Subgradient and Sampling Algorithms for ℓ_1 Regression

Kenneth L. Clarkson*

January 4, 2005

Abstract

Given an $n \times d$ matrix A and an n-vector b, the ℓ_1 *regression* problem is to find the vector x minimizing the objective function $\|Ax - b\|_1$, where $\|y\|_1 \equiv \sum_i |y_i|$ for vector y. This paper gives an algorithm needing $O(n \log n)d^{O(1)}$ time in the worst case to obtain an approximate solution, with objective function value within a fixed ratio of optimum. Given $\epsilon > 0$, a solution whose value is within $1 + \epsilon$ of optimum can be obtained either by a deterministic algorithm using an additional $O(n)(d/\epsilon)^{O(1)}$ time, or by a Monte Carlo algorithm using an additional $O((d/\epsilon)^{O(1)})$ time. The analysis of the randomized algorithm shows that weighted coresets exist for ℓ_1 regression. The algorithms use the ellipsoid method, gradient descent, and random sampling.

1 Introduction

Given a set S of n points, the ℓ_1 regression problem is to fit a hyperplane to the points, minimizing the sum of the vertical distances of the points to the hyperplane. More formally, given an $n \times d$ matrix A and an n-vector b, the problem is to find the d-vector $\hat{x}$ such that the sum of the absolute values of the entries of $Ax - b$ is minimum. The entries of $Ax - b$ are called *residuals*. That is, $\hat{x}$ minimizes the objective function $\|Ax - b\|_1$, yielding the minimum residual vector $\hat{b} \equiv A\hat{x} - b$. Another description of $\hat{x}$ is that it gives the linear combination of the columns of A that is closest in ℓ_1 distance to b.

This problem arises in statistics, as a more robust alternative to least squares regression, for which the sum of the *squares* of the residuals is minimized. We can also regard the residual as the difference between b_i and its prediction $a_{i\cdot}x$, where $a_{i\cdot}$ is the i'th row of A. (The points of S are of the form $[a_{i\cdot}b_i]$, where $a_{i\cdot}$ is the i'th row of A, so "vertical" here would refer to the last coordinate.)

Related work. While ℓ_1 regression is an instance of linear programming, and so has good algorithms in theory and practice, it is still of interest to further understand its computational properties. Moreover, there is the usual gap between theory and practice: the best provably-good algorithms take $O(n)$ time, and are exact,[YKII88, MT93] but are complicated, and take time at least exponential in d. (The stated dependence is 3^{d^2},[MT93] but a dependence that is single-exponential is likely to be achievable with current methods.) In contrast, the algorithms here, while approximate and $O(n \log n)$, are polynomial in d and $1/\epsilon$.

While interior point or simplex methods can be applied, and may be satisfactory in practice, their worst-case dependence on n may be substantial, since the linear programming formulation has $2n$ constraints and $n + d$ unknowns.[Sch86] (While the dual formulation

*Bell Labs; 600 Mountain Avenue; Murray Hill, New Jersey 07974; clarkson@research.bell-labs.com

allows an interior point iteration in $nd^{O(1)}$ time, the number of such iterations can be large in the worst case. Also, the ellipsoid method could be applied using the subgradient, as used here for rounding.)

Subgradient descent. A key algorithmic element used here is a version of gradient descent: starting at an estimate of the optimum, move in the direction of the negative gradient of the objective function

$$F(x) \equiv \|Ax - b\|_1,$$

to obtain a new estimate. Each such step has a certain step length, chosen judiciously (but easily). The descent direction at d-vector x used here is

$$g_x \equiv -A^T \operatorname{sgn}(Ax - b),$$

where $\operatorname{sgn}(y)$, for a vector y, is the vector of equal dimension whose coordinates are the signs of the corresponding coordinates of y. (Note that the corresponding direction for the least squares objective function $\|Ax - b\|_2$ is $-A^T(Ax - b)$; setting that function to zero yields the normal equations.)

Of course, the function $\|Ax-b\|_1$ is only piece-wise linear, and so does not have a gradient everywhere; however, the value g_x is a *subgradient* (as stated below in Lemma 3.1), and the iteration just described is an instance of an unconstrained "subgradient method." Such methods have a long history and rich theory (see, e.g., [Sho85, NB01, Ber01]). However, their analyses seem to typically focus on limiting convergence behavior: as the number of iterations goes to infinity, does the estimate converge to the optimum? Also, most analyses of subgradient methods focus on a fixed sequence of stepsizes, where here the stepsize is computed using the current function value.

Conditioning. Here the results can be stated with bounds in terms of n, d, and ϵ, as in Theorem 3.7, by taking advantage a particular kind of "conditioning" of A for ℓ_1 regression, related to polytope rounding.[Lov86] Elementary column operations and scaling on A amount to a change of variable for x. Since such changes of variable can be tracked readily, this paper will freely use column operations and scaling. As discussed in Section 2, such operations can be done so as to obtain a new version of the matrix A with the property that, for any x,

$$\|x\|_1 \geq \|Ax\|_1 \geq \|x\|_1/d\sqrt{d}.$$

A matrix A will this property will be called ℓ_1 *conditioned.* This relation is similar to, but weaker than, the strict equality $\|Ax\|_2 = \|x\|_2$ that would hold if Gram-Schmidt orthogonalization were done on the columns of A. The conditioning is done by applying the ellipsoid algorithm, as used to find "weak Loewner-John ellipsoids." The subgradient is used as a separation oracle. To obtain a good time bound for the conditioning, without a dependence on the bit-complexity of the input, we first apply elementary column operations to make the columns of A orthogonal, and make b orthogonal to A. That is, a bit of "folklore" advice for getting an initial estimate for ℓ_1 regression, which is to begin with the optimum for ℓ_2 regression, is used here.

The conditioning of A is helpful both for the ℓ_1 subgradient method of Section 3, and for the sampling procedure discussed in Section 4. For the subgradient method, the conditioning assures that each iteration makes good progress. For the randomized sampling procedure, the conditioning allows a bound on the norm of the optimum vector $\hat{x}$, which allows a bound on the variance of the objective function random variable considered in the analysis.

Sampling. The sampling algorithm is not much more than picking a random sample of S, and solving the ℓ_1 regression problem for the resulting *sample problem.* (This again is not far from folklore for how to get an initial estimate for ℓ_1 regression.) The sampling is done after some preprocessing, including the conditioning procedure, and the samples are chosen with probabilities that depend on the point coordinates. Moreover, the resulting sample problem has each point weighted. The sample problem has the provable property that the optimum $\hat{x}$ of the original problem yields the expected $F(\hat{x})$ value as in the original problem, and any point yielding a bad objective function value in the original problem also yields a bad value in the sample. (This is the outline of the proof of Theorem 4.7.) So the solution to the sample problem will be a good solution to the original.

The approximate hardness of the weighted sample problem is a property analogous to that of "coresets" for other fitting problems.[BHPI02, HPV02, AHPVar, BC02] A novelty here is that while most of the known coreset constructions are for problems where the fit involves minimizing a maximum of distances, here the fit is the sum of residuals. However, the "coreset" here is not apparently as useful, but may be one step toward coreset-based algorithms for problems related to ℓ_1 regression. The size of the sample is only polynomially dependent on the dimension, as opposed to some coreset constructions with exponential dependence on the dimension. As shown in Theorem 4.7, the bound is $O(d^3\sqrt{d}/\epsilon^2)\log(d/\gamma\epsilon)$, where γ is a failure probability; this is similar in form to bounds for ϵ-approximations.

The probabilities used in the sampling technique is similar in flavor to that used in some recent randomized approximation algorithms for numerical linear algebra.[DK01, DKM] One difference from those algorithms is that the sampling is done independently for each input point, rather than picking each point of the sample independently from among the input points.

Detailed time bounds. The time bounds for the algorithms are, in more detail,

$$O(nd^5(\log n \log d + \log(d/\epsilon)/\epsilon^2))$$

in the worst case, for the deterministic algorithm, and

$$O(nd^5 \log n \log d) + O(\frac{d^8\sqrt{d}}{\epsilon^4}(\log d)^3 \ln(1/\epsilon)^2 \ln(1/\gamma))$$

for the Monte Carlo algorithm, where the failure probability is γ. (Note the lack of dependence on n in the second term of the latter bound.) These bounds are from Theorem 3.7 and Theorem 4.8, respectively. Although these dependencies are polynomial, they are relatively large, but it is likely that faster performance is possible in practice. This is true particularly if the rounding/conditioning step is skipped; then the main algorithm comprises only the subgradient descent procedure. In that case $O(nd^2)$ time is needed, iterated a data-dependent number of times.

Relation to small-d LP. The form of the time bound for the sampling algorithm is roughly similar to those long known for randomized algorithms for LP (linear programming) in small dimension.[Cla95] Such LP problems arise in, for example, ℓ_∞ regression. The previous algorithms rely on the existence of a "coreset" of size d for the exact problem; that existence follows by basic arguments. Here the coreset existence is much harder to prove, and uses weights, and is only approximate.

Statistical view. Another motivation for studying this problem is that we might regard the regression hyperplane as a statistic, interpreting S itself as a random sample from an underlying population. We might hope that the expected value of the statistic for the sample is equal to the value of the statistic for the whole set, and indeed, an unweighted (unbiased) sample has the appropriate expectation. However, an unweighted sample doesn't have the same provable properties, for clear reasons: while a single outlying point can change the regression plane by an arbitrarily large amount, that outlier may well not be in the sample. That is, the ℓ_1 estimator is not that robust, and so unweighted samples can do poorly. Here the sampling method makes outliers more likely to be picked; this reduces the variance due to their presence.

Outline. Before describing the approximation algorithm in Section 3 and the sampling algorithm in Section 4, the conditioning procedure is described. There are some concluding remarks.

2 Rounding and Conditioning

We will apply a standard rounding procedure, using the ellipsoid method[Lov86], to the polytope $P \equiv P(A) \equiv \{x \mid \|Ax\|_1 \leq 1\}$, which will have the effect of making A well-behaved with respect to the ℓ_1 norm.

This procedure requires a separation oracle, that for a given point $x \notin P$, returns a hyperplane H that separates x and P. Here the subgradient provides such an oracle, since

$$\|Ay\|_1 \geq \|Ax\|_1 + (y-x)^T A^T \operatorname{sgn}(Ax),$$

as shown in Lemma 3.1 below, so that the hyperplane

$$\{y \mid (y-x)^T A^T \operatorname{sgn}(Ax) = ((\|Ax\|_1) + 1)/2\}$$

separates x and P when $\|Ax\|_1 > 1$.

The rounding procedure maintains a series of ellipsoids E_m, $m = 0, \ldots$, where E_0 is a Euclidean ball B containing P. The procedure uses the separation oracle to maintain the condition that each E_m contains P and has provably smaller volume than E_{m-1}. A critical quantity is the ratio of the volume of B to the volume of P; the running time is proportional to the logarithm of this ratio.

We can construct an instance of the rounding problem where the ratio is a function only of n and d, as follows. Use Gram-Schmidt, or an equivalent procedure, to make the columns of A orthogonal to each other. Also, scale the columns of A to have ℓ_1 norm equal to one. This needs only elementary column operations, and as noted above, amounts to a change of variable. The following lemma applies.

Lemma 2.1 *Given an $n \times d$ matrix A where the columns of A are orthogonal and scaled to have unit ℓ_1 norm,*

$$B_{1/\sqrt{d}} \subset P(A) \subset B_{\sqrt{n}},$$

where B_r denotes the Euclidean ball centered at the origin, with radius r.

Proof: If $\|x\|_2 \leq 1/\sqrt{d}$, then $\|x\|_1 \leq 1$, which implies $\|Ax\|_1 \leq 1$ by the column scaling, giving the first inclusion.

For the second inclusion, for x with $\|Ax\|_1 \leq 1$ we have

$$\begin{aligned}\|x\|_2^2 &= \sum_k x_k^2 \|a_{\cdot k}\|_1^2 \\ &\leq n \sum_k x_k^2 \|a_{\cdot k}\|_2^2 \\ &= n\|Ax\|_2^2 \leq n\|Ax\|_1^2 \leq n,\end{aligned}$$

which implies $x \in B_{\sqrt{n}}$. ■

Lemma 2.2 *As applied to the preprocessed $n \times d$ matrix A, the rounding procedure takes*

$$O(nd^5 \log n)$$

time, and returns a pair of concentric ellipsoids E and E' with $E' \subset P(A) \subset E$, and E' arises from E by shrinking by a factor of $1/d$.

Proof: From the previous lemma, the ratio of the volumes of the initial ellipsoid to the final ellipsoid is at most $(dn)^{d/2}$, and the ellipsoids shrink by a factor of $e^{-3/2(d+1)(2d+1)}$ at each step,[Lov86] so the number of steps is $O(d^3 \log(nd))$. A factor $d^2 n$ accounts for the work at each step of checking inclusions, generating separating hyperplanes, and linear algebra to maintain the ellipsoids. Since P is centrally symmetric, the tighter shrinkage term $1/d$ can be used, instead of the general $1/d\sqrt{d}$ for such ellipsoids. ■

The result of this rounding procedure can be more conveniently used here by applying another change of variables to yield the condition that A is "μ-conditioned" for $\mu = d\sqrt{d}$, where this means that for any x,

$$\|x\|_1 \geq \|Ax\|_1 \geq \|x\|_1/\mu. \tag{1}$$

Theorem 2.3 *Given an $n \times d$ matrix A, the rounding procedure and elementary column operations on A yield a new version of A that is $d\sqrt{d}$-conditioned, in time*

$$O(nd^5 \log n).$$

Proof: Having applied the rounding procedure, there is an affine transformation with matrix τ that maps the containing ellipsoid E to B_d, and the contained ellipsoid E' to B_1. That is, if $y = \tau(x)$ for some x, and $\|y\|_1 \leq 1$, then $\|y\|_2 \leq 1$, and so $y \in B_1 \subset \tau P$. Moreover, if $y \in \tau P$, then $y \in B_d$ and $\|y\|_2 \leq d$, so $\|y\|_1 \leq d\sqrt{d}$. Since $\tau P = P(A\tau^{-1})$, renaming $A\tau^{-1}$ to be A yields the conditions that $\|y\|_1 \leq 1$ implies $\|Ay\|_1 \leq 1$, and $\|Ay\|_1 \leq 1$ implies $\|y\|_1 \leq d\sqrt{d}$. These facts imply the inequalities (1) with $\mu = d\sqrt{d}$. ■

3 An Approximation Algorithm

As shown below in Lemma 3.2, the rounding procedure has the useful effect that it results in an optimization problem for which the negative gradient of the objective function points from the current location toward an optimal vector. This and other properties will be stated and proven in the analysis of the subgradient iteration scheme.

This property of the gradient immediately suggests a descent scheme: take a step along the negative gradient toward an optimal vector, re-evaluate the gradient, and repeat. A complication here is that the although the negative gradient points toward an optimum vector, it is difficult to tell how far to go in that direction without overshooting the best possible improvement. It's also difficult to know just how much an improvement will occur.

One approach might be to do a search along the line of the gradient, approximately finding the minimum of the objective function when confined to the given line. This is likely to work well in practice, but seems difficult to analyze.

Here we adopt an approach similar to, but more elaborate than, the smallest ball algorithm of Badiou et al.[BC02]. The step length is calculated using the current function value and other inputs, with a scheme that allows lower and upper bounds on the approximation ratio $\|Ax - b\|_1/\|A\hat{x} - b\|_1$ to be maintained, where again $\hat{x}$ is an optimal vector.

3.1 The subgradient iteration

The approximation algorithm is simply the subgradient iteration procedure described below, under the assumption that A has already been μ-conditioned with $\mu = d\sqrt{d}$, as discussed in the last section, and that b is orthogonal to the subspace spanned by the column vectors of A. The latter is easy to achieve by using the orthogonalization of A produced while A was being conditioned.

The subgradient iteration procedure uses several readily calculated values, which will be specified below. The procedure is as follows: for each entry in a decreasing sequence of values $\tilde{\alpha}_i$, for $i = 0, 1, \ldots$, perform a *minor iteration* with $\tilde{\alpha}_i$ until $\tilde{\alpha}_i \leq 1 + \epsilon$, where $1 + \epsilon$ is the desired approximation ratio. As a special case for $i = 0$, repeatedly perform minor iterations with $\tilde{\alpha}_0$ until a minor iteration is unsuccessful, as described below.

Each minor iteration using some $\tilde{\alpha}$ starts with the current estimate x of the solution, and iteratively attempts to improve on x as follows. Let $x_0 := x$; for $k = 0, 1, \ldots, B(\tilde{\alpha}_i)$, where $B(\tilde{\alpha}_i)$ is given below, let

$$x_{k+1} := x_k + \rho(x_k, \tilde{\alpha}) \frac{g_{x_k}}{\|g_{x_k}\|_2},$$

where $\rho(x_k, \tilde{\alpha})$ is defined below, and

$$g_{x_k} \equiv -A^T \operatorname{sgn}(Ax_k - b),$$

as defined in the introduction.

A minor iteration can quit before $k = B(\tilde{\alpha}_i)$, if

$$F(x_k)/F(x_0) = \|Ax_k - b\|_1 / \|Ax_0 - b\|_1 \leq (1 + 1/\tilde{\alpha})/2.$$

Say that a minor iteration that quits early for this reason is successful, and let the current estimate $x := x_k$. Otherwise the minor iteration is unsuccessful, and the final value of x_k is not used.

This completes the description of the algorithm, except for the definitions of $\tilde{\alpha}_i$, $B(\tilde{\alpha})$ and $\rho(x_k, \tilde{\alpha})$. These are

$$\tilde{\alpha}_i \equiv \begin{cases} 2 & i = 0 \\ \frac{\tilde{\alpha}_{i-1}}{4}(1 + \sqrt{1 + 8/\tilde{\alpha}_{i-1}}) & i > 0. \end{cases}$$

and

$$B(\tilde{\alpha}) \equiv \frac{\ln[2\mu\sqrt{d}(\tilde{\alpha}+1)/(\tilde{\alpha}-1)]}{\kappa((\tilde{\alpha}+1)/2)^2/2},$$

where

$$\kappa(\tilde{\alpha}) \equiv \frac{\tilde{\alpha} - 1}{\mu\sqrt{d}(\tilde{\alpha}+1)}.$$

Finally,

$$\rho(x, \tilde{\alpha}) \equiv F(x)(1 - 1/\tilde{\alpha})/\sqrt{d}.$$

The motivations for these definitions should be clearer from the following analysis. The general idea is that g_x has positive dot product with $\hat{x} - x$, and the stepsize ρ is chosen so that progress toward $\hat{x}$ is made, but the step doesn't overshoot. The values $\tilde{\alpha}_i$ are intended as lower bounds on the approximation ratio $F(x)/F(\hat{x})$. If indeed $\tilde{\alpha}_i \leq F(x)/F(\hat{x})$ then, as will be shown, the minor iteration will be successful, and the approximation ratio will be reduced. The failure of a minor iteration amounts to a proof that $\tilde{\alpha}_i > F(x)/F(\hat{x})$, and so the lower bound $\tilde{\alpha}_i$ can be reduced.

The algorithm and analysis are less elegant than they might be, because for a given step, it hasn't been proven that the objective function value is reduced, only that the ℓ_2 distance to an optimum vector is reduced, and that reduction in distance is not directly known. Hence certain conditions must be inferred from the success or failure of a sufficient number of steps, that is, a minor iteration.

3.2 Analysis of the subgradient iteration

First, a theorem regarding the gradient of the ℓ_1 regression objective function. Some notation will be needed: for vector c, again, let $\operatorname{sgn} c$ denote a vector with the same dimensions as c, but with a $+1$ entry when where c has a positive entry, a -1 when c has a negative entry, and zero where c has a negative entry. With this definition, we can write $\|Ax - b\|_1$ as $\operatorname{sgn}(Ax - b)^T(Ax - b)$.

When the gradient of $\|Ax - b\|_1$ exists, it is $A^T \operatorname{sgn}(Ax - b)$: suppose x is such that for any d-vector δ that is short enough, $\operatorname{sgn}(Ax - b) = \operatorname{sgn}(A(x + \delta) - b)$. Then

$$\begin{aligned}
&\|A(x+\delta) - b\|_1 \\
&\quad = \operatorname{sgn}(A(x+\delta) - b)^T(A(x+\delta) - b) \\
&\quad = \operatorname{sgn}(Ax - b)^T(A(x+\delta) - b) \\
&\quad = \operatorname{sgn}(Ax - b)^T(Ax - b) + \operatorname{sgn}(Ax - b)^T A\delta \\
&\quad = \|Ax - b\|_1 + \delta^T A^T \operatorname{sgn}(Ax - b),
\end{aligned}$$

so the Taylor expansion of

$$\|A(x+\delta) - b\|_1$$

is the above expression, and $A^T \operatorname{sgn}(Ax - b)$ is the gradient at x.

Indeed, $A^T \operatorname{sgn}(Ax - b)$ is a *subgradient* of $\|Ax - b\|_1$, as shown in the next lemma.

Lemma 3.1 *For any d-vectors x and y,*

$$\|Ax - b\|_1 - \|Ay - b\|_1 \le (x - y)^T A^T \operatorname{sgn}(Ax - b).$$

That is, $A^T \operatorname{sgn}(Ax - b)$ is a subgradient.

Proof: We have

$$\begin{aligned}
&\|Ax - b\|_1 - \|Ay - b\|_1 \\
&\quad = \operatorname{sgn}(Ax - b)^T(Ax - b) - \operatorname{sgn}(Ay - b)^T(Ay - b) \\
&\quad \le \operatorname{sgn}(Ax - b)^T(Ax - b) - \operatorname{sgn}(Ax - b)^T(Ay - b) \\
&\quad = \operatorname{sgn}(Ax - b)^T A[x - y] \\
&\quad = (x - y)^T A^T \operatorname{sgn}(Ax - b),
\end{aligned}$$

■

Next we show that the gradient points toward an optimum vector $\hat{x}$. As before, we will use the notation $F(x) \equiv \|Ax - b\|_1$, and $g_x \equiv -A^T \operatorname{sgn}(Ax - b)$.

Lemma 3.2 *Given an ℓ_1 μ-conditioned matrix A and d-vector x, let*

$$\alpha(x, \mu) \equiv F(x)/F(\hat{x}), \tag{2}$$

and writing simply α when x and μ are understood. Let θ be the angle between g_x and $\hat{x} - x$, so

$$\cos\theta \equiv \frac{g_x^T(\hat{x} - x)}{\|g_x\|_2 \|\hat{x} - x\|_2}. \tag{3}$$

Then

$$\|\hat{x} - x\|_2 \cos\theta \geq \rho(x, \alpha) \equiv F(x)(1 - 1/\alpha)/\sqrt{d} \tag{4}$$

and

$$\cos\theta \geq \kappa(\alpha) \equiv \frac{\alpha - 1}{\mu\sqrt{d}(\alpha + 1)}. \tag{5}$$

Proof: At x, we have

$$g^T(\hat{x} - x) \geq F(x) - F(\hat{x}) = F(x)(1 - 1/\alpha),$$

by the previous lemma and definition of α, and omitting the subscript from g. Also $\|g\|_2 \leq \sqrt{d}$ since the column sums of A have been set to 1. From these considerations,

$$\begin{aligned} \|\hat{x} - x\|_2 \cos\theta = \frac{g_x^T(\hat{x} - x)}{\|g_x\|_2} &\geq (F(x) - F(\hat{x}))/\sqrt{d} \\ &= F(x)(1 - 1/\alpha)/\sqrt{d}, \end{aligned}$$

the first conclusion of the lemma.

From the triangle inequality, we have

$$\|A\hat{x} - Ax\|_1 \leq \|Ax - b\|_1 + \|A\hat{x} - b\|_1 = F(x) + F(\hat{x}),$$

and so, from the μ-conditioning of A,

$$\begin{aligned} \|\hat{x} - x\|_2 &\leq \|\hat{x} - x\|_1 \\ &\leq \mu\|A\hat{x} - Ax\|_1 \\ &\leq \mu(F(x) + F(\hat{x})) \\ &\leq \mu(F(x) + F(x)/\alpha) \\ &= \mu F(x)(1 + 1/\alpha), \end{aligned} \tag{6}$$

and so

$$\begin{aligned} \cos\theta &\geq \frac{F(x)(1 - 1/\alpha)/\sqrt{d}}{\|\hat{x} - x\|_2} \\ &\geq \frac{F(x)(1 - 1/\alpha)/\sqrt{d}}{\mu F(x)(1 + 1/\alpha)} \\ &= \frac{\alpha - 1}{\mu\sqrt{d}(\alpha + 1)} \end{aligned}$$

the other conclusion of the lemma. ∎

We need a quantitative statement about how far to go in the direction g_x.

Lemma 3.3 *With conditions as in the previous lemma,*

$$\|x + \rho(x, \alpha)\frac{g_x}{\|g_x\|_2} - \hat{x}\|_2 \leq (1 - \kappa(\alpha)^2/2)\|x - \hat{x}\|_2,$$

where $\rho(x, \alpha)$ and κ are defined above (4,5).

Proof: For convenience in this proof, put $\hat{x}$ at the origin. We have

$$\|x + \rho\frac{g_x}{\|g_x\|_2}\|^2 = \|x\|_2^2 + \rho^2 - 2\|x\|\rho\cos\theta$$

and so, writing κ for $\kappa(\alpha) \equiv \rho/\|x\|_2$,

$$\begin{aligned}\|x + \rho\frac{g_x}{\|g_x\|_2}\|^2/\|x\|_2^2 &= 1 + \frac{\rho^2}{\|x\|_2^2} - 2\frac{\rho}{\|x\|_2}\cos\theta \\ &= 1 + \kappa^2 - 2\kappa\cos\theta \\ &\leq 1 - \kappa^2\end{aligned}$$

using (4) and (5) of the last lemma. The lemma follows, taking square roots of both sides and using $\sqrt{1-w} \leq 1 - w/2$. ∎

A difficulty in using this result is that generally, $F(\hat{x})$ is unknown, which implies α is unknown. However, the following lemma implies that we can tell when an estimate $\tilde{\alpha}$ of α is too big.

Lemma 3.4 *With the conditions of the previous lemma, suppose $\tilde{\alpha} \leq \alpha \equiv \alpha(x)$. Let $x_0 \equiv x$, and let*

$$x_{k+1} \equiv x_k + \rho(x_k, \tilde{\alpha})\frac{g_{x_k}}{\|g_{x_k}\|_2},$$

for $k \geq 0$. There is a value

$$B(\tilde{\alpha}) \equiv \frac{\ln[2\mu\sqrt{d}(\tilde{\alpha}+1)/(\tilde{\alpha}-1)]}{\kappa((\tilde{\alpha}+1)/2)^2/2} \tag{7}$$

such that for some $k \leq B(\tilde{\alpha})$,

$$F(x_k) \leq F(x)(1 + 1/\tilde{\alpha})/2.$$

That is, if $\tilde{\alpha} \leq \alpha \equiv \alpha(x_0)$, a minor iteration succeeds.

Proof: Suppose the claim is false, so that during the $B(\tilde{\alpha})$ iterations, $F(x_k)$ remains above the given bound, implying

$$\begin{aligned}F(x_k) &> F(x_0)(1 + 1/\tilde{\alpha})/2 &(8)\\ &= F(\hat{x})\alpha(x_0)(1 + 1/\tilde{\alpha})/2 \\ &\geq F(\hat{x})(\alpha(x_0) + 1)/2 &(9)\end{aligned}$$

and so $\alpha(x_k) \geq (\alpha + 1)/2$ for $k = 1 \ldots B(\tilde{\alpha})$, where we abbreviate $\alpha(x_0)$ as α. We will show that this assumption implies a contradiction.

Based on this condition, the previous lemma implies that after $B(\tilde{\alpha})$ iterations,

$$\begin{aligned}
&\|x_k - \hat{x}\|_2 \\
&\leq \|x_0 - \hat{x}\|_2 \prod_{1\leq k\leq B(\tilde{\alpha})} (1 - \kappa(\alpha(x_k))^2/2) \\
&\leq (1 - \kappa((\alpha+1)/2)^2/2)^{B(\tilde{\alpha})} \|x_0 - \hat{x}\|_2, \qquad (10)
\end{aligned}$$

using the fact that $\kappa()$ is increasing. Using the triangle inequality and the ℓ_1 conditioning of A,

$$\begin{aligned}
F(x_k) &= \|Ax_k - b\|_1 \\
&\leq \|A\hat{x} - b\|_1 + \|Ax_k - A\hat{x}\|_1 \\
&\leq F(\hat{x}) + \|x_k - \hat{x}\|_1 \\
&\leq F(\hat{x}) + \|x_k - \hat{x}\|_2 \sqrt{d}.
\end{aligned}$$

That is, using (10) and (6), and the facts that $(\alpha - 1)/(\alpha + 1)$ and so $\kappa(\alpha)$ are increasing functions of α,

$$\begin{aligned}
&F(x_k) - F(\hat{x}) \\
&\leq \|x_k - \hat{x}\|_2 \sqrt{d} \\
&\leq (1 - \kappa((\alpha+1)/2)^2/2)^{B(\tilde{\alpha})} \|x_0 - \hat{x}\|_2 \sqrt{d} \\
&\leq \exp(-B(\tilde{\alpha})\kappa((\alpha+1)/2)^2/2) \mu F(x)(1 + 1/\alpha)\sqrt{d} \\
&\leq \frac{\mu F(\hat{x})\alpha(1 + 1/\alpha)\sqrt{d}}{2\mu\sqrt{d}(\tilde{\alpha}+1)/(\tilde{\alpha}-1)} \\
&\leq F(\hat{x})(\alpha - 1)/2,
\end{aligned}$$

which implies $F(x_k) \leq F(\hat{x})(\alpha + 1)/2$, contradicting assumption (9). ■

The following lemma gives the properties of the sequence of $\tilde{\alpha}_i$ that will be used in the analysis of the subgradient iteration algorithm.

Lemma 3.5 *With $\tilde{\alpha}_i$ as defined,*

$$\tilde{\alpha}_{i+1} = \tilde{\alpha}_i(1 + 1/\tilde{\alpha}_{i+1})/2,$$

and $\tilde{\alpha}_i \leq 1 + (3/4)^i$.

Proof: The first claim requires only the quadratic formula. The second can be proven with induction: it holds for $i = 0$, and for $i > 0$, let β denote $(3/4)^i$. Then, using $\beta \leq 1$ and $\sqrt{1+w} \leq 1 + w/2$,

$$\begin{aligned}
&4\tilde{\alpha}_{i+1} \\
&= (1+\beta)(1 + \sqrt{1 + 8/(1+\beta)}) \\
&= 1 + \beta + \sqrt{(1+\beta)^2 + 8(1+\beta)} \\
&= 1 + \beta + 3\sqrt{1 + 10\beta/9 + \beta^2/9} \\
&\leq 1 + \beta + 3\sqrt{1 + 11\beta/9} \\
&\leq 1 + \beta + 3(1 + 11\beta/18) \\
&= 4 + 17\beta/6 \\
&\leq 4 + 3\beta,
\end{aligned}$$

and so $\tilde{\alpha}_{i+1} \leq 1 + 3\beta/4 = 1 + (3/4)^{i+1}$, and the claim follows inductively. ■

Lemma 3.6 *For $\mu \leq d\sqrt{d}$ and $\tilde{\alpha} = 1 + \epsilon$,*

$$B(\tilde{\alpha}) = B(1 + \epsilon) = O(d^4 \log(d/\epsilon))/\epsilon^2$$

as $\epsilon \to 0$.

Proof: From (7) and the definition (5) of $\kappa()$, we have

$$\begin{aligned} B(\tilde{\alpha}) &\equiv \frac{\ln[2\mu\sqrt{d}(\tilde{\alpha}+1)/(\tilde{\alpha}-1)]}{\kappa((\tilde{\alpha}+1)/2)^2/2} \\ &\leq \frac{2\mu^2 d(\tilde{\alpha}+3)^2 \ln[\mu d\sqrt{d}(\tilde{\alpha}+1)/(\tilde{\alpha}-1)]}{(\tilde{\alpha}-1)^2}, \end{aligned}$$

from which the lemma follows. ■

Theorem 3.7 *A d-vector x with ℓ_1 regression value within ϵ of optimal can be found within*

$$O(nd^5(\log n \log d + \log(d/\epsilon)/\epsilon^2))$$

time.

Proof: The preliminary orthogonalization can be done with $O(d^2)$ vector operations on b and the column vectors of A, each requiring $O(n)$ time, for $O(nd^2)$ time overall.

The rounding procedure, yielding the conditioned version of A, needs $O(nd^5 \log n)$, from Lemma 2.2.

The algorithm requires $O(nd)$ time for each of at most $B(\tilde{\alpha}_i)$ steps of the i'th minor iteration. By the previous lemma, $B(2) = O(d^4 \log d)$. When the minor iteration with $\tilde{\alpha}_i = 2$ is successful, the stopping condition implies that the approximation ratio $F(x)/F(\hat{x})$ has decreased by a factor of $(1+1/2)/2 = 3/4$. A minor iteration is unsuccessful only when $\tilde{\alpha}_i > \alpha(x)$, by Lemma 3.4, and so when a minor iteration with $\tilde{\alpha}_i = 2$ is unsuccessful, it must be that the approximation ratio $\alpha(x) < 2$. By beginning with b orthogonal to the columns of A, the initial approximation ratio is at most $\sqrt{n}$, since

$$\|b\|_1 \leq \|b\|_2\sqrt{n} \leq \|\hat{b}\|_2\sqrt{n} \leq \|\hat{b}\|_1\sqrt{n},$$

where again, $\hat{b} \equiv A\hat{x} - b$. Therefore, the work for all the minor iterations with $\tilde{\alpha}_i = 2$ is

$$O(nd)B(2)\log n = O(nd^5 \log d \log n).$$

For the minor iterations with $i > 0$ (and so $\tilde{\alpha}_i < 2$), note that after the minor iteration is done using $\tilde{\alpha}_i$, it holds inductively that $\alpha \leq \tilde{\alpha}_i$: either the iteration fails, which can only happen if $\alpha \leq \tilde{\alpha}_i$, or it succeeds, which implies that the final iterate x_k satisfies

$$\begin{aligned} F(x_k) &\leq F(x_0)(1+\tilde{\alpha}_i)/2 \\ &\leq F(x_0)\tilde{\alpha}_{i-1}(1+\tilde{\alpha}_i)/2 \\ &= F(x_0)\tilde{\alpha}_i, \end{aligned}$$

using the condition for success, the inductive assumption, and Lemma 3.5.

Also from that lemma, $\tilde{\alpha}_i \leq 1 + (3/4)^i$, and so $O(\log(1/\epsilon))$ minor iterations are needed. By the previous lemma, $O(d^4 \log(d/(\tilde{\alpha}_i - 1))/(\tilde{\alpha}_i - 1)^2)$ steps are needed for each minor iteration, and from that bound the number of steps overall is within a constant factor of the number of steps in the last minor iteration, so the total work for the minor iterations with $\tilde{\alpha}_i < 2$ is

$$O(nd)d^4 \log(d/\epsilon)/\epsilon^2.$$

Putting this bound together with the bound for $\tilde{\alpha}_i = 2$ yields the theorem. ■

4 A Sampling Algorithm

First, some probabilistic bounds, and then the sampling construction.

4.1 Tail Estimates

Theorem 4.1 *Let X_i, $i = 1 \ldots n$, be independent random variables, with bounded EX_i^2 and $X_i \geq 0$ for all i. For $S \equiv \sum_i X_i$ and $t \geq 0$,*

$$\log \operatorname{Prob}\{S \leq ES - t\} \leq \frac{-t^2}{2\sum_i EX_i^2}.$$

If there is also some M with $X_i \leq EX_i + M$ for all i, then

$$\log \operatorname{Prob}\{S \geq ES + t\} \leq \frac{-t^2}{2tM/3 + 2\sum_i EX_i^2 - [EX_i]^2}$$

Proof: The bound on the lower tail seems to be due to Maurer[Mau03] and independently McAllester and Ortiz[MO03]; the former gave a simple proof. The upper tail bound is due to Bernstein.[Ber46] ■

4.2 The Sampling Algorithm

The first step of the overall algorithm is to apply the rounding algorithm of Section 2 to $d\sqrt{d}$-condition A. It also is necessary to apply the subgradient algorithm of Section 3 with $\epsilon = 1$ to b, so that $\|b\|_1 \leq 2\|A\hat{x} - b\|_1$. (That is, having obtained an approximate solution $\tilde{x}$, replace b by $b - A\tilde{x}$, and make a change of variable in x to account for the different b.) It will be convenient to then scale b so that $\|b\|_1 = d$.

Following these preprocessing steps, the sampling algorithm is as follows: let

$$f_i \equiv |b_i| + \|a_{i\cdot}\|_1,$$

and let

$$p_i \equiv \min\{1, rf_i/2d\},$$

where r is an integer parameter, noting that $\sum_i f_i = 2d$.

Let Y and Z be diagonal $n \times n$ matrices where

$$Y_{ii} = \begin{cases} 1 & \text{with probability } p_i \\ 0 & \text{with probability } 1 - p_i, \end{cases}$$

and $Z_{ii} \equiv Y_{ii}/p_i$. Now solve the *sample problem*

$$\min_x \|Z(Ax - b)\|_1$$

approximately, and return its output as an approximation to the solution of the original problem. This is the whole of the algorithm.

By construction, we have the expected number of nonzero Y_{ii} to be no more than

$$E\sum_i Y_{ii} = \sum_i EY_{ii} = \sum_i p_i \leq \sum_i rf_i/2d = r,$$

and since $EZ_{ii} = EY_{ii}/p_i = p_i/p_i = 1$,

$$\begin{aligned} E\|Z(Ax - b)\|_1 &= E\sum_i |Z_{ii}(b_i - a_{i\cdot}x)| \\ &= \sum_i EZ_{ii}|b_i - a_{i\cdot}x| \\ &= \|Ax - b\|_1. \end{aligned}$$

So the sample problem gives an unbiased estimate of the objective function value of x, for any x. Of course, this does not show that the sample problem has an objective function value near that of $\hat{x}$, since we don't know how concentrated these values are around their expectations. However, we can apply the tail bounds, Theorem 4.1, to show that with high probability, $\|Z(A\hat{x} - b)\|_1$ is not too large, and for any given x, $\|Z(Ax - b)\|_1$ is not too small. This will require a bound on $\sum_i E[(Z_{ii}|b_i - a_{i\cdot}x|)^2]$, before which, a lemma.

Lemma 4.2 *For $d\sqrt{d}$-conditioned $n \times d$ matrix A, and d-vector b with $\|b\|_1 \leq 2\min_x \|b - Ax\|_1$, we have*

$$\|x\|_1 \leq d\sqrt{d}\|Ax - b\|_1.$$

Proof: Using Lemma 2.3 and the hypothesis,

$$\begin{aligned} \|x\|_1 &\leq d\sqrt{d}\|Ax\|_1 \\ &\leq d\sqrt{d}\,(\|b\|_1 + \|Ax - b\|_1) \\ &\leq d\sqrt{d}\,(2\|A\hat{x} - b\|_1 + \|Ax - b\|_1) \\ &\leq 3d\sqrt{d}\|Ax - b\|_1. \end{aligned}$$

■

Lemma 4.3 *Under the conditions of the sampling algorithm,*

$$\sum_i E[(Z_{ii}|b_i - a_{i\cdot}x|)^2] \leq \frac{6d^2\sqrt{d}}{r}\|Ax - b\|_1^2.$$

Proof: Using the definitions, the previous lemma, standard facts, and ignoring those i for which $p_i = 1 < rf_i/2d$,

$$
\begin{aligned}
&\sum_i E[(Z_{ii}|b_i - a_{i\cdot}x|)^2] \\
&\quad = \sum_i \frac{|b_i - a_{i\cdot}x|^2}{p_i^2} E[Y_{ii}^2] \\
&\quad = \sum_i \frac{|b_i - a_{i\cdot}x|^2}{p_i} \\
&\quad = \sum_i \frac{|b_i - a_{i\cdot}x|^2}{(|b_i| + \|a_{i\cdot}\|_1)r/2d} \\
&\quad \leq \sum_i \frac{|b_i - a_{i\cdot}x|(|b_i| + \|a_{i\cdot}\|_1\|x\|_\infty)}{(|b_i| + \|a_{i\cdot}\|_1)r/2d} \\
&\quad \leq \sum_i \frac{|b_i - a_{i\cdot}x|(|b_i| + \|a_{i\cdot}\|_1)\max\{1, \|x\|_\infty\}}{(|b_i| + \|a_{i\cdot}\|_1)r/2d} \\
&\quad = \frac{2d}{r}\|Ax - b\|_1 \max\{1, \|x\|_\infty\} \\
&\quad \leq \frac{2d}{r}\|Ax - b\|_1 (3d\sqrt{d}\|Ax - b\|_1) \\
&\quad = \frac{6d^2\sqrt{d}}{r}\|Ax - b\|_1^2,
\end{aligned}
$$

and the lemma follows. ■

Lemma 4.4 *For $\epsilon \leq 1/2$ and a given d-vector x, the probability is at most*

$$\exp(-\epsilon^2 r/24d^2\sqrt{d})$$

that

$$\|Z(Ax - b)\|_1 \leq (1+\epsilon)\|\hat{b}\|_1$$

when $\|Ax - b\|_1 \geq (1+2\epsilon)\|\hat{b}\|_1$.

Proof: The lower tail bound of Theorem 4.1 can be applied, with random variables $X_i = Z_{ii}|a_{i\cdot}x - b_i|$, sum $S = \|Z(Ax - b)\|_1$, and

$$t = ES - (1+\epsilon)\|\hat{b}\|_1 = \|Ax - b\|_1 - (1+\epsilon)\|\hat{b}\|_1.$$

Using also the previous lemma and $\epsilon \leq 1/2$, we have

$$
\begin{aligned}
&\log \text{Prob}\{\|Z(Ax - b)\|_1 \leq (1+\epsilon)\|\hat{b}\|_1\} \\
&\leq \frac{-(\|Ax - b\|_1 - \|\hat{b}\|_1(1+\epsilon))^2}{6d^2\sqrt{d}\|Ax - b\|_1^2/r} \\
&\leq \frac{-r}{6d^2}\left(1 - \frac{(1+\epsilon)\|\hat{b}\|_1}{(1+2\epsilon)\|\hat{b}\|_1}\right)^2 \\
&\leq \frac{-r\epsilon^2}{24d^2\sqrt{d}},
\end{aligned}
$$

as claimed. ∎

To apply the upper tail bound of Theorem 4.1, we need an upper bound on each coordinate of $\hat{b}$.

Lemma 4.5 *For all i, $Z_{ii}|\hat{b}_i| \leq 6d^2\sqrt{d}\|\hat{b}\|_1/r$.*

Proof: From Lemma 4.2, we have

$$\|\hat{x}\|_1 \leq 3d\sqrt{d}\|\hat{b}\|_1/r.$$

Therefore

$$\begin{aligned} Z_{ii}|\hat{b}_i| = |\hat{b}_i|/p_i &= |\hat{b}_i|2d/f_i r \\ &\leq \frac{2d}{r}\frac{|b_i| + \|a_{i\cdot}\|_1\|\hat{x}\|_\infty}{|b_i| + \|a_{i\cdot}\|_1} \\ &\leq \frac{2d}{r}\frac{|b_i| + \|a_{i\cdot}\|_1 3d\sqrt{d}\|\hat{b}\|_1}{|b_i| + \|a_{i\cdot}\|_1} \\ &\leq \frac{6d^2\sqrt{d}\|\hat{b}\|_1}{r}. \end{aligned}$$

∎

Lemma 4.6 *For $\epsilon \leq 1$, the probability is at most $\exp(-\epsilon^2 r/16d^2\sqrt{d})$ that*

$$\|Z\hat{b}\|_1 \geq (1+\epsilon)\|\hat{b}\|_1.$$

Proof: The second part of Theorem 4.1 can be applied, with $X_i = Z_{ii}|\hat{b}_i|$, $S = \|Z\hat{b}\|_1$, $t = \epsilon\|\hat{b}\|_1$, and from the previous lemma, $M = 6d^2\sqrt{d}\|\hat{b}\|_1/r$. Consequently,

$$\begin{aligned} &\log \mathrm{Prob}\{\|Z\hat{b}\|_1 > \|\hat{b}\|_1 + t\} \\ &\leq \frac{-t^2}{2tM/3 + 2\sum_i EX_i^2 - [EX_i]^2} \\ &\leq \frac{-(\epsilon\|\hat{b}\|_1)^2}{2(\epsilon\|\hat{b}\|_1)(6d^2\sqrt{d}\|\hat{b}\|_1/r)/3 + 2\frac{6d^2\sqrt{d}}{r}\|\hat{b}\|_1^2} \\ &= \frac{-\epsilon^2}{4\epsilon d^2\sqrt{d}/r + 12d^2\sqrt{d}/r} \\ &\leq \frac{-\epsilon^2 r}{16d^2\sqrt{d}} \end{aligned}$$

for $\epsilon \leq 1$. ∎

These two lemmas can be combined to prove the following theorem.

Theorem 4.7 *For any $n \times d$ matrix A and n-vector b, and given $\epsilon > 0$ and $\gamma > 0$, there is a value R where*

$$R \equiv \frac{288d^3\sqrt{d}}{\epsilon^2}\ln(d/\epsilon\gamma)$$

such that if $r \geq R$, then with probability at least $1-\gamma$, the sample problem with parameter r has optimum solution x_Z with

$$\|Ax_Z - b\|_1 \leq (1+\epsilon)\|\hat{b}\|_1,$$

where $\|\hat{b}\|_1 = \|A\hat{x} - b\|_1$ is the minimum value of $\|Ax - b\|_1$.

Proof: We will show that with high probability, $\hat{x}$ yields an objective function value for the sample problem that is not too big, and that for all vectors, their sample problem objective function value is not too small. Here "too big" means larger than $(1+\epsilon/2)\|\hat{b}\|_1$, and "too small" means smaller than $(1+\epsilon/2)\|\hat{b}\|_1$, when the vector x gives objective function value greater than $1+\epsilon$ times the optimum.

For the latter, it is enough to show that any possible optimum vector x_Z gives a value that is not too small. Such optimum vectors come from a finite set of vectors: since ℓ_1 regression is a linear programming problem, its optima are at vertices of its feasible polytope; here those vertices correspond to hyperplanes determined by d points of the n input points S. There are $\binom{n}{d}$ such hyperplanes. For each such hyperplane, let $p_{i_1}, p_{i_2}, \ldots, p_{i_d}$ denote the points in S that determine it. Let T denote the corresponding set of indices, and let x_T denote the corresponding solution.

The probability that x_T is a bad solution to the sample problem is bounded by the probability that every index $i \in T$ has $Y_{ii} = 1$, and that the resulting objective function value is too small. The former probability is the product of the probabilities $\prod_{i\in T} p_i$. The latter probability is no more than

$$\exp(-\epsilon^2 r/96d^2\sqrt{d})$$

by Lemma 4.4 (using $\epsilon/2$ as the ϵ of the lemma). Let $\binom{N}{d}$ denote the set of all sets of size d of integers between 1 and n. Using these facts and the bound on r of the theorem statement, the probability that some x_T, for $T \in \binom{N}{d}$, has a value that is too small is no more than

$$\begin{aligned}
&\sum_{T\in\binom{N}{d}} \exp(-\epsilon^2 R/96d^2\sqrt{d}) \prod_{i\in T} p_i \\
&\leq \exp(-\epsilon^2 R/96d^2\sqrt{d}) \left(\sum_i p_i\right)^d \\
&= \exp(-\epsilon^2 R/96d^2\sqrt{d}) R^d \\
&\leq \gamma/2.
\end{aligned}$$

Applying Lemma 4.6, the probability that $\hat{x}$ yields an objective function value for the sample problem that is too large is at most

$$\exp(-\epsilon^2 R/64d^2\sqrt{d}) \leq \gamma/2,$$

and so the probability of failure is at most γ, as claimed. ■

Theorem 4.8 *For any $n \times d$ matrix A and n-vector b, and given $\epsilon > 0$ and $\gamma > 0$, in*

$$O(nd^5 \log n \log d) + O(\frac{d^8\sqrt{d}}{\epsilon^4}(\log d)^3 \ln(1/\epsilon)^2 \ln(1/\gamma))$$

time, a vector x^* can be found such that, with probability at least $1-\gamma$, $\|b - Ax^*\|_1 \leq (1+\epsilon)\min_x \|b - Ax\|_1$.

Proof: Applying Theorem 3.7 with $\epsilon = 2$, the time needed for preprocessing of A and b is $O(nd^5 \log n \log d)$.

The sample size R needed to obtain a sampling approximation less than $\epsilon/3$ is given by the last result; applying Theorem 3.7 again, the time to find a solution with $1+\epsilon/3$ of optimum for the sampling problem is

$$\begin{aligned}
&O(Rd^5(\log R \log d + \log(d/\epsilon)/\epsilon^2))\\
&= O(\frac{d^3\sqrt{d}}{\epsilon^2}(\log d)\ln(d/\epsilon\gamma)(\log(d/\epsilon)/\epsilon^2))\\
&= O(\frac{d^8\sqrt{d}}{\epsilon^4}(\log d)^3 \ln(1/\epsilon)^2 \ln(1/\gamma)),
\end{aligned}$$

as claimed. ∎

5 Concluding Remarks

The $\log n$ term in the runtime might be removable by using a sampling approach to estimate the subgradient, somewhat like the work of Nedić and Bertsekas.[NB01]

Given that the ellipsoid method was used, one might ask why that method could not be simply applied to solve the problem. Perhaps it can be, but it is not obvious that a bit-complexity dependence can be avoided; moreover, the subgradient descent scheme may be of independent interest. It seems likely that the conditioning of A can be done using the regression procedure itself, by making each column "ℓ_1 independent" of the others, and then the ellipsoid method is not needed.

References

[AHPVar] P. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *J. ACM*, to appear.

[BC02] Mihai Bădoiu and K. L. Clarkson. Optimal core-sets for balls. Manuscript, available via `http://cm.bell-labs.com/who/clarkson/`, 2002.

[Ber46] S. Bernstein. *The Theory of Probabilities.* Gastehizdat Publishing House, 1946.

[Ber01] D. Bertsekas. *Convex Analysis and Optimization.* Athena Scientific, 2001.

[BHPI02] M. Bădoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proc. 34th Symp. Theory of Comp.*, 2002.

[Cla95] K. L. Clarkson. Las Vegas algorithms for linear and integer programming when the dimension is small. *Journal of the ACM*, 42(2):488–499, 1995.

[DK01] P. Drinneas and R. Kannan. Fast Monte-Carlo algorithms for approximate matrix multiplication. In *EEE Symposium on Foundations of Computer Science*, 2001.

[DKM] P. Drineas, R. Kannan, and M. W. Mahoney. Fast monte carlo algorithms for matrices I: Approximating matrix multiplication. Manuscript. Available via `http://cs-www.cs.yale.edu/homes/mmahoney/`.

[HPV02] S. Har-Peled and K. R. Varadarajan. Projective clustering in high dimensions using core-sets. In *Symp. Comp. Geometry*, 2002.

[Lov86] L. Lovász. *Algorithmic Theory of Numbers, Graphs, and Convexity*. SIAM, 1986.

[Mau03] A. Maurer. A bound on the deviation probability for sums of non-negative random variables. *J. Inequalities in Pure and Applied Mathematics*, 4, 2003.

[MO03] D. McAllester and L. Ortiz. Concentration inequalities for the missing mass and for histogram rule error. *J. Machine Learning Research*, 4:895–911, 2003.

[MT93] N. Megiddo and A. Tamir. Linear time algorithms for some separable quadratic programming problems. *Operations Research Letters*, 13:203–211, March 1993.

[NB01] A. Nedić and D. Bertsekas. Incremental subgradient methods for nondifferentiable optimization. *SIAM J. Opt.*, 12(1):109–138, 2001.

[Sch86] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, New York, 1986.

[Sho85] N. Z. Shor. *Minimization Methods for Non-differentiable Functions*. Springer, 1985.

[YKII88] P. Yamamoto, K. Kato, K. Imai, and H. Imai. Algorithms for vertical and orthogonal L1 linear approximation of points. In *Proceedings of the fourth annual symposium on Computational geometry*, pages 352–361. ACM Press, 1988.