Lecture 11
The Singular Value Decomposition

Jason J. Bramburger

As I'm sure you're already aware, much of computational mathematics is based on linear algebra. We've already seen this when we discretize the heat equation in space to arrive at a coupled linear system of ordinary differential equations. We are going to continue making use of linear algebra as we continue in this course, primarily using a single powerful tool: the singular value decomposition (SVD). The SVD is potentially the most powerful and versatile tool that can be used in data analysis (and computational math in general), but (if you're like me) it is typically not even taught in early courses on linear algebra. This is a tragedy, so today we will discuss much of the theoretical underpinnings to the SVD, as well as discuss the SVD in a broader context.

A Geometric Interpretation

Let's start by building a bit of intuition about the SVD by thinking about it geometrically. Recall that a matrix represents a linear transformation: it takes a vector and transforms it to another vector. To be concrete, let $\mathbf{A}$ be a 2×2 matrix, so that it transforms a vector in $\mathbb{R}^2$ to another vector in $\mathbb{R}^2$. This means that if we have a vector $\mathbf{x} \in \mathbb{R}^2$, then $\mathbf{Ax}$ is also a vector in $\mathbb{R}^2$.

In general, matrix multiplication will rotate and/or scale a vector. For example, the *rotation matrix* is given by

$$\mathbf{R}_\theta = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

so that $\mathbf{R}_\theta \mathbf{x}$ is just the original vector $\mathbf{x}$ rotated counterclockwise around the origin by an angle of θ. Here are some properties of the rotation matrix:

1. We can invert the rotation matrix by simply rotating the vector $\mathbf{R}_\theta \mathbf{x}$ counterclockwise around the origin by an angle of $-\theta$. Hence, the inverse of the rotation matrix is given by

$$\mathbf{R}_\theta^{-1} = \mathbf{R}_{-\theta} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}.$$

   Notice that $\mathbf{R}_\theta^{-1} = \mathbf{R}_\theta^T$, the transpose. Matrices with this property are called **orthogonal** matrices. All of their columns (and rows) are orthonormal.

2. If you have matrices with complex numbers in them, the analogous property to point (1) above is $\mathbf{A}^{-1} = A^*$, where $\mathbf{A}^*$ is the conjugate transpose of the matrix $\mathbf{A}$. Matrices with this property are called **unitary** matrices. Notice that all real orthogonal matrices are also unitary, and so the rotation matrix is both unitary and orthogonal.

3. Unitary and orthogonal matrices have an important property in that they do not stretch or compress a vector. Mathematically, this means that the Euclidean norm (2-norm) of a vector $\mathbf{x} \in \mathbb{R}^N$, given by

$$\|\mathbf{x}\|_2 = \sqrt{\sum_{n=1}^{N} |x_n|^2}$$

   is the same as the Euclidean norm of $\mathbf{Ax}$, for any unitary matrix $\mathbf{A}$. Precisely, $\|\mathbf{x}\|_2 = \|\mathbf{Ax}\|_2$ when $\mathbf{A}$ is unitary.

This last property, geometrically, means that the rotation matrix just maps circles back into circles. Indeed, a circle is given by the set of vectors with $\|\mathbf{x}\|_2$ fixed, and so applying a rotation matrix to every vector in this set means that you arrive at the same circle.

To scale a vector we can consider another special matrix of the form

$$\mathbf{A} = \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix}$$

for some $\alpha > 0$. If $\alpha > 1$ then this matrix just stretches vectors by a factor of α, and if $0 < \alpha < 1$ it compresses them by this factor. Considering $\alpha < 0$ results in the vector flipping to point in the opposite direction, and then stretched or compressed by a factor of $|\alpha|$. Geometrically, this means that circles with radius $r > 0$ are mapped to circles with radius $|\alpha|r$, so the circle is either stretched or compressed too.

Let's now consider the diagonal matrix

$$\mathbf{A} = \begin{bmatrix} 2 & 0 \\ 0 & 1/2 \end{bmatrix}.$$

Applying $\mathbf{A}$ to a vector in $\mathbb{R}^2$ results in a stretching of the first component by a factor of 2 and a compressing of the second component by a factor of $1/2$. This means that a circle would be elongated in the x-direction and compressed in the y-direction, resulting in an ellipse. Precisely, the major semiaxis of this ellipse is on the x-axis and has length 2, while the minor semiaxis is on the y-axis and has length $1/2$.

Although we have illustrated by example above, it turns out that multiplication by any matrix turns circles into ellipses. We can label the **principal semiaxes** of the ellipse resulting from applying the matrix to the unit circle as $\sigma_1\mathbf{u}_1$ and $\sigma_2\mathbf{u}_2$, where σ_1 and σ_2 are the lengths of the vectors and $\mathbf{u}_1$ and $\mathbf{u}_2$ are unit vectors that point in the directions of the axes. Then, there are some unit vectors vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ such that

$$\mathbf{A}\mathbf{v}_1 = \sigma_1\mathbf{u}_1, \quad \mathbf{A}\mathbf{v}_2 = \sigma_2\mathbf{u}_2.$$

In matrix form, this means that

$$\mathbf{AV} = \mathbf{U\Sigma},$$

where $\mathbf{V}$ is a unitary matrix with $\mathbf{v}_1$ and $\mathbf{v}_2$ as its columns, $\mathbf{U}$ is unitary matrix with $\mathbf{u}_1$ and $\mathbf{u}_2$ as its columns, and $\mathbf{\Sigma}$ is a diagonal matrix with σ_j on its diagonals. Since V is unitary (and therefore invertible), we can isolate for $\mathbf{A}$ to get:

$$\mathbf{A} = \mathbf{U\Sigma V}^*,$$

thus giving us a decomposition of the matrix A through unitary matrices $\mathbf{U}$ and $\mathbf{V}$ and a diagonal matrix $\mathbf{\Sigma}$.

The Reduced Singular Value Decomposition

We obviously want to work with matrices that are bigger than just 2×2, and unlike say eigenvalue calculations and determinants, we would like to be able to handle non-square matrices. It turns out that the theory discussed above for 2×2 matrices generalizes nicely to (not necessarily square) matrices of arbitrary size. That is, let $\mathbf{A}$ be an $m \times n$ matrix, representing a transformation from $\mathbb{R}^n$ to $\mathbb{R}^m$. The image of the n-dimensional unit sphere under $\mathbf{A}$ is a hyperellipse in m-dimensional space. In the case that $m \geq n$ and $\mathbf{A}$ is full rank, the hyperellipse will be n-dimensional. There are now n-principal semiaxes with lengths $\sigma_1, \sigma_2, \ldots, \sigma_n$ that point in the directions $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n$. Again there are vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ such that

$$\mathbf{A}\mathbf{v}_1 = \sigma_1\mathbf{u}_1, \quad \mathbf{A}\mathbf{v}_2 = \sigma_2\mathbf{u}_2, \ldots, \quad \mathbf{A}\mathbf{v}_n = \sigma_n\mathbf{u}_n.$$

We can again write this compactly as

$$\mathbf{AV} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}},$$

where the matrices have the same meaning as they did in the 2×2 case. Again using the fact that the columns of $\mathbf{V}$ form an orthonormal basis, we can isolate for $\mathbf{A}$ to get

$$\mathbf{A} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\mathbf{V}^*.$$

The above decomposition of the matrix $\mathbf{A}$ is called the **reduced SVD**. Notice that $\hat{\mathbf{U}}$ is an $m \times n$ matrix with orthonormal columns.

The Singular Value Decomposition

The reduced SVD is not the standard definition of the SVD used in the literature. What is typically done is to construct a matrix $\mathbf{U}$ from $\hat{\mathbf{U}}$ by adding an addition $m-n$ columns that are orthonormal to the already existing set $\hat{\mathbf{U}}$. In this way, the matrix $\mathbf{U}$ becomes a square $m \times m$ matrix, and in order to make the decomposition work, and additional $m-n$ rows of zeros is also added to the matrix $\hat{\mathbf{\Sigma}}$, resulting in the matrix $\mathbf{\Sigma}$. This leads to the **singular value decomposition** of the matrix $\mathbf{A}$:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*,$$

where $\mathbf{U} \in \mathbb{R}^{m\times m}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$ are unitary matrices, and $\mathbf{\Sigma} \in \mathbb{R}^{m\times n}$ is diagonal.

Geometrically, the SVD is doing the following. Multiplying by $\mathbf{V}^*$ on the left rotates the hypersphere to align the $\mathbf{v}_n$ vectors with the axes. Then we multiply on the left by a diagonal matrix which stretches in each direction. Then we multiply on the left by $\mathbf{U}$ to rotate the hyperellipse to the proper orientation.

Definition: The values σ_n on the diagonal of $\mathbf{\Sigma}$ are called the **singular values** of the matrix $\mathbf{A}$. The vectors $\mathbf{u}_n$ which make up the columns of $\mathbf{U}$ are called the **left singular vectors** of $\mathbf{A}$. The vectors $\mathbf{v}_n$ which make up the columns of $\mathbf{V}$ are called the **right singular vectors** of $\mathbf{A}$.

Properties of the SVD

1. Every matrix $\mathbf{A} \in \mathbb{C}^{m\times n}$ has an SVD.
2. The singular values are uniquely determined and are always non-negative real numbers.
3. The singular values are always ordered from largest to smallest, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$, along the diagonal of Σ.
4. The number of nonzero singular values is the rank of $\mathbf{A}$.
5. Letting $r = \text{rank}(\mathbf{A})$ we have that $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r\}$ is a basis for the range of $\mathbf{A}$ and $\{\mathbf{v}_r + 1, \ldots, \mathbf{v}_n\}$ is a basis for the null space of $\mathbf{A}$.
6. MATLAB actually calculates the rank and a basis for the null space using the SVD when you type in the rank() or null() commands.

Computing the SVD

The first point in the above list tells you that the SVD exists for any matrix, but a question that arises is how do we calculate it? This can be done by observing that

$$\begin{aligned}\mathbf{A}^*\mathbf{A} &= (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^*)^*(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^*) \\ &= \mathbf{V}\mathbf{\Sigma}\mathbf{U}^*\mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \\ &= \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^*.\end{aligned}$$

Multiplying on the right by $\mathbf{V}$ gives

$$\mathbf{A}^*\mathbf{A}\mathbf{V} = \mathbf{V}\mathbf{\Sigma}^2,$$

so that the columns of $\mathbf{V}$ are eigenvectors of $\mathbf{A}^*\mathbf{A}$ with eigenvalues given by the square of the singular values. Similarly,

$$\mathbf{A}\mathbf{A}^* = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^*.$$

and so multiplying on the right by $\mathbf{U}$ gives the eigenvalue problem

$$\mathbf{A}\mathbf{A}^*\mathbf{U} = \mathbf{U}\mathbf{\Sigma}^2,$$

which can be used to solve for $\mathbf{U}$.

Comparing SVD and Eigenvalue Decomposition

We just saw that there is a connection between eigenvalues and the SVD. Let's explore this connection a bit more

by considering a square $n \times n$ matrix $\mathbf{A}$. Recall that if $\mathbf{A}$ has n linearly independent eigenvectors, then we can **diagonalize** the matrix, meaning we can write it as

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1},$$

where the columns of $\mathbf{V}$ are the eigenvectors and $\mathbf{\Lambda}$ is a diagonal matrix with eigenvalues on the diagonal. Geometrically, multiplying on the left by $\mathbf{V}^{-1}$ is changing to a basis of eigenvectors, then multiplying by $\mathbf{\Lambda}$ stretches or compresses the eigenvectors, and multiplying by $\mathbf{V}$ goes back to the original basis. Based on this geometric understanding, you should notice some similarity with the SVD. But, how are they different?

1. You can only diagonalize a square matrix, while an SVD can be performed on matrices of any size. This is extremely important since real-world data doesn't always come in the form of a square matrix.

2. Both are about finding bases, but the eigenvalue decomposition finds a single basis while the SVD uses two different ones. Furthermore, the SVD gives orthogonal bases while eigenvectors are not always orthogonal.

3. In the case that the eigenvectors of $\mathbf{A}$ are orthogonal (such as when $\mathbf{A}$ is a symmetric matrix) then the singular values are just the absolute values of the eigenvalues and the singular vectors are eigenvectors.

4. Finally, as we saw above, it is through eigenvectors and eigenvalues that we find the SVD. Be careful though - sometimes I get confused and think that the singular values are just the square of the eigenvalues of the matrix $\mathbf{A}$. Based on our work above, this is of course not true. The singular values are the square of the eigenvalues of $\mathbf{A}^*\mathbf{A}$ (which has the same eigenvalues as $\mathbf{A}\mathbf{A}^*$).

Matrix Norms

To continue presenting some properties of the SVD, we need to briefly discuss matrix norms. Recall that for vectors, norms are a way of asking "how large" a vector is. The question is posed similarly for matrices, while sometimes we think of a matrix norm as measuring the *energy* of the matrix. This will become more clear as we work through some applications in later lectures.

Just like with vectors, there are many different ways to capture the norm of a matrix. We will introduce two: the 2-norm and the Frobenius norm. Let's start with the former. The 2-norm of a matrix is given by

$$\|\mathbf{A}\|_2 := \sup_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2.$$

Now, recall that the SVD of $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$ contains the unitary matrices $\mathbf{U}$ and $\mathbf{V}^*$, This means that multiplying them by any vector does not change the 2-norm (see above), so all the stretching and compressing of $\mathbf{A}\mathbf{x}$ comes from the entries in $\mathbf{\Sigma}$. Hence, the largest that a vector can be stretched by is σ_1, the largest singular value, and so

$$\|\mathbf{A}\|_2 = \sigma_1.$$

Another more geometric way to think about this is that if you take the unit ball ($\|\mathbf{x}\|_2 = 1$), the vector that stretches the most is the one that goes to the longest semiaxis of the hyperellipse.

The 2-norm can sometimes be hard to get your head around since it is indirect. That is, to compute it you don't really do a computation just using the entries of the matrix, you need to understand how it is acting as a matrix transformation. There is another matrix norm, the Frobenius norm, that is more direct. It basically is just the 2-norm for vectors, but over both the rows and columns of the matrix. We write

$$\|\mathbf{A}\|_F := \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{i,j}|^2}.$$

Similar to the 2-norm, the Frobenius norm has a deep connection to the singular values. In fact,

$$\|\mathbf{A}\|_F = \sqrt{\sum_{j=1}^{r} \sigma_j^2},$$

where r is the rank of $\mathbf{A}$. This comes from the fact that the Frobenius norm is of a matrix is the same as the Frobenius norm of that matrix multiplied by a unitary matrix. Since $\mathbf{U}$ and $\mathbf{V}^*$ are unitary, the Frobenius norm of $\mathbf{A}$ is just the Frobenius norm of $\mathbf{\Sigma}$. Of course, the only nonzero entries in $\mathbf{\Sigma}$ are the singular values, giving the above expression for $\|\mathbf{A}\|_F$ in terms of singular values.

Low-Dimensional Approximations

Now that we have a good understanding of the SVD, let's see why it is of so much interest to us in this class. To do this, we first need the following theorem.

Theorem: If $\mathbf{A}$ is a matrix of rank r, then $\mathbf{A}$ is the sum of r rank 1 matrices:

$$\mathbf{A} = \sum_{j=1}^{r} \sigma_j \mathbf{u}_j \mathbf{v}_j^*.$$

The product $\mathbf{u}_j \mathbf{v}_j^*$ is called an *outer product*. If $\mathbf{A}$ is of size $n \times m$, then $\mathbf{u}_j$ is size $m \times 1$ and $\mathbf{v}_j^*$ is size $1 \times n$, meaning that the outer product is the same size as $\mathbf{A}$. The important point is that a matrix that comes from an outer product has rank 1.

The above representation of $\mathbf{A}$ by the sum of rank 1 matrices turns out to give us an excellent method for approximating $\mathbf{A}$. In particular, imagine the rank, r, is large. We don't have to sum all the way up to r, we could just sum the first N terms. This is known as the **best** rank N approximation that is possible for $\mathbf{A}$. This is summarized in the following theorem.

Theorem: For any N so that $0 \leq N \leq r$, we can define the partial sum

$$\mathbf{A}_N = \sum_{j=1}^{N} \sigma_j \mathbf{u}_j \mathbf{v}_j^*.$$

Then,

$$\|\mathbf{A} - \mathbf{A}_N\|_2 = \min_{\mathbf{B} \in \mathbb{C}^{m \times n}, \text{rank}(\mathbf{B}) \leq N} \|\mathbf{A} - \mathbf{B}\|_2 = \sigma_{N+1},$$

and

$$\|\mathbf{A} - \mathbf{A}_N\|_F = \min_{\mathbf{B} \in \mathbb{C}^{m \times n}, \text{rank}(\mathbf{B}) \leq N} \|\mathbf{A} - \mathbf{B}\|_F = \sqrt{\sigma_{N+1}^2 + \cdots + \sigma_r^2}.$$

Mathematically this theorem says that out of all the matrices that are rank N or less, the best approximation of $\mathbf{A}$ (relative to the two different norms) is given by $\mathbf{A}_N$. The reason this is important is because it is going to allow us to do approximations of data. So far, we have been talking about matrices as linear transformations (the geometric interpretation), but we can also use a matrix to store data. Take, for example, a greyscale image. Saving the whole image might take a lot of storage, especially if it is made up of a lot of pixels. The low-rank approximations using the SVD are a tool for doing *dimensionality reduction* or *low-dimensional approximations* of your data. This means that you can represent your image without having to store every single pixel value. Furthermore, the SVD tells you exactly how to keep the most amount of information while keeping the fewest number of dimension to your data. It does this by capturing important trends or correlations in your data. Hence, we can think of the SVD as a trend finder.