Dimensionality Reduction for Tukey Regression

Kenneth L. Clarkson [*1] **Ruosong Wang** [*2] **David P. Woodruff** [*2]

Abstract

We give the first dimensionality reduction methods for the overconstrained Tukey regression problem. The Tukey loss function $\|y\|_M = \sum_i M(y_i)$ has $M(y_i) \approx |y_i|^p$ for residual errors y_i smaller than a prescribed threshold τ, but $M(y_i)$ becomes constant for errors $|y_i| > \tau$. Our results depend on a new structural result, proven constructively, showing that for any d-dimensional subspace $L \subset \mathbb{R}^n$, there is a fixed bounded-size subset of coordinates containing, for every $y \in L$, all the large coordinates, with respect to the Tukey loss function, of y. Our methods reduce a given Tukey regression problem to a smaller weighted version, whose solution is a provably good approximate solution to the original problem. Our reductions are fast, simple and easy to implement, and we give empirical results demonstrating their practicality, using existing heuristic solvers for the small versions. We also give exponential-time algorithms giving provably good solutions, and hardness results suggesting that a significant speedup in the worst case is unlikely.

1. Introduction

A number of problems in numerical linear algebra have witnessed remarkable speedups via the technique of linear sketching. Such speedups are made possible typically by reductions in the dimension of the input (here the number of rows of the input matrix), whereby a large scale optimization problem is replaced by a much smaller optimization problem, and then a slower algorithm is run on the small problem. It is then argued that the solution to the smaller problem provides an approximate solution to the original problem. We refer the reader to several recent surveys on this topic (Kannan & Vempala, 2009; Mahoney, 2011; Woodruff, 2014).

This approach has led to optimal algorithms for approximate overconstrained least squares regression: given an $n \times d$ matrix A, with $n \gg d$, and an $n \times 1$ vector b, output a vector $x' \in \mathbb{R}^d$ for which $\|Ax' - b\|_2 \leq (1+\varepsilon) \min_x \|Ax - b\|_2$. For this problem, one first samples a random matrix $S \in \mathbb{R}^{k \times n}$ with a small number k of rows, and replaces A with $S \cdot A$ and b with $S \cdot b$. Then one solves (or approximately solves) the small problem $\min_x \|SAx - Sb\|_2$. The goal of this *sketch and solve* approach is to choose a distribution S so that if x' is the minimizer to this latter problem, then one has that $\|Ax' - b\|_2 \leq (1+\varepsilon) \min_x \|Ax - b\|_2$ with high probability. Note that $x' = (SA)^+ Sb$, where $(SA)^+$ denotes the pseudoinverse of SA and can be computed in kd^2 time, see, e.g., (Woodruff, 2014) for a survey and further background. Consequently, the overall time to solve least squares regression is $T + kd^2$, where T is the time to compute $S \cdot A$ and $S \cdot b$. Thus, the goal is to minimize both the time T and the sketching dimension k. Using this approach, Sárlos showed (Sarlós, 2006) how to achieve $O(nd \log n) + \mathrm{poly}(d/\varepsilon)$ overall time, which was subsequently improved to the optimal $\mathrm{nnz}(A) + \mathrm{poly}(d/\varepsilon)$ time in (Clarkson & Woodruff, 2013; Meng & Mahoney, 2012; Nelson & Nguyen, 2012).

Recently, a number of works have looked at more robust versions of regression. Sohler and Woodruff (Sohler & Woodruff, 2011), building off of earlier work of Clarkson (Clarkson, 2005) (see also (Dasgupta et al., 2009)), showed how to use the sketch and solve paradigm to obtain a $(1+\varepsilon)$-approximation to ℓ_1 regression, namely, to output a vector $x' \in \mathbb{R}^d$ for which $\|Ax' - b\|_1 \leq (1+\varepsilon) \min_x \|Ax - b\|_1$. This version of regression, also known as least absolute deviations, is known to be less sensitive to outliers than least squares regression, since one takes the absolute values of the errors in the residuals rather than their squares, and so does not try to fit outliers as much. By now, we also have optimal $\mathrm{nnz}(A) + \mathrm{poly}(d/\varepsilon)$ time algorithms for ℓ_1 regression (Li et al., 2013; Clarkson et al., 2016; Woodruff & Zhang, 2013; Clarkson & Woodruff, 2013; Meng & Mahoney, 2012; Wang & Woodruff, 2019), for the related quantile regression problem (Yang et al., 2013), and for ℓ_p regression for every $p \geq 1$ (Dasgupta et al., 2009; Woodruff & Zhang, 2013; Cohen & Peng, 2015).

[*]Equal contribution [1]IBM Research - Almaden, San Jose, California, USA [2]Carnegie Mellon University, Pittsburgh, Pennsylvania, USA. Correspondence to: Ruosong Wang <ruosongw@andrew.cmu.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

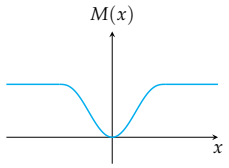


Figure 1. The Tukey loss function.

In this paper we consider more general overconstrained regression problems: given an $n \times d$ matrix A with $n \gg d$, and an $n \times 1$ vector b, output a vector $x' \in \mathbb{R}^d$ for which $\|Ax' - b\|_M \leq (1+\varepsilon)\min_x \|Ax - b\|_M$, where for an n-dimensional vector y and a function $M : \mathbb{R} \to \mathbb{R}^+$, the notation $\|y\|_M$ denotes $\sum_{i=1}^n M(y_i)$. If $M(x) = x^2$ then we have the least squares regression problem, while if $M(x) = |x|$ we have the least absolute deviations problem.

Clarkson and Woodruff showed (Clarkson & Woodruff, 2015b) that for any function $M(\cdot)$ which has at least linear (with positive slope) and at most quadratic growth, as well as some natural other properties, there is a distribution on sketching matrices S with $k = \text{poly}(d \log n)$ rows and corresponding vector $w \in \mathbb{R}^k$, with the following properties. The product $S \cdot A$ can be computed in $\text{nnz}(A)$ time, and if one solves a weighted version of the sketched problem, $\min_x \sum_{i=1}^k w_i M((SAx - b)_i)$, then the minimizer is a constant-factor approximation to the original problem. This gives an algorithm with overall $\text{nnz}(A) + \text{poly}(d)$ running time for the important Huber loss function: given a parameter τ, we have

$$M(x) = \begin{cases} x^2/(2\tau) & |x| \leq \tau \\ |x| - \tau/2 & \text{otherwise} \end{cases}.$$

Unlike least absolute deviations, the Huber loss function is differentiable at the origin, which is important for optimization. However, like least absolute deviations, for large values of x the function is linear, and thus pays less attention to outliers. Other loss functions similar to Huber, such as the $\ell_1 - \ell_2$ and Fair estimators, were also shown to have $\text{nnz}(A) + \text{poly}(d)$ time algorithms. These results were extended to $(1+\varepsilon)$-approximations via sampling-based techniques in (Clarkson & Woodruff, 2015c).

Despite the large body of M-functions handled by previous work, a notable well-studied exception is the Tukey loss function (Fox, 2002), with

$$M(x) = \begin{cases} \frac{\tau^2}{6}(1 - [1 - (\frac{x}{\tau})^2]^3) & |x| \leq \tau \\ \frac{\tau^2}{6} & \text{otherwise} \end{cases}. \quad (1)$$

See Figure 1 for a plot of the Tukey loss function. By a simple Taylor expansion, it is easy to see that $M(x) = \Theta(x^2)$ for $|x| \leq \tau$ and $M(x) = \Theta(\tau^2)$ otherwise. While similar to Huber in the sense that it is quadratic near the origin, it is even less sensitive to outliers than Huber since it is constant when one is far enough away from the origin. Thus, it does not satisfy the linear growth (with positive slope) requirement of (Clarkson & Woodruff, 2015b;c). An important consequence of this distinction is that, while for M-functions with linear growth, a single outlier "at infinity" can radically affect the regression output, this is not the case for the Tukey loss function, due to the bound on its value.

Although the Tukey loss function is not convex, a local minimum of it can be found via iteratively reweighted least squares (Fox, 2002). Also, the dimensionality reduction approach still makes sense for Tukey: here a large non-convex problem is reduced to a smaller non-convex one. Our reduction of a non-convex problem to a smaller one is arguably even more interesting than reducing the size of a convex problem, since inefficient algorithms may now be efficient on the much smaller problem.

Notation. For a matrix $A \in \mathbb{R}^{n \times d}$, we use $A_{i,*}$ to denote its i-th row, $A_{*,j}$ to denote its j-th column, and $A_{i,j}$ to denote a specific entry. For a set of indices $\Gamma \subseteq [n]$, we use $A_{\Gamma,*}$ to denote the submatrix of A formed by all rows in Γ. Similarly, we use $A_{*,\Gamma}$ to denote the submatrix formed by all columns in Γ. For a matrix $A \in \mathbb{R}^{n \times d}$ and a vector $b \in \mathbb{R}^n$, we use $[A\ b] \in \mathbb{R}^{n \times (d+1)}$ to denote the matrix whose first d columns are A and the last column is b. For a matrix $A \in \mathbb{R}^{n \times d}$, we use $\mathbf{im}(A) = \{Ax \mid x \in \mathbb{R}^d\}$ to denote the column span of A.

1.1. Our Assumptions

Before stating our results, we give the general assumptions our algorithms and analyses need. We need the following assumptions on the loss function.

Assumption 1. *There exist real numbers $\tau \geq 0$, constants $p \geq 1$ and $0 < L_M \leq 1 \leq U_M$ such that the function $M : \mathbb{R} \to \mathbb{R}^+$ satisfies:*

1. *Symmetric: $M(a) = M(-a)$ for all a.*
2. *Nondecreasing: $M(a) \geq M(a')$ for $|a| \geq |a'|$.*
3. *Growth condition: for $|a| \geq |a'|$,*
$$\left|\frac{a}{a'}\right|^p \geq \frac{M(a)}{M(a')}.$$
4. *Nearly p th power: for all $|a| \leq \tau$,*
$$L_M|a|^p \leq M(a) \leq U_M|a|^p.$$
5. *Mostly flat: $M(a) = \tau^p$ for $|a| \geq \tau$.*

The conditions in Assumption 1 state that our loss function essentially behaves as an ℓ_p loss function $M(a) = |a|^p$ for $a \leq \tau$, at which point $M(a) = \tau^p$. However, the conditions are more robust in that $M(a)$ just needs to agree with $|a|^p$ up to a fixed constant factor. This is a non-trivial extension of the ℓ_p loss function since we will obtain $(1+\varepsilon)$-approximations in our algorithms, and consequently cannot simply replace the M loss function in our problem with an ℓ_p loss function, as this would result in a constant factor loss. Moreover, such an extension is essential for capturing common robust loss functions, such as the Tukey loss function (1) above: $M(a) = a^2(1-(1-(a/\tau)^2)^3)$ if $|a| < \tau$, and $M(a) = \tau^2$ for $|a| \geq \tau$ (we note that sometimes this loss function is divided by the number 6, but this plays no role from a minimization perspective). Note that the Tukey loss function $M(a)$ does not coincide with the ℓ_2 loss function for $|a| \leq \tau$, though it is within a constant factor of it so our conditions can handle it. For a recent use of this loss function for regression in applications to deep learning, see, e.g., (Belagiannis et al., 2015).

For $\tau = \infty$, we indeed have $M(a) = |a|^p$ and ℓ_p loss functions are widely studied for regression. The case $p = 2$ is just ordinary least squares regression. For $1 \leq p < 2$, the ℓ_p loss function is less sensitive than least squares since one is not squaring the errors, and therefore for such p the loss function is considered to be more robust than ordinary least squares. For $p > 2$, and in particular large values of p, the ℓ_p regression solution approaches the ℓ_∞ regressioin solution, which minimizes the maximum error. The ℓ_p loss functions for every $p \geq 1$ are well-studied in the context of regression, see, e.g., (Clarkson, 2005; Dasgupta et al., 2009; Clarkson et al., 2016; Woodruff & Zhang, 2013; Cohen & Peng, 2015). Loss functions with the polynomial growth condition (Assumption 1.3) are also well-studied (Clarkson & Woodruff, 2015b;c). We also note that already for ℓ_p loss functions, all known dimensionality reduction techniques reduce the original regression problem to a problem of size at least $d^{\Omega(p)}$, which is recently shown to be necessary (Li et al., 2019). Consequently, it is impossible to obtain a considerable dimensionality reduction when p is allowed to grow with n. Since this is a special case of our more general setting of arbitrary τ, we restrict our attention to p that does not grow with n.

For general τ and p, we obtain the Tukey ℓ_p loss function $M(a) = |a|^p$ for $a \leq \tau$, and $M(a) = \tau^p$ for $a \geq \tau$. Note that for $p = 1$, the loss function is the maximum likelihood estimator in the presence of i.i.d. Laplacian noise, while for $p = 2$, the loss function is the maximum likelihood estimator in the presence of i.i.d. Gaussian noise. Thus we obtain the first dimensionality reduction for loss functions that correspond to maximum likelihood estimators of classical noise models where the loss function completely saturates beyond some threshold τ. This is particularly useful for handling large outliers.

We will also need the following assumptions on the input, which we justify below.

Assumption 2. *We assume:*

1. *For given $C_1 \leq \text{poly}(n)$, there is $U = n^{O(d^2)}$ such that $\|\hat{x}\|_2 \leq U$ for any C_1-approximate solution $\hat{x}$ of $\min_x \|Ax-b\|_M$.*
2. *The columns of A and b have ℓ_2 norms in $n^{O(d)}$.*
3. *The threshold $\tau = \Omega(1/n^{O(d)})$.*

As we will show, Assumption 2.1 holds when 2.2 and 2.3 hold, and the entries of A are integers. Alternatively, Assumption 2.1 might hold due to the particular input given, or to an additional explicit problem constraint, or as a consequence of regularization.

We need such a bound on the magnitude of the entries of the Tukey regression optimum, since as the following example shows, they can grow quite large, and behave quite differently from ℓ_p regression solutions. Here we use the case $p = 2$ as an example.

Suppose A is the $n \times 2$ matrix

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & \varepsilon \\ 0 & \varepsilon \\ \vdots & \\ 0 & \varepsilon \end{bmatrix}$$

with $n-2$ rows of $[0\ \varepsilon]$ for $\varepsilon > 0$. Suppose $b \in \mathbb{R}^n$ is the vector of all ones. Then the ℓ_2 regression optimum $x^* = \text{argmin}_x \|Ax-b\|_2^2$ has $\|Ax^*-b\|_2^2$ at most n, since $x = 0$ has that cost. So $n \geq \|Ax^*-b\|_2^2 >= (x_1^*-1)^2 + (x_1^*+x_2^*-1)^2$, from the first two rows, so $(x_1^*)^2 = O(n)$, implying that also $(x_2^*)^2 = O(n)$.

It can also be shown that when $\varepsilon < 1/n$, we have the entries of the ℓ_2 regression optimum x^* in $O(1)$: as x_1 and x_2 get larger, they contribute to the cost, due to the first two rows, more rapidly than the decrease in the $(n-2)(1-x_2/n)^2$ cost due to the $n-2$ last rows.

However, for the Tukey loss function $\|Ax-b\|_M$ with parameter $\tau = 1/2$ and $p = 2$, the cost for $x = [1\ 1/\varepsilon]$ is at most a constant, since the contributions for all rows of A but the second is zero, and contribution made by the second row is at most a constant. However if $x_2 < 1/(2\varepsilon)$, the cost is $\Omega(n)$, since all but the first two rows contribute $\Omega(1)$ to the cost. Thus the optimal x for the Tukey regression has entries $\Omega(1/\varepsilon)$, and no x with entries $o(1/\varepsilon)$ is a constant-factor approximation.

Indeed, given an upper bound on the entries of x, for any $n' < n-2$ there is a large enough version of our example such that no x satisfying that bound can be within an n' factor of optimal for the Tukey regression problem. This example is in fact a near-optimal separation, as one can show that the ℓ_2 regression solution always provides an $O(n)$-approximate solution when $p = 2$.

1.2. Our Contributions

We show that under the assumptions mentioned above, it is possible to obtain dimensionality reductions for Tukey regression. All of our results hold with arbitrarily large constant probability, which can be amplified by independent repetitions and taking the best solution found.

Row Sampling Algorithm. We first give a row sampling algorithm for Tukey regression.

Theorem 1.1. *Given matrix $A \in \mathbb{R}^{n\times d}$ and vector $b \in \mathbb{R}^n$, there is an algorithm that constructs a weight vector w in $\widetilde{O}(\mathrm{nnz}(A) + d^{p/2}\cdot \mathrm{poly}(d\log n)/\varepsilon^2)$[1] time with $\|w\|_0 \le \widetilde{O}(d^{p/2}\cdot\mathrm{poly}(d\log n)/\varepsilon^2)$, for which if $x^*_{M,w}$ is the minimizer to $\min\sum_{i=1}^n w_i M((Ax-b)_i)$, then*

$$\|Ax^*_{M,w} - b\|_M \le (1+\varepsilon)\min_x \|Ax-b\|_M,$$

where M is the Tukey loss function.

Since one can directly ignore those rows $A_{i,*}$ with $w_i = 0$ when solving $\min\sum_{i=1}^n w_i M((Ax-b)_i)$, our row sampling algorithm actually reduces a Tukey regression instance to a weighted version of itself with $\widetilde{O}(d^{p/2}\cdot\mathrm{poly}(d\log n)/\varepsilon^2)$ rows. Notably, the running time of the algorithm and the number of rows in the reduced problem match those given by Lewis weights sampling (Cohen & Peng, 2015), up to $\mathrm{poly}(d\log n)$ factors. However, Lewis weights sampling is designed specifically for the ℓ_p norm, which is a simple special case of the Tukey loss function where $\tau = \infty$.

Our reduction is from the Tukey regression problem to a smaller, weighted version of itself, and since known heuristics for Tukey regression can also handle weights, we can apply them to the reduced problem as well.

Oblivious Sketch. While the row sampling algorithm produces a $(1+\varepsilon)$-approximate solution, where ε can be made arbitrarily small, the algorithm does have some properties that can be a disadvantage in some settings: it makes $\mathrm{polylog}(n)$ passes over the matrix A and vector b, and the rows chosen depend on the input. In the setting of streaming algorithms and distributed computation, on the other hand, a *sketch-and-solve* approach can be more effective. We give such an approach to Tukey regression.

Theorem 1.2. *When $1 \le p \le 2$, there is a distribution $S \in \mathbb{R}^{r\times n}$ over sketching matrices with $r = \mathrm{poly}(d\log n)$, and a corresponding weight vector $w \in \mathbb{R}^r$, for which $S\cdot A$ and $S\cdot b$ can be computed in $O(\mathrm{nnz}(A))$ time and for which if $x^*_{S,M,w}$ is the minimizer to $\min\sum_{i=1}^r w_i M((SAx-Sb)_i)$, then*

$$\|Ax^*_{S,M,w} - b\|_M \le O(\log n)\min_x \|Ax-b\|_M,$$

where M is the Tukey loss function.

Our sketching matrices S are *oblivious*, meaning that their distribution *does not* depend on the data matrix A and the vector b. Furthermore, applying the sketching matrices requires only one pass over these inputs, and thus can be readily implemented in streaming and distributed settings.

We further show that the same distribution S on sketching matrices gives a fixed constant $C \ge 1$ approximation factor if one slightly changes the regression problem solved in the reduced space.

We also remark that for oblivious sketches, the condition that $p \le 2$ is necessary, as shown in (Braverman & Ostrovsky, 2010).

Hardness Results and Provable Algorithms. We give a reduction from MAX-3SAT to Tukey regression, which implies the NP-Hardness of Tukey regression. Under the Exponential Time Hypothesis (Impagliazzo & Paturi, 2001), using Dinur's PCP Theorem (Dinur, 2007), we can strengthen the hardness result and show that even solving Tukey regression with approximation ratio $1+\eta$ requires $2^{\widetilde{\Omega}(d)}$ time for some fixed constant $\eta > 0$.

We complement our hardness results by giving an exponential time algorithm for Tukey regression, using the polynomial system verifier (Renegar, 1992; Basu et al., 1996). This technique has been used to solve a number of numerical linear algebra problems in previous works (Song et al., 2017; Razenshteyn et al., 2016; Clarkson & Woodruff, 2015a; Arora et al., 2016; Moitra, 2016). For the loss function defined in (1), the algorithm runs in $2^{O(n\log n)}\cdot\log(1/\varepsilon)$ time to find a $(1+\varepsilon)$-approximate solution of an instance of size $n\times d$. By further applying our dimensionality reduction methods, the running time can be reduced to $2^{\mathrm{poly}(d\log n)}$, which is significantly faster when $n \gg d$ and comes close to the $2^{\widetilde{\Omega}(d)}$ running time lower bound.

Empirical Evaluation. We test our dimensionality reduction methods on both synthetic datasets and real datasets. Our empirical results quite clearly demonstrate the practicality of our methods.

[1] Throughout the paper we use $\widetilde{O}(f)$ to denote $f\,\mathrm{polylog}\,f$, and $\widetilde{\Omega}(f)$ to denote $f/\mathrm{polylog}\,f$.

2. Technical Overview

In this section, we give an overview of our technical contributions. For convenience, we state our high-level ideas in terms of the loss function

$$M(x) = \begin{cases} x^2 & |x| \leq 1 \\ 1 & |x| \geq 1 \end{cases}$$

where $p = 2$ and $\tau = 1$. We show how to generalize our ideas to other loss functions that satisfy Assumption 1 later.

2.1. Structural Theorem and Algorithms for Finding Heavy Coordinates

Our first main technical contribution is the following structural theorem, which is crucial for each of our algorithms.

Theorem 2.1 (Informal)**.** *For a given matrix $A \in \mathbb{R}^{n\times d}$ and $\alpha \geq 1$, there exists a set of indices $I \subseteq [n]$ with size $|I| \leq \widetilde{O}(d\alpha)$, such that for all $y \in \mathbf{im}(A)$, if y satisfies $\|y\|_M \leq \alpha$ then $\{i \in n \mid |y_i| > 1\} \subseteq I$.*

Intuitively, Theorem 2.1 states that for all vectors y in the column space of A with small $\|y\|_M$, the heavy coordinates of y (coordinates with $|y_i| \geq 1$) must lie in a set I with small cardinality. To prove Theorem 2.1, in Figure 2 we give an informal description of our algorithm for finding the set I. The formal description of the algorithm can be found in the supplementary material.

1. Let $I = \emptyset$.
2. Repeat the following for α times:
   (a) Calculate the leverage scores $\{u_i\}_{i\in[n]\setminus I,*}$ of the matrix $A_{[n]\setminus I,*}$.
   (b) For each $i \in [n] \setminus I$, if $u_i \geq \Omega(1/\alpha)$, then add i into I.
3. Return I.

Figure 2. Polynomial time algorithm for finding heavy coordinates.

For correctness, we first notice that for a vector y with $\|y\|_M \leq \alpha$, the number of heavy coordinates is at most α, since $M(y_i) = 1$ for all $|y_i| > 1$. Now consider the coordinate i with largest $|y_i|$ and $|y_i| > 1$. We must have $\|y\|_2^2 \leq \alpha + \alpha y_i^2$, since the contribution of coordinates with $|y_i| \leq 1$ to $\|y\|_2^2$ is upper bounded by $\|y\|_M \leq \alpha$, and there are at most α coordinates with $|y_i| > 1$, each contributing at most y_i^2 to $\|y\|_2^2$. Now we claim that we must add the coordinate i with largest $|y_i|$ into the set I, which simply follows from

$$\frac{y_i^2}{\|y\|_2^2} \geq \frac{y_i^2}{\alpha + \alpha y_i^2} \geq \Omega(1/\alpha) \tag{2}$$

and thus the leverage score of the row $A_{i,*}$ is at least $\Omega(1/\alpha)$. (Here we use that the i-th leverage score is at least as large as $y_i^2/\|y\|_2^2$ for all $y \in \mathbf{im}(A)$.) After adding i into I, we consider the second largest $|y_i|$ with $|y_i| \geq 1$. A similar argument shows that we will also add i into I in the second repetition. After repeating α times we will add all coordinates i with $|y_i| > 1$ into I, and all coordinates added to I have leverage score $\Omega(1/\alpha)$.

The above algorithm has two main drawbacks. First of all, it returns a set with size $|I| \leq O(d\alpha^2)$ as opposed to $\widetilde{O}(d\alpha)$. Moreover, the algorithm runs in $O(\mathrm{nnz}(A) \cdot \alpha)$ time since we need to calculate the leverage scores of $A_{[n]\setminus I,*}$ a total of α times. When $\alpha = \mathrm{poly}(d)$, such an algorithm does not run in input-sparsity time. An input-sparsity time algorithm for finding such a set I, on the other hand, is an important subroutine for our input-sparsity time row sampling algorithm. In the supplementary material, we give a randomized algorithm for finding a set I with size $|I| \leq \widetilde{O}(d\alpha)$ that runs in input-sparsity time, and we give an informal description of the algorithm in Figure 3. Notice that calculating leverage scores of the matrices $A_{\Gamma_j,*}$ can be done in $\widetilde{O}(\mathrm{nnz}(A) + \mathrm{poly}(d))$ time using existing approaches (Clarkson & Woodruff, 2013; Nelson & Nguyen, 2012).

1. Let $I = \emptyset$.
2. Repeat the following for $O(\log(d\alpha))$ times:
   (a) Randomly partition $[n]$ into $\Gamma_1, \Gamma_2, \ldots, \Gamma_\alpha$.
   (b) For each $j \in [\alpha]$, calculate the leverage scores $\{u_i\}_{i\in\Gamma_j}$ of the matrix $A_{\Gamma_j,*}$.
   (c) For each $j \in [\alpha]$, for each $i \in \Gamma_j$, if $\hat{u}_i \geq \Omega(1)$, then add i to I.
3. Return I.

Figure 3. Input-sparsity time algorithm for finding heavy coordinates.

For correctness, recall that we only need to find those coordinates i for which there exists a vector $y \in \mathbf{im}(A)$ with $\|y\|_M \leq \alpha$ and $|y_i| \geq 1$. Since $\|y\|_M \leq \alpha$, there are most α coordinates in y with absolute value at least 1. Thus, with constant probability, the coordinate i is in a set Γ_j such that it is the only coordinate with $|y_i| \geq 1$ in Γ_j. Moreover, by Markov's inequality, with constant probability the squared

ℓ_2 norm of coordinates in $\Gamma_j \setminus \{i\}$ is at most a constant. Conditioned on these events, using an argument similar to (2), the leverage score of the row $A_{i,*}$ in $A_{\Gamma_j,*}$ is at least a constant, in which case we will add i into I. In order to show that we will add all such i into I with good probability, we repeat the whole procedure for $O(\log(d\alpha))$ times and apply a union bound over all such i. $O(\log(d\alpha))$ repetitions suffice since there are at most $O(\mathrm{poly}(d\alpha))$ different such i, as implied by the existential result mentioned above.

The above algorithm also implies the existence of a set I with better upper bounds on $|I|$, by the probabilisitic method. These algorithms can be readily generalized to general $\tau > 0$, and any $p \geq 1$ using ℓ_p *Lewis weights* in place of leverage scores. We also give a brief overview of Lewis weights and related properties in the supplementary material for readers unfamiliar with these topics.

2.2. The Net Argument

Our second technical contribution is a net argument for Tukey loss functions. Due to the lack of scale-invariance, the net size for the Tukey loss functions need not be $n^{\mathrm{poly}(d)}$. While the M-functions in (Clarkson & Woodruff, 2015b) also do not satisfy scale-invariance, the M-functions in (Clarkson & Woodruff, 2015b) have at least linear growth and so for any value c, and for an orthonormal basis U of A, the set of x for which $\|Ux\|_M = c$ satisfy $c/\mathrm{poly}(n) \leq \|x\|_2 \leq c \cdot \mathrm{poly}(n)$, and so one could use $O(\log n)$ nested nets for the ℓ_2 norm to obtain a net for the M-functions. This does not hold for the Tukey loss function M, e.g., if $c = \tau$, and if the first column of U is $(1, 0, \ldots, 0)^T$, then if $x_1 = \infty$ and $x_2 = x_3 = \cdots = x_d = 0$, one has $\|Ux\|_M = c$. This motivates Assumption 2 above.

Using Assumption 2, we construct a net $\mathcal{N}_\varepsilon$ with size $|\mathcal{N}_\varepsilon| \leq (n/\varepsilon)^{\mathrm{poly}(d)}$, such that for any $y = Ax - b$ with $\|x\|_2 \leq U = n^{\mathrm{poly}(d)}$, there exists $y' \in \mathcal{N}_\varepsilon$ with $\|y' - y\|_M \leq \varepsilon$. The construction is based on a standard volume argument. Notice that such a net only gives an *additive error* guarantee. To give a *relative error* guarantee, we notice that for a vector $y = Ax - b$ with sufficiently small $\|y\|_M$, we must have $\|y\|_\infty \leq 1$, in which case the Tukey loss function $\|\cdot\|_M$ behaves similarly to the squared ℓ_2 norm $\|\cdot\|_2^2$, and thus we can instead use the net construction for the ℓ_2 norm. This idea can be easily generalized to general $p \geq 1$ and $\tau > 0$ if the loss function satisfies $M(x) = |x|^p$ when $|x| \leq \tau$.

To cope with other loss functions that satisfy Assumption 1 for which $M(x)$ can only be *approximated* by $|x|^p$ when $|x| \leq \tau$, we use the nested net construction in (Clarkson & Woodruff, 2015b) when $\|y\|_M$ is sufficiently small. Our final net argument for Tukey loss functions is a careful combination of the two net constructions mentioned above. The full details are given in the supplementary material.

1. Use the algorithm in Figure 3 to find a set I with $\alpha = \mathrm{poly}(d \log n/\varepsilon)$.
2. Calculate the leverage scores $\{u_i\}$ of the matrix $A_{[n]\setminus I,*}$.
3. For each row $A_{i,*}$, we define its sampling probability p_i to be
$$p_i = \begin{cases} 1 & i \in I \\ \min\{1, 1/2 + u_i \,\mathrm{poly}(d/\varepsilon)\} & i \notin I \end{cases}.$$
4. Sample each row with probability p_i.
5. Recursively call the algorithm on the resulting matrix until the number of remaining rows is at most $\mathrm{poly}(d \log n/\varepsilon)$.

Figure 4. The row sampling algorithm.

2.3. The Row Sampling Algorithm

Our row sampling algorithm proceeds in a recursive manner, and employs a combination of *uniform sampling* and *leverage score sampling*, together with the procedure for finding heavy coordinates. We give an informal description in Figure 4. See the supplementary material for the formal description and analysis.

For a vector $y = Ax - b$, we conceptually split coordinates of y into two parts: heavy coordinates (those with $|y_i| > 1$) and light coordinates (those with $|y_i| \leq 1$). Intuitively, we need to apply uniform sampling to heavy coordinates, since all heavy coordinates contribute the same to $\|y\|_M$, and leverage score sampling to light coordinates, since the Tukey loss function behaves similarly to the squared ℓ_2 norm for light coordinates.

In the formal analysis given in supplementary material, we show that if either the contribution from heavy coordinates to $\|y\|_M$ or the contribution from light coordinates to $\|y\|_M$ is at least $\Omega(\mathrm{poly}(d \log n/\varepsilon))$, then with high probability, uniform sampling with sampling probability $1/2$ will preserve $\|y\|_M$ up to ε relative error, for all vectors y in the net. The proof is based on standard concentration inequalities.

If both the contribution from heavy coordinates and the contribution from light coordinates is $O(\mathrm{poly}(d \log n/\varepsilon))$, uniform sampling will no longer be sufficient, and we resort to the structural theorem in such cases. By setting $\alpha = \mathrm{poly}(d \log n/\varepsilon)$ in the algorithm for finding heavy coordinates, we can identify a set I with size $|I| = \mathrm{poly}(d \log n/\varepsilon)$, which includes the indices of all heavy

coordinates. We simply keep all coordinates in I by setting $p_i = 1$. The remaining coordinates must be light, and hence behave very similarly to the squared ℓ_2 norm. Thus, we can use leverage score sampling to deal with the remaining light coordinates. This also explains why we need to use a combination of uniform sampling and leverage score sampling.

Our algorithm will eliminate roughly half of the coordinates in each round, and after $O(\log n)$ rounds there are at most $O(\mathrm{poly}(d \log n/\varepsilon))$ remaining coordinates, in which case we stop the sampling process and return our reduced version of the problem. In each round we calculate the leverage scores and call the algorithm in Figure 3 to find heavy coordinates. Since both subroutines can be implemented to run in $\widetilde{O}(\mathrm{nnz}(A) + \mathrm{poly}(d \log n/\varepsilon))$ time, the overall running time of our row sampling algorithm is also $\widetilde{O}(\mathrm{nnz}(A) + \mathrm{poly}(d \log n/\varepsilon))$.

The above algorithm can be readily generalized to any loss function M that satisfies Assumption 1. Our formal analysis in supplementary material is a careful combination of all ideas mentioned above.

2.4. The Oblivious Sketch

From an algorithmic standpoint, our oblivious sketch S is similar to that in (Clarkson & Woodruff, 2015b). The distribution on matrices S can be viewed roughly as a stack of $h_{\max} = O(\log n)$ matrices, where the i-th such matrix is the product of a CountSketch matrix with $\mathrm{poly}(d \log n)$ rows with a diagonal matrix D which samples roughly $1/(d \log n)^i$ uniformly random coordinates of an n-dimensional vector. Thus, S can be viewed as applying CountSketch to a subsampled set of coordinates of a vector, where the subsampling is more aggressive as i increases. The weight vector w is such that $w_j = (d \log n)^i$ for all coordinates j corresponding to the i-th matrix in the stack. Our main technical contribution here is showing that this simple sketch actually works for Tukey loss functions. One of the main ideas in (Clarkson & Woodruff, 2015b) is that if there is a subset of at least $\mathrm{poly}(d) \log n$ coordinates of a vector y of similar absolute value, then in one of the levels of subsampling of S, with probability $1 - 1/n^{\mathrm{poly}(d)}$ there will be $\Theta(\mathrm{poly}(d) \log n)$ coordinates in this group which survive the subsampling and are *isolated*, that is, they hash to separate CountSketch buckets. Using that the M-function does not grow too quickly, which holds for Tukey loss functions as well if $p \leq 2$, this suffices for estimating the contribution to $\|y\|_M$ from all large subsets of similar coordinates.

The main difference in this work is how estimates for *small subsets of coordinates* of y are made. In (Clarkson & Woodruff, 2015b) an argument based on leverage scores sufficed, since, as one ranges over all unit vectors of the form $y = Ax - b$, there is only a small subset of coordinates which could ever be large, which follows from the condition that the column span of A is a low-dimensional subspace. At first glance, for Tukey loss functions this might not be true. One may think that for any $t \leq \mathrm{poly}(d) \log n$, it could be that for a vector $y = Ax - b$, any subset T of t of its n coordinates could have the property that $M(y_i) = 1$ for $i \in T$, and $M(y_i) < 1$ otherwise. However, our structural theorem in fact precludes such a possibility. The structural theorem implies that there are only $\mathrm{poly}(d \log n)$ coordinates for which $M(y_i)$ could be 1. For those coordinates with $M(y_i) < 1$, the Tukey loss function behaves very similarly to the squared ℓ_2 norm, and thus we can again use the argument based on leverage scores. After considering these two different types of coordinates, we can now apply the perfect hashing argument as in (Clarkson & Woodruff, 2015b).

These ideas can be readily generalized to general $\tau > 0$, and any $1 \leq p \leq 2$, again using ℓ_p Lewis weights in place of leverage scores. We formalize these ideas in the supplementary material.

3. Experiments

In this section we provide experimental results to illustrate the practicality of our dimensionality reduction methods. Figure 5 shows the approximation ratio of our dimensionality reduction methods, when applied to synthetic and real datasets. For all datasets, the number of data points is $n = 10000$. The dimension d is different for different datasets and is marked in Figure 5. We adopt the loss function defined in (1) and use different values of τ for different datasets. To calculate the approximation ratio of our dimensionality reduction methods, we solve the full problems and their sketched counterparts by using the LinvPy software (lin). This software uses iteratively re-weighted least squares (IRLS), and we modify it for $\|\cdot\|_{M,w}$, which requires only to include a "fixed" weighting from w into the IRLS solver.

The Random Gaussian dataset is a synthetic dataset, whose entries are sampled i.i.d. from the standard Gaussian distribution. The remaining datasets are chosen from the UCI Machine Learning Repository. The τ values were chosen roughly so that there would be significant clipping of the residuals. For each dataset, we also randomly select 5% of the entries of the b vector and change them to 10^4, to model outliers. Such modified datasets are marked as "with outliers" in Figure 5.

We tested both the row sampling algorithm and the oblivious sketch. We varied the size of the sketch from $2d$ to $10d$ (d is the dimension of the dataset) and calculated the approximation ratio $\|A\hat{x} - b\|_M / \|Ax^* - b\|_M$ using the

modified LinvPy software, where x^* is the solution returned by solving the full problem and $\hat{x}$ is the solution returned by solving the sketched version. We repeated each experiment ten times and took the best result among all repetitions.

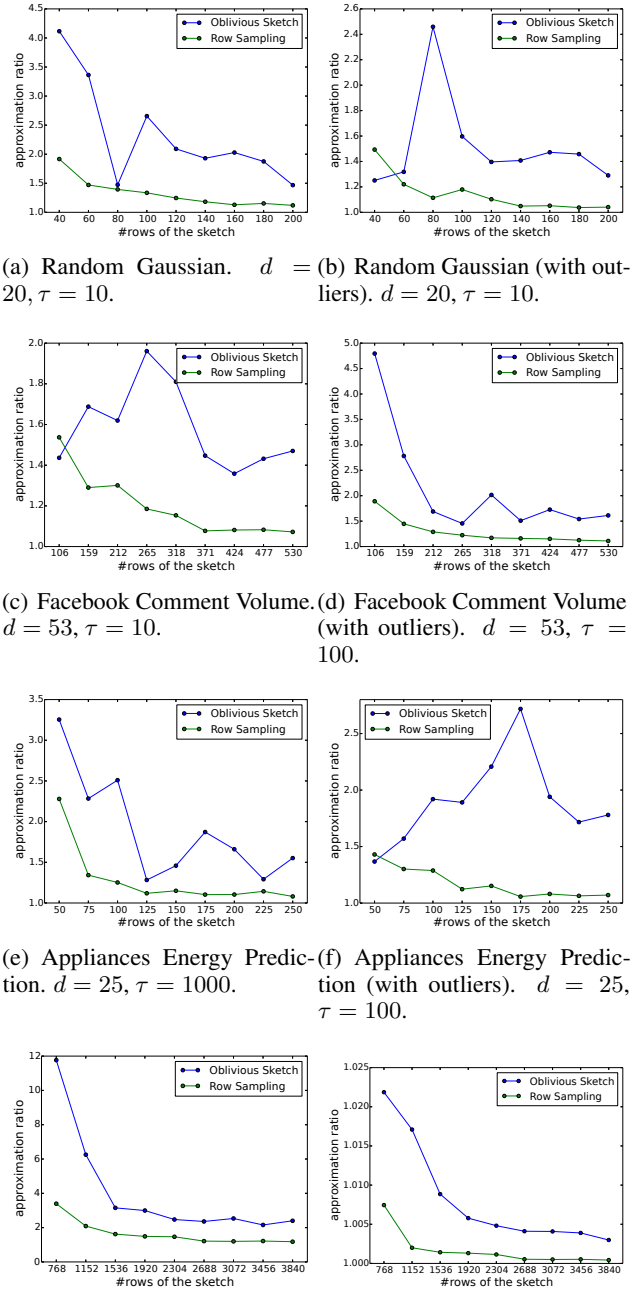
Discussions As can be seen from Figure 5, the row sampling algorithm has better approximation ratio as we increase the sketch size. The same is not always true for the oblivious sketch, since the oblivious sketch only guarantees an $O(\log n)$ approximation instead of a $(1+\varepsilon)$-approximate solution, as returned by the row sampling algorithm. Moreover, the row sampling algorithm consistently outperforms the oblivious sketch in the experiments, except for extremely small sketch sizes (around $2d$). However, applying the oblivious sketch requires only one pass over the input, and the distribution of the sketching matrices does not depend on the input. These advantages make the oblivious sketch preferable in streaming and distributed settings. Another advantage of the oblivious sketch is its simplicity.

Our empirical results demonstrate the practicality of our dimensionality reduction methods. Our methods successfully reduce a Tukey regression instance of size $10000 \times d$ to another instance with $O(d)$ rows, without much sacrifice in the quality of the solution. In most cases, the row sampling algorithm reduces the size to $3d$ rows while retaining an approximation ratio of at most 2.

4. Conclusions

We give the first dimensionality reduction methods for the overconstrained Tukey regression problem. We first give a row sampling algorithm which takes $\widetilde{O}(\mathrm{nnz}(A) + \mathrm{poly}(d \log n/\varepsilon))$ time to return a weight vector with $\mathrm{poly}(d \log n/\varepsilon)$ non-zero entries, such that the solution of the resulting weighted Tukey regression problem gives a $(1+\varepsilon)$-approximation to the original problem. We further give another way to reduce Tukey regression problems to smaller weighted versions, via an oblivious sketching matrix S, applied in a single pass over the data. Our dimensionality reduction methods are simple and easy to implement, and we give empirical results demonstrating their practicality. We also give hardness results showing that the Tukey regression problem cannot be efficiently solved in the worst-case.

From a technical point of view, our algorithms for finding heavy coordinates and our structural theorem seem to be of independent interest. We leave it as an intriguing open problem to find more applications of them.



(a) Random Gaussian. $d = 20, \tau = 10$.

(b) Random Gaussian (with outliers). $d = 20, \tau = 10$.

(c) Facebook Comment Volume. $d = 53, \tau = 10$.

(d) Facebook Comment Volume (with outliers). $d = 53, \tau = 100$.

(e) Appliances Energy Prediction. $d = 25, \tau = 1000$.

(f) Appliances Energy Prediction (with outliers). $d = 25, \tau = 100$.

(g) CT Slice Localization. $d = 384, \tau = 100$.

(h) CT Slice Localization (with outliers). $d = 384, \tau = 1000$.

Figure 5. Approximation ratio of our dimensionality reduction methods.

Acknowledgements

The authors would like to thank Lijie Chen and Peilin Zhong for helpful discussions, and the anonymous ICML reviewers for their insightful comments. Ruosong Wang and David P. Woodruff were supported in part by Office of Naval Research (ONR) grant N00014-18-1-2562. Part of this work was done while the authors were visiting the Simons Institute for the Theory of Computing.

References

The `linvpy` package. `https://github.com/LCAV/linvpy/`.

Arora, S., Ge, R., Kannan, R., and Moitra, A. Computing a nonnegative matrix factorization—provably. *SIAM Journal on Computing*, 45(4):1582–1611, 2016.

Basu, S., Pollack, R., and Roy, M.-F. On the combinatorial and algebraic complexity of quantifier elimination. *Journal of the ACM (JACM)*, 43(6):1002–1045, 1996.

Belagiannis, V., Rupprecht, C., Carneiro, G., and Navab, N. Robust optimization for deep regression. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2830–2838, 2015.

Braverman, V. and Ostrovsky, R. Zero-one frequency laws. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pp. 281–290. ACM, 2010.

Clarkson, K. L. Subgradient and sampling algorithms for ℓ_1 regression. In *SODA*, 2005.

Clarkson, K. L. and Woodruff, D. P. Low rank approximation and regression in input sparsity time. In *STOC*, 2013. Full version at `http://arxiv.org/abs/1207.6365`.

Clarkson, K. L. and Woodruff, D. P. Input sparsity and hardness for robust subspace approximation. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pp. 310–329. IEEE, 2015a.

Clarkson, K. L. and Woodruff, D. P. Sketching for M-estimators: A unified approach to robust regression. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pp. 921–939, 2015b.

Clarkson, K. L. and Woodruff, D. P. Input sparsity and hardness for robust subspace approximation. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pp. 310–329, 2015c.

Clarkson, K. L., Drineas, P., Magdon-Ismail, M., Mahoney, M. W., Meng, X., and Woodruff, D. P. The fast cauchy transform and faster robust linear regression. *SIAM Journal on Computing*, 45(3):763–810, 2016.

Cohen, M. B. and Peng, R. ℓ_p row sampling by Lewis weights. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pp. 183–192, 2015.

Dasgupta, A., Drineas, P., Harb, B., Kumar, R., and Mahoney, M. W. Sampling algorithms and coresets for ℓ_p regression. *SIAM J. Comput.*, 38(5):2060–2078, 2009.

Dinur, I. The pcp theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12, 2007.

Fox, J. *Robust Regression (web appendix)*. Sage Publications, 2002. URL `http://socserv.mcmaster.ca/jfox/books/companion/appendix.html`.

Impagliazzo, R. and Paturi, R. On the complexity of k-sat. *Journal of Computer and System Sciences*, 62(2): 367–375, 2001.

Indyk, P. and Woodruff, D. P. Optimal approximations of the frequency moments of data streams. In *STOC*, pp. 202–208, 2005.

Kannan, R. and Vempala, S. Spectral algorithms. *Foundations and Trends in Theoretical Computer Science*, 4 (3-4):157–288, 2009.

Lewis, D. Finite dimensional subspaces of L_p. *Studia Mathematica*, 63(2):207–212, 1978.

Li, M., Miller, G. L., and Peng, R. Iterative row sampling. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pp. 127–136. IEEE, 2013.

Li, Y., Wang, R., and Woodruff, D. P. Tight bounds for the subspace sketch problem with applications. *arXiv preprint arXiv:1904.05543*, 2019.

Mahoney, M. W. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2): 123–224, 2011.

Meng, X. and Mahoney, M. W. Low-distortion Subspace Embeddings in Input-sparsity Time and Applications to Robust Linear Regression. *ArXiv e-prints*, October 2012.

Moitra, A. An almost optimal algorithm for computing nonnegative rank. *SIAM Journal on Computing*, 45(1): 156–173, 2016.

Nelson, J. and Nguyen, H. L. OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings. *CoRR*, abs/1211.1002, 2012.

Razenshteyn, I., Song, Z., and Woodruff, D. P. Weighted low rank approximations with provable guarantees. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pp. 250–263. ACM, 2016.

Renegar, J. On the computational complexity and geometry of the first-order theory of the reals. part i: Introduction. preliminaries. the geometry of semi-algebraic sets. the decision problem for the existential theory of the reals. *Journal of symbolic computation*, 13(3):255–299, 1992.

Sarlós, T. Improved approximation algorithms for large matrices via random projections. In *FOCS*, pp. 143–152, 2006.

Sohler, C. and Woodruff, D. P. Subspace embeddings for the l_1-norm with applications. In *STOC*, pp. 755–764, 2011.

Song, Z., Woodruff, D. P., and Zhong, P. Low rank approximation with entrywise ℓ_1-norm error. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 688–701. ACM, 2017.

Verbin, E. and Zhang, Q. Rademacher-sketch: A dimensionality-reducing embedding for sum-product norms, with an application to earth-mover distance. In *ICALP (1)*, pp. 834–845, 2012.

Wang, R. and Woodruff, D. P. Tight bounds for ℓ_p oblivious subspace embeddings. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1825–1843. SIAM, 2019.

Wojtaszczyk, P. *Banach spaces for analysts*, volume 25. Cambridge University Press, 1996.

Woodruff, D. P. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.

Woodruff, D. P. and Zhang, Q. Subspace embeddings and l_p-regression using exponential random variables. *CoRR*, abs/1305.5580, 2013.

Yang, J., Meng, X., and Mahoney, M. W. Quantile regression for large-scale applications. In *ICML (3)*, pp. 881–887, 2013.