Dimensionality Reduction for Tukey Regression

Kenneth L. Clarkson, Ruosong Wang and David P. Woodruff

IBM

Carnegie Mellon University

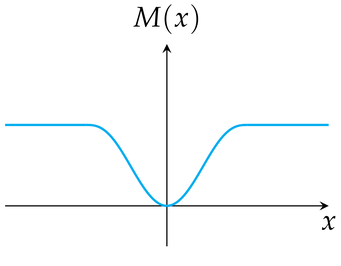
Background

Motivation

- A number of problems in numerical linear algebra have witnessed remarkable speedups via the technique of linear sketching.
- For linear regression, we have $\mathrm{nnz}(A) + \mathrm{poly}(d/\varepsilon)$ time algorithms when the loss function is the ℓ_p norm, the quantile loss function, or M-estimators.
- Can we apply the technique of linear sketching to non-convex loss functions, e.g., the Tukey loss function?

Tukey Loss Function

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 $|a| \geq \tau$.



Row Sampling Algorithm

1. Algorithm for finding heavy coordinates
   (a) Let $I = \emptyset$.
   (b) Repeat the following for α times:
      i. Calculate the leverage scores $\{u_i\}_{i\in[n]\setminus I,*}$ of the matrix $A_{[n]\setminus I,*}$.
      ii. For each $i \in [n] \setminus I$, if $u_i \geq \Omega(1/\alpha)$, then add i into I.
   (c) Return I.
2. Row sampling algorithm
   (a) Use the above algorithm to find a set I with $\alpha = \mathrm{poly}(d \log n/\varepsilon)$.
   (b) Calculate the leverage scores $\{u_i\}$ of the matrix $A_{[n]\setminus I,*}$.
   (c) For each row $A_{i,*}$, we define its sampling probability p_i to be
$$p_i = \begin{cases} 1 & i \in I \\ \min\{1, 1/2 + u_i\,\mathrm{poly}(d\log n/\varepsilon)\} & i \notin I \end{cases}.$$
   (d) Sample each row with probability p_i.
   (e) Recursively call the algorithm on the resulting matrix until the number of remaining rows is at most $\mathrm{poly}(d \log n/\varepsilon)$.

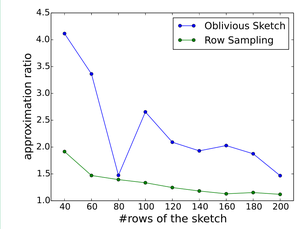
Technical Overview

- For a vector $y \in \mathbb{R}^n$, let $\|y\|_M = \sum_{i=1}^n M(y_i)$ and $\|y\|_{M,w} = \sum_{i=1}^n w_i M(y_i)$.
- Consider the loss function
$$M(x) = \begin{cases} x^2 & |x| \leq 1 \\ 1 & |x| \geq 1 \end{cases}.$$
- **Theorem 1** For a given matrix $A \in \mathbb{R}^{n\times d}$, for any $y = Ax$ such that $\|y\|_M \leq \alpha$, for all $i \in [n]$ such that $|y_i| \geq 1$, we have $i \in I$.
- **Theorem 2** For a matrix $A \in \mathbb{R}^{n\times d}$ and $b \in \mathbb{R}^n$, the algorithm returns a weight vector $w \in \mathbb{R}^n$, such that for $\hat{x} = \mathrm{argmin}\|Ax - b\|_{M,w}$, we have $\|A\hat{x} - b\|_M \leq (1+\varepsilon)\min\|Ax - b\|_M$. The weight vector w has almost $\mathrm{poly}(d \log n/\varepsilon)$ non-zero entries.
- *Proof.* Theorem 1 + concentration bound + net argument.

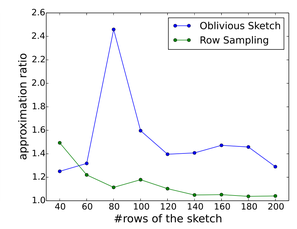
Oblivious Sketch

- **Theorem 3** There is a distribution $S \in \mathbb{R}^{\mathrm{poly}(d\log n)\times n}$ over matrices and weight vector $w \in \mathbb{R}^n$, such that for $\hat{x} = \mathrm{argmin}\|SAx - Sb\|_{M,w}$, we have $\|A\hat{x} - b\|_M \leq O(\log n)\min\|Ax - b\|_M$.
- Can be readily implemented in streaming and distributed settings.

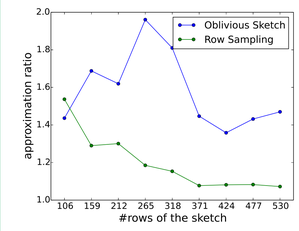
Experiments



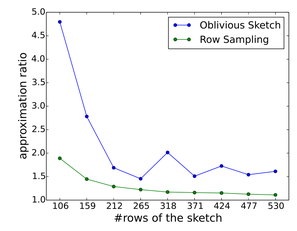
(a) Random Gaussian. $d = 20, \tau = 10$.



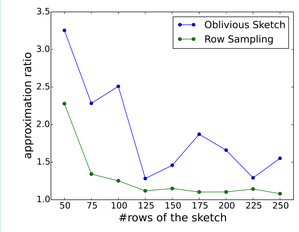
(b) Random Gaussian (with outliers). $d = 20, \tau = 10$.



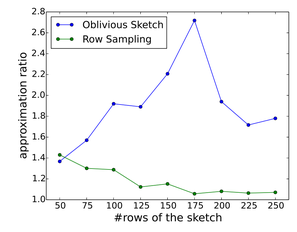
(c) Facebook Comment Volume. $d = 53, \tau = 10$.



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



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



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