



# R- Package Clustering de Variables

**Maissa Lajimi**  
**Yassine Cheniour**  
**Lamia Hatem**

Lien du dépôt :

[https://github.com/maissaladjimi/SISE\\_Clustering\\_Variables\\_R/](https://github.com/maissaladjimi/SISE_Clustering_Variables_R/)

# SOMMAIRE

01 ARCHITECTURE DU PACKAGE

02 K-MEANS → VAR QUANTI

03 VARCLUS → VAR QUANTI

04 CAH (DICE/ACM) → VAR QUALI

05 APPLICATION SHINY

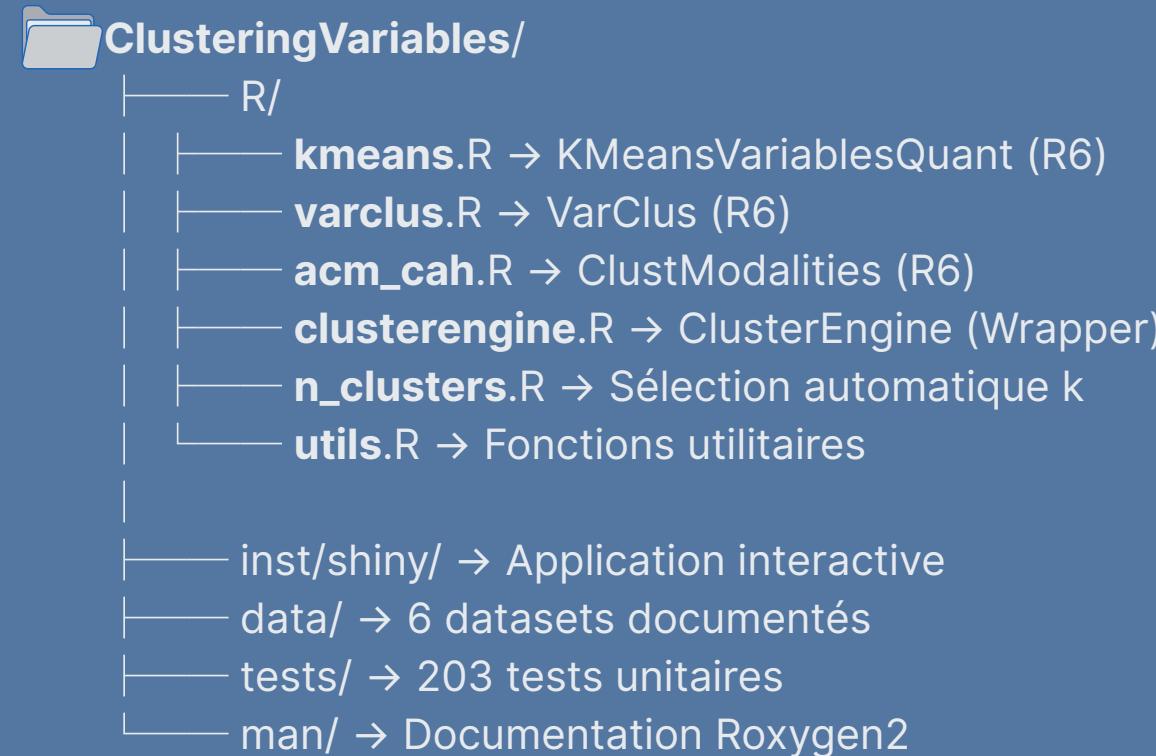


# ARCHITECTURE DU PACKAGE



3 algorithmes = 3 classes indépendantes :

- Indépendance des développements
- Faciliter maintenance et correction



## Classes R6 :

- **Constructeur** : km <- KMeansVariablesQuant\$new(k = 3)
- Entrainement : km\$fit(data)
- Infos succinctes : km\$print()
- Résultats : km\$summary()
- Visualisation et Interprétation : km\$plot\_correlation\_circle()

# ARCHITECTURE : CLUSTER\_ENGINE

## ClusterEngine

- └── **data** : Données
- └── **method** : "kmeans" / "varclus" / ...
- └── **n\_clusters** : Nombre de clusters
- └── **model** : Instance de la classe R6

### Méthodes :

- └── **fit()** → Crée et ajuste le modèle
- └── **predict()** → Délègue à `model$predict()`
- └── **summary()** → Délègue à `model$summary()`
- └── **elbow()** → Sélection auto k

### Fonctionnement du Wrapper :

- Création de l'instance et délégation selon la méthode souhaitée
- Le constructeur de `clusterengine` créé en réalité un objet d'une des 3 classes R6 et appelle ses méthodes

### Améliorations :

- L'app Shiny n'utilise pas `clusterengine`, appelle directement les classes (déboggage)
- `elbow` ne fonctionne pas très bien pour CAH

Exemples →

```
library(ClusteringVariables)

data(crime)

# quasi-unifiée pour les 3 méthodes
engine <- ClusterEngine$new(crime[,-1], "kmeans", n_clusters=3)

engine$fit() # Délègue vers KMeansVariablesQuant

engine$summary() # Appelle model$summary()

engine$predict(new_vars)
```

# ARCHITECTURE : N\_CLUSTERS

Contient des fonctions de détection du nombre de k optimal

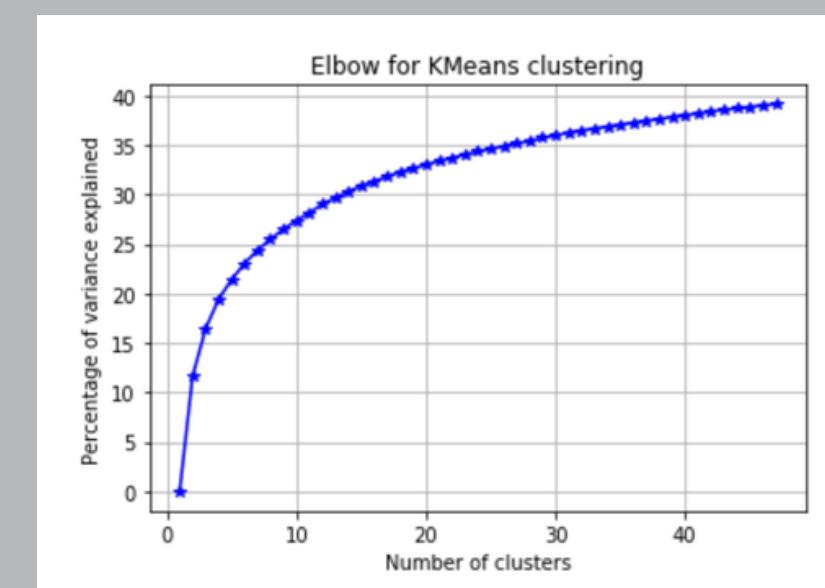
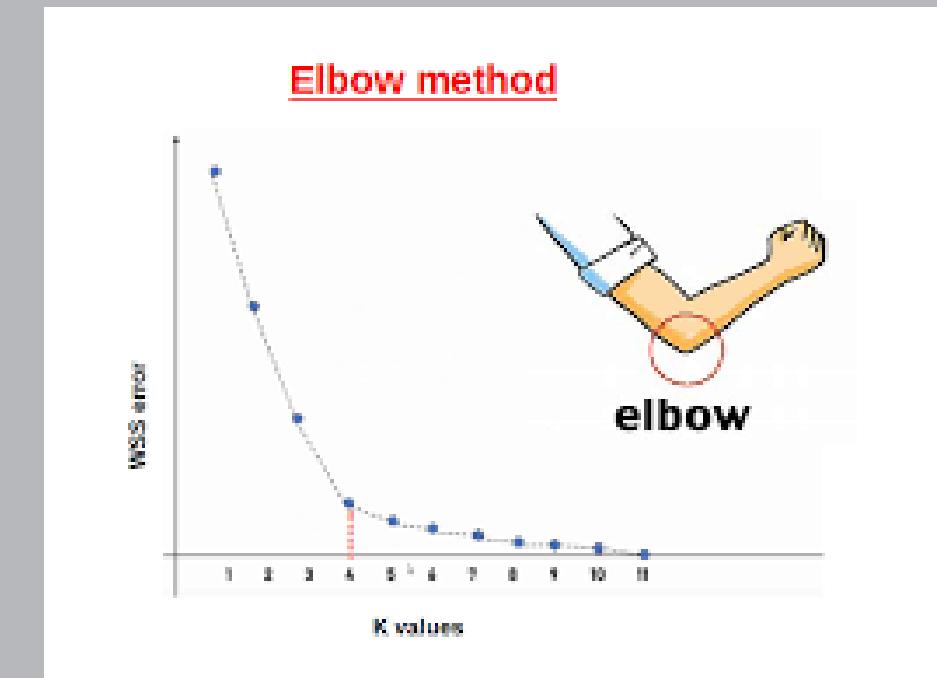
2 approches selon l'algorithme :

→ Kmeans et Varclus : **distance perpendiculaire** (droite de min à max : on cherche le point le plus éloigné de la droite, c'est celui qui indique le meilleur k.)

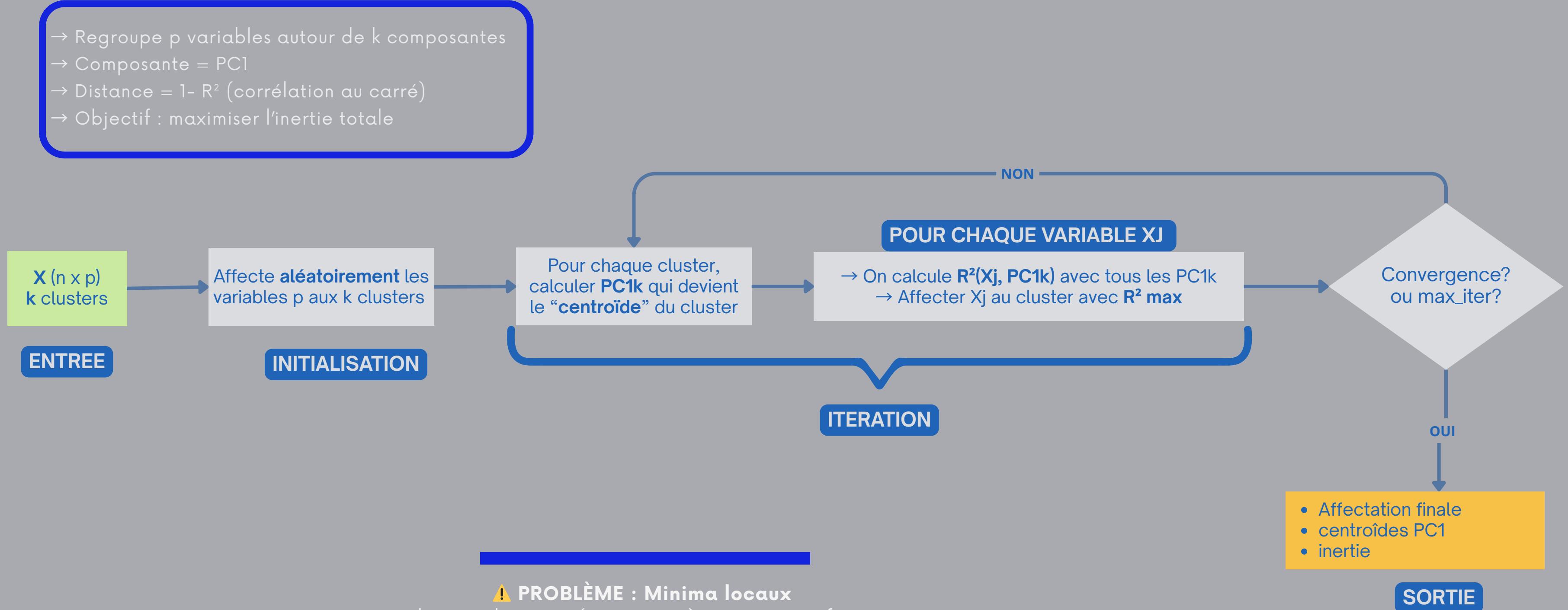
→ CAH : A partir du dendrogramme, on prend le plus grand saut entre deux hauteurs, c'est ce qu'on appelle le **gain marginal** (= hauteur( $k-1$ ) - hauteur( $k$ ),  $k$  optimal est le plus grand saut)

Le k optimal proposé, n'est pas toujours le plus optimal

→ Il est plus fiable de regarder l'elbow plot



# K-MEANS : PRINCIPE



## ⚠ PROBLÈME : Minima locaux

SOLUTION : Multi-initialisations ( $n_{init} = 10$ ) → Lancer 10 fois avec affectations initiales différentes → Garder la solution avec inertie maximale

# K-MEANS : CLASSE R6

## ATTRIBUTS : 3 TYPES

### PARAMETRES ALGO :

- k :
- max\_iter
- n\_init
- tol
- seed

### DONNÉES :

- data
- X\_scaled
- scale\_center
- scale\_scale

### RÉSULTATS :

- clusters
- centers
- r2\_matrix
- inertia\_total
- inertia\_by\_cluster
- n\_iter

- : Nombre de cluster ( $\geq 2$ )
- : Max itérations (défaut: 100)
- : Nb initialisations (défaut: 10)
- : Tolérance (défaut: 1e-4)
- : Graine aléatoire (optionnel)
- 
- : data.frame original
- : Matrice centrée-réduite ( $n \times p$ )
- : Paramètres centrage (predict)
- : Paramètres échelle (predict)
- 
- : Vecteur affectation (long.  $p$ )
- : Matrice centroïdes PC1 ( $n \times k$ )
- : Matrice  $R^2$  ( $p \times k$ )
- : Inertie totale  $\sum k$
- : Inertie par cluster
- : Nb itérations convergence

## METHODES PRINCIPALES

### OBLIGATOIRES :

- **\$new(k, max\_iter, n\_init, tol, seed)**
  - → Constructeur avec validation paramètres
- **\$fit(X, k=NULL)**
  - → Ajustement : centrage-réduction + clustering
  - → Lance  $n_{init}$  runs, garde meilleure inertie
- **\$print()**
  - → Affichage succinct ( $k, n, p, \text{inertie}, \text{tailles}$ )
- **\$summary(print\_output=TRUE)**
  - → Statistiques détaillées (qualité,  $R^2$ , corrél.)
- **\$predict(new\_data)**
  - → Affecte nouvelle variable au cluster optimal
  - → Retourne  $R^2$  avec chaque centroïde

### AUTRES METHODES :

- **\$illustrative(X\_illust, plot=TRUE)**
  - → Analyse complète variables illustratives
  - → Régression sur PC1, visualisation ACP
- **\$plot\_elbow(k\_range=2:10)**
  - → Sélection automatique  $k$  optimal
- **\$plot\_correlation\_circle(dims=c(1,2))**
  - → Cercle des corrélations (ACP des variables)
- **\$plot\_biplot(dims=c(1,2))**
  - → Biplot variables dans l'espace factoriel
- **\$get\_clusters\_table()** → DataFrame avec affectations triées

## METHODES PRIVEES

### private\$latent\_center(X\_cluster)

- → Calcule PC1 d'un cluster par ACP
- → Gère cas particuliers : cluster vide, 1 variable, variance nulle
- → Retourne centroïde normalisé (mean=0, sd=1)

### private\$single\_run(X, K, max\_iter, tol)

- → Une exécution complète de l'algorithme
- → Init aléatoire → Boucle itérative → Convergence →
- Retourne : clusters, centers, r2\_matrix, inertia

### private\$detect\_elbow(k\_vals, y\_vals)

- → Détection  $k$  optimal par distance perpendiculaire
- → Utilisé dans \$plot\_elbow()

# K-MEANS : INTERPRÉTATION

## Métriques

- **Print** (Nb\_clusters (k), max\_iter, Nb\_init, Convergence tolerance, Inertie totale)
- **Summary**
  - **Global quality**
    - Inertia\_total : Variance totale capturée par toutes les PC1
    - avg\_inertia : Inertie moyenne par cluster
    - pct\_explained : % de l'inertie maximale atteinte
  - **Cluster Summary**
    - members : nb de variables
    - Inertia (variation explained) : Valeur propre de la PC1 du cluster
    - proportion explained(  $\lambda_1/\text{members}$ ) : proportion moyenne de variance expliquée par PC1
  - **Cor\_latent** : correlation entre PC1 de chaque cluster
  - **R<sup>2</sup> matrix** : R<sup>2</sup> entre chaque variable et chaque PC1
- **Cluster table**
  - Table d'affectations

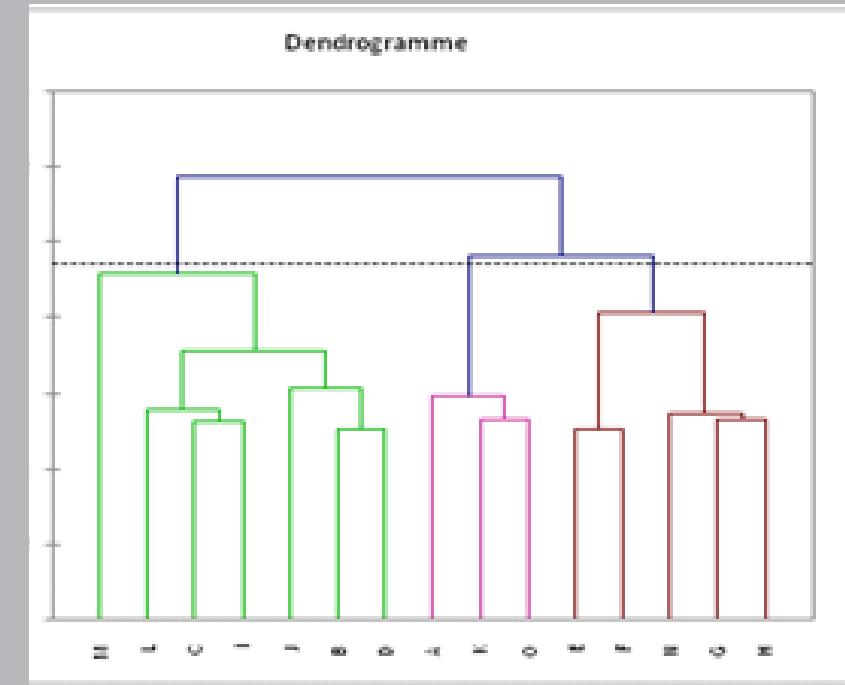
## Visualisations

- **Cercle de corrélation**
  - Variables projetées sur les 2 premiers axes factoriels d'une ACP globale
- **Biplot**
  - Projection variables dans espace factoriel
- **Elbow plot**
  - Inertie totale en fonction de k
  - k optimal → calcul avec distance perpendiculaire

# CAH- PRINCIPE

## Contexte :

- > On ne clusterise pas directement les variables mais leurs **modalités**
- > 2 méthodes proposées : **DICE** ou **ACM**
- > Résultat final : **Groupes de MODALITÉS**
  - Variables regroupées par associations de modalités similaires



## DICE

- > **Table disjonctive** complète (Matrice indicatrice 0/1)
- > Calcule **distance DICE<sup>2</sup>** entre chaque paire de modalités
  - $d^2\text{Dice}(mj, mj') = \frac{1}{2} \sum_i (x_{ij} - x_{ij'})^2$
- > **CAH** avec critère de ward (ward.D2) pour agréger modalités
  - Agrégation **ascendante** (bottom-up)
  - **Minimise** variance intra-cluster

## ACM

- > **Table disjonctive** complète (Matrice indicatrice 0/1)
- > **Projection** des modalités dans un espace réduit (avec ACM)
- > **Pondération** des coordonnées factorielles par la racine des VP
  - donnent plus de poids aux axes plus informatifs
- > **CAH** sur les coordonnées pondérées avec ward.D
- > **Correction** des VP pour interprétation (Greenacre et Benzécri)
  - Corriger la redondance artificielle

# CAH: CLASSE R6

ATTRIBUTS : 4 TYPES	METHODES PRINCIPALES	AUTRES METHODES	
<p><b>PARAMETRES ALGO :</b></p> <ul style="list-style-type: none"><li>• method</li><li>• n_axes</li></ul> <p><b>DONNÉES :</b></p> <ul style="list-style-type: none"><li>• data</li><li>• disj</li></ul> <p><b>RÉSULTATS ACM :</b></p> <ul style="list-style-type: none"><li>• acm</li><li>• eig_raw</li><li>• eig_benzecri</li><li>• eig_greenacre</li><li>• mod_coords</li><li>• ind_coords</li></ul> <p><b>RÉSULTATS CLUSTERING :</b></p> <ul style="list-style-type: none"><li>• dist_mat</li><li>• hclust</li><li>• k</li><li>• mod_clusters</li></ul>	<p>: "dice" ou "acm"</p> <p>: Nb axes ACM (NULL = auto)</p> <p>: data.frame variables quali</p> <p>: Table disjonctive complète</p> <p>: Objet ade4::dudi.coa()</p> <p>: Valeurs propres brutes</p> <p>: VP corrigées (Benzecri)</p> <p>: VP corrigées (Greenacre)</p> <p>: Coord. modalités (pondérées)</p> <p>: Coord. individus</p> <p>: Matrice distances modalités</p> <p>: Objet hclust (dendrogramme)</p> <p>: Nombre de clusters</p> <p>: Affectation modalités</p>	<p><b>OBLIGATOIRES :</b></p> <ul style="list-style-type: none"><li>• <b>\$new(method, n_axes)</b><ul style="list-style-type: none"><li>◦ → Constructeur avec choix méthode</li></ul></li><li>• <b>\$fit(X, k=NULL)</b><ul style="list-style-type: none"><li>◦ → DICE : TDC → distance Dice<sup>2</sup> → CAH (ward.D2)</li><li>◦ → ACM : TDC → ACM → coord pondérées → CAH (ward.D) + corrections VP</li></ul></li><li>• <b>\$print()</b><ul style="list-style-type: none"><li>◦ → Affichage succinct (méthode, nb modalités, k)</li></ul></li><li>• <b>\$summary(print_output=TRUE)</b><ul style="list-style-type: none"><li>◦ → Statistiques détaillées :<ul style="list-style-type: none"><li>▪ Basic stats (nb vars, nb modalités, k)</li><li>▪ ACM stats (VP brutes/corrigées, % variance)</li><li>▪ Cluster stats (tailles, composition)</li></ul></li></ul></li><li>• <b>\$predict(new_data)</b><ul style="list-style-type: none"><li>◦ → Affecte nouvelles observations</li><li>◦ → Retourne cluster + distances</li></ul></li></ul>	<ul style="list-style-type: none"><li>• <b>\$illustrative(X_illust_quali)</b><ul style="list-style-type: none"><li>◦ → Projette modalités illustratives qualitatives</li><li>◦ → Affichage table croisée</li></ul></li><li>• <b>\$illustrative_numeric(X_illust_quant)</b> [ACM only]<ul style="list-style-type: none"><li>◦ → Cercle des corrélations variables quanti</li><li>◦ → Interprétation axes factoriels</li></ul></li><li>• <b>\$plot_dendrogram(k=NULL)</b><ul style="list-style-type: none"><li>◦ → Dendrogramme coloré par cluster</li></ul></li><li>• <b>\$plot_factorial_map(dims=c(1,2), k=NULL)</b> [ACM only]<ul style="list-style-type: none"><li>◦ → Carte factorielle modalités colorées</li></ul></li><li>• <b>\$plot_scree(cumulative=FALSE)</b> [ACM only]<ul style="list-style-type: none"><li>◦ → Éboulis valeurs propres (brutes/corrigées)</li></ul></li><li>• <b>\$plot_contrib(dim, top=10)</b> [ACM only]<ul style="list-style-type: none"><li>◦ → Contributions modalités aux axes</li></ul></li><li>• <b>\$elbow(k_max=10, plot=TRUE)</b><ul style="list-style-type: none"><li>◦ → Sélection k optimal (gain marginal max)</li></ul></li><li>• <b>\$cluster_table()</b><ul style="list-style-type: none"><li>◦ → Table croisée variables × clusters</li></ul></li><li>• <b>\$get_clusters_table()</b><ul style="list-style-type: none"><li>◦ → DataFrame (modalité, cluster)</li></ul></li></ul>

# CAH : INTERPRÉTATION

## Métriques

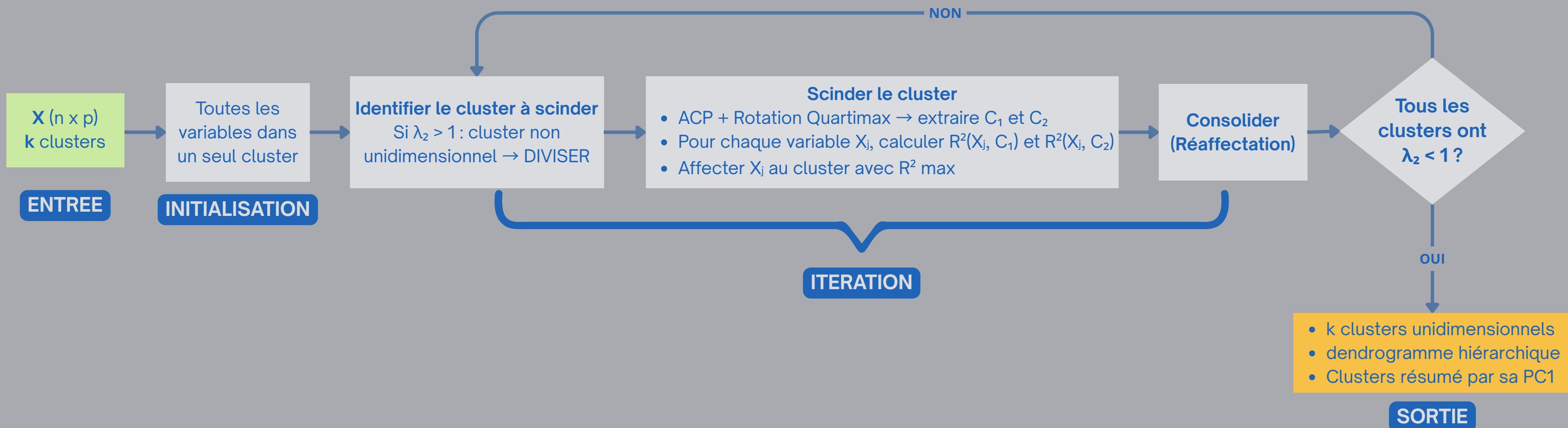
- **Print** (method : Dice/acm, nb var, nb mod, nb clusters(k))
- **Summary**
  - **Basic stats** (nb\_obs, nb\_var, nb\_mod, k)
  - **ACM Stats :**
    - Axe
    - Valeurs Propres brutes
    - Valeurs Propres corrigées Benzécri
    - Valeurs Propres corrigées Greenace
    - % variance expliquée
    - % variance cumul
- **Cluster table**
  - Table d'affectations

## Visualisations

- **Dendrogramme**
  - Variables projetées sur les 2 premiers axes factoriels d'une ACP globale
  - Distance :  $1 - \text{Dice}^2$  ou euclidienne (ACM)
  - linkage : ward.D2
- **Carte factorielle**
  - Modalités projetées sur axes 1-2 de l'ACM
- **Elbow**
  - Inertie totale en fonction de k
  - k optimal → Gain marginal
- **Contributions (ACM) :**
  - Barplot op N modalités contribuant le plus à un axe

# VARCLUS: PRINCIPE

- Clustering hiérarchique divisif de variables
- Mesure = Corrélation
- Objectif : clusters unidimensionnels ( $\lambda_2 < 1$ )
- Utilise Hmisc::varclus()



# VARCLUS : CLASSE R6

## ATTRIBUTS : 2 TYPES

### PARAMETRES ALGO :

- similarity
- n\_clusters

### DONNÉES :

- data
- model
- clusters
- dendo

- : "pearson" ou "spearman"
- : Nb clusters (null= auto)
- : Matrice numérique (n x p)
- : Objet Hmisc::varclus()
- : data.frame (variable, cluster)
- : Objet dendrogramme

## Résultats (calculés à la volée) :

- PC1 par cluster
- Valeurs propres  $\lambda_1$  par cluster
- Proportion variance expliquée
- $R^2$  own/next par variable

## METHODES PRINCIPALES

### OBLIGATOIRES :

- **\$new(similarity, n\_clusters)**
  - → Constructeur avec mesure similarité
- **\$fit(X)**
  - → Appelle Hmisc::varclus()
  - → Coupe dendrogramme si n\_clusters spécifié
  - → Sinon, utilise critère  $\lambda_2 \geq 1$  automatique
- **\$print()**
  - → Affichage succinct (nb vars, clusters,  $\lambda$ )
- **\$summary(print\_output=TRUE)**
  - → Statistiques détaillées par cluster
  - →  $R^2$  own/next/ratio pour chaque variable
- **\$predict(new\_var)**
  - → Affecte variable au cluster avec  $R^2$  max

### AUTRES METHODES :

- **\$illustrative(X\_illust, plot=TRUE)**
  - → Régression sur PC1 des clusters
  - → Visualisation ACP
- **\$get\_dendrogram**
  - → Retourne fonction de plot du dendrogramme
- **\$get\_heatmap()**
  - → Heatmap matrice similarité (corrélations)
- **\$plot\_elbow(k\_range)**
  - → Aide choix k optimal (coude similarité)
- **\$get\_clusters\_table()**
  - → DataFrame avec affectations triées

## METHODES PRIVEES

### private\$compute\_cluster\_pcs\_list()

- → Calcule PC1 de chaque cluster par ACP
- → Gère cluster à 1 variable (PC1 = variable normalisée)

### private\$compute\_cluster\_pcs()

- → Calcule statistiques résumées par cluster :
- Nb variables
- $\lambda_1$  (eigenvalue)
- Proportion variance expliquée par PC1

### private\$compute\_cluster\_R2()

- → Pour chaque variable :
  - $R^2_{own}$  :  $R^2$  avec PC1 de son cluster
  - $R^2_{next}$  :  $R^2$  max avec PC1 autres clusters
  - Ratio :  $(1-R^2_{own})/(1-R^2_{next})$
- → Ratio < 1 = bonne affectation

# VARCLUS : INTÉRPRÉTATION

## Métriques

- **Cluster Summary**
  - Nombre de variables dans chaque cluster
  - $\lambda_1$  (variance de PC1)
  - % de variance expliquée par PC1
- **R<sup>2</sup> Details**
  - Own\_Cluster: Corrélation<sup>2</sup> avec PC1 de son propre cluster
  - Next\_Cluster: R<sup>2</sup> maximum avec la PC1 du cluster voisin le plus proche
  - 1-R<sup>2</sup>\_Ratio: Ratio de qualité d'affectation
- **Cluster Quality**
  - mean\_R2\_own: Moyenne des R<sup>2</sup> dans le cluster

## Visualisations

- **Dendrogramme**
- **Heatmap de corrélation**
- **Elbow plot**
  - Sélection automatique du nombre k optimal
  - $S_k$  = similarité intra-cluster moyenne
  - k optimal = max distance à la droite

# APPLICATION SHINY



inst/shiny/

```
    └── app.R # Point d'entrée
    └── ui.R # Interface utilisateur globale
    └── server.R # Logique serveur principale
    └── modules/ # Modules Shiny (un par algorithme)
        └── kmeans_module.R # UI + Server K-Means
        └── varclus_module.R # UI + Server VarClus
        └── acm_cah_module.R # UI + Server ACM-CAH
    └── www/ # Assets statiques (CSS, images)
```

## ARCHITECTURE MODULAIRE

- **Indépendances** entre les algorithmes
- Facilite la **maintenance** et le déboggage
- **Scalabilité** (ajouter un algorithme)
- N'utilise pas clusterEngine
  - **Instancie** directement les classes

## FONCTIONNEMENT

- Choix algo puis "**Run Clustering**"
  - **eventReactive** instancie la classe choisie
  - Affichage conditionnel
- Un **algorithme** sélectionné=une **instance R6**  
créé= un **module** actif= une **UI** affichée

## STRUCTURE D'UN MODULE

- Chaque module a une **ui** et un **server**
- **UI** → 4 tabs:
    - Summary
    - Visualizations
    - Illustrative Variables
    - Detailed Stats
  - **Server** : Logique server de la classe concerné

# BILAN ET CONLUSION



## RÉALISÉ

→ 3 **algorithmes** complets :

- K-means (Var. Quantitatives)
- Varclus Hiérarchique (Var. Quantitatives)
- DICE/ACM + CAH (Var Qualitatives)

→ **Package R** :

- Architecture R6
- Documentation
- Tests et datasets inclus

→ Application **Shiny** interactive :

- Architecture modulaire
- Sélection k optimal



## AMÉLIORATIONS

→ **Consolidation et fiabilisation**

- Améliorer la robustesse
- Améliorer la flexibilité
- Confirmer la justesse des algos
- Corriger si besoin

• **Enrichissements** :

- Ajout d'autres algorithmes
- Gestion des variables mixtes
- Ajout de visualisations
- Ajout d'outils d'interprétation

# PASSONS À LA SUITE!

