Principal Component Analysis

Maciej Staniszewski and Adam Foster

# Executive Summary

Principal Component Analysis (PCA) is a technique used for reducing the dimensionality of a dataset allowing for a more intuitive data exploration and analysis.

Table of Contents

[Executive Summary 1](#__RefHeading___Toc97_4112998524)

[Problem Description 2](#__RefHeading___Toc101_4112998524)

[Theoretical Background 2](#__RefHeading___Toc122_4112998524)

[Mathematical Foundations 2](#__RefHeading___Toc138_4112998524)

[Project Specification 3](#__RefHeading___Toc126_4112998524)

[Overview of the Data 3](#__RefHeading___Toc103_4112998524)

[Methodology Used 3](#__RefHeading___Toc105_4112998524)

[Results 4](#__RefHeading___Toc107_4112998524)

[Summary and Recommendations 4](#__RefHeading___Toc109_4112998524)

# Problem Description

In finance, similarly to other domains of science which are heavily reliant on concepts of data analysis and exploration, we are often facing large sets of data. These can be often difficult to analyse manually, even when plotted graphically. Noticing recurring patterns is highly impractical when the number of variables in the data exceeds reasonable bounds.

Interpreting data on aggregate level is especially difficult. Estimating joint distributions is an elementary step when determining the data generating process behind the data set. This in turn is a necessary step in inference-based thinking and causal discovery in data. Dimensionality curse is resticting us to three easily observable dimensions. When analysing complex data sets consisting of hundreds of variables, it is crucial that a dimensionality reducing operation is performed, enabling scientists to consider these joint distributions of every variable in a simplified form.

## Theoretical Background

A very well known method of representing variables for such analysis is the Principal Component Analysis, commonly known as PCA[[1]](#footnote-1). The main idea behind it is to create new variables from existing ones, that maximize the ratio of original observed variance to the variance captured by them. To do that, n-dimensional dataset needs to be reduced to k-dimensional one, where k is smaller than n.

The coordinates of that new system are the namesake principal components. They represent directions in n-dimensional space which retain the most of the original variance when casting the data points onto them. The first vector created this way is said to explain the biggest portion of original variance and is thus called the first principal component. Data points projected on this newly created coordinate system can be explored manually and jointly due to the reduced dimensionality.

## Mathematical Foundations

The approach can be broken down into five key steps. At it core, PCA incorporates eigendecomposition and maximisation on pre-processed data in achieving an alternative simpler function form of an equation expressing the relationship between several variables.

### Step 1 - Standardisation

Standardisation involves the normalisation using mean and standard deviation and makes varying domains comparable. This prevents variables with larger domains dwarfing the impact of smaller domains, reducing bias.

### Step 2 - Covariance Matrix

In searching for all possible relationships between the variables and their joint variance, a covariance matrix is created. For variables, an covariance matrix is formed.

We can observe a couple of interesting facts about the covariance matrix:

* Diagonal elements of the covariance matrix describe the variance of individual variables. This is because they contain information about the variability of the pair of values which is essentially the same as the variability of for some variable .
* Another observation is that the matrix should be symmetrical about the diagonal. This means that elements vary in the same exact way as elements . Note: that python libraries will raise a warning if the covariance matrix does not have that property.

### Step 3 - Eigendecomposition

Principal components are new variables that are constructed as linear combinations of the initial variables. These combinations are done in such a way that the principal components are uncorrelated and most of the information within the initial variables (and therefore the variance) is compressed into the first components.

Across all principal components, the same information is maintained. The advantage is that the minimum amount of information is lost when removing the least important components, hence a large proportion of the data can be represented by few components - they pick up the maximal amount of variance.

The first principal component is calculated to maximise the average of the squared distances from projected points to the origin. The second one is calculated the same way, provided it is uncorrelated with and therefore orthogonal to the first component. This can be continued for the number of dimensions of the data and each component has an associated eigenvector-eigenvalue pair. Eigenvectors describe the direction of the axes where there is the most variance and eigenvalues show the amount of variance carried. Ranking eigenvectors by eigenvalues in descending order shows the principal components in order of significance. The eigenvalue relative to the sum of eigenvalues denotes the percentage of variance covered by the associated principal component.

#### Methodology

A point gets projected onto a unit vector , a new point is created whose magnitude is calculated by the inner product between and : .

can be thought of as the amount of information preserved and is maximal when is parallel to and minimal when is orthogonal to . PCA is then an optimisation problem whereby it seeks to maximise information preserved subject to the unit vector usage:

subject to

which simplifies to

subject to where (covariance matrix)

This can be expressed as a Lagrangean and solved:

Therefore, and represent eigenvectors and eigenvalues, respectively. The best direction to pick is coupled with the eigenvector with the largest eigenvalue . can then be selected such that with and so on.

### Step 4 - Feature Vector

At this stage, a process of elimination occurs based on principal component significance and the resulting components form the feature vector.

### Step 5 - Recast the Data along the PC Axes

Eigenvectors of the feature vector principal components are used to reorient the data from the original axes to the ones represented by the principal components.

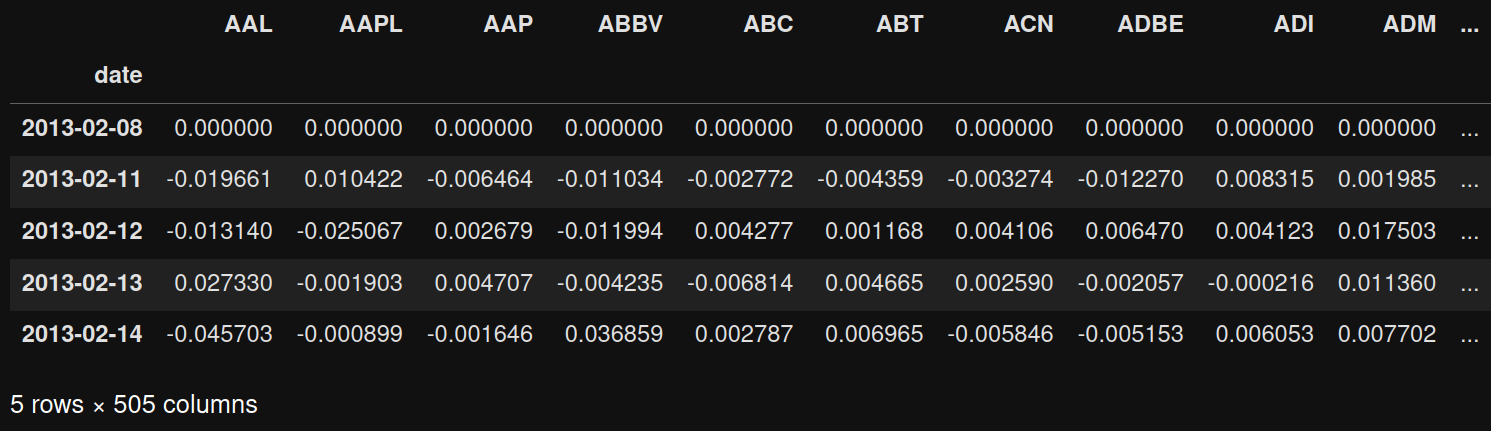
This data can be visualised in a more intuitive manner across fewer dimensions.

## Project Specification

In our project we decided to explore a well-known financial equity index – *Standard & Poor’s 500* or *S&P500[[2]](#footnote-2)*. Its price[[3]](#footnote-3) is determined by the joint market performance of five hundred largest   
US-based companies traded publicly. We wanted to find the companies which drive the price variability of the index and determine if the performance of *S&P500* can be expressed as a linear combination of the underlying assets. We relied on PCA to decompose daily returns of the index into principal components and obtained main drivers from the coordinates of the first component. We arrived at a quantitative solution which helps analysts to understand price moves of complex financial products, such as an equity index.

# Overview of the Data

We used a dataset which includes all *S&P500* single-name stock daily close prices in a time period between 2013 and 2018[[4]](#footnote-4). To obtain the index daily close prices in that period we used the aforementioned Python yfinance library. We combined these two datasets into one pandas dataframe and cleaned the data: transformed prices into returns, pivoted the data for compatiiblity with other libraries, ensured coherency between all starting dates and ending dates. We obtained the following data set:

Figure 1: Five rows of preprocessed dataset including daily returns of all S&P500 single-name stocks and the index price betwen 2013 and 2018.

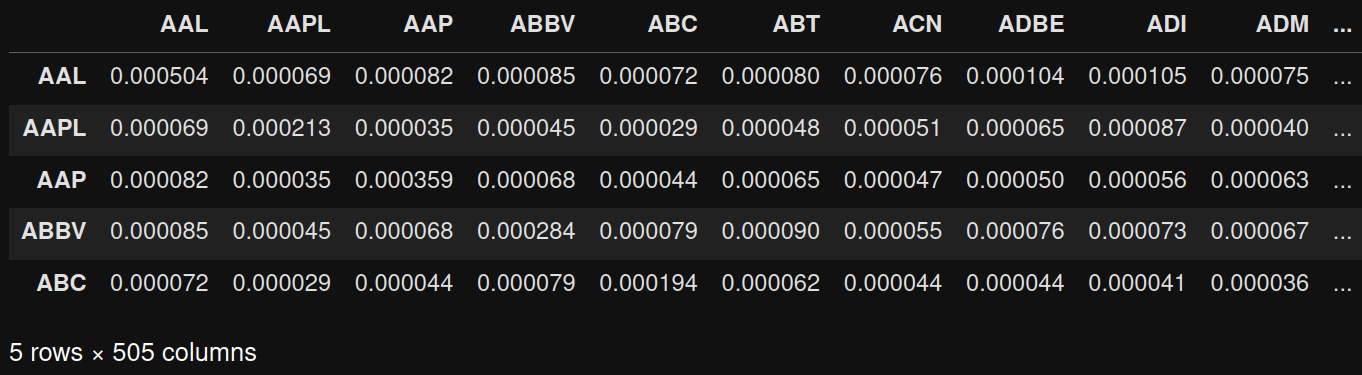
# Methodology Used

We used Python programming language inside a Jupyter Notebook[[5]](#footnote-5) to interactively display both code and its results. The following packages were included:

* numpy[[6]](#footnote-6) for basic mathematical operations
* pandas[[7]](#footnote-7) for dataset manipulations
* ipykernel Jupyter kernel for Python code execution
* matplotlib[[8]](#footnote-8) for data visualisation with plots
* yfinance for financial data from Yahoo Finance[[9]](#footnote-9)

We were operating on the dataset described in detail in section Overview of the Data. We used cov() Python method to compute the covariance matrix which sample is presented below:

After such preprocessing we were ready to start considering eigenvectors and eigenvalues. The results of our analysis are present in the next section - Results.

Figure 2: Five rows of covariance matrix of daily returns of S&P500 single-name stocks and the index price betwen 2013 and 2018.

# Results

# Summary and Recommendations

1. [Principal Component Analysis - Wikipedia](https://en.wikipedia.org/wiki/Principal_component_analysis) [↑](#footnote-ref-1)
2. [Standard and Poor's 500 - Wikipedia](https://en.wikipedia.org/wiki/S%26P_500) [↑](#footnote-ref-2)
3. [S&P500 Index Price - S&P Global](https://www.spglobal.com/spdji/en/indices/equity/sp-500/#overview) [↑](#footnote-ref-3)
4. [S&P500 Single Name Prices - Kaggle Dataset](https://www.kaggle.com/datasets/camnugent/sandp500) [↑](#footnote-ref-4)
5. [Jupyter Notebook - Main Page](https://jupyter.org/) [↑](#footnote-ref-5)
6. [NumPy - Python](https://numpy.org/) [↑](#footnote-ref-6)
7. [Pandas - Python Data Analysis Library](https://pandas.pydata.org/) [↑](#footnote-ref-7)
8. [Matplotlib - Visualizing with Python](https://matplotlib.org/) [↑](#footnote-ref-8)
9. [Yahoo Finance - Top ETFs](https://finance.yahoo.com/etfs) [↑](#footnote-ref-9)