T. Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Trans. Information Theory*, 49:682–691, 2003.