Matrix methods – Principal component analysis

MATH 2740 – Mathematics of Data Science – Lecture 11

Julien Arino

`julien.arino@umanitoba.ca`

Department of Mathematics @ University of Manitoba

Fall 202X

The University of Manitoba campuses are located on original lands of Anishinaabeg, Ininew, Anisininew, Dakota and Dene peoples, and on the National Homeland of the Red River Métis. We respect the Treaties that were made on these territories, we acknowledge the harms and mistakes of the past, and we dedicate ourselves to move forward in partnership with Indigenous communities in a spirit of Reconciliation and collaboration.

Outline

A crash course on probability

A running example: hockey players

Change of basis

Back to PCA

Dimensionality reduction

One of the reasons the SVD is used is for dimensionality reduction. However, SVD has many many other uses

Now we look at another dimensionality reduction technique, PCA

PCA is often used as a blackbox technique, here we take a look at the math behind it

What is PCA?

Linear algebraic technique

Helps reduce a complex dataset to a lower dimensional one

Non-parametric method: does not assume anything about data distribution (distribution from the statistical point of view)

A crash course on probability

A running example: hockey players

Change of basis

Back to PCA

Brief “review” of some probability concepts

Proper definition of *probability* requires to use *measure theory*.. will not get into details here

A **random variable** X is a *measurable* function $X : \Omega \to E$, where Ω is a set of outcomes (*sample space*) and E is a measurable space

$\mathbb{P}(X \in S \subseteq E) = \mathbb{P}(\omega \in \Omega | X(\omega) \in S)$

Distribution function of a r.v., $F(x) = \mathbb{P}(X \leq x)$, describes the distribution of a r.v.

R.v. can be discrete or continuous or .. other things.

Definition 1 (Variance)

Let X be a random variable. The **variance** of X is given by

$$\operatorname{Var} X = E\left[(X - E(X))^2\right]$$

where E is the expected value

Definition 2 (Covariance)

Let X, Y be jointly distributed random variables. The **covariance** of X and Y is given by

$$\operatorname{cov}(X, Y) = E\left[(X - E(X))\,(Y - E(Y))\right]$$

Note that $\operatorname{cov}(X, X) = E\left[(X - E(X))^2\right] = \operatorname{Var} X$

In practice: “true law” versus “observation”

In statistics: we reason on the *true law* of distributions, but we usually have only access to a sample

We then use **estimators** to .. estimate the value of a parameter, e.g., the mean, variance and covariance

Definition 3 (Unbiased estimators of the mean and variance)

Let $x_1, \ldots, x_n$ be data points (the *sample*) and

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

be the **mean** of the data. An unbiased estimator of the variance of the sample is

$$\sigma^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

Definition 4 (Unbiased estimator of the covariance)

Let $(x_1, y_1), \ldots, (x_n, y_n)$ be data points,

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \text{ and } \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

be the means of the data. An estimator of the covariance of the sample is

$$\mathrm{cov}(x, y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

What does covariance do?

Variance explains how data disperses around the mean, in a 1-D case

Covariance measures the relationship between two dimensions. E.g., height and weight

More than the exact value, the sign is important:

- $\text{cov}(X, Y) > 0$: both dimensions change in the same "direction"; e.g., larger height usually means higher weight
- $\text{cov}(X, Y) < 0$: both dimensions change in reverse directions; e.g., time spent on social media and performance in this class
- $\text{cov}(X, Y) = 0$: the dimensions are independent from one another; e.g., sex/gender and "intelligence"

The covariance matrix

Typically, we consider more than 2 variables..

Definition 5

Suppose p random variables $X_1, \ldots, X_p$. Then the covariance matrix is the symmetric matrix

$$\begin{pmatrix} \mathrm{cov}(X_1, X_1) & \mathrm{cov}(X_1, X_2) & \cdots & \mathrm{cov}(X_1, X_p) \\ \mathrm{cov}(X_2, X_1) & \mathrm{cov}(X_2, X_2) & \cdots & \mathrm{cov}(X_2, X_p) \\ \vdots & \vdots & & \vdots \\ \mathrm{cov}(X_p, X_1) & \mathrm{cov}(X_p, X_2) & \cdots & \mathrm{cov}(X_p, X_p) \end{pmatrix}$$

i.e., using the properties of covariance,

$$\begin{pmatrix} \mathrm{Var}\, X_1 & \mathrm{cov}(X_1, X_2) & \cdots & \mathrm{cov}(X_1, X_p) \\ \mathrm{cov}(X_1, X_2) & \mathrm{Var}\, X_2 & \cdots & \mathrm{cov}(X_2, X_p) \\ \vdots & \vdots & & \vdots \\ \mathrm{cov}(X_1, X_p) & \mathrm{cov}(X_2, X_p) & \cdots & \mathrm{Var}\, X_p \end{pmatrix}$$

A crash course on probability

A running example: hockey players

Change of basis

Back to PCA

A 2D example

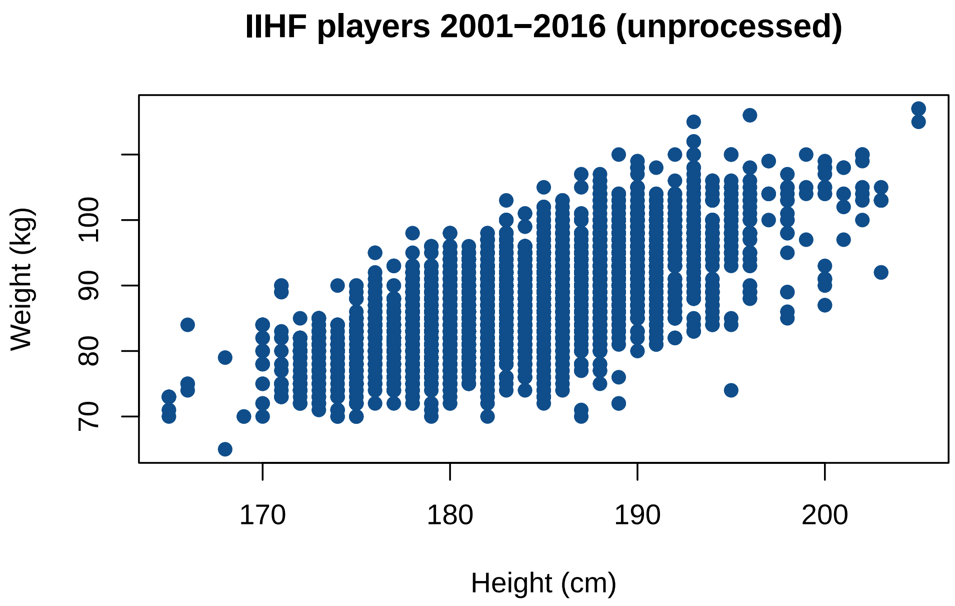
See a dataset on this page for a dataset of height and weight of some hockey players

```
# From https://figshare.com/ndownloader/files/5303173
data = read.csv("https://github.com/julien-arino/math-of-data-science/raw/re
head(data, n=3)

##   year country no                      name position side height weight      bi
## 1 2001     RUS 10    tverdovsky oleg               D    L    185     84 1976-05
## 2 2001     RUS  2  vichnevsky vitali               D    L    188     86 1980-03
## 3 2001     RUS 26 petrochinin evgeni               D    L    182     95 1976-02
##                      club      age cohort      bmi
## 1   anaheim mighty ducks  24.95277   1976 24.54346
## 2   anaheim mighty ducks  21.11978   1980 24.33228
## 3 severstal cherepovetal  25.22930   1976 28.68011

dim(data)
```

In case you are wondering, this is a database of ice hockey players at IIHF world championships, 2001-2016, assembled by the dataset's author

See some comments here

As usual, it is a good idea to plot this to get a sense of the lay of the land


IIHF players 2001–2016 (unprocessed)
Weight (kg)
70
80
90
100
170
180
190
200
Height (cm)

The author of the study is interested in the evolution of weights, so it is likely that the same person will be in the dataset several times

Let us check this: first check will be `FALSE` if the number of unique names does not match the number of rows in the dataset

```
length(unique(data$name)) == dim(data)[1]

## [1] FALSE

length(unique(data$name))

## [1] 3278
```

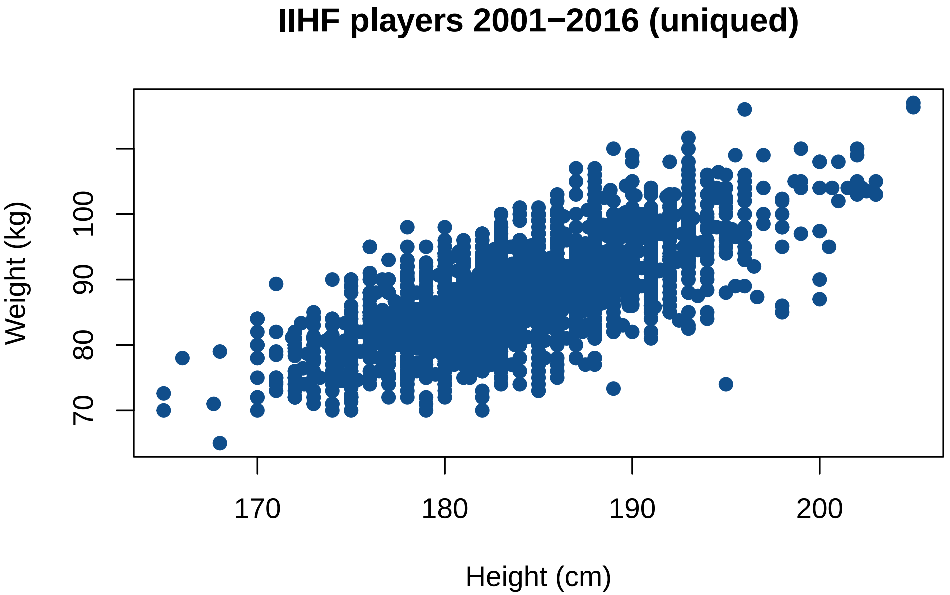
Not interested in the evolution of weights, so simplify: if more than one record for someone, take average of recorded weights and heights

To be extra careful, could check as well that there are no major variations on player height (homonymies?)

```
data_simplified = data.frame(name = unique(data$name))
w = c()
h = c()
for (n in data_simplified$name) {
    tmp = data[which(data$name == n),]
    h = c(h, mean(tmp$height))
    w = c(w, mean(tmp$weight))
}
data_simplified$weight = w
data_simplified$height = h
```

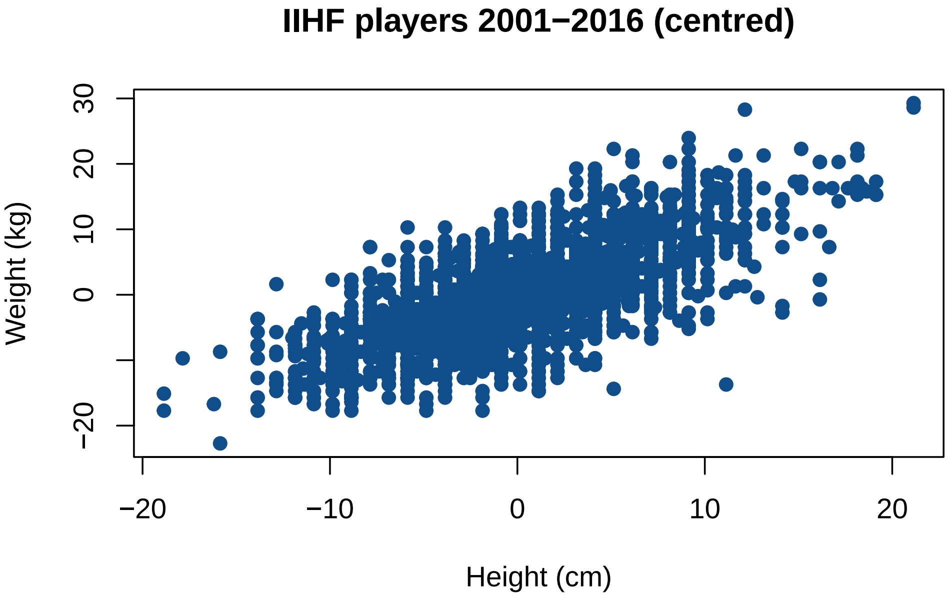
```r
data = data_simplified
head(data_simplified, n = 6)
```

```
##                    name weight height
## 1    tverdovsky oleg    84.0  185.0
## 2  vichnevsky vitali    86.0  188.0
## 3 petrochinin evgeni    95.0  182.0
## 4    zhdan alexander    85.5  178.5
## 5    orekhovsky oleg    88.0  175.0
## 6      zhukov sergei    92.5  193.0
```


IIHF players 2001–2016 (uniqued)
Weight (kg)
70
80
90
100
170
180
190
200
Height (cm)

Centre the data

```
mean(data$weight)

## [1] 87.71555

mean(data$height)

## [1] 183.8596

data$weight.c = data$weight-mean(data$weight)
data$height.c = data$height-mean(data$height)
```


IIHF players 2001−2016 (centred)
Weight (kg)
Height (cm)
30
20
10
0
−20
−20
−10
0
10
20

Setting things up

Each participant is a row in the matrix (an *observation*)

Each variable is a column

So we have an 200×10 matrix (we discard the “Participant number” column)

We want to find what carries the most information

For this, we are going to project the information in a new basis in which the first “dimension” will carry most variance, the second dimension will carry a little less, etc.

In order to do so, we need to learn how to change bases

A crash course on probability

A running example: hockey players

Change of basis

Back to PCA

In the following slide,

$$[\boldsymbol{x}]_{\mathcal{B}}$$

denotes the coordinates of $\boldsymbol{x}$ in the basis $\mathcal{B}$

The aim of a change of basis is to express vectors in another coordinate system (another basis)

We do so by finding a matrix allowing to move from one basis to another

Change of basis

Definition 6 (Change of basis matrix)

$\mathcal{B} = \{\boldsymbol{u}_1, \dots, \boldsymbol{u}_n\}$ and $\mathcal{C} = \{\boldsymbol{v}_1, \dots, \boldsymbol{v}_n\}$ bases of vector space V
The **change of basis matrix** $P_{\mathcal{C}\leftarrow\mathcal{B}} \in \mathcal{M}_n$,

$$P_{\mathcal{C}\leftarrow\mathcal{B}} = [[\boldsymbol{u}_1]_{\mathcal{C}} \cdots [\boldsymbol{u}_n]_{\mathcal{C}}]$$

has columns the coordinate vectors $[\boldsymbol{u}_1]_{\mathcal{C}}, \dots, [\boldsymbol{u}_n]_{\mathcal{C}}$ of vectors in $\mathcal{B}$ with respect to $\mathcal{C}$

Theorem 7

$\mathcal{B} = \{\boldsymbol{u}_1, \dots, \boldsymbol{u}_n\}$ and $\mathcal{C} = \{\boldsymbol{v}_1, \dots, \boldsymbol{v}_n\}$ bases of vector space V and $P_{\mathcal{C}\leftarrow\mathcal{B}}$ a change of basis matrix from $\mathcal{B}$ to $\mathcal{C}$

1. $\forall \boldsymbol{x} \in V$, $P_{\mathcal{C}\leftarrow\mathcal{B}}[\boldsymbol{x}]_{\mathcal{B}} = [\boldsymbol{x}]_{\mathcal{C}}$
2. $P_{\mathcal{C}\leftarrow\mathcal{B}}$ *s.t.* $\forall \boldsymbol{x} \in V$, $P_{\mathcal{C}\leftarrow\mathcal{B}}[\boldsymbol{x}]_{\mathcal{B}} = [\boldsymbol{x}]_{\mathcal{C}}$ *is* ***unique***
3. $P_{\mathcal{C}\leftarrow\mathcal{B}}$ *invertible and* $P^{-1}_{\mathcal{C}\leftarrow\mathcal{B}} = P_{\mathcal{B}\leftarrow\mathcal{C}}$

Row-reduction method for changing bases

Theorem 8

$\mathcal{B} = \{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$ and $\mathcal{C} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ bases of vector space V. Let $\mathcal{E}$ be any basis for V,

$$B = [[\mathbf{u}_1]_{\mathcal{E}}, \ldots, [\mathbf{u}_n]_{\mathcal{E}}] \text{ and } C = [[\mathbf{v}_1]_{\mathcal{E}}, \ldots, [\mathbf{v}_n]_{\mathcal{E}}]$$

and let $[C|B]$ be the augmented matrix constructed using C and B. Then

$$RREF\left([C|B]\right) = [\mathbb{I}|P_{\mathcal{C}\leftarrow\mathcal{B}}]$$

If working in $\mathbb{R}^n$, this is quite useful with $\mathcal{E}$ the standard basis of $\mathbb{R}^n$ (it does not matter if $\mathcal{B} = \mathcal{E}$)

So the question now becomes

How do we find what new basis to look at our data in?

(Changing the basis does not change the data, just the view you have of it)

(Think of what happens when you do a headstand.. your up becomes down, your right and left switch, but the world does not change, just your view of it)

(Changes of bases are *fundamental* operations in Science)

A crash course on probability

A running example: hockey players

Change of basis

Back to PCA

Setting things up

I will use notation (mostly) as in Joliffe's *Principal Component Analysis* (PDF of older version available for free from UofM Libraries)

$\boldsymbol{x} = (x_1, \ldots, x_p)$ vector of p random variables

We seek a linear function $\boldsymbol{\alpha}_1^T \boldsymbol{x}$ with maximum variance, where $\boldsymbol{\alpha}_1 = (\alpha_{11}, \ldots, \alpha_{1p})$, i.e.,

$$\boldsymbol{\alpha}_1^T \boldsymbol{x} = \sum_{j=1}^{p} \alpha_{1j} x_j$$

Then we seek a linear function $\boldsymbol{\alpha}_2^T \boldsymbol{x}$ with maximum variance, uncorrelated to $\boldsymbol{\alpha}_1^T \boldsymbol{x}$

And we continue...

At kth stage, we find a linear function $\boldsymbol{\alpha}_k^T \boldsymbol{x}$ with maximum variance, uncorrelated to $\boldsymbol{\alpha}_1^T \boldsymbol{x}, \ldots, \boldsymbol{\alpha}_{k-1}^T \boldsymbol{x}$

$\boldsymbol{\alpha}_i^T \boldsymbol{x}$ is the ith **principal component** (PC)

Case of known covariance matrix

Suppose we know Σ, covariance matrix of $\boldsymbol{x}$ (i.e., typically: we know $\boldsymbol{x}$)

Then the kth PC is

$$z_k = \boldsymbol{\alpha}_k^T \boldsymbol{x}$$

where $\boldsymbol{\alpha}_k$ is an eigenvector of Σ corresponding to the kth largest eigenvalue λ_k

If, additionally, $\|\boldsymbol{\alpha}_k\| = \boldsymbol{\alpha}_k^T \boldsymbol{\alpha} = 1$, then $\lambda_k = \text{Var } z_k$

Why is that?

Let us start with

$$\boldsymbol{\alpha}_1^T \boldsymbol{x}$$

We want maximum variance, where $\boldsymbol{\alpha}_1 = (\alpha_{11}, \ldots, \alpha_{1p})$, i.e.,

$$\boldsymbol{\alpha}_1^T \boldsymbol{x} = \sum_{j=1}^{p} \alpha_{1j} x_j$$

with the constraint that $\|\boldsymbol{\alpha}_1\| = 1$

We have

$$\text{Var } \boldsymbol{\alpha}_1^T \boldsymbol{x} = \boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1$$

Objective

We want to maximise Var $\alpha_1^T \boldsymbol{x}$, i.e.,

$$\alpha_1^T \Sigma \alpha_1$$

under the constraint that $\|\alpha_1\| = 1$

$\Longrightarrow$ use **Lagrange multipliers**

Maximisation using Lagrange multipliers

(A.k.a. super-brief intro to multivariable calculus)

We want the max of $f(x_1, \ldots, x_n)$ under the constraint $g(x_1, \ldots, x_n) = k$

1. Solve

$$\begin{aligned} \nabla f(x_1, \ldots, x_n) &= \lambda \nabla g(x_1, \ldots, x_n) \\ g(x_1, \ldots, x_n) &= k \end{aligned}$$

   where $\nabla = (\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_n})$ is the **gradient operator**

2. Plug all solutions into $f(x_1, \ldots, x_n)$ and find maximum values (provided values exist and $\nabla g \neq \mathbf{0}$ there)

λ is the **Lagrange multiplier**

The gradient

(Continuing our super-brief intro to multivariable calculus)

$f : \mathbb{R}^n \to \mathbb{R}$ function of several variables, $\nabla = \left(\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_n}\right)$ the gradient operator

Then

$$\nabla f = \left(\frac{\partial}{\partial x_1} f, \ldots, \frac{\partial}{\partial x_n} f\right)$$

So ∇f is a *vector-valued* function, $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$; also written as

$$\nabla f = f_{x_1}(x_1, \ldots, x_n)\boldsymbol{e}_1 + \cdots f_{x_n}(x_1, \ldots, x_n)\boldsymbol{e}_n$$

where f_{x_i} is the partial derivative of f with respect to x_i and $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$ is the standard basis of $\mathbb{R}^n$

Bear with me..

(You may experience a brief period of discomfort)

$\boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1$ and $\|\boldsymbol{\alpha}_1\|^2 = \boldsymbol{\alpha}_1^T \boldsymbol{\alpha}_1$ are functions of $\boldsymbol{\alpha}_1 = (\alpha_{11}, \ldots, \alpha_{1p})$

In the notation of the previous slide, we want the max of

$$f(\alpha_{11}, \ldots, \alpha_{1p}) := \boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1$$

under the constraint that

$$g(\alpha_{11}, \ldots, \alpha_{1p}) := \boldsymbol{\alpha}_1^T \boldsymbol{\alpha}_1 = 1$$

and with gradient operator

$$\nabla = \left(\frac{\partial}{\partial \alpha_{11}}, \ldots, \frac{\partial}{\partial \alpha_{1p}} \right)$$

Effect of ∇ on g

g is easiest to see:

$$
\begin{aligned}
\nabla g(\alpha_{11}, \ldots, \alpha_{1p}) &= \left(\frac{\partial}{\partial \alpha_{11}}, \ldots, \frac{\partial}{\partial \alpha_{1p}}\right) (\alpha_{11}, \ldots, \alpha_{1p}) \begin{pmatrix} \alpha_{11} \\ \vdots \\ \alpha_{1p} \end{pmatrix} \\
&= \left(\frac{\partial}{\partial \alpha_{11}}, \ldots, \frac{\partial}{\partial \alpha_{1p}}\right) \left(\alpha_{11}^2 + \cdots + \alpha_{1p}^2\right) \\
&= \left(2\alpha_{11}, \ldots, 2\alpha_{1p}\right) \\
&= 2\boldsymbol{\alpha}_1
\end{aligned}
$$

(And that's a general result: $\nabla \|\boldsymbol{x}\|_2^2 = 2\boldsymbol{x}$ with $\|\cdot\|_2$ the Euclidean norm)

Effect of ∇ on f

Expand (write $\Sigma = [s_{ij}]$ and do not exploit symmetry)

$$\begin{aligned}
\boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1 &= (\alpha_{11}, \ldots, \alpha_{1p}) \begin{pmatrix} s_{11} & s_{12} & \cdots & s_{1p} \\ s_{21} & s_{22} & \cdots & s_{2p} \\ \vdots & \vdots & & \vdots \\ s_{p1} & s_{p2} & & s_{pp} \end{pmatrix} \begin{pmatrix} \alpha_{11} \\ \alpha_{12} \\ \vdots \\ \alpha_{1p} \end{pmatrix} \\
&= (\alpha_{11}, \ldots, \alpha_{1p}) \begin{pmatrix} s_{11}\alpha_{11} + s_{12}\alpha_{12} + \cdots + s_{1p}\alpha_{1p} \\ s_{21}\alpha_{11} + s_{22}\alpha_{12} + \cdots + s_{2p}\alpha_{1p} \\ \vdots \\ s_{p1}\alpha_{11} + s_{p2}\alpha_{12} + \cdots + s_{pp}\alpha_{1p} \end{pmatrix} \\
&= (s_{11}\alpha_{11} + s_{12}\alpha_{12} + \cdots + s_{1p}\alpha_{1p})\alpha_{11} \\
&\quad + (s_{21}\alpha_{11} + s_{22}\alpha_{12} + \cdots + s_{2p}\alpha_{1p})\alpha_{12} \\
&\quad \vdots \\
&\quad + (s_{p1}\alpha_{11} + s_{p2}\alpha_{12} + \cdots + s_{pp}\alpha_{1p})\alpha_{1p}
\end{aligned}$$

We have

$$
\begin{aligned}
\boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1 = {} & (s_{11}\alpha_{11} + s_{12}\alpha_{12} + \cdots + s_{1p}\alpha_{1p})\alpha_{11} \\
& + (s_{21}\alpha_{11} + s_{22}\alpha_{12} + \cdots + s_{2p}\alpha_{1p})\alpha_{12} \\
& \vdots \\
& + (s_{p1}\alpha_{11} + s_{p2}\alpha_{12} + \cdots + s_{pp}\alpha_{1p})\alpha_{1p}
\end{aligned}
$$

$$
\begin{aligned}
\Longrightarrow \frac{\partial}{\partial \alpha_{11}} \boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1 &= (s_{11}\alpha_{11} + s_{12}\alpha_{12} + \cdots + s_{1p}\alpha_{1p}) + s_{11}\alpha_{11} \\
&\quad + s_{21}\alpha_{12} + \cdots + s_{p1}\alpha_{1p} \\
&= s_{11}\alpha_{11} + s_{12}\alpha_{12} + \cdots + s_{1p}\alpha_{1p} \\
&\quad + s_{11}\alpha_{11} + s_{21}\alpha_{12} + \cdots + s_{p1}\alpha_{1p} \\
&= 2(s_{11}\alpha_{11} + s_{12}\alpha_{12} + \cdots + s_{1p}\alpha_{1p})
\end{aligned}
$$

(last equality stems from symmetry of Σ)

In general, for $i = 1, \ldots, p$,

$$\begin{aligned} \frac{\partial}{\partial \alpha_{1i}} \boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1 &= s_{i1}\alpha_{11} + s_{i2}\alpha_{12} + \cdots + s_{ip}\alpha_{1p} \\ &\quad + s_{i1}\alpha_{11} + s_{2i}\alpha_{12} + \cdots + s_{pi}\alpha_{1p} \\ &= 2(s_{i1}\alpha_{11} + s_{i2}\alpha_{12} + \cdots + s_{ip}\alpha_{1p}) \end{aligned}$$

(because of symmetry of Σ)

As a consequence,

$$\nabla \boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1 = 2\Sigma\boldsymbol{\alpha}_1$$

So solving

$$\nabla f(x_1, \ldots, x_n) = \lambda \nabla g(x_1, \ldots, x_n)$$

means solving

$$2\Sigma\boldsymbol{\alpha}_1 = \lambda 2\boldsymbol{\alpha}_1$$

i.e.,

$$\Sigma\boldsymbol{\alpha}_1 = \lambda\boldsymbol{\alpha}_1$$

$\implies$ $(\lambda, \boldsymbol{\alpha}_1)$ eigenpair of Σ, with $\boldsymbol{\alpha}_1$ having unit length

Picking the right eigenvalue

$(\lambda, \boldsymbol{\alpha}_1)$ eigenpair of Σ, with $\boldsymbol{\alpha}_1$ having unit length

But which λ to choose?

Recall that we want Var $\boldsymbol{\alpha}_1^T \boldsymbol{x} = \boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1$ maximal

We have

$$\text{Var } \boldsymbol{\alpha}_1^T \boldsymbol{x} = \boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_1 = \boldsymbol{\alpha}_1^T (\Sigma \boldsymbol{\alpha}_1) = \boldsymbol{\alpha}_1^T (\lambda \boldsymbol{\alpha}_1) = \lambda (\boldsymbol{\alpha}_1^T \boldsymbol{\alpha}_1) = \lambda$$

$\Longrightarrow$ we pick $\lambda = \lambda_1$, the largest eigenvalue (covariance matrix symmetric so eigenvalues real)

What we have this far..

The first principal component is $\alpha_1^T \boldsymbol{x}$ and has variance λ_1, where λ_1 the largest eigenvalue of Σ and α_1 an associated eigenvector with $\|\alpha_1\| = 1$

We want the second principal component to be *uncorrelated* with $\alpha_1^T \boldsymbol{x}$ and to have maximum variance Var $\alpha_2^T \boldsymbol{x} = \alpha_2^T \Sigma \alpha_2$, under the constraint that $\|\alpha_2\| = 1$

$\alpha_2^T \boldsymbol{x}$ uncorrelated to $\alpha_1^T \boldsymbol{x}$ if $\mathrm{cov}(\alpha_1^T \boldsymbol{x}, \alpha_2^T \boldsymbol{x}) = 0$

We have

$$
\begin{aligned}
\mathrm{cov}(\boldsymbol{\alpha}_1^T \boldsymbol{x}, \boldsymbol{\alpha}_2^T \boldsymbol{x}) &= \boldsymbol{\alpha}_1^T \Sigma \boldsymbol{\alpha}_2 \\
&= \boldsymbol{\alpha}_2^T \Sigma^T \boldsymbol{\alpha}_1 \\
&= \boldsymbol{\alpha}_2^T \Sigma \boldsymbol{\alpha}_1 \quad [\Sigma \text{ symmetric}] \\
&= \boldsymbol{\alpha}_2^T (\lambda_1 \boldsymbol{\alpha}_1) \\
&= \lambda \boldsymbol{\alpha}_2^T \boldsymbol{\alpha}_1
\end{aligned}
$$

So $\boldsymbol{\alpha}_2^T \boldsymbol{x}$ uncorrelated to $\boldsymbol{\alpha}_1^T \boldsymbol{x}$ if $\boldsymbol{\alpha}_1 \perp \boldsymbol{\alpha}_2$

This is beginning to sound a lot like Gram-Schmidt, no?

In short

Take whatever covariance matrix is available to you (known Σ or sample S_X) – assume sample from now on for simplicity

For $i = 1, \dots, p$, the ith principal component is

$$z_i = \boldsymbol{v}_i^T \boldsymbol{x}$$

where $\boldsymbol{v}_i$ eigenvector of S_X associated to the ith largest eigenvalue λ_i

If $\boldsymbol{v}_i$ is normalised, then $\lambda_i = \text{Var } z_k$

Covariance matrix

Σ the covariance matrix of the random variable, S_X the sample covariance matrix

$X \in \mathcal{M}_{mp}$ the data, then the (sample) covariance matrix S_X takes the form

$$S_X = \frac{1}{n-1} X^T X$$

where the data is centred!

Sometimes you will see $S_X = 1/(n-1)XX^T$. This is for matrices with observations in columns and variables in rows. Just remember that you want the covariance matrix to have size the number of variables, not observations, this will give you the order in which to take the product

Covariance

The function `cov` returns the covariance of two samples

Note that the functions deals equally well with data that is not centred as with data that is centred

```
cov(data$height, data$weight)

## [1] 26.63506

cov(data$height.c, data$weight.c)

## [1] 26.63506
```

Covariance matrix

As we could see from plotting the data, there is a positive linear relationship between the two variables

Let us compute the sample covariance matrix

```
X = as.matrix(data[,c("height.c", "weight.c")])
S = 1/(dim(X)[1]-1)*t(X) %*% X
S

##          height.c weight.c
## height.c 29.66176 26.63506
## weight.c 26.63506 47.81112
```

Covariance matrix

The off-diagonal entries do match the computed covariance. Let us check that the variances are indeed a match too.

```
var(X[,1])

## [1] 29.66176

var(X[,2])

## [1] 47.81112
```

Hey, that works. Is math not cool? ;)

Principal components

Now compute the principal components. We need eigenvalues and eigenvectors

```
ev = eigen(S)
ev

## eigen() decomposition
## $values
## [1] 66.87496 10.59793
##
## $vectors
##           [,1]       [,2]
## [1,] 0.5820222 -0.8131729
## [2,] 0.8131729  0.5820222
```

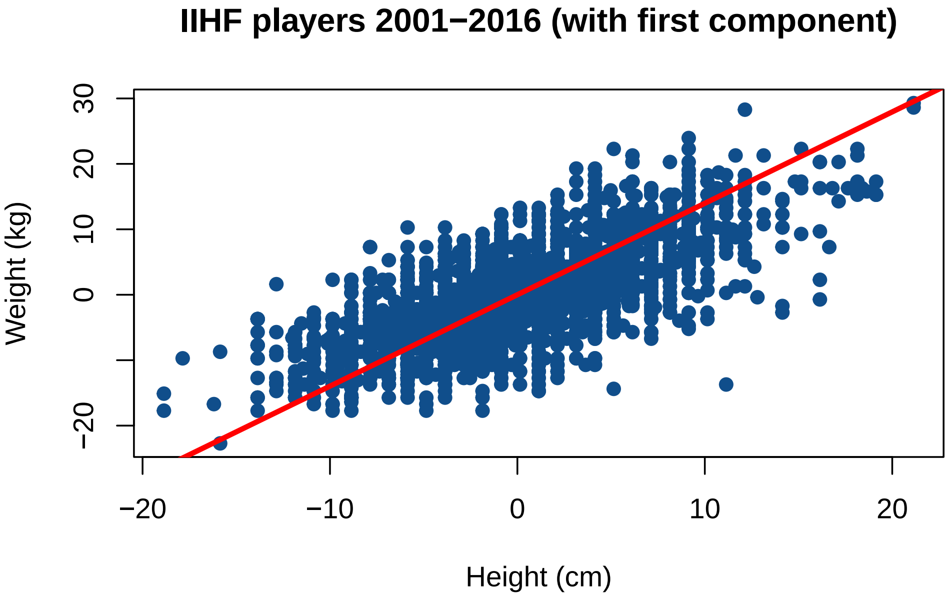
(`eigen` returns eigenvalues sorted in decreasing order and normalised eigenvectors)

First principal component

Let us plot this first eigenvector (well, the line carrying this first eigenvector)

To use the function `abline`, we need to give the coefficients of the line in the form of (intercept,slope). Intercept is easy, as the line goes through the origin (by construction and because we have centred the data). The slope is also quite simple..

```
plot(data$height.c, data$weight.c,
    pch = 19, col = "dodgerblue4",
    main = "IIHF players 2001-2016 (with first component)",
    xlab = "Height (cm)", ylab = "Weight (kg)")
abline(a = 0, b = ev$vectors[2,1]/ev$vectors[1,1],
       col = "red", lwd = 3)
```


IIHF players 2001–2016 (with first component)
Weight (kg)
30
20
10
0
−20
−20
−10
0
10
20
Height (cm)

Rotating the data

Let us rotate the data so that the red line becomes the *x*-axis

To do that, we use a rotation matrix

$$R_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

To find the angle θ, recall that $\tan\theta$ is equal to opposite length over adjacent length, i.e.,

$$\tan\theta = \frac{\texttt{ev\$vectors}[2,1]}{\texttt{ev\$vectors}[1,1]}$$

So we just use the $\arctan$ of this

Note that angles are in radians

Rotating the data

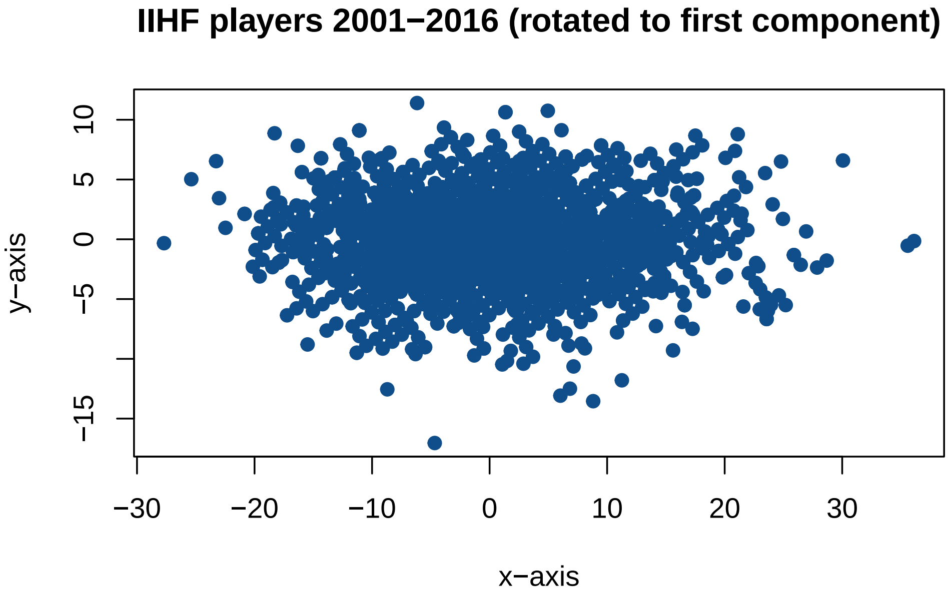
```
theta = atan(ev$vectors[2,1]/ev$vectors[1,1])
theta

## [1] 0.949583

R_theta = matrix(c(cos(theta), -sin(theta),
                   sin(theta), cos(theta)),
                 nr = 2, byrow = TRUE)
R_theta

##           [,1]       [,2]
## [1,] 0.5820222 -0.8131729
## [2,] 0.8131729  0.5820222
```

Rotating the data

And now we rotate the points

(In this case, we think of the points as vectors, of course)

```
tmp_in = matrix(c(data$weight.c, data$height.c),
                 nc = 2)
tmp_out = c()
for (i in 1:dim(tmp_in)[1]) {
    tmp_out = rbind(tmp_out,
                    t(R_theta %*% tmp_in[i,]))
}
data$weight.c_r = tmp_out[,1]
data$height.c_r = tmp_out[,2]
```

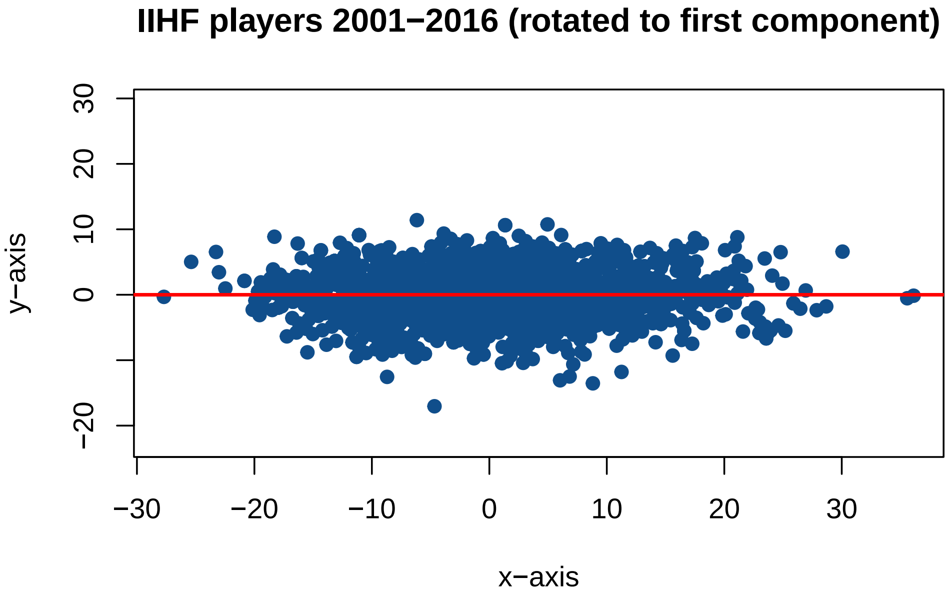

IIHF players 2001−2016 (rotated to first component)
y−axis
10
5
0
−5
−15
−30
−20
−10
0
10
20
30
x−axis

Principal components

Note that the axes have changed quite a lot, hence the very different aspect

Let us plot with the same range as for the non-rotated data for the y-axis

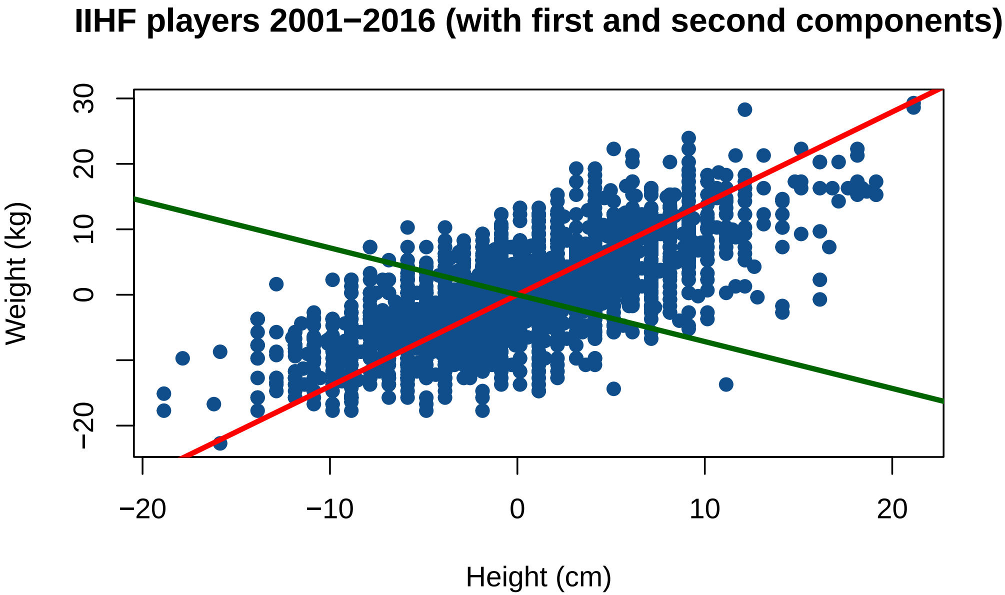
```
plot(data$height.c_r, data$weight.c_r,
    pch = 19, col = "dodgerblue4",
    xlab = "x-axis", ylab = "y-axis",
    main = "IIHF players 2001-2016 (rotated to first component)",
    ylim = range(data$weight.c))
abline(h = 0, col = "red", lwd = 2)
```


IIHF players 2001−2016 (rotated to first component)
y−axis
x−axis
30
20
10
0
−20
−30
−20
−10
0
10
20
30

First and second principal components

Plot the first and second eigenvectors

```
plot(data$height.c, data$weight.c,
    pch = 19, col = "dodgerblue4",
    main = "IIHF players 2001-2016 (with first and second components)",
    xlab = "Height (cm)", ylab = "Weight (kg)")
abline(a = 0, b = ev$vectors[2,1]/ev$vectors[1,1],
       col = "red", lwd = 3)
abline(a = 0, b = ev$vectors[2,2]/ev$vectors[1,2],
       col = "darkgreen", lwd = 3)
```


IIHF players 2001–2016 (with first and second components)
Weight (kg)
Height (cm)
30
20
10
0
−20
−20
−10
0
10
20

Proper change of basis

Let us change the basis so that, in the new basis, the first component is the x-axis and the second component is the y-axis

We want to use Theorem 8

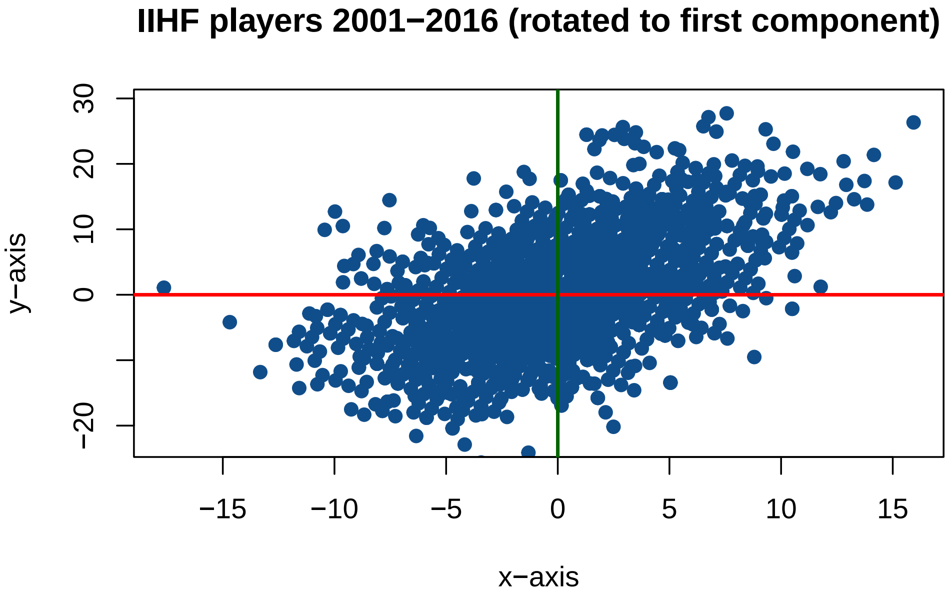
We need the coordinates of the new basis in the canonical basis of $\mathbb{R}^2$

Since both axes go through the origin, we can just use $y = ax$, with a the slope of the lines and, say, $x = 1$, i.e., $(x, y) = (1, a)$

We then normalise the resulting vectors

Proper change of basis

```
red_line = c(1, ev$vectors[2,1]/ev$vectors[1,1])
red_line = red_line/sqrt(sum(red_line^2))
green_line = c(1, ev$vectors[2,2]/ev$vectors[1,2])
green_line = green_line/sqrt(sum(green_line^2))
augmented_M = cbind(red_line,green_line, diag(2))
P = rref(augmented_M)[,3:4]

tmp_in = matrix(c(data$weight.c, data$height.c), nc = 2)
tmp_out = c()
for (i in 1:dim(tmp_in)[1]) {
    tmp_out = rbind(tmp_out, t(P %*% tmp_in[i,]))
}
data$weight.c_r2 = tmp_out[,1]
data$height.c_r2 = tmp_out[,2]
```


IIHF players 2001−2016 (rotated to first component)
y−axis
x−axis
30
20
10
0
−20
−15
−10
−5
0
5
10
15

PCA using built-in functions

Now do things “properly”

```
GS = pracma::gramSchmidt(A = ev$vectors, tol = 1e-10)
GS

## $Q
##            [,1]       [,2]
## [1,] 0.5820222 -0.8131729
## [2,] 0.8131729  0.5820222
##
## $R
##      [,1] [,2]
## [1,]    1    0
## [2,]    0    1
```

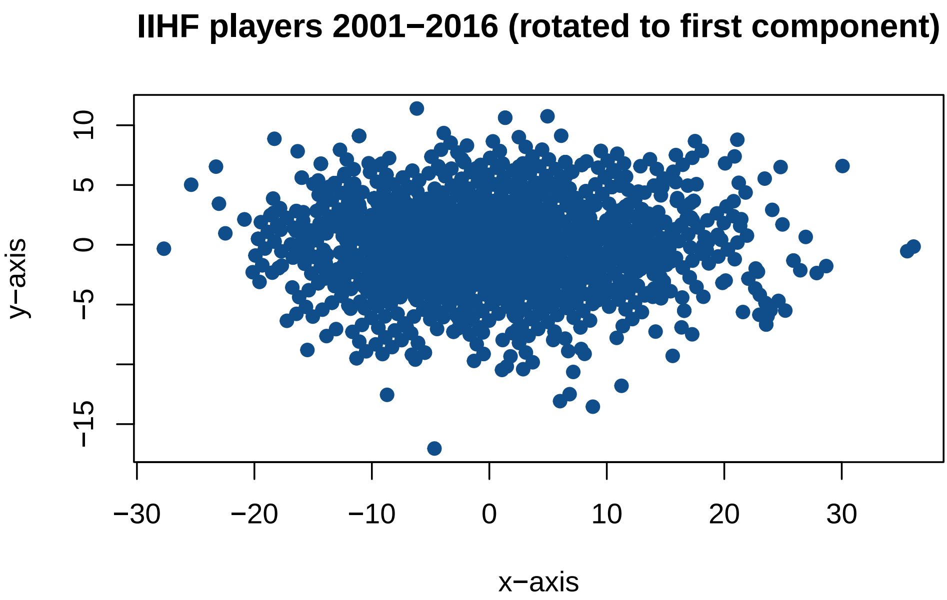
PCA using built-in functions

Now recall we saw a theorem that told us how to construct a new basis..

```
A=matrix(c(GS$Q,1,0,0,1), nr = 2)
A

##           [,1]       [,2] [,3] [,4]
## [1,] 0.5820222 -0.8131729    1    0
## [2,] 0.8131729  0.5820222    0    1

pracma::rref(A)

##      [,1] [,2]       [,3]      [,4]
## [1,]    1    0  0.5820222 0.8131729
## [2,]    0    1 -0.8131729 0.5820222
```

PCA using built-in functions

```
P = pracma::rref(A)[,c(3,4)]

##              [,1]      [,2]
## [1,]  0.5820222 0.8131729
## [2,] -0.8131729 0.5820222

X.new = X %*% t(P)
```


IIHF players 2001−2016 (rotated to first component)
y−axis
10
5
0
−5
−15
−30
−20
−10
0
10
20
30
x−axis