Discussion Session 4

Luis Ángel Larios Cárdenas

January 29, 2016

1 Singular Value Decomposition

The *Singular Value Decomposition* or **SVD** allows us to break any matrix (square or rectangular) into the product of three matrices. Let A be an $n \times p$ matrix, then, its SVD is given by:

$$A = U\ S\ V'$$

where U is an $n \times n$ orthogonal matrix whose columns are the *left singular vectors* $\hat{\mathbf{v}}_i$, S is an $n \times p$ diagonal matrix that holds the *singular values* σ_i, and U is a $p \times p$ orthogonal matrix whose columns are the *right singular vectors* $\hat{\mathbf{u}}_i$. Figure 1 is an schematic representation of the SVD of matrix A.

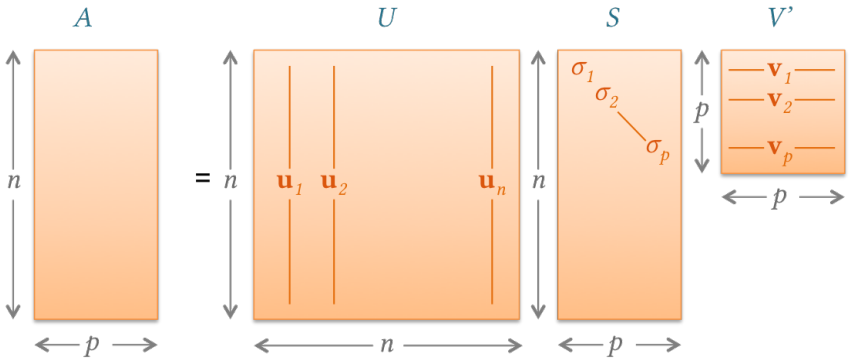

Figure 1: Block representation of the SVD of A.

We can analyze matrix the SVD of matrix A from two perspectives: when A represents a *transformation*, and when A is a *dataset*.

1.1 SVD of a Transformation Matrix

Like in the *Eigendecomposition* analysis, a matrix A is a transformation or change of basis that can be broken into three steps: a rotation, a dilation, and another rotation. The SVD is not different, but is more general than the Eigenstructure. Consider, for instance, the following transformation matrix T:

$$T = \begin{pmatrix} 1.5 & 2.0 & 1.0 \\ 2.1 & 1.0 & 0.5 \end{pmatrix}$$

T is a 2×3 matrix. As a transformation matrix, it takes points in 3D space, and project them (i.e. removes one dimension) onto a 2D space. Figures 2 and 3 show what T does the unit 3D sphere: if flattens, stretch, and tilt the sphere into a 2D ellipsoid.

But, how can we explain this behavior of T? We just need to look at its SVD. Given that $T = U\ S\ V'$, V is an orthogonal 2×2 matrix, S is a diagonal 2×3 matrix, and U is an orthogonal 3×3 matrix. In the same

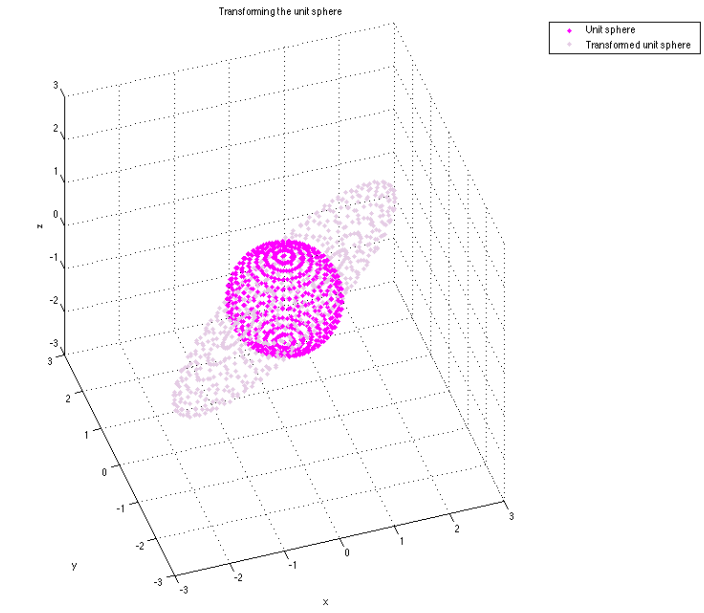

Figure 2: The unit sphere as transformed by T (one view).

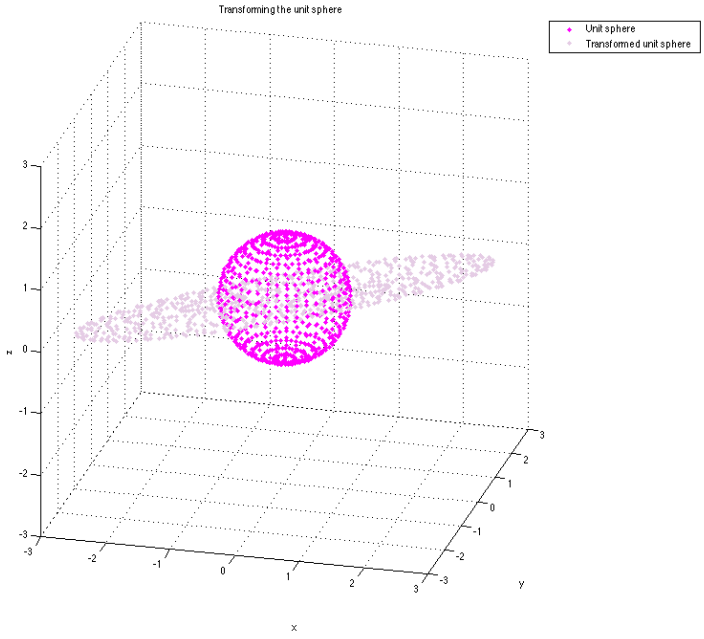

Figure 3: The unit sphere as transformed by T (another view).

way we approached the *Eigendecomposition* in our previous notes[1], T is just a series (or concatenation) of transforms. Let $\mathbf{a}$ be a vector in 3D, then:

[1]Notes available at CCLE or at http://youngmin.one/documents/ta/cs170A-W16/Session2.zip

$$\mathbf{b} = f(\mathbf{a})$$
$$\mathbf{b} = T\mathbf{a}$$
$$\mathbf{b} = USV'\mathbf{a}$$
$$\mathbf{b} = {}_0U_1 \ {}_1S_2 \ {}_2(V')_3 \ \mathbf{a}$$

where the transformation matrix ${}_iM_j$ indicates that M maps points expressed in terms of the j coordinate system, into its equivalence with respect to coordinate system i.

So, given our 2×3 transformation matrix T, the following happens:

1. ${}_2(V')_3$ gives us the coordinates of the 3D point $\mathbf{a}$, but now with respect to the coordinate system ${}_0U_1 \ {}_1S_2$ —here, the point is still in 3D, it was just rotated around.

2. ${}_1S_2$ gives us the coordinates of the (still) 3D point ${}_2(V')_3 \ \mathbf{a}$, but now with respect to the coordinate system ${}_0U_1$ —here, notice what has happened: we have lost one dimension! S stretched 2 of the axes and zeroed the other one! After the mapping, we are no longer in 3D, but in 2D.

3. Finally, ${}_0U_1$ gives us the coordinates of the 2D point ${}_1S_2 \ {}_2(V')_3 \ \mathbf{a}$, but now with respect to the coordinate system 0, which is the "world frame". The transform U just rotated the poor 2D vector that it was left with after the ${}_1S_2 \ {}_2(V')_3 \ \mathbf{a}$ mutilation.

In summary, our transformation matrix T departed from a 3D coordinate system, and arrived at a 2D coordinate system. Along the way, the S matrix was the one who took care of removing one of the 3 dimensions by introducing a 'zero' singular value (which doesn't appear in S, of course).

Again, like in the *Eigendecomposition*, each singular value is matched to a singular vector. Thus, the largest singular value represents the maximum stretch that T performs on any point. Figure 4 shows the left singular vectors (i.e. U) being stretched by their respective singular values. You can notice that the largest ellipsoid axis corresponds to $\hat{\mathbf{u}}_1$ scaled by σ_1. Also, the effect of T on the unit sphere (see figure 2), does not only flattens the sphere, but it 'twists' the sphere too, as you may notice in the two subtle pink ellipses that actually correspond to the original sphere poles.

1.2 SVD of a Dataset

Any $n \times p$ matrix A, like the one portrayed in figure 1, can be something different than a transformation: it can be a dataset with n observations and p features. In this case, the SVD can help us extract hidden characteristics and further analyze the structure of matrix A.

In particular, when A is an image, we can do *image compression* by first decomposing $A = USV'$, then keeping only the k largest (significant) singular values: $\sigma_1, \sigma_2, \cdots, \sigma_k$, where $k < p$ and $k < n$, and reconstructing A as:

$$A^{(k)} = U^{(k)} \ S^{(k)} \ \left(V^{(k)}\right)'$$

where $U^{(k)}$ is an $n \times k$ matrix, $S^{(k)}$ is a $k \times k$ square matrix, and $V^{(k)}$ is a $p \times k$ matrix. We can better see this assemblage from figure 5, where we only need to keep k left singular vectors $\hat{\mathbf{u}}_i$ and k right singular vectors $\hat{\mathbf{v}}_i$, for $1 \leq i \leq k$. At the end, $A^{(k)}$ has the same dimensions than A, but the image quality will depend greatly on how many *insignificant* singular values we threw away.

As an example, if $A = USV'$ is a 600×500 gray-scale image (figure 6), we can use the following Matlab code:

```
[U, S, V] = svd( A );       % U is nxn, S is nxp, and V is pxp.

k = 100;
Uk = U(:, 1:k);             % Throw away left singular vectors beyond the k^th.
```

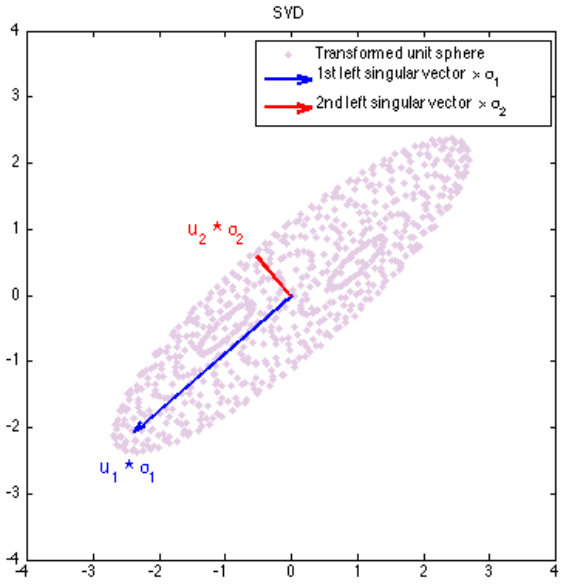

Figure 4: Left singular vectors of T, scaled by their respective singular value.

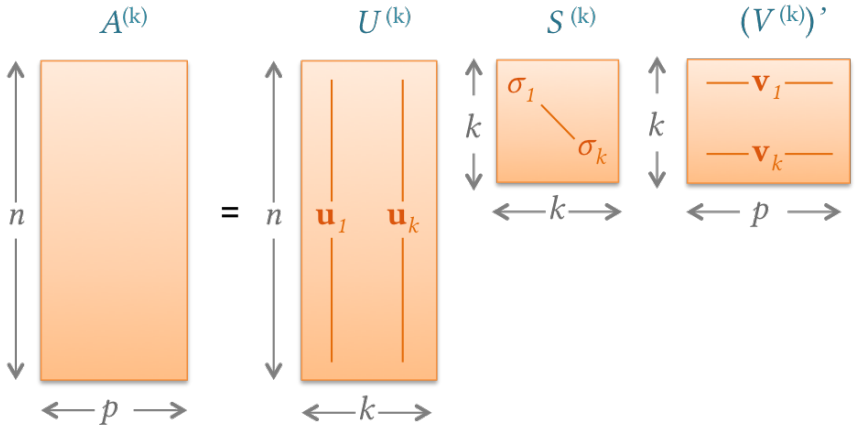

Figure 5: Block diagram for approximating A but using only the largest k singular values.

```
Vk = V(:, 1:k);              % Throw away right singular vectors beyond the k^th.
Sk = S(1:k, 1:k);            % We only need k singular values.

Ak = Uk * Sk * Vk';          % k^th approximation of Taeyeon.
```

to approximate A, but only with $k = 100$ singular values. The resulting $A^{(k)}$ can be seen in figure 7.

Figure 6: Original image A.

Figure 7: "Compressed" image $A^{(k)}$, when we preserve $k = 100$ singular values.