Projet Analyse de données

Olivier Berthier

2024-08-21

L'objet de ce rapport consiste à étudier un jeu de données concernant les caractéristiques chimiques de différents vins, auxquels sont associés une note de qualité.

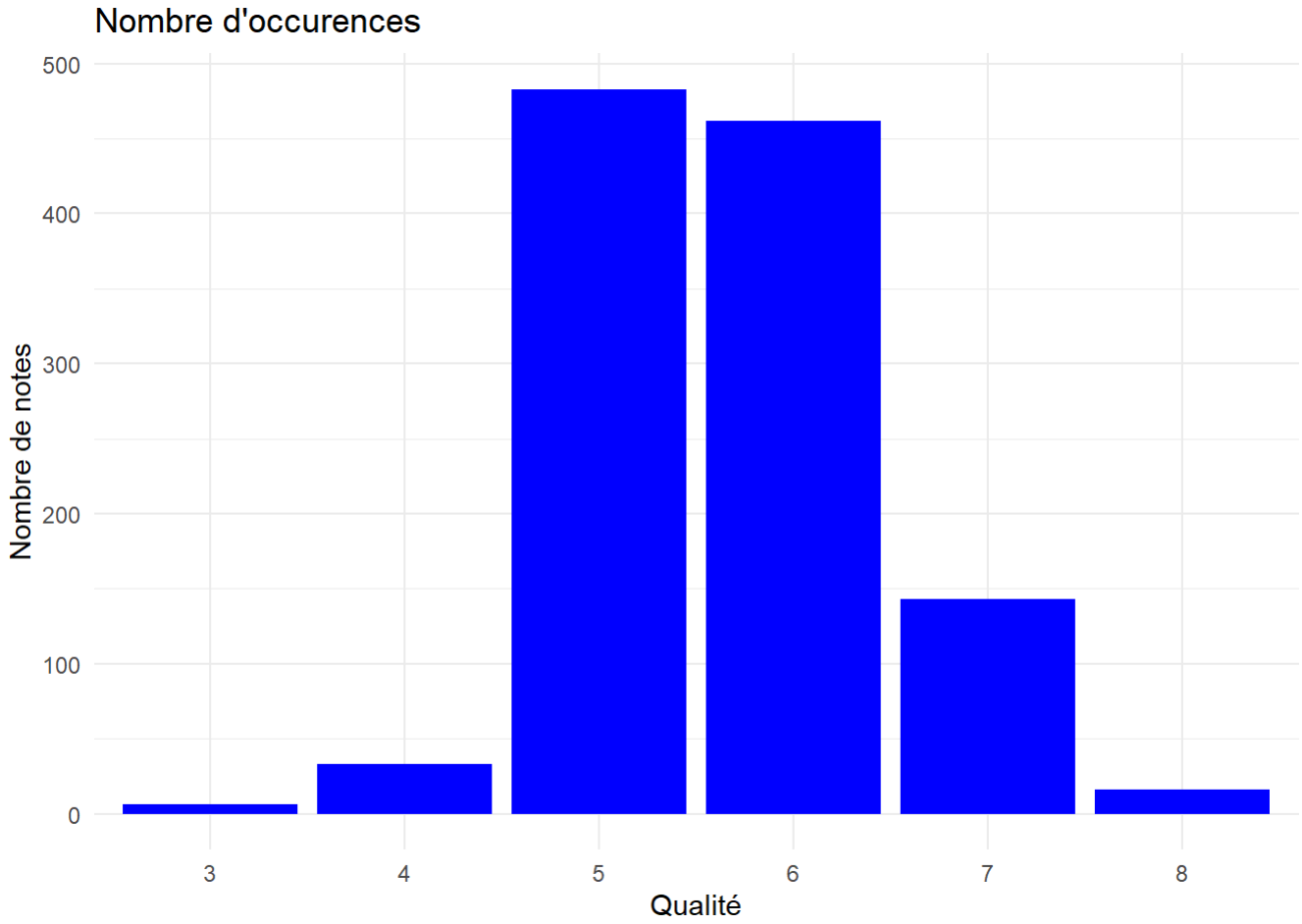
Importation et exploration des données

```
## Rows: 1143 Columns: 12
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (12): fixed acidity, volatile acidity, citric acid, residual sugar, chlo...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
##  fixed acidity    volatile acidity  citric acid     residual sugar
##  Min.   : 4.600   Min.   :0.1200   Min.   :0.0000   Min.   : 0.900
##  1st Qu.: 7.100   1st Qu.:0.3925   1st Qu.:0.0900   1st Qu.: 1.900
##  Median : 7.900   Median :0.5200   Median :0.2500   Median : 2.200
##  Mean   : 8.311   Mean   :0.5313   Mean   :0.2684   Mean   : 2.532
##  3rd Qu.: 9.100   3rd Qu.:0.6400   3rd Qu.:0.4200   3rd Qu.: 2.600
##  Max.   :15.900   Max.   :1.5800   Max.   :1.0000   Max.   :15.500
##    chlorides       free sulfur dioxide total sulfur dioxide    density
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00       Min.   :0.9901
##  1st Qu.:0.07000   1st Qu.: 7.00       1st Qu.: 21.00       1st Qu.:0.9956
##  Median :0.07900   Median :13.00       Median : 37.00       Median :0.9967
##  Mean   :0.08693   Mean   :15.62       Mean   : 45.91       Mean   :0.9967
##  3rd Qu.:0.09000   3rd Qu.:21.00       3rd Qu.: 61.00       3rd Qu.:0.9978
##  Max.   :0.61100   Max.   :68.00       Max.   :289.00       Max.   :1.0037
##        pH          sulphates         alcohol         quality
##  Min.   :2.740   Min.   :0.3300   Min.   : 8.40   Min.   :3.000
##  1st Qu.:3.205   1st Qu.:0.5500   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.310   Median :0.6200   Median :10.20   Median :6.000
##  Mean   :3.311   Mean   :0.6577   Mean   :10.44   Mean   :5.657
##  3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10   3rd Qu.:6.000
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :8.000
```

```
## [1] 1143   12
```

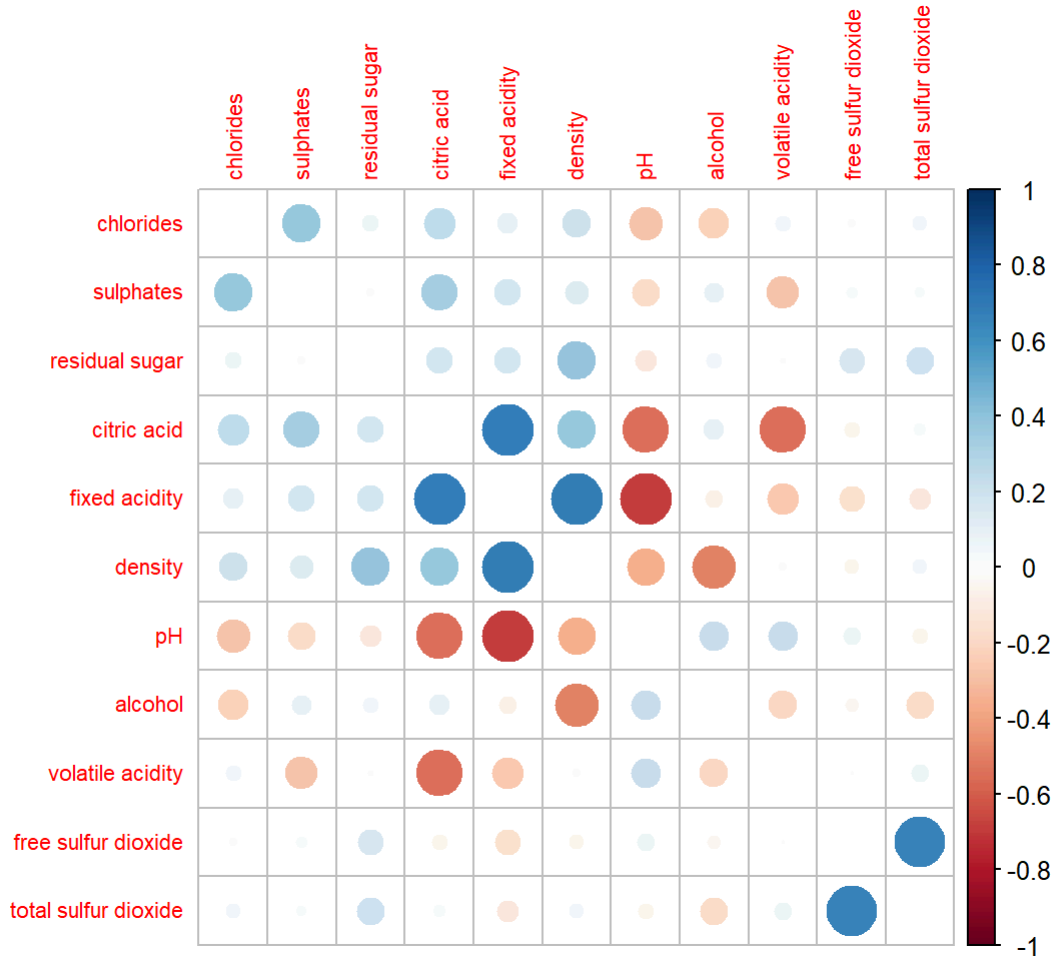
```
quality_freq <- as.data.frame(table(vin$quality))
colnames(quality_freq) <- c("Quality", "Frequency")

ggplot(quality_freq, aes(x = Quality, y = Frequency)) +
  geom_bar(stat = "identity", fill = "blue") +
  labs(title = "Nombre d'occurences",
       x = "Qualité",
       y = "Nombre de notes") +
  theme_minimal()
```



Une grande majorité des vins sont notés 5 ou 6 et très peu reçoivent une valeur “extrême” (3 ou 8).

Dans cette partie, nous ne nous intéressons qu’aux 11 premières variables, nous excluons donc la variable *quality* au sein d’un nouveau dataframe **vin_11**.
Heatmap pour afficher les corrélations entre les variables (l’argument *hclust* affiche les variables dans la matrice par cluster):

```
vin_11 <- vin %>%              # nouveau dataframe sans la variable "quality"
dplyr::select(1:11)

corrplot::corrplot(cor(vin_11),
         order = "hclust",
         tl.cex=0.7,
         diag =FALSE)
```



Les fortes corrélations (>0.5) concernent principalement les variables de même famille chimique (acidité, dioxyde de souffre).
Peu de groupes sont mis en évidence par l'argument *cluster* de *corplot*.

ACP

```
#standardisation dans vin_11_s
vin_11_s <- scale(vin[, 1:11])
acp_vin <- PCA(vin_11_s, scale.unit = FALSE, ncp = 11, graph=FALSE)  # montre que le scale un
it est automatique
```

Afin d'éviter le biais dû à l'échelle des variables, nous standardisons ici les variables dans le dataframe **vin_11_s**. Chaque variable est donc transformée pour qu'elle ait une moyenne de 0 et un écart-type de 1. Sans cela, celles avec des variances plus élevées pourraient excessivement dominer et fausser l'analyse. Les standardiser les rend en quelque sorte "compatibles".

Notre code crée autant de composantes principales qu'il y a de variables, soit 11 composantes.
Voyons lesquelles retenir pour notre analyse.

Selection des composantes

Critère de Kaiser

Le critère de Kaiser retient uniquement les composantes dont la valeur propre est supérieure à 1. Autrement dit, selon ce critère, une composante, pour être retenue, doit expliquer davantage de variance qu'une variable d'origine standardisée (qui a une variance de 1).

```
# Extraire les valeurs propres de l'ACP
valeus_propres <- acp_vin$eig[, 1]  # la première colonne contient les valeurs propres
print(valeus_propres)
```

```
##     comp 1     comp 2     comp 3     comp 4     comp 5     comp 6     comp 7
## 3.15339664 1.87661813 1.57227532 1.21044484 0.95737522 0.66478705 0.55710956
##     comp 8     comp 9    comp 10    comp 11
## 0.41869216 0.34419257 0.17707639 0.05840834
```

```
# conserver les composantes avec des valeurs propres >= 1
kaiser <- sum(valeus_propres >= 1)

cat("Composantes principales à retenir selon le critère de Kaiser :", kaiser, "\n")
```

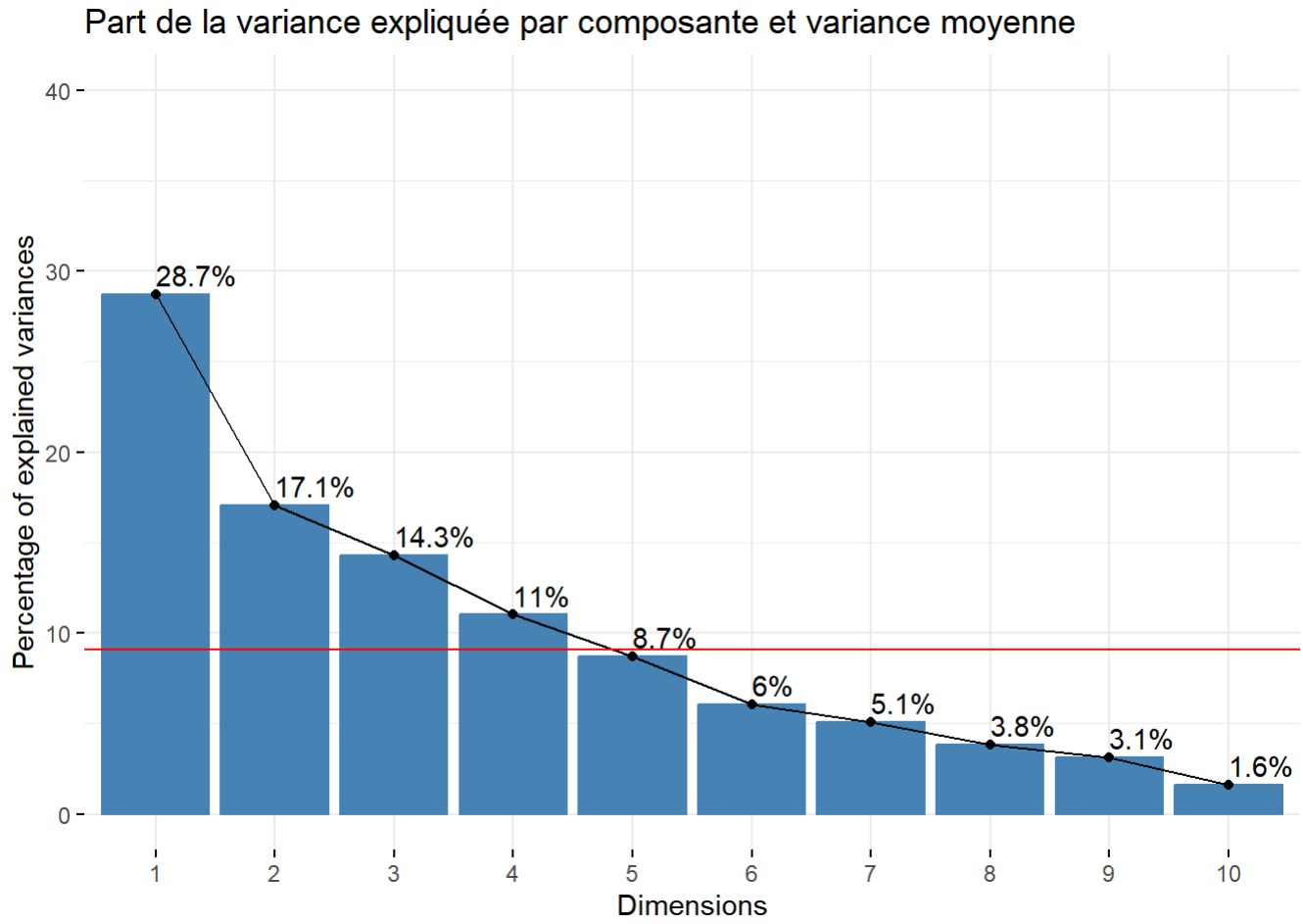
```
## Composantes principales à retenir selon le critère de Kaiser : 4
```

4 composantes sont retenues.

Comparaison avec la variance moyenne

Cette méthode retient uniquement les composantes qui expliquent plus que la moyenne de la variabilité totale du jeu de données.
Alors que le critère de Kaiser fixe un seuil strict (valeur propre > 1), l'approche basée sur la moyenne permet une évaluation plus fine. Ainsi, une composante peut être retenue même si sa valeur propre est inférieure à 1.

```
# calcul de la variance moyenne
p<- 11
val_propres <- acp_vin$eig %>%
  as.data.frame() %>%
  rename_all(.funs = function(x) str_replace_all(x, " ", "_")) %>%
  mutate(comp = 1:p)
moy_val_propres <- mean(val_propres$eigenvalue)

# affichage de la part de la variance expliquée par composante + variance moyenne
fviz_screeplot(acp_vin, addlabels = TRUE, ylim = c(0, 40)) +
  geom_hline(yintercept = moy_val_propres*100 / p, linetype = 1, color = "red") +
  labs(title = "Part de la variance expliquée par composante et variance moyenne")
```



Ici, les 4 premières composantes valident le critère, la 5e se situant à la limite.

Variance Expliquée Cumulée

Une autre méthode consiste à définir un seuil (par exemple, 80% ou 90%) de variance cumulée par rapport à la variance totale.

Au seuil de 80%, 4 composantes sont assurément retenues; la 5e se situe à la limite de notre seuil et peut être retenue ou écartée.
Au seuil de 90%, 6 composantes sont retenues.

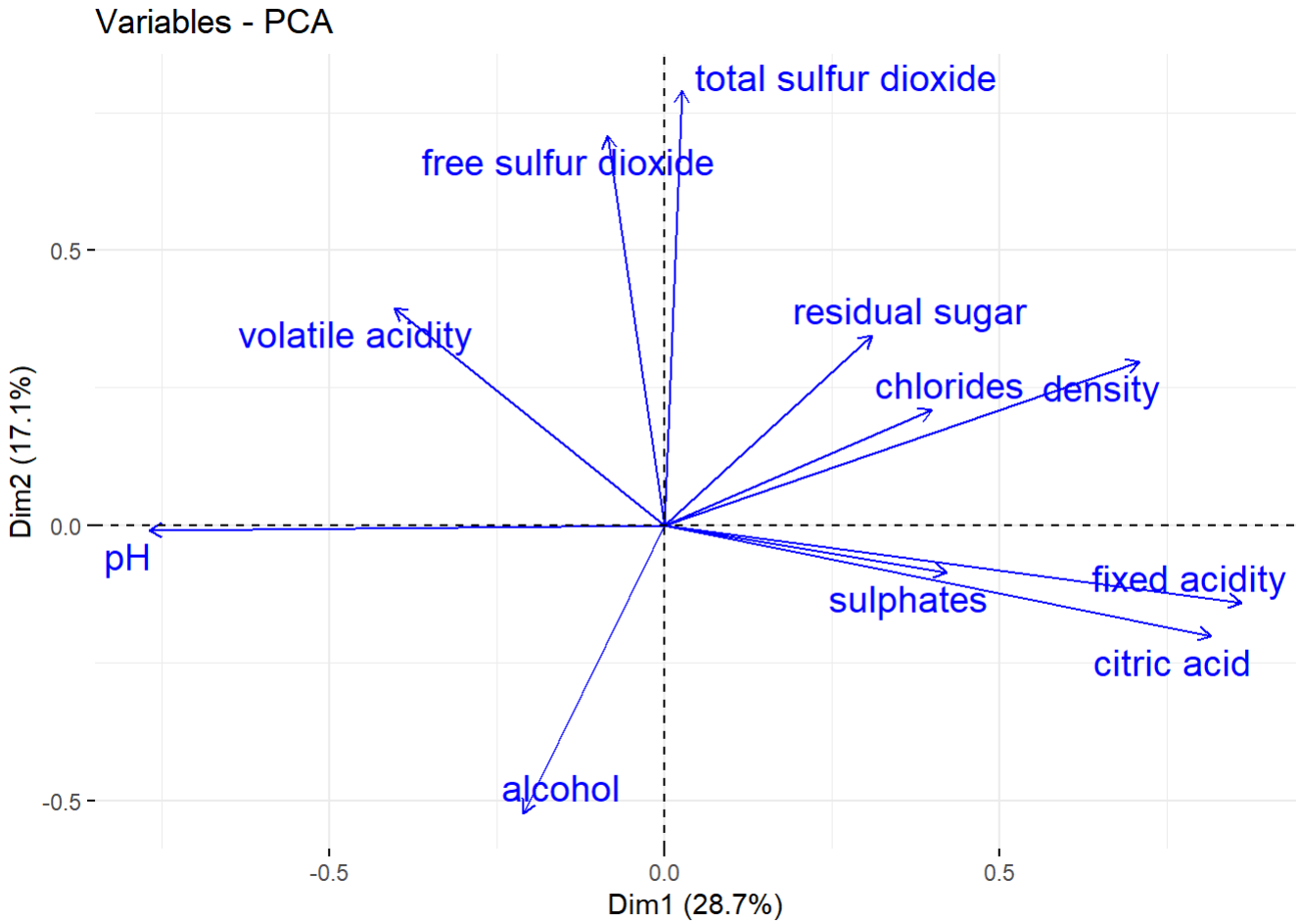
Conclusion: Chaque méthode a ses avantages et ses inconvénients. Le critère de Kaiser est simple mais peut être restrictif.
La comparaison avec la moyenne des valeurs propres est plus flexible mais peut être moins intuitive.
L'approche basée sur la variance expliquée cumulée est visuellement informative et permet d'adapter le seuil selon les besoins spécifiques de l'analyse.

On voit que ces différentes méthodes arrivent dans notre cas à des résultats similaires.

Visualisation des variables

Axes 1 et 2

```
acp_var_12 <- fviz_pca_var(acp_vin,
             axes = c(1,2),
             repel = TRUE,
             labelsize = 5,
             col.var = "blue")
acp_var_12
```

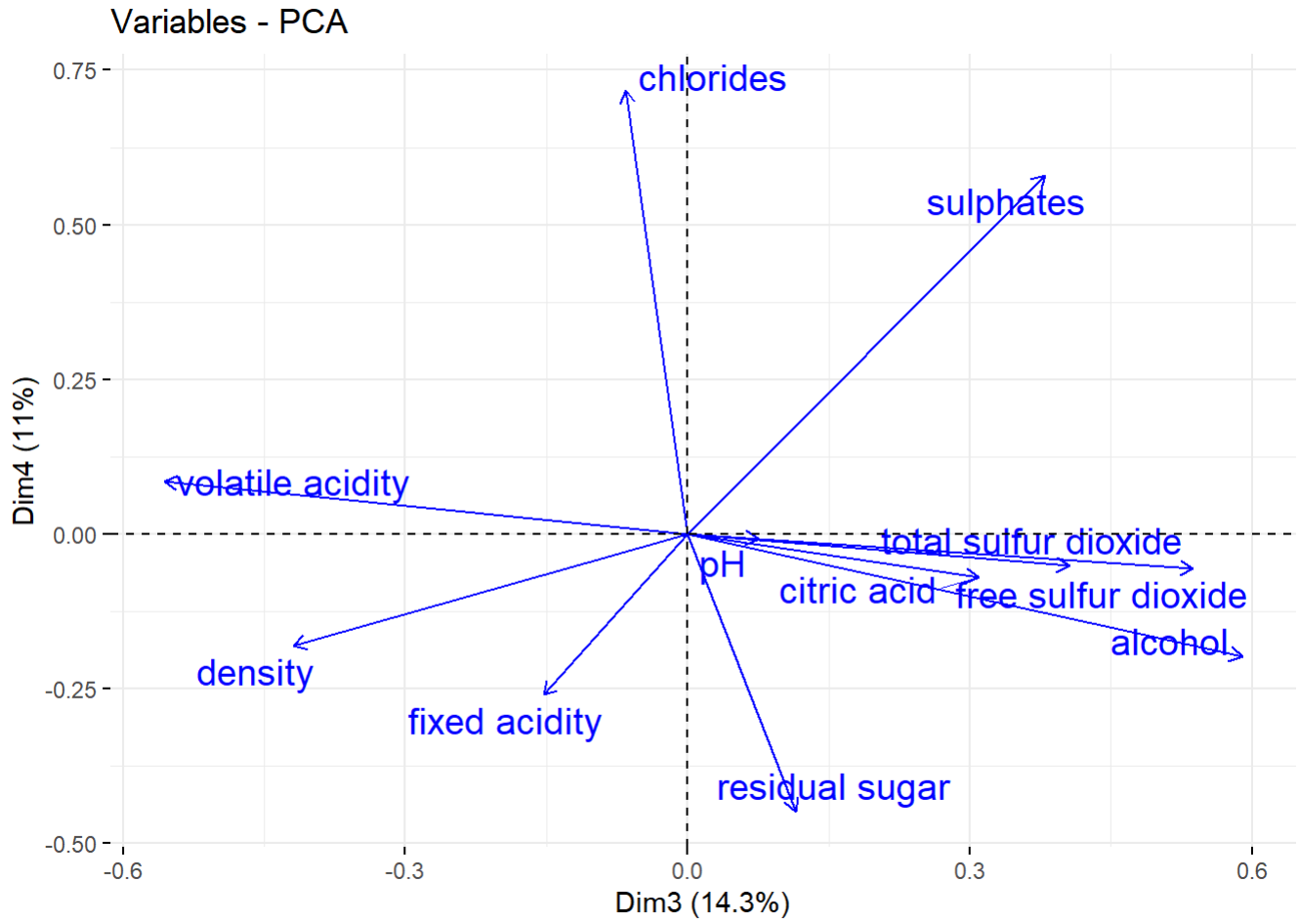


La dimension 1 (28.7% de la variance totale) sépare les variables principalement en fonction de l'acidité: à droite *fixed acid* et *citric acid*, à gauche *PH*. (Un fort PH correspond à un vin de nature "basique", les vins de nature acide ont un PH plus faible, que ces variables soient anti-corrélées est donc logique.)

La dimension 2 (17.1% de la variance totale) oppose principalement les variables en fonction du taux de *total sulfur dioxide* et *free sulfur dioxide* d'un côté, et du pourcentage d'alcool de l'autre.
Le dioxyde de souffre est en général lié à la protection et à la conservation du vin (il améliore la protection contre l'oxydation et les bactéries). Un vin avec un haut niveau d'alcool pourrait nécessiter moins de SO_2 pour atteindre une stabilité acceptable.

Axes 3 et 4

```
acp_var_34 <- fviz_pca_var(acp_vin,
             axes = c(3,4),
             repel = TRUE,
             labelsize = 5,
             col.var = "blue")
acp_var_34
```



Les variables qui s'opposent le plus selon la dimension 3 (14.3% de la variance totale) sont le dioxyde de souffre et l'alcool d'un côté, et la *volatile acidity* de l'autre. Ces trois variables sont les seules dont les projections sur l'axe des abscisses sont supérieures à 0.5 (leur cosinus).

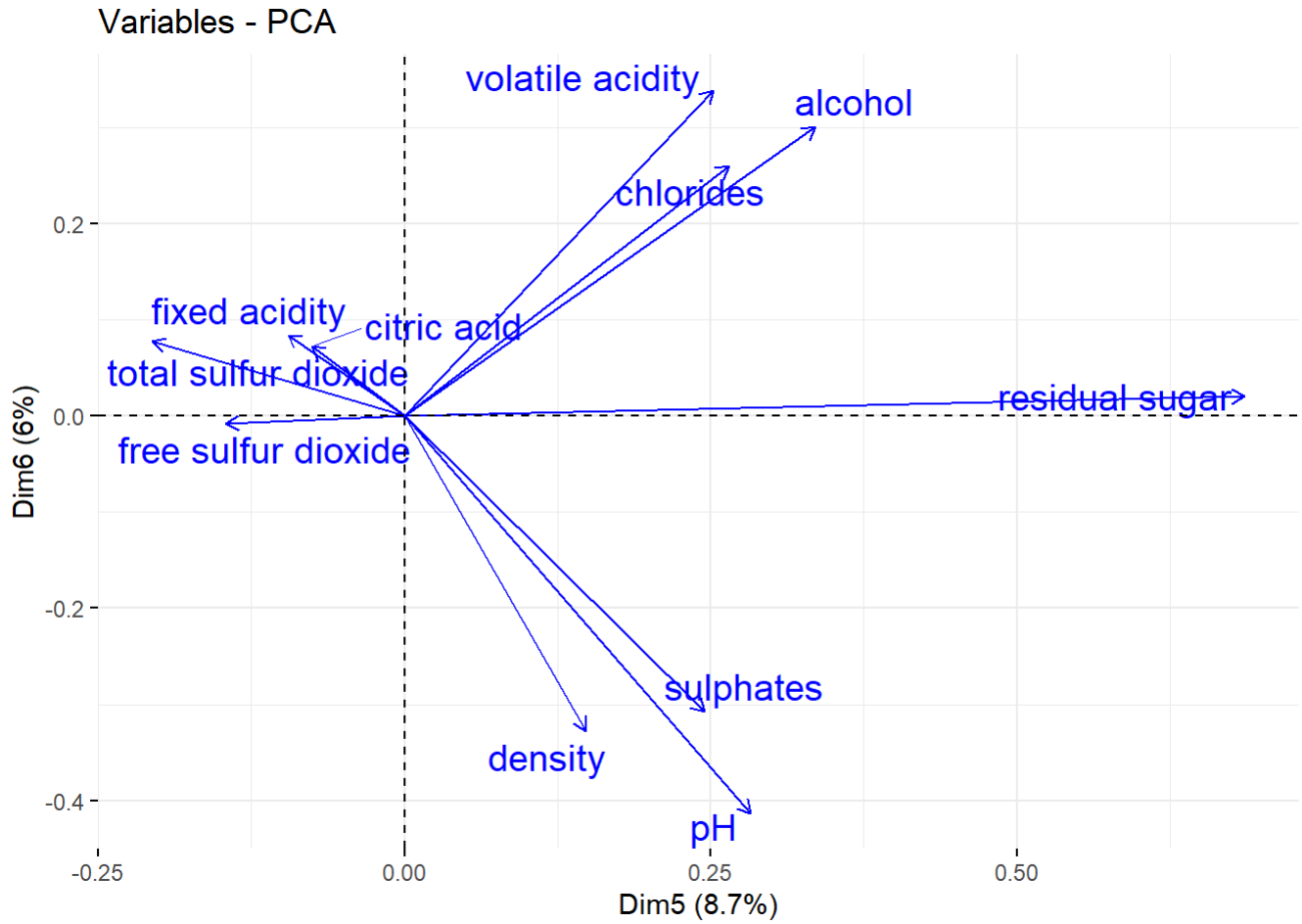
La dimension 4 (11% de la variance totale) est elle dominée par les variables *chlorides* et *sulphates*, celles-ci participant dans le même sens à cette dimension. La variable *chlorides* a le plus grand sinus (environ 0.7) et nous pouvons noter qu'elle est presque neutre sur la dimension 3.

Notons également que la variable *PH*, déterminante dans la dimension 1, est ici neutre dans les dimensions 3 et 4 (ainsi que dans la 2).

Axes 5 et 6

Même si nos critères ont tendance à ne retenir que 4 dimensions, observons les dimensions 5 et 6 et voyons si elles peuvent aider à l'analyse de notre jeu de données:

```
acp_var_56 <- fviz_pca_var(acp_vin,
             axes = c(5,6),
             repel = TRUE,
             labelsize = 5,
             col.var = "blue")
acp_var_56
```

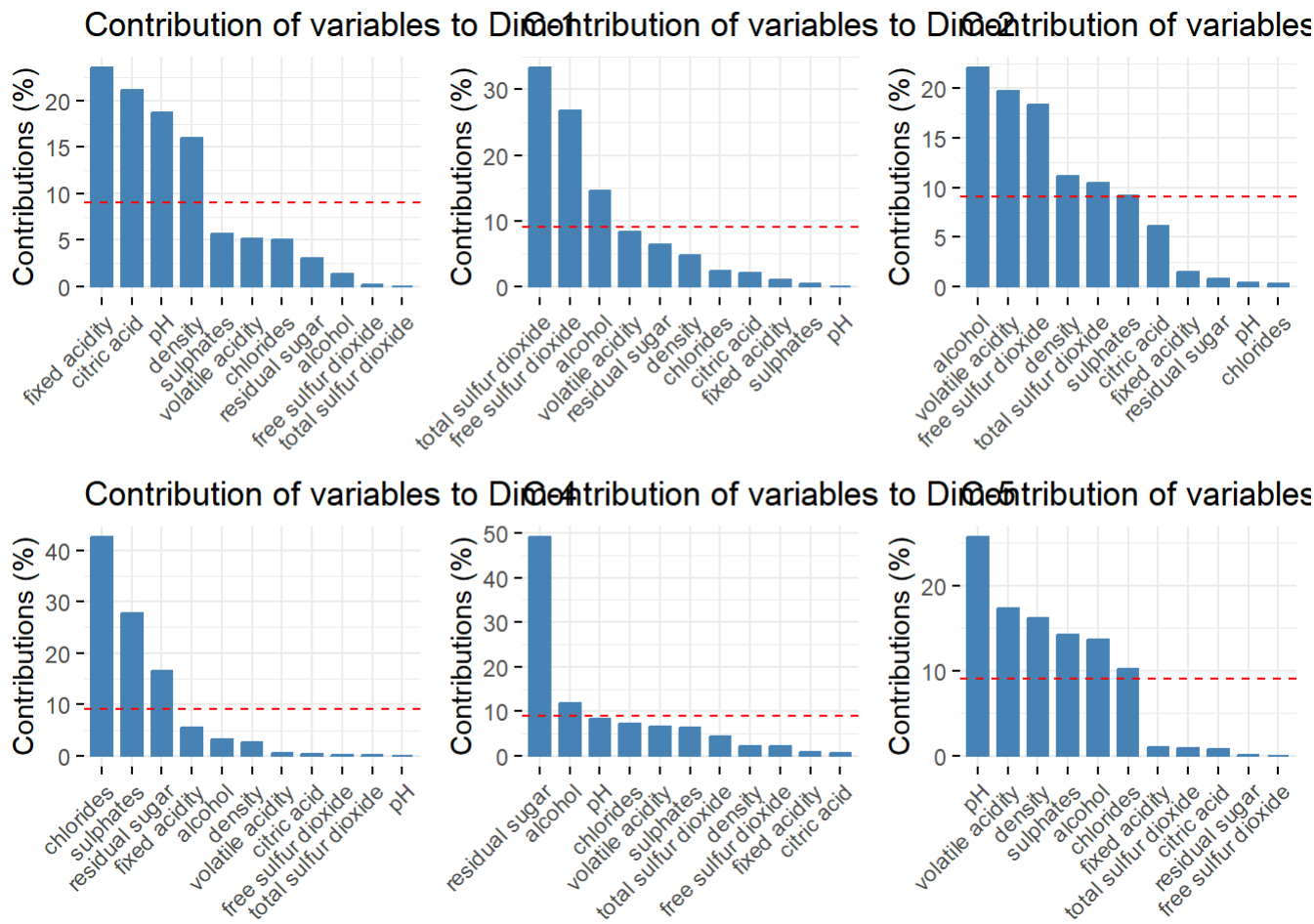


Seule la variable *residual sugar* possède une projection sur les axes assez importante avec un cosinus d'environ 0.7 (donc sur l'axe des abscisses) dans la dimension 5 (qui représente 8.7% de la variance totale).

La dimension 6 (6% de la variance totale) oppose principalement 2 groupes (*volatile acidity*, *alcohol* et *chlorides* d'une part et *PH*, *density* et *sulphates* de l'autre).
L'interprétation est rendue ici difficile au vu de "l'éparpillement" des variables qui construisent cette dimension. Mais cette interprétation n'est pas des plus décisive étant donné le niveau relativement faible de la variance expliquée par la dimension 6.

Contribution des différentes variables dans les 4 premières composantes

```
plot1 <- fviz_contrib(acp_vin, choice = "var", axes = 1)
plot2 <- fviz_contrib(acp_vin, choice = "var", axes = 2)
plot3 <- fviz_contrib(acp_vin, choice = "var", axes = 3)
plot4 <- fviz_contrib(acp_vin, choice = "var", axes = 4)
plot5 <- fviz_contrib(acp_vin, choice = "var", axes = 5)
plot6 <- fviz_contrib(acp_vin, choice = "var", axes = 6)

grid.arrange(plot1, plot2, plot3, plot4, plot5, plot6, ncol = 3, nrow = 2)
```

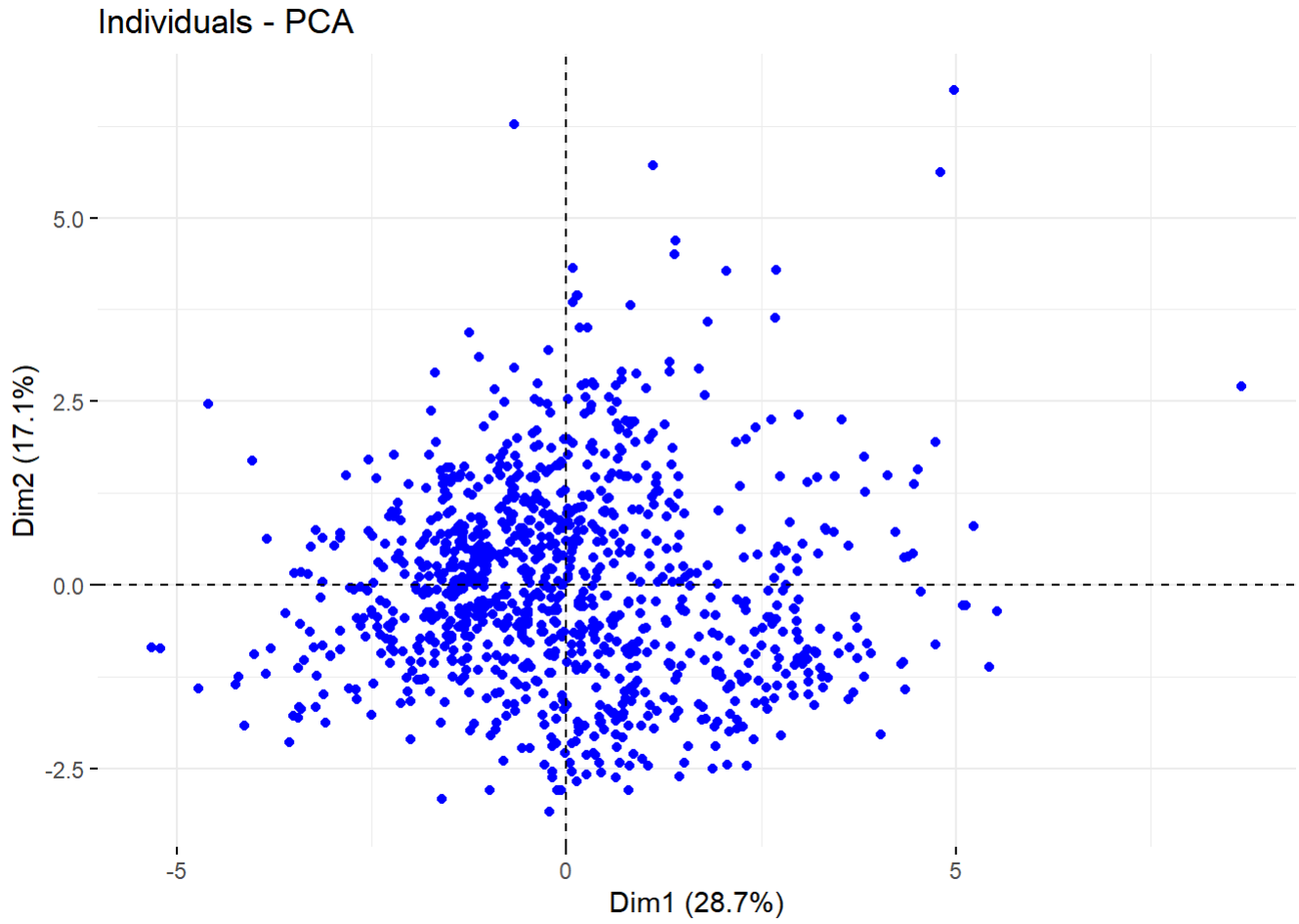


Cette représentation permet de visualiser, cette fois-ci, une dimension à la fois, la contribution des variables aux différentes dimensions.
Elle a l'avantage d'être très lisible mais ne permet pas de détecter les interactions entre les dimensions - contrairement au cercle des corrélations-, ni les relations des variables au sein de la dimension.

Visualisation des individus

Axes 1 et 2

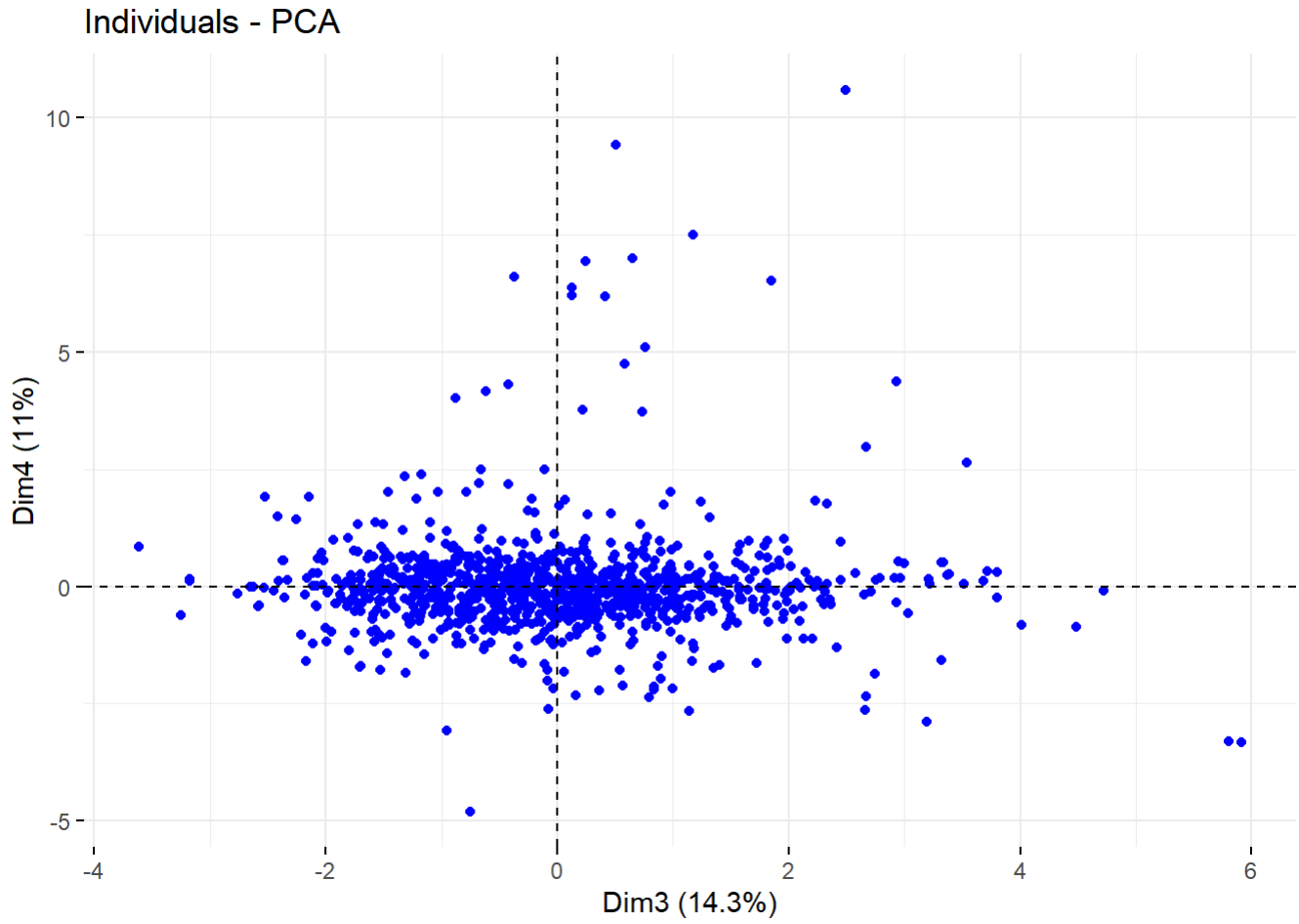
```
fviz_pca_ind(acp_vin,
             axes = c(1,2),
             label = "none",
             col.ind = "blue")
```



Nous n'affichons pas les *labels*, les vins étant ici numérotés et non nommés: cela n'apporterait pas d'informations supplémentaires.
Notons une forte dispersion des observations selon les dimensions 1 et 2.
Nous pouvons simplement remarquer une densité plus forte à gauche de l'axe des ordonnées (soit selon la dimension 1) et une répartition plus étalée à droite de cet axe.
Aucun groupe ne se détache réellement mais nous pouvons noter la présence de quelques *outliers* surtout dans le cadran 1 (en haut à droite ou cadran nord-est). Ces individus sont donc très "forts" dans les 2 premières dimensions et contribuent fortement à la construction de ces mêmes dimensions. On peut donc s'attendre à ce que ces individus aient un faible PH et un taux important en dioxyde de souffre.

Axes 3 et 4

```
fviz_pca_ind(acp_vin,
             axes = c(3,4),
             label = "none",
             col.ind = "blue")
```

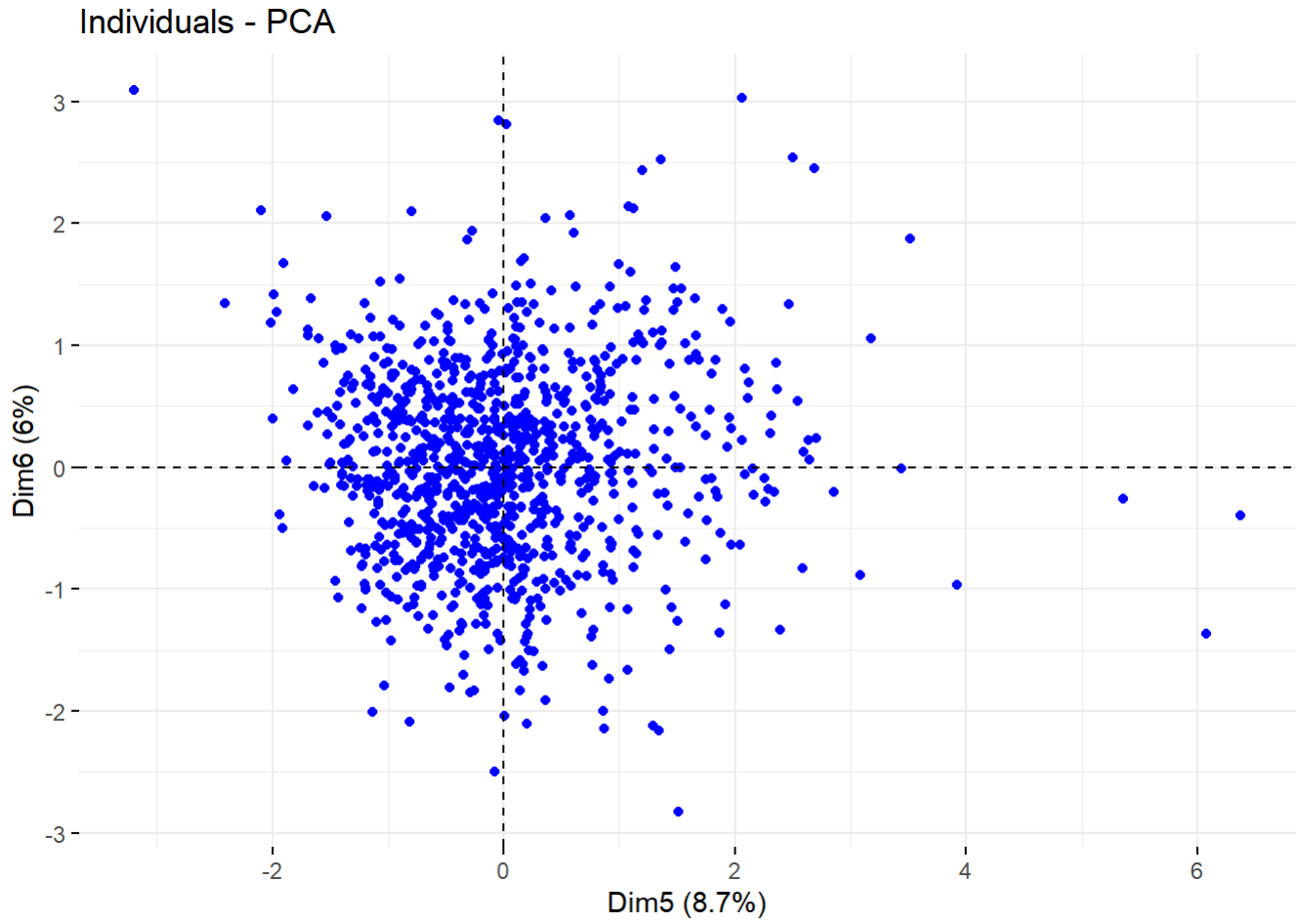


Les individus sont très étalés selon la dimension 1, aucun groupe n'apparaît clairement et nous observons au contraire une répartition assez homogène et symétrique autour de zéro.
Cependant, quelques outliers, surtout à droite, indiquent que certains vins sont fortement associés à cette dimension et peuvent être considérés comme de forts contributeurs à la composante.
On peut s'attendre à ce que ces individus soient "forts" en dioxyde de souffre et alcool.

La dimension 4 est au contraire très "compressée", la plupart des individus sont rassemblés autour de sinus 0. Une vingtaine de forts contributeurs se détachent et sont plus fortement associés à cette dimension, sans toutefois former un groupe homogène. On peut s'attendre à ce que ces individus soient "forts" en *chlorides* et *sulphates*, et "faibles" en *residual sugar*.

Axes 5 et 6

```
fviz_pca_ind(acp_vin,
             axes = c(5,6),
             label = "none",
             col.ind = "blue")
```

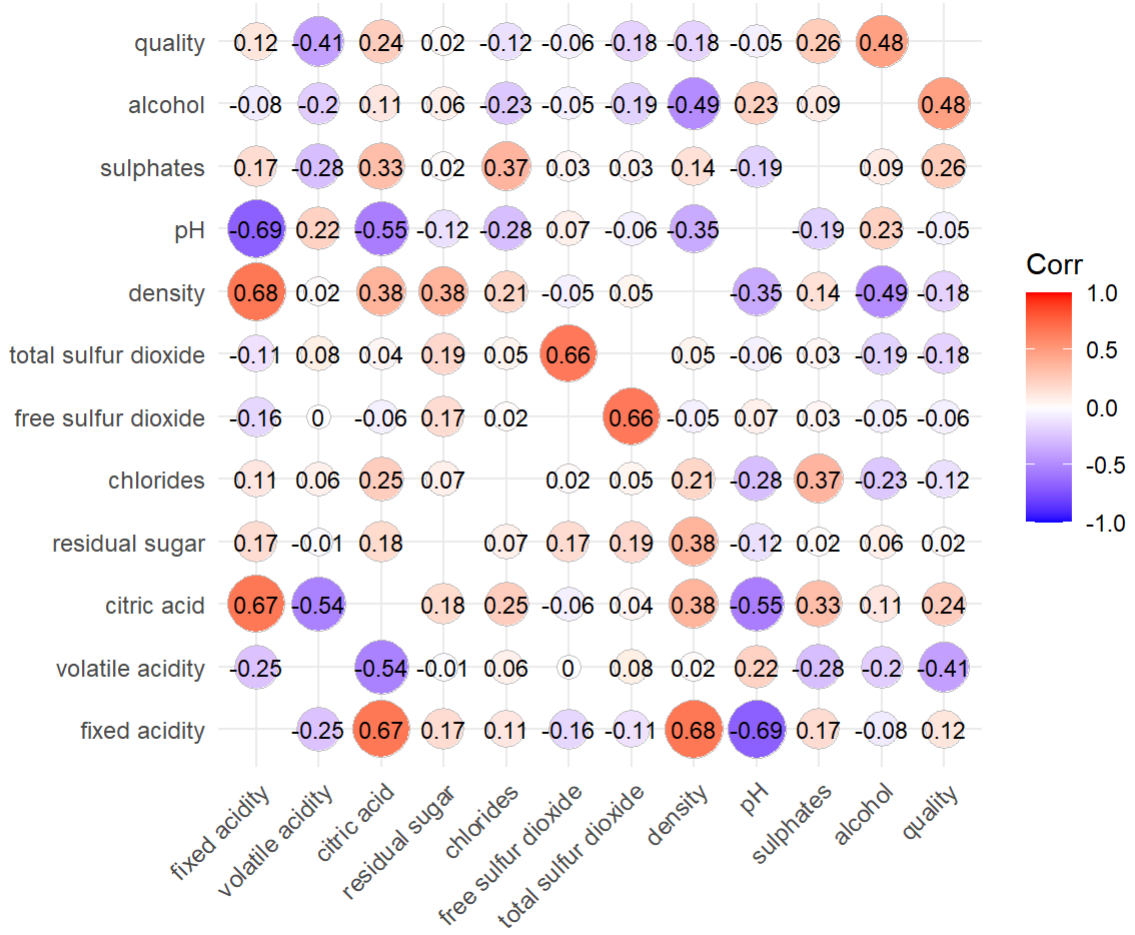


Gardons à l'esprit que ces 2 dimensions sont moins déterminantes.
Les individus sont essentiellement rassemblés autour de zéro, et la densité diminue progressivement en s'en éloignant. 3 *outliers* sont fortement associés à la direction positive de la dimension 1 (acidité).

Les variables chimiques expliquent-elles la qualité ?

Regardons maintenant les corrélations en intégrant la varible cible *quality*:

```
ggcorrplot(cor(vin),
           method = "circle",
           lab_size = 3,
           lab = TRUE,
           show.diag=FALSE,
           tl.cex = 9)
```



La note de qualité est surtout corrélée positivement avec les variables *alcohol* (0.48), *volatile acidity* (0.41) et, dans une moindre mesure, *sulphates* et *citric acid* (respectivement 0.26 et 0.24).

Régression linéaire sur les composantes

Une méthode consiste à pratiquer une simple régression linéaire sur les composantes:

```
coord_acp <- acp_vin$ind$coord        # coordonnées des composantes principales
vin_reg_lin <- data.frame(quality = vin$quality, coord_acp)  # nouveau dataframe
modele <- lm(quality ~ ., data = vin_reg_lin)    # régression
summary(modele)
```

```
##
## Call:
## lm(formula = quality ~ ., data = vin_reg_lin)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.49977 -0.36903 -0.04658  0.43956  2.00117
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.65704    0.01895 298.585  < 2e-16 ***
## Dim.1        0.04979    0.01067   4.667 3.42e-06 ***
## Dim.2       -0.23867    0.01383 -17.257  < 2e-16 ***
## Dim.3        0.26057    0.01511  17.245  < 2e-16 ***
## Dim.4       -0.05057    0.01722  -2.937 0.003385 **
## Dim.5        0.07019    0.01936   3.625 0.000302 ***
## Dim.6       -0.01529    0.02324  -0.658 0.510572
## Dim.7        0.08232    0.02538   3.243 0.001217 **
## Dim.8       -0.07903    0.02928  -2.699 0.007055 **
## Dim.9       -0.11445    0.03229  -3.544 0.000410 ***
## Dim.10      -0.11053    0.04502  -2.455 0.014242 *
## Dim.11      -0.07194    0.07839  -0.918 0.358991
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6405 on 1131 degrees of freedom
## Multiple R-squared:  0.3742, Adjusted R-squared:  0.3682
## F-statistic: 61.49 on 11 and 1131 DF,  p-value: < 2.2e-16
```

Beaucoup de composantes semblent avoir une influence significative sur la note du vin. Les plus gros coefficients sont dans l'ordre décroissant: les dimensions 3,2,9 et 10. Notons que la dimension 10 est moins significative mais sa p-valeur reste cependant acceptable (inférieure à 0.05).
Notre R^2 est relativement faible. Le modèle semble peu adapté.

Une régression ordinale est sans doute plus adaptée ici, la variable *quality* étant une note (valeur entière ordonnée) et non une variable continue.

Régression ordinale sur les composantes principales

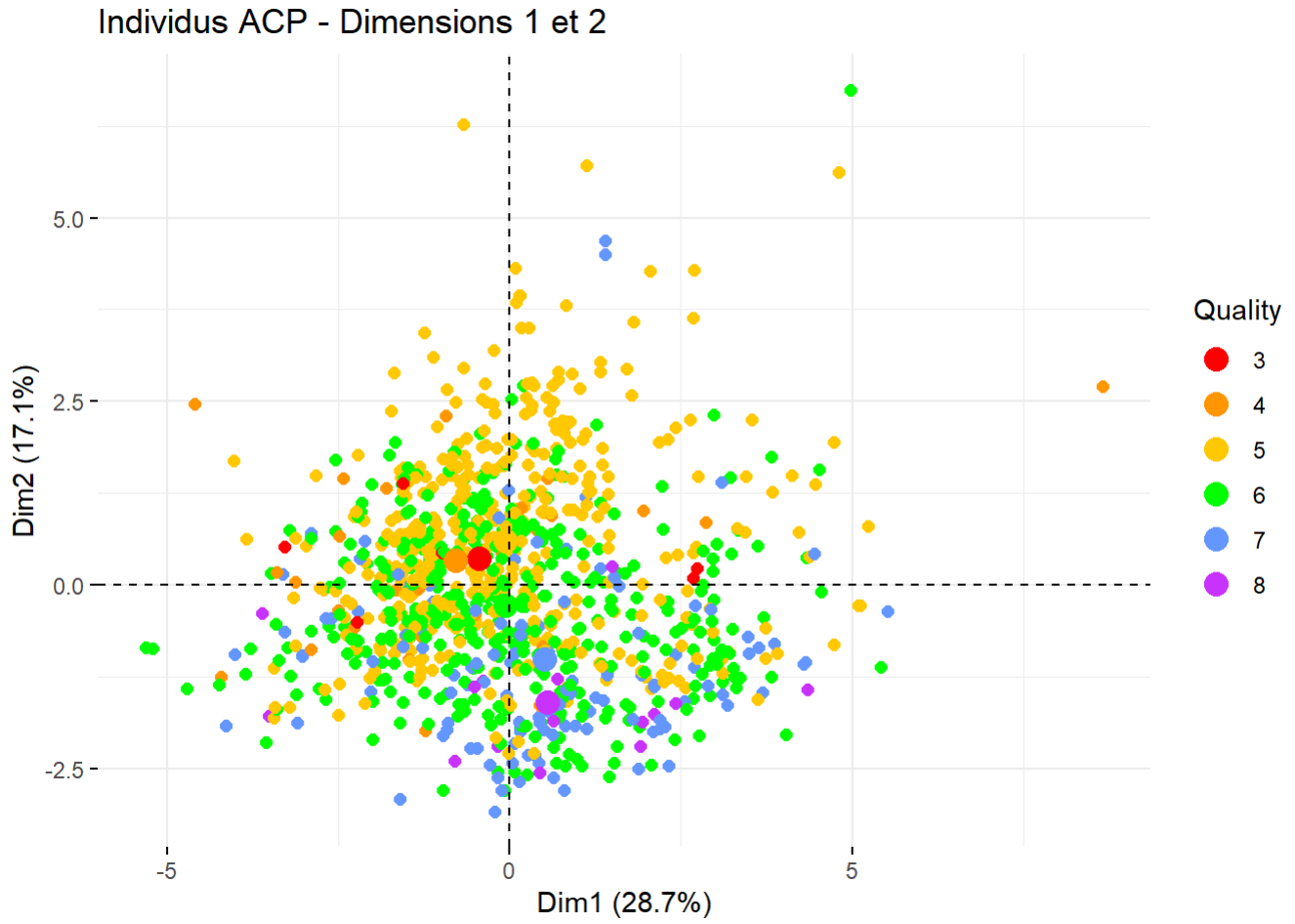
```
df_vin_reg_ord <- vin_reg_lin
df_vin_reg_ord$quality <- factor(vin_reg_lin$quality, ordered = TRUE)  # dataframe pour régre
ssion ordinale
modele_ord <- clm(quality ~ ., data = df_vin_reg_ord)
summary(modele_ord)
```

```
## formula:
## quality ~ Dim.1 + Dim.2 + Dim.3 + Dim.4 + Dim.5 + Dim.6 + Dim.7 + Dim.8 + Dim.9 + Dim.10 +
Dim.11
## data:    df_vin_reg_ord
##
##  link  threshold nobs logLik   AIC     niter max.grad cond.H
##  logit flexible  1143 -1081.89 2195.78 7(0)  2.55e-07 1.5e+02
##
## Coefficients:
##        Estimate Std. Error z value Pr(>|z|)
## Dim.1   0.15777    0.03500   4.508 6.54e-06 ***
## Dim.2  -0.75616    0.04943 -15.299  < 2e-16 ***
## Dim.3   0.80387    0.05489  14.644  < 2e-16 ***
## Dim.4  -0.15223    0.05788  -2.630  0.00854 **
## Dim.5   0.27262    0.06326   4.309 1.64e-05 ***
## Dim.6  -0.07355    0.07490  -0.982  0.32607
## Dim.7   0.24445    0.08198   2.982  0.00287 **
## Dim.8  -0.22715    0.09399  -2.417  0.01566 *
## Dim.9  -0.40829    0.10478  -3.897 9.75e-05 ***
## Dim.10 -0.40133    0.14478  -2.772  0.00557 **
## Dim.11 -0.12492    0.25179  -0.496  0.61981
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 3|4 -6.16325    0.41762 -14.758
## 4|5 -4.21762    0.18006 -23.424
## 5|6 -0.36339    0.07242  -5.018
## 6|7  2.55954    0.11322  22.607
## 7|8  5.40615    0.27251  19.838
```

Nous ne poussons pas plus avant l'étude de ces modèles, ce n'est pas l'objet ici. Notons néanmoins l'intérêt de faire des régressions en utilisant les composantes de l'ACP, ces composantes étant indépendantes entre elles.

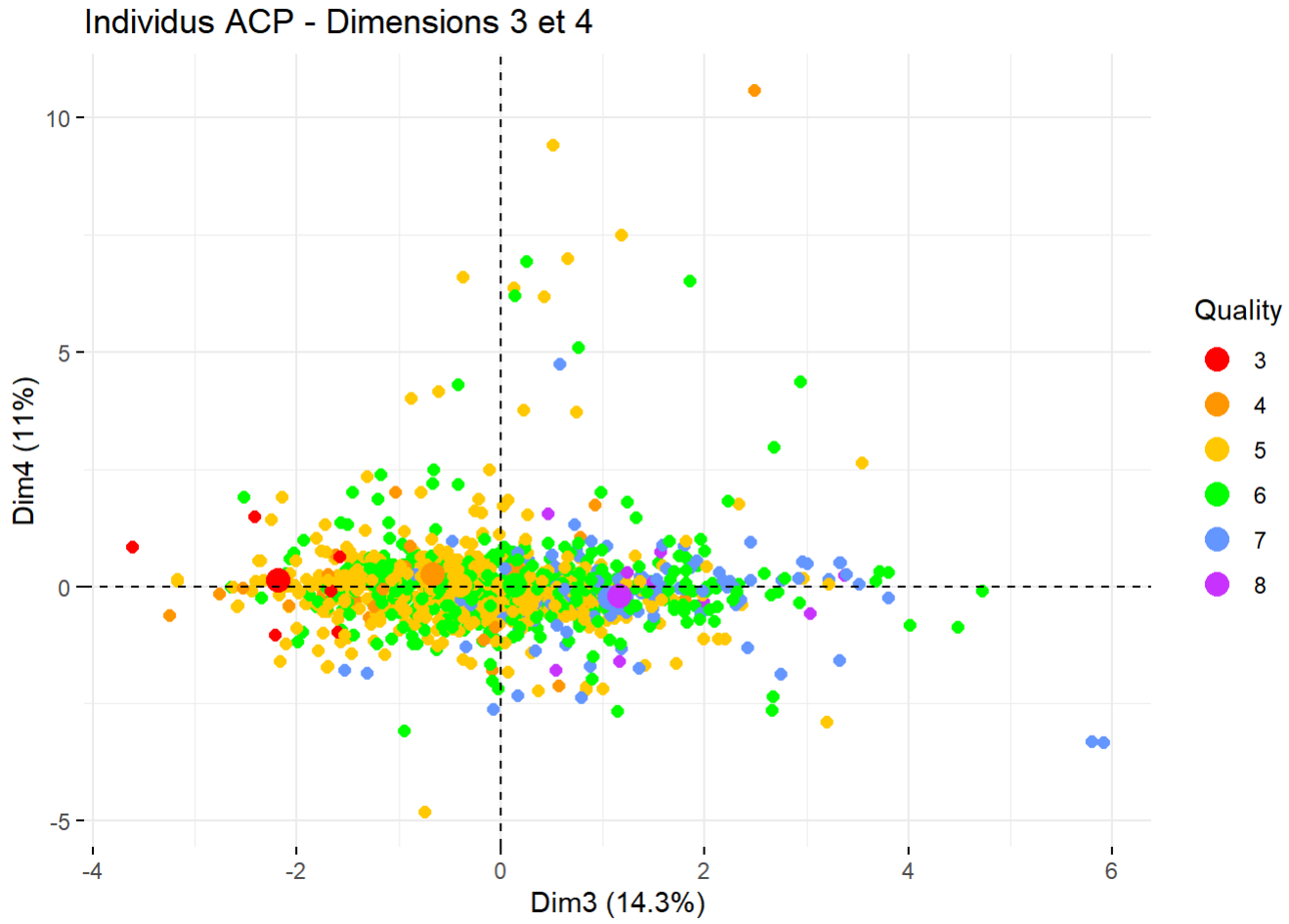
Cercle des corrélations avec variable d'intêret

```
quality_f <- as.factor(vin$quality)  # dataframe de la colonne quality en facteur

acp_vin_12 <-  fviz_pca_ind(acp_vin,
             axes=c(1,2),
             repel = TRUE,
             geom.ind = "point",
             col.ind = quality_f,
             palette = "ucscgb",
             pointshape = 16,
             pointsize = 2,
             legend.title = "Quality",
             title = "Individus ACP - Dimensions 1 et 2")
acp_vin_12
```



Gardons à l'esprit que la grande majorité des vins sont notés 5 ou 6.
Première remarque: la dimension 1 explique assez mal la qualité des vins.
Par contre, la dimension 2 explique mieux notre variable cible. Le "pattern" qui apparaît est que les notes tendent à augmenter lorsque l'on se déplace vers les valeurs négatives de la dimension 2.
Les vins les mieux notés sont donc ceux qui ont globalement le plus haut taux d'alcool et le moins de dioxyde de souffre.

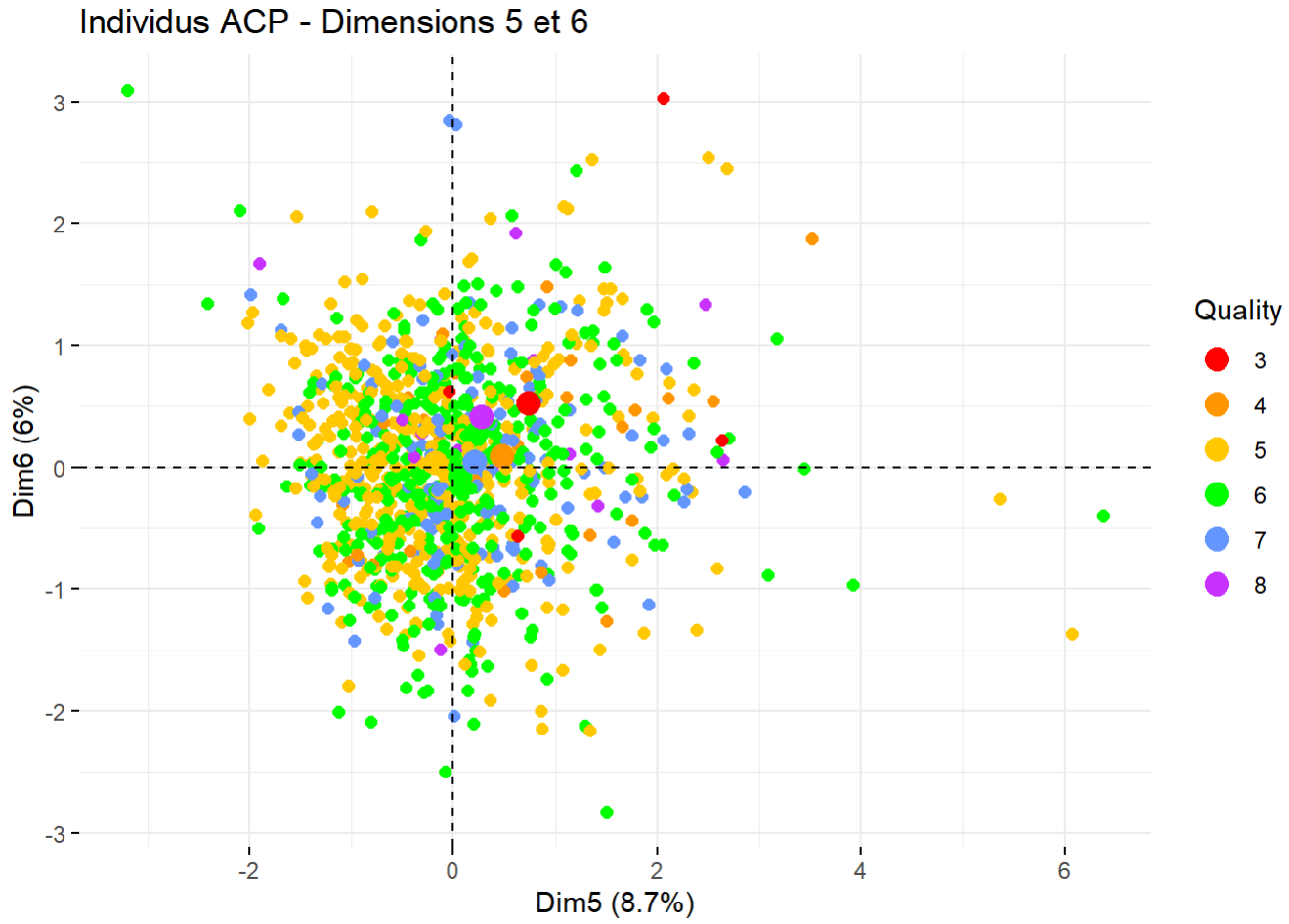
```
acp_vin_34 <- fviz_pca_ind(acp_vin,
             axes=c(3,4),
             repel = TRUE,
             geom.ind = "point",
             col.ind = quality_f,
             palette = "ucscgb",
             pointshape = 16,
             pointsize = 2,
             legend.title = "Quality",
             title = "Individus ACP - Dimensions 3 et 4")
acp_vin_34
```



On observe que les vins de meilleure qualité ont tendance à être plus vers la droite. La dimension 3 fait donc bien apparaître une relation avec la note donnée au vin. Notons les deux “outliers” à droite, qui sont des vins notés 7.
Par contre, une interprétation de ce phénomène n’est pas aisée (du moins sans connaissance solide en chimie), la composante 3 n’ayant pas de variables véritablement dominantes dans sa composition.

La dimension 4 ne fait pas apparaître de relation claire avec la qualité des vins.

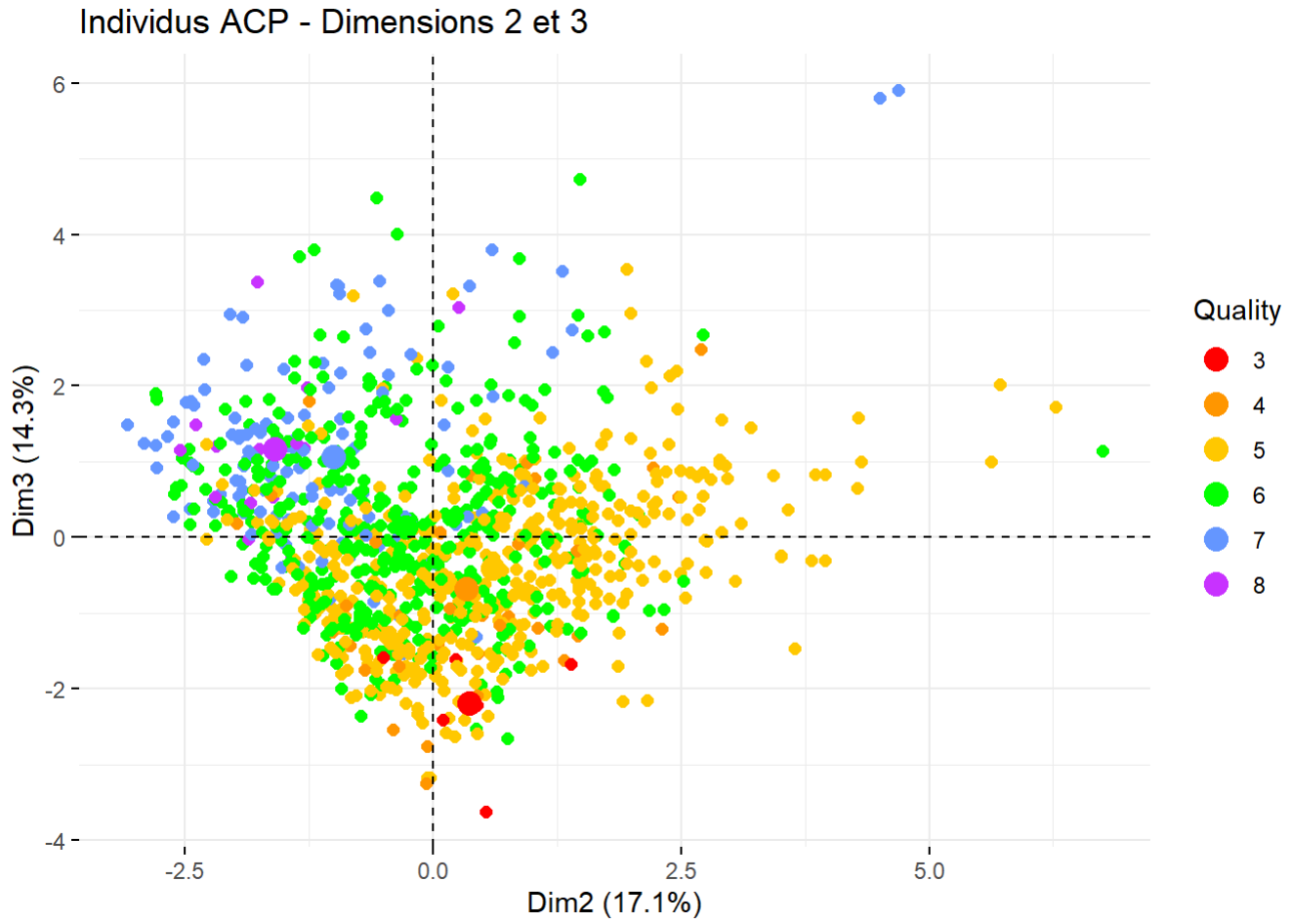
```
acp_vin_56 <- fviz_pca_ind(acp_vin,
             axes=c(5,6),
             repel = TRUE,
             geom.ind = "point",
             col.ind = quality_f,
             palette = "ucscgb",
             pointshape = 16,
             pointsize = 2,
             legend.title = "Quality",
             title = "Individus ACP - Dimensions 5 et 6")
acp_vin_56
```



La composante 5 est peu explicative, même si on remarque que les vins mieux notés ont tendance à s'étaler plus vers la droite.
La dimension 6 ne permet pas d'expliquer la note.

Les dimensions les plus explicatives sont donc la 2 et la 3, nous pouvons les combiner au sein d'un même graphique:

```
acp_vin_23 <- fviz_pca_ind(acp_vin,
             axes=c(2,3),
             repel = TRUE,
             geom.ind = "point",
             col.ind = quality_f,
             palette = "ucscgb",
             pointshape = 16,
             pointsize = 2,
             legend.title = "Quality",
             title = "Individus ACP - Dimensions 2 et 3")
acp_vin_23
```



Le résultat est le meilleur obtenu jusque-là. Une classification en fonction de la qualité apparaît ici nettement.

PLS

Afin de savoir si les variables chimiques expliquent bien la qualité du vin, la méthode Partial Least Square semble bien adaptée.
En plus de maximiser la quantité de variance au sein des variables indépendantes comme le fait l'ACP, elle maximise également la variance qui explique la variable cible (la qualité du vin, ici).

```
vin_12_s <- as.data.frame(vin_11_s)
vin_12_s$quality <- vin$quality

set.seed(100)
pls_model <- plsr(quality ~ ., data = vin_12_s, scale= FALSE)
summary(pls_model)
```

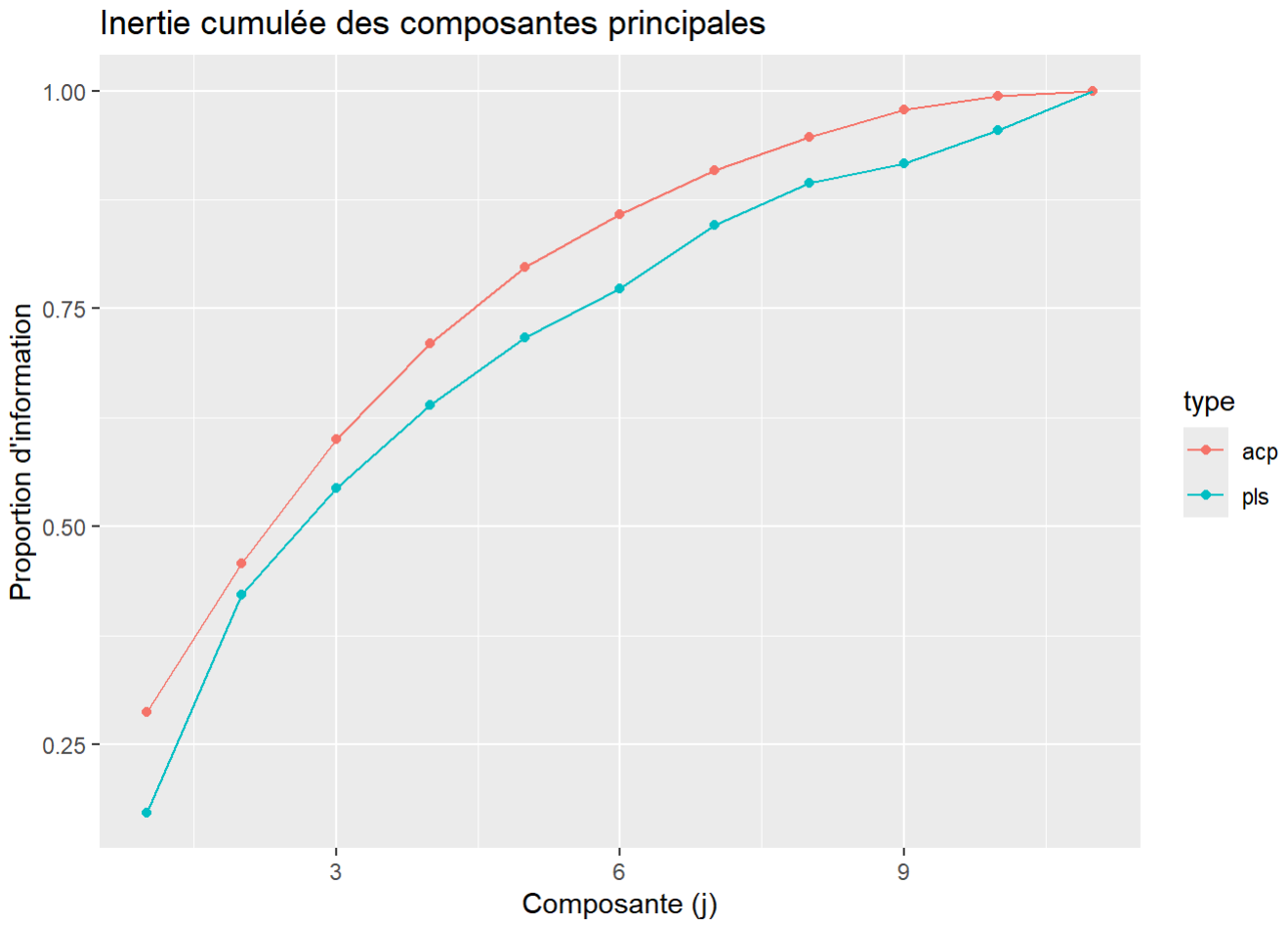
```
## Data:    X dimension: 1143 11
##  Y dimension: 1143 1
## Fit method: kernelpls
## Number of components considered: 11
## TRAINING: % variance explained
##          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X          17.18    42.12    54.35    63.94    71.71    77.28    84.57    89.44
## quality    34.95    36.37    37.01    37.29    37.37    37.40    37.41    37.42
##          9 comps  10 comps  11 comps
## X          91.68     95.45    100.00
## quality    37.42     37.42     37.42
```

La première composante explique 17.18% de la variance de X (les variables de nature chimique) et 37.95% de la variance de la variable cible.
Nous observons que la première composante explique environ 10% de moins la variance de X que ne le fait la première dimension de l'ACP (respectivement 17.18% contre 28.7%), ce qui peut s'expliquer par le fait que les composantes en PLS représentent un compromis entre la variance des variables explicatives et celle de la qualité, par construction.

Nous pouvons observer graphiquement cette différence entre la variance expliquée par les composantes ACP et les composantes PLS:

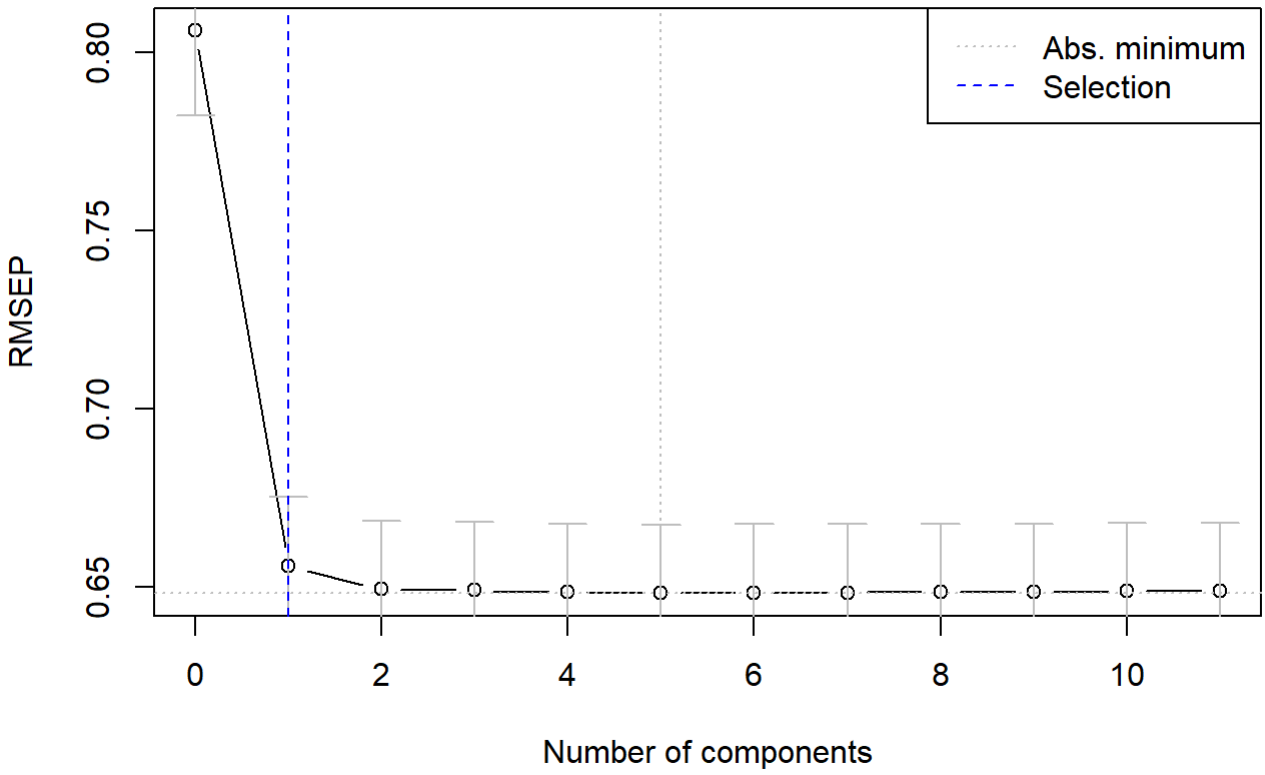
```
val_propres %>% mutate(pls = cumsum(pls_model$Xvar/pls_model$Xtotvar),
                       acp = cumsum(eigenvalue / p)) %>%
  dplyr::select(comp, pls, acp) %>%
  gather(key ="type", value ="y",-comp) %>%
ggplot() +
  aes(x = comp, y =y, color =type) +
  geom_point() +
  geom_line() +
  labs(x = "Composante (j)", y = "Proportion d'information",
       title = "Inertie cumulée des composantes principales")
```

```
## Warning: attributes are not identical across measure variables; they will be
## dropped
```



Le graphique confirme bien la moindre performance des composantes PLS pour expliquer uniquement les variables indépendantes (sauf évidemment quand toutes les composantes sont présentes).
La différence est particulièrement importante pour la première composante, celle qui justement explique le mieux la variable cible.

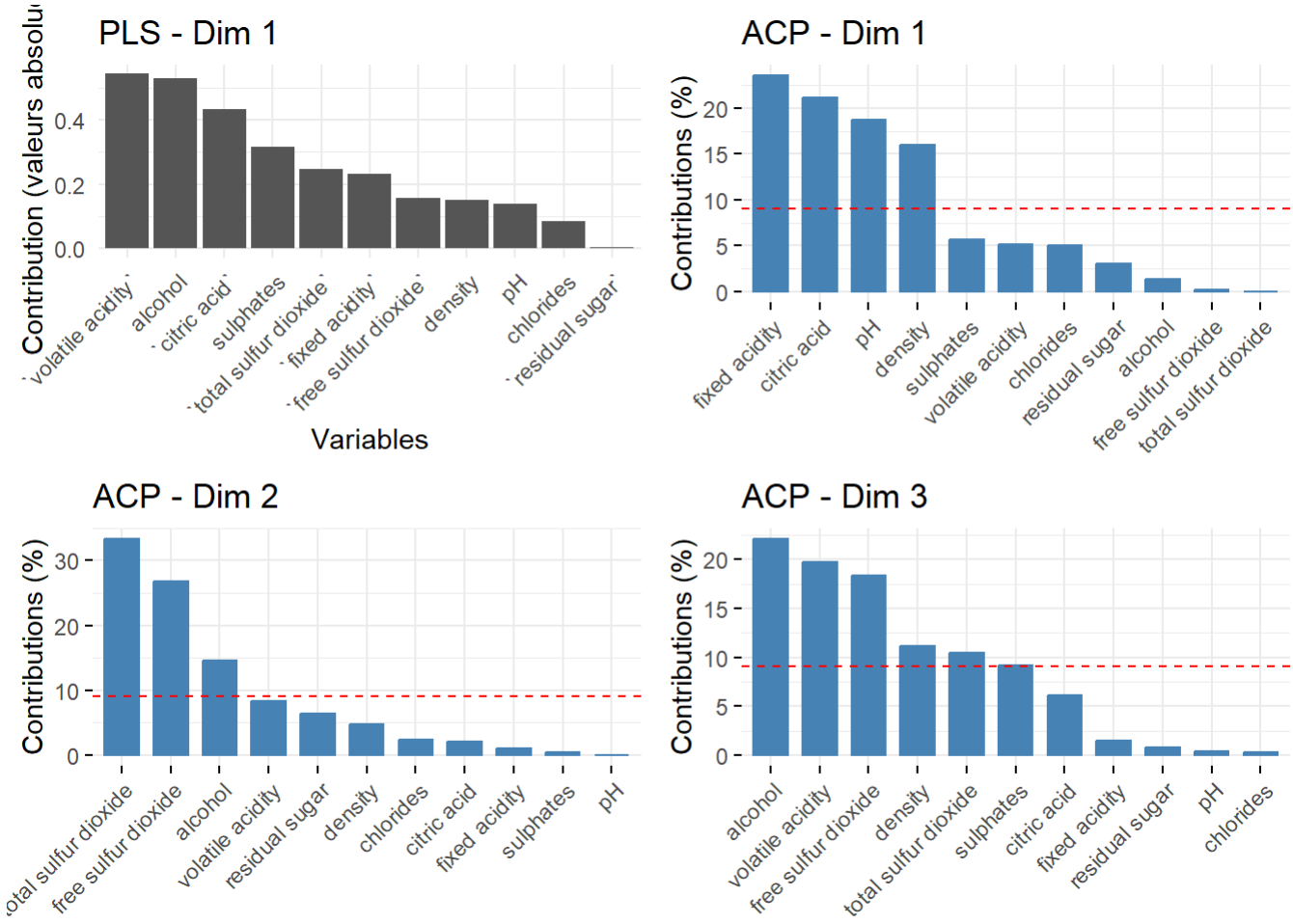
```
pls_model_cv <- plsr(quality ~ ., data = vin_12_s, scale= FALSE, validation ="CV")
ncomp.onesigma <- selectNcomp(pls_model_cv, method = "onesigma", plot = TRUE)
```



Par validation croisée et en utilisant le critère d'une *standard error* de distance maximale par rapport à l'erreur minimale (l'erreur minimale est atteinte en comp.5), c'est, sans surprise, uniquement la première variable qui est sélectionnée.

Comparons les variables de la première composante PLS et les trois premières composantes ACP:

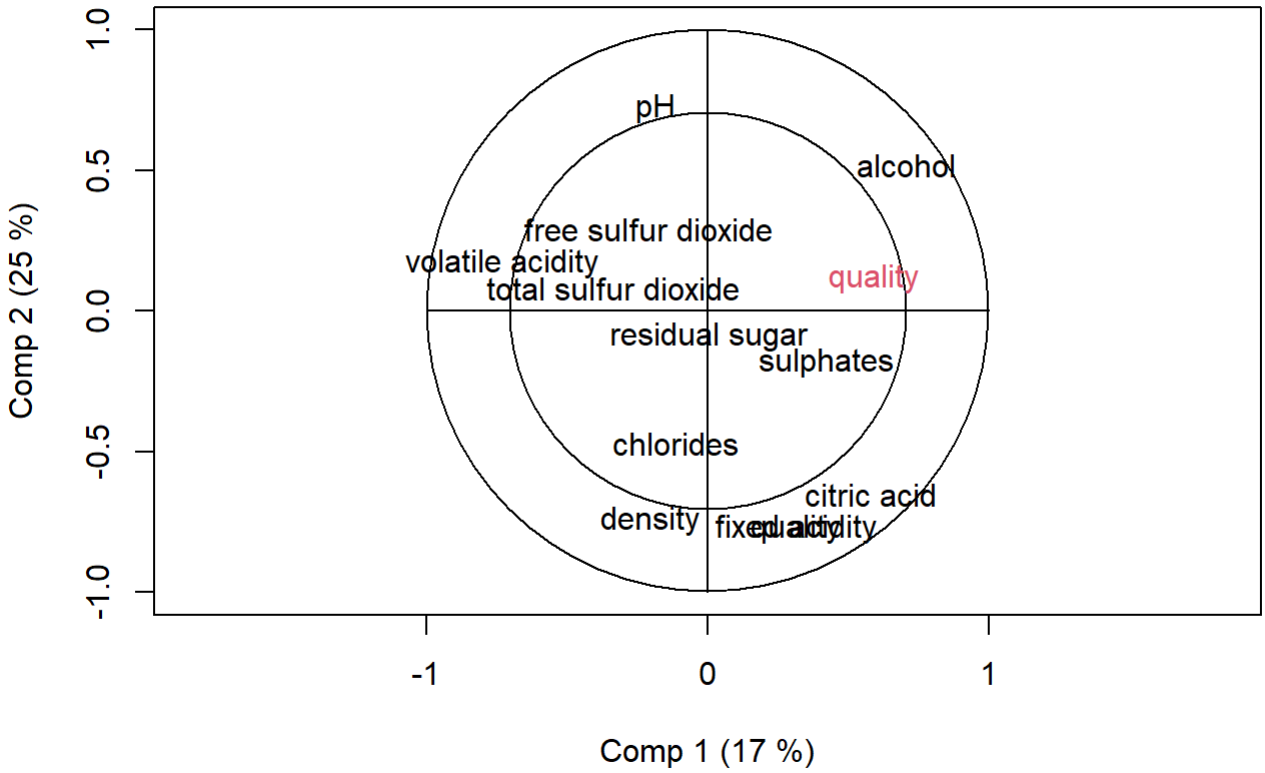
```
loadings_pls <- loadings(pls_model) # Extraire les contributions des variables
loadings_dim1 <- abs(loadings_pls[, 1])  # Prendre les valeurs absolues des contributions
loadings_df <- data.frame(Variables = rownames(loadings_pls), Contribution = loadings_dim1)

dim_pls <- ggplot(loadings_df, aes(x = reorder(Variables, -Contribution), y = Contribution))
+
  geom_bar(stat = "identity") +
  labs(title = "PLS - Dim 1",
       x = "Variables", y = "Contribution (valeurs absolues)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

plot1 <- fviz_contrib(acp_vin, choice = "var", axes = 1) + ggtitle("ACP - Dim 1")
plot2 <- fviz_contrib(acp_vin, choice = "var", axes = 2) + ggtitle("ACP - Dim 2")
plot3 <- fviz_contrib(acp_vin, choice = "var", axes = 3) + ggtitle("ACP - Dim 3")

#plot1 + plot2 + plot3
grid.arrange(dim_pls, plot1,plot2, plot3, ncol = 2, nrow = 2)
```



Il est intéressant de noter que la dimension 1 en PLS ressemble à une combinaison des dimensions 2 et 3 de l'ACP, celles qui expliquaient le mieux la qualité des vins.
Nous voyons ici tout l'intérêt de l'approche PLS qui "capte" rapidement les variables les plus explicatives de la variable cible.

Pour une compréhension plus fine, nous pouvons afficher le cercle des corrélations des deux premières dimensions:

```
corrplot(pls_model, label = c(colnames(vin),"quality"), ploty=TRUE, comps=c(1,2))
```

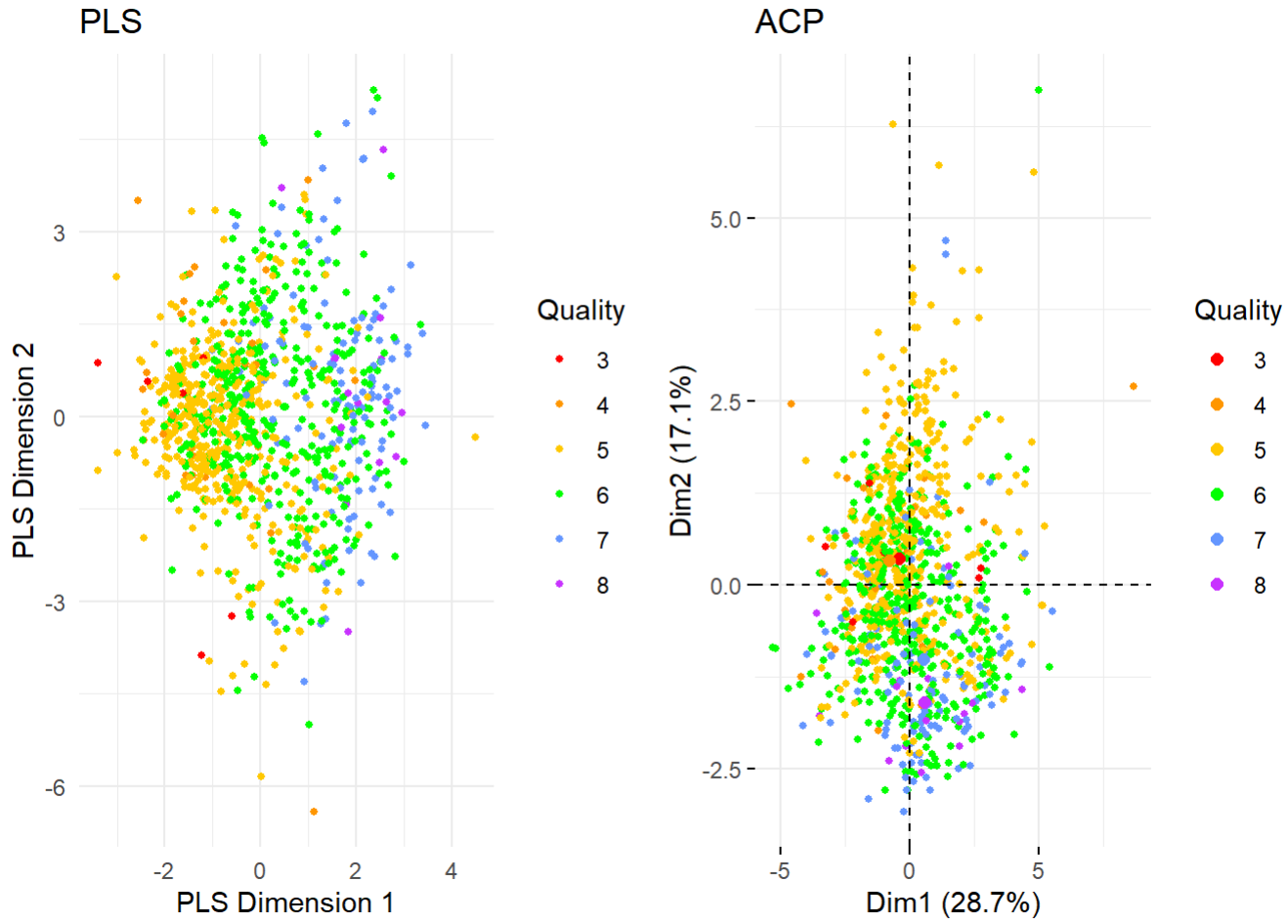


Remarque: la variance expliquée de la composante 2 est indiquée supérieure à celle de la composante 1 car l'ordre de numérotion des variables dépend de l'explication de la variable cible et non de l'explication de la variance totale.

Ce graphique montre une certaine ressemblance entre la composante 1 de la PLS et la composante 2 de l'ACP.

Comparons la répartition des individus selon les deux premières composantes de la PLS et de l'ACP:

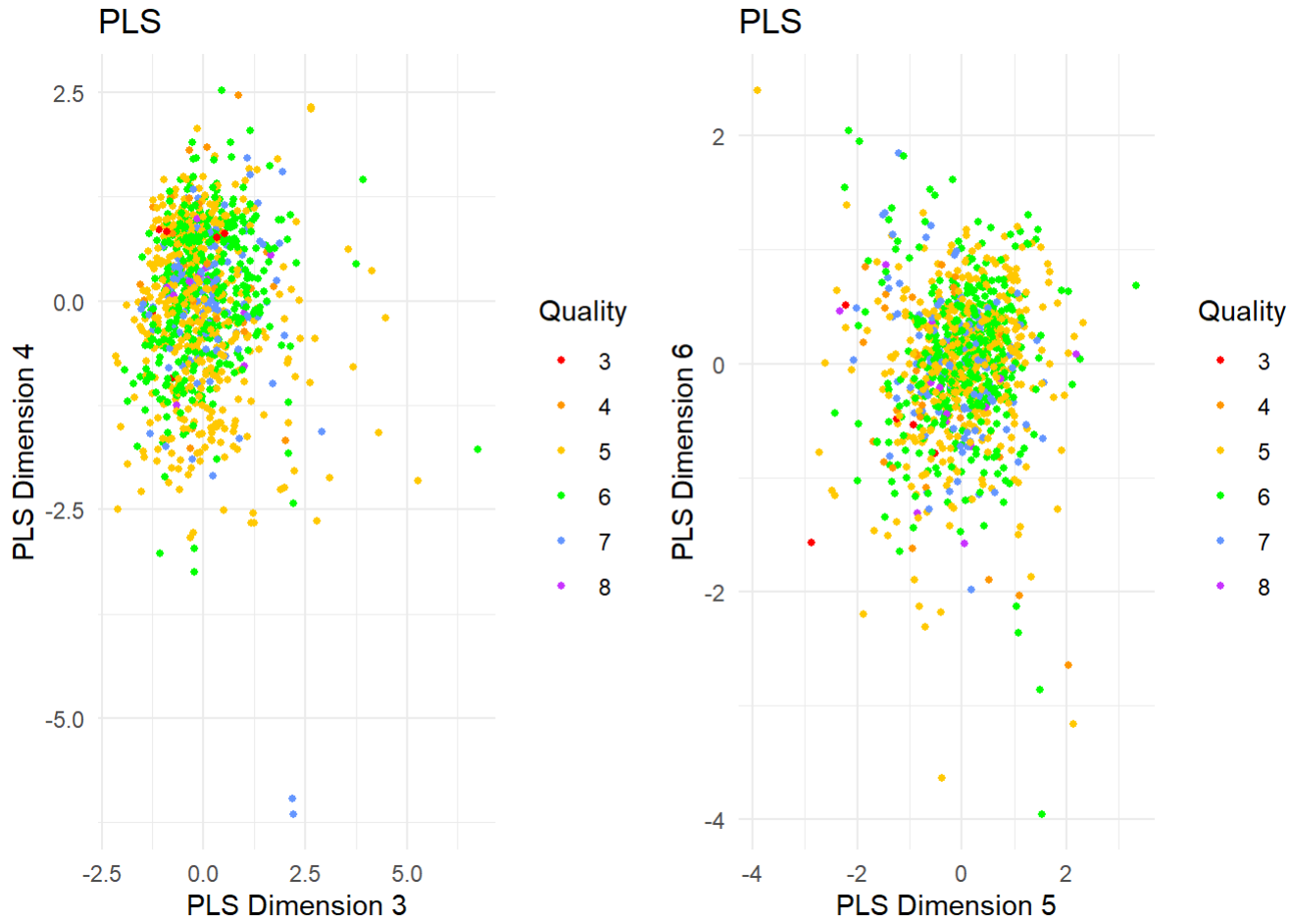
```
# Extraire les coordonées des deux premières composantes
scores <- pls_model$scores[, 1:2]  # Prendre les deux premières composantes
df_scores <- data.frame(scores, Quality = factor(vin$quality))

# Créer le graphique en utilisant la palette "ucscgb"
pls_vin_12 <- ggplot(df_scores, aes(x = Comp.1, y = Comp.2, color = Quality)) +
  geom_point(size = 1) +
  labs(title = "PLS",
       x = "PLS Dimension 1",
       y = "PLS Dimension 2") +
  theme_minimal() +
  scale_color_ucscgb()

grid.arrange(pls_vin_12, acp_vin_12, ncol = 2, nrow = 1)
```



Cette autre représentation arrive aux mêmes conclusions: la première dimension parvient assez bien à expliquer la qualité des vins, mieux encore que la dimension 2 de l'ACP.

Affichons les dimensions suivantes:

```
scores <- pls_model$scores[, 3:4]  # extraire les  composantes 3 et 4
df_scores <- data.frame(scores, Quality = factor(vin$quality))

pls_vin_34 <- ggplot(df_scores, aes(x = Comp.3, y = Comp.4, color = Quality)) +
  geom_point(size = 1) +
  labs(title = "PLS",
       x = "PLS Dimension 3",
       y = "PLS Dimension 4") +
  theme_minimal() +
  scale_color_ucscgb()

scores <- pls_model$scores[, 5:6]  # extraire les  composantes 3 et 4
df_scores <- data.frame(scores, Quality = factor(vin$quality))

pls_vin_56 <- ggplot(df_scores, aes(x = Comp.5, y = Comp.6, color = Quality)) +
  geom_point(size = 1) +
  labs(title = "PLS",
       x = "PLS Dimension 5",
       y = "PLS Dimension 6") +
  theme_minimal() +
  scale_color_ucscgb()

grid.arrange(pls_vin_34, pls_vin_56, ncol = 2, nrow = 1)
```



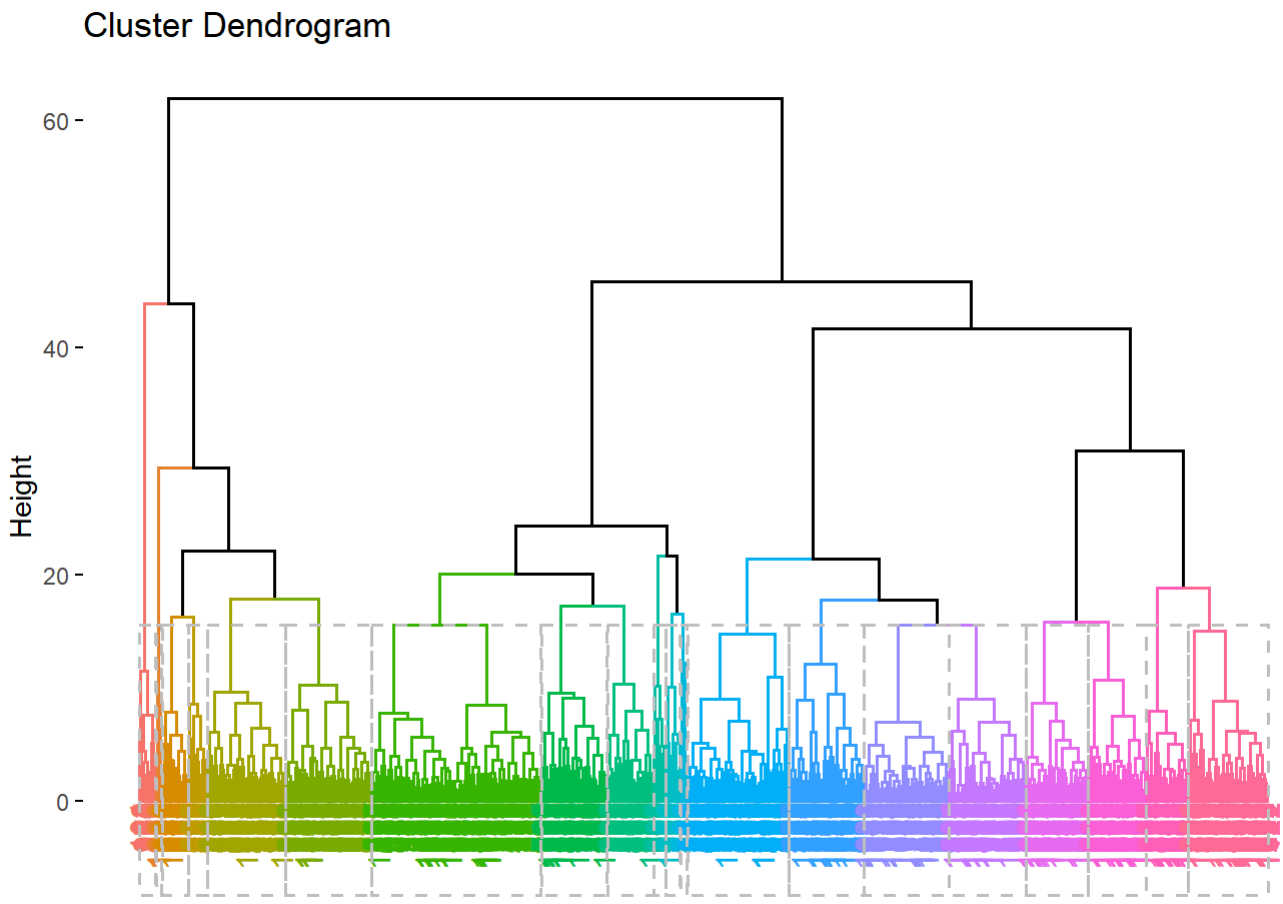
Nous passons rapidement sur les autres dimensions représentées ici, elles ne parviennent à expliquer correctement la qualité (à relier à la faible variance expliquée de la variable cible par ces composantes).

Heatmap avec 20 groupes de vins

```
hc <- hclust(dist(vin_11_s), method ="ward.D2")
dend<- fviz_dend(hc, rect = TRUE, k=20)
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## ℹ The deprecated feature was likely used in the factoextra package.
##   Please report the issue at <https://github.com/kassambara/factoextra/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

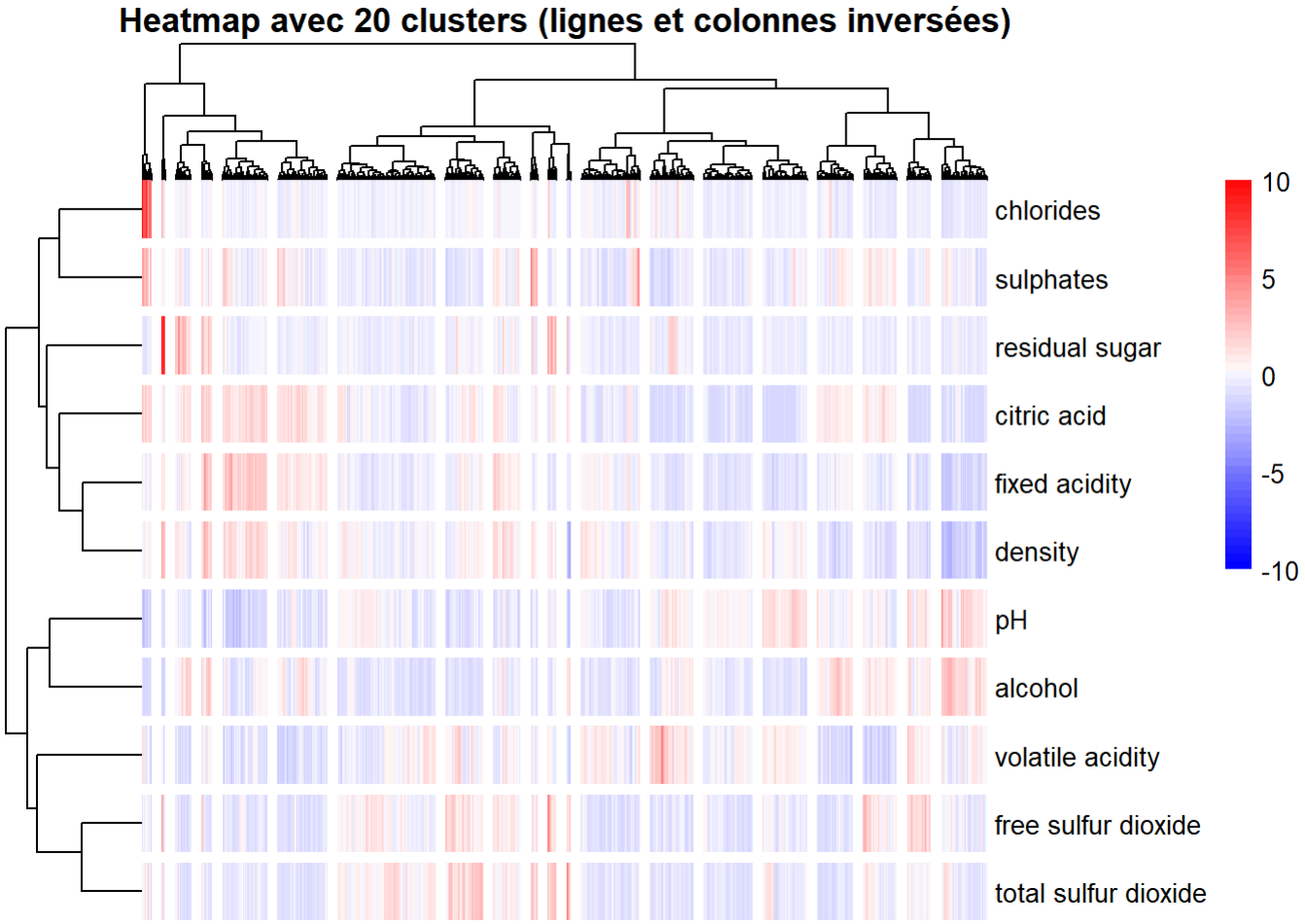
```
dend
```



La méthode de *ward* minimise la variance totale au sein de chaque cluster.
Notons que les 20 clusters sont assez hétérogènes en nombre d'individus.
Le cluster le plus à gauche se distingue: les deux *fusions* qui le relient au cluster qui contient tous les individus sont situées très haut, ce qui indique une grande distance par rapport aux autres clusters.

Utilisons cet arbre dans un *heatmap*:

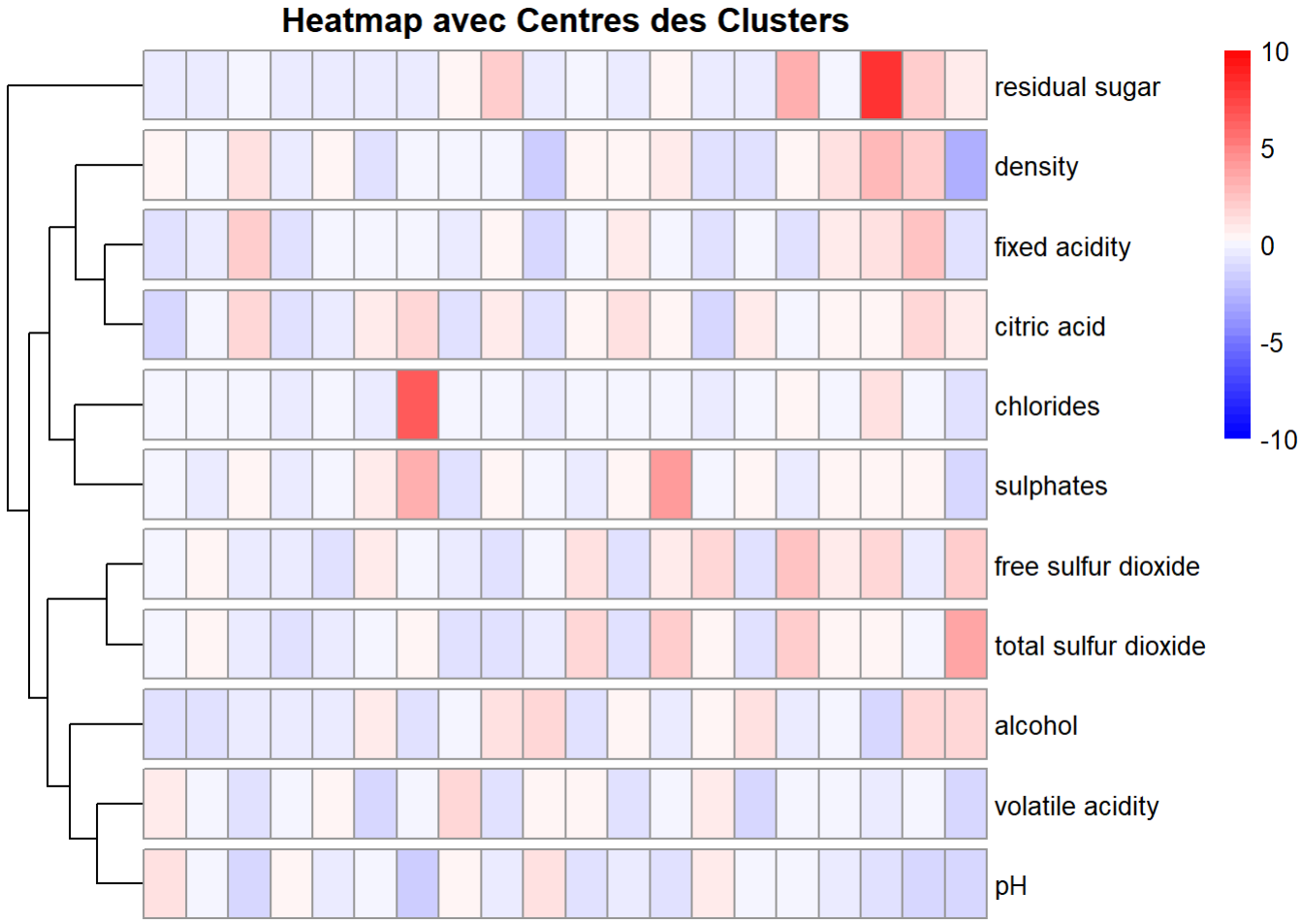
```
hc <- hclust(dist(vin_11_s), method ="ward.D2")
vin_11_s_transposed <- t(vin_11_s)

hm_all <- pheatmap(vin_11_s_transposed,
         cluster_cols = hc,
         show_rownames = TRUE,
         cutree_cols = 20,
         cutree_rows = 11,
         breaks = seq(-10, 10, length.out = 50),  # Limiter l'échelle des couleurs
         color = colorRampPalette(c("blue", "white", "red"))(50),  # Choisir une palette de c
ouleurs
         main = "Heatmap avec 20 clusters (lignes et colonnes inversées)")
```



Nous pouvons observer que la CAH parvient bien à créer des familles en fonction des variables, i.e.: à une forte proximité dans l'arbre, l'intensité par rapport à une variable donnée est le plus souvent aussi assez proche. À l'opposé, pour les clusters isolés, donc reliés au sommet par peu de fusions (comme le premier à gauche, ici), le profil se différencie plus nettement des autres clusters.

Cette représentation affiche **tous les individus** au sein de chaque cluster, elle permet notamment de vérifier que le cluster est homogène (en son sein) par rapport à une variable donnée.
Pour une lecture plus claire mais moins précise, nous pouvons n'afficher que le centre des clusters.

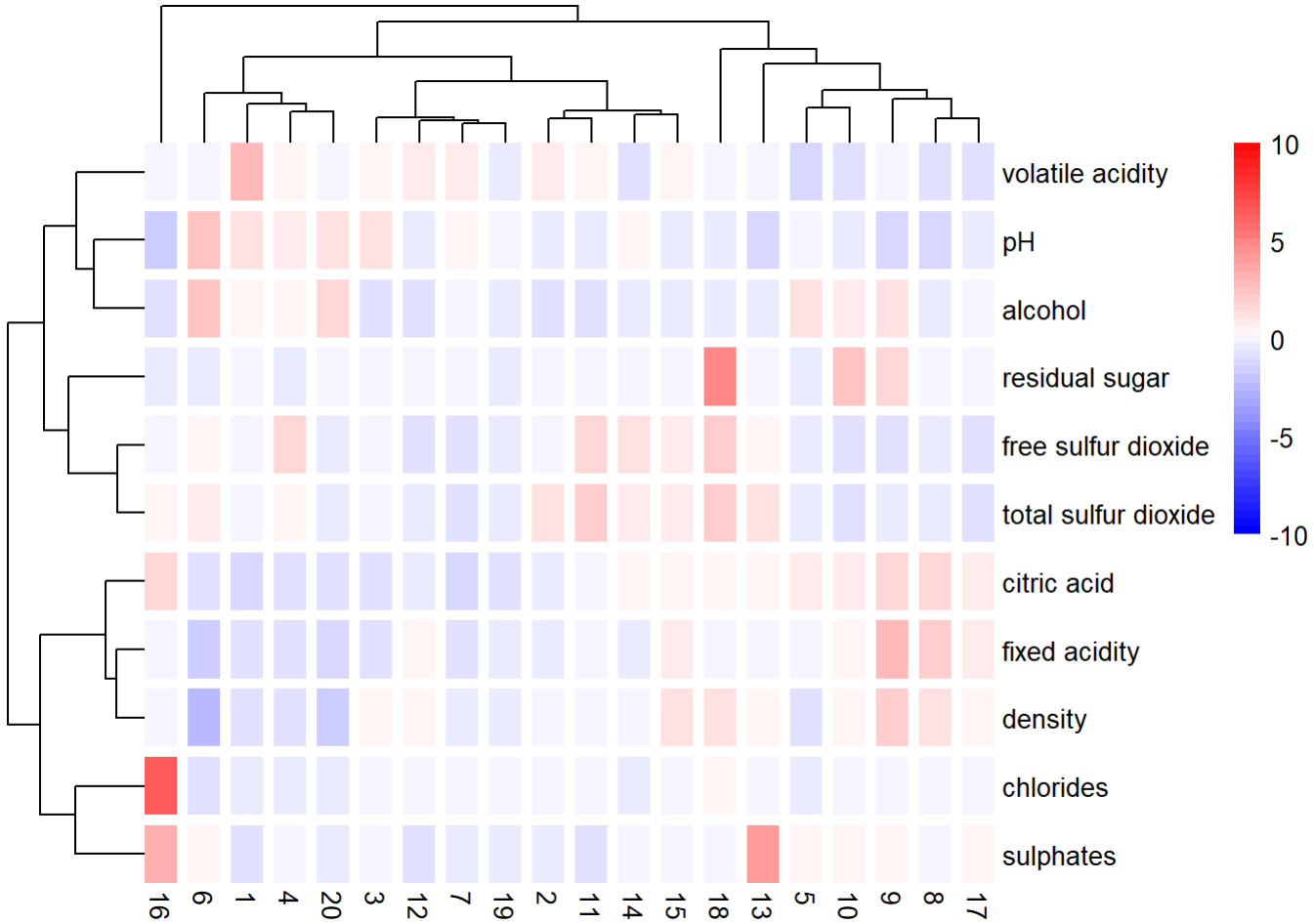
```
clusters <- cutree(hc, k = 20)
# Calcule les centres des clusters
centres_clusters <- aggregate(vin_11_s, by = list(cluster = clusters), FUN = mean)
centres_clusters <- centres_clusters[,-1]  # Supprimer la colonne de cluster
centres_clusters_transposed <- t(centres_clusters)

# heatmap avec les centres des clusters
pheatmap(centres_clusters_transposed,
         cutree_cols = 20,
         cluster_cols = FALSE,
         cutree_rows = 11,
         show_rownames = TRUE,
         breaks = seq(-10, 10, length.out = 50),
         color = colorRampPalette(c("blue", "white", "red"))(50),
         main = "Heatmap avec Centres des Clusters")
```



Il est ici plus facile de repérer les clusters qui sont “forts” dans certaines variables au prix d’une perte d’information (dimension des clusters et comportement des individus en leur sein).

méthode des k-means

```
set.seed(400)
km <- kmeans(vin_11_s, centers = 20)

cols <- colorRampPalette(c( "#0073c2","white","#efc000"))(100)
pheatmap(t(km$centers), border_color = NA,
         color =colorRampPalette(c("blue", "white", "red"))(50),
         clustering_method = "ward.D2",
         breaks = seq(-10, 10, length.out = 50),
         cutree_rows = 11,
         cutree_cols =20)
```



Cette approche permet d'associer les deux méthodes: d'abord les clusters sont calculés par la méthode des k-means, puis les centres des clusters sont hiérarchisés par CAH (méthode souvent utilisée quand il y a beaucoup d'individus).
L'avantage des k-means est que nous pouvons changer les points de départ en fixant des graines différentes et ne conserver que les résultats les plus pertinents.

Concernant notre jeu de données, les deux méthodes donnent des résultats assez proches.
Dans le cadre d'une étude chimique poussée, réalisée avec un expert, nous pourrions analyser quels individus composent tel cluster fort en telle variable, etc.
Par exemple: les vins composant le cluster "fort" en *residual sugar* sont-ils issus des mêmes terroirs ? Ou bien proviennent-ils des mêmes cépages ?

Comparaison quantitative des deux méthodes:

```
########## CAH

clusters_hc <- cutree(hc, k = 20)
# inertie intra-classe pour le clustering hiérarchique
inertie_intra_hc <- sum(sapply(unique(clusters_hc), function(cluster) {
  sum(dist(vin_11_s[clusters_hc == cluster, ])^2)
}))

# Calcul de l'inertie totale
inertie_totale <- sum(dist(vin_11_s)^2)

# Ratio pour le modèle hiérarchique
ratio_hc <- (inertie_totale - inertie_intra_hc) / inertie_totale
cat("Ratio Inertie inter / Inertie totale pour le clustering hiérarchique :", round(ratio_hc,
3), "\n")
```

```
## Ratio Inertie inter / Inertie totale pour le clustering hiérarchique : 0.976
```

```
########## k-means

# Calcul de l'inertie inter et intra pour le k-means
inertie_inter_km <- km$betweenss
inertie_intra_km <- km$tot.withinss

# Ratio pour le modèle k-means
ratio_km <- inertie_inter_km / (inertie_inter_km + inertie_intra_km)
cat("Ratio Inertie inter / Inertie totale pour le clustering k-means :", round(ratio_km, 3),
"\n")
```

```
## Ratio Inertie inter / Inertie totale pour le clustering k-means : 0.661
```

Pour la CAH, un ratio de 0.976 indique que la majorité de la variance totale des données est expliquée par les différences entre les clusters. Cela signifie que les clusters sont très distincts les uns des autres, ce qui est un bon point. En d'autres termes, presque toutes les données sont bien séparées en groupes homogènes et cohérents.

Pour les k-means, le ratio plus faible indique que seulement 66,1 % de la variance totale est expliquée par les différences entre les clusters. Cela signifie que les clusters formés par k-means sont moins distincts comparés à ceux formés par le clustering hiérarchique. Il y a donc plus de variabilité ici **au sein** des clusters.
Certains individus peuvent être plus proches d'autres clusters que de leur propre cluster (limite de la méthode k-means).
La méthode des k-means est réputée plus grossière que la CAH, elle est le plus souvent utilisée en pré-traitement lorsque l'on a affaire à un grand nombre d'individus, la méthode CAH étant très gourmande en ressources (le temps de traitement observé ici avec 10000 individus le confirme).

Conclusion

Pour aller plus loin, nous aurions pu utiliser un modèle de PLS ordinale comme le permet le package **plsRglm**, plutôt que le package *pls* qui a traité la qualité du vin comme une variable continue.

Dans le cadre d'une analyse chimique poussée et maîtrisée, une ACP parcimonieuse aurait permis d'être plus sélectif et de mieux orienter l'analyse.
À l'inverse de l'ACP classique où toutes les composantes principales sont des combinaisons linéaires de toutes les variables, l'ACP parcimonieuse permet de sélectionner un sous-ensemble de variables pour chaque composante principale, ce qui permet de simplifier l'interprétation des résultats.