Dimension reduction, principal component analysis (PCA) and principal component regression (PCR)

Søren Højsgaard

Thu Sep 26 10:10:08 2024

Principal component analysis (PCA)

- Principal component analysis (PCA) is a an exploratory technique which can sometimes help identifying low dimensional structures in high dimensional data.
- There are many such methods, often called *dimension reduction* methods. Presumably the most classical method (dating back to 1904) is called *principal component analysis* or *PCA*.
- PCA is a linear method and is based on the idea of finding a new set of variables which are linear combinations of the original variables and which are uncorrelated and explain as much of the variation in the data as possible.
- A related method is *factor analysis* (not discussed here)

Example: Crime rate in the US in 1977

The crime_rate data contains the crime rate per 100.000 people in 50 US states in 1977.

```
crime <- doBy::crime_rate
state <- rownames(crime)
crime_state <- cbind(crime, state=state)
rownames(crime_state) <- NULL
```

```
crime |> head(3)
```

```
##         murder rape assault robbery burglary larceny autotheft
## Alabama   14.2   25     278      97     1136    1882       281
## Alaska    10.8   52     284      97     1332    3370       753
## Arizona    9.5   34     312     138     2346    4467       440
```

```
crime_state |> head(3)
```

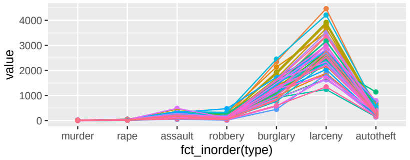
```
##   murder rape assault robbery burglary larceny autotheft   state
## 1   14.2   25     278      97     1136    1882       281 Alabama
## 2   10.8   52     284      97     1332    3370       753  Alaska
## 3    9.5   34     312     138     2346    4467       440 Arizona
```

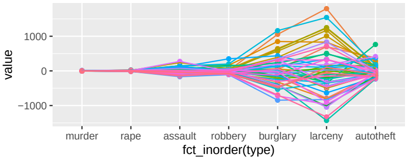
Summarizing multivariate data

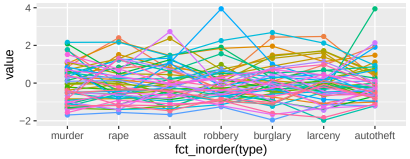
Plotting multivariate data

```r
## Two handy utilities:
make_long <- function(data., x){
    data. |>
        as.data.frame() |>
        pivot_longer(-!!sym(x), names_to = "type", values_to = "value")
}

do_plot1 <- function(data.){
    data. |>
        ggplot(aes(x=fct_inorder(type), y=value, group=state, color=state)) +
        geom_point() + geom_line() + theme(legend.position="none")
}


## Uncentered data
dat_long <- crime_state |> make_long("state")
## Centered but unscaled data
dat_c <- crime_state |> doBy::scale_df(center=T, scale=F)
dat_c_long <- dat_c |> make_long("state")
## Centered and scaled data
dat_cs <- crime_state |> doBy::scale_df(center=T, scale=T)
dat_cs_long <- dat_cs |> make_long("state")
```

```r
p1 <- do_plot1(dat_long)
p2 <- do_plot1(dat_c_long)
p3 <- do_plot1(dat_cs_long)
cowplot::plot_grid(p1, p2, p3)
```







Correlations

The relationship between some variables appear to be approximately linear, so we might want to calculate all pairwise correlations:

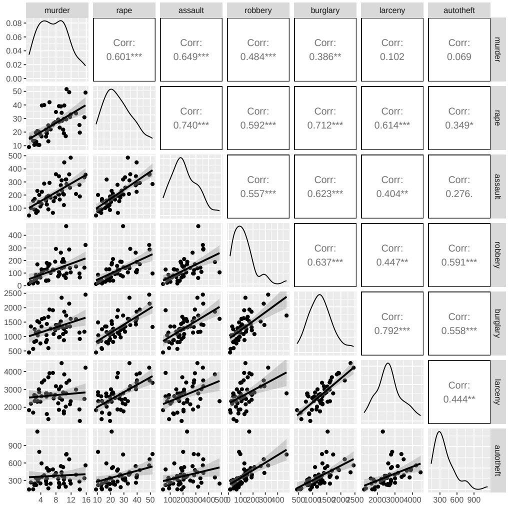
```
cormat <- cor(crime)
round(10*cormat)
```

```
##           murder rape assault robbery burglary larceny autotheft
## murder        10    6       6       5        4       1         1
## rape           6   10       7       6        7       6         3
## assault        6    7      10       6        6       4         3
## robbery        5    6       6      10        6       4         6
## burglary       4    7       6       6       10       8         6
## larceny        1    6       4       4        8      10         4
## autotheft      1    3       3       6        6       4        10
```

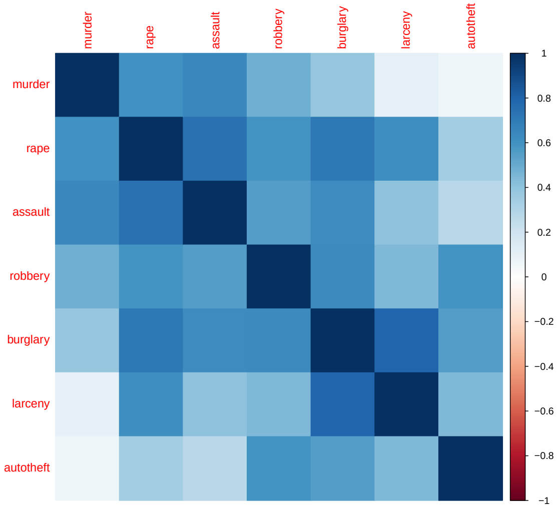
```r
doBy::truncate0(cormat, tol=0.6)
```

```
## 7 x 7 sparse Matrix of class "dgCMatrix"
##           murder rape assault robbery burglary larceny autotheft
## murder      1.00 0.60    0.65    .       .        .              .
## rape        0.60 1.00    0.74    .       0.71     0.61           .
## assault     0.65 0.74    1.00    .       0.62     .              .
## robbery     .    .       .       1.00    0.64     .              .
## burglary    .    0.71    0.62    0.64    1.00     0.79           .
## larceny     .    0.61    .       .       0.79     1.00           .
## autotheft   .    .       .       .       .        .              1
```

```
library(GGally)
crime |>
    ggpairs(lower = list(continuous = "smooth"),
            progress=FALSE)
```



```r
library(corrplot)
corrplot::corrplot(cormat, method="color")
```



Principal component analysis (PCA)

Basic idea

Easy to explain with only three variables

```
violent <- crime[,1:3]
violent |> head(3)
```

```
##         murder rape assault
## Alabama   14.2   25     278
## Alaska    10.8   52     284
## Arizona    9.5   34     312
```

```
violent |> cor()
```

```
##         murder rape assault
## murder    1.00 0.60    0.65
## rape      0.60 1.00    0.74
## assault   0.65 0.74    1.00
```

3D-plot of the data

```
library(plotly)
crime <- doBy::crime_rate
violent <- crime[,1:3]
violent |> head()
plot_ly(violent, x=~murder, y=~rape, z=~assault)

pc <- prcomp(violent)
violent_rec <- doBy::recover_pca_data(pc, 1)
violent2 <- rbind(violent, violent_rec)

plot_ly(violent2, x=~murder, y=~rape, z=~assault,
        color=c(rep('green', 50), rep('blue', 50)))
```

Denote columns of observed variables by y_1, y_2 and y_3. The idea is to find a new set of variables x_1, x_2 and x_3 such that the x variables 1) are uncorrelated and 2) explain as much of the variation in the data as possible.

The first principal component x_1 is a linear combination (weighted sum) of the y variables:

$$x_1 = r_{11} \cdot y_1 + r_{21} \cdot y_2 + \cdots + r_{31} \cdot y_3$$

where r_{11}, r_{21}, r_{31} are the weights. The weights are chosen so that x_1 explains as much of the variation in the data as possible.

The second principal component x_2 is also a linear combination of the y variables

$$x_2 = r_{12} \cdot y_1 + r_{22} \cdot y_2 + \cdots + r_{32} \cdot y_3$$

where r_{12}, r_{22}, r_{32} are the weights. The weights are chosen so that 1) x_2 explains second most variation in the data as possible and 2) is uncorrelated with x_1.

The third principal component x_3 is a linear combination of the y variables:

$$x_3 = r_{13} \cdot y_1 + r_{23} \cdot y_2 + \cdots + r_{33} \cdot y_3$$

where r_{13}, r_{23}, r_{33} are the weights. The weights are chosen so that 1) x_3 explains third most variation in the data as possible and 2) is uncorrelated with x_1 and x_2.

The weights r_{ij} are chosen so that (there are a couple of extra details)

1. Each x_j has length 1
2. $var(x_1) \geq var(x_2) \geq var(x_1)$
3. All principal components x_1, x_2, x_3 are uncorrelated.

It can be shown that

$$v_{total} = var(y_1) + var(y_2) + var(y_3) = var(x_1) + var(x_2) + var(x_3)$$

So

$$var(x_1)/v_{total}, (var(x_1) + var(x_2))/v_{total}$$

is the variance explained by the first and the first two principal components, respectively.

```r
pca <- prcomp(violent, center=T, scale.=T)
summary(pca)
```

```
## Importance of components:
##                          PC1   PC2    PC3
## Standard deviation     1.526 0.646 0.5046
## Proportion of Variance 0.776 0.139 0.0849
## Cumulative Proportion  0.776 0.915 1.0000
```

```r
pca$x |> head(3) ## Principal components / scores
```

```
##          PC1   PC2   PC3
## Alabama -1.3 -1.28 -0.28
## Alaska  -2.3  0.71  1.10
## Arizona -1.4  0.23 -0.19
```

```r
pca$x |> cov()
```

```
##          PC1      PC2     PC3
## PC1  2.3e+00 -4.6e-16 1.8e-16
## PC2 -4.6e-16  4.2e-01 6.4e-17
## PC3  1.8e-16  6.4e-17 2.5e-01
```

```r
pca$rotation ## Loadings / weights
```

```
##           PC1   PC2   PC3
## murder  -0.55 -0.82  0.15
## rape    -0.58  0.51  0.63
## assault -0.59  0.26 -0.76
```

Interpretation:

```r
cor(violent, pca$x[,1:2])
```

```
##           PC1   PC2
## murder  -0.85 -0.53
## rape    -0.89  0.33
## assault -0.91  0.17
```

Interpretation:

- x_1 is a weighted sum of the three variables. The weights are approximately equal. Hence x_1 is a measure of the average crime rate.
- x_2 gets negative contributions from `murder` and positive contributions from `assault` and `rape`. Hence x_2 is a measure of the violent crime rate.

The full dataset

Principal components can be obtained with `prcomp()`
Default is that variables are centered to have mean zero (`center=T`), but when the variation of the variables are very different it is often a good idea to standardize variables to have variance one (`scale.=T`).

```r
pca <- prcomp(crime, center=T, scale. = T)
summary(pca)
```

```
## Importance of components:
##                          PC1   PC2   PC3    PC4    PC5    PC6    PC7
## Standard deviation     2.029 1.113 0.852 0.5625 0.5079 0.4712 0.3522
## Proportion of Variance 0.588 0.177 0.104 0.0452 0.0369 0.0317 0.0177
## Cumulative Proportion  0.588 0.765 0.869 0.9137 0.9506 0.9823 1.0000
```

The first PC explains about 60% of the variation and with the first three components about 85% of the variation is explained.

Interpretation - loadings / rotations

```r
cor(crime, pca$x[,1:3])
```

```
##             PC1    PC2    PC3
## murder    -0.61 -0.700 -0.152
## rape      -0.88 -0.189  0.208
## assault   -0.80 -0.382  0.059
## robbery   -0.81  0.047 -0.422
## burglary  -0.89  0.226  0.179
## larceny   -0.72  0.448  0.459
## autotheft -0.60  0.559 -0.484
```

- x_1 is a weighted sum of all variables. The weights are approximately equal (when data is centered and scaled). Hence x_1 is a measure of the average crime rate.
- x_2 gets negative contributions violent crime and positive contributions from economical crime. A state with much voilence tends to have a negative value of x_2; a state with economical crime tends to have a positive value of x_3.
- From thereof the picture is more blurred.

Principal components / scores

The principal components / scores:

```
X <- pca$x
head(X, 5)
```

```
##                PC1   PC2    PC3    PC4   PC5   PC6    PC7
## Alabama       0.05 -2.10 -0.502  0.251 0.498 -0.43  0.118
## Alaska       -2.42  0.17  0.070  1.160 1.470  1.50  0.465
## Arizona      -3.01  0.84  1.752 -0.116 0.280 -1.07  0.058
## Arkansas      1.05 -1.35  0.018  0.022 0.023  0.39 -0.311
## California -4.28  0.14 -0.276  0.025 0.058  0.38 -0.464
```

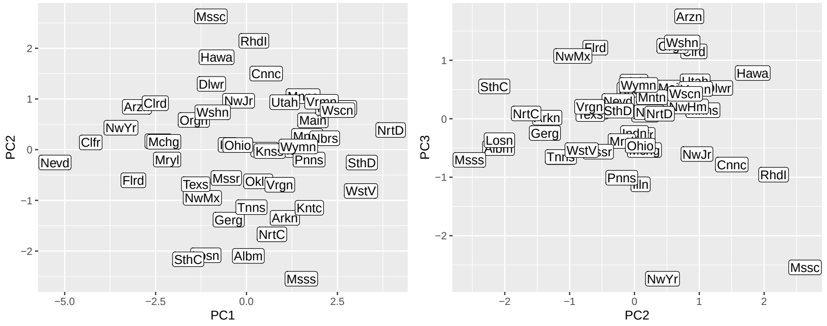
Scores plot

```r
state2 <- abbreviate(state, 4)
state2 |> head(10)
```

```
##     Alabama      Alaska     Arizona    Arkansas  California    Colorado
##      "Albm"      "Alsk"      "Arzn"      "Arkn"      "Clfr"      "Clrd"
## Connecticut    Delaware     Florida     Georgia
##      "Cnnc"      "Dlwr"      "Flrd"      "Gerg"
```

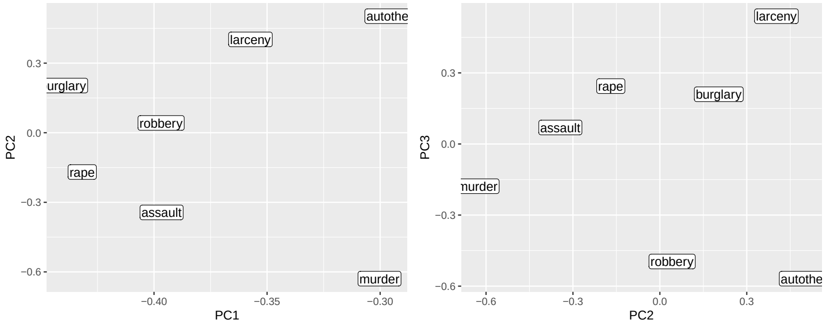
```
do_plot2 <- function(dat, x, y, lab){
    dat <- dat |> as.data.frame()
    dat |> ggplot(aes(x=!!sym(x), y=!!sym(y), label=lab)) +
        geom_point() +
        geom_label(label.padding = unit(0.1, "lines"))
}
```

```
X <- pca$x
p1 <- X  |> do_plot2("PC1", "PC2", state2)
p2 <- X  |> do_plot2("PC2", "PC3", state2)
cowplot::plot_grid(p1, p2, nrow=1)
```



Loading plots

```
W <- pca$rotation
p1 <- W  |> do_plot2("PC1", "PC2", rownames(W))
p2 <- W  |> do_plot2("PC2", "PC3", rownames(W))
plot_grid(p1, p2, nrow=1)
```



Example: NIRmilk

In these data NIR (near infrared) measurements are made at 152 wavelengths on 17 milk samples: Milk runs trough a glass tube. Near infrared light is sent through the tube. The transmittance (fraction of electromagnetic power) at different wavelengths is recorded.

The samples are also analyzed for contents of `fat`, `lactose`, `protein` and `drymatter`.

A natural question is: Can `fat`, `lactose`, `protein` and `drymatter` content be predicted from the NIR measurements (which are easy and cheap to obtain).

PCA is an excellent tool in this connections

```r
nir_milk <- doBy::nir_milk
ynir <- nir_milk$y |> as_tibble()
xnir <- nir_milk$x |> as_tibble()
ynir  |> head(3)
```

```
## # A tibble: 3 x 5
##   sample   fat protein lactose    dm
##   <chr>  <dbl>   <dbl>   <dbl> <dbl>
## 1 s01     4.17    3.64    4.53  13.1
## 2 s02     4.23    3.55    5.56  14.1
## 3 s03     3.90    4.30    5.49  14.5
```

```r
xnir[,1:6] |> head(3)
```

```
## # A tibble: 3 x 6
##   sample   X964   X968   X972   X976  X979
##   <chr>   <dbl>  <dbl>  <dbl>  <dbl> <dbl>
## 1 s01    0.0536 0.0556 0.0608 0.0782 0.109
## 2 s02    0.0693 0.0713 0.0775 0.0980 0.135
## 3 s03    0.0677 0.0703 0.0773 0.0989 0.136
```

```r
ynir0 <- ynir |> select(-sample)
xnir0 <- xnir |> select(-sample)
samp <- xnir$sample
wave <- xnir0 |> colnames()  |> gsub("X", "", x=_)  |> as.numeric()
samp |> head(3)
```

```
## [1] "s01" "s02" "s03"
```

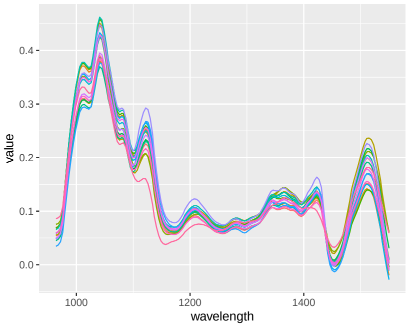
```r
wave |> head(3)
```

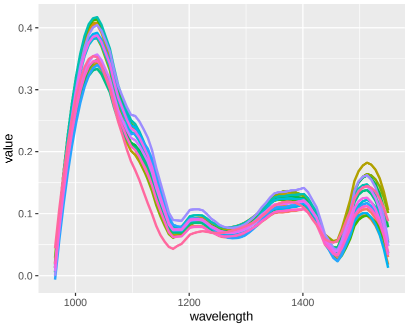
```
## [1] 964 968 972
```

```r
xnir_long <- xnir |>
    pivot_longer(-sample, names_to="wave", values_to="value") |>
    mutate(wavelength = as.numeric(gsub("X", "", wave)))
head(xnir_long, 3)
```

```
## # A tibble: 3 x 4
##   sample wave   value wavelength
##   <chr>  <chr>  <dbl>      <dbl>
## 1 s01    X964  0.0536        964
## 2 s01    X968  0.0556        968
## 3 s01    X972  0.0608        972
```

```r
plot_xnir <- xnir_long |>
    ggplot(aes(x=wavelength, y=value, group=sample, colour=sample))
plot_grid(plot_xnir + geom_line(),
          plot_xnir + geom_smooth(span=.25, se=F), nrow=1)
```



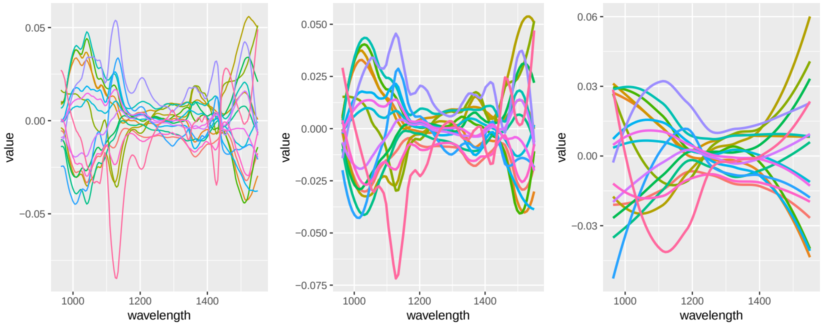


A good question is if we should scale the variables to have the same variance or not.

1. All measurements are the same quantity but at different wavelengths which suggests that scaling might not be necessary.
2. On the other hand, the plot indicates that the variances are different for different wavelengths.

```r
xnir_center <- doBy::scale_df(xnir, center = TRUE, scale = FALSE)

xnir_center_long <-
    xnir_center  |>
    pivot_longer(-sample, names_to = "wave", values_to = "value") |>
    mutate(wavelength = as.numeric(gsub("X", "", wave)))
```

```r
plot_xnir_center <- xnir_center_long |>
    ggplot(aes(wavelength, value, group=sample, colour=sample))

plot_grid(plot_xnir_center + geom_line(),
          plot_xnir_center + geom_smooth(span=.2, se=F),
          plot_xnir_center + geom_smooth(span=.7, se=F), nrow=1)
```



We continue with unscaled data (not terribly important in this case)

```
pca0 <- prcomp(xnir0, rank=6)
summary(pca0)
```

```
## Importance of first k=6 (out of 17) components:
##                          PC1   PC2    PC3     PC4     PC5     PC6
## Standard deviation     0.158 0.122 0.0958 0.02138 0.00488 0.00371
## Proportion of Variance 0.506 0.298 0.1853 0.00923 0.00048 0.00028
## Cumulative Proportion  0.506 0.804 0.9898 0.99898 0.99946 0.99974
```

```
pca_scaled <- prcomp(xnir0, scale=T, rank=6)
summary(pca_scaled)
```

```
## Importance of first k=6 (out of 17) components:
##                          PC1   PC2   PC3    PC4     PC5     PC6
## Standard deviation     8.679 6.934 5.246 1.3434 0.38062 0.24169
## Proportion of Variance 0.492 0.314 0.180 0.0118 0.00095 0.00038
## Cumulative Proportion  0.492 0.807 0.986 0.9983 0.99923 0.99962
```

Hence, 80 % of the total variation in a 150–dimensional data set is explained by the first two principal components and practically all variation is explained by the first three principal components. This is quite a substantial reduction in dimension.

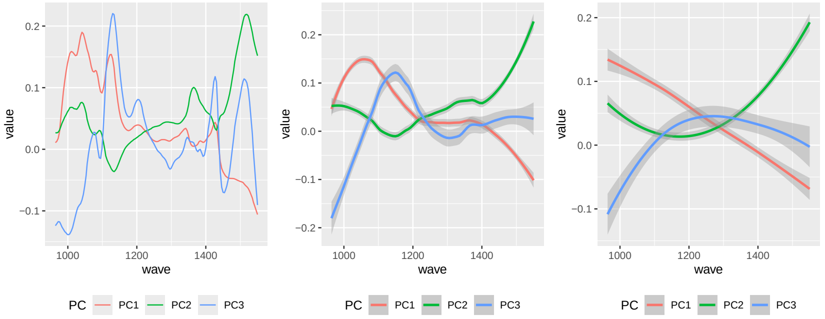
Plotting rotations

Recall: rotations / loadings are the weights given to each measurement when forming the principal components. We plot loadings against wavelengths:

```
rot <- pca0$rotation[, 1:3] |> as.data.frame()
rot$wave <- wave

rot_long <- rot |> pivot_longer(-wave, names_to = "PC")
rot_long |> head(3)
```

```
## # A tibble: 3 x 3
##    wave PC      value
##   <dbl> <chr>   <dbl>
## 1   964 PC1    0.0108
## 2   964 PC2    0.0268
## 3   964 PC3   -0.124
```

```r
plot_rotation <- rot_long |> ggplot(aes(x=wave, y=value, color=PC)) +
    theme(legend.position="bottom")
plot_grid(plot_rotation + geom_line() ,
          plot_rotation + geom_smooth(span=.5),
          plot_rotation + geom_smooth(span=2), nrow=1)
```



Principal component regression (PCR)

Next we turn to the regression problem: We want to predict, say, the `fat` content from the NIR measurements.

We have 17 samples and 153 predictor variables.

In the usual regression setting the model matrix *ynir* is 17×153 (an extra column of 1s corresponding to the intercept is added).

This is an example of a *p larger than n* problem; there are more predictors than observations.

A straight forward alternative is principal component regression or PCR

1. First derive principal components of the explanatory variables and
2. Then use these principal components as explanatory variables.

Combine the first three principal components with the ynirs and regress the ynirs on (some of) the principal components.

```
doBy::truncate0(cor(ynir0, pca0$x[,1:6]), tol=0.4)
```

```
## 4 x 6 sparse Matrix of class "dgCMatrix"
##          PC1  PC2  PC3 PC4 PC5 PC6
## fat     0.47 .    0.84   .   .   .
## protein .    0.61 0.74   .   .   .
## lactose 0.93 .    .      .   .   .
## dm      0.61 .    0.79   .   .   .
```

```
nir_ext <- cbind(pca0$x[, 1:6], ynir0)
nir_ext |> head(3)
```

```
##     PC1   PC2     PC3    PC4     PC5      PC6 fat protein lactose dm
## 1 -0.11 -0.18 -0.0079 0.0074 0.00099  0.00064 4.2     3.6     4.5 13
## 2  0.18 -0.11 -0.0881 0.0069 0.00181  0.00025 4.2     3.5     5.6 14
## 3  0.13  0.12 -0.0454 0.0156 0.01020 -0.00358 3.9     4.3     5.5 14
```

```r
cor(ynir0, pca0$x[,1:4])
```

```
##           PC1     PC2   PC3     PC4
## fat     0.474 -0.2610  0.84 -0.0095
## protein 0.045  0.6146  0.74  0.2364
## lactose 0.928  0.1677 -0.32 -0.0027
## dm      0.614  0.0032  0.79  0.0370
```

Now, we can try to make a (multiple) regression explaining the ynirs not directly in terms of the wavelengths but in terms of the principal components (which in turn are derived from the wavelengths):

```r
m1 <- lm(fat ~ PC1 + PC3 , data = nir_ext)
```

The model explains practically all variation in data:

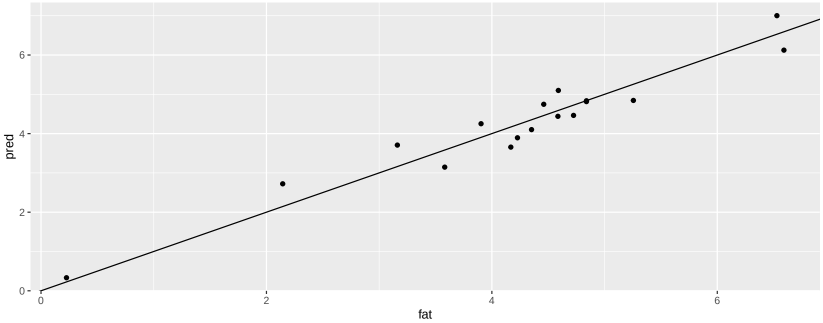
```r
m1 |> broom::glance() |> pander::pander()
```

Table 1: Table continues below

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik
0.931	0.9212	0.4168	94.51	7.414e-09	2	-7.595

AIC	BIC	deviance	df.residual	nobs
23.19	26.52	2.432	14	17

```
nir_ext2 <- nir_ext |> modelr::add_predictions(m1)
nir_ext2 |> ggplot(aes(x=fat, y=pred)) +
    geom_point() + geom_abline(slope=1, intercept=0)
```



Notice: Since the first three PCs explain practically all variation in data we need not include any additional PCs in the regression model. However, it could be the case that not all of the PCs had a significant effect as a predictor, but that is not the case here.

Cross validation

The predictive ability of the regression model above is assessed on the basis of the data and here we should do cross validation. The `pls` package has a function for this purpose.

```
library(pls)
ynir$X <- as.matrix(xnir0)
ynir[1:3,1:4]
```

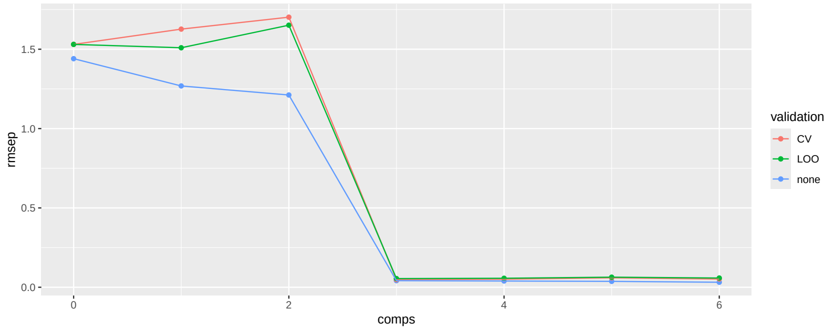
```
## # A tibble: 3 x 4
##   sample   fat protein lactose
##   <chr>  <dbl>   <dbl>   <dbl>
## 1 s01     4.17    3.64    4.53
## 2 s02     4.23    3.55    5.56
## 3 s03     3.90    4.30    5.49
```

```
ynir$X[1:3, 1:5]
```

```
##       X964  X968  X972  X976 X979
## [1,] 0.054 0.056 0.061 0.078 0.11
## [2,] 0.069 0.071 0.078 0.098 0.13
## [3,] 0.068 0.070 0.077 0.099 0.14
```

```r
m3 <- pcr(fat ~ X, data=ynir, scale=F, ncomp=6, validation="none")
r3 <- RMSEP(m3) |> as.data.frame()

m4 <- pcr(fat ~ X, data=ynir, scale=F, ncomp=6, validation="CV",
          segments=5)
r4 <- RMSEP(m4, estimate="CV") |> as.data.frame()

m5 <- pcr(fat ~ X, data=ynir, scale=F, ncomp=6, validation="LOO")
r5 <- RMSEP(m5, estimate="CV") |> as.data.frame()
```

Best predictive results with three principal components:



Whichever evaluation method is used, the RMSEP is smallest when three components are used.

Notice that scores we compute “manually” are the same as those computed by `pcr`:

```
m3$scores |> head(3)
```

```
##   Comp 1 Comp 2  Comp 3 Comp 4  Comp 5   Comp 6
## 1  -0.11  -0.18 -0.0079 0.0074 0.00099  0.00064
## 2   0.18  -0.11 -0.0881 0.0069 0.00181  0.00025
## 3   0.13   0.12 -0.0454 0.0156 0.01020 -0.00358
```

```
pca0$x[, 1:6] |> head(3)
```

```
##         PC1   PC2     PC3    PC4     PC5      PC6
## [1,] -0.11 -0.18 -0.0079 0.0074 0.00099  0.00064
## [2,]  0.18 -0.11 -0.0881 0.0069 0.00181  0.00025
## [3,]  0.13  0.12 -0.0454 0.0156 0.01020 -0.00358
```

Take-home message

- PCR is not a feature selection method as such: It does not extract significant wavelengths and throw the rest away.
- But for these data, it probably makes very little sense to talk about a few significant wavelengths. Instead it is perhaps relevant to look at collections of wavelengths.
- Can see PCA as a necessary preprocessing step (dimension reduction) before regression.
- Very common to do such dimension reductions before other analyses.