Institutional Repository - Research Portal
Dépôt Institutionnel - Portail de la Recherche
researchportal.unamur.be
University of Namur

THESIS / THÈSE

MASTER EN SCIENCES MATHÉMATIQUES

Etude de la stabilité de méthodes de détermination du nombre de classes en classification par la méthode du bootstrap

Gobert, Laurence

Award date:
1998

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 04. May. 2026

FACULTÉS UNIVERSITAIRES N.D. DE LA PAIX
NAMUR
FACULTÉS DES SCIENCES

Etude de la stabilité de méthodes de détermination du nombre de classes en classification par la méthode du Bootstrap

Mémoire présenté pour l'obtention du grade
de Licencié en Sciences
mathématiques
par

Promoteur: André HARDY

Laurence GOBERT

Année académique 1997-1998

Je tiens à remercier, tout particulièrement, Mr A. Hardy pour son aide et sa disponibilité apportées tout au long de ce travail.
Il me faut également remercier Mr V. Bertholet pour ses nombreux conseils en informatique.
Mes pensées se tournent également vers ma famille et mon entourage pour leurs encouragements et leur patience durant ces quatre années.

A mon mari et mon petit garçon, qu'ils me pardonnent mon peu de patience ces derniers mois.

Résumé

Le but de la classification automatique est de décomposer un ensemble de données de n objets décrits par un ensemble de p caractéristiques, en un nombre relativement restreint de classes d'objets "semblables".
Un des problèmes fondamentaux de la classification automatique est la détermination du nombre de classes et par conséquent le choix de la règle d'arrêt à utiliser.
De manière à faciliter ce choix, nous avons établi dans ce mémoire des critères basés sur la stabilité des règles d'arrêt.

Abstract

The aim of classification is to decompose a given set of n objects described by a set of p features in a relatively small number of clusters of similar objects.
One of fundamentals problems in classification is the determination of the number of true clusters and consequently the choice of stopping rule to use.
So as to make this choice easier, we have tried, in this work, to establish some criteria based on the stability of these stopping rules.

Table des matières

1 La classification **4**

1.1 Qu'est-ce que la classification automatique? 4

1.2 Problèmes liés aux méthodes de classification 6

1.2.1 Choix de la mesure de similitude 6

1.2.2 Choix de la méthode de classification 6

1.2.3 Détermination du nombre de classe 6

1.3 Mesure de proximité utilisée et position du problème 7

1.3.1 Introduction et notations 7

1.3.2 Mesures de proximité entre deux individus 8

1.3.3 Position du problème 9

2 Méthodes de classification et de détermination du nombre de classes **10**

2.1 Classement des méthodes de classification 10

2.2 Présentation des méthodes de classification utilisées 13

2.2.1 La méthode du voisin le plus proche ou du lien simple . . . 13

2.2.2 La méthode du voisin le plus éloigné ou du lien complet . . 16

2.2.3 La méthode de la moyenne ou du lien moyen (group average link) 19

2.2.4 La méthode de WARD 20

2.3 Calcul pratique des distances pour les méthodes utilisées 21

2.4 Méthodes de détermination du nombre de classes 22

2.4.1 Etude comparative de Milligan et Cooper 22

2.4.2 Présentation de quatre méthodes de Milligan et Cooper . . 25

2.4.3 Présentation du programme utilisé 36

2.5 Méthodes de détermination du nombre de classes : approche par la technique du bootstrap 37

2.5.1 Méthode de Jain et Moreau 37

3 Etude de la stabilité de quelques méthodes **41**

3.1 Introduction 41

3.2 Méthodologie employée 42

3.2.1 Choix des méthodes et techniques d'échantillonnage 42

3.2.2 Présentation des programmes utilisés 44
3.3 Critères d'étude de la stabilité 45
3.3.1 Mesures d'association traditionnelles 48
3.3.2 Le coefficient λ 50
3.4 Applications 53
3.4.1 Aux données de Ruspini : 4 classes séparées 53
3.4.2 Aux données ALLON : 3 classes allongées 65
3.4.3 Aux données SOURIRE : 2 classes 75

Annexes **87**

A Partitions et hiérarchies **87**
A.1 Partitions 87
A.2 Familles de partitions 87
A.3 Hiérarchies de parties 88

B Jeux de données **89**
B.1 Données de Ruspini 89
B.2 Données ALLON 90
B.3 Données SOURIRE 91

C Programmes utilisés **92**
C.1 Calcul des indices 92
C.2 Formation des échantillons 100
C.3 Renumération des échantillons 101
C.4 Calcul du coefficient λ 102

D Calculs du coefficient λ **105**
D.1 Pour les données de Ruspini 105
D.2 Pour les données ALLON 106
D.3 Pour les données SOURIRE 107

Introduction

La littérature en matière de classification automatique a longtemps été orientée vers le développement de nouvelles méthodes de classification. L'utilisateur est alors confronté à toute une série de procédures avec pratiquement pas de conseils pour faire le choix entre elles.

Par conséquent, il est important de trouver des moyens pour faciliter l'interprétation des résultats fournis par l'une ou l'autre méthode de classification proposée.
En effet, les problèmes liés aux méthodes de classification sont nombreux. On peut citer le problème du choix de la mesure de dissimilarité ou de similarité entre individus, le problème de la stabilité et de la validité des classes obtenues, et celui du nombre de classes à considérer comme significatives.

Nous nous intéresserons dans ce mémoire, au problème de la détermination du nombre de classe et notre objectif est de présenter des procédures qui permettront d'évaluer les performances des règles d'arrêt.

Dans le premier chapitre nous parlerons de la classification automatique et des différents problèmes liés à celle-ci.
Le deuxième chapitre sera consacré à l'explication de différentes méthodes existantes.
Enfin, le troisième chapitre présentera la méthodologie d'approche que l'on va utiliser pour étudier la stabilité des différentes méthodes de détermination du nombre de classes.

Chapitre 1

La classification

1.1 Qu'est-ce que la classification automatique?

La classification[1], au sens large du terme, est l'une des plus anciennes activités entreprises par l'homme. D'une façon tout à fait générale, la classification désigne le processus qui consiste à désigner par un même nom des objets semblables. Le développement du langage, par exemple, est une activité de classification puisqu'il consiste à désigner du même terme une classe d'objets, d'événements ou de personnes qui ont un certain nombre de caractéristiques en commun.
La classification a joué un rôle important dans le développement de plusieurs sciences, notamment la chimie avec la classification périodique des éléments qui a contribué énormément à la compréhension de la structure de la matière, mais aussi en biologie (zoologie et botanique), archéologie, médecine, etc.
En d'autres mots, la classification désigne le processus de répartir un ensemble d'individus caractérisés par un certain nombre de variables en groupes ou classes, de telle sorte que les individus d'une même classe soient aussi semblables que possible, et les individus de classes différentes aussi dissemblables que possible, vis-à-vis d'un certain critère. Les groupes ou classes ne sont pas connus a priori, ce qui distingue, ainsi, la classification de l'analyse discriminante qui consiste à affecter des individus nouveaux à des classes préalablement définies.
La classification permet ainsi de simplifier une réalité complexe par la constitution de groupes d'individus "semblables".
La classification permet aussi , selon l'appartenance d'un individu à une classe ou à une autre (par exemple la classe des mammifères), d'en préciser les caractéristiques (par exemple, pour les mammifères, ils allaitent leurs petits), le comportement (par exemple, pour une campagne électorale, une campagne publicitaire, ...), ...

1. Aussi appelée analyse typologique, classification automatique, classification non-supervisée, taxonomie, ...

Remarquons que la classification est une méthode descriptive où l'utilisateur fournit le minimum d'hypothèses sur les données et où toutes les variables jouent le même rôle (il peut y avoir éventuellement des poids sur les variables, mais nous n'étudions pas ce cas).
Enfin, le processus de classification comporte trois grandes étapes :

- l'obtention des données
- la constitution des groupes
- l'analyse des résultats.

Remarque :

Par la suite, nous dirons qu'une méthode donne une bonne classification si cette méthode retrouve les classes naturelles.
Par classes naturelles, nous entendons les classes qui sont repérées par l'oeil lorsque les individus sont représentés par des points dans un espace à deux ou trois dimensions (nous pouvons généraliser à p dimensions, mais cela n'est pas aussi facilement réalisable).

1.2 Problèmes liés aux méthodes de classification

1.2.1 Choix de la mesure de similitude

Le problème du choix de l'indice de similitude i.e. du paramètre utilisé pour quantifier la similarité ou la dissimilarité entre individus se pose. Ces mesures ou indices sont nombreux dans la littérature et se distinguent par leur manière de définir le concept de ressemblance ou de dissemblance, par leur propriétés théoriques et par le type de données auxquelles elles sont les plus appropriées.

1.2.2 Choix de la méthode de classification

Une fois que la mesure de similitude est définie, reste alors à décider de la méthode de classification à utiliser.
Mais aucune méthode ne "marche" à chaque fois sur tous les exemples. Donc, il n'y a pas de méthode miracle, qui donnerait à chaque fois un meilleur résultat que toutes les autres. On préconise donc, en général, de travailler avec plusieurs méthodes.

1.2.3 Détermination du nombre de classe

C'est le but de ce mémoire. Nous allons essayer de donner des critères sur lesquels nous pourrons nous baser pour choisir une "bonne" méthode de détermination du nombre de classes.

1.3 Mesure de proximité utilisée et position du problème

1.3.1 Introduction et notations

Soit un ensemble de **n individus** :

$$E = \{x_1, x_2, ..., x_n\}$$

Chaque individu est caractérisé par un ensemble de **p variables** :

$$V = \{v_1, v_2, ..., v_p\}$$

Cette situation peut se représenter par une matrice $n \times p$, que l'on appellera matrice des données :

$$\mathcal{M} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

Pour pouvoir visualiser ces objets, on a l'habitude de représenter chacun d'entre eux par un point dans un espace à p dimensions, chacune de ces dimensions représentant une des p variables descriptives. Remarquons que pour cela, il faut bien entendu que les variables utilisées soient des variables quantitatives (ce que nous supposerons dans ce mémoire).

Chaque point i représentant l'objet x_i a alors pour coordonnées :

$$(x_{i1}, x_{i2}, ..., x_{ip})$$

On se rappelle alors que le but est de trouver des classes de façon à ce que les membres de chaque classe aient en commun certaines caractéristiques qui les distinguent des membres des autres classes (on peut alors essayer de coller une "étiquette" sur chaque classe). Mais comment faire?
C'est ce que nous verrons dans les paragraphes suivants.

N.B. : pour p=2 ou p=3, cela peut se faire de façon visuelle.

1.3.2 Mesures de proximité entre deux individus

A chaque paire d'objets x_i et x_j, on va associer un nombre que l'on appellera indice de similarité ou de dissimilarité selon le cas, et qui mesure la proximité entre deux objets (pour pouvoir voir à quel point ils sont semblables ou dissemblables). Dans ce mémoire, la proximité entre deux objets sera mesurée par la distance euclidienne non pondérée :

$$d_{ij}^2 = \sum_{k=1}^{p} (x_{ik} - x_{jk})^2$$

C'est un indice de dissimilarité car, vu la représentation adoptée au paragraphe précédent, plus la distance entre deux objets est grande, plus ils sont dissemblables.
On peut ainsi définir une matrice de proximité :

$$\mathcal{D} = \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \ldots & d_{nn} \end{pmatrix}$$

Remarques :

- Cette distance dépend de l'unité de mesure et de la variance de chaque variable.

- Lorsque les unités de mesure diffèrent d'une variable à l'autre, la variable ayant la plus forte variance prendra une importance prépondérante dans la distance entre les objets. La distance euclidienne est donc biaisée en direction des variables qui ont la plus grande dispersion. Afin de pallier à ces difficultés, on procède couramment à la standardisation des variables en soustrayant à chaque mesure la moyenne de la variable et en divisant par l'écart-type. En conséquence, les distances entre objets sont elles-mêmes mesurées en écarts-types. De plus, la standardisation des variables ne préserve pas l'ordre des distances obtenu sur les données brutes. Remarquons enfin que la standardisation peut entraîner une perte d'informations pas nécessairement bénéfique à l'analyse. Il faut alors bien réfléchir pour voir si la standardisation est (ou était si on analyse les résultats) pertinente.

- On peut aussi choisir d'accorder un poids différent à chaque variable, indépendamment de la standardisation (par exemple, selon la qualité ou la fiabilité des mesures effectuées) :

$$d_{ij}^2 = \sum_{k=1}^{p} p_k (x_{ik} - x_{jk})^2$$

1.3.3 Position du problème

Reprenons notre ensemble d'individus :

$$E = \{x_1, x_2, ..., x_n\}$$

On recherche une partition[2] de E en k classes (k fixé)

$$P = \{C_1, C_2, ..., C_k\}$$

A chaque partition P, on peut associer un critère de classification :

$$\begin{array}{rccc} W: & P_k & \longrightarrow & R^+ \\ & P & \rightsquigarrow & W(P,k) \end{array}$$

Le problème est alors :

Trouver la partition "optimale",

$P^* = \{C_1^*, C_2^*, ..., C_k^*\}$ **telle que** $W(P^*,k) = \min_{P \in P_k} W(P,k)$

ou

$W(P^*,k) = \max_{P \in P_k} W(P,k)$

Ce problème peut avoir différentes facettes :

- Soit k est connu et il faut trouver la partition qui donne une valeur optimale de $W(P,k)$,
- Soit k est inconnu et, dans ce cas,il faut déterminer k pour que le critère soit optimal. **C'est le cas auquel nous nous intéresserons.**

Remarque :

On peut calculer le nombre de partitions possibles : il est énorme. En général, les diverses méthodes de classification essaient de ne s'intéresser qu'aux "meilleures" partitions possibles.

2. Définition en annexe .

Chapitre 2

Méthodes de classification et de détermination du nombre de classes

2.1 Classement des méthodes de classification

Tout d'abord, il faut savoir qu'il existe deux grandes classes de méthodes :

1. *Les méthodes monothétiques.*
   Leur objectif est la recherche d'une hiérarchie[1] de partitions, construite à partir de la matrice des données, par une suite de divisions en deux classes ne tenant compte que d'une seule variable à la fois. A chaque étape, la division peut se faire suivant une variable différente.
   Dans ces méthodes, une classe d'objets est définie par la possession en commun d'un attribut. Par exemple, on peut désirer classer ensemble tous les individus qui ont répondu de la même façon à l'une des questions d'une enquête. Le problème se pose alors de sélectionner la question la plus discriminante ou la plus sélective, c'est-à-dire celle qui apporte le plus d'informations sur l'ensemble des réponses.

2. *Les méthodes polythétiques.*
   Les méthodes polythétiques sont celles auxquelles on va s'intéresser dans ce mémoire. Elles tiennent compte simultanément de l'ensemble des variables décrivant les objets. Ainsi, deux objets pourront appartenir à la même classe sans posséder un seul caractère commun pourvu qu'ils se ressemblent suffisamment du point de vue de l'indice de similarité choisi pour mesurer leur ressemblance. L'information de base manipulée par les méthodes polythétiques est la matrice de proximité et non la matrice des données.

1. Définition en annexe.

Ces méthodes peuvent être conçues, ainsi que le remarquent Jardine et Sibson ([22]), comme transformant la matrice des proximités en une nouvelle matrice dans laquelle les groupes d'objets sont plus apparents que dans la matrice des proximités initiale. La transformation remplace les dissimilarités initiales par de nouvelles dissimilarités vérifiant des propriétés plus fortes.

Les méthodes hiérarchiques

Leur objectif est la recherche d'une famille[2] de partitions telle que les groupements ou les divisions successifs des objets forment une hiérarchie.
On a alors deux cas possibles :

1. *Algorithmes agglomératifs.*
   On part de n classes constituées chacune d'un individu :

$$C_1 = \{x_1\}, C_2 = \{x_2\}, ..., C_n = \{x_n\}$$

   A chaque étape, on regroupe les deux classes les "plus proches" (pour avoir dans une même classe des objets qui se ressemblent).

   On aura donc étape 0 : n classes
   1 : n-1 classes
   ⋮
   n-1 : 1 classe$\equiv E = \{x_1, x_2, ..., x_n\}$

   Pour chaque définition différente de la distance entre deux classes, on aura une méthode différente (à chaque étape, les deux classes fusionnées seront différentes).

2. *Algorithmes divisifs.*
   On part d'une classe :

$$E = \{x_1, x_2, ..., x_n\}$$

   A chaque étape, on va choisir parmi toutes les divisions possibles d'une classes en deux, celle dont la distance entre les classes obtenues par cette division est maximale.

   On aura donc étape 0 : 1 classe$\equiv E = \{x_1, x_2, ..., x_n\}$
   1 : 2 classes
   ⋮
   n-1 : n classes

2. Définition en annexe.

A nouveau, pour chaque définition différente de la distance entre deux classes, on aura une méthode différente.

Enfin, il existe différentes classes de méthodes hiérarchiques :

1. Les méthodes hiérarchiques ordinales.
   Elles n'utilisent pas d'autre information que le classement de paires d'objets par ordre de proximité.

2. Les méthodes hiérarchiques non-ordinales.
   Ces méthodes, contrairement aux précédentes, utilisent les valeurs numériques des dissimilarités entre paires d'objets.

Remarques :

- La hiérarchie peut dépendre de l'ordre d'introduction des données.
- Le problème avec les méthode hiérarchiques est qu'une fois qu'un individu est mis dans un groupe, il y restera jusqu'à la fin. En d'autres mots, il n'est pas possible de corriger une mauvaise partition.
- Il existe aussi des méthodes hiérarchiques générant une hiérarchie de recouvrements (ordonnés par la relation de finesse) au lieu d'une hiérarchie de partitions.

Les méthodes de réallocation

Ces méthodes ont pour but de construire une partition unique des objets en k classes où le nombre k est soit spécifié a priori, soit déterminé par l'algorithme. L'idée centrale de ces méthodes est de choisir une partition initiale des objets et de déplacer les objets d'un groupe à l'autre pour obtenir une meilleure partition. Les nombreux algorithmes existants diffèrent par le choix de la partition initiale, par la définition qu'ils donnent à une "meilleure partition" et par la méthode utilisée pour améliorer la partition. Ils partent donc d'une partition initiale, puis génèrent un ensemble de partitions successives permettant d'améliorer la valeur d'une fonction objective (d'un critère) jusqu'à ce qu'un minimum soit atteint. Ces méthodes sont en général simples et économiques.

Les méthodes de recherche de densité

Comme on considère les objets comme des points dans un espace à p dimensions, il est naturel de penser aux groupes d'objets en terme de régions de l'espace où la densité de points est élevée, séparées par des régions où elle ne l'est pas. De façon générale, on recherche des régions de forte densité pour constituer des groupes.

2.2 Présentation des méthodes de classification utilisées

Elles sont au nombre de quatre. Ce sont toutes des méthodes hiérarchiques. Pour ces méthodes, il existe à la fois des algorithmes ascendants et descendants, mais nous n'utiliserons que les algorithmes ascendants. Rappelons que pour les méthodes hiérarchiques ascendantes, nous partons d'une partition où chaque élément forme une classe, et à chaque étape, nous fusionnons les deux classes les plus proches.
La seule chose qui distinguera donc ces quatre méthodes sera la façon de mesurer la distance entre deux groupes d'objets.

2.2.1 La méthode du voisin le plus proche ou du lien simple

1. Description

   Cette méthode mesure la distance entre deux classes C_i et C_j d'une partition par la plus petite distance séparant un point d'une classe et un point de l'autre :

   $$d_{C_iC_j} = \min_{x \in C_i, y \in C_j} d(x, y)$$

   On appelle cette distance : la distance du saut minimum.

Exemple :

1.	2.	3.			
4.	5.	6.			
7.	8.	9.	.10	.11	.12
			.13	.14	.15
			.16	.17	.18

Si $C_1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
et $C_2 = \{10, 11, 12, 13, 14, 15, 16, 17, 18\}$
Alors $d_{C_1C_2} = d(9, 10)$

2. Exemple: $E = \{1, 2, 3, 4\}$

Voici la matrice des distances entre ces quatre objets :

$$D_0 = \begin{pmatrix} 0 & 5 & 9 & 8 \\ 5 & 0 & 4 & 5 \\ 9 & 4 & 0 & 3 \\ 8 & 5 & 3 & 0 \end{pmatrix}$$

Etape 0 : $P_0 = \{\{1\}, \{2\}, \{3\}, \{4\}\}$

Etape 1 : Calculons les distances entre chaque paire de classes. Comme toutes les classes sont constituées d'un point, la distance les séparant est la distance entre ces points !
On a $d_{12} = 5$,$d_{13} = 9$,$d_{14} = 8$,$d_{23} = 4$,$d_{24} = 5$,$d_{34} = 3$
Les deux points les plus proches sont 3 et 4. Nous allons donc les regrouper :

$$P_1 = \{\{1\}, \{2\}, \{3, 4\}\}$$

Etape 2 : Calculons les distances entre chaque paire de classes.

$$\begin{aligned} d_{12} &= 5 \\ d_{1\{3,4\}} &= \min\{d_{13}, d_{14}\} = \min\{9, 8\} = 8 \\ d_{2\{3,4\}} &= \min\{d_{23}, d_{24}\} = \min\{4, 5\} = 4 \end{aligned}$$

On regroupe les deux classes les plus proches : $\{2\}$ et $\{3, 4\}$

$$P_2 = \{\{1\}, \{2, 3, 4\}\}$$

Etape 3 : Le seul groupe possible est : $\{1\}$ et $\{2, 3, 4\}$

$$P_3 = \{\{1, 2, 3, 4\}\}$$

3. Propriétés :

   (a) La hiérarchie peut dépendre de l'ordre de lecture des données (par exemple, si la distance entre deux objets est la même que la distance entre deux autres objets, l'algorithme choisira comme distance minimale la première qu'il a calculée).

   (b) La méthode est peu robuste car en perturbant un peu les données, on peut modifier beaucoup la hiérarchie obtenue. Cela est dû au fait que l'on mesure la distance entre deux groupes par la plus petite distance les séparant.

   (c) A chaque fusion, les objets non encore classés tendent à être incorporés aux groupes existants plutôt qu'à former de nouveaux groupes([17]). En conséquence, la méthode donne des résultats peu satisfaisants lorsque des objets intermédiaires sont présents entre deux groupes ou lorsque les groupes ne sont pas nettement séparés (propriété de chaînage).

2.2.2 La méthode du voisin le plus éloigné ou du lien complet

1. Description

Cette méthode mesure la distance entre deux classes C_i et C_j d'une partition par la plus grande distance séparant un point d'une classe avec un point de l'autre :

$$d_{C_iC_j} = \max_{x \in C_i, y \in C_j} d(x, y)$$

On appelle cette distance : la distance du saut maximum.

Exemple :

1.	2.	3.			
4.	5.	6.			
7.	8.	9.	.10	.11	.12
			.13	.14	.15
			.16	.17	.18

Si $C_1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
et $C_2 = \{10, 11, 12, 13, 14, 15, 16, 17, 18\}$
Alors $d_{C_1C_2} = d(1, 18)$

Attention :

Pour calculer la distance entre deux classes, on prend la distance **maximale** entre deux éléments (un de chaque classe), mais ensuite, on regroupe à nouveau les deux classes dont la distance est **minimale**.

2. Exemple: $E = \{1, 2, 3, 4\}$

Voici la matrice des distances entre ces quatre objets:

$$\mathcal{D}_0 = \begin{pmatrix} 0 & 5 & 9 & 8 \\ 5 & 0 & 4 & 5 \\ 9 & 4 & 0 & 3 \\ 8 & 5 & 3 & 0 \end{pmatrix}$$

Etape 0: $P_0 = \{\{1\}, \{2\}, \{3\}, \{3\}, \{4\}\}$

Etape 1: De nouveau, calculons les distances entre chaque paire de classes. Comme toutes les classes sont constituées d'un point, la distance les séparant est la distance entre ces points!
On a $d_{12} = 5$,$d_{13} = 9$,$d_{14} = 8$,$d_{23} = 4$,$d_{24} = 5$,$d_{34} = 3$.
Les deux points les plus proches sont 3 et 4,donc on les regroupe:

$$P_1 = \{\{1\}, \{2\}, \{3, 4\}\}$$

Etape 2: Calculons les distances entre chaque paire de classes:

$$\begin{aligned} d_{12} &= 5 \\ d_{1\{3,4\}} &= \max\{d_{13}, d_{14}\} = \max\{9, 8\} = 9 \\ d_{2\{3,4\}} &= \max\{d_{23}, d_{24}\} = \max\{4, 5\} = 5 \end{aligned}$$

On regroupe les deux classes les plus proches: $\{1\}$ et $\{2\}$ ou $\{2\}$ et $\{3, 4\}$
Si d_{12} est la première distance calculée, alors

$$P_2 = \{\{1, 2\}, \{3, 4\}\}$$

Si $d_{2\{3,4\}}$ est la première distance calculée, alors

$$P_2 = \{\{1\}, \{2, 3, 4\}\}$$

Etape 3: Le seul groupement possible dans chaque cas est $\{1, 2\}$ et $\{3, 4\}$ ou $\{1\}$ et $\{2, 3, 4\}$

$$P_3 = \{\{1, 2, 3, 4\}\}$$

3. Propriétés :

   (a) L'algorithme ascendant et l'algorithme descendant ne fournissent pas toujours la même hiérarchie.

   (b) La hiérarchie dépend de l'ordre de lecture des données.

   (c) La méthode est peu robuste.

   (d) La méthode a tendance à former des classes hypersphériques.

2.2.3 La méthode de la moyenne ou du lien moyen (group average link)

1. Description

   Cette méthode mesure la distance entre deux classes C_i et C_j comportant respectivement n_i et n_j objets, par la valeur moyenne des distances inter-classes :

$$d_{C_iC_j} = \sum_{x \in C_i, y \in C_j} \frac{d(x,y)}{(n_i + n_j)}$$

2. Propriétés :

   (a) Cette méthode ne fait pas intervenir les dissimilarités intra-classes. En d'autres termes, elle garantit que les deux classes fusionnées sont composées d'objets proches en moyenne mais elle ne garantit pas que les classes soient les plus "naturelles" possible.

   (b) Remarque : les groupes obtenus diffèrent peu, en général, de ceux obtenus par le critère du voisin le plus éloigné.

2.2.4 La méthode de WARD

1. Description

La distance entre deux classes C_i et C_j comportant respectivement n_i et n_j objets, est mesurée par la différence entre le moment centré d'ordre 2 des classes fusionnées et le moment centré d'ordre 2 de chacune des classes :

$$d^2_{C_iC_j} = M^2(C_i \cup C_j) - M^2(C_i) - M^2(C_j)$$

où $M^2(C_i)$ représente la somme des carrés des écarts des objets au centre de gravité de la classe, ou en d'autres mots, la moyenne des distances euclidiennes entre toutes les paires d'objets de la classe :

$$M^2(C_i) = \sum_{k=1}^{p} \sum_{l \in C_i} (x_{lk} - \bar{x}_k)^2 = \sum_{l,j \in C_i} \frac{d_{lj}}{n_i}$$

Il est facile de démontrer ([21]) que ce critère du moment centré d'ordre 2 se ramène à une pondération de la distance entre centres de gravité :

$$d_{C_iC_j} = \left(\frac{n_i n_j}{n_i + n_j}\right)^{\frac{1}{2}} \|g(C_i) - g(C_j)\|$$

où $g(C_i)$ et $g(C_j)$ sont les centres de gravité des classes C_i et C_j respectivement.

En fait, on fusionne les deux groupes qui conduisent à un accroissement minimum du critère des moindres carrés.

2. Propriétés :

   (a) Croissance monotone des distances à chaque fusion.

   (b) Cette méthode tend aussi à former des classes hypersphériques (cela est dû au fait qu'elle est basée sue le critère des moindres carrés).

2.3 Calcul pratique des distances pour les méthodes utilisées

En fait, les distances entre groupes pour les quatre méthodes utilisées satisfont une relation de récurrence ([12]).
Quand on veut calculer la distance entre un groupe C_k et un groupe C_{ij} formé par la fusion des groupes C_i et C_j (ce qui est le cas à chaque étape d'une méthode hiérarchique ascendante), la formule suivante est vérifiée :

$$d_{C_kC_{ij}} = \alpha_i d_{C_kC_i} + \alpha_j d_{C_kC_j} + \beta d_{C_iC_j} + \gamma(d_{C_kC_i} - d_{C_kC_j})$$

où α_i, α_j, β et γ sont des paramètres dont les valeurs changent selon la méthode utilisée.

1. Voisin le plus proche : $\alpha_i = \alpha_j = \frac{1}{2}$; $\beta = 0$; $\gamma = \frac{-1}{2}$
2. Voisin le plus éloigné : $\alpha_i = \alpha_j = \frac{1}{2}$; $\beta = 0$; $\gamma = \frac{1}{2}$
3. Méthode de la moyenne : $\alpha_i = \frac{n_i}{n_i+n_j}$; $\alpha_j = \frac{n_j}{n_i+n_j}$; $\beta = \gamma = 0$
4. Méthode de WARD : $\alpha_i = \frac{n_k+n_i}{n_k+n_i+n_j}$; $\alpha_j = \frac{n_k+n_j}{n_k+n_i+n_j}$; $\beta = \frac{-n_k}{n_k+n_i+n_j}$; $\gamma = 0$

La formule de récurrence ci-dessus a été utilisée pour la programmation des méthodes de classification utilisées car elle est très facile à implémenter.

2.4 Méthodes de détermination du nombre de classes

2.4.1 Etude comparative de Milligan et Cooper ([24])

En 1985, Milligan et Cooper ont fait une étude comparative de trente méthodes de détermination du nombre de classes. Pour cela, ils ont pris beaucoup d'ensembles de points générés de façon aléatoire. Chaque ensemble de points contenait 2, 3, 4 ou 5 groupes différents, ne se recouvrant pas, avec un total de cinquante points chacun et dans un espace à 2, 4 ou 6 dimensions.
Pour disposer d'une variété de solutions, les ensembles de points ont été analysés par quatre méthodes de classification (toutes hiérarchiques).

Ce travail a donc permis de classer les trente règles d'arrêt de la meilleure à la moins bonne. La meilleure étant celle qui a donné le plus souvent les résultats attendus. Evidemment, même si ce classement n'est valable que pour les exemples utilisés, il est fort peu probable qu'une méthode classée première après plusieurs centaines d'exemples (108 jeux de données* 4 méthodes) ne s'avère être "mauvaise". Donc, ce classement donne une indication sur les meilleures méthodes parmi les trente méthodes analysées et permet de les comparer. Il est également important de remarquer que le classement reflète les performances des méthodes sur des ensembles de données où la structure est assez "nette". Ce classement peut donc être altéré pour des structures de données plus complexes. De plus, le fait que les structures des différents groupes étaient assez marquées permet aussi d'affirmer que les méthodes qui donnaient des mauvais résultats dans ce travail n'avaient aucune chance de donner de bons résultats dans des exemples plus complexes.

Maintenant, décrivons ce travail d'un point de vue plus pratique.
Si nous définissons deux types d'erreur :

- le premier : dire qu'il y a k groupes présents dans un ensemble de points alors qu'en réalité, il y en a moins que k.
- le deuxième : dire qu'il y a k groupes dans un ensemble de points alors qu'en réalité, il y en a plus que k.

Le deuxième type d'erreur a été considéré comme plus "grave" car de l'information est perdue en fusionnant[3] deux groupes distincts.
De plus, les ensembles de points ont été générés par un processus très particulier (Milligan et Cooper, 1985) favorisant la création de classes "naturelles". Sur 432 solutions (108 ensembles de points*4 méthodes), 400 avaient une structure "nette".

3. On procède par fusions car on est dans le cas de méthodes agglomératives.

Remarquons encore que les méthodes de détermination du nombre de classes choisies étaient toutes indépendantes des méthodes de classification utilisées et qu'il est possible de les adapter pour être utilisées avec des méthodes de classification non-hiérarchiques.
Signalons aussi que toutes les méthodes où intervenaient la subjectivité humaine pour décider du nombre de classes (comme par exemple certaines méthodes graphiques) ont été évitées. Les seules méthodes qui ont été choisies étaient donc celles où les règles de décision sur le nombre de groupes présents étaient "automatiques".

Enfin, chaque méthode était adaptée pour donner les meilleurs résultats possibles. Par exemple, quand une méthode permettait de choisir soit la valeur maximale de l'indice, soit la différence maximale entre deux valeurs successives de l'indice, celle qui indiquait le bon nombre de groupes était choisie pour représenter le résultat. Ceci dit, dans notre travail, nous avons préféré choisir le maximum et dans le cas où nous avions deux maxima, nous avons choisi celui qui correspondait à la différence maximale.
De plus, pour les tests statistiques, les valeurs des niveaux α étaient parfois assez modérées pour donner des résultats optimaux. Notons également que seules les 25 dernières valeurs de l'indice (c'est-à-dire celles correspondant à 26 ou moins de 26 groupes) étaient prises en compte pour déterminer le nombre de classes de l'ensemble de points examiné. Cela est dû au fait que certains indices ont des problèmes quand il y a presque autant de groupes que de points dans l'ensemble de données.

Les résultats étaient quant à eux présentés sous la forme de tableaux à deux entrées. En voici un exemple pour la méthode de Calinski et Harabasz :

	Number of true clusters				
Calinski et Harabasz	2	3	4	5	Overall
2 or fewer	-	-	1	0	1
1 too fewer	-	12	6	0	18
correct level	96	95	97	102	390
2 too many	3	0	3	6	12
3 or more	5	1	0	0	6

L'entrée de la deuxième ligne de la troisième colonne indique par exemple que parmi les ensembles de points constitués de 4 groupes, la méthode de Calinski et Harabasz en a retrouvé un en moins, c'est-à-dire 3, à six reprises.

Outre le classement proposé, il est important de signaler que huit des dix meilleures méthodes présentaient leur plus mauvais résultat quand le nombre de groupes présents étaient 2. Ce cas semble être le plus difficile à traiter.

2.4.2 Présentation de quatre méthodes de Milligan et Cooper ([24])

Nous allons maintenant exposer quatre des meilleures méthodes de détermination du nombre de classes selon Milligan et Cooper ([24]).

La méthode Gamma ([19])

Description

Soit un ensemble de n objets : $\{o_1, o_2, ..., o_n\}$.
Notons $d(o_i, o_j)$ la distance euclidienne entre o_i et o_j .
Supposons que ces objets sont partitionnés en k classes.
Définissons

$$T_l(o_i, o_j) = \begin{cases} 0 & \text{si } o_i \text{ et } o_j \text{ sont dans la même classe} \\ 1 & \text{sinon} \end{cases}$$

De plus, si $T_l(o_i, o_j)$ vaut 0 (o_i et o_j sont dans la même classe), définissons aussi :

$$n_l(o_i, o_j) = \sharp\{\{o_r, o_t\} : T_l(o_r, o_t) = 1 \text{ et } d(o_r, o_t) < d(o_i, o_j)\}$$

En fait, $n_l(o_i, o_j)$ représente le nombre de couples d'objets n'appartenant pas à la même classe et qui sont plus proches entre eux que le sont o_i et o_j.

Nous pouvons maintenant définir l'indice α_l :

$$\alpha_l = \frac{\sum_{i<j} n_l(o_i, o_j)}{\max \sum_{i<j} n_l(o_i, o_j)}$$

où

- le maximum est pris sur toutes les partitions possibles en k classes en gardant le même nombre d'objets par classe.
- les sommes sont sur les paires d'objets $\{o_i, o_j\}$ qui appartiennent à la même classe (telles que $T_l(o_i, o_j) = 0$).

Or, $\sum_{i<j} n_l(o_i, o_j)$ représente la somme sur les paires d'objets appartenant à une même classe, du nombre de paires d'objets n'appartenant pas à une même classe et qui sont plus proches que la paire d'objets considérée dans la somme.

Donc, α_l **est la proportion de paires d'objets qui sont dans des "mauvais groupes".**

Et cela dans le sens où deux objets d'une même classe sont strictement plus proches que deux objets de classes différentes (cela correspond bien à nos définitions du premier chapitre).

Enfin, l'indice γ est défini par :

$$\boxed{\gamma = 1 - 2\alpha_l}$$

Remarquons que l'indice γ varie de -1 à $+1$.

En effet : Si aucun objet n'est mal placé, alors $\sum_{i<j} n_l(o_i, o_j) = 0$.
Par conséquent, $\alpha_l = 0$ et $\gamma = 1$.
Si tous les objets sont mal placés, alors $\alpha_l = 1$ et $\gamma = -1$.

Cela montre aussi que γ vaut 1 si et seulement si la partition est "parfaite" (c'est-à-dire que, par rapport à notre définition, tous les objets sont "bien placés").

Bien évidemment, **la méthode Gamma cherche la valeur maximale de l'indice, celle-ci devant être le plus proche possible de 1.**
Rappelons que dans notre cas, cette valeur était cherchée parmi toutes celles correspondant aux partitions de la hiérarchie de partitions obtenue par la méthode de classification utilisée.

Que vaut $\max \sum_{i<j} n_l(o_i, o_j)$?

Soit x le nombre de couples d'objets appartenant à une même classe.
y le nombre de couples d'objets appartenant à des classes différentes.

Combien peut-il y avoir au maximum d'objets "mal placés" ?

On doit comparer chaque paire d'objets appartenant à une même classe avec chaque paire d'objets appartenant à des classes différentes.
Donc, on fait $x * y$ comparaisons. Le nombre maximum de cas "défavorables" est alors $x * y$. **Mais**, de ce nombre, il faut retrancher tous les cas où il y a égalité, c'est-à-dire où des objets appartenant à la même classe sont à la même distance que des objets appartenant à des classes différentes. En effet, ces objets ne sont pas considérés comme mal placés car, dans la définition de $n_l(o_i, o_j)$, on a $d(o_r, o_t) < d(o_i, o_j)$.

En résumé, $\max \sum_{i<j} n_l(o_i, o_j) \equiv xy -$ "égalités".

La méthode de Duda et Hart ([7])

Description

Définissons

- X_i l'ensemble des points du i$^{\text{ième}}$ groupe
- m_i la moyenne des points du i$^{\text{ième}}$ groupe
- $J(k) = \sum_{i=1}^{k} \sum_{x \in X_i} \|x - m_i\|^2$

Il est évident que $J(k)$ diminue de façon monotone avec k car la somme des carrés des erreurs peut être réduite à chaque fois que k augmente en formant un nouveau groupe avec un seul objet. On peut aussi montrer que si il y a k_0 groupes bien séparés, on s'attend à ce que $J(k)$ diminue rapidement jusque $k = k_0$ et diminue beaucoup moins rapidement ensuite jusqu'à ce qu'il atteigne 0 quand $k = n$.

Maintenant, on voudrait pouvoir voir, grâce à une amélioration statistiquement significative de $J(k)$, que décrire l'ensemble de points avec $k + 1$ groupes est mieux adapté qu'avec k groupes.
Une façon formelle de faire est d'avancer l'hypothèse nulle qu'il y a exactement k groupes et de calculer la distribution d'échantillonnage pour $J(k+1)$ sous cette hypothèse. Cette distribution nous dirait alors à quel genre d'amélioration on doit s'attendre pour $J(k)$ quand une description de l'ensemble de points en k groupes est correcte. La règle de décision serait alors d'accepter l'hypothèse nulle si la valeur de $J(k + 1)$ tombe au-delà d'une limite correspondant à une probabilité raisonnable de "rejet à tort" de l'hypothèse nulle. Malheureusement, on ne peut généralement rien faire d'autre que d'estimer grossièrement la distribution de $J(k + 1)$. Les solutions obtenues ne sont alors pas au-dessus de tout soupçon et le problème statistique de tester la validité des groupes est encore irrésolu.

Duda et Hart ont alors essayé l'approximation du critère de la "somme des carrés des erreurs" qui suit.

Supposons que l'on ait un ensemble X de n points et que nous voulions décider si oui ou non on peut justifier l'hypothèse qu'ils forment plus qu'un groupe.

Avançons l'hypothèse nulle :
H_0 : les points viennent d'une population normale de moyenne μ et de matrice de covariance $\sigma^2 I$.

Si cette hypothèse était vraie, tout groupe trouvé parmi les points aurait été formé par chance, et toute diminution observée de la somme des carrés des erreurs dans la classification n'aurait aucune signification.
Considérons la somme des carrés des erreurs $J_e(1)$ comme une variable aléatoire, car elle dépend de l'ensemble particulier de points choisi.

$$J_e(1) = \sum_{x \in X} \|x - m\|^2$$

où m est la moyenne des n objets.

Duda et Hart affirment alors que, sous l'hypothèse nulle, la distribution de $J_e(1)$ est approximativement normale, de moyenne $nd\sigma^2$ et de variance $2nd\sigma^4$, où d est le nombre de dimensions des données.

Ils supposent ensuite que l'on peut diviser l'ensemble des objets en deux sous-ensembles X_1 et X_2 de façon à minimiser $J_e(2)$ où

$$J_e(2) = \sum_{i=1}^{2} \sum_{x \in X_i} \|x - m_i\|^2$$

où m_i est la moyenne des objets de X_i.

Ils expliquent alors que sous l'hypothèse nulle, cette partition n'est pas la meilleure mais il en résulte néanmoins une valeur de $J_e(2)$ plus petite que $J_e(1)$. Si on connaissait la distribution d'échantillonnage de $J_e(2)$, on pourrait alors déterminer à quel point $J_e(2)$ devrait être petit pour que l'on soit obligé d'abandonner l'hypothèse nulle d'un seul groupe.
Comme nous n'avons pas la solution "analytique" de la partition optimale, on ne peut pas trouver une solution exacte pour la distribution d'échantillonnage. Heureusement, Duda et Hart disent que l'on peut obtenir une bonne estimation en considérant la partition "sous-optimale" obtenue en prenant un hyperplan passant par la moyenne de l'échantillon. Pour n grand, on peut montrer que la somme des carrés des erreurs pour cette partition est approximativement normale, de moyenne $n(d - \frac{2}{\pi})\sigma^2$ et de variance $2n(d - \frac{8}{\pi^2})\sigma^4$. Ce résultat s'accorde bien avec le fait que $J_e(2)$ est plus petit que $J_e(1)$, puisque la moyenne pour $J_e(2)$ $(n(d - \frac{2}{\pi})\sigma^2)$ est plus petite que la moyenne pour $J_e(1)$ $(nd\sigma^2)$. Mais pour pouvoir être considérée comme significative, la réduction de la somme des carrés des erreurs doit être plus grande que cela.
Duda et Hart expliquent à ce moment que l'on peut obtenir une approximation de la valeur critique pour $J_e(2)$ en supposant que la valeur sous-optimale est presqu'optimale, en utilisant l'approximation normale pour la

distribution d'échantillonnage et en estimant σ^2 par :

$$\hat{\sigma}^2 = \frac{1}{nd} \sum_{x \in X} \|x - m\|^2 = \frac{1}{nd} J_e(1)$$

Le résultat final est alors :

Rejeter l'hypothèse nulle "au niveau p" si

$$\frac{J_e(2)}{J_e(1)} < 1 - \frac{2}{nd} - \alpha \sqrt{\frac{2(1 - \frac{8}{\pi^2 d})}{nd}}$$

où α est déterminé par

$$p = 100 \int_{\alpha}^{\infty} \frac{1}{\sqrt{2\pi}} e^{\frac{-1}{2u^2}} du$$

Dans notre cas, on va inverser la procédure. Rappelons-nous en effet que, à chaque étape, les méthodes hiérarchiques agglomératives regroupent les deux classes les plus proches. On utilise alors le test pour décider si oui ou non la fusion de deux groupes est justifiée.
Il est évident que ce test s'applique très bien aux hiérarchies de partitions obtenues par des méthodes hiérarchiques. Nous avons alors qu'à chaque étape, le test n'est appliqué qu'aux points des classes concernées par la fusion (ce sera le cas aussi pour la méthode de Beale).
En pratique, la méthode calcule à chaque étape (pour toute partition en k classes, en partant de k=n) :

$$\frac{-\frac{J_e(2)}{J_e(1)} + 1 - \frac{2}{\pi d}}{\sqrt{\frac{2(1 - \frac{8}{\pi^2 d})}{nd}}}$$

On a alors que **dès que cette expression dépasse la valeur choisie pour α (pour $k = k_0$), on rejette l'hypothèse nulle qui correspond à l'existence d'un seul groupe. C'est donc que la fusion des deux groupes concernés n'était plus justifiée[4]. On prend alors comme bon nombre de classes $k_0 + 1$.**

4. Chaque α correspond à un niveau p. En général, dans la pratique, on choisit alors α pour avoir les meilleurs résultats possibles.

Dans la littérature, plusieurs valeurs de α ont été proposées. En effet, Milligan et Cooper ont indiqué que 3.20 semblait être la valeur qui donnait les résultats optimaux ([24]). Tandis que Gordon utilisait quant à lui 4 comme valeur pour α ([16]), tout en signalant que devoir spécifier ainsi une valeur critique était un inconvénient de la méthode.
Dans nos applications, nous prendrons $\alpha = 3.20$.

La méthode de Beale ([3])

Notons W_1 la somme des carrés des distances à l'intérieur d'un ensemble de points
W_2 la somme des carrés des distances à l'intérieur des deux groupes obtenus en divisant l'ensemble initial en deux.

Le test proposé par Beale permet, tout comme celui de Duda et Hart, de voir si la fusion de deux groupes de points est justifiée ou pas.
Ce test implique la comparaison de

$$\frac{(\frac{W_1-W_2}{W_2})}{((\frac{n-1}{n-2})2^{\frac{2}{p}}-1)}$$

avec une distribution F de Fisher-Snedecor à p degrés de liberté au numérateur et $(n-2)p$ degrés de liberté au dénominateur.

Tout comme pour le test de Duda et Hart, si k_0 est la première valeur qui conduit à un rejet de l'hypothèse correspondant à la fusion de deux groupes de points, le test de Beale indique que le bon nombre de classes est k_0+1.

Dans notre cas, p vaut 2 et $(n-2)p$ est toujours plus grand que 120. Ce qui conduit à des valeurs de $F_{p,(n-2)p}$ de 5.30 pour un niveau de précision 0.005 et de 4.61 pour un niveau de précision de 0.01.

Dans nos applications, c'est le niveau 4.61 qui donne en général les meilleurs résultats, c'est pourquoi nous nous restreindrons à celui-là.

La méthode de Calinski et Harabasz ([4])

Notations

Supposons que nous ayons n objets classés en k groupes. Dans cette section, nous noterons :

- d_{ij} la distance euclidienne entre les objets i et j.
- $\bar{d}^2 = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{d_{ij}^2}{\frac{n(n-1)}{2}}$ la moyenne des d_{ij}^2.
- $\bar{d}_g^2 = \sum_{i=1}^{n_g} \sum_{\substack{j=1 \\ j \neq i}}^{n_g} \frac{d_{ij}^2}{\frac{n_g(n_g-1)}{2}}$ la moyenne des d_{ij}^2 dans le g$^{\text{ième}}$ groupe.
- $R = B + W$ où R est la matrice de dispersion totale
  B est la matrice de dispersion inter-groupes
  W est la matrice de dispersion intra-groupes.

  Les éléments des matrices R, W et B étant les suivants :

$$r_{jl} = \sum_{i=1}^{N} (x_{ij} - \bar{x}_j)(x_{il} - \bar{x}_l)$$
$$w_{jl} = \sum_{i=1}^{k} \sum_{i=1}^{n_c} (x_{ij} - \bar{x}_{jc})(x_{il} - \bar{x}_{lc})$$
$$b_{jl} = \sum_{i=1}^{k} n_c (\bar{x}_{jc} - \bar{x}_c)(\bar{x}_{lc} - \bar{x}_c)$$

  où k est le nombre de groupes
  n_c est l'effectif du groupe C
  N est l'effectif de la population
  p est le nombre de variables
  j, l sont les indices de variables, $1 \leq j \leq p$, $1 \leq l \leq p$
  $\bar{x}_j$ la moyenne générale de la variable j
  $\bar{x}_{lc}$ la moyenne de la variable l dans le groupe C.

De plus, nous utiliserons les abréviations suivantes :

- $BGSS$ qui désigne la "Between Group Sum of Squares"
- $WGSS$ qui désigne la "Within Group Sum of Squares"
- TSS qui désigne la "Total Sum of Squares"

Nous avons alors :

- $TraceW = WGSS = TraceR_1 + \cdots + TraceR_k$
  où $TraceR_g = n_g^{-1}(d_{12(g)}^2 + d_{13(g)}^2 + \cdots + d_{n_{g-1},n_g(g)}^2)$
- $TraceR = \frac{1}{n}(d_{12}^2 + d_{13}^2 + \cdots + d_{n-1,n}^2)$

Description

Le critère de rapport de variance (VRC) est :

$$VRC = \frac{\frac{BGSS}{k-1}}{\frac{WGSS}{n-k}}$$

Bien qu'il n'y ait aucune justification théorique en probabilité pour utiliser VRC, ce critère possède quelques propriétés mathématiques intéressantes.

1. $TSS = \frac{1}{2}(n-1)\bar{d}^2$
   En effet,

$$\begin{aligned} \frac{1}{2}(n-1)\bar{d}^2 &= \frac{\frac{1}{2}(n-1)(d_{12}^2 + d_{13}^2 + \cdots + d_{n-1,n}^2)}{\frac{n(n-1)}{2}} \\ &= \frac{1}{n}(d_{12}^2 + d_{13}^2 + \cdots + d_{n-1,n}^2) \end{aligned}$$

2. $WGSS = \frac{1}{2}((n_1-1)\bar{d}_1^2 + \cdots + (n_k-1)\bar{d}_k^2)$
   En effet,

   Comme $TraceW = TraceR_1 + \cdots + TraceR_k$, il suffit d'appliquer le résultat précédent pour chaque $TraceR_i$ ($\equiv TSS$ du groupe i).

3. $BGSS = \frac{1}{2}((k-1)\bar{d}^2 + (n-k)A_k)$

$$\text{où } A_k = \frac{1}{n-k}((n_1-1)(\bar{d}^2 - \bar{d}_1^2) + \cdots + (n_k-1)(\bar{d}^2 - \bar{d}_k^2))$$

   En effet,

   - $R = B + W$
     Par conséquent, $TraceB = TraceR - TraceW$ et
     $TraceB = \frac{1}{2}(n-1)\bar{d}^2 - [\frac{1}{2}((n_1-1)\bar{d}_1^2 + \cdots + (n_k-1)(\bar{d}^2 - \bar{d}_k^2)]$
   - Or $\frac{1}{2}((k-1)\bar{d}^2 + (n-k)A_k)$
     $= \frac{1}{2}[(k-1)\bar{d}^2 + \frac{n-k}{n-k}((n_1-1)(\bar{d}^2 - \bar{d}_1^2) + \cdots + (n_k-1)(\bar{d}^2 - \bar{d}_k^2))]$
     $= \frac{1}{2}[\bar{d}^2(n-1) - \bar{d}_1^2(n_1-1) - \cdots - \bar{d}_k^2(n_k-1)]$

     car $(k-1) + (n_1-1) + \cdots + (n_k-1) = (n-1)$

Maintenant, nous pouvons écrire :

$$\boxed{VRC = \frac{(\bar{d}^2 + \frac{n-k}{k-1} A_k)}{(\bar{d}^2 - A_k)}}$$

En effet,

- $VRC = \frac{\frac{BGSS}{k-1}}{\frac{WGSS}{n-k}}$

  Mais $\frac{BGSS}{k-1} = \frac{\frac{1}{2}((k-1)\bar{d}^2 + (n-k)A_k))}{k-1} = \frac{1}{2}(\bar{d}^2 + \frac{n-k}{k-1} A_k)$

  $\frac{WGSS}{n-k} = \frac{\frac{1}{2}((n_1-1)\bar{d}_1^2 + \cdots + (n_k-1)\bar{d}_k^2)}{n-k}$

  Or,

$$\begin{aligned}(\bar{d}^2 - A_k) &= \bar{d}^2 - \frac{1}{n-k}((n_1-1)(\bar{d}^2 - \bar{d}_1^2) + \cdots + (n_k-1)(\bar{d}^2 - \bar{d}_k^2)) \\ &= \bar{d}^2 - \frac{1}{n-k}\bar{d}^2((n_1-1)\bar{d}_1^2 + \cdots + (n_k-1)) \\ &\quad + \frac{1}{n-k}((n_1-1)\bar{d}_1^2 + \cdots + (n_k-1)\bar{d}_k^2) \\ &= \frac{1}{n-k}((n_1-1)\bar{d}_1^2 + \cdots + (n_k-1)\bar{d}_k^2)\end{aligned}$$

  et donc, $\frac{WGSS}{n-k} = \frac{1}{2}(\bar{d}^2 - A_k)$

- En remplaçant, nous obtenons :

$$VRC = \frac{\frac{1}{2}(\bar{d}^2 + \frac{n-k}{k-1} A_k}{\frac{1}{2}(\bar{d}^2 - A_k)} = \frac{(\bar{d}^2 + \frac{n-k}{k-1})}{(\bar{d}^2 - A_k)}$$

On peut alors montrer que ([4]):

- Quand toutes les paires de points sont à égale distance, A_k vaut 0 et VRC vaut 1 (évident en regardant la définition de A_k).
- Le critère du $WGSS$ minimum maximise A_k pour un k donné.

Mettons cela en rapport avec l'indice VRC. Réécrivons :

$$\frac{\frac{BGSS}{k-1}}{\frac{WGSS}{n-k}} = \frac{(1 + \frac{n-k}{k-1} a_k)}{(1 - a_k)}$$

où $a_k = \frac{A_k}{\bar{d}^2}$

Nous avons alors que ([4]):

◇ a_k varie entre 0 et 1. a_k vaut 0 si toutes les paires de points sont à égale distance et vaut 1 pour des groupements "idéaux" (où il n'y a pas de "variation" à l'intérieur des groupes[5]).
En effet :

- Il est évident que $a_k \geq 0$.

- Si tous les points sont à égale distance, on a $A_k = 0$
  car $\bar{d}^2 = \bar{d}_i^2,\ i = 1, \ldots, k$.
  Ce qui entraîne : $a_k = 0$.
- Si on a des groupements "idéaux" : $\bar{d}_i^2 = 0,\ i = 1, \ldots, k$.

$$\begin{aligned} \text{Or, } a_k &= \frac{A_k}{\bar{d}^2} = \frac{\frac{1}{n-k}((n_1-1)(\bar{d}^2-\bar{d}_1^2)+\cdots+(n_k-1)(\bar{d}^2-\bar{d}_k^2))}{\bar{d}^2} \\ &= \frac{\frac{1}{n-k}\bar{d}^2((n_1-1)+\cdots+(n_k-1))}{\bar{d}^2} + \frac{\frac{1}{n-k}\bar{d}^2((n_1-1)\bar{d}_1^2+\cdots+(n_k-1)\bar{d}_k^2)}{\bar{d}^2} \\ &= 1 + \frac{\frac{1}{n-k}\bar{d}^2((n_1-1)\bar{d}_1^2+\cdots+(n_k-1)\bar{d}_k^2)}{\bar{d}^2} \end{aligned}$$

Et donc, $a_k = 1$.

◇ Si les points sont uniformément distribués dans l'espace, a_k va augmenter doucement et plus ou moins régulièrement avec k. VRC a quant à lui tendance à décroître quand k augmente si a_k est constant, ceci étant plus ou moins contrebalancé par l'augmentation de a_k.
De toute façon, une distribution uniforme dans l'espace va généralement provoquer une variation régulière des valeurs de VRC.

◇ Par contre, si les points sont naturellement groupés en k_0 ensembles, le passage de $k_0 - 1$ à k_0 va provoquer une augmentation considérable de a_k et même de VRC. Plus précisément, le passage de $k_0 - 1$ à k_0 fera croître énormément VRC si $\frac{a_{k_0}}{a_{k_0-1}}$ est supérieur au rapport $\frac{(k_0-1}{a_{k_0-1}+k_0-2)}$
(ce rapport n'est jamais plus petit que 1).

Nous savons maintenant que le calcul de VRC pour $k = 2, 3, \ldots$ aide à décider quel est le "meilleur" nombre de groupes. Calinski et Harabasz ([4]) suggèrent de **choisir le nombre k pour lequel VRC a un maximum relatif ou absolu, ou au moins une croissance plus rapide.**
Si jamais les valeurs de VRC ont une croissance monotone avec celles de k, on peut conclure que la meilleure partition des points est celle où chaque point forme un groupe.

5. Si tous les points sont différents, cela n'arrive que quand k atteint n

2.4.3 Présentation du programme utilisé

Signalons pour commencer que la version initiale de ce programme a été réalisée par A.D. Gordon.
Maintenant, nous allons expliquer un peu à quoi sert ce programme et comment il procède.
En fait, le programme calcule[6] pour chaque méthode de classification[7], la hiérarchie de partitions correspondante. De plus, pendant la formation de chaque hiérarchie de partitions, il calcule aussi les valeurs de quatre indices (tous ceux présentés au paragraphe précédent) pour chaque partition de la hiérarchie. Enfin, l'algorithme utilisé procède de façon agglomérative. Il part donc de la partition en n classes pour terminer par la partition en une classe.
On obtient donc à chaque fois un résultat du type :

Ensemble de données 1.
Méthode du voisin le plus proche.

	M_1	M_2	M_3	M_4
$k = n$				
$n-1$				
$\vdots$				
1				

où les colonnes du tableau contiennent les valeurs des indices correspondant aux méthodes $M_1, \ldots, M_4$ pour chaque partition en $n, n-1, \ldots, 1$ classe(s).

Méthode du voisin le plus éloigné.
$\vdots$

Ensuite pour chaque tableau, il faut appliquer la règle de décision du nombre de classes correspondant à chaque méthode $M_1, \ldots, M_4$ suivant les différentes valeurs des indices.

6. A chaque fois qu'il est exécuté pour un certain jeu de données.
7. Rappelons que les méthode utilisées sont celles du voisin le plus proche, du voisin le plus éloigné, de WARD et de la moyenne.

2.5 Méthodes de détermination du nombre de classes : approche par la technique du bootstrap

Jusquà présent, nous n'avons vu que des méthodes de détermination du nombre de classes qui calculaient des indices basés sur une hiérarchie de partitions de l'ensemble de données original.

Attardons nous maintenant un peu sur une autre méthode qui utilise la technique du bootstrap.

2.5.1 Méthode de Jain et Moreau ([20])

Introduction

Jain et Moreau ([20]) pour déterminer le nombre de classes d'un ensemble de données, ont adopté une méthodologie basée sur la technique du bootstrap.

Cette méthodologie consiste à générer une série d'échantillons à partir des données initiales par la technique du bootstrap, c'est-à-dire qu'on génère des échantillons de même taille que l'ensemble de données original dont les objets ont été prélevés avec remplacement dans les données initiales (certains objets seront donc répétés et d'autres omis).

On applique alors à chaque échantillon une méthode de classification et on détermine la valeur d'un critère de qualité de classification pour les différents nombres de classes ($k = 1, 2, \ldots$). Le nombre de classes retenu sera celui pour lequel les valeurs du critère seront les plus stables à travers les différents échantillons (ceci sera expliqué un peu plus tard).

Description

Soit E un ensemble de n objets de dimension d.
On suppose que E est classifié et nous notons K^* le "vrai" nombre de classes dans E.
Notre problème est d'estimer la valeur de K^*.
Notons P_K une partition en K classes de E obtenue par un algorithme de classification donné.
L'algorithme de Jain et Moreau est basé sur les idées intuitives suivantes :

(1) Une solution en K^* classes sera stable. Par stabilité, nous entendons que les objets présents dans la classe resteront les mêmes, même si on perturbe légèrement les données dans E.

(2) N'importe quelle autre solution en K classes ($K \neq K^*$) ne sera pas aussi stable que la solution en K^* classes.
Dans ce cas, les points des classes dans P_K peuvent dépendre du choix de l'algorithme de classification et des paramètres d'entrée qui initialise le processus de classification (ce sera le cas pour la méthode K-means). Donc, les classes dans P_K pourraient complètement changer si certains objets de E sont modifiés.

L'idée de "perturber" les données de E a été implémentée par la technique du bootstrap.
Soient donc p[8] échantillons bootstrap de n objets $B_1, B_2, \ldots, B_p$.
Notons $P_{K,r}$ les partitions en K classes obtenues pour B_r, $r = 1, \ldots, p$ et définissons un critère pour caractériser la structure de $P_{K,r}$:

$$W_K = \frac{1}{K} \sum_{i=1}^{K} R_i$$ [9]

où $R_i = \max_{\substack{j \\ j \neq i}} \frac{S_i + S_j}{T_{i,j}}$

$S_i = \{\frac{1}{|C_i|} \sum_{x_j \in C_i} |x_j - g_i|^s\}^{\frac{1}{s}}$
: dispersion de la classe C_i

$T_{i,j} = \{\sum_{l=1}^{d} |g_i^l - g_j^l|^t\}^{\frac{1}{t}}$
: séparation de la classe i et de la classe j

d : dimension des données

g_i : centre de gravité de la classe C_i

$|C_i|$: le nombre d'objets dans C_i

s, t : paramètres naturels non nuls sélectionnés indépendemment pour définir des distances particulières.

W_K est évidemment calculé pour chaque $P_{K,r}$.

Le critère défini par Jain et Moreau pour mesurer la stabilité de la partition P_K est ΔW_K qui représente l'intervalle de confiance à 68% de la variation de $W_K(P_{K,r})$ sur les p échantillons pour un nombre K de classes donné.
La valeur de K qui minimise ΔW_K sera prise comme une estimation de K^*.

Mais les expériences faites par Jain et Moreau montrent que ΔW_K est généralement un meilleur critère pour détecter les partitions en K classes pour lesquelles $K > K^*$ que les partitions en K classes pour lesquelles $K < K^*$.
En d'autres mots, les expériences ont généralement montré que les valeurs de ΔW_K pour toutes les classifications avec $K < K^*$ étaient les mêmes que pour

8. En général, $p = 100$ est un bon nombre ([20]).

$K = K^*$. Cela veut dire que les partitions en K classes, souvent obtenues en fusionnant certaines des classes de la partition en K^*, peuvent aussi fournir des classes bien séparées et, dès lors, des classifications stables.

Pour résoudre ce problème, Jain et Moreau ont considéré un critère supplémentaire : une mesure de compacité des classes.
Une bonne façon de distinguer la partition en K^* classes de la partition en K classes (pour $K < K^*$) est de mesurer la compacité des classes. La compacité des classes dans une solution en K^* classes est généralement plus haute comparée à celle de la solution en K classes où $K < K^*$. La mesure de compacité que nous utilisons est définie en terme de dispersion intra-groupe M.
Notons δM_K la décroissance moyenne sur tous les échantillons bootstrap de la valeur de m quand on passe de K à $K+1$ classes.
Pour chaque nombre de classes K, nous mesurons donc δM_K qui caractérise la compacité de la partition P_K relative à celle de P_{K+1} et ΔW_K qui mesure la stabilité de P_K.

Nous pouvons alors supposer que :

- Pour $K < K^*$, les classes sont stables mais la croissance de la compacité est grande (ΔW_K petit et δM_K grand).
- Pour $K = K^*$, les classes sont stables et la croissance de la compacité est petite (ΔW_K petit et δM_K petit).
- Pour $K > K^*$, les classes sont instables et la croissance en compacité est petite (ΔW_K grand et δM_K petit).

Une combinaison de ces deux paramètres identifie le cas $K = K^*$ comme le seul cas où ΔW_K et δM_K ont les plus petites valeurs.

La statistique de classification est alors définie comme :

$$R_K = a\left(\frac{\Delta W_K}{\|\Delta W\|}\right) + b\left(\frac{\delta M_K}{\|\delta M\|}\right)$$

où $\|\Delta W\| = \sqrt{\sum_{k=K_1}^{K_2} (\Delta W_K)^2}$

$\|\delta M\| = \sqrt{\sum_{k=K_1}^{K_2} (\delta M_K)^2}$

K_1 = le nombre minimum de classes testé
K_2 = le nombre maximum de classes testé
a et b sont des paramètres de poids dont les valeurs respectives 0.75 et 0.25 donnent les meilleurs résultats.

La valeur de K qui minimise R_K est prise comme K^*.

Remarque :
Ce critère convient particulièrement pour des classes hypersphériques ([20]).
Pour d'autres formes de classes (hyperellipsoïdales), on considère un autre critère :

$$S_K = \Delta G_K$$

où $G_K = \frac{\sum \text{des dispersions intra-classes}}{\sum \text{des dispersions inter-classes}}$ pour une partition en K classes.
S_K est l'intervalle de variation de G_K sur les échantillons bootstrap.

Chapitre 3

Etude de la stabilité de quelques méthodes

3.1 Introduction

Dans le chapitre précédent, nous avons exposé toute une série de méthodes de détermination du nombre de classes.
Le nombre de ces méthodes est tellement élevé (nous n'en avons exposé que cinq, mais il en existe bien d'autres) qu'il est très difficile d'en choisir une.

Il est donc intéressant de réaliser une évaluation de ces règles (ou méthodes) à travers trois ensembles de données de structure différente de manière à faciliter le choix de la méthode dans ce genre de contexte.

La première qualité que doit avoir une méthode de détermination du nombre de classes est qu'elle donne des résultats qui soient stables. Par stabilité, nous signifions que les résultats obtenus ne changent pas beaucoup lorsqu'une perturbation est apportée aux données.

Une des perturbations pouvant être apportée à une matrice de données initiale serait la répétition et l'omission d'un certain nombre d'individus. Si une méthode est stable, le fait de répéter et d'omettre aléatoirement une partie des individus ne changera pas énormément les résultats obtenus[1].

Dans cette optique, et dans ce chapitre, nous exposerons des procédures pour étudier la stabilité des résultats de quelques règles d'arrêt par application de la technique du bootstrap.

1. Seulement si on ne tient pas compte de la densité des points.

3.2 Méthodologie employée

3.2.1 Choix des méthodes et techniques d'échantillonnage

Nous allons étudier la stabilité de quatre méthodes de détermination du nombre de classes faisant partie du "top five" de Milligan et Cooper :

- la méthode Gamma
- la méthode de Duda et Hart
- la méthode de Beale
- la méthode de Calinski et Harabasz.

Chacune de ces méthodes étant combinée avec quatre méthodes de classification :

- la méthode du voisin le plus proche
- la méthode du voisin le plus éloigné
- la méthode de la moyenne
- la méthode de WARD.

Le principe de ce travail est donc de générer une série d'échantillons (ils seront au nombre de dix) pour chaque ensemble de données (qui sont au nombre de trois) auxquels nous appliquerons les différentes méthodes de classification.

Il se présente deux manières de former des échantillons (c'est-à-dire d'apporter une perturbation aux données) :

- la technique du jackknife
- la technique du bootstrap.

La technique du jackknife génère des échantillons d'effectif moindre que celui de l'ensemble initial. De plus, il y a prélèvement sans remplacement des objets de l'ensemble de données original et donc élimination d'une partie des données initiales.

La technique du bootstrap, quant à elle, procède avec remplacement. Elle génère alors des échantillons de même taille que l'ensemble de départ, certains objets sont donc répétés et d'autres omis. C'est cette technique que nous utiliserons dans nos applications.

Une autre manière d'apporter des perturbations aux données est la suivante : c'est une technique qui consiste à ajouter un "bruit" ou une erreur aléatoire aux rangs des mesures de proximité.

3.2.2 Présentation des programmes utilisés

Nous avons donc procédé comme suit :

- Génération de 50 échantillons bootstrap pour chaque ensemble de données par le programme echantillonprgm.for[2].
  Pour chaque échantillon, nous spécifions dans le programme un numéro de table de nombres aléatoires. Ensuite, ce programme tire autant de nombres aléatoires qu'il y a de données initiales et retranscrit dans un nouveau fichier les données dont les numéros correspondent aux nombres aléatoires tirés (chacun de ces nombres aléatoires étant compris entre 1 et n où n est le nombre de données initiales).
  Nous obtenons donc, à chaque exécution du programme, un nouvel échantillon, à condition de spécifier à chaque fois un numéro de table de nombres aléatoires différent.

  Exemple : Si notre ensemble de données est $\{5, 10, 7, 6\}$
  et si nous tirons les nombres aléatoires 2, 4, 3 et 3
  Alors notre échantillon bootstrap sera constitué des données $\{10, 6, 7, 7\}$.

- A chacun de ces échantillons, nous appliquons chacune des quatre méthodes de classification et ce, pour chaque règle d'arrêt. Nous disposons donc pour chaque échantillon des nombres de classes qu'il contient et des partitions en ces classes (obtenus en appliquant le programme de A.D. Gordon expliqué au chapitre précédent).

- Suit alors un travail d'encodage de ces partitions de manière à pouvoir employer le programme lambda.for[3] qui compare ces partitions.

Remarque :
Nous avons dû modifier le programme de A.D. Gordon (en annexes) de manière à pouvoir l'utiliser sur des ensembles contenant plusieurs fois le même objet.

2. Annexes.
3. Annexes.

3.3 Critères d'étude de la stabilité

Disposant d'une série de solutions pour un même jeu de données, on peut donc évaluer la variabilité des résultats et par conséquent, la stabilité des différentes méthodes de détermination du nombre de classes (plus la variabilité sera grande, moins stable sera la méthode).

La première manière de juger la stabilité des règles d'arrêt est de dresser un histogramme des nombres de classes obtenues. De même, pour chiffrer ces variations, on peut calculer des écarts-types ou des coefficients de variation des nombres de classes.

Cependant, ceci reste un jugement global, car il se peut que dans certaines situations, on obtienne le même nombre de classes sans que la constitution des classes ne soit la même. Par conséquent, on doit utiliser des mesures permettant d'évaluer la similitude ou l'association entre les différentes solutions.

Ces mesures seront appelées mesures d'association ([15]) et seront donc utilisées pour évaluer la ressemblance entre les solutions obtenues pour les différents échantillons d'un même ensemble de données (et ce pour toutes les méthodes envisagées).

Nous commençerons par donner quelques propriétés des mesures d'association. Ensuite, nous en définirons quelques-unes.

Propriétés.

(1) Par convention, une mesure d'association λ a les propriétés suivantes :

Soit $-1 \leq \lambda \leq 1$
$\lambda = \pm 1$ dans le cas d'association complète
$\lambda = 0$ en cas d'indépendance[4].

Soit $0 \leq \lambda \leq 1$
$\lambda = 1$ dans le cas d'association complète
$\lambda = 0$ en cas d'indépendance.

(2) Toute mesure d'association comprise entre 0 et 1 est indépendante de l'ordre des classes[5].

(3) Toute mesure d'association est symétrique.

4. Nous éclaircirons cette notion d'indépendance plus tard.
5. Cela sera expliqué plus en détail par la suite.

Toutes les mesures d'association que nous allons présenter ici sont basées sur la table de contingence suivante :

Tab. 1.1 : Table de contingence pour les mesures d'association.

	Classification B					
Classif. A	1	$\cdots$	j	$\cdots$	β	Totaux
1	n_{11}		n_{1j}		$n_{1\beta}$	$n_{1.}$
$\vdots$						
i	n_{i1}		n_{ij}		$n_{i\beta}$	$n_{i.}$
$\vdots$						
α	$n_{\alpha 1}$		$n_{\alpha j}$		$n_{\alpha\beta}$	$n_{\alpha.}$
Totaux	$n_{.1}$	$\cdots$	$n_{.j}$	$\cdots$	$n_{.\beta}$	n

où n_{ij} désigne le nombre d'individus qui sont à la fois dans la classe i de la classification A et la classe j de la classification B.

Remarques :

- Eclaircissons maintenant la notion d'indépendance vue à la page précédente :
  il y a indépendance entre les deux classifications, si $n_{ij} = \frac{n_{i.}n_{.j}}{n}$,
  $i = 1, \ldots, \alpha$, $j = 1, \ldots, \beta$.
- Si il y a indépendance, la mesure d'association est nulle mais l'inverse n'est pas vrai.

Exemple : Prenons le cas où la classification A donne une partition en 1 classe et où la classification B donne une partition en 2 classes.

Nous obtenons donc une table de la forme :

	Classification B		
Classif. A	1	2	Totaux
1	n_{11}	n_{12}	$n_{1.}$
2	0	0	0
Totaux	$n_{.1}$	$n_{.2}$	n

On a bien que la mesure d'association est nulle (il faudrait le montrer pour toutes les mesures d'association).

Prenons $n_{11} = 30$ et $n_{12} = 20$. Nous avons donc que $n = 50$, $n_{1.} = 50$, $n_{2.} = 0$, $n_{.1} = 30$ et $n_{.2} = 20$.

A partir de là, nous pouvons facilement voir que $n_{ij} \neq n_{i.}n_{.j}$.

3.3.1 Mesures d'association traditionnelles ([15],[28],[23])

Beaucoup de mesures d'association contiennent la statistique standard χ^2 sur laquelle est basé un test d'indépendance.
Soit une population finie de n membres.

Posons $n_{ij} = n\rho_{ij}$
$n_{i.} = n\rho_{i.}$
$n_{.j} = n\rho_{.j}$

où ρ_{ij} est la proportion d'individus présents dans la classe i de la classification A et dans la classe j de la classification B.

$\rho_{i.}$ est la proportion d'individus présents dans la classe i de la classification A.

$\rho_{.j}$ est la proportion d'individus présents dans la classe j de la classification B.

La statistique χ^2 dans le cas où on compare deux classifications est :

$$\begin{aligned}\chi^2 &= \sum_{i=1}^{\alpha}\sum_{j=1}^{\beta}\frac{(n_{ij}-\frac{n_{i.}n_{.j}}{n})^2}{\frac{n_{i.}n_{.j}}{n}} \\ &= n\sum_i\sum_j\frac{(\rho_{ij}-\rho_{i.}\rho_{.j})^2}{\rho_{i.}\rho_{.j}} \\ &= n\sum_i\sum_j\frac{\rho_{ij}^2}{\rho_{i.}\rho_{.j}} - n\end{aligned}$$

qui est le test χ^2 d'indépendance.

Pour le cas $\alpha = \beta = 2$, Yule ([15],[28]) a défini le coefficient d'association

$$\mathcal{Q} = \frac{n_{11}n_{22} - n_{12}n_{21}}{n_{11}n_{22} + n_{12}n_{21}}$$

Un autre coefficient suggéré par Yule ([15],[28]) pour le cas 2×2 est

$$\mathcal{Y} = \frac{\sqrt{n_{11}n_{22}} - \sqrt{n_{12}n_{21}}}{\sqrt{n_{11}n_{22}} + \sqrt{n_{12}n_{21}}}$$

Pour le cas général $\alpha \times \beta$, un coefficient souvent utilisé est le "mean square contingency" défini par :

$$\Phi^2 = \frac{\chi^2}{n}$$

Karl Pearson en a proposé un autre appelé "coefficient de contingence" :

$$\mathcal{C} = \sqrt{\frac{\frac{\chi^2}{n}}{1 + \frac{\chi^2}{n}}}$$

Un autre coefficient a été introduit par Tschuprow :

$$\mathcal{T} = \sqrt{\frac{\frac{\chi^2}{n}}{(\alpha - 1)(\beta - 1)}}$$

Selon Kendall ([23]), les deux dernières mesures $\mathcal{C}$ et $\mathcal{T}$ ont été faites dans le but de normer χ^2 entre 0 et 1 et de façon à ce que les valeurs extrêmes 0 et 1 correspondent aux cas d'indépendance et d'association complète respectivement.

Cramér quant à lui suggère ([6]) le coefficient :

$$\frac{\frac{\chi^2}{n}}{\min(\alpha - 1, \beta - 1)}$$

qui donne une meilleure norme que $\mathcal{C}$ et $\mathcal{T}$, puisqu'il est compris entre 0 et 1 et que les valeurs extrêmes correspondent bien aux cas d'indépendance et d'association complète.

Mais le fait qu'un excellent test d'indépendance soit basé sur une χ^2 ne signifie pas du tout que χ^2, ou simplement une fonction de celle-ci, est une mesure appropriée du degré d'association ([13]).

C'est pourquoi nous allons utiliser une autre mesure d'association que celles exposées jusqu'ici.

3.3.2 Le coefficient λ ([15])

Le coefficient λ est une mesure d'association basée sur le pouvoir prédicatif d'une variable par une autre.

Si les individus sont classés selon deux classifications A et B, alors on peut estimer la probabilité d'erreur de prédiction de la classe d'un individu pour la classification B :

(1) en ne tenant pas compte de sa classe dans la classification A[6].

(2) en tenant compte de sa classe dans la classification A.

La diminution relative de cette probabilité entre le cas (1) et le cas (2) peut être utilisée comme mesure d'association entre les deux classifications.

Le coefficient λ est ainsi défini par :

$$\lambda = \frac{\sum_{i=1}^{\alpha} r_i + \sum_{j=1}^{\beta} c_j - r - c}{2n - r - c}$$

où $r_i = \max_j(n_{ij})$

$c_j = \max_i(n_{ij})$

$r = \max_j(n_{.j})$

$c = \max_i(n_{i.})$

et où les n_{ij}, $n_{.j}$ et $n_{i.}$ sont obtenus à partir de la table de contingence *1.1.*

Propriétés.

(1) λ est déterminé sauf lorsque la population entière se trouve dans une seule cellule de la table.

(2) sinon $0 \leq \lambda \leq 1$.

(3) $\lambda = 1$ si et seulement si toute la population est concentrée dans des cellules dont deux ne sont jamais sur la même ligne ou colonne, c'est-à-dire dans la situation suivante (pour le cas 3×3) :

	ClassificationB			
Classif.A	1	2	3	Totaux
1	n_{11}	0	0	$n_{1.}$
2	0	0	n_{23}	$n_{2.}$
3	0	n_{32}	0	$n_{3.}$
Totaux	$n_{.1}$	$n_{.2}$	$n_{.3}$	n

6. En supposant que la classification A précède la classification B chronologiquement.

Cette table correspond bien à une association parfaite car prenons par exemple n_{23} : comme il n'y a rien d'autre que n_{23} sur la deuxième ligne et la troisième colonne de la table, cela veut dire que n_{23} est le nombre de points appartenant à la classe 2 de la classification A et à la classe 3 de la classification B, et donc que n_{23} est le nombre de points appartenant à une seule classe de chaque classification. Dès lors, il y a forcément association complète entre ces deux classes.

(4) $\lambda = 0$ si il y a indépendance entre les deux classifications, mais l'inverse n'est pas vrai.

(5) λ est invariant par permutation des lignes ou des colonnes.

(6) λ est symétrique, c'est-à-dire que si on permute "classification A" et "classification B", la table sera modifiée, mais λ ne changera pas.

Remarque :
Une des propriétés générales d'une mesure d'association était :

Toute mesure d'association est indépendante de l'ordre des classes.

Ce n'est pas toujours le cas. Il se peut que, dans certaines situations, on veuille distinguer les deux tables suivantes (pour le cas 3×3) :

n_{11}	0	0
0	n_{22}	0
0	0	n_{33}

(a)

et

0	0	n_{13}
0	n_{22}	0
n_{31}	0	0

(b)

Par convention, la mesure d'association vaudra 1 dans le cas d'**association** (cas (a)) et vaudra -1 dans le cas d' **association contraire**[7] (cas (b)).

Une mesure proposée est alors :

$$\gamma = \frac{\pi_s - \pi_d}{1 - \pi_t}$$

où $\pi_s = Pr[(a_1 < a_2 \text{ et } b_1 < b_2) \text{ ou } (a_1 > a_2 \text{ et } b_1 > b_2)]$
$\pi_d = Pr[(a_1 < a_2 \text{ et } b_1 > b_2) \text{ ou } (a_1 > a_2 \text{ et } b_1 < b_2)]$
$\pi_t = Pr[(a_1 = a_2 \text{ ou } b_1 = b_2)]$
$a_i = 1, \ldots, \alpha \quad i = 1, 2$
$b_i = 1, \ldots, \beta$

7. Nous nous retrouvons alors dans le cas (1) des propriétés des mesures d'association.

En d'autres mots,

a_1 (a_2) représente la classe de la classification A pour l'individu 1 (2).
b_1 (b_2) représente la classe de la classification B pour l'individu 1 (2).

π_s représente la probabilité que les individus 1 et 2 soient dans des classes différentes (pour la même classification) et que l'ordre des classes soit le même pour les deux classifications.

π_d représente la probabilité que les individus 1 et 2 soient dans des classes différentes (pour la même classification) et que l'ordre des classes soit différent pour les deux classifications.

π_t représente la probabilité que les individus 1 et 2 soient dans la même classe soit pour la classification A, soit pour la classification B.

On peut réécrire γ, puisque $\pi_s + \pi_d = 1 - \pi_t$, comme

$$\gamma = \frac{2\pi_s - 1 + \pi_t}{1 - \pi_t}$$

en utilisant $\pi_s = 2\sum_a \sum_b \rho_{ab} \{ \sum_{a'>a} \sum_{b'>b} \rho_{a'b'} \}$

$$\pi_t = \sum_a \rho_{a.}^2 + \sum_b \rho_{.b}^2 - \sum_a \sum_b \rho_{ab}^2$$

Propriétés.

(1) γ est indéterminé si la population est entièrement dans une seule ligne ou colonne de la table.

(2) $\gamma = 1$ si la population est concentrée dans une diagonale qui descend de gauche à droite.
$\gamma = -1$ si la population est concentrée dans une diagonale qui monte de gauche à droite.

(3) $\gamma = 0$ si il y a indépendance entre les deux classifications, mais l'inverse n'est pas vrai, sauf dans le cas 2×2.

3.4 Applications

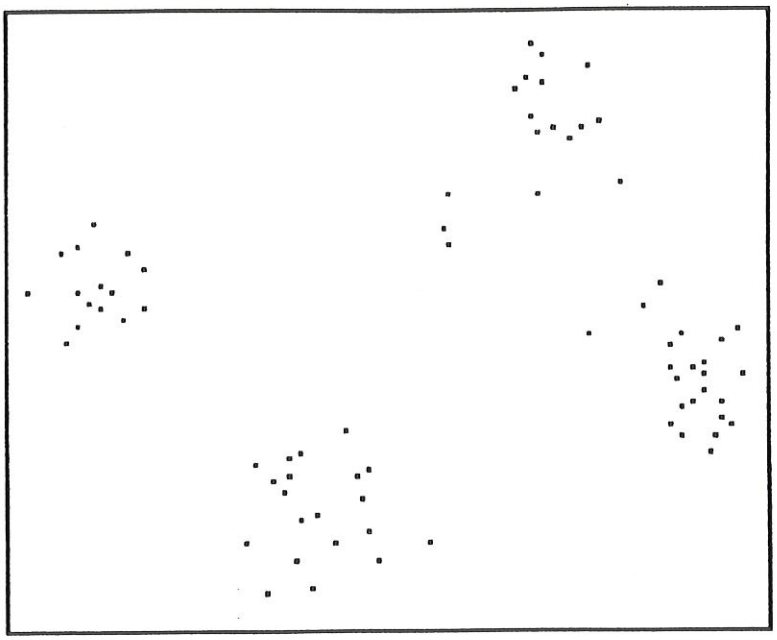
Les procédures que nous avons mises au point ont été appliquées sur trois jeux de données :

- les données de Ruspini
- les données ALLON
- les données SOURIRE.

Rappelons que pour les méthodes de Duda et Hart et de Beale, nous prenons les niveaux $\alpha = 3.20$ et $\alpha = 4.61$ respectivement.

3.4.1 Aux données de Ruspini : 4 classes séparées

Cet exemple comporte 75 données et est souvent utilisé pour vérifier l'applicabilité des méthodes de classification et de détermination du nombre de classes.



Données de Ruspini .

Si nous appliquons les quatre méthodes de détermination du nombre de classes aux partitions obtenues par les méthodes de classification choisies, on obtient :

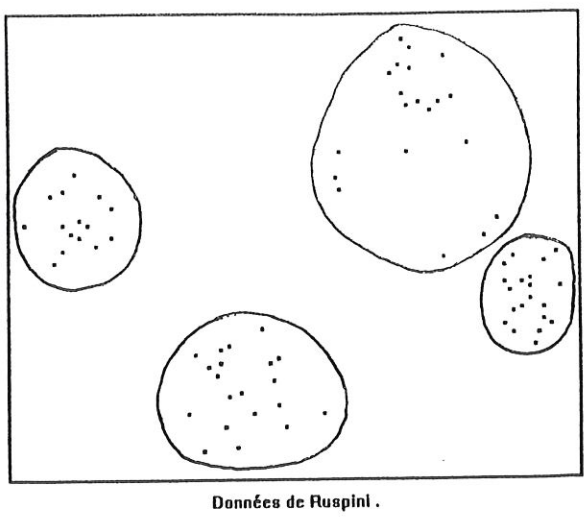
Données de Ruspini	Gam	D-H	Beale	C-H
Voisin le plus proche	4	4	4	4
Voisin le plus éloigné	4	4	4	4
Moyenne	4	4	4	4
Ward	4	4	4	4

Pour examiner la constitution des classes, nous avons calculé le coefficient λ entre les partitions en 4 classes obtenues par les différentes méthodes de classification et les classes naturelles. Pour ce jeu de données, nous obtenons :

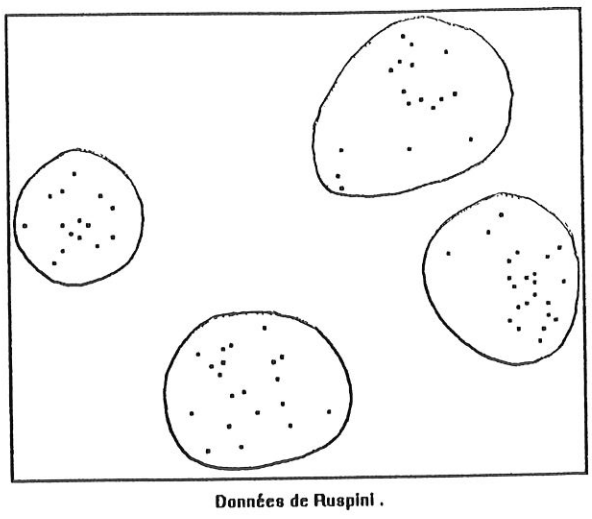
Tab. 1.2. : Coefficient λ pour Ruspini avec les classes naturelles.

	Gam	D-H	Beale	C-H
Voisin le plus proche	1	1	1	1
Voisin le plus éloigné	0.9439	0.9439	0.9439	0.9439
Moyenne	1	1	1	1
Ward	1	1	1	1

Il y a donc parfaite association entre les classes naturelles et les classes obtenues par les différentes méthodes sauf pour la méthode du voisin le plus éloigné qui donne une autre classification toute aussi bonne que les autres[8] (trois points ont été déplacés d'une classe dans une autre).



Classif. pour le voisin le plus éloigné

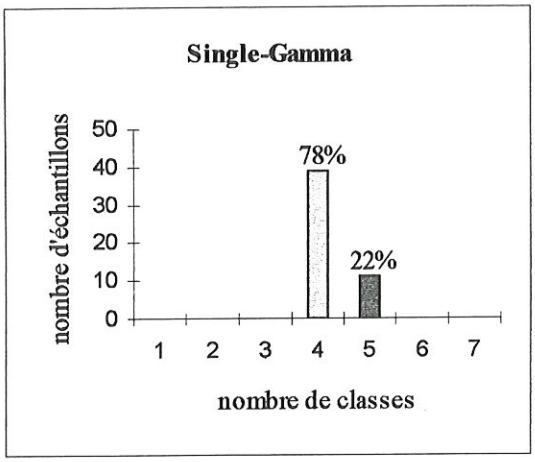


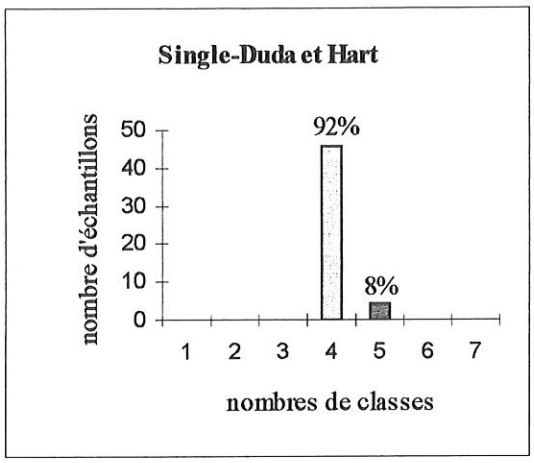
Classif. pour les autres méthodes

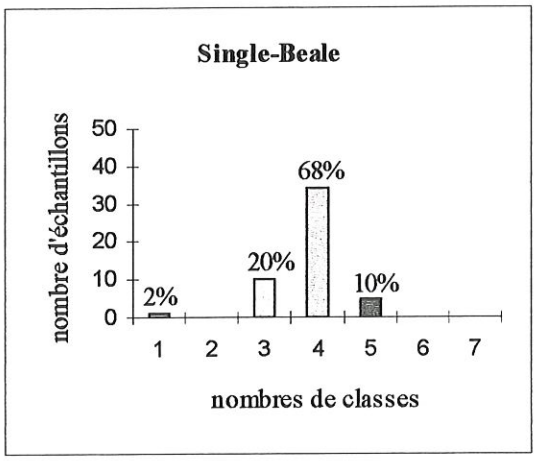
8. Cela dépend de ce qu'on entend par classes naturelles.

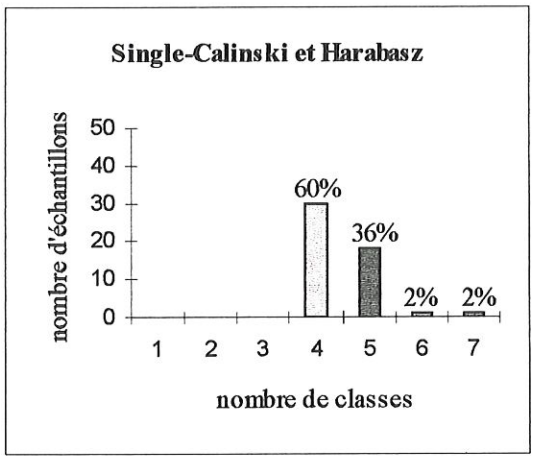
Appliquons maintenant les mêmes procédures aux différents échantillons. Notons que, pour la détermination du nombre de classes, nous travaillerons sur 50 échantillons, mais nous nous restreindrons à 10 pour les calculs du coefficient λ. Nous allons donc, dans un premier temps, dresser un histogramme des différents nombres de classes obtenus pour les différents échantillons. Ensuite, nous travaillerons sur les 10 premiers échantillons pour étudier la stabilité des différentes méthodes de détermination du nombre de classes.

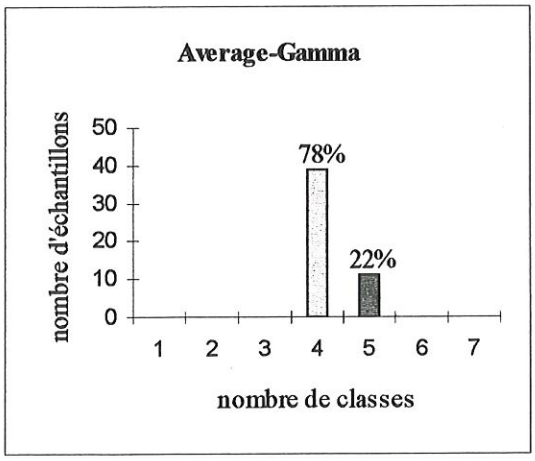
Histogrammes pour Ruspini

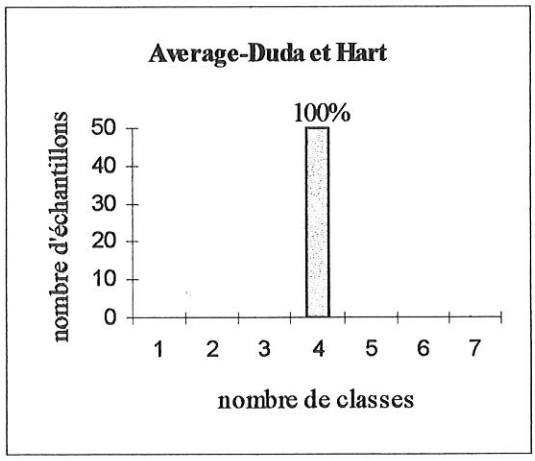


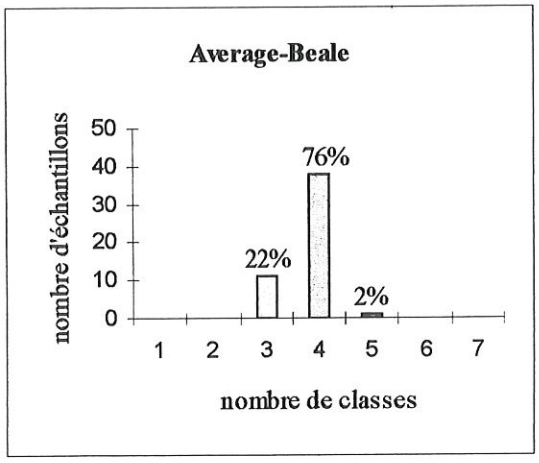


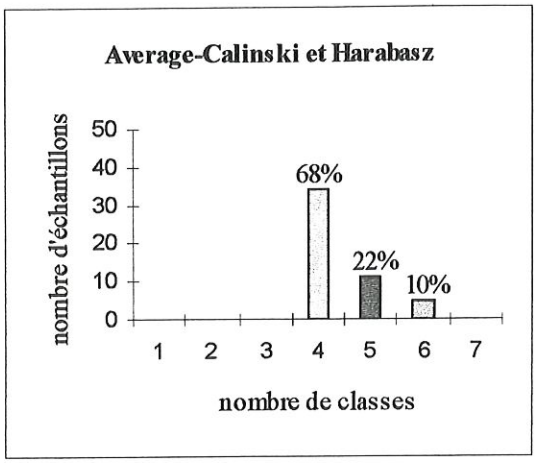


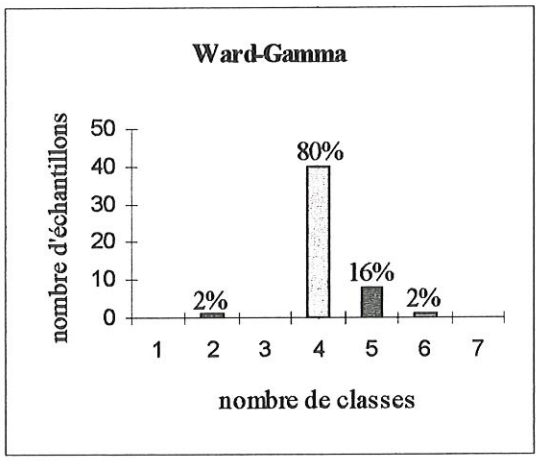


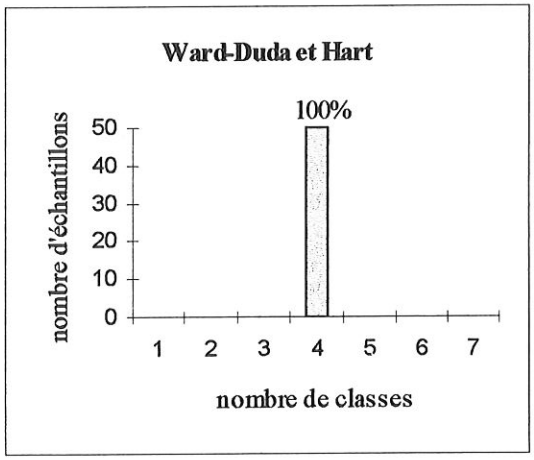


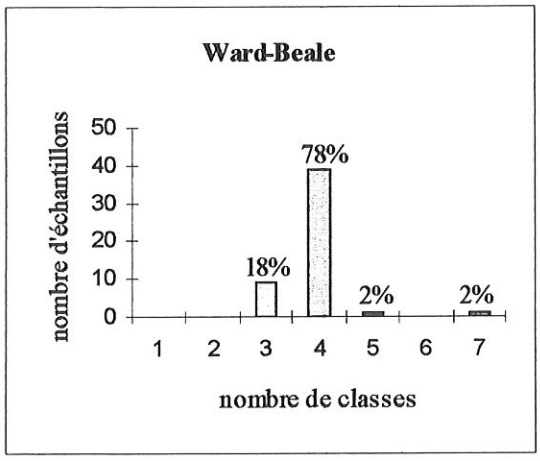


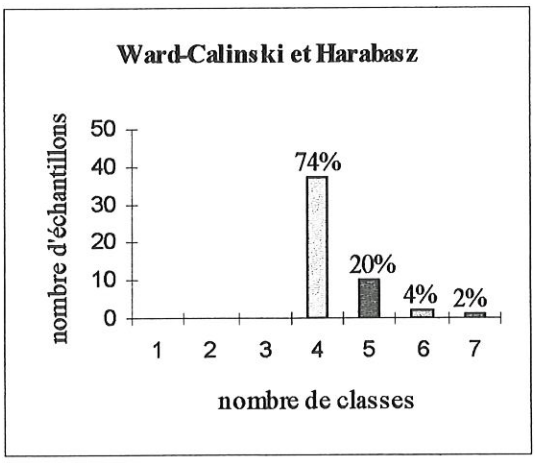

Average-Beale
nombre d'échantillons
50
40
30
20
10
0
22%
76%
2%
1 2 3 4 5 6 7
nombre de classes

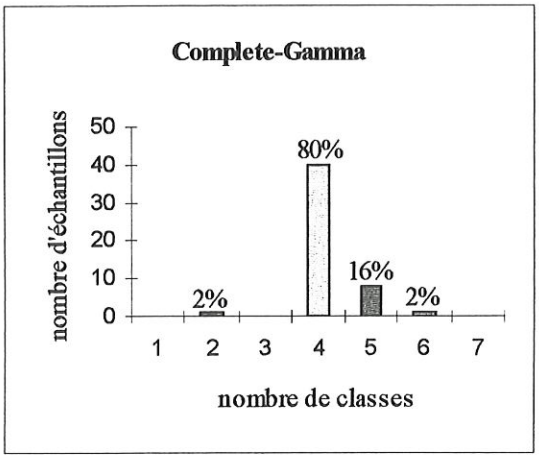

Average-Calinski et Harabasz
nombre d'échantillons
50
40
30
20
10
0
68%
22%
10%
1 2 3 4 5 6 7
nombre de classes

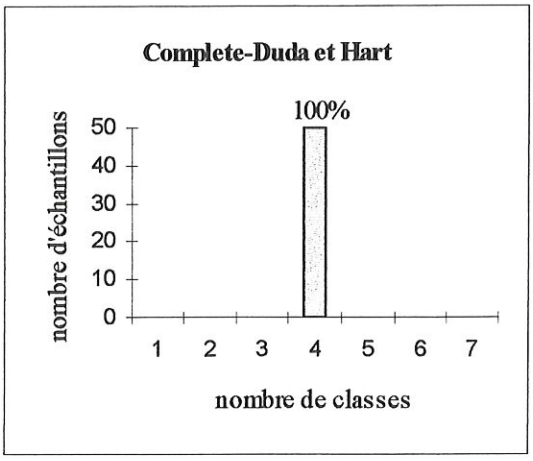

Ward-Gamma
nombre d'échantillons
50
40
30
20
10
0
2%
80%
16%
2%
1 2 3 4 5 6 7
nombre de classes

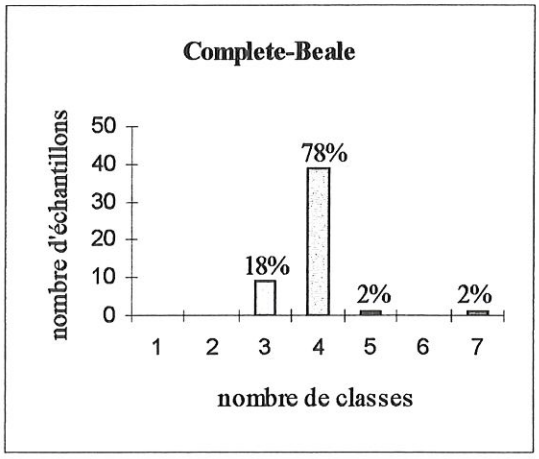

Ward-Duda et Hart
nombre d'échantillons
50
40
30
20
10
0
100%
1 2 3 4 5 6 7
nombre de classes

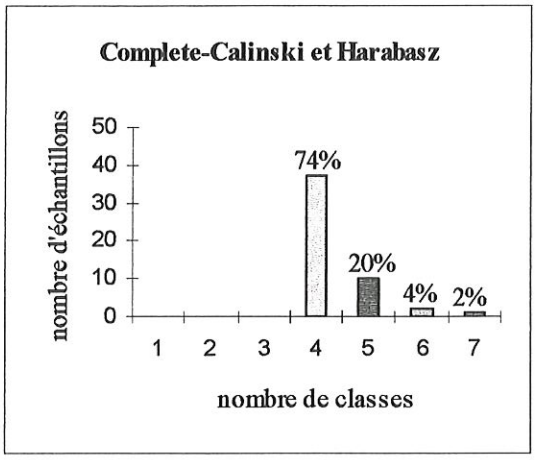

Ward-Beale
nombre d'échantillons
50
40
30
20
10
0
18%
78%
2%
2%
1 2 3 4 5 6 7
nombre de classes


Ward-Calinski et Harabasz
nombre d'échantillons
50
40
30
20
10
0
74%
20%
4%
2%
1 2 3 4 5 6 7
nombre de classes









On peut remarquer que dans tous les cas, la règle de Duda et Hart retrouve toujours le bon nombre de classes (pour tous les échantillons). Mais les autres méthodes donnent aussi de bons résultats, ce qui est logique car elles ont toutes tendance à priviligier les classes hypersphériques.

Maintenant, nous allons travailler sur les 10 échantillons.

A. Méthode du voisin le plus proche

Le tableau suivant donne les nombres de classes obtenus, pour les différents échantillons, par les différentes règles d'arrêt ainsi que les moyennes et les écart-types de ces nombres.

Tab. 1.3. : Nombres de classes obtenus pour les différents échantillons Méthode du voisin le plus proche

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	4	4	4	4
2	4	4	3	5
3	4	4	4	4
4	4	4	3	4
5	5	4	3	7
6	4	4	5	5
7	4	4	4	4
8	4	4	4	5
9	4	4	4	5
10	4	4	4	4
Moyenne	4.1	4	3.8	4.7
E-T	0.2998	0	0.6	0.9

Du point de vue du nombre de classes obtenu, on constate que la règle d'arrêt de Duda et Hart donne les résultats les plus stables, suivie de la règle Gamma et Beale. Ici, la règle de Calinski et Harabasz s'est montrée la moins stable.
En ce qui concerne la constitution des classes obtenues, on a calculé les coefficients λ entre les solutions des différents échantillons et la solution du jeu de données initial d'une part et entre les échantillons deux à deux d'autre part. Les résultats détaillés de ces calculs sont présentés en annexes. Ces résultats sont résumés dans le tableau suivant en termes de moyennes, d'écarts-types et de valeurs extrêmes.

Tab. 1.4. : Coefficient λ entre les classes obtenues pour les différents échantillons Méthode du voisin le plus proche

Règle	Moyenne	Min	Max	E-T
Gam	0.9872	0.92	1	0.02234
D-H	0.9959	0.9565	1	0.01187
Beale	0.8859	0.5517	1	0.13049
C-H	0.9626	0.8462	1	0.04477

De nouveau, en examinant ces résultats, on constate que les règles de Duda-Hart et Gamma sont les plus stables. Par conséquent, nous retrouvons en troisième et quatrième positions les règles de Calinsky-Harabasz et Beale.
Or, pour le nombre de classes, nous avions le contraire : Beale s'était avérée plus stable que Calinski-Harabasz.
Cela est dû au fait que nous considérons qu'il est plus "grave" de fusionner deux classes naturelles (car alors on perd de l'information) que de laisser une division d'une classe naturelle en deux sous-classes. Et ce problème se voit mieux dans le calcul du coefficient λ que dans le tableau du nombre de classes.
C'est pourquoi nous dirons que la méthode de Calinski-Harabasz est plus stable que la méthode de Beale.

B. Méthode du voisin le plus éloigné

Tab. 1.5. : Nombres de classes obtenus pour les différents échantillons Méthode du voisin le plus éloigné

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	4	4	4	4
2	4	4	4	5
3	4	4	4	4
4	4	4	3	4
5	5	4	4	4
6	4	4	5	5
7	4	4	4	4
8	4	4	4	5
9	4	4	4	4
10	4	4	4	4
Moyenne	4.1	4	4	4.3
E-T	0.2998	0	0.4472	0.4583

Nous pouvons faire les mêmes constatations que pour la méthode du voisin le plus proche. La règle de Duda et Hart est la plus stable suivie des règles Gamma, Beale et Calinski-Harabasz.

Tab. 1.6. : Coefficient λ entre les classes obtenues pour les différents échantillons Méthode du voisin le plus éloigné

Règle	Moyenne	Min	Max	E-T
Gam	0.9872	0.92	1	0.02234
D-H	0.9959	0.9565	1	0.01187
Beale	0.9539	0.7429	1	0.07618
C-H	0.9811	0.92	1	0.02426

De nouveau, nous pouvons faire les mêmes remarques que précédemment. Nous dirons que la méthode de Duda et Hart est la plus stable. Ensuite viennent les règles Gamma, Calinski-Harabasz et Beale.

C. Méthode de la moyenne

Tab. 1.7. : Nombres de classes obtenus pour les différents échantillons Méthode de la moyenne

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	4	4	4	4
2	4	4	3	5
3	4	4	4	4
4	4	4	3	4
5	5	4	3	5
6	4	4	5	5
7	4	4	4	4
8	4	4	4	5
9	4	4	4	5
10	4	4	4	4
Moyenne	4.1	4	3.8	4.5
E-T	0.2998	0	0.6	0.5

A nouveau, nous classons la méthode de Duda et Hart comme étant la plus stable suivie de la méthode Gamma, et des méthodes de Beale et Calinski-Harabasz.

On préférera pour les mêmes raisons que précédemment la méthode de Calinski-Harabasz à celle de Beale.

Tab. 1.8. : Coefficient λ entre les classes obtenues pour les différents échantillons Méthode de la moyenne

Règle	Moyenne	Min	Max	E-T
Gam	0.9872	0.92	1	0.02234
D-H	0.9959	0.9565	1	0.01187
Beale	0.8859	0.5517	1	0.1249
C-H	0.9794	0.92	1	0.02344

Conclusion : Duda et Hart est la méthode la plus stable suivie des méthodes Gamma, Calinski-Harabasz et Beale, ce qui confirme nos conclusions précédentes.

D. Méthode de Ward

Tab. 1.9. : Nombres de classes obtenus pour les différents échantillons
Méthode de Ward

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	4	4	4	4
2	4	4	4	5
3	4	4	4	4
4	4	4	3	4
5	5	4	4	4
6	4	4	5	5
7	4	4	4	4
8	4	4	4	5
9	4	4	4	4
10	4	4	4	4
Moyenne	4.1	4	4	4.3
E-T	0.2998	0	0.4472	0.4583

Nous obtenons le même tableau (pour le nombre de classes) que celui de la méthode du voisin le plus éloigné, nous pouvons donc en tirer les mêmes conclusions :

la méthode de Duda et Hart est la plus stable.

Tab. 1.10. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode de Ward

Règle	Moyenne	Min	Max	E-T
Gam	0.9872	0.92	1	0.02234
D-H	0.9959	0.9565	1	0.01187
Beale	0.9539	0.7429	1	0.07618
C-H	0.9811	0.92	1	0.02426

A nouveau, nous concluons que la méthode de Duda et Hart est la plus stable suivie des méthodes Gamma, Calinski-Harabasz. La méthode de Beale s'avère la moins stable.

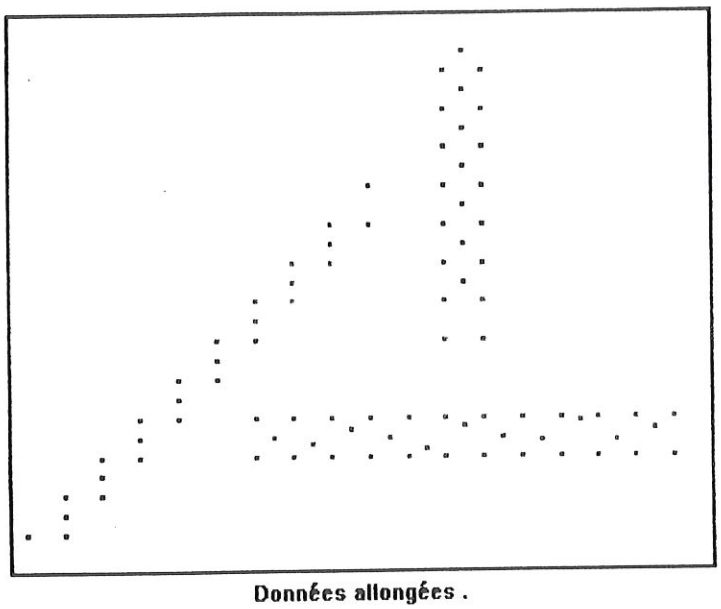
Conclusions générales.

Pour les données de Ruspini, nous pouvons classer les méthodes de détermination du nombre de classes de la plus stable à la moins stable :

- la méthode de Duda et Hart
- la méthode Gamma
- la méthode de Calinski et Harabasz
- la méthode de Beale.

3.4.2 Aux données ALLON : 3 classes allongées

Cet exemple comporte 85 données.
On y remarque trois classes naturelles telles qu'aucune d'entre elles n'est séparable des autres par un hyperplan.



Données allongées .

Si nous appliquons les quatre méthodes de détermination du nombre de classes aux partitions obtenues par les méthodes de classification choisies, on obtient :

Données ALLON	Gam	D-H	Beale	C-H
Voisin le plus proche	1	1	1	3
Voisin le plus éloigné	1	3	1	1
Moyenne	1	1	1	1
Ward	1	3	1	1

Pour examiner la constitution des classes, nous avons calculé le coefficient λ entre les partitions obtenues par les différentes méthodes de classification et les classes naturelles. Pour ce jeu de données, nous obtenons :

Tab. 1.11. : Coefficient λ pour ALLON avec les classes naturelles.

	Gam	D-H	Beale	C-H
Voisin le plus proche	0	0	0	1
Voisin le plus éloigné	0	0.6098	0	0
Moyenne	0	0	0	0
Ward	0	0.6098	0	0

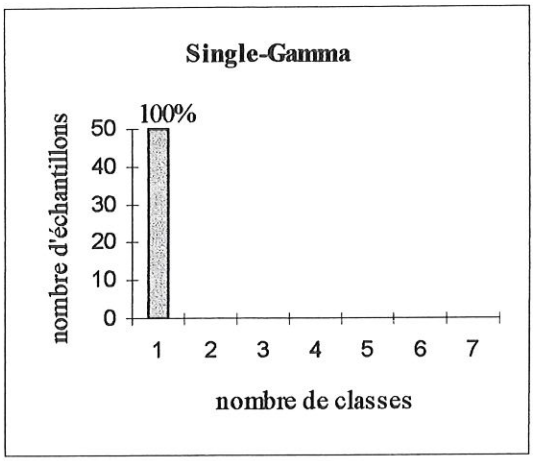
Nous pouvons constater que seule la méthode du voisin le plus proche retrouve les classes naturelles.
Remarquons aussi que λ vaut 0 dans beaucoup de situations. Ce sera toujours le cas lorsqu'on voudra calculer le degré d'association entre une partition en 1 classe et une partition en k classes ($k \neq 1$), car par définition de λ, tous les individus se trouveront alors sur une seule ligne ou colonne de la table de contingence ce qui entraîne une valeur nulle pour le coefficient λ.

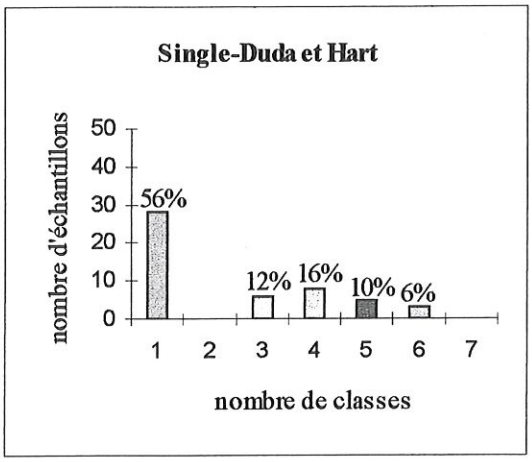
Remarque :

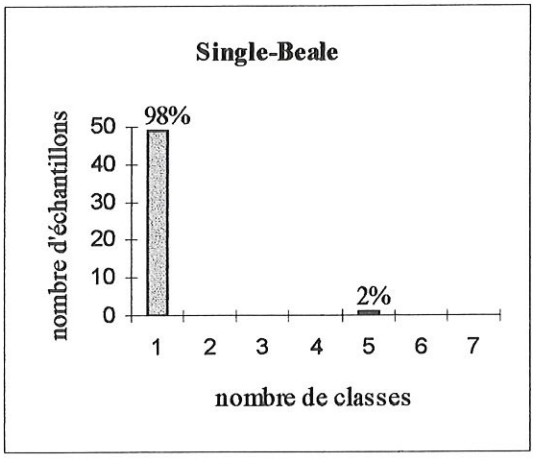
> Il se peut qu'une partition en 5 classes comparées à une partition en 4 classes donne $\lambda = 1$. En effet, il suffit que les points de la cinquième classe ne soient repris dans aucune des classes de la partition en 4 classes.

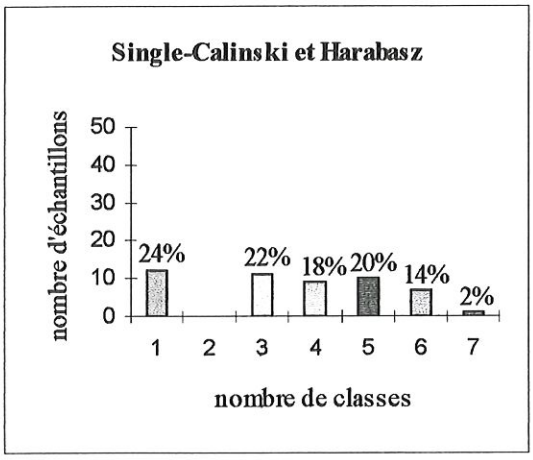
Comme pour les données de Ruspini, nous allons dresser un histogramme des différents nombres de classes obtenus par les différentes méthodes.

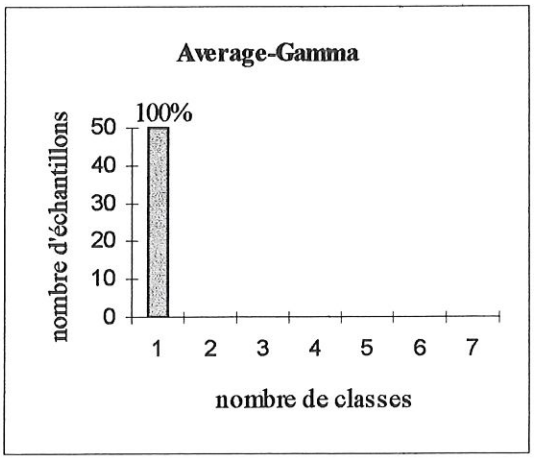
Histogrammes pour ALLON

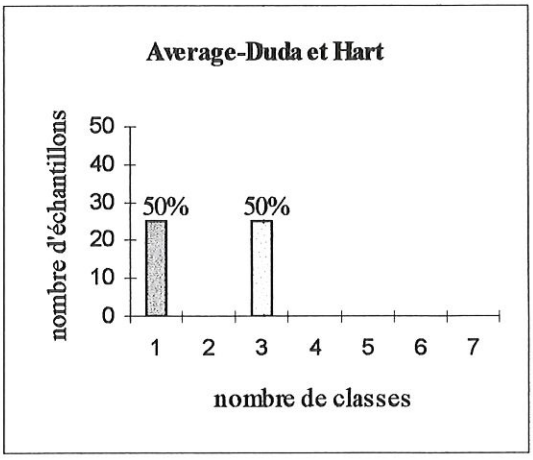


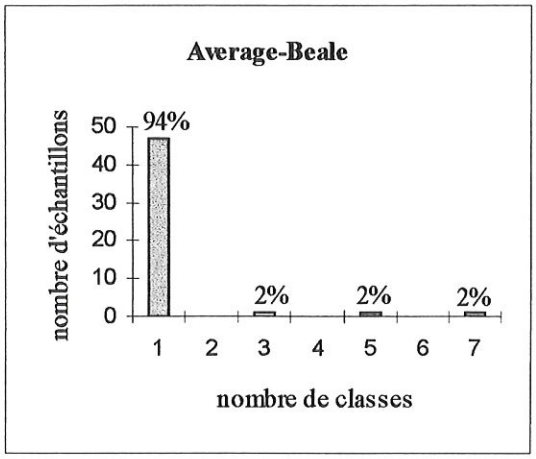


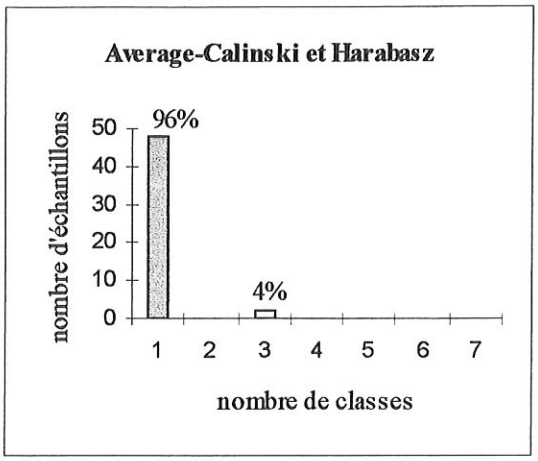


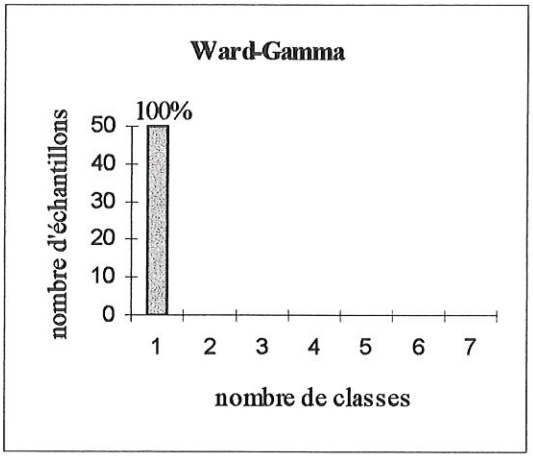


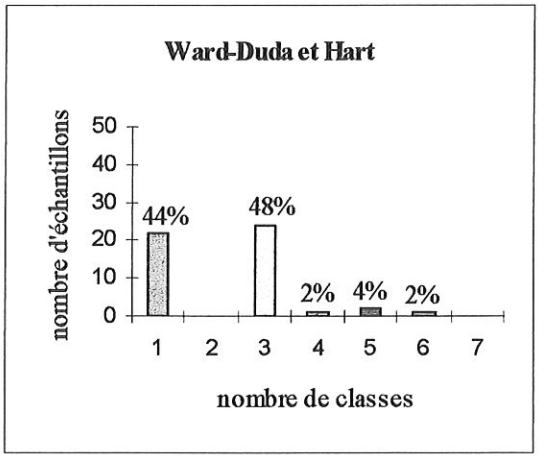


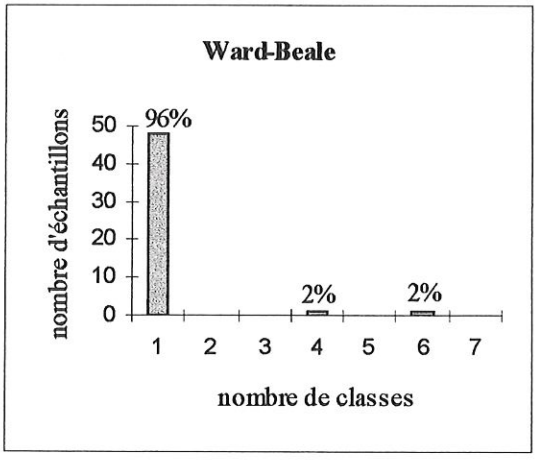


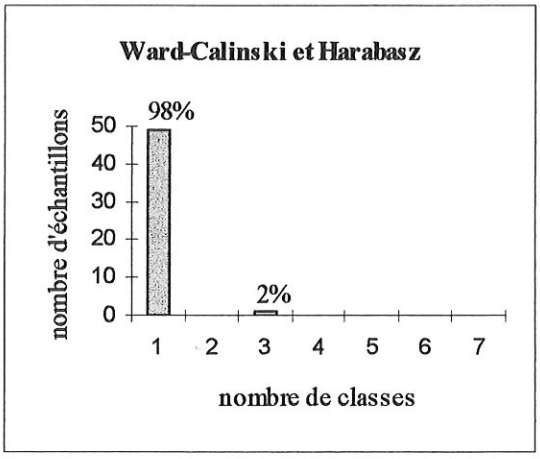

Average-Beale
nombre d'échantillons
50
40
30
20
10
0
94%
2%
2%
2%
1 2 3 4 5 6 7
nombre de classes

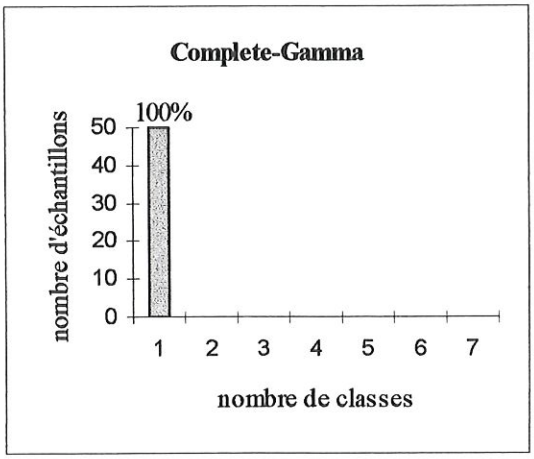

Average-Calinski et Harabasz
nombre d'échantillons
50
40
30
20
10
0
96%
4%
1 2 3 4 5 6 7
nombre de classes

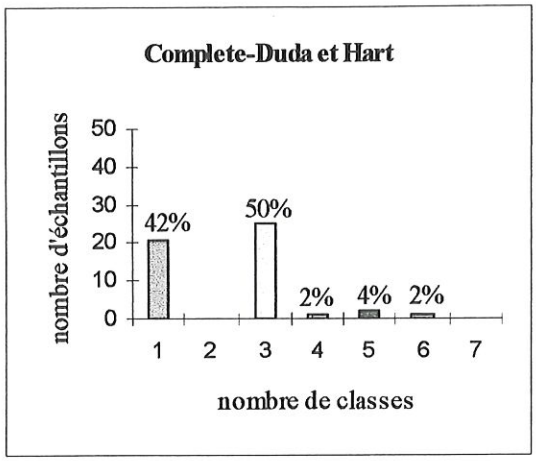

Ward-Gamma
nombre d'échantillons
50
40
30
20
10
0
100%
1 2 3 4 5 6 7
nombre de classes

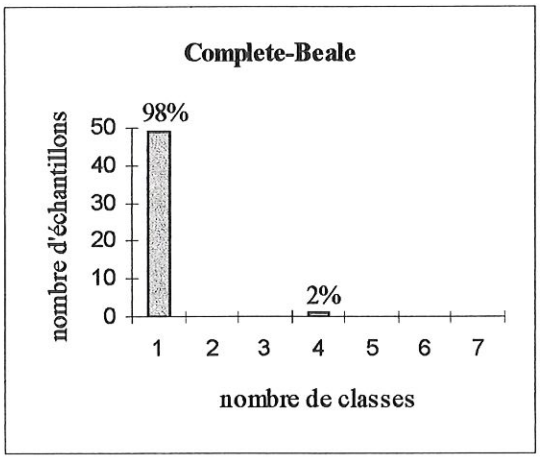

Ward-Duda et Hart
nombre d'échantillons
50
40
30
20
10
0
44%
48%
2%
4%
2%
1 2 3 4 5 6 7
nombre de classes

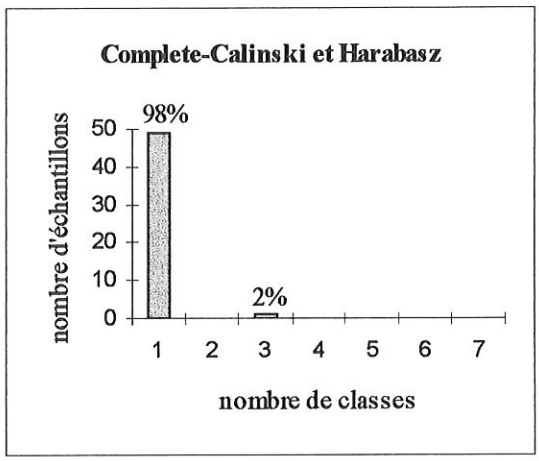

Ward-Beale
nombre d'échantillons
50
40
30
20
10
0
96%
2%
2%
1 2 3 4 5 6 7
nombre de classes


Ward-Calinski et Harabasz
nombre d'échantillons
50
40
30
20
10
0
98%
2%
1 2 3 4 5 6 7
nombre de classes









Nous pouvons remarquer que les méthodes Gamma et Beale donnent dans (presque) tous les cas le même nombre de classes : 1. En effet, ces deux méthodes ont tendance à retrouver une absence de structure dans un ensemble de classes allongées.

Appliquons maintenant la même procédure aux différents échantillons et ce pour chaque méthode de classification.

A. Méthode du voisin le plus proche

Tab. 1.12. : Nombres de classes obtenus pour les différents échantillons
Méthode du voisin le plus proche

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	1	1	1	3
2	1	1	1	1
3	1	1	1	1
4	1	1	1	3
5	1	6	1	6
6	1	6	1	1
7	1	1	1	3
8	1	4	1	1
9	1	5	1	6
10	1	6	1	6
Moyenne	1	3.2	1	3.1
E-T	0	2.2716	0	2.0712

Du point de vue du nombre de classes, ce sont les méthodes Gamma et Beale qui paraissent les plus stables suivies des règles de Calinski-Harabasz et Duda-Hart (ceci indépendamment du fait que les méthodes retrouvent le "bon" nombre de classes ou non).
Examinons maintenant la constitution des classes.

Tab. 1.13. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode du voisin le plus proche

Règle	Moyenne	Min	Max	E-T
Gam	1	1	1	0
D-H	0.4227	0	1	0.6378
Beale	1	1	1	0
C-H	0.3949	0	1	0.4179

Comme précédemment, nous obtenons beaucoup de valeurs nulles pour le coefficient λ (voir résultats en annexes). Ce qui influence fortement vers le bas la moyenne de ce coefficient.

Nous tirons à nouveau comme conclusions que les méthodes Gamma et Beale sont les plus stables suivies des méthodes de Calinski-Harabasz et Duda-Hart.

B. Méthode du voisin le plus éloigné

Tab. 1.14. : Nombres de classes obtenus pour les différents échantillons
Méthode du voisin le plus éloigné

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	1	1	4	1
2	1	5	1	1
3	1	3	1	1
4	1	3	1	1
5	1	5	1	1
6	1	6	1	1
7	1	3	1	1
8	1	3	1	1
9	1	4	1	1
10	1	3	1	1
Moyenne	1	3.6	1.3	1
E-T	0	1.0789	0.9	0

Nous constatons que, du point de vue du nombre de classes, cette fois ce sont les méthodes Gamma et Calinski-Harabasz qui sont les plus stables suivies des méthodes de Beale et de Duda-Hart.

Tab. 1.15. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode du voisin le plus éloigné

Règle	Moyenne	Min	Max	E-T
Gam	1	1	1	0
D-H	0.5839	0	1	0.3055
Beale	0.818	0	1	0.3857
C-H	1	1	1	0

De nouveau, ce sont les méthodes Gamma et Calinski-Harabasz qui sont les plus stables suivies des méthodes de Beale et de Duda-Hart.

C. Méthode de la moyenne

Tab. 1.16. : Nombres de classes obtenus pour les différents échantillons
Méthode de la moyenne

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	1	1	4	3
2	1	1	1	1
3	1	3	1	1
4	1	3	1	1
5	1	3	1	1
6	1	1	1	1
7	1	3	1	1
8	1	3	1	1
9	1	3	1	1
10	1	1	1	1
Moyenne	1	2.2	1	1.2
E-T	0	0.9798	0	0.6

Ici, on retrouve les mêmes conclusions que celles faites pour la méthode du voisin le plus proche :
les méthodes Gamma et Beale sont les plus stables suivies des méthodes de Calinski-Harabasz et de Duda-Hart.

Tab. 1.17. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode de la moyenne

Règle	Moyenne	Min	Max	E-T
Gam	1	1	1	0
D-H	0.4119	0	1	0.4584
Beale	1	1	1	0
C-H	0.818	0	1	0.3857

Ce tableau confirme les résultats obtenus pour le nombre de classes :
les méthodes Gamma et Beale sont les plus stables suivies des méthodes de Calinski-Harabasz et de Duda-Hart.

D. Méthode de Ward

Tab. 1.18. : Nombres de classes obtenus pour les différents échantillons
Méthode de Ward

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	1	1	4	1
2	1	5	1	1
3	1	3	1	1
4	1	3	1	1
5	1	5	1	1
6	1	6	1	1
7	1	3	1	1
8	1	3	1	1
9	1	4	1	1
10	1	1	1	1
Moyenne	1	3.4	1.3	1
E-T	0	1.5875	0.9	0

De ces résultats, nous pouvons tirer les mêmes conclusions que celles faites pour la méthode du voisin le plus éloigné :
les méthodes Gamma et Calinski-Harabasz sont les plus stables suivies des méthodes de Beale et de Duda-Hart.

Tab. 1.19. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode de Ward

Règle	Moyenne	Min	Max	E-T
Gam	1	1	1	0
D-H	0.4854	0	1	0.3624
Beale	0.818	0	1	0.3857
C-H	1	1	1	0

Nous constatons la même chose que pour le nombre de classes :
les méthodes Gamma et Calinski-Harabasz sont les plus stables suivies des méthodes de Beale et de Duda-Hart.

Conclusions générales :

Pour les données ALLON, nous pouvons classer les règles de la plus stable à la moins stable selon les méthodes de classification.

pour les méthodes du voisin le plus proche et de la moyenne :

- les méthodes Gamma et Beale
- la méthode de Calinski-Harabasz
- la méthode de Duda-Hart

pour les méthodes du voisin le plus éloigné et de Ward :

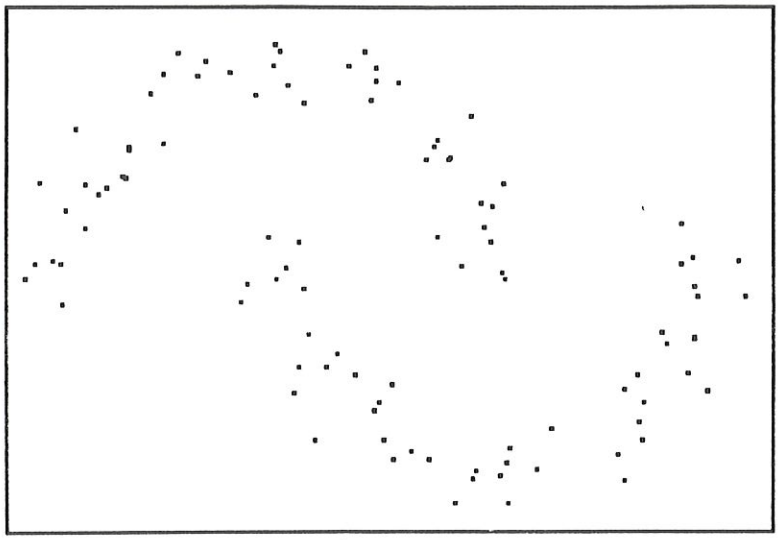
- les méthodes Gamma et de Calinski-Harabasz
- la méthode de Beale
- la méthode de Duda-Hart

Remarques :

- La méthode Gamma est placée comme étant la plus stable pour toutes les méthodes de classification.
- Nous avons un résultat différent de celui obtenu sur les données de Ruspini : la règle de Duda-Hart est la plus stable pour les données de Ruspini et la moins stable pour les donées ALLON. Mais la règle Gamma est très bien placée pour ces deux jeux de données (deuxième et première positions).

3.4.3 Aux données SOURIRE : 2 classes

Cet exemple comporte 100 données.
On peut remarquer que les classes naturelles ne sont pas convexes et elles ne sont pas séparables l'une de l'autre par un hyperplan.



Données en sourire .

Si nous appliquons les quatre méthodes de détermination du nombre de classes aux partitions obtenues par les méthodes de classification choisies, on obtient :

Données SOURIRE	Gam	D-H	Beale	C-H
Voisin le plus proche	1	1	1	5
Voisin le plus éloigné	1	1	1	1
Moyenne	1	1	1	1
Ward	1	1	1	1

Pour examiner la constitution des classes, nous avons calculé le coefficient λ entre les partitions obtenues par les différentes méthodes de classification et les classes naturelles.
A nouveau, nous aurons beaucoup de zéros car la plupart des méthodes (pratiquement toutes) donnent une partition en une classe.

Nous obtenons :

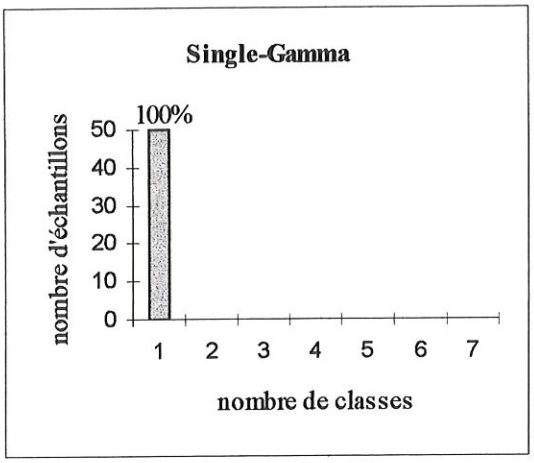
Tab. 1.20. : Coefficient λ pour SOURIRE avec les classes naturelles.

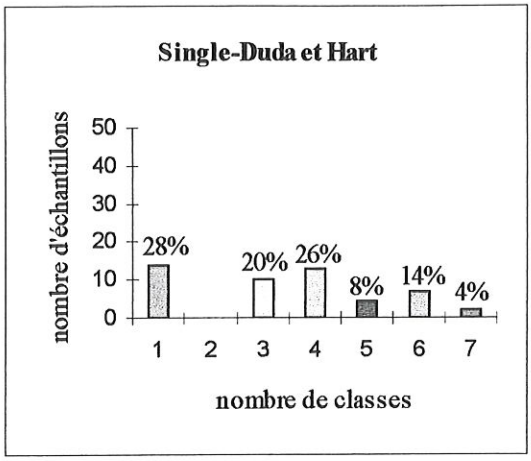
	Gam	D-H	Beale	C-H
Voisin le plus proche	0	0	0	0.8116
Voisin le plus éloigné	0	0	0	0
Moyenne	0	0	0	0
Ward	0	0	0	0

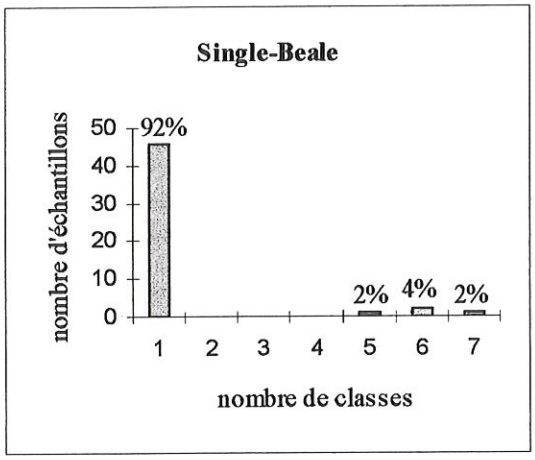
Remarquons que si on demande aux différentes méthodes de classification une partition en deux classes, aucune d'entre elles ne retrouve les classes naturelles (excepté la méthode du voisin le plus proche).

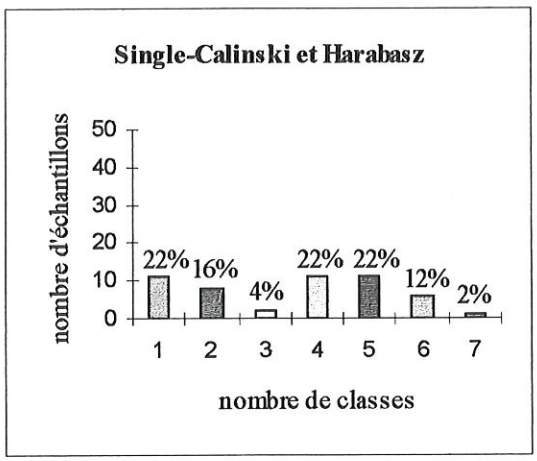
Comme pour les deux jeux de données précédents, nous allons dresser un histogramme des différents nombres de classes obtenus par les différentes méthodes.

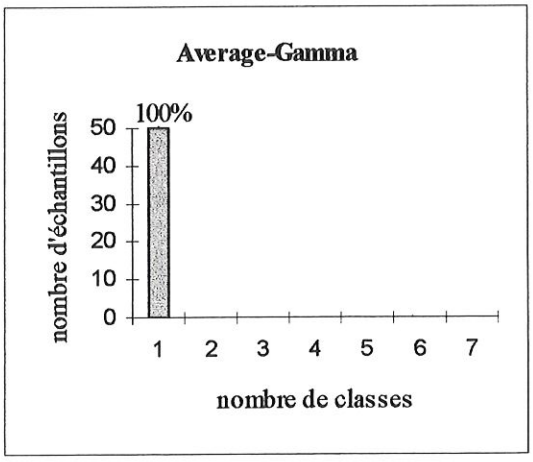
Histogrammes pour SOURIRE

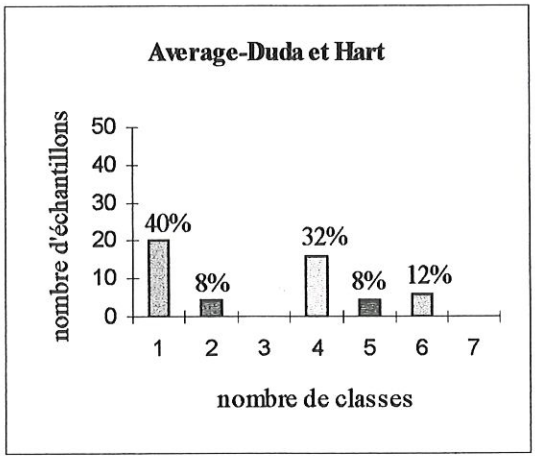


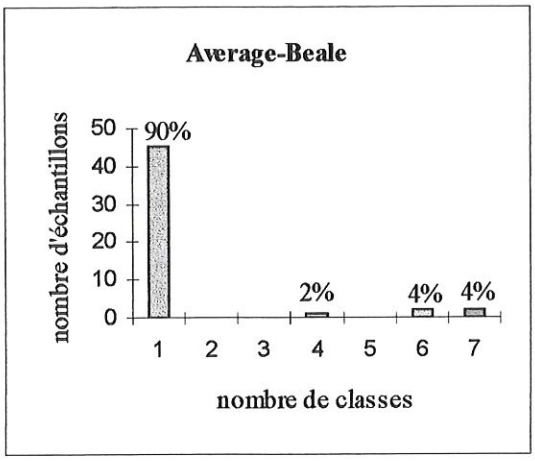


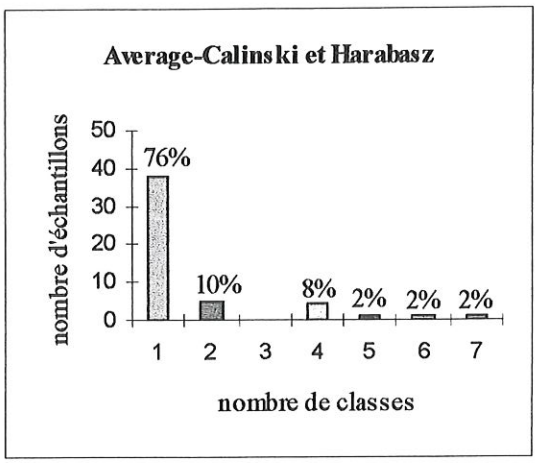


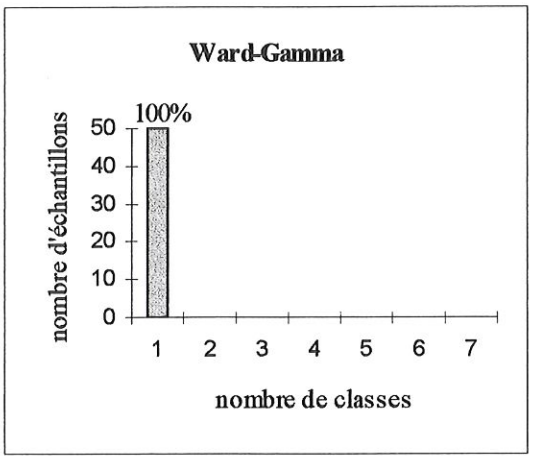


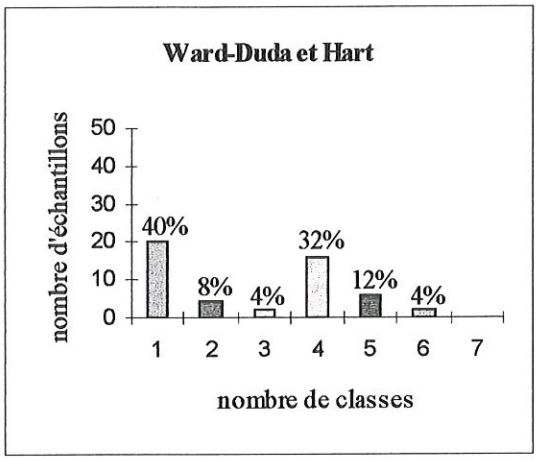


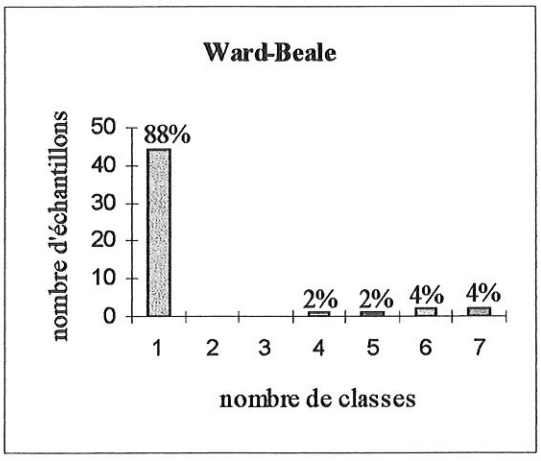


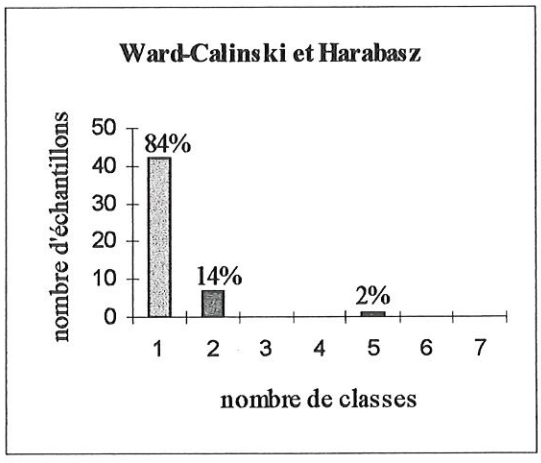

Average-Beale
nombre d'échantillons
50
40
30
20
10
0
90%
2%
4%
4%
1 2 3 4 5 6 7
nombre de classes

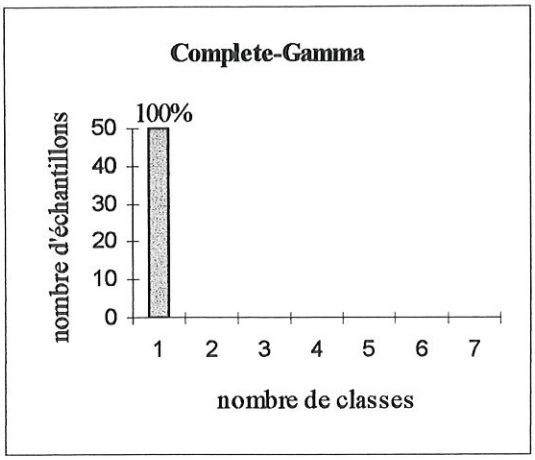

Average-Calinski et Harabasz
nombre d'échantillons
50
40
30
20
10
0
76%
10%
8%
2%
2%
2%
1 2 3 4 5 6 7
nombre de classes

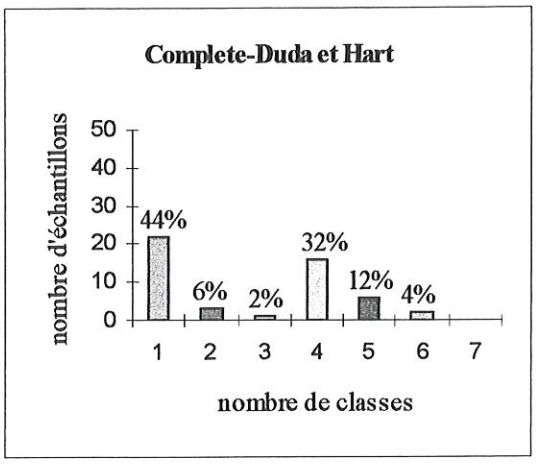

Ward-Gamma
nombre d'échantillons
50
40
30
20
10
0
100%
1 2 3 4 5 6 7
nombre de classes

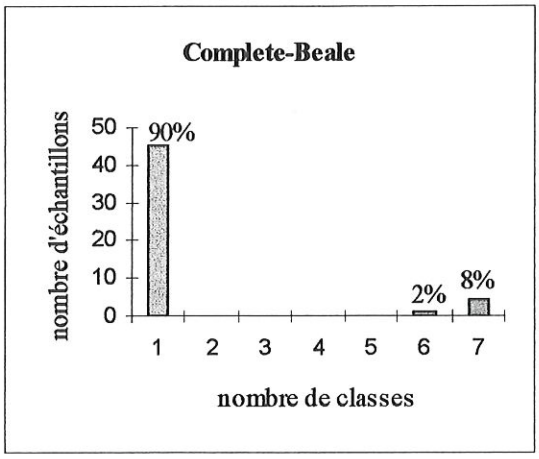

Ward-Duda et Hart
nombre d'échantillons
50
40
30
20
10
0
40%
8%
4%
32%
12%
4%
1 2 3 4 5 6 7
nombre de classes

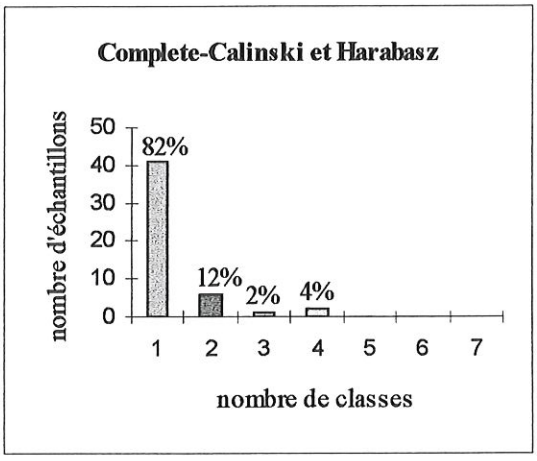

Ward-Beale
nombre d'échantillons
50
40
30
20
10
0
88%
2%
2%
4%
4%
1 2 3 4 5 6 7
nombre de classes


Ward-Calinski et Harabasz
nombre d'échantillons
50
40
30
20
10
0
84%
14%
2%
1 2 3 4 5 6 7
nombre de classes









On constate que les deux méthodes Gamma et Beale donnent dans la plupart des cas une structure en une classe. Quant aux deux autres règles, c'est un peu n'importe quoi. Nous pouvions nous y attendre car les données en sourire sont les plus difficiles à classifier.

Examinons maintenant les 10 échantillons pour les quatre méthodes de classification.

A. Méthode du voisin le plus proche

Tab. 1.21. : Nombres de classes obtenus pour les différents échantillons
Méthode du voisin le plus proche

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	1	4	1	1
2	1	1	1	1
3	1	6	1	1
4	1	7	1	2
5	1	3	1	7
6	1	6	1	1
7	1	1	1	5
8	1	3	1	1
9	1	4	1	1
10	1	1	1	1
Moyenne	1	3.6	1	2.1
E-T	0	2.1071	0	2.0224

A partir de ce tableau, on peut conclure que les méthodes Gamma et Beale sont les plus stables suivies des règles de Calinski-Harabasz et de Duda-Hart.

Tab. 1.22. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode du voisin le plus proche

Règle	Moyenne	Min	Max	E-T
Gam	1	1	1	0
D-H	0.40478	0	1	0.4142
Beale	1	1	1	0
C-H	0.4889	0	1	0.4854

Nous tirons les mêmes conclusions que pour les nombres de classes, c'est-à-dire que les méthodes les plus stables sont Gamma et Beale suivies des méthodes de Calinski-Harabasz et de Duda-Hart.

B. Méthode du voisin le plus éloigné

Tab. 1.23. : Nombres de classes obtenus pour les différents échantillons
Méthode du voisin le plus éloigné

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	1	1	1	1
2	1	4	1	1
3	1	4	1	1
4	1	1	1	1
5	1	5	7	1
6	1	5	1	1
7	1	4	1	1
8	1	5	1	1
9	1	2	1	1
10	1	1	1	1
Moyenne	1	3.2	1.6	1
E-T	0	1.6613	1.8	0

D'après ces résultats, les méthodes les plus stables sont Gamma et Calinski-Harabasz, suivies des méthodes de Beale et de Duda-Hart.

Tab. 1.24. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode du voisin le plus éloigné

Règle	Moyenne	Min	Max	E-T
Gam	1	1	1	0
D-H	0.3539	0	1	0.3966
Beale	0.818	1	1	0.3857
C-H	1	1	1	0

Ce tableau confirme les résultats obtenus pour les nombres de classes :
les méthodes Gamma et Calinski-Harabasz sont les plus stables, suivies des méthodes de Beale et de Duda-Hart.

C. Méthode de la moyenne

Tab. 1.25. : Nombres de classes obtenus pour les différents échantillons
Méthode de la moyenne

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	1	1	1	1
2	1	4	1	1
3	1	4	1	1
4	1	2	1	1
5	1	4	1	1
6	1	5	1	1
7	1	4	1	1
8	1	5	1	1
9	1	2	1	1
10	1	6	1	1
Moyenne	1	3.7	1	1
E-T	0	1.4866	0	0

Ce tableau nous permet de conclure que pour la méthode de la moyenne, les règles Gamma, Beale et Calinski-Harabasz sont les plus stables, la règle de Duda-Hart s'avérant la moins stable.

Confirmons cette conclusion par le calcul du coefficient λ :

Tab. 1.26. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode de la moyenne

Règle	Moyenne	Min	Max	E-T
Gam	1	1	1	0
D-H	0.4508	0	1	0.3671
Beale	1	1	1	0
C-H	1	1	1	0

D. Méthode de Ward

Tab. 1.27. : Nombres de classes obtenus pour les différents échantillons
Méthode de Ward

	Règles d'arrêt			
Echant.	Gam	D-H	Beale	C-H
1	1	1	1	1
2	1	4	1	1
3	1	4	1	1
4	1	2	1	1
5	1	5	7	1
6	1	5	1	1
7	1	4	1	1
8	1	5	1	1
9	1	2	1	1
10	1	1	1	1
Moyenne	1	3.2	1.6	1
E-T	0	1.6149	1.8	0

A partir de là, nous pouvons constater que les règles Gamma et Calinski-Harabasz sont les plus stables. La méthode de Duda-Hart s'avère la moins stable alors que la méthode de Beale occupe une place intermédiaire.

Tab. 1.28. : Coefficient λ entre les classes obtenues pour les différents échantillons
Méthode de Ward

Règle	Moyenne	Min	Max	E-T
Gam	1	1	1	0
D-H	0.3917	0	1	0.3716
Beale	0.818	0	1	0.3857
C-H	1	1	1	0

Nous avons donc que :
les méthodes Gamma et Calinski-Harabasz sont les plus stables, suivies des méthodes de Beale et de Duda-Hart.

Conclusions générales :

Pour les données SOURIRE, nous pouvons classer les règles de la plus stable à la moins stable selon les méthodes de classification.

pour la méthode du voisin le plus proche :

- les méthodes Gamma et Beale
- la méthode de Calinski-Harabasz
- la méthode de Duda-Hart

pour les méthodes du voisin le plus éloigné et de Ward :

- les méthodes Gamma et de Calinski-Harabasz
- la méthode de Beale
- la méthode de Duda-Hart

pour la méthode de la moyenne :

- les méthode Gamma, Calinski-Harabasz et Beale
- la méthode de Duda-Hart

Remarques :

- La méthode Gamma est toujours placée comme étant la plus stable pour toutes les méthodes de classification.
- La règle de Duda-Hart est toujours en dernière position en opposition aux résultats obtenus pour les données de Ruspini.

Conclusions

Ce mémoire a été consacré à l'élaboration d'une méthodologie qui nous a permis d'étudier la stabilité de différentes méthodes de détermination du nombre de classes.
A notre sens, la stabilité est considérée comme l'aptitude de la règle d'arrêt à donner les mêmes résultats quand on l'applique à différents échantillons (générés par un processus bootstrap) choisis aléatoirement parmi les données initiales.

Nous avons alors appliqué notre procédure à trois jeux de données de structures connues et différentes. Ceci étant, nous ne prétendons en aucun cas pouvoir tirer des conclusions générales sur la supériorité d'une méthode ou d'une autre: il faudrait pour cela, tester d'autres jeux de données et peut-être augmenter le nombre déchantillons sur lesquels on travaille.

Les exemples que nous avons traités nous ont permis de mettre le doigt sur les difficultés que pose la détermination du nombre de classes adéquat pour un jeu de données.
Les résultats montrent que les solutions fournies par certaines méthodes changent plus ou moins fortement d'un échantillon à l'autre.

Un comportement particulièrement stable a été observé pour la règle utilisant le coefficient Gamma. Quant à la règle de Duda-Hart, elle semble stable pour les données de Ruspini, mais pas pour les deux autres jeux de données. Donc, le choix d'une méthode de détermination du nombre de classes peut dépendre de la structure des données.
Nous avons le même problème pour la méthode de Beale qui paraît peu stable pour les données de Ruspini et assez stable pour les deux autres jeux de données.

Mais nous ne pouvons nous limiter à des conclusions uniquement sur base de la stabilité, car en effet, une règle peut être stable sans que le résultat qu'elle fournit soit le "bon" (c'est notamment le cas de la règle Gamma pour les jeux de données ALLON et SOURIRE). On peut se demander si une règle n'a pas tendance à sous-estimer ou à surestimer le nombre de classes.

Pour pouvoir détecter et caractériser de tels biais dans une règle, il faut à la fois se baser sur un fondement théorique et sur les résultats fournis en pratique sur des données dont on connaît la structure.

Nous avons pu constater également que les règles d'arrêt sont conditionnées par l'algorithme de classification, par conséquent, on peut conclure que l'une des conditions d'application d'une règle d'arrêt est d'avoir choisi la "bonne" méthode de classification, choix pouvant être basé essentiellement sur la nature des données.

Il serait évidemment intéressant de continuer ce travail en testant d'autres ensembles de données et d'autres méthodes de manière à généraliser nos conclusions. Mais il est certain qu'on ne pourra tirer aucune conclusion qui permettrait de dire qu'une telle méthode est la meilleure. Pour cela, il faudrait un critère qui tienne compte à la fois du nombre de classes, de la constitution des classes, mais aussi du fait d'avoir ou non le "bon" nombre de classes : ce qui est impossible par la subjectivité de la situation.

Annexe A

Partitions et hiérarchies

A.1 Partitions

Une partition de $E = \{x_1, x_2, \ldots, x_n\}$ en k classes $C_1, C_2, \ldots, C_k$ est définie par :

- $C_i \neq \emptyset \;\; i = 1, 2, \ldots, k$
- $C_i \cap C_j \neq \emptyset \;\; i, j = 1, 2, \ldots, k, \;\; i \neq j$
- $\sum_{i=1}^{k} C_i = E.$

Exemple : $E = \{a, b, c\}$
$P = \{C_1, C_2\}$ où $C_1 = \{a\}, C_2 = \{b, c\}$
On notera alors ici $P = a/bc$.

A.2 Familles de partitions

Une famille F de partitions est un ensemble de partitions $P_1, P_2, \ldots, P_n$ indicées par le nombre de classes.

Exemple : $F = \{P_1, P_2, P_3, P_4\}$ où $P_1 = abcd, P_2 = ac/bd, P_3 = ab/c/d$
et $P_4 = a/b/c/d$.

A.3 Hiérarchies de parties

Une hiérarchie H de parties de $E = \{x_1, x_2, \ldots, x_n\}$ est un ensemble de sous-ensembles de E qui vérifie :

- $E \subset H$
- $\forall x_i \in E : \{x_i\} \in H$
- $\forall E_1, E_2 \in H : E_1 \cap E_2 = \emptyset$ ou $E_1 \subset E_2$ ou $E_2 \subset E_1$.

Exemple : $H' = \{P_1, P_2, P_3\}$ où $P_1 = abcd, P_2 = a/bcd, P_3 = a/bc/d$ et $P_4 = a/b/c/d$.

Annexe B

Jeux de données

B.1 Données de Ruspini

45	17	50	23	41	26	74	26	51	30
62	34	63	39	46	37	43	40	51	42
125	42	127	48	127	51	122	51	120	45
131	56	119	55	122	57	118	61	127	62
103	63	116	72	94	88	77	82	100	98
102	100	105	101	90	107	95	108	95	113
2	71	9	62	11	71	13	69	15	68
17	71	20	78	23	68	53	18	65	23
57	26	63	28	54	31	61	38	48	35
49	38	49	41	59	46	126	45	129	47
124	53	120	50	118	47	124	56	118	57
124	53	120	63	130	64	113	68	109	90
78	88	78	79	94	99	97	100	93	102
92	109	103	111	93	115	8	78	11	65
11	79	14	83	15	72	19	66	23	75

B.2 Données ALLON

1.	1.	2.	1.	2.	2.	3.	2.	3.	3.
4.	3.	5.	4.	5.	5.	6.	5.	6.	6.
7.	3.	7.	4.	7.	6.	7.	7.	8.	3.
8.	4.	8.	7.	8.	8.	9.	3.	9.	4.
9.	8.	9.	9.	10.	3.	10.	4.	10.	9.
10.	10.	11.	3.	11.	4.	12.	3.	12.	4.
12.	6.	12.	7.	12.	8.	12.	9.	12.	10.
12.	11.	12.	12.	12.	13.	13.	3.	13.	4.
13.	6.	13.	7.	13.	8.	13.	9.	13.	10.
13.	11.	13.	12.	13.	13.	14.	3.	14.	4.
15.	3.	15.	4.	16.	3.	16.	4.	17.	3.
17.	4.	18.	3.	18.	4.	7.5	3.5	8.5	3.5
9.5	3.5	10.5	3.5	11.5	3.5	12.5	3.5	13.5	3.5
14.5	3.5	15.5	3.5	16.5	3.5	17.5	3.5	12.5	7.5
12.5	8.5	12.5	9.5	12.5	10.5	12.5	11.5	12.5	12.5
12.5	13.5	2.	1.5	3.	2.5	5.	4.5	6.	5.5
7.	6.5	8.	7.5	9.	8.5	4.	3.5	4.	4.

B.3 Données SOURIRE

3.03749	2.08972	3.01959	2.11360	2.84041	2.14835	2.94592	2.32273
2.97207	2.25784	2.97850	2.41149	3.03286	2.51179	2.73136	2.27549
2.72975	2.71330	2.78165	2.62101	2.92549	2.42649	2.88239	2.82270
2.78689	2.63322	2.67921	2.62151	2.72049	2.68161	2.46044	3.03574
2.55941	2.97249	2.43778	2.88818	2.33448	3.04500	2.45439	2.97990
2.40827	3.10743	2.13212	2.87629	2.02023	3.10982	2.05756	2.95561
1.99680	3.04692	2.00106	3.13467	1.91087	2.90987	1.79698	3.01289
1.65148	3.00094	1.68947	3.06231	1.56870	3.10504	1.50411	3.00967
1.44260	2.92068	1.50275	2.69405	1.34608	2.66919	1.34784	2.68142
1.11398	2.75796	1.31790	2.54864	1.33780	2.54085	1.15175	2.51401
0.95454	2.52368	1.24538	2.50266	1.06596	2.40284	1.20864	2.47407
1.15333	2.31807	1.04184	2.15729	0.92931	2.15855	1.01266	2.17444
0.88934	2.09182	1.04967	1.97989	1.97185	2.27974	2.11182	2.25781
2.05393	2.13898	1.88092	2.06591	1.85163	1.98431	2.13474	2.04519
2.01125	2.08862	2.10791	1.69614	2.15545	1.83981	2.28026	1.75617
2.08820	1.57978	2.23043	1.69960	2.36499	1.65885	2.44829	1.50030
2.52634	1.62005	2.18283	1.36992	2.47398	1.53694	2.49632	1.36780
2.53709	1.28101	2.61891	1.32204	2.80921	1.08642	2.69863	1.28269
3.05937	1.33163	2.90662	1.23645	2.89489	1.19962	3.04129	1.26645
3.01197	1.20938	3.17961	1.24294	3.05367	1.08800	3.24804	1.42186
3.63755	1.45096	3.54171	1.30403	3.57400	1.19066	3.65959	1.53642
3.65309	1.36909	3.57058	1.59867	3.63100	1.66048	3.75925	1.79951
3.85134	1.66683	3.94241	1.58877	3.88385	1.82265	3.88180	2.05372
3.73750	1.85008	4.10882	2.00944	3.88437	1.83018	3.82376	2.15118
3.87529	2.17916	3.89741	2.01047	4.08177	2.16492	3.82280	2.33583

Annexe C

Programmes utilisés

C.1 Calcul des indices

```
c*****************************************************************
c*         Calcul des indices correspondant aux methodes Gamma,     *
c*          de Duda-Heart, de Beale, de Calinski-Harabasz,          *
c*                    et du C-index.                                *
c*****************************************************************

c Declarations:
c---------------
      dimension D(150,150),X(150,20),BW(150,150),W(22500),BB(22500)
      DIMENSION Y(150,20)
      dimension A(150),B(150),H(150),P(150),Q(150),TITLE(20),FMT(6)
      dimension DLO(11250),DHI(11250)
      integer BW,A,B,P,Q,QIC,QJC,N,Z,Y
      data PI,EPS,ZD,ZX/3.14159,1.0e-6,'d','x'/

c Les variables seront expliquees au fur et a mesure.

c*****************************************************************

      open(unit=30,file='ruspini.clus',status='old')
      open(unit=40,file='resultat.dat',status='new',err=5)
      goto 6
 5    open(unit=40,file='resultat.dat',status='old')
 6    continue
c Les fichiers ruspini.clus (et resultat.dat) contiennent (contiendront)
c evidemment les donnees a analyser (et les resultats).

c*****************************************************************

c Lecture des donnees:
c----------------------

      read(30,506) (TITLE(J),J=1,20)
      write(40,506) (TITLE(J),J=1,20)

c On lit la premiere ligne du fichier de donnees qui doit comporter
c son nom.
```

```
      read(30,503) F

c On lit la deuxieme ligne du fichier de donnees qui comporte un "d" si
c les donnees sont sous la forme d'une matrice de dissimilarite
c (partie triangulaire inferieure de cette matrice), et
c un "x" si elles sont sous la forme d'une "pattern matrix"
c (une ligne=un objet ; une colonne=une variable caracteristique).

      if (F.eq.ZD) go to 71

c Si la matrice donnee est la matrice de dissimilarite, on passe a 71.
c Sinon on va la calculer :

      read(30,501) N,NP,NHI
      write(40,605) N,NP
c     write(*,*) N,NP
      FP=float(NP)

c     -on commence par lire le nombre de donnees ("N"), le nombre de
c       variables ("NP", puis "FP"), et le nombre de groupes ("NHI") pour
c       lequel on veut afficher la partition obtenue (une serie de nombres
c       indiquant a quel groupe appartient le premier objet, le deuxieme,
c       etc).

      do 72 i=1,N
72    read(30,502) (X(I,K),K=1,NP)
c     DO 2000 I=1,N
c     DO 2000 J=1,NP



c     write(*,*) X(I,J)
c2000 CONTINUE

c     -on lit ensuite la "pattern matrix" (ne pas oublier d'adapter
c        le format (502)).

      call xtod(N,NP,X,D)
      go to 73

c     -on calcule la matrice de dissimilarite ("D=(D(I,J)") grace a la
c        sous-routine "xtod", et on va en 73.
c        Cette matrice est symetrique et a des 0 sur la diagonale.
c     ATTENTION: D(I,J) = distance entre I et J.

c On arrive alors en 71 pour le cas ou la matrice de dissimilarite
c etait donnee. Dans ce cas,

71    read(30,501) N,NHI
      write(40,607) N

c     -on lit le nombre de donnees ("N"), on l'ecrit dans le fichier
c        resultat et on lit le nombre de groupes pour lequel
c        on veut afficher la partition obtenue ("NHI").

      do 74 i=2,N
      I1=I-1
      read(30,fmt) (d(i,j),j=1,i1)
      do 74 j=1,i1
74    d(j,i)=d(i,j)

c     -on lit cette matrice de dissimilarite ("D=(D(I,J)").

73    continue
*****************************************************************
```

```
c Programme:
c------------

      do 121 nclust=1,4

c Chaque valeur de "nclust" represente une methode de classification.
c Donc, pour chacune des quatres methodes, on va faire ce qui suit.

      do 75 i=2,n
      i1=i-1
      do 75 j=1,i1
75    bw(i,j)=0

c On remplit la partie de la matrice "BW" sous la diagonale avec des
c zeros.
c On verra apres ce que represente cette matrice.

      go to (81,82,83,84) nclust
81    write(40,611)
      go to 90
82    write(40,612)
      go to 90
83    write(40,613)
      go to 90
84    write(40,614)
90    write(40,608)

c On ecrit le nom de la methode de classification utilisee dans le
c fichier resultat.

      if(f.eq.zd) write(40,609)



      write(40,610)

c On indique dans le fichier resultat quelle etait la matrice donnee.
c Ensuite, on ecrit les titres des colonnes qui vont contenir les
c resultats (voir format 610).

      ttot=0.0
      k=0

      do 101 i=1,n-1
      do 101 j=i+1,n
      k=k+1
      dlo(k)=d(j,i)
101   ttot=ttot+d(j,i)

      ttot=ttot/float(n)

c On calcule le tableau "DLO" qui contient PROVISOIREMENT les
c dissimilarites de la matrice triangulaire inferieure "D=(D(I,J))"
c prises colonne par colonne.
c De plus, on calcule la moyenne de ces dissimilarites "TTOT".

      nc2=n*(n-1)/2
      call badsrt(dlo,dhi,nc2)

c Soit "NC2", le nombre d'elements de "DLO".
c On trie "DLO" grace a la sous-routine "badsrt".
c DONC, "DLO" contient toutes les dissimilarites triees par
c ordre CROISSANT.
```

```
      wtot=0.0
      dtot=0.0
      ndtot=0
      hold=0.0
      n1=n-1
      k=0

      do 1 i=1,n
      p(i)=i
1     q(i)=1

c On initialise "P" par [1,2,...,N] et "Q" par [1,1,...,1].
c "P" contient des valeurs qui sont les memes (ex: P[2]=2 et P[4]=2)
c     si les points correspondants sont dans la meme classe (les 2eme
c     et 4eme points sont dans la meme classe).
c     ATTENTION: le premier point d'une classe a toujours sa valeur
c     dans "P" (P[2]=2).
c "Q" est lie a "P". Il contient, quand plusieurs points sont dans la
c     meme classe, le nombre de points dans cette classes. Seulement,
c     ce nombre est indique dans la case correspondant au
c     premier de ces elements (dans l'exemple, Q[2]=2 et Q[4]=1).

C++++++++++++++++++++++++++++++++++++++++++++++++
c Tant que K est different de N1=N-1, on fait:
c (car quand K=N1, a un moment, on a "goto 22" qui fait passer
c au dessus de l'instruction "goto 2").

2     k=k+1
      dmax=1.0e20

c Partir de maintenant, on va travailler sur la partie superieure
c de "D=(D(I,J))". Donc "D"=PARTIE SUPERIEURE DE "D" POUR APRES.
c Cette partie va contenir les distances entre les
c points ou les groupes de points formes. En fait, c'est la matrice
c que l'on transforme habituellement au fur et a mesure des etapes
c (et donc des groupements ou des divisions)pour les methodes




c hierarchiques.

      do 4 i=1,n1
      if (p(i).ne.i) go to 4
      i1=i+1
      do 3 j=i1,n
      if (p(j).ne.j) go to 3
      t=d(i,j)
      if (t.gt.dmax) go to 3
      ic=i
      jc=j
      dmax=t
3     continue
4     continue

c On recherche le plus petit element de "D=(D(I,J))" actuel, SAUF
c    parmi ceux regroupes avant (c'est pourquoi on inspecte les
c    valeurs de "P=(P(I))". En fait, on ne regarde que les distances
c    entre les groupes deja formes.
c Cet element est note "DMAX". Il se trouve a la ligne "IC" et a la
c    colonne "JC" dans "D". Ce sont alors les deux classes comprenant
c    "IC" et "JC" respectivement que l'on va regrouper.

      call wgss(d,p,q,n,ic,ssq1,dw1)

c On calcule la somme "DW1" et la moyenne "SSQ1" des dissimilarites
c des elements appartenant a la meme classe que "IC".
```

```
      call wgss(d,p,q,n,jc,ssq2,dw2)

c On calcule la somme "DW2" et la moyenne "SSQ2" des dissimilarites
c des elements appartenant a la meme classe que "JC".

      dwt=dw1+dw2
      nw1=(q(ic)*(q(ic)-1))/2
      nw2=(q(jc)*(q(jc)-1))/2
      nwt=nw1+nw2
      ssqt=ssq1+ssq2
C      write(*,*)'ssqt',ssqt
C      write(*,*)'ssq1,ssq2,',SSQ1,SSQ2
c On calcule:
c       -la somme sur les deux classes regroupees des dissimilarites
c         "intra-classes" "DWT".
c       -la somme sur les deux classes regroupees des moyennes des
c         dissimilarites "intra-classes" "SSQT".
c       -la somme sur les deux classes regroupees des nombres de D(I,J)
c         par classes (4 objets -> 6 D(I,J)).

      do 21 j=1,n
      if (p(j).ne.jc) go to 21
      do 23 i=1,n
      if (p(i).ne.ic) go to 23
      ii=max0(i,j)
      jj=min0(i,j)
      bw(ii,jj)=1
23    continue
      p(j)=ic
21    continue
      a(k)=ic
      b(k)=jc
      h(k)=dmax
      qic=q(ic)
      qjc=q(jc)
      z=qic+qjc
      ff=1.0/float(z)




      do 20 i=1,n
      if (i.eq.ic.or.p(i).ne.i) go to 20
      if (nclust.ne.3) go to 27
      qi=q(i)
      z=qi+qic+qjc
      ff=1.0/float(z)
27    j=min0(ic,i)
      l=max0(ic,i)
      k1=min0(jc,i)
      k2=max0(jc,i)
      dj=d(j,l)
      dk=d(k1,k2)
      go to (111,112,113,114) nclust
c     SINGLE LINK
111   d(j,l)=amin1(dj,dk)
      go to 20
c     GROUP AVERAGE LINK
112   d(j,l)=ff*(qic*dj+qjc*dk)
      go to 20
c     WARD'S METHOD
113   d(j,l)=ff*((qic+qi)*dj+(qjc+qi)*dk-qi*dmax)
      go to 20
c     COMPLETE LINK
114   d(j,l)=amax1(dj,dk)
20    continue
      q(ic)=q(ic)+q(jc)
      call wgss(d,p,q,n,ic,ssq,dw)
      if (ssq.le.(1.0e-6)) goto 2
```

```
      nqc=q(ic)
      nw=(nqc*(nqc-1))/2
      fq=float(nqc)
      if (f.eq.zd) go to 93
      if (ssqt-eps) 93,93,94
94    beale=(ssq-ssqt)/ssqt
      p2=2.0/fp
      div=((fq-1.0)/(fq-2.0))*(2.0**p2)-1.0
      beale=beale/div
      go to 95
93    beale=0.0
95    continue
      if (f.eq.zd) go to 92
      hold=ssqt/ssq-1.0+2.0/(pi*fp)
      hold2=(2.0-16.0/(pi*pi*fp))/(fp*fq)
      hold=-hold/sqrt(hold2)
92    wtot=wtot+ssq-ssqt
      btot=ttot-wtot
      if (k.eq.n1) go to 22
      z=n-k-1
      ch=btot*float(k)/(wtot*float(z))
      dtot=dtot+dw-dwt
      ndtot=ndtot+nw-nwt
      cind=(dtot-dlo(ndtot))/dhi(ndtot)
      kin=0
      kout=0
      do 31 i=2,n
      i1=i-1
      do 31 j=1,i1
      if (bw(i,j)) 32,32,33
32    kout=kout+1
      bb(kout)=d(i,j)
      go to 31
33    kin=kin+1
      w(kin)=d(i,j)
31    continue
      u=0.0
      dd=0.0



      do 51 kk=1,kin
      din=w(kk)
      do 51 jj=1,kout
      if (din-bb(jj)) 51,52,53
52    dd=dd+0.5
      go to 51
53    u=u+1.0
51    continue
      denmr=float(kin)*float(kout)/2.0-dd
      gamma=1.0-u/denmr
      ngp=n-k
      write(40,601) ngp,h(k),a(k),b(k),gamma,hold,beale,ch,SSQ
      if (ngp-12) 41,42,41
41    if (ngp-nhi) 2,42,2
42    write(40,602) (p(ll),ll=1,n)
      go to 2
C++++++++++++++++++++++++++++++++++++++
```

```
22      ngp=n-k
        write(40,616) ngp,h(k),a(k),b(k),hold,beale
121     continue
501     format(6i4)
502     format(f3.0,1x,f3.0)
503     format(a1)
506     format(20a4)
601     format(i4,f10.3,2i4,2x,f10.6,2x,2f10.6,f12.6,f10.6)
602     format(15i4)
603     format(10f8.1)
604     format(10f8.5)
605     format('data provided as pattern matrix',/,'number of objects=',
       1   i4,'  number of dimensions=',i4,/)
606     format(i4,10f8.4/(4x,10f8.4))
607     format('data provided as dissimilarity matrix',/,
       1  'number of objects=',i4,/)
608     format(/,' evaluation of partition gamma, duda-heart, beale,'
       1  ' calinski-harabasz',/,'   and c statistics',/)
609     format(/,' data not allow evaluation of duda-heart and beale'
       1  ' statistics')
610     format(/,' ngps   height  leaders',
       1  '     gamma    duda-heart   beale    c-h          c')
611     format(/,' single link analysis')
612     format(/,' group average link analysis')
613     format(/,' wards method analysis')
614     format(/,' complete link analysis')
616     format(i4,f10.3,2i4,14x,2f10.6)
        close(30)
        close(40)
        stop
        end

        subroutine xtod(n,np,x,d)
        dimension x(150,20),d(150,150)
        do 1 i=2,n
        i1=i-1
        do 3 j=1,i1
        tot=0.0
        do 2 k=1,np
        z=x(i,k)-x(j,k)
2       tot=tot+z*z
        d(i,j)=tot
3       d(j,i)=d(i,j)
1       continue
        return
        end

        subroutine wgss(d,p,q,n,ic,ssq,dw)

        dimensiond(150,150),p(150),q(150)
        integer p,q
        ssq=0.0
        dw=0.0
        if (q(ic).le.1) return
        do 1 i=1,n-1
        if (p(i).ne.ic) go to 1
        do 2 j=i+1,n
        if (p(j).ne.ic) go to 2
        ssq=ssq+d(j,i)
2       continue
1       continue
        dw=ssq
        ssq=ssq/q(ic)
        return
        end
```

```
      subroutine badsrt(dlo,dhi,nc2)
      dimension dlo(11250),dhi(11250)
      do 1 i=1,nc2-1
      small=dlo(i)
      do 1 j=i+1,nc2
      if (small-dlo(j)) 1,1,2
2     dlo(i)=dlo(j)
      dlo(j)=small
      small=dlo(i)
1     continue
      dhi(1)=dlo(nc2)
      do 4 i=2,nc2
4     dhi(i)=dhi(i-1)+dlo(nc2-i+1)
      do 5 i=2,nc2
5     dlo(i)=dlo(i-1)+dlo(i)
      do 3 i=1,nc2
3     dhi(i)=dhi(i)-dlo(i)
      return
      end
```

C.2 Formation des échantillons

```
C*************************************************************
C*    Formation d;echantillons de meme effectif que          *
C*    l'ensemble de donnees initial et dont certains         *
C*    points seront repetes et d'autres omis.                *
C*************************************************************

C Declarations:
C -------------

      dimension x(150,20),Y(150,20)
      dimension TITLE(20)
      integer N,NP,NHI
      integer M(1)

C ---------------------------------------------------------------

      open(unit=30,file='ruspini.clus',status='old')
      open(unit=40,file='ruspini1.clus',status='new',err=5)
      goto 6
 5    open(unit=40,file='ruspini1.clus',status='old')
 6    continue

C ruspini.clus est le fichier initial. Le fichier ruspini1.clus
C contiendra un echantillon de meme effectif que ruspini.clus mais
C dont certains points sont repetes et d'autres omis.

C Lecture des donnees:
C --------------------

      read(30,506) (TITLE(J),J=1,20)
      write(40,506) (TITLE(J),J=1,20)
      read(30,503) F
      write(40,503) F
      read(30,501) N,NP,NHI
      write(40,501) N,NP,NHI
      FP=float(NP)

      call rnset(234)

C On appelle la table de nombres aleatoires 234

      do 72 i=1,N
 72      read(30,502) (X(I,K),K=1,NP)

      do 100 l=1,N
         call rnsri(1,75,M)
 100     write(40,502) (X(M(1),K),K=1,NP)

C On tire un nombre aleatoire (ici dans la table 234) inferieur ou
C egal a 75 et on le stocke dans M. Ensuite, on va chercher la donnee
C correspondant a M dans le fichier initial et on la retrnscrit dans
C le fichier de sortie.

 501  format(6i4)
 502  format(f4.1,1x,f4.1)
 503  format(a1)
 506  format(20a4)

      end
```

C.3 Renumération des échantillons

```
C*****************************************************************
C*   Renumeration des echantillons pour pouvoir encoder          *
C*   les partitions.                                             *
C*   Ce programme compare un echantillon avec l'ensemble         *
C*   de donnees initial de maniere a numeroter les donnees       *
C*   de l'echantillon par rapport aux numeros de l'ensemble      *
C*   initial.                                                    *
C*****************************************************************


C     declarations
C     ------------

      integer X(150,20),Y(150,20)
      integer N,NP

C     ouverture des fichiers
C     ----------------------

      open(unit=30,file='ruspini.clus',status='old')
      open(unit=40,file='ruspini1.clus',status='old')
      open(unit=50,file='classeru1.dat',status='new',err=5)
      goto 6
 5    open(unit=50,file='classeru1.dat',status='old')
 6    continue


C     lecture des donnees
C     -------------------

      N=75
      NP=3

      do 72 I=1,N
         read(30,501)  (X(I,K),K=1,NP)
         read(40,502)  (Y(I,K),K=1,2)
 72   continue


C     programme
C     ---------


      do 73 I=1,N
         do 74 J=1,N
            if ((Y(I,1)).eq.(X(J,1))) then
               if ((Y(I,2)).eq.(X(J,2))) then
                  write(50,500) X(J,3)
               endif
            endif
 74      continue
 73   continue



 500  format(i3)
 501  format(f7.5,1x,f7.5,1x,i4)
 502  format(f7.5,1x,f7.5)

      end
```

C.4 Calcul du coefficient λ

```
C*****************************************************************
C*        Calcul du coefficient lambda (degre d'association      *
C*        entre 2 classes)                                       *
C*****************************************************************


C     declaration
C     -----------
      implicit none
      integer X(150,20),Y(150,20),Z(7,7),W(75),U(75)
      integer N,NP,I,J,K,M
      integer N1,N2,N3,N4,N5,N6,N7,M1,M2,M3,M4,M5,M6,M7,R1,R2,R3,R4
      integer R5,R6,R7,C1,C2,C3,C4,C5,C6,C7,R,C,NBR
      real lambda
      real a,b

C     N1=N1.= la somme des elements de la premiere ligne de la table
C             de contingence
C     N2=N2.= la somme des elements de la deuxieme ligne de la table
C     N3=N3.= la somme des elements de la troisieme ligne de la table
C     N4=N4.= la somme des elements de la quatrieme ligne de la table
C     M1=N.1= la somme des elements de la premiere colonne de la table
C     M2=N.2= la somme des elements de la deuxieme colonne
C     M3=N.3= la somme des elements de la troisieme colonne
C     M4=N.4= la somme des elements de la quatrieme colonne

C     ouverture des fichiers
C     ----------------------

      open(unit=30,file='classeru.dat',status='old')
      open(unit=40,file='classeru1.dat',status='old')
      open(unit=50,file='coefflambda.dat',status='new',err=5)
      goto 6
 5    open(unit=50,file='coefflambda.dat',status='old')
 6    continue

C Le fichier classeru.dat contient la partition en k classes (k variant
C d'une methode a l'autre) de RUSPINI et le fichier classeru1.dat
C contient la partition en k' classes (k' variant d'une methode a l'autre)
C Le fichier coefflambda.dat contiendra la valeur du coefficient lambda
C pour classeru.dat et classeru1.dat

C     lecture des donnees
C     -------------------

      N=75
      NP=2

      do 72 I=1,N
         read(30,*) (X(I,K),K=1,NP)
         read(40,*) (Y(I,K),K=1,NP)
 72   continue

C On commence par lire les 2 fichiers d'entree qui donnent les partitions
C a comparer

C     initialisations
C     ---------------

      do 10 J=1,7
         do 20 M=1,7
            Z(J,M)=0
 20      continue
 10   continue

      do 11 I=1,N
```

```
          W(I)=0
          U(I)=0
11      continue

        M1=0
        M2=0
        M3=0
        M4=0
        M5=0
        M6=0
        M7=0
        N1=0
        N2=0
        N3=0
        N4=0
        N5=0
        N6=0
        N7=0
        NBR=0

C On initialise la table de contingence et d'autres variables

C       programme
C       ---------

        do 73 I=1,N
           do 76 J=1,N
              if ((X(I,1).eq.Y(J,1)).and.(W(I).eq.0).and.
     $             (U(J).eq.0)) then
                 Z(X(I,2),Y(J,2))=Z(X(I,2),Y(J,2))+1
                 NBR=NBR+1
                 W(I)=1
                 U(J)=1
              endif
 76        continue
 73      continue

C On remplit la table de contingence et on calcule le nombre total
C de points presents dans les deux partitions. Par la suite, on
C calculera toutes les variables necessaires au calcul du coefficient
C lambda

        do 74 J=1,7
           M1=M1+Z(J,1)
           M2=M2+Z(J,2)
           M3=M3+Z(J,3)
           M4=M4+Z(J,4)
           M5=M5+Z(J,5)
           M6=M6+Z(J,6)
           M7=M7+Z(J,7)
  74     continue


        do 75 I=1,7
           N1=N1+Z(1,I)
           N2=N2+Z(2,I)
           N3=N3+Z(3,I)
           N4=N4+Z(4,I)
           N5=N5+Z(5,I)
           N6=N6+Z(6,I)
           N7=N7+Z(7,I)
  75     continue


        R1=max(Z(1,1),Z(1,2),Z(1,3),Z(1,4),Z(1,5),Z(1,6),Z(1,7))
```

```
      R2=max(Z(2,1),Z(2,2),Z(2,3),Z(2,4),Z(2,5),Z(2,6),Z(2,7))
      R3=max(Z(3,1),Z(3,2),Z(3,3),Z(3,4),Z(3,5),Z(3,6),Z(3,7))
      R4=max(Z(4,1),Z(4,2),Z(4,3),Z(4,4),Z(4,5),Z(4,6),Z(4,7))
      R5=max(Z(5,1),Z(5,2),Z(5,3),Z(5,4),Z(5,5),Z(5,6),Z(5,7))
      R6=max(Z(6,1),Z(6,2),Z(6,3),Z(6,4),Z(6,5),Z(6,6),Z(6,7))
      R7=max(Z(7,1),Z(7,2),Z(7,3),Z(7,4),Z(7,5),Z(7,6),Z(7,7))


      C1=max(Z(1,1),Z(2,1),Z(3,1),Z(4,1),Z(5,1),Z(6,1),Z(7,1))
      C2=max(Z(1,2),Z(2,2),Z(3,2),Z(4,2),Z(5,2),Z(6,2),Z(7,2))
      C3=max(Z(1,3),Z(2,3),Z(3,3),Z(4,3),Z(5,3),Z(6,3),Z(7,3))
      C4=max(Z(1,4),Z(2,4),Z(3,4),Z(4,4),Z(5,4),Z(6,4),Z(7,4))
      C5=max(Z(1,5),Z(2,5),Z(3,5),Z(4,5),Z(5,5),Z(6,5),Z(7,5))
      C6=max(Z(1,6),Z(2,6),Z(3,6),Z(4,6),Z(5,6),Z(6,6),Z(7,6))
      C7=max(Z(1,7),Z(2,7),Z(3,7),Z(4,7),Z(5,7),Z(6,7),Z(7,7))


      R=max(M1,M2,M3,M4,M5,M6,M7)
      C=max(N1,N2,N3,N4,N5,N6,N7)


      a=R1+R2+R3+R4+R5+R6+R7+C1+C2+C3+C4+C5+C6+C7-R-C
      b=2*NBR-R-C

      lambda=a/b

      write(50,503) lambda


503   format('lambda= ',f6.4)
      close(30)
      close(40)
      close(50)
      stop

      end
```

Annexe D

Calculs du coefficient λ

D.1 Pour les données de Ruspini

Single-Gamma

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	0.9524	0.9464	0.95	0.975	0.9412					
6	1	1	1	1	1	0.92				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	0.9524	1	1		
9	1	1	1	1	1	0.9231	1	1	1	
10	0.9714	1	1	1	0.9636	0.9615	0.96	0.9565	0.9565	0.9661

Single-Duda-Hart

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	0.9714	1	1	1	0.9636	1	0.96	0.9565	0.9565	0.9661

Single-Beale

	R	1	2	3	4	5	6	7	8	9
1	1									
2	0.7347	0.7333								
3	1	1	0.64							
4	0.7925	0.7568	1	0.8205						
5	0.7347	0.65	1	0.5517	1					
6	0.9706	0.9565	0.7778	0.9444	0.8478	0.7317				
7	1	1	0.7317	1	0.8889	0.7586	1			
8	1	1	0.6667	1	0.8	0.7097	0.98	1		
9	1	1	0.7368	1	0.82	0.8293	0.9333	1	1	
10	0.9714	1	0.8205	1	0.8163	0.8182	0.92	0.9565	0.9565	0.9661

Single-Calinski-Harabasz

	R	1	2	3	4	5	6	7	8	9
1	1									
2	0.9677	0.95								
3	1	1	0.9412							
4	1	1	0.9773	1						
5	0.8571	0.8214	0.9091	0.85	0.8462					
6	0.9706	0.9565	1	0.9444	0.9615	0.9231				
7	1	1	1	1	1	0.9167	1			
8	0.9697	0.9583	1	1	0.9583	0.913	1	1		
9	0.9722	0.9636	1	0.9783	0.9643	0.9107	1	1	1	
10	0.9714	1	0.9787	1	0.9636	0.9038	0.92	0.9565	0.9348	0.9322

Complete-Gamma

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	0.9524	0.9464	0.95	0.975	0.9412					
6	1	1	1	1	1	0.92				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	0.9524	1	1		
9	1	1	1	1	1	0.9231	1	1	1	
10	0.9714	1	1	1	0.9636	0.9615	0.96	0.9565	0.9565	0.9661

Complete-Duda-Hart

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	0.9714	1	1	1	0.9636	1	0.96	0.9565	0.9565	0.9661

Complete-Beale

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	0.7925	0.7568	0.7429	0.8205						
5	1	1	1	1	0.75					
6	0.9706	0.9565	0.9545	0.9444	0.8478	0.92				
7	1	1	1	1	0.8889	1	1			
8	1	1	1	1	0.8	1	0.98	1		
9	1	1	1	1	0.82	1	0.9333	1	1	
10	0.9714	1	1	1	0.8163	1	0.92	0.9565	0.9565	0.9661

Complete-Calinski-Harabasz

	R	1	2	3	4	5	6	7	8	9
1	1									
2	0.9677	0.95								
3	1	1	0.9412							
4	1	1	0.9773	1						
5	1	1	0.95	1	1					
6	0.9706	0.9565	1	0.9444	0.9615	0.92				
7	1	1	1	1	1	1	1			
8	0.9697	0.9583	1	1	0.9583	0.9524	1	1		
9	1	1	0.9796	1	1	1	0.9333	1	0.9808	
10	0.9714	1	0.9787	1	0.9636	1	0.92	0.9565	0.9348	0.9661

Average-Gamma

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	0.9524	0.9464	0.95	0.975	0.9412					
6	1	1	1	1	1	0.92				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	0.9524	1	1		
9	1	1	1	1	1	0.9231	1	1	1	
10	0.9714	1	1	1	0.9636	0.9615	0.96	0.9565	0.9565	0.9661

Average-Duda-Hart

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	0.9714	1	1	1	0.9636	1	0.96	0.9565	0.9565	0.9661

Average-Beale

	R	1	2	3	4	5	6	7	8	9
1	1									
2	0.7347	0.7333								
3	1	1	0.64							
4	0.7925	0.7568	1	0.8205						
5	0.7347	0.65	1	0.5517	1					
6	0.9706	0.9565	0.7778	0.9444	0.8478	0.7317				
7	1	1	0.7317	1	0.8889	0.7586	1			
8	1	1	0.6667	1	0.8	0.7097	0.98	1		
9	1	1	0.7368	1	0.82	0.8293	0.9333	1	1	
10	0.9714	1	0.8205	1	0.8163	0.8182	0.92	0.9565	0.9565	0.9661

Average-Calinski-Harabasz

	R	1	2	3	4	5	6	7	8	9
1	1									
2	0.9677	0.95								
3	1	1	0.9412							
4	1	1	0.9773	1						
5	0.9524	0.9464	1	0.975	0.9412					
6	0.9706	0.9565	1	0.9444	0.9615	1				
7	1	1	1	1	1	1	1			
8	0.9697	0.9583	1	1	0.9583	1	1	1		
9	0.9722	0.9636	1	0.9783	0.9643	1	1	1	1	
10	0.9714	1	0.9787	1	0.9636	0.9615	0.92	0.9565	0.9348	0.9322

Ward-Gamma

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	0.9524	0.9464	0.95	0.975	0.9412					
6	1	1	1	1	1	0.92				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	0.9524	1	1		
9	1	1	1	1	1	0.9231	1	1	1	
10	0.9714	1	1	1	0.9636	0.9615	0.96	0.9565	0.9565	0.9661

Ward-Duda-Hart

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	0.9714	1	1	1	0.9636	1	0.96	0.9565	0.9565	0.9661

Ward-Beale

	R	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	0.7925	0.7568	0.7429	0.8205						
5	1	1	1	1	0.75					
6	0.9706	0.9565	0.9545	0.9444	0.8478	0.92				
7	1	1	1	1	0.8889	1	1			
8	1	1	1	1	0.8	1	0.98	1		
9	1	1	1	1	0.82	1	0.9333	1	1	
10	0.9714	1	1	1	0.8163	1	0.92	0.9565	0.9565	0.9661

Ward-Calinski-Harabasz

	R	1	2	3	4	5	6	7	8	9
1	1									
2	0.9677	0.95								
3	1	1	0.9412							
4	1	1	0.9773	1						
5	1	1	0.95	1	1					
6	0.9706	0.9565	1	0.9444	0.9615	0.92				
7	1	1	1	1	1	1	1			
8	0.9697	0.9583	1	1	0.9583	0.9524	1	1		
9	1	1	0.9796	1	1	1	0.9333	1	0.9808	
10	0.9714	1	0.9787	1	0.9636	1	0.92	0.9565	0.9348	0.9661

D.2 Pour les données ALLON

Remarque : pour ce jeu de données, il arrive souvent que l'on compare deux partitions en une classe. Dès lors, λ est indéterminé, mais nous le prendrons à un puisque l'association entre deux partitions en une classe ne peut être que parfaite. Si vous rencontrez un 1*, cela veut dire que c'est la vraie valeur de λ et qu'il n'est donc pas dû à deux partitions en une classe.

Single-Gamma

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Single-Duda-Hart

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	0	0	0	0	0					
6	0	0	0	0	0	0.7818				
7	1	1	1	1	1	0	0			
8	0	0	0	0	0	0.7755	0.7083	0		
9	1	1	1	1	1	0.8689	0.7959	0	0.902	
10	0	0	0	0	0	0.8824	0.7447	0	0.8085	0.9804

Single-Beale

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Single-Calinski-Harabasz

	A	1	2	3	4	5	6	7	8	9
1	0.6458									
2	0	0								
3	0	0	1							
4	0.931	0.4545	0	0						
5	0.7419	0.6316	0	0	0.825					
6	0	0	1	1	0	0				
7	1*	0.5897	0	0	0.9545	0.7045	0			
8	0	0	1	1	0	0	1	0		
9	0.7813	0.5484	0	0	0.8077	0.7705	0	0.7727	0	
10	0.7763	0.5263	0	0	0.7447	0.8824	0	0.7451	0	0.8868

Complete-Gamma

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Complete-Duda-Hart

	A	1	2	3	4	5	6	7	8	9
1	0									
2	0.6429	0								
3	0.8723	0	0.7021							
4	0.8039	0	0.6667	0.9459						
5	0.6607	0	0.75	0.6531	0.5238					
6	0.5254	0	0.6531	0.5455	0.5417	0.7778				
7	1*	0	0.5116	0.9462	0.8378	0.6429	0.6			
8	0.9259	0	0.6341	1*	1*	0.6304	0.4565	0.9		
9	0.5472	0	0.8519	0.5588	0.7826	0.6949	0.75	0.5526	0.7083	
10	0.8667	0	0.7	0.7931	0.5806	0.675	0.575	0.8667	0.8065	0.5556

Complete-Beale

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	0								
3	1	0	1							
4	1	0	1	1						
5	1	0	1	1	1					
6	1	0	1	1	1	1				
7	1	0	1	1	1	1	1			
8	1	0	1	1	1	1	1	1		
9	1	0	1	1	1	1	1	1	1	
10	1	0	1	1	1	1	1	1	1	1

Complete-Calinski-Harabasz

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Average-Gamma

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Average-Duda-Hart

	A	1	2	3	4	5	6	7	8	9
1	0									
2	1	1								
3	0	0	0							
4	0	0	0	0.9459						
5	0	0	0	0.6364	0.6667					
6	1	1	1	0	0	0				
7	0	0	0	0.8462	0.8378	0.9259	0			
8	0	0	0	1*	1*	0.6333	0	0.9		
9	0	0	0	0.76	0.9459	0.8824	0	0.7778	0.8947	
10	1	1	1	0	0	0	1	0	0	0

Average-Beale

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Average-Calinski-Harabasz

	A	1	2	3	4	5	6	7	8	9
1	0									
2	1	0								
3	1	0	1							
4	1	0	1	1						
5	1	0	1	1	1					
6	1	0	1	1	1	1				
7	1	0	1	1	1	1	1			
8	1	0	1	1	1	1	1	1		
9	1	0	1	1	1	1	1	1	1	
10	1	0	1	1	1	1	1	1	1	1

Ward-Gamma

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Ward-Duda-Hart

	A	1	2	3	4	5	6	7	8	9
1	0									
2	0.6429	0								
3	0.8723	0	0.7021							
4	0.8039	0	0.6667	0.9459						
5	0.6607	0	0.75	0.6531	0.5238					
6	0.5254	0	0.6531	0.5455	0.5417	0.7778				
7	1*	0	0.5116	0.8462	0.8378	0.6429	0.6			
8	0.9259	0	0.6341	1*	1*	0.6304	0.4565	0.9		
9	0.5472	0	0.8519	0.5588	0.7826	0.6949	0.75	0.5526	0.7083	
10	0	1	0	0	0	0	0	0	0	0

Ward-Beale

	A	1	2	3	4	5	6	7	8	9
1	0									
2	1	0								
3	1	0	1							
4	1	0	1	1						
5	1	0	1	1	1					
6	1	0	1	1	1	1				
7	1	0	1	1	1	1	1			
8	1	0	1	1	1	1	1	1		
9	1	0	1	1	1	1	1	1	1	
10	1	0	1	1	1	1	1	1	1	1

Ward-Calinski-Harabasz

	A	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

D.3 Pour les données SOURIRE

Remarque : pour ce jeu de données, il arrive souvent que l'on compare deux partitions en une classe. Dès lors, λ est indéterminé, mais nous le prendrons à un puisque l'association entre deux partitions en une classe ne peut être que parfaite. Si vous rencontrez un 1*, cela veut dire que c'est la vraie valeur de λ et qu'il n'est donc pas $d\hat{u}$ à deux partitions en une classe.

Single-Gamma

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Single-Duda-Hart

	S	1	2	3	4	5	6	7	8	9
1	1*									
2	1	0								
3	0	0.8095	0							
4	0	0.6429	0	0.907						
5	0	0.5526	0	0.8372	0.6744					
6	0	0.7547	0	0.9074	0.7105	0.6216				
7	1	0	1	0	0	0	0			
8	0	0.9762	0	0.6341	0.6757	0.6053	0.7308	0		
9	0	0.5429	0	0.75	0.6667	0.9189	0.8293	0	0.5152	
10	1	0	1	0	0	0	0	1	0	0

Single-Beale

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Single-Calinski-Harabasz

	S	1	2	3	4	5	6	7	8	9
1	1*									
2	0	1								
3	0	1	1							
4	0.6471	0	0	0						
5	0.9577	0	0	0	0.6512					
6	0	1	1	1	0	0				
7	0.8983	0	0	0	0.7353	1*	0			
8	0	1	1	1	0	0	1	0		
9	0	1	1	1	0	0	1	0	1	
10	0	1	1	1	0	0	1	0	1	1

Complete-Gamma

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Complete-Duda-Hart

	S	1	2	3	4	5	6	7	8	9
1	1									
2	0	1								
3	0	0	0.4595							
4	1	1	0	0						
5	0	0	0.7826	0.8462	0					
6	1	1	0.7609	0.5476	0	1				
7	0	0	0.6848	0.6429	0	0.7381	0.7727			
8	0	0	0.8727	0.8298	0	0.7358	0.6607	0.8627		
9	0	0	0.3438	0.5833	0	0.625	0.7353	0.3256	0.6579	
10	1	1	0	0	0	0	0	0	0	0

Complete-Beale

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	0	0	0	0	0					
6	1	1	1	1	1	0				
7	1	1	1	1	1	0	1			
8	1	1	1	1	1	0	1	1		
9	1	1	1	1	1	0	1	1	1	
10	1	1	1	1	1	0	1	1	1	1

Complete-Calinski-Harabasz

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Average-Gamma

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Average-Duda-Hart

	S	1	2	3	4	5	6	7	8	9
1	1									
2	0	0								
3	0	0	0.525							
4	0	0	0.7273	0.6944						
5	0	0	0.425	0.8261	0.6316					
6	0	0	0.7442	0.5952	0.5417	0.6744				
7	0	0	0.9762	0.6429	0.7188	0.5263	0.7727			
8	0	0	0.7447	0.6818	0.3946	0.7917	0.6607	0.8627		
9	0	0	0.4474	0.6389	0.6774	0.5	0.7353	0.3256	0.6579	
10	0	0	0.7647	0.6042	0.5745	0.7679	0.8364	0.7609	0.8367	0.5217

Average-Beale

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Average-Calinski-Harabasz

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Ward-Gamma

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Ward-Duda-Hart

	S	1	2	3	4	5	6	7	8	9
1	1									
2	0	0								
3	0	0	0.4595							
4	0	0	0.4138	0.7222						
5	0	0	0.7826	0.8462	0.6216					
6	0	0	0.7609	0.5476	0.5417	1				
7	0	0	0.6818	0.6429	0.7188	0.7381	0.7727			
8	0	0	0.8727	0.8292	0.3846	0.7358	0.6607	0.8627		
9	0	0	0.3438	0.5833	0.6774	0.625	0.7353	0.3256	0.6579	
10	1	1	0	0	0	0	0	0	0	0

Ward-Beale

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	0	0	0	0	0					
6	1	1	1	1	1	0				
7	1	1	1	1	1	0	1			
8	1	1	1	1	1	0	1	1		
9	1	1	1	1	1	0	1	1	1	
10	1	1	1	1	1	0	1	1	1	1

Ward-Calinski-Harabasz

	S	1	2	3	4	5	6	7	8	9
1	1									
2	1	1								
3	1	1	1							
4	1	1	1	1						
5	1	1	1	1	1					
6	1	1	1	1	1	1				
7	1	1	1	1	1	1	1			
8	1	1	1	1	1	1	1	1		
9	1	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	1	1	1	1	1

Bibliographie

[1] G.J. BABU et K. SINGH, Inference on means using the Bootstrap, *Annals of Statistics*, 1983, Vol. 11, N.3, 999-1003.

[2] F.B. BAKER, Stability of Two Hierarchical Grouping Techniques. Case I: Sensitivity to Data Errors, *Journal of the American Statistical Association*, 1974, Vol.69, N.346, 440-445.

[3] E.M.L. BEALE, Cluster Analysis, London : Scientific Control Systems, 1969.

[4] R.B. CALINSKI et J. HARABASZ, A Dendrite Method for Cluster Analysis, *Communications in Statistics*,1974, 3, 1-27.

[5] J.L. CHANDON et S. PINSON, Analyse typologique : théorie et applications, Masson, Paris, 1981.

[6] H. CRAMER, Mathematical Methods of Statistics, Princeton, Princeton University Press, 1946.

[7] R.O. DUDA et P.E. HART, Pattern Classification and Scene Analysis, Wiley-Interscience, New York, 1973.

[8] A.C. DAVISON, D.V. HINKLEY et E. SCHECHTMAN, Efficient Bootstrap Simulation, *Biometrika*, 1986, Vol.73, N.3, 555-566.

[9] B. EFRON et R.J. TIBSHIRANI, An introduction to the Bootstrap, Monographs on Statistics and Applied Probability 57, Chapman & Hall, International Thomson Publishing, 1993.

[10] B. EFRON, The Jackknife, the Bootstrap and Other Resampling Plans, Society for Industrial and Applied Mathematics, Philadelphia, pennsylvania, 1982.

[11] B. EFRON, Bootstrap Methods : Another Look at the Jackknife, *Annals of Statistics*, 1979, Vol.7, N.1, 1-26.

[12] B. EVERITT, Cluster Analysis, Halsted press, London,1980.

[13] R.A. FISHER, Statistical Methods for research Workers, Tenth Edition, New York, Hafner publishing Co, 1948.

[14] L. FISHER et J.W. VAN NESS, Admissible Clustering Procedures, *Biometrika*, 1971, 58, 91-100.

[15] L.A. GOODMAN et W.H. KRUSKAL, Measures of Association for Cross Classifications, Springer Series in Statistics, Springer Verlag, New York, Heidelberg Berlin, 1954, 1959, 1963, 1972.

[16] A.D. GORDON, How many clusters? An investigation of five Procedures for detecting nested Cluster Structure, *Data Sciences, Classification and related methods*, C. Hazashi et al., 1998, 109-116.

[17] P. HALL, On the Number of Bootstrap Simulations required to construct a Confidence Interval, *Annals of Statistics*, 1986, Vol. 14, N.4, 1453-1462.

[18] F.R. HODSON, P.M.A. SNEATH et J.E. DORAN, Some Experiments in Numerical Analysis of Archeological Data, *Biometrika*, 1966, 53, 311-324.

[19] L.J. HUBERT et J.R. BAKER, Measuring the Power of Hierarchical Cluster Analysis, *Journal of the American Statistical Association*, 1975, 70, 31-38.

[20] A.K. JAIN et J.V. MOREAU, Bootstrap Technique in Cluster Analysis, *Pattern Recognition*, 1987, Vol. 20, N.5, 547-568.

[21] M. JAMBU, Techniques de classification automatique appliquées à des données de sciences humaines, Thèse de Doctorat de 3ème Cycle, Paris, 1972.

[22] N. JARDINE et R. SIBSON, Mathematical Taxonomy, John Wiley & Sons, Ltd, New York, 1971.

[23] M.G. KENDALL, The advanced Theory of Statistics, London, Charles Griffin and Co, Ltd, 1948.

[24] G.W. MILLIGAN et M.C. COOPER, An examination of procedures for determining the number of clusters in a data set, *Psychometrika*, 1985, Vol. 50, N.2, 159-179.

[25] Ch.Z. MOONEY et R.D. DUVAL Bootstrapping: A nonparametric Approach to Statistical Inference, Series: Quantitative Applications in the Social Sciences, Sage Publications, London, 1993.

[26] W.M. RAND, Objective criteria for the Evaluation of clustering Methods, *Journal of the American Statistical Association*, 1971, Vol. 66, N.336, 846-850.

[27] Jun SHAO et Dongsheng TU, The Jackknife and Bootstrap, Springer Series in Statistics, Springer, 1995.

[28] YULE, G. UNDY et M.G. KENDALL, An introduction to the Theory of Statistics, London, Charles Griffin and Co, Ltd, 1950.

ERRATA

- ◇ page 34 : $VRC = \frac{\frac{1}{2}(\bar{d}^2 + \frac{n-k}{k-1} A_k)}{\frac{1}{2}(\bar{d}^2 - A_k)} = \frac{(\bar{d}^2 + \frac{n-k}{k-1} A_k)}{(\bar{d}^2 - A_k)}$

- ◇ page 36 : Les méthodes M_1, M_2, M_3 et M_4 sont les méthodes Gamma, de Duda et Hart, de Beale et de Calinski-Harabasz.

- ◇ page 38 : $W_K = \frac{1}{K} \sum R_i$

- ◇ page 39 : - $K = K^*$. Cela veut dire que les partitions en K classes, souvent obtenues en fusionnant certaines des classes de la partition en K^* classes, peuvent ainsi fournir des classes bien séparées et, dès lors, des classifications stables.

  - ligne 11 : Notons δM_K la décroissance moyenne sur tous les échantillons bootstrap de la valeur de M quand on passe de K à K+1 classes.

- ◇ page 47 : dernière ligne :
  A partir de là, nous pouvons facilement voir que $n_{ij} \neq \frac{n_{i.} n_{.j}}{n}$