Multivariate Verfahren

7.3. Principal Component Analysis (PCA)

Hannah Kümpel

Institut für Statistik, LMU München

Sommersemester 2024

Contents

Introduction I

- **Principal component analysis (PCA)** is a feature extraction method.

$\longrightarrow$ I.e. it is a method that reduces the dimension of data points by calculating fewer new variables while retaining as much information as possible.

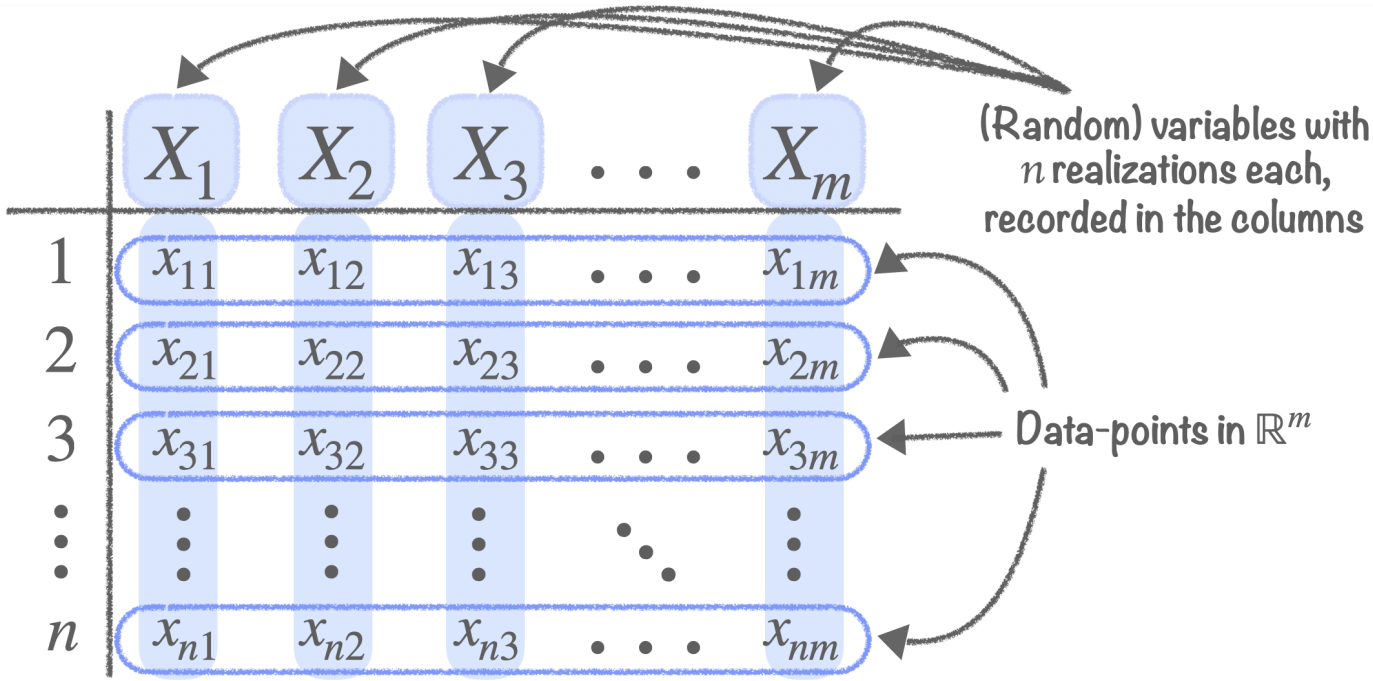
Introduction II

- Reminder: We consider given data as a matrix $\boldsymbol{X}$:

	X_1	X_2	X_3	$\cdots$	X_m
1	x_{11}	x_{12}	x_{13}	$\cdots$	x_{1m}
2	x_{21}	x_{22}	x_{23}	$\cdots$	x_{2m}
3	x_{31}	x_{32}	x_{33}	$\cdots$	x_{3m}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
n	x_{n1}	x_{n2}	x_{n3}	$\cdots$	x_{nm}

Introduction II

- Reminder: We consider given data as a matrix $\boldsymbol{X}$. Specifically, in the case of dimension reduction we consider the following aspects:



- Applying feature extraction means computing n values of new (random) variables $\tilde{X}_1, \ldots, \tilde{X}_p$, $p < m$, which make up the columns of a new matrix $\tilde{\boldsymbol{X}}$ **that contains data points in $\mathbb{R}^p$, $p < m$,** so that we can reasonably perform analysis on the data $\tilde{\boldsymbol{X}}$ instead of $\boldsymbol{X}$.

Introduction III

- In PCA, specifically, this is achieved via

  1. First, an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (called the *first principal component*), the second greatest variance on the second coordinate, and so on.[1]

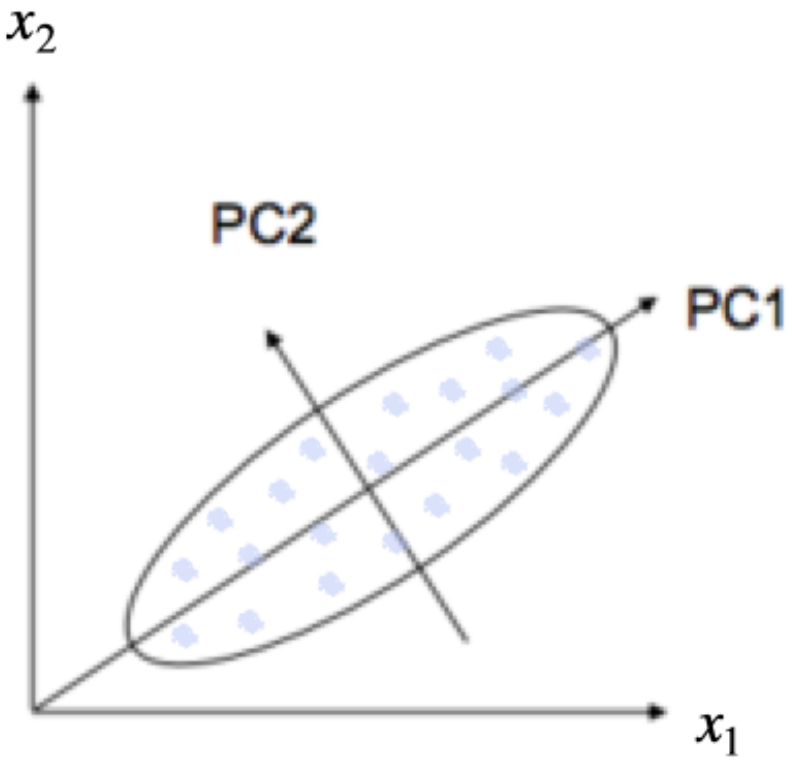
  2. Then, the dimension is reduced by projecting each data point on the first p principal components, so that *most of* the variance remains explained.

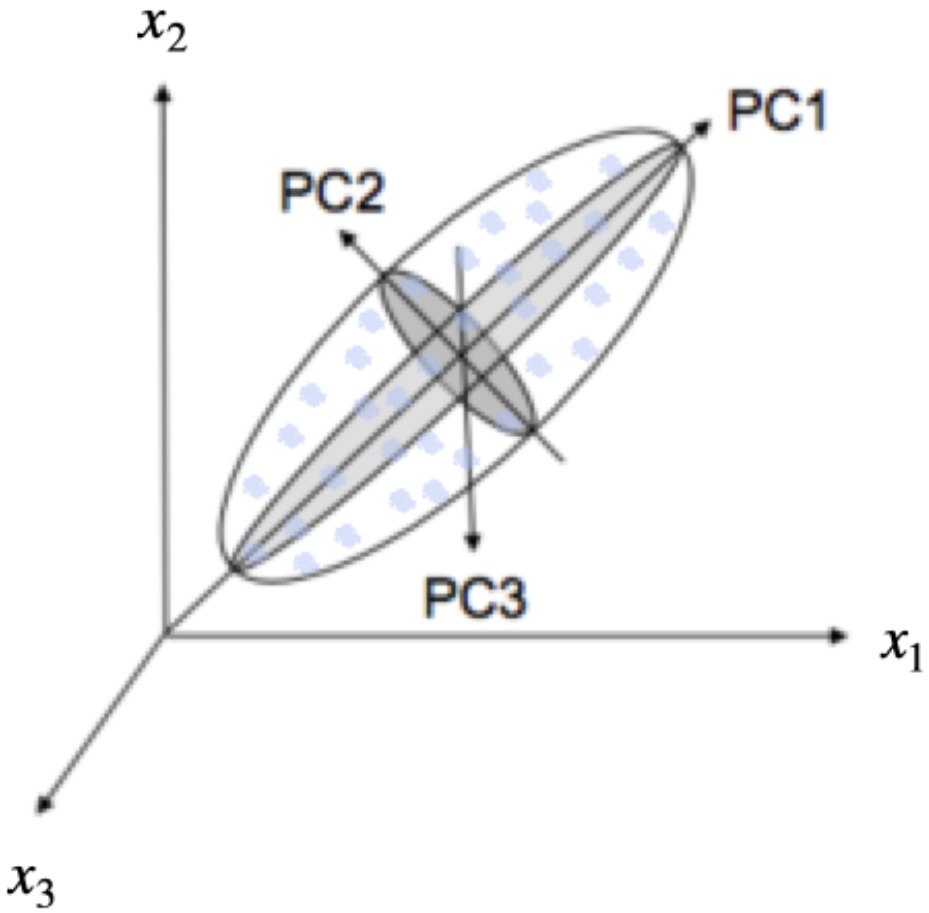
[1] Jolliffe, I. T. (2002). Principal Component Analysis. Springer Series in Statistics. New York: Springer-Verlag. doi:10.1007/b98835. ISBN 978-0-387-95442-4.

PCA Motivation: Principal components I

- First, let us understand the **principal components (PCs)**
  (Calculation of which is itself not a dimension reduction technique!)
- Intuitively, for a $\mathbb{R}^{n \times m}$ data matrix we can think of the principal components as the axes of a m-dimensional ellipsoid fitted to the n data points in $\mathbb{R}^m$, ordered by length in descending order.
- Examples for $m = 2$ and $m = 3$ are given by the following:

PCA Motivation: Principal components II





PCA Motivation: Projecting data points on the PCs I

- The next section will formalize what constitutes principal components given data n data points in $\mathbb{R}^m$.

- For now, please keep the following in mind:

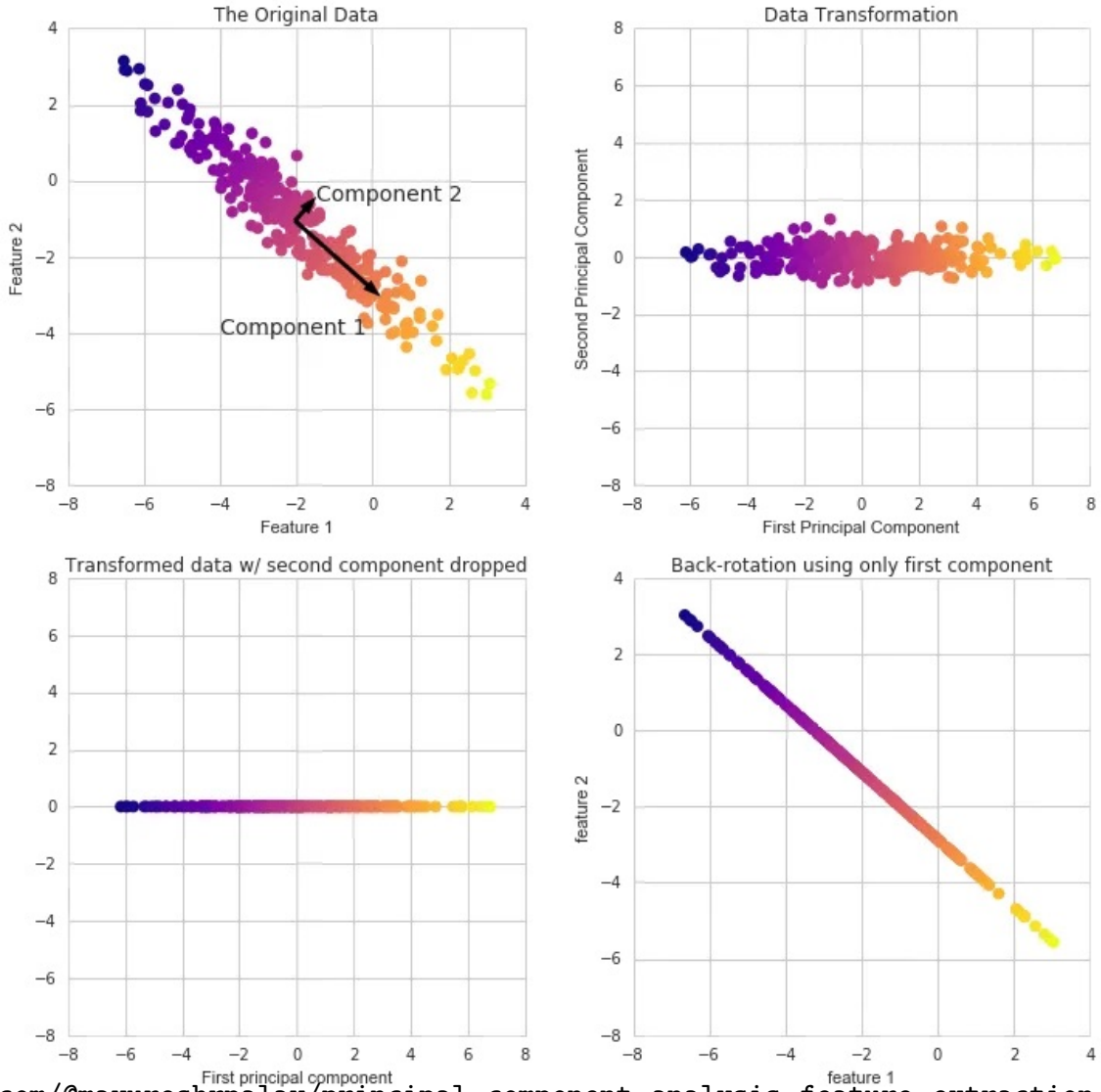
  1. The PCs (as axes of an ellipsoid) are orthogonal to each other and there are exactly as many as dimensions of the data points

  $\longrightarrow$ We can easily consider the continuation of each PC as an axis in an m-dimensional coordinate system.

  2. Heuristically, we can observe that the longer a PC/axis, the higher the variance of the data "in that direction".

  3. One can use a matrix P with $P = P^2 = P^\top$ to orthogonally project data points $x \in \mathbb{R}^m$ onto a vector (line) **such as a principal component** in $\mathbb{R}^m$.

PCA Motivation: Projecting data points on the PCs II



Source: https://medium.com/@mayureshrpalav/principal-component-analysis-feature-extraction-technique-3f480d7b9697

Contents

Spatial intuition for matrix multiplication

- To gain a deeper understanding of PCA, it is helpful to have some spatial intuition for the linear transformation of a point in $\mathbb{R}^n$, $n \in \mathbb{N}$, that is achieved by multiplication with a matrix $A \in \mathbb{R}^{n \times n}$.

- To this end, we will first take a closer look at the linear transformation of the unit circle using matrices in $\mathbb{R}^{2 \times 2}$.

- Throughout, please keep the following in mind:
  - Any matrix $A \in \mathbb{R}^{n \times n}$, $n \in \mathbb{N}$, defines a mapping $\mathbb{R}^n \longrightarrow \mathbb{R}^n$, $x \mapsto Ax$. If it additionally holds that $A = A^2$, this mapping is a projection.
  - For any square matrix $A \in \mathbb{R}^{n \times n}$, $n \in \mathbb{N}$, $\boldsymbol{v} \in \mathbb{R}^n$ is an eigenvector of A corresponding to the eigenvalue λ of A, iff

$$A\boldsymbol{v} = \lambda \boldsymbol{v}\,.$$

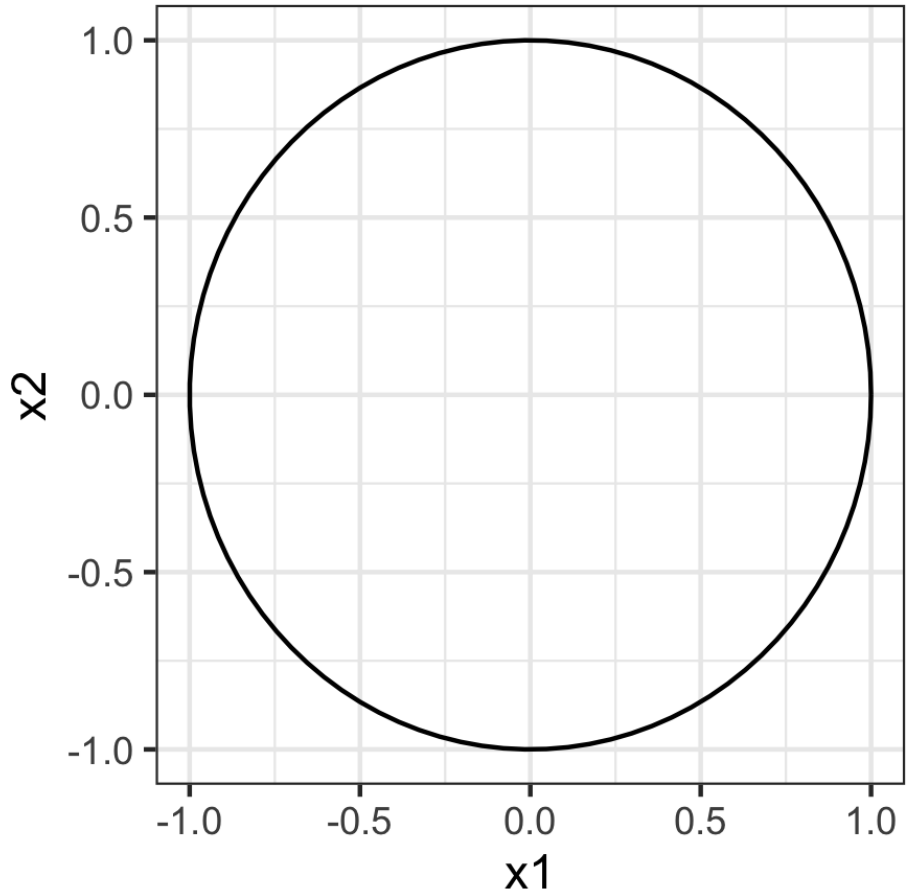
Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ I

- We begin with the unit circle and the identity matrix $\boldsymbol{I}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.
- Note that the eigenvalues of $\boldsymbol{I}_2$ are $\lambda_1 = \lambda_2 = 1$ and any 2 orthogonal vectors in $\mathbb{R}^2$ of length 1 may be chosen as corresponding eigenvectors, such as $\boldsymbol{v}_1 = (1,0)^\top$, $\boldsymbol{v}_2 = (0,1)^\top$; $\boldsymbol{v}_1 = (0,1)^\top$, $\boldsymbol{v}_2 = (-1,0)^\top$; etc.

Setup in R:

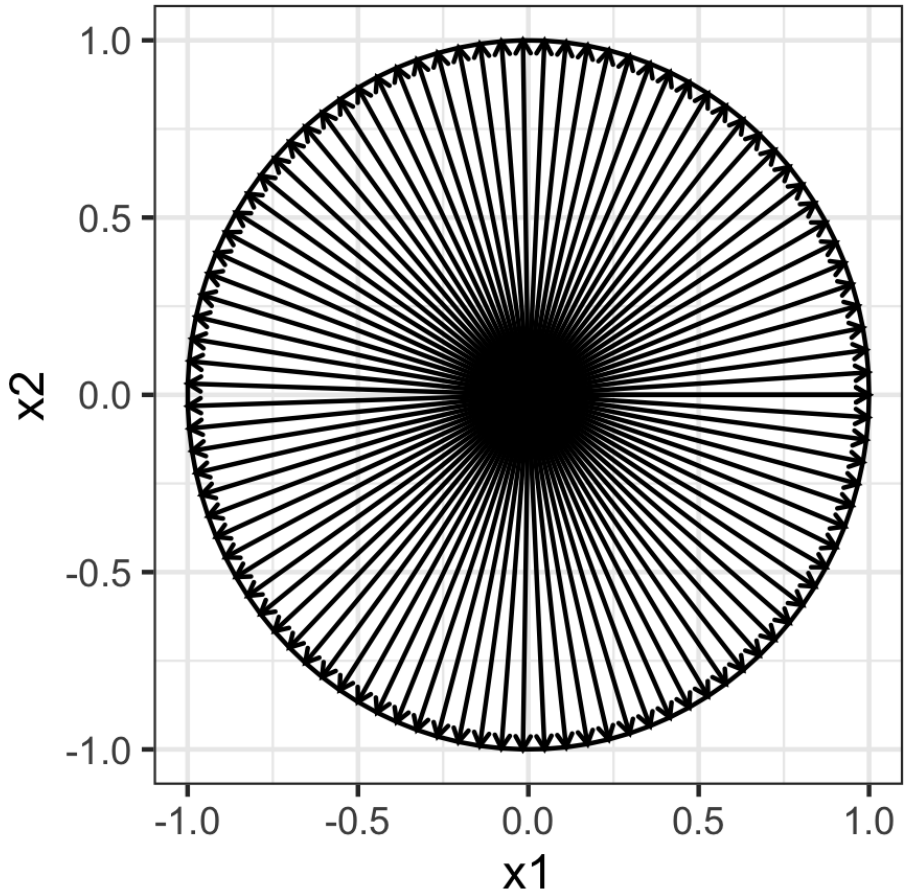
```
library(ggplot2)
library(tidyverse)
library(ordr)
library(metR)
library(RColorBrewer)

colors<-brewer.pal(8,"Set2")

circle <- function(center = c(0,0),r = 1, npoints = 100){
  tt <- seq(0,2*pi,length.out = npoints)
  xx <- center[1] + r * cos(tt)
  yy <- center[2] + r * sin(tt)
  return(data.frame(x1 = xx, x2 = yy))}

unit_circle <- circle()
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ II



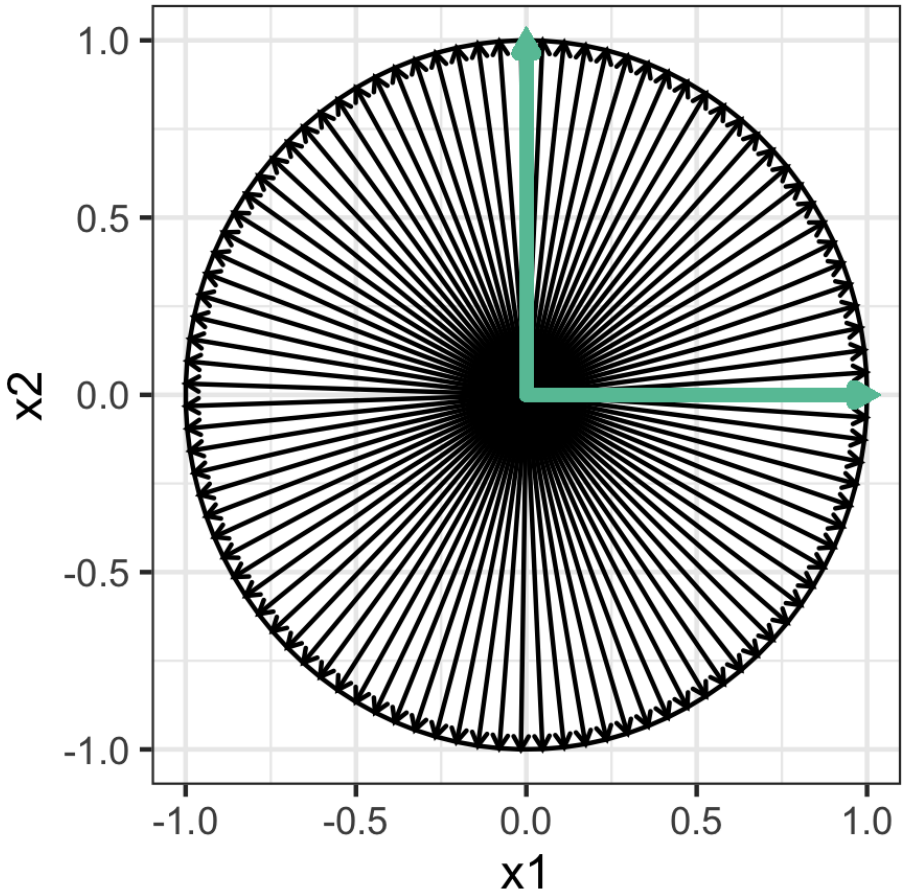

```
ggplot(unit_circle,aes(x1,x2))+geom_path()+theme_bw()
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ II



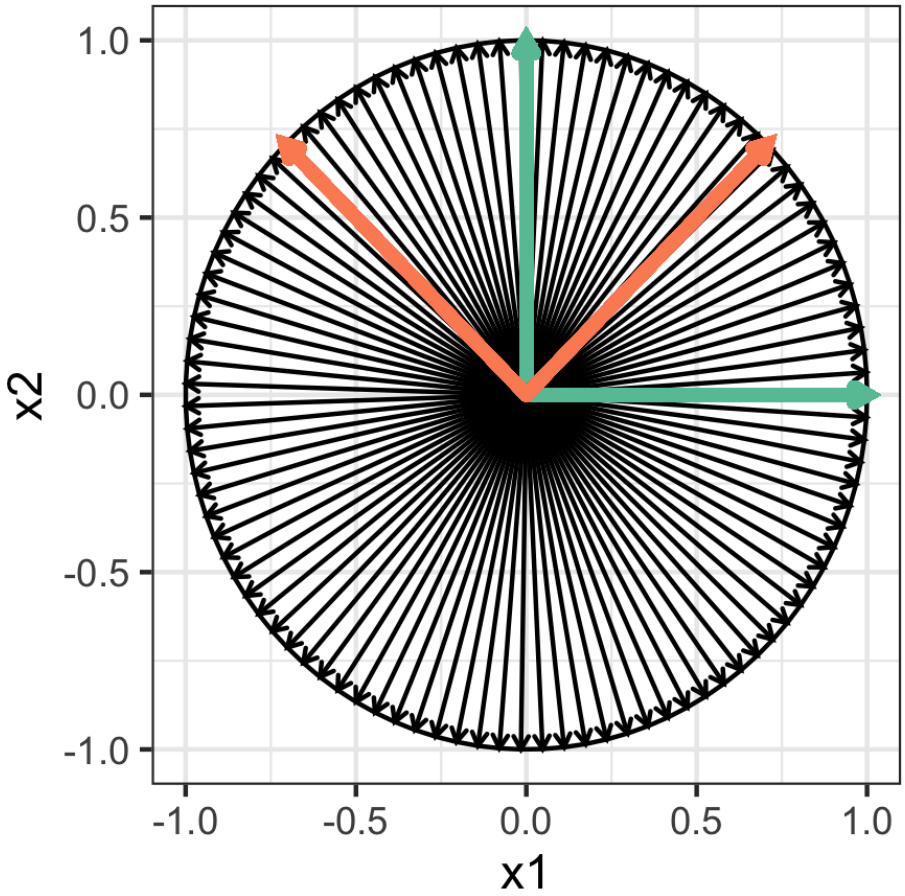

```
ggplot(unit_circle,aes(x1,x2))+geom_path()+theme_bw()+geom_vector()
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ II



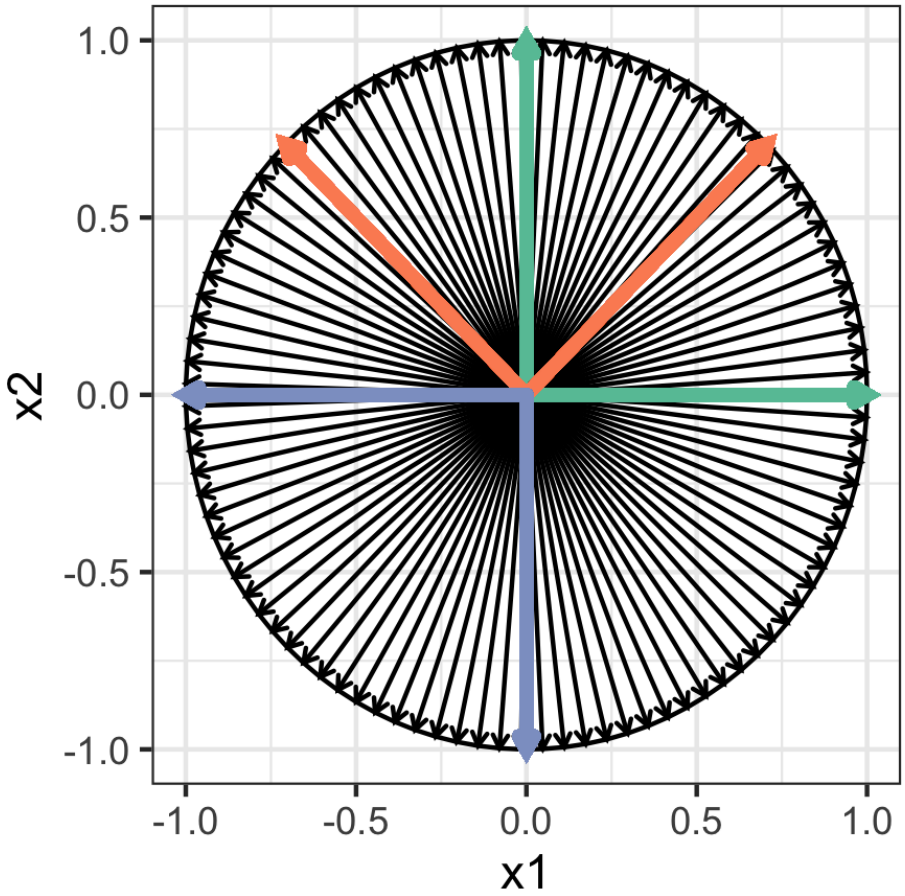

```
ggplot(unit_circle,aes(x1,x2))+geom_path()+theme_bw()+geom_vector()+
geom_vector(aes(x=1,y=0),color=colors[1],size=1.5)+geom_vector(aes(x=0,y=1),color=colors[1],size=1.5)
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ II



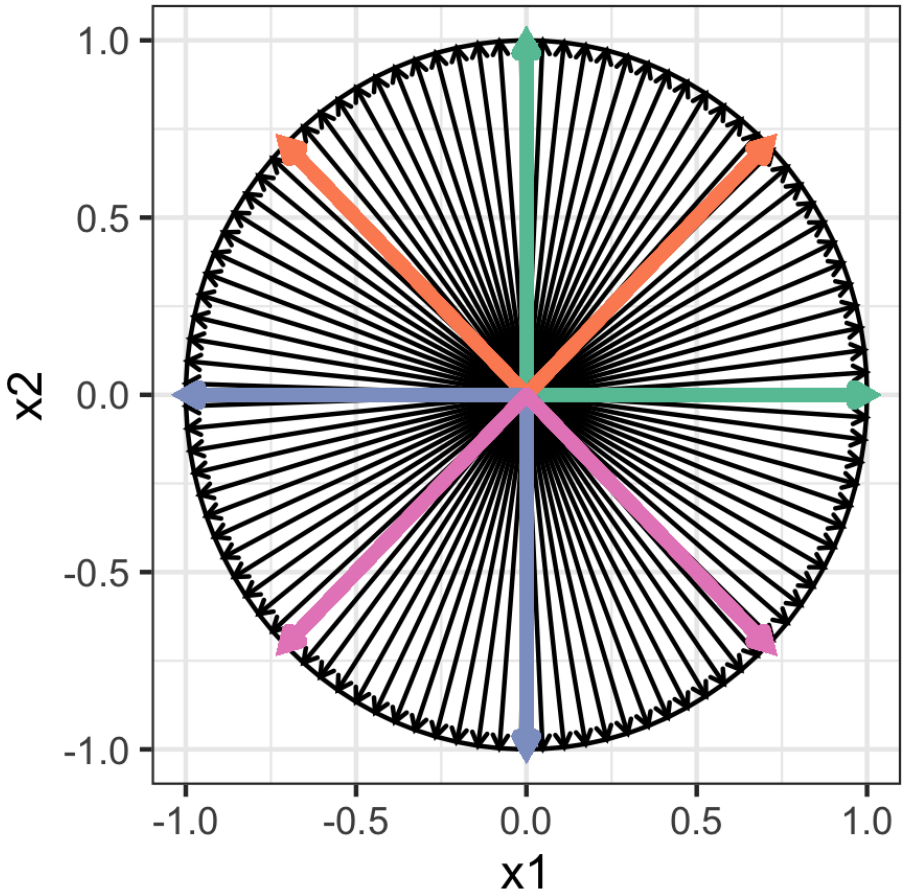

```
ggplot(unit_circle,aes(x1,x2))+geom_path()+theme_bw()+geom_vector()+
geom_vector(aes(x=1,y=0),color=colors[1],size=1.5)+geom_vector(aes(x=0,y=1),color=colors[1],size=1.5)+
geom_vector(aes(x=sqrt(1/2),y=sqrt(1/2)),color=colors[2],size=1.5)+geom_vector(aes(x=-sqrt(1/2),y=sqrt(1/2)),
color=colors[2],size=1.5)
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ II




```
ggplot(unit_circle,aes(x1,x2))+geom_path()+theme_bw()+geom_vector()+
geom_vector(aes(x=1,y=0),color=colors[1],size=1.5)+geom_vector(aes(x=0,y=1),color=colors[1],size=1.5)+
geom_vector(aes(x=sqrt(1/2),y=sqrt(1/2)),color=colors[2],size=1.5)+geom_vector(aes(x=-sqrt(1/2),y=sqrt(1/2)),
color=colors[2],size=1.5)+geom_vector(aes(x=-1,y=0),color=colors[3],size=1.5)+geom_vector(aes(x=0,y=-1),
color=colors[3],size=1.5)
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ II




```
ggplot(unit_circle,aes(x1,x2))+geom_path()+theme_bw()+geom_vector()+
geom_vector(aes(x=1,y=0),color=colors[1],size=1.5)+geom_vector(aes(x=0,y=1),color=colors[1],size=1.5)+
geom_vector(aes(x=sqrt(1/2),y=sqrt(1/2)),color=colors[2],size=1.5)+geom_vector(aes(x=-sqrt(1/2),y=sqrt(1/2)),
color=colors[2],size=1.5)+ geom_vector(aes(x=-1,y=0),color=colors[3],size=1.5)+geom_vector(aes(x=0,y=-1),
color=colors[3],size=1.5)+geom_vector(aes(x=-sqrt(1/2),y=-sqrt(1/2)),color=colors[4],size=1.5)+
geom_vector(aes(x=sqrt(1/2),y=-sqrt(1/2)),color=colors[4],size=1.5)
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ III

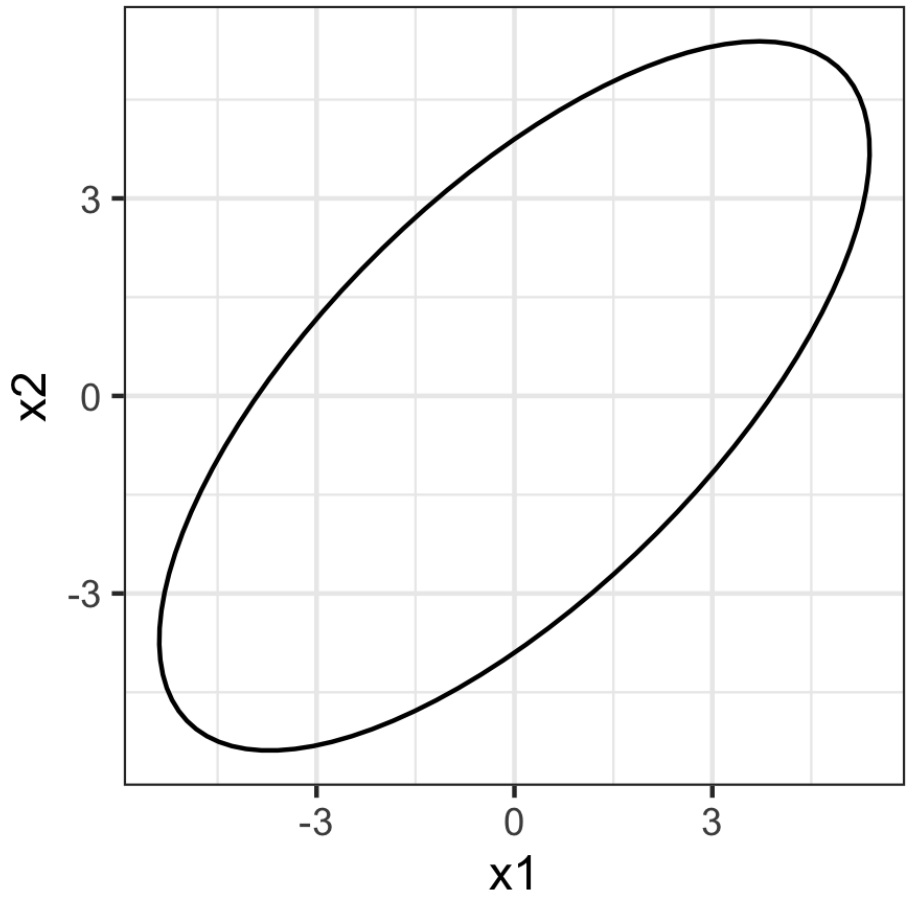
- Next, consider the **symmetric** matrix $A = \begin{pmatrix} 2 & 5 \\ 5 & 2 \end{pmatrix}$.

- We can get the eigenvalues and corresponding normalized eigenvectors using the `eigen()` function in R:

Continue setup in R from slide 12:

```
A<-matrix(c(2,5,5,2),ncol=2)

Aevals<-eigen(A)$values
Aevecs<-as.list(as.data.frame(eigen(A)$vectors))

Aevals
#[1] 7 -3

Aevecs
#$V1
#[1] 0.7071068 0.7071068
#
#$V2
#[1] -0.7071068  0.7071068
```

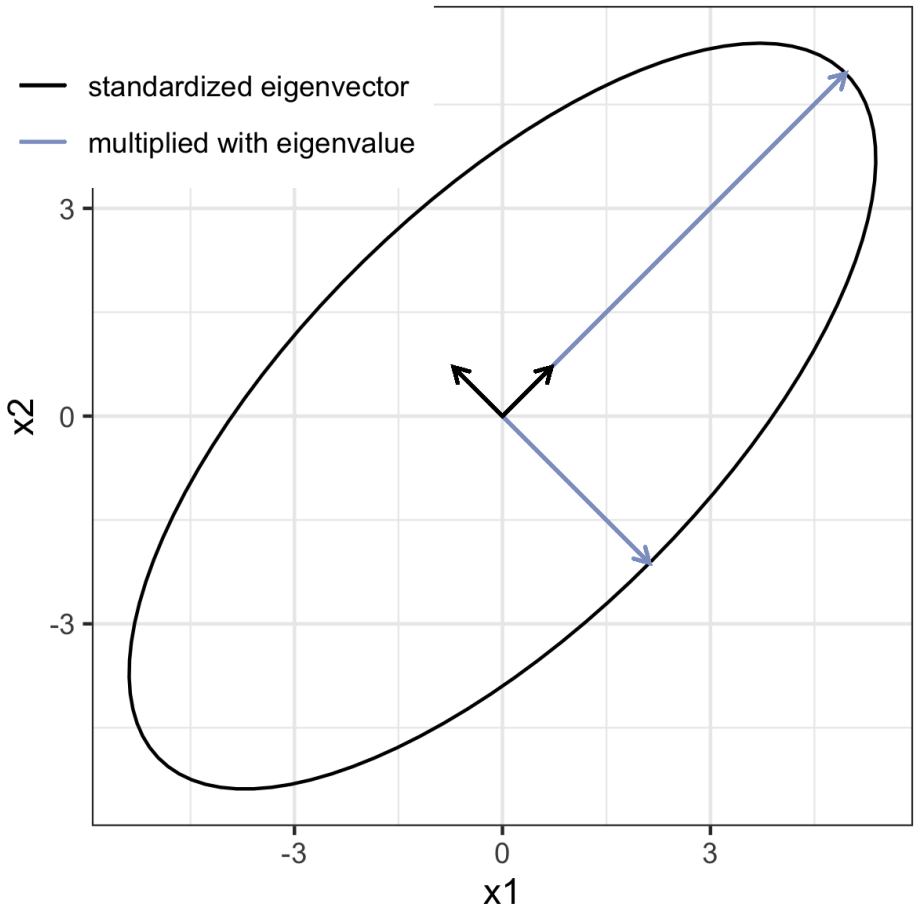
Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ IV



Multiplying A with every point $u \in \mathbb{R}^2$ on the unit circle results in the above circle!

```
ggplot(data.frame(x1=apply(unit_circle,1,function(x)A%*%x)[1,],
                  x2=apply(unit_circle,1,function(x)A%*%x)[2,]),aes(x1,x2))+geom_path()+theme_bw()
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ IV




```
ggplot(data.frame(x1=apply(unit_circle,1,function(x)A%*%x)[1,],x2=apply(unit_circle,1,function(x)A%*%x)[2,]),
aes(x1,x2))+geom_path()+theme_bw()+geom_vector(aes(x=Aevals[1]*Aevecs$V1[1],y=Aevals[1]*Aevecs$V1[2],color=
"multiplied with eigenvalue"))+geom_vector(aes(Aevals[2]*Aevecs$V2[1],y=Aevals[2]*Aevecs$V2[2],color=
"multiplied with eigenvalue"))+geom_vector(aes(Aevecs$V1[1],y=Aevecs$V1[2],color="standardized eigenvector"))+
geom_vector(aes(Aevecs$V2[1],y=Aevecs$V2[2],color="standardized eigenvector"))+labs(color = "")+
scale_color_manual(values = c("standardized eigenvector"="black","multiplied with eigenvalue"=colors[3]))+
theme(legend.position = c(0.15,0.9))
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ V

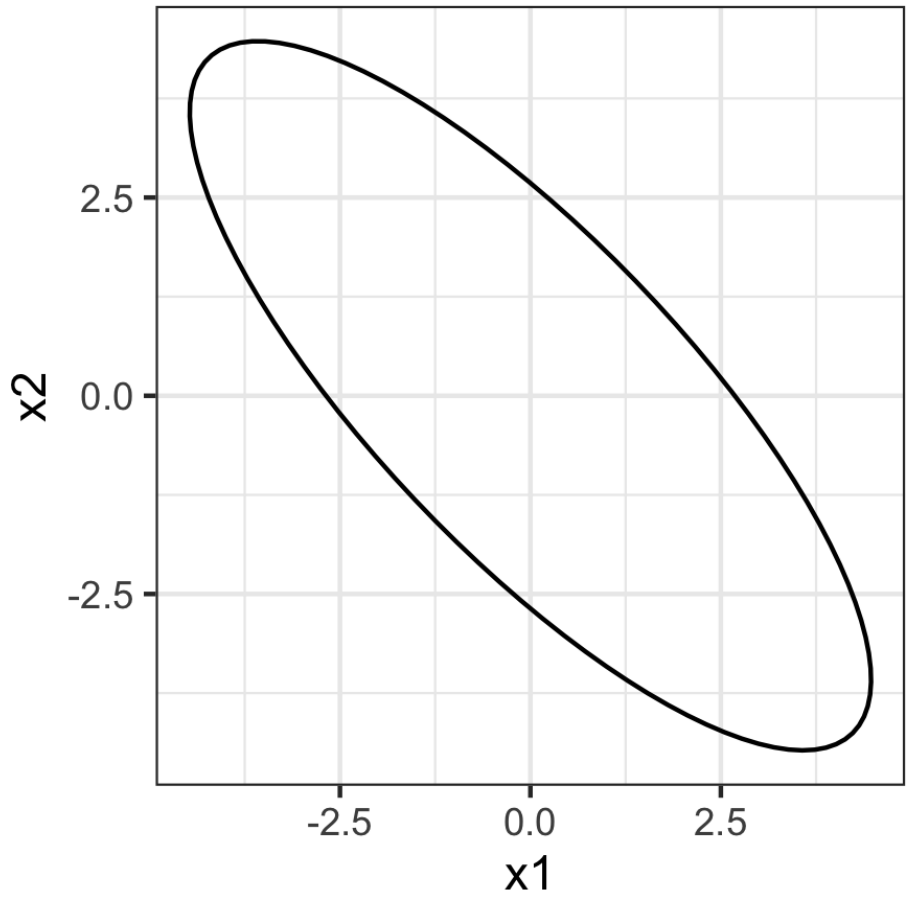
- Next, consider the **symmetric** matrix $B = \begin{pmatrix} -4 & 2 \\ 2 & -4 \end{pmatrix}$.

Again, continuing the setup in R from slide 12:

```
B<-matrix(c(-4,2,2,-4),ncol=2)
Bevals<-(-1)*eigen(B)$values
Bevecs<-as.list((-1)*as.data.frame(eigen(B)$vectors))

Bevals
#[1] 2 6
Bevecs
#$V1
#[1] -0.7071068 -0.7071068
#
#$V2
#[1] -0.7071068  0.7071068
```

- Note that, if we have determined an eigenvalue λ and and corresponding eigenvector $\boldsymbol{v}$, we can clearly always replace them by the eigenvalue $-\lambda$ and and corresponding eigenvector $-1 \cdot \boldsymbol{v}$!

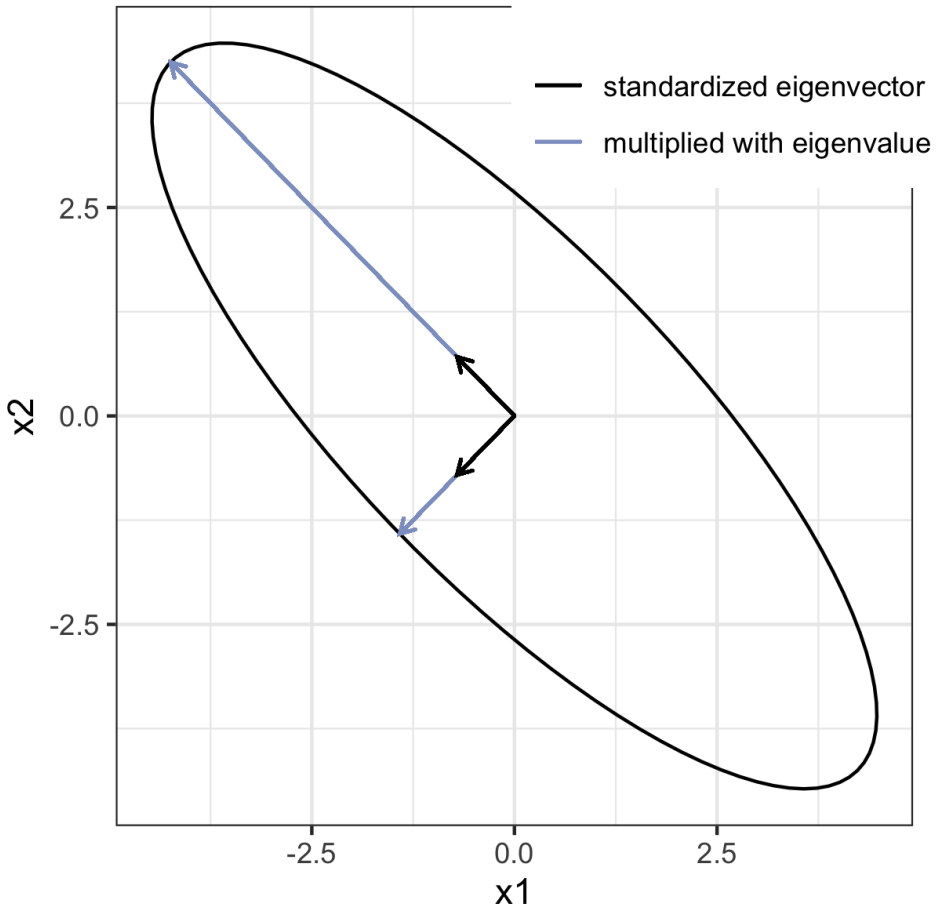
Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ VI



Multiplying B with every point $u \in \mathbb{R}^2$ on the unit circle results in the above circle!

```
ggplot(data.frame(x1=apply(unit_circle,1,function(x)B%*%x)[1,],
                  x2=apply(unit_circle,1,function(x)B%*%x)[2,]),aes(x1,x2))+geom_path()+theme_bw()
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ VI




```
ggplot(data.frame(x1=apply(unit_circle,1,function(x)B%*%x)[1,],x2=apply(unit_circle,1,function(x)B%*%x)[2,]),
aes(x1,x2))+geom_path()+theme_bw()+geom_vector(aes(x=Bevals[1]*Bevecs$V1[1],y=Bevals[1]*Bevecs$V1[2],color=
"multiplied with eigenvalue"))+geom_vector(aes(Bevals[2]*Bevecs$V2[1],y=Bevals[2]*Bevecs$V2[2],color=
"multiplied with eigenvalue"))+geom_vector(aes(Bevecs$V1[1],y=Bevecs$V1[2],color="standardized eigenvector"))+
geom_vector(aes(Bevecs$V2[1],y=Bevecs$V2[2],color="standardized eigenvector"))+labs(color = "")+
scale_color_manual(values = c("standardized eigenvector"="black","multiplied with eigenvalue"=colors[3]))+
theme(legend.position = c(0.771,0.9))
```

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ VI

- **Question:** Can we do the same for square, but **non-symmetric** matrices?

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ VI

- **Question:** Can we do the same for square, but **non-symmetric** matrices?
- **Answer:** Not exactly. While their multiplication with every point on the unit circle *would definitely result in a* 2 *dimensional ellipsoid (ellipse)*, it's axes would no longer coincide with the eigenvectors multiplied with their corresponding eigenvalues.
- **Why?**

Transforming the unit circle with matrices in $\mathbb{R}^{2\times 2}$ VI

- **Question:** Can we do the same for square, but **non-symmetric** matrices?
- **Answer:** Not exactly. While their multiplication with every point on the unit circle *would definitely result in a* 2 *dimensional ellipsoid (ellipse)*, it's axes would no longer coincide with the eigenvectors multiplied with their corresponding eigenvalues.
- **Why?** Because the eigenvectors of a non-symmetric matrix are not orthogonal.

Outlook: We have already established that covariance matrices are, by definition, always symmetric!

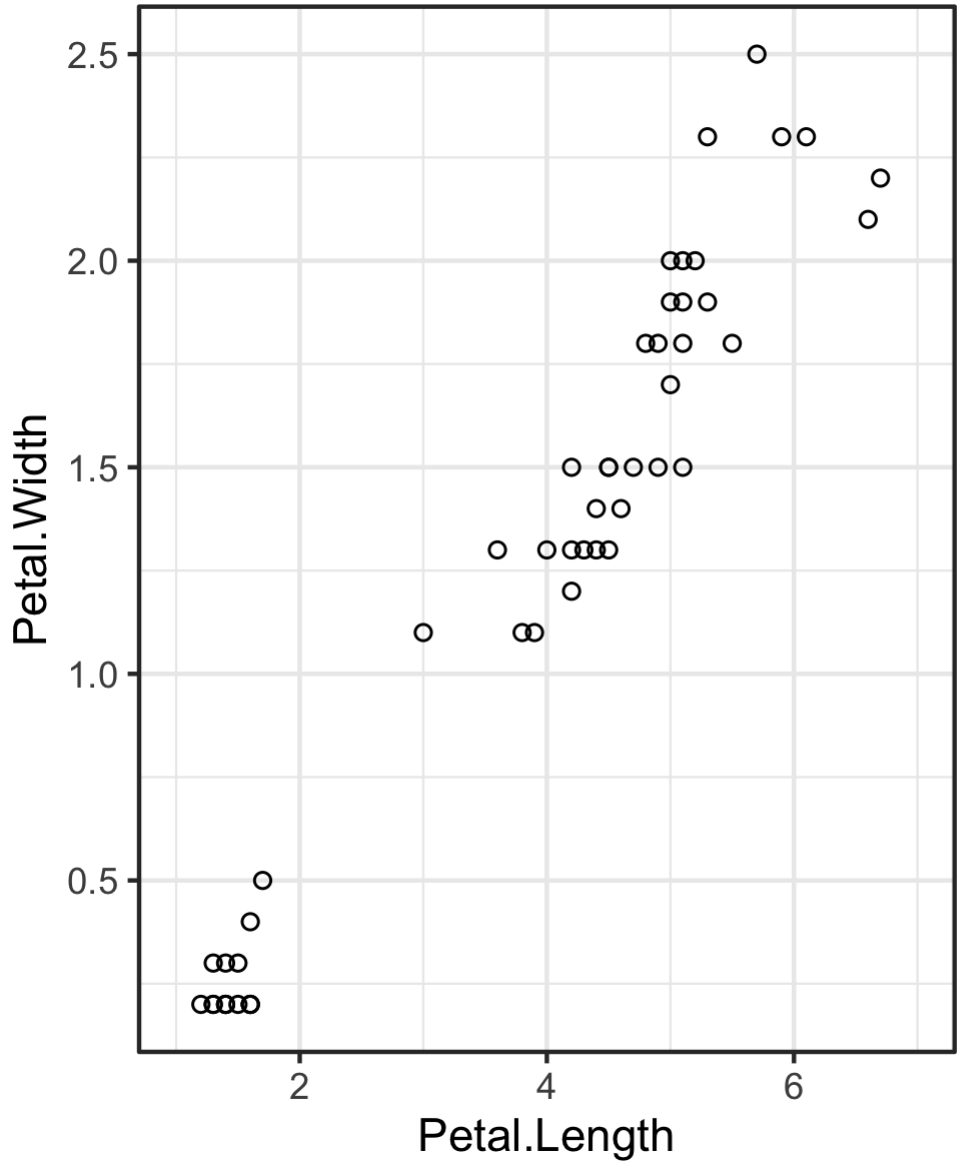
Explained and residual variance: Linear regression I

- The previous plot displayed a projection that minimized the distance in both directions. Let us instead consider linear regression with one regressor.

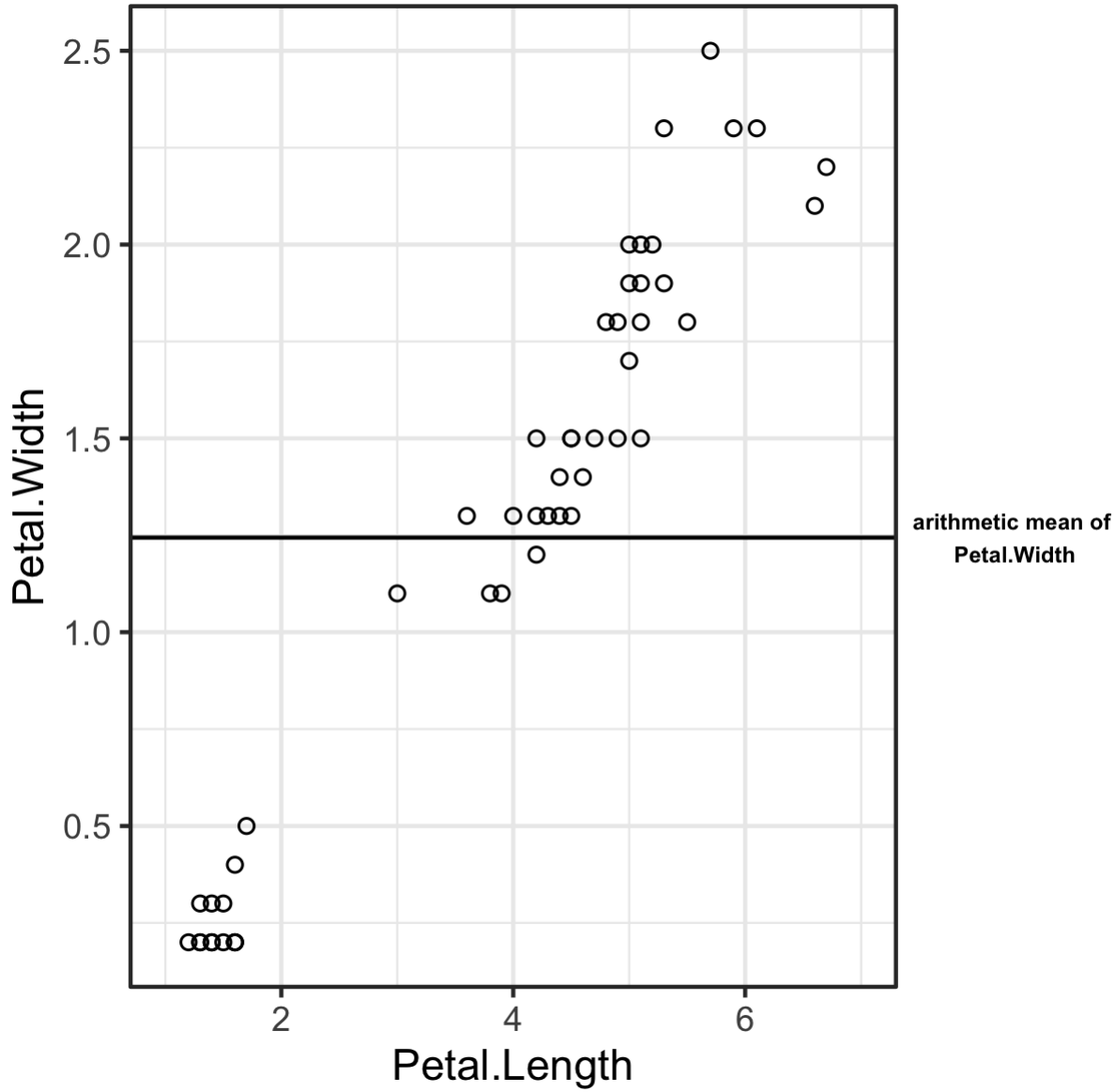
- Specifically, let us again consider the Iris-regression from the Linear Algebra lecture, where we fit a linear model with
  - `Petal.Width` as dependent and
  - `Petal.Length` as an independent variable.

```
library(ggplot2)
library(ggarchery)
set.seed(735)

data<-iris[sample(1:nrow(iris),10), c("Petal.Width","Petal.Length")]
lm(Petal.Width~Petal.Length,data=data)
```

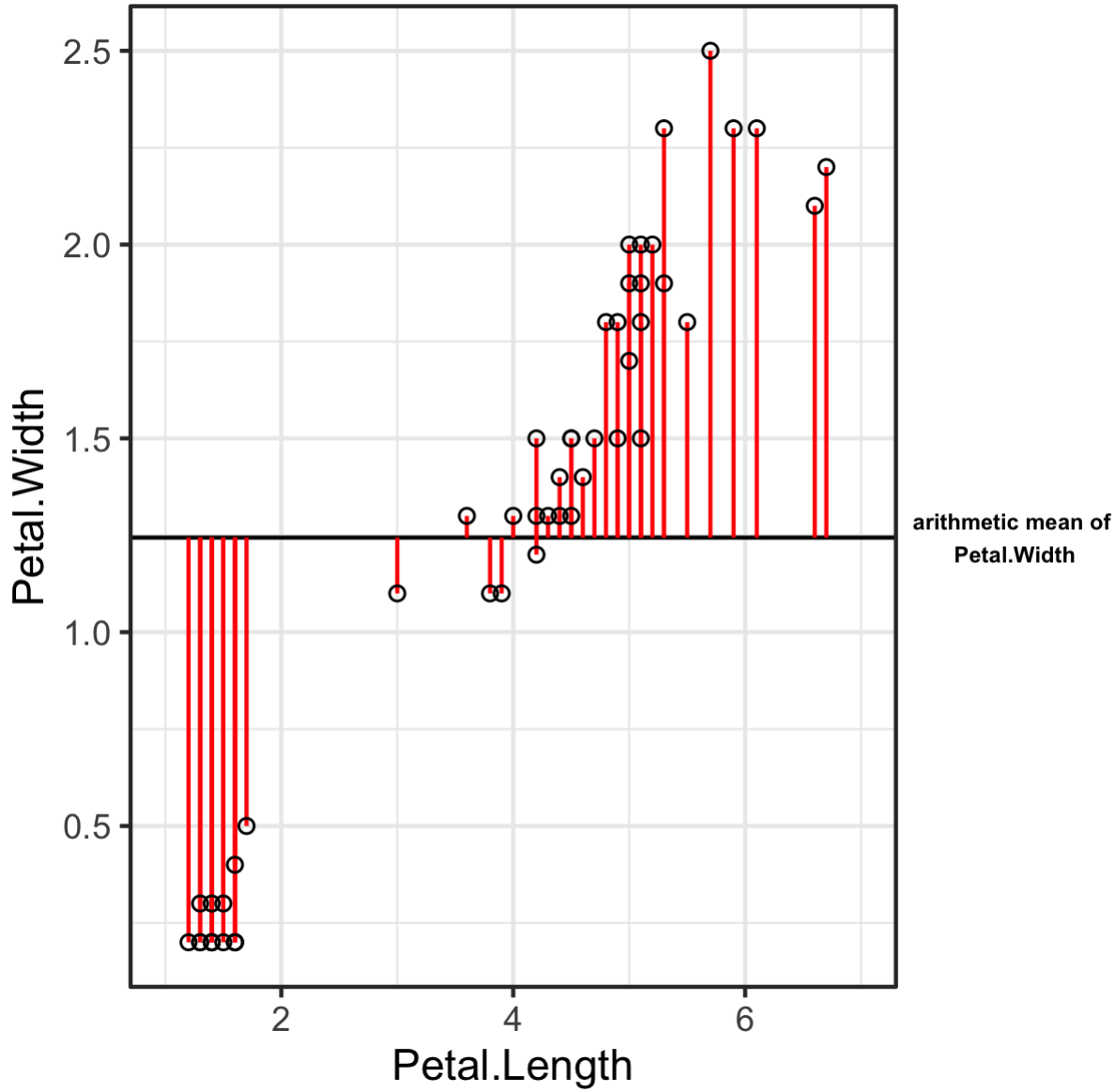
Explained and residual variance: Linear regression II



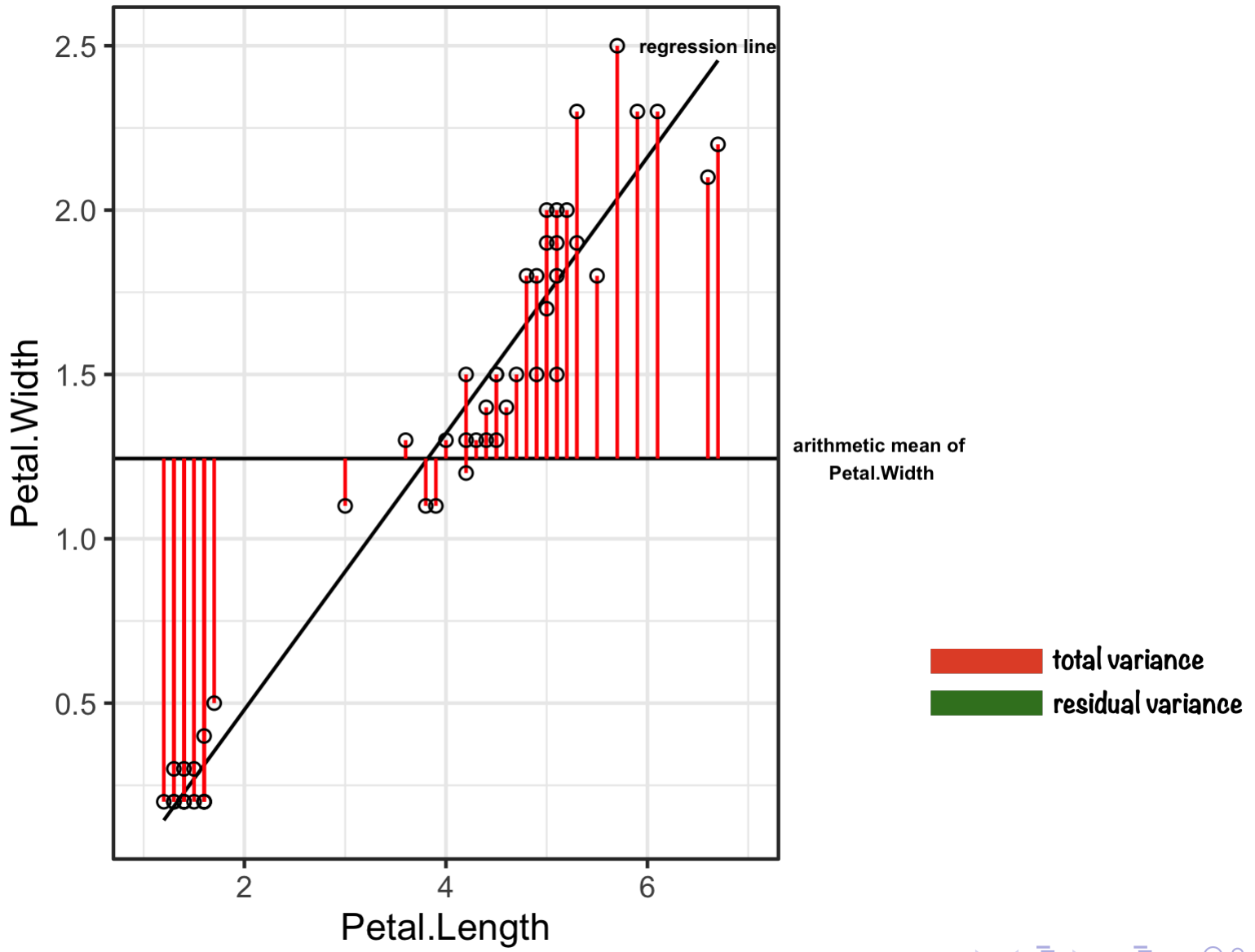
Explained and residual variance: Linear regression II



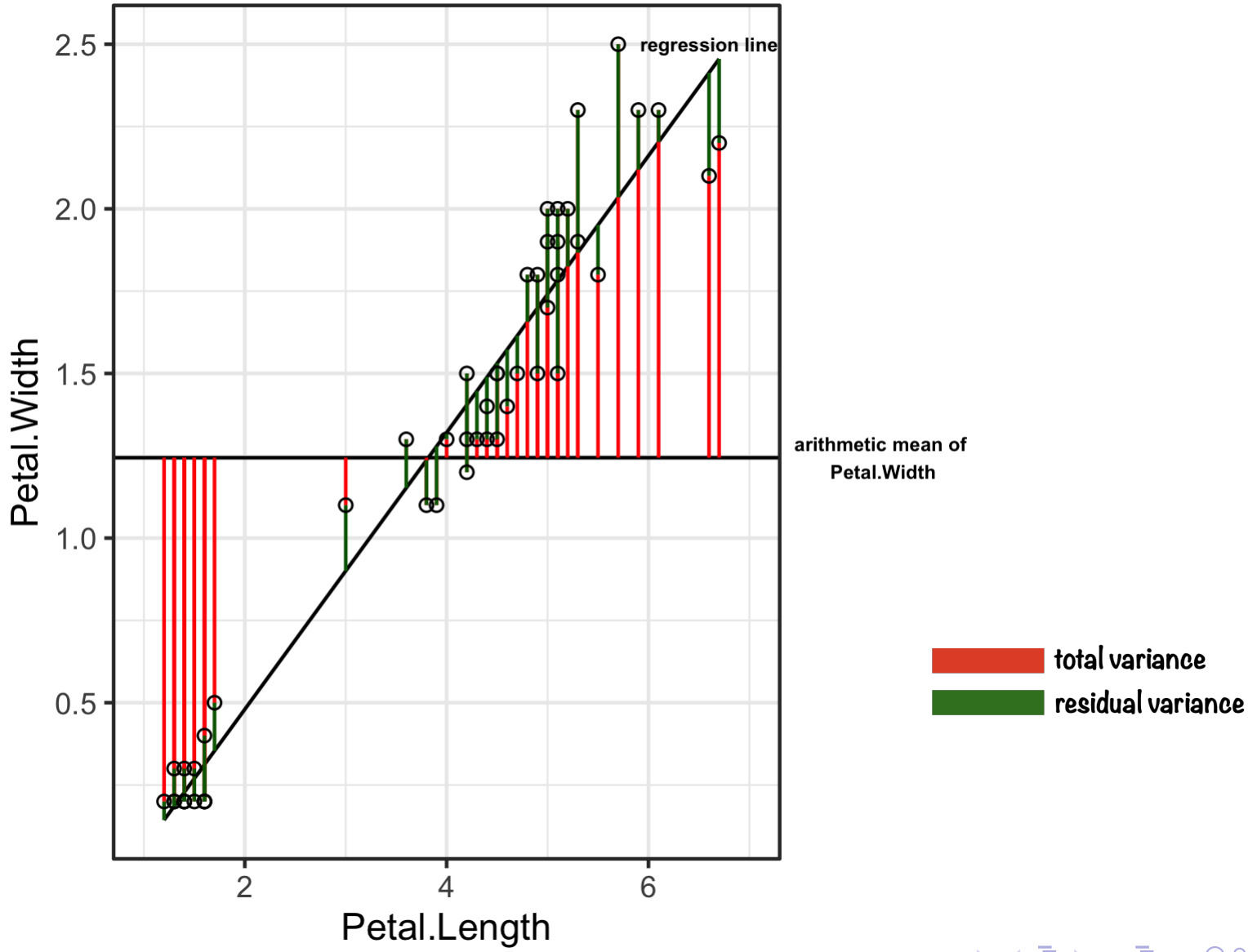
Explained and residual variance: Linear regression II



Explained and residual variance: Linear regression II



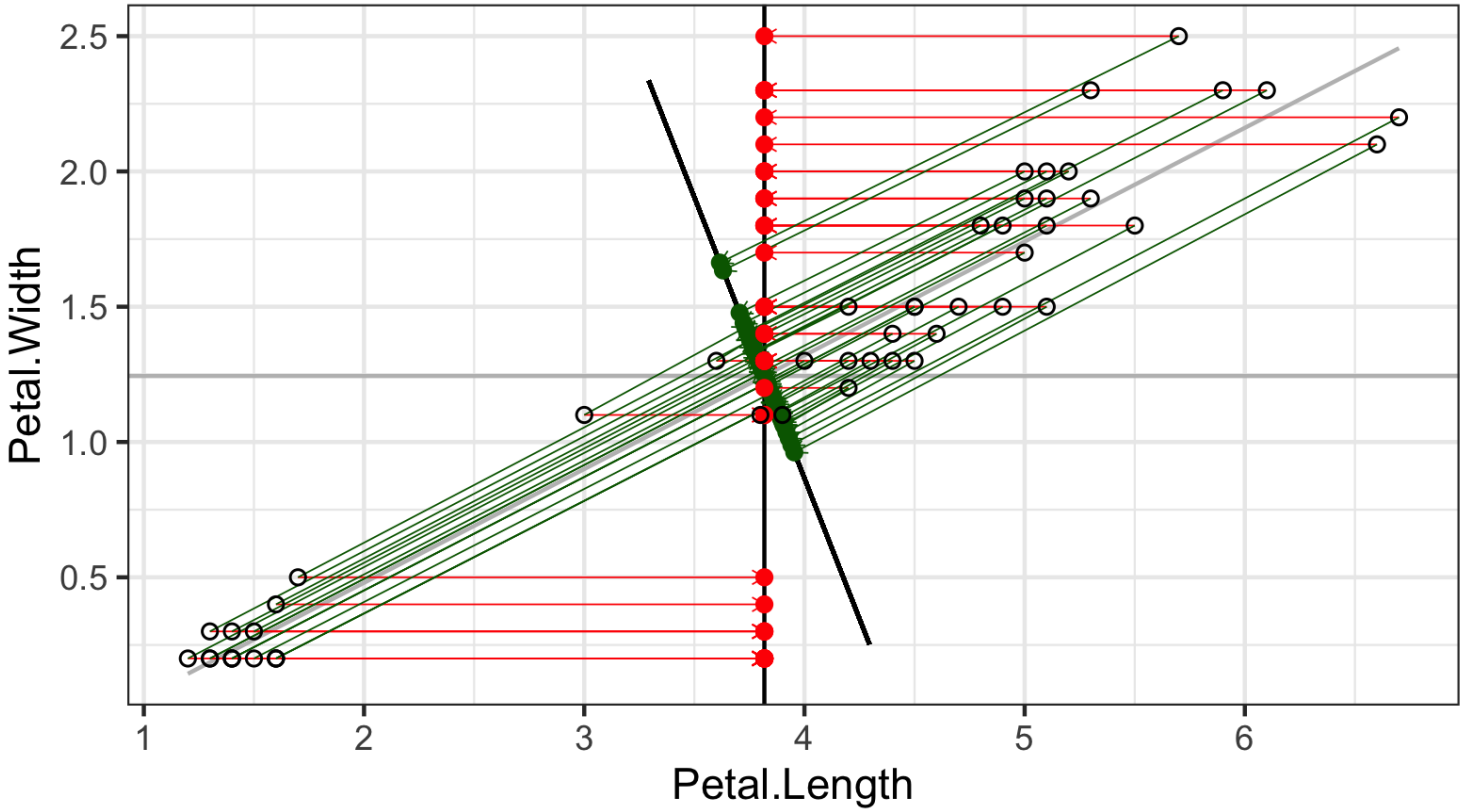
Explained and residual variance: Linear regression II



Explained and residual variance: Linear regression III

- Now, we could, e.g., clearly calculate $\frac{1}{n-1} \cdot RSS$ and $\frac{1}{n-1} \cdot TSS$ as the sample variance of the residual and total variances, respectively.

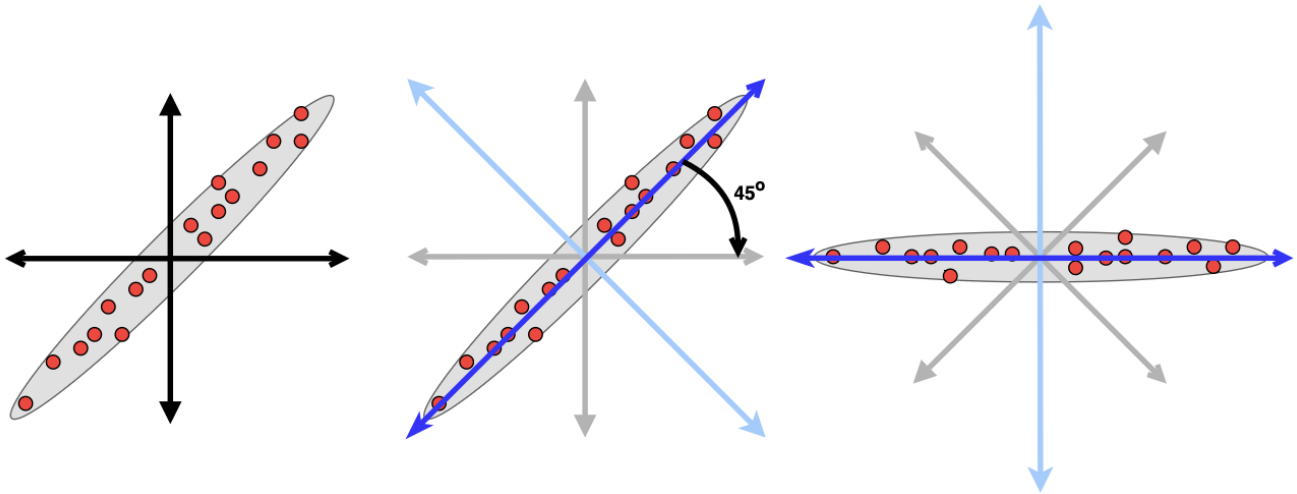
- To do so equivalently to slide ??, we need only orthogonally project our observations as follows:



Explained and residual variance: Linear regression IV

⟶ Given the mean of the target variable and the regression line as well as the explained- and residual- variances for every observation we can infer the coordinates of every point!

- If the horizontal line at the mean of the target variable and the regression line *were* orthogonal, we could clearly draw the a "point cloud" with the exact same structure but with different individual coordinates by using the continuation of these two lines as axes:



Source: https://www.baeldung.com/cs/principal-component-analysis

Standardizing Variables I

- Before we are finally ready for the definition of principal components, let us briefly consider the topic of standardization.

Reminder

For a real random variable X (i.e. X takes values in (a subset of) $\mathbb{R}$) with expected value $\mathbb{E}[X]$ and variance $\mathrm{Var}(X)$, the random variable

$$Z := \frac{X - \mathbb{E}[X]}{\sqrt{\mathrm{Var}(X)}}$$

has expected value $\mathbb{E}[Z] = 0$ and variance $\mathrm{Var}(Z) = 1$.

- This clearly holds because

$$\mathbb{E}[Z] = \mathbb{E}[\tfrac{X-\mathbb{E}[X]}{\sqrt{\mathrm{Var}(X)}}] = \sqrt{\mathrm{Var}(X)}^{-1}(\mathbb{E}[X] - \mathbb{E}[X]) = 0 \text{ and}$$

$$\mathrm{Var}[Z] = \mathrm{Var}[\tfrac{X-\mathbb{E}[X]}{\sqrt{\mathrm{Var}(X)}}] = \mathrm{Var}(X)^{-1}(\mathrm{Var}(X) - 0) = 1.$$

Standardizing Variables II

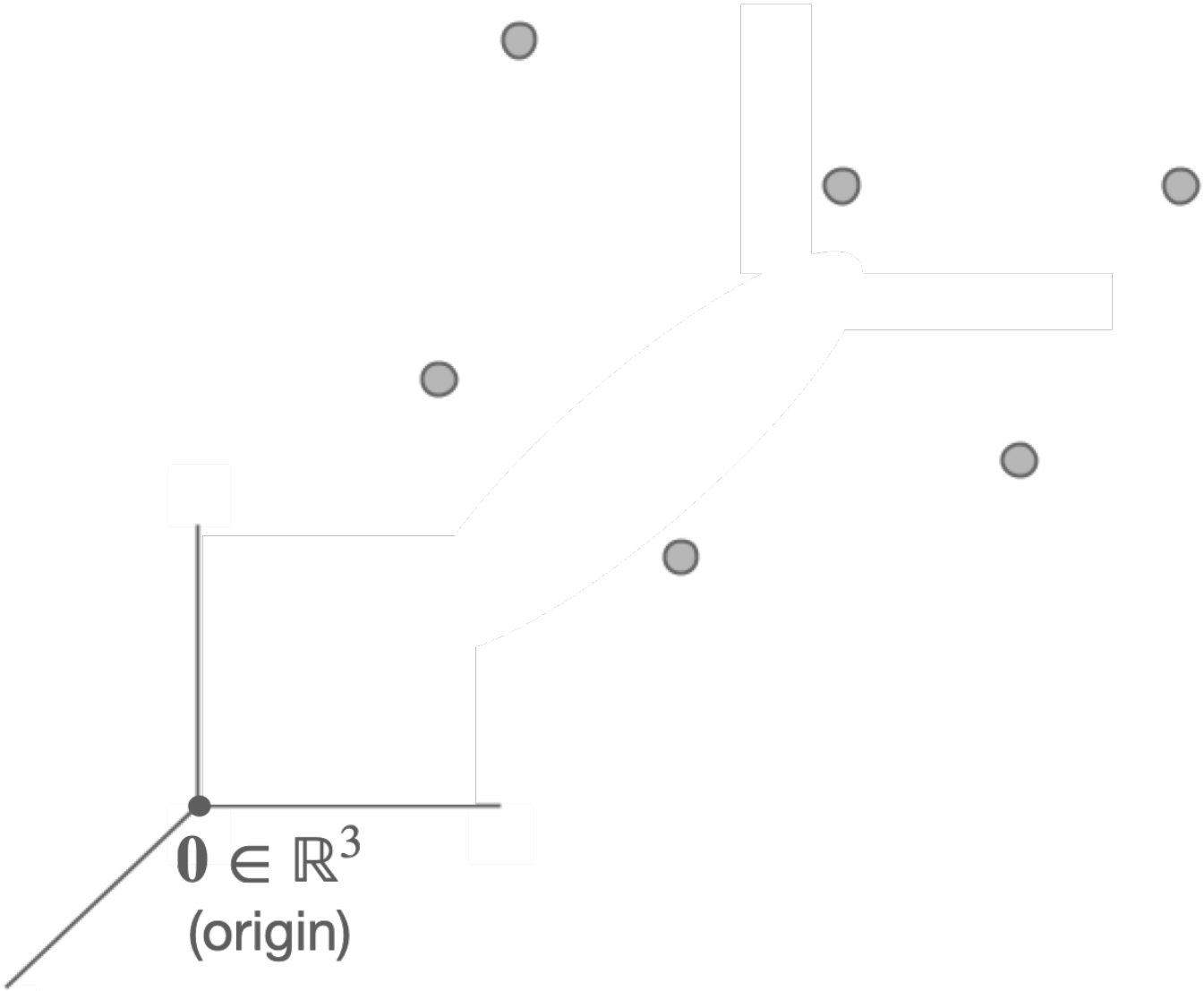
- Extending the intuition of this theoretical fact, we can standardize a sequence of data points $\{\boldsymbol{x}_i\}_{i=1,\dots,n}$, $\boldsymbol{x}_i \in \mathbb{R}^m$, $m, n \in \mathbb{N}$, by defining a sequence of standardized points $\{\tilde{\boldsymbol{x}}_i\}_{i=1,\dots,n}$ with, for $s_j^2 := \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x}_{\cdot j})^2$ denoting the sample variance of the jth data variable,

$$\tilde{\boldsymbol{x}}_i = (x_{i1}, \dots, x_{im})^\top \quad \text{with } \tilde{x}_{ij} := \frac{x_{ij} - \bar{x}_{\cdot j}}{s_j}\,,\ j = 1, \dots m\,.$$

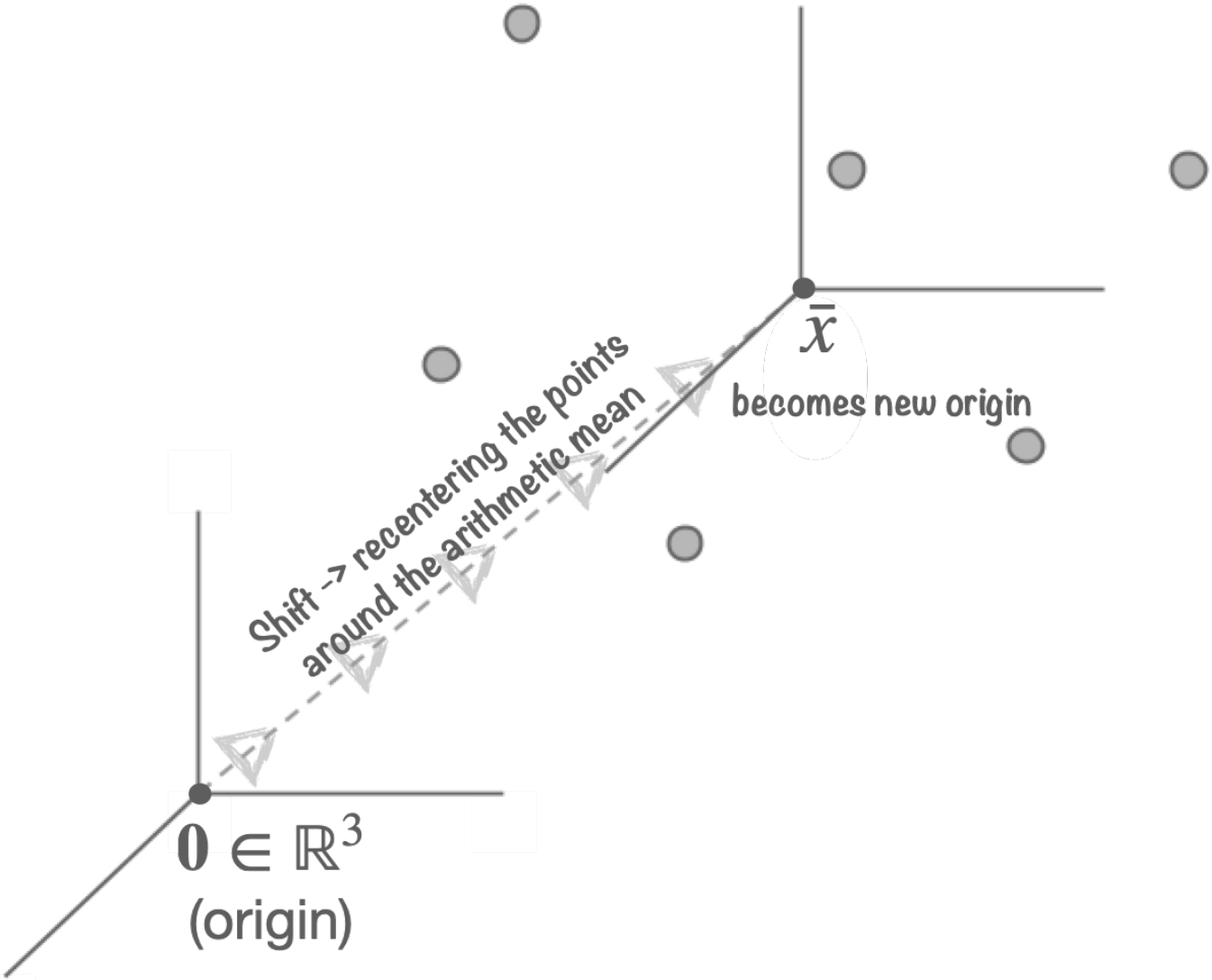
- We can break the above standardization down into two steps:
  1. Shifting each point by the arithmetic mean
  2. Scaling each shifted point by the sample standard deviation.

- Going back to the visualization of a "point cloud" this process can be thought of as *either* a shifting and scaling of the cloud *or* a shifting and relabeling of the axes:

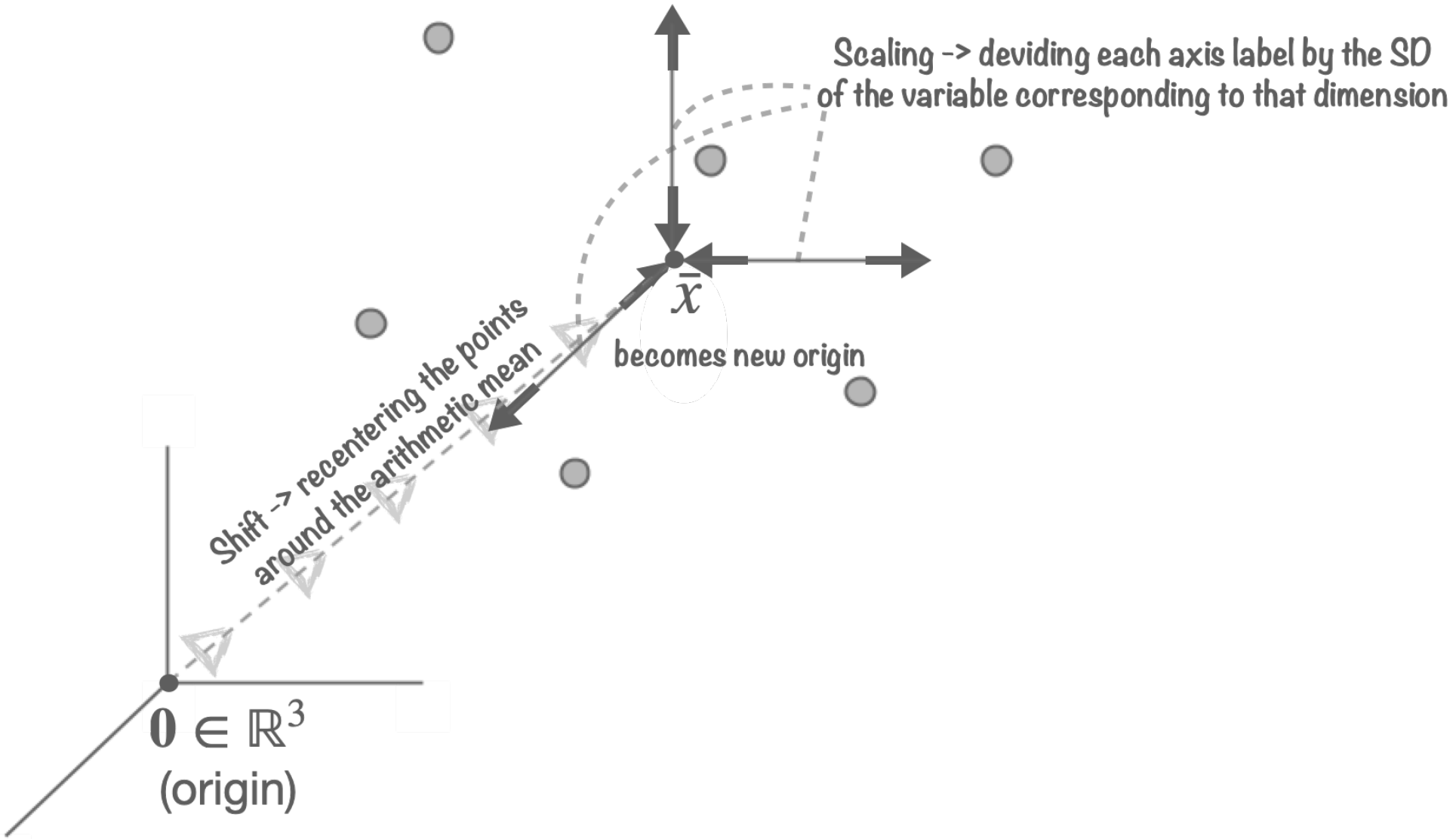
We recall: Standardizing data points



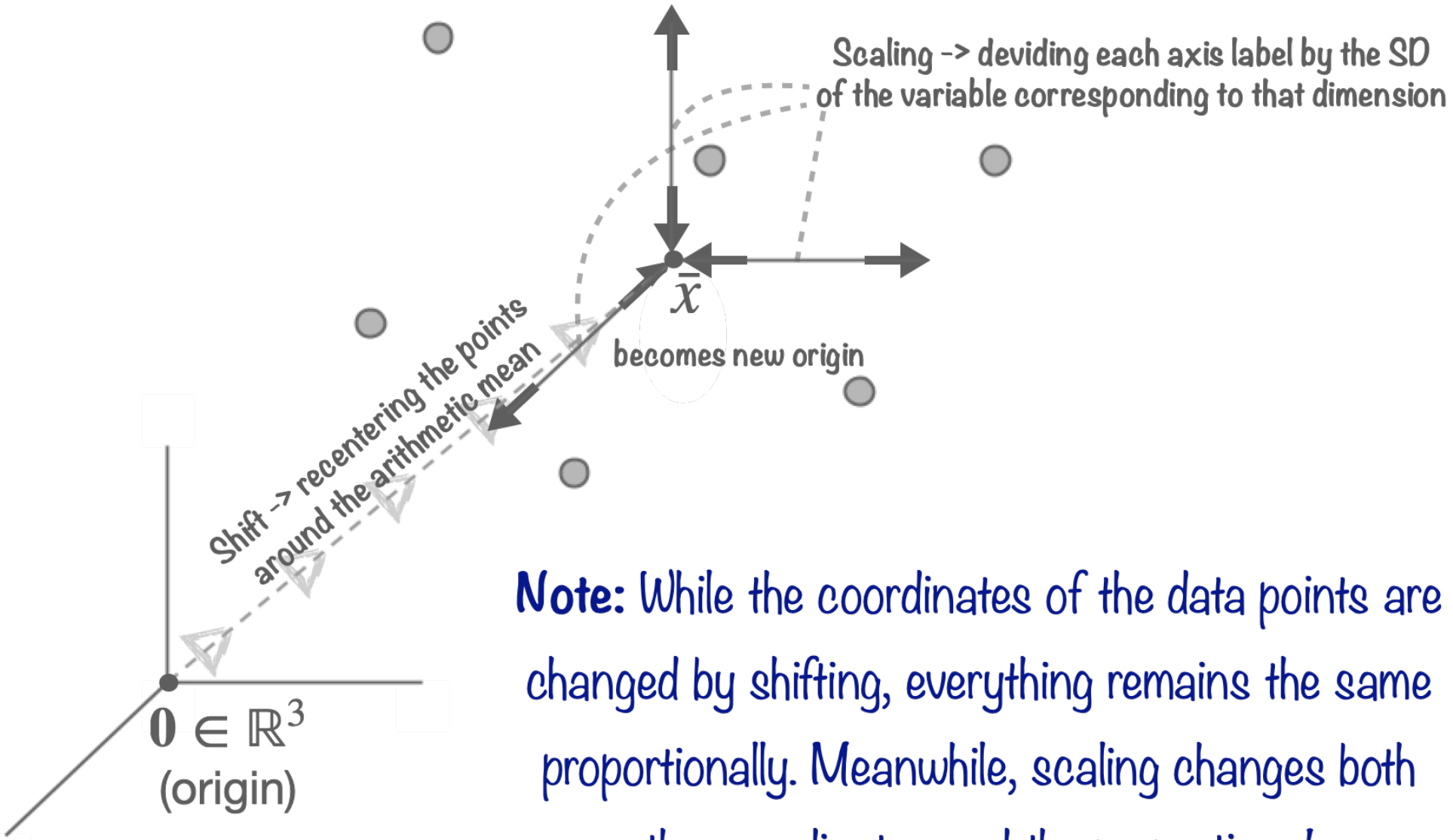
We recall: Standardizing data points



We recall: Standardizing data points



We recall: Standardizing data points



Note: While the coordinates of the data points are changed by shifting, everything remains the same proportionally. Meanwhile, scaling changes both the coordinates and the proportions!

Principal Components I

- Given a data-matrix $\boldsymbol{X} = \begin{pmatrix} x_{11} & \dots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \dots & x_{nm} \end{pmatrix} \in \mathbb{R}^{n \times m}, n, m \in \mathbb{N},$

  with rows given by the sequence of data points $\{\boldsymbol{x}_i\}_{i=1,\dots,n}$, $\boldsymbol{x}_i \in \mathbb{R}^m$

- If we can find m orthogonal vectors $\boldsymbol{a}_1, \dots, \boldsymbol{a}_m \in \mathbb{R}^m$, we have already learned how they each may be used both
  - as new axes of the m-dimensional Cartesian coordinate system
  - to calculate the data's variance in one of m directions after orthogonally projecting each data point on the line produced by the respective vector $\boldsymbol{a}_p$, $p \in \{1, \dots, m\}$.

Principal Components II

- Specifically, in PCA, we want to find such m orthogonal vectors iteratively, ordered by the proportion of the data's overall variance in that direction.

- Equivalently, we can say that the **first Principal Component** $\boldsymbol{a}_1$ is the vector of coefficients in a linear combination

$$\boldsymbol{a}^\top x = a_1 x_1 + a_2 x_2 + \cdots + a_m x_m = \sum_{j=1}^{m} a_j x_j \tag{1}$$

of the original variables that explains the most variance, the **second Principal Component** the vector of coefficients in above linear combination of the original variables that explains the second most variance, and so on.

Why not just a straightforward definition? I

- While algorithms to carry out principal component analysis are well documented and implemented in a huge variety of softwares, the literature is lacking in a clear and formally sound definition of Principal Components.

- For example, the book *Principal Component Analysis* by I.T. Jolliffe gives the following definition, in the setting of eq. 1, of PCs on pages 2 and 5:

  1. The kth derived variable, $\boldsymbol{a}_k^\top \boldsymbol{x}$ is the kth PC.
  2. To derive the form of the PCs, consider first $\boldsymbol{a}_1^\top \boldsymbol{x}$; the vector $\boldsymbol{a}_1$ maximizes $\mathrm{Var}(\boldsymbol{a}_1^\top \boldsymbol{x}) = \boldsymbol{a}_1^\top \boldsymbol{\Sigma} \boldsymbol{a}_1$.

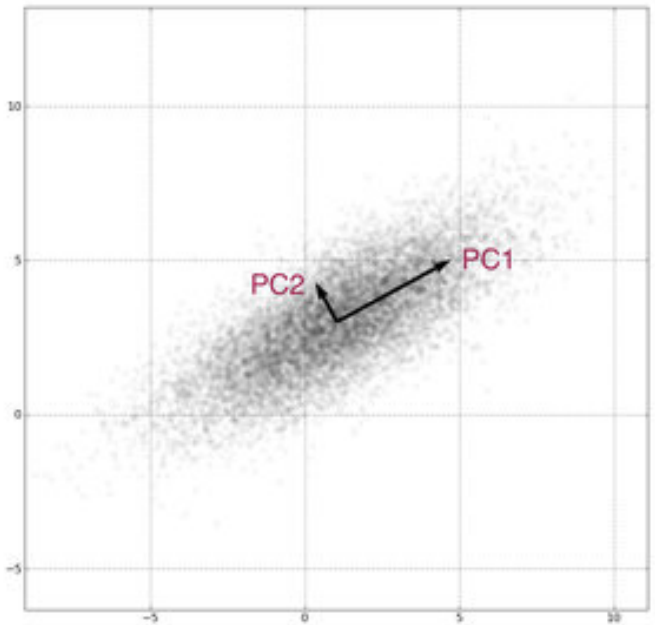
     (where $\boldsymbol{\Sigma}$ is the "known" covariance matrix of the random vector $X = (X_1, \ldots, X_m)^\top$, i.e. the vector of random variables of which we see each column of $\boldsymbol{X}$ as n realizations.)

Why not just a straightforward definition? II

Please note a few things:

1. $\boldsymbol{a}_k^\top \boldsymbol{x}$, whether considered as random or not, takes values in $\mathbb{R}$. As such, common PC plots such as on the right, would not make any sense if the PCs are defined as $\boldsymbol{a}_1^\top \boldsymbol{x}, \ldots, \boldsymbol{a}_m^\top \boldsymbol{x}$.

2. Considering the covariance matrix $\boldsymbol{\Sigma}$ of the random vector $X = (X_1, \ldots, X_m)^\top$ as known makes little sense, since we simply transforming data and not making any distributional assumptions.



Source: ResearchGate

Why not just a straightforward definition? III

3. Of course, the last issue is easily remedied by replacing the term

$$\mathrm{Var}(\boldsymbol{a}_k^\top \boldsymbol{x}) = \boldsymbol{a}_k^\top \boldsymbol{\Sigma} \boldsymbol{a}_k$$

with

$$\widehat{\mathrm{Var}}(\boldsymbol{a}_k^\top \boldsymbol{x}) = \boldsymbol{a}_k^\top \boldsymbol{S} \boldsymbol{a}_k \,,$$

where $\boldsymbol{S}$ denotes the sample covariance matrix, i.e. the matrix with the sample covariances as entries (and the sample variances on the diagonal).

4. Lastly, please note that viewing any elements as random variables in the context of PCA may legitimately be considered superfluous.

*Specifically, we could simply use the concepts of **inertia** and **center of gravity** instead of variance and mean to obtain all necessary information.*

Quick reminder: Sample covariance/correlation matrices I

- Recall that for each of the $m \in \mathbb{N}$ variables (or columns) of given data, the **sample variance** is given by

$$s_j^2 := \frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{x}_{\cdot j})^2 \, , j = 1, \ldots, m$$

and the **sample covariance** between the jth and kth variable/column by

$$s_{kj} := \frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{x}_{\cdot j}) (x_{ik} - \bar{x}_{\cdot k}) \, , \quad 1 \leq k, j \leq m, j \neq k.$$

Quick reminder: Sample covariance/correlation matrices II

- And the **sample covariance matrix**, which we denote by $\boldsymbol{S}$ but is often still denoted by $\boldsymbol{\Sigma}$, is defined as

$$\boldsymbol{S} = \begin{bmatrix} s_1^2 & s_{12} & \dots & s_{1m} \\ s_{21} & s_2^2 & \dots & s_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ s_{m1} & s_{m2} & \dots & s_m^2 \end{bmatrix} \in \mathbb{R}^{m \times m}.$$

- It then immediately follows that:

Quick reminder: Sample covariance/correlation matrices III

$$\boldsymbol{S} = \frac{1}{n-1}\sum_{i=1}^{n}\begin{pmatrix} (x_{i1}-\bar{x}_{\cdot 1})^2 & \cdots & (x_{i1}-\bar{x}_{\cdot 1})(x_{im}-\bar{x}_{\cdot m}) \\ \vdots & \ddots & \vdots \\ (x_{im}-\bar{x}_{\cdot m})(x_{i1}-\bar{x}_{\cdot 1}) & \cdots & (x_{im}-\bar{x}_{\cdot m})^2 \end{pmatrix}$$

$$= \frac{1}{n-1}\sum_{i=1}^{n}\begin{pmatrix} (x_{i1}-\bar{x}_{\cdot 1}) \\ \vdots \\ (x_{im}-\bar{x}_{\cdot m}) \end{pmatrix}\begin{pmatrix} (x_{i1}-\bar{x}_{\cdot 1}) & \ldots & (x_{im}-\bar{x}_{\cdot m}) \end{pmatrix}$$

$$= \frac{1}{n-1}\sum_{i=1}^{n}(\bar{\boldsymbol{x}}_{\boldsymbol{i}}-\bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_{\boldsymbol{i}}-\bar{\boldsymbol{x}})^{\top}.$$

$\Rightarrow$ Let $\boldsymbol{X}_C$ denote the matrix of the data points shifted by the arithmetic mean. It then follows that $\boldsymbol{S} = \frac{1}{n-1}\boldsymbol{X}_C^{\top}\boldsymbol{X}_C$.

Quick reminder: Sample covariance/correlation matrices IV

- Recall that for two of the $m \in \mathbb{N}$ variables (or columns) of given data, the **sample correlation** is given by, for $i, j \in \{1, \ldots, m\}$,

$$r_{ij} := \begin{cases} 1\,, & \text{if } i = j, \\ \dfrac{s_{ij}}{\sqrt{s_i^2 s_j^2}}\,, & \text{otherwise.} \end{cases}$$

- And the **sample correlation matrix**, which we denote by $\boldsymbol{R}$, is defined as

$$\boldsymbol{R} = \begin{bmatrix} 1 & r_{12} & \ldots & r_{1m} \\ r_{21} & 1 & \ldots & r_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{m1} & r_{m2} & \ldots & 1 \end{bmatrix} \in \mathbb{R}^{m \times m}\,.$$

Determining the principal component vectors I

Regardless of the arguable lack of formal definition, the following is clear:

The m Principal Component (vectors) $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ are derived...

... by **iteratively maximizing**

$$\boldsymbol{a}_p^\top \boldsymbol{S} \boldsymbol{a}_p \,, \quad p = 1, \ldots, m \,.$$

under the following constraints:

1. $\boldsymbol{a}_p$ is normalized, i.e. $\boldsymbol{a}_p^\top \boldsymbol{a}_p = 1$

2. a_p is orthogonal to all previous PCs, i.e.

$$\boldsymbol{a}_p^\top \boldsymbol{a}_j = 0 \,, \quad p = 2, \ldots, m \,; \; j = 1, \ldots, p-1.$$

(Clearly this leads to a set of PCs that are all orthogonal.)

Determining the principal component vectors II

- **Iteratively** solving $\underset{\boldsymbol{a} \in \mathbb{R}^{\boldsymbol{m}}}{\arg\max}\ \boldsymbol{a}_p^\top \boldsymbol{S} \boldsymbol{a}_p$ under these constraints may be achieved using one of the following three ways:

  1. the method of Lagrange multipliers
  2. Eigendecomposition of $\boldsymbol{S}$
  3. Singular value decomposition of the centered data matrix $\boldsymbol{X}_C$

PCA via Lagrange Multipliers I

- First things first: **The method of Lagrange multipliers is a strategy for finding the local Extrema of a function subject to equation constraints.** Formally this can be expressed as follows:

- (See p.285 of *Fuente, A. (2000). Mathematical Methods and Models for Economists. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511810756* for more.)

PCA via Lagrange Multipliers II

The method of Lagrange Multipliers

Consider, for two twice continuously differentiable functions $f : \mathbb{R}^m \supseteq X \longrightarrow \mathbb{R}^c$ and $g : \mathbb{R}^m \supseteq X \longrightarrow \mathbb{R}$, with $m, c \in \mathbb{N}$ and $c \leq m$, the optimization problem

$$\max_{x \in \mathbb{R}^m} \{ f(x) \text{ s.t. } g(x) = \mathbf{0} \}.$$

Let x^* be an optimal solution to the above optimization problem such that $\operatorname{rank}(\mathrm{D}\, g(x^*)) = c$. Then there exists a unique Lagrange multiplier $\lambda^* \in \mathbb{R}^c$ such that

$$\mathrm{D}\, f(x^*) = \lambda^{*\top}\, \mathrm{D}\, g(x^*).$$

(where $\mathrm{D}\, g(x^*)$ and $\mathrm{D}\, g(x^*)$ denote the matrices of partial derivatives, $\left[\mathrm{D}\, g(x^*)\right]_{j,k} = \left[\frac{\partial g_j}{\partial x_k}\right]$ and $\left[\mathrm{D}\, g(x^*)\right]_{j,k} = \left[\frac{\partial f_j}{\partial x_k}\right]$,respectively.)

PCA via Lagrange Multipliers III

- Applying Lagrange multipliers to the iterative maximization problem of slide 35 gives:

Determining the 1. PC $\boldsymbol{a}_1^\top =$

maximizing $\boldsymbol{a}_1^\top \boldsymbol{S}\boldsymbol{a}_1$ under the constraint $\boldsymbol{a}_1^\top \boldsymbol{a}_1 = 1$:

$$L(\boldsymbol{a}_1) = \boldsymbol{a}_1^T \boldsymbol{S}\boldsymbol{a}_1 - \lambda(\boldsymbol{a}_1^\top \boldsymbol{a}_1 - 1) \to \max$$

$$\Leftrightarrow \frac{\partial L}{\partial \boldsymbol{a}_1} = 2\boldsymbol{S}\boldsymbol{a}_1 - 2\lambda \boldsymbol{a}_1 \stackrel{!}{=} 0$$

$$\Rightarrow (\boldsymbol{S} - \lambda I)\boldsymbol{a}_1 \stackrel{!}{=} 0$$

Determining the 2. PC $\boldsymbol{a}_2^\top =$

maximizing $\boldsymbol{a}_2^\top \boldsymbol{S}\boldsymbol{a}_2$ under the constraints $\boldsymbol{a}_2^\top \boldsymbol{a}_2 = 1$ **and** $\boldsymbol{a}_2^\top \boldsymbol{a}_1 = 0$:

$$L(\mathbf{a}_2) = \mathbf{a}_2^\top \boldsymbol{\Sigma}\mathbf{a}_2 - \lambda_1 (\mathbf{a}_2^\top \mathbf{a}_2 - 1) - \lambda_2 (\mathbf{a}_1^\top \mathbf{a}_2) \to \max$$

$$\Leftrightarrow \frac{\partial L}{\partial \boldsymbol{a}_2} = 2\boldsymbol{S}\boldsymbol{a}_2 - 2\lambda_1 \boldsymbol{a}_2 - 2\lambda_2 \boldsymbol{a}_1 \stackrel{!}{=} 0$$

$\Rightarrow$ Choosing $\boldsymbol{\lambda} = (\lambda_1, \lambda_2)^\top = \begin{pmatrix} \lambda \\ 0 \end{pmatrix}$ we get

$$(\boldsymbol{S} - \lambda I)\boldsymbol{a}_2 \stackrel{!}{=} 0$$

and so on for the 3rd to mth PCs...

PCA via Eigendecomposition I

- Considering the last equations to be solved in the previous slide were of the form $(\boldsymbol{S} - \lambda I)\boldsymbol{a}_i \stackrel{!}{=} 0$, it stands to reason that the PCs that solve

$$\boldsymbol{a}_p^\top \boldsymbol{S} \boldsymbol{a}_p \to \max$$

  are just the eigenvectors of the sample covariance matrix $\boldsymbol{S}$.

- Specifically, we can derive the m Principal Component (vectors) $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ as follows:

PCA via Eigendecomposition II

Eigendecomposition of $\boldsymbol{S}$ with eigenvalues ordered by size

Consider an Eigendecomposition of the sample covariance matrix

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}_{\text{ord}}\boldsymbol{V}^{-1},$$

where $\boldsymbol{\Lambda}_{\text{ord}}$ denotes a diagonal $m \times m$ matrix with the largest to smallest eigenvalues of $\boldsymbol{S}$ on the diagonal, i.e. $\lambda_{11} \geq \lambda_{22} \geq \ldots \geq \lambda_{mm}$.

The m Principal Component (vectors) $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ are then given by the columns of the matrix $\boldsymbol{V}$, i.e.

$$\boldsymbol{a}_i = \begin{pmatrix} v_{1i} \\ \vdots \\ v_{mi} \end{pmatrix} \quad \forall i \in \{1, \ldots, m\}.$$

PCA via Eigendecomposition III

An alternative motivation: Mean-centered ellipse

- Note that since the eigenvectors of the sample covariance matrix are orthogonal per definition, we have $\boldsymbol{V}^{-1} = \boldsymbol{V}^\top$ in the previous decomposition and, therefore,

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}_{\text{ord}}\boldsymbol{V}^{-1} = \boldsymbol{V}\boldsymbol{\Lambda}_{\text{ord}}\boldsymbol{V}^\top = \sum_{j=1}^{m} \lambda_{jj}\boldsymbol{v}_{\cdot j}\boldsymbol{v}_{\cdot j}^\top$$

- Furthermore,

$$\boldsymbol{S}^{-1} = \boldsymbol{V}\boldsymbol{\Lambda}_{\text{ord}}^{-1}\boldsymbol{V}^\top = \sum_{j=1}^{m} \frac{1}{\lambda_{jj}}\boldsymbol{v}_{\cdot j}\boldsymbol{v}_{\cdot j}^\top .$$

PCA via Eigendecomposition IV

An alternative motivation: Mean-centered ellipse

- For some $r \in \mathbb{R}$, the set

$$\left\{ x \in \mathbb{R}^m : (x - \bar{x})^\top \boldsymbol{S}^{-1} (x - \bar{x}) = r^2 \right\}$$

  gives the surface of an m-dimensional ellipsoid.

- This ellipsoid is equal to the one that results from multiplying an m-ball (circle when $m = 2$) with radius r around the origin with the sample covariance matrix $\boldsymbol{S}$.

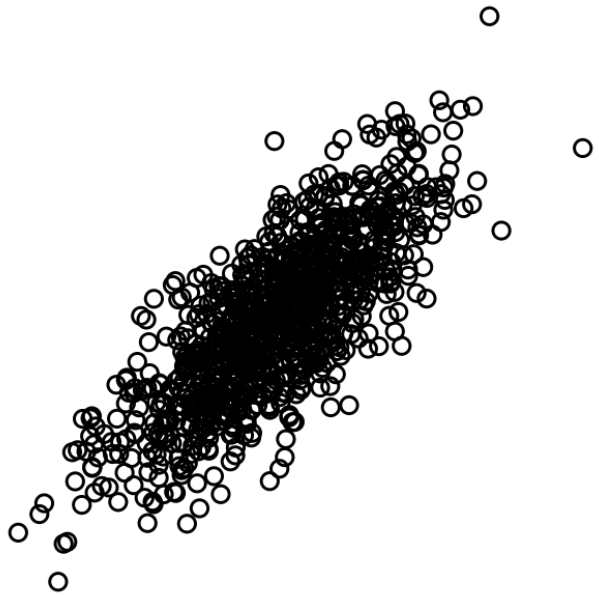
- The axes of such an ellipsoid are then given by $r\lambda_{11}\boldsymbol{a}_1, \ldots, r\lambda_{mm}\boldsymbol{a}_m$, i.e. the eigenvectors of $\boldsymbol{S}$ multiplied with the corresponding eigenvalues and r.

PCA via Eigendecomposition V

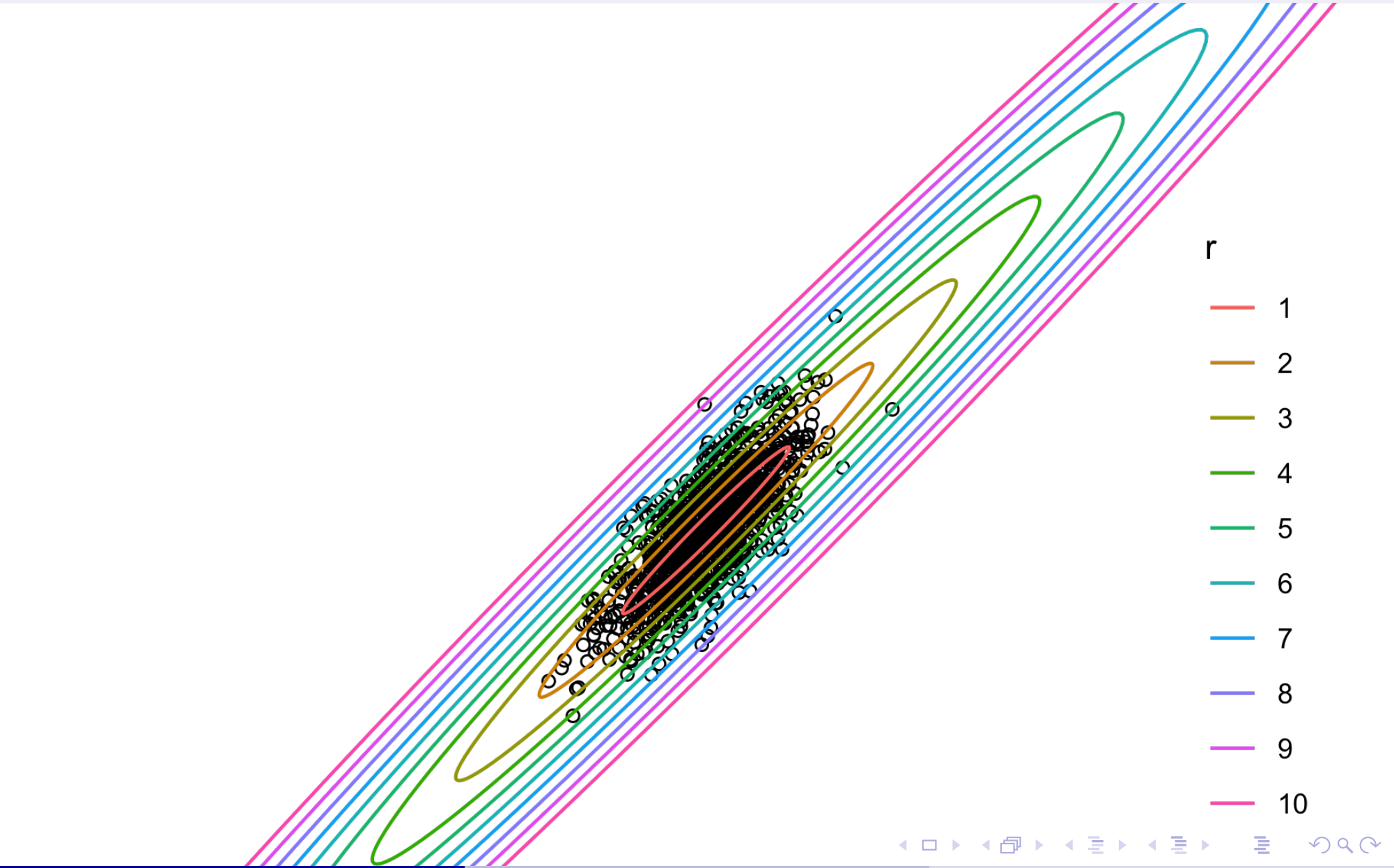
This is nicely visualized in $\mathbb{R}^2$ using R:

```
library(mvtnorm)
circle <- function(center = c(0,0),r = 1, npoints = 100){
  tt <- seq(0,2*pi,length.out = npoints)
  xx <- center[1] + r * cos(tt)
  yy <- center[2] + r * sin(tt)
  return(data.frame(x1 = xx, x2 = yy))
}

set.seed(4321)
example_dat<-as.data.frame(rmvnorm(1000,mean=c(12,10),sigma=matrix(c(3,1,1,0.5),ncol=2)))
S<-cov(example_dat)

circs<-do.call("rbind",lapply(as.list(seq(1,10,by=1)),function(x){cbind(circle(center=c(0,0),
        r=as.numeric(x)),r=x,npoints=100)}))

p<-ggplot(circs)+labs(x="x1",y="x2")+
  geom_point(aes(example_dat$V1-colMeans(example_dat)[1],example_dat$V2-colMeans(example_dat)[2]),shape=1)+
  theme_void()
p

p+geom_path(aes(apply(cbind(x1,x2),1,function(x)S%*%x)[1,],
                apply(cbind(x1,x2),1,function(x)S%*%x)[2,],color=as.factor(r)))+
  labs(color="r")+theme(legend.position = c(0.8,0.33))
```

PCA via Eigendecomposition VI



PCA via Eigendecomposition VI



PCA via SVD I

- Finally, let $\boldsymbol{X}_C$ denote the matrix of the data points shifted by the arithmetic mean and consider its singular value decomposition

$$\boldsymbol{X}_C = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top .$$

- It clearly follows that

$$\boldsymbol{S} = \frac{\boldsymbol{X}_C^\top \boldsymbol{X}_C}{n-1} = \frac{\left(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top\right)^\top \left(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top\right)}{n-1}$$

$$= \frac{\boldsymbol{V}\boldsymbol{\Sigma}^\top \boldsymbol{U}^\top \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top}{n-1}$$

and, since the singular values are always ordered in SVD, the m Principal Component (vectors) $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ are again given by the

PCA via SVD II

columns of the matrix $\boldsymbol{V}$, i.e.

$$\boldsymbol{a}_i = \begin{pmatrix} v_{1i} \\ \vdots \\ v_{mi} \end{pmatrix} \quad \forall i \in \{1, \ldots, m\}\,.$$

- **Thereby, we have shown that the PCs may also be derived directly via the singular value decomposition of the centered matrix $\boldsymbol{X}_C$.**

- In practice, this is only relevant for computational reasons, since for data with extremely many observations, the SVD solution saves the expense of having to compute a sample covariance matrix.

Standardized version I

- Importantly, **PCA is NOT scale-invariant!**
- In cases where the variables/columns/features take values in vastly different ranges, it therefore makes sense to standardize all variables instead of only centering them.
- Luckily, the procedure stays the same:

  Since the sample correlation matrix $\boldsymbol{R}$ already gives us the sample variance of the standardized variables, the issue simply becomes $\boldsymbol{a}_p^\top \boldsymbol{R} \boldsymbol{a}_p \to \max$, $p = 1, \ldots, m$.

Standardized version II

- **Iteratively** solving $\underset{\boldsymbol{a}\in\mathbb{R}^{\boldsymbol{m}}}{\arg\max}\ \boldsymbol{a}_p^\top \boldsymbol{R}\boldsymbol{a}_p$ under the same constraints may equivalently be achieved using one of the following three ways:

1. the method of Lagrange multipliers
2. Eigendecomposition of $\boldsymbol{R}$
3. Singular value decomposition of the standardized data matrix $\boldsymbol{X}_Z$, i.e. the matrix with entries

$$\left(\boldsymbol{X}_Z\right)_{ij} = \frac{x_{ij} - \bar{x}_{\cdot j}}{s_j}$$

Dimension reduction via PCA:
Projecting onto the first $k < m$ PCs I

- Once the matrix $\boldsymbol{V} \in \mathbb{R}^{m \times m}$ containing the 1st to mth PCs as columns has been determined and we have chosen a $k \in \mathbb{N}$ with $k < m$ as the "goal dimension", the rest is very straightforward:

Dimension reduction via PCA

Consider the matrix $\boldsymbol{V}_K = \begin{pmatrix} v_{11} & \dots & v_{1k} \\ \vdots & \vdots & \vdots \\ v_{m1} & \dots & v_{mk} \end{pmatrix} \in \mathbb{R}^{m \times k}$, which is the matrix of the first k columns of $\boldsymbol{V}$ (i.e. PCs of $\boldsymbol{X}$).

The data matrix with reduced dimension k is then given by

$$\widetilde{\boldsymbol{X}} = \boldsymbol{X}\boldsymbol{V}_K \quad \in \mathbb{R}^{n \times k}.$$

Dimension reduction via PCA:
Projecting onto the first $k < m$ PCs II

- **Different criteria for choosing k appropriately will be discussed in section 4.**

- For now, note that the entries of the data matrix $\widetilde{\boldsymbol{X}}$ with n observations in the lower dimension k are given by

$$\widetilde{x}_{ij} := \sum_{l=1}^{m} x_{il}\, \mathrm{v}_{lj}, \quad i = 1, \ldots, n;\ j = 1, \ldots, k\,.$$

- This is equivalent to an orthogonal projection of each data points onto the plane spanned by the first k PCs.

Contents

Proportion of variance explained I

- Recall that
  - The *trace* of a Matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, $n \in \mathbb{N}$, with entries $(a_{ij})_{i,j=1,\ldots,n}$ is defined as the sum of the diagonal entries, i.e.

$$\operatorname{tr}(\boldsymbol{A}) := \sum_{i=1}^{n} a_{ii}$$

  - and, for two matrices $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ and $\boldsymbol{B} \in \mathbb{R}^{\boldsymbol{m} \times \boldsymbol{n}}$, $n, m \in \mathbb{N}$

$$\operatorname{tr}(\boldsymbol{AB}) = \operatorname{tr}(\boldsymbol{BA}) .$$

- Since the diagonal entries of the sample covariance matrix $\boldsymbol{S}$ are the sample variances of the m variables/columns/features, the sum of sample variances is equal to $\operatorname{tr}(\boldsymbol{S})$.

Proportion of variance explained II

- In the context of PCA the sum of sample variances is sometimes also referred to as *total variance* or *overall variance* and is **also** equal to the sum of eigenvalue of the sample covariance matrix $\boldsymbol{S}$, since

$$\operatorname{tr}(\boldsymbol{S}) = \operatorname{tr}(\boldsymbol{V}\boldsymbol{\Lambda}_{\text{ord}}\boldsymbol{V}^\top) = \operatorname{tr}(\boldsymbol{V}^\top\boldsymbol{V}\boldsymbol{\Lambda}_{\text{ord}}) = \operatorname{tr}(\boldsymbol{\Lambda}_{\text{ord}}) = \sum_{i=1}^{m} \lambda_i \,.$$

- For the first $k \in \mathbb{N}$ PCs, $k \leq m$, the **proportion of variance explained** is defined as

$$\frac{\sum_{i=1}^{k} \lambda_i}{\operatorname{tr}(\boldsymbol{\Lambda}_{\text{ord}})} \,,$$

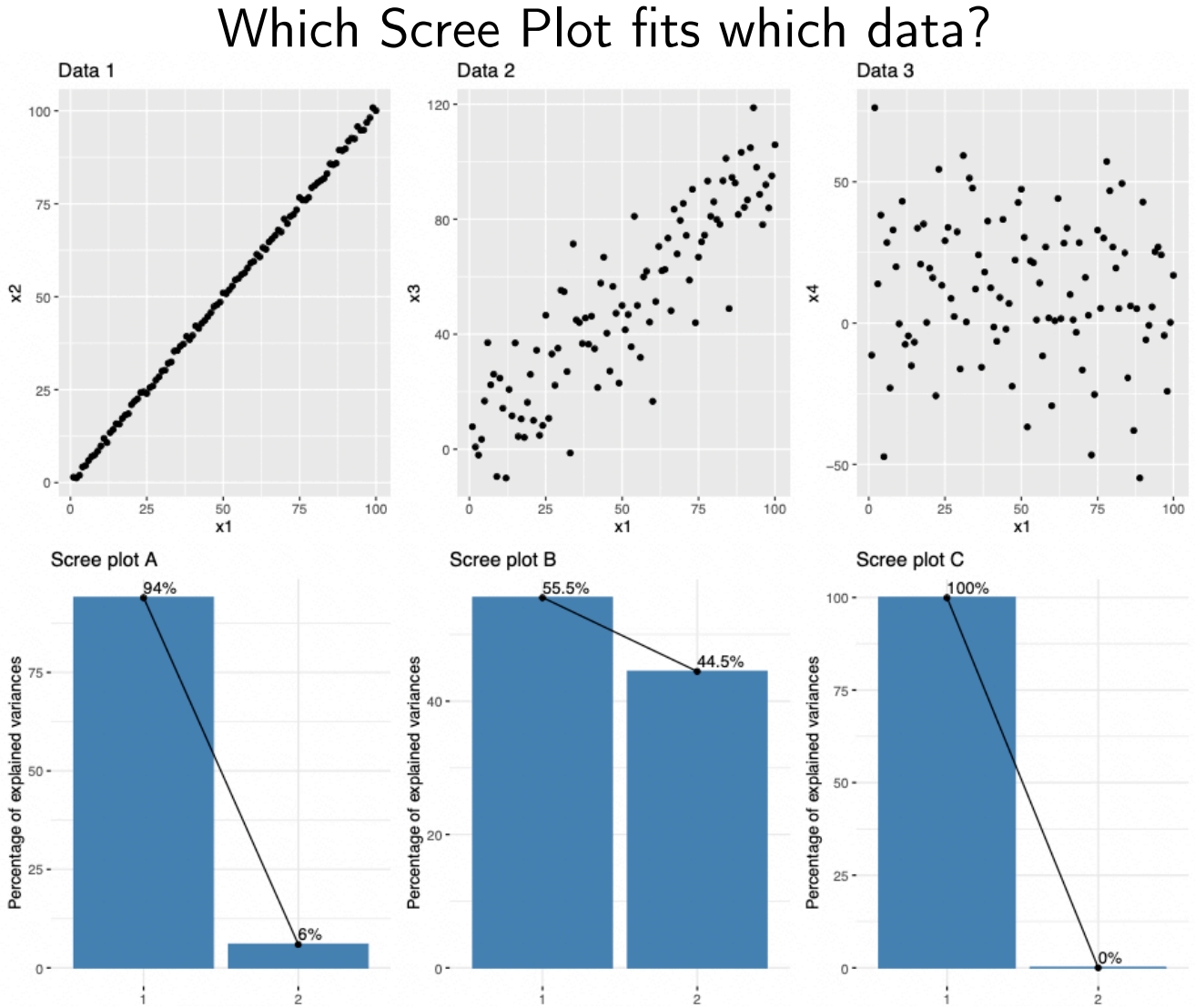
  i.e. the proportion of the sum of variances explained by the first k PCs.

Scree plots I

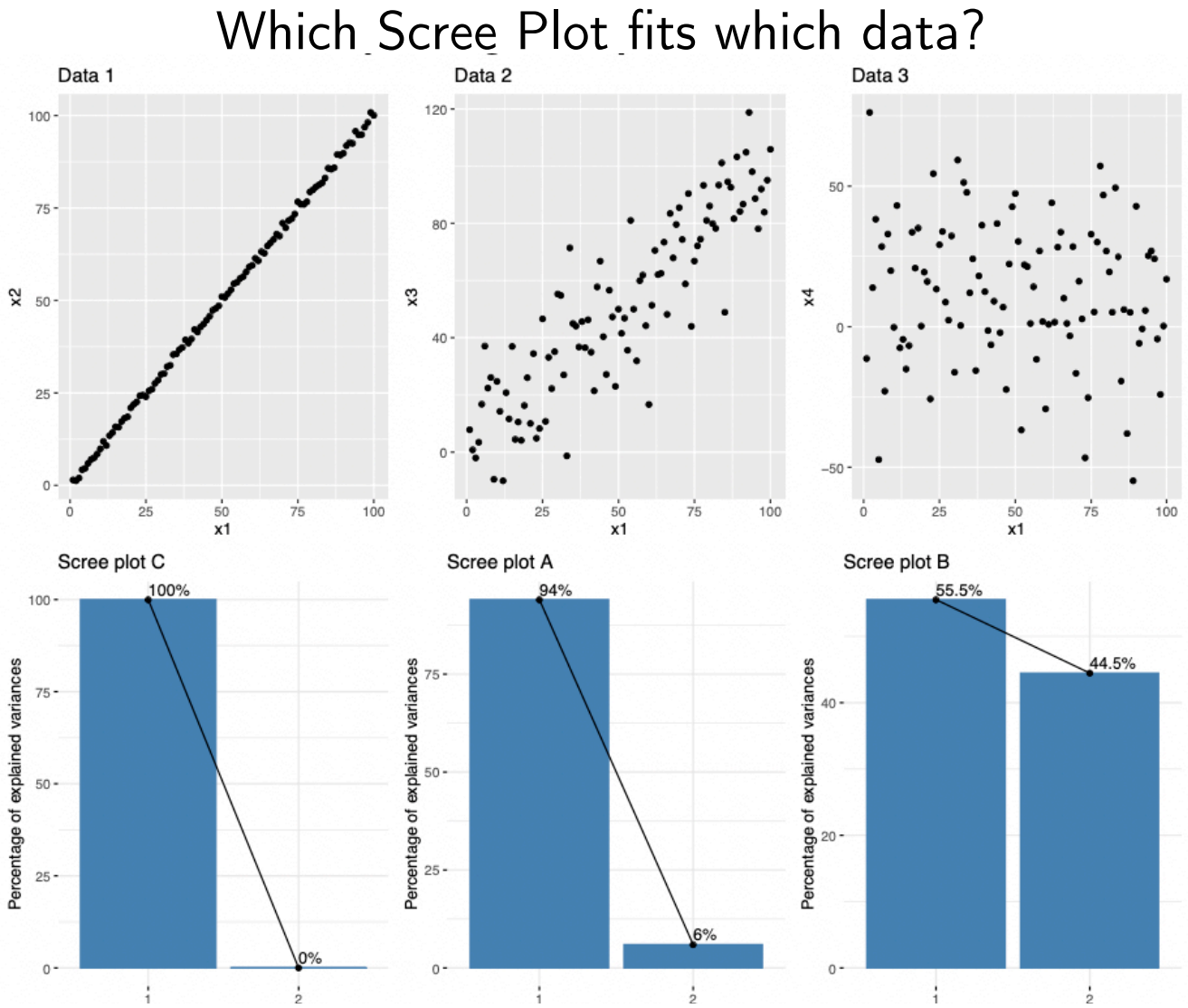
- Generally, a **scree plot** is a line plot of the eigenvalues of factors or principal components in an analysis, ordered in descending fashion.

- For our purposes, specifically, it is a line plot (possibly with additional bars) with the PC-indices on the x-axis and the value of the corresponding eigenvalue **or** its proportion of variance explained on the y-axis.

Scree plots II



Scree plots III



Loadings and scores

1. Thus far, we have referred to $\boldsymbol{a}_p$, $p = 1, \ldots, m$, as *principal components* or *principal component vectors* (both slightly vaguely abbreviated by PCs). Sometimes, however, $\boldsymbol{a}_p$, $p = 1, \ldots, m$, are also referred to as **loadings of the pth PC**.

2. Let $\boldsymbol{x}_i$, $i = 1, \ldots, n$, denote the ith row/observation of the data matrix $\boldsymbol{X}$ and $\bar{\boldsymbol{x}}$ the arithmetic mean of all observations. Then, the **score of the pth PC corresponding to the ith observation** is defined as

$$y_{ip} = \boldsymbol{a}_p^\top (\boldsymbol{x}_i - \bar{\boldsymbol{x}}), \qquad i = 1, \ldots, n\,.$$

The value y_{ip} is the point where the centered ith observation projects onto the direction vector for the pth component.

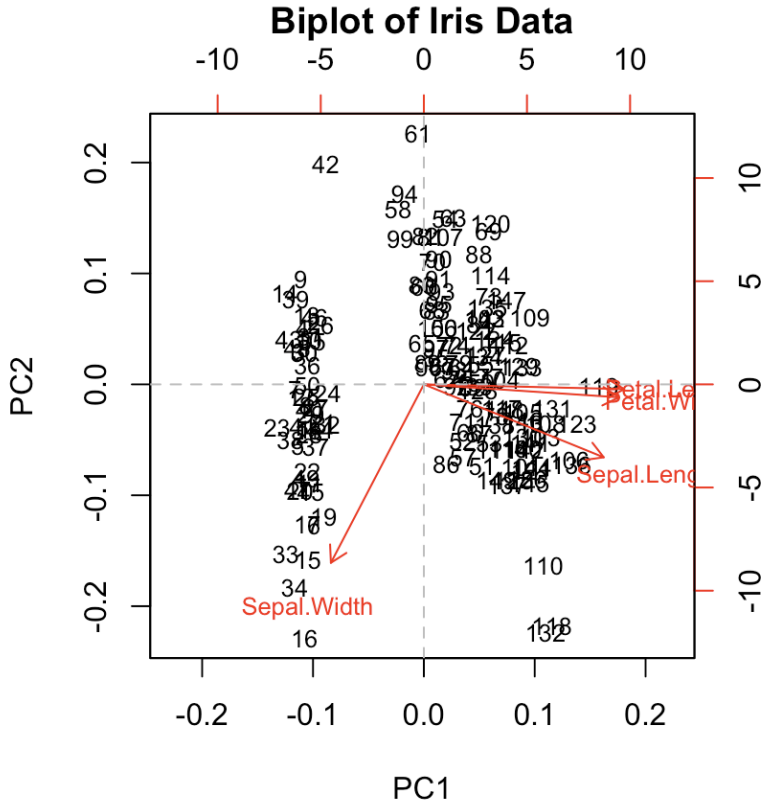
Biplots I

- Biplots are exploratory graphs that overlay a
  - ***score plot***, which is a plot with the first $k \leq m$ PCs as axes and the score values of each observation plotted as points, i.e. the points given by $(y_{i1}, \ldots, y_{ik})^\top \in \mathbb{R}^k$, $i = 1, \ldots, n$
  - with a ***loading plot***, where the first $k \leq m$ PCs can again be seen as the axes, but instead of plotting score values as points, the loadings of each variable are plotted as labelled, directed vectors. This is equivalent to plotting **each row (observation) of the following matrix as labelled, directed vectors:**

$$\begin{matrix} \overbrace{\phantom{v_{11}}}^{\boldsymbol{a}_1} & \overbrace{}^{\boldsymbol{a}_{\ldots}} & \overbrace{\phantom{v_{1k}}}^{\boldsymbol{a}_k} \end{matrix}$$
$$\begin{bmatrix} v_{11} & \cdots & v_{1k} \\ \vdots & \vdots & \vdots \\ v_{m1} & \cdots & v_{mk} \end{bmatrix}$$

Biplots II

- Biplots can be considered as a generalization of simple two-dimensional scatterplots.
- One example from `https://rpubs.com/ssipa/281715`:



Biplots III

Some helpful interpretation tips for Biplots:[2]

- Like for any scatterplot we may look for patterns, clusters, and outliers in a Biplot. In addition, note the following for the loading vectors:
  - The *orientation* (direction) of the vector, with respect to the principal component space, in particular, its angle with the principal component axes: the more parallel to a principal component axis is a vector, the more it contributes only to that PC.
  - The *length in the space*; the longer the vector, the more variability of this variable is represented by the two displayed principal components; short vectors are thus better represented in other dimension.
  - The *angles between vectors* of different variables show their correlation in this space: small angles represent high positive correlation, right angles represent lack of correlation, opposite angles represent high negative correlation.

[2] Source: Hartmann, K., Krois, J., Waske, B. (2018): E-Learning Project SOGA: Statistics and Geospatial Data Analysis. Department of Earth Sciences, Freie Universitaet Berlin.

Contents

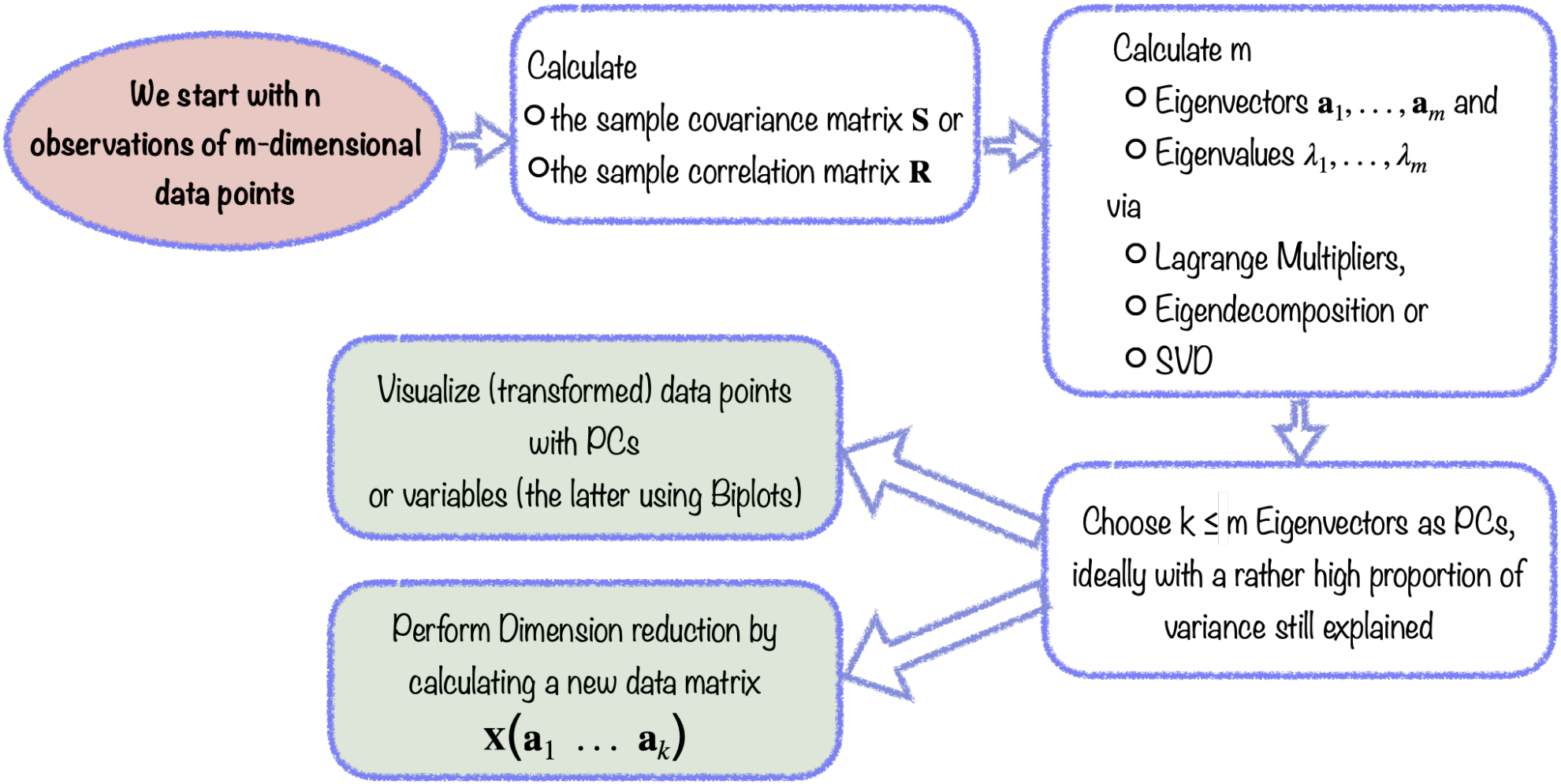
Number of PCs to use for dimension reduction

The goal is to have as much of the variance in the data explained by the first k principal components as possible.
$\Rightarrow$ Directions of the ellipsoid with short principal axes may be negligible.

SOME possible criteria:

- Kaiser criterion: Principal components with eigenvalue greater than 1. (I.e. the maximal k s.t. $\lambda_k > 1$)
- All principal components needed to get a total of 80% of the variance. (I.e. the minimal k s.t. $\mathrm{tr}(\mathbf{\Lambda}_{\text{ord}})^{-1} \cdot \sum_{i=1}^{k} \lambda_k \geq 0.8$)
- Scree Plot: Consider a graphical representation of the eigenvalues. Use as many principal components up to the bend of the graph (elbow).
- Simply choose k so that it is convenient (e.g. for a planned visualization).

Contents


We start with n observations of m-dimensional data points
Calculate
the sample covariance matrix $\mathbf{S}$ or
the sample correlation matrix $\mathbf{R}$
Calculate m
Eigenvectors $\mathbf{a}_1, \ldots, \mathbf{a}_m$ and
Eigenvalues $\lambda_1, \ldots, \lambda_m$
via
Lagrange Multipliers,
Eigendecomposition or
SVD
Choose k ≤ m Eigenvectors as PCs, ideally with a rather high proportion of variance still explained
Visualize (transformed) data points with PCs or variables (the latter using Biplots)
Perform Dimension reduction by calculating a new data matrix
$\mathbf{x}(\mathbf{a}_1 \ \ldots \ \mathbf{a}_k)$