**Application of Principal Component Analysis in Assessment of Soil Heavy Metal Relationships**

Jinling Li

**Abstract**

This study investigated soil heavy metals, i.e. Cu, Ni, Pb, Cd, Zn and Cr in the Atlantic region and applied principal component analysis (PCA) to obtain the most significant component among all elements. The first principal component include Cu, Ni, Pb and Cd, and the second principal component includes Zn and Cr. The computational data modeling was implemented in Python and R. These results demonstrated that principal component analysis can be helpful for data modeling and engineering in the soil science area. Further studies including correlation analysis and clustering analysis could be conducted to strengthen the conclusions.

Key word: Principal Component Analysis, Heavy Metals, Data Modelling, Python

**Introduction**

Soil is basis for crop growth. Soil investigation is of great significance to agriculture and environment. Those soil properties, including heavy metal content in soils, indicate the status of soil quality and health. Questions like, “which heavy metal will have positive/negative impacts on soil quality?”, “are those soils in one site in poor/good condition?”, “which heavy metal affect soil condition most?” etc., require to be answered. Relationships between various soil heavy metals are to be explored as well. However, there is a gap between field studies and computational analysis. Agricultural scientists and experts often are not able to well interpret and understand data obtained from field studies. By contrast, data scientists and analysts do not usually find good data set to digest and justify their models.

In this study, a numerical model was set up and applied in soil data set. According to Wikipedia, “Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables. This transformation is defined in such a way that the first principal component has the largest possible variance, and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors are an uncorrelated orthogonal basis set”. In another word, PCA is a method of extracting important variables from a large set of variables available in a data set. It extracts low dimensional set of features from a high dimensional data set to capture information. PCA can be done by eigenvalue decomposition of a data covariance (or correlation) matrix or singular value decomposition of a data matrix. The matrix should be numeric and have standardized data.

Given the definition, PCA is able to analyze the relationships between factors and potentially can serve as a tool to explore the significance of the data set. The goal of this study is to explore factor relationships between heavy metal content in soils using PCA method implemented in Python.

**Materials and Methods**

The original samples were collected in the Atlantic region, then stored and transported to laboratory for further analysis. A total of 24 soil subsamples were combined and selected for heavy metal investigation. The contents of total heavy metals, i.e., Ni, Cu, Pb, Cd, Zn, and Cr, were determined in each subsample, respectively. To clarify the mechanics of mathematical analysis, only 24 subsamples were selected for heavy metal determination, and a numerical data set with 24 (samples) x 6 (heavy metals) observations was generated.

Principal Component Analysis (PCA) will be applied to find out which factor (heavy metal, i.e., Ni, Cu, Pb, Cd, Zn, and Cr) is the most important one regarding to soil properties. It is likely that some factors will have priorities that control soil quality over the others.

In the computer programming level, Python will be used to implement the PCA modelling. A pseudo code or command lines in R platform are also available for comparative analysis.

**Principal Component Analysis**

The purpose of PCA is to reduce dimensions and identify patterns from the dataset. The general steps of implementing PCA are as below:

1. Obtain the dataset consisting of n x p samples
2. Compute covariance matrix
3. Compute eigenvectors and eigenvalues
4. Sort eigenvectors by decreasing eigenvalues, and form a principal component matrix

**Data Set**

The dataset is 24 observations x 6 heavy metals. It is neither sparse nor large, therefore, PCA can treat this dataset very well.

Table 1: Heavy metal observations in the soil.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Heavy Metal Content (mg kg-1) | | | | | |
| Observation | Cu | Zn | Ni | Pb | Cr | Cd |
| 1 | 30.87 | 95.44 | 36.12 | 27.65 | 91.8 | 0.21 |
| 2 | 30.07 | 94.47 | 35.12 | 26.65 | 90.8 | 0.2 |
| 3 | 29.2 | 93.64 | 35.06 | 26.48 | 89.9 | 0.19 |
| 4 | 28.33 | 93.26 | 34.15 | 24.89 | 89.37 | 0.18 |
| 5 | 25.83 | 87.8 | 29.17 | 21.81 | 79.6 | 0.17 |
| 6 | 30.87 | 95.44 | 36.12 | 25.65 | 91.8 | 0.21 |
| 7 | 30.07 | 94.47 | 35.12 | 26.65 | 90.8 | 0.2 |
| 8 | 24.01 | 85.27 | 28.82 | 24.09 | 79.28 | 0.16 |
| 9 | 32.34 | 116.89 | 34.48 | 29.55 | 108 | 0.23 |
| 10 | 32.28 | 112.87 | 35.82 | 29.83 | 98 | 0.23 |
| 11 | 31.02 | 95.44 | 38.12 | 29.65 | 97 | 0.21 |
| 12 | 31.34 | 95.92 | 39.9 | 29.98 | 97.74 | 0.21 |
| 13 | 32.82 | 96.01 | 35.85 | 28.49 | 97.4 | 0.2 |
| 14 | 32.72 | 96.16 | 34.71 | 28.98 | 97.83 | 0.22 |
| 15 | 31.02 | 95.46 | 38.56 | 29.79 | 96.3 | 0.19 |
| 16 | 30.97 | 95.82 | 36.38 | 28.94 | 96.93 | 0.19 |
| 17 | 46.23 | 105.52 | 39.92 | 32.66 | 97.8 | 0.29 |
| 18 | 46.49 | 103.17 | 37.03 | 32.97 | 97.24 | 0.29 |
| 19 | 45.15 | 102.11 | 39.08 | 32.44 | 99 | 0.26 |
| 20 | 44.8 | 101.71 | 31.9 | 30.34 | 99.54 | 0.27 |
| 21 | 39.67 | 97.19 | 40.08 | 33.28 | 83.8 | 0.3 |
| 22 | 39.36 | 92.19 | 41.04 | 32.72 | 80.8 | 0.31 |
| 23 | 38.07 | 99.88 | 40.42 | 32.49 | 79.7 | 0.28 |
| 24 | 38.27 | 96.26 | 38.38 | 28.82 | 79.68 | 0.22 |

**Data Modelling**

PCA is achieved through several steps.

* Compute covariance matrix

The covariance is calculated by:

The covariance matrix V is calculated by:

Where  is the mean vector

* Compute eigenvectors and eigenvalues

Find eigenvalue: det(V−λI)=0

Find ith eigenvector: Vei = λiei

e: eigenvector

λ: eigenvalue

* Sort eigenvectors by decreasing eigenvalues, and form a principal component matrix

Sort eigenvalues in descending sequence, construct the matrix showings.

Through PCA matrix or loadings, a few principal components can be extracted. The first principal component captures the maximum variance in the dataset. It determines the direction of highest variability in the data. The second principal component captures the remaining variance in the dataset. The correlation between first and second component is zero, but their directions should be orthogonal. In general, for n × p dimensional data, min(n-1, p) principal component can be constructed.

These steps are the primary procedures to obtain a PCA matrix. However, we might not be satisfied with the obtained results and would like to explore more. Therefore, a full package of PCA is applied in Python and more detailed results are generated.

**Model Implementation**

As mentioned above, the model is implemented on Python 2.7. The python codes are as below:

**import** numpy **as** np  
**from** sklearn.decomposition **import** PCA  
**import** pandas **as** pd  
**import** matplotlib.pyplot **as** plt  
**from** sklearn.preprocessing **import** scale  
  
*#Load dataset*data = pd.read\_csv(**'work.csv'**)  
  
*#convert it to numpy arrays*X=data.values  
  
*#Scale the values*X = scale(X)  
  
**#Part I: Step by step to calculate PCA** *#compute Covariance Matrix*mean\_vec = np.mean(X, axis=0)  
cov\_mat = (X - mean\_vec).T.dot((X - mean\_vec)) / (X.shape[0]-1)  
*#print('Covariance Matrix \n%s' %cov\_mat)***print**(**'NumPy Covariance Matrix: \n%s'** %np.cov(X.T))  
  
*#compute eigenvectors and eigenvalues*cov\_mat1 = np.cov(X.T)  
eig\_vals, eig\_vecs = np.linalg.eig(cov\_mat1)  
**print**(**'Eigenvectors \n%s'** %eig\_vecs)  
**print**(**'\nEigenvalues \n%s'** %eig\_vals)  
  
*# Make a list of eigenvalue and eigenvector*eig\_pairs = [(np.abs(eig\_vals[i]), eig\_vecs[:,i]) **for** i **in** range(len(eig\_vals))]  
*# Sort eigenvalue from high to low*eig\_pairs.sort()  
eig\_pairs.reverse()  
**print**(**'Eigenvalues in descending order:'**)  
**for** i **in** eig\_pairs:  
 **print**(i[0])  
  
*#Construct matrix W*matrix\_w = np.hstack((eig\_pairs[0][1].reshape(6,1),  
 eig\_pairs[1][1].reshape(6,1)))  
**print**(**'Matrix W:\n%s'** %matrix\_w)  
Y = X.dot(matrix\_w)

**#Part II: Use package to calculate PCA**pca = PCA(n\_components=6)  
  
*#pca.fit(X)*pca.fit\_transform(X)  
**print**(pca)  
 *#The amount of variance that each PC explains*var= pca.explained\_variance\_ratio\_  
  
*#Cumulative Variance explains*var1=np.cumsum(np.round(pca.explained\_variance\_ratio\_, decimals=6)\*100)  
  
var2=pca.components\_  
**print** var1  
**print** var2  
plt.plot(var1)  
plt.show()

**Results and Discussion**

Python outputs are obtained and analyzed as below:

**Data Analysis**

* Compute covariance matrix

**Covariance Matrix:**

**[ 1.04347826 0.4941216 0.55359562 0.86174531 0.21426171 0.92168311]**

**[ 0.4941216 1.04347826 0.27587118 0.56228141 0.69435848 0.47816189]**

**[ 0.55359562 0.27587118 1.04347826 0.82287931 0.0732937 0.6476081 ]**

**[ 0.86174531 0.56228141 0.82287931 1.04347826 0.29280126 0.90970147]**

**[ 0.21426171 0.69435848 0.0732937 0.29280126 1.04347826 0.04562658]**

**[ 0.92168311 0.47816189 0.6476081 0.90970147 0.04562658 1.04347826]**

* Compute eigenvectors and eigenvalues

**Eigenvectors**

**[ 0.4642147 0.11948664 -0.44148623 0.41346637 -0.62236337 -0.13048738]**

**[ 0.35036065 -0.53453781 -0.13859744 -0.7229192 -0.20477624 -0.08810234]**

**[ 0.38884537 0.27965242 0.77338436 -0.10389764 -0.32387472 0.23828741]**

**[ 0.50202766 0.10150749 0.13973534 0.08469065 0.49430751 -0.68310033]**

**[ 0.19333317 -0.74231192 0.22578503 0.52921751 0.14153333 0.24599504]**

**[ 0.47025728 0.2458781 -0.34242145 -0.09162423 0.44884752 0.62553218]**

**Eigenvalues**

**[3.83497954 1.42623191 0.56136242 0.28694041 0.0962703 0.05508498]**

* Sort eigenvectors by decreasing eigenvalues, and form a principal component matrix

**Eigenvalues in descending order:**

**3.83497954277**

**1.42623190743**

**0.561362422287**

**0.286940413318**

**0.0962702999233**

**0.0550849794884**

**Matrix W:**

**PC1 PC2**

**Cu[ 0.4642147 0.11948664]**

**Zn[ 0.35036065 -0.53453781]**

**Ni[ 0.38884537 0.27965242]**

**Pb[ 0.50202766 0.10150749]**

**Cr[ 0.19333317 -0.74231192]**

**Cd[ 0.47025728 0.2458781 ]**

The eigenvalues in descending order generated two principal components PC1 and PC2. The PC matrix W grouped these two components. As is shown, Zn and Cr are similar, while the other four elements are grouped together.

If implemented using python package, PCA can also be calculated as below:

**PCA(copy=True, n\_components=6, whiten=False)**

**Cumulative Explained Variance**

**[61.2531 84.0332 92.9994 97.5825 99.1202 100.]**

The cumulative explained variances revealed the 6 principal components defined, but we do not need that many. As is seen, the first and second components explained 84% of total variances. It is reasonable to conclude that two principal components are extracted from the dataset.

**The vectors of Loadings**

**Cu Zn Ni Pb Cr Cd**

**PC1[ 0.4642147 0.35036065 0.38884537 0.50202766 0.19333317 0.47025728]**

**PC2[ 0.11948664 -0.53453781 0.27965242 0.10150749 -0.74231192 0.2458781 ]**

**PC3[-0.44148623 -0.13859744 0.77338436 0.13973534 0.22578503 -0.34242145]**

**PC4[ 0.41346637 -0.7229192 -0.10389764 0.08469065 0.52921751 -0.09162423]**

**PC5[ 0.62236337 0.20477624 0.32387472 -0.49430751 -0.14153333 -0.44884752]**

**PC6[ 0.13048738 0.08810234 -0.23828741 0.68310033 -0.24599504 -0.62553218]**

The vector of loadings revealed that the 6 heavy metals are included in two various group. PC1 includes Cu, Ni, Pb, and Cd, while PC2 includes Zn and Cr. This list is fully implemented and thus is complement to the results above using step by step method.

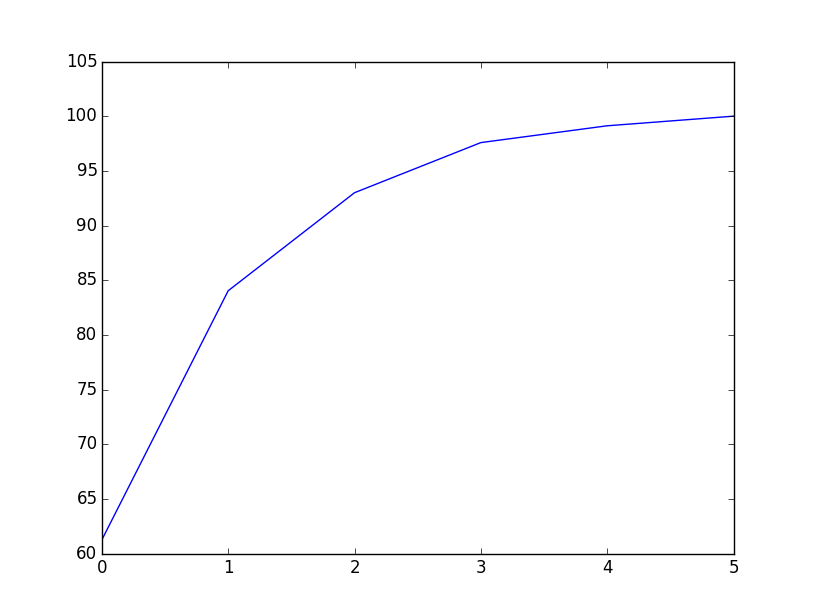
****

Figure 1. The contributions of six heavy metals of total variances

From this figure, we can see that the first and second components explained 84% of total variances. Therefore, it is safe to draw conclusion that there are two principal components existing in the dataset.

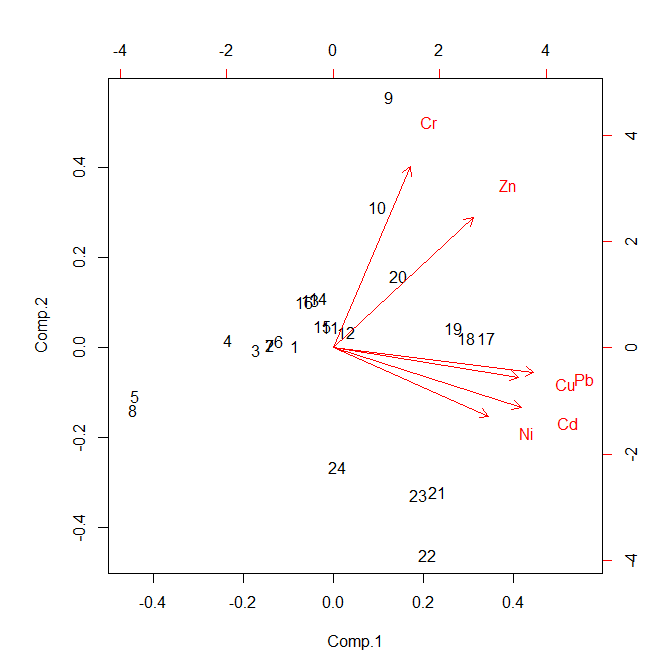


Figure 2. The principal components of six heavy metals

The figure above shows clearly the relationships among the 6 heavy metals. From the matrix W and vector loadings, we can confirm the two principal components, with the 1st principal components including Cu, Ni, Pb and Cd, and the 2nd principal components including Zn and Cr. These results comply with those from previous analysis. The two groups might be useful to disclose the relationships between soil heavy metals and provide evidence for explaining the potential sources of heavy metals.

**Conclusion**

This study investigated soil heavy metals, i.e. Cu, Ni, Pb, Cd, Zn and Cr in the Atlantic region and applied principal component analysis (PCA) to obtain the most significant component among all elements. The first principal component include Cu, Ni, Pb and Cd, and the second principal component includes Zn and Cr. The computational data modeling was implemented in Python and R. These results demonstrated that principal component analysis can be helpful for data modeling and engineering in the soil science area. Further studies including correlation analysis and clustering analysis could be conducted to strengthen the conclusions.

**Reference**

Wikipedia. Principal Component Analysis. <https://en.wikipedia.org/wiki/Principal_component_analysis>

Sebastian Raschka, Apr 13, 2014. Implementing a Principal Component Analysis (PCA) – in Python, step by step. <http://sebastianraschka.com/Articles/2014_pca_step_by_step.html>

Analytics Vidhya. Mar 21, 2016. Practical Guide to Principal Component Analysis (PCA) in R & Python. <https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principal-component-analysis-python/>

Jinling Li, Ming He, Wei Han, Yifan Gu, 2009. "Analysis and assessment on heavy metal sources in the coastal soils developed from alluvial deposits using multivariate statistical methods". Journal of Hazardous Material, vol.164, no.2, pp. 976-981

K. Thielen and V. Tien, "Python and Physical Modeling," in Computing in Science & Engineering, vol. 18, no. 3, pp. 8-10, May-June 2016

M. Partridge and M. Jabri, "Robust principal component analysis," Neural Networks for Signal Processing X. Proceedings of the 2000 IEEE Signal Processing Society Workshop (Cat. No.00TH8501), Sydney, NSW, 2000, pp. 289-298 vol.1

A. Papaioannou and S. Zafeiriou, "Principal Component Analysis With Complex Kernel: The Widely Linear Model," in IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 9, pp. 1719-1726, Sept. 2014