8 rincipal Component Analysis, Image Compression

8.1 Principal Component Analysis (PCA)

PCA is a method of working with n points in $\mathbb{R}^m$, and finding a k-dimensional space ($k < m$) spanned by orthonormal vectors $\vec{v}_1, \cdots, \vec{v}_k$ so thatthe points have the greatest variance relative to direction $\vec{v}_i$ (reduce the dimensionality so that all of the points look pretty spread out).

Definition 8.1
Given $\{\vec{a}_1, \cdots, \vec{a}_n\}$, the **sample variance relative to** μ of the set is

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n} ||\vec{a}_i - \vec{\mu}||^2$$

We assume everything relative to the origin, so

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n} ||\vec{a}_i||^2$$

The **covariance matrix** is

$$C = \frac{1}{n-1}A^T A$$

Observe

$$\frac{1}{n-1}\begin{bmatrix} \vec{a}_1 & \longrightarrow \\ \vec{a}_2 & \longrightarrow \\ \vdots & \\ \vec{a}_n & \longrightarrow \end{bmatrix}\begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \cdots & \vec{a}_n \\ \downarrow & \downarrow & & \downarrow \end{bmatrix} = \frac{1}{n-1}\begin{bmatrix} ||\vec{a}_1||^2 & & & \\ & ||\vec{b}_2||^2 & & \\ & & \ddots & \\ & & & ||\vec{a}_n||^2 \end{bmatrix}$$

Which means that $\text{tr}(\frac{1}{n-1}A^T A) = \frac{1}{n-1}\sum_{i=1}^{n} ||\vec{a}_i||^2 = \sigma^2 =$ the variance!

But this is also the sum of the eigenvalues of the matrix $\frac{1}{n-1}A^T A$ as we discussed last lecture.

But we also know that the singular values $s_1, \cdots, s_k$ in the SVD of A are the square roots of the shared positive eigenvalues of $A^T A$ and AA^T.

Definition 8.2
The proportion of total variance of the data $\{a_1, \cdots, a_n\}$ in the direction of $\vec{v}_i$ is

$$\frac{s_i^2}{s_1^2 + s_2^2 + \cdots + s_k^2} = \frac{\lambda^i}{\sum_{j=1}^{n} \lambda_j}$$

The **total variance presevered** by $t < k$ singular values is

$$\frac{s_1^2 + \cdots + s_t^2}{s_1^2 + \cdots + s_k^2} = \frac{\sum_{i=1}^{t} \lambda_i}{\sum_{i=1}^{k} \lambda_i}$$

8.2 Spread of Data

Let $\vec{a}_1, \cdots, \vec{a}_n$ be data points. The SVD is

$$A\begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \cdots \vec{a}_n & \\ \downarrow & \downarrow & & \downarrow \end{bmatrix} = \begin{bmatrix} \vec{u}_1 & \cdots & \vec{u}_m \\ \downarrow & U & \downarrow \end{bmatrix}\left[\begin{array}{ccc|c} s_1 & & \Sigma & \\ & \ddots & & 0 \\ & & s_k & \\ \hline & 0 & & 0 \end{array}\right]\begin{bmatrix} \vec{v}_1 & \longrightarrow \\ \vdots & V^T \\ \vec{v}_n & \longrightarrow \end{bmatrix}$$

Where

$$V^T = \begin{bmatrix} v_{11} & v_{12} & \cdots \\ v_{21} & & \\ \vdots & & \end{bmatrix}$$

Expanding the product, we find

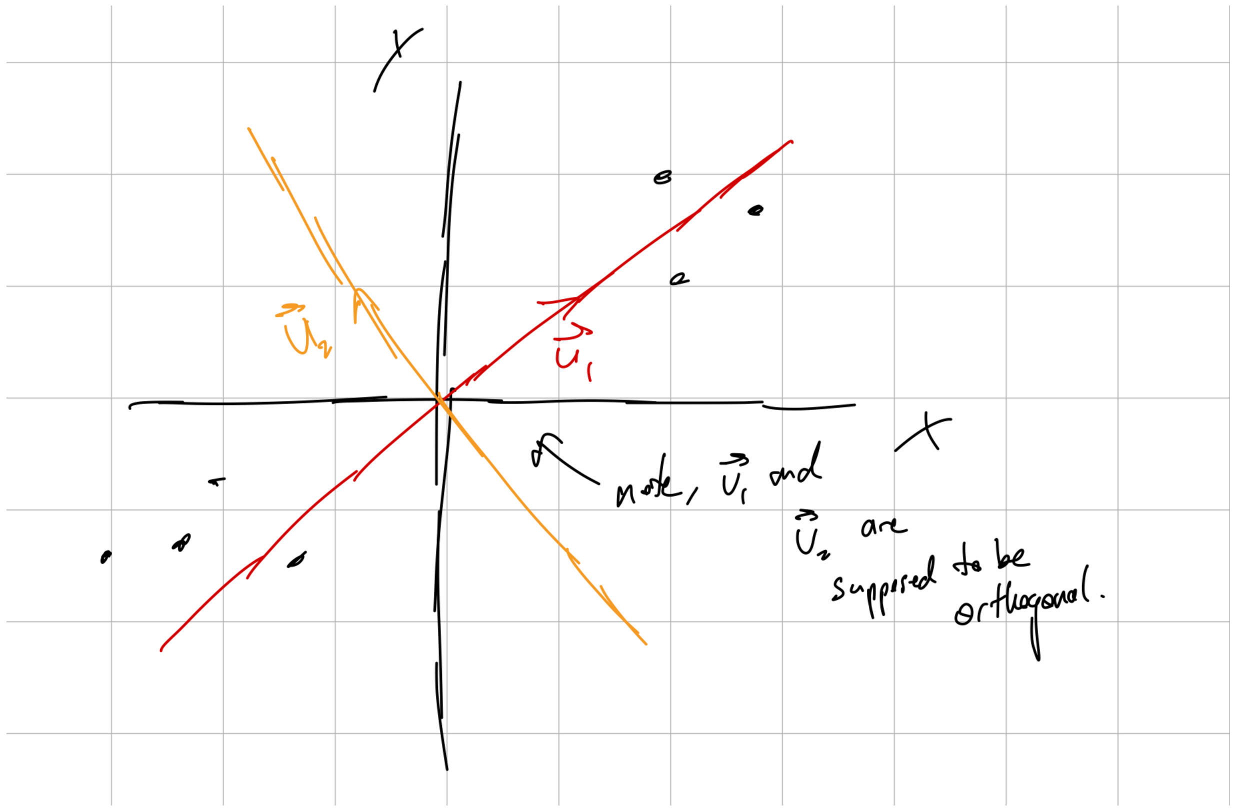
$$\vec{a}_1 = s_1 v_{11} \vec{u}_1 + s_2 v_{21} \vec{u}_2 + \cdots + s_k v_{k1} \vec{u}_k$$

$$\vec{a}_n = s_1 v_{1n} \vec{u}_1 + \cdots + s_k v_{kn} \vec{u}_k$$

Recall the singular values are non-increasing, i.e. $s_1 \geq s_2 \geq \cdots \geq s_k > 0$.
Note that s_1 is the largest, and v_{11} is a small positive or negative number.

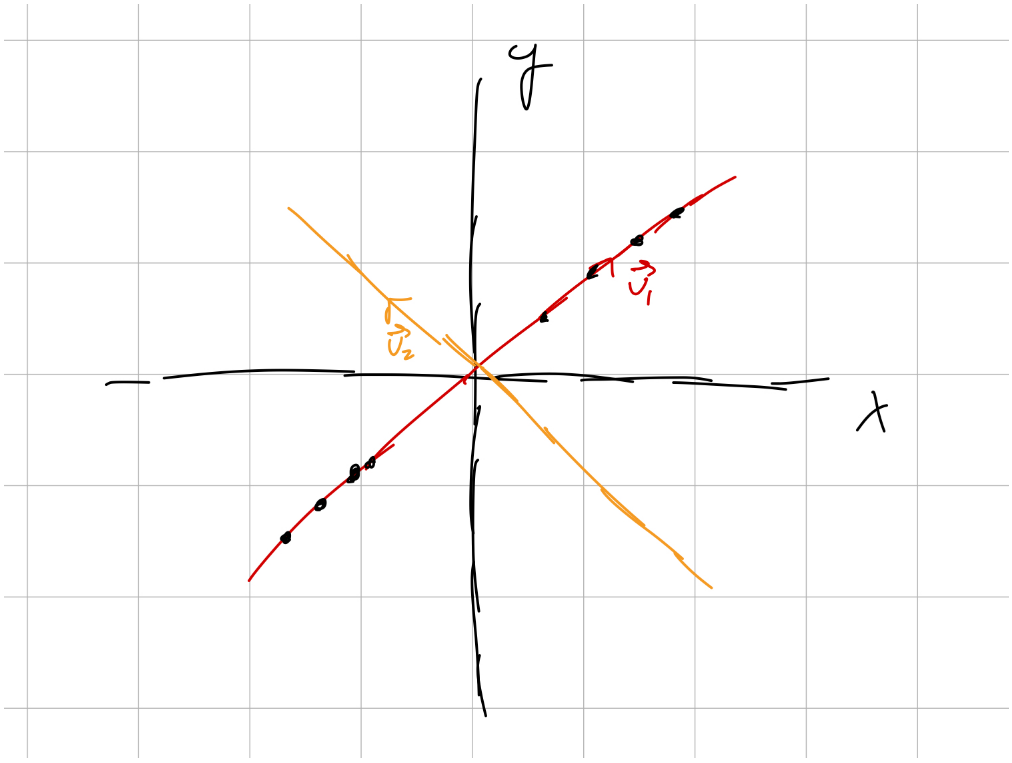
Data points $\vec{a}_i$ are linear combinations of $\vec{u}_i$'s.

$\vec{u}_1$ is scaled by the largest amount by its corresponding eigenvector, and it can be scaled either positively or negatively depending on v_{11}.
In other words, the line formed by u has the greatest scaling factor (s_1), so we can expect points $\vec{a}_1, \cdots, \vec{a}_n$ to be greatly spread out relative to this line.



In the above picture, the points vary a lot more along the line formed by $\vec{u}_1$ compared to the line formed by $\vec{u}_2$.

What happens if $s_2 = 0$?



Then all of the points are on the line formed by $\vec{u}_1$! The data is still relatively spread out because $\vec{u}_1$ was originally affecting the spread of the data the most.

Thus the spread is dependent on the largest singular values.

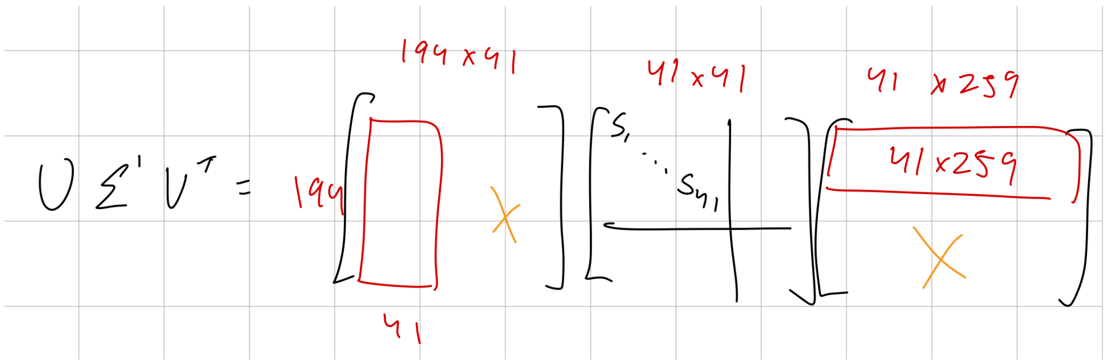
8.2.1 Image Compression

We represent an image using grayscale, where each pixel is assigned a value between 0 (black) to 1 (white).

The matrix A represents the grayscale values of the image.

The original image (handed out in class) is $194 \times 259 = 50246$ pixels.
The SVD of A is $U\Sigma V^T$. We then set the small singular values equal to 0. We want a small effect on the total variance.
We get a new image with matrix $A' = U\Sigma' V^T$.

Using only 41 singular values, we can compress the matrices as follows:



We now need to only store $(41)(194) + 41^2 + (41)(259) = 20254$ pixels. We compressed to less than half of the original data, while still capturing over 99% of the variance in the image. This turns out to be a decent compression.