# irlbpy – A fast partial SVD for Python

J. Baglama [1], M. Kane [2], B. Lewis [3], and L. Reichel [4]

[1]Department of Mathematics, The Unversity of Rhode Island
[2]Department of Biostatistics, Yale University
[3]Paradigm4
[4]Department of Mathematics, Kent State University

## Overview

The singular value decomposition (SVD) is central to many important analysis methods and applications including principal component analysis, canonical correlation analysis, correspondence analysis, latent semantic indexing and non-linear iterative partial least squares to name a few. However, numerical implementations of the SVD are computationally intensive, generally incurring a computational complexity of $O(m^2n + n^3)$ for an $m \times n$ matrix with $m$ greater than $n$. As a result, data scientist's have fewer analytical tools to understand the structure of data as those data become large and the resulting computational cost becomes too expensive to carry out.

However, many of these methods and applications only require a few singular values and corresponding singular vectors. With this in mind, some researchers have focused on computational efficient ***truncated*** SVD algorithm that calculates the largest or smallest singular value information for a matrix. The *implicitly restarted Lanczos bidiagonalization* (IRLB) algorithm [BR06] is a fast and efficient approach for calculating truncated singular values, generally scaling linearly in the size of the matrix. This innovative approach to calculating a key numerical decomposition for statistical and machine learning procedures allows many standard analyses to scale to much larger data sets than previously possible and suggests new approximation algorithms where current approaches fall short.

1. introduce the irbpy package
2. reference irlba R, and Matlab implementation
3. works with sparse and dense matrices
4. give the github address

## Mathematical Approach

## Algorithm Implementation

### References

[BR06] J. Baglama and L. Reichel. Restarted block lanczos bidiagonalization methods. *Numerical Algorithms*, 43:251–272, 2006.

## Dense Matrix Comparison

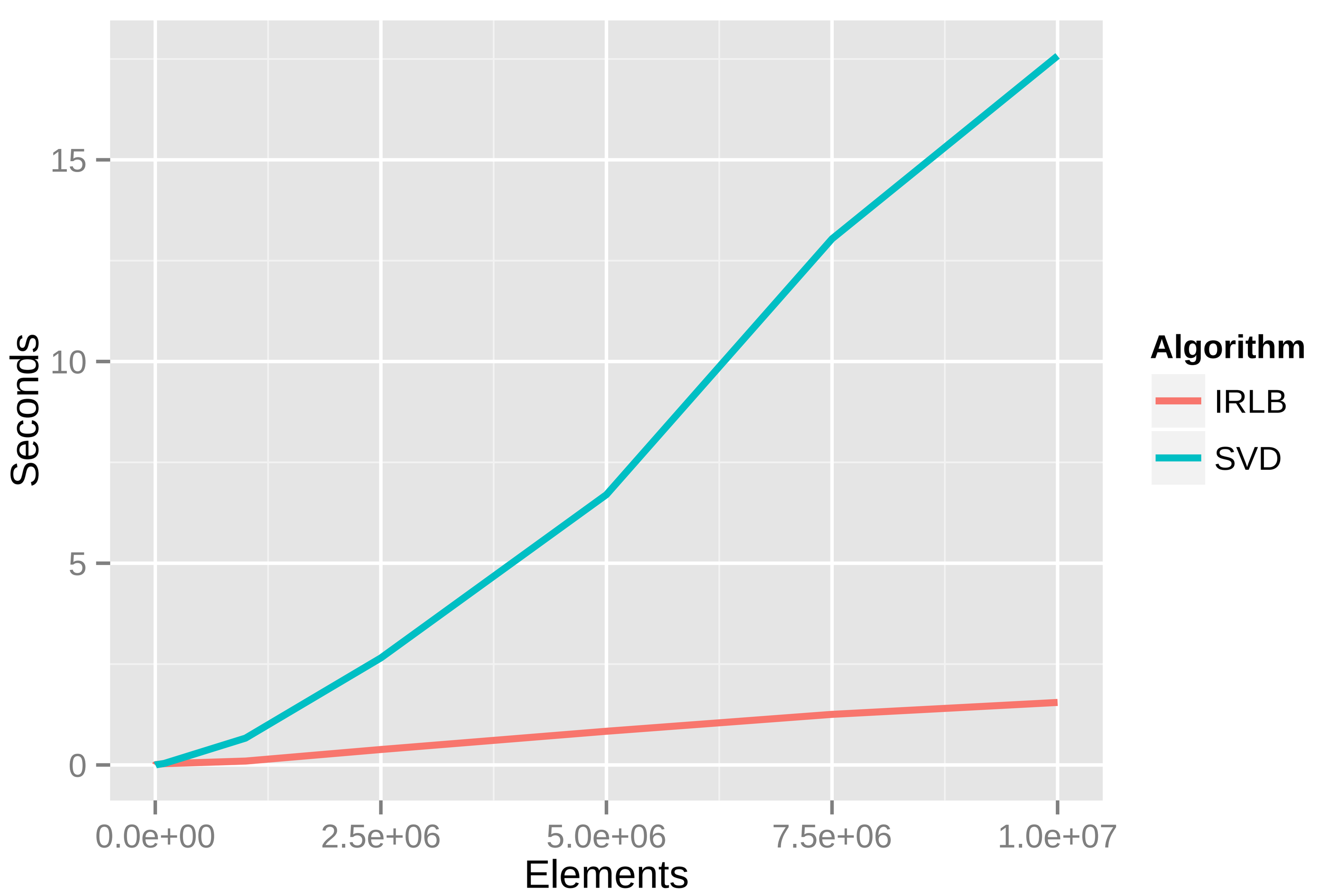


FIGURE 1: Perfomance comparison of the IRLB and the numpy implementation of the SVD.

## Dense Matrix Scaling

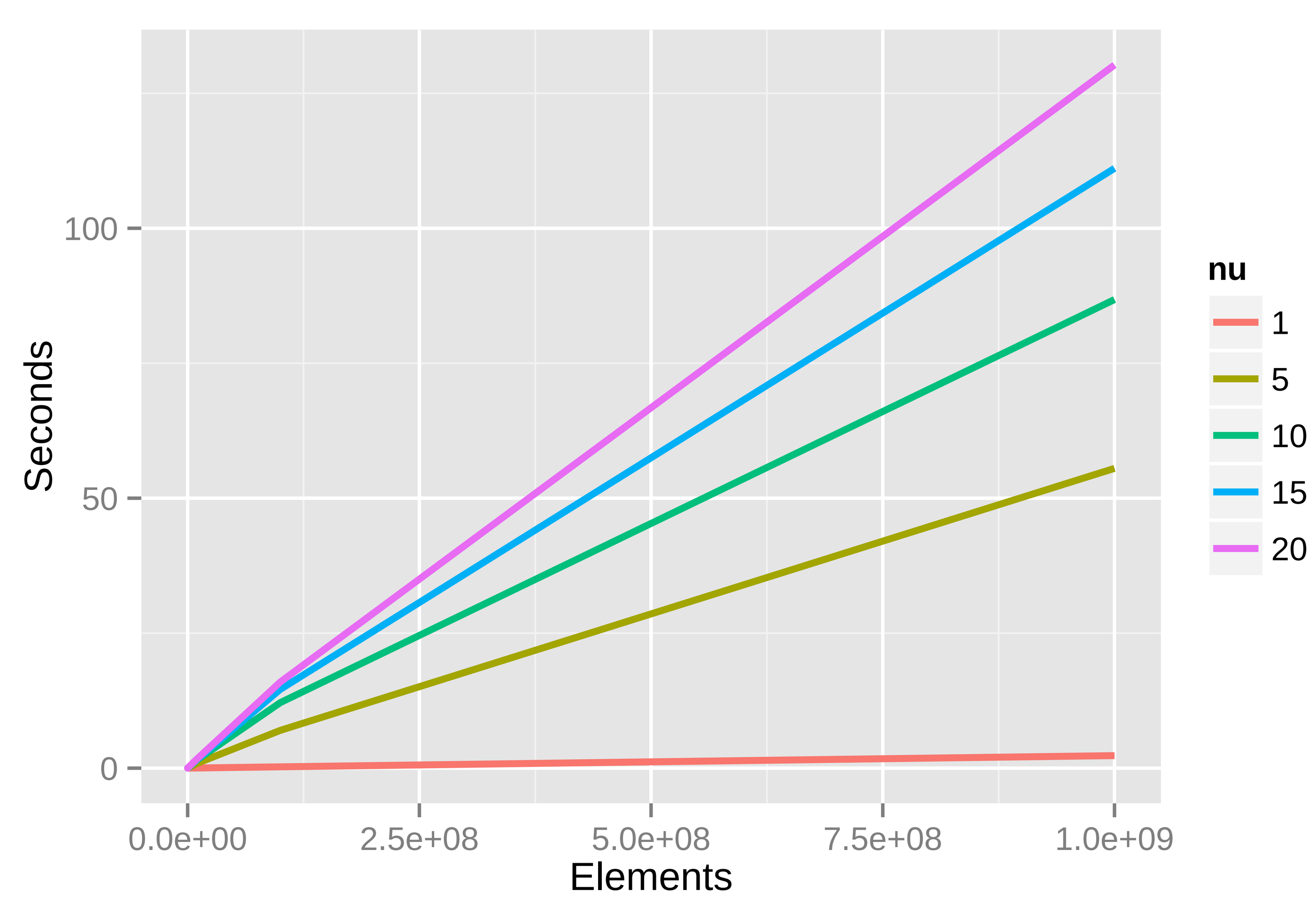


FIGURE 2: The time required to calculate the IRLB on dense matrices for specified values of nu (the number of singular vectors).

## Sparse Matrix Scaling

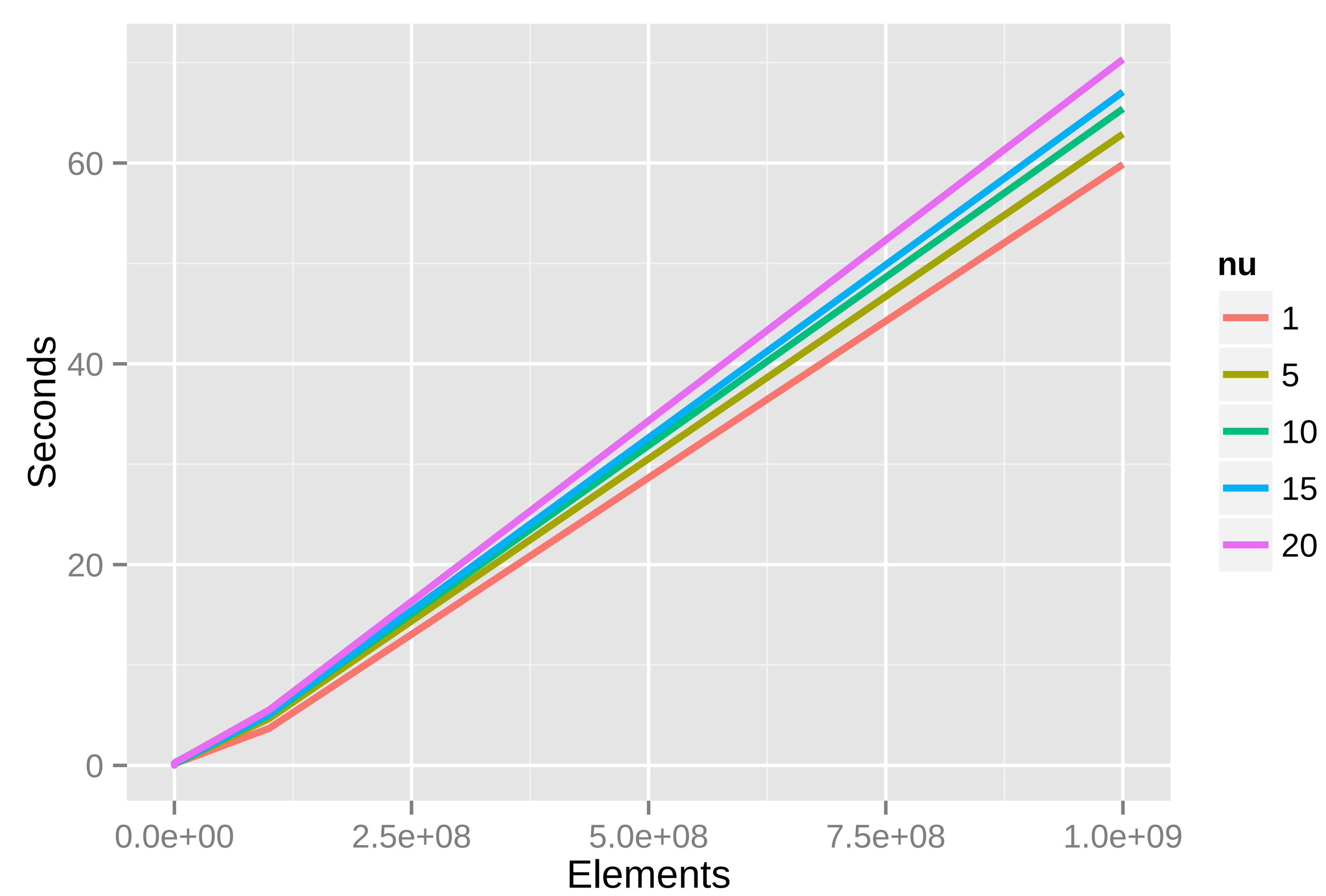


FIGURE 3: The time required to calculate the IRLB on sparse matrices for specified values of nu (the number of singular vectors).