SVD

Ben Southworth

31 May 2013

Abstract

In this paper we will first introduce the singular value decomposition (SVD) of a matrix and basic theory behind it. We will then go over the general algorithm, along with implementations in SAGE and computational efficiency comparisons with built in libraries. At the end we will discuss principal component analysis and its relation with the SVD as one of many applications.

1 Introduction to SVD

The singular value decomposition (SVD) of an $m \times n$ matrix A is a factorization

$$A = U\Sigma V^*,$$

where U is an $m \times m$ unitary matrix, Σ an $m \times n$ rectangular diagonal matrix, and V^* an $n \times n$ unitary matrix. V^* is the conjugate transpose of V, i.e. the (i, j)th entry of V is the complex conjugate of the (j, i)th entry of V^*. Note, a square matrix $A \in \mathbb{C}^{m\times m}$ is unitary if $A^* = A^{-1}$, and the singular values of A are the square roots of the eigenvalues of A^*A, all of which are real and non-negative.

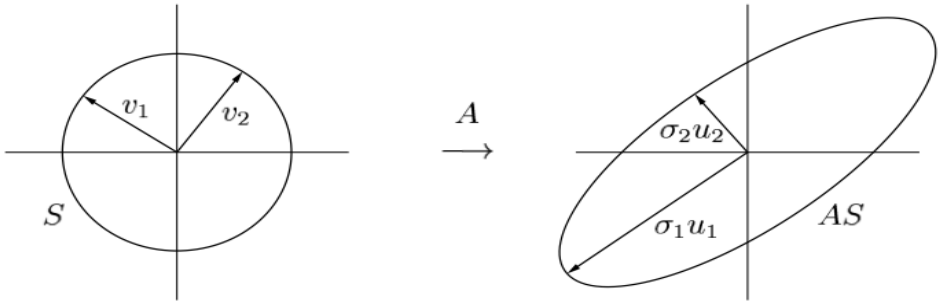

Figure 1: SVD of a 2×2 matrix [9].

Since V is unitary, $V^* = V^{-1}$, and thus we can rewrite the SVD as $AV = U\Sigma$, which can be broken down into individual components as $Av_i = \sigma_i u_i$. Using this format it is easiest to think of the SVD of a matrix A by breaking down its action on the unit sphere as shown in Figure 1. For the sake of simplicity, let A be some real matrix $A \in \mathbb{R}^{m\times n}$, where $m \geq n$. Then A maps the unit sphere $S \in \mathbb{R}^n$ to a hyper ellipse $AS \in \mathbb{R}^m$. The unit sphere is "stretched" by some factors $\sigma_1, ..., \sigma_m$ in some orthogonal directions $u_1, ..., u_m \in \mathbb{R}^m$. The vectors $\{\sigma_i u_i\}$ are called the

principal semi axes of the hyper ellipse, with lengths $\sigma_1, ..., \sigma_m$. If A has rank r, then exactly r of the lengths σ_i will be nonzero, and if $m \geq n$, at most n will be nonzero.

We can also connect this action on the unit sphere to the fundamental theorem of linear algebra and the four fundamental subspaces of a matrix A with rank r. Figure 2 illustrates the idea nicely. The subspace of $\mathbb{R}^m$ spanned by the columns of A is the range of A, also called the column space, and has dimension r. This is the space mapped to by A, and an orthonormal basis for the column space makes up the first r columns of U in our SVD. Similarly, the row space is the subspace of $\mathbb{R}^n$ spanned by the rows of A and is also of dimension r. These are the vectors that are mapped by A to the column space. The first r columns of V (or rows in V^*) are an orthonormal basis for the row space of A.

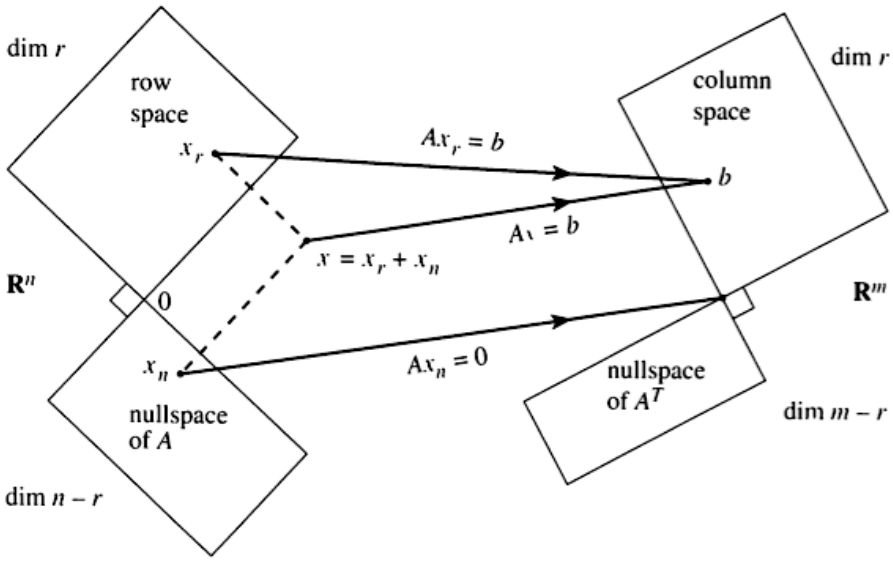

Figure 2: The 4 fundamental subspaces of a matrix [8].

By the fundamental theorem of linear algebra, there are two corresponding null spaces as well, the (right) null space and left null space with dimensions $m-r$ and $n-r$, respectively. The null space of A is the set of vectors mapped to zero by A, $N(A) = \{x \in \mathbb{R}^n | Ax = 0\}$, and an orthonormal basis for $N(A)$ makes up columns $\{r+1, ..., m\}$ of U. One important property of the null space is that it is orthogonal to the row space. Consider some $x \in N(A)$ and y in the row space of A. If y is in the row space of A, we can also write y as a linear combination of rows of A, where $y = \sum_{i=1}^{m} a_i b_i$ for some $b_i \in \mathbb{R}$ and row of A, $a_i \in \mathbb{R}^n$. Recall by definition of the null space, if $x \in N(A)$, $a_i x = 0$ for $i = 1, ..., m$. As a linear combination of rows of A, let y be a row vector, and let x naturally be a column vector as it is in the null space. Then if we consider the dot product of y and x we have

$$\begin{aligned} yx &= \Big(\sum_{i=1}^{m} a_i b_i\Big) x \\ &= \sum_{i=1}^{m} b_i (a_i x) \\ &= 0, \end{aligned}$$

thus proving their orthogonality. The left null space of A is the set of vectors y such that $yA = 0$, equivalently defined as the null space of A^*. A similar proof to that shown for the orthogonality of the null space and row space of A will confirm the orthogonality of the left null space and column space of A. An orthonormal basis for the left null space of A makes up columns of $\{r+1, ..., n\}$ of V. We call the column vectors of U and V the left singular vectors and right singular vectors, respectively. Figure 3 puts all of this together to show the form of each matrix in our decomposition, with the orthonormal bases for the row space, column space, null space and left null space, along with r nonzero singular values σ_i along the diagonal of Σ.

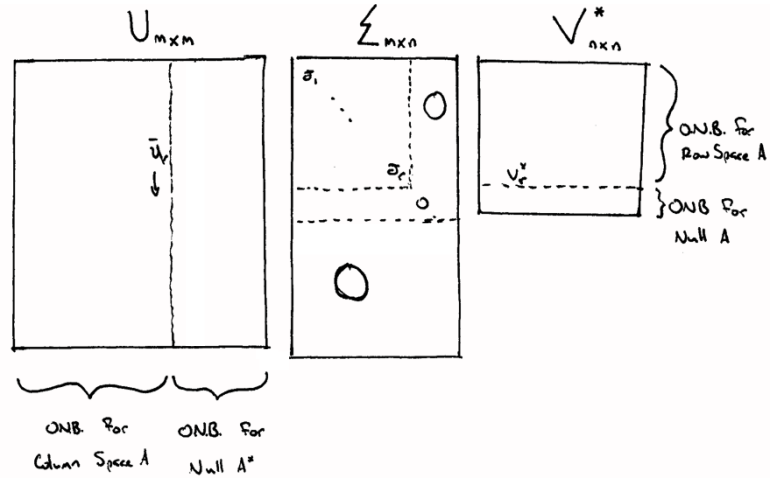

Figure 3: The 4 fundamental subspaces of a matrix.

We should note that the orthogonality of the null space and left null space with the row space and column space, respectively, is critical to choosing the columns of U, V in our SVD. This orthogonality allows a set consisting of an orthonormal basis for the column space (row space) and an orthonormal basis for the left null space (null space) to form an orthonormal basis for $\mathbb{R}^m$ ($\mathbb{R}^n$). A unitary matrix has several equivalent necessary conditions, and one is that an $m \times m$ matrix is unitary if and only if its rows and columns each form an orthonormal basis for $\mathbb{R}^m$. The orthogonality previously discussed allows us to choose the columns of U, V as we did. A nice proof on the existence and uniqueness of the SVD for *any* matrix A can be seen in [9].

2 Computing the SVD

It is possible to form the computation of left and right singular vectors and singular values of A as an eigenvalue problem, using the product A^*A. This process however can cause a significant loss in accuracy of smaller singular values. An alternative method that does not experience this problem is to use the cyclic matrix $\begin{pmatrix} 0 & A \\ A^* & 0 \end{pmatrix}$, however this accuracy is augmented with a corresponding increase in computational cost [5].

The best general method to computing the SVD can be broken down into two phases. The first is to bidiagonalize the matrix A into an upper bidiagonal matrix,

$A = PBQ^*$, where P, Q are unitary matrices and B an upper bidiagonal matrix. The second step is to then compute the SVD of $B = X\Sigma Y^*$, Plugging this into our formula for A gives us

$$\begin{aligned} A &= PX\Sigma Y^* Q^* \\ &= (PX)\Sigma(QY)^* \end{aligned}$$

Because the product of two unitary matrices with compatible dimensions is also unitary, if we let $U = PX$ and $V = QY$, we then have the SVD of A in the form $A = U\Sigma V^*$.

2.1 Phase 1: Bidiagonalization

Two standard methods to bidiagonalize a matrix are through Householder transformations or Lanczos recurrences. The use of Householder transformations to bidiagonalize a matrix is an extension of the standard QR factorization algorithm using Householder transformations. In the QR factorization, we decompose A into $A = QR$, for some orthogonal matrix Q and upper triangular matrix R. The Householder method involves applying a set of unitary matrices Q_k to the left of A to form an upper triangular matrix R,

$$Q_n ... Q_1 A = R$$

where $Q_n ... Q_1 = Q^*$, and thus $A = QR$. We choose each matrix Q_k such that when applied to the left of A, zeros are introduced under the diagonal in the kth column. Figure 4 illustrates this process on a 5×3 matrix.

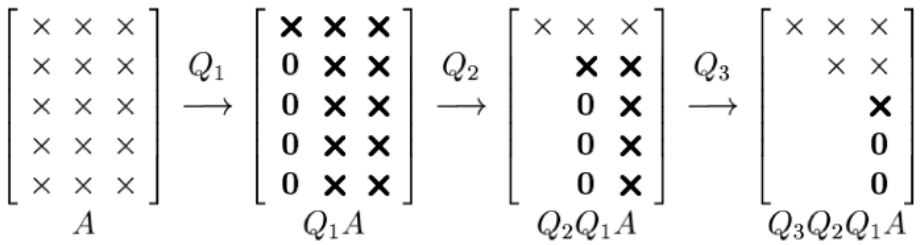

Figure 4: Householder bidiagonalization of a 5×3 matrix. [9]

We construct these unitary matrices Q_k through Householder reflectors F

$$Q_k = \begin{pmatrix} I & 0 \\ 0 & F \end{pmatrix},$$

where I is the $(k-1) \times (k-1)$ identity matrix, and F an $(m-k+1) \times (m-k+1)$ unitary matrix. Let $x \in \mathbb{C}^{m-k+1}$ be the entries below and including the diagonal of the kth column of A. We then need

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_{m-k+1} \end{pmatrix} \quad \xrightarrow{F} \quad Fx = \begin{pmatrix} \|x\| \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

which can also be written as $Fx = \|x\|e_1$, where e_1 is the $(m-k+1)$-dimensional vector $e_1 = (1, 0, ..., 0)^T$. The appropriately named reflector F will do this by reflecting the space $\mathbb{C}^{m-k+1}$ across the hyperplane orthogonal to $v = \|x\|e_1 - x$. Trefethen defines the reflector matrix as

$$F = I - 2\frac{vv^*}{v^*v},$$

which we will then use to apply n Householder reflections to the left of A [9].

The extension of householder transformations to bidiagonalization is called Golub-Kahan bidiagonalization [4]. Instead of applying Householder reflectors to just the left side of A, we alternate, applying reflectors to the left and right side of A,

$$A \quad \longrightarrow \quad U_n^*...U_1^*AV_1...V_{n-2}$$

Reflectors applied to the left introduce column zeros beneath the diagonal, and reflectors applied to the right introduce row zeros, starting at the two entries past the diagonal, as shown in Figure 5 At the end of the algorithm, we have applied n reflectors to the left and $n-2$ reflectors to the right [9], leaving A as the product of an upper bidiagonal matrix with a set of unitary Householder transformation matrices.

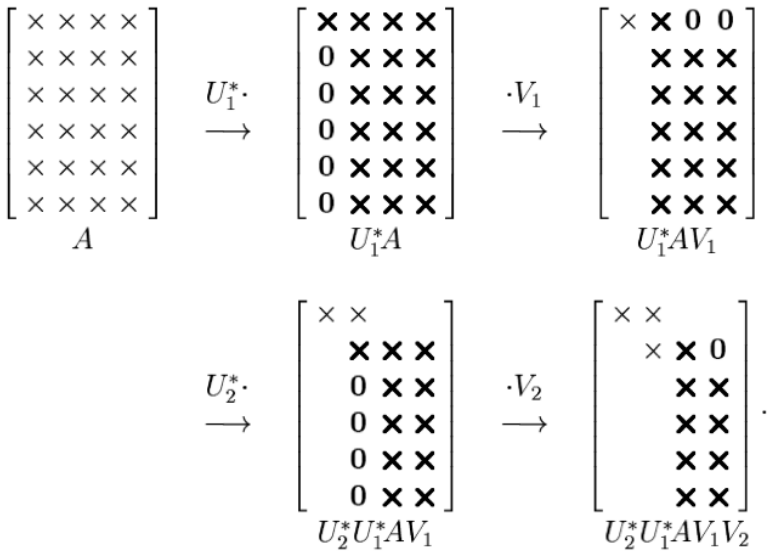

Figure 5: Householder bidiagonalization of a 5×3 matrix. [9]

Householder.sage is a SAGE script to upper-triangularize a matrix using the QR algorithm. Implementing the two sided householder transformations proved to be significantly trickier problem, and so we instead switched our bidiagonalization method to Lanczos recurrences, discussed below. The bidiagonalization script for Lanczos recurrences can be found in *Lanczos.sage*.

An alternative bidiagonalization method without using Householder transformations is using Lanczos recurrences. Suppose we have the equation $A = PBQ^*$, where P, Q are unitary matrices and B an upper bidiagonal matrix. We can rewrite

this as

$$B = P^*AQ = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ & \alpha_2 & \beta_2 & & \\ & & \ddots & \ddots & \\ & & & \alpha_n & \beta_n \end{pmatrix},$$

where

$$\alpha_j = p_j^* A q_j$$
$$\beta_j = p_j^* A q_{j+1}.$$

As discussed in [5], if we choose an arbitrary unit vector q_1, the rest of the column vectors making up P, Q are uniquely determined. Given matrix equations $AQ = PB$ and $A^*P = QB^*$, we can equate the first k columns to yield a double recursion algorithm as follows:

$$\alpha_j p_j = Aq_j - B_{j-1}p_{j-1}$$
$$B_j q_{j+1} = A^* p_j - \alpha_j q_j,$$

where $\alpha_j = \|Aq_j - B_{j-1}p_{j-1}\|$ and $\beta_j = \|A^*p_j - \alpha_j q_j\|$ because the columns of P, Q are normalized [5]. An implementation of this recurrence can be found in *Lanczos.sage*.

2.2 Phase 2: Diagonalization

The second phase of computing the SVD of a matrix A is to compute the SVD of the bidiagonal matrix B computed in phase 1. This is done by diagonalizing B as the product of unitary matrices, which, as discussed previously, will then give us the SVD of A. In computation, the second phase is not an analytic algorithm however, instead iterating to some level of error convergence.

The generic algorithm to diagonalize B is applying an implicit QR algorithm to B^TB. We compute a sequence of matrices B_i, starting at $B_0 = B$, with some shift σ^2, generally taken to be the smallest eigenvalue of the 2×2 matrix in the bottom right of $B_iB_i^T$. The implicit QR factorization is then done on the shifted matrix $B)_i^T B_i - \sigma^2 I = QR$. We then let $B_{i+1}^T B_{i+1} = RQ + \sigma^2 I$, which will converge to a diagonal matrix for large i [2].

It is worth noting there are a number of different, more specific methods for the second phase. The one presented here works well for well-conditioned matrices, but can fail on ill-conditioned matrices. There is an implicit zero-shift QR algorithm, which removes the act of shifting our sequence of matrices and performs much better on ill-conditioned matrices, albeit at a high computational cost [3]. There are also divide and conquer algorithms to diagonalize B that are relatively efficient and robust [1].

3 Principal Component Analysis

Many uses of the SVD can be seen in Carla Martin and Mason Porter's "The extraordinary SVD" [6]. One of the most prominent applications is the close relation of the SVD to principal component analysis (PCA). PCA is perhaps best described

by Jonathon Schlens as "a simple non parametric method of extracting relevant information from confusing data sets" [7]. Given some set of n samples of m measurements, $X \in \mathbb{R}^{m\times n}$, the goal of PCA is to find a new basis as a linear combination of the original basis of X that best re-expresses the data. In performing a change of basis, we hope to both remove noise and extract an underlying structure to the data. We can write this change of basis as a matrix equation

$$PX = Y,$$

where $X \in \mathbb{R}^{m\times n}$ is our data, $Y \in \mathbb{R}^{m\times n}$ our re-expressed data, and $P \in \mathbb{R}^{m\times m}$ a change of basis matrix. Let us now index the rows of P as p_i, for $i = 1, ..., m$ and the columns of X as x_i for $i = 1, ..., n$, and rewrite the equation as a set of vector dot products

$$\begin{aligned} PX &= \begin{pmatrix} p1 \\ \vdots \\ p_m \end{pmatrix} \begin{pmatrix} x_1 & ... & x_n \end{pmatrix} \\ &= \begin{pmatrix} p1 \cdot x_1 & ... & p1 \cdot x_n \\ \vdots & \ddots & \vdots \\ p_m \cdot x_1 & ... & p \cdot x_n \end{pmatrix} \\ &= Y \end{aligned}$$

We can then see that the ith column of Y is given by $y_i = \begin{pmatrix} p_1 \cdot x_i, ..., p_m \cdot x_i \end{pmatrix}^T$, where the jth coefficient of y_i is a projection onto the jth row of P. Thus $PX = Y$ is a change of basis on the columns of X to the basis $\{p_1, ..., p_m\}$. We call these vectors $\{p_1, ..., p_m\}$ the principal components of X.

The next question is how do we define the *best* basis to re-express our data? We are trying to rid the data of both noise and redundancy. A scale of noise is the signal to noise ratio (SNR), given by

$$SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}},$$

where a low SNR implies noisy data and a high $SNR >> 1$ implies minimal noise. Thus a basis to reduce noise will be found by maximizing the variance of the signal, thereby minimizing the SNR. The other potential problem with data is redundancy, which refers to measurements that give unnecessary information. A simple example is videotaping movement along a 1-dimensional place using multiple cameras. Clearly one camera is sufficient, so data supplied by additional cameras add redundancy to our data. Redundancy is best measured by the covariance between vectors. A high covariance implies a strong linear relationship and thus redundancy, while a low covariance implies higher independence and low redundancy. In choosing a new basis, we thus want to minimize the covariance between measurements. If we assume zero mean for two measurements x_1, x_2, the formula for covariance between them reduces to a normalized dot product,

$$\sigma^2_{x_1,x_2} = \frac{1}{n-1} x_1 x_2^T.$$

Generalizing this to our matrix X of m measurements, we have a covariance matrix given by

$$C_X = \frac{1}{n-1} X X^T,$$

where the ith diagonal entry is the variance of the ith measurement, and the (i, j)th entry is the covariance between the ith and jth measurements. Two important things to note are the covariance of a vector with itself is just its variance, and C_X is a symmetric $m \times m$ matrix, where $C_X(i, j) = C_X(j, i)$. If you recall our goals, it is clear that we want C_X to be diagonal, removing redundancy altogether and then maximizing the SNR in the diagonal entries. Reframing this to our equation $PX = Y$, we want some orthonormal matrix P such that $C_Y = \frac{1}{n-1} YY^T$ is diagonal. Writing C_Y in terms of P gives us

$$\begin{aligned} C_Y &= \frac{1}{n-1} YY^T \\ &= \frac{1}{n-1} (PX)(PX)^T \\ &= P\Big(\frac{XX^T}{n-1}\Big)P^T \end{aligned}$$

If we now note that XX^T is a symmetric matrix, we can use the relevant linear algebra theorem that a symmetric matrix is diagonalized by an orthogonal matrix of its eigenvectors, $A = EDE^T$ [8]. This tells us that we want to choose P such that each row p_i is an eigenvector of $C_X = \frac{1}{n-1} XX^T$.

With this, we can finalize the relation of the SVD and PCA. Let us define a new matrix $Y = \frac{1}{n-1} X^T$, and expand $Y^T Y$ to see that $Y^T Y = C_X$. As previously discussed, the principal components of X are the eigenvectors of C_X, and we will now calculate these eigenvectors as the eigenvectors of $Y^T Y$. If we decompose $Y^T Y$ into its SVD, the columns of the matrix V are the eigenvectors for $Y^T Y$, and thus the columns of V are the principal components of X.

References

[1] Cline, Alan, and Dhillon, Inderjit. "Computation of the Singular Value Decomposition." Handbook of Linear Algebra. Chapman and Hall, 2007.

[2] Deift, Percy, et al. "The bidiagonal singular value decomposition and Hamiltonian mechanics." SIAM journal on numerical analysis 28.5 (1991): 1463-1516.

[3] Demmel, James, and William Kahan. "Accurate singular values of bidiagonal matrices." SIAM Journal on Scientific and Statistical Computing 11.5 (1990): 873-912.

[4] Golub, Gene, and William Kahan. "Calculating the singular values and pseudo-inverse of a matrix." Journal of the Society for Industrial & Applied Mathematics, Series B: Numerical Analysis 2.2 (1965): 205-224.

[5] Hernandez, V., et al. "Restarted Lanczos bidiagonalization for the SVD in SLEPc." (2007).

[6] Martin, Carla D., and Mason A. Porter. "The extraordinary SVD." The American Mathematical Monthly 119.10 (2012): 838-851.

[7] Shlens, Jonathon. "A tutorial on principal component analysis." Systems Neurobiology Laboratory, University of California at San Diego (2005).

[8] Strang, Gilbert. "The fundamental theorem of linear algebra." The American Mathematical Monthly 100.9 (1993): 848-855.

[9] Trefethen, Lloyd N., and David Bau III. Numerical linear algebra. No. 50. Society for Industrial and Applied Mathematics, 1997.