

DM2: Analyse Discriminante PLS exploratoire

Kateryna Stetsun

2025-11-21

Introduction

L'Analyse Discriminante PLS exploratoire (ADPLS) est une méthode multivariée qui combine les principes de la régression PLS (Partial Least Squares) et de l'analyse discriminante.

Son objectif est de construire des composantes linéaires des variables explicatives X maximisant la capacité de discrimination des groupes définis par la variable qualitative Y .

Autrement dit, on cherche à extraire des axes factoriels $f = XMu$ expliquant à la fois une grande part de la variance de X (inertie totale) et une forte part de la variance expliquée par les classes de Y .

Les notations utilisées sont celles du cours :

- $X \in \mathbb{R}^{n \times p}$: matrice des variables quantitatives centrées ;
- $Y \in \mathbb{R}^{n \times q}$: matrice d'indicatrices des q classes (non centrée) ;
- $W = \text{diag}(w_i)$: matrice diagonale des poids individuels ;
- M : métrique définie positive sur l'espace des variables ;
- $f = XMu$: composante discriminante associée à la direction u ;
- $\Pi_Y = Y(Y'WY)^{-1}Y'W$: projecteur sur l'espace engendré par Y ;
- $\hat{X} = \Pi_Y X$: projection de X sur cet espace.

Partie 2 — Programmation

Dans cette seconde partie, nous détaillons l'implémentation algorithmique de l'Analyse Discriminante PLS (ADPLS).

L'objectif est de reconstruire pas à pas les éléments constitutifs de la méthode : extraction des composantes discriminantes, calcul des indicateurs associés et préparation des représentations graphiques permettant l'interprétation multivariée des données.

Nous adoptons une approche modulaire afin de distinguer clairement les opérations matricielles générales (produits pondérés, normalisations, métriques) des étapes spécifiques au calcul des composantes ADPLS.

Définition des variables d'entrée et fonctions utilitaires

Les données d'entrée nécessaires à l'algorithme sont les suivantes :

- la matrice X de dimension $n \times p$, contenant les variables explicatives numériques (non nécessairement centrées) ;
- la matrice Y de dimension $n \times q$, formée des indicatrices (0/1) correspondant aux q modalités de la variable de groupe ;
- le paramètre H , représentant le nombre de composantes discriminantes à extraire ;
- le vecteur de poids w , de longueur n , qui peut être laissé nul pour signifier des poids égaux ;
- la matrice métrique M de dimension $p \times p$, éventuellement identitaire.

Une option `center_scale` permet en outre de centrer et réduire les variables de X si nécessaire.

```
# Définition des entrées
X <- as.matrix(X)          # matrice n x p
Y <- as.matrix(Y)          # matrice n x q
H <- 3                      # nombre de composantes
w <- rep(1 / nrow(X), nrow(X))  # poids égaux
M <- diag(ncol(X))        # matrice métrique identité
center_scale <- TRUE       # centrage-réduction activée
```

Afin d'assurer une formulation claire et homogène des opérations, nous définissons quelques fonctions utilitaires.

Elles permettent notamment de gérer les produits scalaires pondérés, les normes induites par un vecteur de poids, ainsi que les transformations associées à la métrique M .

```
# matrice de pondération diagonale à partir d'un vecteur w
Wmat <- function(w) {
  if (is.null(w)) return(NULL)
  diag(as.numeric(w), nrow = length(w))
}

# norme pondérée avec w vecteur poids ou W matrice diag
w_norm2 <- function(v, w) {
  if (is.null(w)) return(as.numeric(t(v) %*% v))
  as.numeric(t(v) %*% (w * v))
}

# produit scalaire pondéré
w_prod <- function(a, b, w) {
  if (is.null(w)) return(as.numeric(t(a) %*% b))
  as.numeric(t(a) %*% (w * b))
}

# racine symétrique de M (M supposée SPD ou semi-définie)
M_half <- function(M) {
  ev <- eigen(M, symmetric = TRUE)
  vals <- pmax(ev$values, 0)
  vecs <- ev$vectors
  vecs %*% diag(sqrt(vals), length(vals)) %*% t(vecs)
}

# inverse de la racine symétrique de M (pseudo-inverse si valeurs nulles)
M_half_inv <- function(M) {
  ev <- eigen(M, symmetric = TRUE)
  vals <- ev$values
```

```

vecs <- ev$vectors
invvals <- ifelse(vals > 0, 1 / sqrt(vals), 0)
vecs %*% diag(invvals, length(invvals)) %*% t(vecs)
}

```

1. Calcul des H premières composantes ADPLS

La première étape de l'Analyse Discriminante PLS (ADPLS) consiste à extraire successivement les H composantes discriminantes maximisant la covariance pondérée entre les variables explicatives et la structure de groupes encodée dans Y .

Pour chaque rang h , la composante discriminante est définie par $f_h = XMu_h$, où u_h désigne le vecteur propre associé à la plus grande valeur propre de la matrice $E_h = \hat{X}'W\hat{X}$ avec $\hat{X} = \Pi_Y X$ la projection de X sur l'espace engendré par les indicatrices de groupes.

Pour $h > 1$, le vecteur u_h est en outre calculé sous la contrainte d'orthogonalité aux composantes précédentes, en imposant $u_h^\top Mu_{h'} = 0$ pour tout $h' < h$.

L'algorithme ci-dessous réalise ces étapes successives :

- projection de X sur l'espace de Y ,
- diagonalisation de la matrice E_h ,
- extraction du vecteur propre principal,
- normalisation selon la métrique M ,
- construction des scores f_h puis de leurs versions normalisées \tilde{f}_h .

Le code renvoie également les vecteurs u_h , les composantes discriminantes $F = (f_h)$, les composantes normalisées \tilde{F} , ainsi que les quantités intermédiaires \hat{X} , E et le projecteur Π_Y .

```

adpls_part1 <- function(X, Y, H = 2, w = NULL, M = NULL, center_scale = TRUE) {

#préparation
# conversion explicite en matrices et extraction des dimensions.
X <- as.matrix(X)
Y <- as.matrix(Y)
n <- nrow(X)
p <- ncol(X)

# vérification : X et Y doivent décrire les mêmes individus
if (nrow(Y) != n) stop("X et Y doivent avoir le même nombre de lignes (individus)!")

# construction du vecteur de poids w
if (is.null(w)) w <- rep(1 / n, n)
w <- as.numeric(w)
if (length(w) != n) stop("longueur de w pas la même que de n (nrow(X))!")
W <- diag(w)

# centrage-réduction éventuel de X
if (center_scale) X <- scale(X, center = TRUE, scale = TRUE)

# matrice métrique M (M=I par défaut)
if (is.null(M)) M <- diag(1, p)
if (!all(dim(M) == c(p, p))) stop("M doit être p x p!")

}

```

```

# calcul de racines de M
sqrtM <- M_half(M)
inv_sqrtM <- M_half_inv(M)

# projecteur Pi_Y
YYw <- t(Y) %*% (W %*% Y)
inv_YYw <- solve(YYw)
PiY <- Y %*% inv_YYw %*% t(Y) %*% W

# projection de X sur l'espace de Y
Xhat <- PiY %*% X

# matrice E (pour la diagonalisation)
E <- t(Xhat) %*% (W %*% Xhat)

# réservation des objets de sortie
if (is.null(H) || !is.numeric(H) || H < 1) stop("H doit être un entier positif!")
H <- as.integer(H)
U_list <- vector("list", H)
F <- matrix(0, n, H)
Ftilde <- matrix(0, n, H)

# boucle sur h = 1..H
for (h in seq_len(H)) {
  if (h == 1) {

    # première composante
    Estar <- sqrtM %*% E %*% sqrtM
    ev <- eigen(Estar, symmetric = TRUE)
    # vecteur propre principal en métrique transformée
    ustar <- ev$vectors[, 1]
    # retour dans l'espace u
    u <- inv_sqrtM %*% ustar
    # normalisation selon la contrainte u'Mu = 1
    unorm <- as.numeric(t(u) %*% (M %*% u))
    if (unorm <= 0) stop("norme u'Mu non strictement positive!")
    u <- u / sqrt(unorm)

  } else {

    # composantes suivantes (h > 1)
    Fprev <- F[, 1:(h - 1), drop = FALSE]
    Dprime <- t(Fprev) %*% (W %*% X)
    D <- t(Dprime)
    # construction Pi_D_perp
    inner <- t(D) %*% (M %*% D)
    inv_inner <- solve(inner)
    PiDperp <- diag(1, p) - D %*% inv_inner %*% t(D) %*% M
    # matrice à diagonaliser dans l'espace réduit
    Estar_proj <- sqrtM %*% PiDperp %*% E %*% t(PiDperp) %*% sqrtM
    ev <- eigen(Estar_proj, symmetric = TRUE)
    # premier vecteur propre dans l'espace réduit.
    ustar <- ev$vectors[, 1]
  }
}

```

```

u <- inv_sqrtM %*% ustar
# normalisation selon la contrainte  $u'Mu = 1$ 
unorm <- as.numeric(t(u) %*% (M %*% u))
if (unorm <= 0) stop(paste("norme  $u'Mu$  non positive (h=", h, ")!", sep = ""))
u <- u / sqrt(unorm)
}

# calcul de score  $f_h$ 
f_h <- X %*% (M %*% u)
# normalisation pondérée
fnorm2 <- as.numeric(t(f_h) %*% (W %*% f_h))
if (fnorm2 <= 0) stop(paste("norme de  $f_h$  <= 0 (h=", h, ")!", sep = ""))
ftilde_h <- f_h / sqrt(fnorm2)

# stockage des résultats
U_list[[h]] <- as.numeric(u)
F[, h] <- as.numeric(f_h)
Ftilde[, h] <- as.numeric(ftilde_h)
}

# liste de vecteurs en matrice p x H
U_mat <- do.call(cbind, U_list)
colnames(U_mat) <- paste0("u", seq_len(ncol(U_mat)))
colnames(F) <- paste0("f", seq_len(ncol(F)))
colnames(Ftilde) <- paste0("f_tilde", seq_len(ncol(Ftilde)))

return(list(U = U_mat, F = F, Ftilde = Ftilde, Xhat = Xhat,
            E = E, PiY = PiY, M = M, w = w))
}

```

2. Calcul des indicateurs $S(f_h)$ et $R^2(f_h, Y)$

Dans cette étape, nous évaluons la qualité des composantes discriminantes obtenues à partir de l'ADPLS. Deux indicateurs complémentaires sont utilisés :

$$S(f_h) = \frac{\|f_h\|_W^2}{\text{tr}(X'WX)}, \quad R^2(f_h, Y) = \frac{\|\hat{f}_h\|_W^2}{\|f_h\|_W^2},$$

où $\hat{f}_h = \Pi_Y f_h = \Pi_Y X M u_h$ et $\Pi_Y = Y(Y'WY)^{-1}Y'W$.

L'indicateur $S(f_h)$ mesure la proportion d'inertie totale de X capturée par la composante f_h . Une valeur élevée signale que l'axe contribue significativement à la structuration globale du nuage d'individus.

L'indicateur $R^2(f_h, Y)$ quantifie la part de variance de f_h expliquée par les classes. Un R^2 proche de 1 indique que la composante est fortement alignée avec la structure induite par la variable de groupe et qu'elle possède une capacité discriminante élevée.

L'implémentation suivante exploite les résultats de la fonction `adpls_part1` et procède, pour chaque composante, au calcul des normes pondérées nécessaires à l'obtention des deux indicateurs.

```

adpls_part2 <- function(X, Y, H = 2, w = NULL, M = NULL, center_scale = TRUE) {
  # réutilise la fonction adpls_part1
}

```

```

part1 <- adpls_part1(X, Y, H, w, M, center_scale)

# préparation des objets utiles
X <- as.matrix(X)
W <- diag(part1$w)
M <- part1$M
F <- part1$F
U <- part1$U
Xhat <- part1$Xhat
E <- part1$E

total_inertia <- sum(diag(t(X) %*% (W %*% X)))

# réservation des vecteurs résultats
S_vec <- numeric(H)
R2_vec <- numeric(H)

# boucle sur h
for (h in seq_len(H)) {
# extraction de f_h
  f_h <- F[, h, drop = FALSE]
# norme pondérée
  fnorm2 <- as.numeric(t(f_h) %*% (W %*% f_h))
# calcul de S(f_h)
  S_vec[h] <- fnorm2 / total_inertia
# projection PiY*f_h
  fhat_h <- part1$PiY %*% f_h
# norme pondérée de la projection
  top_val <- as.numeric(t(fhat_h) %*% (W %*% fhat_h))
# calcul du coefficient de détermination R^2
  R2_vec[h] <- top_val / fnorm2
}

# les indicateurs au résultat final
part1$S <- S_vec
part1$R2 <- R2_vec

return(part1)
}

```

Le vecteur `S_vec` regroupe les proportions d'inertie globale expliquées par chacune des composantes discriminantes, tandis que le vecteur `R2_vec` fournit quant à lui une mesure synthétique de la part de variance des composantes imputable aux classes.

Ces deux indicateurs constituent la base de l'interprétation exploratoire de l'ADPLS : les composantes associées à un $S(f_h)$ élevé et un $R^2(f_h, Y)$ proche de 1 sont celles qui contribuent le plus efficacement à la discrimination entre groupes.

3. Calcul des coordonnées des centres de gravité des q classes sur les H axes discriminants

L'objectif de cette étape est de déterminer, pour chacune des q classes considérées, les coordonnées de son centre de gravité dans l'espace défini par les H premières composantes discriminantes issues de l'ADPLS.

On note $F = [f_1, f_2, \dots, f_H]$ la matrice des composantes discriminantes ($n \times H$). Chaque individu i a un vecteur de scores factoriels $f_i = (f_{i1}, \dots, f_{iH})$.

La matrice $Y(n \times q)$ est une matrice indicatrice telle que $Y_{ik} = 1$ si l'individu i appartient à la classe k , et 0 sinon.

La coordonnée barycentrique d'une classe k sur l'axe h correspond à la moyenne pondérée des scores individuels, selon les poids définis par la matrice W :

$$g_{kh} = \frac{\sum_{i=1}^n W_i Y_{ik} f_{ih}}{\sum_{i=1}^n W_i Y_{ik}}$$

Sous forme matricielle, cette expression devient :

$$G = (Y'WY)^{-1}Y'WF$$

où :

- G est une matrice $q \times H$,
- chaque ligne correspond au barycentre d'une classe dans l'espace factoriel,
- chaque colonne correspond à un axe discriminant.

Ainsi, G fournit la position moyenne de chaque classe dans le sous-espace discriminant, selon la métrique pondérée par W , assurant la cohérence avec la construction précédente des composantes.

L'implémentation en R est donnée ci-dessous.

```
adpls_part3 <- function(X, Y, H = 2, w = NULL, M = NULL, center_scale = TRUE) {
  # réutilise la fonction adpls_part2
  part2 <- adpls_part2(X, Y, H, w, M, center_scale)

  # préparation des objets utiles
  W <- diag(part2$w)
  F <- part2$F
  Y <- as.matrix(Y)

  # calcul des centres de gravité
  YtWY <- t(Y) %*% W %*% Y
  YtWF <- t(Y) %*% W %*% F
  G <- solve(YtWY, YtWF)

  rownames(G) <- colnames(Y)
  colnames(G) <- paste0("Axe_", seq_len(H))

  # résultats
  part2$centers <- G
}

return(part2)
}
```

La matrice G obtenue contient donc les coordonnées pondérées des centres de gravité des classes dans le sous-espace discriminant. Chaque ligne de G représente la position barycentrique d'une classe sur les H axes principaux.

Ces coordonnées constituent une étape essentielle pour l'interprétation géométrique : elles permettent de visualiser la disposition relative des groupes dans le plan factoriel (h, m) , ce qui sera exploité ultérieurement lors de la représentation graphique (question 5).

4. Calcul des coordonnées des variables de X

Cette étape consiste à déterminer la position des variables initiales X_j dans l'espace engendré par les H omposantes discriminantes obtenues par l'ADPLS. L'étude de ces coordonnées permet d'évaluer la contribution de chaque variable à la construction des axes discriminants et, par conséquent, son rôle dans la discrimination entre les groupes.

Pour une variable centrée-réduite X_j et une composante discriminante f_h , la corrélation pondérée est définie par :

$$r_{jh} = \frac{\langle X_j, f_h \rangle_W}{\|X_j\|_W \|f_h\|_W}$$

où :

- X_j : j -ième variable centrée-réduite de X ,
- f_h : h -ième composante discriminante,
- $\langle \cdot, \cdot \rangle_W$: le produit scalaire pondéré par la matrice W ,
- $\|\cdot\|_W$: la norme associée à ce produit scalaire.

Sous forme matricielle, la matrice des corrélations entre les variables et les axes discriminants s'écrit :

$$C = \text{corr}(X, F) = D_X^{-1} (X'WF) D_F^{-1}$$

avec :

- D_X la matrice diagonale des normes pondérées des variables de X_j ,
- D_F la matrice diagonale des normes pondérées des composantes f_h .

La matrice C (de dimension $p \times H$) regroupe les coordonnées factorielles des variables sur les différents axes.

```
adpls_part4 <- function(X, Y, H = 2, w = NULL, M = NULL, center_scale = TRUE) {
  # réutilise la fonction adpls_part3
  part3 <- adpls_part3(X, Y, H, w, M, center_scale)

  # préparation des objets utiles
  W <- diag(part3$w)
  F <- part3$F

  # X supposé centré-réduit
  X <- scale(X, center = TRUE, scale = TRUE)
  n <- nrow(X)
  p <- ncol(X)

  # matrice des corrélations
  C <- matrix(0, nrow = p, ncol = H)
```

```

# corrélations pondérées entre X_j et f_h
for (j in 1:p) {
  xj <- X[, j]
  norm_xj <- sqrt(as.numeric(t(xj) %*% (W %*% xj)))

  for (h in 1:H) {
    fh <- F[, h]
    norm_fh <- sqrt(as.numeric(t(fh) %*% (W %*% fh)))
    if (norm_xj == 0 || norm_fh == 0) {
      C[j, h] <- 0
    } else {
      C[j, h] <- as.numeric(t(xj) %*% (W %*% fh)) / (norm_xj * norm_fh)
    }
  }
}

rownames(C) <- colnames(X)
colnames(C) <- paste0("Axe_", seq_len(H))

# résultats
part3$var_coords <- C

return(part3)
}

```

Les coefficients de la matrice C représentent les corrélations pondérées entre les variables initiales et les axes discriminants issus de l'ADPLS. Une valeur absolue élevée (proche de 1) indique une forte contribution de la variable à la construction de l'axe considéré, tandis qu'une valeur faible (proche de 0) traduit une influence limitée, voire nulle.

Ainsi, l'analyse de la matrice C constitue une étape fondamentale pour relier les composantes discriminantes aux variables d'origine. Elle permet d'interpréter la signification géométrique des axes obtenus, en identifiant les variables qui structurent le plus efficacement la séparation entre les groupes.

Fonction générale — intégration du processus ADPLS

Afin de rassembler l'ensemble des étapes de calcul présentées précédemment (questions 1 à 4), nous avons implémenté une fonction générale `adpls_full()`.

Cette fonction constitue une synthèse opérationnelle de l'approche ADPLS et permet d'obtenir, à partir des matrices d'entrée X et Y , l'ensemble des résultats requis pour l'analyse discriminante partielle par moindres carrés.

```

adpls_full <- function(X, Y, H=2, w=NULL, M=NULL, center_scale=TRUE){

  # fonctions utilitaires internes
  Wmat <- function(w) {
    if (is.null(w)) return(NULL)
    diag(as.numeric(w), nrow = length(w))
  }

  w_norm2 <- function(v, w) {
    if (is.null(w)) return(sum(v^2))
    as.numeric(t(v) %*% (w * v))
  }
}

```

```

}

w_prod <- function(a, b, w) {
  if (is.null(w)) return(as.numeric(t(a) %*% b))
  as.numeric(t(a) %*% (w * b))
}

M_half <- function(M) {
  ev <- eigen(M, symmetric = TRUE)
  vals <- pmax(ev$values, 0)
  vecs <- ev$vectors
  vecs %*% diag(sqrt(vals), length(vals)) %*% t(vecs)
}

M_half_inv <- function(M) {
  ev <- eigen(M, symmetric = TRUE)
  vals <- ev$values
  vecs <- ev$vectors
  invvals <- ifelse(vals > 0, 1 / sqrt(vals), 0)
  vecs %*% diag(invvals, length(invvals)) %*% t(vecs)
}

if (is.null(w)) w <- rep(1 / nrow(X), nrow(X))

# étapes de calcul
part1 <- adpls_part1(X = X, Y = Y, H = H, w = w, M = M, center_scale = center_scale)
part2 <- adpls_part2(X = X, Y = Y, H = H, w = w, M = M, center_scale = center_scale)
part3 <- adpls_part3(X = X, Y = Y, H = H, w = w, M = M, center_scale = center_scale)
part4 <- adpls_part4(X = X, Y = Y, H = H, w = w, M = M, center_scale = center_scale)

# résultats
return(list(F=part1$F,
            Ftilde=part1$Ftilde,
            S=part2$S,
            R2=part2$R2,
            centers=part3$centers,
            var_coords=part4$var_coords))
}

```

5. Représentation graphique des individus, des centres de gravité et des variables

Dans cette dernière étape, nous devons visualiser les résultats obtenus dans les étapes précédentes.

L'objectif comporte trois volets :

1. Afficher les individus dans le plan factoriel (h, m) choisi par l'utilisateur.
2. Afficher les centres de gravité des q classes sur ce même plan.
3. Afficher les variables de X projetées sur le plan choisi et tracer le cercle unité.

Pour cela, nous utilisons la fonction `plot_adpls()` qui génère la figure dans le plan (h, m) .

Cette fonction affiche :

- les individus colorés selon leur classe,

- les centres de gravité des q classes,
- les variables de X projetées sur le plan choisi,
- le cercle unité, pour faciliter l'interprétation des corrélations.

```

library(RColorBrewer)

plot_adpls <- function(out, Y = NULL, Y_labels = NULL, h = 1, m = 2,
                      show_variables = TRUE, circle = TRUE,
                      main = NULL, cex_ind = 0.7, cex_cent = 1.0,
                      xlim = NULL, ylim = NULL) {

  Ftilde <- out$Ftilde
  centers <- out$centers
  var_coords <- out$var_coords
  n <- nrow(Ftilde)

  # identification des étiquettes de classes
  if (is.null(Y_labels) & !is.null(Y)) {
    lab <- apply(Y, 1, function(r) which(r == 1))
    Y_labels <- factor(lab)
  } else if (is.null(Y_labels)) {
    stop("Y_labels ou Y doivent être fournis!")
  }

  # individus
  xind <- Ftilde[, h]
  yind <- Ftilde[, m]
  classes <- as.factor(Y_labels)
  cols <- rainbow(length(levels(classes)))

  # palette de couleurs
  n_classes <- length(levels(classes))
  cols <- brewer.pal(min(max(n_classes, 3), 8), "Set2")
  if (n_classes > 8) cols <- rep(cols, length.out = n_classes)

  # tracé des individus
  plot(xind, yind,
        col = adjustcolor(cols[as.numeric(classes)], alpha.f = 0.7),
        pch = 19,
        cex = cex_ind,
        xlab = paste0("Axe ", h),
        ylab = paste0("Axe ", m),
        main = main,
        bg = "white",
        xlim = xlim,
        ylim = ylim)

  legend("topright",
         legend = levels(classes),
         col = cols,
         pch = 19,
         cex = 1,
         title = "Classes",
         bg = "white",

```

```

    box.lwd = 0.5,
    box.col = "black",
    text.col = "black")

# centres de gravité
points(centers[, h], centers[, m], col = "black", pch = 8, cex = cex_cent)
text(centers[, h], centers[, m], labels = rownames(centers), pos = 3)

# variables dans le plan
if (show_variables) {
  arrows(rep(0, nrow(var_coords)), rep(0, nrow(var_coords)),
         var_coords[, h], var_coords[, m],
         length = 0.07, col = "grey40", lwd = 1)

  text(var_coords[, h], var_coords[, m],
        labels = rownames(var_coords),
        cex = 0.7, col = "grey20")

# cercle unité pour corrélations
  if (circle) {
    angles <- seq(0, 2*pi, length.out = 200)
    lines(cos(angles), sin(angles), lty = 2, col = "black")
  }
}

# opérateur utilitaire : renvoie a si non nul, sinon b
`%||%` <- function(a, b) if (!is.null(a)) a else b

```

Conclusion

Dans cette deuxième partie, nous avons développé l'ensemble des étapes nécessaires à la mise en œuvre de l'ADPLS. Les fonctions programmées permettent d'obtenir les composantes discriminantes, d'évaluer leur qualité, de calculer les centres de gravité des classes et de mesurer la contribution des variables. La fonction graphique rassemble ensuite ces résultats dans une représentation claire sur les plans factoriels, facilitant ainsi l'interprétation. L'ensemble constitue un outil fonctionnel et accessible, qui automatise l'application de la méthode ADPLS et rend l'analyse plus efficace.

Partie 3 — Application: types forestiers du bassin du Congo

Chargement des données et ADPLS exploratoire

Tout d'abord, nous chargeons le fichier *genus* et réalisons une analyse ADPLS exploratoire de la variable $Y = \text{forest}$ sur les 27 variables de composition arborée $X = [\text{gen}1, \dots, \text{gen}27]$. Avant l'analyse, toutes les variables X sont centrées et réduites.

```

data <- read.table("Datagenus.csv", header = TRUE)
X <- scale(as.matrix(data[, paste0("gen", 1:27)])), center = TRUE, scale = TRUE)
grp <- as.factor(data$forest)
Y <- model.matrix(~ grp - 1)

```

Une attention particulière est portée au choix de la métrique M . Nous allons comparer deux options :

- $M = I$ — matrice identité, métrique standard sans pondération.
- $M = \text{diag}\left(\frac{1}{\text{var}(X_j)}\right)$ — inverse des variances des variables, pour tenir compte des différences d'échelle.

En comparant les résultats obtenus avec ces différentes métriques, nous pourrons choisir celle qui met le mieux en évidence la structure discriminante entre les types forestiers.

Metrique

$M = I$

```
M1 <- diag(1, ncol(X))

#adls
out1 <- adpls_full(X, Y, H = 3, w = rep(1 / nrow(X), nrow(X)), M = M1, center_scale = FALSE)

data.frame(
  Axe = 1:3,
  S = round(out1$S, 3),
  R2 = round(out1$R2, 3),
  Discriminant = round(out1$S * out1$R2, 3)
)

##   Axe      S     R2 Discriminant
## 1    1 0.184 0.200      0.037
## 2    2 0.116 0.178      0.021
## 3    3 0.051 0.090      0.005
```

Le tableau des indicateurs montre que la première composante (axe 1) concentre la part la plus importante de l'inertie totale ($S_1 = 18,4\%$) et qu'elle est la plus corrélée avec la variable de groupe ($R_1^2 = 0,20$). Son pouvoir discriminant (0,037) est donc le plus élevé. Elle représente l'axe principal de séparation entre les types de forêts. La deuxième composante (0,021) contribue de manière secondaire, tandis que la troisième (0,005) joue un rôle marginal.

Metrique

$M = \text{diag}\left(\frac{1}{\text{var}(X_j)}\right)$

```
X2 <- as.matrix(data[, paste0("gen", 1:27)])
grp <- as.factor(data$forest)
Y <- model.matrix(~ grp - 1)

variances2 <- apply(X2, 2, var)
M2 <- diag(1 / variances2)

# standartisation
X_scaled2 <- scale(X, center = TRUE, scale = TRUE)

# adpls
```

```

out_M2 <- adpls_full(X_scaled2, Y, H = 3, M = M2)

# S, R2, Discriminant
data.frame(
  Axe = 1:3,
  S = round(out_M2$S, 3),
  R2 = round(out_M2$R2, 3),
  Discriminant = round(out_M2$S * out_M2$R2, 3)
)

```

	Axe	S	R2	Discriminant
## 1	1	0.017	0.181	0.003
## 2	2	0.009	0.130	0.001
## 3	3	0.008	0.051	0.000

Interpretation des résultats

```

data <- read.table("Datagenus.csv", header = TRUE)

X <- scale(as.matrix(data[, paste0("gen", 1:27)]), center = TRUE, scale = TRUE)
grp <- as.factor(data$forest)
Y <- model.matrix(~ grp - 1)

#choosing M
M <- diag(1, ncol(X))

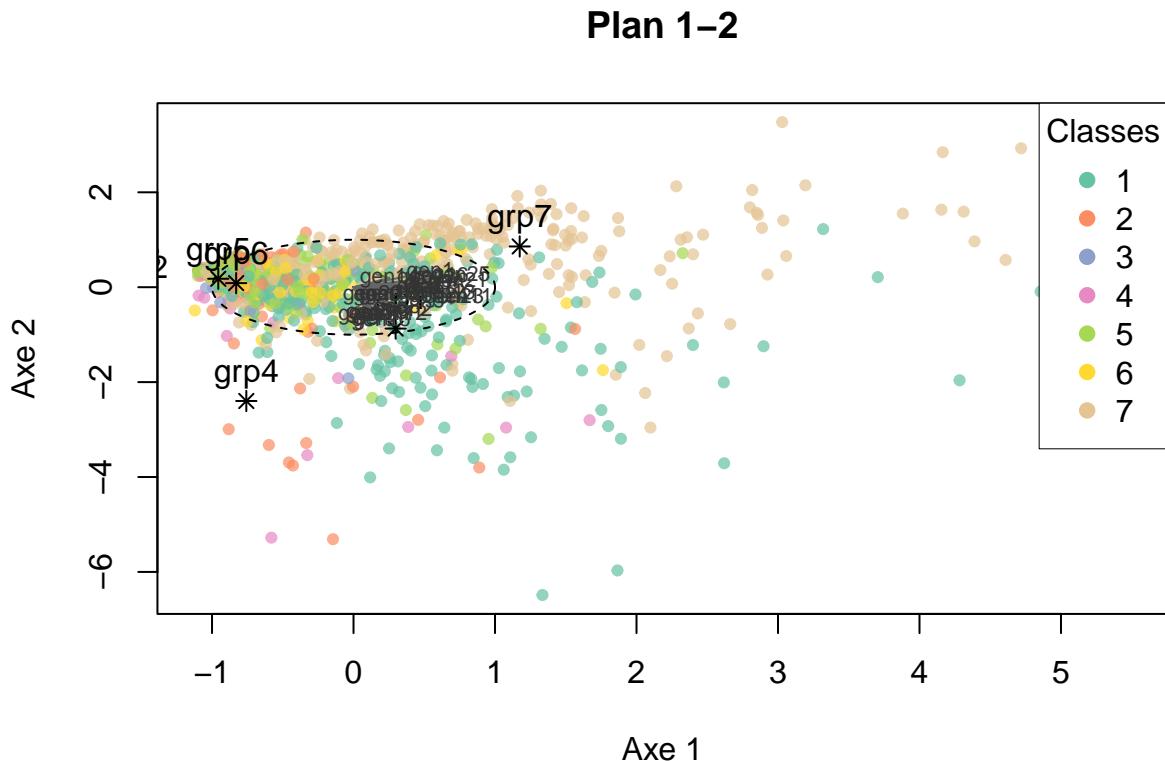
#adls
out <- adpls_full(X, Y, H = 3, w = rep(1 / nrow(X), nrow(X)), M = M, center_scale = FALSE)

data.frame(
  Axe = 1:3,
  S = round(out$S, 3),
  R2 = round(out$R2, 3),
  Discriminant = round(out$S * out$R2, 3)
)

```

	Axe	S	R2	Discriminant
## 1	1	0.184	0.200	0.037
## 2	2	0.116	0.178	0.021
## 3	3	0.051	0.090	0.005

```
plot12 <- plot_adpls(out, Y = Y, Y_labels = grp, h = 1, m = 2, show_variables = TRUE, main = "Plan 1-2")
```

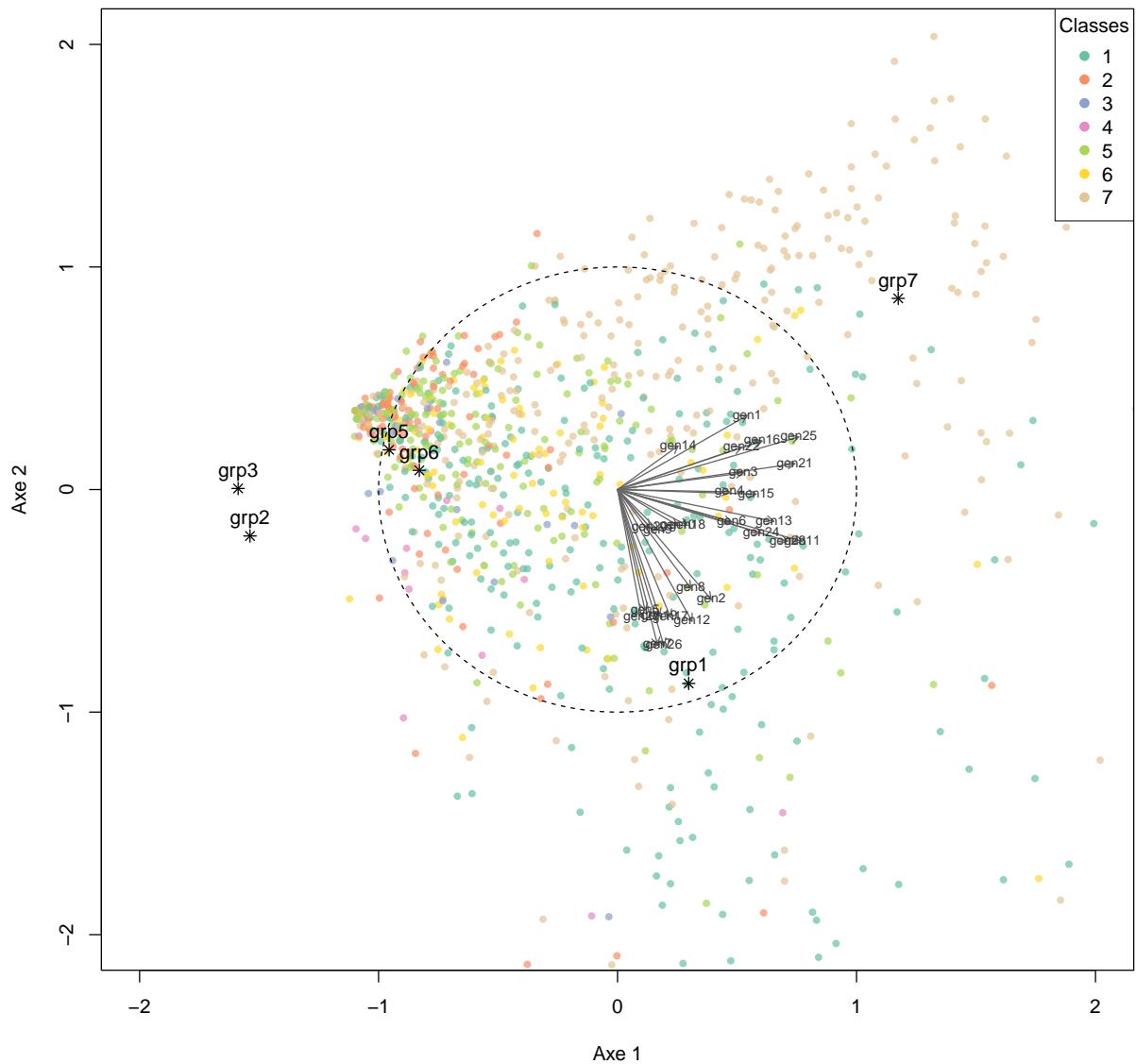


Le graphique de projection sur le plan formé par les axes 1 et 2 illustre cette structure. Chaque point représente une forêt colorée selon sa classe, et les croix indiquent les centres de gravité des groupes. On observe une séparation nette entre les classes 4 et 7, qui apparaissent bien distinctes dans l'espace factoriel. Cela suggère que ces deux types de forêts possèdent des compositions en genres d'arbres clairement différencierées.

En revanche, les classes 5 et 6 sont proches l'une de l'autre : leurs centres de gravité sont situés dans une même zone du plan, ce qui traduit une similarité partielle dans leur composition floristique ou un recouvrement des caractéristiques discriminantes identifiées par les deux premières composantes. Cette proximité peut indiquer soit une continuité écologique entre ces deux types de forêts, soit une difficulté du modèle à les séparer de manière nette.

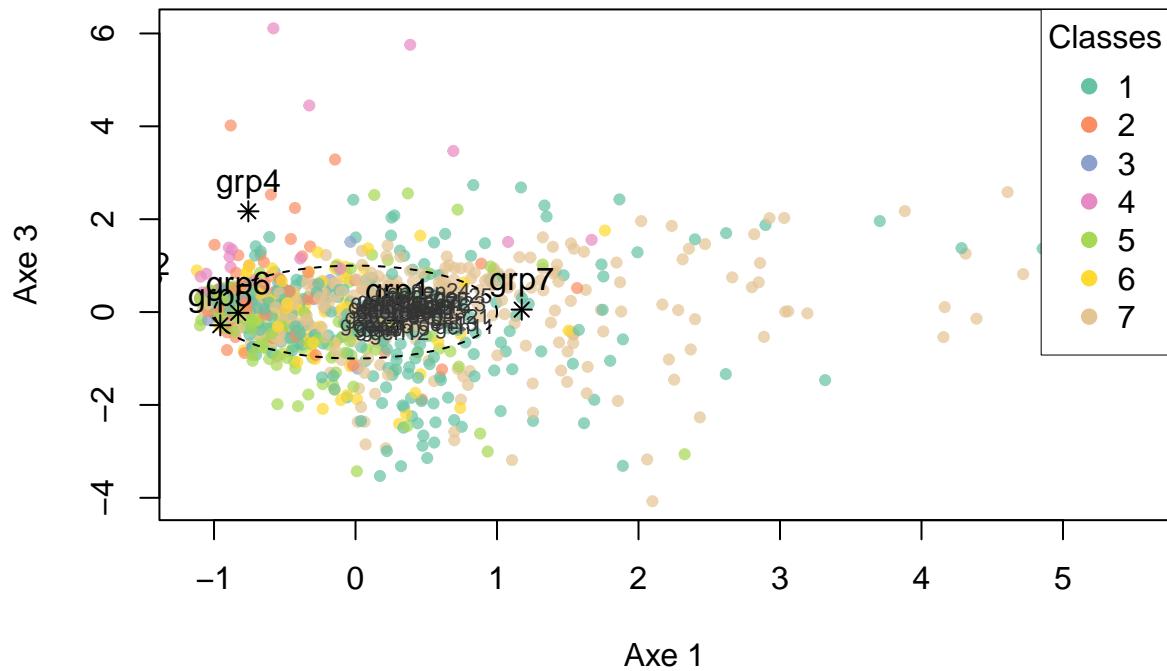
```
plot_adpls(
  out,
  Y = Y,
  Y_labels = grp,
  h = 1,
  m = 2,
  show_variables = TRUE,
  main = "Plan 1-2",
  xlim = c(-2, 2),
  ylim = c(-2, 2)
)
```

Plan 1–2



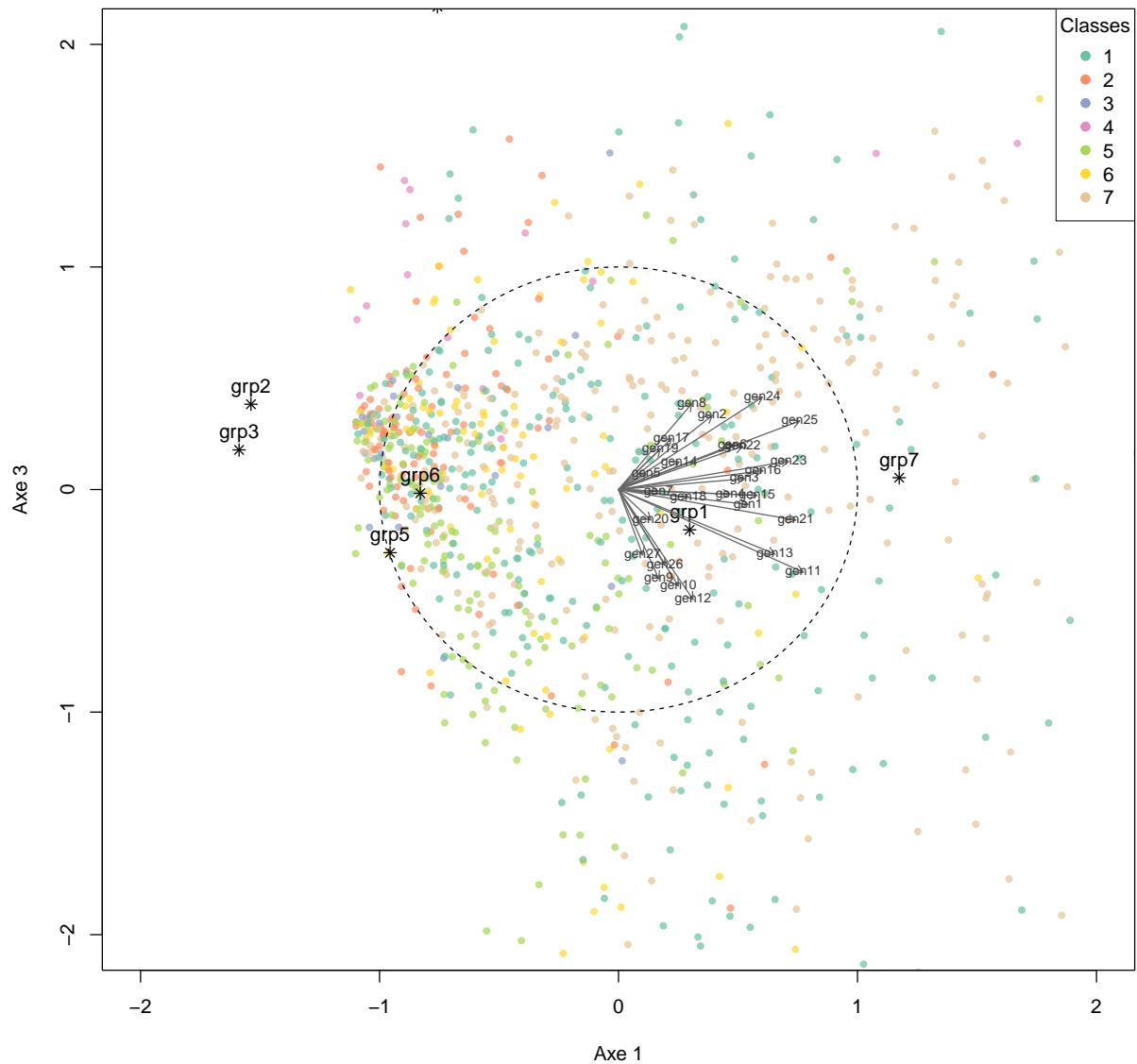
```
plot13 <- plot_adpls(out, Y = Y, Y_labels = grp, h = 1, m = 3, show_variables = TRUE, main = "Plan 1-3")
```

Plan 1–3



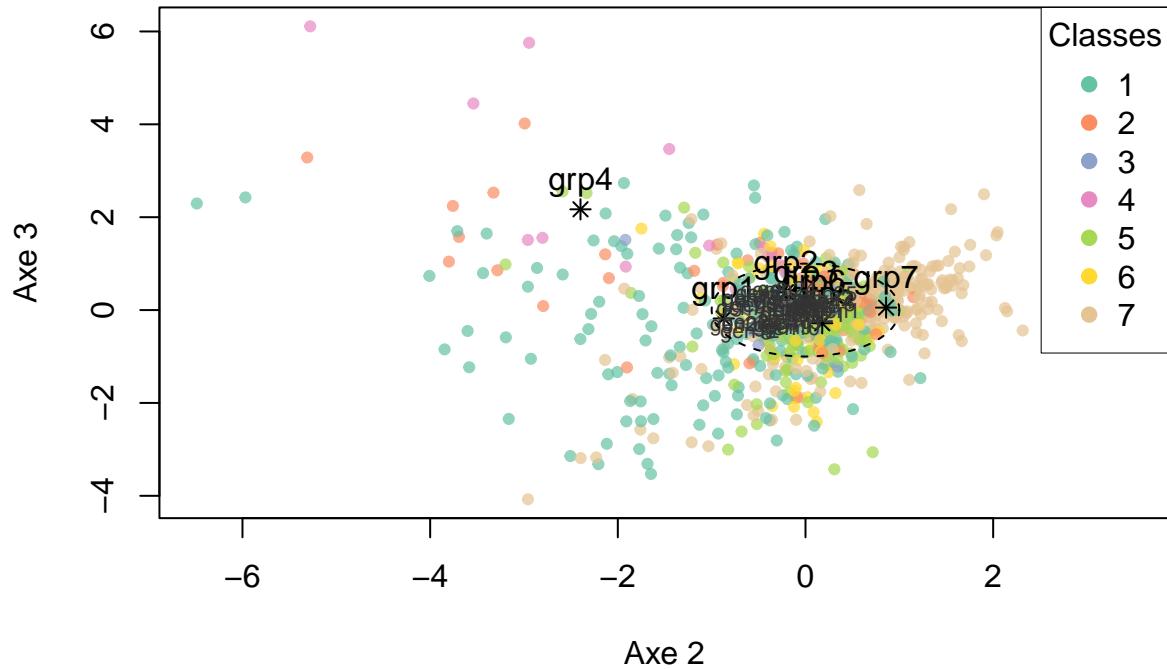
```
plot_adpls(  
  out,  
  Y = Y,  
  Y_labels = grp,  
  h = 1,  
  m = 3,  
  show_variables = TRUE,  
  main = "Plan 1-3",  
  xlim = c(-2, 2),  
  ylim = c(-2, 2))
```

Plan 1–3



```
plot23 <- plot_adpls(out, Y = Y, Y_labels = grp, h = 2, m = 3, show_variables = TRUE, main = "Plan 2-3")
```

Plan 2-3



```
plot_adpls(
  out,
  Y = Y,
  Y_labels = grp,
  h = 2,
  m = 3,
  show_variables = TRUE,
  main = "Plan 2-3",
  xlim = c(-1, 1),
  ylim = c(-1, 1)
)
```

Plan 2–3

