Optimal Sketching for Kronecker Product Regression and Low Rank Approximation

Huain Diao[†], Rajesh Jayaram*, Zhao Song[α], Wen Sun[b], and David Woodruff*
CMU*, Northeast Normal University[†], University of Washington[α], MSR New York[b]

Overview

Over-constrained ℓ_p-Norm Regression

Given $\epsilon > 0, A \in \mathbb{R}^{n\times d}$ $b \in \mathbb{R}^n$ where $n \gg d$, compute $x' \in \mathbb{R}^d$ such that

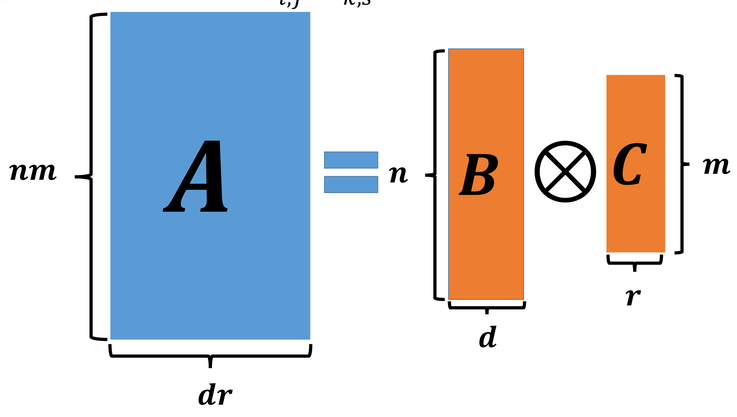
$$\|Ax' - b\|_p^p \leq (1+\epsilon)\min_{x\in\mathbb{R}^d}\|Ax - b\|_p^p \quad (1)$$

- $p = 2$: *Least Squares Regression*
- $p = 1$: *Least Absolute Deviation Regression*

In this work, we utilize **sketching** techniques to obtain fast solutions to (1), when the input matrix A is a *Kronecker Product* of q smaller matrices.

Kronecker Products

Given $B \in \mathbb{R}^{n\times d}$ and $C \in \mathbb{R}^{m\times r}$ Kronecker product $A = B \otimes C \in \mathbb{R}^{nm\times dr}$ consists of all $nmdr$ entry-wise products of the form $B_{i,j} \cdot C_{k,s}$



More generally, given $A_i \in \mathbb{R}^{n_i\times d_i}$, can define $\otimes_{i=1}^q A_i \in \mathbb{R}^{n_1 n_2\cdots n_q \times d_1 d_2 \cdots d_q}$, and minimize:

$$\min_{x\in\mathbb{R}^d}\left\|\left(\otimes_{i=1}^q A_i\right)x - b\right\|_p^p$$

Even forming $\mathrm{A} = \left(\otimes_{i=1}^q A_i\right)$ requires $\prod_i^q n_i d_i$ time! However, [1] shows that a solution x' to (1) can be found in time *sublinear* in the size of A.

Rank Regression: Let $\bar{A} \in \mathbb{R}^{n^2\times d}$ be the matrix of all pair-wise differences of the rows of $A \in \mathbb{R}^{n\times d}$, and similarly define $\bar{b} \in \mathbb{R}^{n^2}$. The goal is to minimize: $\left\|\bar{A}x - \bar{b}\right\|_1$

Low Rank Approximation (LRA): Given $\mathrm{A} = \left(\otimes_{i=1}^q A_i\right)$, find B' such that

$$\|B' - A\|_F^2 \leq (1+\epsilon)\min_{\text{B rank-k}}\|B - A\|_F^2$$

Our Results

Problem	Our Runtime	Prior Runtime
Regression $p = 2$	$\sum_i \boldsymbol{nnz(A_i)}$	$\sum_i \boldsymbol{nnz(A_i)}$ $+\boldsymbol{nnz(b)}$ [1]
Regression $p \in [0,2)$	$\sum_i \boldsymbol{nnz(A_i)}$ $+\boldsymbol{nnz(b)}$	$\boldsymbol{q = 2, n_1 = n_2}$**:** $\boldsymbol{n^{3/2}poly(d_1 d_2)}$ $+\boldsymbol{nnz(b)}$ [1]
All-Pairs Regression	$\boldsymbol{nnz(A)}$ $(\boldsymbol{A \in \mathbb{R}^{n\times d}})$	**LP with n^2 constraints**
LRA	$\sum_i \boldsymbol{nnz(A_i)}$	$\prod_i \boldsymbol{nnz(A_i)}$ [3]

Best known fast LRA algorithm [3] would require $nnz(A) = \prod_i nnz(A_i)$ time! For the related and special case of polynomial kernel LRA, a $\sum_i nnz(A_i)$ algorithm was known [4].

Applications

Kronecker Product Regression arises in Spline Regression, Signal Processing, and Multivariate Data fitting. Many statistical problems can be modeled as Kronecker product regression, such as **rank-regression** (all-pairs regression estimation).

- **Rank Regression:** Robust Estimator, minimizing $\left\|\bar{A}x - \bar{b}\right\|_1$ more robust than ℓ_1: $\min\|Ax - b\|_1$.
  - Highly successful estimator for regression with both heavy tailed and Gaussian error [2].
  - Downside: requires solving a LP with n^2constraints. **This work:** we give much faster algorithms!
  - Observe: $\bar{A} = A \otimes \mathbf{1} - \mathbf{1} \otimes A$ is a Kronecker product!

References

1. Diao, H., Song, Z., Sun, W., & Woodruff, D. P. (2017). Sketching for kronecker product regression and p-splines. In *International Conference on Artificial Intelligence and Statistics*.
2. Wang, L., Kai, B., & Li, R. (2009). Local rank inference for varying coefficient models. Journal of the American Statistical Association.
3. Clarkson, K. L., & Woodruff, D. P. (2017). Low-rank approximation and regression in input sparsity time. Journal of the ACM (JACM).
4. Avron, H., Nguyen, H., & Woodruff, D. (2014). Subspace embeddings for the polynomial kernel. In *Advances in Neural Information Processing Systems*.

Our Results

Our sketching algorithms rely on solving

$$\min_{x\in\mathbb{R}^d}\|S(Ax - b)\|_p^p$$

Where $S \in \mathbb{R}^{s\times n}$, $s \ll n$ is a random sketching matrices which must *depend* on the matrix [A,b].

- For $p = 2$ cannot even read b! Our sketches each factor $R_i A_i$ by random $R_i$to precondition A_i, then estimate and sample from leverage scores without reading b.
- For p<2, our main algorithm involves a multi-part sketching procedure to sample from the *residual error*. First, we show how to compute a rough approximation x' to the optimal. Then, we sample rows $i \in [\prod_i n_i]$ to include in the sketch S from the distribution: $\rho_i \propto |(Ax' - b)_i|^p$

Challenge: computing all τ_i takes $\prod n_i d_i$ time! To avoid this, we utilize tools sketching & sampling tools from streaming:
Precision Sampling: to reduce sampling from τ_i to finding heavy hitters & **Count-sketch with Dyadic Trick** for quickly finding heavy hitters after applying precision sampling.

- Our algorithm for all-pairs $\ell_1$regression proceeds similarly.

Low Rank Approximation: Use *sparse random projections* matrices S_i **to** preserve the cost of projections onto small dim. Subspaces. Then can just compute a good LRA to:

$$\left(\boldsymbol{S_1} \otimes \cdots \otimes \boldsymbol{S_q}\right)\left(\boldsymbol{A_1} \otimes \cdots \otimes \boldsymbol{A_q}\right) = \left(\boldsymbol{S_1 A_1} \otimes \cdots \otimes \boldsymbol{S_q A_q}\right)$$

Experiments

	$\boldsymbol{m}$	m/n	$\boldsymbol{r_e}$	$\boldsymbol{r_e'}$	$\boldsymbol{r_t}$	$\boldsymbol{r_t'}$
ℓ_2	8100	.09	2.48%	1.51%	0.05	0.22
	12100	.13	1.55%	0.98%	0.06	0.24
	16129	.18	1.20%	0.71%	0.07	0.08
ℓ_1	2000	.02	7.72%	9.10%	0.02	0.59
	4000	.04	4.26%	4.00%	0.03	0.75
	8000	.09	1.85%	1.60%	0.07	0.83
	12000	.13	1.29%	0.99%	0.09	0.79
	16000	.18	1.01%	0.70%	0.14	0.90

$r_e = 100\left(\frac{Error_{ours} - Error_{OPT}}{Error_{OPT}}\right), r_e' = 100\left(\frac{Error_{ours} - Error_{[1]}}{Error_{[1]}}\right).$

$r_t = \frac{\text{our runtime}}{\text{brute force}}$, $r_t' = \frac{\text{our runtime}}{\text{alg from [1]}}$ $n_1 = n_2 = 300$, $d_1 = d_2 = 15, n_1 n_2 = 90{,}000$

$m :=$ sketch size (# of rows of matrix S)

Experiments run on synthetic data show dramatic runtime improvements over both brute-force (optimal) and prior algorithm of [1], with only a mild decrease in accuracy. Error rate of $\epsilon \approx .02$ achievable with sketch sizes of $m \approx \frac{n}{10}$ for both ℓ_1 and ℓ_2 regression.