Principal Components Analysis (PCA)

BIOE 210

Classification vs. Understanding

The SVM algorithm used training data to classify unknown samples.

We do not always understand how the SVM classifier makes decisions.

In biology we are often interested in **understanding** the differences between two classes, not assigning new samples to classes.

Understanding is difficult in high-dimensional systems.

Are high-dimensional data really high-dimensional?

Imagine you measured gene expression levels for multiple subtypes of a tumor.

There are often **hundreds** of genes that are differentially expressed.

Is it reasonable to think that the subtypes differ by hundreds of independent processes?

Usually there are a small number of differential functions that each involve lots of genes.

Dimensionality Reduction

Dimensionality reduction converts lots of individual variables into a smaller number of **composite** variables.

The components of the composite variables function together.

- Composite variables are linearly independent.
- Variables inside a composite variable are dependent.

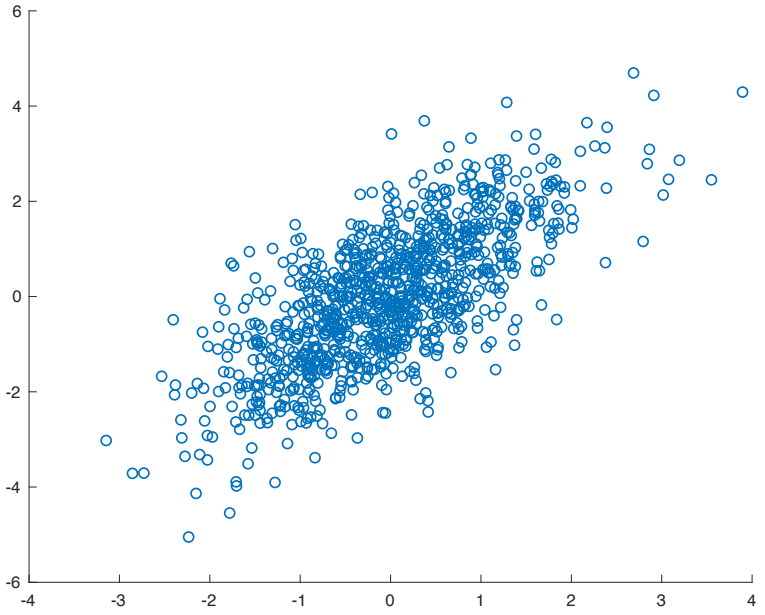
Our goal is to find the fewest composite variables that explain the maximum amount of the data.

Principal Component Analysis

Principal Component Analysis (PCA) chooses composite variables from a matrix of data.

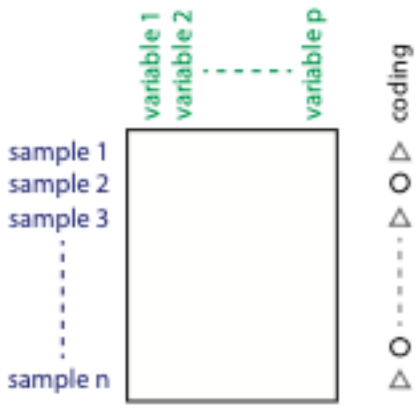
The composite variables (principal components) are always mutually orthogonal.

PCA also calculates the importance of each component, i.e. the amount of explained variance in the data.

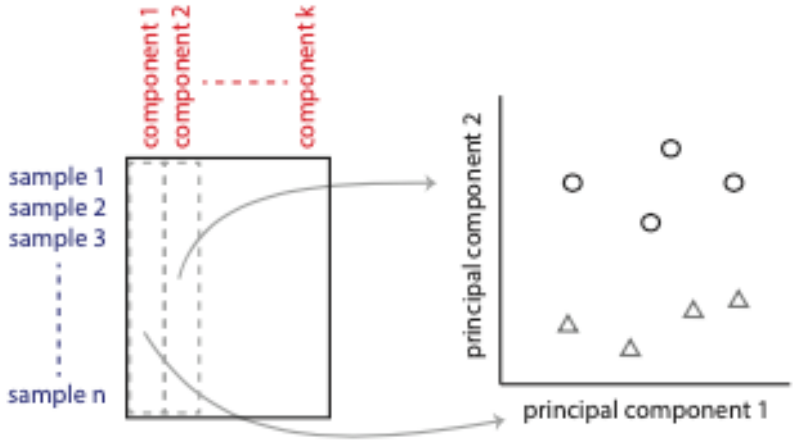


Visual Guide to PCA using Matlab

data



score

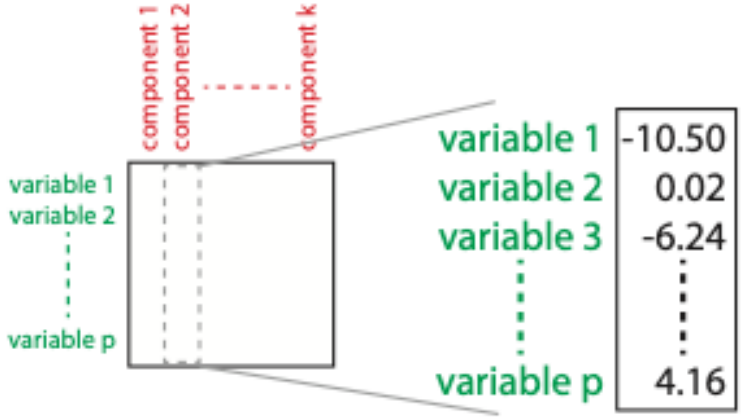


[coeff,score,~,~,explained] = pca(data)

explained

component 1	35.26
component 2	23.02
component 3	8.94
component k	0.02

coeff



How do we calculate Principal Components?

`[coeff,score,~,~,explained] = pca(X)`

$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ (the SVD)

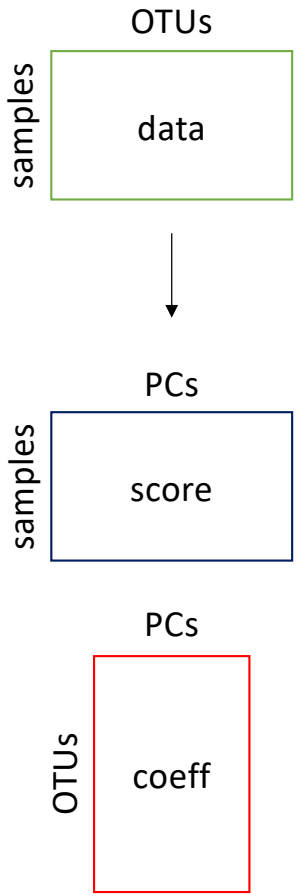
score = $\boldsymbol{U}\boldsymbol{\Sigma}$

explained = diag($\boldsymbol{\Sigma}$), normalized to 100%

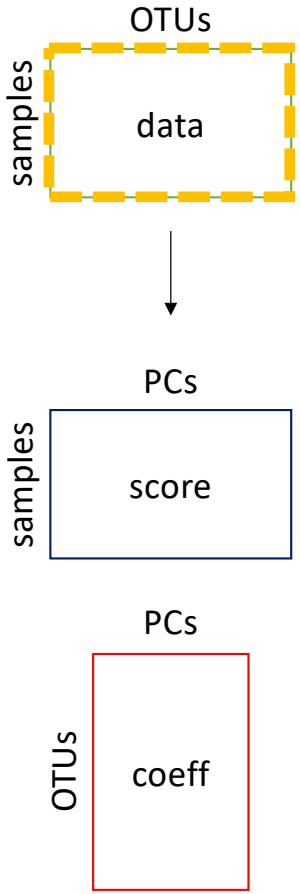
coeff = $\boldsymbol{V}$

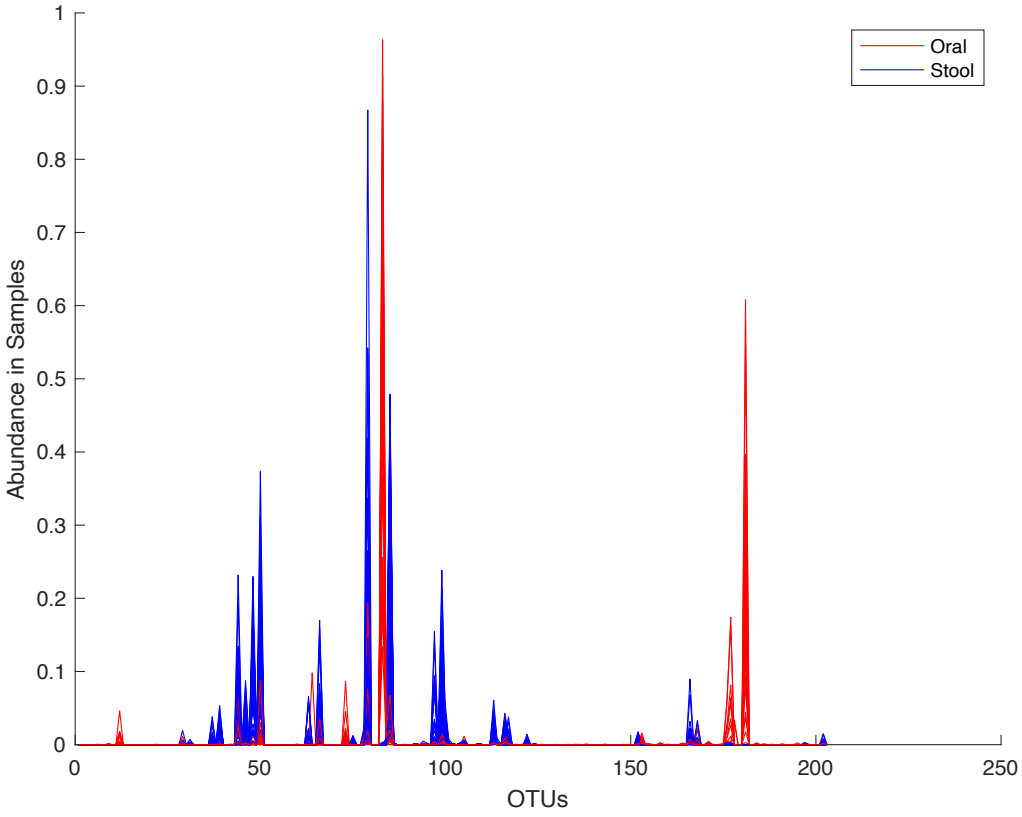
Example: Fluoride effects on the Microbiome

1. Study examined mice given no, low, or high levels of fluoride in drinking water for 12 weeks.
2. Microbiome samples taken from mouth and stool were sequenced to identify changes in microbial composition.
3. Variables are the abundances of species in the samples (called OTUs, or operational taxonomic units). ~10,000-30,000 OTUs are commonly seen in human microbiome samples.
4. Source: Yasuda K, et al. 2017. Fluoride depletes acidogenic taxa in oral but not gut microbial communities in mice. *mSystems* 2: e00047-17. https://doi.org/10.1128/mSystems.00047-17.

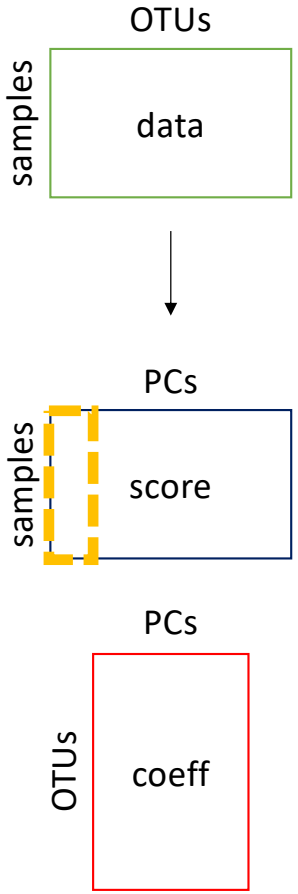


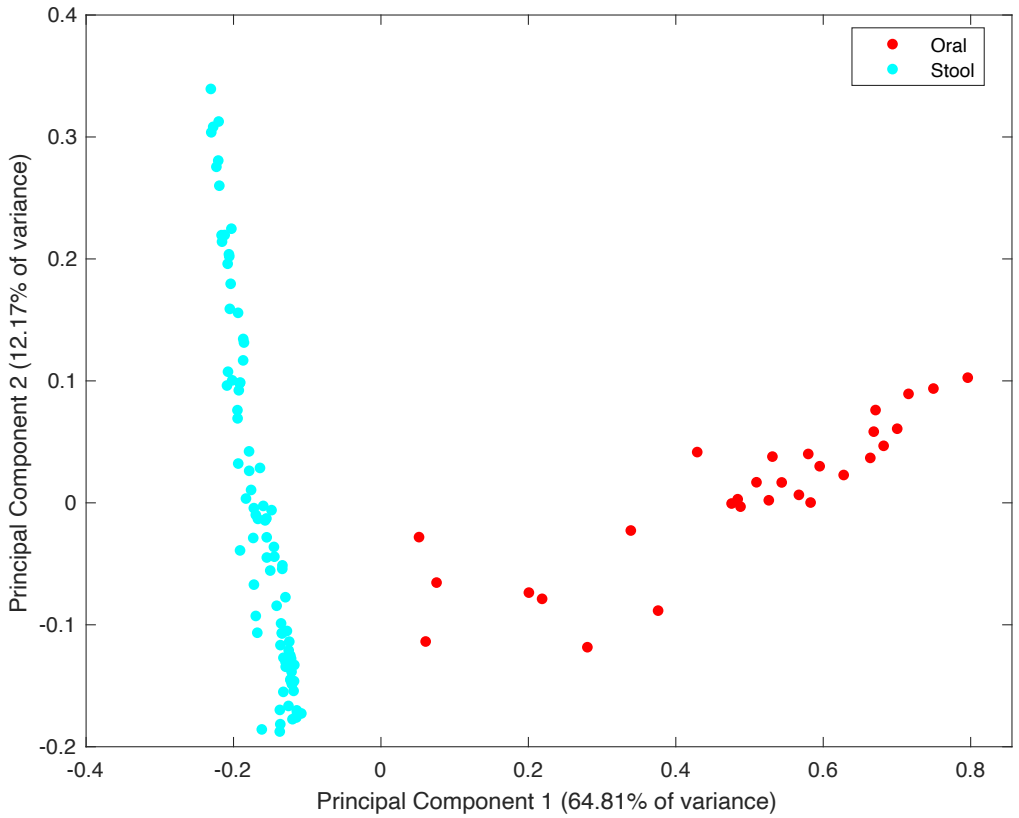
Many species (OTUs) vary between the oral and gut microbiomes.



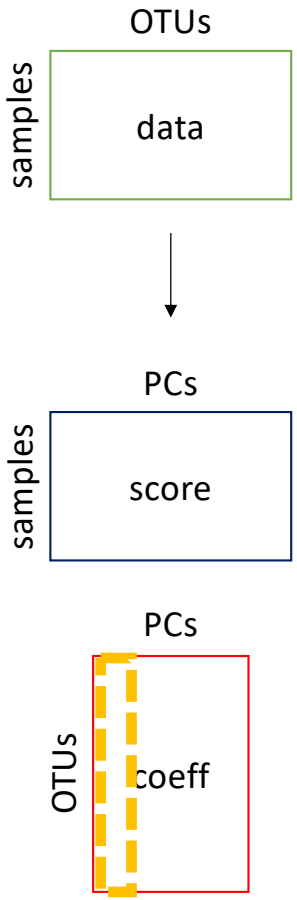


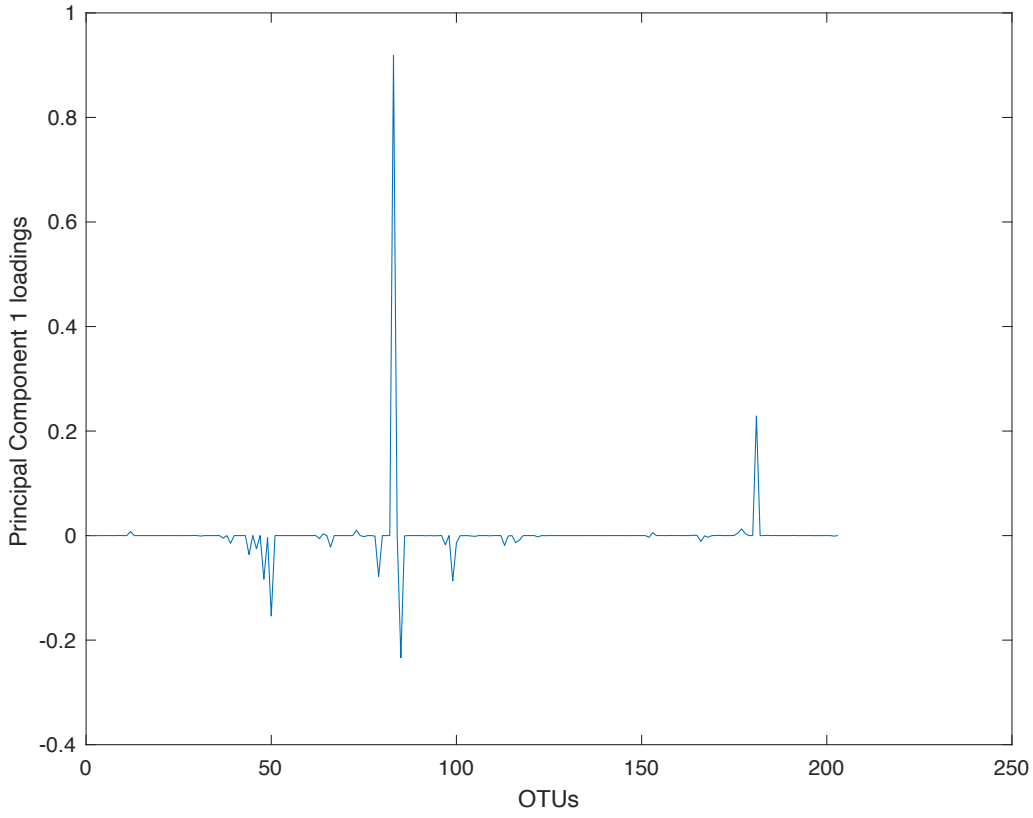
The microbiomes can be separated by the 1st P.C.





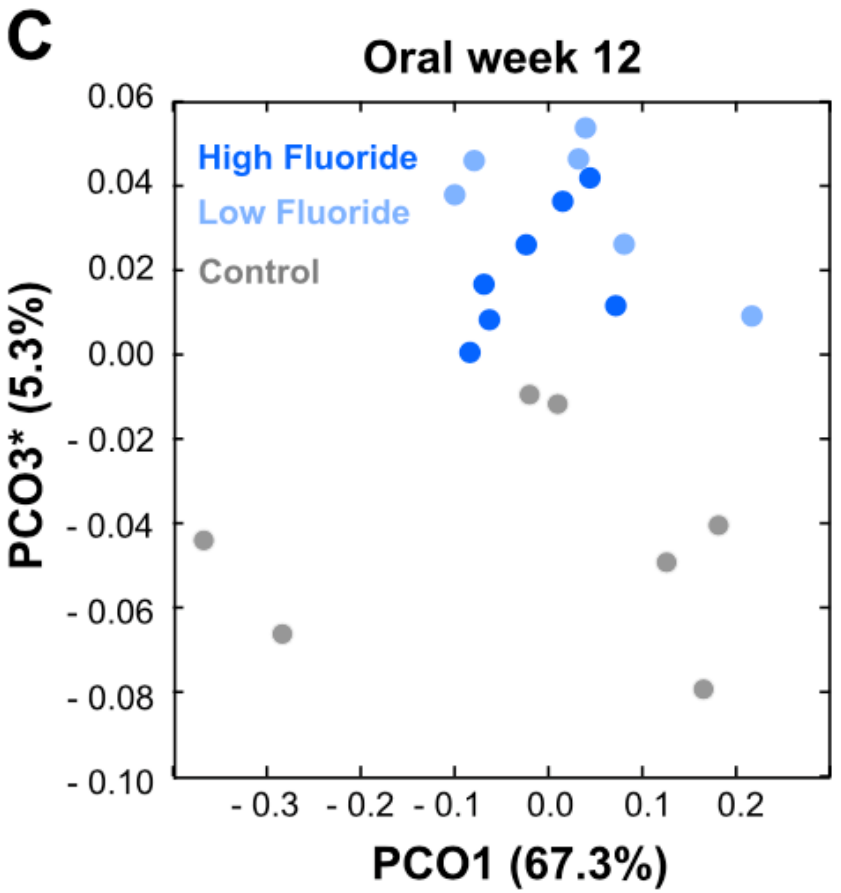
The loadings of PC1 identify differentially abundant species.





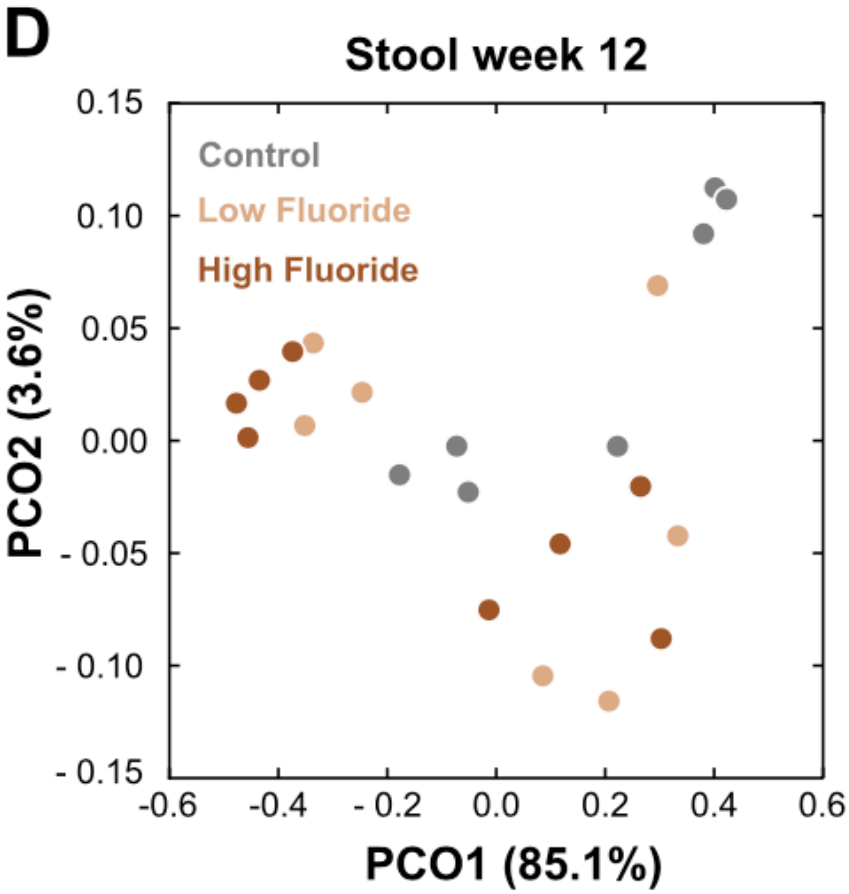
Result 2: Fluoride changes oral microbiome composition

1. PCs 1&3 explain 67.3 + 5.3 = 72.3% of the total variance in the dataset.
2. PC1 & PC2 do not separate the samples by fluoride levels.
3. PC3 does, however PC2 explains only 5.3% of the total variation.
4. The variables loaded in PC3 explain differences between fluoride levels, but the total effect is not large; the effects of PC1 must be removed first.
5. The authors confirmed several of the species loaded onto PC3 were affected by fluoride levels.



Result 3: Fluoride changes are limited to the oral cavity

1. Neither PC1 or PC2 separate the stool microbiome samples by fluoride levels.
2. Since these PCs explain 85.1 + 3.6 = 88.7% of the total variation, any effects of fluoride on the stool microbiome must be very small.



Summary

- The number of independent components is usually far smaller than the number of variables.
- PCA finds orthogonal combinations of variables of decreasing importance.
- Visualizing “lesser” components can identify signals that are lost in the full dataset.

Standard PCA Workflow

1. Make sure data are rows=observations and columns=variables.
2. Convert columns to Z-scores. (optional, but recommended)
3. Run [coeff,score,latent,tsquared,explained] = pca(X)
4. Using the %variance in "explained", choose k = 1, 2, or 3 components for visual analysis.
5. Plot score(:,1), ..., score(:,k) on a k-dimensional plot to look for clustering along the principal components.
6. If clustering occurs along principal component j, look at the loadings coeff(:,j) to determine which variables explain the clustering.

Principal Components Analysis in Matlab

`[coeff,score,latent,tsquared,explained] = pca(X)`

- **X**: input data
  - Matrix with *n* rows and *p* columns
  - Each row is an observation or sample
  - Each column is a predictor variable
  - All columns **must** be zero-centered

    `X(:,i) = X(:,i) – mean(X(:,i))`
  - pca will zero-center automatically, but any reconstructed output will not match X
  - Recommended that you scale the variance of columns to 1 by converting X to Z-scores

    `[...] = pca(zscore(X))`

Principal Components Analysis in Matlab

`[coeff,score,latent,tsquared,explained] = pca(X)`

- **coeff**: coefficients (loadings) for each PC
  - Square *pxp* matrix
  - Each column is a principal component
  - Each entry -- coeff(i,j) -- is the loading of variable *i* in principal component *j*
  - The matrix is orthonormal and each column is a right singular vector of **X**; **coeff** is the matrix **V** from the SVD of **X**.
  - The first column explains the most variance. The variance explained by each subsequent column decreases.

Principal Components Analysis in Matlab

`[coeff,score,latent,tsquared,explained] = pca(X)`

- **score**: Data (**X**) transformed into PC space
  - Rectangular *nxp* matrix
  - Each row corresponds to a row in the original data matrix **X**.
  - Each column corresponds to a principal component.
  - If row i in **X** was decomposed over the principal component vectors, the coefficients would be score(i,j):

```
X(i,:) = score(i,1)*coeff(:,1) + score(i,2)*coeff(:,2)
             + ... + score(i,p)*coeff(:,p)
```

Principal Components Analysis in Matlab

`[coeff,score,latent,tsquared,explained] = pca(X)`

- **latent**: Variance explained by each PC
- **explained**: % of total variance explained by each PC
  - Both **latent** and **explained** are vectors of length p (one entry for each PC
  - explained = latent/sum(latent) * 100
  - Variance explained is used when deciding how many PCs to keep.

Principal Components Analysis in Matlab

`[coeff,score,latent,tsquared,explained] = pca(X)`

- **tsquared**: Hotelling's T-squared statistic
  - Vector of length *n*, one entry for every observation in **X**.
  - Statistic measuring how far each observation is from the "center" of the entire dataset.
  - Useful for identifying outliers.