Fast Iterative Hard Thresholding Methods with Pruning Gradient Computations

Y. Ida[1], S. Kanai[1], A. Kumagai[1], T. Iwata[2], and Y. Fujiwara[2]

[1]NTT Computer and Data Science Laboratories,

[2]NTT Communication Science Laboratories

Sparse Linear Regression

NTT

- We consider the problem of identifying the top-k important parameters in a linear regression model using the least squares method.
- This is a crucial problem that crosses feature selection, sparse coding, and compressed sensing.

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^n} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\theta}\|_0 \leq k.$$

$\boldsymbol{y}$: continuous responses
$\boldsymbol{X}$: input matrix
$\boldsymbol{\theta}$: parameter vector
$\|\cdot\|_0$: # of nonzero elements

We will let $f(\boldsymbol{\theta}) = \frac{1}{2}\|\boldsymbol{y} - X\boldsymbol{\theta}\|_2^2$ for simplicity.

- **Iterative Hard Thresholding (IHT)** is a practical method to tackle this problem.

Iterative Hard Thresholding (IHT)

- IHT repeats the following steps util convergence:

Step 1: Update all parameters using gradient descent.

Step 2: Select the top-k parameters with the largest absolute values, and set the remaining parameters to zero (hard thresholding operator).

Algorithm 1 Iterative Hard Thresholding

Input: sparsity level k, step size η
Initialization: $\boldsymbol{\theta}^1 \leftarrow \mathbf{0}$, $t \leftarrow 1$
repeat
  $\boldsymbol{z}^t \leftarrow \boldsymbol{\theta}^t - \eta \nabla f(\boldsymbol{\theta}^t)$;
  $\boldsymbol{\theta}^{t+1} \leftarrow H_k(\boldsymbol{z}^t)$; $t \leftarrow t+1$;
until a stopping criterion is met

Speeding up IHT using a pruning strategy

- Step 1 requires $\mathcal{O}(mn)$ or $\mathcal{O}(n^2)$ time per iteration and is dominant in the overall cost.
  - m and n are the numbers of rows and columns of the input matrix $\boldsymbol{X}$, respectively.
- We reduce the cost of Step 1 per iteration using a pruning strategy to speed up IHT.
  - Previous methods mainly reduce the number of iterations to speed up IHT, e.g. Nesterov acceleration.

Algorithm 1 Iterative Hard Thresholding

Input: sparsity level k, step size η
Initialization: $\boldsymbol{\theta}^1 \leftarrow \mathbf{0}$, $t \leftarrow 1$
repeat
  $\boldsymbol{z}^t \leftarrow \boldsymbol{\theta}^t - \eta \nabla f(\boldsymbol{\theta}^t)$;
  $\boldsymbol{\theta}^{t+1} \leftarrow H_k(\boldsymbol{z}^t)$; $t \leftarrow t+1$;
until a stopping criterion is met

$\mathcal{O}(mn)$ or $\mathcal{O}(n^2)$

Pruning unnecessary updates via upper bounds

- We efficiently update the top-k parameters in each iteration by pruning unnecessary parameters.
- The unnecessary parameters are identified by comparing the updated previous iteration's top-k parameters with the upper bounds of the absolute values of the remaining parameters.

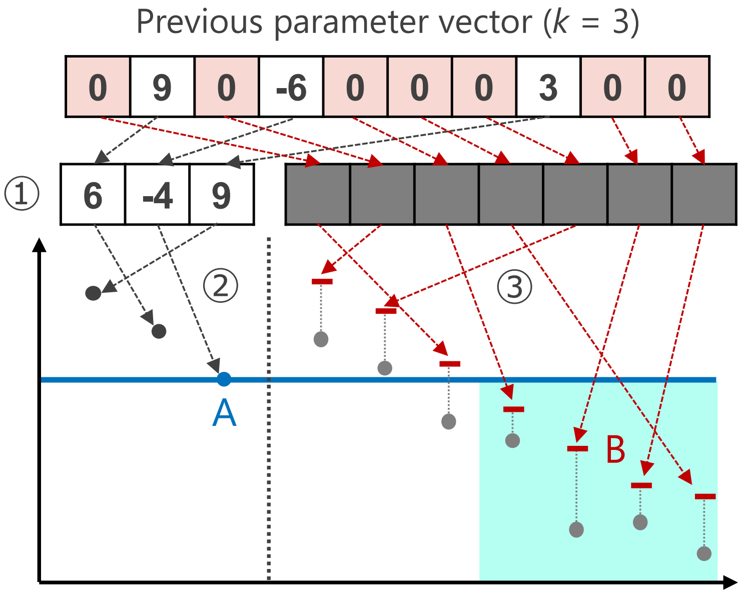
Subroutine of main idea

① Update the previous top-k parameters.

② Compute the smallest absolute value among the top-k parameters. (A)

③ Compute the upper bounds of the absolute values of the remaining parameters. (B)

If B ≦ A holds, the parameter can be safely pruned; otherwise, update the parameter precisely.



Property of upper bound computation

- The upper bounds for all parameters can be efficiently computed at $\mathcal{O}(n)$ time.
- Since the pruning is safe, the subroutine of the main idea yields the same result as $H_k(\cdot)$.
  - Our method achieves the same optimization result as the plain IHT.

Efficient computation of upper bound

The upper bound is computed based on the difference between the previous and current parameter vectors.

Definition 1 *Let t^* be $1 \leq t^* < t$ in Algorithm 1. Then, $\bar{z}_j^t$ at the t-th iteration in Algorithm 1 is computed as follows:*

$$\bar{\boldsymbol{z}}_j^t = |\boldsymbol{G}_j \boldsymbol{\theta}^{t^*} + \eta(\boldsymbol{X}^\top \boldsymbol{y})_j| + \|\boldsymbol{G}_j\|_2 \|\boldsymbol{\theta}^t - \boldsymbol{\theta}^{t^*}\|_2,$$

where $\boldsymbol{G} = \boldsymbol{I} - \eta \boldsymbol{X}^\top \boldsymbol{X}$.

Consistency of processing result

The subroutine of the main idea returns the same result of line 5 in IHT.

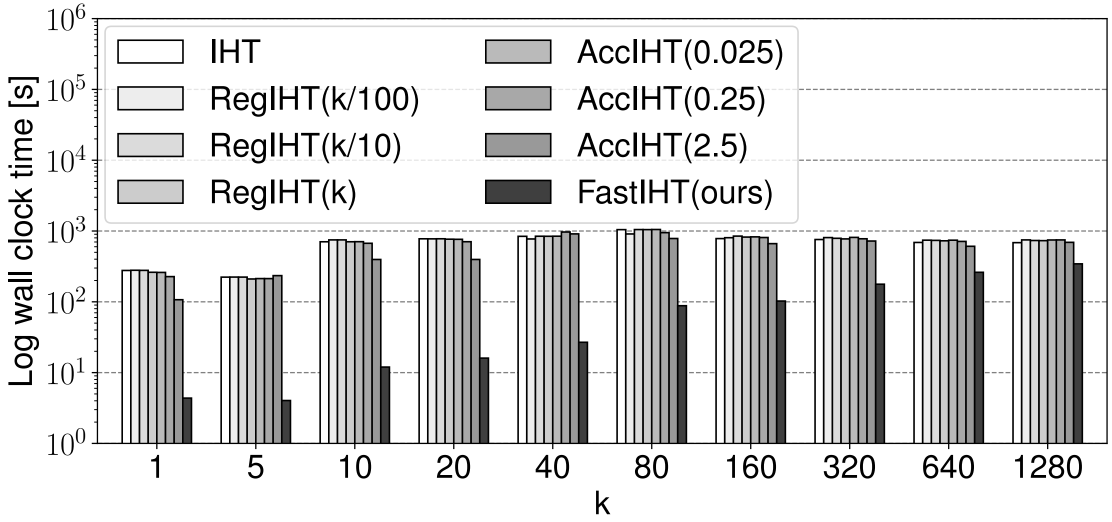
Algorithm 1 Iterative Hard Thresholding

```
1: Input: sparsity level k, step size η
2: Initialization: θ^1 ← 0, t ← 1
3: repeat
4:     z^t ← θ^t − η∇f(θ^t);
5:     θ^{t+1} ← H_k(z^t); t ← t + 1;
6: until a stopping criterion is met
```

Processing time

- Our method was up to 73 times faster than the plain IHT in our experiment.
- Our method achieved a large speedup factor for smaller *k* because it is based on the pruning.
- Our method does not need additional hyperparameter tuning while other baselines need it.

  Please see our paper for details.



Optimization result

- Our method achieved the same objective values and parameter vectors as those of the plain IHT.
- Our method ensures that the parameter vector of each iteration matches perfectly with that of the plain IHT.

  Please see our paper for details.

dataset	method	$k = 1$	$k = 20$	$k = 160$	$k = 1280$
gisette	IHT	56.01×10^{-2}	31.99×10^{-2}	14.01×10^{-2}	80.73×10^{-3}
	ours	56.01×10^{-2}	31.99×10^{-2}	14.01×10^{-2}	80.73×10^{-3}
robert	IHT	99.03×10^{-1}	91.23×10^{-1}	73.56×10^{-1}	66.24×10^{-1}
	ours	99.03×10^{-1}	91.23×10^{-1}	73.56×10^{-1}	66.24×10^{-1}
ledgar	IHT	12.76×10^{2}	82.48×10^{1}	50.70×10^{1}	35.35×10^{1}
	ours	12.76×10^{2}	82.48×10^{1}	50.70×10^{1}	35.35×10^{1}
real-sim	IHT	86.47×10^{-2}	63.84×10^{-2}	40.32×10^{-2}	23.16×10^{-2}
	ours	86.47×10^{-2}	63.84×10^{-2}	40.32×10^{-2}	23.16×10^{-2}
epsilon	IHT	93.50×10^{-2}	67.24×10^{-2}	44.93×10^{-2}	43.03×10^{-2}
	ours	93.50×10^{-2}	67.24×10^{-2}	44.93×10^{-2}	43.03×10^{-2}

Summary

- We accelerate IHT that finds the top-k important parameters in a linear regression model.

- Our method prunes unnecessary computations by using upper bounds.
- Our method guarantees the same optimization results as the plain IHT.
- Our method does not need additional hyperparameter tuning.

- Experiments demonstrate that our method is up to 73 times faster than the plain IHT without degrading accuracy.