Ruimin Zhang
Feb 21th, 2020

Principle Component Analysis(PCA)

Abstract:

This is the homework 3 of AMATH 482 in the University of Washington. This assignment focuses on extracting data from several videos by Principle Component Analysis(PCA).

Sec. I. Introduction and Overview

Our aim in this assignment is to use a number of cameras (probes) to extract out data concerning the behavior of the spring-mass system and then to extract empirically the governing equations of motion.

Sec. II. Theoretical Background

2.1 SVD

A singular value decomposition (SVD) is a factorization of a matrix into a number of constitutive components all of which have a specific meaning in applications. The SVD, much as illustrated in the preceding paragraph, is essentially a transformation that stretches/compresses and rotates a given set of vectors. In particular, the following geometric principle will guide our forthcoming discussion: the image of a unit sphere under any m × n matrix is a hyper-ellipse.

Thus in total, there are n vectors that are transformed under A. We can interpret A as

$$A = \hat{U}\hat{\Sigma}V^* \tag{1}$$

where the matrix $\hat{\Sigma}$ is an $n \times n$ diagonal matrix with positive entries provided the matrix A is of full rank. The matrix $\hat{U}$ is an $m \times n$ matrix with orthonormal columns, and the matrix V is an $n \times n$ unitary matrix and V is unitary. This factorization is known as the reduced singular value decomposition, or reduced SVD, of the matrix A.

Theorem 1: A is the sum of r rank-one matrices.

$$A = \sum_{j=1}^{r} \sigma_j \mathbf{u}_j \mathbf{v}_j^* \tag{2}$$

Theorem 2: For any N so that $0 \leq N \leq r$, we can define the partial sum

$$A_N = \sum_{j=1}^{N} \sigma_j \mathbf{u}_j \mathbf{v}_j^* \tag{3}$$

And if N = min{m,n}, define $\sigma_{N+1} = 0$. Then

$$\|A - A_N\|_2 = \sigma_{N+1} \tag{4}$$

Likewise, if using the Frobenius norm, then

$$\|A - A_N\|_N = \sqrt{\sigma_{N+1}^2 + \sigma_{N+2}^2 + \cdots + \sigma_r^2} \tag{5}$$

This is to say, after r steps, the total energy in A is completely captured. Thus the SVD gives a type of least-square fitting algorithm, allowing us to project the matrix onto low-dimensional representations in a formal, algorithmic way. Herein lies the ultimate power of the method.

2.2 PCA

One of the key applications of the SVD is Principal Component Analysis (PCA).
If we are given two sets of measurements with zero means expressed in row vector form:

$$\boldsymbol{a} = [a1\, a2\ \cdots\ an] \text{ and } \boldsymbol{b} = [b1\, b2\ \cdots\ bn] \tag{6}$$

where the subscript denotes the sample number. The variances of a and b are given by

$$\sigma_a^2 = \frac{1}{n-1}\mathbf{a}\mathbf{a}^T \tag{7}$$

$$\sigma_b^2 = \frac{1}{n-1}\mathbf{b}\mathbf{b}^T \tag{8}$$

while the covariance between these two data sets is given by

$$\sigma_{ab}^2 = \frac{1}{n-1}\mathbf{a}\mathbf{b}^T \tag{9}$$

where the normalization constant of 1/(n − 1) is for an unbiased estimator.
Therefore, the appropriate covariance matrix for this case is then

$$C_X = \frac{1}{n-1}\mathbf{X}\mathbf{X}^T \tag{10}$$

The off-diagonal terms are the covariances between measurement types. Thus CX captures the correlations between all possible pairs of measurements. Redundancy is thus easily captured since if two data sets are identical (identically redundant), the off-diagonal term and diagonal term would be equal since $\sigma_{ab}^2 = \sigma_a^2 = \sigma_b^2$ if **a**=**b**. Thus large off-diagonal terms correspond to redundancy while small off-diagonal terms suggest that the two measured quantities are close to statistically independent and have low redundancy. It should also be noted that large diagonal terms, or those with large variances, typically represent what we might consider the dynamics of interest since the large variance suggests strong fluctuations in that variable. Thus the covariance matrix is the key component to understanding the entire data analysis.

The SVD can diagonalize any matrix by working in the appropriate pair of bases U and V. Thus by defining the transformed variable

$$\mathbf{Y} = \mathbf{U}^*\mathbf{X} \tag{11}$$

where U is the unitary transformation associated with the SVD: $\mathbf{X} = \mathbf{U\Sigma V}^*$. Just as in the eigenvalue/eigenvector formulation, we then compute the variance in **Y**:

$$\begin{aligned}\mathbf{C_Y} &= \frac{1}{n-1}\mathbf{YY}^T \\ &= \frac{1}{n-1}(\mathbf{U}^*\mathbf{X})(\mathbf{U}^*\mathbf{X})^T \\ &= \frac{1}{n-1}\mathbf{U}^*\mathbf{XX}^{\boldsymbol{T}}\boldsymbol{U} \\ &= \frac{1}{n-1}\mathbf{U}^*\mathbf{U\Sigma^2UU}^* \\ \mathbf{C_Y} &= \frac{1}{n-1}\mathbf{\Sigma^2}\end{aligned} \tag{12}$$

This gives the SVD method for producing the principal components. Overall, the SVD method is the more robust method and should be used. However, the connection between the two methods becomes apparent in these calculations.

Sec. III. Algorithm Implementation and Development

Sec. IV. Computational Results

Sec. V. Summary and Conclusions

To sum up, we use PCA on data of a spring-mass system recorded by several cameras. Test 1 is the ideal case. Test 2 is the noisy case. Test 3 is the horizontal displacement. Test 4 is the horizontal displacement and rotation.

Appendix A. MATLAB functions used and brief implementation explanation

1. `[row,col] = find(__)` returns the row and column subscripts of each nonzero element in array X using any of the input arguments in previous syntaxes.

2. `[U,S,V] = svd(A,'econ')` produces an economy-size decomposition of m-by-n matrix A:

3. `B = repmat(A,r1,...,rN)` specifies a list of scalars, `r1,..,rN`, that describes how copies of A are arranged in each dimension. When A has N dimensions, the size of B is `size(A).*[r1...rN]`. For example, `repmat([1 2; 3 4],2,3)` returns a 4-by-6 matrix

4. `[x,y] = ginput(n)` allows you to identify the coordinates of n points. To choose a point, move your cursor to the desired location and press either a mouse button or a key on the keyboard. Press the **Return** key to stop before all n points are selected. MATLAB® returns the coordinates of your selected points. If there are no current axes, calling `ginput` creates a set of Cartesian axes.

Appendix B.MATLAB codes

Reference

Kutz, Nathan. *Data Driven Modeling & Scientific Computing*