Singular Value Decomposition (SVD)

Nate Islip

Eastern Washington University

2/12/2020

Table of Contents I

1 Slide Contents

2 Introduction & Application

3 SVD Notation and Properties
- Definitions:
- Computation:
- Approximation:
- Truncation:
- Properties and Manipulations

4 Examples: PCA and Eigenfaces
- PCA Introduction & Computation:
- Eigenfaces Example:

5 Truncation & Alignment

6 Computational Cost:

7 Bibliography

Introduction:

- **SVD** is one of the most important **matrix factorization** and is a foundational concept to other concepts (PCA, FFT).
- Use **SVD** to obtain **low-rank approximations** to matrices and to perform **pseudo-inverses of non-square matrices** to find the solution of a system of equations $Ax = b$.
- SVD is the basis for many techniques in **dimensional reduction**
  - PCA, KLT, EOF's, CCA to name a few

Applications:

- Fast Fourier Transform (FFT)
- Principal Component Analysis (PCA) in Statistics
- Dynamic Mode Decomposition (DMO) in Fluid Dynamics
  - Proper orthogonal decomposition (POD)
    - SVD Algorithm applied to PDE.
    - Important in studying complex spatio temporal systems

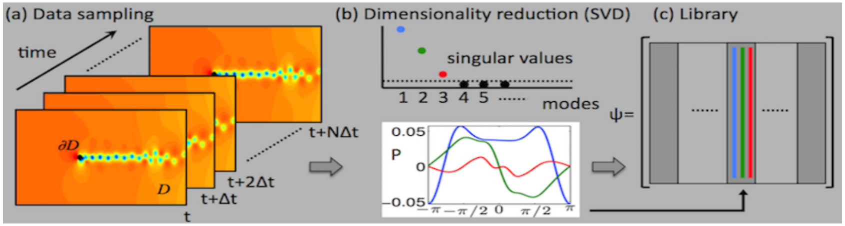

Figure 1: Model Reduction for flow around a cylinder

Notation: Full SVD

- Let X be a large data matrix where $X \in \mathbb{C}^{n\times m}$

$$\mathbf{X} = \begin{bmatrix} \vdots & \vdots & & \vdots \\ x_1 & x_2 & \dots & x_m \\ \vdots & \vdots & & \vdots \end{bmatrix} \tag{1}$$

  and the columns $x_k \in \mathbb{C}^n$ may be measurements from simulations or experiments (kth distinct set of measurements).

- The SVD is a *unique matrix decomposition* that exists for every complex valued matrix $\mathbf{X} \in \mathbb{C}^{n\times m}$.

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \tag{2}$$

  $\mathbf{U} \in \mathbb{C}^{n\times n}$ and $\mathbf{V} \in \mathbb{C}^{m\times m}$ are **unitary matrices** with orthonormal columns, and $\mathbf{\Sigma} \in \mathbb{R}^{n\times m}$ is a matrix with real, non-negative entries on the diagonal zeros off the diagonal.[1]

[1] $*$ denote the complex conjugate transpose

Notation: Economy SVD

- When $n \geq m$, the matrix $\mathbf{\Sigma}$ has at most m non-zero elements on the diagonal. Therefore, we can represent $\mathbf{X}$ as the **economy SVD**.

$$\mathbf{x} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* = \begin{bmatrix}\hat{\mathbf{U}} & \hat{\mathbf{U}}\end{bmatrix} \begin{bmatrix}\hat{\mathbf{\Sigma}} \\ \mathbf{0}\end{bmatrix} \mathbf{V}^* = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\mathbf{V}^* \tag{3}$$

- The columns of $\mathbf{U}$ are called *left singular vectors* of $\mathbf{X}$ and the columns of $\mathbf{V}$ are called *right singular vectors*.
- Diagonal elements of $\mathbf{\Sigma} \in \mathbb{C}^{m \times m}$ are called *singular values* and they are ordered from *largest to greatest*.

Notation: SVD Schematic

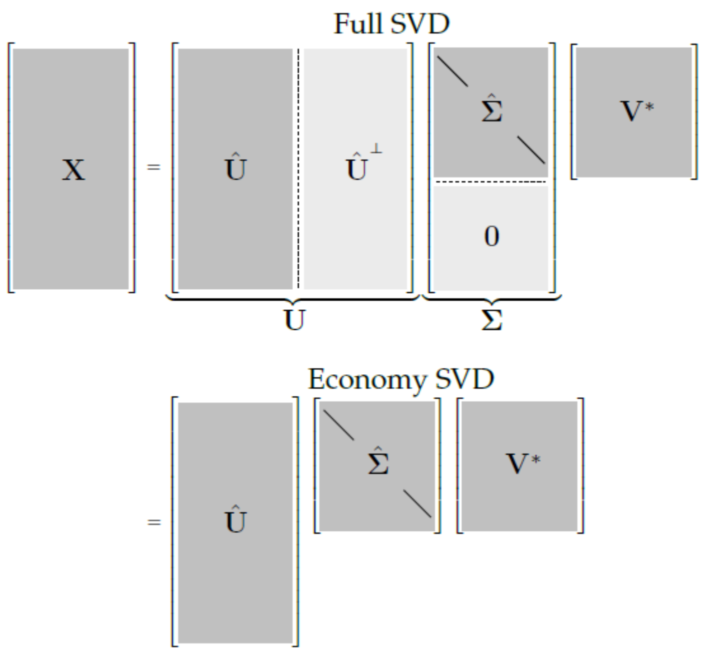

Figure 2: SVD and Economy SVD schematic

Coding Full and Economy SVD: Input

Python Implementation

```python
import numpy as np
import pandas as pd

X = np.random.rand(5,3) # random matrix
U,S,V = np.linalg.svd(X, full_matrices=True) # Full SVD
Uhat, Shat, Vhat = np.linalg.svd(X, full_matrices=True) # Economy SVD

dfU, dfS, dfV = pd.DataFrame(U), pd.DataFrame(S), pd.DataFrame(V)

print(dfU.to_latex(index=False, caption="the matrix $\mathbf{U}$"))
print(dfS.to_latex(index=False, caption="the matrix $\mathbf{S}$"))
print(dfV.to_latex(index=False, caption="the matrix $\mathbf{V}$"))
```

Matlab/Octave Implementation

```matlab
X = randn(5,3); % 5 x 3 random matrix
[U,S,V] = svd(X);
[Uhat, Shat, Vhat] = svd(X,'econ');
```

Coding Full SVD: Output

$$\mathbf{U} = \begin{bmatrix} 0.090948 & -0.87 & 0.18788 & 0.26935 & -0.35633 \\ 0.69151 & 0.078063 & 0.6658 & -0.15117 & 0.22267 \\ 0.0066421 & 0.15537 & 0.099151 & -0.5997 & -0.77868 \\ -0.26758 & 0.41429 & 0.48811 & 0.63234 & -0.34446 \\ -0.66476 & -0.20303 & 0.52281 & -0.38092 & 0.31375 \end{bmatrix}_{5\times5}$$

$$\mathbf{S} = \begin{bmatrix} 2.3183 & 0 & 0 \\ 0 & 1.6126 & 0 \\ 0 & 0 & 0.74531 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}_{5\times3}$$

$$\mathbf{V} = \begin{bmatrix} 0.44677 & 0.84636 & -0.28993 \\ 0.057677 & -0.35065 & -0.93473 \\ -0.89279 & 0.40089 & -0.20548 \end{bmatrix}_{3\times3}$$

Coding Economy SVD: Output

$$\hat{\mathbf{U}} = \begin{bmatrix} 0.090948 & -0.87 & 0.18788 \\ 0.69151 & 0.078063 & 0.6658 \\ 0.0066421 & 0.15537 & 0.099151 \\ -0.26758 & 0.41429 & 0.48811 \\ -0.66476 & -0.20303 & 0.52281 \end{bmatrix}_{5\times 3}$$

$$\hat{\mathbf{S}} = \begin{bmatrix} 2.3183 & 0 & 0 \\ 0 & 1.6126 & 0 \\ 0 & 0 & 0.74531 \end{bmatrix}_{3\times 3}$$

$$\hat{\mathbf{V}} = \begin{bmatrix} 0.44677 & 0.84636 & -0.28993 \\ 0.057677 & -0.35065 & -0.93473 \\ -0.89279 & 0.40089 & -0.20548 \end{bmatrix}_{3\times 3}$$

Notation: Matrix Approximation

Theorem (Eckart-Young 170)

The optimal rank-r approximation to $\mathbf{X}$, in a least squares sense, is given by the rank-r SVD truncation $\tilde{\mathbf{X}}$

$$\text{argmin}_{\tilde{X}, s.tr(\tilde{X})=r} ||\mathbf{X} - \tilde{\mathbf{X}}||_F = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}} \quad (4)$$

Here, $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{V}}$ denote the first r leading columns of $\mathbf{U}$ and $\mathbf{V}$, and $\tilde{\mathbf{\Sigma}}$ contains the leading $r \times r$ sub-block of $\mathbf{\Sigma}$. $||\cdot||_F$ is the Frobenius norm[2]. Because $\mathbf{\Sigma}$ is diagonal, the rank-r SVD approximation is given by the sum of r distinct rank-1 matrices.

[2] $||\cdot||$ is the matrix norm of an $m \times n$ matrix A (Euclidean norm)

Notation: Truncation

- Truncated SVD Basis is denoted as $\tilde{\mathbf{X}} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*$. The resulting **dyadic summation**

$$\tilde{\mathbf{X}} = \sum_{k=1}^{r} \sigma_k \mathbf{u}_k \mathbf{v}_k^* \tag{5}$$

- For a given rank, r, there is no better approximation for $\mathbf{X}$, in the ℓ_2 sense, then the truncated SVD approximation.

Remark:

Numerous examples of data sets contain **high dimensional measurements**, however, there are **dominant low dimensional patterns** in data, and $\tilde{\mathrm{U}}$ provides a transformation from *High* to *Low* dimensional pattern space. This allows for better analysis and visualization.

Properties and Manipulations: Interpretation as Dominant Correlations

- The SVD is closely related to an eigenvalue problem involving the correlation matrices $\mathbf{X}\mathbf{X}^*$ and $\mathbf{X}^*\mathbf{X}$.
- Plugging equation (3) into the row wise correlation matrix $\mathbf{X}\mathbf{X}^*$ and the column-wise correlation matrix $\mathbf{X}^*\mathbf{X}$ we find,

$$\mathbf{X}\mathbf{X}^* = U \begin{bmatrix} \hat{\Sigma}^2 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{U}^* \tag{6}$$

$$\mathbf{X}^*\mathbf{X} = \mathbf{V}\hat{\mathbf{\Sigma}}^{\mathbf{2}}\mathbf{V}^* = \mathbf{V}\hat{\mathbf{\Sigma}}^{\mathbf{2}} \tag{7}$$

- If $\mathbf{X}$ is **self-adjoint** (i.e. $\mathbf{X} = \mathbf{X}^*$), then the singular values of $\mathbf{X}$ are equal to the absolute value of the eigenvalues of $\mathbf{X}$.
- $\mathbf{\Sigma}$ are the square roots of the **eigenvalues** of the column-wise correlation matrix $\mathbf{X}\mathbf{X}^*$
- Columns of $\mathbf{V}$ are **eigenvectors** of $\mathbf{X}^*\mathbf{X}$
- $\mathbf{V}$ captures correlation in the rows of $\mathbf{X}$

Properties and Manipulations: Interpretation as Dominant Correlations

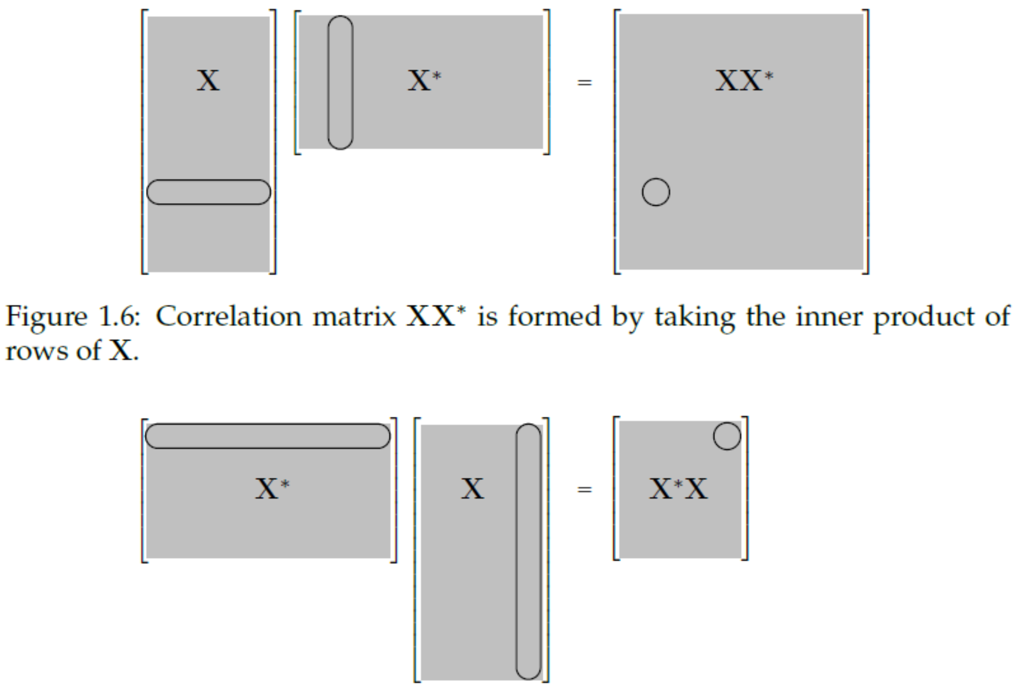

Figure 3: Schematic of Correlation matrices

PCA Set-up:

- PCA provides **hierarchical coordinate system to represent high-dimensional correlated data**.
- PCA pre-processes the data by **mean subtraction and setting the variance to unity before performing the SVD**.

$$\mathbf{X} = \begin{bmatrix} \dots & \dots & x_1 & \dots & \dots \\ \dots & \dots & x_1 & \dots & \dots \\ & & \vdots & & \\ \dots & \dots & x_n & \dots & \dots \end{bmatrix}$$

- PCA Steps:
  1. Compute mean row
  2. Subtract Mean $\mathbf{B} = \mathbf{X} - \overline{\mathbf{X}}$
  3. Compute Covariance Matrix of rows of $\mathbf{B}$
  4. $\mathbf{T} = \mathbf{B}\mathbf{V} \implies \mathbf{T} = \mathbf{U}\mathbf{\Sigma}$ where $B = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathbf{T}}$ [3]

[1] Here, we use SVD, we could also compute the eigen values and vectors of the covariance matrix.

Eigenfaces Example: Setup

- Eigenfaces are an example of SVD and PCA. We apply PCA to facial images to extract the most dominant correlation between images.

Figure 4: (Left) single image for each person (Right) and all images for each person.

- Use PCA to extract **most dominant correlations between images**.
- Result? set of **eigenfaces** that define a **new coordinate system**

Eigenfaces Example: Computation

- First 36 individuals used as **training data**, holding back two people as a **test set**.
  1. Re-shape each image into a large column vector
  2. Average face is computed and subtracted from each column vector.
  3. Mean-subtracted image vectors are stacked HZ as columns in $\mathbf{X}$.
  4. Take **SVD of mean-subtracted matrix** $\mathbf{X}$, giving the PCA
- Attempt to approximately represent an image that was not in the training data.
- How well does a rank-r SVD basis approximates the image using $\tilde{\mathbf{x}}_{test} = \tilde{\mathbf{U}}\tilde{\mathbf{U}}^*\mathbf{x_{test}}$

Eigenfaces Example: Schematic

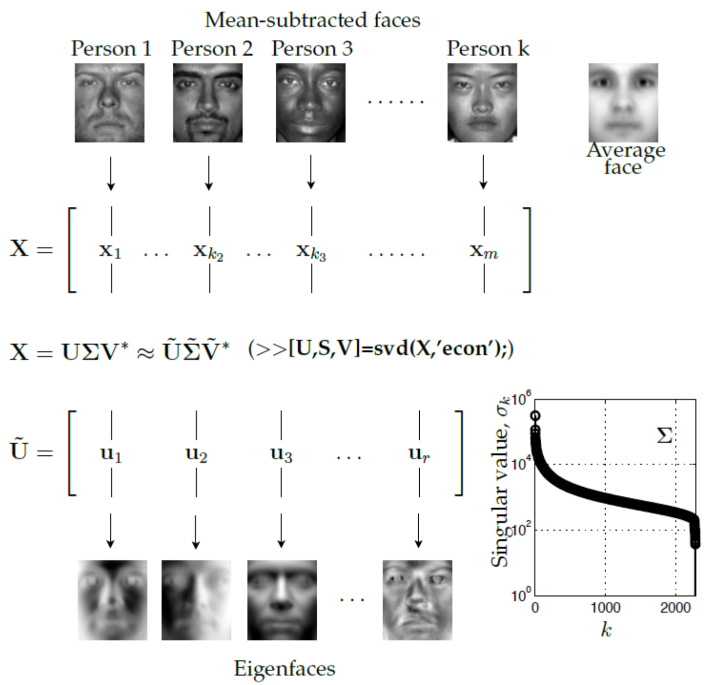

Figure 5: Schematic of procedure to obtain eigenfaces

Eigenfaces Example: Eigenface approximation

Figure 6: Face of various order r

Truncation & Alignment

- Deciding **how many singular values** is an important concept when discussing the SVD.
- There are two, commonly used, techniques used in truncation of singular values
  - **Method 1:** truncate SVD at a rank r that captures a pre-determined amount of the variance or energy in the original data (90% or 99% truncation).
  - **Method 2:** Identify ”knees” or ”elbows”. Helps distinguish *important patterns, from noise*.

Optimal Threshold

- So, how do we choose our optimal rank to truncate the **SVD**

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \qquad\qquad \mathbf{X} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\mathbf{T}}$$

- **Method 1:** plot the log of the singular values, $\log \sigma_j$, versus j and identify elbow (unfortunately, does not work well) .
- **Why?** Create a balance between model **simplicity** and **complexity**.
- **Method 2:** Technique designed to identify an optimal rank r to truncate (Gavish and Donoho, 2014). Consider,

$$\mathbf{X} = \mathbf{X}_{true} + \gamma \mathbf{X}_{noise} \tag{8}$$

Optimal Threshold: Continued

- Now, let us observe *two cases* where we can use **Method 2**, given by Gavish and Donoho [2].

  Case (1): $\mathbf{X}$ square, and γ is known.

$$\tau = \frac{4}{\sqrt{3}}\gamma\sqrt{n} \tag{9}$$

  Case (2): $\mathbf{X}$ rectangular, γ unknown

$$\tau = \lambda(\beta)\sigma_{med} \tag{10}$$

Computational Cost:

- Assuming, if A is $m \times n$ then $m >> n$ s.t. n^2 fits in memory on a single machine [3].
- Example: $m = 1$trillion and $n = 1,000$ (1 trillion movies each has a thousand features.
- Computing SVD requires $O(mn^2)$ work. Computing the top k singular values and vectors costs $O(mk^2)$ work.
  - Here we set k accordingly to how many singular values we would like.

Citations:

1. Brunton, S. L., Kutz, J. N. (2019). Data-driven science and engineering: Machine learning, dynamical systems, and control. Cambridge University Press.
2. The optimal hard threshold for singular values is $4/\sqrt{3}$, by M. Gavish and D. L. Donoho, IEEE Transactions on Information Theory, 2014
3. `https://stanford.edu/~rezab/classes/cme323/S17/notes/lecture17/cme323_lec17.pdf`