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 [BR06a] 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.

While irlb implementations have existed for some time now in both Matlab [BR06b] and R [BR12] the scientific Python community has not enjoyed the computational savings offered by the algorithm until now. This poster introduces the irlbpy package for Python, a pip-installable open-source implementation of the IRLB algorithm that is available from github at https://github.com/bwlewis/irlbpy. The package leverages numpy for dense matrices as well as scipy for sparse matrices. The rest of this poster gives an overview of the algorithm and benchmarks performance. The actual benchmarks were performed on a Mac Book Pro with a quad-core 2.7 GHz Intel Core i7 with 16 GB of 1600 MHz DDR3 RAM running Python version 2.7.3, Numpy version 1.7.0, and SciPy version 0.12.0.

Dense Matrix Comparison

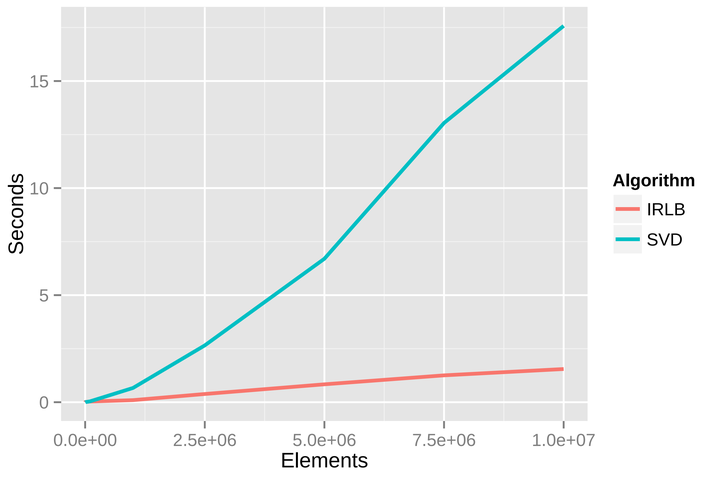

Figure 1: Perfomance comparison of the IRLB and the numpy implementation of the SVD.

Mathematical Approach

Dense Matrix Scaling

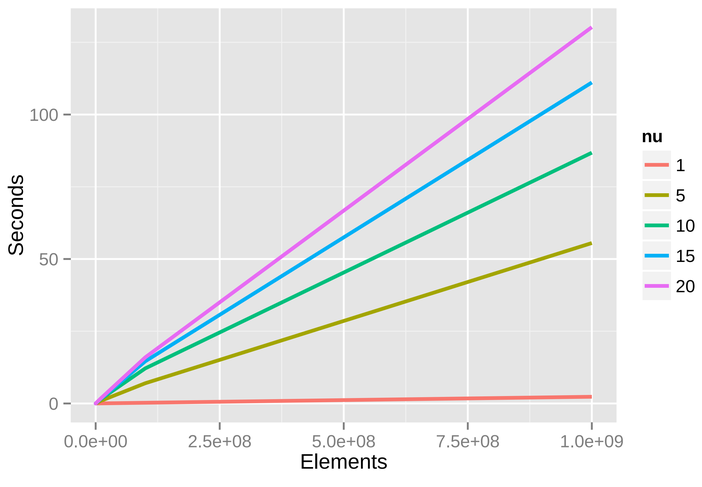

Figure 2: The time required to calculate the IRLB on dense matrices for specified values of nu (the number of singular vectors).

Algorithm Implementation

Sparse Matrix Scaling

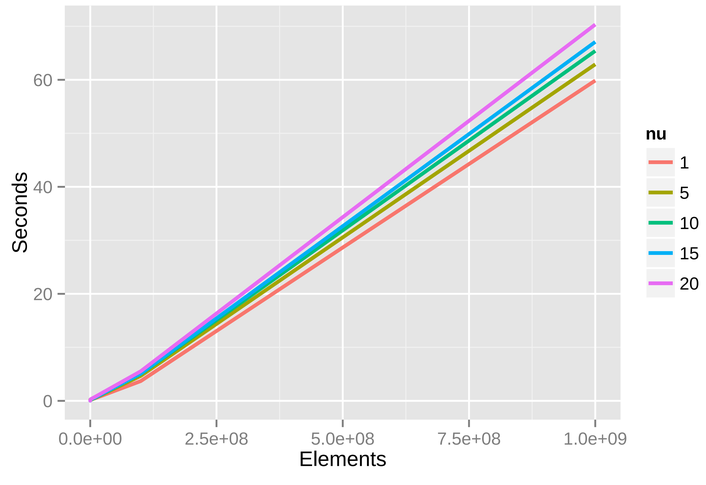

Figure 3: The time required to calculate the IRLB on sparse matrices for specified values of nu (the number of singular vectors).

References

[BR06a] J. Baglama and L. Reichel. Restarted block lanczos bidiagonalization methods. *Numerical Algorithms*, 43:251–272, 2006.

[BR06b] Jim Baglama and Lothar Reichel. *irlba*, 2006. Matlab package version 1.0.

[BR12] Jim Baglama and Lothar Reichel. *irlba: Fast partial SVD by implicitly-restarted Lanczos bidiagonalization*, 2012. R package version 1.0.2, Implemented by Bryan Lewis.