VIPs comparison

Paula Solé Vallés

2025-01-11

```r
# Load libraries
library(mixOmics)
```

```
## Loading required package: MASS

## Loading required package: lattice

## Loading required package: ggplot2

##
## Loaded mixOmics 6.30.0
## Thank you for using mixOmics!
## Tutorials: http://mixomics.org
## Bookdown vignette: https://mixomicsteam.github.io/Bookdown
## Questions, issues: Follow the prompts at http://mixomics.org/contact-us
## Cite us:  citation('mixOmics')
```

```r
library(readxl)
library(diffdf)
library(ggplot2)
```

Obtenció de les dades

Carreguem les dades:

- Abans de ser pre-processades

```r
setwd('/Volumes/ftp/Paula Sole')

# Upload files
transcriptomics <- read_excel("./dades/FPKM_tomate_inundacion_all.xlsx")
physiological <-
read_excel("./dades/Summary_Physiological_parameters_lukullus_notabilis_PA_20
19.xlsx", sheet = "Full3")
```

```
## New names:
## • `` -> `...1`
```

```r
metabolites_PA <- read.table("./processed_data/metabolites_PA.txt", sep="\t",
row.names=1, header=T)
metabolites_AR <- read.table("./processed_data/metabolites_AR.txt", sep="\t",
row.names=1, header=T)

# Set the row names for the dataframe
transcriptomics <- as.data.frame(transcriptomics)
```

```
rownames(transcriptomics) <- transcriptomics$Locus
# Remove the first column as it’s now used as row names
transcriptomics <- transcriptomics[-1]

# Divide the transcriptomic dataset by tissue
# Create the "PA" dataset by selecting columns with "PA" in their names
transcriptomics_PA <- transcriptomics[, grepl("PA",
colnames(transcriptomics))]
transcriptomics_PA <- t(transcriptomics_PA)
transcriptomics_PA <- as.data.frame(transcriptomics_PA)

# Create the "AR" dataset by selecting columns with "AR" in their names
transcriptomics_AR <- transcriptomics[, grepl("AR",
colnames(transcriptomics))]
transcriptomics_AR <- t(transcriptomics_AR)
transcriptomics_AR <- as.data.frame(transcriptomics_AR)

# Create the PA dataframe
PA_dataframe <- cbind(physiological, transcriptomics_PA, metabolites_PA)
PA_dataframe <- PA_dataframe[-1]

# Create the AR dataframe
AR_dataframe <- cbind(physiological, transcriptomics_AR, metabolites_AR)
AR_dataframe <- AR_dataframe[-1]
```

- Després de ser pre-processades i normalitzades:

```
PA_log <- read.table("/Volumes/ftp/Paula
Sole/processed_data/PA_dataframe_log.txt", sep="\t")
PA_quantile <- read.table("/Volumes/ftp/Paula
Sole/processed_data/PA_dataframe_quantile.txt", sep="\t")

AR_log <- read.table("/Volumes/ftp/Paula
Sole/processed_data/AR_dataframe_log.txt", sep="\t")
AR_quantile <- read.table("/Volumes/ftp/Paula
Sole/processed_data/AR_dataframe_quantile.txt", sep="\t")
```

Extracció dels VIPs de les dades de PA

```
# VIPs with the raw data
Labels_PA <- as.matrix(rownames(PA_dataframe))

plsda_PA <-plsda(PA_dataframe, as.factor(Labels_PA), ncomp=3)

## Warning in cor(A[[k]], variates.A[[k]]): the standard deviation is zero

VIP_PA <- as.data.frame(vip(plsda_PA))
VIP_PA <- subset(VIP_PA, comp1 > 1 & comp2 > 1)

# VIPs with the normalized data
Labels_PA <- as.matrix(rownames(PA_log))
```

```r
plsda_PA_log <-plsda(PA_log, as.factor(Labels_PA), ncomp=3)

## Warning in cor(A[[k]], variates.A[[k]]): the standard deviation is zero

VIP_PA_log <- as.data.frame(vip(plsda_PA_log))
VIP_PA_log <- subset(VIP_PA_log, comp1 > 1 & comp2 > 1)

Labels_PA <- as.matrix(rownames(PA_quantile))

plsda_PA_quantile <-plsda(PA_quantile, as.factor(Labels_PA), ncomp=3)

## Warning in cor(A[[k]], variates.A[[k]]): the standard deviation is zero

VIP_PA_quantile <- as.data.frame(vip(plsda_PA_quantile))
VIP_PA_quantile <- subset(VIP_PA_quantile, comp1 > 1 & comp2 > 1)
```

Extracció dels VIPs de les dades de AR

```r
# VIPs with the raw data
Labels_AR <- as.matrix(rownames(AR_dataframe))

plsda_AR <-plsda(AR_dataframe, as.factor(Labels_AR), ncomp=3)

## Warning in cor(A[[k]], variates.A[[k]]): the standard deviation is zero

VIP_AR <- as.data.frame(vip(plsda_AR))
VIP_AR <- subset(VIP_AR, comp1 > 1 & comp2 > 1)

# VIPs with the normalized data
Labels_AR <- as.matrix(rownames(AR_log))

plsda_AR_log <-plsda(AR_log, as.factor(Labels_AR), ncomp=3)

## Warning in cor(A[[k]], variates.A[[k]]): the standard deviation is zero

VIP_AR_log <- as.data.frame(vip(plsda_AR_log))
VIP_AR_log <- subset(VIP_AR_log, comp1 > 1 & comp2 > 1)

Labels_AR <- as.matrix(rownames(AR_quantile))

plsda_AR_quantile <-plsda(AR_quantile, as.factor(Labels_AR), ncomp=3)

## Warning in cor(A[[k]], variates.A[[k]]): the standard deviation is zero

VIP_AR_quantile <- as.data.frame(vip(plsda_AR_quantile))
VIP_AR_quantile <- subset(VIP_AR_quantile, comp1 > 1 & comp2 > 1)
```

Comparació dels resultats

```r
# Standardize row names for PA
rownames(VIP_PA) <- tolower(rownames(VIP_PA))
rownames(VIP_PA_log) <- tolower(rownames(VIP_PA_log))
rownames(VIP_PA_quantile) <- tolower(rownames(VIP_PA_quantile))

# Compare raw vs log normalization for PA
print("differences between VIP_PA_log and VIP_PA")

## [1] "differences between VIP_PA_log and VIP_PA"

table(rownames(VIP_PA_log) %in% rownames(VIP_PA))

##
## FALSE  TRUE
##   453  3353

comparison1 <- arsenal::comparedf(VIP_PA, VIP_PA_log)
summary1 <- summary(comparison1) # Extract summary

# Compare raw vs quantile normalization for PA
print("differences between VIP_PA_quantile and VIP_PA")

## [1] "differences between VIP_PA_quantile and VIP_PA"

table(rownames(VIP_PA_quantile) %in% rownames(VIP_PA))

##
## FALSE  TRUE
##   895  2848

comparison2 <- arsenal::comparedf(VIP_PA, VIP_PA_quantile)
summary2 <- summary(comparison2) # Extract summary

# Compare log vs quantile normalization for PA
print("differences between VIP_PA_quantile and VIP_PA_log")

## [1] "differences between VIP_PA_quantile and VIP_PA_log"

table(rownames(VIP_PA_quantile) %in% rownames(VIP_PA_log))

##
## FALSE  TRUE
##   794  2949

comparison3 <- arsenal::comparedf(VIP_PA_log, VIP_PA_quantile)
summary3 <- summary(comparison3) # Extract summary

# Standardize row names for AR
rownames(VIP_AR) <- tolower(rownames(VIP_AR))
rownames(VIP_AR_log) <- tolower(rownames(VIP_AR_log))
rownames(VIP_AR_quantile) <- tolower(rownames(VIP_AR_quantile))

# Compare raw vs log normalization for PA
print("differences between VIP_AR_log and VIP_AR")
```

```
## [1] "differences between VIP_AR_log and VIP_AR"
```

```
table(rownames(VIP_AR_log) %in% rownames(VIP_AR))
```

```
##
## FALSE  TRUE
##   665  1115
```

```
comparison4 <- arsenal::comparedf(VIP_AR, VIP_AR_log)
summary4 <- summary(comparison4) # Extract summary

# Compare raw vs quantile normalization for PA
print("differences between VIP_AR_quantile and VIP_AR")
```

```
## [1] "differences between VIP_AR_quantile and VIP_AR"
```

```
table(rownames(VIP_AR_quantile) %in% rownames(VIP_AR))
```

```
##
## FALSE  TRUE
##   795  1036
```

```
comparison5 <- arsenal::comparedf(VIP_AR, VIP_AR_quantile)
summary5 <- summary(comparison5) # Extract summary

# Compare log vs quantile normalization for PA
print("differences between VIP_AR_log and VIP_AR_quantile")
```

```
## [1] "differences between VIP_AR_log and VIP_AR_quantile"
```

```
table(rownames(VIP_AR_log) %in% rownames(VIP_AR_quantile))
```

```
##
## FALSE  TRUE
##   670  1110
```

```
comparison6 <- arsenal::comparedf(VIP_AR_log, VIP_AR_quantile)
summary6 <- summary(comparison6) # Extract summary
```

Representació gràfica dels resultats

```
# Subset the top 20 VIPs and convert the row names into a column for easier
plotting
top_VIP_PA <- VIP_PA[order(-VIP_PA$comp1), ][1:20, ]
top_VIP_PA$feature <- rownames(top_VIP_PA)

top_VIP_PA_quantile <- VIP_PA_quantile[order(-VIP_PA_quantile$comp1), ][1:20,
]
top_VIP_PA_quantile$feature <- rownames(top_VIP_PA_quantile)

top_VIP_PA_log <- VIP_PA_log[order(-VIP_PA_log$comp1), ][1:20, ]
top_VIP_PA_log$feature <- rownames(top_VIP_PA_log)
```

```r
top_VIP_AR <- VIP_AR[order(-VIP_AR$comp1), ][1:20, ]
top_VIP_AR$feature <- rownames(top_VIP_AR)

top_VIP_AR_quantile <- VIP_AR_quantile[order(-VIP_AR_quantile$comp1), ][1:20,
]
top_VIP_AR_quantile$feature <- rownames(top_VIP_AR_quantile)

top_VIP_AR_log <- VIP_AR_log[order(-VIP_AR_log$comp1), ][1:20, ]
top_VIP_AR_log$feature <- rownames(top_VIP_AR_log)


pdf(file = "VIPs_barplots.pdf", width = 5, height = 5);
par(mfrow = c(2,3));

# Create the bar plot
ggplot(top_VIP_PA, aes(x = reorder(feature, -comp1), y = comp1)) +
  geom_bar(stat = "identity", fill = "darkseagreen") +
  coord_flip() + # Flip the coordinates for better readability
   labs(title = "top VIPs for PA",
       x = "Feature",
       y = "comp1 Value") +
  theme_minimal()

ggplot(top_VIP_PA_quantile, aes(x = reorder(feature, -comp1), y = comp1)) +
  geom_bar(stat = "identity", fill = "darkseagreen") +
  coord_flip() + # Flip the coordinates for better readability
   labs(title = "top VIPs of quantile normalization for PA",
       x = "Feature",
       y = "comp1 Value") +
  theme_minimal()

ggplot(top_VIP_PA_log, aes(x = reorder(feature, -comp1), y = comp1)) +
  geom_bar(stat = "identity", fill = "darkseagreen") +
  coord_flip() + # Flip the coordinates for better readability
   labs(title = "top VIPs of log normalization for PA",
       x = "Feature",
       y = "comp1 Value") +
  theme_minimal()

# Create the bar plot
ggplot(top_VIP_AR, aes(x = reorder(feature, -comp1), y = comp1)) +
  geom_bar(stat = "identity", fill = "burlywood") +
  coord_flip() + # Flip the coordinates for better readability
   labs(title = "top VIPs for AR",
       x = "Feature",
       y = "comp1 Value") +
  theme_minimal()
```

```
ggplot(top_VIP_AR_quantile, aes(x = reorder(feature, -comp1), y = comp1)) +
  geom_bar(stat = "identity", fill = "burlywood") +
  coord_flip() + # Flip the coordinates for better readability
   labs(title = "top VIPs of quantile normalization for AR",
       x = "Feature",
       y = "comp1 Value") +
  theme_minimal()

ggplot(top_VIP_AR_log, aes(x = reorder(feature, -comp1), y = comp1)) +
  geom_bar(stat = "identity", fill = "burlywood") +
  coord_flip() + # Flip the coordinates for better readability
   labs(title = "top VIPs of log normalization for AR",
       x = "Feature",
       y = "comp1 Value") +
  theme_minimal()



dev.off()
```

```
## quartz_off_screen
##                 2
```