SVD and Beyond

Ilan Man
Statistical Science, Duke University

Objectives

- Discuss singular value decomposition (SVD) and its applications
- Introduce Randomized SVD
- Highlight the Johnson-Lindenstrauss Lemma
- Identify how power iterations make for much faster RSVD convergence
- Show empirical results

Introduction

Many statistical and machine learning applications require matrix manipulations of large datasets. Very commonly, we attempt to factorize a matrix A into smaller components and process those. Singular Value Decomposition is among the most popular approaches, for its wide range of applications. However, SVD is a computationally expensive algorithm so we seek to find faster, more efficient ways to factorize matrices. One relatively new approach is to use Randomized methods.

Matrix Factorization

We seek a factorization:

$$\underset{n\times m}{A} = \underset{n\times k}{R} \times \underset{k\times m}{Q^T} \tag{1}$$

- Common methods are **QR**, **Cholesky** and **SVD**
- Objective is to find $||A - RQ^T||_F \leq \varepsilon$, where $||A||_F = \sqrt{\sum_{ij} a_{ij}^2}$ is the Frobenius Norm

Applications: Low-rank Approximation

Rank: *maximal number of linearly independent rows (or columns).*
A rank k approximation is:

$$\min_B ||A - B||_F : \text{rank}(B) = k \tag{2}$$

Where $k \leq \text{rank}(A)$

Applications: Least Squares Solution

In order to solve $A\mathbf{x} = \mathbf{b}$ we seek a minimization of the form

$$||A\mathbf{x} - \mathbf{b}||_2 \tag{3}$$

The best minimizer is $\mathbf{x} = (A^TA)^{-1}A^T\mathbf{b}$.

SVD is the most numerically stable solution, especially when A is rank-deficient. However can be computationally expensive.

Singular Value Decomposition

The SVD of a real n x m matrix A:

$$\underset{n\times m}{A} = \underset{m\times k}{U} \times \underset{k\times k}{\Sigma} \times \underset{k\times m}{V}^T \tag{4}$$

We can also write it as:

$$A = \sum_{j=1}^{n} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{5}$$

Randomized SVD: Vanilla algorithm

Given: m by n matrix A and a desired rank k:

1. Draw an $n \times k$ Gaussian random matrix Ω with elements $\omega_{ij} \overset{iid}{\sim} N(0,1)$

```
Omega = np.random.randn(m, k)
```

2. Compute $H = A \times \Omega$

```
H = np.dot(A,Omega)
```

3. Construct $Q \in R^{m\times k}$ with columns forming an orthonormal basis for the range of H

```
Q, R = np.qr(H)
```

4. Form the $k \times n$ matrix, $B_k = Q^TA$.

```
B = np.dot(Q.T,A)
```

5. Return B_k

- Note that $||A - B_k||_F \leq \varepsilon$. Theoretically minimum error is σ_{k+1}.
- ε can be shown to be close to σ_{k+1} if oversampled.

Johnson-Lindenstrauss Lemma

The JL lemma allows us to project onto a low-dimensional subpace, keeping all structure unchanged, and do this in polynomial time.

$$(1-\varepsilon)||u-v||^2 \leq ||f(u) - f(v)||^2 \leq (1+\varepsilon)||u-v||^2$$

Randomized SVD: Comments

- $A \times \Omega$ is the bottleneck $\rightarrow$ easily **parallelizable**.
- **Fixed-rank** approach - assumes known target rank k. In practice, not always known in advance
- Number of columns, l, to minimize ε is $> k$
- Set $k = l + p$ where p is **oversampling** parameter, usually set at 5 or 10.
- Important to distinguish between in-sample error (fixed-precision, **numerical perspective**) vs out-of-sample (**statistical perspective**)

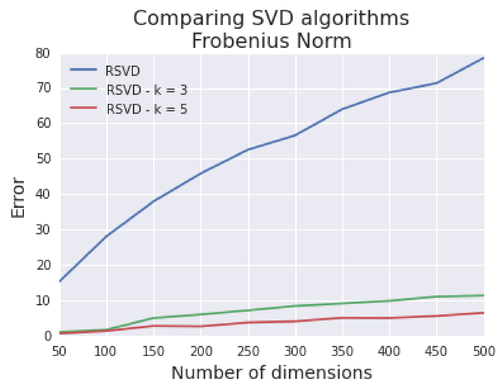
Power Scheme - RSVD improvement

Use power iterations to separate eigenvalues.

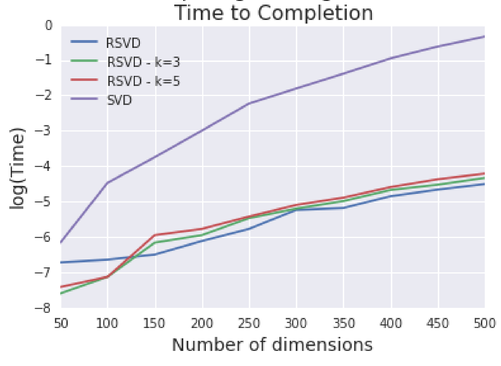
$$A^k = \sum_{i=1}^{n} {\sigma_i}^k \mathbf{v}_i \mathbf{v}_i^t$$

- If $\sigma_1 > \sigma_2$ then $\frac{1}{\sigma_1}A^k \rightarrow \mathbf{v}_1\mathbf{v}_1^T$. Can normalize by F-norm, and matrix converges to rank 1 matrix
- **Heuristic**: if $\sigma_i < 1, A_k \rightarrow 0$
- **Takeaway**: RSVD algorithm converges much faster

Empirical Results



For only a few power iterations, ε decreases dramatically. But there is little gain after 5 iterations.



RSVD computes a low rank approximation orders of magnitude faster than traditional SVD. Increasing the number of power iterations does not materially decrease computation time.

Further discussion

Next phase is to parallelize the algorithm by distributing the A^TA computation step. Allow us to compute SVD on matrices too large to fit in RAM.

Contact Information

- Web: http://www.ilanthedataman.com
- Email: ilanman@gmail.com
- Phone: +1 (917) 941 7323