PH21 Assignment 6 report

Helen Xue

March 16, 2018

1 Method for finding Principal Components

Following the treatment in the Shiens paper, we can find the principal component for a set of generated 2D data as follows:

1. Generate the data: let x be uniform, and y be a linear function of x with some error. The error is generated using a Gaussian random variable with a mean of 1 and a standard deviation of 0.5. Here I generated two arrays of size 10.
2. Combine x and y into one matrix xy and normalize the matrix to have 0 mean.
3. Use numpy.cov to find the covariance matrix for xy
4. Use numpy.linalg.eig to find the eigenvalues and eigenvectors of the covariance matrix.
5. Use numpy.transpose to get the principal components from the eigenvectors.

The process is similar for a higher-dimensional data set (here there are 3 dependent variables):

1. Generate the data: let x be uniform, and y1,y2,y3 be linear function of x with some error. The error is generated using a Gaussian random variable with a mean of 1 and a standard deviation of 0.5. Here I generated two arrays of size 10.
2. Combine x and y1,y2,y3 into one matrix xy and normalize the matrix to have 0 mean.
3. Use numpy.cov to find the covariance matrix for xy
4. Use numpy.linalg.eig to find the eigenvalues and eigenvectors of the covariance matrix.
5. Use numpy.transpose to get the principal components from the eigenvectors.

2 Results

The smaller the eigenvalue, the less important the associated principal component: the eigenvalue lets us know the spread, and the lower the eigenvalue, the noisier that particular degree of freedom is.

2D case:
eigenvalue: 0.0009190906545804012; pc: [-0.83392113 0.55188364]
eigenvalue: 28.920427851861454; pc: [-0.55188364 -0.83392113]

4D case:
eigenvalue: 23.40471194722344; pc: [0.61791707 0.66880666 0.07743473 0.4060542]
eigenvalue: 0.065051557053434; pc: [-0.75523464 0.5628972 -0.20985111 0.26216388]
eigenvalue: 0.178672491231527; pc: [0.18695206 -0.20384939 -0.93324546 0.2292319]
eigenvalue: 0.259501559246700; pc: [-0.11334932 -0.44078319 0.28110368 0.8448922]

We see that the noise-free channel had a high corresponding eigenvalue.