1 Singular Value Decomposition

The singular vector decomposition allows us to write any matrix $A$ as

$$A = USV^T,$$

where $U$ and $V$ are orthogonal matrices (square matrices whose columns form an orthonormal basis), and $S$ is a diagonal matrix (a matrix whose only non-zero entries lie along the diagonal):

$$S = \begin{bmatrix} s_1 & & \\ & s_2 & \\ & & \ddots \\ & & & s_n \end{bmatrix}$$

The columns of $U$ and $V$ are called the left singular vectors and right singular vectors, respectively. The diagonal entries $\{s_i\}$ are called singular values. The singular values are always $\geq 0$.

The SVD tells us that we can think of the action of $A$ upon any vector $\vec{x}$ in terms of three steps (Fig. 1):

1. rotation (multiplication by $V^T$, which doesn’t change vector length of $\vec{x}$).
2. stretching along the cardinal axes (where the $i^{th}$ component is stretched by $s_i$).
3. another rotation (multiplication by $U$).

$$\vec{y} = A\vec{x} = (USV^T)\vec{x}$$

Figure 1: Schematic illustration of SVD in terms of three linear transformations.
2 Inverses

The SVD makes it easy to compute (and understand) the inverse of a matrix. We exploit the fact that $U$ and $V$ are orthogonal, meaning their transposes are their inverses, i.e., $U^T U = U U^T = I$ and $V^T V = V V^T = I$.

The inverse of $A$ (if it exists) can be determined easily from the SVD, namely:

$$A^{-1} = V S^{-1} U^T,$$

where

$$S^{-1} = \begin{bmatrix} \frac{1}{s_1} & \frac{1}{s_2} & \cdots & \frac{1}{s_n} \end{bmatrix}$$

(2)

The logic is that we can find the inverse mapping by undoing each of the three operations we did when multiplying $A$: first, undo the last rotation by multiplying by $U^T$; second, un-stretch by multiplying by $1/s_i$ along each axis, third, un-rotate by multiplying by $V$. (See Fig 2).

Another way to see that this definition of the inverse is correct is via:

$$A^{-1} A = (V S^{-1} U^T)(U S V^T)$$

$$= V S^{-1} (U^T U) S V^T$$

$$= V (S^{-1} S) V^T$$

$$= V V^T$$

$$= I$$

We can do a similar analysis of $AA^{-1}$. 
3 Pseudo-inverse

The SVD also makes it easy to see when the inverse of a matrix doesn’t exist. Namely, if any of the singular values \( s_i = 0 \), then the \( S^{-1} \) doesn’t exist, because the corresponding diagonal entry would be \( 1/s_i = 1/0 \).

In other words, if a matrix \( A \) has any zero singular values (let’s say \( s_j = 0 \)), then multiplying by \( A \) effectively destroys information because it takes the component of the vector along the right singular vector \( \vec{v}_j \) and multiplies it by zero. We can’t recover this information, so there’s no way to “invert” the mapping \( A\vec{x} \) to recover the original \( \vec{x} \) that came in. The best we can do is to recover the components of \( \vec{x} \) that weren’t destroyed via multiplication with zero.

The matrix that recovers all recoverable information is called the pseudo-inverse, and is often denoted \( A^\dagger \). We can obtain the pseudoinverse from the SVD by inverting all singular values that are non-zero, and leaving all zero singular values at zero.

Suppose we have an \( n \times n \) matrix \( A \), which has only \( k \) non-zero singular values. Then the \( S \) matrix obtained from SVD will be

\[
S = \begin{bmatrix}
 s_1 \\
 \vdots \\
 s_k \\
 0 \\
 \vdots \\
 0
\end{bmatrix}.
\]

The pseudoinverse of \( A \) can then be written similarly to the inverse:

\[
A^\dagger = VS^\dagger U^\top,
\]

where

\[
S^\dagger = \begin{bmatrix}
 1/s_1 \\
 \vdots \\
 1/s_k \\
 0 \\
 \vdots \\
 0
\end{bmatrix}.
\]

4 Condition number

In practical situations, a matrix may have singular values that are not exactly equal to zero, but are so close to zero that it is not possible to accurately compute them. In such cases, the matrix is what we call ill-conditioned, because dividing by the singular values \( 1/s_i \) for singular values \( s_i \) that are arbitrarily close to zero will result in numerical errors. Such matrices are theoretically but not practically invertible. (If you try to invert such a matrix, you likely (hopefully) get a warning like: “Matrix is close to singular”).
The degree to which ill-conditioning prevents a matrix from being inverted accurately depends on the ratio of its largest to smallest singular value, a quantity known as the **condition number**:

\[
\text{condition number} = \frac{s_1}{s_n}.
\]

The larger the condition number, the more practically non-invertible it is. When using double floating point precision, matrices with condition numbers greater than \( \approx 10^{14} \) cannot be stably inverted.

You can compute the condition number yourself from the SVD, or using the built-in Matlab command `cond`, or the numpy command `numpy.linalg.cond`.

## 5 SVD of non-square matrix

If \( A_{m \times n} \) is a non-square matrix, then \( U \) is \( m \times m \) and \( V \) is \( n \times n \), and \( S_{m \times n} \) is non-square (and therefore has only \( \min(m, n) \) non-zero singular values. Such matrices are (obviously) non-invertible, though we can compute their pseudo-inverses using the formula above.

![SVD of non-square matrices](image)

Figure 3: SVD of non-square matrices. The gray regions of the matrices are not needed, since they consist of all zeros in the \( S \) matrix, and are only hit by zeros in those portions of the \( U \) or \( V^\top \) matrices. Dropping them results in the more compact “reduced” SVD for tall, skinny (above) or short, fat (below) matrices.