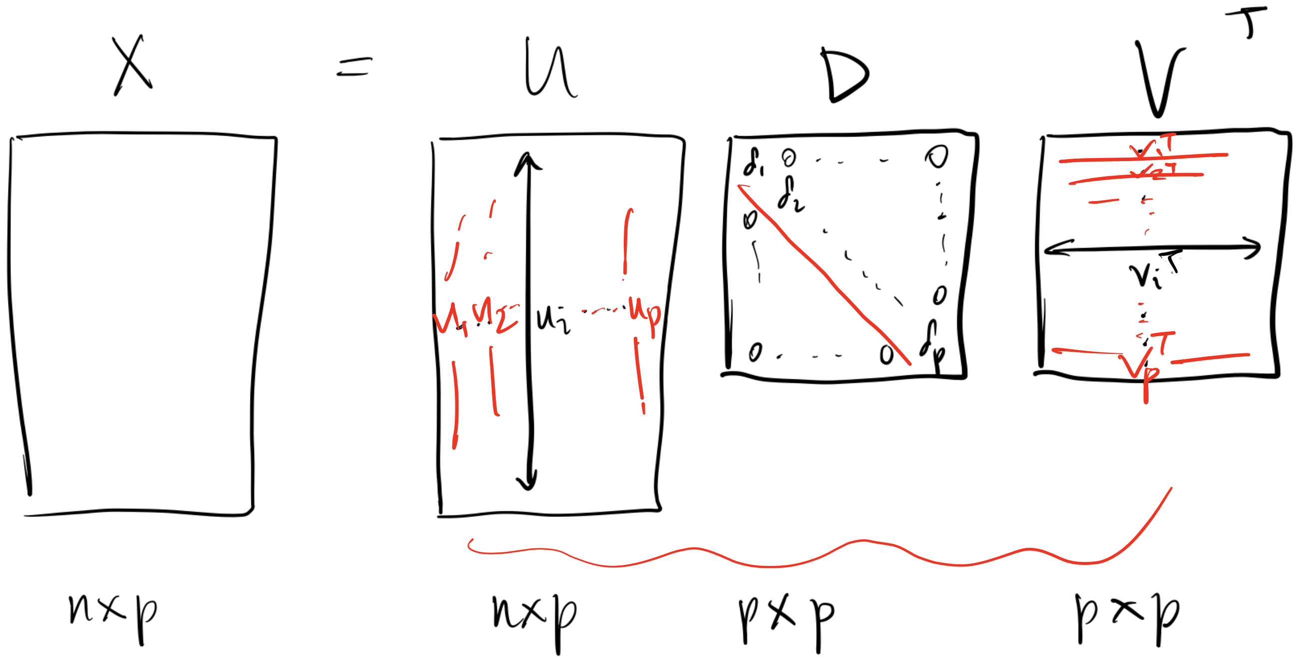
Singular Value Decomposition

Suppose we have a matrix X with n rows & p columns (without loss of generality, assume $n \geq p$). It turns out that every matrix X has a *singular value decomposition*:



where the following statements are true:

1. The columns of U: $u_1, ..., u_p$ are orthonormal vectors and are called the *left singular vectors* of X.
2. $U^T U = I_p$
3. The rows of V: $v_1, ..., v_p$ are orthonormal vectors and are called the *right singular vectors* of X.
4. $V^T V = I_p$ (i.e. the rows of V: $v_1, ..., v_p$ are orthonormal vectors)
5. The diagonal entries of D, $\{\delta_1, \ldots, \delta_p\}$, are nonnegative & called the *singular values* of X. Without loss of generality we can order them such that they are always nonincreasing:

$$\delta_1 \geq \delta_2 \geq \ldots \geq \delta_p \geq 0.$$

Note: If $n < p$, an SVD still exists, but now U is square and V is not.

Alternative form of the SVD:

If we carry out the matrix multiplication, we can formulate the decomposition as:

$$X = \sum_{i=1}^{\cancel{d}\; p} \underbrace{\delta_i u_i v_i^T}_{\text{outerproduct}} \qquad \text{Why?}$$

Things to notice:

- We have expressed the design matrix X as a weighted sum of rank 1 matrices: $u_1 v_1^T, ..., u_p v_p^T$ with weights equal to the singular values, $\{\delta_1, \ldots, \delta_p\}$.
- Some of the singular values could be zero! If there are r ~~nonnegative~~ nonzero singular values, then $\text{rank}(X) = r$.
- Recall that when X is a design matrix, then its rank is the dimension of the space in which variation of its data actually exists! If all the sample data points $x_1, ..., x_n$ (defined by the rows of the design matrix X) lie on a line, there is only one nonzero singular value. If all the sample data points lie on a plane, there are only two nonzero singular values. In general, if all of the points span a subspace of dimension r, there are r nonzero singular values.

Practical Question:

How do I find out what $U, D,$ & V are?

Consider $X^T X$:

~~$n \times n$~~

$$X^T X = (UDV^T)^T (UDV^T) = V(D^T D)V^T$$

$$= V \begin{pmatrix} \delta_1^2 & 0 & \cdots & 0 \\ 0 & \delta_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_p^2 \end{pmatrix} V^T$$

Does this structure/factorization look familiar?

Claim: V is the matrix of eigenvectors of $X^T X$ and $\delta_1^2, \ldots, \delta_p^2$ are the corresponding eigenvalues.

Proof:

$$X^T X v_1 = V(D^T D)V^T v_1$$

$$= V(D^T D) \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_p^T \end{bmatrix} v_1$$

$$= V(D^T D) \begin{bmatrix} v_1^T v_1 \\ v_1^T v_2 \\ \vdots \\ v_1^T v_p \end{bmatrix} = V(D^T D) \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = V \begin{bmatrix} \delta_1^2 & & 0 \\ & \delta_2^2 \ddots & \\ 0 & & \delta_p^2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$= V \begin{bmatrix} \delta_1^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$= \delta_1^2 v_1$$

$$X = \begin{bmatrix} u_1 & u_2 & \cdots & u_p \end{bmatrix} \underbrace{\begin{bmatrix} \delta_1 & 0 & \cdots & 0 \\ 0 & \delta_2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & \cdots & \delta_p \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_p^T \end{bmatrix}}$$

$$= \begin{bmatrix} u_1 & \cdots & u_p \end{bmatrix} \left[\begin{pmatrix} \delta_1 & 0 & \cdots & 0 \\ & 0 & & \vdots \\ & & \ddots & 0 \\ 0 & \cdots & & 0 \end{pmatrix} + \begin{pmatrix} 0 & & & 0 \\ & \delta_2 & & \\ & & \ddots & \\ 0 & & & 0 \end{pmatrix} + \cdots + \begin{pmatrix} 0 & & & 0 \\ & \ddots & & \\ & & 0 & \\ 0 & & & \delta_p \end{pmatrix} \right] \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_p^T \end{bmatrix}$$

$$= U \begin{pmatrix} \delta_1 & & \\ & 0 & \\ & & \ddots & \\ & & & 0 \end{pmatrix} V^T + U \begin{pmatrix} 0 & & \\ & \delta_2 & \\ & & \ddots & \\ & & & 0 \end{pmatrix} V^T + \cdots + U \begin{pmatrix} 0 & & \\ & \ddots & \\ & & 0 & \\ & & & \delta_p \end{pmatrix} V^T$$

$$= \begin{bmatrix} u_1 & u_2 & \cdots & u_p \end{bmatrix} \begin{bmatrix} \delta_1 & & 0 \\ & 0 & \\ & & \ddots & \\ 0 & & & 0 \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_p^T \end{bmatrix} + \cdots$$

$$= u_1 \cdot \delta_1 \cdot v_1^T + u_2 \delta_2 v_2^T + \cdots + u_p \delta_p v_p^T$$

$$= \underbrace{\delta_1 u_1 v_1^T + \delta_2 u_2 v_2^T + \cdots + \delta_p u_p v_p^T}_{\text{rank 1 matrices}}$$

$$= \sum_{j=1}^{p} \delta_j u_j v_j^T$$

Same argument holds for $v_2, \ldots, v_p$.

Similar claim:

$$XX^T \text{ has an eigendecomposition } UD^2U^T.$$

Why do we care about the SVD? A few reasons:

1. Analysis using the SVD can lend insights and different perspectives into other ML methods (e.g. regression)
2. SVD can help with dimension reduction
   $\Longrightarrow$ has a close tie to PCA (more on this later, but a sneak preview: row i of UD describes the coordinates of the cenetered sample point x_i^C in principal component space)

Geometric Interpretation of the SVD

Suppose we know that for an arbitrary vector w, we know $Xw = z$; i.e. multiplying w by X maps it to another vector z. Recall that orthonormal matrices represent rotation operations & diagonal matrices correspond to scaling operations. So if we break X down into its SVD components, we can consider the linear transformation induced by applying the matrix X on an arbitrary vector w as three consecutive transformations:

$$Xw = UDV^Tw$$

1. First multiplication: multiplication by V^T rotates w onto a new vector; call this $V^Tw = \tilde{w}$.

$$Xw = UD\tilde{w}$$

2. Second multiplication: multiplication by the diagonal matrix D rescales axes by the singular values, but does not change the direction of the vector. Call the transformed vector after this multiplication $D\tilde{w} = \tilde{\tilde{w}}$

$$Xw = U\tilde{\tilde{w}}$$

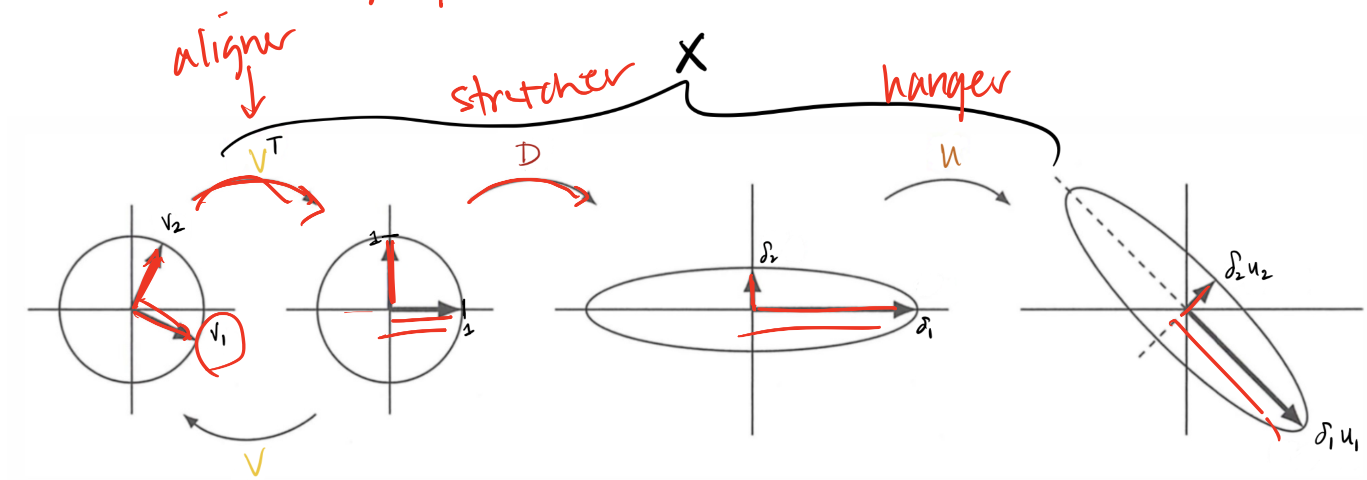
(3) Second multiplication: multiplication by U rotates $\tilde{\tilde{w}}$ again; call this final vector z.

$$Xw = U\tilde{\tilde{w}} = z$$

A nice way of visualizing this is to consider transforming the basis vectors of V, i.e. choose $w = v_i$ for any $i = 1, ..., p$.

EX:

$Xv_1 = \delta_1 u_1$ $\qquad$ $Xv_2 = \delta_2 u_2$



Intuition Building via SVD

Ex. OLS through SVD:

Consider the OLS setup

$$y = X\beta + \epsilon, \quad \overset{\varepsilon}{\cancel{epsilon}} \sim N(0, \sigma^2 I_n).$$

Derive the OLS estimator $\hat{\beta}$ in terms of the SVD of X. You can assume X has full rank.

Sol:

Normal Eq: $(X^T X)\hat{\beta} = X^T y$

$$\implies \hat{\beta} = (X^T X)^{-1} X^T y$$

$$= (VD^2V^T)^{-1}(UDV^T)^T y$$

$$= (VD^2V^T)^{-1}(VDU^T) y$$

$$= VD^{-2}V^T VDU^T y$$

$$= VD^{-2}DU^T y$$

$$= VD^{-1}U^T y$$

How does this relate to multicollinearity & instability in $\hat{\beta}$?

Notice: $\hat{\beta} = VD^{-1}U^T y = \sum_{j=1}^{p} \delta_j^{-1}(u_j^T y)v_j$ $\leftarrow$
so if δ_j is very close to zero, small changes in δ_j explode and result in very large changes in $\hat{\beta}$.

Question: In regression, how could regularization (e.g. ridge/lasso) stabilize the OLS estimate?

$Q(\beta) = \|Y - X\beta\|^2 + \lambda \|\beta\|^2$

Exercise: For a given penalty parameter λ, calculate the $\mathcal{L}^2$−regularized (i.e. ridge) estimate $\hat{\beta}_\lambda$ in terms of the SVD of X.

$$\hat{\beta}_\lambda = (X^T X + \lambda I)^{-1} X^T Y$$

$$= (VD^2V^T + \lambda I)^{-1} (UDV^T)^T Y$$

$$= (VD^2V^T + \lambda VV^T)^{-1} (UDV^T)^T Y$$

$$= (V(D^2 + \lambda I)V^T)^{-1} (UDV^T)^T Y$$

$$= (V^T)^{-1}(D^2 + \lambda I)^{-1} V^{-1} V D U^T Y$$

$$= V(D^2 + \lambda I)^{-1} D U^T Y$$

$$= V \begin{bmatrix} \delta_1^2+\lambda & & & 0 \\ & \delta_2^2+\lambda & & \\ & & \ddots & \\ 0 & & & \delta_p^2+\lambda \end{bmatrix}^{-1} \begin{bmatrix} \delta_1 & & \\ & \ddots & \\ & & \delta_p \end{bmatrix} U^T y$$

$$= V \begin{bmatrix} \frac{\delta_1}{\delta_1^2+\lambda} & & & 0 \\ & \frac{\delta_2}{\delta_2^2+\lambda} & & \\ & & \ddots & \\ 0 & & & \frac{\delta_p}{\delta_p^2+\lambda} \end{bmatrix} U^T y$$

the λ prevents me from having to divide by zero:

$$\frac{\delta_j}{\delta_j^2+\lambda} \to 0 \quad \text{as} \quad \delta_j \to 0$$

cf. $$\frac{1}{\delta_j} \to \infty \quad \text{as} \quad \delta_j \to 0$$

Dimension Reduction via SVD

Idea: If at some point k, the remaining singular values, $\{\delta_j\}_{j>k}$, are very small in magnitude, their relative contribution to the weighted sum is very small:

$$X = \delta_1 u_1 v_1^T + \cdots + \delta_k u_k v_k^T + \delta_{k+1} u_{k+1} v_{k+1}^T + \cdots + \delta_p u_p v_p^T$$

If they aren't adding much, **who needs them?**

Let's zero them out.

Doing so results in a *low rank/rank-k approximation* of the data matrix X:

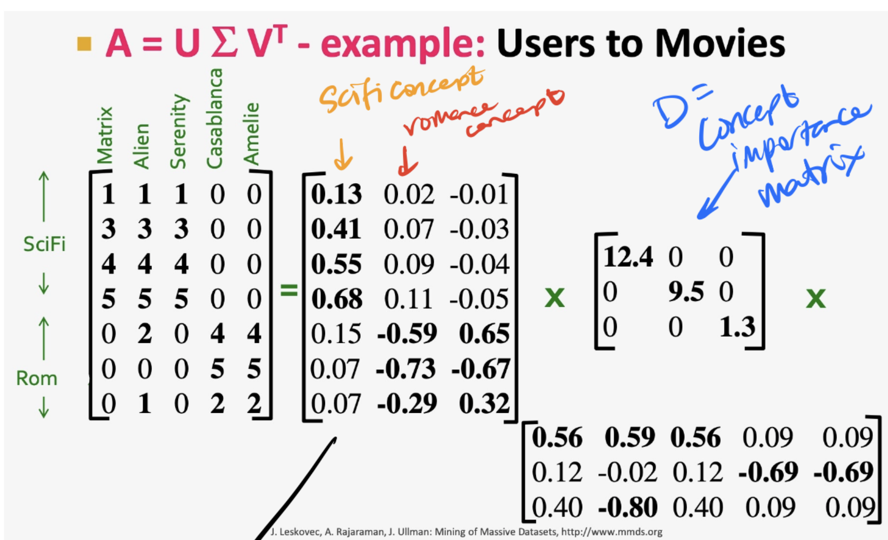
$$X \approx \delta_1 u_1 v_1^T + \cdots + \delta_k u_k v_k^T \quad (+0 + \cdots + 0)$$

$$= \underbrace{[u_1 \cdots u_k]}_{n\times k} \underbrace{\begin{pmatrix} \delta_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \delta_k \end{pmatrix}}_{k\times k} \underbrace{\begin{pmatrix} v_1^T \\ \vdots \\ v_k^T \end{pmatrix}}_{k\times p}$$

Call these truncated matrices U_k, D_k, V_k.

Instead of storing $n \times p$ values, now I only need to store $(n \times k) + (k \times k) + (k \times p) = (n + k + p)k$. This can matter a lot for high dimensional data, i.e. when p is very large!!

Example: Preview of Application to Recommender Systems



U= user-to-concept matrix

V= movie-to-concept matrix

Reading Assignment:

Chapter 11.3 of *Mining of Massive Datasets.*