Outline

1. Initial thoughts (in layman's terms)
2. Setting the scene
3. Principle of PCA
4. SVD vs PCA
5. PCA in R

Which kinds of data?

Principal Components Analysis (from now on **PCA**) applies to data tables where rows are considered as **individuals** and columns as **quantitative variables**

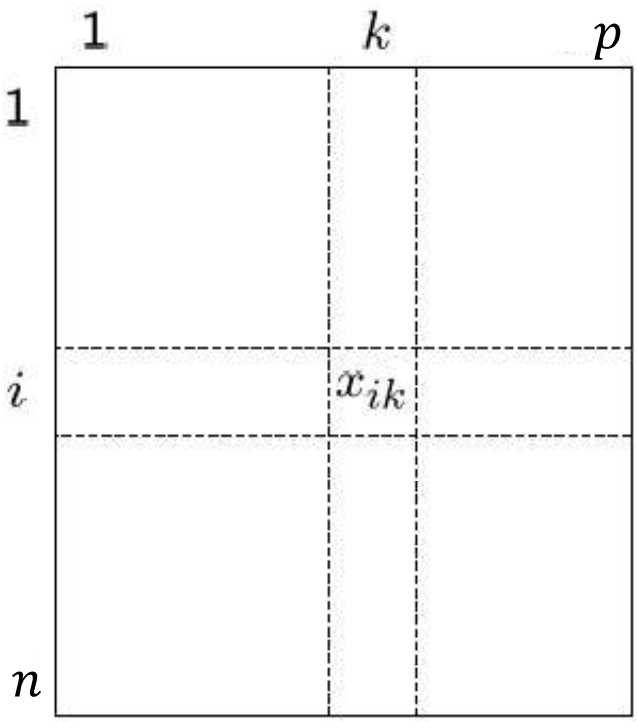


Figure: Data table in PCA

Examples:

- Ecology: concentration of pollutant k in river i
- Economics: indicator value k in year i
- Genetics: expression of gene k for patient i
- Biology: measure k for animal i
- Marketing: value of measure k for brand i
- Sociology: time spent on activity k by individuals from social class i
- etc.

Initial thoughts (in layman's terms)

- **What PCA does?**

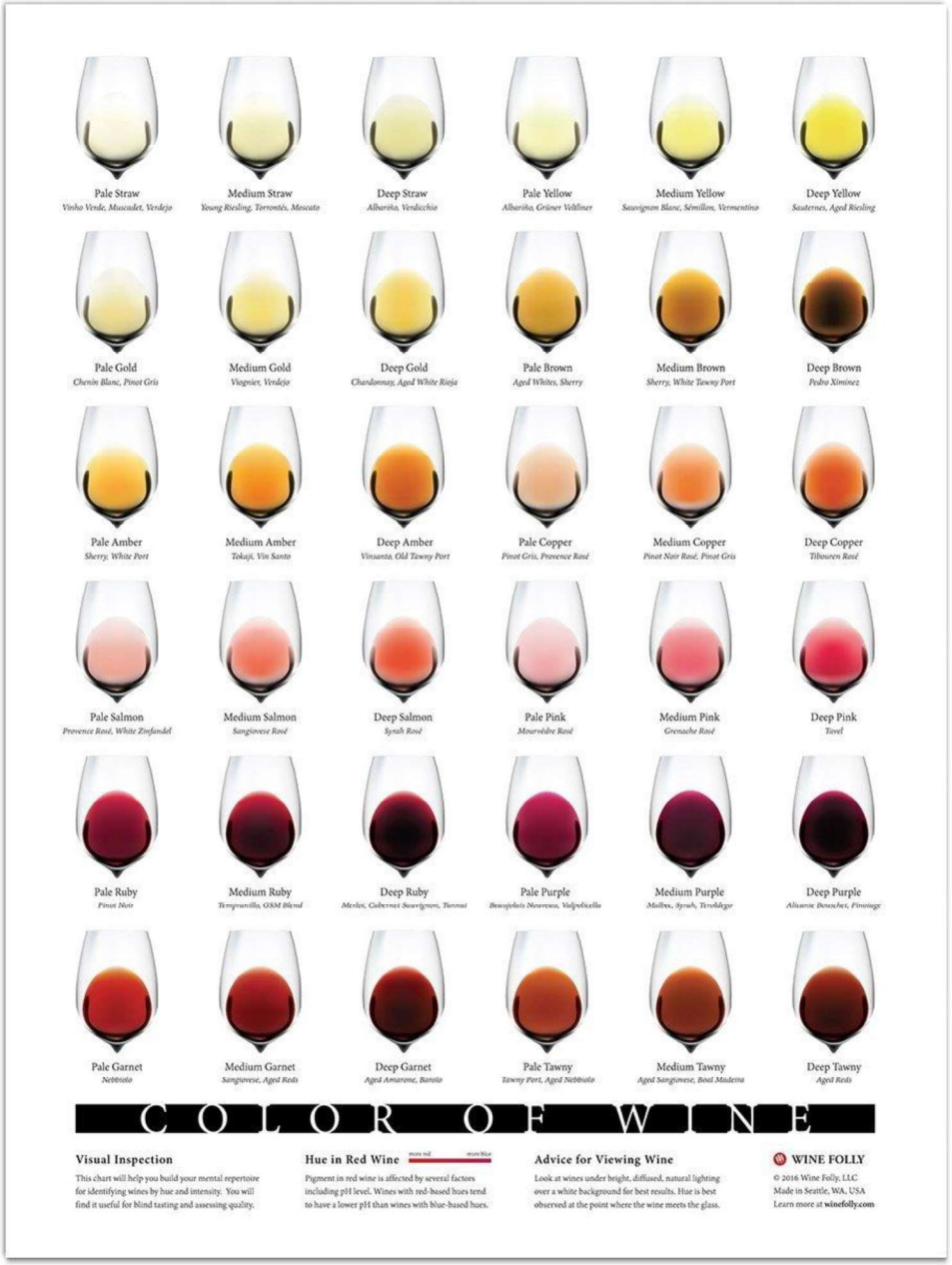
It's just a method of **summarizing some data**.

For instance, we have some wine bottles standing on a table. We can describe each wine by its colour, by how strong it is, by how old it is, and so on. Thus, we can compose a whole list of different characteristics of each wine in our cellar.

But many of them will measure related properties and so **will be redundant**. If so, we should be able to **summarize each wine with fewer characteristics**. This is what PCA does.

https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues

Initial thoughts (in layman's terms)



Source: http://winefolly.com/tutorial/wine-color-chart/

Initial thoughts (in layman's terms)

- **Does PCA check what characteristics are redundant and discards them?**

No. PCA is not selecting some characteristics and discarding the others.

PCA constructs some ***new* characteristics** that turn out to summarize our list of wines well.

These new characteristics are constructed using the old ones.

For example, a new characteristic might be computed as *wine age minus wine acidity level* or some other combination like that (we call them ***linear combinations***)

PCA finds the **best possible characteristics**, the ones that summarize the list of wines as well as only possible (among all conceivable linear combinations).

Initial thoughts (in layman's terms)

- **What do we actually mean when we say that these new PCA characteristics "summarize" the list of wines?**

Two possible answers here:

1) We look for some wine properties (characteristics) that strongly differ across wines.

Imagine that you come up with a *property* that is the same for most of the wines.

This would not be very useful (bad summary of the data) because wines are very different and this *property* makes them all look the same.

PCA looks for properties that show **as much variation across wines as possible**.

Initial thoughts (in layman's terms)

- **What do we actually mean when we say that these new PCA characteristics "summarize" the list of wines?**

2) We look for the properties that would allow you to **predict**, or "reconstruct", the original wine characteristics

Imagine that you come up with a *property* that has no relation to the original characteristics.

This would not be very useful as there is no way you could reconstruct the original ones (bad summary of the data, again).

PCA looks for properties that allow to **reconstruct the original characteristics as well as possible**.

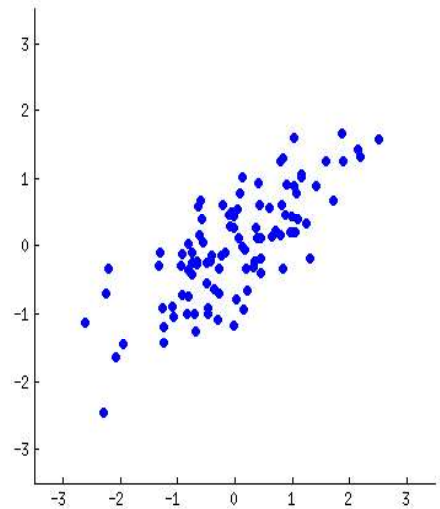
Those two goals (reconstruct and variation) are equivalent and PCA can do both.

Initial thoughts (in layman's terms)

- **Why would these two goals be equivalent?**

Let's select two wine features: wine darkness & alcohol content.

They're **correlated**



Scatter plot: each dot is one particular cloud.

A *new* property can be constructed. How?

Drawing a line through the center of the cloud and projecting all points into this line. Let's do it

Initial thoughts (in layman's terms)

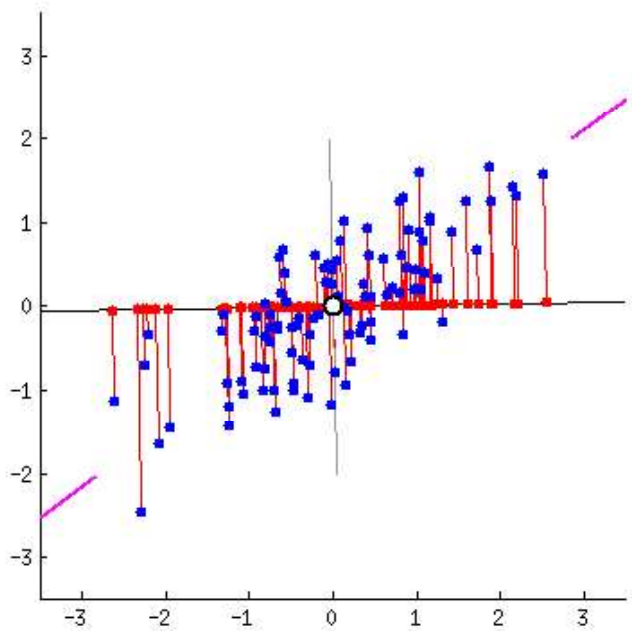
- **Why would these two goals be equivalent?**

The *new* property will be given by a linear combination

$$a_1\, x + a_2\, y$$

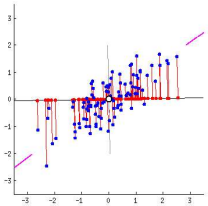
x : wine darkness and y: alcohol content

The values of a_1 and a_2 will determine **different lines**



Initial thoughts (in layman's terms)

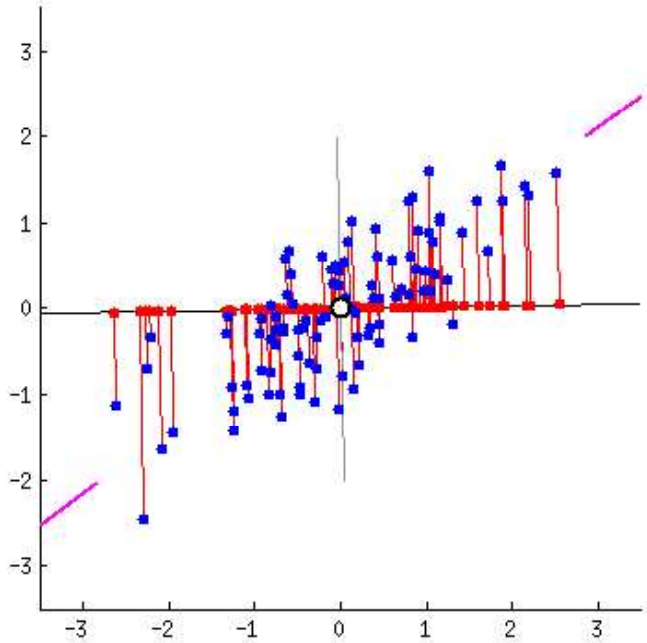
- **Why would these two goals be equivalent?**



PCA will find the "best" line according to two different criteria/goals of what is the "best"

1. The **variation** of values along this line should be maximal. Pay attention to how the "spread" (we call it "variance") of the red dots changes while the line rotates; *can you see when it reaches maximum?*
2. The **reconstruction**: if we reconstruct the original two characteristics (position of a **blue dot**) from the new one (position of a **red dot**), the reconstruction error will be given by the length of the connecting **red line.** Observe how the length of these red lines changes while the line rotates; *can you see when the total length reaches minimum?*

Initial thoughts (in layman's terms)



<u>The maximum variance and the minimum error</u> are reached at the same time.

when the red line reaches magenta line.

This magenta line corresponds to the **new wine property** that will be constructed by PCA.

This new property is called **first principal component** or PC1

Setting the scene

- The data table can be seen as a set of rows or a set of columns

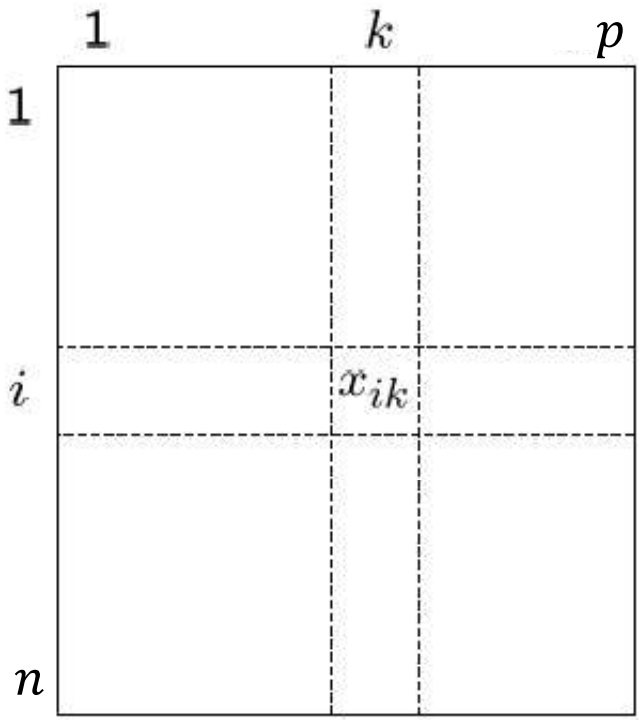
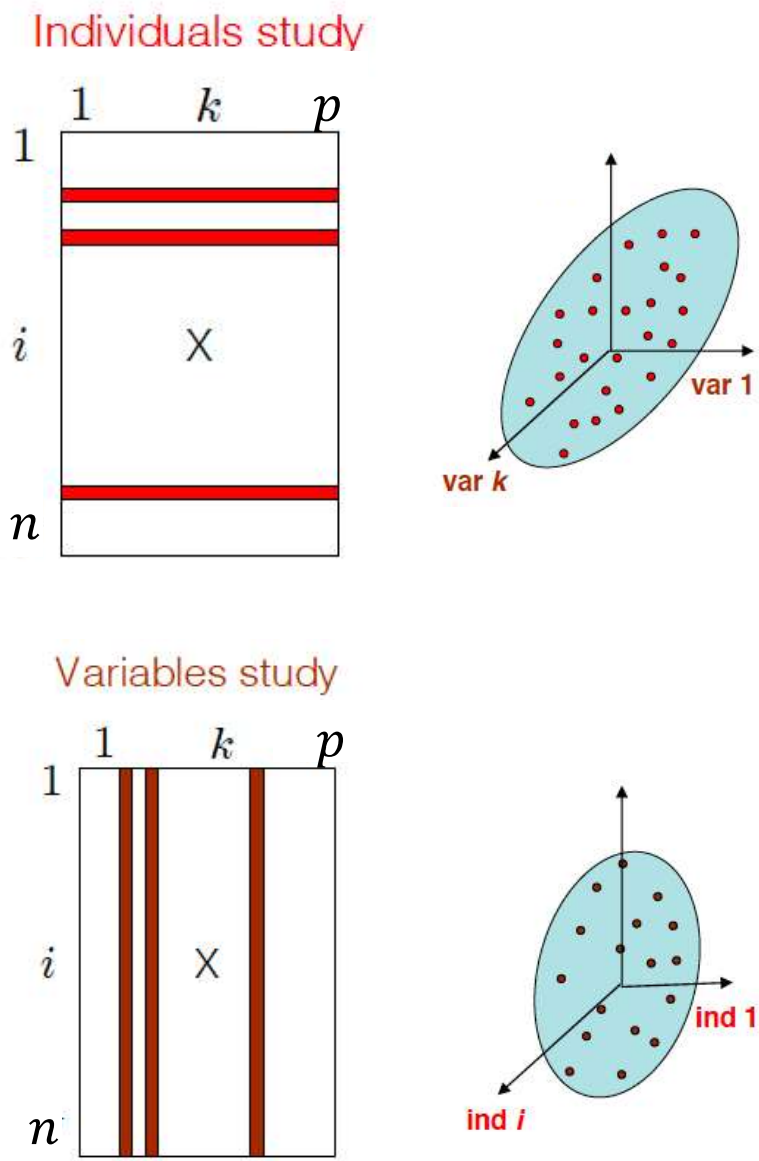


Figure: Data table in PCA



Setting the scene

- The basic aim of PCA is to **describe variation** in a set of **correlated variables** $x_1, x_2, \cdots, x_p$ in terms of a new set of **uncorrelated variables** $y_1, y_2, \cdots, y_p$.
  - Note that p is the same number in the original variables and the new uncorrelated ones.
- Each of $y_1, y_2, \cdots, y_p$ is a **linear combination** of the x variables. For instance,

$$y_1 = a_{11}\, x_1 + a_{12}\, x_2 + \cdots + a_{1p}\, x_p$$

- The new variables are derived in **decreasing order of "importance",** in the sense that
  - y_1 accounts for as much of the variation (**variance**) in the original data amongst all linear combinations of $x_1, x_2, \cdots, x_p$.
  - Then, y_2 is chosen to account for **as much as possible** of the remaining variation, subject to being uncorrelated with y_1, and so on.

Dimensionality reduction

- The new variables defined by this process, $y_1, y_2, \cdots, y_p$, are the **principal components** (PCs).

- The hope is that the first few PCs will account for a **substantial proportion** of the **variation** in the original variables $x_1, x_2, \cdots, x_p$

- If so, the first few PCs can be used to provide a **lower dimensional summary** of the data.

- The PCs form an **orthogonal coordinate system**.

Finding the sample principal components

- The first PC of the observations is the linear combination

$$y_1 = a_{11}\, x_1 + a_{12}\, x_2 + \cdots + a_{1p}\, x_p$$

whose <u>sample variance is greatest among all such linear combinations.</u>

- How can we increase the variance of the first PC?
  - Simply by increasing the **coefficients (a.k.a. loadings)** $a_{11}, a_{12}, \cdots, a_{1p}$ (like in linear regression)
  - Since the variance of y_1 could be increased without limit, a restriction must be placed on those coefficients.

- A sensible **constraint** is to require that the <u>sum of squares of the coefficients for each PC should take the value one</u>.

How can we found the sample PCs?

1) Eigendescomposition of the sample covariance matrix

2) Singular Value Descomposition (SVD)

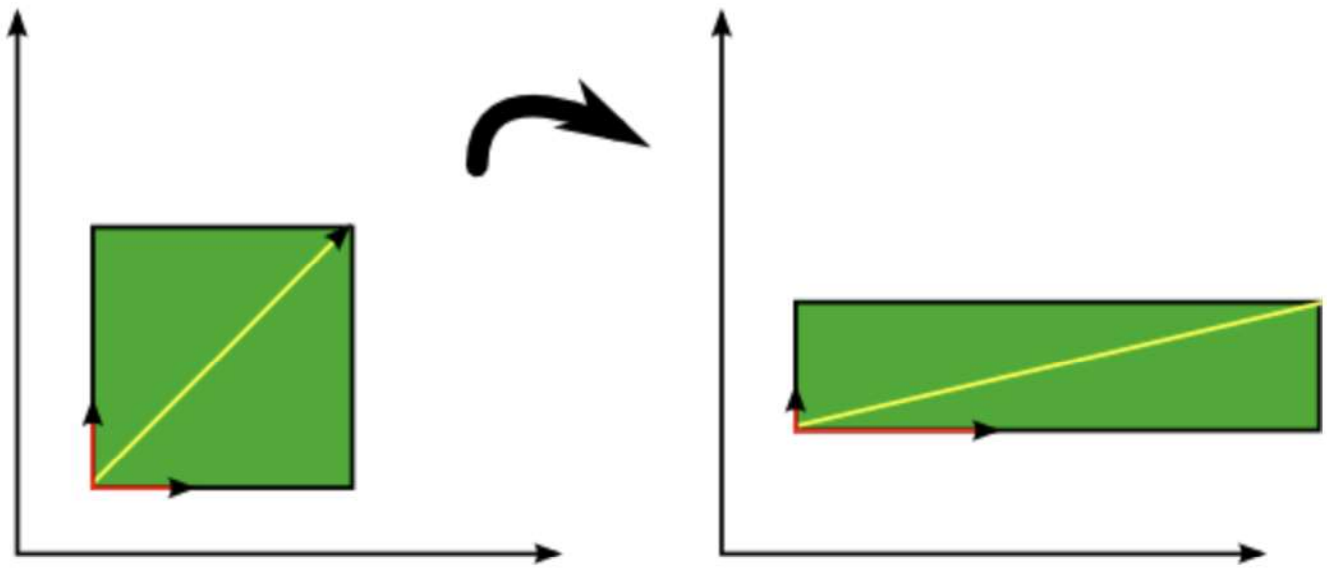
Dot product and matrix multiplication: the product ***C***=***AB*** of two matrices ***A*** *(n×m)* and ***B*** *(m×p)* should have a shape of *n×p.* Two matrices can be multiplied only when the second dimension of the former matches the first dimension of the latter. The element c_{ij} in the resultant matrix ***C*** is computed as:

$$c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj} = \mathbf{a}_{i,*} \cdot \mathbf{b}_{*,j}$$

Eigendescomposition of the sample covariance matrix

Do you remember what is an eigenvalue and an eigenvector?

An **eigenvector** is a vector whose direction remains unchanged when a linear transformation is applied to it



Eigenvectors (red) do not change direction when a linear transformation (e.g. scaling) is applied to them.

The **eigenvalue** is the scalar the defines that linear transformation

https://www.youtube.com/watch?v=PFDu9oVAE-g

Eigendescomposition of the sample covariance matrix

- Let $\mathbf{S}$ be the positive semi-definite covariance matrix of a mean-centered data matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ with $\text{rank}(\mathbf{S}) = r$ $(r \leq p)$.

https://en.wikipedia.org/wiki/Definite_symmetric_matrix

Rank(S): maximal number of linearly independent columns of S

- The eigenvalue decomposition (or spectral decomposition) of $\mathbf{S}$ can be written as

$$\mathbf{S} = \mathbf{A}\mathbf{\Lambda}\mathbf{A}^\top = \sum_{i=1}^{r} \lambda_i \mathbf{a}_i \mathbf{a}_i^\top \quad ,$$

where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_r)$ is an $r \times r$ diagonal matrix containing the positive eigenvalues of $\mathbf{S}$, $\lambda_1 \geq \cdots \geq \lambda_r > 0$, on its main diagonal and $\mathbf{A} \in \mathbb{R}^{p \times r}$ is a column-wise orthonormal matrix whose columns $\mathbf{a}_1, \ldots, \mathbf{a}_r$ are the corresponding unit-norm eigenvectors of $\lambda_1, \ldots, \lambda_r$.

Orthonormal means they are orthogonal (90 degrees from each other, therefore, independent) and they have length 1.

PCA via the eigendescomposition

- Two vectors are **orthogonal** if they are perpendicular to each other, i.e., the <u>dot product of the two vectors is zero</u>
- A vector is **orthonormal** if it has length 1, i.e., if the <u>dot product of the vector with itself is one</u>

- PCA looks for r vectors $\mathbf{a}_j \in \mathbb{R}^{p\times 1}$ $(j = 1, \ldots, r)$ which

$$\begin{array}{ll} \text{maximize} & \mathbf{a}_j^\top \mathbf{S}\mathbf{a}_j \\ \text{subject to} & \mathbf{a}_j^\top \mathbf{a}_j = 1 \quad \text{for } j = 1, \ldots, r \quad \text{and} \\ & \mathbf{a}_i^\top \mathbf{a}_j = 0 \quad \text{for } i = 1, \ldots, j-1 \quad (j \geq 2) \ . \end{array}$$

- It turns out that $\mathbf{y}_j = \mathbf{X}\mathbf{a}_j$ is the j-th sample PC with zero mean and variance λ_j, where $\mathbf{a}_j$ is an eigenvector of $\mathbf{S}$ corresponding to its j-th largest eigenvalue λ_j $(j = 1, \ldots, r)$.

- The total variance of the r PCs will equal the total variance of the original variables so that $\sum_{j=1}^{r} \lambda_j = \mathrm{tr}(\mathbf{S})$.

Singular Value Descomposition (SVD)

- Singular value decomposition (SVD) is commonly used **dimensionality reduction** approaches

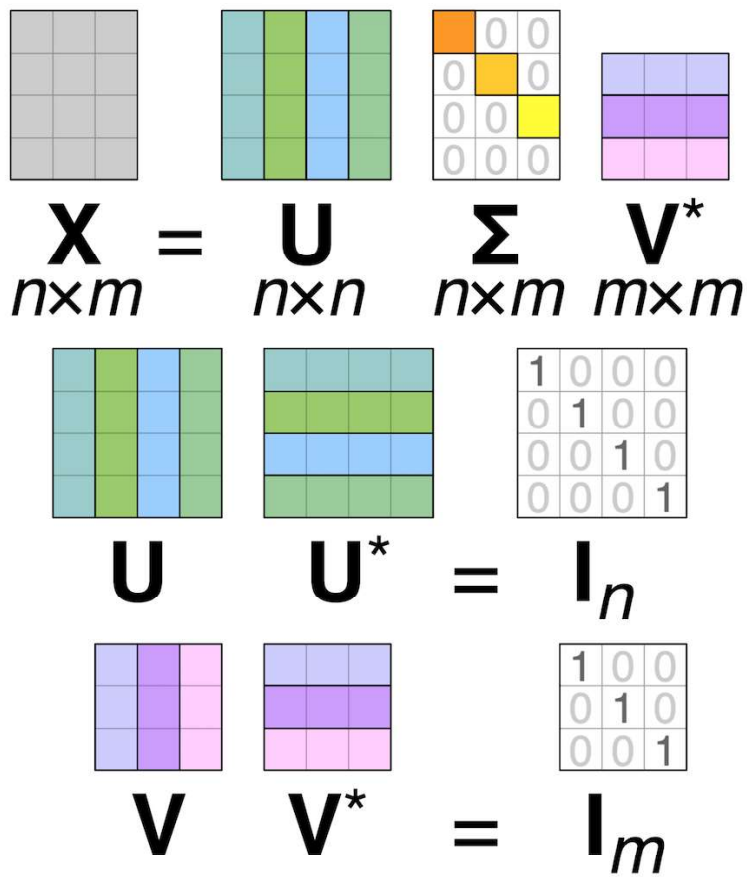
- As PCA, SVD attempts to **find linear combination**s of features in the original high dimensional data matrix to construct meaningful representation of the dataset.

- SVD is very popular in the field of **natural language processing** to achieve a representation of the gigantic while sparse word frequency matrices

- The resultant representations from PCA and SVD are similar in some data. In fact, **PCA and SVD are closely related.**

Singular Value Descomposition (SVD)

- SVD decomposes a matrix into the product of two unitary matrices (***U***, ***V****) and a rectangular diagonal matrix of singular values (***Σ***):



Singular **values are the absolute values** of the eigenvalues

Singular Value Descomposition (SVD)

- PCA and SVD are closely related approaches and can be both applied to decompose any rectangular matrices.
- We can look into their relationship by performing SVD on the covariance matrix **S***:*

$$\begin{aligned}\mathbf{S} &= \frac{\mathbf{X}^\top \mathbf{X}}{n-1} \\ &= \frac{\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^\top \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top}{n-1} \\ &= \mathbf{V}\frac{\boldsymbol{\Sigma}^2}{n-1}\mathbf{V}^\top\end{aligned}$$

- We notice that the result is in the same form with eigen decomposition of **S** (slide 23, with V as A), we can easily see the relationship between singular values (**Σ**) and eigenvalues (**Λ**):

$$\boldsymbol{\Lambda} = \frac{\boldsymbol{\Sigma}^2}{n-1}$$

Relationship between eigenvalue and singular values

- Therefore, we can actually perform PCA using SVD.

More references:

https://stats.stackexchange.com/questions/134282/relationship-between-svd-and-pca-how-to-use-svd-to-perform-pca

https://math.stackexchange.com/questions/3869/what-is-the-intuitive-relationship-between-svd-and-pca

https://stats.stackexchange.com/questions/79043/why-pca-of-data-by-means-of-svd-of-the-data

https://stats.stackexchange.com/questions/121162/is-there-any-advantage-of-svd-over-pca

PCA in R

- In R, PCA can be done using the functions `princomp()` and `prcomp()` (both contained in the R package stats).
- The `princomp()` function carries out PCA via an **eigendecomposition** of the sample covariance matrix **S**.
- When the variables are on very **different scales**, PCA is usually carried out on the **correlation matrix R**.
  - These components (sometimes) **are not fully equal** to those derived from **S**.

PCA using the function `princomp()`

- First of all, PCA assumes that of the variables in the data matrix X has been centered to have mean zero

- Centering **does not modify** the shape of the cloud

- 6 ways of centering in R:
  https://www.gastonsanchez.com/visually-enforced/how-to/2014/01/15/Center-data-in-R/

- Perhaps the most simple, quick and direct way to mean-center your data is by using the function `scale()`.
  - By default, this function will standardize the data (mean zero, unit variance).
  - To indicate that we just want to subtract the mean, we need to turn off the argument `scale = FALSE`.

PCA using the function `princomp()`

Example of centering in R.

For illustration purposes we'll use the following random small dataset:

```
# small dataset
set.seed(212)
Data = matrix(rnorm(15), 5, 3)

Data
```

```
##         [,1]    [,2]    [,3]
## [1,] -0.2392  0.1545  0.1503
## [2,]  0.6769  1.0369  0.5097
## [3,] -2.4403 -0.7796 -0.7733
## [4,]  1.2409  0.6213  1.8757
## [5,] -0.3265  0.2994  0.7883
```

PCA using the function `princomp()`

Example of centering in R.

We create a new R function only centering:

```
# centering with 'scale()'
center_scale <- function(x) {
    scale(x, scale = FALSE)
}

# apply it
center_scale(Data)
```

```
##          [,1]     [,2]       [,3]
## [1,] -0.02153 -0.11200 -0.3597876
## [2,]  0.89458  0.77038 -0.0004599
## [3,] -2.22270 -1.04610 -1.2834512
## [4,]  1.45853  0.35477  1.3655295
## [5,] -0.10887  0.03294  0.2781693
## attr(,"scaled:center")
## [1] -0.2176  0.2665  0.5101
```

Variance explained by PCs

- The <u>total variance of the *p* PCs</u> will **equal** the total <u>variance of the original variables</u> so that

$$\sum_{j=1}^{p} \lambda_j = s_1^2 + s_2^2 + \cdots + s_p^2 \ ,$$

where λ_j is the variance of the jth PC and s_j^2 is the sample variance of x_j.

- Consequently, the jth PC accounts for a proportion: $\dfrac{\lambda_j}{\sum_{j=1}^{p} \lambda_j}$

  and the first *k* PCs account for a proportion: $\dfrac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{p} \lambda_j}$

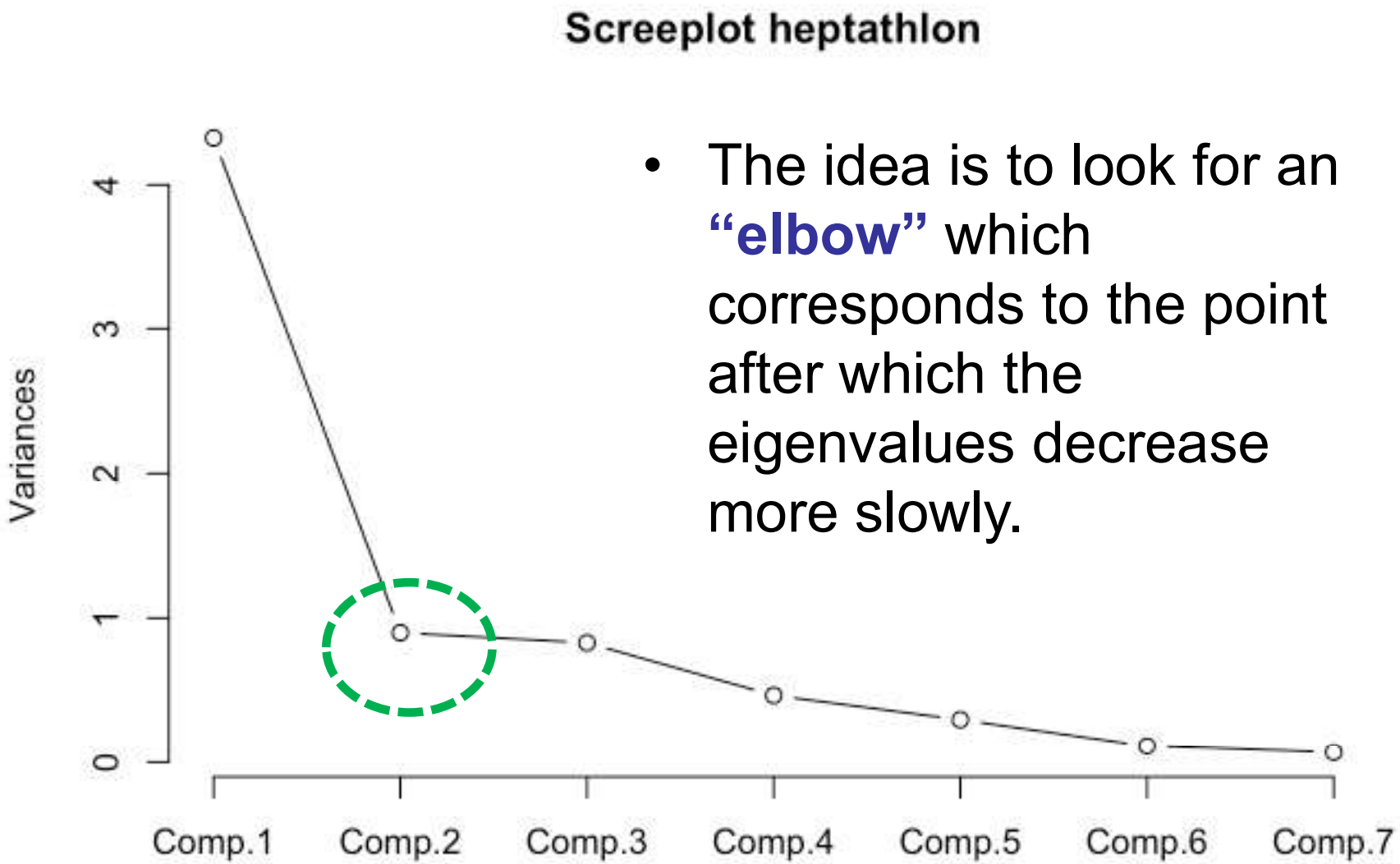
Criteria for choosing the number of PCs

- Retain the first **k** components which explain a **large proportion** of the total variation, say 70-80%.

- Examine a **scree plot**.
  - This is a plot of the component variances versus the component number.
  - The idea is to look for an **"elbow"** which corresponds to the point after which the eigenvalues decrease more slowly.

- Consider whether the component has a **sensible and useful interpretation**.

Example: The Olympic heptathlon data



- The idea is to look for an **"elbow"** which corresponds to the point after which the eigenvalues decrease more slowly.

PCA biplots

- **Biplots** are a graphical method for simultaneously displaying the variables and sample units described by a multivariate data matrix.

- A **PCA biplot** displays the component scores and the variable loadings obtained by PCA in two or three dimensions.

- The computations are based on the singular value decomposition (SVD) of the (centered and possibly scaled) data matrix **X**.

Let's come back to our heptathlon example.

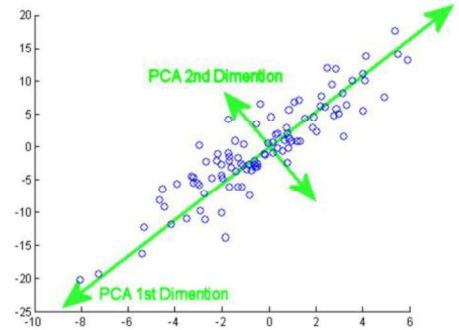
Other packages

In R, there are several functions from different packages that allow us to perform PCA.

Here, we list 5 different ways to do a PCA using the following functions (with their corresponding packages in parentheses)

- `prcomp()` (stats)
- `princomp()` (stats)
- `PCA()` (FactoMineR)
- `dudi.pca()` (ade4)
- `acp()` (amap)

Principal Components Analysis (PCA)



PCA example

•Type of learning: Unsupervised.

•Task: Dimension reduction.

•Used:

- Derive a set of variables of reduced dimension with respect to a total set of variables
- It reduces the dimensions of the data and helps us understand, graph the data with a smaller dimension compared to the original data.

•Type of data:

- PCA is designed for continuous variables (p> 10)

•Application areas: data compression, image processing, visualization, exploratory data analysis, pattern recognition and time series prediction.