Fast and Accurate Pseudoinverse with Sparse Matrix Reordering and Incremental Approach

Numerical linear algebra course by Professor Oseledets
Final Project by Team 'Untitled'
Anton Antonov, Nicholas Babaev

2020

PROBLEM STATEMENT

FORMAL STATEMENT

The solution of overdetermined equation $AZ \approx Y$ is obtained by minimizing the least square error $\|AZ - Y\|_F^2$, which results in the closed form solution $Z = A^+Y$. Using SVD decomposition, A^+ can be approximated as $A^+ \approx V_{n\times r}\Sigma^+_{r\times r}U^T_{r\times m}$.

LESS FORMAL

The goal of our project was to test a recently published [1] algorithm for finding the pseudoinverse matrix using SVD decomposition.

[1] FastPI: Jinhong Jung, Lee Sael

WHY

Pseudoinverse matrix is widely used mathematical object, especially in areas such as optimization, data science, etc.
It is still quite difficult to compute the pseudoinverse matrix by standard methods, but in some situations it is possible to optimize the computation, as result we can get a significant increase in computational speed.

Hypothesis

FastPI execution time is lower than others methods

The accuracy of FastPI is approximately equal to others methods

Quality measurement

Reconstruction error

$$\|\mathbf{A} - \mathbf{U}_{m\times r}\mathbf{\Sigma}_{r\times r}\mathbf{V}_{r\times n}^{\top}\|_{\mathrm{F}}$$

Computational performance

$$\mathbf{T}$$

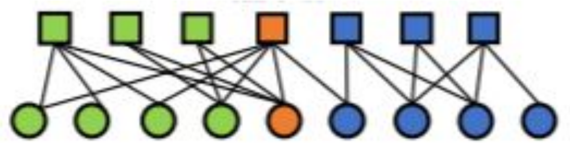
ALGORITHM

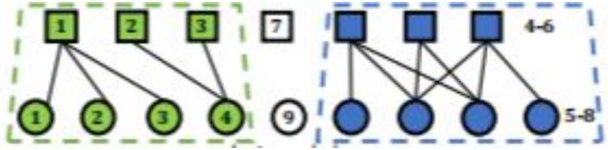
Step 1. Reordering

- Spokes (green)

- Hubs (orange)

- Giant Connected Component (blue)

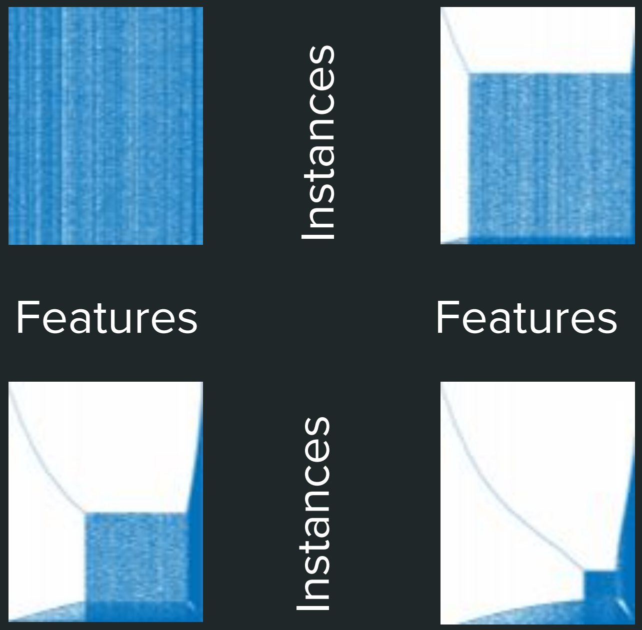




Step 1. Reordering

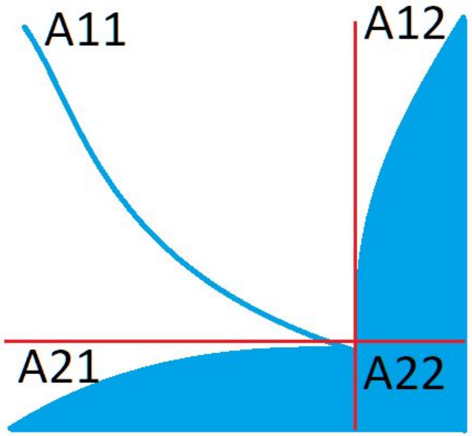
Algorithm

1. Find hubs
2. Put hubs into end of axis
3. Find spokes
4. Put spokes into beginning of axis
5. Repeat for GCC



Step 2. SVD of diagonal blocks

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$



$$A_{11} \qquad \mathbf{U}_{m_1 \times s} \mathbf{\Sigma}_{s \times s} \mathbf{V}^{\top}_{s \times n_1} = \mathrm{bdiag}(\mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(B)}) \times \mathrm{bdiag}(\mathbf{\Sigma}^{(1)}, \cdots, \mathbf{\Sigma}^{(B)}) \times \mathrm{bdiag}(\mathbf{V}^{(1)\top}, \cdots, \mathbf{V}^{(B)\top})$$

Step 3

$$
\begin{aligned}
\begin{bmatrix} \mathbf{A}_{11} \\ \mathbf{A}_{21} \end{bmatrix} &\simeq \begin{bmatrix} \mathbf{U}_{m_1 \times s} \mathbf{\Sigma}_{s \times s} \mathbf{V}_{s \times n_1}^{\top} \\ \mathbf{A}_{21} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{U}_{m_1 \times s} \mathbf{O}_{m_1 \times m_2} \\ \mathbf{O}_{m_2 \times s} \mathbf{I}_{m_2 \times m_2} \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}_{s \times s} \mathbf{V}_{s \times n_1}^{\top} \\ \mathbf{A}_{21} \end{bmatrix} \\
&\simeq \begin{bmatrix} \mathbf{U}_{m_1 \times s} \mathbf{O}_{m_1 \times m_2} \\ \mathbf{O}_{m_2 \times s} \mathbf{I}_{m_2 \times m_2} \end{bmatrix} \underbrace{\tilde{\mathbf{U}}_{(s+m_2) \times s} \tilde{\mathbf{\Sigma}}_{s \times s} \tilde{\mathbf{V}}_{s \times n_1}^{\top}}_{\text{Low-rank approximation with } s} \\
&= \mathbf{U}_{m \times s} \mathbf{\Sigma}_{s \times s} \mathbf{V}_{s \times n_1}^{\top}
\end{aligned}
$$

Step 4

$$
\begin{aligned}
\begin{bmatrix} \mathbf{A}_{11}\mathbf{A}_{12} \\ \mathbf{A}_{21}\mathbf{A}_{22} \end{bmatrix} &\simeq \left[\mathbf{U}_{m\times s}\boldsymbol{\Sigma}_{s\times s}\mathbf{V}_{s\times n_1}^{\top}\mathbf{T}\right] \\
&= \left[\mathbf{U}_{m\times s}\boldsymbol{\Sigma}_{s\times s}\ \mathbf{T}\right] \begin{bmatrix} \mathbf{V}_{s\times n_1}^{\top} & \mathbf{O}_{s\times n_2} \\ \mathbf{O}_{n_2\times n_1} & \mathbf{I}_{n_2\times n_2} \end{bmatrix} \\
&= \underbrace{\tilde{\mathbf{U}}_{m\times r}\tilde{\boldsymbol{\Sigma}}_{r\times r}\tilde{\mathbf{V}}_{r\times(s+n_2)}^{\top}}_{\text{Low-rank approximation with } r} \begin{bmatrix} \mathbf{V}_{s\times n_1}^{\top} & \mathbf{O}_{s\times n_2} \\ \mathbf{O}_{n_2\times n_1} & \mathbf{I}_{n_2\times n_2} \end{bmatrix} \\
&= \underline{\mathbf{U}_{m\times r}\boldsymbol{\Sigma}_{r\times r}\mathbf{V}_{r\times n}^{\top}}
\end{aligned}
$$

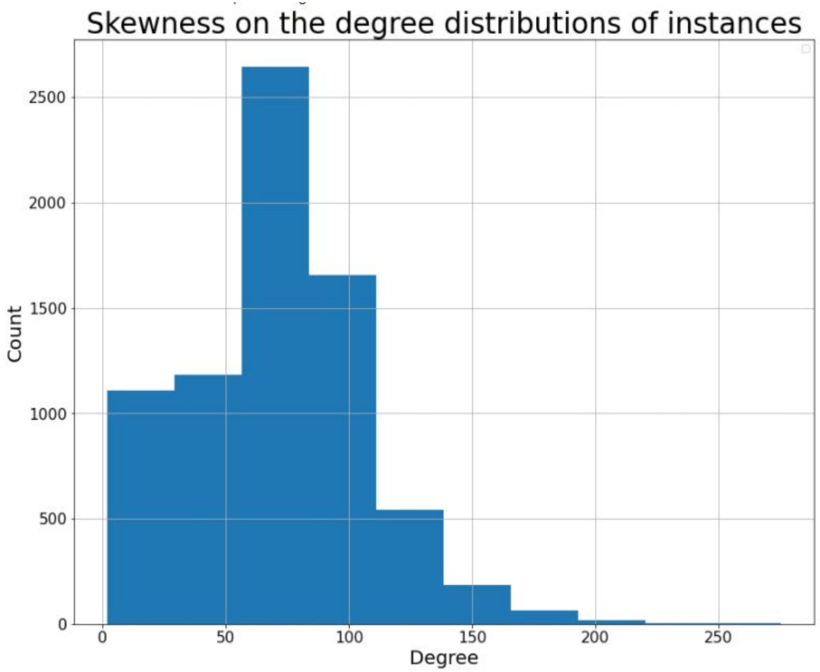
Complexity

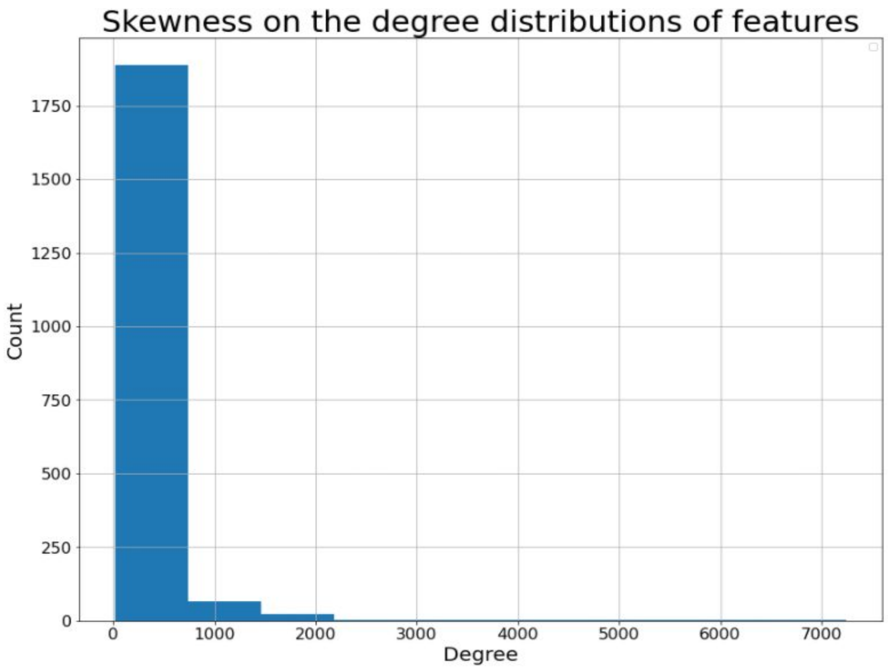
1. Reordering (t times):
   1.1 Sorting nodes and features: $O(m \log m)$
   1.2 Searching for spokes,hubs, GCC: $O(|A|)$
2. SVD of diagonal blocks: $\sum m_{1i} n_{1i} s_i$
3. SVD in step 3: $O((m_2 + s) n_1 s)$
4. Multiplication in step 3: $O(m_1 r^2 + n_1 r^2 + m_2 n_1 r)$
5. SVD in step 4: $O(m(n_2 + s) r)$
6. Multiplication in step 4: $O(n_1 r^2 + mr^2 + mn_2 r)$

Total:
$$O(m_1 r^2 + n_1 r^2 + m_2 n_1 r + n_1 r^2 + mr^2 + mn_2 r + m(n_2 + s) r + (m_2 + s) n_1 s + T(\sum m_{1i} n_{1i} s_i + |A|)) = \underline{O(mn \log(r) + (m+n) r^2)}$$

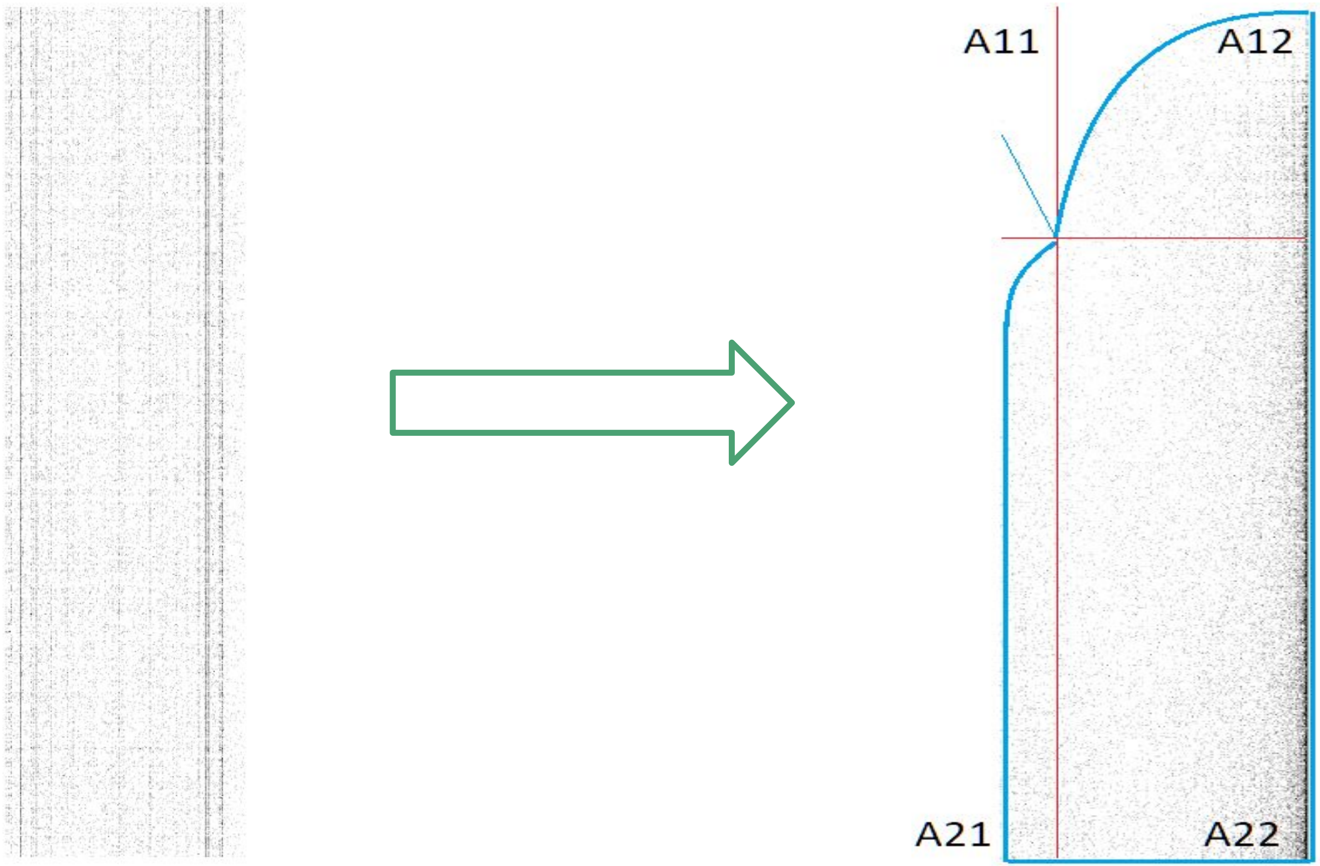
RESULTS

Dataset (Bibtex)

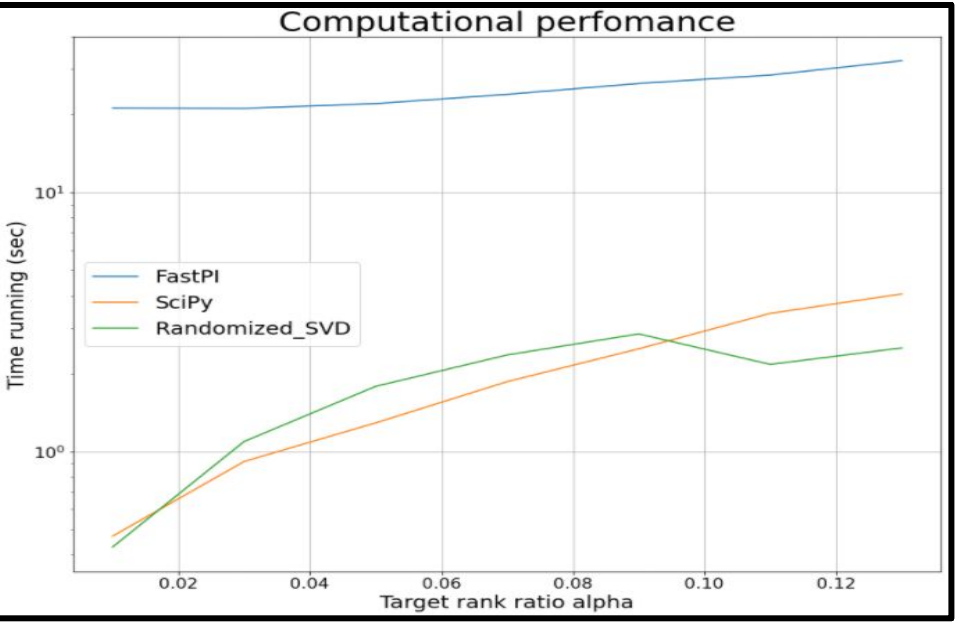




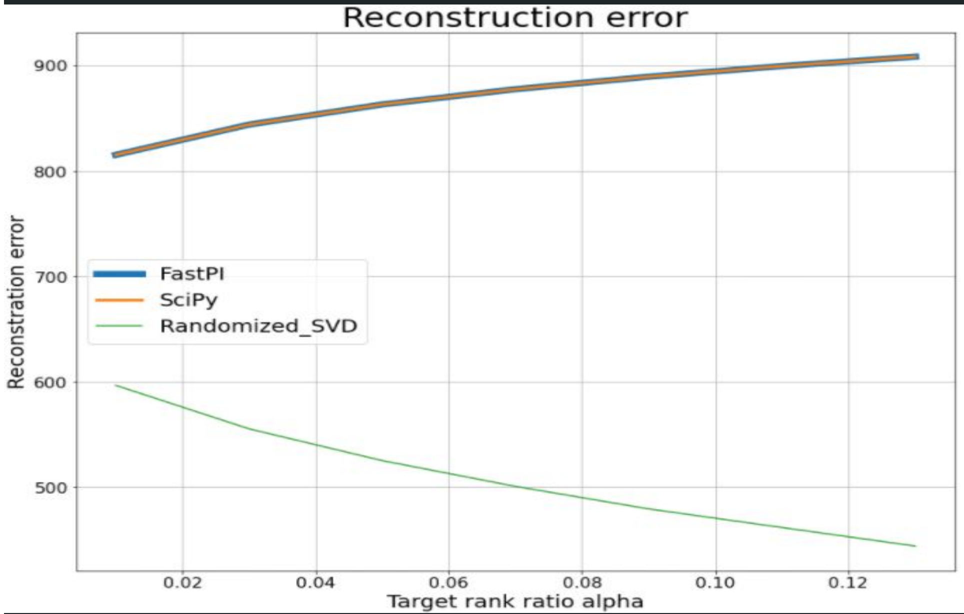
Matrices



RESULTS ANALYSIS



Execution time is higher than others methods



Reconstruction error is about SciPy implementation error

REASONS LED TO RESULT

Not fully optimized implementation

Inability to use large datasets

FUTURE PLANS

Code optimization

Multi-label regression metrics checking

OUR TEAM:

Anton Antonov

Nicholas Babaev

References

Jinhong J., Lee S. (2020) **Fast and Accurate Pseudoinverse with Sparse Matrix Reordering and Incremental Approach**, https://arxiv.org/pdf/2011.04235.pdf