Interaction Hard Thresholding: Consistent Sparse Quadratic Regression in Sub-quadratic Time and Space

Shuo Yang*, Yanyao Shen*, Sujay Sanghavi

Introduction

Going from linear model to quadratic model:

Linear Model $y \sim \boldsymbol{\theta}^\top \mathbf{x}$ Quadratic Model $y \sim \mathbf{x}^\top \boldsymbol{\Theta} \mathbf{x}$

Solving the quadratic regression:

(Quadratic Structure) $$\min_{\boldsymbol{\Theta}:\|\boldsymbol{\Theta}\|_0 \le K} \frac{1}{n}\sum_{i=0}^{n-1} f\left(\mathbf{x}_i^\top \boldsymbol{\Theta}\mathbf{x}_i, y_i\right) \quad := F_n(\boldsymbol{\Theta})$$

Key Ideas and Merits

We list several key ideas of our work, which leads to the merits in both statistics and computation.

IHT for Sparse Regression: With iterative hard thresholding, we can keep the parameter estimation sparse in all iterations. If the $\widehat{\boldsymbol{\Theta}}$ is k-sparse, then only the **top-2k elements in gradient will be useful.**

Fast Gradient Estimation: Since the gradient also has the quadratic structure, finding the top-2k elements in the gradient can be reduced to **finding top-2k elements in matrix multiplication**.

Algorithm Design: Our method (algorithm 1) proceeds by updating the parameter estimation via iterative hard thresholding. The gradient estimation is accomplished by first **approximately find the top-2k elements of the gradient (algorithm 2)** and then calculate the corresponding gradient exactly. This scheme can be combined with various convex optimization algorithm, and we provide detailed analysis for SGD and SVRG in our paper.

Statistical Optimal: Our method yield consistent estimation of the sparse parameter. The sample complexity also matches the optimal for sparse recovery.

Computational Efficiency: Despite the increased dimension, our method achieves an **overall sub-quadratic complexity**. This can be pushed to higher order polynomial regression with little modification.

Algorithm

Algorithm 1 INTERACTION HARD THRESHOLDING (INTHT)

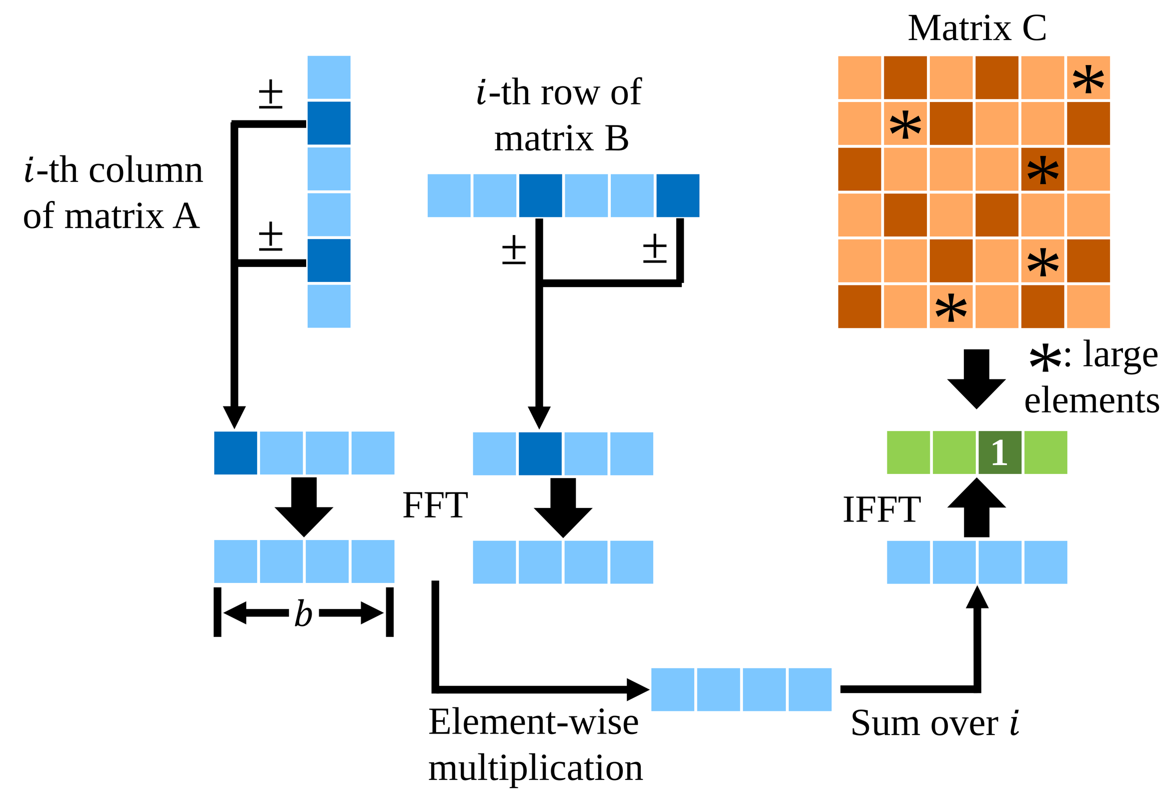
Input: Dataset $\{\mathbf{x}_i, y_i\}_{i=1}^n$, dimension p
Parameters: Step size η, estimation sparsity k, batch size m, round number T
Output: The parameter estimation $\widehat{\boldsymbol{\Theta}}$
Initialize $\boldsymbol{\Theta}^0$ as a $p \times p$ zero matrix.
for $t = 0$ **to** $T-1$ **do**
  Draw a subset of indices $\mathcal{B}_t$ from $[n]$ randomly.
  Calculate the residual $u_i = u(\boldsymbol{\Theta}^t, \mathbf{x}_i, y_i)$ based on for every $i \in \mathcal{B}_t$.
  Set $\mathbf{A}_t \in \mathbb{R}^{p\times m}$, where each column of $\mathbf{A}_t$ is $u_i\mathbf{x}_i$, $i \in \mathcal{B}_t$.
  Set $\mathbf{B}_t \in \mathbb{R}^{p\times m}$, where each column of $\mathbf{B}_t$ is $\mathbf{x}_i$, $i \in \mathcal{B}_t$. (where $\frac{\mathbf{A}_t\mathbf{B}_t^\top}{m}$ gives the gradient)
  Compute $\widetilde{S}_t = \mathrm{ATEE}(\mathbf{A}_t, \mathbf{B}_t, 2k)$. —-/* *approximate top elements extraction* */—-
  Set $S_t = \widetilde{S}_t \cup \mathrm{supp}(\boldsymbol{\Theta}^t)$. —-/* *inaccurate hard thresholding update* */—-
  Compute $\mathcal{P}_{S_t}(\mathbf{G}^t) \leftarrow$ the gradient value $\mathbf{G}^t = \frac{1}{m}\sum_{i\in\mathcal{B}^t} u_i\mathbf{x}_i\mathbf{x}_i^\top$ only calculated on S_t.
  Update $\boldsymbol{\Theta}^{t+1} = \mathcal{H}_k\left(\boldsymbol{\Theta}^t - \eta\mathcal{P}_{S_t}(\mathbf{G}^t)\right)$.
Return: $\widehat{\boldsymbol{\Theta}} = \boldsymbol{\Theta}^T$

Algorithm 2 APPROXIMATED TOP ELEMENTS EXTRACTION (ATEE)

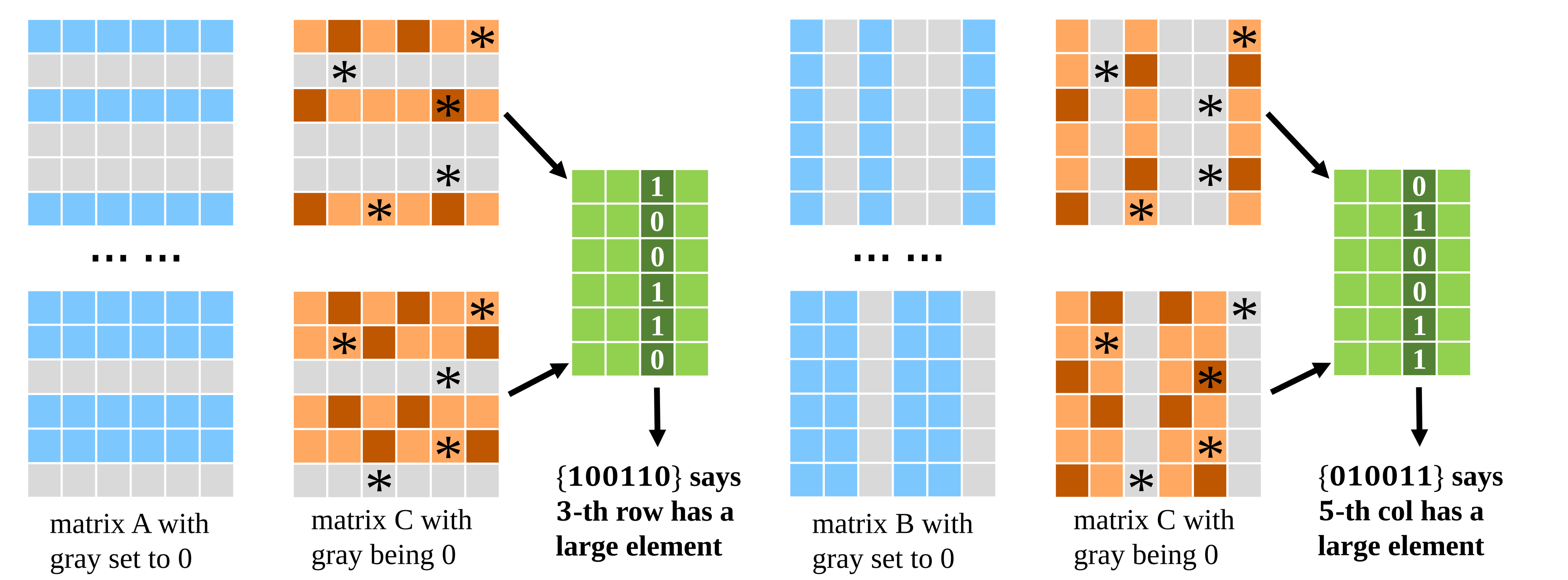
Input: Matrix $\mathbf{A}$, matrix $\mathbf{B}$, top selection size k
Parameters: Output set size upper bound b, repetition number d, significant level Δ
Expected Output: Set Λ: the top-k elements in $\mathbf{AB}^\top$ with absolute value greater than Δ
Output: Set $\widetilde{\Lambda}$ of indices, with size at most b (approximately contains Λ)

Algorithm Illustration

Compressed Matrix Multiplication: Calculate $C = AB$



Sub-linear Elements Extraction[1]: Extract the index of large elements



Main Results

Denote the batch to be $\mathcal{B}$ and the loss function to be $f_{\mathcal{B}}$. Note that we define b to be the output size of ATEE, p to be the input dimension and m to be the batch-size. We then have (adapted from [1]):

Theorem 1 (Recovering top-$2k$ elements of the gradient)**.** *With the setting above, if we choose b, d, Δ so that $b\Delta^2 \ge 432\,\|\nabla f_{\mathcal{B}}(\boldsymbol{\Theta})\|_F^2$ and $d \ge 48\log 2ck$, then the index set $(\widetilde{\Lambda})$ returned by ATEE contains the desired index set (Λ) with probability at least $1 - 1/c$.*

Also in this case the time complexity of ATEE is $\widetilde{\mathcal{O}}\left(m(p+b)\right)$, and space complexity is $\widetilde{\mathcal{O}}\left(m(p+b)\right)$.

In the following result, we keep all previous definition and define k_Δ to be the number of elements in top-$2k$ elements with magnitude below Δ, K is the true sparsity of $\boldsymbol{\Theta}^\star$, and for simplicity, we define $\nu = 1 + (\rho + \sqrt{(4+\rho)\rho})/2$, $\rho = K/k$. Then we have

Theorem 3 (Per-round convergence of IntHT)**.** *Following the above notations, the per-round convergence of Algorithm 1 satisfies the following:*

- *If ATEE succeeds, i.e., $\Lambda \subseteq \widetilde{\Lambda}$, then*

$$\mathbb{E}_{B_t}\left[\left\|\boldsymbol{\Theta}^t - \boldsymbol{\Theta}^\star\right\|_F^2\right] \le \kappa_1 \mathbb{E}_{B_{t-1}}\left[\left\|\boldsymbol{\Theta}^{t-1} - \boldsymbol{\Theta}^\star\right\|_F^2\right] + \sigma_{GD}^2 + \sigma_{\Delta|GD}^2,$$

where $\kappa_1 = \nu\left(1 - 2\eta\alpha_{2k} + 2\eta^2 L_{2k}^2\right)$, $\sigma_{\Delta|GD}^2 = 4\sqrt{k_\Delta}\eta\sqrt{k}\omega\Delta + 2k_\Delta\eta^2\Delta^2$, and

$$\sigma_{GD}^2 = \max_{|\Omega|\le 2k+K}\left[4\nu\eta\sqrt{k}\omega\left\|\mathcal{P}_\Omega\left(\nabla F\left(\boldsymbol{\Theta}^\star\right)\right)\right\|_F + 2\nu\eta^2\mathbb{E}_{\mathcal{B}_t}\left[\left\|\mathcal{P}_\Omega\left(\nabla f_{\mathcal{B}_t}\left(\boldsymbol{\Theta}^\star\right)\right)\right\|_F^2\right]\right].$$

- *If ATEE fails, i.e., $\Lambda \not\subseteq \widetilde{\Lambda}$, then,*

$$\mathbb{E}_{B_t}\left[\left\|\boldsymbol{\Theta}^t - \boldsymbol{\Theta}^\star\right\|_F^2\right] \le \kappa_2 \mathbb{E}_{B_{t-1}}\left[\left\|\boldsymbol{\Theta}^{t-1} - \boldsymbol{\Theta}^\star\right\|_F^2\right] + \sigma_{GD}^2 + \sigma_{Fail|GD}^2,$$

where $\kappa_2 = \kappa_1 + 2\nu\eta L_{2k}$, $\sigma_{Fail|GD}^2 = \max_{|\Omega|\le 2k+K}\left[4\nu\eta\sqrt{k}\omega\mathbb{E}_{\mathcal{B}_t}\left[\left\|\mathcal{P}_\Omega\left(\nabla f_{\mathcal{B}_t}\left(\boldsymbol{\Theta}^\star\right)\right)\right\|_F\right]\right].$

Combining the previous two results, we have the main theorem for convergence analysis:

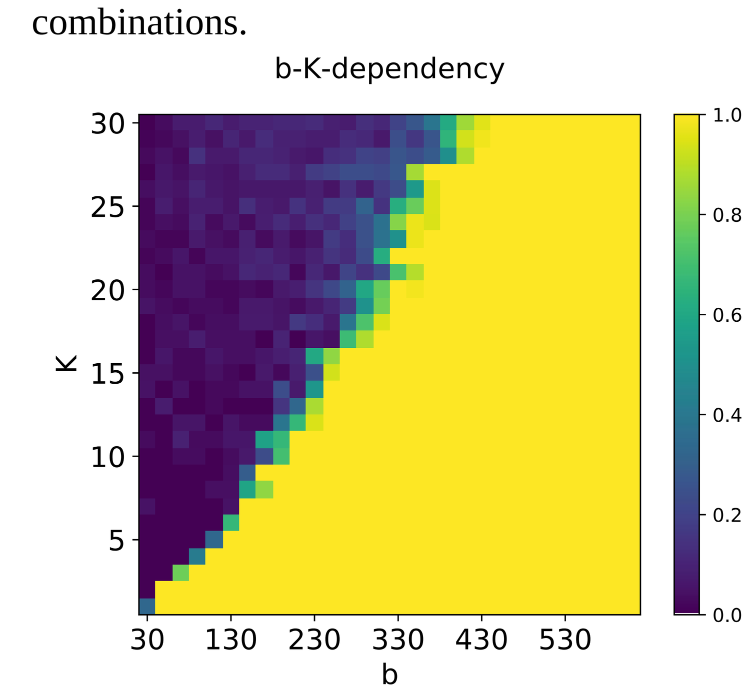
Theorem 4 (Main result)**.** *Following the above notations, the expectation of the parameter recovery error of Algorithm 1 is bounded by*

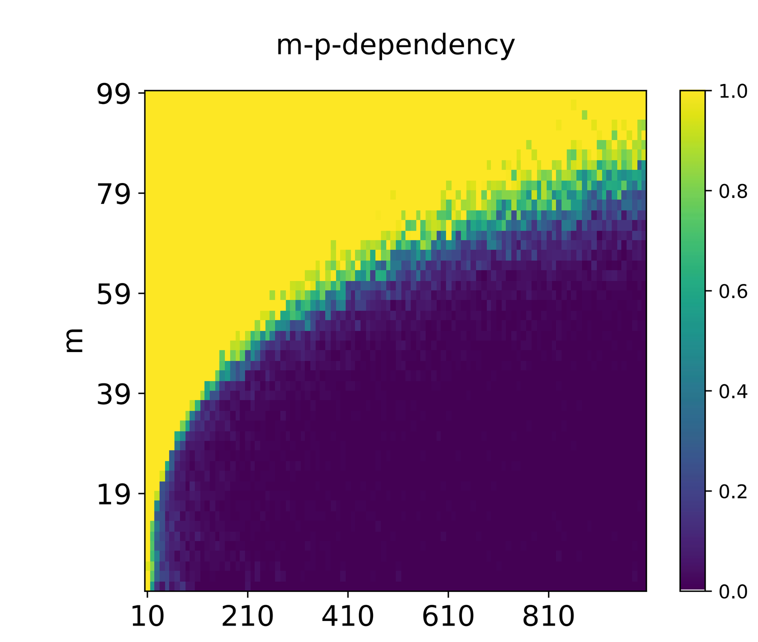
$$\mathbb{E}_{B_t,\Phi_t}\left[\left\|\boldsymbol{\Theta}^t - \boldsymbol{\Theta}^\star\right\|_F^2\right] \le \left(\kappa_1 + \frac{1}{c}(\kappa_2-\kappa_1)\right)^t \left\|\boldsymbol{\Theta}^0 - \boldsymbol{\Theta}^\star\right\|_F^2$$
$$+\left[\left(\kappa_1 + \frac{1}{c}(\kappa_2-\kappa_1)\right)^t - 1\right]\left(\frac{\sigma_1^2}{\kappa_1 - 1}\right) + \frac{\kappa_2 - 1}{c - c\kappa_1 + \kappa_1 - \kappa_2}\left(\frac{\sigma_2^2}{\kappa_2-1} - \frac{\sigma_1^2}{\kappa_1-1}\right).$$

Experiment Results

We designed the quadratic regression problem $y = \mathbf{x_i^T}\boldsymbol{\Theta}^\star\mathbf{x_i}$. We generate input feature $\mathbf{x_i}$ uniformly from [-1, 1] and uniformly random choose K elements in $\boldsymbol{\Theta}^\star$ as the ground truth interaction.

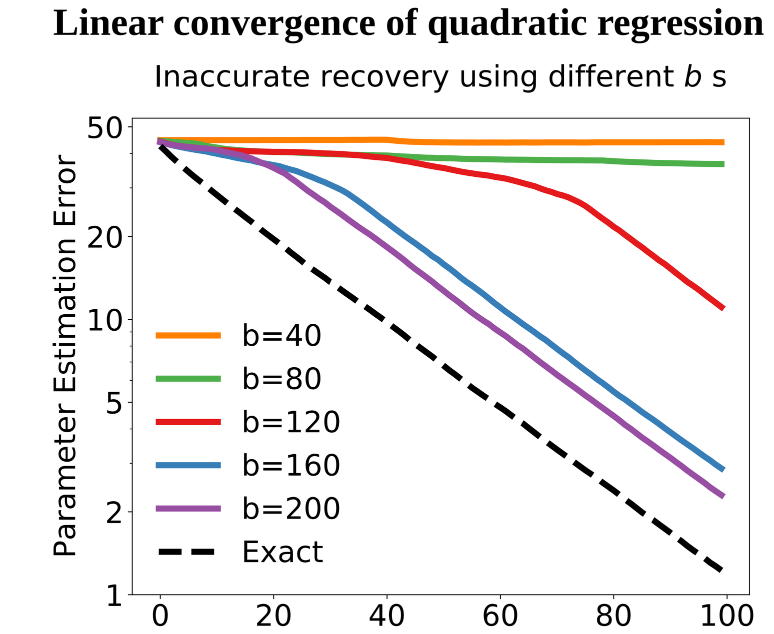
Note that the per-step complexity is $\widetilde{O}(m(p+b))$ and we claim that in order to achieve sub-quadratic complexity we can set $b = O(K)$ and $m = O(\log p)$. Here we first tested with different b-K and m-p combinations.



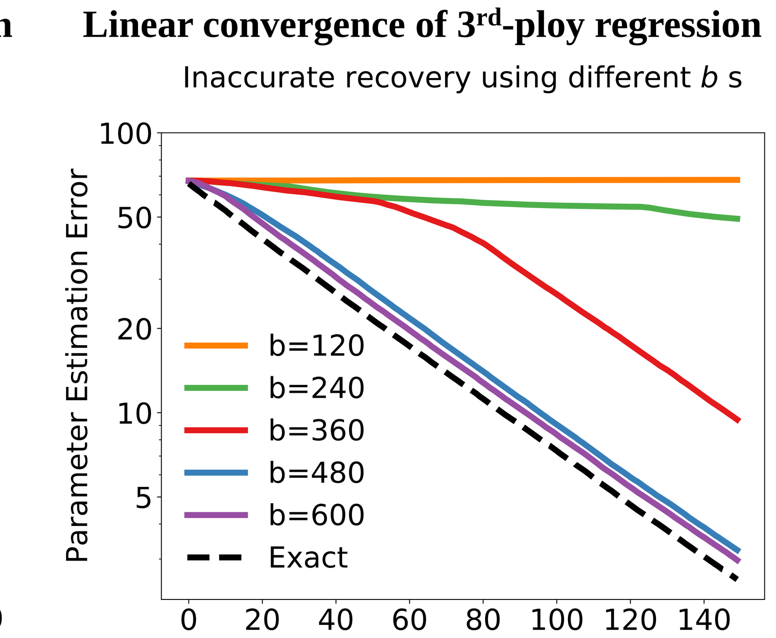


The two plots above shows the fraction of $\boldsymbol{\Theta}^\star$ sparse support recovery. Dark purple meaning no recovery and yellow meaning perfect recovery. As we claimed, setting $b = O(K)$ and p scales with $\log p$ is sufficient for parameter recovery.

Linear convergence of quadratic regression



Linear convergence of 3rd-ploy regression



The graph on the left is a quadratic regression problem, trying to recover a 30-sparse support in of a 200x200 $\boldsymbol{\Theta}^\star$ matrix. On the right hand side is an extension to higher order polynomial regression. The algorithm aims to recover a 30-sparse support of a 30x30x30 tensor, which models the 3-rd order interaction. The error is measured by L2 distance to $\boldsymbol{\Theta}^\star$.

[1] Pagh, R. (2013). Compressed matrix multiplication. ACM Transactions on Computation Theory (TOCT), 5(3), 9.