# PHIL 7010: Formal Methods for AI, Data, and Algorithms

## Week 5
## Principle Component Analysis

Boris Babic,
HKU 100 Associate Professor of Data Science, Law and Philosophy

# Today

**Learning goals**

- The Basics of Principal Component Analysis (PCA)
- Exploring the Process and Mechanics of PCA
- Identifying and Interpreting Principal Components
- PCA for Dimensionality Reduction

# High-Dimensional Data

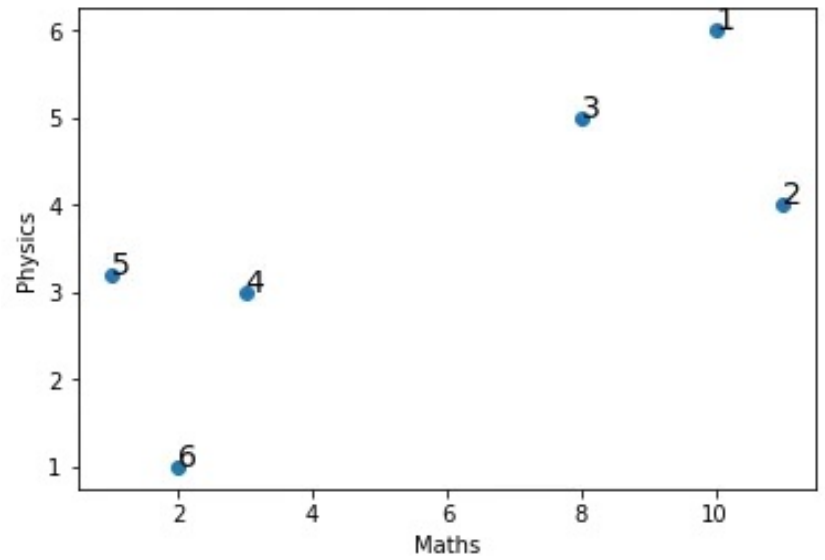
- A common issue for data scientists when creating an algorithm is having too many variables. Naturally, you would think that adding more information would only make your model better, but with every feature you add comes another dimension.
- In the real world, data can have an infinite amount of dimensions
- To deal with these complicated datasets, Principal Component Analysis is an ideal method to reduce the data dimensionality.

# Example

| | Student1 | Student2 | Student3 | Student4 | Student5 | Student6 |
|---|---|---|---|---|---|---|
| **Maths** | 10.0 | 11.0 | 8.0 | 3.0 | 1.0 | 2.0 |
| **Physics** | 6.0 | 4.0 | 5.0 | 3.0 | 3.2 | 1.0 |

- Imagine you have a dataset that records the marks of six students in two subjects: Mathematics and Physics.
- Each student's performance is represented as a point on a two-dimensional graph, with Math scores on the x-axis and Physics scores on the y-axis.

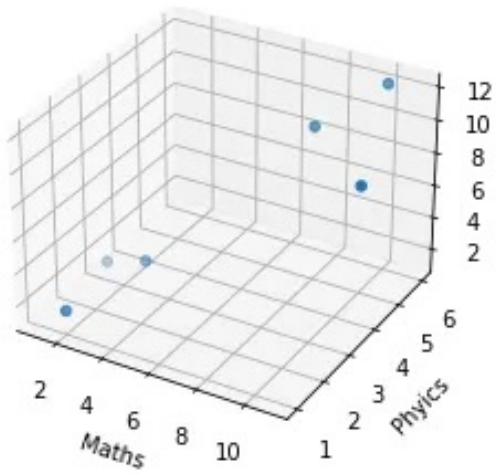
| | Student1 | Student2 | Student3 | Student4 | Student5 | Student6 |
|---|---|---|---|---|---|---|
| **Maths** | 10.0 | 11.0 | 8.0 | 3.0 | 1.0 | 2.0 |
| **Physics** | 6.0 | 4.0 | 5.0 | 3.0 | 3.2 | 1.0 |




- To compare and understand the student's performance in the two subjects, it would be quite easy for us to plot a 2-D graph.
- We can see that Students 1, 2, and 3 cluster on the right side while Students 4,5,6 on the lower left side of the graph.

Now, suppose we add marks for another subject, and then we can plot the data on a 3D graph.

| | Student1 | Student2 | Student3 | Student4 | Student5 | Student6 |
|---|---|---|---|---|---|---|
| **Maths** | 10.0 | 11.0 | 8.0 | 3.0 | 1.0 | 2.0 |
| **Physics** | 6.0 | 4.0 | 5.0 | 3.0 | 3.2 | 1.0 |
| **History** | 12.0 | 9.0 | 10.0 | 2.5 | 1.3 | 2.0 |




**But what if we want to measure students's scores for one more subject?** Here, comes the PCA.

# Principal Component Analysis (PCA)

- **Principal Component Analysis (PCA)** is a dimensionality reduction technique. It's a powerful **unsupervised** method for visualizing and simplifying high-dimensional data while preserving essential information.
- PCA extracting hidden structure from high dimensional datasets by transforming the original data into a new coordinate system where the axes are called **principal components**.
- In simpler words, PCA works by finding the directions (principal components) along which the data varies the most. These principal components are **orthogonal** to each other, and the first principal component captures the most variance in the data, the second captures the second-most variance, and so on.
- Applications:
  - fields such as face recognition and image compression
  - finding patterns in data of high dimension.

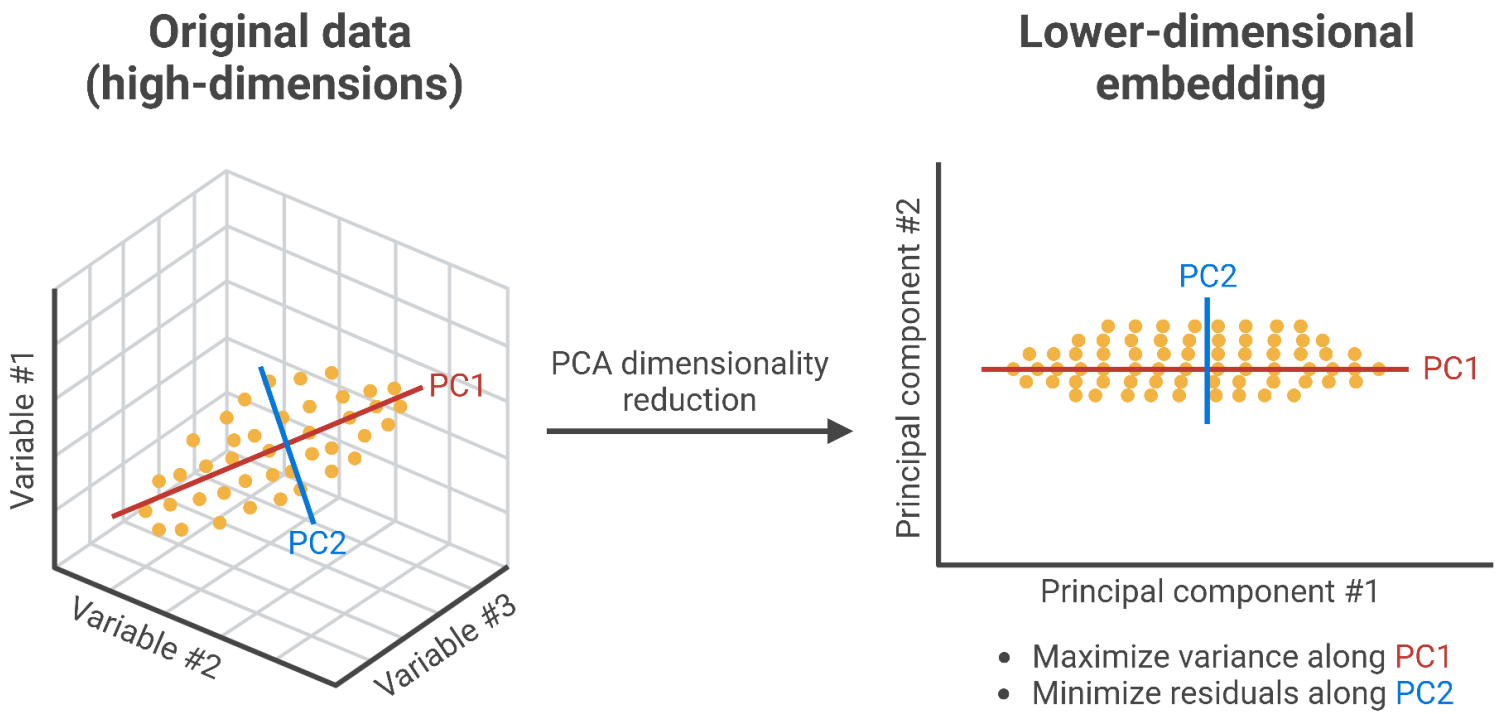
# Principal Component Analysis (PCA)

In PCA analysis, each principal component is a linear combination of the original variables, with weights (or coefficients) corresponding to the importance or contribution of each variable to that component. These weights are determined based on the variance that each variable contributes to the dataset as a whole.

The overall goal is to maximize the variance captured by each component, under the constraint that each component is orthogonal (i.e., statistically independent) to the others. Thus, **identifying a PC can be viewed as a sophisticated way of taking a weighted average of variables** in a way that captures the most significant patterns of variation in the data.

# High-Dimensional Data

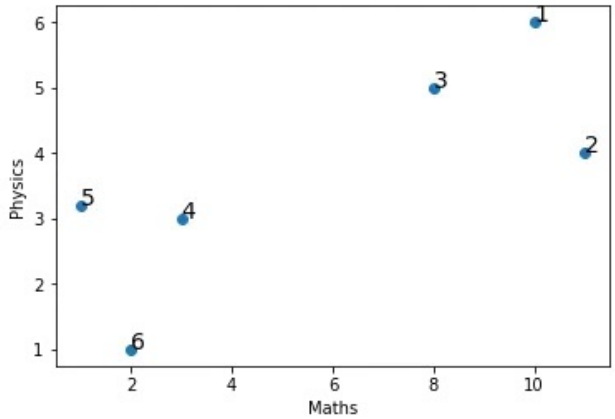
## Principal Component Analysis (PCA) Transformation



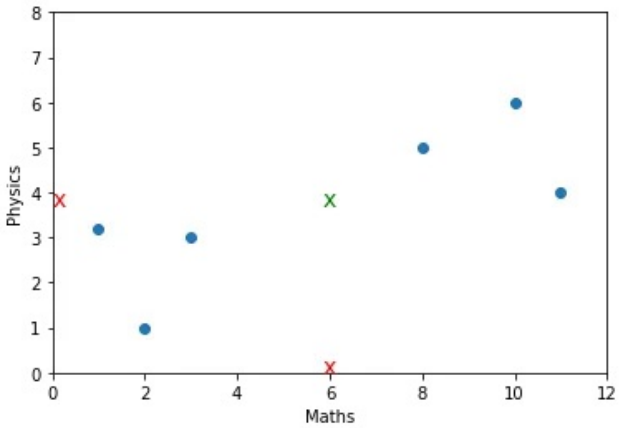

- Maximize variance along PC1
- Minimize residuals along PC2

# PCA - How it works

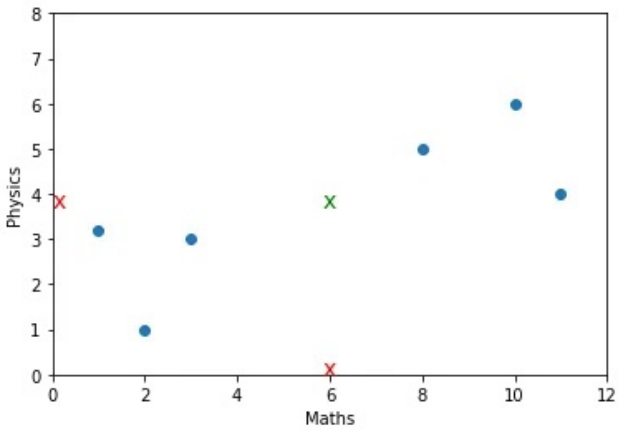
To understand the underlying process of PCA, let's take the example of data with two variables; Students' scores recorded in 2 subjects.

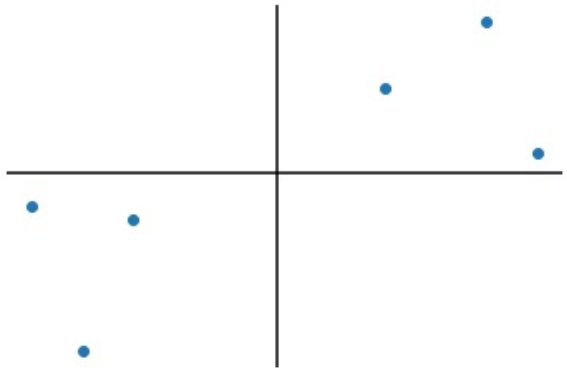



We start with calculating the average measurement for Maths and for Physics. With these average values, the center of the data is calculated (green cross in the figure below).



Now, we'll shift the data so that the center (the green cross) is on the top of the origin (0,0) in the graph. Note that shifting the data won't disturb the relative positioning of the sample points in the data.



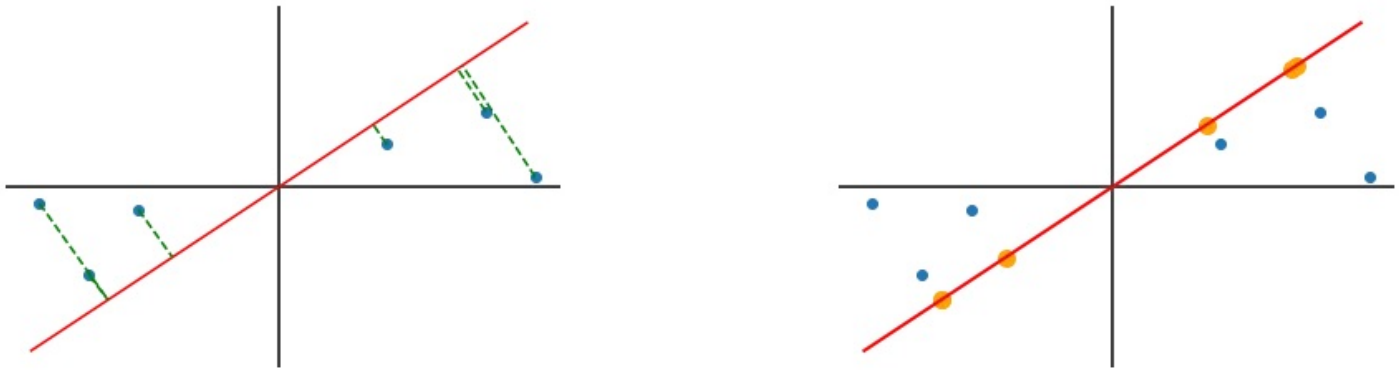



Now that the data is centered on the origin, we can try to fit a line to it. To do this, we start by drawing a random line that goes through the origin and rotate the line to find the best fit.

To quantify, how well the line fits, PCA projects the data points on the line and then it does **either one of the following**:

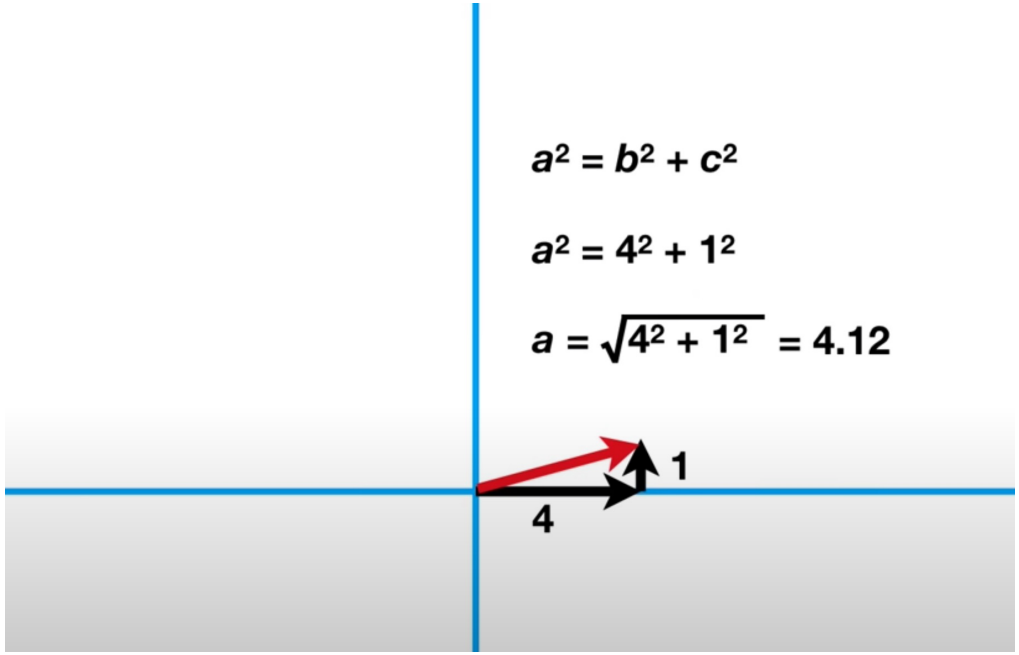
- Measure the distance from the data to the line and try to find the line that minimizes those distances. (green dotted line in the figure below)
- Project all data to the line and find the line that maximizes the squared distance from the projected points (orange dots) to the origin.

The second approach is more commonly used. However, the two methods are mathematically equivalent. This is because the total variance in the data can be partitioned into the variance explained by the principal components and the residual variance (the orthogonal distances to the principal component lines). Thus, maximizing the former inherently minimizes the latter.



**We'll call the best-fit line the 1st principal component or the PC1 for our data**

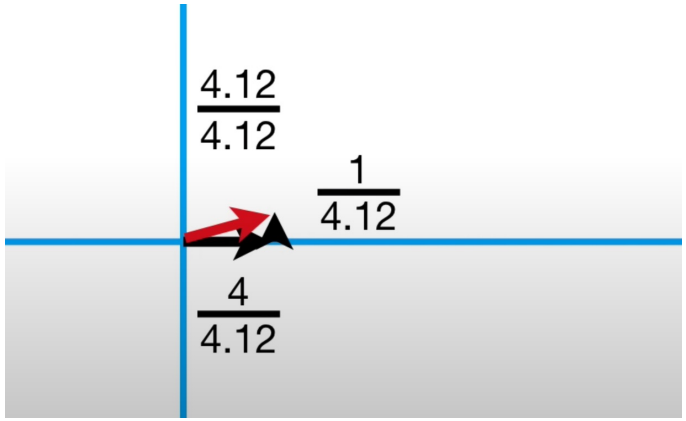
# Interpret the principal component




Suppose the best-fitted line (PC1) has a slope of 0.25. In other words, for every 4 units that we go out along the x-axis (Maths), we go out 1 unit along the y-axis (Physics)

- That means the data are mostly spread out along the x-axis. The ratio tells us that Maths is more important when it comes to describing how the data are spread out.
- Next, we will scale the vector (4, 1) into a unit vector along the direction of PC1.

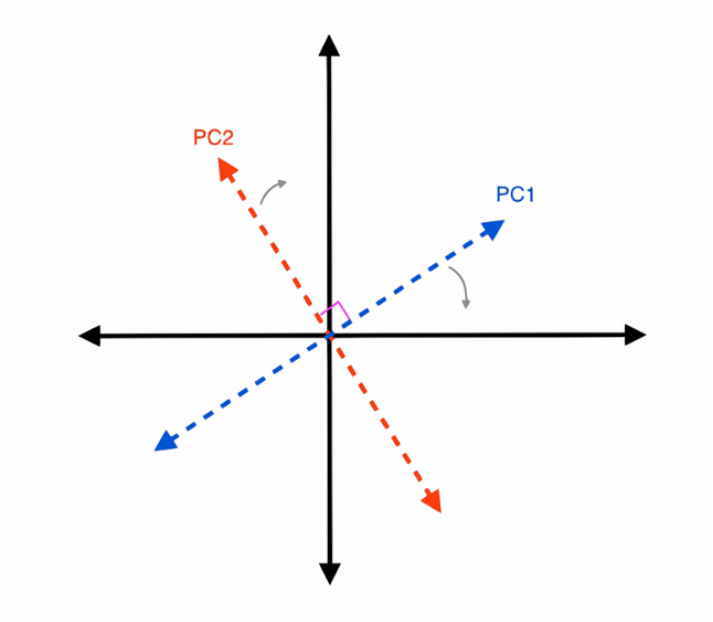
To get the unit vector, we simply scale the triangle so that the red line is 1 unit long.




This one-unit-long vector, which is (0.97, 0.242) in this case, is called the '**Singular vector**' or the '**Eigenvector**' for PC1.

The average value of the sum of squares of the distances between the projected points on the best-fit line and the origin is called the '**Eigenvalue**' for PC1

Because this is a 2D graph, PC2 is simply the line through the origin that is perpendicular to PC1.




And we can calculate the Eigenvector and Eigenvalue for PC2 just like what we did for PC1.

Remember the Eigenvalues we calculated for PC1 and PC2? These values are actually the measures of variations.

- Suppose that the variation (or eigenvalue) we calculated for PC1 is 15, and the variation for PC2 is 3.
- The total variations around both PCs would be 15+3 =18
- **We can infer that PC1 accounts for 15/18 = 83% of the total variations, while PC2 accounts for 17% of the total variations.**

Now that you get a sense of how PCA works for two-dimensional data. What if we now have three variables?

The processes are pretty much the same.

- Centered the data
- Fit a random line that goes through the origin and projects all the data onto it.
- Rotate the line to find the best-fitting line by maximizing the sum squared distance from the projected points to the origin. This best-fitting line will be the PC1.
- We then find PC2, which is the next best-fitting line given that it goes through the origin and **is perpendicular to PC1**.
- Lastly, we find PC3, the best-fitting line given that it goes through the origin and **is perpendicular to both PC1 and PC2**
- If we have more variables, we'd just keep on finding more principle components by adding perpendicular lines

In theory, there is one principal component per each variable but in practice, the number of PCs is either the number of variables or the number of samples, whichever is smaller.

Once we have all the principal components figured out, we can calculate their eigenvectors and eigenvalues. To wrap up, eigenvectors help us determine which variable is more important for each PC, and eigenvalues can help us determine the proportion of variation that each PC accounts for.

| | Student1 | Student2 | Student3 | Student4 | Student5 | Student6 |
|---|---|---|---|---|---|---|
| **Maths** | 10.0 | 11.0 | 8.0 | 3.0 | 1.0 | 2.0 |
| **Physics** | 6.0 | 4.0 | 5.0 | 3.0 | 3.2 | 1.0 |
| **History** | 12.0 | 9.0 | 10.0 | 2.5 | 1.3 | 2.0 |

The Eigenvectors along each PC are as follows:

```
Eigen Vector along PC1 : [-2.68191702e+00  1.78779473e+00  5.00135483e-16]
Eigen Vector along PC2 : [ 6.04528374e+00 -1.29486389e-01  5.00135483e-16]
Eigen Vector along PC3 : [-3.36336672e+00 -1.65830834e+00  5.00135483e-16]
```

PC1 is majorly comprised of variable 1, 'Maths', followed by 'Physics' and 'History'. PC2 being perpendicular to PC1, is majorly comprised of 'Maths'. PC3 is perpendicular to both PC1 and PC2 and has the greatest proportion for 'Maths'.

By calculating the eigenvalues, we get the proportion of variation that each PC accounts for.

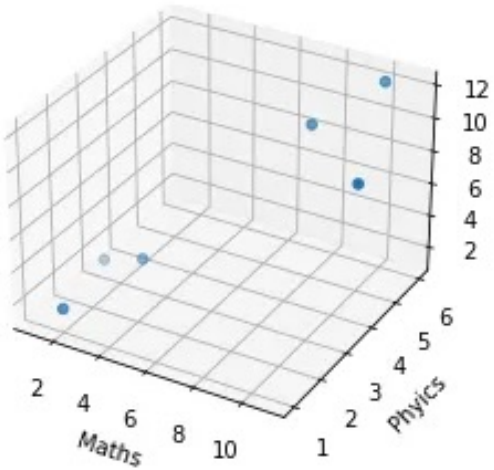
```
PC1 accounts for : 90.22678664377429%
PC2 accounts for : 9.773213356225702%
PC3 accounts for : 1.2299057648569933e-30%
```

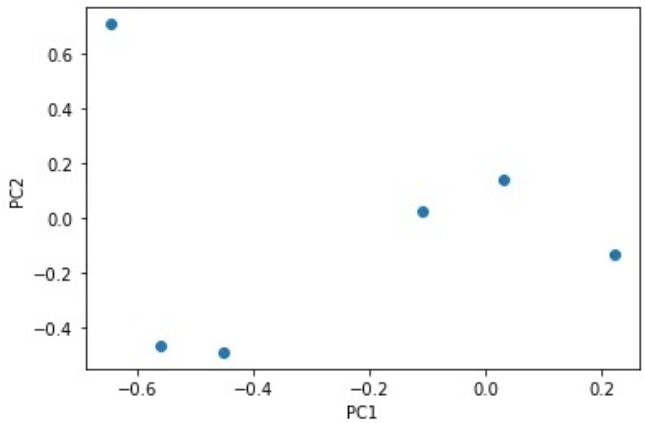
This tells us that:

- PC1 accounts for 90% of the variation.
- PC2 accounts for 9.7% of the variation.
- PC3 accounts for $\tilde{0}$% of the variation.

PC1 and PC2 account for the vast majority of the variation. That means that a 2D graph using only PC1 and PC2 would be a good approximation of the original 3D graph since it would account for 99.7% of the variation in the data.

After selecting the desired PCs, we convert the 3D graph to a 2D graph by striping away the PC3 and original axis. We will be left with PC1, PC2, and data points. The data points need to be projected on PC1 and PC2, and then spotting the position of the transformed point using the coordinate pair from the projection on each PC. Thus we end up transforming 3D data to a 2D data while capturing 99.7% of the variation.



# R example

Now, let's explore further how we could conduct PCA analysis in R.
The dataset ('mtcars') we're going to work on contains automobile design and performance information for 32 automobiles.

```
## Rows: 32
## Columns: 5
## $ displacement <dbl> 160.0, 160.0, 108.0, 258.0, 360.0, 225.0, 360.0, 146.7, 1…
## $ horsepower   <dbl> 110, 110, 93, 110, 175, 105, 245, 62, 95, 123, 123, 180, …
## $ weight       <dbl> 2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570, 3.190, 3…
## $ gear         <dbl> 4, 4, 4, 3, 3, 3, 3, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 4, 4, …
## $ carburetors  <dbl> 4, 4, 1, 1, 2, 1, 4, 2, 2, 4, 4, 3, 3, 3, 4, 4, 4, 1, 2, …
```

- disp: Displacement (cu.in.)
- hp: Gross horsepower
- wt: Weight (1000 lbs)
- gear: Number of forward gears
- carb: Number of carburetors

In R, there's a built-in function called 'prcomp()' for performing PCA analysis. By setting 'scale. $=$ TRUE',we standardize all variables in the dataset, ensuring each contributes equally to the analysis. This standardization involves converting values to z-scores, calculated as $z = \frac{x-\mu}{\sigma}$.

```
pca_result <- prcomp(mtcars, scale. = TRUE)
```

From the output, PC1 captures 62.56% of the total variance and PC1 captures 29.11% of the total variance. By examining the cumulative proportion, we can decide how many PCs to consider for a simplified yet comprehensive representation of our data. For example, if we want to set the criterion to explain around 90% of the variance, the first two PCs (cumulatively 91.67%) would be sufficient.

```
summary(pca_result)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5
## Standard deviation     1.7686 1.2065 0.46967 0.39587 0.19773
## Proportion of Variance 0.6256 0.2911 0.04412 0.03134 0.00782
## Cumulative Proportion  0.6256 0.9167 0.96084 0.99218 1.00000
```

To determine which variable(s) the PCs represent or are most strongly associated with in our analysis, we need to look at the eigenvectors (also known as loadings), which indicate the contribution of each original variable to each principal component.

```
loadings <- pca_result$rotation

pc_loadings <- loadings[, c(1, 2)]
print(pc_loadings)
```

```
##                     PC1        PC2
## displacement  0.5369859  0.1493075
## horsepower    0.4985817 -0.2973280
## weight        0.5213334  0.1780703
## gear         -0.2680366 -0.6956855
## carburetors   0.3455868 -0.6112413
```

- Displacement, Horsepower, and Weight have strong positive loadings on PC1. This indicates that PC1 is heavily influenced by the size and power of the cars.
- Gear and Carburetors have strong negative loadings on PC2, indicating that PC2 largely captures variations in the transmission system and fuel system complexity. The negative signs mean that as we move along PC2, the number of gears and carburetors tends to decrease.

# Principle Component Regression (PCR)

Principal Component Regression (PCR) involves **using the principle components as the predictors in a linear regression model**. The idea is that often a small number of principal components suffice to explain most of the variability of the data. Thus, instead of directly regressing on the independent variables, we utilize the principle components as the model's predictors.

Suppose now we want to build a prediction model for the car's horsepower using the available information in the dataset. We'll use the package 'pls' in R to perform PCR.

```
pcr_model <- pcr(horsepower~displacement+weight+gear+carburetors,
                 data=mtcars, scale=TRUE, validation = 'CV')
```

- Again, by setting scale=TRUE, we normalize the data before fitting the regression model
- The'validation = 'CV' parameter specifies that cross-validation (CV) is used to evaluate the model's predictive performance. CV involves using part of the data to fit the model and another part of the data to test the model, iteratively, to get a more accurate measure of its predictive performance.

```
summary(pcr_model)
```

```
## Data:    X dimension: 32 4
##  Y dimension: 32 1
## Fit method: svdpc
## Number of components considered: 4
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps
## CV           69.66    47.98    35.74    37.61    28.69
## adjCV        69.66    47.87    35.45    37.22    28.34
##
## TRAINING: % variance explained
##             1 comps  2 comps  3 comps  4 comps
## X             61.57    93.17    97.48   100.00
## horsepower    52.56    78.86    80.11    88.64
```

Based on the summary output, the PCR findings align with our PCA analysis conclusions. The initial two principal components account for over 90% of the variance in the covariate data.

```
pcr_pred <- predict(pcr_model, test_data, ncomp=2)
```

We can then reduce dimensionality effectively by utilizing a PCR model with two principal components for predictions, by specifying 'ncomp=2'.

# Conclusion

- Principal Component Analysis (PCA) is a fundamental technique for dimensionality reduction
- PCA enables us to simplify and visualize high-dimensional data by identifying the principal components, which are orthogonal directions capturing the maximum variance in the data.
- By doing so, PCA transforms complex datasets into a lower-dimensional space while preserving important information.
- Practically, PCA allows us to choose the number of principal components that best represent the data while discarding less significant components.

# Today

**Learning goals**

- The Basics of Principal Component Analysis (PCA)
- Exploring the Process and Mechanics of PCA
- Identifying and Interpreting Principal Components
- PCA for Dimensionality Reduction