# UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS

École Doctorale MIPTIS

Laboratoire d'Informatique, équipe de recherche FOVEA

**THÈSE** présentée par :

**Abdelheq Et-Tahir GUETTALA**

soutenue le : 5 septembre 2013

pour obtenir le grade de : Docteur de l'université François - Rabelais de Tours

Discipline/ Spécialité : INFORMATIQUE

## VizAssist : un assistant utilisateur pour le choix et le paramétrage des méthodes de fouille visuelle de données

THÈSE DIRIGÉE PAR :

| | |
|---|---|
| VENTURINI Gilles | Professeur, Université François Rabelais de Tours |
| BOUALI Fatma | Maître de conférences, Université Lille 2 |

RAPPORTEURS :

| | |
|---|---|
| LUTTON Evelyne | Directeur de recherche, UMR GMPA, INRA-AgroParisTech |
| MELANCON Guy | Professeur, Université Bordeaux 1 |

JURY :

| | |
|---|---|
| HAO Jin-Kao | Professeur, Université Angers, Président du jury |
| BOUALI Fatma | Maître de conférences, Université Lille 2 |
| LUTTON Evelyne | Directeur de recherche, UMR GMPA, INRA-AgroParisTech |
| MELANCON Guy | Professeur, Université Bordeaux 1 |
| VENTURINI Gilles | Professeur, Université François Rabelais de Tours |

# Remerciements

En tout premier lieu, je tiens à exprimer ma profonde reconnaissance à mon directeur de thèse, Pr. Gilles Venturini, et je le remercie pour son soutien scientifique et pour la confiance qu'il m'a accordée tout au long de mon parcours de doctorat ainsi que mon stage de M2R. Je remercie également tout particulièrement ma co-encadrante, Dr. Fatma Bouali, pour son implication et ses conseils inestimables.

Je tiens à remercier vivement Dr. Evelyne Lutton et Pr. Guy Melançon pour avoir accepté de rapporter ce travail, pour les commentaires critiques sur ce mémoire et leur participation à mon jury de thèse. Je remercie très chaleureusement Pr. Jin-Kao Hao pour avoir accepté de présider mon jury de thèse.

Je remercie également les membres du laboratoire d'Informatique de l'Université François Rabelais de Tours. Un grand merci à l'équipe FOVEA et tous ceux qui m'ont aidé de près ou de loin à la réalisation de ce travail. Je n'oublie pas de remercier l'équipe du CE.R.I.E.S. et plus particulièrement Dr. Christiane Guinot et Julie Latreille, pour leur support lors de l'analyse des résultats des évaluations utilisateurs de notre système.

Finalement, je remercie ma famille pour ses encouragements et son soutien incontestable. En particulier, mes parents, ma femme et mon fils Mohammed-Adam, ma sœur Lina, mes frères Abdellatif, Abdeldjalil et Abderraouf. Vous êtes ceux à qui je dédie ce travail.

# Résumé

En combinant les capacités perceptives des utilisateurs et la puissance de calcul et de stockage dont sont dotés les ordinateurs, les systèmes de fouille visuelle de données proposent plusieurs avantages. Parmi ces avantages on peut citer leur capacité à faciliter la compréhension d'un jeu de données volumineux. Cependant, l'utilisation efficace de ces systèmes par des utilisateurs novices nécessite généralement la présence d'un expert en visualisation ou une phase d'acquisition de connaissances sur les visualisations.

Nous nous intéressons dans cette thèse au problème de l'automatisation du processus de choix et de paramétrage des visualisations en fouille visuelle de données. Pour résoudre ce problème, nous avons développé un assistant utilisateur *"VizAssist"* dont l'objectif principal est de guider les utilisateurs (experts ou novices) durant le processus d'exploration et d'analyse de leur ensemble de données. Dans le but de faciliter l'intégration de nouvelles connaissances sur les visualisations et les préférences utilisateurs, ou d'adapter notre système à d'éventuelles extensions du nombre de visualisations qu'il gère, nous avons fondé son architecture sur un modèle simple et générique. Ses principaux éléments sont un modèle des données et des objectifs utilisateur, une base de connaissances sur les visualisations, un module d'appariement et de suggestion de visualisations, et un module de paramétrage interactif.

Nous illustrons, l'approche sur laquelle s'appuie VizAssit pour guider les utilisateurs dans le choix et le paramétrage des visualisations. En effet, VizAssist propose un processus en deux étapes. La première étape consiste à recueillir les objectifs annoncés par l'utilisateur ainsi que la description de son jeu de données à visualiser, pour lui proposer un sous ensemble de visualisations candidates pour le représenter. Dans cette phase, VizAssist suggère différents appariements entre la base de données à visualiser et les visualisations qu'il gère. La seconde étape permet d'affiner les différents paramétrages suggérés par le système. Dans cette phase, VizAssist utilise un algorithme génétique interactif qui a pour apport de permettre aux utilisateurs d'évaluer et d'ajuster visuellement ces paramétrages.

Finalement, nous présentons les résultats que nous avons obtenus à l'issue de l'évaluation utilisateur que nous avons réalisée pour évaluer les avantages et inconvénients de VizAssist. Nous présentons également les apports de notre outil à accomplir quelques tâches de fouille de données à travers quelques cas d'études.

**Mots clés :** Assistant utilisateur, fouille visuelle de données, interaction visuelle, interaction Homme-Machine, algorithme génétique interactif.

# Abstract

By combining the perceptual capabilities of users and the computing and storage power of computers, visual data mining systems propose several advantages. This includes for example the ability of minimizing effort provided by the user to understand a large dataset. However, an efficient use of these systems, by novice users, usually requires the presence of an expert in visualization or a practice time to acquire basic knowledge on data visualization.

In this thesis, we deal with the problem of automating the process of choosing an appropriate visualization and its parameters in the context of visual data mining. To solve this problem, we developed a user assistant *"VizAssist"* which mainly assist users (experts and novices) during the process of exploration and analysis of their dataset. In order to facilitate the process of integration of new knowledge about visualizations and user preferences, or to adjust our system to a possible extension of the number of visualizations it manages, we based its architecture on a simple and a generic model. Its main components are a data and a user's objective model, a knowledge base on the visualization, a mapping and suggestion visualization module, and the interactive parameterization module that is based on an interactive genetic algorithm.

We also illustrate the approach used by VizAssit to help users in the visualization selection and parameterization process. VizAssist proposes a process based on two steps. In the first step, VizAssist collects the user's objectives and the description of his dataset, and then proposes a subset of candidate visualizations to represent them. In this step, VizAssist suggests a different mapping between the database for representation and the set of visualizations it manages. The second step allows user to adjust the different mappings suggested by the system. In this step, VizAssist uses an interactive genetic algorithm to allow users to visually evaluate and adjust such mappings.

Finally, we present the results that we have obtained during a user evaluation we performed to evaluate the advantages and weaknesses of VizAssist. We also present the contributions of our tool to accomplish some tasks of data mining through some case studies.

**Keywords :** User Assistant, Visual Data Mining, Visual Interaction, Human-Computer Interaction, Interactive Genetic Algorithm.

# Table des matières

# Liste des tableaux

# Table des figures

# Introduction Générale

Nous assistons aujourd'hui à une véritable révolution technologique qui s'est accompagnée d'une augmentation extraordinaire du volume de données engendrant de nouveaux défis aux chercheurs dans la voie de faciliter les processus décisionnels. Étant une approche qui contribue efficacement à leur analyse, la visualisation d'informations est devenue donc incontournable dans presque tous les domaines. En effet, l'utilisation d'un outil de visualisation d'informations pour accomplir un processus de fouille visuelle de données, sur un jeu de données volumineux, a pour vocation de réduire l'effort intellectuel à fournir pour le comprendre. Cela est dû essentiellement à la capacité de communication dont sont dotées les méthodes de visualisations pour faciliter les différentes tâches d'analyse et d'exploration.

Cependant, les systèmes de fouille visuelle de données peuvent se transformer en outils complexes à l'usage, une complexité liée principalement au manque de connaissances chez ces derniers. Ces connaissances sont généralement nécessaires pour une meilleure manipulation de ces systèmes durant le processus de visualisation et dont l'intérêt est de bien comprendre les résultats proposés par les méthodes de visualisations qu'ils gèrent. Pour cette raison, [Wong, 1999] considère qu'un véritable système de fouille visuelle de données ne doit pas exiger des connaissances de la part des utilisateurs, mais plutôt les guider dans le processus d'exploration et d'analyse de leurs ensembles de données. Dans le but de réduire cette complexité, nous nous sommes donc intéressés, dans nos travaux de recherche au problème de l'automatisation du processus de choix et de paramétrage des visualisations en fouille visuelle de données. Nous présentons ci-dessous le problème que nous avons abordé durant nos recherches et pour lequel nous tentons de proposer de nouvelles solutions.

## Sujet de la thèse

Beaucoup de méthodes de fouille visuelle de données s'adressent à des spécialistes, et les utilisateurs novices ou même des experts du domaine d'application visé peuvent avoir des difficultés, et parfois échouer, lors de l'utilisation de telles méthodes [Mackinlay, 1986] [Senay et Ignatius, 1992] [Lange *et al.*, 1995] [Healey *et al.*, 1999]. Deux raisons sont responsables de ces difficultés. Tout d'abord, les utilisateurs doivent choisir la ou les visualisations qui vont représenter de manière efficace leurs données. Cela nécessite de bien connaître les visualisations, et plus précisément de savoir quelles données elles peuvent représenter et quels objectifs elles peuvent atteindre. Ensuite, les utilisateurs doivent trouver le meilleur paramétrage possible de ces visualisations, c'est à dire la meilleure correspondance possible entre les attributs des données et les signes visuels utilisés dans la visualisation. Si

une visualisation est mal choisie ou mal paramétrée, il y a de grandes chances pour que l'utilisateur ne puisse pas atteindre les objectifs qu'il s'est fixés. Enfin, il ne faut pas oublier l'aspect subjectif (esthétique, préférences visuelles, niveau de perception visuelle) propre à chaque personne et qui intervient aussi beaucoup dans le choix et le paramétrage d'une visualisation.

L'objectif de cette thèse est donc de fournir un outil interactif d'aide au choix et de paramétrage des visualisations dans un processus de fouille visuelle de données. Cet outil est un assistant utilisateur, et doit fonctionner de la manière suivante :

1. L'utilisateur fournit en entrée du système une description symbolique de ses données et de ses objectifs.
2. Le système lui propose une ou plusieurs visualisations parmi les plus adaptées à ses objectifs et aux caractéristiques de ces données, avec pour chacune un paramétrage.
3. Dans une boucle interactive, l'assistant propose à l'utilisateur d'affiner le paramétrage proposé si ce dernier ne répond pas à ses attentes. Cette partie s'appuie sur un algorithme génétique interactif.

Les objectifs que nous nous sommes efforcés de suivre sont donc de : 1) proposer un nouvel assistant utilisateur pour le choix et le paramétrage des méthodes de fouille visuelle de données en s'appuyant sur une architecture générique et simple, et résolvant certaines limitations des systèmes existants, 2) comparer notre outil à un système de visualisation utilisant un paramétrage manuel en menant une évaluation utilisateur afin d'évaluer notre approche via des résultats statistiques, 3) évaluer la capacité de notre outil à accomplir des tâches de fouille de données à travers des cas d'études pour mettre en avant ses apports vis-à-vis des systèmes existants.

## Organisation du document

– Nous nous intéressons dans le premier chapitre à la visualisation d'informations en général, et plus particulièrement aux méthodes de fouille visuelle de données. Pour cela, nous donnons quelques définitions et notions de base du domaine de la visualisation d'informations, puis nous mettons en avant les différents avantages des représentations graphiques par rapport aux méthodes classiques. Nous exposons aussi quelques limitations d'utilisation des méthodes de fouille visuelle de données liées aux systèmes de visualisation actuels. Quelques nouvelles tendances du domaine de la visualisation d'informations sont également présentées.

– Nous décrivons dans le chapitre 2 les assistants utilisateurs existants pour le choix et le paramétrage des méthodes de fouille visuelle de données. Dans le but d'étudier les avantages et inconvénients de chacun de ces systèmes, nous décrivons leur architecture générale, le mode de leur fonctionnement et leurs contributions et limitations. Une étude comparative des différents assistant utilisateurs est proposée.

– Nous présentons dans le chapitre 3 un état de l'art des algorithmes génétiques interactifs (AGIs). Dans notre illustration de ce type d'algorithmes stochastiques, nous nous appuyons sur les algorithmes génétiques classiques. Dans notre description des

AGIs, nous mettons en avant leurs principes de base et les problématiques de leur utilisation. Nous exposons aussi certaines solutions proposées dans la littérature pour les résoudre. Nous illustrons également les différents domaines d'applications des AGIs en mettant l'accent sur le domaine de la fouille visuelle de données.

– Nous proposons dans le quatrième chapitre, un aperçu général de notre assistant utilisateur **VizAssist** pour le choix et le paramétrage des méthodes de fouille visuelle de données. Un système que nous définissons sur la base d'un modèle simple et générique. En effet, pour résoudre les limitations constatées dans les systèmes existants, VizAssist propose deux étapes. La première étape consiste à suggérer à l'utilisateur différents appariements entre la base de données à visualiser et les visualisations qu'il gère en s'appuyant sur les objectifs qu'il annonce et les caractéristiques de ses données. Ces appariements sont générés par une heuristique utilisant une base de connaissances sur les visualisations et la perception visuelle. Ensuite, afin d'affiner les différents paramétrages suggérés, VizAssist utilise dans la seconde étape un algorithme génétique interactif qui permet aux utilisateurs d'évaluer et d'ajuster visuellement ces paramétrages.

– Nous détaillons dans le cinquième chapitre le modèle des données et des objectifs utilisateur ainsi que la base de connaissances sur les visualisations sur laquelle s'appuie VizAssist. En effet, dans notre assistant utilisateur, le modèle des données a pour intérêt principal de définir formellement la structure dans laquelle les données utilisateur doivent être décrites pour dérouler le processus de visualisation. Le modèle des objectifs utilisateur quant à lui, a pour but de permettre aux utilisateurs de fixer leurs objectifs d'analyse et d'exploration. La base de connaissances sur les visualisations sert en effet à représenter la description des visualisations proposées par VizAssist.

– Nous illustrons dans le chapitre 6 le module d'appariement et de suggestion des visualisations constituant l'architecture de VizAssist. Ce module s'appuie sur un processus qui se déroule en deux étapes. La première étape consiste à pré-sélectionner, à partir d'une base de connaissances sur les visualisations, celles qui sont les plus appropriées aux objectifs utilisateurs. La seconde étape permet de proposer un paramétrage à chacune des visualisations retenues. Enfin, nous présentons à travers un exemple concret le processus d'appariement proposé par VizAssist.

– Étant conscient de la difficulté du processus de paramétrage des visualisations dans les systèmes existants, liée à la subjectivité et au niveau de perception visuelle des utilisateurs, nous proposons dans le chapitre 7 notre nouvelle approche pour résoudre ce problème. En effet, dans VizAssist, nous nous appuyons sur une étape visuelle et interactive qui utilise un algorithme génétique interactif (AGI). L'objectif principal de notre méthode est donc d'affiner et d'optimiser l'appariement entre les attributs de données et les attributs visuels en permettant aux utilisateurs de participer à ce processus. Afin d'illustrer les apports de notre approche, nous décrivons à travers l'accomplissement d'une tâche de clustering, le principe général de notre AGI.

– Dans le but d'expérimenter notre système, nous avons réalisé une évaluation uti-

lisateur dans laquelle nous avons comparé notre assistant à un autre système de visualisation appelé VRMiner [Azzag *et al.*, 2005]. Nous présentons dans le huitième chapitre le protocole expérimental que nous avons défini pour mener cette évaluation et les résultats obtenus. Notons que nous nous sommes appuyés sur deux tâches pour comparer les deux systèmes et que 27 personnes ont participé aux tests. De plus, nous avons collecté durant les expérimentations différents aspects subjectifs ressentis par les participants.

– Nous proposons dans le dernier chapitre plusieurs cas d'études qui ont pour intérêt principal de mettre en avant les apports et avantages de VizAssist et de ses interfaces. Le premier cas concerne la technique adoptée pour ajouter de nouvelles visualisations dans le système. Les deuxième et troisième cas d'études permettent de montrer davantage la capacité de notre système à accomplir différentes tâches de fouille de données à travers un processus visuel et interactif. Le dernier cas d'étude porte sur l'intérêt que peut révéler le processus interactif sur lequel se base notre outil pour la réorganisation des dimensions dans des visualisations multidimensionnelles

Le travail que nous avons effectué dans cette thèse a fait l'objet de plusieurs publications scientifiques présentées dans l'annexe B.

# Première partie

# État de l'art

# Chapitre 1

# Visualisation d'informations et fouille visuelle de données

**Résumé** : Nous nous intéressons dans ce chapitre à la visualisation d'informations en général, et plus particulièrement aux méthodes de fouille visuelle de données. Pour cela, nous donnons quelques définitions et notions de base du domaine de la visualisation d'informations, puis nous mettons en avant les différents avantages des représentations graphiques par rapport aux méthodes classiques. Nous exposons aussi quelques limitations d'utilisation des techniques de fouille visuelle de données liées aux systèmes de visualisation actuels. Quelques nouvelles tendances du domaine de la visualisation d'informations sont également présentées.

## 1.1 Introduction

Nous assistons aujourd'hui à une véritable révolution technologique qui s'est accompagnée d'une nouvelle ère d'utilisation des différents outils numériques dans notre société. Cette évolution a engendré, selon de nombreuses études [Lesk, 1997] [Dienes, 2012] [Hilbert et López, 2011], une augmentation extraordinaire du volume de données. La visualisation d'informations est une approche qui contribue efficacement à leur analyse. La visualisation, en tant que support de communication, permet d'extraire plus aisément un maximum de connaissances [Chen, 2005] [Shneiderman, 2008] afin de faciliter le processus décisionnel sur de grands ensembles de données. D'ailleurs, l'importance de la visualisation est traduite par plusieurs expressions littéraires très connues comme *"A picture is worth a thousand words"* ou *"Seeing is believing"*.

Dans ce chapitre, nous nous intéressons à la visualisation d'informations en général, et plus particulièrement aux méthodes de fouille visuelle de données. Ces dernières représentent l'axe principal de notre recherche dans cette thèse. Nous commençons par présenter la visualisation d'informations et ses apports par rapport aux approches classiques. Nous décrivons ensuite le principe de la construction graphique ainsi que le processus de visualisation communément utilisé dans la littérature et adopté par la majorité des systèmes de visualisation. Nous définissons ensuite la fouille visuelle de données et donnons quelques exemples de ses méthodes. Puis, nous exposons les enjeux actuels de la visualisation et les nouvelles approches qui commencent à émerger dans ce domaine. Nous terminons par une conclusion.

## 1.2 La visualisation d'informations

Selon [Card *et al.*, 1999], la visualisation d'informations peut être définie par l'utilisation des représentations graphiques interactives de données abstraites dans le but d'amplifier la cognition humaine. Plusieurs autres définitions existent en littérature, mais convergent toutes vers son apport fondamental résidant dans sa capacité à réduire l'effort cognitif. C'est donc cet avantage de la visualisation d'informations qui explique son émergence actuelle dans toutes les disciplines. En effet, la cognition étant la façon dont nous organisons nos pensées afin de donner un sens à notre propre représentation de la réalité qui nous entoure, la visualisation d'informations, à travers l'utilisation des représentations graphiques, permet l'amélioration des processus cognitifs en simplifiant l'acquisition et la compréhension de grands volumes de données (voir figure 1.1). Cela se justifie surtout par le fait que le système visuel humain est directement sollicité dans le processus d'interprétation des données. De plus, différentes études du système visuel humain ont prouvé qu'il permet d'absorber plus rapidement une grande quantité d'informations à partir des représentations graphiques [Fekete et Plaisant, 2002], contrairement aux approches classiques (textes et tableaux) dans lesquelles des mots et des chiffres doivent être décomposés, interprétés et analysés pour être compris.

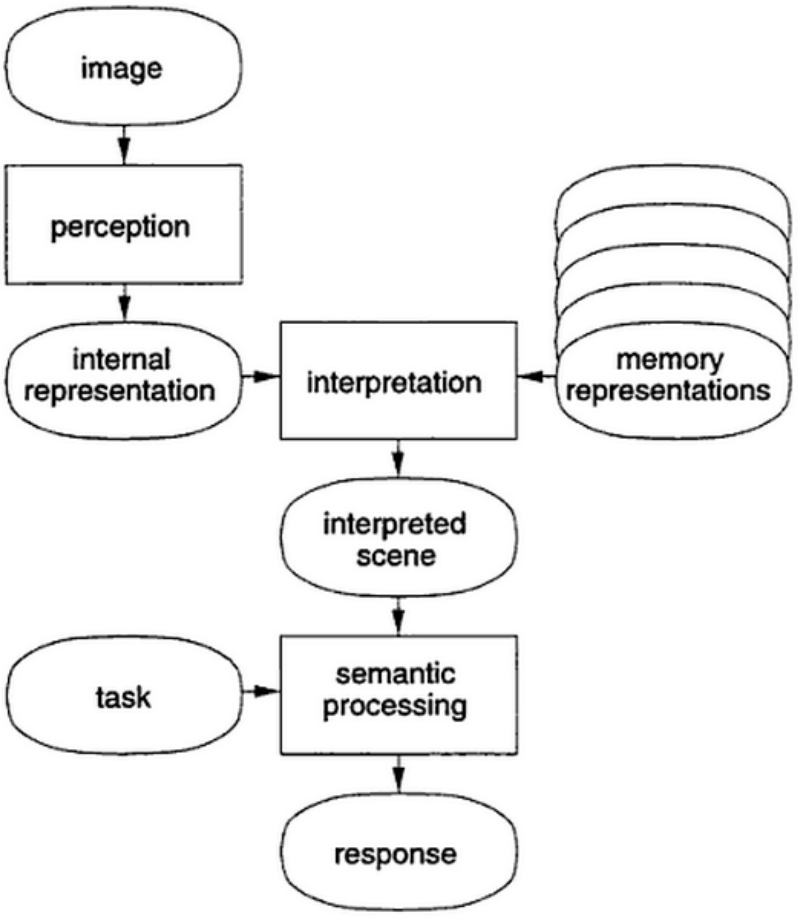


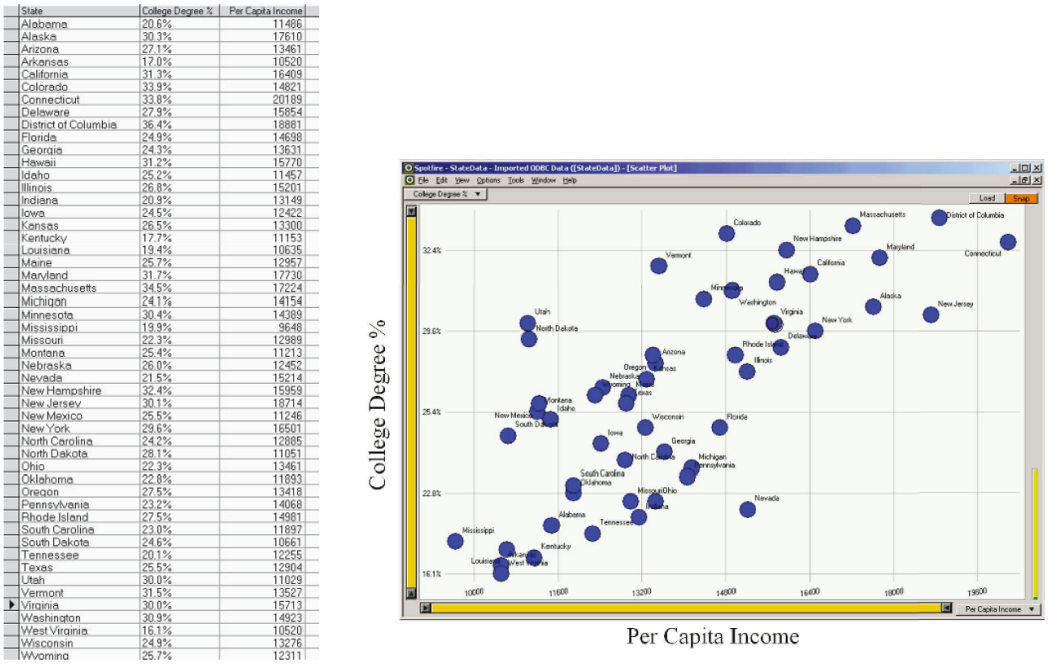

FIGURE 1.1 – Représentation schématique du processus visuo-cognitif des images selon [Janssen, 2001].

### 1.2.1 Entre approches classiques et approches visuelles

De nos jours, analyser des bases de données pour extraire des informations, en s'appuyant sur les approches classiques (textuelles ou tabulaires), est devenue une tâche difficile, voir même impossible [Eick, 2000]. Ce constat se justifie principalement par la grande masse de données caractérisant les jeux de données actuels. Pour [Derthick *et al.*, 1997], la meilleure approche pour résoudre les contraintes liées aux approches classiques est d'utiliser des techniques de visualisation interactives. En effet, ils considèrent que ces dernières sont très avantageuses pour accomplir un processus d'exploration car les capacités perceptives humaines sont directement sollicitées. Ceci permet d'assurer une continuité dans l'analyse d'une grande quantité de données sur une seule image.

Fekete et al. dans [Fekete *et al.*, 2008] ont présenté à travers une étude comparative entre une représentation graphique et une représentation tabulaire (voir figure 1.2), d'un même jeu de données, les avantages et les atouts de la visualisation d'informations. En effet, dans leur exemple, ils ont essayé d'accomplir trois tâches d'analyse dont le degré

de difficulté est graduellement différent. Les résultats de leur expérimentation prouvent que l'utilisation d'une représentation tabulaire pour satisfaire des tâches simples sur une colonne ou une ligne d'un tableau est possible et parfois même plus facile à accomplir que sur des visualisations. Par contre, dès que la tâche demandée nécessite une exploration plus détaillée de l'ensemble de données (ex. extraire des informations séparées, puis les combiner pour accomplir une tâche donnée), la difficulté d'utilisation des approches classiques se pose. De plus, la durée nécessaire pour achever l'objectif demandé devient très longue et la tâche à accomplir fastidieuse.



| State | College Degree % | Per Capita Income |
|---|---|---|
| Alabama | 20.6% | 11486 |
| Alaska | 30.3% | 17610 |
| Arizona | 27.1% | 13461 |
| Arkansas | 17.0% | 10520 |
| California | 31.3% | 16409 |
| Colorado | 33.9% | 14821 |
| Connecticut | 33.8% | 20189 |
| Delaware | 27.9% | 15854 |
| District of Columbia | 36.4% | 18881 |
| Florida | 24.9% | 14698 |
| Georgia | 24.3% | 13631 |
| Hawaii | 31.2% | 15770 |
| Idaho | 25.2% | 11457 |
| Illinois | 26.8% | 15201 |
| Indiana | 20.9% | 13149 |
| Iowa | 24.5% | 12422 |
| Kansas | 26.5% | 13300 |
| Kentucky | 17.7% | 11153 |
| Louisiana | 19.4% | 10635 |
| Maine | 25.7% | 12957 |
| Maryland | 31.7% | 17730 |
| Massachusetts | 34.5% | 17224 |
| Michigan | 24.1% | 14154 |
| Minnesota | 30.4% | 14389 |
| Mississippi | 19.9% | 9648 |
| Missouri | 22.3% | 12989 |
| Montana | 25.4% | 11213 |
| Nebraska | 26.0% | 12452 |
| Nevada | 21.5% | 15214 |
| New Hampshire | 32.4% | 15959 |
| New Jersey | 30.1% | 18714 |
| New Mexico | 25.5% | 11246 |
| New York | 29.6% | 16501 |
| North Carolina | 24.2% | 12685 |
| North Dakota | 28.1% | 11051 |
| Ohio | 22.3% | 13461 |
| Oklahoma | 22.8% | 11893 |
| Oregon | 27.5% | 13418 |
| Pennsylvania | 23.2% | 14068 |
| Rhode Island | 27.5% | 14981 |
| South Carolina | 23.0% | 11897 |
| South Dakota | 24.6% | 10661 |
| Tennessee | 20.1% | 12255 |
| Texas | 25.5% | 12904 |
| Utah | 30.0% | 11029 |
| Vermont | 31.5% | 13527 |
| Virginia | 30.0% | 15713 |
| Washington | 30.9% | 14923 |
| West Virginia | 16.1% | 10520 |
| Wisconsin | 24.9% | 13276 |
| Wyoming | 25.7% | 12311 |



**(a)** A thousand words **(b)** A picture

FIGURE 1.2 – Comparaison entre une représentation graphique (nuage de points) et une représentation tabulaire d'un même jeu de données [Fekete *et al.*, 2008].

La figure 1.3 illustre une comparaison entre la description d'un itinéraire de voyage sous deux formes différentes : une représentation graphique (carte géographique) et une représentation textuelle [Mazza, 2009]. En analysant les deux représentations, [Mazza, 2009] a mis en avant l'apport d'utilisation des représentations graphiques qui permettent parfois d'extraire des informations très utiles et même inattendues (trouver une station de services, trouver des lieux de loisirs, trouver un hôtel, prendre un raccourci, etc.). En effet, même si l'intérêt principal de la carte géographique (la visualisation) illustrée dans la figure 1.3 est de guider une personne pour se déplacer entre deux points, il lui est très facile de se situer par rapport à des lieux qui l'intéressent si nécessaire. On peut conclure donc que la visualisation en tant que moyen de communication représente un avantage incontournable pour simplifier le processus d'extraction de connaissances.

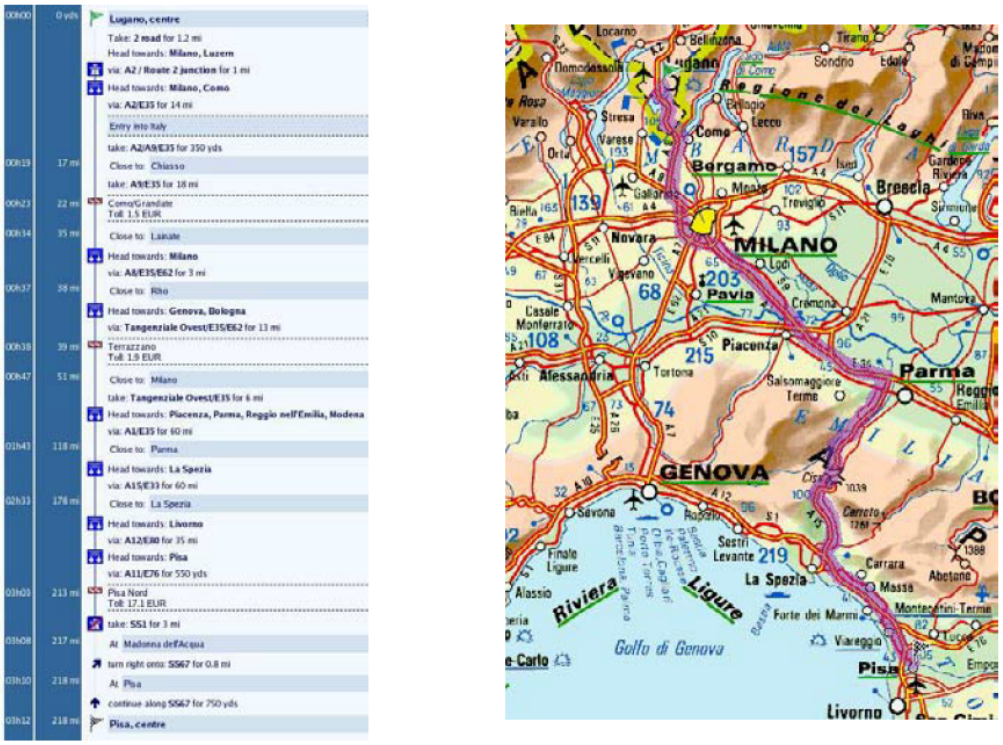



FIGURE 1.3 – Comparaison entre une représentation visuelle et textuelle pour la description d'un itinéraire de voyage [Mazza, 2009].

### 1.2.2 Apports de la visualisation d'informations

Devenue de très grande utilité et même parfois incontournable, la visualisation d'information est utilisée aujourd'hui pratiquement dans tous les domaines. Cette nouvelle pratique a incité beaucoup de chercheurs de la communauté de la visualisation d'informations à s'intéresser à l'étude des différents critères pouvant influencer l'efficacité des représentations graphiques. Shneiderman dans [Shneiderman, 1996] a lié l'efficacité d'une visualisation à son aptitude à présenter les données dans différentes formes, selon l'intérêt qu'elles peuvent susciter. Dans [Bertin, 1998], Bertin s'est intéressé plutôt à la notion de la qualité des représentations graphiques, en mettant en avant le rapport entre sa capacité à répondre à un objectif d'analyse donné et la durée d'observation nécessaire pour le satisfaire. Knight [Knight, 2001] a abordé l'efficacité des visualisations en se focalisant sur leurs utilisations et les interactions qu'elles offrent. Pour cela, elle a fixé deux critères importants. Le premier critère concerne la capacité d'une visualisation à accomplir des tâches qu'elle est censée supporter. Le deuxième critère concerne plutôt l'aspect conceptuel de la visualisation et la façon qu'elle représente les données. Plus récemment, Fekete et al. dans [Fekete *et al.*, 2008] ont listé quelques apports de la visualisation d'informations à travers les différentes réponses qu'ils ont proposées à la question *"how and why is InfoVis useful ?"*. Parmi les atouts qu'ils ont recensés, on peut citer la capacité d'une représentation graphique à assurer le même raisonnement que la mémoire humaine lors de

l'accomplissement d'un processus cognitif en s'appuyant sur la perception visuelle. De plus, ils considèrent qu'une visualisation permet d'expliquer plus facilement des phénomènes très complexes à découvrir comme dans [Gilbert, 1958].

Les représentations graphiques, fondées dans leur conception sur les travaux de Jacques Bertin [Bertin, 1983], ont beaucoup d'avantages. Dans [González et Kobsa, 2003], González et Kobsa mettent en avant la capacité dont sont dotées les visualisations pour faciliter plusieurs processus d'analyses de données. Ils considèrent que l'importance de ces visualisations réside dans le fait qu'elles permettent de donner un aperçu global de l'ensemble de données, de les manipuler de manière flexible et surtout de simplifier la transformation des résultats en représentations graphiques facile à comprendre. Pour [Tufte, 2001], elles peuvent aussi illustrer, avec clarté et précision, des concepts complexes, parfois difficiles à expliquer verbalement. La représentation graphique qu'a utilisé Charles Joseph Minard (voir figure 1.4) en est l'exemple de référence. D'ailleurs, selon [Tufte, 1983], cette figure est le meilleur graphique statistique qui a jamais été produit. Bien qu'il s'est appuyé sur un espace très restreint pour illustrer les pertes successives en hommes de l'armée Française dans la compagne de Russie 1812-1813, Charles Joseph Minard a réussi à raconter avec succès plusieurs étapes de cette dernière à travers son graphique. En effet, la lecture de sa visualisation permet de percevoir très facilement que les troupes françaises ont pris différentes voies pour se rendre à Moscou et que leur nombre change à chaque étape du trajet. Aussi, on peut aisément observer que les affrontements entre les deux armées (russe et française) ne sont pas la seule cause des pertes constatées dans cette campagne. La baisse importante de la température en hiver en Russie en est également une autre cause. En résumé, les représentations graphiques peuvent donc être considérées comme un support visuel de communication très efficace dont la capacité d'illustration d'une grande masse d'informations permet de réduire au minimum l'effort intellectuel à fournir pour les comprendre.

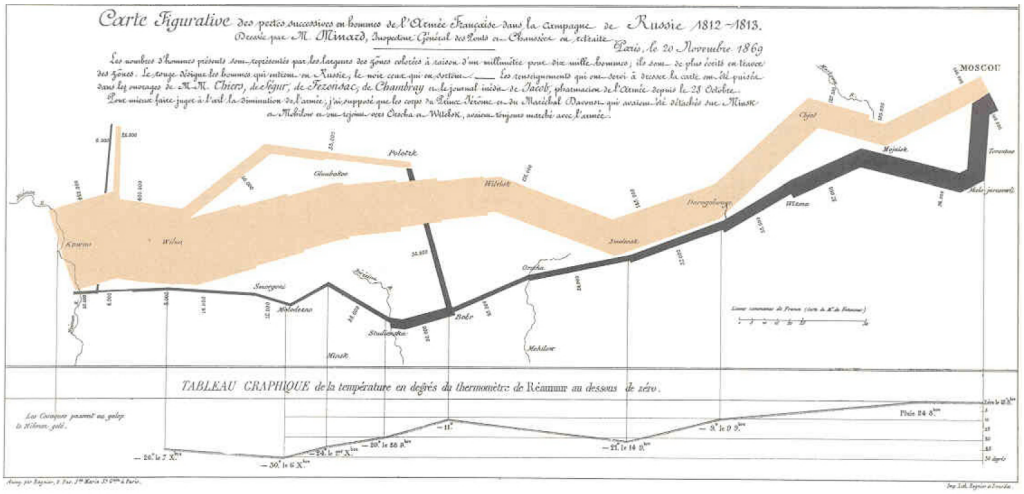



FIGURE 1.4 – Une représentation graphique des pertes successives en hommes de l'armée Française dans la compagne de Russie 1812-1813. [Tufte, 1983].

### 1.2.3 Principe de la construction graphique

Pour [Bertin, 1998], les trois fonctions de base d'une représentation graphique sont : l'enregistrement, la communication et le traitement des informations. Dans le but de donner un sens à ces fonctionnalités, Bertin s'est intéressé au principe de base de la construction graphique. En effet, conscient des capacités du système visuel humain à percevoir des signes graphiques sur un plan, il identifie trois symboles graphiques de base pouvant caractériser toute représentation graphique : un point, une ligne et une surface. L'utilisation de ces derniers pour véhiculer des informations nécessite d'après Bertin de s'appuyer sur un *"système d'expression"* regroupant huit composantes (voir figure 1.6). Ces composantes sont en effet les huit variations perceptives (variables visuelles) qui peuvent influencer l'interprétation des trois symboles graphiques cités ci-dessus, dans un plan. Dans la suite de ce manuscrit, nous utilisons l'expression *"attribut visuels"* pour désigner les variables visuelles d'une représentation graphique. A noter, que le symbole graphique *"volume"* présenté dans la figure 1.5 représente une extension du symbole *"surface"* dans un repère 3D.

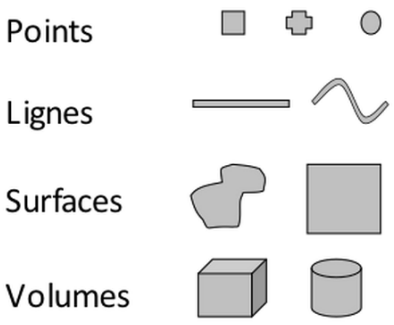



FIGURE 1.5 – Les symboles graphiques de base caractérisant toute représentation graphique.

### 1.2.4 Processus de visualisation

Représenter des données sous forme d'un graphique en se fondant sur un système de visualisation nécessite de s'appuyer sur un processus de visualisation [Dos Santos et K., 2004] [Haber et Mcnabb, 1990] [Card *et al.*, 1999]. La description de ce processus telle qu'elle a été proposée dans [Card *et al.*, 1999] peut être définie par un modèle scindé en quatre états allant de la forme initiale des données *"données brutes"*, représentée habituellement sous forme de textes ou de tableaux, à leur forme finale *"Image"* (voir figure 1.7). Généralement, dans ce processus, les données brutes subissent trois étapes de transformation.

1. **Pré-traitement des données** : elle consiste à faire un pré-traitement sur les données brutes à visualiser afin d'extraire les caractéristiques nécessaires (nombre de variables, nom des variables, type de données des variables, etc.) pour les mettre sous un format prédéfini. Les caractéristiques récupérées durant cette phase sont indispensables car elles forment les informations de base du reste du processus de visualisation. En effet, la nouvelle structuration des données rend plus facile l'exploitation et la manipulation des données par les systèmes de visualisations.

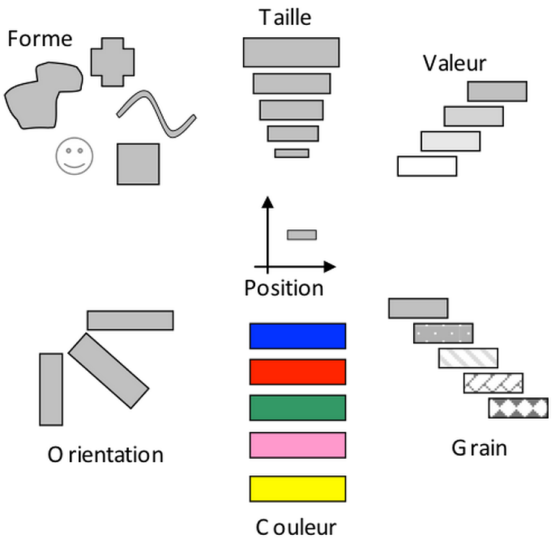



FIGURE 1.6 – Les 8 variables visuelles caractérisant les variations perceptives de symboles graphiques selon [Bertin, 1983].

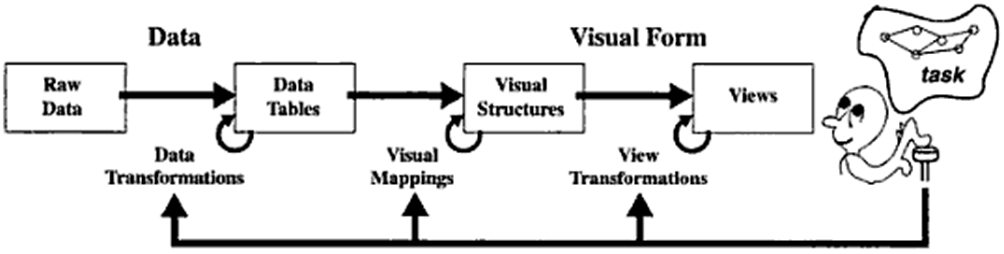



FIGURE 1.7 – Description du processus de visualisations proposé par [Card *et al.*, 1999].

2. **Appariement entre attributs de données et variables visuelles** : cette étape consiste à trouver l'appariement le plus approprié entre les variables décrivant les données à visualiser (appelées aussi attributs ou dimensions de données) et les variables visuelles (couleur, taille, position, etc.) caractérisant les symboles graphiques (ligne, point, surface, etc.) de la visualisation à générer par le système. Dans le cas des visualisations multidimensionnelles, ce processus dépend généralement de la nature des ensembles de données à représenter (nombre d'attributs de données, type de données, etc.) et est lié à la perception visuelle humaine. En effet, l'association des attributs de données avec les attributs visuels des symboles graphiques d'une visualisation exige de s'appuyer sur des règles de perception visuelle assurant une meilleure interprétation des représentations graphiques. Dans la littérature, on peut distinguer deux types de ces règles : 1) l'efficacité des attributs visuels à représenter de manière significative tel ou tel attribut de données, 2) la capacité des attributs visuels à représenter un nombre donné de valeurs discrètes ou continues d'un attribut de données.

   Parmi les travaux tenant compte du premier type de règles de perception visuelle,

on trouve [Cleveland et McGill, 1984] qui a proposé une classification des différentes tâches perceptives (voir figure 1.8). Dans cette dernière, les attributs visuels décrits dans la section précédente sont ordonnés selon leur efficacité à représenter seulement des données qualitatives (ordinale et nominale). Recensant plus d'attributs visuels, Mackinlay dans [Mackinlay, 1986] propose une extension de la classification de [Cleveland et McGill, 1984] prenant en considération les données quantitatives (voir figure 1.9). A noter, qu'il existe d'autres classifications dans la littérature comme [Salisbury, 2001]. La figure 1.10 illustre une des solutions proposées pour tenir compte du second type de règles de perception visuelle. [Wilkins, 2003] considéra que seules 12 couleurs peuvent être perçues simultanément dans une représentation graphique. Dans [Healey *et al.*, 2008] le nombre de couleur est réduit à 7.

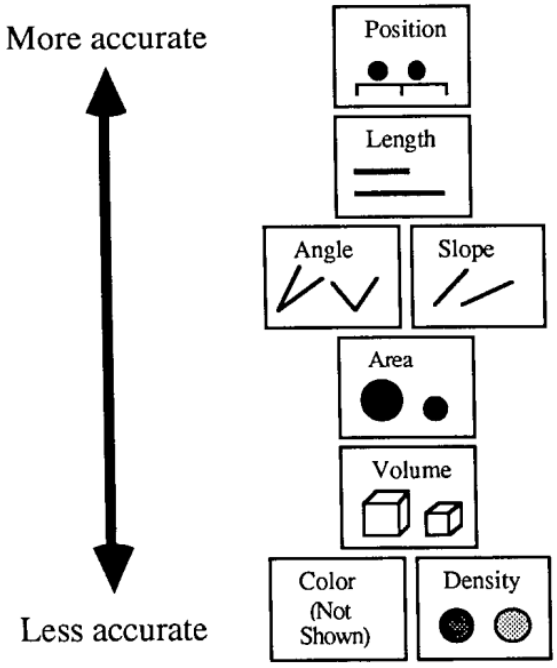



FIGURE 1.8 – Proposition de [Cleveland et McGill, 1984] pour la classification des différentes tâches perceptives [Mackinlay, 1986].

Le résultat obtenu à l'issue de cette deuxième étape du processus de visualisation est donc basé sur : 1) les caractéristiques des attributs de données de l'ensemble de données à visualiser, 2) les caractéristiques des attributs visuels de la visualisation choisie pour représenter ces données. Le modèle décrivant l'appariement entre les attributs de données et les attributs visuels permet donc de définir la description de la structure visuelle de base (symbole graphique) de la visualisation à générer par le système de visualisation.

3. **Générations des représentations visuelles** : la description structurelle créée lors de la deuxième étape du processus de visualisation sert de paramétrage pour générer le rendu visuel de l'ensemble de données à représenter. Pour générer ce dernier, un

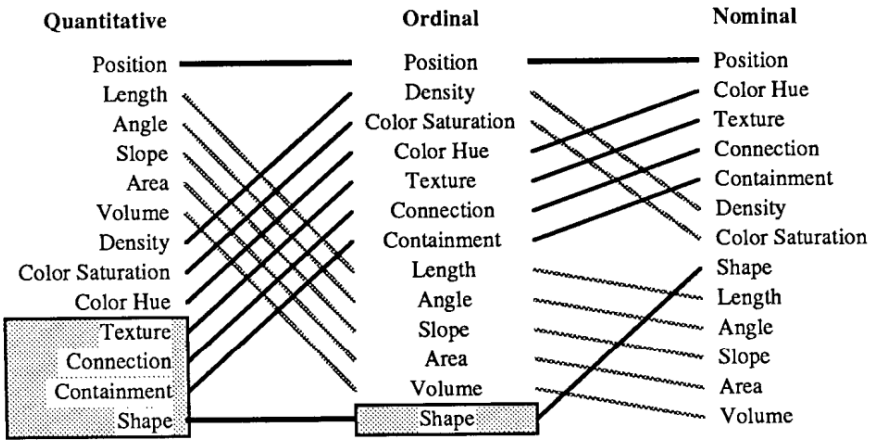



FIGURE 1.9 – Classification des attributs visuels selon leur capacité à renseigner les attributs de données proposée par [Mackinlay, 1986]. La première ligne du tableau désigne les attributs visuels les plus efficace à représenter chaque type de données.

| Visual Feature | Limit | Visual Feature | Limit | Visual Feature | Limit |
|---|---|---|---|---|---|
| Horizontal Position | 100 | Area | 10 | Line Thickness | 3 |
| Vertical Position | 100 | Shading | 4 | Line Dashing | 2 |
| Height | 50 | Connectivity | 8 | Shape | 5 |
| Width | 50 | Colour | 12 | Visibility | 2 |
| Line Length | 50 | Labels | ∞ | Tabular | ∞ |

FIGURE 1.10 – La limite maximale des valeurs que peuvent représenter les différents attributs visuels selon [Wilkins, 2003].

système de visualisation s'appuie généralement sur une librairie de visualisations. En effet, cette dernière utilise l'appariement entre les attributs de données et les variables visuelles défini dans la description structurelle des symboles graphiques pour créer les représentations graphiques. A noter que dans le cas des visualisations multidimensionnelles, chaque symbole graphique représente une donnée du jeu de données fourni en entrée du système.

### 1.2.5 Limites de la visualisation d'informations

Utiliser la visualisation d'informations comme seule technique pour accomplir quelques tâches avancées d'analyse sur des jeux de données volumineux (classification, clustering, détection des éléments aberrants, etc.) est loin d'être suffisante. En effet, exécuter ce type de processus d'analyse de données (connu aussi sous le nom de fouille de données), sur des visualisations nécessite de doter ces dernières d'approches automatiques dont le résultat de traitement est directement visualisé pour être interprété. Pour pallier les difficultés liées à la fouille de données sur des représentations graphiques statiques, plusieurs chercheurs ont focalisé leurs efforts sur le développement des méthodes de ***fouille visuelle de données***. Ces méthodes ont pour objectif de passer de l'utilisation traditionnelle des visualisations (processus d'exploration visuelle simple), à un nouveau type d'utilisation plus efficace permettant de faciliter le processus d'extraction des connaissances à partir des jeux

de données volumineux [Ferreira de Oliveira et Levkowitz, 2003]. Nous décrivons dans la section suivante la fouille visuelle de données en exposant les différentes définitions qui lui sont attribuées dans la littérature, ses avantages ainsi que quelques exemples de techniques.

## 1.3 Fouille visuelle de données (FVD)

L'application des différentes approches et algorithmes utilisés dans un processus de fouille de données permettent principalement d'extraire des connaissances de manière automatique à partir d'un jeu de données utilisateur. Cependant, la présentation des résultats de ce processus sous une seule forme appropriée, pouvant satisfaire tous les utilisateurs, est une activité parfois difficile. Cette difficulté est due essentiellement à la différence d'intérêt qui peut exister dans l'analyse du même résultat par différents types d'utilisateurs. Selon [Ankerst, 2001], la fouille visuelle de données est l'approche la plus efficace qui permet d'aborder cette problématique. Pour Ankerst cette efficacité se justifie par la capacité de la fouille visuelle de données à cerner cette nuance d'intérêt en intégrant l'utilisateur dans le processus d'interprétation des résultats fournis par le traitement opéré avec les algorithmes de fouille de données.

### 1.3.1 Définitions

Dans la littérature, différentes définitions sont attribuées à la fouille visuelle de données. Pour [Wong, 1999], la FVD se caractérise par une intégration d'un processus analytique et différentes techniques de visualisation d'informations dans un même outil de fouille de données. Selon Niggemann dans [Niggemann, 2001], l'idée principale de la FVD est de combiner 1) les capacités des ordinateurs à traiter une grande masse de données dans un délai très court, et 2) l'aptitude du système visuel humain à identifier très aisément des formes ou des modèles sur des représentations graphiques. Keim et al. dans [Keim *et al.*, 2002] décrivent la FVD comme une nouvelle approche de la fouille de données combinant des techniques de fouille de données avec des méthodes de visualisation d'informations. Pour [Ferreira de Oliveira et Levkowitz, 2003], une technique de fouille visuelle de données est un algorithme de fouille de données dans lequel la visualisation joue un rôle important. Dans [R. Albertoni et Hauska, 2003], la FVD est définie comme étant une nouvelle approche du processus d'extraction de connaissances dans lequel la visualisation est utilisée comme un moyen de communication entre l'utilisateur et l'ordinateur. [Chen *et al.*, 2007] décrit la FVD comme une nouvelle étape qui s'ajoute au cycle traditionnel du processus d'extraction de connaissances. Dans leurs travaux, Simoff et al. [Simoff *et al.*, 2008] définissent la fouille visuelle de données par un processus d'interaction et de raisonnement analytique opéré sur une ou plusieurs visualisations dont le principal intérêt est de découvrir visuellement des modèles robustes dans les données. Ces derniers permettent en effet d'extraire des informations et des connaissances pouvant être utilisées dans un processus décisionnel. En résumé, la fouille visuelle de données en tant que discipline se situe dans l'intersection entre le domaine de la visualisation d'informations et le domaine de la fouille de données.

### 1.3.2 Avantages de la fouille visuelle de données

En combinant les capacités perceptives des utilisateurs et la puissance de calcul et de stockage dont sont dotés les ordinateurs, les systèmes de fouille visuelle de données proposent plusieurs avantages aux utilisateurs. Pour [Keim *et al.*, 2002], l'un des domaines où l'utilisation des techniques de FVD a fait ses preuves est le domaine d'analyse exploratoire des données. D'ailleurs [Keim et Kriegel, 1996] [Keim *et al.*, 2002] [Rossi, 2006] [Kimani *et al.*, 2008] considèrent que le principal apport de ces techniques réside dans leur potentiel à faciliter les différentes tâches d'exploration sur de grandes bases de données multidimensionnelles. L'autre apport de ces méthodes est leur capacité à fournir un degré très élevé de confiance dans les résultats qu'elles fournissent, à l'issue du processus d'exploration [R. Albertoni et Hauska, 2003]. Ces deux apports sont justifiés surtout par le fait que même si les objectifs des utilisateurs, durant le processus d'exploration des données, sont très vagues ou que les bases de données à analyser sont hétérogènes et très bruitées, le processus d'exploration n'est en aucun cas compromis. Les techniques de FVD présentent aussi d'autres avantages. On peut citer à titre d'exemple leur intuitivité d'utilisation pour effectuer différentes tâches d'analyses statistiques d'une part, et le temps nécessaire pour les accomplir qui est nettement inférieur à celui nécessité par les systèmes traditionnels utilisant des visualisations statiques [Bogacz et Trafton, 2005] [R. Albertoni et Hauska, 2003]. Pour un expert en fouille de données, l'utilisation des techniques de FVD est aussi très utile car elles offrent des mécanismes d'interactions pour contrôler et paramétrer des algorithmes de fouille de données et les résultats qu'ils fournissent à chaque étape du processus d'extraction de connaissances [Wegman, 2003] [Schulz *et al.*, 2006]. Elles peuvent aussi permettre à un utilisateur de sélectionner plus facilement un sous ensemble de données pour accomplir une tâche d'analyse donnée. Ces utilisateurs peuvent ainsi, tester plusieurs scénarios et hypothèses d'exécution d'une même tâche sur différentes répartitions des données, pouvant ainsi conduire à des conclusions plus pertinentes pour un décideur.

### 1.3.3 Techniques de fouille visuelle de données

Beaucoup de méthodes de fouille visuelle de données existent dans la littérature. Cependant, le choix de la meilleure technique pour accomplir une tâche donnée est une opération difficile. En effet, cela nécessite certaines connaissances a priori sur les avantages proposées par chacune de ces méthodes. Les principales différences dans les techniques de FVD concernent leurs descriptions conceptuelles, les tâches d'analyse qu'elles permettent d'effectuer sur un jeu de données et les différentes possibilités d'interactions qu'elles proposent. Afin de rationaliser leurs utilisations et profiter des divers avantages qu'elles offrent, quelques chercheurs ont tenté de classifier ces méthodes en se basant sur divers critères [Hoffman, 1977] [Grinstein *et al.*, 2001] [Hoffman et Grinstein, 2002] [Chan, 2006] [Mazza, 2009]. On peut distinguer deux principales classifications de ces méthodes dans la littérature. Une classification par méthodes de visualisation et une classification par type de données représentées. Afin d'illustrer chacune de ces deux classifications, nous donnons dans ce qui suit un exemple de travaux mettant en avant cette répartition des techniques de visualisation.

En s'appuyant sur l'intérêt d'utilisation des techniques de fouille visuelle de données,

[Ferreira de Oliveira et Levkowitz, 2003] regroupent ces dernières en deux catégories. La première catégorie concerne les méthodes servant surtout à extraire de la connaissance ou à exécuter une tâche spécifique de fouille de données. La deuxième catégorie comporte les techniques destinées plutôt à afficher les résultats fournis à l'issue de l'exécution des algorithmes de fouille de données. Le principal intérêt de cette deuxième catégorie est l'amélioration du processus d'interprétation des résultats fournis aux utilisateurs.

Dans leur classification des techniques de FVD, Keim et Kriegel [Keim et Kriegel, 1996] se sont fondés sur la nature des données qu'elles peuvent représenter et ont proposé ainsi une répartition en 5 groupes. Les méthodes de la première catégorie s'organisent dans leur conception sur une projection géométrique des données dans un espace 2D ou 3D [Andrews, 1972] [Carr *et al.*, 1987] [Inselberg et Dimsdale, 1990] [Pirolli et Rao, 1996] [Bendix *et al.*, 2005]. La deuxième catégorie regroupe les méthodes fondées sur une représentation iconique comme [Chernoff, 1973] [Pickett et Grinstein, 1988]. Les visualisations appartenant à la troisième catégorie sont définies par une conception graphique à base de pixels [Ankerst, 2001]. Les représentations graphiques du $4^{ième}$ groupe sont définies sous forme arborescente ou hiérarchique [Shneiderman, 1992]. La dernière classe des visualisations comporte toutes les visualisations permettant de visualiser des données relationnelles [Collberg *et al.*, 2003].

Comme nous nous sommes intéressés dans nos recherches aux visualisations multidimensionnelles, nous détaillons dans ce qui suit les principales représentations graphiques de cette catégorie. Un survol plus exhaustif de ces techniques est proposé dans [Chan, 2006]. Nous illustrons pour chacune des visualisations présentées son principe de base et un exemple d'utilisation.

#### 1.3.3.1 Nuages de points *"scatterplot"*

La visualisation *"nuage de points"* est considérée comme l'une des représentations graphiques les plus connues et les plus utilisées dans les outils de visualisation actuels. Cela est dû surtout à sa capacité à fournir un aperçu général permettant de découvrir des tendances, des données atypiques et surtout des concentrations de données (ex. classes de données). La description conceptuelle de cette visualisation dépend étroitement de sa dimension visuelle (2D ou 3D). En effet, si les données à visualiser sont à représenter dans un espace 2D, seul deux attributs de données quantitatives peuvent être visualisés en s'appuyant sur les deux axes X et Y d'un plan (voir figure 1.11). Cette visualisation permet aussi de représenter des attributs de données de type qualitatif en utilisant d'autres signes visuels pouvant caractériser un point comme sa couleur et sa forme (voir figure 1.12). La figure 1.13 illustre un exemple d'utilisation de cette technique de visualisation dans un espace 3D. On peut remarquer que cette dernière permet de représenter plusieurs attributs de données simultanément. Plusieurs travaux dans le domaine de la fouille de données s'appuient sur cette technique de visualisation pour résoudre des problèmes liés aux différentes tâches d'exploration et d'analyse de données [Boudjeloud-Assala et Poulet, 2008].

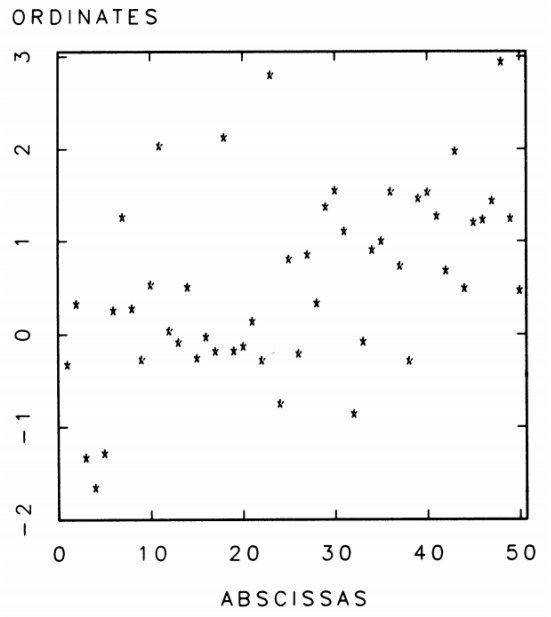



FIGURE 1.11 – Aperçu de la visualisation nuage de points décrite par 2 attributs visuels [Cleveland, 1979].

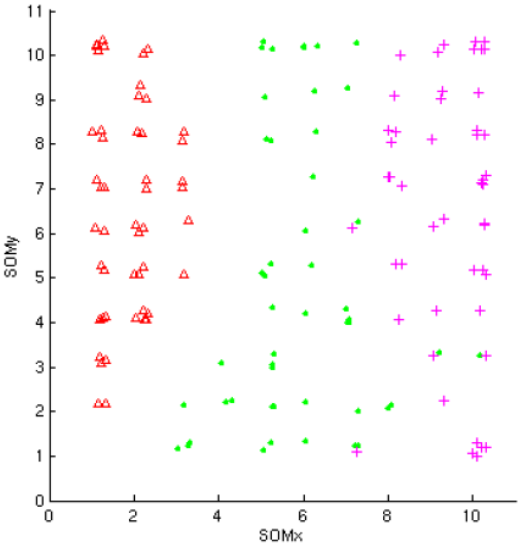



FIGURE 1.12 – Aperçu de la visualisation nuage de points décrite par 4 attributs visuels [Grinstein *et al.*, 2001].

#### 1.3.3.2 Matrice de nuages de points *"Scatterplot Matrix"*

La figure 1.14 illustre un exemple d'utilisation de la visualisation *"Scatterplot Matrix"* dans le domaine de la fouille de données [Becker et Cleveland, 1987] En effet, cette dernière est une extension de la représentation graphique *"nuage de points"* sous forme de matrice

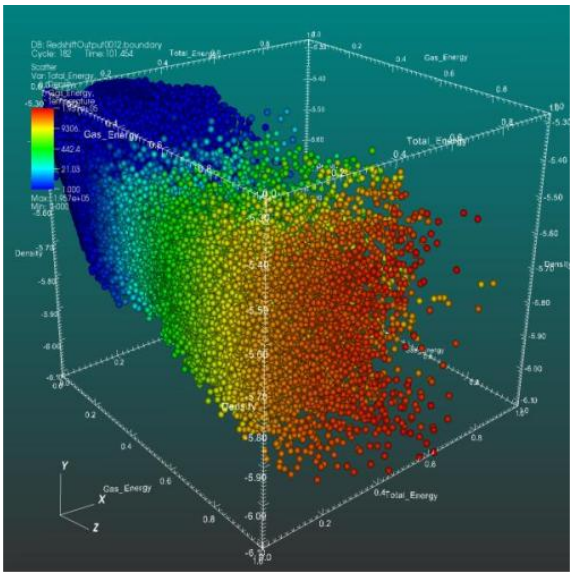


FIGURE 1.13 – Aperçu de la visualisation nuage de points représentée dans un espace 3D [2].

regroupant plusieurs instances de cette dernière. Cette visualisation permet de résoudre les contraintes d'utilisation de la représentation graphique *"nuage de points"* quand le nombre de points affichées sur un même plan est très important [Carr *et al.*, 1987]. Son principe de base est de s'appuyer sur une matrice carrée pour visualiser un ensemble d'attributs de données sur les deux axes : vertical et horizontal. Ces derniers sont alignés dans le même ordre à la fois sur les lignes et les colonnes de la matrice. Ceci permet d'étudier toute relation qui peut exister entre les attributs de données en comparant leurs valeurs deux à deux. La figure 1.15 illustre une autre forme de la visualisation *"Scatterplot Matrix"* proposée dans [Grinstein *et al.*, 2001], où deux visualisations sont combinées (des nuages de points et des histogrammes) pour représenter les attributs de données dans un même plan (2D).

#### 1.3.3.3 Coordonnées parallèles

Les *"coordonnées parallèles"* [Ocagne, 1885] [Inselberg et Dimsdale, 1990] est l'une des représentations graphiques les plus utilisées pour visualiser les jeux de données multidimensionnels. En effet, dans cette visualisation, chaque attribut de donnée de l'ensemble de données à visualiser est représenté par un axe vertical. Une fois créés, les axes sont disposés de manière parallèle dans un plan $L$ dans lequel la distance entre deux axes adjacents est identique. Dans cette visualisation, représenter une donnée consiste à la transformer graphiquement en une ligne brisée *"polyligne"* dans un plan. Cette dernière est formée par une combinaison de plusieurs lignes connectant deux axes qui se suivent. Le point de connexion de la ligne brisée avec chaque axe représente donc la valeur qui caractérise une donnée pour un attribut de données. Afin de profiter d'une meilleure utilisation de la visualisation *"coordonnées parallèles"* dans un processus d'analyse et d'exploration de

2. https ://wci.llnl.gov/codes/visit/gallery.html

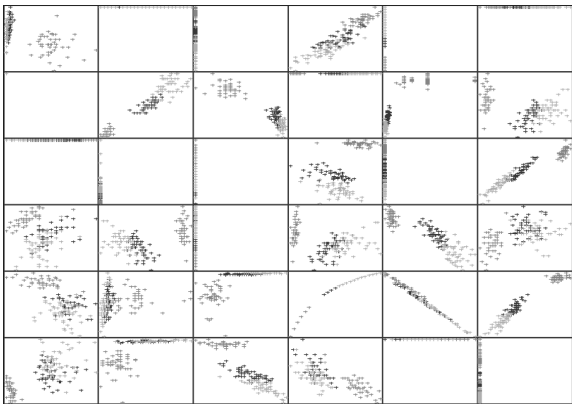


FIGURE 1.14 – Aperçu de la visualisation matrice de nuages de points représentée dans [Venturini *et al.*, 1997].

données, il est souhaitable de limiter le nombre d'axes pouvant être visualisés dans une même représentation.

La figure 1.17 illustre un exemple d'utilisation de la visualisation *"Coordonnées parallèles"* générée à partir des données décrites dans le tableau de la figure 1.16. En effet, on peut remarquer que même si les types des attributs de données contenus dans le tableau à représenter ne sont pas identiques (2 attributs quantitatifs et 1 attribut qualitatif), cette représentation graphique permet de générer un axe vertical pour chacun d'eux. A noter que dans d'autres approches d'application de cette visualisation, les attributs de type qualitatif (ex. classes de données) sont plutôt représentés par la variable visuelle *couleur* et non pas par des axes.

#### 1.3.3.4 Visages de Chernoff

Partant du constat que la détection des expressions faciales dans un visage humain est très facile à percevoir, Chernoff dans [Chernoff, 1973] a proposé une nouvelle visualisation multidimensionnelle appelée *"Visages de Chernoff"*. Le principe de base de cette visualisation consiste à s'appuyer sur la forme, la taille, la couleur des différentes parties du visages (bouche, nez, yeux, etc.) pour représenter des attributs de données d'un jeu de données. Même si quelques travaux comme [Mazza, 2009] ont recensés les principales caractéristiques qui peuvent être utilisées pour représenter des jeux de données avec cette représentation graphiques (voir figure 1.18), ces dernières peuvent être étendues (voir figure 1.19). En effet, dans la figure 1.18, [Mazza, 2009] définit un visage par 6 caractéristiques faciales (attributs visuels) uniquement, tandis que dans la figure 1.19 ce dernier est décrit par 11 caractéristiques. Cette extension du nombre d'attributs visuels définissant cette visualisation est d'un apport majeur dans le cas des bases de données multidimensionnelles car on peut visualiser ainsi plus d'attributs de données simultanément. La figure 1.20 illustre un exemple d'utilisation de cette représentation graphique. A noter que les *"Visages de Chernoff"* peuvent être utilisés dans un plan (2D) et même dans des cartes géographiques (ex. pour représenter des concentrations de populations ou distinguer des communautés).

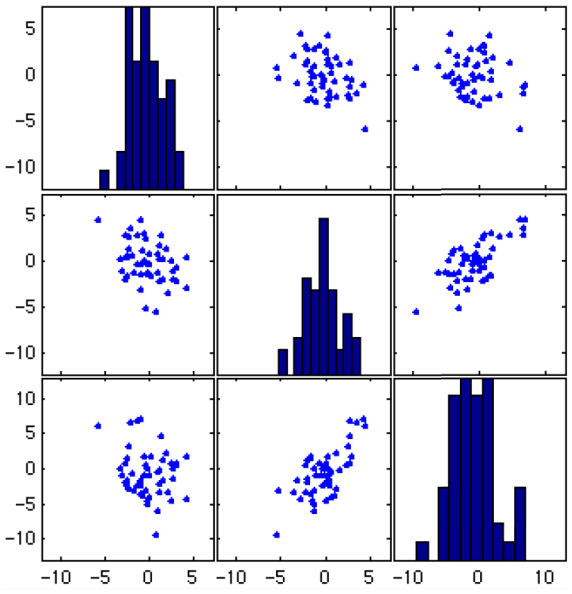



FIGURE 1.15 – Aperçu de la visualisation matrice de nuages de points représentée dans [Grinstein *et al.*, 2001].

| | Age | Weight | Sex |
|---|---|---|---|
| Vincenzo | 32 | 75 | M |
| Piero | 24 | 63 | M |
| Luisa | 28 | 60 | F |
| Giulia | 18 | 58 | F |

FIGURE 1.16 – Tableau descriptif des données visualisées dans la figure 1.17 [Mazza, 2009].

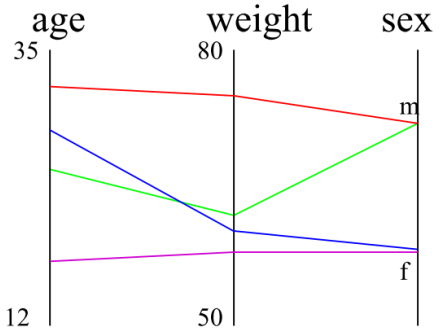



FIGURE 1.17 – Aperçu de la visualisation *"Coordonnées parallèles"* appliquée sur le jeu de données représenté dans la figure 1.16 [Mazza, 2009].

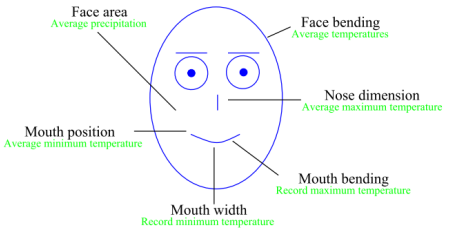



FIGURE 1.18 – Description de quelques caractéristiques faciales d'un visage de Chernoff selon [Mazza, 2009].

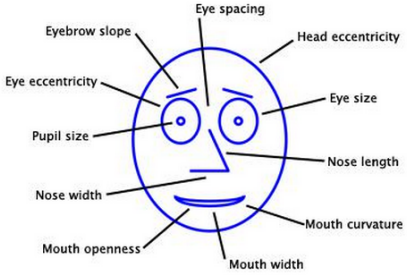



FIGURE 1.19 – Extension des caractéristiques faciales d'un visage de Chernoff. [3]

#### 1.3.3.5 Les icônes *"Sticks-Figure"*

La visualisation *"Sticks-Figures"* proposée dans [Pickett et Grinstein, 1988] appartient à la catégorie des représentations graphiques iconiques. La figure 1.22 illustre un aperçu du rendu visuel de cette technique [Grinstein *et al.*, 2001], représentant un affichage iconographique de 5 images satellitaires de la région des Grands Lacs. À travers cette représentation, [Erbacher et Grinstein, 1994] ont voulu démontrer la capacité remarquable des icônes à montrer toutes les nuances contenues dans un ensemble d'images en fusionnant leurs caractéristiques dans un affichage sous forme de texture avec un effet 3D. En effet, à l'inverse d'un affichage fondé sur les niveaux de gris utilisés dans les images ordinaires, il considère que cette fusion permet surtout de préserver les caractéristiques de chacune des images en les rendant beaucoup plus évidente à travers la texture formée par les icônes.

Le principe de base de cette visualisation consiste à représenter les données utilisateurs par des icônes. En effet, la figure 1.21 illustre quelques exemples d'icônes proposés par [Pickett et Grinstein, 1988] et qui peuvent être utilisés comme base pour représenter des données brutes dans une représentation graphique *"Sticks-Figures"*. On remarque que chacune des icônes est formée par 5 segments. Les angles entre les segments servent à représenter les valeurs des attributs de données. Cela signifie que les 12 exemples présentés dans la figure 1.21 permettent essentiellement de visualiser 4 dimensions simultanément. Représenter une dimension supplémentaire dans cette visualisation consiste à utiliser d'autres propriétés visuelles des icônes. Par exemple si on veut rajouter une cinquième dimensions aux symboles graphiques représentés dans la figure 1.21, on peut exploiter la variable vi-

3. http ://talent.paperblog.fr/3533721/mystere-du-premier-mai-un-sourire-enigmatique/

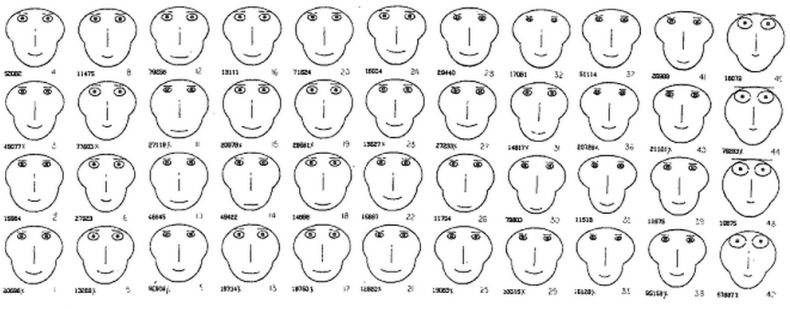


FIGURE 1.20 – Visages de Chernoff [Chernoff, 1973].

suelle *"orientation"* des icônes.

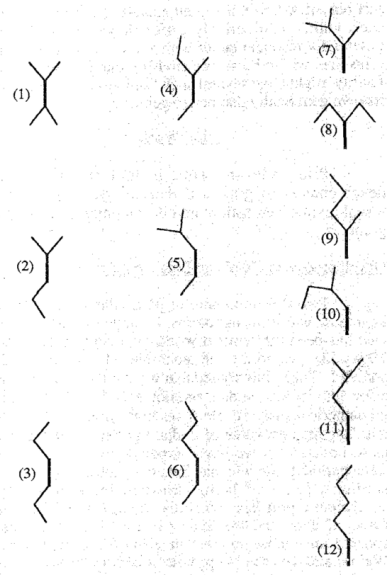



FIGURE 1.21 – Formes générale des icônes proposées dans [Pickett et Grinstein, 1988] pour représenter les données.

#### 1.3.3.6 "Star Plots"

La visualisation *"Star Plots"* s'appuie dans sa description conceptuelle sur des symboles graphiques sous forme d'étoiles pour représenter des données. Pour visualiser les attributs de données caractérisant chaque individu d'un jeu de données, le principe de cette visualisation consiste à s'appuyer sur le rayon de l'étoile (la longueur entre le centre d'une étoile et ses sommets). Cela signifie que si les données sont caractérisées par $n$ attributs de données, alors chacune des étoiles de cette représentation graphique est décrite par $n$ sommets. L'amplitude entre le centre d'une étoile et chacun de ses sommets représente la valeur des attributs de données (voir figure 1.23). Notons que l'angle entre les rayons représentants les $n$ attributs de données est identique autour du cercle formé à partir du centre de chaque

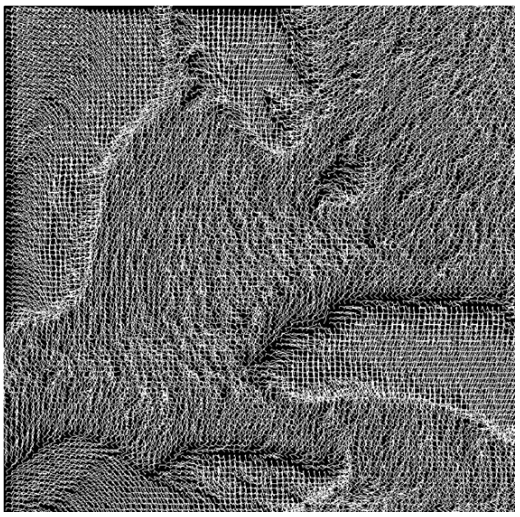


FIGURE 1.22 – Aperçu de la visualisation *"Sticks-Figures Icônes"* [Grinstein *et al.*, 2001].

étoile.

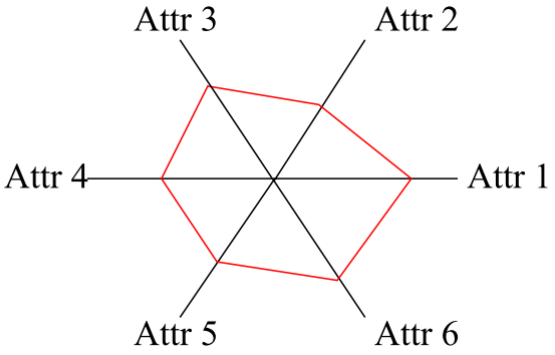



FIGURE 1.23 – Principe d'utilisation de la visualisation *"Star Plots"* [Mazza, 2009].

La figure 1.24 illustre un exemple d'utilisation de la visualisation *"Star Plots"* utilisée pour représenter un sous ensemble (voir tableau 1.1) du jeu de données IRIS [Fisher, 1936]. En effet, on peut distinguer dans cette visualisation que chacune des 15 étoiles (stars) est caractérisée par 4 rayons dont l'amplitude est différente. Cette variation se traduit en effet par les valeurs que prennent les 4 attributs de données (sepal lenght, sepal width, petal lenght, petal width) de chacun des 15 individus du jeu de données représenté dans le tableau 1.1.

Selon le contexte et les objectifs des utilisateurs, l'utilisation de l'une des visualisations décrites ci-dessus peut être d'un apport précieux pour accomplir un processus d'exploration et d'analyse sur un jeu de données multidimensionnel. Cependant, réussir un tel processus dépend aussi des caractéristiques du système de visualisation pour achever cette tâche. Nous exposons dans la section suivante les différents caractéristiques identifiées et proposées par [Wong, 1999] pour une meilleure utilisation des différentes méthodes de fouille visuelle de

| n° individu | Sepal length | Sepal width | petal length | petal width | classe |
|---|---|---|---|---|---|
| 1 | 5,1 | 3,5 | 1,4 | 0,2 | setosa |
| 2 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 3 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 4 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 5 | 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| 6 | 7.0 | 3.2 | 4.7 | 1.4 | versicolor |
| 7 | 6.4 | 3.2 | 4.5 | 1.5 | versicolor |
| 8 | 6.9 | 3.1 | 4.9 | 1.5 | versicolor |
| 9 | 5.5 | 2.3 | 4.0 | 1.3 | versicolor |
| 10 | 6.5 | 2.8 | 4.6 | 1.5 | versicolor |
| 11 | 6.3 | 3.3 | 6.0 | 2.5 | virginica |
| 12 | 5.8 | 2.7 | 5.1 | 1.9 | virginica |
| 13 | 7.1 | 3.0 | 5.9 | 2.1 | virginica |
| 14 | 6.3 | 2.9 | 5.6 | 1.8 | virginica |
| 15 | 6.5 | 3.0 | 5.8 | 2.2 | virginica |

TABLE 1.1 – Description d'un sous ensemble du jeu de données IRIS [Fisher, 1936].

données.

### 1.3.4 Systèmes de fouille visuelle de données

Selon [Wong, 1999] un véritable système de fouille visuelle de données ne doit pas exiger des connaissances de la part des utilisateurs, mais plutôt les guider dans le processus d'exploration et d'analyse de leur ensemble de données. Dans le but de doter les systèmes de fouille visuelle de données de cette capacité, [Wong, 1999] considère que la conception de ce type d'applications doit s'appuyer sur les caractéristiques suivantes :

1. simplicité et utilité : dans un système de FVD, la notion de simplicité concerne plusieurs aspects qu'on peut résumer à : une simplicité d'utilisation (systèmes dotés de mécanismes intuitifs facilitant l'interprétation des résultats fournis), une simplicité d'application (systèmes dotés d'un protocole de communication efficace avec les utilisateurs), une simplicité de relance du processus de visualisation (systèmes dotés de possibilité de personnaliser la structure de données à visualiser pour accélérer le processus de recherche) et une simplicité d'exécution (systèmes fondés sur un minimum d'étapes pour aboutir à des résultats significatifs).
2. autonomie : un système de FVD devrait guider ses utilisateurs dans leur processus d'exploration et d'analyse pour aboutir aux bonnes décisions sur leurs jeux de données plutôt que de leur imposer d'avoir des pré-connaissances pour son utilisation.
3. fiabilité : un système de FVD fiable devrait permettre de fournir une estimation de l'erreur qui peut être engendrée à chaque étape du processus de fouille de données. Lors d'un processus d'analyse visuelle, cette erreur peut être très utile pour mesurer l'impact d'un manque d'informations dans le résultat final.

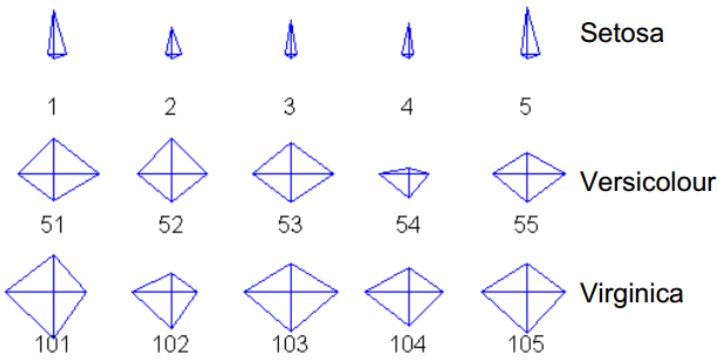



FIGURE 1.24 – Représentation d'un extrait du jeu de données IRIS [Fisher, 1936] en utilisant la visualisation *"Star Plots"*. [5]

4. réutilisabilité : un système de FVD doit pouvoir être utilisé dans différents domaines d'applications afin de réduire l'effort d'adaptation pour toute utilisation dans un nouvel environnement de travail. Cela permet d'augmenter la performance et la portabilité du système.
5. généricité : un système de FVD doit permettre une extension plus facile des connaissances exploitées et/ou une adaptation plus facile de ces dernières à travers des mécanismes simples.
6. sécurité : un système de FVD doit être sécurisé. Utiliser des informations personnelles lors d'une session d'utilisation nécessite donc la présence de mécanismes assurant la protection de ces données. Cela permet en effet de sécuriser toutes informations relatives aux utilisateurs (ex. leurs profils).

Bien que les caractéristiques citées ci-dessus ont permis de définir un modèle général pour résoudre plusieurs lacunes lors du développement des systèmes de fouille visuelle de données, il en demeure quelques limitations. En effet, ces dernières sont liées principalement au processus de visualisation, à la description des jeux de données et au niveau de perception visuelle variable chez les utilisateurs. Nous décrivons dans la section suivante les impacts de ces différentes contraintes pour une meilleure exploitation des techniques de fouille visuelle de données.

### 1.3.5 Le processus de visualisation en fouille visuelle de données

Satisfaire les objectifs d'un utilisateur pour accomplir une tâche donnée en s'appuyant sur une méthode de fouille visuelle de données nécessite à la fois de 1) choisir la meilleure visualisation qui correspond aux caractéristiques des données à représenter, 2) trouver le meilleur appariement entre les attributs de données, décrivant le jeu à visualiser, et les attributs visuels de la visualisation choisie. Or, dans la majorité des systèmes de visualisation, accomplir ces deux tâches nécessite soit une pré-connaissance du domaine de la

5. http ://www.cs.uiuc.edu/∼yyz/teaching/InfoVis-s10/charts.pdf

visualisation d'informations ou l'aide d'un expert du domaine. Si on se réfère donc au processus de visualisation, satisfaire les objectifs d'un utilisateur pour accomplir une tâche donnée, dépend du résultat de la deuxième phase du processus de visualisation.

En effet, c'est durant cette étape que le meilleur appariement entre les variables de données et les variables visuelles satisfaisant le but à atteindre est trouvé. Cependant, plusieurs problèmes concernant cette phase sont à noter. La principale difficulté qui peut se poser concerne la visualisation des ensembles de données multidimensionnelles. En effet, bien que beaucoup d'efforts ont été fournis pour concevoir des visualisations permettant de visualiser simultanément plusieurs attributs de données, les visualisations proposées jusqu'à présent restent limitées dans le nombre de variables visuelles caractérisant un symbole graphique. Cela signifie que la visualisation proposée, à l'issue du processus de visualisation, pour un jeu de données décrit par un nombre important d'attributs de données, ne peut représenter qu'un sous ensemble d'attributs de données. Dans ce cas de figure, l'analyse de l'ensemble de données dans sa globalité est impossible et nécessite un processus d'exploration répétitif consistant à visualiser à chaque fois un sous ensemble d'attributs de données. Cette difficulté peut s'amplifier dans le cas où seule une combinaison donnée des attributs de données (inconnu a priori) peut satisfaire un objectif utilisateur. En effet, dans une telle situation, l'utilisateur doit tester beaucoup de combinaisons pour aboutir à celle qui permet de satisfaire son objectif. Une autre contrainte, concernant le volume de données à représenter, peut s'ajouter. En effet, il est parfois impossible de représenter dans une même visualisation toutes les données caractérisant un jeu de données et quand cela est possible, leur compréhension devient une activité fastidieuse.

Dans le but de résoudre les difficultés indiquées ci-dessus et de faire face aux différents enjeux de la visualisation d'informations, plusieurs groupes de travail, regroupant des chercheurs venant de différentes communautés scientifiques, ont été formés. Le but principal de ces groupes est de travailler en étroite collaboration avec des équipes de recherches académiques et industriels de diverses disciplines pour converger vers des solutions appropriées. C'est dans cette même perspective que s'inscrit le projet VisMaster [6] [Keim *et al.*, 2010] dont les différents acteurs s'appuient sur la nouvelle discipline qui émerge dans le domaine de la visualisation : **la visualisation analytique** *("Visual Analytics")*. Nous présentons dans ce qui suit un survol de cette nouvelle tendance dont l'utilisation est très prometteuse pour résoudre quelques défis cités ci-dessus.

## 1.4 Visualisation Analytique *"Visual Analytics"*

Deux principales définitions de la visualisation analytique (VA) existent dans la littérature. Selon [Thomas et Cook, 2005] la VA peut être définie comme une science analytique de raisonnement simplifié par l'utilisation des interfaces visuelles interactives avancées. Pour [Keim *et al.*, 2010], la VA est le résultat d'une combinaison entre des techniques d'analyse automatique avec des visualisations interactives dont le but est de rendre plus efficace tout processus décisionnel sur des jeux de données complexes et très volumineux. Dans [Keim *et al.*, 2008], l'apport de la VA est présenté comme une combinaison des capacités de calcul des ordinateurs et les caractéristiques du système visuel humain dans un proces-

6. http ://www.vismaster.eu/

sus interactif pour améliorer le processus d'extraction de connaissance à partir des grandes masses de données. La figure 1.25 illustre plus explicitement les différents avantages de la discipline. En effet, on peut constater que l'intérêt grandissant pour cette nouvelle tendance dans le domaine de la visualisation est justifié par sa capacité à fusionner différents processus et traitements avec la visualisation d'informations. Pour [Kohlhammer *et al.*, 2011] les techniques et outils de la VA sont très prometteurs surtout qu'ils permettent de : 1) synthétiser les informations qui peuvent être extraites de grande masse de données, parfois ambigües ou même conflictuelles, 2) découvrir plus facilement des connaissances attendues et même inattendues, 3) fournir dans un délai opportun, une évaluation concise et simple à comprendre pour une meilleure prise de décision.

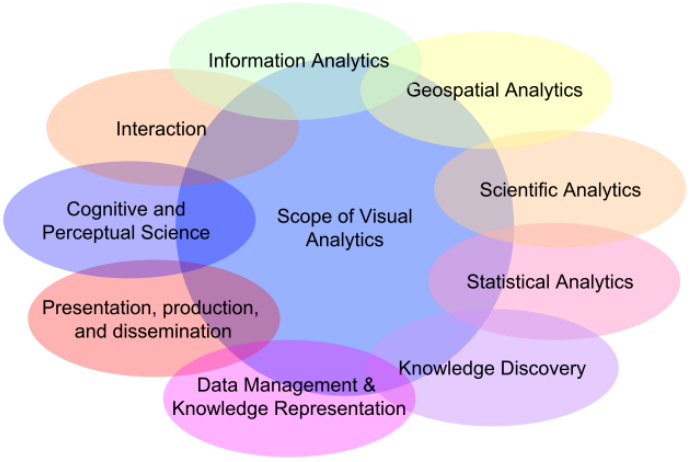



FIGURE 1.25 – Positionnement de la *Visual Analytics* parmi les différents champs et disciplines de recherche scientifique [Keim *et al.*, 2006].

Inversement au principe d'exploration de données en visualisation d'informations proposé dans [Shneiderman, 1996], *"Overview first, zoom/filter, details on demand"*, Keim et al. dans [Keim *et al.*, 2006] ajoute celui de la VA et le définit comme suit : *"Analyze first, Show the Important, Zoom, filter and analyze further, Details on demand"* . Les modifications apportées au processus sont justifiées surtout par la grande taille des données qui peuvent être visualisées. En effet, dans ce cas, il est parfois impossible de présenter une vue d'ensemble des données d'une grande masse de données, puisque cela peut impacter directement les deux opérations de zoom et de filtrage et diminuer leurs effets pour guider un utilisateur durant son processus d'exploration. Pour [Kohlhammer *et al.*, 2011], ce nouveau principe représente une approche astucieuse de jumelage des techniques analytiques avec les techniques interactives de visualisation . La figure 1.26 traduit un aperçu général du processus de la VA.

## 1.5 Conclusion

Nous avons abordé dans ce chapitre les différents aspects caractérisant le domaine de la visualisation d'informations. Nous avons montré à travers plusieurs exemples les atouts

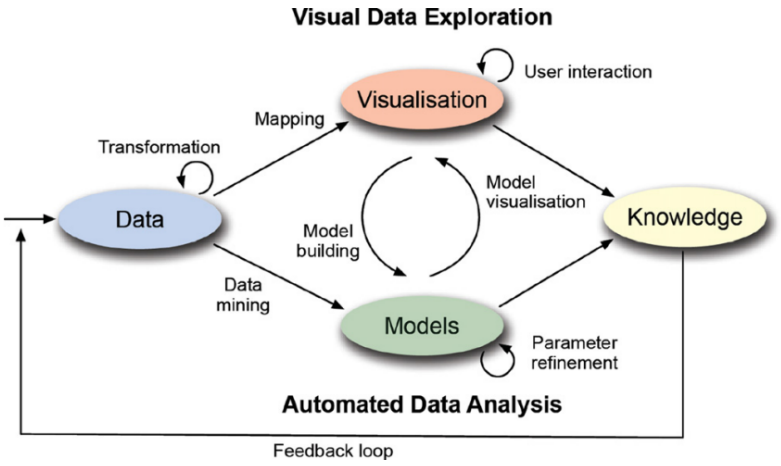



FIGURE 1.26 – Processus de la Visual Analytics [Kohlhammer *et al.*, 2011].

de l'utilisation des différentes techniques de visualisations. En effet, en s'appuyant sur les propriétés perceptives du système visuel humain ces dernières permettent de représenter une grande masse de données, de faciliter les différentes tâches d'exploration et d'analyse de données et surtout d'expliquer des phénomènes très complexes. Cependant, beaucoup de méthodes de fouille visuelle de données s'adressent à des spécialistes, et les utilisateurs novices ou même des experts du domaine d'application visé peuvent avoir des difficultés, et parfois échouer, lors de l'utilisation de telles méthodes. Traiter ce cas d'étude en s'appuyant sur un système de visualisation classique (basé sur un paramétrage - appariement entre les attributs de données et les attributs visuels - manuel), est une tâche très difficile. Dans la littérature, résoudre ce problème passe par l'utilisation d'un assistant permettant d'automatiser ce processus de choix et de paramétrage des visualisations. Un survol des assistants pour la fouille visuelle de données existants dans la littérature est présenté dans le chapitre suivant.

# Chapitre 2

# Assistants Utilisateurs pour la FVD

**Résumé** : Nous nous intéressons dans ce chapitre aux assistants utilisateurs existants pour le choix et le paramétrage des méthodes de fouille visuelle de données. Dans le but d'étudier les avantages et inconvénients de chacun de ces systèmes, nous décrivons leur architecture générale, le mode de leur fonctionnement et leurs contributions et limitations. Une étude comparative des différents assistants utilisateurs est proposée.

## 2.1 Introduction

Beaucoup de méthodes de fouille visuelle de données s'adressent à des spécialistes. Les utilisateurs novices ou même des experts du domaine d'application visé peuvent donc avoir des difficultés, et parfois échouer, lors de l'utilisation de telles méthodes pour deux raisons : 1) les utilisateurs doivent choisir la ou les visualisations représentant de manière efficace leurs données. Cela nécessite une bonne connaissance des visualisations, des données qu'elles peuvent représenter et des objectifs qu'elles peuvent atteindre ; 2) les utilisateurs doivent être capables de trouver le meilleur paramétrage possible de ces visualisations, c'est à dire la meilleure correspondance possible entre les attributs des données et les signes visuels utilisés dans la visualisation. Si une visualisation est mal choisie ou mal paramétrée, il y a de grandes chances que l'utilisateur ne puisse pas atteindre les objectifs qu'il s'est fixés. Dans ce chapitre, nous allons donc nous intéresser aux assistants utilisateurs pour la fouille visuelle de données. L'objectif de ces assistants est de permettre à l'utilisateur de choisir et de paramétrer automatiquement des visualisations. Ces assistants s'appuient généralement sur une modélisation des visualisations, des données et des objectifs de l'utilisateur. Ils utilisent généralement un moteur d'inférence ou un algorithme d'appariement afin de mettre en correspondance les attributs des données avec les attributs (ou signes) visuels des visualisations.

Nous présentons dans la section 2.2 les principaux assistants utilisateurs utilisés dans le domaine de la fouille visuelle de données. Dans notre description de ces assistants, nous exposons leurs objectifs, leurs contributions, leurs architectures, leurs modes de fonctionnement et les différentes limitations qu'ils présentent. Nous donnons dans la section 2.3 un aperçu général des assistants utilisateurs permettant d'automatiser le processus de visualisations dans les autres domaines. Nous terminons par une conclusion.

## 2.2 Assistants de visualisation en fouille de données

Nous présentons dans cette section les principaux assistants utilisateurs recensés dans la littérature et qui correspondent aux problématiques traitées dans notre recherche. Nous illustrons dans le tableau 2.1 un récapitulatif des avantages et des inconvénients des ces systèmes.

### 2.2.1 BHARAT [Gnanamgari, 1981]

Le système BHARAT [Gnanamgari, 1981] peut être considéré comme l'un des premiers travaux de recherche sur les assistants utilisateurs dans le domaine de la fouille visuelle de données. Son objectif principal était de proposer une technique intelligente qui pouvait permettre le remplacement d'un expert en visualisation d'informations dont la présence permanente était nécessaire pour des utilisateurs novices. En effet, pour cette catégorie d'utilisateurs, cet expert avait pour tâche principale de les assister dans : 1) le choix de la meilleure visualisation possible pour bien représenter graphiquement leurs jeux de données, et 2) le paramétrage de la visualisation choisie.

Le système BHARAT a traité deux principales contraintes constatées lors d'une étude

sur l'utilisation des différents systèmes de visualisation d'informations de l'époque. La première contrainte d'utilisation concerne l'absence d'une étape permettant aux utilisateurs d'indiquer leurs objectifs d'analyse et d'exploration à accomplir sur leurs jeux de données. Or, cette étape est indispensable et très importante si l'objectif de la représentation graphique est de générer des visualisations adaptées aux besoins utilisateurs. La seconde contrainte constatée concerne le manque de mécanismes permettant de proposer au moins une visualisation par défaut qui peut s'adapter aux données utilisateur. Cette deuxième problématique se pose lorsqu'un utilisateur n'a pas de préférences lors du processus de choix et de sélection d'une visualisation pour son jeu de données.

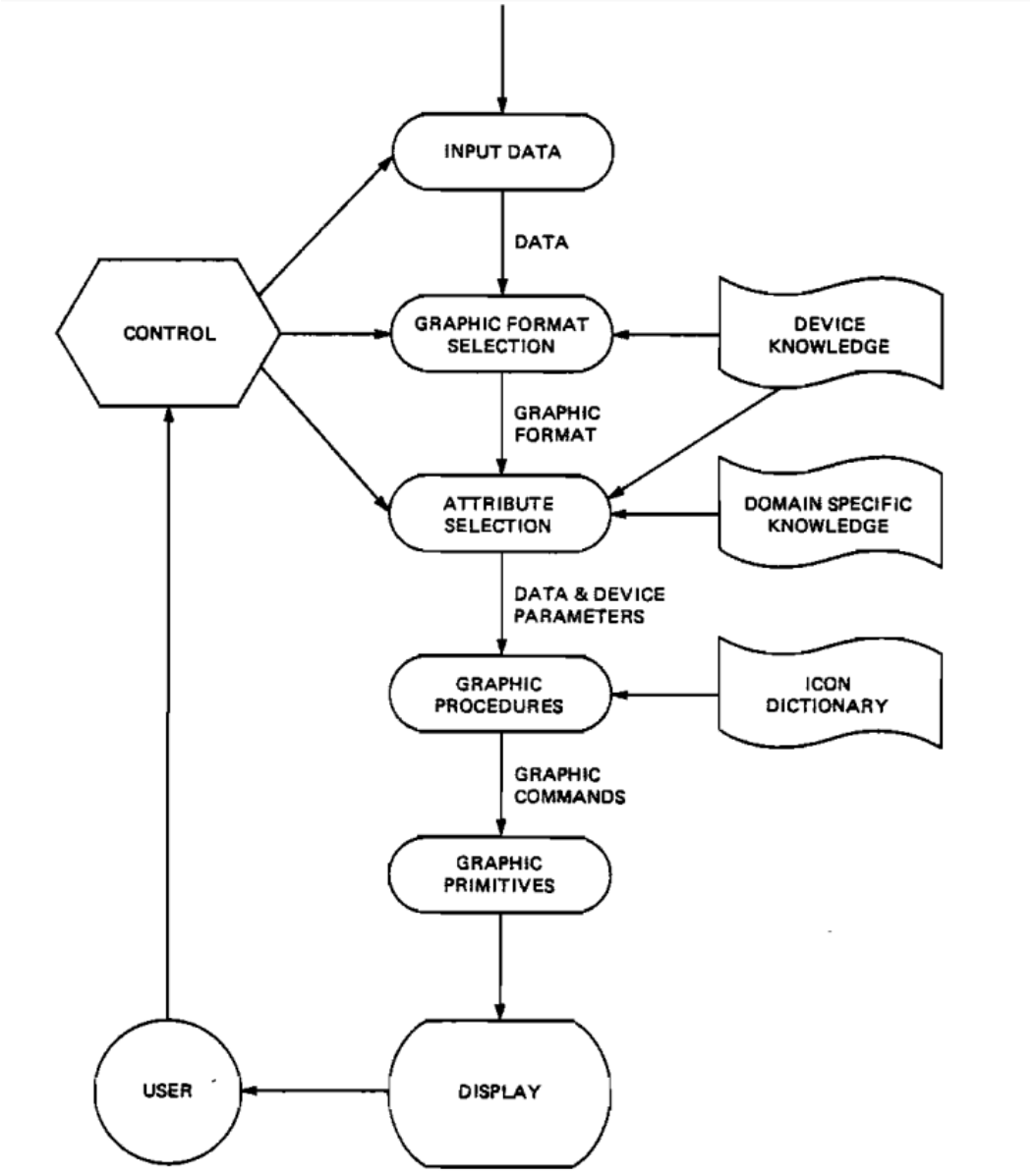



FIGURE 2.1 – Architecture de base du système BHARAT [Gnanamgari, 1981].

Tenant compte du processus d'aide fourni par un expert en visualisation d'informations, l'architecture du système BHARAT a été définie selon un modèle qui caractérise surtout les principales étapes de son raisonnement (voir figure 2.1). Le système s'appuie sur un ensemble de connaissances, défini a priori et nécessaire pour l'accomplissement de chacune des phases de son raisonnement dans le but de guider les utilisateurs novices à bien représenter graphiquement leurs données. Ces connaissances concernent les données à visualiser, les spécifications du domaine d'application, les caractéristiques perceptives des utilisateurs ainsi que quelques détails matériels du périphérique sur lequel les représentations graphiques sont affichées (résolution de l'écran, couleurs, etc.).

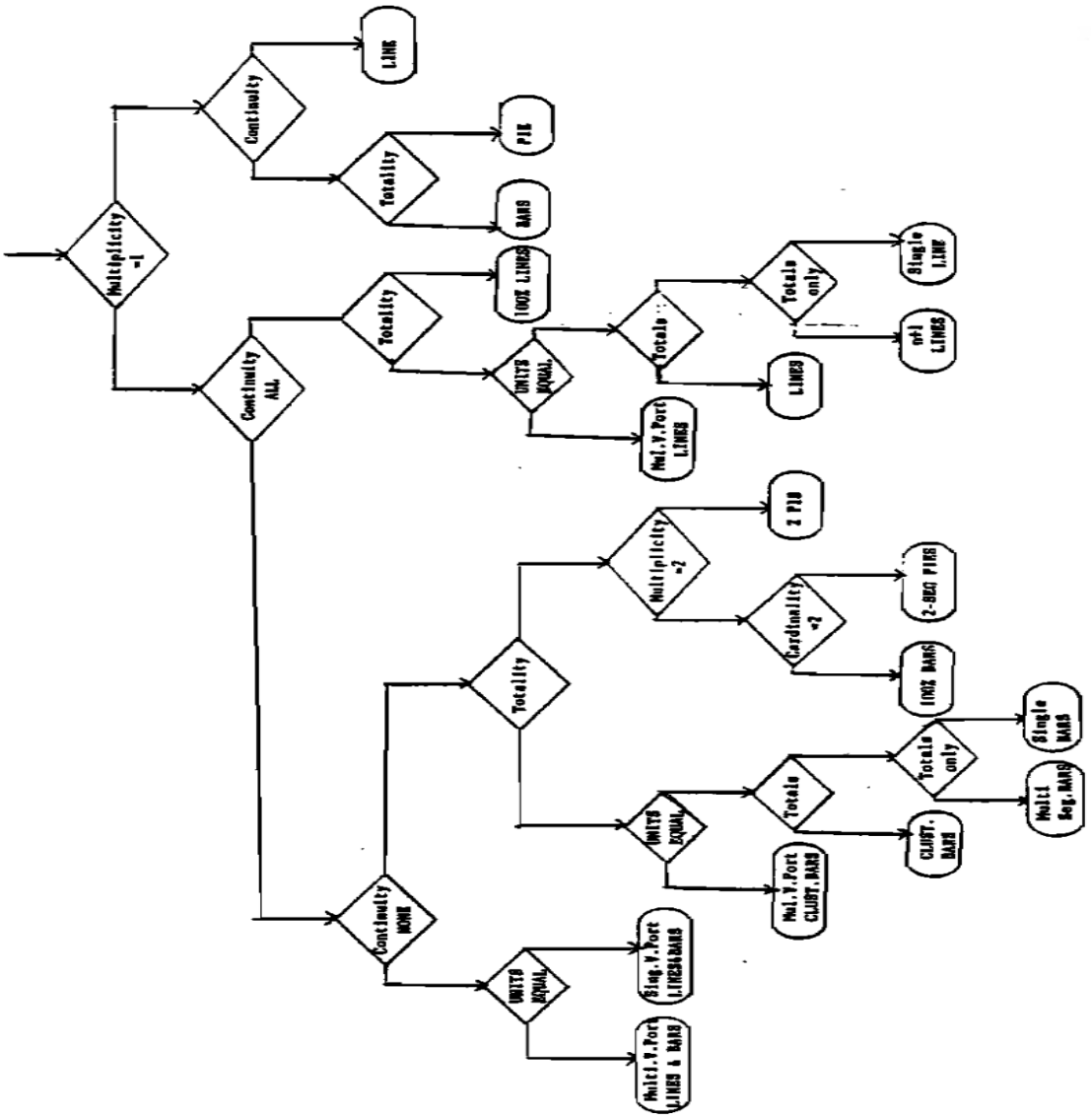



FIGURE 2.2 – L'arbre de décision utilisé par le système BHARAT pour sélectionner et proposer des représentations graphiques aux utilisateurs [Gnanamgari, 1981].

Dans son modèle de données, BHARAT s'appuie sur six propriétés des données (continuité, totalité, cardinalité, multiplicité/nombre d'attributs de données, unité de mesure des données et rang). Pour les objectifs utilisateurs, le système propose un questionnaire défini principalement pour permettre aux utilisateurs de les indiquer avant le choix d'une visualisation par le système. Ce questionnaire comporte deux questions (voir figure 2.3). La première question concerne la nature de la tâche d'analyse à exécuter sur les données (comparaison d'un sous-ensemble de valeurs des données avec le total des valeurs des données, comparaison de l'évolution des données, analyse des tendances dans les données, etc.). La seconde question a pour but d'apporter plus de précision sur les préférences utilisateurs dans la manière de comparer ces données (faut-il faire une comparaison absolue entre données ou pas ?). BHARAT propose ce questionnaire aux utilisateurs avant le chargement des données à visualiser. Ainsi, ce système résout la première problématique indiquée ci-dessus.

Une fois les propriétés des données connues et les objectifs utilisateurs fixés, le système peut procéder à la sélection de la représentation graphique la plus appropriée aux données à visualiser. Durant cette phase, BHARAT utilise un algorithme simple qui s'appuie sur un arbre de décision (voir figure 2.2) dans lequel les visualisations (manipulées par le système) sont classées selon les propriétés des données qu'elles peuvent interpréter. Ainsi, BHARAT permet de proposer aux utilisateurs trois techniques de visualisation (diagrammes en camembert, histogrammes et courbes). Ces visualisations sont représentées graphiquement en 2 dimensions (2D). Dans son processus de choix des visualisations, BHARAT intègre quelques règles déductives dont le but est de résoudre la deuxième problématique mentionnée ci-dessus. En effet, si l'utilisateur ne change pas les réponses au questionnaire sur les objectifs utilisateurs, proposées par le système par défaut (voir figure 2.3), ce dernier peut suggérer au moins une visualisation par défaut aux utilisateurs. Dans ce cas, la visualisation choisie est un diagramme en camembert. Dans le cas où l'utilisateur fixe ses objectifs et répond au questionnaire par des réponses différentes de celles proposées par défaut, le système choisit parmi les deux restantes, la visualisation qui convient le mieux aux besoins utilisateur.

En résumé, les principales contributions du système BHARAT sont la mise en place d'un outil intelligent qui représente un nouveau type de système de visualisation d'informations pour les utilisateurs novices. À travers son architecture et surtout son modèle de données, le système BHARAT a permis de définir le modèle général sur lequel s'appuient les assistants utilisateurs actuels ainsi que les différentes connaissances dont ils devraient se doter (les propriétés des données, les objectifs utilisateurs, les principes de la conception graphique, les résultats des différentes études sur la perception graphique). Cependant, pour représenter ses données, l'utilisateur du système BHARAT est limité à trois visualisations en 2D. De plus, pour paramétrer une visualisation choisie, l'utilisateur est obligé d'accomplir cette tâche de manière manuelle. Ceci peut poser à l'utilisateur plusieurs difficultés pour trouver le meilleur appariement entre les attributs de données et les signes visuels de la représentation graphique sélectionnée, voire même parfois échouer à en trouver un. De plus, cette phase peut devenir une étape fastidieuse à cause de la durée qu'elle peut nécessiter. Même si BHARAT propose un questionnaire aux utilisateurs pour fixer leurs objectifs d'analyse, ce dernier est limité à deux questions. Pour les interactions sur les visualisations, BHARAT n'en propose aucune. Bien que l'évaluation des représentations graphiques par les utilisateurs est une activité qui s'appuie sur leurs niveaux de perception

visuelle, donc se fonde sur des critères subjectifs, aucun mécanisme de contrôle pour cette étape n'est proposé dans le système BHARAT. Une autre limitation du système BHARAT est qu'il ne permet pas de représenter les modèles de visualisations et les bases de données relationnelles.

```
Is this data for TREND analysis (Y/N) ? N

Do you prefer ABSOLUTE comparison (Y/N) ? N
```

FIGURE 2.3 – Le questionnaire sur les objectifs utilisateurs proposé par le système BHARAT [Gnanamgari, 1981].

### 2.2.2 APT [Mackinlay, 1986]

APT [Mackinlay, 1986] est un autre assistant qui a été développé dans la même perspective que BHARAT. Il a été conçu dans le but d'automatiser le processus de génération des visualisations (voir figure 2.4), indépendamment du domaine d'application. La conception de APT est fondée sur un ensemble de règles et permet de représenter graphiquement des bases de données relationnelles. Mackinlay a mis l'accent sur l'importance des interfaces utilisateurs et leurs impacts sur l'accomplissement des différentes tâches opérées par les utilisateurs, dans le but de visualiser leurs jeux de données. Il a aussi évoqué la dépendance des utilisateurs au soutien des experts en visualisation pour l'évaluation des représentations graphiques de leurs données. En effet, l'intervention d'un expert en visualisation d'informations leur était nécessaire du fait qu'elle leur permettait de garantir l'obtention de la meilleure interprétation possible satisfaisant les objectifs fixés.

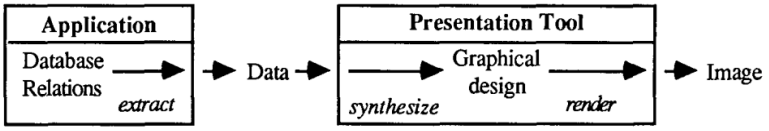



FIGURE 2.4 – Description du processus général de génération automatique des visualisations avec le système APT [Mackinlay, 1986].

À travers ses travaux, Mackinlay s'est fixé deux problématiques importantes à résoudre pour réussir la réalisation de son outil. La première problématique soulevée porte sur l'intégration aisée des représentations graphiques dans un système de visualisation d'informations. Pour cela, Mackinlay a proposé une nouvelle technique pour codifier plus aisément les visualisations fondées sur les différents critères utilisés en conception graphique. La deuxième problématique soulignée concerne la capacité de ces outils à proposer une variété de représentations graphiques pour s'accommoder des différents types de données à visualiser. Pour cela, Mackinlay a proposé de doter ces applications d'un mécanisme pouvant leur permettre l'adaptation des différents symboles graphiques aux jeux de données issues de différents domaines d'application.

Ainsi, Mackinlay a défini un langage graphique pour son système. Ce langage a permis de simplifier la formalisation des différentes visualisations lors du processus de leurs intégrations. La figure 2.5 illustre un exemple d'encodage d'une base de données relationnelles, avec le langage graphique proposé par Mackinlay. En effet, cet encodage traduit la description conceptuelle prise en compte par APT. À travers cette formalisation, on peut constater que le langage graphique utilisé permet d'indiquer plus facilement le nom de la représentation graphique à sélectionner, de fixer les échelles des axes et de simplifier la désignation des positions des données sur les axes. La figure 2.6 illustre la représentation graphique générée à partir de la codification présentée dans la figure 2.5.

*Encodes*(*VertAxis*, [3500, 13000], *ScatterPlot*)
*Encodes*(*HorzAxis*, [10, 40], *ScatterPlot*)
*Encodes*(*Points*, *Cars*, *ScatterPlot*)
*Encodes*(*Position*(*Points*, *VertAxis*), *Price*(*Cars*), *ScatterPlot*)
*Encodes*(*Position*(*Points*, *HorzAxis*), *Mileage*(*Cars*), *ScatterPlot*)

FIGURE 2.5 – Exemple de codification d'une description conceptuelle d'une base de donnée sous le système APT [Mackinlay, 1986].

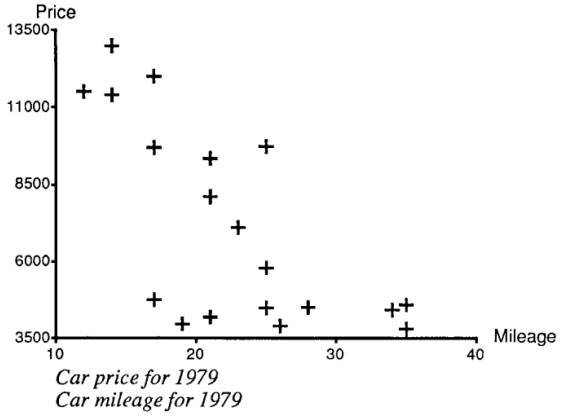



FIGURE 2.6 – La représentation graphique générée par le système APT à partir de la description conceptuelle de la figure 2.5 [Mackinlay, 1986].

Pour évaluer les différentes visualisations qu'il génère et vérifier leur convenance structurelle et sémantique aux données qu'elles interprètent, APT s'appuie sur deux critères. Le premier critère dit d'***expressivité***, détermine si le langage graphique, utilisé pour définir la description conceptuelle de la base de données fournie en entrée, permet d'exprimer exactement toutes les informations contenues dans les données. La nature des données (quantitative, ordinale ou nominale) à visualiser est aussi un facteur discriminant pour le choix des visualisations expressives. En effet, si les données à représenter graphiquement sont ordinales, l'utilisation des histogrammes est la manière la plus significative (expressive) pour les exprimer. Par contre, pour des données nominales, une représentation graphique en nuages de points est la plus appropriée. Le deuxième critère dit d'***efficacité***, détermine si le langage graphique exploite les capacités du support de sortie/affichage (couleurs, ré-

solution de l'écran, etc.), et surtout les caractéristiques du système visuel humain. Sachant que le niveau de perception visuelle des utilisateurs est un facteur déterminant pour juger l'efficacité d'une visualisation, Mackinlay a proposé une classification des différentes tâches perceptives (voir figure 1.9). Cette classification représente une extension de celle proposée par [Cleveland et McGill, 1984] (voir figure 1.8). Pour la définir, Mackinlay s'est basé sur les résultats des différentes études psychophysiques sur la perception visuelle. L'avantage de cette classification est qu'elle permet de tenir compte non seulement des données non quantitatives (nominale et ordinale), mais aussi d'autres signes visuels non mentionnés dans celles de [Cleveland et McGill, 1984]. La figure 1.9 illustre le principe de base de cette classification, dans laquelle l'ordre des signes visuels varie selon le type de donnée. Ainsi, le système APT tient compte des caractéristiques du système visuel humain et fixe l'importance des signes visuels dans chacune des représentations graphiques selon les données qu'elles peuvent visualiser.

En plus des deux critères (d'expressivité et d'efficacité), APT s'appuie sur un ensemble de règles de composition graphique pour générer les différentes représentations graphiques qu'il propose. Ces règles permettent de combiner les différentes primitives graphiques (points, lignes, formes, etc.) à travers des opérateurs de composition graphique pour générer différentes visualisations. Trois opérateurs de composition graphique sont utilisés par APT (*double-axes composition*, *single-axis composition* et *mark composition*). La figure 2.7 illustre un exemple d'utilisation de l'opérateur mark composition. En effet, dans ce cas, APT combine deux signes visuels (la taille et la couleur) pour représenter plus d'informations sur le même graphique.

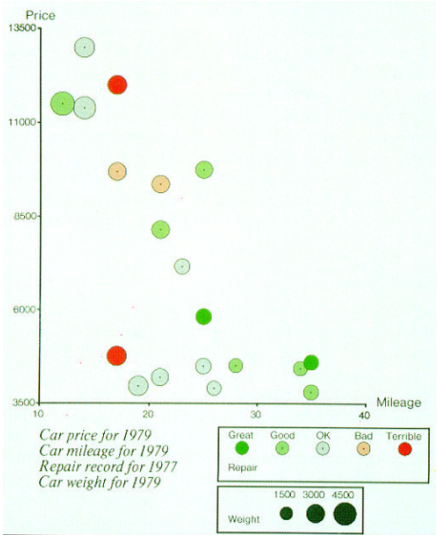



FIGURE 2.7 – Exemple d'utilisation de l'opérateur de composition graphique *mark composition* par APT [Mackinlay, 1986].

Dans son processus de visualisation, APT s'appuie sur un algorithme implémenté avec une approche logique. En effet, cet algorithme utilise la technique de chaînage arrière d'un algorithme déductif appelé *Residue* [Finger et Genesereth, 1985]. Le déroulement de cet algorithme, est basé sur la stratégie *diviser pour régner*. Son déroulement se fait en trois

phases : division, sélection et composition. Les critères d'expressivité et d'efficacité sont appliqués à chaque étape. Un partitionnement des attributs de données, établi selon leur importance, est effectué durant l'étape de division. Cela permet, entre autres, au système APT de les apparier aux primitives graphiques qui peuvent les visualiser. À l'étape de sélection, le système APT sélectionne les attributs de données à visualiser et associe à chacun de ces attributs un indice visuel. La dernière étape consiste à créer une visualisation par composition des attributs visuels sélectionnés. Cependant, dans cette dernière étape, le système APT ne tient pas compte des différentes tâches d'analyse que chacune de ses représentations graphiques permet d'accomplir. Les utilisateurs peuvent être limités dans l'exploration de leurs jeux de données. De plus, estimer la durée réelle, nécessaire pour aboutir au choix approprié d'une représentation graphique peut être difficile à fixer. Un autre inconvénient d'APT est qu'il ne propose que certaines représentations graphiques statiques (2D). L'autre limitation du système est liée au niveau de perception visuelle des utilisateurs. En effet, bien que leurs capacités perceptives puissent varier, APT se base uniquement sur un ensemble de règles de perception définies a priori dans le processus d'évaluation des appariements (attributs de données/attributs visuels) qu'il propose.

### 2.2.3 VISTA [Senay et Ignatius, 1992, Senay et Ignatius, 1994]

Dans le but de faciliter le processus de visualisation pour des experts scientifiques, [Senay et Ignatius, 1992] [Senay et Ignatius, 1994] ont développé un assistant de visualisation appelé VISTA. Ce dernier a pour principal but de générer des visualisations efficaces et expressives en se basant sur un ensemble de connaissances (caractéristiques des données, vocabulaire visuel, primitives visuelles des techniques de visualisations) et de règles (règles de composition graphique et règles de perception visuelle) [Bertin, 1983] [Cleveland et McGill, 1984] [Tufte, 1983]. En effet, VISTA peut être considéré comme une extension du système APT [Mackinlay, 1986] incluant en plus des représentations graphiques en 2D des visualisations en 3D.

Constatant que l'étape d'appariement entre les données à visualiser et les primitives visuelles (voir figure 2.9) est une phase cruciale du processus de visualisation, Senay et Ignatius ont doté leur système VISTA de mécanismes d'évaluation supplémentaires. Ces derniers ont pour intérêt principal d'améliorer le processus d'évaluation de la qualité de l'appariement proposé par le système. Ces mécanismes reposent essentiellement sur une base de connaissances sur les visualisations (voir figure 2.8) gérées par le système. En effet, pour concevoir VISTA, Senay et Ignatius ont basé leur raisonnement sur l'étude des différentes lacunes constatées dans les deux systèmes ApE [Dyer, 1990] et AVS [Upson *et al.*, 1989]. Ces lacunes sont liées principalement à l'absence de support d'aide pour guider les utilisateurs dans la transformation de leurs données de leur état initial brut vers une visualisation représentative.

Conforme au modèle de base du processus de visualisation proposé par [Card *et al.*, 1999], dans son architecture (voir figure 2.8), le système VISTA s'appuie sur trois unités principales. Une unité de données *(Data Unit)* dont le but est de traiter et manipuler les données utilisateurs fournies en entrée du système. Une unité de conception graphique *(Design Unit)* qui a pour principal rôle de définir un appariement entre les données à visualiser et la représentation graphique à générer par le système. Une unité d'interprétation *(Ren-*

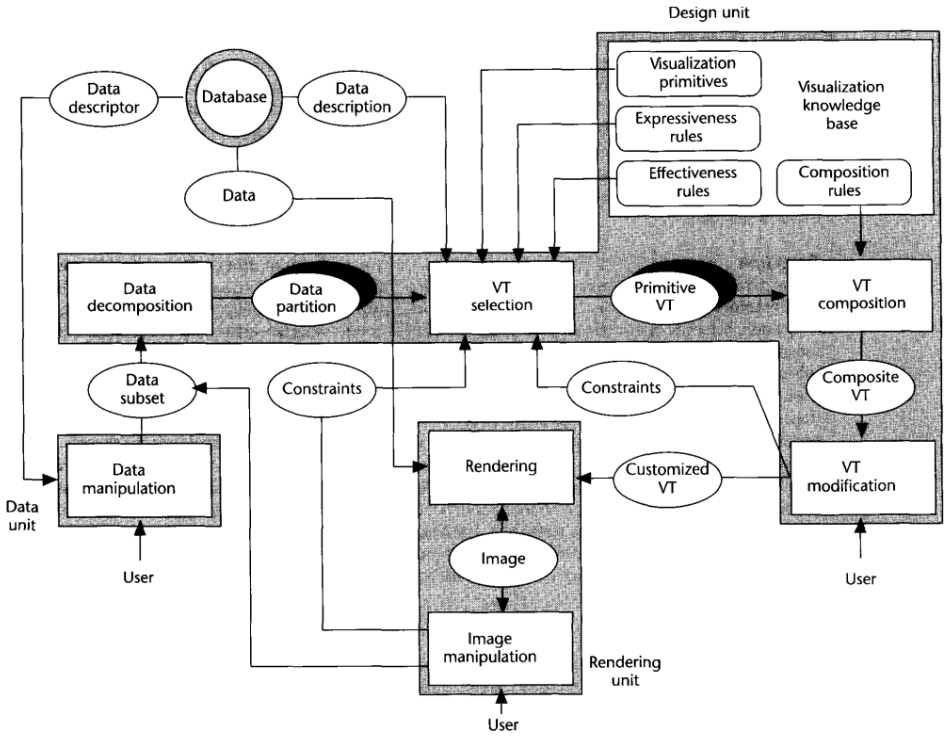



FIGURE 2.8 – Architecture du système VISTA [Senay et Ignatius, 1994].

*dering Unit)* pour générer le rendu graphique (image) correspondant à la description de l'appariement fourni par l'unité de conception graphique. La figure 2.8 montre aussi que le système VISTA inclut divers modules dans chaque unité de son architecture et que l'unité de conception graphique *(Design Unit)* constitue le noyau du système.

La méthodologie de conception graphique employée par le système VISTA est fondée sur une séquence de transformations. La première étape de ce processus consiste à décomposer l'ensemble initial d'attributs de données en sous-ensembles (partitions) d'attributs de données. Pour la deuxième phase, chacune des partitions générées est appariée avec l'une des primitives visuelles gérées par le système. Dans la dernière étape du processus de transformation, les primitives visuelles formées sont combinées pour générer une visualisation composée. La figure 2.10 illustre quelques exemples de visualisations générées avec les règles de composition graphique utilisées par VISTA. Ainsi, en s'appuyant sur une base de connaissances sur les visualisations lors de l'étape de conception graphique VISTA a pour avantage principal de permettre au système de générer des visualisations efficaces et expressives.

Cependant, bien que le système VISTA permette aux utilisateurs de modifier interactivement la description de la visualisation générée par le système, VISTA est limité dans le nombre de règles de perception visuelle qu'il propose. De plus, VISTA ne permet pas de gérer les objectifs des utilisateurs sur les visualisations. Basé sur des règles de perception visuelle définies empiriquement, le niveau de perception visuelle des utilisateurs n'est pas

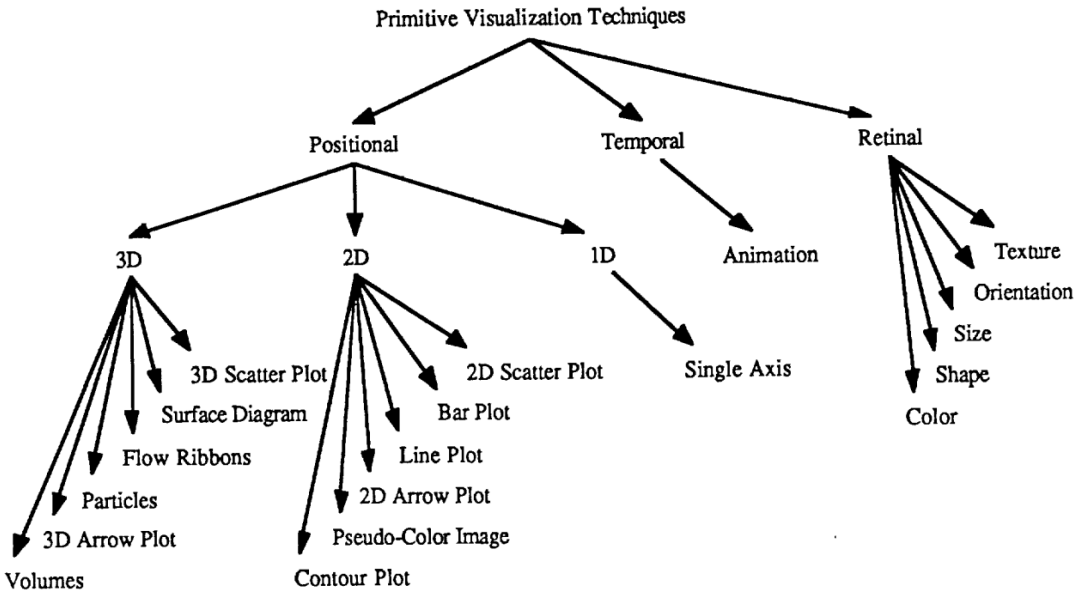



FIGURE 2.9 – Primitives visuelles définies dans le système VISTA [Senay et Ignatius, 1992].

pris en compte dans la phase d'appariement entre les partitions d'attributs de données et les primitives visuelles. Cela a pour inconvénient de n'offrir aucun mécanisme permettant de personnaliser les visualisations à générer par le système.

### 2.2.4 Vis-WIZZ [Lange *et al.*, 1995]

[Lange *et al.*, 1995] ont proposé un système d'aide à la visualisation nommé Vis-Wizz. Un outil qui a été conçu principalement pour assister des utilisateurs non experts durant le processus de visualisation de leurs données. Son principal objectif était de renforcer les systèmes de visualisations qui existaient en leur intégrant un mécanisme permettant de générer automatiquement des visualisations de données. En outre, [Lange *et al.*, 1995] ont tenté à travers leur application d'apporter des solutions aux plus importantes lacunes d'utilisations des systèmes de visualisation de l'époque (connaitre les principes de la conception graphique, module de traitement de données, etc.). En effet, ils avaient remarqué que peu de systèmes de visualisation étaient dotés d'interfaces pour importer les données utilisateurs. En plus, la majorité de ces systèmes ne proposaient pas de techniques de visualisation pour des données multidimensionnelles.

Dans leurs travaux, [Lange *et al.*, 1995] ont considéré d'abord que la mise en place d'une taxonomie regroupant les paramètres les plus importants dans le processus de visualisation était nécessaire dans leur système. Le but de cette taxonomie était de fournir une liste de facteurs pouvant influencer l'évaluation d'une visualisation (voir figure 2.11) quelle que soit la nature du domaine d'application et/ou des données à représenter. Elle offre aussi la possibilité d'établir une classification plus aisée des différentes techniques de visualisations. Cette taxonomie permet également de mieux raffiner le mécanisme de sélection dans le

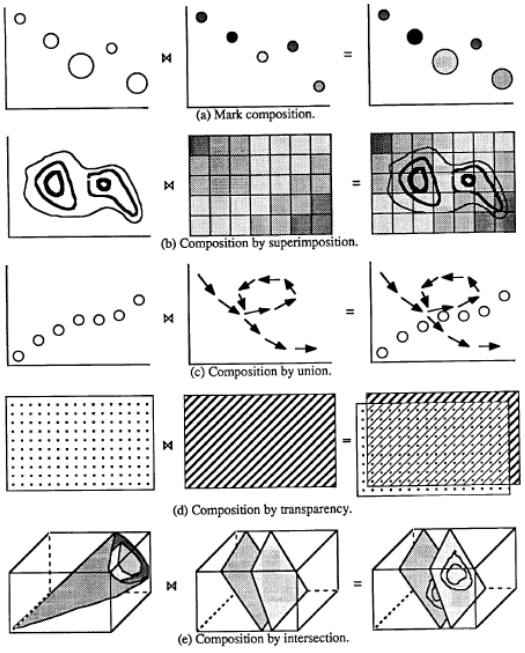



FIGURE 2.10 – Exemple d'utilisation des règles de composition graphique utilisées dans le système VISTA [Senay et Ignatius, 1992].

sens où ces facteurs d'influence réduisent l'espace de recherche exploré par le système. Les principaux paramètres qui ont été recensés et pris en considération sont : les caractéristiques (structurelles et sémantiques) des données, les objectifs et les propriétés des visualisations, le domaine d'application et le niveau de perception visuelle des utilisateurs.

Contrairement au modèle général sur lequel s'appuyaient les assistants de visualisation cités précédemment, [Lange *et al.*, 1995] ont doté leur système d'un module supplémentaire d'évaluation des visualisations (voir figure 2.12). Son apport principal était d'améliorer la qualité des visualisations proposées par le système et générées à partir des règles perceptives. Étant conscients que l'ensemble de ces règles perceptives étaient fondées en général sur des tests empiriques, et ne permettait qu'une évaluation a priori des visualisations, [Lange *et al.*, 1995] ont proposé cette nouvelle technique qui avait pour avantage d'offrir aux utilisateurs une technique d'évaluation a posteriori. Le résultat du processus d'évaluation, avec cette nouvelle technique, était une information complémentaire sous forme de critique (informative ou constructive) sur l'applicabilité des visualisations proposées par le système ainsi que leurs paramétrages.

Afin de simplifier l'évaluation des visualisations, le système définit chacune des techniques de visualisation qu'il gère sous forme d'un descripteur. Pour chaque descripteur, un vecteur de poids est calculé afin de mesurer le taux de convenance de chaque visualisation pour chaque objectif d'analyse. Les degrés d'influence de chacune des caractéristiques des données (incluse dans la taxonomie utilisée par le système) sur chaque visualisation et sur chaque objectif d'analyse sont aussi pris en compte dans la définition des valeurs de chaque vecteur de poids. De plus, afin de simplifier davantage l'évaluation des visualisations, l'uti-

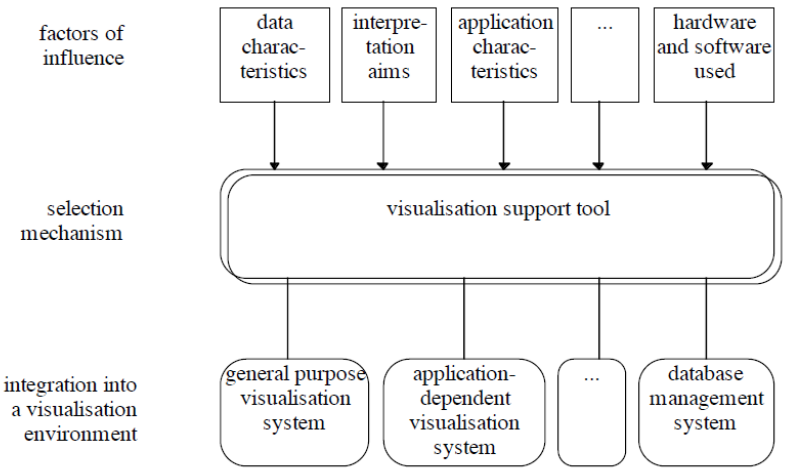



FIGURE 2.11 – Liste des facteurs d'influence du système Vis-Wizz [Lange *et al.*, 1995].

lisateur fixe ses préférences, au début de chaque session d'utilisation du système Vis-Wizz, sur une échelle de priorités pour chacun des objectifs d'analyse qu'il souhaite accomplir sur ses données. Ces préférences sont ensuite traduites par le système sous forme d'un vecteur de poids dit vecteur de priorité.

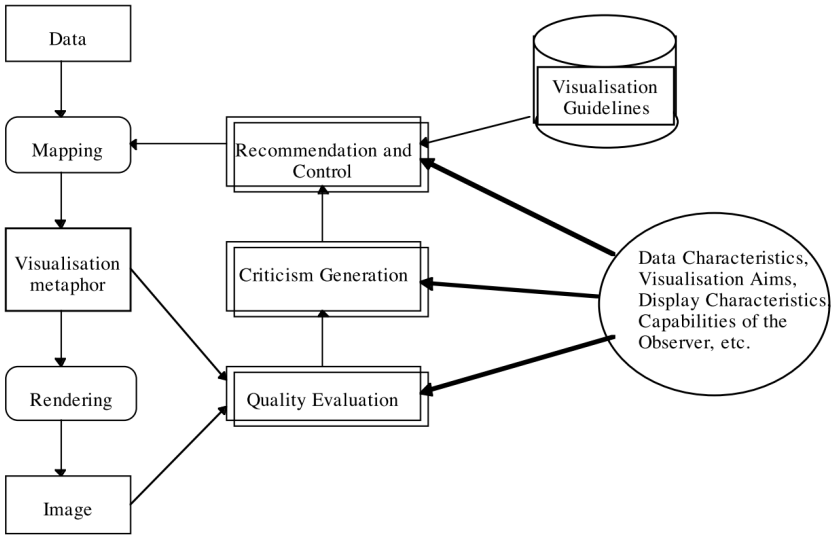



FIGURE 2.12 – Modèle général du système Vis-Wizz [Lange *et al.*, 1995].

Dans le but de proposer aux utilisateurs les visualisations les plus appropriées à leurs jeux de données, Vis-Wizz s'appuie sur le vecteur de priorités qu'il fixe en entrée. Ce dernier est comparé, selon un seuil de similarité fixé a priori par l'utilisateur, à chacun des vecteurs de poids des descripteurs des visualisations gérées par l'outil. À l'issue de cette étape de comparaison, si le système trouve une ou plusieurs techniques de visualisation(s)

satisfaisant le seuil de similarité, elle(s) est(sont) suggérée(s) à l'utilisateur. Sinon (aucune visualisation ne satisfait le seuil de similarité), le système propose aux utilisateurs deux solutions pour y remédier. Dans sa première solution, le système analyse pour chaque visualisation la cause de sa non-conformité avec le vecteur de priorité utilisateur. Les résultats de cette analyse sont affichés par le système directement à l'utilisateur sur une interface (voir figure 2.13) illustrant les contraintes d'évaluation constatées sur ses priorités d'analyse. L'utilisateur pourra ainsi réajuster interactivement ses préférences initiales. Dans le cas où aucune visualisation n'est trouvée, le système propose à l'utilisateur d'utiliser la deuxième solution qui consiste à combiner au moins deux techniques de visualisations utilisées par Vis-Wizz (voir figure 2.14) pour représenter le jeu de données utilisateur. En effet, dans un premier temps, le système découpe le jeu de données utilisateur en sous-ensembles de données. Ensuite, pour chaque sous-ensemble, une technique de visualisation est choisie. Afin de permettre aux utilisateurs une bonne compréhension de la représentation graphique finale, Vis-Wizz repère le minimum de sous-ensembles de données.

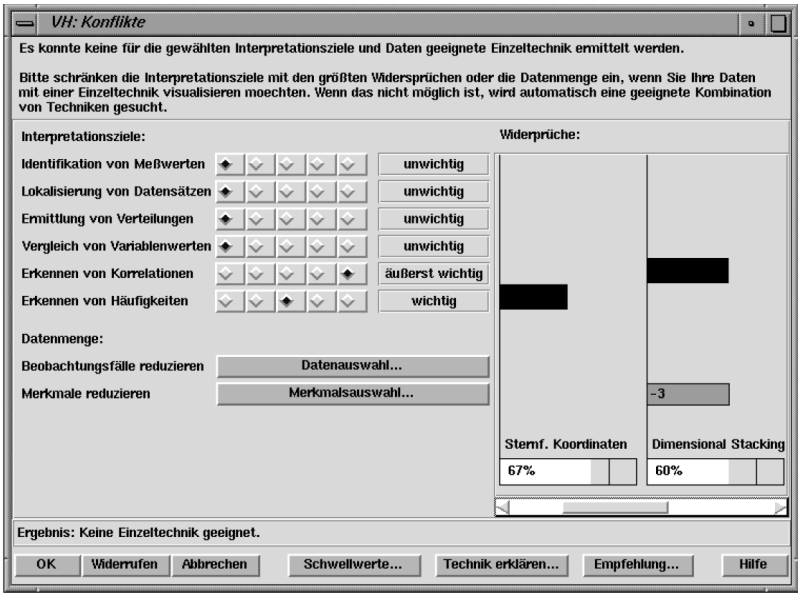



FIGURE 2.13 – Interface proposée par le système Vis-Wizz à l'utilisateur pour réajuster ses préférences initiales sur les propriétés des objectifs d'analyse [Lange *et al.*, 1995].

Malgré ses différents avantages, le système Vis-Wizz a quelques limitations. Sa première limitation est liée à son interface. En fait, l'utilisateur doit choisir manuellement les variables de données à afficher, et indiquer aussi manuellement au système celles qui doivent être corrélées. Cette étape peut rapidement devenir fastidieuse et coûteuse en temps d'exécution, à chaque fois que l'utilisateur désire créer une nouvelle visualisation de ses données. Sa seconde limitation est liée à la nature de ses visualisations. En effet, le système est restreint à des représentations graphiques 2D statiques et sert donc juste de support d'aide pour des utilisateurs novices mais ne permet pas de générer des visualisations réelles.

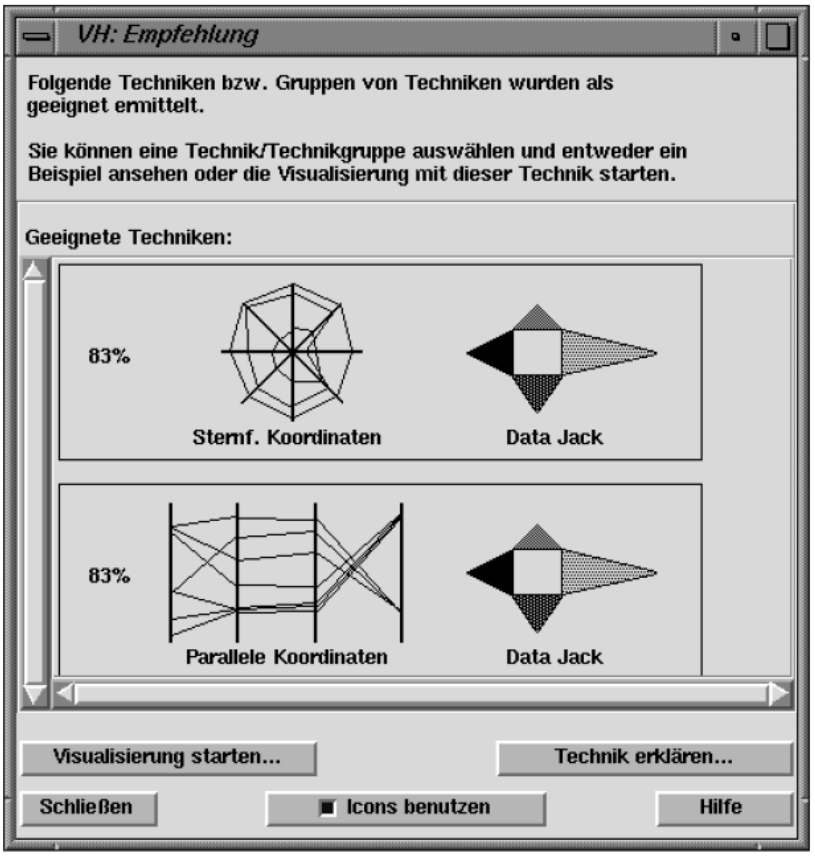



FIGURE 2.14 – Interface proposée par le système Vis-Wizz à l'utilisateur pour combiner deux visualisations afin de satisfaire ses préférences d'analyse initiales à accomplir sur ses données [Lange *et al.*, 1995].

### 2.2.5 ViA [Healey *et al.*, 1999, Healey *et al.*, 2008]

ViA est un autre assistant visuel interactif [Healey *et al.*, 1999] qui a été conçu dans le même but que les précédents outils mais qui utilise un autre type de raisonnement. L'objectif principal de ViA est d'aider les utilisateurs à créer des visualisations perceptivement efficaces pour représenter, explorer et analyser un large et complexe ensemble de données multidimensionnelles. De plus, le défi attendu à travers la conception du système ViA était de développer un outil assurant l'affichage de plusieurs données simultanément sans que les utilisateurs perdent la capacité de comprendre ce qu'ils voient. C'est pour cette raison que l'approche proposée dans ViA s'appuie principalement sur un ensemble de directives visuelles [Healey, 1996] [Healey et Enns, 1998] [Healey et Enns, 1999] définies sur la base d'études expérimentales psychophysiques. L'intérêt de l'utilisation de ces directives était aussi d'exploiter les caractéristiques du système visuel humain pour atteindre le double objectif : une visualisation efficace et expressive.

Dans son architecture (voir figure 2.15), le système ViA s'appuie sur deux modules. Le premier module comporte un moteur de recherche qui sert à générer des appariements

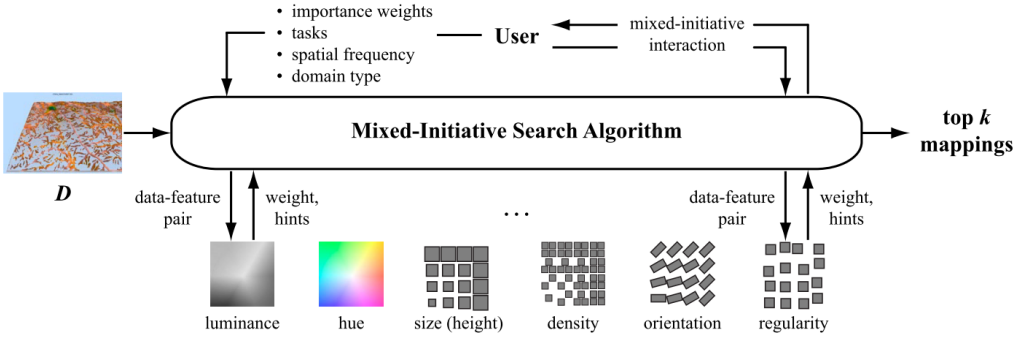



FIGURE 2.15 – Architecture du système ViA [Healey *et al.*, 2008].

entre les attributs de données et les attributs visuels. Ce dernier est fondé sur un algorithme incrémental de recherche en temps réel (LRTA*) [Korf, 1990]. Le deuxième module regroupe un ensemble de moteurs d'évaluations dont chacun est associé à une directive visuelle. Le but de chaque moteur d'évaluation est d'évaluer un attribut visuel (couleur, luminance, etc.) dans l'appariement proposé par le moteur de recherche selon un ordre prédéfini de son degré d'importance [Cleveland et McGill, 1984] et les critères perceptifs de son affectation aux valeurs de données. Par exemple, dans ViA l'attribut visuel *"couleur"* ne peut pas avoir plus de sept occurrences possibles (sept couleurs différentes) en même temps dans la même visualisation. Ainsi, l'attribut de données avec lequel est associé l'attribut visuel *couleur* ne peut pas prendre plus de sept valeurs distinctes.

Le processus d'appariement entre les attributs de données et les attributs visuels proposé par ViA est une tâche itérative dont le résultat final (fourni par le système aux utilisateurs) dépend de : 1) un ensemble de règles de perception visuelle [Healey *et al.*, 2008], 2) les réactions de l'utilisateur vis-à-vis des différentes recommandations (ex. modification des poids des attributs de données) que lui propose le système pour améliorer l'appariement proposé. Dans cette phase du processus de visualisation, le système ViA s'appuie sur deux types d'informations. Le premier type d'informations est dit informations objectives et concerne les caractéristiques des attributs de données du jeu de données à visualiser. Le deuxième type est dit informations subjectives et concerne les préférences utilisateurs. Pour chaque affectation possible entre les attributs de données et les attributs visuels, le système ViA teste : 1) les interférences visuelles qui peuvent exister entre deux affectations différentes dans le même appariement généré, 2) l'applicabilité des tâches fixées sur les attributs visuels choisis, 3) l'adéquation entre le type de données de l'attribut de données et le type visuel de l'attribut visuel de chaque affectation possible, 4) la représentation de la fréquence spatiale d'un attribut de données (nombre de valeurs qu'il peut prendre) par l'attribut visuel choisi.

Visant la simplification d'utilisation de leur outil pour épargner aux utilisateurs tout effort de manipulation ou de compréhension des directives visuelles utilisées, Healey et al. ont fondé le principe de fonctionnement de leur système sur un ensemble de questions simples et limitées. Ces questions concernent la structure de l'ensemble de données à visualiser et le type des tâches d'analyse et d'exploration qu'ils veulent accomplir. Les réponses à ces questions sont récupérées sous forme de poids (importance). Ces poids sont combinés

avec les moteurs d'évaluation pour identifier et rechercher intelligemment l'appariement répondant efficacement aux besoins visés par les utilisateurs. Une fois obtenu, cet appariement final est utilisé par le système pour générer la visualisation correspondante qui sera présentée à l'utilisateur.

En permettant aux utilisateurs de spécifier leurs préférences, le système ViA favorise plus d'interactivités durant le processus de recherche du meilleur appariement entre les attributs de données et les attributs visuels. Bien que le système ViA ait été conçu pour visualiser des bases de données multidimensionnelles, le système ne gère qu'un nombre limité d'attributs visuels. Cela a pour inconvénient de n'afficher qu'un nombre restreint d'attributs de données simultanément. Un autre inconvénient de ViA est qu'il est limité à une seule visualisation, une sorte de cartographie bidimensionnelle. En s'appuyant sur une petite variation des poids dans l'évaluation des appariements générés par le moteur de recherche, ViA réduit les possibilités de tester une multitude d'appariements possibles. Cela diminue la chance d'obtenir à chaque session d'utilisation la meilleure solution possible (le meilleur appariement possible).

## 2.3 Autres assistants utilisateurs

Proposé par [Casner, 1991], le système BOZ a pour principal but d'automatiser le processus de génération des visualisations. Parmi les avantages de ce système on peut citer sa capacité à proposer différentes représentations graphiques pour visualiser un même ensemble d'informations. Permettant de rationaliser la recherche des informations nécessaires à un utilisateur, BOZ a donné des résultats significatifs à travers son application pour l'extraction d'informations sur les horaires de compagnies aériennes. Cependant, BOZ nécessite une description textuelle des tâches à accomplir sur les représentations graphiques qu'il propose, ce qui est contraignant pour ses utilisateurs. Une autre contrainte de ce système est qu'il ne prend pas en considération les différentes règles de perception visuelle pour faciliter le processus d'interprétation des visualisations. Cela nécessite parfois un effort considérable de la part de ses utilisateurs.

Constatant les difficultés d'utilisation des outils de visualisation pour analyser de grande quantité de données, [Miceli, 1994] a proposé MDV un assistant utilisateur pour accompagner des scientifiques durant ce processus. Le principal apport de MDV réside dans sa capacité à résoudre les contraintes liées au temps que peuvent rencontrer ces scientifiques pour comprendre des données provenant de différentes sources. En effet, avant d'exploiter ces données, les utilisateurs des outils de visualisation scientifique doivent comprendre leurs différentes structures. Cela engendre généralement une durée considérable avant de pouvoir générer une visualisation permettant d'analyser les données à interpréter. Même si le système MDV propose à travers son architecture une solution prometteuse pour réduire la complexité du processus de génération des visualisations, il s'appuie sur un paramétrage manuel de ces dernières. Une autre contrainte de ce système est qu'il ne propose qu'une seule visualisation à ses utilisateurs. De plus, il ne leurs permet pas de spécifier leurs objectifs.

Proposé dans [Nakamura *et al.*, 1995], le système $C^2IMA$ a pour principal but d'assister des scientifiques dans le choix des visualisations appropriées à leurs objectifs et les

propriétés des données qu'ils veulent visualiser. S'appuyant sur la classification des visualisations proposée dans [Wehrend et Lewis, 1990] pour suggérer des visualisations aux utilisateurs, le système $C^2IMA$ ne fournit aucun mécanisme permettant de faire le meilleur choix si le nombre de représentations graphiques proposées est important. $C^2IMA$ s'appuie sur un autre système de visualisation AVS [Upson *et al.*, 1989] pour générer le rendu visuel des visualisations retenues par les utilisateur. Toute modification du paramétrage de la visualisation choisie se fait donc sur un autre outil que $C^2IMA$.

Basé dans sa première version uniquement sur la matrice des objectifs des visualisations proposée dans [Wehrend et Lewis, 1990], le système GADGET [Fujishiro *et al.*, 1997] a été amélioré dans sa seconde version [Fujishiro *et al.*, 2000] par l'ajout de la taxonomie proposée dans [Shneiderman, 1986]. L'apport de cette dernière réside dans la description détaillée des tâches qu'un utilisateur désire accomplir sur ses données. On y trouve dans la taxonomie proposée dans [Shneiderman, 1986] une distinction entre les différents types de données (1D, 2D, 3D, temporel, multidimensionnel , hiérarchique, relationnel). Cela permet de faciliter le processus de suggestion des visualisations en fonction de la nature des données qu'un utilisateur peut spécifier au système. L'ajout de cette taxonomie permet donc une utilisation plus large du système GADGET. En effet, en plus de ses capacités dans le domaine de la visualisation scientifique, la combinaison des deux taxonomies proposées dans [Wehrend et Lewis, 1990] et [Shneiderman, 1986] étend son utilisation dans d'autres domaines de la visualisation d'informations. Parmi les limites du système GADGET, notons l'absence d'interactions et de mécanismes de navigation dans les visualisations qu'il gère. Cela peut impacter de manière directe l'interprétation des visualisations par les utilisateurs. Bien que le système gère plusieurs visualisations, il ne propose qu'une seule visualisation à la fois aux utilisateurs. GADGET ne propose aucun mécanisme aux utilisateurs pour renseigner l'importance des attributs de données qu'il veulent représenter pour trouver l'appariement qu'ils souhaitent dans la visualisation proposée. Le système ne prend pas en considération les règles de perception visuelle dans son processus de visualisation.

## 2.4 Conclusion

Nous avons illustré dans ce chapitre les principaux assistants utilisés dans le domaine de la fouille visuelle de données. Nous avons expliqué pour chacun de ces outils la technique sur laquelle il se base pour permettre à des utilisateurs novices de choisir et de paramétrer automatiquement des visualisations. Ces approches s'appuient généralement sur une modélisation des visualisations, des données et des objectifs de l'utilisateur. Ils utilisent bien souvent un moteur d'inférence ou un algorithme d'appariement afin de mettre en correspondance les attributs des données avec les attributs visuels des visualisations. Cependant, les approches existantes sont peu nombreuses et ont certaines limitations (voir tableau 2.1). Par exemple dans certains systèmes le nombre de visualisations gérées est limité, ou bien les objectifs utilisateurs ne sont pas pris en considération lors du processus de suggestion des visualisations. Le mode d'appariement proposé dans la majorité de ces système est aussi contraignant pour des utilisateurs novices car il est manuel. Une autre contrainte importante constatée dans les assistants de visualisation existants concerne l'absence de personnalisation des paramétrages des visualisations en fonction du niveau de perception

| Assistant utilisateur | Visualisations | Objectifs utilisateur | Modèle de données | Paramétrage visualisations | Suggestion de visualisations | Interactions |
|---|---|---|---|---|---|---|
| BHARAT | visualisations 2D statique (pas d'interactions) | tient compte via un questionnaire | propriétés des données | appariement manuel | une seule. Néglige les règles de perception visuelle | non |
| APT | certaines visualisations 2D statique (pas d'interactions) règles de composition | négligés | absent | appariement manuel | une seule. Utilise modèle général règles de perception visuelle | non |
| VISTA | visualisations 2D et 3D statique (pas d'interactions) règles de composition | négligés | propriétés des données | appariement manuel | une seule. Utilise modèle général règles de perception visuelle | non |
| Vis-WIZZ | visualisations 2D statique (pas d'interactions) | tient compte via un questionnaire | propriétés des données | appariement manuel | plusieurs. Utilise modèle général règles de perception visuelle | via menu classique |
| ViA | cartographie 6 attributs visuels pour générer les visualisations | tient compte via un questionnaire | propriétés des données | appariement automatique | une seule. Utilise modèle général règles de perception visuelle | via menu classique |

TABLE 2.1 – Tableau comparatif recensant les principales caractéristiques des assistants utilisateurs utilisés dans le domaine de la fouille visuelle de données.

visuelle des utilisateurs. Pour cette raison, nous nous sommes intéressés à une approche interactive ***"Algorithme Génétique Interactif"*** qui a démontré des résultats significatifs pour résoudre cette contrainte (voir chapitre 7). Nous présentons donc dans le chapitre suivant un survol de ce type d'algorithmes génétiques.

# Chapitre 3

# Algorithme Génétique Interactif

**Résumé** : Nous présentons dans ce chapitre un état de l'art des algorithmes génétiques interactifs (AGIs). Dans notre illustration de ce type d'algorithmes stochastiques, nous nous appuyons sur les algorithmes génétiques classiques. Dans notre description des AGIs, nous mettons en avant leurs principes de base et les problématiques de leur utilisation. Nous exposons aussi certaines solutions proposées dans la littérature pour les résoudre. Nous illustrons également les différents domaines d'application des AGIs en mettant l'accent sur le domaine de la fouille visuelle de données.

## 3.1 Introduction

Les algorithmes génétiques (AG) [Holland, 1975] [Goldberg, 1989] [Man *et al.*, 1996] sont des approches d'exploration stochastique basées sur le principe de la sélection naturelle et les mécanismes de la théorie d'évolution [Darwin, 1859]. L'efficacité de ces techniques à résoudre des problèmes complexes (ex. problèmes d'optimisation) et leurs capacités à leur trouver des solutions optimales, forment leurs principaux atouts d'utilisation dans différents domaines scientifiques. Dans ce chapitre, nous présentons un aperçu général de la version interactive de ce type d'algorithme appelée ***"Algorithme Génétique Interactif (AGI)"*** [Dawkins, 1986]. Dans notre assistant, cette approche est utilisée pour résoudre les difficultés liées à l'automatisation du processus d'optimisation du paramétrage entre un jeu de données utilisateur et une visualisation donnée, que nous considérons comme étant un problème d'optimisation subjectif. Pour [Smith, 1991], l'utilisation d'un AGI pour résoudre ce type de problèmes est justifié principalement par le fait que l'évaluation humaine est la mesure qu'ils utilisent pour estimer la qualité des résultats fournis par un processus génétique. L'utilisation d'un AGI est donc une solution prometteuse qui peut contribuer à résoudre les différentes contraintes du processus de paramétrage des visualisations en fouille visuelle de données, constatées dans les systèmes existants (voir chapitre 2).

Dans les sections suivantes, nous commençons par présenter le principe général des algorithmes génétiques. Ensuite, nous définissons ce qu'est un AGI et les motivations du choix d'utilisation de cette approche pour résoudre le problème d'optimisation du paramétrage des visualisations dans notre assistant. Puis, nous présentons la différence qui existe dans le processus génétique d'un AG et d'un AGI. Nous proposons aussi un survol des différents domaines d'applications de ce dernier (AGI) en mettant l'accent sur ses différentes utilisations en fouille visuelle de données. Nous exposons ensuite, les difficultés d'utilisations des AGIs que nous avons recensées dans la littérature ainsi que les solutions proposées pour les résoudre. Nous terminons par une conclusion.

## 3.2 Principe général des algorithmes génétiques (AG)

Résoudre un problème donné en s'appuyant sur un AG consiste dans un premier temps à définir la représentation générale de ses solutions potentielles et une fonction objectif pour les évaluer. Une solution est généralement appelée *individu* ou *chromosome* et se constitue d'un ensemble de paramètres *(gènes)* caractérisant le problème à résoudre. Une fois établie, cette solution est utilisée pour générer de manière aléatoire un ensemble de solutions dit *population* (notée également $P(t)$). Cette opération représente la première étape du processus de base du déroulement d'un AG (voir figure 3.1). Une fois générés, les individus de cette population sont évalués en se basant sur la fonction de qualité prédéfinie afin de sélectionner ceux qui représentent les meilleures solutions du problème traité. Sachant que le processus génétique s'exécute de manière itérative, à chaque itération deux opérateurs génétiques (croisement et mutation) sont appliqués sur les individus sélectionnés pour créer une nouvelle génération de la population (notée également $P(t+1)$ à la génération $t+1$). Le cycle précédent (évaluation/sélection, croisement et mutation) est appliqué sur les nouveaux individus générés. Ce processus est répété un grand nombre de fois jusqu'à ce qu'une

solution suffisamment satisfaisante soit trouvée ou qu'un critère d'arrêt, prédéfini en entrée de l'AG, soit vérifié.

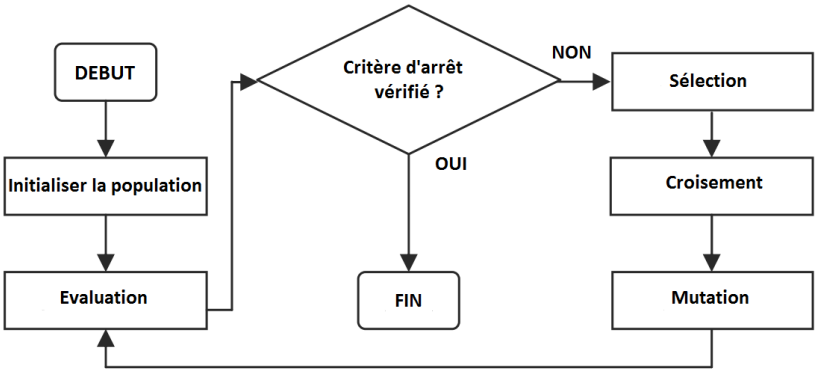



FIGURE 3.1 – Principe de base d'un algorithme génétique.

## 3.3 Définition d'un Algorithme Génétique Interactif (AGI)

Bien que l'utilisation des AGs a contribué à résoudre des problèmes complexes dans différentes disciplines, l'efficacité des résultats fournis par ces algorithmes dépend étroitement de la fonction de *"fitness"* (connue aussi sous les noms : fonction de qualité et fonction objectif) utilisée dans le processus génétique. En effet, cette dernière a pour but de mesurer la qualité des solutions générées dans une population $P(t)$ à chaque itération de l'algorithme. Si pour un problème donné à traiter avec un AG, la fonction de fitness est difficile à définir et dépend de la subjectivité et la cognition humaine, cette dernière est remplacée par une évaluation directe d'un utilisateur. Dans la littérature, cette version des AGs est connue sous le nom de : ***Algorithme Génétique Interactif (AGI)***. Pour [Takagi, 1998a], un AGI est l'approche qui permet d'intégrer dans un système d'optimisation l'un des processus psychologiques de l'être humain (intuition, perception, cognition, etc.). La figure 3.2 illustre quelques processus appartenant à cette catégorie (fonctions psychologiques) désignés dans [Takagi, 1998a] [Takagi, 1998b] sous le nom de ***KANSEI***. Pour [Takagi, 2001], les AGIs ont deux principales définitions. La première, la plus répandue dans la littérature (concerne le processus génétique), soulève l'aspect subjectif de ce type d'algorithme - *"the technology that EC optimizes the target systems based on subjective human evaluation as fitness values for system outputs."*. Tandis que la seconde définition qui est un peu plus générale dans le concept des AGIs met l'accent sur l'interactivité entre le système et l'utilisateur - *"the technology that EC optimizes the target systems having an interactive human-machine interface"*.

## 3.4 Apports d'un AGI dans la fouille visuelle de données

Bien que différentes approches ont été proposées pour résoudre le problème d'automatisation du paramétrage des visualisations en fouille visuelle de données (voir chapitre 2),

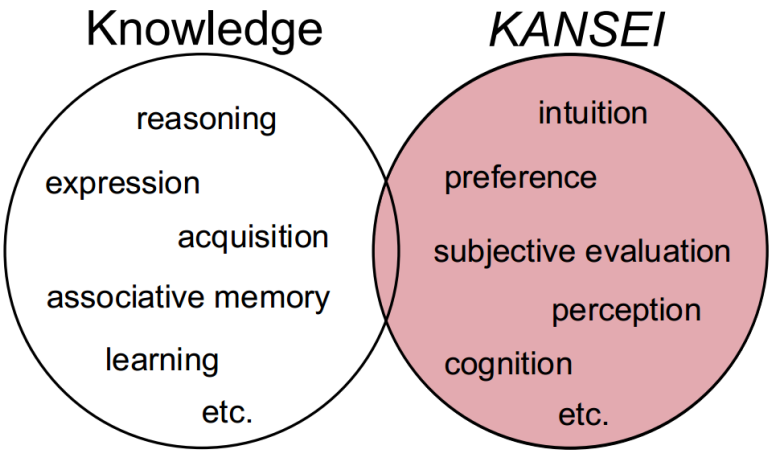



FIGURE 3.2 – La classification des capacités humaines [Takagi, 2003].

aucune d'elles ne permet vraiment de personnaliser ce processus en temps réel. Dans notre cas d'étude, cette personnalisation signifie l'adaptation des résultats aux divers profils utilisateurs. Cette personnalisation est relative soit à la description des données à visualiser, soit aux caractéristiques de leurs systèmes visuels ou à la description des visualisations utilisées. De plus, trouver un appariement entre un jeu de données et une visualisation consiste à tester un large espace de solutions possibles de mise en correspondance entre les attributs de données d'un jeu de données utilisateur d'un côté, et les attributs visuels d'une visualisation donnée. Généralement, l'accomplissement de cette tâche nécessite de répéter le même processus jusqu'à l'aboutissement à un paramétrage approprié aux objectifs utilisateurs. Ainsi, au vu de toutes les caractéristiques de notre problème, nous pouvons conclure qu'il s'agit d'un problème d'optimisation combinatoire.

Par ailleurs, notre problème dépendant étroitement de la subjectivité et du niveau individuel de perception visuelle des utilisateurs, il est nécessaire de s'appuyer sur une approche qui tient compte de la dimension subjective de ce processus (processus de paramétrage) et de la complexité combinatoire. C'est pour cette raison que nous nous sommes intéressés dans nos travaux de recherche aux AGI [Dawkins, 1986] [Smith, 1991] [Sims, 1992].

## 3.5 Principe de base d'un AGI

La figure 3.3 illustre la différence du processus génétique qui existe entre un AG et un AGI. Cette différence met en avant les avantages de ce dernier dans la résolution de la dimension subjective du problème que nous traitons. Comme déjà indiqué ci-dessus, dans l'AGI, l'utilisateur joue un rôle important et intervient directement dans le processus génétique pour évaluer les individus proposés dans chaque population. Bien que l'opération de génération d'une population reste toujours une opération automatique, le processus de sélection des individus (solutions) devient une tâche manuelle ou semi-automatique. L'intervention de l'utilisateur dans cette étape, assure ainsi d'accommoder le résultat final du déroulement de l'algorithme aux préférences de chaque utilisateur. Ceci passe par la

définition d'une fonction de fitness personnalisée et en temps réel. Des cas d'études de cette personnalisation sont proposés dans le chapitre 8 et 9. En effet, dans chaque cas d'étude, l'AGI est utilisé pour accomplir un objectif différent (découverte de variables discriminantes pour distinguer des classes de données, clustering, réorganisation de dimensions, etc.), mais sans aucune modification dans la structure initiale de l'algorithme (étapes de déroulement de l'AGI), ni dans la représentation des individus.

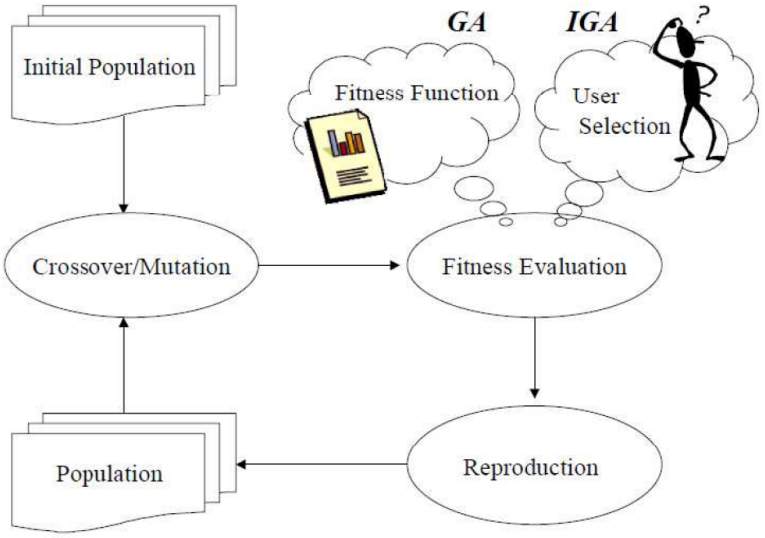



FIGURE 3.3 – Schéma illustratif de la différence dans le principe de base d'un AG et un AGI selon [Kim et Cho, 2000].

## 3.6 Représentation des individus et taille d'une population d'un AGI

La première étape dans la résolution d'un problème donné en utilisant un AGI consiste à définir la représentation générale de ses solutions potentielles. Chaque solution (individu) est définie par un nombre fixe de paramètres (gènes) caractérisant le problème à résoudre. Une fois la forme générale d'un individu établie, un codage (binaire, réel, etc.) lui est appliqué. En s'appuyant sur la structure générale d'un individu et le domaine des valeurs de ses gènes, une population regroupant un grand ensemble de solutions est générée de manière aléatoire. Contrairement à un AG où la taille d'une population $P$ n'a pas de limite théorique et dépend du problème traité [Mahfoud, 1994] [Smith, 1997] [Weicker, 2000], cette taille est bornée dans un AGI [Sims, 1991]. Le faible nombre d'individus d'une population utilisée par un AGI est justifié principalement par le mode d'évaluation dans ce type d'algorithmes. En effet, à chaque itération de l'AGI ses individus (parfois un sous ensemble d'individus) sont présentés directement à l'utilisateur pour être évalués. Toute augmentation du nombre d'individus d'une population signifie donc un affaiblissement et une diminution dans l'efficacité de l'AGI à résoudre le problème à traiter. Une description plus détaillée des impacts

de la taille d'une population d'un AGI est présentée dans la section 3.10.

## 3.7 Opérateurs génétiques

### 3.7.1 Évaluation/Sélection

Dans un processus génétique, l'étape d'évaluation permet de déterminer si un individu $I$ d'une population $P$ forme une solution optimale ou non à l'instant $t$. Le but de cette opération est donc de décider si une solution proposée dans $P(t)$ doit être retenue ou écartée pour la génération des individus de $P(t+1)$. Les individus sélectionnés dans chaque itération d'un AG, représentent généralement des optima locaux dans un grand espace de solutions. Plusieurs techniques de sélection existent dans la littérature [Goldberg et Deb, 1990] et le choix de la meilleure méthode dépend du problème traité et des résultats attendus.

Dans un AG, les qualités des individus d'une population sont mesurées en s'appuyant sur une fonction de fitness. La définition de cette dernière dépend du problème à traiter et sert de critère pour déterminer la meilleure solution pour le résoudre. Ainsi, dans un AG la sélection des individus à retenir à chaque itération de l'algorithme est une tâche automatique basée sur la fonction de fitness. Par contre, dans un AGI cette sélection se fait de manière manuelle car la qualité des individus est évaluée directement par l'utilisateur. Dans un AGI, généralement aucun critère de sélection ne peut être prédéfini pour préciser ce qu'est une solution optimale.

### 3.7.2 Croisement

L'étape de croisement dans un AG permet de reproduire des nouveaux individus à partir d'un ensemble d'individus déjà existants. En effet, après l'opération de sélection, les individus retenus dans $P(t)$ sont combinés deux à deux pour échanger leurs gènes de manière à créer de nouvelles solutions permettant de constituer la nouvelle population $P(t+1)$. On peut donc dire que le croisement est une association des caractéristiques génétiques de deux individus dits parents pour générer deux autres individus dit fils. La figure 3.4 illustre un exemple d'utilisation de l'opérateur de croisement à 1 point.

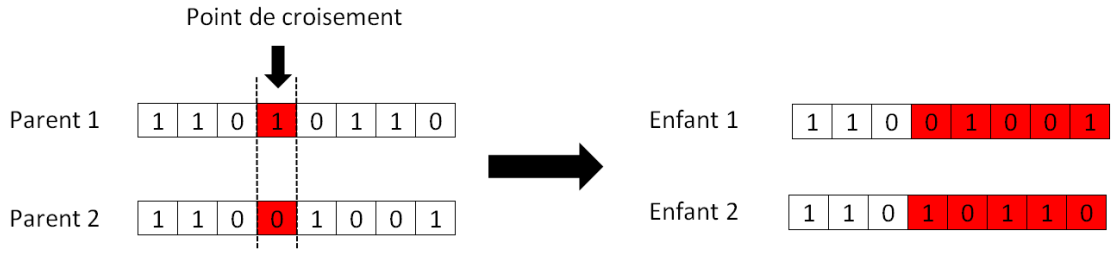



FIGURE 3.4 – Opérateur de croisement à 1 point.

Une multitude d'opérateurs de croisement est proposée dans la littérature [Sywerda, 1989] [De Jong et Spears, 1992] [Beasley *et al.*, 1993] [Haj-Rachid *et al.*, 2010]. Cependant, certaines contraintes sont imposées dans leurs choix pour quelques catégories de problèmes

[Starkweather *et al.*, 1991].

### 3.7.3 Mutation

La fonction principale de l'opérateur de mutation est de modifier la valeur d'un gène dans un chromosome, de manière aléatoire. Cette modification dépend du codage des individus d'une population. Par exemple, dans le cas d'utilisation d'un codage binaire pour représenter les individus de $P$, i.e. les valeurs des gènes $\in [0, 1]$, l'opérateur de mutation remplace la valeur d'un gène par son complémentaire (voir figure 3.5). Dans le cas d'un codage réel des individus, l'opérateur de mutation permet de changer la valeur des gènes à muter dans un intervalle prédéfini $[min, max]$.

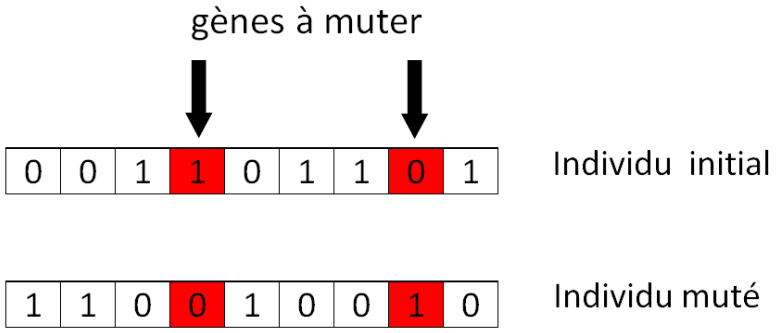



FIGURE 3.5 – Opérateur de mutation.

La mutation en tant qu'étape dans le processus génétique est considérée également comme une étape stochastique. L'aspect probabilistique de cette opération est due à sa fréquence d'exécution dans le processus génétique qui est soumise à une faible probabilité à chaque itération de l'algorithme. Cette étape permet donc de maintenir une certaine diversité dans les solutions proposées par l'algorithme dans chaque population [Schaffer *et al.*, 1989]. En effet, l'opération de croisement permettant uniquement d'échanger les gènes entre les individus sélectionnés, le risque que les individus engendrés à l'issue de cette étape permutent les mêmes caractéristiques est très élevé. Cette probabilité augmente encore plus quand aucun critère limitant le nombre d'itérations de l'algorithme est prédéfini. Ainsi, pour éviter cette éventuelle stagnation dans l'évolution des individus générés dans les différentes populations, la mutation est utilisée pour assurer cette variation nécessaire.

Plusieurs opérateurs de mutation existent dans la littérature et le choix d'un opérateur approprié dépend généralement de la nature du problème à traiter. Notons que la mutation la plus utilisée est la mutation uniforme [Michalewicz, 1996].

## 3.8 Domaines d'applications des AGIs

Les AGIs ont prouvé leur efficacité dans différents domaines où l'une des fonctionnalités psychologiques de l'être humain (voir figure 3.2) est sollicitée. Dans [Takagi, 2001], les

différents domaines d'applications des AGIs sont regroupés en deux courants de recherche distincts. Le premier courant concerne les applications dans les différentes disciplines de l'intelligence artificielle comme la conception graphique (infographie), la composition musicale et la création de modèles artistiques. Tandis que le second courant englobe les différentes disciplines de l'ingénierie et les systèmes ludo-éducatifs. Appartenant à la première catégorie (intelligence artificielle), le travail de [Caldwell et Johnston, 1991] s'appuie sur un AGI afin d'optimiser le processus d'identification des visages des suspects. En effet, l'utilisation de l'AGI a pour intérêt de combiner des images des différentes parties faciales à travers un processus interactif jusqu'à l'aboutissement à une solution optimale (un prototype du visage de la personne suspectée). Utilisé dans [Tokui et Iba, 2000] [Horowitz, 1994] pour résoudre un problème de composition musicale, l'AGI a prouvé sa puissance pour améliorer des rythmes musicaux en se basant sur les préférences d'un utilisateur pour désigner les meilleures solutions possibles. Dans le but d'aider des concepteurs de vêtements à concevoir de nouveaux styles, [Kim et Cho, 2000] propose d'utiliser un AGI. Ce dernier peut même être exploité directement par des clients pour personnaliser de manière interactive leurs préférences (couleurs, formes, tailles, etc.). En permettant d'incorporer les utilisateurs dans le processus de recherche d'images à partir de grandes base de données, l'utilisation des AGIs dans [Cho et Lee, 2002] [Lai et Chen, 2011] a contribué à améliorer significativement les résultats obtenus par rapport aux méthodes classiques (recherche par mot clés). Cette approche a contribué également dans le même cadre à améliorer les résultats fournis par des moteurs de recherche de base de données d'images [Are, 2011]. [Jakša et Takagi, 2003] proposent quelques exemples expérimentaux d'utilisation d'un AGI pour améliorer les résultats d'un processus de traitement d'image. Dans le domaine de la visualisation, [Cancino *et al.*, 2012] utilise l'AGI pour guider les utilisateurs pour explorer de manière visuelle et interactive des ensembles de données complexes et multidimensionnelles. Plusieurs utilisations de l'AGI dans le domaine de traitement de la parole ont prouvé l'efficacité de ces techniques a résoudre les différentes distorsions constatées dans le déroulement de ses processus [Watanabe et Takagi, 1995]. Parmi les travaux s'appuyant sur un AGI dans le domaine de la réalité virtuelle, on trouve celui de [Iwasaki *et al.*, 2000] qui l'utilise pour doter un aquarium virtuel de différentes espèces de poissons. Son principal but est de personnaliser la forme des poissons et les voir ainsi nager dans un aquarium virtuel. Plusieurs travaux dans le domaine de la robotique utilise les AGIs comme [Kamohara *et al.*, 1997] [Lund *et al.*, 1998] qui l'exploitent dans le cadre de l'amélioration des contrôleurs des jeux robots (*"LEGO robots"*). En ingénierie, les résultats de l'application des AGIs dans le domaine des systèmes mécano électronique [Kamalian *et al.*, 2004] a démontré aussi l'efficacité de cette catégorie d'algorithmes génétiques à optimiser leurs modèles de conceptions.

Comme nous nous intéressons à la fouille (visuelle) de données dans notre recherche, une étude plus détaillée de l'application des AGIs dans ces deux domaines est donnée dans la section suivante. Cependant, un résumé plus exhaustif des différentes applications des AGIs (citées ci-dessus) est proposé dans [Takagi, 2001].

## 3.9 Utilisation des AGIs en fouille visuelle de données

Bien que l'utilisation des AGIs comme technique d'aide à la décision peut représenter un atout majeur en fouille de données, à travers l'incorporation de l'utilisateur dans le processus d'extraction de connaissances, peu de travaux de ce domaine utilisent cette approche. Dans [Venturini *et al.*, 1997], l'utilisation de l'AGI a pour intérêt majeur d'améliorer le processus de coopération entre un expert et un outil de fouille de données en prenant en compte ses préférences et intuitions. Cela permet donc de faciliter le processus d'extraction de connaissances à partir de bases de données numériques. En s'appuyant sur une représentation graphique 2D, l'AGI que propose [Venturini *et al.*, 1997] permet de sélectionner et de valider visuellement toute relation qui peut exister entre des variables numériques et symboliques (générées par un outil de fouille de données) et qui peuvent intéresser un expert du domaine. Pour [Venturini *et al.*, 1997], les résultats obtenus à travers l'application de l'AGI en fouille de données démontre l'intérêt d'utiliser une évaluation visuelle et interactive qui s'appuie sur une optimisation génétique pour l'extraction de connaissances. Cependant, leur approche est limitée dans son utilisation sur une seule visualisation statique (matrices de scatter-plot) et pour un seul type de données (numérique).

Dans [Terano et Inada, 2003], l'AGI a pour intérêt principal d'aider des experts médicaux à analyser des données cliniques afin d'identifier de manière interactive les facteurs pertinents dans le diagnostic des maladies. Dû au bruit qu'ils peuvent contenir, l'exploitation des données cliniques nécessite généralement une phase de nettoyage avant toute utilisation dans un processus décisionnel. Même si des méthodes de pré-traitement permettent d'assurer cette tâche, une intervention directe d'un praticien hospitalier est généralement nécessaire à cause de la dimension subjective des différentes pathologies. Dans un outil d'aide à la décision dans le domaine médical, ce dernier intervient surtout pour cerner les caractéristiques qui peuvent être discriminantes ou pas dans les données cliniques. Étant fondée sur des arbres de décision, l'approche proposée par [Terano et Inada, 2003] utilise l'AGI pour sélectionner ce qui convient le mieux au problème traité. En effet, cette approche met en valeur les règles de décisions importantes pouvant guider au mieux le processus décisionnel. Cependant, même si les résultats fournis par le système proposé par [Terano et Inada, 2003] prouve l'efficacité d'utilisation d'un AGI pour accomplir des tâches de fouilles de données, la solution proposée s'appuie uniquement sur un affichage classique des résultats (texte). D'ailleurs, c'est principalement cette technique de représentation qui justifie la durée nécessaire (20 heures) pour aboutir à un résultat lors de l'application de leur approche sur un jeu de données volumineux pour accomplir une tâche de classification. L'absence de représentations graphiques pour interpréter plus facilement et plus rapidement les données cliniques durant le processus génétique est donc une contrainte qui limite la solution proposée par [Terano et Inada, 2003] et impacte ainsi le processus décisionnel de leur système.

Selon [Boudjeloud-Assala et Poulet, 2008], l'utilisation d'une approche semi-interactive pour réduire la complexité d'un processus de sélection de dimensions [Dash *et al.*, 1997] dans un jeu de données décrit par un nombre important de variables, est une solution très prometteuse. Pour cela, ils combinent une approche automatique avec un AGI pour proposer un compromis entre la qualité des solutions obtenues au problème traité, et le temps de calcul nécessaire pour aboutir à des résultats significatifs. En effet, la méthode de

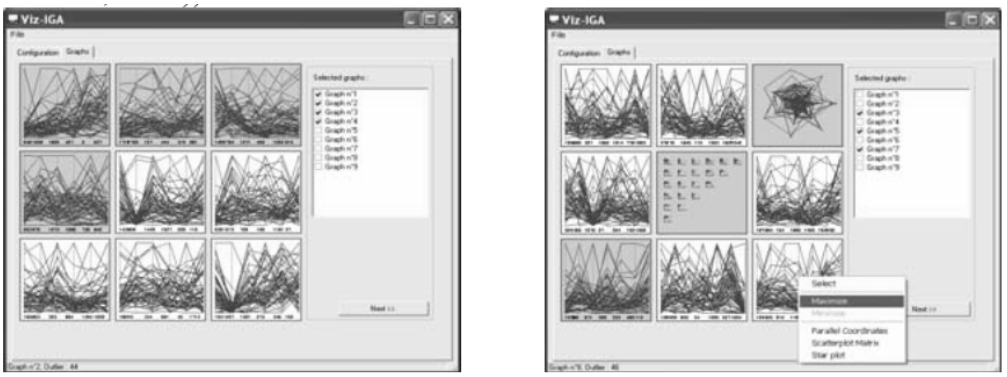



FIGURE 3.6 – Interfaces du système Viz-IGA [Boudjeloud-Assala et Poulet, 2008].

[Boudjeloud-Assala et Poulet, 2008] consiste à utiliser dans un premier temps une fonction d'évaluation, fondée sur des critères permettant d'éliminer des solutions redondantes ou bruitées. Cela permet de sélectionner un ensemble initial de sous-espaces représentant des solutions potentielles au problème traité. Ensuite, en s'appuyant sur ce dernier ensemble, les données sont visualisées et présentées à l'utilisateur. L'utilisateur peut ainsi choisir de manière interactive à l'aide de l'AGI les sous-ensembles les plus pertinents. Les caractéristiques des sous-ensembles sélectionnés par l'utilisateur à chaque itération de l'AGI servent donc à générer des solutions adaptées à ses préférences. La figure 3.6 illustre deux interfaces parmi celles proposées par le système Viz-IGA [Boudjeloud-Assala et Poulet, 2008]. À travers leurs travaux, [Boudjeloud-Assala et Poulet, 2008] [Boudjeloud et Poulet, 2005] ont confirmé la conclusion de [Venturini *et al.*, 1997] concernant l'avantage d'utilisation d'un AGI pour résoudre des tâches de fouille de données, en l'occurrence la détection d'individus aberrants et le clustering. Cependant, l'utilisation de leur approche est limitée à deux tâches de fouille de données et appliquée uniquement sur deux visualisations 2D statiques. Le système "EvoGraphDice", proposé par [Cancino *et al.*, 2012], est un autre système de fouille visuelle de données qui s'appuie sur un AGI (voir figure 3.7). Plus précisément, "EvoGraphDice" permet de trouver un bon point de vue dans des "scatterplots" multidimensionnels par combinaison linéaire [Cancino *et al.*, 2012] et non-linéaire [Cancino *et al.*, 2013] d'axes. À travers les résultats obtenus lors des expérimentations menées avec "EvoGraphDice", [Cancino *et al.*, 2012] ont démontré l'avantage de l'utilisation d'une approche interactive lors d'un processus d'analyse visuelle. Cependant, le système "EvoGraphDice" s'appuie uniquement sur une seule visualisation.

## 3.10 Contraintes d'utilisations d'un AGI et solutions proposées

Même si l'utilisation des AGIs constitue généralement une solution très avantageuse pour résoudre des problèmes d'optimisation subjective, plusieurs travaux [Banzhaf, 1997] [Takagi et Ohsaki, 1999] [Tokui et Iba, 2000] [Takagi, 2001] [Wang, 2007] soulèvent quelques contraintes d'application de ces approches. La principale contrainte concerne la fatigue constatée chez les utilisateurs durant l'étape d'évaluation des individus d'une population.

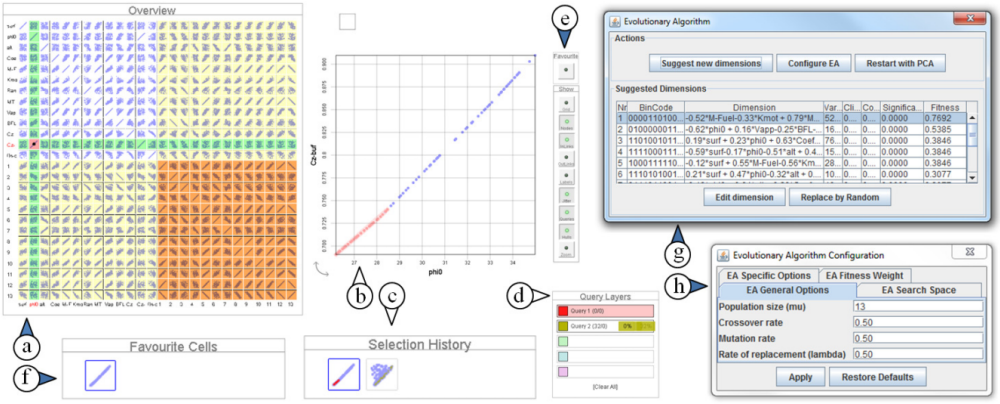



FIGURE 3.7 – Interfaces du système EvoGraphDice [Cancino *et al.*, 2012].

En effet, cette difficulté peut s'expliquer par le mode d'évaluation des individus utilisé dans le processus génétique d'un AGI. Comme l'utilisateur doit mesurer directement la qualité de l'ensemble des individus générés à chaque itération de l'algorithme, ce processus peut très vite devenir une tâche fastidieuse. Cela a pour inconvénient d'augmenter le risque d'abandon du déroulement de l'AGI chez les utilisateurs au bout de quelques générations de populations.

La réduction du nombre d'individus dans une population ou la limitation du nombre d'itérations de l'algorithme peuvent être des solutions possibles au problème de la fatigue humaine dans un AGI. Cependant, certains comme [Tokui et Iba, 2000] les considèrent comme étant d'autres contraintes de l'algorithme génétique interactif. En effet, en l'absence d'un critère permettant d'accélérer la convergence vers des solutions optimales, toute limitation opérée sur une population peut influencer négativement le résultat final et ne pas garantir l'efficacité attendue par les utilisateurs.

La troisième contrainte d'utilisation d'un AGI se pose uniquement pour quelques domaines d'applications. Par exemple, si dans la population utilisée par un AGI, un individu représente une séquence vidéo comme [Unemi, 2000] ou une composition musicale comme [Tokui et Iba, 2000], l'évaluation de ce dernier (individu) se fait généralement par une comparaison entre deux solutions générées à deux instants qui se suivent. Dans le cas de ce type d'applications, la durée nécessaire à un utilisateur pour comparer deux solutions est très difficile à mesurer. Ainsi, même si le nombre d'individus à évaluer dans une population ainsi que le nombre d'itérations sont limités, la durée nécessaire globale du déroulement de l'AGI est difficile à estimer et peut être très longue.

Pour résoudre les contraintes mentionnées ci-dessus, différentes stratégies ont été proposées dans la littérature. L'amélioration des interfaces des systèmes utilisant un AGI est l'une des solutions proposées par plusieurs travaux [Takagi et Ohya, 1996] [Ohsaki *et al.*, 1998] [Takagi et Ohsaki, 1999] [Hayashida et Takagi, 2002] pour réduire l'impact de la fatigue constatée chez les utilisateurs. Pour accélérer la convergence vers des solutions optimales sous les deux contraintes citées ci-dessus (limite du nombre d'individus et de génération d'une population), [Takagi et Kishi, 1999] [Takagi, 2000] [Hayashida et Takagi, 2000] [Tokui et Iba, 2000] [Boschetti et Takagi, 2001] [Hsu et Huang, 2005] suggèrent de faire in-

tervenir l'utilisateur dans les différentes étapes du processus génétique en plus de la phase d'évaluation automatique. Pour [Tokui et Iba, 2000], la combinaison d'un AGI avec une approche automatique représente une solution prometteuse pour améliorer l'efficacité des AGIs durant l'étape d'évaluation du processus génétique. Cette technique a pour but de faire une pré-évaluation de tous les individus d'une population avant de présenter un sous ensemble d'individus aux utilisateurs pour être évalués. Une autre solution suggérée par [Baker, 1993] consiste à assurer plus de diversité entre deux populations qui se succèdent en augmentant le taux de mutation utilisé dans le processus génétique pour générer les individus à chaque itération de l'AGI. [Kim et Cho, 2000] résout le problème de la taille d'une population dans les AGIs par un regroupement d'individus sous forme de clusters. L'évaluation est ensuite fondée sur le score fourni par une fonction de fitness sur celui du meilleur représentant de chaque cluster. Il propose aussi d'utiliser une interface utilisateur *multi-field* [Unemi, 1998] qui a pour but d'assurer une diversité dans les solutions proposées à chaque itération de l'AGI même si la taille de la population est limitée. Dans la même catégorie d'approches s'appuyant sur une évaluation automatique ou au moins semi-automatique, on trouve aussi les solutions proposées par [Gong *et al.*, 2009] [Llorà *et al.*, 2005].

## 3.11 Conclusion

En s'appuyant sur la description des algorithmes génétiques classiques, nous avons présenté dans ce chapitre un aperçu général des algorithmes génétiques interactifs. Pour cela, nous avons mis en avant leurs atouts dans la résolution de certains types de problèmes d'optimisation, en l'occurrence ceux qui ont un rapport avec l'une des fonctionnalité psychologique de l'être humain. Contrairement aux AGs, ces techniques permettent d'incorporer les utilisateurs dans le processus génétique en leur offrant la possibilité d'évaluer les individus d'une population. Nous avons exposé aussi les différents domaines d'applications en mettant l'accent sur le domaine de la fouille visuelle de données qui concerne notre domaine d'étude. Malgré la faible utilisation de ces techniques en fouille visuelle de données, l'utilisation des AGI à travers un processus interactif et visuel dans la résolution de quelques tâches d'extraction de connaissances a donné des résultats très significatifs. Enfin, nous avons recensé quelques contraintes d'utilisation des AGIs dont nous devons tenir compte lors de l'élaboration de notre approche.

Deuxième partie

# VizAssist : Assistant Utilisateur pour la fouille visuelle de données

# Chapitre 4

# Architecture de VizAssist

**Résumé** : Nous proposons dans ce chapitre, un aperçu général de notre assistant utilisateur **VizAssist** pour le choix et le paramétrage des méthodes de fouille visuelle de données. Un système que nous définissons sur la base d'un modèle simple et générique. En effet, pour résoudre les limitations constatées dans les systèmes existants, VizAssist propose deux étapes. La première étape consiste à proposer à l'utilisateur différents appariements entre la base de données à visualiser et les visualisations qu'il gère en s'appuyant sur les objectifs qu'il annonce et les caractéristiques de ses données. Ces appariements sont générés par une heuristique utilisant une base de connaissances sur les visualisations et la perception visuelle. Ensuite, afin d'affiner les différents paramétrages suggérés par le système, VizAssist utilise dans la seconde étape un algorithme génétique interactif qui permet aux utilisateurs d'évaluer et d'ajuster visuellement ces paramétrages.

## 4.1 Introduction : Aperçu général du système VizAssist

L'utilisation des systèmes de visualisation d'informations est devenue de plus en plus fréquente dans tous les domaines. Cela est dû essentiellement aux capacités des visualisations à représenter de manière assez claire de grands ensemble de données, et aussi aux capacités perceptives des utilisateurs à les interpréter. Cela signifie que l'utilisation d'une visualisation a pour principal intérêt de réduire l'effort intellectuel à fournir pour comprendre un nombre important de données. Cependant, réussir à proposer un tel processus dans un système de visualisation nécessite de le doter d'une architecture simple et générique permettant :

- la description facile et efficace des différentes caractéristiques des données et objectifs utilisateurs,
- la collecte des différentes connaissances pouvant définir une représentation graphique aussi bien sur le plan conceptuel que sur le plan sémantique,
- la mise à jour de ces connaissances.

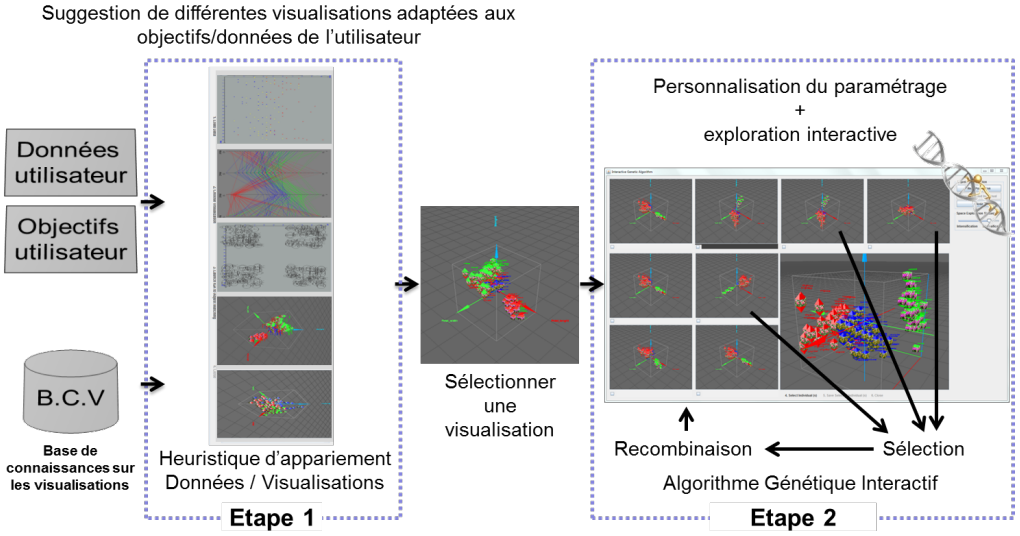



FIGURE 4.1 – Aperçu général de VizAssist.

Afin de faciliter l'intégration de nouvelles connaissances sur les visualisations et les préférences utilisateurs, et la prise en considération d'éventuelles extensions du nombre de visualisations gérées, nous proposons un nouvel assistant utilisateur : **VizAssist**. L'architecture de base de notre système est définie sur un modèle simple et générique dont les principaux éléments sont :

1. un modèle des données et des objectifs utilisateur (voir chapitre 5).
2. une base de connaissances sur les visualisations (voir chapitre 5).
3. un module d'appariement et de suggestion de visualisations (voir chapitre 6).
4. un module de paramétrage interactif avec un algorithme génétique interactif (voir chapitre 7).

La figure 4.1 illustre le processus général de visualisation utilisé par notre système VizAssist dans lequel on peut remarquer que notre outil est fondé sur un processus défini par deux étapes.

### 4.1.1 Première étape

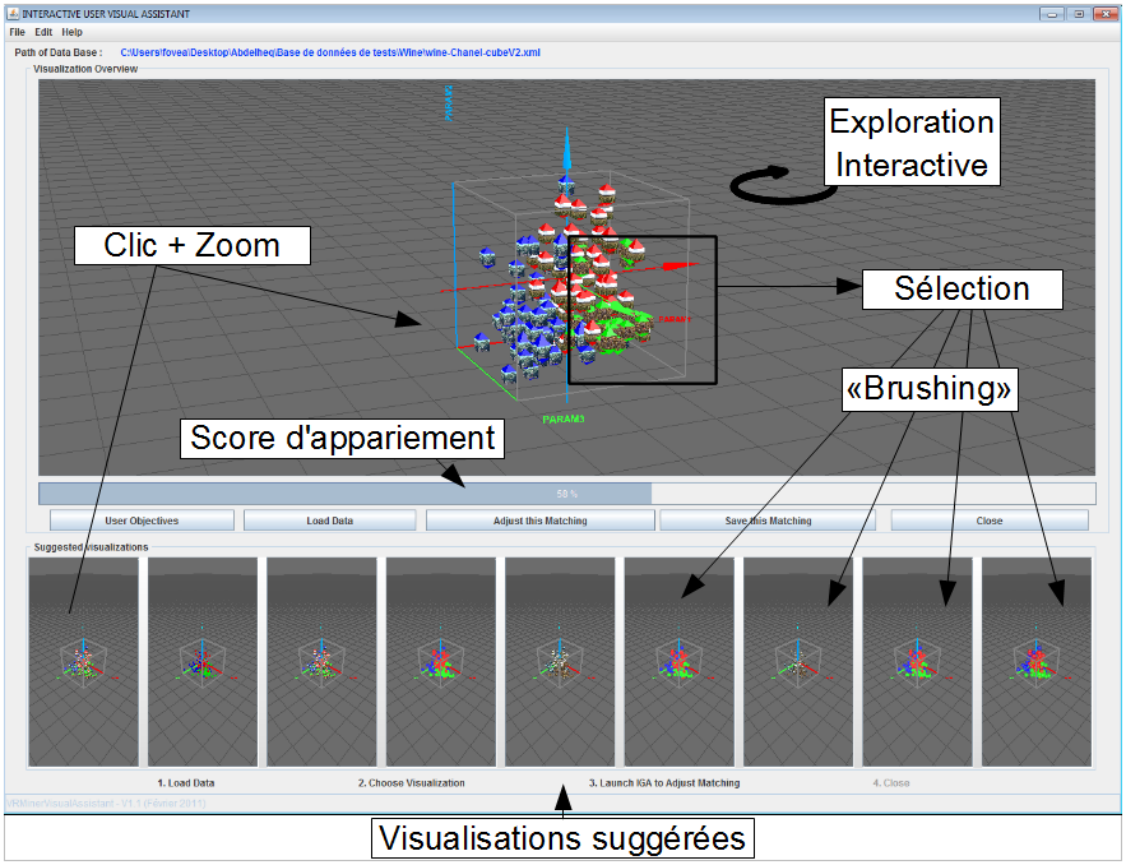



FIGURE 4.2 – Aperçu général du processus de la première étape de VizAssist. L'interface utilisée propose plusieurs interactions (clic, zoom, sélection, exploration, "brushing", etc.) et permet de suggérer différentes méthodes de visualisation appliquées sur le jeu de données utilisateur.

L'objectif de la première étape du processus de visualisation de VizAssist est de suggérer différentes visualisations adaptées à la description des données et des objectifs des utilisateurs. En effet, conscient de l'importance d'un tel mécanisme dans un système de fouille visuelle de données, nous avons doté notre système d'une interface intuitive (voir figure 4.2) qui s'appuie sur une heuristique d'appariement simple. Cette dernière a pour but principal d'automatiser le processus de mise en correspondance entre les visualisations gérées par le système et les jeux de données utilisateurs à visualiser en s'appuyant sur leurs descriptions. Notons, que durant cette première étape, VizAssist s'appuie sur une base de connaissances sur les visualisations et un modèle de données et objectifs utilisateurs (voir

chapitre 5).

### 4.1.2 Seconde étape

Étant conscient de la difficulté du processus de paramétrage manuel (appariement entre les attributs de données et les attributs visuels) utilisé dans les systèmes de visualisation existants (voir chapitre 2), nous proposons dans notre système une nouvelle approche pour accomplir cette tâche. L'avantage principal de notre méthode est qu'elle permet de personnaliser, à travers une interface simple et intuitive (voir figure 4.3), le paramétrage des visualisations de manière interactive et visuelle. Lors de cette seconde étape, VizAssist s'appuie sur un algorithme génétique interactif dont l'intérêt est de proposer un processus itératif permettant d'explorer un espace de solutions (appariements) large pouvant se dérouler plus facilement et plus rapidement. Notons que si l'utilisateur n'est pas satisfait des appariements proposés dans la seconde étape (ex. il considère que le nombre d'attributs de données affichées est limité), il a la possibilité de revenir à la première étape pour choisir une autre visualisation.

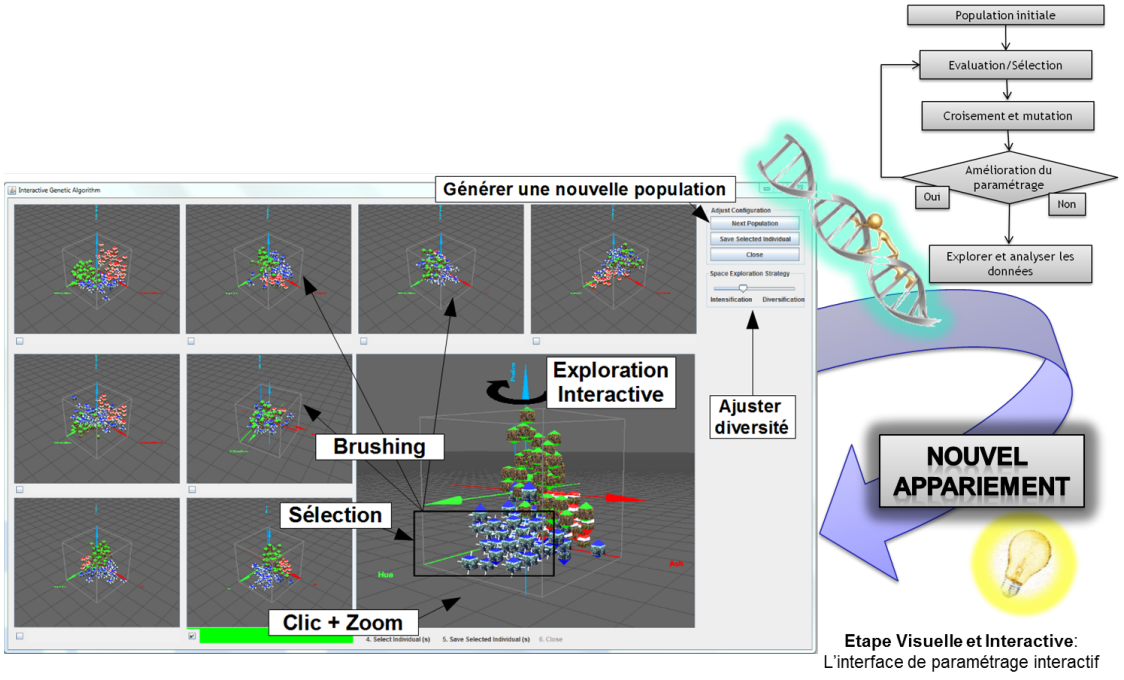



FIGURE 4.3 – Aperçu général du processus de la seconde étape de VizAssist.

Nous détaillons dans la suite de cette deuxième partie, les différents modules constituants l'architecture de notre outil VizAssist en illustrant quand cela est nécessaire un exemple concret d'utilisation.

# Chapitre 5

# Modèles de données et base de connaissances sur les visualisations

**Résumé** : Nous détaillons dans ce chapitre le modèle des données et des objectifs utilisateur ainsi que la base de connaissances sur les visualisations gérées par VizAssist. En effet, dans notre outil, le modèle des données a pour intérêt principal de définir formellement la structure dans laquelle les données utilisateur doivent être décrites pour dérouler le processus de visualisation. Le modèle des objectifs utilisateur quant à lui, a pour but de permettre aux utilisateurs de fixer leurs objectifs d'analyse et d'exploration. La base de connaissances sur les visualisations sert en effet à représenter la description des visualisations proposées par VizAssist.

## 5.1 Introduction

Nous décrivons dans ce chapitre les deux premiers modèles de VizAssist : ***(1) le modèle des données et des objectifs utilisateur, (2) la base de connaissances sur les visualisations***. Nous commençons par présenter son modèle des données et ses apports par rapport aux systèmes existants (voir chapitre 2). Nous décrivons ensuite le modèle des objectifs utilisateurs et la manière dont ces derniers sont pris en compte par VizAssist pour pré-sélectionner les visualisations. Nous présentons aussi l'architecture générique de la base de connaissances sur les visualisations que nous avons mises en place pour enregistrer la description conceptuelle des visualisations gérées par VizAssist. Pour cela, nous définissons toutes les visualisation déjà implémentées ou rajoutées à la base. Nous terminons par une conclusion.

## 5.2 Modèle de données

Utiliser un système de visualisation pour générer des représentations graphiques nécessite d'exécuter un processus comportant plusieurs étapes. La première phase consiste à transformer les données à visualiser de leur format brut initial (forme originale) vers un format manipulable et exploitable par le système. La transformation des données est donc une opération qui consiste à structurer un jeu de données en s'appuyant sur une représentation prédéfinie.

Pour [Thomas et Cook, 2005], la mise en place d'une représentation de données appropriée est une étape primordiale pour générer des représentations graphiques significatives. L'importance de cette phase est liée essentiellement à la description des caractéristiques des données originales à visualiser. En effet, ces dernières peuvent contenir des informations devant être préservées afin d'aboutir à des visualisations efficaces et expressives.

Partant du constat que la représentation d'un jeu de données utilisateur revient à définir formellement ses différentes caractéristiques, nous avons défini un modèle de données dont le but principal est de faciliter la phase d'appariement du processus de visualisation. Nous nous sommes appuyés dans la définition de ce processus sur la conceptualisation des visualisations de notre base de connaissances (voir section 5.5).

Dans notre modèle de données, nous notons $D = \{d_1, ..., d_n\}$ la base des $n$ données à visualiser. Chaque donnée $d_i$ est définie par $k$ attributs de données $A_1$, .., $A_k$. Chaque attribut de données $A_i$ est caractérisé par un type $t_i$ et une importance $u_i$ (voir tableau 5.1). Notre système gère différents types de données (numérique/quantitatif, symbolique/ordinal ou nominal, temporel, image, son, texte, lien Web, etc.). La valeur de l'importance $u_i$ est définie dans l'intervalle $[0, 100]$. Elle représente l'intérêt que porte l'utilisateur à l'attribut $A_i$, et peut être déterminée manuellement par l'utilisateur en fonction de ses connaissances a priori ou automatiquement via des méthodes de sélection de variables [Guyon, 2006]. Si aucune connaissance n'est disponible, alors les $u_i$ prennent toutes la même valeur ($u_i = 50$). Le tableau 5.2 illustre le format de données géré par VizAssist où les lignes du tableau représentent les données $d_i$ de $D$ et les colonnes représentent les attributs de données $A_i$. Nous notons $p_{ji}$ la valeur que prend une données $d_j$ pour l'attribut de données $A_i$, et qui correspond dans le tableau 5.2 à la cellule d'intersection entre une ligne et une colonne.

| Attributs de données | | |
|---|---|---|
| Nom | Type | Importance |
| Age | numérique | 100 |
| Poids | numérique | 80 |
| Yeux | symbolique | 20 |
| ... | ... | ... |
| ... | ... | ... |

TABLE 5.1 – Représentation des caractéristiques des attributs d'un jeu de données utilisateur $D$.

| | $A_1$ | $A_2$ | . | . | $A_k$ |
|---|---|---|---|---|---|
| Type | $t_1$ | $t_2$ | . | . | $t_i$ |
| Importance | $u_1$ | $u_2$ | . | . | $u_k$ |
| $d_1$ | $p_{11}$ | $p_{12}$ | . | . | $p_{1k}$ |
| $d_2$ | $p_{21}$ | $p_{22}$ | . | . | $p_{2k}$ |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| $d_n$ | $p_{n1}$ | $p_{n2}$ | . | . | $p_{nk}$ |

TABLE 5.2 – Le format des données (jeu de données utilisateur) à fournir en entrée de l'assistant utilisateur pour être visualiées.

Nous illustrons à travers le tableau 5.3 un exemple de représentation d'une base de données utilisateur avec notre modèle de données. La base de données utilisée est la base des IRIS [Fisher, 1936]. Cette dernière est décrite par 5 attributs de données dont 4 attributs sont de type *"numérique"* et 1 attribut de type *"symbolique"* représentant les classes des données. Par défaut, l'importance des attributs est fixée à 50. Dans la section suivante, nous illustrons les apports de cette représentation.

| Nom | Sepal length | Sepal width | Petal length | Petal width | Classe |
|---|---|---|---|---|---|
| Type | numérique | numérique | numérique | numérique | symbolique |
| Importance | 50 | 50 | 50 | 50 | 50 |
| $d_1$ | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| $d_2$ | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| $d_{150}$ | 5.9 | 3.0 | 5.1 | 1.8 | virginica |

TABLE 5.3 – Représentation de la base de données IRIS [Fisher, 1936].

## 5.3 Apports de notre modèle de données

En comparant notre modèle de données à ceux des assistants utilisateurs cités dans le chapitre 2, nous pouvons constater que le fait d'utiliser plusieurs types de données pour

renseigner les attributs de données représente un des atouts de notre système. En effet, pour préciser les caractéristiques de leurs données, les utilisateurs des précédents systèmes ont l'habitude de fixer seulement : 1) leur nature qui peut être soit quantitative ou qualitative (ordinale ou nominale), 2) leur type qui est soit continu ou discret. Ces informations restent générales et peuvent ne pas permettre le choix approprié des visualisations à proposer aux utilisateurs. À titre d'exemple, on peut citer le cas où un utilisateur désire représenter une base de données comportant des images. Si ce dernier s'appuie uniquement sur la description des données telle qu'elle a été proposée dans les modèle de données des précédents assistants (nature et type de données), il lui est impossible d'introduire cette information. Par contre, dans notre modèle de données, il suffit à l'utilisateur de choisir le type de données *"image"* lors de la définition de l'attribut de données en question pour que le système tienne compte de cette information dans le choix des visualisations à proposer.

L'autre atout de notre modèle de données réside dans l'intérêt à travers l'utilisation de *"l'importance"* dans la définition des attributs de données. Même si cette caractéristique est prise en compte dans les systèmes ViA [Healey *et al.*, 1999] [Healey *et al.*, 2008], Vis-WIZZ [Lange *et al.*, 1995] et VISTA [Senay et Ignatius, 1992, Senay et Ignatius, 1994], elle est utilisée uniquement pour désigner l'ordre dans lequel les attributs de données à visualiser doivent être considérés par le système, dans le processus d'appariement avec les attributs visuels. Cela signifie que la désignation d'une importance dans ces systèmes est restreinte uniquement aux attributs de données choisis pour être représentés. Par ailleurs, si dans un jeu de données utilisateur, il y a un nombre important d'attributs de données et si l'utilisateur a besoin de modifier à chaque fois leurs ordres ou sélectionner de nouveaux attributs de données à visualiser, il est obligé d'effectuer ce processus manuellement. Cette opération peut donc devenir une étape fastidieuse à cause de la durée qu'elle peut nécessiter. Dans notre système, l'attribution d'une importance aux attributs de données a deux avantages. Le premier avantage est de faciliter (à l'utilisateur), la tâche de spécification des attributs de données pertinents, et donc ceux qui devraient être visualisés en priorité. Le second avantage est que le vecteur initial des importances des attributs de données (fixées par l'utilisateur en entrée) sert de support dans le processus d'optimisation de paramétrage (appariement entre attributs de données et attributs visuels) proposé par VizAssist. En effet, ce processus permet d'éviter aux utilisateurs de modifier manuellement les importances des attributs de données déjà visualisés pour changer leur ordre d'appariement. Il permet aussi de sélectionner de nouveaux attributs de données à visualiser. Une explication plus détaillée de ce processus est donnée dans le chapitre 7.

## 5.4 Modèles des objectifs utilisateurs

Dans le but de permettre aux utilisateurs de VizAssist de spécifier leurs objectifs d'exploration et d'analyse sur leurs jeux de données, nous avons défini un modèle des objectifs utilisateurs (assez proche de celui utilisé dans Vis-WIZZ). Nous nous sommes appuyés dans l'élaboration de ce modèle sur une liste prédéfinie d'objectifs recensés dans la littérature (découvrir des classes, avoir une vue d'ensemble, détecter des individus aberrants, détecter des corrélations, sélectionner des données, sélectionner des attributs, etc.). Afin d'identifier plus facilement les préférences utilisateurs, nous avons associé à chaque objectif, noté $O_i$,

une importance renseignée par une priorité $p_j$. Cette dernière est définie sur une échelle de priorité ("pas important", "peu important", "assez important" et "très important") dont chaque occurrence est décrite par une valeur numérique dans l'intervalle $[0, 3]$ : pas important = 0, peu important = 1, assez important = 2 et très important = 3. Le tableau 5.4 illustre un exemple de spécification des objectifs utilisateurs fixés pour une session d'utilisation de VizAssist. Les utilisateurs fixent les priorités de leurs objectifs d'exploration et d'analyse via une interface (voir figure 5.1).

| Objectifs utilisateur | Importance |
|---|---|
| Découvrir des classes | 1 |
| Avoir une vue d'ensemble | 3 |
| Détecter des individus aberrants | 2 |
| Détecter des corrélations | 1 |
| . | . |
| . | . |

TABLE 5.4 – Représentation des objectifs utilisateurs dans notre système. L'importance désigne l'ordre de priorité des objectifs fixés par un utilisateur pour visualiser son jeu de données.

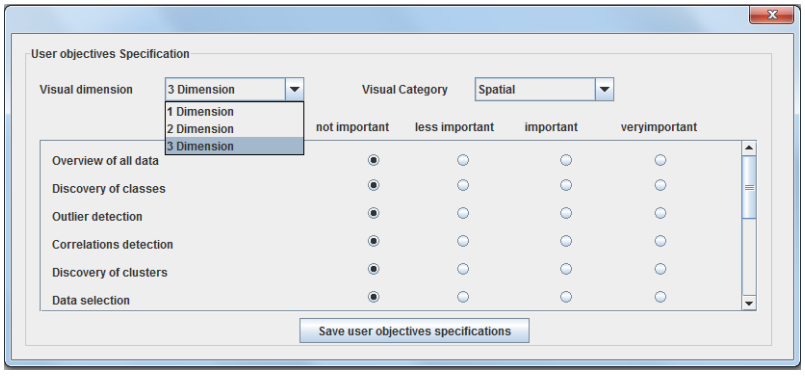



FIGURE 5.1 – Interface de spécification des objectifs utilisateurs proposée par VizAssist.

## 5.5 Base de connaissances sur les visualisations

Représenter dans un système de visualisation des méthodes de visualisation nécessite la définition d'une représentation claire et concise de ces dernières. L'élaboration d'une telle représentation est habituellement fondée sur le modèle de données utilisé par ce type d'outils. D'une manière générale, cela a pour principal avantage de simplifier le processus de visualisation, et plus particulièrement l'étape d'appariement entre les données utilisateur et les visualisations. Représenter une visualisation de manière claire et concise passe par la prise en compte de deux critères indispensables dans sa définition : ***expressivité*** et ***efficacité*** [Healey *et al.*, 2008]. Nous rappelons qu'une visualisation est dite ***expressive***

si elle permet de représenter un jeu de données utilisateur dans sa totalité ainsi que les différentes relations qui existent entre ses données. La notion d'expressivité concerne donc l'aspect structurel des ensembles de données à visualiser. Quant à l'efficacité, c'est une notion qui dépend plutôt de l'aspect sémantique d'une représentation graphique. En effet, une visualisation est dite ***efficace*** si l'encodage visuel qu'elle utilise permet de faciliter le processus d'interprétation des informations qu'elle représente, et ainsi de simplifier les différentes tâches d'exploration et d'analyse qu'elle permet d'accomplir.

Dans la littérature, la majorité des travaux du domaine de la visualisation d'informations s'appuient sur la sémiologie graphique de [Bertin, 1983] pour conceptualiser les différentes techniques de visualisation. De plus, en s'appuyant sur les types de données qu'elles permettent de représenter (relationnel, géo-spatiale, spatial, etc.), quelques travaux ont listé de manière formelle les méthodes d'encodage visuels pour chaque type de représentation graphique. On peut citer à titre d'exemple, les travaux de [Salisbury, 2001] qui regroupe les différentes techniques de visualisation en 4 catégories (voir figure 5.2).

| | *Encoding Methods* | |
|---|---|---|
| Base Type | objects | attributes |
| Graphs | polygons, bars, lines, points, icons | position, color hue, color intensity, pattern, size (length, width, height, diameter), shape |
| Maps | polygons, surfaces, lines, points | color hue, color intensity, pattern, point size, point shape, line width |
| Charts | polygons, bars, lines, points, icons | position, color hue, color intensity, pattern, size (length, width, height, diameter), shape |
| Tables | labels, lines | color, line width |

FIGURE 5.2 – Description des méthodes d'encodage des visualisations [Salisbury, 2001].

Dans le but de tenir compte des différentes caractéristiques pouvant permettre la définition d'une représentation graphique, nous avons mis en place une base de connaissances sur les visualisations pour modéliser les visualisations dans VizAssist. Cette dernière a pour avantages de :

1. faciliter la modélisation des visualisations en s'appuyant sur notre modèle de données.
2. tenir compte des différents niveaux de perception visuelle des utilisateurs ainsi que les résultats des différentes études psychophysiques sur les caractéristiques du système visuel humain.

### 5.5.1 Modélisation des visualisations

En s'inspirant principalement des travaux de [Bertin, 1983] et de [Card *et al.*, 1999], nous avons développé une base de connaissances permettant de modéliser les différentes visualisations gérées par VizAssist (voir figure 5.4). Dans notre conceptualisation, nous définissons une visualisation $V_i$ par ses éléments graphiques (points, lignes, formes 2D ou 3D, etc.). Chaque élément graphique de $V_i$ est caractérisé par un ou plusieurs attributs

visuels. Dans notre modèle de visualisations, nous notons l'ensemble des attributs visuels de $V_i$ par $A_{i1}$, .., $A_{im}$. Nous associons à chaque attribut visuel $A_{ij}$ un type visuel $vt_{ij}$ (position, taille, couleur, etc.), un type d'attribut de données $dt_{ij}$ dont la valeur est utilisée pour renseigner l'attribut visuel, et un degré d'importance $v_{ij}$. Les valeurs $v_{ij}$ sont déterminées selon la capacité que peut avoir un attribut visuel à représenter tel ou tel type d'attribut de données ainsi que le fait que $A_{ij}$ soit obligatoirement représenté dans une visualisation donnée ou ne l'est pas (ex. dans un nuage de points, l'utilisation de la couleur comme attribut visuel est optionnelle à l'inverse des deux axes X et Y). Pour tenir compte de cette contrainte dans notre base de connaissances, chaque attribut visuel $A_{ij}$ est donc renseigné par une variable booléenne appelée ***"Obligatoire"***. Si cette dernière prend la valeur *"Oui"* alors l'attribut visuel désigné est obligatoire, sinon (la valeur est *"Non"*) l'attribut visuel est optionnel. Afin de déterminer les valeurs $A_{ij}$ dans notre base de connaissances, nous avons défini une matrice d'importance "type d'attribut visuel × type d'attribut de données" (voir tableau 5.5). Les valeurs renseignées dans cette dernière sont déterminées en s'appuyant sur l'étude de [Mackinlay, 1986] (voir figure 1.9) et de [Salisbury, 2001] (voir figure 5.3).

| types visuels | types de données | | |
|---|---|---|---|
| | Quantitative | Ordinal | Nominal |
| Position | 50 | 50 | 50 |
| Longueur | 45 | 15 | 18 |
| Angle | 40 | 10 | 14 |
| Orientation (slope) | 35 | 5 | 10 |
| Surface/Volume | 30 | 1 | 6 |
| Saturation | 25 | 45 | 30 |
| Couleur | 20 | 40 | 46 |
| Texte | 15 | 30 | 22 |
| Texture | 0 | 35 | 42 |
| Connexion | 0 | 25 | 38 |
| Inclusion | 0 | 20 | 34 |
| Forme | 0 | 0 | 26 |

TABLE 5.5 – Matrice des importances "type d'attribut visuel × type d'attribut de données" utilisée par VizAssist.

Notons que dans certains travaux comme [Healey *et al.*, 1999], fixer l'importance des attributs visuels dépend aussi du nombre de valeurs discrètes qu'elles peuvent représenter (voir figure 1.10) pour un attribut de données. Dans [Healey *et al.*, 2008], cette caractéristique est utilisée par les moteurs d'évaluation (voir figure 2.15) comme une unité de mesure de la capacité visuelle humaine à percevoir une plage de valeurs pour un attribut visuel donné. Cela signifie donc que cette dernière peut influencer directement le choix des attributs visuels. Dans VizAssist, cette caractéristique est gérée directement dans l'algorithme d'appariement, sous forme de contrainte à vérifier avant de mettre en correspondance les attributs de données, d'un jeu de données utilisateur, avec les attributs visuels d'une visualisation $V_i$ (voir chapitre 6).

De plus, pour chacune des visualisations de la base de connaissances, nous décrivons

| **Quantitative Data** | **Ordinal Data** | **Nominal Data** | |
|---|---|---|---|
| Position | Position | Position | *most accurate* |
| Length | Color Intensity | Color Hue | ↑ |
| Angle | Color Hue | Pattern/Texture | |
| Slope | Pattern/Texture | Connection | |
| Area/Volume | Text | Containment | |
| Color Intensity | Connection | Shape | |
| Text | Containment | Text | |
| | Length | Length | |
| | Angle | Angle | |
| | Slope | Slope | ↓ |
| | Area/Volume | | *least accurate* |

FIGURE 5.3 – Classification des attributs visuels selon leur capacités à renseigner plus efficacement les attributs de données proposée par [Salisbury, 2001].

quels sont les objectifs à atteindre, sa dimension visuelle (1D, 2D, 3D) et sa catégorie visuelle (temporelle, relationnelle, etc.). Pour [Nakamura *et al.*, 1995], l'objectif d'une visualisation est généralement désigné par une action et un ou plusieurs types de données qu'elle peut satisfaire. Selon [Nakamura *et al.*, 1995], les différentes actions qu'une visualisation peut proposer sont : identification, localisation, distinction, catégorisation, clusterisation, classification, comparaison, association et corrélation. [Fujishiro *et al.*, 2000] étend cette liste en y rajoutant : trouver une structure de, émerger une structure de, révéler un changement de, révéler un "cluster" de, révéler une distribution de, révéler des modèles de, révéler une structure et révéler des relations. Dans notre base de connaissances, nous nous sommes appuyés sur cette liste d'actions pour spécifier les objectifs $O_j$ à atteindre par une visualisation $V_i$. Nous attribuons à chaque objectif un poids $o_{ij}$ qui peut varier d'une visualisation à une autre. Ces poids sont calculés en fonction des caractéristiques et travaux connus sur chacune des visualisations représentées comme [Keller et Keller, 1993].

Pour résumer une visualisation $V_i$ est modélisée dans VizAssist par :

1. un ensemble d'attributs visuels dont le tout constitue un élément graphique. Chaque attribut visuel est caractérisé par un type visuel, un type d'attribut de données servant à renseigner l'attribut visuel et une importance,
2. une dimension visuelle,
3. une catégorie visuelle,
4. un ensemble d'objectifs d'analyse qu'elle permet d'atteindre.

### 5.5.2 Administration de la base de connaissances

Dans le but de représenter les différentes visualisations gérées par notre assistant, nous avons développé une interface web. Cette dernière a pour principal intérêt d'administrer notre base de connaissances en facilitant :

1. l'ajout de la description conceptuelle des nouvelles visualisations à intégrer dans notre assistant.
2. la personnalisation du paramétrage des visualisations selon les nouvelles études sur les caractéristiques du système visuel humain :
   (a) mise-à-jour des importances des attributs visuels.
   (b) mise-à-jour des importances des objectifs utilisateurs pour chaque visualisation.
   (c) extension de la limite visuelle (nombre de valeurs discrètes possibles) qu'un attribut visuel peut représenter dans une visualisation.

### 5.5.3 Schéma conceptuel de la base de connaissances

La figure 5.4 illustre le schéma relationnel de notre base de connaissances. En effet, la table *"méthode"* désigne les caractéristiques d'une technique de visualisation (nom de la méthode, description de son principe et un lien vers sa source si elle existe). La table *"bibliographie"* permet de renseigner pour chaque visualisation le nom de son (ses) auteur (s), l'année de son apparition, sa description et éventuellement le lien hypertexte vers cette bibliographie. Les autres tables sont décrites dans la section 5.5.1.

A noter que chaque table de notre base de connaissances est gérée par un formulaire dans notre assistant web. L'utilisation de l'assistant web a donc pour apport principal, de séparer le processus de paramétrage des visualisations du processus de génération des visualisations dans VizAssist. Ainsi, l'intégration de nouvelles visualisations dans notre base de connaissances devient une tâche facile et possible via le web.

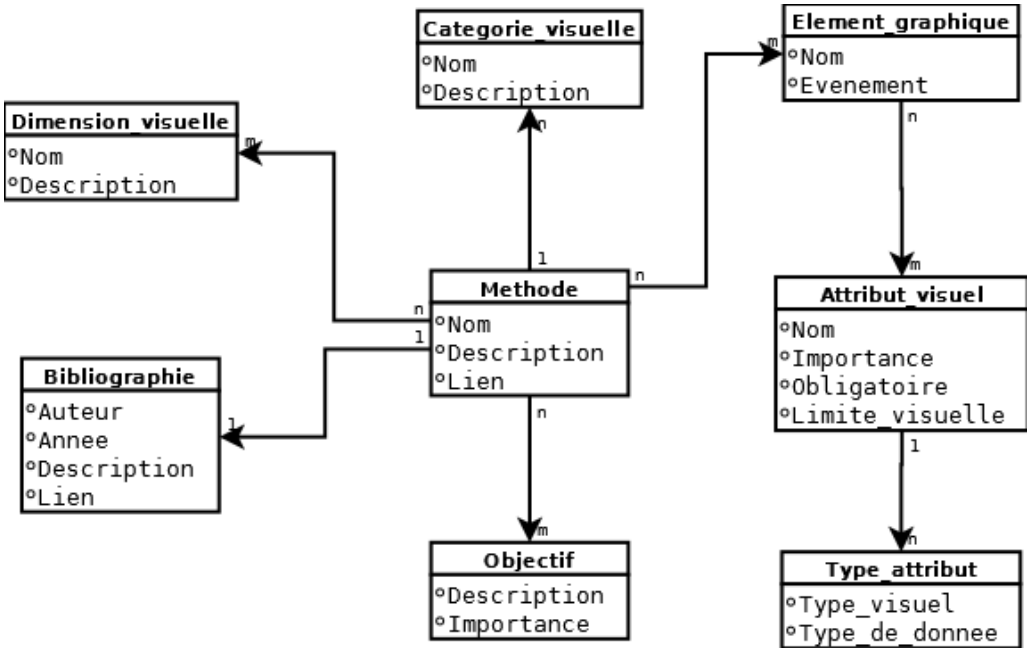



FIGURE 5.4 – Schéma de la base de connaissances sur les visualisations utilisée par notre système.

### 5.5.4 Méthodes de visualisation représentées dans notre base de connaissances

Dans le but de faciliter le processus de développement et d'intégration des visualisations dans VizAssist, nous avons défini un encodage visuel spécifique pour renseigner la description conceptuelle des visualisations dans notre base de connaissances. Pour élaborer ce dernier, nous nous sommes appuyés sur le modèle de visualisations décrit dans la section 5.5.1. Le tableau 5.6 illustre les méthodes d'encodage visuel spécifiées dans VizAssist. Ces méthodes servent à décrire toute nouvelle visualisation à ajouter dans la base de connaissances, et à développer son code source pour générer son rendu visuel. Dans la version actuelle de notre système, nous avons intégré 2 catégories de visualisations (2 dimensions et 3 dimensions). Toutes les techniques de visualisations gérées par VizAssist sont multi-dimensionnelles et sont de nature (catégorie visuelle) spatiale. Nous détaillons dans ce qui suit chacune de ces visualisations en présentant leurs descriptions conceptuelles renseignées dans la base de connaissances ainsi qu'un exemple de leurs rendus visuels générés par VizAssist. A noter que pour le format 3D des visualisations proposées, nous nous appuyons sur des éléments ***"graphiques composés"***. Un élément graphique est dit ***"composé"*** s'il est défini par une collection d'éléments graphiques simples (point, ligne, surface, volume) et le résultat forme une seule unité perceptive [Senay et Ignatius, 1992].

| Visualisations | Éléments graphiques | Attributs visuels |
|---|---|---|
| Nuage de points 3D | cube chanel v2, cube chanel v1<br>cube, sphère, facette | axeX, axeY, axeZ, Image, Texte Haut<br>Texte Bas, TaillePyrHaut, TaillePyrBas<br>CouleurPyrHaut, CouleurPyrBas |
| Nuage de points 2D | point | axeX, axeY, couleur |
| Coordonnées parallèles | polyligne | Axis1, Axis2, Axis3, Axis4<br>Axis5, Axis6, Axis7, Axis8<br>Axis9, Axis10, Axis11, Axis12<br>Axis13, Axis14, Axis15, Axis16<br>Axis17, Axis18, Axis19, Axis20, couleur |
| Visages de Chernoff | visage | forme visage, taille pupilles, forme sourcils,<br>forme yeux, taille yeux, écartement yeux,<br>forme nez, courbure bouche, axeX, axeY<br>taille bouche, forme bouche, couleur |

TABLE 5.6 – Système d'encodage des méthodes de visualisation utilisées par notre assistant pour les représenter dans la base de connaissances. Pyr = pyramidion.

#### 5.5.4.1 Nuage de points 3D

Dans notre assistant, la définition des visualisations 3D proposées est fondée sur la description de la représentation graphique *"nuage de points"* dans un espace 3 dimensions. La seule différence qui existe entre ces visualisations réside dans la description de la forme des points 3D dont chacune représente l'élément graphique de base de la visualisation. Dans notre base de connaissances, nous nous sommes appuyés sur l'élément graphique *"point 3D"* pour définir 5 formes géométrique différentes (***"Cube Chanel V2", "Cube Chanel V1", "Cube", "Sphère", "Facette"***). La principale différence entre ces formes concerne le nombre d'attributs visuels qu'un *"point 3D"* permet de représenter simultanément. En s'appuyant sur notre modèle de visualisations (une visualisation est décrite par un élément

graphique), VizAssist permet donc de générer 5 visualisations différentes à partir de ces éléments graphiques.

1. **Nuage 3D Cube Chanel V2**
   Dans cette visualisation, nous pouvons représenter jusqu'à 11 attributs de données simultanément. En effet, l'élément graphique de base de cette visualisation est un symbole graphique composé, nommé *"Cube Chanel V2"* (voir figure 5.5). Ce dernier est formé d'une composition d'un cube avec deux pyramidions. Le premier pyramidion est positionné au dessus du cube et le second au dessous. Dans notre base de connaissances, un *"Cube Chanel V2"* est défini par 11 attributs visuels (voir tableau 5.7). 3 attributs décrivent la position de l'objet dans l'espace 3D (*axe X, axe Y et axe Z*) et permettent de représenter les attribut de données de type *"numérique"*. 1 attribut *couleur* (plaqué sur les 4 faces du cube) qui permet de désigner les attributs de données de type *"symbolique"* (ex. classes des données) et 1 attribut *image*. Chacun des deux pyramidions est défini par 2 attributs visuels : *couleur* (*couleur pyramidion haut* et *couleur pyramidion bas*) et *taille* (*taille pyramidion haut* et *taille pyramidion bas*). La taille des pyramidions peut servir par exemple à désigner une amplitude de la valeur d'un attribut de données, tandis que la couleur à distinguer des classes de données. Les 2 autres attributs caractérisant l'élément graphique *"Cube Chanel V2"* sont *Texte Haut* et *Texte Bas*. Ces derniers peuvent être utilisés pour renseigner une information supplémentaire de type *"texte"* qu'on peut placer au dessus et au dessous de chaque élément graphique (ex. numéro ou nom de la classe).
   A noter que les deux attributs visuels (*couleur* et *image*) de l'élément graphique *"Cube Chanel V2"* ne peuvent pas être représentés ensemble dans la même visualisation. Cela est dû au fait que si on met l'image sur les faces d'un cube, la couleur permettant de distinguer des classes de données serait cachée. Pour tenir compte de cette contrainte et surtout des préférences utilisateurs dans cette visualisation (ex. afficher des images sur les cubes pour visualiser des photos ou utiliser les faces des cubes pour distinguer des classes de données), la visualisation ***Nuage 3D Cube Chanel V2*** est décrite dans notre assistant en 2 versions différentes (avec et sans attribut visuel *image*). La figure 5.5 illustre un exemple de cette visualisation (version utilisant l'attribut visuel *image*) proposée par notre assistant et appliquée sur une base de données artificielle.

| Attributs visuels | | | | | |
|---|---|---|---|---|---|
| Nom | Type visuel | Type de données | Obligatoire | limite visuelle | Importance |
| axe X | position | numérique | oui | 100 | 50 |
| axe Y | position | numérique | oui | 100 | 50 |
| axe Z | position | numérique | oui | 100 | 50 |
| taille pyramidion haut | longueur | numérique | non | 50 | 45 |
| taille pyramidion bas | longueur | numérique | non | 50 | 45 |
| couleur pyramidion haut | couleur | symbolique | non | 12 | 40 |
| couleur pyramidion bas | couleur | symbolique | non | 12 | 40 |
| couleur | couleur | symbolique | non | 12 | 40 |
| image | texture | image | non | $\infty$ | 35 |
| Texte Haut | texte | texte | non | $\infty$ | 22 |
| Texte Bas | texte | texte | non | $\infty$ | 22 |

TABLE 5.7 – Description de la visualisation ***"Nuage 3D Cube Chanel V2"***.

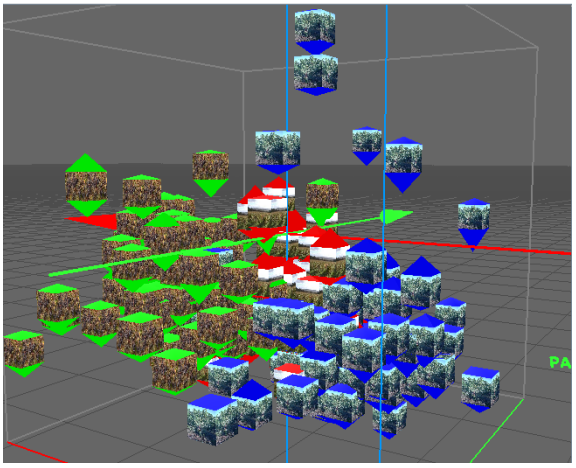


FIGURE 5.5 – La visualisation *"Nuage 3D Cube Chanel V2"* proposée par VizAssist appliquée sur une base de données artificielle.

2. **Nuage 3D Cube Chanel V1**
   Basée sur un élément graphique composé, la visualisation ***Nuage 3D Cube Chanel V1*** est décrite dans notre base de connaissances par 9 attributs visuels (voir tableau 5.8). La seule différence entre cette visualisation et la première ***Nuage 3D Cube Chanel V2*** est qu'elle est caractérisée uniquement par un seul pyramidion placé au dessus du cube. Cette deuxième visualisation est également définie dans notre base de connaissances en 2 versions différentes (avec et sans attribut visuel *image*).

| Attributs visuels | | | | | |
|---|---|---|---|---|---|
| Nom | Type visuel | Type de données | Obligatoire | limite visuelle | Importance |
| axe X | position | numérique | oui | 100 | 50 |
| axe Y | position | numérique | oui | 100 | 50 |
| axe Z | position | numérique | oui | 100 | 50 |
| taille pyramidion haut | longueur | numérique | non | 50 | 45 |
| couleur pyramidion haut | couleur | symbolique | non | 12 | 40 |
| couleur | couleur | symbolique | non | 12 | 40 |
| image | texture | image | non | $\infty$ | 35 |
| Texte Haut | texte | texte | non | $\infty$ | 22 |
| Texte Bas | texte | texte | non | $\infty$ | 22 |

TABLE 5.8 – Description de la visualisation ***"Nuage 3D Cube Chanel V1"***.

3. **Nuage 3D Cube**
   Dans sa définition, la visualisation ***Nuage 3D Cube*** est fondée sur un élément graphique simple *"Cube"*. Dans notre base de connaissances, cette dernière est décrite par 7 attributs visuels (*axe X, axe Y, axe Z, couleur, image, Texte Haut et Texte Bas*). A noter que cette visualisation est également prise en compte dans notre assistant en 2 versions différentes (avec et sans attribut visuel *image*). La figure 5.6 illustre un exemple d'utilisation de cette visualisation, proposé par notre assistant, pour représenter une base de données artificielle comportant un attribut de données de type *"image"*. Le tableau 5.9 illustre la description détaillée des attributs visuels de la visualisation ***Nuage 3D Cube***.

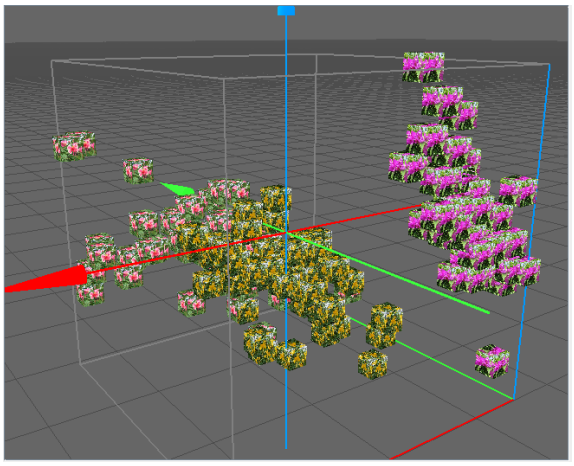


FIGURE 5.6 – La visualisation *"nuage 3D Cube"* proposée par VizAssist appliquée sur une base de données artificielle.

| Attributs visuels | | | | | |
|---|---|---|---|---|---|
| Nom | Type visuel | Type de données | Obligatoire | limite visuelle | Importance |
| axe X | position | numérique | oui | 100 | 50 |
| axe Y | position | numérique | oui | 100 | 50 |
| axe Z | position | numérique | oui | 100 | 50 |
| couleur | couleur | symbolique | non | 12 | 40 |
| image | texture | image | non | ∞ | 35 |
| Texte Haut | texte | texte | non | ∞ | 22 |
| Texte Bas | texte | texte | non | ∞ | 22 |

TABLE 5.9 – Description des deux visualisations ***"Nuage 3D Cube"/"Nuage 3D Facette"***.

4. **Nuage 3D Facette**
   La description conceptuelle de cette visualisation est identique à celle de la visualisation ***Nuage 3D Cube*** du point de vue du nombre d'attributs visuels. Nous illustrons dans la figure 5.7 le rendu visuel proposé par notre assistant pour représenter une base de données artificielle. Nous définissons aussi cette visualisation en 2 versions différentes dans notre base de connaissances (avec et sans attribut visuel *image*).
5. **Nuage 3D Sphère**
   À l'inverse des autres visualisations 3D, la visualisation ***Nuage 3D Sphère*** ne permet pas de représenter les attributs de données de type *"image"* (voir tableau 5.10). Cela est dû essentiellement à la forme ronde de l'élément graphique qui la définit. Dans notre base de connaissances, cette visualisation est décrite par 6 attributs visuels (*axe X, axe Y, axe Z, couleur, Texte Haut et Texte Bas*). La figure 5.8 illustre un aperçu de la visualisation ***Nuage 3D Sphère*** générée par notre assistant.

#### 5.5.4.2 Nuage de points 2D

La représentation graphique ***Nuage de points 2D*** est définie par 3 attributs visuels (*axe X, axe Y et couleur*) dans notre base de connaissances (voir tableau 5.11). Bien que

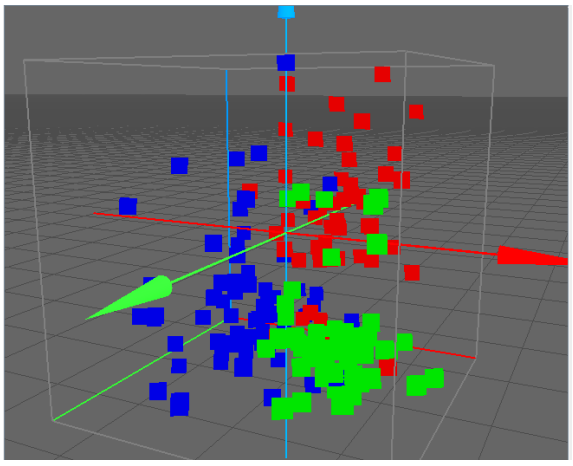


FIGURE 5.7 – La visualisation *"nuage 3D Facette"* proposée par VizAssist appliquée sur une base de données artificielle.

| Attributs visuels | | | | | |
|---|---|---|---|---|---|
| Nom | Type visuel | Type de données | Obligatoire | limite visuelle | Importance |
| axe X | position | numérique | oui | 100 | 50 |
| axe Y | position | numérique | oui | 100 | 50 |
| axe Z | position | numérique | oui | 100 | 50 |
| couleur | couleur | numérique | non | 12 | 40 |
| Texte Haut | texte | texte | non | $\infty$ | 22 |
| Texte Bas | texte | texte | non | $\infty$ | 22 |

TABLE 5.10 – Description de la visualisation ***"Nuage 3D Sphère"***.

cette visualisation ne permet de représenter que 2 attributs de données de type *"numérique"* et un attribut de données de type *"symbolique"*, elle reste l'une des représentations graphiques les plus connues et les plus utilisées. En effet, à l'issue d'une évaluation utilisateur formative que nous avons menée avec un groupe d'utilisateurs de 15 personnes, nous avons constaté que la majorité des participants ont déjà utilisé cette représentation graphique au moins une fois. Pour cette raison, nous l'avons intégrée dans notre système (voir figure 5.9).

| Attributs visuels | | | | | |
|---|---|---|---|---|---|
| Nom | Type visuel | Type de données | Obligatoire | limite visuelle | Importance |
| axe X | position | numérique | oui | 100 | 50 |
| axe Y | position | numérique | oui | 100 | 50 |
| couleur | couleur | symbolique | non | 12 | 40 |

TABLE 5.11 – Description de la visualisation ***"Nuage de points 2D"***.

#### 5.5.4.3 Coordonnées parallèles

La représentation graphique ***Coordonnées parallèles*** est considérée comme l'une des rares visualisations permettant de visualiser un nombre important d'attributs de données

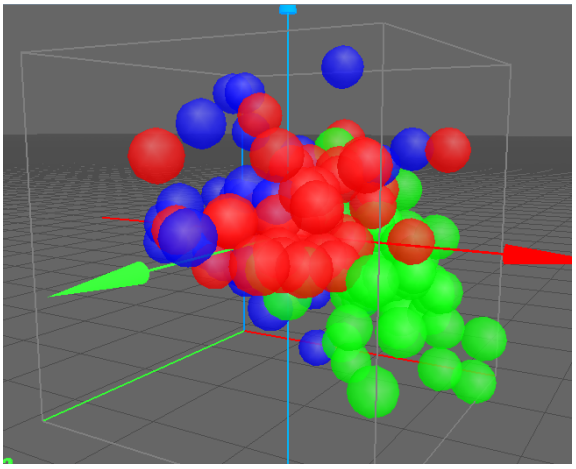


FIGURE 5.8 – La visualisation *"nuage 3D Sphère"* proposée par VizAssist appliquée sur une base de données artificielle.

sur un plan 2D. Nous avons intégré cette visualisation dans notre base de connaissances et nous la définissons par 21 attributs visuels (voir tableau 5.12) dont 20 axes verticaux $Axis_{1,2,...,20}$ et un attribut *couleur*. Notre choix de limiter le nombre d'axes verticaux possibles générés dans cette visualisation a pour but de faciliter la compréhension des données utilisateurs lors de l'utilisation de cette dernière. La figure 5.10 est le résultat proposé par VizAssist en appliquant la visualisation ***Coordonnées parallèles*** sur une base de données artificielle. Chaque axe vertical de la visualisation permet de représenter un attribut de données de type *"numérique"*. Dans cette visualisation, l'attribut visuel *"couleur"* permet de distinguer les classes de données, et donc est utilisé pour représenter des attributs de données de type *"symbolique"*. Pour intégrer cette visualisation dans VizAssist, nous nous sommes appuyés sur le code source de [Hauser *et al.*, 2002] [7] (voir section 9.2).

| Attributs visuels | | | | | |
|---|---|---|---|---|---|
| Nom | Type visuel | Type de données | Obligatoire | limite visuelle | Importance |
| Axis1 | position | numérique | oui | 100 | 50 |
| Axis2 | position | numérique | oui | 100 | 50 |
| Axis3 | position | numérique | non | 100 | 50 |
| Axis4 | position | numérique | non | 100 | 50 |
| . | position | numérique | non | 100 | 50 |
| . | position | numérique | non | 100 | 50 |
| Axis19 | position | numérique | non | 100 | 50 |
| Axis20 | position | numérique | non | 100 | 50 |
| couleur | couleur | symbolique | non | 12 | 40 |

TABLE 5.12 – Description de la visualisation ***"Nuage de points 2D"***.

7. http ://www.mediavirus.org/parvis/

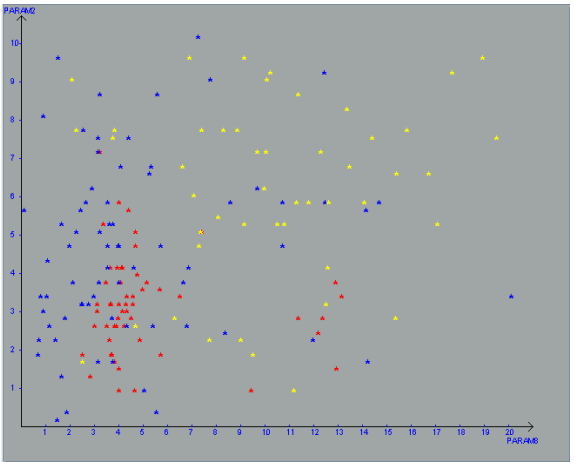


FIGURE 5.9 – La visualisation *"nuage de points 2D"* proposée par VizAssist appliquée sur une base de données artificielle.

#### 5.5.4.4 Visages de Chernoff

Dans notre base de connaissances, nous définissons la représentation graphique ***Visages de Chernoff*** (voir figure 5.11) par 13 attributs visuels. Nous nous sommes appuyés sur un plan 2D pour représenter cette visualisation, 2 des 13 attributs visuels représentent donc les deux attributs de position *axeX* et *axeY*. Pour distinguer des classes dans les données, nous décrivons aussi cette visualisation par un attribut visuel *couleur*. Le reste des attributs (les 10 autres attributs visuels) décrivent les caractéristiques faciales de l'élément graphique visage (*forme visage, forme yeux, taille pupilles, forme sourcils, forme nez, courbure bouche, écartement yeux, taille yeux, taille bouche et forme bouche*). Le tableau 5.13 illustre la description détaillée des attributs visuels de la visualisation ***Visages de Chernoff*** tels qu'ils sont renseignés dans notre base de connaissances.

| Attributs visuels | | | | | |
|---|---|---|---|---|---|
| Nom | Type visuel | Type de données | Obligatoire | limite visuelle | Importance |
| axeX | position | numérique | oui | 100 | 50 |
| axeY | position | numérique | oui | 100 | 50 |
| forme visage | longueur | numérique | oui | 50 | 45 |
| forme nez | longueur | numérique | oui | 50 | 45 |
| forme bouche | longueur | numérique | oui | 50 | 45 |
| forme yeux | longueur | numérique | oui | 50 | 45 |
| forme sourcils | longueur | numérique | oui | 50 | 45 |
| taille pupilles | longueur | numérique | oui | 50 | 45 |
| taille yeux | longueur | numérique | oui | 50 | 45 |
| écartement yeux | longueur | numérique | oui | 50 | 45 |
| courbure bouche | longueur | numérique | oui | 50 | 45 |
| couleur | couleur | symbolique | non | 12 | 40 |

TABLE 5.13 – Description de la visualisation ***"Visages de Chernoff"***.

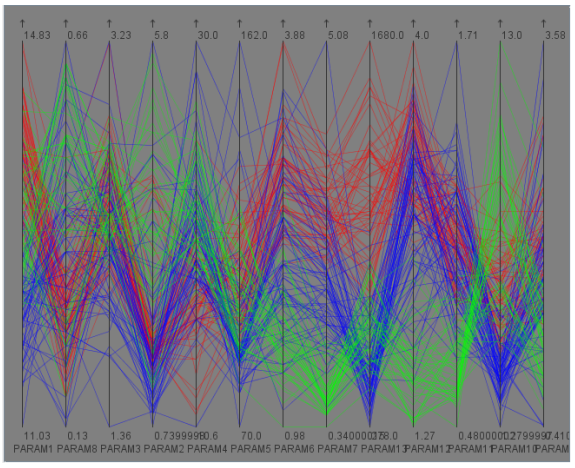



FIGURE 5.10 – La visualisation *"coordonnées parallèles"* proposée par VizAssist appliquée sur une base de données artificielle.

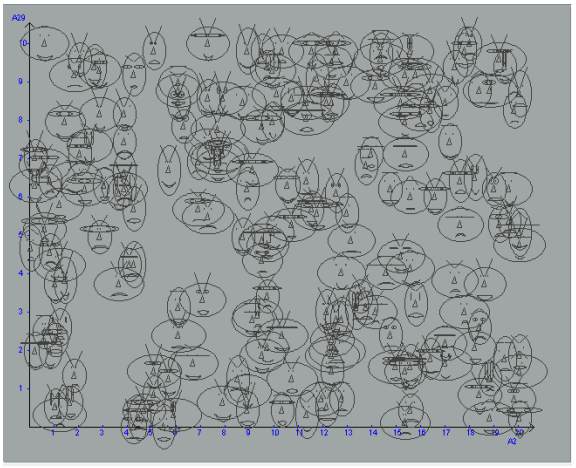


FIGURE 5.11 – La visualisation *"visages de chernoff"* proposée par VizAssist appliquée sur une base de données artificielle.

## 5.6 Conclusion

Nous avons abordé dans ce chapitre la description de deux composantes de l'architecture de VizAssist, en l'occurrence, le modèle de données et des objectifs utilisateurs, et la base de connaissances sur les visualisations. Après avoir décrit le modèle de données, nous avons mis en avant ses différents apports par rapport à ceux des systèmes existants. Nous avons aussi présenté le modèle conceptuel des visualisations sur lequel s'appuie VizAssist ainsi que les différentes méthodes de visualisation intégrées dans la base de connaissances. Nous avons mis en avant, son aspect générique dont l'intérêt est de faciliter l'intégration de nouvelles visualisations. Nous poursuivons dans le chapitre suivant l'illustration de l'aspect générique de notre système via le troisième module ***"module d'appariement et de suggestion des visualisations"***.

# Chapitre 6

# Module d'appariement et de suggestion des visualisations

**Résumé** : Nous décrivons dans ce chapitre le module d'appariement et de suggestion des visualisations constituant l'architecture de VizAssist. Ce module s'appuie sur un processus qui se déroule en deux étapes. La première étape consiste à présélectionner, à partir d'une base de connaissances sur les visualisations, celles qui sont les plus appropriées aux objectifs utilisateurs. La seconde étape permet de proposer un paramétrage à chacune des visualisations retenues. Pour illustrer le processus d'appariement proposé par VizAssist, nous présentons un exemple concret.

## 6.1 Introduction

Selon [Fayyad *et al.*, 2002], sélectionner une visualisation appropriée nécessite de prendre en considération différents facteurs importants. Parmi ces facteurs, on trouve les objectifs d'exploration et d'analyse qu'une visualisation peut permettre d'accomplir sur un jeu de données. Tenir compte de la description des données (nombre d'attributs de données, leur types de données, etc.) à visualiser représente aussi un critère décisif dans le choix des visualisations [Robertson, 1991]. Nous décrivons dans ce chapitre le troisième module ***"module d'appariement et de suggestion des visualisations"*** constituant l'architecture de VizAssist. Ce module s'appuie sur un processus qui se déroule en deux étapes. La première étape consiste à pré-sélectionner, à partir de la base de connaissances (voir chapitre 5) les visualisations les plus appropriées aux objectifs utilisateurs. La seconde étape permet de proposer un paramétrage à chacune des visualisations retenues. L'objectif de ce module est donc de permettre à l'utilisateur de choisir et de paramétrer automatiquement les visualisations qui conviennent le mieux à ses objectifs d'exploration et d'analyse. Pour simplifier le processus d'appariement à toutes les catégories d'utilisateurs, nous avons défini une interface simple et intuitive.

Nous décrivons dans la section suivante le processus d'appariement entre le jeu de données utilisateur et les visualisations à générer par le système. Dans la section 6.3, nous illustrons la technique de choix des visualisations que nous proposons et présentons l'interface que nous avons développée. Ensuite, nous expliquons le protocole de visualisation des données utilisateur utilisé par VizAssist. La section 6.4 décrit à travers un exemple illustratif les résultats obtenus à l'issue du processus d'appariement entre un jeu de données réel (IRIS [Fisher, 1936]) et les visualisations de notre base de connaissances. Nous concluons par une discussion sur les avantages et inconvénients du module d'appariement et de suggestion des visualisations de VizAssist.

## 6.2 Processus d'appariement

Dans VizAssist, la phase d'appariement entre les visualisations et le jeu de données utilisateur se déroule en deux étapes. La première étape permet de pré-sélectionner une liste de visualisations compatibles avec les objectifs utilisateurs. La seconde étape permet quant à elle, de générer les appariements entre chacune de ces visualisations sélectionnées et le jeu de données utilisateur. Les appariements obtenus sont utilisés par VizAssist pour générer les visualisations pré-sélectionnées. Aider les utilisateurs à choisir la visualisation la plus appropriée à leurs données et leurs objectifs devient ainsi une tâche facile à réaliser. Une description plus détaillée des deux étapes du processus d'appariement est proposée dans ce qui suit.

### 6.2.1 Première étape : Sélection des visualisations

La première étape du processus d'appariement consiste à sélectionner les visualisations qui sont les plus compatibles avec les objectifs utilisateurs. Dans notre système, ces derniers sont spécifiés via une liste préétablie qui est proposée sous forme de questionnaire

(voir figure 5.1). L'utilisateur est donc amené à indiquer pour chaque objectif proposé dans cette liste une importance $p_{ij}$ définie sur une échelle de priorité ("pas important", "peu important", "assez important" et "très important"). À l'issue de cette opération, un vecteur des importances des objectifs utilisateurs est déterminé, noté $P$. L'utilisateur peut aussi fixer un seuil de similarité minimum $w'$, utilisé par notre système dans cette première phase comme critère de comparaison pour la pré-sélection des visualisations à partir de la base de connaissances. Si un seuil $w'$ est déterminé, l'assistant calcule un score d'appariement, noté $w_i$, entre le vecteur $P$ et les objectifs que chacune des visualisations, de la base de connaissances, permet d'atteindre. Seules les visualisations utiles et dépassant $w'$ sont conservées pour la suite du processus. Pour une visualisation donnée, le score $w_i$ est le produit scalaire entre le vecteur des importances $P$ annoncées par l'utilisateur, et le vecteur des importances des objectifs $O_j$ de chaque visualisation renseignée dans la base de connaissances. La formule utilisée pour calculer le score d'appariement entre le vecteur des importances des objectifs d'une visualisation $V_i$ et le vecteur des importances des objectifs utilisateur fournis en entrée du système est la suivante (produit scalaire) :

$$W_i = \overrightarrow{P}.\overrightarrow{O_j} = \sum_j p_{ij} \times o_{ij}$$

A noter que si l'utilisateur ne fixe pas le seuil de similarité minimum $w'$, toutes les visualisations de notre base de connaissances seront proposées à l'issue de cette première étape. En effet, nous avons fixé à 0 la valeur de $w'$ par défaut.

### 6.2.2 Seconde étape : Paramétrage des visualisations retenues

La seconde étape du processus d'appariement consiste à générer un paramétrage (appariement) possible entre les attributs de données du jeu de données utilisateur $D$ avec les attributs visuels de chaque visualisation sélectionnée dans la première étape. Cela signifie que pour chaque attribut visuel $A_{ij}$ d'une visualisation $V_i$, on associe un attribut de données $A_i$ de $D$ . Pour cela, nous utilisons une heuristique simple (voir algorithme 1) qui s'appuie sur les deux vecteurs d'importances $U$ (vecteur des importances des attributs de données) et $V$ (vecteur des importances des attributs visuels). Cette heuristique s'appuie aussi sur la limite visuelle de chaque attribut visuel $A_{ij}$ d'une visualisation $V_i$ récupérée à partir de notre base de connaissances (voir section 5.5).

Pour générer le paramétrage d'une visualisation $V_i$ pré-sélectionnée, lors de la première étape du processus d'appariement, avec le jeu de données utilisateur $D$, l'heuristique d'appariement se déroule en 3 phases :

1. la première phase consiste à trier les attributs de données $A_i$ par type puis par ordre décroissant de leurs importances et à placer le résultat dans une liste notée $L_A$.
2. les attributs visuels $A_{ij}$ sont triés de la même manière, et le résultat est mis dans une liste notée $L_V$. Les importances $v_{ij}$ des attributs visuels $A_{ij}$ récupérées depuis la base de connaissances subissent une modification avant d'être triés. En effet, une nouvelle importance notée $v'_{ij}$ est calculée pour chaque $A_{ij}$ en fonction de sa valeur $v_{ij}$ (voir tableau 5.5) ainsi que la valeur que prend la variable ***"Obligatoire"*** (voir section 5.5.1) désignée dans la base de connaissances. Cette dernière peut varier d'une

visualisation à une autre et prend la valeur "1" si $A_{ij}$ est obligatoire et la valeur "0" dans le cas échéant. Nous utilisons la formule suivante pour générer les nouvelles importances $v'_{ij}$ utilisées pour créer $L_V$ :

$$v'_{ij} = (50 \times Obligatoire) + v_{ij}$$

De cette manière, un attribut visuel obligatoire aura une importance toujours supérieure à celle d'un attribut non obligatoire. Il sera donc apparié avant les attributs non obligatoires. Dans le tableau 6.1, nous illustrons à titre d'exemple les nouvelles importances des attributs visuels décrivant la visualisation ***"Nuage 3D Cube Chanel V2"*** décrite dans la section 5.5.4.2.

3. Dans la dernière phase de notre heuristique, une mise en correspondance entre les deux listes $L_A$ et $L_V$ est effectuée. Cette opération consiste à affecter à chaque $A_{ij}$ de $L_V$ le premier $A_i$ de $L_A$ ayant le même type et permettant de représenter toutes les valeurs possibles de $A_i$ en respectant la limite visuelle de $A_{ij}$.

| Attributs visuels | | | |
|---|---|---|---|
| Nom | Obligatoire | Importance $v_{ij}$ | Importance $v'_{ij}$ |
| axe X | oui | 50 | 100 |
| axe Y | oui | 50 | 100 |
| axe Z | oui | 50 | 100 |
| taille pyramidion haut | non | 45 | 45 |
| taille pyramidion bas | non | 45 | 45 |
| couleur pyramidion haut | non | 40 | 40 |
| couleur pyramidion bas | non | 40 | 40 |
| couleur | non | 40 | 40 |
| image | non | 35 | 35 |
| Texte Haut | non | 22 | 22 |
| Texte Bas | non | 22 | 22 |

TABLE 6.1 – Description de la visualisation ***"Nuage 3D Cube Chanel V2"*** avec les nouvelles importances utilisées dans le processus d'appariement.

L'algorithme 1 décrit en détail le processus d'appariement défini dans VizAssist. Notre méthode permet de favoriser d'une part les visualisations associant aux attributs de données les plus pertinents, des signes visuels faciles à percevoir, et d'autre part les visualisations, toute chose étant égale par ailleurs, représentant plus d'attributs de données que les autres. Notons que si un attributs $A_{ij}$ est obligatoire et n'est pas renseigné (apparié), alors la visualisation $V_i$ ne peut pas être utilisée.

À la fin de la seconde étape du processus d'appariement, VizAssist génère pour chacune des visualisations retenues dans la première étape son appariement avec le jeu de données utilisateur. Les appariements obtenus sont utilisés ensuite par VizAssist comme paramétrage pour générer toutes les visualisations pré-sélectionnées sur la même interface (voir figure 6.2). Nous décrivons dans la section suivante la procédure de choix des visualisations et les critères que nous proposons pour guider les utilisateurs dans cette tâche.

### 6.2.3 Heuristique d'appariement

**Algorithm 1** algorithme d'appariement

**ENTREE :**
**1.** $D$ : le jeu de données utilisateur.
**1.** $U$ : le vecteur des importances $u_i$ des attributs $A_i$ de $D$.
**2.** $P$ : le vecteur des importances $p_{ij}$ fixés par l'utilisateur.
**3.** $w'$ : le seuil d'appariement minimum fixé a priori par l'utilisateur.

**SORTIE :**
**1.** $L_{param}$ : liste des paramétrages des visualisations satisfaisant $P$.

**Début**

$L_{param} \leftarrow \phi$ ;
Extraire $P$ à partir de l'interface des objectifs utilisateurs ;
Extraire $U$ à partir de $D$ ;
$L_A \leftarrow$ trier attributs de données selon type de données et importances $U$ ;
**Pour** (toutes les visualisations $V_i$ de la base de connaissances) **Faire**
  Extraire le vecteur $O_j$ des importances d'objectifs de la visualisation $V_i$ ;
  Calculer $w_i$ ;
  **Si** ($w_i \geq w'$) **Alors**
    $V_i$ est sélectionnée pour être proposée à l'utilisateur ;
  **Finsi**
**Fin pour**
$l_{param} \leftarrow \phi$ ; // liste de paramétrage (mise en correspondance) d'une visualisation $V_i$ ;
**Pour** (chaque visualisation pré-sélectionnée $V_i$ ) **Faire**
  Extraire le vecteur des importances $V$ de la visualisation $V_i$ ;
  $L_V \leftarrow$ trier les attributs visuels selon type de données et importances $V$ ;
  **Pour** (chaque $A_{ij}$ de $L_V$) **Faire**
    Chercher le premier attribut $A_i \in L_A$ de type compatible avec $A_{ij}$ ;
    **Si** (limite visuelle de $A_{ij} \leq$ nombre de valeurs possibles de $A_i$) **Alors**
      $l_{param} \leftarrow$ affectation($A_{ij}$, $A_i$) ;
    **Finsi**
    **Si** (un des attributs $A_{ij}$ obligatoire n'est pas renseigné) **Alors**
      $V_i$ ne peut être utilisée ;
    **Finsi**
  **Fin pour**
  $L_{param} \leftarrow l_{param}$ ;
**Fin pour**
**Fin.**

## 6.3 Choix d'une visualisation

À l'issue du processus d'appariement, l'assistant prévisualise chacune des visualisations sélectionnées avec le jeu de données utilisateur (voir section 6.3.2). Contrairement aux précédents assistants, dans VizAssist toutes ces visualisations sont affichées sur une même interface. Cela permet en effet d'éviter aux utilisateurs un processus itératif de choix des visualisations et de se concentrer plutôt sur les tâches d'exploration et d'analyse de leurs données. La figure 6.1 illustre un exemple de résultat fourni par l'interface de suggestion des visualisations, que propose VizAssist, appliquées directement sur le jeu de données ecoli [8]. En effet, dans cette première interface, les visualisations proposées sont disposées dans la partie inférieure et la visualisation la plus appropriée (ayant le score d'appariement le plus élevé) est affichée directement dans la partie supérieure. Cette dernière fait partie aussi de la liste des visualisations proposées et elle est placée en bas à droite de cette liste. Dans le but d'aider l'utilisateur à choisir une visualisation parmi celles proposées, l'ensemble des visualisations est triée par ordre décroissant (de gauche à droite) sur le score d'appariement calculé (voir section 6.3.3).

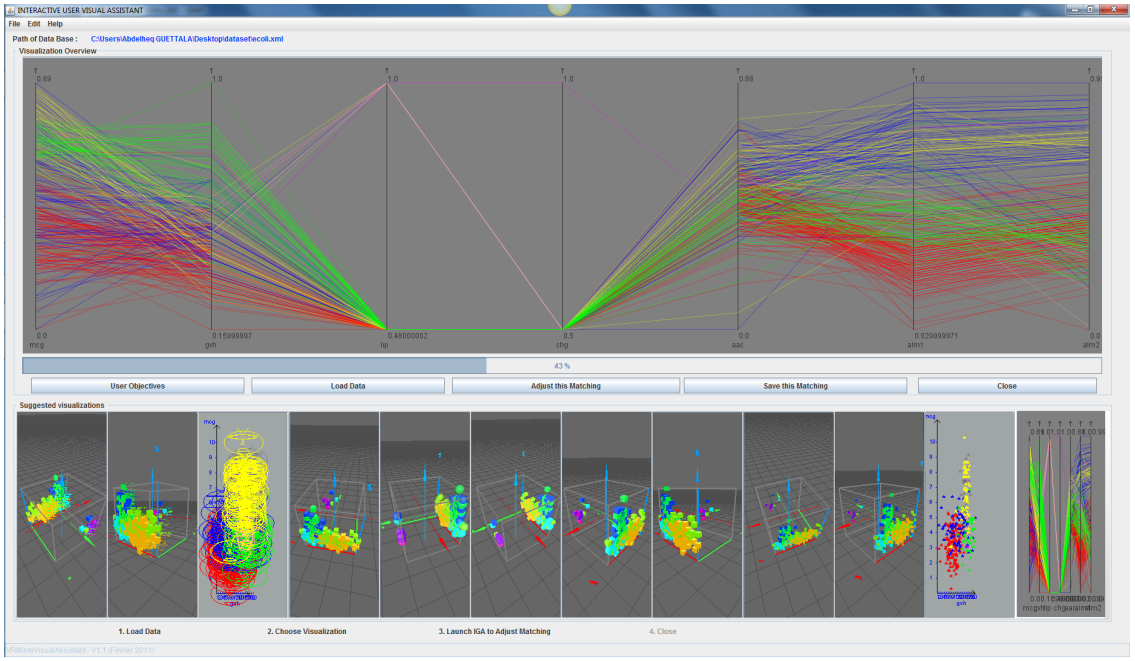



FIGURE 6.1 – Interface de suggestion des visualisations proposée par VizAssist (la base de données utilisée dans cette exemple est ecoli).

### 6.3.1 Interface de sélection des visualisations

La figure 6.2 illustre les différents avantages de la première interface de VizAssist (interface de suggestion des visualisations). Parmi ces avantages on peut citer, sa capacité à

8. http ://archive.ics.uci.edu/ml/datasets/Ecoli

permettre à l'utilisateur d'explorer ses données avec toutes les fonctionnalités associées à chaque visualisation. Le second avantage de cette interface est qu'elle offre la possibilité d'explorer de manière comparative l'ensemble des visualisations proposées directement sur les données de l'utilisateur. Ce qui signifie que l'utilisateur n'a pas un aperçu statique de ce que donne la visualisation mais plutôt la possibilité de tester dynamiquement toutes les fonctionnalités des visualisations sur ses données. Il peut ainsi vérifier si la visualisation lui convient (informations présentées, vitesse d'affichage, etc.) et si les interactions proposées l'aident à atteindre ses objectifs. Ensuite, lorsque cela est possible, nous avons ajouté la possibilité de sélectionner des données dans une visualisation et de voir apparaître les données sélectionnées dans les autres grâce à la technique nommée "brushing" (voir figure 6.3) et décrite dans [Becker et Cleveland, 1987]. Cette partie de VizAssist peut donc être vue non seulement comme un outil permettant à l'utilisateur de choisir interactivement une visualisation sans avoir à spécifier manuellement un paramétrage, mais également comme une interface multi-visualisations.

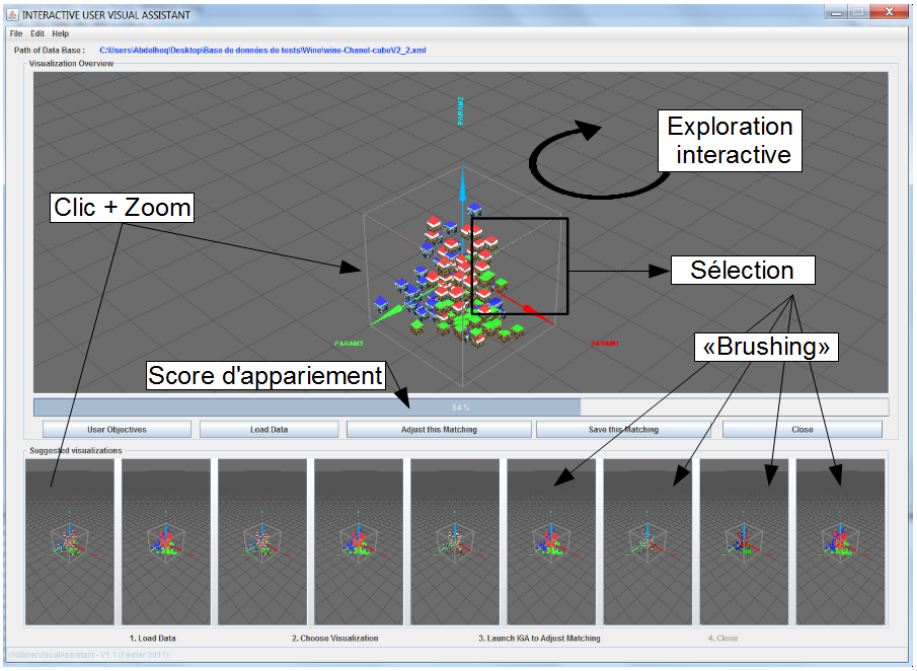



FIGURE 6.2 – Fonctionnalités et interactions proposées dans l'interface de suggestion des visualisations de VizAssist.

### 6.3.2 Protocole de prévisualisation des données utilisateurs

Dans le but de faciliter le choix des visualisations, VizAssist permet de prévisualiser chacune des visualisations proposées avec les données utilisateur à représenter. Pour cela, nous avons mis au point un protocole permettant d'intégrer dans VizAssist toute librairie JAVA représentant une visualisation (ex. coordonnées parallèle [Hauser *et al.*, 2002]) [9] (voir

9. http ://www.mediavirus.org/parvis/

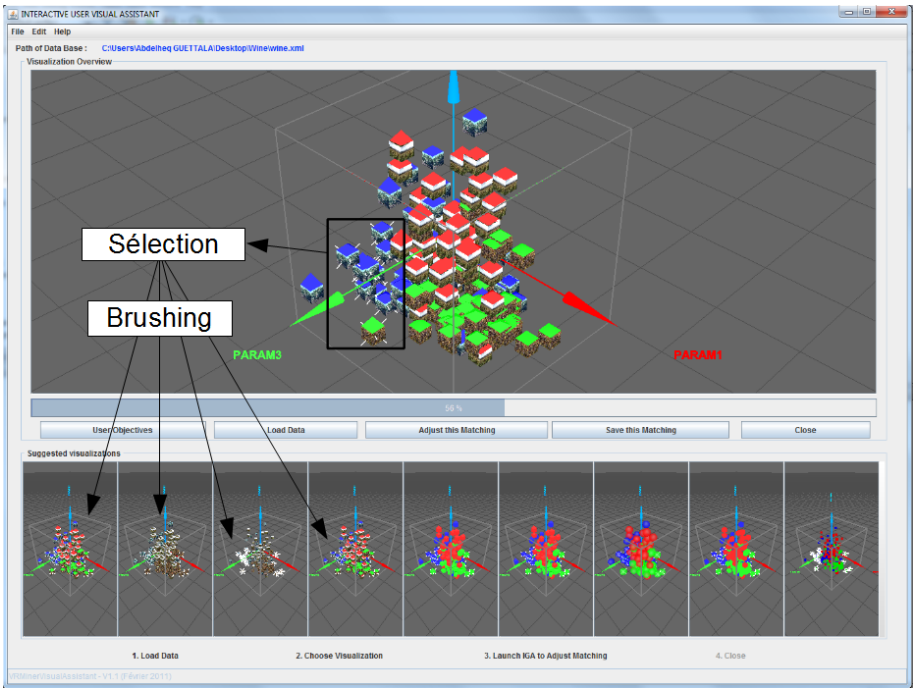



FIGURE 6.3 – Utilisation de la technique de "brushing" dans l'interface de suggestion des visualisations de VizAssist.

section 9.4). Ce protocole passe par un échange au format XML permettant d'indiquer à la librairie de visualisation, quel est le paramétrage trouvé par l'assistant.

La figure 6.4 illustre le format du fichier XML utilisé par VizAssist. Ce dernier est décomposé en 4 parties. La première partie nommée *"structure"* contient la description des attributs de données $A_i$ du jeu de données utilisateur. Dans cette partie nous renseignons pour chaque attribut de données du jeu de données à visualiser son nom, son type et son importance. La deuxième partie *"data"* comporte les valeurs des $n$ données $d_i$ de $D$. L'enrichissement de cette deuxième partie dépend de la description des attributs de données définie dans la partie *"structure"*. Cela signifie que pour chaque donnée du jeu de données à représenter, nous renseignons les valeurs correspondantes à tous les attributs de données. La troisième et quatrième partie du fichier concerne les paramétrages des visualisations. En effet, la partie *"visualisations"* sert à enregistrer le paramétrage des visualisations qu'un utilisateur désire ré-exploiter dans des sessions d'utilisation ultérieures. Tandis que la partie *"geneticalgorithm"* est utilisée dans un processus d'optimisation interactif des paramétrages proposés à l'issue de la phase d'appariement du système. Ce processus est présenté en détails dans le chapitre suivant (voir chapitre 7). La figure 6.5 illustre le paramétrage de la visualisation ***"Coordonnées parallèles"*** représentée dans la figure 6.1 tel qu'il est décrit dans le fichier XML et passé en paramètre à la librairie JAVA permettant de générer cette visualisation.

```
<?xml version="1.0" encoding="UTF-8"?>
<database version="2">
  <structure>
    <attribute>
      <name>sepal_length</name>
      <type>numeric</type>
      <importance>50</importance>
    </attribute>
    <attribute>
      <name>sepal_width</name>
      <type>numeric</type>
      <importance>50</importance>
    </attribute>
    <attribute>
      <name>petal_length</name>
      <type>numeric</type>
      <importance>50</importance>
    </attribute>
    <attribute>
      <name>petal_width</name>
      <type>numeric</type>
      <importance>50</importance>
    </attribute>
    <attribute>
      <name>classe</name>
      <type>symbolic</type>
      <importance>50</importance>
    </attribute>
  </structure>
  <data>
  <visualizations />
  <geneticalgorithm />
</database>
```

FIGURE 6.4 – Format du fichier XML utilisé par VizAssist.

### 6.3.3 Calcul du score d'appariement

Nous notons $U = \{u_1, ..., u_k\}$ le vecteur des importances des $k$ attributs de données de $D$ et $V = \{v_{1i}, ..., v_{mi}\}$ le vecteur des importances des $m$ attributs visuels de la visualisation $V_i$. Nous rappelons que les valeurs $u_i$ représentent l'intérêt que porte l'utilisateur aux attributs de données $A_i$ et les valeurs $v_{ij}$ représentent l'importance des attributs visuels $A_{ij}$ fixées dans notre matrice "attribut de données × attributs visuels" (voir chapitre 5). Nous rappelons aussi que toutes les valeurs $u_i$ et $v_{ij}$ sont définies dans l'intervalle $[0, 100]$.

Afin de mesurer le degré d'adéquation entre un jeu de données utilisateur et chacune des visualisations $V_i$ pré-sélectionnées par notre système, nous calculons le produit scalaire entre $U$ et $V$ de chaque $V_i$ en considérant les attributs appariés entre eux. Le résultat de cette opération permet de mesurer le score d'appariement, noté $sim(U, V)$, entre le jeu de données utilisateur et chacune des visualisations retenues lors de la première phase du

```
<nuage3D>
<CoordonneesParalleles>
  <profil1 valeur="profil1">
    <Axis1>mcg</Axis1>
    <Axis2>gvh</Axis2>
    <Axis3>lip</Axis3>
    <Axis4>chg</Axis4>
    <Axis5>aac</Axis5>
    <Axis6>alm1</Axis6>
    <Axis7>alm2</Axis7>
    <classe>Classe</classe>
  </profil1>
</CoordonneesParalleles>
<ScatterPlot>
<Chernoff>
```

FIGURE 6.5 – Description du paramétrage de la visualisation *"Coordonnées parallèles"* dans le fichier XML utilisé par VizAssist, représentée dans la figure 6.1.

processus d'appariement. L'utilisation de ce score a pour intérêt de fournir une information supplémentaire à l'utilisateur du système pour l'aider à choisir entre les différentes visualisations proposées. La *"barre de progression"* située entre la partie supérieure et la partie inférieure de la première interface (voir figure 6.2) de notre système (interface de suggestion des visualisations) est réservée à l'affichage de ce score d'appariement. La formule utilisée pour calculer le score d'appariement est la suivante :

$$sim(U, V) = \sum_{ij} u_i \times v_{ij} \in [0, 1]$$

Dans VizAssist, nous utilisons ce score pour trier les visualisations dans un ordre décroissant selon leur seuil de similarité. Afin de présenter le score d'appariement entre le jeu de données utilisateur et chacune des visualisations sélectionnée comme un pourcentage, nous divisons la valeur $sim(U, V)$ de chaque $V_i$ sur la similarité maximale obtenue parmi les visualisations retenues.

Partant du constat que les deux vecteurs $U$ et $V$ peuvent ne pas avoir la même dimension (le nombre d'attribut de données de $D$ est différent du nombre d'attributs visuels de $V_i$), une normalisation de ces derniers est effectuée avant de calculer le score d'appariement $sim(U, V)$. Cette normalisation concerne le nombre d'attributs de données/visuels qui devrait être le même entre les deux vecteurs.

## 6.4 Exemple illustratif

Dans le but d'expliquer en détail le processus d'appariement tel qu'il est proposé par VizAssist, nous illustrons son déroulement à travers un exemple. Nous commençons par montrer comment l'utilisateur spécifie ses objectifs et la manière dont ils sont pris en compte par VizAssist. Ensuite, nous illustrons comment l'assistant récupère les descripteurs des visualisations depuis la base de connaissances en prenant comme exemple la visualisation *"Nuage 3D Sphère"*. Nous présentons aussi comment sont générés les paramétrages des visualisations pré-sélectionnées pour visualiser le jeu de données utilisateur. Nous terminons cette section par la présentation de l'affichage de la liste des visualisations proposées aux utilisateurs à l'issue du processus d'appariement.

### 6.4.1 Descripteurs des objectifs utilisateur

Comme déjà indiqué ci-dessus, la première étape du processus d'appariement consiste à pré-sélectionner les visualisations les plus appropriées aux objectifs de l'utilisateur. Pour cela, ce dernier doit indiquer les priorités de chaque objectif de la liste proposée par le système ainsi que la dimension et la catégorie visuelle des visualisations qu'il souhaite. La figure 6.6 illustre un exemple de spécification des objectifs utilisateur. En effet, on peut remarquer que l'utilisateur désire visualiser ses données avec des visualisations en *"3 dimensions"* dont la catégorie visuelle est *"spatiale"*. Par ailleurs, l'utilisateur a fixé une importance très élevée *"very important"* aux deux objectifs *"Overview of all data"* et *"Discovery of classes"*. L'importance de l'objectif *"Data selection"* est fixée à *"important"*, tandis que *"Outlier detection"* à *"less important"*. Le reste des objectifs n'a aucune importance pour l'utilisateur. Nous présentons dans le tableau 6.2, la description des objectifs utilisateur telle qu'elle est prise en compte par notre système, pour générer le vecteur des importances $P$.

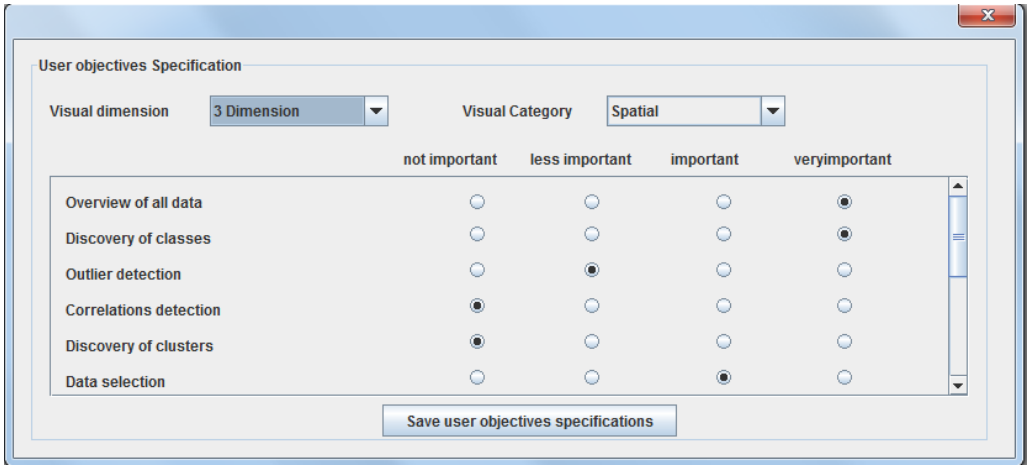



FIGURE 6.6 – Description des objectifs utilisateur.

| Objectifs utilisateur | Importance |
|---|---|
| Avoir une vue d'ensemble | 3 |
| Découvrir des classes | 3 |
| Sélectionner des données | 2 |
| Détecter des individus aberrants | 1 |
| Détecter des clusters | 0 |
| Déterminer des tendances | 0 |
| . | 0 |
| . | 0 |

TABLE 6.2 – Représentation des importances (ordre de priorité) des objectifs de la visualisations *"Nuage 3D Sphère"*, récupérées à partir de la base de connaissances (vecteur des importances $P$).

### 6.4.2 Description du jeu de données utilisateur

La base de données (jeu de données utilisateur) choisie dans notre exemple est la base des IRIS [Fisher, 1936]. Nous rappelons que cette base de données est décrite par 150 données dont chacune est définie par 4 attributs de données de type *"numérique"* et 1 attribut de données de type *"symbolique"*. Le tableau 6.3 illustre la description des attributs de données dont les importances sont fixées à 50 et servent à générer le vecteur des importances $U$.

| Attributs de données | | |
|---|---|---|
| Nom | Type de données | Importance |
| Sepal length | numérique | 50 |
| Sepal width | numérique | 50 |
| Petal length | numérique | 50 |
| Petal width | numérique | 50 |
| Classe | symbolique | 50 |

TABLE 6.3 – Liste des attributs de données $L_A$.

### 6.4.3 Descripteurs des visualisations $V_i$

Avant de commencer la première étape du processus d'appariement, notre système récupère les descripteurs de chacune des visualisations à partir de la base de connaissances. Deux types de descripteurs sont extraits pour chacune des visualisations. Nous illustrons à titre d'exemple, dans les deux tableaux 6.4 et 6.5, les deux descripteurs de la visualisation *"Nuage 3D Sphère"*. En effet, le tableau 6.4 représente l'ordre des priorités des objectifs de la visualisations *"Nuage 3D Sphère"*, récupérées à partir de la base de connaissances contenant le vecteur des importances $O_i$. Le tableau 6.5 illustre la dimension et la catégorie visuelle de la visualisation *"Nuage 3D Sphère"* ainsi qu'une description détaillée de ses attributs visuels. En effet, cette dernière permet de créer le vecteur des importances $V$ ainsi que de vérifier la limite visuelle des attributs visuels lors de leurs mises en correspondance avec les attributs de données du jeu de données à représenter (voir algorithme 1).

### 6.4.4 Générer les paramétrages des visualisations pré-sélectionnées

Le tableau 6.6 illustre le résultat fourni à l'issue de la phase de mise en correspondance entre les attributs de données $A_i$ de la base de données des IRIS (voir tableau 6.3) et les attributs visuels $A_{ij}$ de la visualisation *"Nuage 3D Sphère"* (voir tableau 6.5). Une fois générés, les paramétrages des visualisations pré-sélectionnées à l'issue du processus d'appariement sont appliqués directement sur le jeu de données à représenter. L'utilisateur peut ainsi explorer directement ses données dans chacune des visualisations pour choisir celle qui convient le mieux à ses besoins. La figure 6.8 illustre le rendu visuel de la base de données IRIS avec la visualisation *"Nuage 3D Sphère"*, généré à partir du paramétrage décrit dans le tableau 6.6. On peut constater que dans ce dernier, seulement 3 des 4 attributs

| Objectifs utilisateur | Importance |
|---|---|
| Avoir une vue d'ensemble | 3 |
| Découvrir des classes | 3 |
| Sélectionner des données | 3 |
| Détecter des clusters | 2 |
| Déterminer des tendances | 2 |
| Détecter des individus aberrants | 2 |
| Détecter des corrélations | 1 |
| Sélectionner des attributs | 0 |
| . | 0 |
| . | 0 |

TABLE 6.4 – Représentation des importances (ordre de priorité) des objectifs de la visualisations *"Nuage 3D Sphère"*, récupérées à partir de la base de connaissances (vecteur des importances $O_i$).

| Nuage 3D Sphère | | | | | |
|---|---|---|---|---|---|
| 3 dimensions | | | spatiale | | |
| Attributs visuels | | | | | |
| Nom | Type visuel | Type de données | Obligatoire | limite visuelle | Importance |
| axe X | position | numérique | oui | 100 | 100 |
| axe Y | position | numérique | oui | 100 | 100 |
| axe Z | position | numérique | oui | 100 | 100 |
| couleur | couleur | symbolique | non | 12 | 41 |
| Text Haut | texte | texte | non | $\infty$ | 32 |
| Text Bas | texte | texte | non | $\infty$ | 32 |

TABLE 6.5 – Description de la visualisation ***"Nuage 3D Sphère"*** incluant la liste des attributs visuels $L_V$.

de données de type *"numériques"*, du jeu de données IRIS sont visualisés. Cela justifie le score d'appariement (86%) de la visualisation *"Nuage 3D Sphère"* affiché sur la barre de progression dans l'interface de suggestion des visualisations (voir figure 6.7).

### 6.4.5 Liste des visualisations proposées et leurs paramétrages

Nous supposons que dans cet exemple illustratif, l'utilisateur n'a pas fixé de seuil de similarité minimum. Cela signifie que toutes les visualisations *3D* et *spatiale* de notre base de connaissances seront proposées à l'utilisateur. La figure 6.7 illustre le résultat fourni par notre système à l'issue du processus d'appariement. En effet, toutes les visualisations suggérées sont appliquées directement sur le jeu de données IRIS [10].

10. http ://archive.ics.uci.edu/ml/datasets/Iris

| Attributs de donnée | Attributs visuels |
|---|---|
| Sepal length | axe X |
| Sepal width | axe Y |
| Petal length | axe Z |
| Classe | couleur |

TABLE 6.6 – Résultat de la mise en correspondance entre la liste des attributs de données $L_A$ et la liste des attributs visuels $L_A$ de la visualisation *"Nuage 3D Sphère"*.

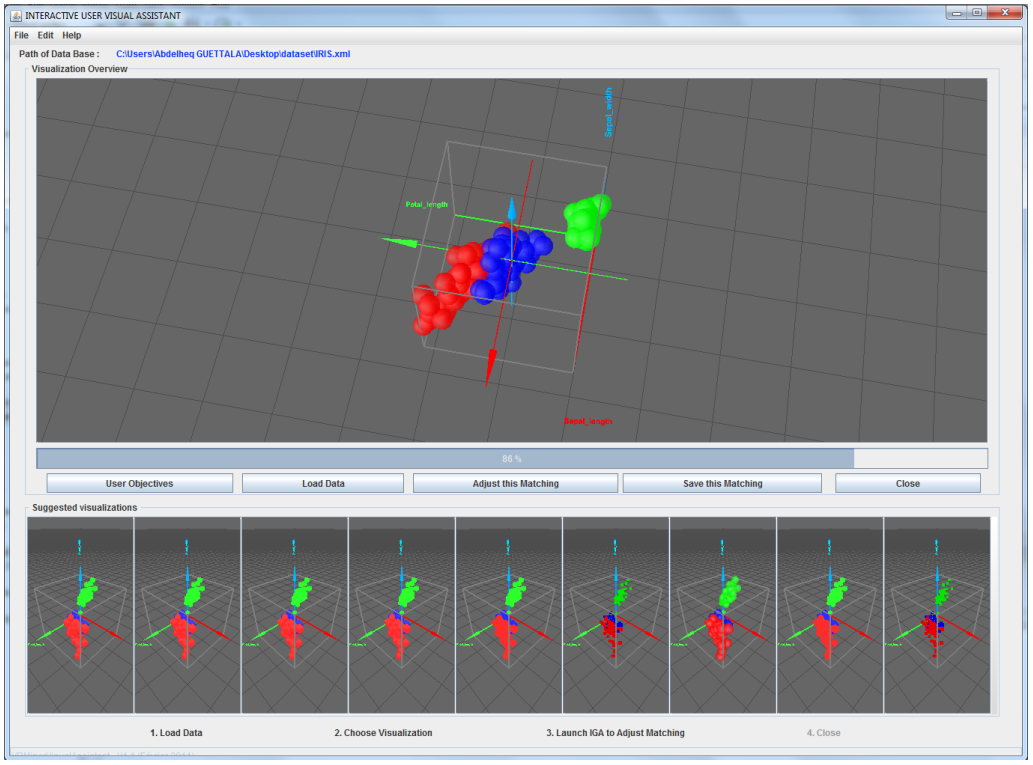



FIGURE 6.7 – Liste des visualisations proposées et générées à l'issue de la phase d'appariement.

## 6.5 Conclusion

Dans ce chapitre, nous avons présenté le module d'appariement et de suggestions des visualisations de VizAssist. L'apport majeur de ce module est qu'il permet de suggérer plusieurs visualisations avec les meilleurs paramétrages possibles appliquées directement sur les données utilisateur. L'autre avantage de ce module est qu'il repose sur un protocole simple de visualisation, passant par un échange au format XML pour communiquer les paramétrages à la librairie de visualisation (codées en JAVA). L'interface de suggestion des visualisations utilisée par VizAssist a deux principaux avantages. Son premier avantage est qu'elle permet une exploration interactive et comparative des jeux de données utilisateurs sur l'ensemble des visualisations proposées en s'appuyant sur un affichage

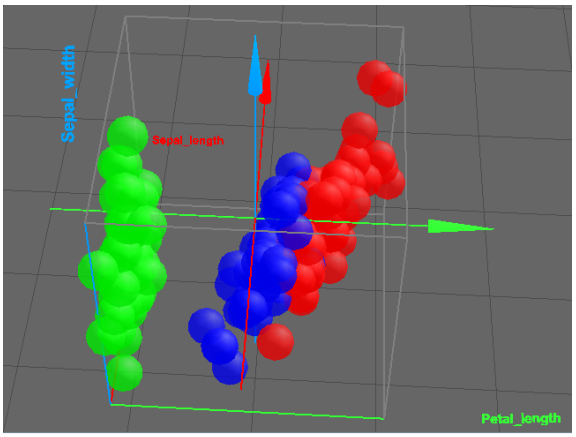



FIGURE 6.8 – Aperçu de la visualisation *"Nuage 3D Sphère"* avec le paramétrage proposé à l'issue du processus d'appariement.

multi-visualisations. Le second avantage de l'interface est qu'elle propose une vue dynamique, ce qui permet aux utilisateurs de tester les fonctionnalités des visualisations sur leurs données.

Même si le module d'appariement et de suggestions des visualisations propose les meilleurs paramétrages possibles des visualisations suggérées, ces derniers (paramétrages) peuvent ne pas être suffisamment adaptés à l'utilisateur ou à ses données, et ceci pour deux raisons. Tout d'abord, ces appariements sont effectués en fonction de critères généraux sur la perception visuelle. Les paramétrages peuvent donc être considérés comme suffisamment bons pour donner une idée de l'intérêt des visualisations pour l'utilisateur. Cependant, la perception visuelle étant subjective, il est possible que l'utilisateur souhaite modifier le paramétrage d'une visualisation donnée. Par ailleurs, l'utilisateur peut ne pas connaître l'importance réelle des attributs de données, cette importance pouvant se révéler au cours d'une exploration interactive des données. Pour ces deux raisons, nous avons défini une nouvelle approche fondée sur un ***Algorithme Génétique Interactif*** dont l'objectif est d'optimiser de manière interactive et visuelle le paramétrage d'une visualisation choisie par les utilisateurs sur la première interface de VizAssist. Cette nouvelle approche est décrite en détail dans le chapitre suivant.

# Chapitre 7

# Module de paramétrage interactif

**Résumé** : Étant conscient de la difficulté du processus de paramétrage des visualisations dans les systèmes existants, lié à la subjectivité et au niveau de perception visuelle des utilisateurs, nous proposons dans ce chapitre notre nouvelle approche pour résoudre ce problème. En effet, dans VizAssist, nous nous appuyons sur une étape visuelle et interactive qui utilise un algorithme génétique interactif (AGI). L'objectif principal de notre méthode est donc d'affiner et d'optimiser l'appariement entre les attributs de données et les attributs visuels en permettant aux utilisateurs de participer au processus. Afin d'illustrer les apports de notre approche, nous décrivons le principe général de notre AGI à travers l'accomplissement d'une tâche de clustering.

## 7.1 Introduction

Trouver le paramétrage le plus adéquat d'une visualisation $V_i$ pour accomplir une tâche donnée $T_i$, est une opération très difficile. Cette difficulté vient du fait que ce processus dépend étroitement de la subjectivité et du niveau individuel de perception visuelle des utilisateurs. En effet, quand un utilisateur désire représenter un ensemble de données $D$, défini par un nombre important de dimensions $k$, sélectionner le meilleur appariement entre les attributs de données $A_i$ de $D$ et les signes visuels $A_{ij}$ de $V_i$, satisfaisant ses objectifs, nécessite : 1) soit une pré-connaissance sur le jeu de données à visualiser, 2) soit l'appui d'un expert en visualisation d'informations, 3) ou soit un test de toutes les combinaisons possibles des attributs $A_i$ avec $A_{ij}$, jusqu'à aboutir à celle qui lui semble la meilleure. Pour des utilisateurs novices, cette dernière technique est généralement la plus utilisée car elle garantit de trouver le paramétrage exact. Cependant, le temps nécessaire à l'exploration de tout l'espace de combinaisons possibles est très difficile à estimer et dépend principalement du nombre de dimensions $k$ du jeu de données $D$ à représenter. Pour résoudre ce problème, il est nécessaire de combiner un type d'évaluation basé sur : 1) la prémisse que le concept *d'intérêt* chez les utilisateurs est subjectif plutôt qu'objectif, et 2) une technique de recherche permettant de parcourir un grand espace de solutions (combinaisons de $A_i$ avec $A_{ij}$) de manière rapide et personnalisée.

Peu de solutions traitant les contraintes du processus mentionné ci-dessus sont proposées dans le domaine de la visualisation. D'ailleurs, le problème d'appariement entre les attributs de données d'un ensemble de données à visualiser et les attributs visuels d'une visualisation donnée est considéré comme l'un des principaux problèmes posé actuellement pour une automatisation complète du processus de visualisation (voir figure 1.7). Parmi ces solutions, on trouve [Venturini *et al.*, 1997] [Ceglar *et al.*, 2003] [Boudjeloud et Poulet, 2005] [Cancino *et al.*, 2012] qui ont montré que l'utilisation d'une évaluation interactive s'appuyant sur une optimisation génétique représente un avantage considérable. Cependant, ces approches sont limitées dans leurs utilisations car elles sont appliquées seulement dans le cadre de visualisations statiques et pour un seul type de visualisation.

Dans VizAssist, nous proposons une étape visuelle et interactive qui s'appuie sur un algorithme génétique interactif (AGI) pour résoudre le problème cité ci-dessus. L'objectif principal de notre méthode est d'affiner et d'optimiser l'appariement entre les attributs de données et les attributs visuels. À l'inverse des assistants qui existent dans la littérature (voir chapitre 2), s'appuyant seulement sur un modèle général à base de directives visuelles, VizAssist propose une personnalisation du paramétrage fondée sur les préférences visuelles individuelles des différentes catégories d'utilisateurs (novices ou experts). De plus, notre méthode peut s'appliquer sur plusieurs visualisations statiques ou dynamiques représentées en 2 ou 3 dimensions (voir chapitre 8). Nous décrivons dans ce chapitre, le principe de base de notre algorithme, la représentation des individus d'une population $P$ (adoptée pour le problème que nous traitons) et la description des différents opérateurs génétiques utilisés par l'AGI. Ensuite, nous illustrons à travers un exemple les étapes du déroulement de l'AGI en détaillant l'interface assurant cette opération. Nous terminons le chapitre par une conclusion synthétisant les apports de notre approche.

## 7.2 Principe de l'algorithme génétique interactif (AGI)

À l'issue de la première étape proposée par VizAssist qui consiste à choisir une visualisation $V_i$ (voir chapitre 6), l'utilisateur peut améliorer son paramétrage. Cette amélioration concerne, le choix de la combinaison la plus adéquate des attributs de données $A_i$ décrivant son jeu de données $D$ et les signes visuels $A_{ij}$ décrivant $V_i$. Cette opération est possible grâce à une étape visuelle et interactive qui s'appuie sur un algorithme génétique interactif (AGI).

Dans notre AGI, une population $P$ comporte 8 individus dont chacun représente un paramétrage possible de la visualisation $V_i$ (un appariement entre $A_i$ et $A_{ij}$). Tous les individus de $P$ sont représentés visuellement sur une seule interface (voir figure 7.2). Une explication détaillée de la représentation des individus utilisée dans notre AGI est proposée dans la section 7.3.

D'une manière générale, la seule différence entre un AGI et un algorithme génétique classique (AG) réside dans l'étape d'évaluation des individus (voir chapitre 3). En effet, dans un AGI l'utilisateur remplace la fonction objectif utilisée dans les AG pour évaluer les individus de la population $P$ (notée également $P(t)$ à la génération $t$). La figure 7.1 illustre l'architecture générale de l'AGI utilisée par VizAssist.

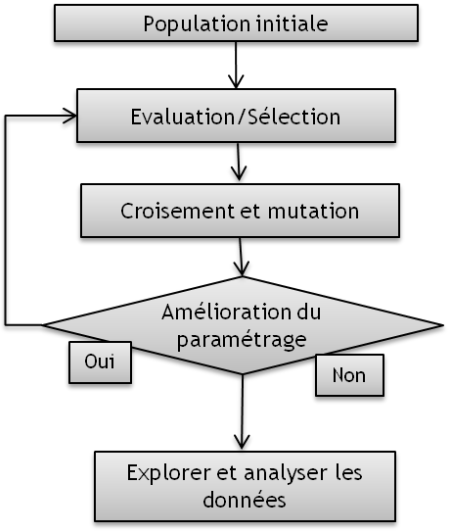



FIGURE 7.1 – Architecture générale de l'AGI.

Le principe général du fonctionnement de notre AGI est le suivant :

1. On commence par générer une population d'individus (voir section 7.3) de façon aléatoire à partir du paramétrage de la visualisation choisie lors de la phase de suggestion (voir chapitre 6). Chacun des paramétrages de la population $P$ est appliqué à la visualisation $V_i$ avec l'ensemble de données $D$, et chaque visualisation est affichée sur la même interface (voir la figure 7.2).
2. L'utilisateur peut observer chacune des visualisations résultant des 8 paramétrages potentiels, avec les mêmes possibilités d'interaction que dans la première interface

(voir figure 6.2). L'utilisateur sélectionne alors les individus à retenir ou à écarter.

3. Si l'utilisateur trouve le paramétrage qui lui convient, il peut arrêter le déroulement de l'AGI à cette étape et enregistrer les meilleurs paramétrages pour une utilisation ultérieure, sinon, il passe à l'étape suivante.
4. Pour générer une nouvelle population $P(t+1)$, l'AGI conserve les $N$ individus sélectionnés dans $P(t)$ et remplace les autres par de nouveaux individus générés en s'appuyant sur les différents opérateurs génétiques (croisement, mutation). L'application des opérateurs génétiques se fait selon le nombre d'individus $N$ sélectionnés par l'utilisateur. Différents scénarios de génération de $P(t+1)$ sont envisageables avec notre AGI :
   (a) $\mathbf{N = 0}$ **:** nous générons tous les individus de $P$ de manière aléatoire.
   (b) $\mathbf{N = 1}$ **:** nous générons les nouveaux individus par l'application de l'opérateur de mutation seul.
   (c) $\mathbf{N = 8}$ **:** la nouvelle génération est identique à la précédente.
   (d) $\mathbf{N \in [2,\ 7]}$ **:** parmi les $N$ individus sélectionnés deux individus sont choisis de manière aléatoire et uniforme, puis nous leur appliquons un croisement. Ensuite, nous conservons un des deux descendants de cette opération et nous lui appliquons une mutation avec une probabilité de modifier chaque gène notée $P_{mutation}$. L'enfant ainsi engendré vient remplacer un individu non sélectionné dans $P(t)$. Cette étape s'arrête une fois que tous les individus non sélectionnés ont été remplacés.
5. Retour à l'étape 2.

## 7.3 Représentation des individus

Dans notre AGI, un individu $I$ de la population $P$ représente un paramétrage possible des attributs visuels $A_{ij}$, d'une visualisation $V_i$, avec les attributs de données $A_i$, d'un jeu de données $D$ à visualiser. Plus précisément, on utilise pour $I$ un encodage indirect défini sous la forme d'un vecteur de poids $(g_1, g_2, ..., g_k)$ de l'ensemble des attributs de données $A_i$, où chacune des importances représente un *gène* dont la valeur est définie dans l'intervalle $[0, 100]$. Ces poids remplacent les importances initiales $u_i$ utilisées dans le processus de mise en correspondance (voir chapitre 6), et leurs valeurs changent à chaque itération de l'AGI.

| $g_1$ | $g_2$ | $\cdots$ | $g_{k-1}$ | $g_k$ |
|---|---|---|---|---|

$$g_i \in \{g_1, g_2, ..., g_k\}$$
$$\forall i, j \in [1, k], g_i \neq g_j$$

Donc, un individu $I$ représente indirectement un appariement, puisque il vient influencer directement l'ordre dans lequel les attributs de données sont fixés dans l'appariement avec les attributs visuels. Pour notre problème, l'avantage de cette représentation est qu'à

chaque itération de l'AGI, de nouveaux vecteurs de poids sont générés et donc de nouveaux appariements, entre les attributs visuels et les attributs de données, sont proposés. Le codage des individus que nous avons utilisé est une représentation en nombres réels [Wright, 1991] [Goldberg, 1991].

## 7.4 Opérateurs génétiques

### 7.4.1 Initialisation de la population

Fixer la taille d'une population $P$ dans un AGI pour éviter le problème de fatigue chez les utilisateurs [Takagi, 2001] constitue le problème le plus notable dans ce type d'algorithmes. Contrairement aux algorithmes génétiques classiques, les individus d'une population de l'AGI sont présentés visuellement pour une évaluation directe par l'utilisateur. En effet, si le nombre d'individus de $P$ affichés est assez important, le risque qu'un utilisateur abandonne le déroulement de l'AGI est très élevé à cause du nombre d'individus à évaluer à chaque itération. Pour cette raison, nous avons opté pour une population définie par 8 individus seulement.

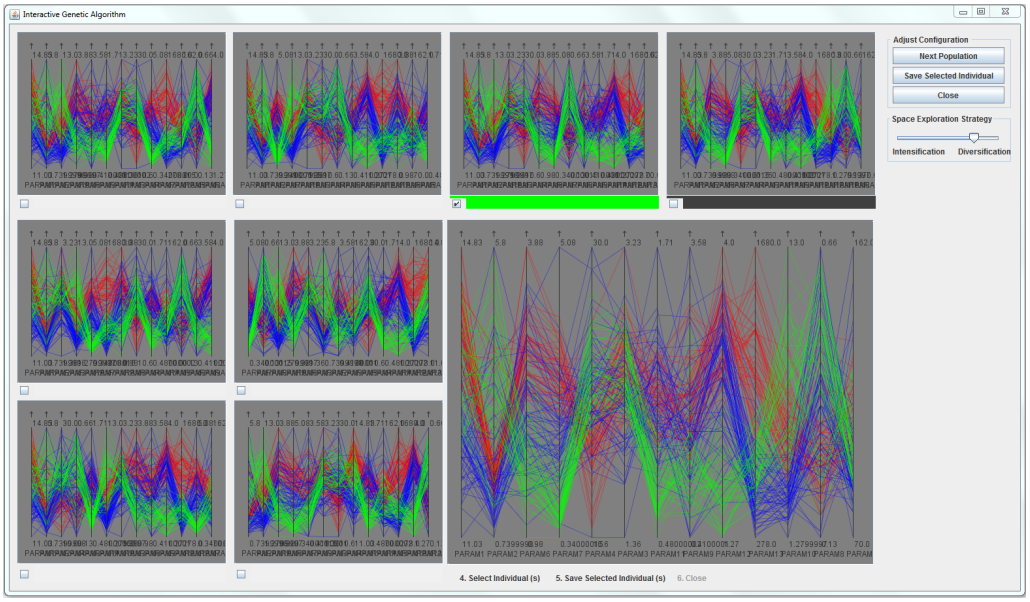



FIGURE 7.2 – Seconde interface de VizAssist. A travers un processus interactif et visuel, cette interface permet de dérouler l'AGI pour affiner et optimiser l'appariement entre les attributs de données et les attributs visuels d'une visualisation $V_i$. Exemple d'application de l'AGI sur un jeu de données de données $D$ représenté avec les coordonnées parallèles [Inselberg et Dimsdale, 1990].

Pour initialiser la population $P$, notre système s'appuie sur le vecteur de poids $U = (u_1, u_2, ..., u_k)$ des attributs de données $A_i$ qui caractérise le paramétrage de la visualisation $V_i$, sélectionnée lors de la phase de suggestion (voir chapitre 6). En modifiant ce vecteur avec un opérateur de mutation (voir section 7.4.4), 8 nouveaux vecteurs d'importances

sont générés $I_{i=1..8}$. Pour chaque vecteur, un appariement basé sur les nouvelles valeurs d'importance est effectué entre les attributs de données $A_i$ et les attributs visuels $A_{ij}$. Ces 8 nouveaux appariements représentent les 8 individus de $P$, donc 8 nouveaux paramétrages de la visualisation $V_i$ sont créés. Ensuite, tous ces paramétrages sont appliqués sur le jeu de données $D$ et affichés sur une seule interface (voir figure 7.2) pour être évalués visuellement par l'utilisateur.

### 7.4.2 Évaluation/Sélection

L'évaluation des individus de $P(t)$ dans notre AGI se fait de manière visuelle et interactive. En effet, à l'issue de la phase d'initialisation de notre algorithme, tous les individus de $P(t)$ générés sont affichés sur une seule interface (voir figure 7.2). L'utilisateur peut ainsi, selon ses préférences visuelles individuelles, évaluer la qualité de chaque appariement généré par une simple comparaison visuelle. Pour des raisons ergonomiques nous avons réservé sur la seconde interface de l'assistant une partie assez importante pour visualiser les individus dans une vue plus large et plus facile à explorer. Pour bien exploiter cette fonctionnalité, et pour une meilleure évaluation de $P$, un seul clic sur n'importe quel individu de $P$ permet de l'afficher directement dans l'emplacement réservé (voir figures 7.2, 7.3, 7.4). De plus, l'interface proposée par VizAssist permet à l'utilisateur de bénéficier de différentes interactions (zoom, sélection d'individu, brushing, etc.). Les figures 7.3 et 7.4 illustrent ces interactions.

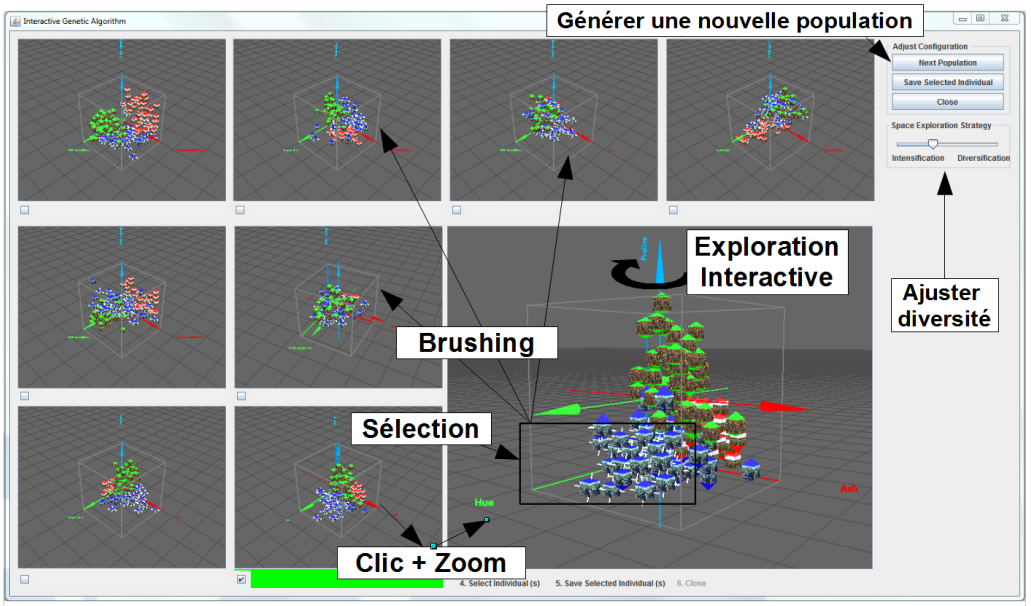



FIGURE 7.3 – Seconde interface de VizAssist. Plusieurs interactions sont proposées aux utilisateurs afin de faciliter le processus d'optimisation génétique du paramétrage d'une visualisation avec l'AGI.

Dans notre AGI, la sélection des individus dans une population $P(t)$ dépend du degré d'adéquation du paramétrage qu'ils traduisent par rapport aux préférences utilisateur. En

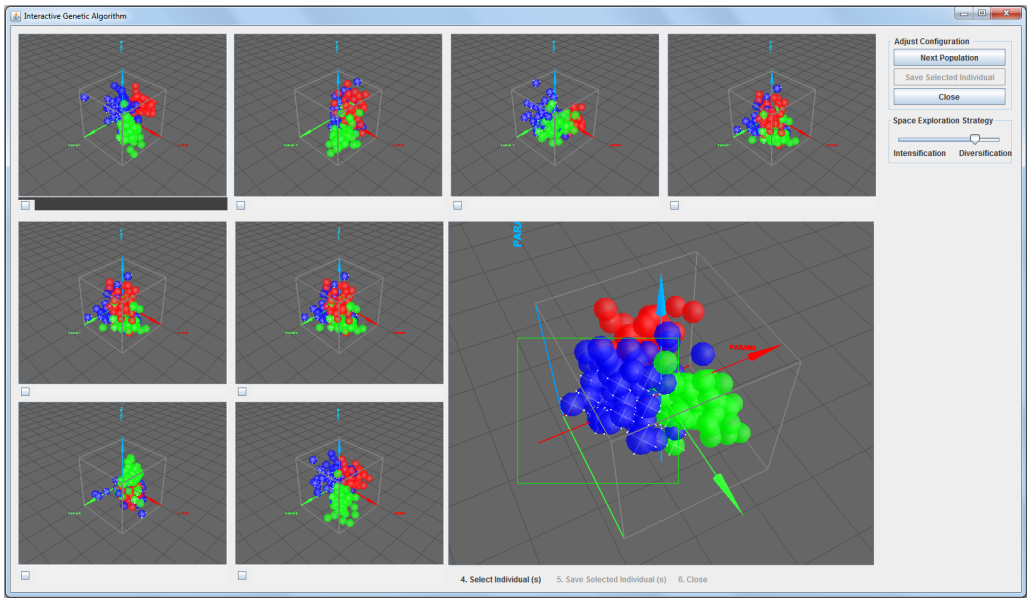



FIGURE 7.4 – Seconde interface de VizAssist : affichage d'un individu $I$ de $P$ dans une vue plus grande sur la partie inférieure à droite de l'interface, par un simple clic.

effet, deux modes de sélection d'individus sont offerts aux utilisateurs. Le premier mode concerne une sélection classique qui consiste à retenir des individus afin de leur appliquer un croisement et une mutation pour générer une nouvelle population $P(t+1)$. Par contre, le second mode consiste à sélectionner des individus pour une utilisation ultérieure du paramétrage qu'elles proposent. Dans ce mode de sélection, si l'utilisateur juge par exemple qu'un appariement lui est pertinent pour l'accomplissement d'une tâche ultérieure mais pas pour celle qu'il est en train d'exécuter, 1) il peut sélectionner l'individu désiré, 2) le sauvegarder, 3) puis le dé-sélectionner. À chaque itération de l'AGI, l'utilisateur peut sélectionner de 1 à 7 individus dans $P(t)$ par un simple clic.

### 7.4.3 Croisement

À l'issue de l'étape d'évaluation des individus de notre AGI, l'utilisateur peut retenir de 1 à 7 individus de $P(t)$. Dans la nouvelle population $P(t+1)$, les individus non sélectionnés sont les seuls à être remplacés. La génération des nouveaux individus nécessite l'exécution successive d'une étape de croisement puis de mutation si le nombre d'individus sélectionnés à l'instant $t$ est supérieur ou égal à 2. Dans le cas contraire, l'étape de mutation vient directement après l'étape de sélection (moins de 2 individus sélectionnés). En effet, si le nombre d'individus sélectionnés est supérieur ou égal à 2, seulement deux sont choisis pour l'étape de croisement. Ce choix est opéré de manière aléatoire et uniforme. Ensuite, nous leur appliquons un croisement uniforme. Le principe de ce dernier (croisement uniforme) consiste à permuter, avec une probabilité de 50%, les gènes des deux individus retenus (voir figure 7.5). Dans notre AGI, un seul descendant (fils) est conservé à l'issue de l'étape de croisement. En s'appuyant sur ce dernier, les individus à créer dans $P(t+1)$ sont générés.

A noter que dans ce type de croisement le nombre de gènes à échanger entre deux individus n'est pas connu a priori. Pour notre problème, l'utilisation de ce type de croisement a permis d'apporter plus de diversité dans les appariements proposés à chaque étape, contrairement aux deux croisements : 1-point et n-points.

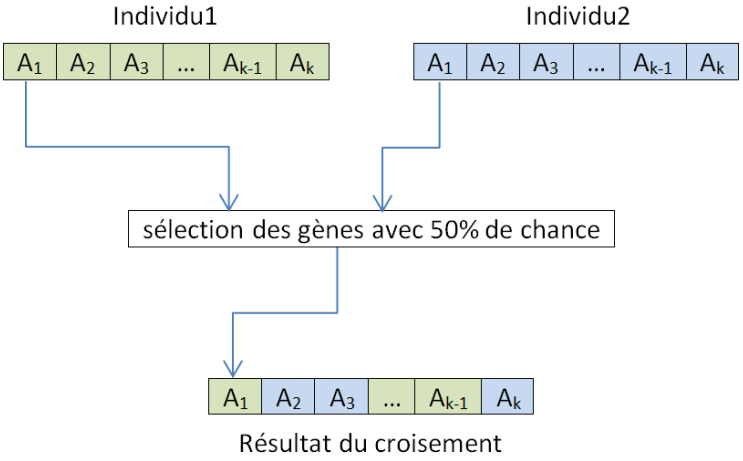



FIGURE 7.5 – Principe du croisement uniforme utilisé dans notre AGI.

### 7.4.4 Mutation

Ayant choisi une représentation génétique qui s'appuie sur un codage réel, pour assurer plus de diversité dans la population $P$ à chaque itération de l'AGI, nous avons opté pour un opérateur de mutation multi-uniforme. L'application de cet opérateur dans notre cas a pour apport principal de permettre à l'utilisateur d'explorer le plus de paramétrages possibles. Son principe consiste à décider selon une probabilité $P_{mutation}$ si on ajoute ou non à chaque gène de $I$ de $P$ un bruit $\delta$. Dans notre AGI, la probabilité de mutation $P_{mutation}$ n'est pas fixe et varie dans l'intervalle $[P_{min}, P_{max}]$. Pour la fixer nous utilisons la fonction suivante :

$$P_{mutation}(i) = \beta(i) \times P_{min} + (1 - \beta(i)) \times P_{max}$$

Où :

$\beta(i) = 1 - \frac{i-1}{n'-1}$

$i$ : l'indice du $i^{i\grave{e}me}$ individu à générer.

$n'$ : le nombre d'individus à générer.

L'utilisateur peut augmenter ou diminuer la diversité des individus en fonction de ce qu'il observe, sans avoir à connaître les détails de l'algorithme génétique et sans avoir une quelconque expertise dans ce domaine. Simplement, il dispose d'une réglette (voir figure

7.3) pour augmenter ou diminuer une valeur appelée *"diversification"* dans l'interface, et en fonction de la valeur fixée, nous mettons à jour la valeur de $\delta$ dans l'opérateur de mutation. Dans notre AGI, le bruit $\delta$ est généré en s'appuyant sur la fonction suivante :

$$\delta = (r - 0.5) \times 2 \times diversification$$

Où :

$r$ : un nombre réel aléatoirement choisi dans l'intervalle $[0, 1]$.

$diversification$ : un nombre réel aléatoirement choisi dans l'intervalle $[-20, +20]$.

En résumé, la formule que nous avons définie pour calculer la valeur $P_{mutation}$ a pour intérêt principal d'assurer plus de diversité dans les paramétrages (appariement entre attribut de données et attributs visuels) proposés à chaque itération de notre AGI. Par ailleurs, comme nous nous appuyons dans notre processus d'appariement sur une heuristique fondée sur une opération de tri des importances et que la mise en correspondance entre les attributs $A_i$ et $A_{ij}$ dépend de ces valeurs, l'utilisation de la fonction $\delta$ a pour avantage d'assurer une variance des importances dans l'intervalle $[0, 100]$. Cela signifie que si les données utilisateurs sont définies par un nombre important d'attributs de données, le bruit $\delta$ généré à chaque instant $t+1$ permet d'augmenter la chance que de nouveaux attributs de données $A_i$ (ayant une faible importance initialement) soient choisis dans le processus d'appariement des nouvelles visualisations générées. Ainsi, nous traitons dans notre approche les problèmes que peuvent rencontrer des utilisateurs de VizAssist liés à l'utilisation de l'AGI.

## 7.5 Exemple illustratif du déroulement de l'AGI

Dans le but de présenter le mode de fonctionnement de l'AGI et d'expliquer la seconde interface de l'assistant servant à le dérouler, nous présentons dans ce qui suit un exemple d'utilisation de notre algorithme pour accomplir une tâche d'optimisation de paramétrage. Nous rappelons que cette dernière consiste à trouver le meilleur appariement entre des attributs de données d'une base de données artificielle $D$ avec les attributs visuels d'une visualisation $V_i$ pour accomplir une tâche $T$.

1. **Tâche** (T) :
   Dans le but de présenter un exemple du déroulement de l'AGI sur la seconde interface de l'assistant, nous avons défini une tâche $T$. Cette tâche a pour objectif d'obtenir un paramétrage d'une visualisation $V_i$ qui permet de distinguer 4 clusters dans les données à représenter (voir figure 7.6). Le but est donc de tester avec l'AGI plusieurs combinaisons entre les attributs de données d'une base de données $D$ et les attributs visuels d'une visualisation $V_i$. Sachant que la visualisation $V_i$ à utiliser pour accomplir $T$ est représentée en 2D, distinguer les 4 clusters, revient à trouver 2 attributs de $D$ à positionner sur les deux axes X et Y de la visualisation choisie et dont la combinaison sépare les données en 4 groupes.

2. **Base de données** $(D)$ :
   Afin d'accomplir la tâche $T$, nous avons généré une base de données artificielle $BD$ à l'aide du logiciel *Microsoft Excel*. La $BD$ comporte 200 données. Chaque donnée est caractérisée par 30 attributs numériques. Parmi les 30 attributs, 2 attributs seulement donnent la meilleure combinaison permettant de générer la visualisation dans laquelle on distingue les 4 clusters à obtenir. 2 sont générés à partir des meilleurs attributs mais avec un bruit croissant et le reste des attributs (les 26 autres attributs) représentent un bruit généré de manière aléatoire.
3. **Visualisation choisie** $(V_i)$ :
   Nous avons choisi comme visualisation pour accomplir la tâche $T$ la représentation graphique *"Visages de Chernoff"* (voir figure 7.6) [Chernoff, 1973]. Dans notre base de connaissance, cette visualisation est décrite par 13 attributs visuels (AxeX, AxeY, forme visage, forme yeux, taille pupilles, etc.). Avec un outil fondé sur un paramétrage manuel, trouver le paramétrage exact (réponse exacte pour accomplir $T$) entre $D$ et $V_i$ peut nécessiter de tester toutes les combinaisons possibles entre les 30 attributs de données et les 2 attributs visuels X et Y. En réalité, cela peut consister à faire $C_{30}^2$ (435) tests.

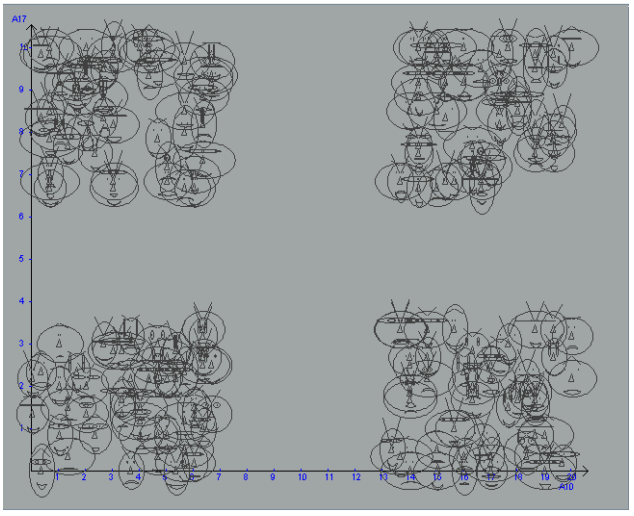


FIGURE 7.6 – Aperçu de la visualisation à obtenir avec le paramétrage distinguant les 4 clusters.

4. **Description des étapes d'accomplissement de** $T$
   ***Initialisation de*** $P$ ***:*** après avoir choisi la visualisation *"Visages de Chernoff"* sur la première interface de l'assistant, l'utilisateur clique sur le bouton *"Adjust this Matching"* (voir figure 6.2) pour lancer la seconde interface. À partir du vecteur de poids initial $U$ des attributs de données $A_i$ du jeu de données $D$, 8 nouveaux vecteurs (8 individus de $P$) sont créés. Ces derniers sont ensuite utilisés pour générer 8 nouveaux appariements de la même visualisation choisie lors de la première étape (*"Visages de Chernoff"*). Les résultats de cette étape sont affichés sur la même interface (voir figure 7.7) à l'utilisateur pour être évalués.

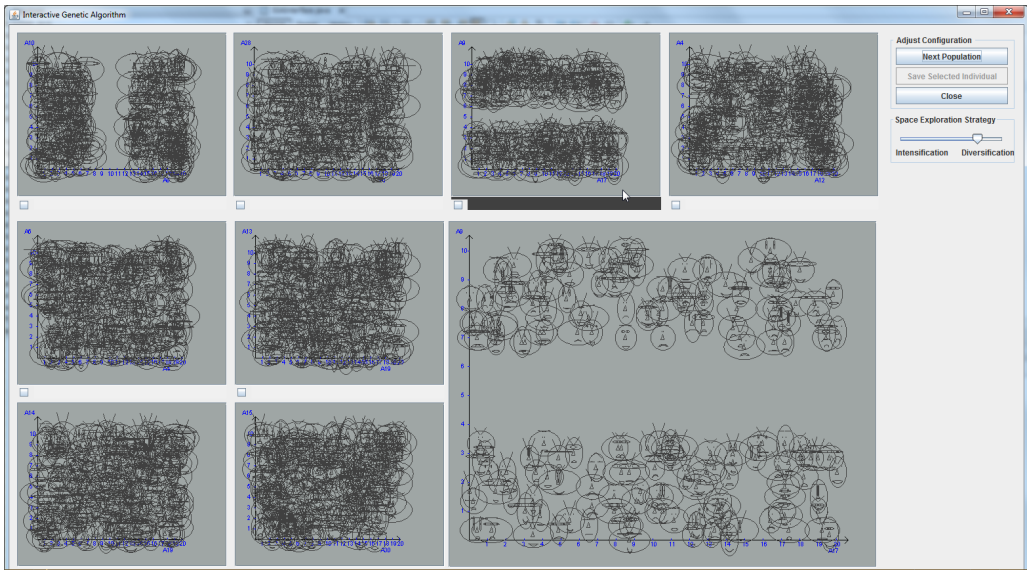



FIGURE 7.7 – Initialisation de la population $P$. Les individus obtenus à l'instant $t = 1$ affichés sur la seconde interface de l'assistant appliqués au jeu de données $D$.

***Évaluation visuelle et sélection interactive des individus :*** généralement les aperçus des visualisations 2D affichées sur la deuxième interface peuvent suffire à l'utilisateur pour faire une évaluation visuelle rapide de la population visualisée et choisir les visualisations qui l'intéressent. Cela est dû principalement au nombre limité des individus qui définissent $P$. Par contre, pour des visualisations en 3D, un survol rapide de la seconde interface n'est pas suffisant et l'utilisateur est obligé de faire une exploration plus détaillée. Pour cette raison, nous avons réservé une zone assez importante pour représenter les individus dans un format plus large et facile à percevoir. Un simple clic sur l'individu à explorer en détail suffit pour l'afficher sur cette zone (voir itération 3 sur la figure 7.8). Après avoir évalué $P(t)$, l'utilisateur peut sélectionner les individus qui lui semblent donner le résultat recherché (voir itération 2 sur la figure 7.8). Quand l'utilisateur sélectionne tous les individus qu'il désire retenir dans $P(t)$, il clique sur le bouton *"Next Population"* pour générer $P(t+1)$.

***Croisement/Mutation :*** Quand l'utilisateur clique sur le bouton *"Next Population"*, l'AGI génère une nouvelle population $P(t+1)$. Dans $P(t+1)$, seuls les individus non sélectionnés dans $P(t)$ sont remplacés. Dans la figure 7.8, on peut voir que les deux individus sélectionnés dans l'itération 3 sont retenus dans l'itération 4. Les 6 nouveaux paramétrages de la visualisation *"Visages de Chernoff"* affichés dans l'interface de la quatrième itération (voir figure 7.8) sont les résultats de l'exécution d'un croisement puis d'une mutation appliqués sur les deux individus sélectionnés dans la troisième itération. Si après quelques itérations, la combinaison d'individus, sélectionnés pour leur appliquer les deux opérateurs croisement et mutations, ne donne pas de résultats satisfaisants (une stagnation dans les paramétrages proposés), l'utilisateur peut soit 1) dé-sélectionner ceux qu'il juge avoir peu d'impact sur le résultat

à atteindre (voir un exemple entre les itérations 4 et 5 dans la figure 7.8), 2) utiliser la réglette réservée pour augmenter ou diminuer la variable de *"diversification"* (voir section 7.4.4).

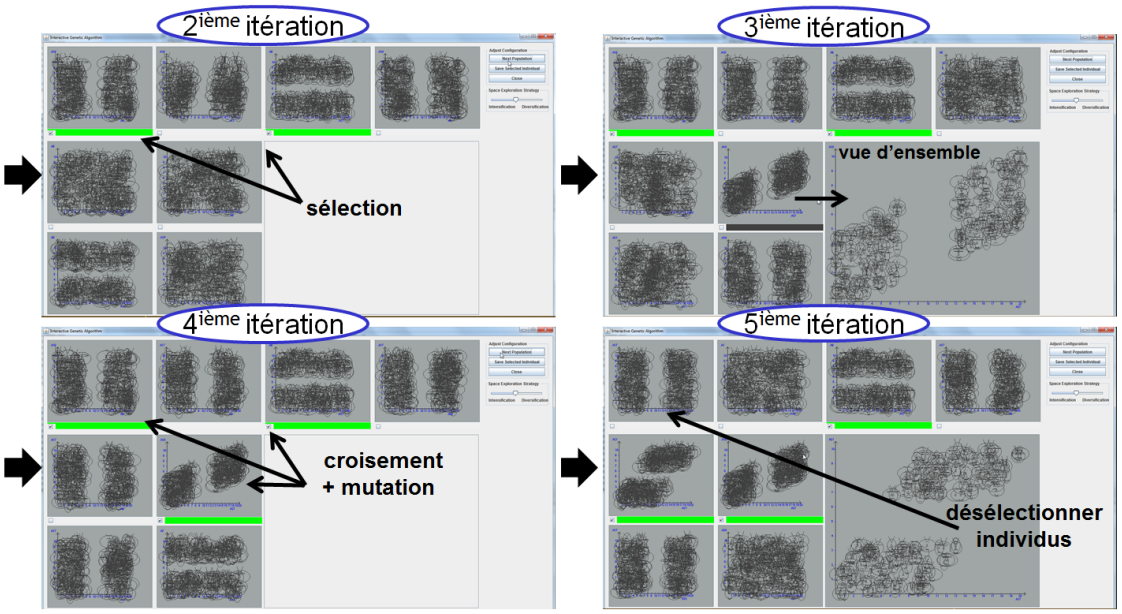



FIGURE 7.8 – Aperçu des résultats obtenus dans 4 itérations successives avec illustration des interactions possibles fournies par la seconde interface de VizAssist.

5. **Résultat :**
   Les deux dernières techniques proposées pour éviter une éventuelle stagnation dans les paramétrages proposés par l'AGI permettent de 1) simplifier l'exploration d'un grand espace de recherche, 2) réduire le temps pour converger vers des solutions optimales. Dans notre exemple, le résultat à trouver pour accomplir $T$ est atteint après seulement 9 itérations et au bout de 48 secondes (voir figure 7.9). Cela met en avant les avantages d'utilisation de notre AGI ainsi que la simplicité de la seconde interface de l'assistant, dédiée à son déroulement pour accomplir $T$. Plus de résultats sont présentés dans le chapitre 8 et 9.

## 7.6 Conclusion

Nous avons décrit dans ce chapitre le module de paramétrage interactif de VizAssist. Ce dernier s'appuie sur un algorithme génétique interactif dont l'objectif principal est d'aider les utilisateurs d'un système de fouille visuelle de données à affiner et optimiser un paramétrage initial entre les attributs de données d'un jeu de données $D$ et les attributs visuels d'une visualisation $V_i$. Nous avons détaillé aussi l'interface dédiée à son déroulement, à travers un exemple d'application, visant à détecter des clusters dans un jeu de données. Cet exemple, nous a permis de mettre en avant quelques avantages d'utilisation de VizAssist pour des utilisateurs novices. Pour plus de détails sur les autres avantages de l'approche que

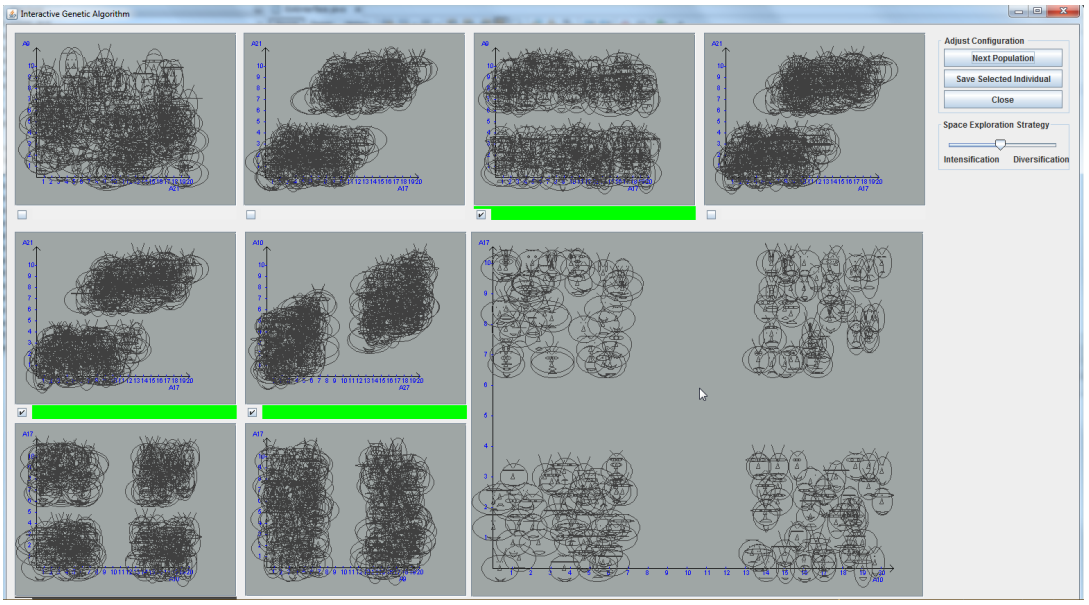



FIGURE 7.9 – Aperçu de la dernière population contenant la visualisation ayant le paramétrage distinguant les 4 clusters. Le résultat est obtenu au bout de 9 itérations. Le test a été accompli en 48 secondes.

nous proposons, plusieurs cas d'études sont proposés dans le chapitre 9. Nous présentons dans la partie suivante du manuscrit plus de détails sur les expérimentations menées pour évaluer VizAssist.

# Troisième partie

# Expérimentations

# Chapitre 8

# Évaluation Utilisateur

**Résumé** : Dans le but d'expérimenter notre système, nous avons réalisé une évaluation utilisateur dans laquelle nous avons comparé notre assistant à un autre système de visualisation appelé VRMiner [Azzag *et al.*, 2005]. Nous présentons dans ce chapitre le protocole expérimental que nous avons défini pour mener cette évaluation et les résultats obtenus. Notons que nous nous sommes appuyés sur deux tâches pour comparer les deux systèmes et que 27 personnes ont participé aux tests. De plus, nous avons collecté durant les expérimentations différents aspects subjectifs ressentis par les participants.

## 8.1 Introduction

Dans le but d'expérimenter notre système nous avons réalisé une évaluation utilisateur dans laquelle nous avons comparé notre assistant à un autre système de visualisation appelé VRMiner [Azzag *et al.*, 2005], utilisant les mêmes visualisations mais avec un paramétrage manuel. L'évaluation utilisateur a pour objectif de : 1) comparer la fluidité et la durée des tâches de configuration et d'exploration dans les deux systèmes ; 2) identifier lequel des deux systèmes satisfait le mieux les objectifs des participants ; 3) identifier le ressenti des participants face à deux types de paramétrage : un paramétrage visuel et interactif, et un paramétrage manuel.


FIGURE 8.1 – Interface de paramétrage manuel du système VRMiner [Azzag *et al.*, 2005].

Nous avons défini un protocole expérimental structuré en six étapes que nous décrivons dans la section 8.2. Nous avons utilisé durant les expérimentations un questionnaire qui regroupe trois parties. La première partie comporte des questions sur : le profil utilisateur (age, sexe, niveau d'étude), son niveau dans le domaine de la visualisation d'informations (utilisation des graphiques, maîtrise des applications de visualisation, etc.) et son niveau dans un environnement 3D (voir tableau 8.1). La deuxième partie contient la procédure du déroulement du processus de réalisation des deux tâches (objectif de la tâche, système utilisé, bases de données utilisées, réponses obtenues et la durée du test) que nous avons fixées pour comparer les deux systèmes (voir section 8.2.2). Quant à la troisième partie, elle englobe des questions sur les différents aspects subjectifs ressentis par les participants durant les tests avec les deux systèmes (voir tableau 8.7).

Pour réaliser ces deux tâches, nous avons généré deux catégories de bases de données différentes que nous décrivons en détail dans la section 8.2.3. Les deux systèmes ont utilisé rigoureusement les mêmes informations. Notre protocole a été testé sur 27 participants.

## 8.2 Protocole expérimental

### 8.2.1 Participants

Le tableau 8.1 résume les différentes informations recueillies lors de l'administration du questionnaire sur le profil des utilisateurs, leurs niveaux dans le domaine de la visualisation d'informations ainsi que dans un environnement 3D. En effet, notre protocole a porté sur 27 participants (6 femmes et 21 hommes) dont l'âge varie de 22 ans à 40 ans. Leur niveau d'étude est supérieur à Bac+2 dans le domaine de l'Informatique. Toutes les personnes ayant participé à notre évaluation connaissent ce qu'est une base de données multidimensionnelle et ont au moins utilisé auparavant une représentation graphique (2D et/ou 3D). Plus de la moitié des participants ont déclaré avoir déjà utilisé un nuage de points comme représentation graphique pour visualiser leurs données. 25 des 27 participants ont l'habitude d'utiliser *Microsoft Excel* (qui s'appuie sur un paramétrage manuel pour configurer les visualisations). Ils utilisent cet outil avec un niveau de maîtrise moyen de l'ordre de $3.40 \pm 0.82$ sur une échelle de 1 à 5 [Likert, 1932]. Par ailleurs, l'évaluation utilisateur montre que les principaux objectifs justifiant le recours de l'ensemble des participants à l'utilisation de la visualisation d'informations auparavant sont : la présentation de leurs résultats, l'obtention d'une vue d'ensemble des données et l'extraction des connaissances à partir d'un jeu de données.

La réalisation d'une expérimentation durait en moyenne une heure pour chaque participant. Toutes les expérimentations ont eu lieu au sein de notre laboratoire sur une machine équipée d'outils de navigation 3D. Les séances de test se sont déroulées de la façon suivante :

1. accueil des participants,
2. administration du questionnaire de caractérisation des participants (profils, niveau d'étude, niveau en base de données multidimensionnelles, usage d'une application pour la représentation graphique en précisant le niveau de maîtrise, niveau dans un environnement 3D, etc.),
3. illustration du scénario d'utilisation des deux systèmes pour le choix et le paramétrage des visualisations avec deux exemples de bases de données multidimensionnelles,
4. définition de quelques tâches de familiarisation avec les deux systèmes, puis orientation des participants pour les accomplir,
5. explication du protocole de test,
6. début de l'expérimentation.

Le questionnaire final qui a servi à dérouler les expérimentations de notre évaluation utilisateur est détaillé dans l'annexe A.

| | |
|---|---|
| **Nombre de participants** | **27 personnes :** 6 femmes et 21 hommes |
| **Profils** | **Age :** 22 ans à 40 ans |
| | **Niveau d'étude :** Bac+2 à >Bac+5 en informatique |
| | **Connaissez vous ce qu'est une BD multi-D :** Oui : 27/27 \| Non : 0/27 |
| **Niveau en visualisation d'informations** | **Avoir déjà utilisé des graphiques :** Oui : 27/27 \| Non : 0/27 |
| | **Graphiques déjà utilisés :** nuages de points, histogrammes, courbes et camembert |
| | **Objectifs habituels :** présentation des résultats, obtention d'une vue d'ensemble des données et extraction des connaissances |
| | **Outils de visualisation fréquemment utilisés :** Microsoft Excel (25/27) et Matlab (15/27) |
| | **Fréquences d'utilisation des outils de visualisation :** 1 fois par mois pour 33% des participants et moins pour le reste des participants |
| **Les caractéristiques importantes pour les participants pour le choix d'une visualisation ordonnées selon leurs priorités** | La facilité d'analyse et d'interprétation des données<br>La présentation des résultats<br>La dimension visuelle (1D, 2D, 3D, etc.)<br>Les interactions et les tâches d'exploration |
| **Les avantages les plus pertinents dans le choix d'un outil de visualisation pour les participants** | Facilité du processus de paramétrage ($4.35 \pm 1.05$)<br>Temps nécessaire pour générer une visualisation ($3.87 \pm 1.12$)<br>Nombre de visualisations proposées ($3.81 \pm 1.05$) |

TABLE 8.1 – Informations recueillies lors de l'administration du questionnaire sur le profil des utilisateurs, leurs niveaux dans le domaine de la visualisation d'informations ainsi que dans un environnement 3D. Les scores utilisés dans la dernière ligne du tableau sont définis sur une échelle de 1 à 5 [Likert, 1932].

### 8.2.2 Tâches

Comme nous l'avons déjà indiqué ci-dessus, nous avons fixé deux tâches T1 et T2 pour évaluer les avantages et inconvénients de VizAssist par rapport au système VRMiner. Ces deux tâches permettent de mesurer les différents aspects sur lesquels un système de fouille visuelle de données doit se fonder (voir section 1.3.4). Nous avons défini deux principales caractéristiques à comparer entre les deux outils (VizAssist et VRMiner). Dans un premier temps, nous mesurons la durée d'une session utilisateur pour obtenir une visualisation ayant un paramétrage prédéfini en s'appuyant sur deux types d'interfaces différentes : interface avec paramétrage manuel (VRMiner) et interface avec paramétrage automatique (VizAssist). Ensuite, nous évaluons la qualité des réponses obtenues par les participants dans la réalisation d'une tâche de classification, de manière manuelle avec VRMiner, et de manière visuelle et interactive avec VizAssist. Pour chacune des deux tâches, les participants ont répété le même test 3 fois. Notre choix (des 3 répétitions) est motivé par notre intérêt à comparer l'évolution des résultats obtenus par les participants, dans les 3 tests pour chaque tâche, afin d'étudier la stabilité d'utilisation des deux outils, et d'obtenir un effectif de tests suffisant pour la comparaison statistique des résultats.

La tâche T1 a pour but de tester la première interface de VizAssist (voir figure 4.2) et consiste à obtenir une visualisation fixée d'avance. Avant de commencer le test, on fournit à chaque participant la description du paramétrage prédéfini à obtenir. Pour le paramétrage à trouver, on indique à l'utilisateur, qu'on souhaite représenter 3 attributs numériques, un

attribut image et un attribut texte. On lui indique aussi qu'il doit obtenir une visualisation représentant l'ensemble des données regroupées en 3 classes (voir figure 8.2).

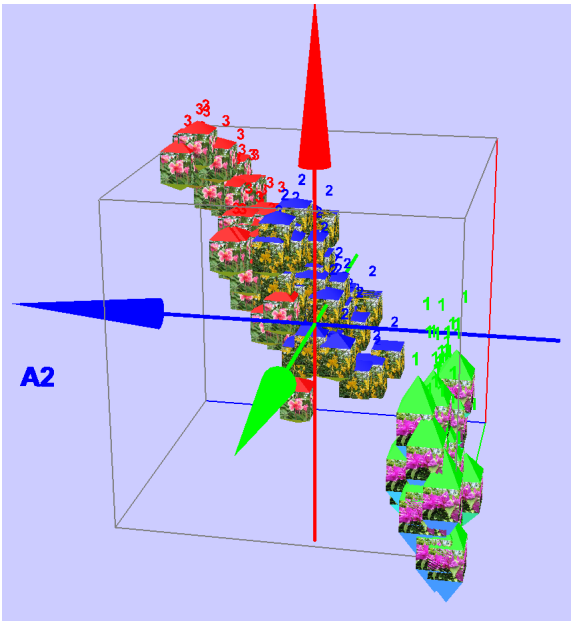



FIGURE 8.2 – Aperçu du résultat à obtenir à l'issue de la réalisation de T1. On peut distinguer dans cette figure que chaque élément graphique de la visualisation est caractérisé par sa position dans le repère 3D, le numéro de sa classe d'appartenance représenté avec du texte au-dessus, une image et une couleur reflétant sa classe.

La tâche T2 a pour but de tester la seconde interface de VizAssist (voir figure 4.3) et consiste à obtenir une visualisation qui soit la moins bruitée possible et dans laquelle 8 classes se distinguent. Cette tâche a donc pour objectif de tester l'AGI et d'améliorer un paramétrage initial proposé par VizAssist dans sa première interface. Avant de commencer le test, on montre à chaque participant un aperçu de la visualisation la moins bruitée possible dans laquelle on distingue les 8 classes représentant la meilleure classification des données à visualiser (voir figures 8.3 et 8.4). Sachant que toutes les visualisations utilisées par les deux systèmes pour réaliser cette seconde tâche sont représentées en 3D, on indique à l'utilisateur qu'on souhaite trouver la meilleure combinaison des 3 axes (axe X, axe Y et axe Z) donnant le meilleur paramétrage.

### 8.2.3 Bases de données

Dans le but de réaliser les deux tâches que nous avons définies dans notre protocole, nous avons généré 12 bases de données spécifiques regroupées en deux catégories selon la méthode de leur création (voir tableau 8.2). En effet, la première catégorie comporte les bases de données de BD1 à BD6 et elles sont destinées à être utilisées pour réaliser la tâche T1. La seconde catégorie contient les bases de données de BD7 à BD12 qui sont utilisées pour accomplir la tâche T2. Toutes les bases de données utilisées dans les tests ont été générées avec le logiciel *Microsoft Excel*. Pour accomplir chaque tâche, nous avons utilisé

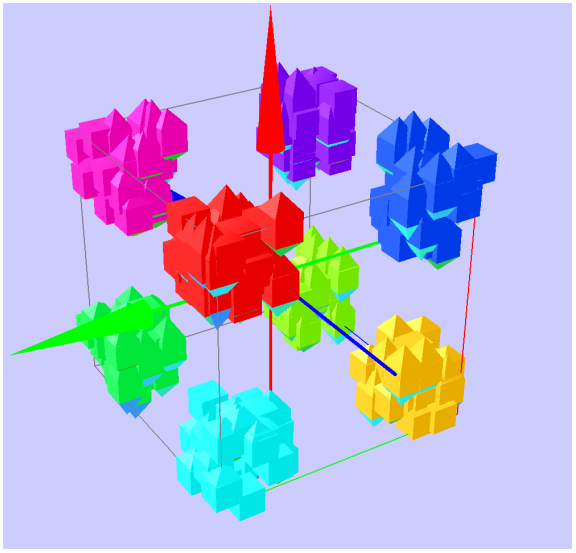


FIGURE 8.3 – Aperçu du résultat à obtenir à l'issue de la réalisation de T2 dans lequel on peut distinguer 8 classes séparées.

les mêmes bases de données pour les deux systèmes mais avec un renommage des attributs de données et un changement d'ordre afin d'éviter un effet d'apprentissage. L'ordre des tests a été effectué de manière aléatoire.

Les bases de données de la première catégorie sont caractérisées par 150 données dont chacune est décrite par 5 attributs numériques, 1 attribut image, 1 attribut texte et 1 attribut classe. Pour les bases de données de la seconde catégorie, elles comportent 240 données. Chaque donnée est caractérisée par 30 attributs numériques et 1 attribut classe. Parmi les 30 attributs de données numériques, 3 attributs seulement donnent la meilleure combinaison permettant de générer une visualisation dans laquelle on distingue les 8 classes à obtenir. 3 sont générés à partir des meilleurs attributs mais avec un bruit croissant et le reste des attributs (les 24 autres attributs) représentent un bruit généré de manière aléatoire et croissante aussi.

| Tâches | Tests | Bases de données | Nombre d'attributs | Description des attributs |
|---|---|---|---|---|
| Tâche 1 | Test 1<br>Test 2<br>Test 3 | BD1, BD2<br>BD3, BD4<br>BD5, BD6 | 7 | 5 att. numériques<br>1 att. image, 1 att. texte<br>1 att. classe |
| Tâche 2 | Test 1<br>Test 2<br>Test 3 | BD7, BD8<br>BD9, BD10<br>BD11, BD12 | 31 | 30 att. numériques<br>1 att. classe |

TABLE 8.2 – Description des données des deux catégories de base de données utilisées dans l'évaluation utilisateur et le protocole de choix de ces dernières pour chaque tâche et pour chaque test de l'experimentation.

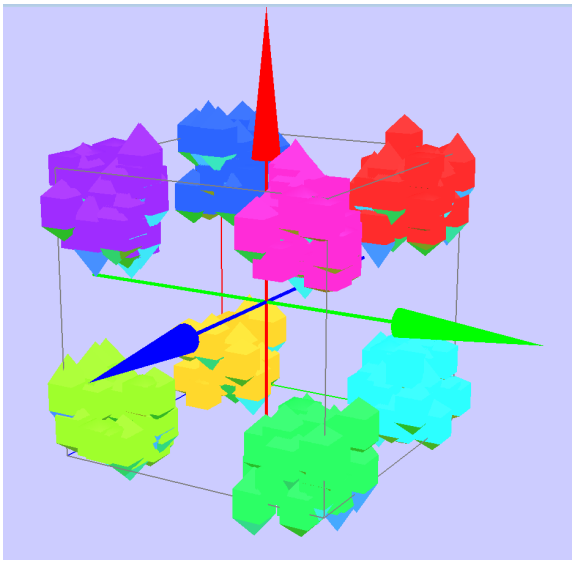


FIGURE 8.4 – Aperçu de la même visualisation que dans la figure 8.3 mais prise du côté opposé.

### 8.2.4 Résultats

1. **Tâche 1 :**
   Les résultats obtenus pour la tâche T1 montrent que tous les utilisateurs ont résolu cette tâche avec les deux systèmes (voir tableau 8.3). Cependant, la durée passée par les participants sur T1 avec VizAssist est 21.93 ± 11.34 secondes, tandis qu'avec VRMiner cette durée est de 54.25 ± 23.64 secondes. Nous constatons ainsi que les temps pour répondre à T1 sont très nettement en faveur de VizAssist. Pour appuyer notre conclusion sur la rapidité de VizAssist pour réaliser T1, nous avons réalisé une étude statistique (ANOVA). Les résultats obtenus ont démontré que le temps pour accomplir la tâche T1 est statistiquement différent entre les deux outils. De plus, si on se réfère au tableau 8.1, on peut constater que même si la majorité des participants ont l'habitude d'utiliser un système de visualisation d'informations (*Microsoft Excel ou Matlab*) qui s'appuie sur un paramétrage manuel et ont un bon niveau de maîtrise de cette application, VizAssist leur a permis d'accomplir T1 (une tâche de paramétrage) plus rapidement, plus facilement et sans aucune pré-connaissance de l'outil. Nous pouvons donc conclure que l'interface de paramétrage que propose VizAssist est très avantageuse et en adéquation avec les propriétés que devrait garantir un système de fouille visuelle de données pour des utilisateurs novices.
2. **Tâche 2 :**
   Pour la tâche T2, nous avons défini une mesure de qualité $Q$ pour évaluer graduellement les réponses obtenues par les participants avec les deux outils. Cette mesure de qualité permet de donner un score à chaque attribut, ce score étant inversement proportionnel au bruit que nous y avons rajouté. Si les attributs non bruités ont été trouvés, la qualité est de 2, et si les attributs les plus bruités ont été choisis, la qualité est de 0. Par ailleurs, dans le but de mesurer l'impact du temps nécessaire pour ac-

| Tests | Système | Moyenne ± Écart type (secondes) | P-valeur | Abandons/total |
|---|---|---|---|---|
| Test1 | VizAssist | **29.74 ± 16.38** | | 0/27 |
| | VRMiner | 73.04 ± 23.71 | | 0/27 |
| Test2 | VizAssist | **18.52 ± 3.76** | | 0/27 |
| | VRMiner | 48.48 ± 15.49 | | 0/27 |
| Test3 | VizAssist | **17.52 ± 4.20** | | 0/27 |
| | VRMiner | 41.22 ± 18.43 | | 0/27 |
| Total | VizAssist | **21.93 ± 11.34 *** | $< 2e^{-16}$ | 0/81 |
| | VRMiner | 54.25 ± 23.64 | | 0/81 |

TABLE 8.3 – Indicateur statistique de la **distibution du temps** pour la tâche T1 selon le numéro de test, avec VRMiner et VizAssist. La réponse en gras et avec * signifie que ce résultat est statistiquement significatif entre les deux outils.

complir la tâche T2 sur la qualité des réponses obtenues avec VRMiner et VizAssist, nous avons réalisé T2 selon deux protocoles différents. Dans le premier protocole, nous avons fixé comme contrainte une limitation de la durée des tests à 3 minutes, tandis que pour le second protocole les tests ont été accomplis sans limite de temps. Les résultats sont illustrés dans les deux tableaux 8.4 et 8.5. Dans le tableau 8.6 nous présentons une synthèse des durées des tests obtenues en réalisant la tâche T2 avec VRMiner et VizAssist dans le cas où les participants n'ont aucune contrainte sur la durée des tests.

| Tests | Système | Moyenne ± Écart type | P-valeur | Abandons/total |
|---|---|---|---|---|
| Test1 | VizAssist | **0.69 ± 0.21** | | 0/27 |
| | VRMiner | 0.44 ± 0.29 | | 0/27 |
| Test2 | VizAssist | **0.72 ± 0.15** | | 0/27 |
| | VRMiner | 0.41 ± 0.31 | | 0/27 |
| Test3 | VizAssist | **0.79 ± 0.13** | | 0/27 |
| | VRMiner | 0.43 ± 0.32 | | 0/27 |
| Total | VizAssist | **0.73 ± 0.17 *** | $< 9.29e^{-13}$ | 0/81 |
| | VRMiner | 0.43 ± 0.31 | | 0/81 |

TABLE 8.4 – Indicateur statistique de la **distribution de la qualité** des réponses pour la tâche T2 selon le numéro de test, avec VRMiner et VizAssist (la durée des tests est limitée à 3 minutes). La réponse en gras et avec * signifie que ce résultat est statistiquement significatif entre les deux outils.

Les valeurs de qualité obtenues, quand nous limitons la durée des tests à 3 minutes, sont de 0.73 ± 0.17 avec VizAssist, et de 0.43 ± 0.31 avec VRMiner. Et dans le cas contraire (sans limitation de la durée des tests), les valeurs de qualité obtenues sont de 0.76 ± 0.17 avec VizAssist, et de 0.50 ± 0.34 avec VRMiner. Donc, après avoir réalisé une analyse statistique (ANOVA) nous avons constaté que la qualité des réponses avec VizAssist est supérieure à celles obtenues avec VRMiner et la contrainte qu'on a fixée sur le temps de réalisation de la tâche T2 a un faible impact sur les

| Tests | Système | Moyenne ± Écart type | P-valeur | Abandons/total |
|---|---|---|---|---|
| Test1 | VizAssist | **0.76 ± 0.18** | | 0/27 |
| | VRMiner | 0.52 ± 0.30 | | 0/27 |
| Test2 | VizAssist | **0.72 ± 0.17** | | 0/27 |
| | VRMiner | 0.46 ± 0.35 | | 0/27 |
| Test3 | VizAssist | **0.81 ± 0.13** | | 0/27 |
| | VRMiner | 0.51 ± 0.38 | | 0/27 |
| Total | VizAssist | **0.76 ± 0.17 *** | $< 2.55e^{-09}$ | 0/81 |
| | VRMiner | 0.50 ± 0.34 | | 0/81 |

TABLE 8.5 – Indicateur statistique de la **distibution de la qualité** des réponses pour la tâche T2 selon le numéro de test, avec VRMiner et VizAssist (aucune limite de la durée des tests n'est fixée). La réponse en gras et avec * signifie que ce résultat est statistiquement significatif entre les deux outils.

résultats obtenus dans les 2 cas expérimentés. Nous avons remarqué que les participants trouvaient beaucoup de difficultés pour accomplir T2 avec VRMiner car ils réalisaient de nombreux essais d'appariement manuellement. Mais comme il s'agit de tester 30 variables sur trois axes (X, Y, Z) pour trouver la réponse correcte, après quelques tentatives d'appariements manuels, les participants optaient soit pour un choix aléatoire des attributs ou abandonnaient la réalisation de T2. En revanche, l'utilisation de l'AGI dans la deuxième interface de VizAssist a permis de faciliter et d'accélérer la convergence vers les meilleures réponses. D'ailleurs, les utilisateurs passaient à peu près la même durée pour réaliser T2 avec les deux systèmes (voir tableau 8.6) mais obtenaient des qualités de réponses différentes, ce qui est dû principalement à la simplicité du processus d'amélioration du paramétrage que propose VizAssist. Cela met en avant les capacités de VizAssist pour l'ajustement des paramètres, la sélection d'attributs et l'exploration des données.

| Tests | Système | Moyenne ± Écart type (secondes) | P-valeur | Abandons/total |
|---|---|---|---|---|
| Test1 | Assistant | 193.81 ± 94.45 | | 0/27 |
| | VRMiner | 213.59 ± 98.12 | | 0/27 |
| Test2 | Assistant | 184.41 ± 88.04 | | 0/27 |
| | VRMiner | 183.56 ± 88.43 | | 0/27 |
| Test3 | Assistant | 189.59 ± 91.09 | | 0/27 |
| | VRMiner | 173.22 ± 80.93 | | 0/27 |
| Total | Assistant | 189.27 ± 90.17 | 0.952 | 0/81 |
| | VRMiner | 190.12 ± 89.98 | | 0/81 |

TABLE 8.6 – Indicateur statistique de la **distibution du temps** pour la tâche T2 selon le numéro de test, avec VRMiner et VizAssist.

### 8.2.5 Bilan utilisateur

Pour évaluer les réponses des participants sur les questions subjectives à l'issue de la réalisation des deux tâches T1 et T2, nous avons opté pour un recueil d'informations utilisant l'échelle de Likert [Likert, 1932]. Le tableau 8.7 résume les observations que nous avons collectées sur l'utilisation des deux outils comparés.

Les résultats obtenus montrent que VizAssist est pour la majorité des participants un système simple à utiliser et permet de trouver les meilleurs paramétrages des visualisations. D'ailleurs, au moins 23 sur les 27 participants ont déclaré qu'ils choisiraient, pour une prochaine utilisation, VizAssist pour visualiser leurs données. La facilité d'utilisation des interfaces de VizAssist est un avantage très important pour les utilisateurs, et permet d'avoir un degré de fiabilité élevé pour accomplir les tâches demandées. Les participants à notre expérimentation jugent aussi que l'utilisation de l'AGI est très utile surtout qu'il leur fait éviter de faire plusieurs tests d'appariement manuellement pour trouver un paramétrage adéquat. Cependant, les utilisateurs ont déclaré que la vitesse de VizAssist en général et de génération des visualisations en particulier n'est pas très avantageuse par rapport au système VRMiner.

| Questions | Système | Moyenne ± Écart type | P-valeur |
|---|---|---|---|
| Simplicité d'utilisation | VizAssist | **4.52 ± 0.58 *** | $< 2.61e^{-11}$ |
| | VRMiner | 3.00 ± 0.73 | |
| Rapidité de génération des visualisations | VizAssist | 3.78 ± 1.01 | 0.532 |
| | VRMiner | 3.63 ± 0.69 | |
| Obtention du meilleur paramétrage de vos visualisations | VizAssist | **4.33 ± 0.78 *** | $< 1.27e^{-10}$ |
| | VRMiner | 2.56 ± 0.85 | |
| Quel système choisiriez-vous pour visualiser vos données ? | VizAssist | **4.37 ± 0.74 *** | $< 1.75e^{-07}$ |
| | VRMiner | 3.00 ± 0.92 | |
| Facilité d'utilisation des interfaces | VizAssist | **4.56 ± 0.64 *** | $< 1.9e^{-09}$ |
| | VRMiner | 2.74 ± 1.13 | |
| Fiabilité du système (permis d'accomplir les tâches demandées) | VizAssist | **4.26 ± 0.66 *** | $< 2.56e^{-05}$ |
| | VRMiner | 3.15 ± 1.06 | |
| Vitesse du logiciel | VizAssist | 3.52 ± 0.89 | 0.873 |
| | VRMiner | 3.48 ± 0.80 | |
| Comment trouvez-vous les deux systèmes (frustrant/satisfaisant) | VizAssist | **4.48 ± 0.80 *** | $< 1.38e^{-05}$ |
| | VRMiner | 3.15 ± 1.20 | |
| Comment trouvez-vous les deux systèmes (ennuyeux/stimulant) | VizAssist | **4.44 ± 0.75 *** | $< 2.38e^{-08}$ |
| | VRMiner | 2.89 ± 0.97 | |
| Comment trouvez-vous l'AGI | VizAssist | **4.63 ± 0.69** | |

TABLE 8.7 – Les réponses aux différentes questions subjectives proposées aux participants à l'issue de la réalisations des tâches T1 et T2 (les scores des réponses sont définis sur l'échelle de Likert [Likert, 1932]). Les réponses en gras et avec * signifient que cette question a donné des résultats statistiquement significatifs entre les deux outils.

## 8.3 Conclusion

Nous avons présenté dans ce chapitre les résultats de l'évaluation utilisateur que nous avons menée pour comparer VizAssist avec un système de visualisation classique fondé sur un paramétrage manuel. L'analyse des résultats obtenus montre que les interfaces que nous avons développées sont avantageuses pour des utilisateurs novices car elles n'exigent pas de connaissances a priori de leur part. Les utilisateurs ont montré aussi, de manière subjective, l'intérêt qu'ils ont porté pour l'utilisation ultérieure de notre système. En effet, ils considèrent que la capacité d'explorer des bases de données avec un mode multi-visualisations, offert par VizAssist, représente un atout majeur à la fois pour le choix des visualisations et aussi pour l'accomplissement de différentes tâches sur une même interface et sur différentes visualisations. De plus, même si quelques participants ont préféré avoir un mécanisme pour fixer quelques affectations dans les paramétrages proposés par l'AGI, ils le considèrent comme très utile pour ce processus quand il s'agit de visualiser des bases de données multidimensionnelles. Afin de présenter les avantages de VizAssist dans l'accomplissement de différentes tâches de fouille de données et surtout la résolution de quelques problématiques liées à l'utilisation de quelques techniques de fouille visuelle de données, nous présentons quelques cas d'études dans le chapitre suivant.

# Chapitre 9

# Cas d'études

**Résumé** : Nous proposons dans ce chapitre plusieurs cas d'études qui ont pour intérêt principal de mettre en avant les apports et avantages de VizAssist et de ses interfaces. Le premier cas concerne la technique adoptée pour ajouter de nouvelles visualisations dans le système. Les deuxième et troisième cas d'études permettent de montrer davantage la capacité de notre système à accomplir différentes tâches de fouille de données à travers un processus visuel et interactif. Le dernier cas d'étude porte sur l'intérêt que peut révéler le processus interactif sur lequel s'appuie notre outil pour la réorganisation (réduction du clutter) des dimensions dans des visualisations multidimensionnelles.

## 9.1 Introduction

Dans le but de mettre en avant les différents avantages de notre assistant VizAssist, nous proposons dans ce chapitre quelques cas d'utilisation. Nous commençons par présenter la méthodologie que nous avons définie pour l'intégration d'une nouvelle visualisation dans notre système. Nous avons pris pour cela comme exemple la représentation graphique en *"coordonnées parallèles"*. Nous montrons ensuite les avantages de notre système pour accomplir une tâche de fouille de données, en l'occurrence la découverte de variables discriminantes pour distinguer des classes de données à travers une application sur des jeux de données réels. Nous exposons enfin, l'intérêt que peut révéler le processus interactif sur lequel s'appuie notre outil pour la réorganisation des dimensions dans des visualisations multidimensionnelles. Nous terminons par une conclusion.

## 9.2 Ajouter une nouvelle visualisation dans VizAssist

Un des atouts de l'architecture modulaire de notre assistant est la possibilité d'intégrer de nouvelles visualisations de manière simple en s'appuyant sur des librairies de visualisation existantes, codées en langage JAVA. Nous illustrons dans ce qui suit les étapes du processus d'ajout de toute nouvelle visualisation dans notre outil à travers un exemple réel d'intégration de la visualisation *"coordonnées parallèles"*. En effet, pour ajouter cette dernière dans notre assistant, nous nous sommes appuyés sur une librairie JAVA utilisée par l'application ***parvis***[11]. La figure 9.1 illustre un aperçu de la visualisation *"coordonnées parallèles"* telle qu'elle est proposée par le système ***parvis***.

Notre choix de cette librairie est justifié essentiellement par trois critères : 1) l'aspect multidimensionnel de cette visualisation et son adéquation au type de jeux de données visualisés par VizAssist, 2) les différentes interactions proposées par cette visualisation dont l'avantage est de faciliter la réalisation de beaucoup de tâches de fouille de données (voir 5.4), 3) la libre utilisation du code source de l'application ***parvis***, puisque ce dernier est considéré comme un projet *"open source"*[12]. Le processus d'intégration de cette nouvelle visualisation dans notre assistant consiste donc à procéder comme suit :

1. **première étape :** nous commençons par définir la description conceptuelle de la visualisation dans la base de connaissances (voir section 5.5). Dans le cas de la visualisation *"coordonnées parallèles"*, nous nous sommes appuyés sur la description définie dans le tableau 5.6. Ce dernier illustre seulement la partie de la description qui concerne la définition de son élément graphique de base (polyligne) et les attributs visuels qui le caractérisent (Axis1, ...,Axis20, couleur). Afin de compléter cette description par les autres caractéristiques nécessaires pour renseigner une visualisation dans notre base de connaissances, nous avons ensuite étendu cette description avec les paramètres décrits dans le tableau 5.12. Concernant la catégorie et la dimension visuelle de cette visualisation, cette dernière est *"spatiale"* et représentée dans un espace *"2D"* (plan). Partant du constat de la multitude des objectifs que la visualisation *"coordonnées parallèles"* peut permettre d'accomplir, nous lui avons attribué

11. http ://www.mediavirus.org/parvis/
12. http ://fr.wikipedia.org/wiki/Open_Source_Definition

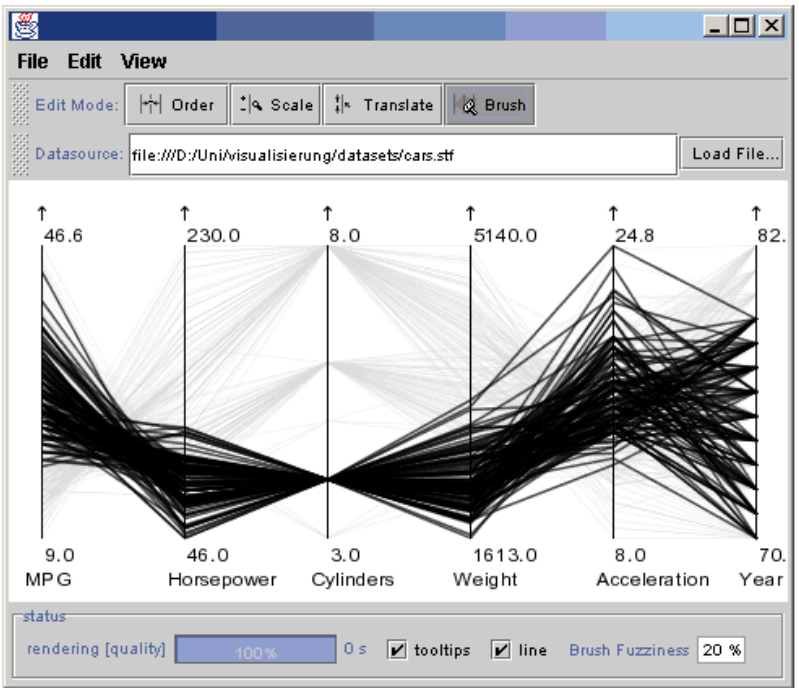



FIGURE 9.1 – Aperçu de l'application ***"parvis"*** utilisant la visualisation *"coordonnées parallèles"*.

tous les objectifs définis dans 5.4. L'ajout de tout nouvel objectif à cette visualisation peut se faire de manière simple via l'interface web de VizAssist (voir section 5.5.2). Pour la bibliographie, nous avons mentionné la références des auteurs de cette visualisation en l'occurrence Inselberg et Dimsdale [Inselberg et Dimsdale, 1990].

2. **deuxième étape :** cette phase consiste à adapter le format du fichier de données fourni en entrée de la libraire de visualisation à intégrer au modèle de données utilisé par VizAssist. À ce jour, toutes les librairies de visualisations utilisées par VizAssist acceptent deux formats de fichiers en entrée : un fichier Excel dont l'extension est "xls" et dont la représentation est conforme au tableau 5.2 et un fichier XML dont la structure doit être décrite comme dans la figure 6.4.
3. **troisième étape :** c'est la dernière phase du processus d'intégration de visualisations dans VizAssist et elle consiste en effet à charger la librairie JAVA de la visualisation dans le projet principal de notre application. A noter qu'aucune modification n'est nécessaire pour adapter le code source de la visualisation ajoutée pour que le module de suggestion des visualisations ou celui d'optimisation des paramétrages tienne compte de cette dernière (librairie de visualisation). Ceci est rendu possible grâce au format du fichier XML (voir figure 6.4) utilisé dans le protocole de communication entre la librairie de visualisation et la base de connaissances qui a été défini de manière standard et générique (voir section 6.3.2). Ainsi, après avoir chargé un jeu de données à représenter, VizAssist extrait à partir de la base de connaissances, la description conceptuelle de toutes les visualisations pré-sélectionnées pour être générées. Cette description est ensuite exploitée par le module d'appariement de notre outil

pour créer une mise en correspondance entre les attributs de données et les attributs visuels de chaque visualisation sélectionnée. Le résultat est également mis dans le fichier XML. Ensuite, ce même fichier XML est exploité par notre outil pour créer les rendus visuels des visualisations en s'appuyant sur les libraires de visualisations utilisées par VizAssist. Les deux figures 9.2 et 9.3 illustrent respectivement l'aperçu de la visualisation *"coordonnées parallèles"* sur la première et seconde interface obtenues suite à son intégration dans VizAssist.

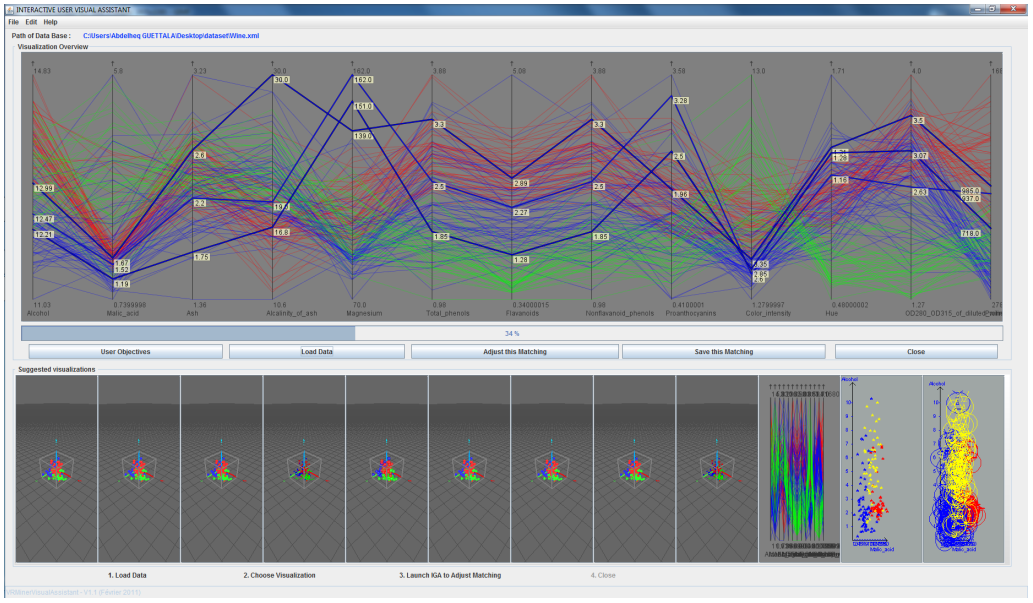



FIGURE 9.2 – Aperçu de la visualisation *"coordonnées parallèles"* dans la première interface de VizAssist après son intégration . Le jeu de données visualisé est la base de données WINE [Blake et Merz, 1998].

## 9.3 Découvertes de variables discriminantes (pour des classes existantes)

Découvrir les variables permettant de construire des classes séparées peut être considéré comme une tâche de fouille de données. En effet, cette tâche consiste à regrouper des individus d'un jeu de données dans des sous-ensembles de manière pertinente pour accomplir une opération d'analyse de données. Bien qu'il existe plusieurs techniques et algorithmes pour effectuer cette opération, l'exécution de ces derniers se fonde généralement sur un processus automatique, dans la majorité des outils de fouille de données. Cela signifie que les utilisateurs de ces systèmes ne peuvent pas intervenir dans le processus, ce qui peut réduire l'efficacité des résultats présentés en sortie (à leurs utilisateurs). De plus, il se peut que parfois un expert ou même un utilisateur novice désire étudier les résultats intermédiaires (fournis) afin d'extraire des connaissances pouvant être discriminantes dans un processus décisionnel (ex. découverte de connaissances). Malheureusement,

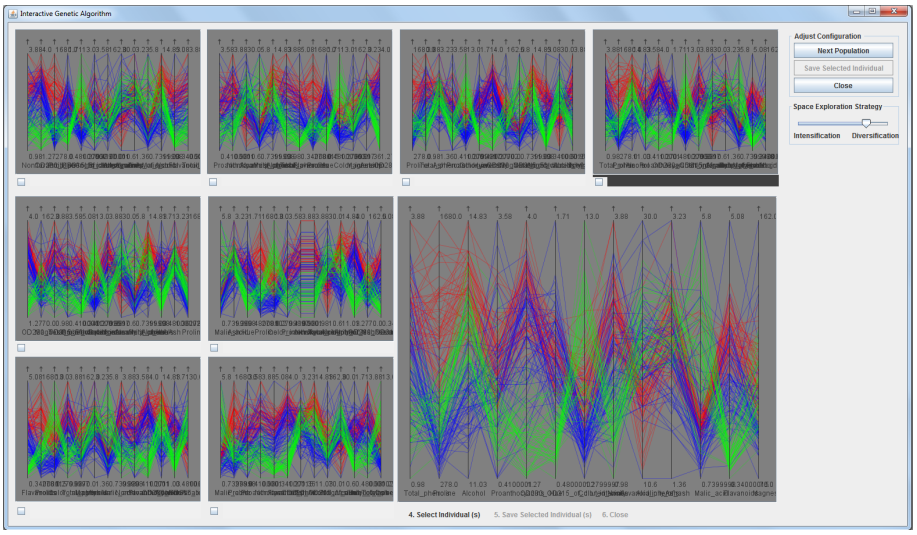



FIGURE 9.3 – Aperçu de la visualisation *"coordonnées parallèles"* dans la seconde interface de VizAssist après son intégration . Le jeu de données visualisé est la base de données WINE [Blake et Merz, 1998].

ces résultats ne sont pas toujours accessibles dans un processus de fouille de données. Pour remédier à cette contrainte liée aux méthodes automatiques, nous proposons dans VizAssist une approche basée sur un processus visuel et interactif. En effet, en s'appuyant sur un algorithme génétique interactif et des signes visuels pré-attentifs (couleur, forme, position), VizAssist permet de faciliter pour différentes catégories d'utilisateurs (experts ou novices) la détection des variables discriminantes pouvant constituer des classes de données séparées. Notons que le déroulement de cette opération dans VizAssist ne nécessite aucun paramétrage de l'AGI et s'exécute sur la seconde interface (voir figure 4.3).

| Base de données | Nombre de données | Nombre d'attributs | Nombre de classes |
|---|---|---|---|
| IRIS | 150 | 5 | 3 |
| PIMA | 768 | 9 | 2 |
| Breast Tissue | 106 | 10 | 6 |
| Heart | 270 | 14 | 2 |

TABLE 9.1 – Description des jeux de données utilisés pour illustrer le déroulement du processus de découvertes de variables discriminantes avec VizAssist.

Les figures 9.4, 9.6, 9.5, 9.7 et 9.8 illustrent quelques résultats intermédiaires obtenus durant le processus de découverte de variables discriminantes des jeux de données présentés dans le tableau 9.1. À travers les visualisations affichées sur l'interface et en s'appuyant sur les couleurs, on peut percevoir des classes pour chaque jeu de données choisi. Cela permet en effet aux utilisateurs de sélectionner à chaque étape du déroulement de l'AGI les individus présentant les meilleurs résultats jusqu'à l'aboutissement à des solutions satisfaisantes.

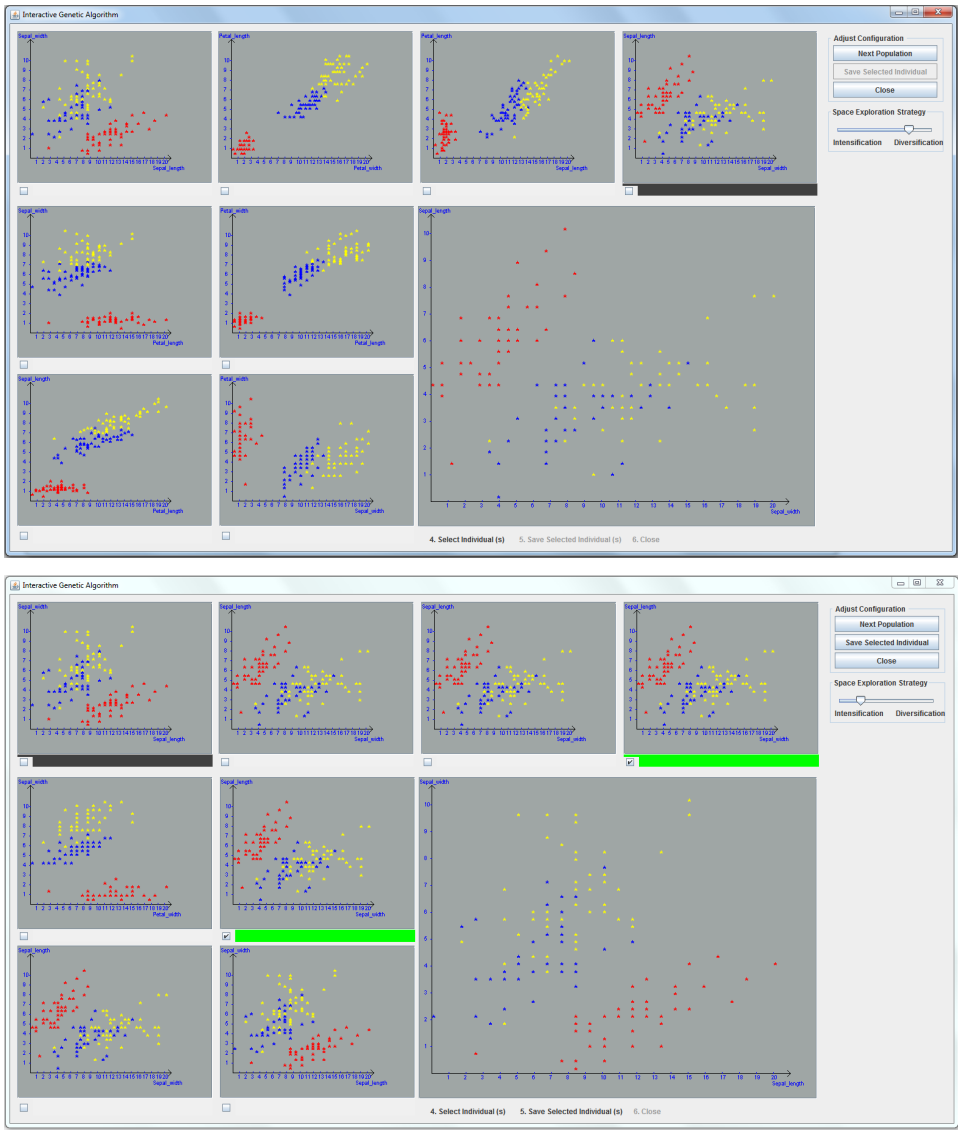



FIGURE 9.4 – Processus visuel et interactif de séparation des classes du jeu de données IRIS [Fisher, 1936] en s'apputyant sur la visualisation *"Nuage de points 2D"*.

## 9.4 Réorganisation interactive des dimensions dans une visualisation multidimensionnelle : cas des *"Coordonnées parallèles"*

Selon [Peng *et al.*, 2004], la réorganisation des dimensions dans les visualisations multidimensionnelles est un facteur décisif qui peut jouer un rôle important dans leurs expressivités. En effet, Peng et al. considèrent que dans ce type de représentations graphiques toute variation opérée sur l'ordre des dimensions peut réduire le bruit qu'elles peuvent contenir sans que cela impacte les données visualisées. Lors d'un processus d'exploration et d'ana-

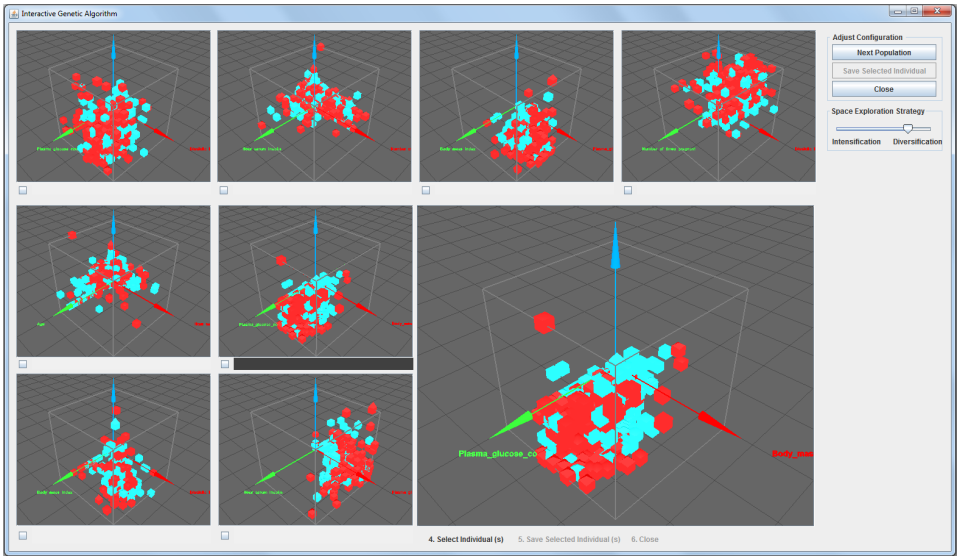



FIGURE 9.5 – Processus visuel et interactif de séparation des classes du jeu de données PIMA [Blake et Merz, 1998] en s'appuyant sur la visualisation *"Nuage 3D Cube Chanel V2"*.

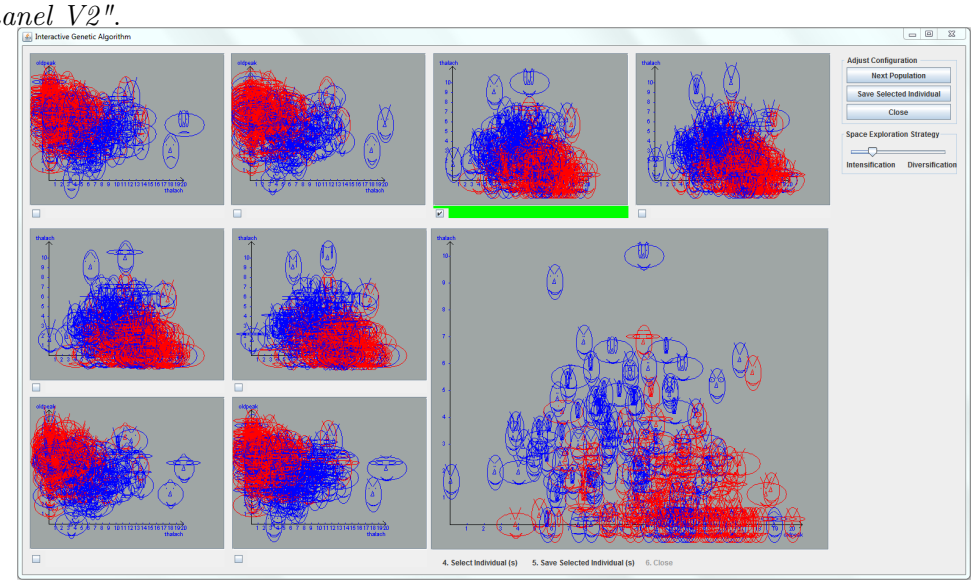



FIGURE 9.6 – Processus visuel et interactif de séparation des classes du jeu de données PIMA [Blake et Merz, 1998] en s'appuyant sur la visualisation *"Visages de Chernoff"*.

lyse d'un jeu de données, cette opération pourra donc révéler de nouvelles connaissances ce qui permettra d'appuyer les utilisateurs dans leur processus décisionnel.

Actuellement, dans la littérature, il existe seulement deux approches pour réorganiser les dimensions dans une visualisation. La première approche dite ***"manuelle"*** consiste à donner un contrôle direct aux utilisateurs pour modifier l'ordre des dimensions selon leur be-

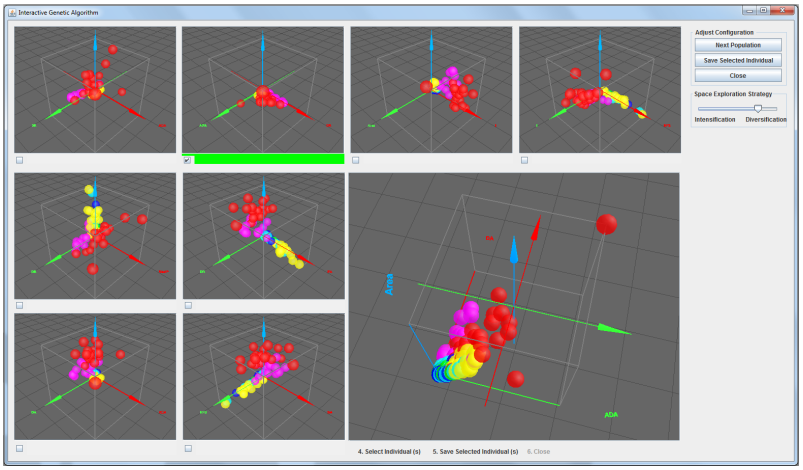



FIGURE 9.7 – Processus visuel et interactif de séparation des classes du jeu de données Breast Tissue [Blake et Merz, 1998] en s'appuyant sur la visualisation *"Nuage 3D Sphère"*.

soins [Stolte *et al.*, 2002]. Tandis que la seconde approche dite ***"automatique"*** permet de générer des combinaisons de dimensions sans intervention des utilisateurs [Cancino *et al.*, 2012]. Bien que ces deux approches permettent d'assurer la réorganisation des dimensions dans les visualisations multidimensionnelles, cette opération peut devenir très rapidement fastidieuse quand il s'agit d'exécuter cette tâche sur des jeux de données décrits par un nombre important de dimensions. En effet, dans les deux cas d'utilisation (manuel et automatique), cette difficulté découle du nombre de combinaisons nécessaires pour trouver le meilleur appariement répondant aux objectifs d'analyse de l'utilisateur. Pour résoudre cette contrainte, nous proposons dans notre outil une nouvelle approche qui consiste à réorganiser les dimensions de manière interactive en s'appuyant sur un AGI (voir chapitre 7).

Dans le but d'illustrer les avantages de notre technique sur des visualisations multidimensionnelles, nous avons appliqué notre méthode de réorganisation sur la représentation graphique "coordonnées parallèles". Dans cette visualisation, le changement des importances des attributs de données par l'AGI effectue un changement dans l'ordre d'affichage des axes parallèles. Les figures 9.9, 9.10 et 9.11 illustre les résultats obtenus pendant le déroulement du processus interactif de réorganisation des axes (dimensions) qui a été appliqué sur le jeu de données *Ecoli*[13]. Cet ensemble de données est décrit par 336 données dont chacune est définie par 9 attributs de données dont 7 sont numériques, 1 attribut de type texte et 1 attribut classe. On peut remarquer que l'utilisation de l'AGI dans notre outil dans ce cas d'étude contribue à faciliter la réorganisation des axes. En effet, il permet aux utilisateurs d'affiner leurs préférences concernant l'ordre des dimensions et diminuant ainsi l'espace de recherche des meilleures combinaisons possibles. Cela permet donc de rendre cette visualisation plus claire lors de son utilisation.

13. http ://archive.ics.uci.edu/ml/datasets/Ecoli

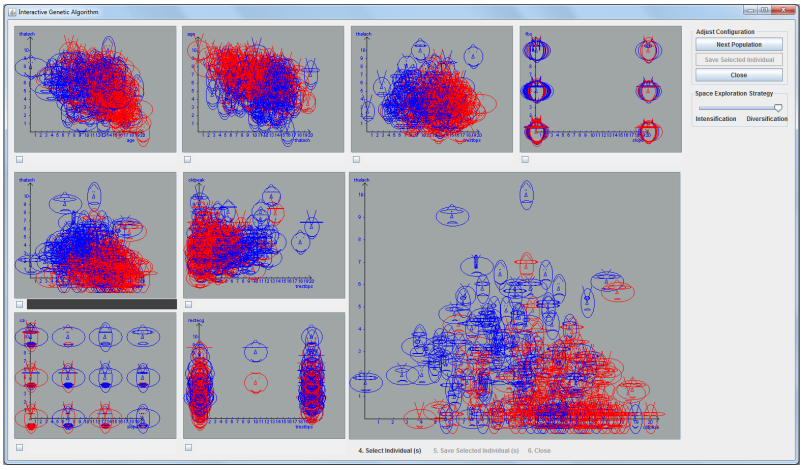


FIGURE 9.8 – Processus visuel et interactif de séparation des classes du jeu de données Heart [Blake et Merz, 1998] en s'appuyant sur la visualisation *"Visages de Chernoff"*

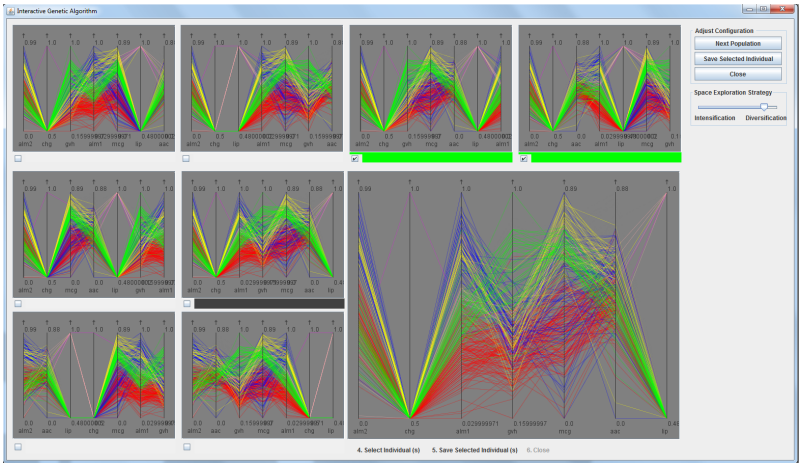


FIGURE 9.9 – Déroulement du processus interactif de réorganisation des axes (dimensions) dans la visualisation *"coordonnées parallèles"*. Application sur le jeu de données Ecoli [Blake et Merz, 1998]. Croisement de l'ordre de paramétrage de deux visualisations.

## 9.5 Conclusion

Nous avons présenté dans ce chapitre quelques cas d'étude résumant les différents apports de notre système qui viennent s'ajouter au choix et paramétrage des visualisations des méthodes de fouille visuelle de données. En effet, nous avons commencé par mettre en avant les avantages de l'architecture générique et modulaire de VizAssist qui facilite l'intégration de toute nouvelle visualisation dans sa base de connaissances et sa librairie de visualisations. Pour cela, nous avons illustré à travers un exemple concret la méthodologie à suivre pour ajouter la visualisation *"coordonnées parallèles"*. Nous avons montré ensuite, à travers des applications sur des jeux de données réels, l'efficacité de l'approche interactive et visuelle sur laquelle repose notre système pour accomplir un processus de découverte

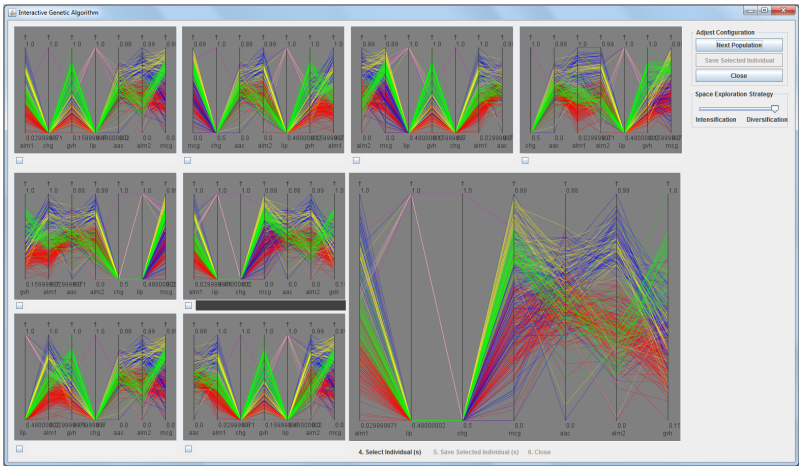



FIGURE 9.10 – Déroulement du processus interactif de réorganisation des axes (dimensions) dans la visualisation *"coordonnées parallèles"*. Application sur le jeu de données Ecoli [Blake et Merz, 1998]. Génération aléatoire de l'ordre des axes.

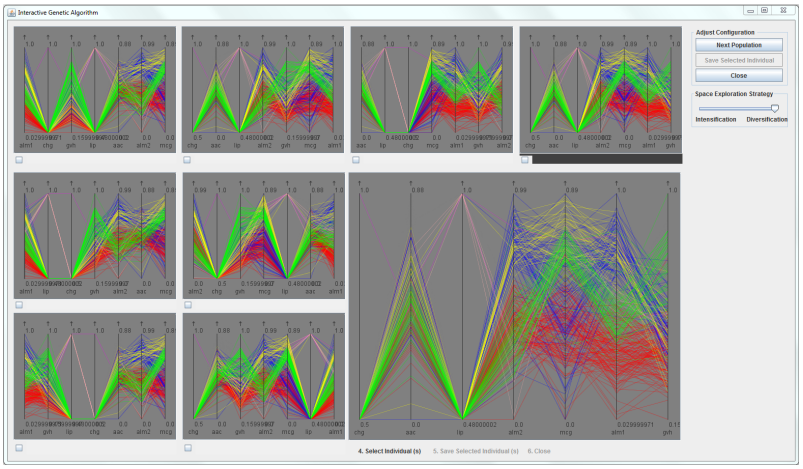



FIGURE 9.11 – Déroulement du processus interactif de réorganisation des axes (dimensions) dans la visualisation *"coordonnées parallèles"*. Application sur le jeu de données Ecoli [Blake et Merz, 1998]. Génération aléatoire de l'ordre des axes.

de variables discriminantes pour distinguer des classes de données dans une visualisation. Conscient de l'apport majeur de l'utilisation d'un AGI pour réorganiser des dimensions dans une visualisation multidimensionnelle, nous avons illustré via un cas d'étude l'avantage de son application dans la visualisation *"coordonnées parallèles"*. On peut conclure donc que notre outil propose à ses utilisateurs : 1) le choix et le paramétrage des visualisations, et 2) l'accomplissement de plusieurs tâches de fouille de données de manière facile et intuitive.

# Conclusion et perspectives

## Conclusion

Dans cette thèse, nous avons abordé le problème d'automatisation du processus de choix et de paramétrage des visualisations en fouille visuelle de données. En effet, si on se limite strictement à ce domaine, les approches existantes permettant de résoudre ce problème sont peu nombreuses (voir chapitre 2) et ont certaines limitations. Par exemple dans certains systèmes le nombre de visualisations gérées est limité (parfois à une seule visualisation), ou bien les objectifs utilisateurs ne sont pas pris en considération lors du processus de suggestion des visualisations. Ces systèmes utilisent aussi bien souvent un moteur d'inférence ou un algorithme d'appariement afin de mettre en correspondance les attributs des données avec les attributs visuels des visualisations. Proposant des interfaces à base de paramétrage manuel pour réajuster les paramétrages qu'ils suggèrent, les utilisateurs de ces systèmes sont généralement obligés d'exécuter une tâche d'appariement fastidieuse. Cette tâche est d'autant plus difficile, dans le cas de la visualisation des jeux de données décrits par un nombre important d'attributs de données. De plus, les systèmes existants se fondent sur un modèle général des préférences visuelles des utilisateurs et négligent donc toute possibilité de personnalisation.

La contribution principale de cette thèse est la conception et le développement d'un nouvel assistant utilisateur *"VizAssist"* proposant une nouvelle approche pour résoudre le problème d'automatisation du processus de choix et de paramétrage des visualisations en fouille visuelle de données. Un outil qui peut également servir pour explorer de manière interactive une base de données, soit avec un mode multi-visualisations comme dans sa première interface, soit avec un mode exploration interactive comme dans sa seconde interface. VizAssist repose donc sur deux étapes dont la première permet de suggérer plusieurs visualisations avec des paramétrages définis sur la base de la description du jeu de données utilisateur et des visualisations proposées. Quant à la seconde étape, elle permet d'optimiser et d'affiner les paramétrages proposés de manière visuelle et interactive. Les paramétrages générés par VizAssist dans les deux étapes sont appliqués directement sur les données utilisateurs. Cela signifie que les utilisateurs n'ont pas un aperçu statique de ce que donne la visualisation mais plutôt la possibilité de tester dynamiquement toutes les fonctionnalités des visualisations sur leurs jeux de données. Lorsque cela est possible, VizAssist donne la possibilité de sélectionner des données dans une visualisation et de voir apparaître les données sélectionnées dans les autres grâce à la technique de "brushing". Un autre atout de VizAssist est qu'il est fondé sur une architecture générique et modulaire

simple. Une architecture qui intègre plusieurs modules : un modèle des données et des objectifs utilisateur, une base de connaissances sur les visualisations, un module d'appariement et de suggestion de visualisations, un module de paramétrage interactif avec un algorithme génétique interactif.

Les principales contributions des modules constituant l'architecture de VizAssist sont les suivantes :

- L'utilisation de plusieurs types de données pour renseigner les attributs des jeux de données utilisateurs à visualiser, représente un des atouts du modèle de données de VizAssist. En effet, cet avantage nous a permis de visualiser efficacement différentes bases de données multidimensionnelles et hétérogènes. L'autre atout de notre modèle de données réside dans sa capacité à gérer les appariements entre les attributs de données et les attributs visuels à travers l'utilisation d'une *"importance"*. Cela signifie que quand un utilisateur désire visualiser un jeu de données, défini par un nombre important d'attributs, il peut fixer une priorité à ses attributs de données les plus importants pour les faire apparaître dans les visualisations proposées par VizAssist. De plus, cette caractéristique (importance) a pour apport d'influencer les appariements proposés par VizAssist lors de la seconde étape proposée par notre outil en permettant de faire varier l'ordre d'importance des attributs à visualiser. Étant indispensables dans le processus de choix des visualisations, les objectifs utilisateurs sont pris aussi en compte dans VizAssist à travers un modèle des objectifs utilisateurs. Afin d'élaborer ce modèle, nous nous sommes appuyés sur une liste prédéfinie d'objectifs recensés dans la littérature. Dans VizAssist, les utilisateurs fixent les priorités de leurs objectifs d'exploration et d'analyse via une interface en s'appuyant sur une échelle de priorité.

- La conception d'une base de connaissances sur les visualisations dont l'apport majeur est de faciliter la description des modèles de visualisations (la description conceptuelle des visualisations) gérées par VizAssist. L'intérêt principal, à travers l'utilisation de cette base de connaissances, est la simplification du processus d'intégration de nouvelles connaissances sur les visualisations et les préférences visuelles des utilisateurs (esthétiques, perceptives, etc.). Cela permet donc de personnaliser les visualisations sur la base du profil de chaque utilisateur. De plus, dû à son aspect générique, la base de connaissances sur les visualisations ne nécessite pas de modification dans le code source du projet lors de l'extension du nombre de visualisations. Notons que le modèle conceptuel de la base de connaissances de VizAssist permet d'ajouter de nouvelles visualisations selon deux méthodes : 1) développer une nouvelle visualisation en s'appuyant sur l'encodage visuel existant, et 2) intégrer des visualisations existantes en accommodant leurs encodages à celui utilisé par VizAssist ainsi qu'en adaptant le format de fichier de données qu'ils prennent en entrée à ceux gérés par notre assistant : XML et XLS.

- Le module d'appariement et de suggestion de visualisations a pour apport majeur de permettre de suggérer plusieurs visualisations (2D et 3D) dynamiques avec des paramétrages possibles appliquées directement sur les données utilisateur. L'autre avantage de ce module est qu'il repose sur un protocole simple de visualisation, passant par un échange au format XML pour communiquer à la libraire de visuali-

sations de VizAssist (codées en JAVA) les paramétrages des visualisations à générer. L'interface de suggestion des visualisations utilisée par VizAssist a deux principaux avantages. Son premier avantage est qu'elle permet une exploration interactive et comparative des jeux de données utilisateurs sur l'ensemble des visualisations proposées en s'appuyant sur un affichage multi-visualisations. Le second avantage de l'interface est qu'elle propose une vue dynamique, ce qui permet aux utilisateurs de tester les fonctionnalités des visualisations sur leurs données.

– Le module de paramétrage interactif a pour atout principal d'aider les utilisateurs de VizAssist à affiner et optimiser un paramétrage initial entre les attributs de données d'un jeu de données et les attributs visuels d'une visualisation choisie pour accomplir les tâches d'analyse et d'exploration. Ce dernier utilise un algorithme génétique interactif dont l'utilisation permet de résoudre les limitations liées au paramétrage manuel constatées dans les systèmes existants. Le module de paramétrage interactif s'appuie sur une interface simple et intuitive qui ne nécessite aucune pré-connaissances sur les algorithmes génétiques de la part des utilisateurs. Cela permet d'éviter aux utilisateurs de fixer manuellement les appariements, leur permettant ainsi de se concentrer sur l'évaluation des appariements proposés à chaque étape du déroulement de l'AGI.

L'analyse des résultats obtenus à travers les différentes expérimentations de VizAssist, en l'occurrence l'évaluation utilisateur et les différents cas d'études, a permis de montrer que :

– Les interfaces que nous avons développées sont avantageuses pour des utilisateurs novices car elles n'exigent pas de connaissances a priori de leur part.

– Les utilisateurs ont montré aussi, de manière subjective, l'intérêt qu'ils ont porté à l'utilisation ultérieure de notre système. En effet, ils considèrent que la capacité d'explorer des bases de données avec un mode multi-visualisations, offert par VizAssist, représente un atout majeur à la fois pour le choix des visualisations et aussi pour l'accomplissement de différentes tâches sur une même interface et sur différentes visualisations.

– L'avantage de l'architecture générique et modulaire de VizAssist qui permet de faciliter l'intégration de toute nouvelle visualisation dans sa base de connaissances et sa librairie de visualisations.

– L'utilité du module de paramétrage interactif dans le cas de la visualisation des bases de données multidimensionnelles décrites par un nombre important d'attributs de données. Ce constat a été confirmé par les utilisateurs à l'issue de l'évaluation utilisateur.

– L'apport majeur de l'utilisation d'un AGI pour réorganiser des dimensions dans une visualisation multidimensionnelle.

– L'efficacité de l'approche interactive et visuelle sur laquelle repose notre système pour accomplir différentes tâches de fouille de données comme le clustering et la

détection des variables discriminantes pour trouver des classes existants dans des jeux de données réels, de manière facile et intuitive.

## Perspectives

Les perspectives faisant suite à ce travail sont :

- Le développement d'un module de retour d'expériences afin de permettre la prise en compte des retours utilisateurs et améliorer ainsi les recommandations de VizAssist. En effet, partant du constat de la variabilité du niveau de perception visuelle chez les utilisateurs et la différence de leurs niveaux d'expériences en visualisation d'informations, ce processus assurera une prise en considération du retour d'expériences utilisateur pour :
  1. personnaliser leurs préférences sur les visualisations et adaptant ainsi les suggestions de VizAssist au profils de chaque utilisateur (ex. une préférence d'utilisation des visualisations en dimension 2D ou de visualisation des données toujours avec un nuage de points),
  2. personnaliser le paramétrage des visualisations de la base de connaissances (voir 5.5) en mettant à jour l'ordre de priorités des objectifs utilisateur et/ou les valeurs d'importances des attributs visuels décrivant une visualisation,
  3. personnaliser le paramétrage du jeu de données à visualiser. Dans ce cas, les utilisateurs peuvent être amenés à modifier les importances des attributs de données pour trouver le paramétrage satisfaisant les objectifs d'exploration qu'ils désirent accomplir sur une visualisation donnée,
  4. améliorer les caractéristiques de notre système : prise en compte des suggestions de visualisations à intégrer au système, augmentation de la fiabilité et la vitesse du système, etc.

- Le développement d'une version web de VizAssist. En effet, notre objectif dans le futur est de pouvoir proposer à tout concepteur d'une méthode de visualisation de la modéliser dans notre base de connaissances et de l'inclure dans notre assistant. Cela nous permettrait d'avoir plus de visualisations dans notre base de connaissances et de pouvoir tester davantage le module d'appariement et de suggestion de visualisations.

- L'utilisation de VizAssist dans le cas des *"Open data"*. En effet, L'open data en tant que concept peut être considéré comme un nouveau mécanisme de diffusion sur lequel s'appuient plusieurs institutions publiques (ex. gouvernements) ou privés (ex. des entreprises) pour mettre à la disposition de la population des jeux de données liées à la société. Pour publier ce type de données, ces institutions s'appuient généralement sur des sites internet où des milliers de jeux de données sont consultables gratuitement (ex. plus de trois cent cinquante mille ensembles de données sont proposés sur le site : http ://www.data.gouv.fr/). Notons que ces données peuvent porter sur divers domaines comme les finances publiques, les statistiques sur la population,

des résultats d'élections ou de sondages. Cependant, la majorité des sites offrant ces données publiques (open data) repose dans leurs publications sur des formats textuels ou tabulaires. L'exploration de ces données par les utilisateurs peut donc être contraignante (voir section 1.2.1). Étant conscient que l'intérêt principal d'une visualisation est de permettre la compréhension d'un nombre important de données en réduisant l'effort intellectuel à fournir, nous considérons que notre assistant VizAssist peut être d'un grand apport dans la représentation de ces données publiques. De plus, VizAssist peut s'appuyer sur la description des jeux de données à visualiser (nombre d'attributs, type de données, etc.) ainsi que les objectifs d'exploration fixés par les utilisateurs pour les guider dans le choix des représentations graphiques les plus adéquates.

– Une perspective, en cours d'étude également, concerne l'amélioration de l'AGI utilisé par VizAssist de manière à permettre de limiter le nombre d'itération pour aboutir à des solutions optimales. En effet, nous pensons que l'utilisation d'une heuristique de type recherche tabou [Soriano et Gendreau, 1997] avec notre AGI devrait permettre d'éviter la génération des mêmes appariements (entre attributs de données et attributs visuels) durant le processus génétique.

# Annexes

# Annexe A

# Protocole Évaluation Utilisateur

## A.1 Questionnaire

Dans notre évaluation utilisateur, nous nous sommes basés sur un questionnaire regroupant trois parties :

### A.1.1 Identification de l'utilisateur

#### A.1.1.1 Profil utilisateur

1. Nom
2. Age
3. Sexe
4. Niveau d'études

#### A.1.1.2 Niveau de l'utilisateur en visualisation d'informations

1. Savez-vous ce qu'est une base de données multidimensionnelle ?
2. Avez-vous utilisé auparavant des représentations graphiques de vos données (histogrammes 2D ou 3D, courbes, nuages de points 2D ou 3D, camembert, boîtes à moustaches, etc.) ?
3. Si oui, dans quel but :
   (a) Avoir une vue d'ensemble de vos données.
   (b) Présenter de vos résultats.
   (c) Extraire des connaissances depuis vos données.
   (d) Si autre, merci de préciser :
4. Si oui, quelle représentation graphique utilisez-vous le plus souvent ?
5. Quelles sont les caractéristiques importantes qui ont motivé votre choix de cette (ces) visualisation (s) ?
   (a) Représentation des visualisations en : (1D , 2D, 3D ou nD).

(b) Interaction / Tâches d'exploration (zoom, clique, etc. / sélection des données, etc.).

(c) Facilité d'analyse de vos données et d'interprétation de l'ensemble de la représentation graphique.

(d) Présentation à d'autres personnes.

6. Avec quel type d'application avez-vous l'habitude de générer vos visualisations des données ?

   (a) Microsoft Excel.

   (b) MATLAB.

   (c) R, SAS ou autre logiciel de statistiques.

   (d) Si autres, merci de préciser.

7. Avec quelle fréquence utilisez-vous cette application ?

   (a) 1 fois par jour.

   (b) 1 fois par semaine.

   (c) 1 fois par mois.

   (d) 1 fois par an.

   (e) Plusieurs fois par jour.

   (f) Plusieurs fois par semaine.

   (g) Plusieurs fois par mois.

   (h) Plusieurs fois par an.

8. Niveau de maîtrise de (s) l'application (s) ?

9. D'après votre utilisation de cette application, et dans le but d'une prochaine utilisation pour la représentation graphique de vos données, quels sont les avantages qui vous semblent les plus pertinents dans le choix d'une application de visualisation ?

   (a) Fonctionnalités (le nombre de visualisations proposées).

   (b) Facile d'utilisation (facilité du processus de paramétrage des données avec les visualisations gérées par l'application).

   (c) Performance (le temps mis par l'application pour la génération des visualisations, à partir du paramétrage créé, est très rapide).

   (d) Autres.

10. Avez-vous déjà utilisé une visualisation de données multidimensionnelle en 3D ?

11. Quel est votre niveau de maîtrise des outils de navigation (ex. souris 3D) dans un environnement 3D (de débutant à expert) ?

12. Fréquentation des cinémas 3D (de jamais à tous les jours) ?

13. Utilisation de jeux vidéo en 3D (de jamais à tous les jours) ?

### A.1.2 Évaluation utilisateur (accomplissement de tâches)

#### A.1.2.1 Tâche 1

1. **Description de la tâche :** cette tâche s'intéresse à la comparaison de la fluidité des tâches de configuration dans les 2 systèmes : VRMiner et l'assistant utilisateur. Pour réaliser cette tâche avec l'assistant les utilisateurs s'appuient uniquement sur la première interface.
2. **Question :** Obtenir une visualisation de données dans laquelle on représente la classe des individus, un attribut de type texte et un attribut de type image.
3. **Réponses :** afin de répondre à cette question, le participant devrait choisir une visualisation sur les deux systèmes qui satisfont cette tâche. Les réponses possibles sont : (oui : visualisation trouvée, non : visualisation non trouvée).
4. **Résultat souhaité :** (oui ou non) plus mesure de la durée (en seconde) de la réalisation de cette tâche pour chaque participant.
5. **Tests :** Pour la réalisation de cette tâche, les participants devraient faire trois tests. Pour chaque test, deux bases de données sont associées dont chacune est utilisée par un système (voir tableau A.1).

| Tests | Base de données |
|---|---|
| Test 1 | BD1, BD2 |
| Test 2 | BD3, BD4 |
| Test 3 | BD5, BD6 |

TABLE A.1 – Bases de données utilisées dans chaque test. Le choix des bases de données s'effectue de manière aléatoire d'un utilisateur à un autre.

6. **Description des bases de données :** Les bases de données sont caractérisées par :
   (a) 5 attributs numériques (randomisation des noms d'attributs dans chaque base de données).
   (b) 1 attribut symbolique (classe).
   (c) 1 attribut image (à chaque classe on associe une image pour tous les individus qui la caractérisent).
   (d) 1 attribut texte (un mot associé à chaque individu).
7. **Questions nécessaires pour cette tâche (utilisées dans le formulaire) :**
   (a) Quel est le logiciel choisi pour le test ?
   (b) Quelle est la base de données choisie ?
   (c) La visualisation est-elle trouvée ?
   (d) Temps en secondes.

#### A.1.2.2 Tâche 2

1. **Description de la tâche :** cette tâche s'intéresse à la comparaison de la fluidité des tâches de paramétrage et d'exploration dans les 2 systèmes : VRMiner et l'assistant utilisateur. En effet, l'intérêt est de mettre en avant la différence entre un paramétrage manuel avec VRMiner (interface manuelle) et un paramétrage semi-automatique avec l'assistant (interface basée sur l'AGI). Pour réaliser cette tâche avec l'assistant les utilisateurs s'appuient sur la deuxième interface.
2. **Question :** Trouver une représentation graphique qui soit la moins bruitée possible.
3. **Réponses :** le participant devrait indiquer obligatoirement (même s'il n'a pas trouvé la bonne réponse) le nom des trois attributs de données affectés aux trois axes (X, Y et Z) renseignés dans la visualisation qui lui semble la plus appropriée pour satisfaire cette tâche.
4. **Résultat souhaité :** mesure de la durée (en seconde) de la réalisation de cette tâche pour chaque participant plus qualité de la réponse donnée.
5. **Tests :** Pour la réalisation de cette tâche, les participants devraient effectuer trois tests. Pour chaque test, deux bases de données sont associées dont chacune est utilisée par un système (voir tableau A.2).

| Tests | Base de données |
|---|---|
| Test 1 | BD7, BD8 |
| Test 2 | BD9, BD10 |
| Test 3 | BD11, BD12 |

TABLE A.2 – Bases de données utilisées dans chaque test. Le choix des bases de données s'effectue de manière aléatoire d'un utilisateur à un autre.

6. **Description des bases de données :** Les bases de données sont caractérisées par :
   (a) 30 attributs numériques (3 donnent la bonne réponse, tandis que les 27 autres forment un bruit généré de manière aléatoire et graduelle).
   (b) 1 attribut symbolique (classe).
7. **Questions nécessaires pour cette tâche (utilisées dans le formulaire) :**
   (a) Quel est le logiciel choisi pour le test ?
   (b) Quelle est la base de données choisie ?
   (c) Quel sont les indices des attributs ?
   (d) Temps en secondes.

### A.1.3 Bilan de l'utilisateur

1. Sur une échelle d'évaluation de 1 à 5 des caractéristiques mentionnées ci-dessous, merci d'indiquer votre appréciation pour les deux systèmes (VRMiner et l'assistant utilisateur) ?
   (a) Simplicité d'utilisation.

   (b) Rapidité de génération des visualisations.
   (c) Obtention du meilleur paramétrage de vos visualisations.
   (d) Quel système choisiriez-vous pour visualiser vos données ?
   (e) Facilité d'utilisation des interfaces.
   (f) Fiabilité du système (permis d'accomplir les tâches demandées).
   (g) Vitesse du logiciel (trop lent ou très rapide : grille de 1 à 5).
   (h) Comment trouvez-vous les deux systèmes (frustrant ou satisfaisant : grille de 1 à 5) ?
   (i) Comment trouvez-vous les deux systèmes (ennuyeux ou stimulant : grille de 1 à 5) ?
2. Comment trouvez-vous l'AGI qu'utilise l'assistant utilisateur (de pas utile à utile) ?
3. Quels autres types de fonctionnalités voudriez-vous rajouter à l'assistant utilisateur ?

# Annexe B

# Publications

1. Abdelheq Et-tahir Guettala, Fatma Bouali, Christiane Guinot, Gilles Venturini : A User Assistant for the selection and parameterization of the visualizations in visual data mining, $16^{th}$ International Conference on Information Visualisation, (IV'2012) p. 252-257, Montpellier, France.

2. Abdelheq Et-tahir Guettala, Fatma Bouali, Christiane Guinot, Gilles Venturini : Un assistant utilisateur pour le choix et le paramétrage des méthodes de fouille visuelle de données, $12^{ième}$ Conférence Internationale Francophone sur l'Extraction et la Gestion de Connaissance, (EGC'2012) p. 399-404, janvier, Bordeaux, France.

3. Abdelheq Et-tahir Guettala, Fatma Bouali, Christiane Guinot, Gilles Venturini : Premiers résultats pour un assistant utilisateur en fouille visuelle de données, $18^{ième}$ Rencontres de la Société Francophone de Classification, (SFC'2011) p. 71-74, septembre, Orléans, France.

# Bibliographie


[Are, 2011] (2011). Distance-based relevance feedback using a hybrid interactive genetic algorithm for image retrieval. *Applied Soft Computing*, 11(2):1782–1791.

[Andrews, 1972] Andrews, D. F. (1972). Plots of high-dimensional data. *Biometrics*, 28(1):pp. 125–136.

[Ankerst, 2001] Ankerst, M. (2001). "visual data mining with pixel-oriented visualization techniques," presented at the acm sigkdd workshop on visual data mining.

[Azzag *et al.*, 2005] Azzag, H., Picarougne, F., Guinot, C. et Venturini, G. (2005). Vrminer : A tool for multimedia database mining with virtual reality. *Processing and Managing Complex Data for Decision Support*, (Ea 2101):318–339.

[Baker, 1993] Baker, E. (1993). Evolving line drawings. *In Proceedings of the 5th International Conference on Genetic Algorithms*, pages 627–, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Banzhaf, 1997] Banzhaf, W. (1997). Interactive evolution. *Handbook of Evolutionary Computation*, pages 1–5.

[Beasley *et al.*, 1993] Beasley, D., Bull, D. R. et Martin, R. R. (1993). An overview of genetic algorithms : Part 2 , research topics. *University Computing*, 15(4):1–15.

[Becker et Cleveland, 1987] Becker, R. A. et Cleveland, W. S. (1987). Brushing scatterplots. *Technometrics*, 29(2):127–142.

[Bendix *et al.*, 2005] Bendix, F., Kosara, R. et Hauser, H. (2005). Parallel sets : visual analysis of categorical data. *In Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pages 133–140.

[Bertin, 1983] Bertin, J. (1983). *Semiology of graphics.* University of Wisconsin Press, Berlin.

[Bertin, 1998] Bertin, J. (1998). Sémiologie graphique : les diagrammes, les réseaux, les cartes (3e éd.). *Paris : Éditions de l'EHESS.*

[Blake et Merz, 1998] Blake, C. L. et Merz, C. J. (1998). Uci repository of machine learning databases. *UCI repository of machine learning databases*, page http ://archive.ics.uci.edu/ml/.

[Bogacz et Trafton, 2005] Bogacz, S. et Trafton, J. G. (2005). Understanding dynamic and static displays : using images to reason dynamically. *Cogn. Syst. Res.*, 6(4):312–319.

[Boschetti et Takagi, 2001] Boschetti, F. et Takagi, H. (2001). Visualization of ec landscape to accelerate ec conversion and evaluation of its effect. *In Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 2, pages 880–886. IEEE.

[Boudjeloud et Poulet, 2005] Boudjeloud, L. et Poulet, F. (2005). Visual interactive evolutionary algorithm for high dimensional data clustering and outlier detection. *In* Ho, T. B., Cheung, D. W.-L. et Liu, H., éditeurs : *PAKDD*, volume 3518 de *Lecture Notes in Computer Science*, pages 426–431. Springer.

[Boudjeloud-Assala et Poulet, 2008] Boudjeloud-Assala, L. et Poulet, F. (2008). Algorithme interactif pour la sélection de dimensions en détection d'outlier. *Revue d'Intelligence Artificielle*, 22(3-4):401–420.

[Caldwell et Johnston, 1991] Caldwell, C. et Johnston, V. S. (1991). Tracking a criminal suspect through "face-space" with a genetic algorithm. *In Proceedings of the Fourth International Conference on Genetic Algorithm*, pages 416–421. Morgan Kaufmann Publisher.

[Cancino *et al.*, 2013] Cancino, W., Boukhelifa, N., Bezerianos, A. et Lutton, E. (2013). Evolutionary visual exploration : Experimental analysis of algorithm behaviour. *In VizGEC 2013, Workshop on Visualisation Methods in Genetic and Evolutionary Computation. Genetic and Evolutionary Computation Conference, GECCO 2013.*

[Cancino *et al.*, 2012] Cancino, W., Boukhelifa, N. et Lutton, E. (2012). Evographdice : Interactive evolution for visual analytics. *In IEEE Congress on Evolutionary Computation, June 10-15.* June 10-15, Brisbane, Australia.

[Card *et al.*, 1999] Card, S. K., Mackinlay, J. D. et Shneiderman, B., éditeurs (1999). *Readings in information visualization : using vision to think.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Carr *et al.*, 1987] Carr, D. B., Littlefield, R. J., Nicholson, W. L. et Littlefield, J. S. (1987). Scatterplot matrix techniques for large n. *Journal of the American Statistical Association*, 82(398):pp. 424–436.

[Casner, 1991] Casner, S. M. (1991). Task-analytic approach to the automated design of graphic presentations. *ACM Trans. Graph.*, 10(2):111–151.

[Ceglar *et al.*, 2003] Ceglar, A., Roddick, J. et Calder, P. (2003). Managing data mining technologies in organizations. chapitre Guiding knowledge discovery through interactive data mining, pages 45–87. IGI Publishing, Hershey, PA, USA.

[Chan, 2006] Chan, W. (2006). A survey on multivariate data visualization. *Department of Computer Science and Engineering. Hong Kong University of Science and Technology.*

[Chen, 2005] Chen, C. (2005). Top 10 unsolved information visualization problems. *Computer Graphics and Applications, IEEE*, 25(4):12–16.

[Chen *et al.*, 2007] Chen, J., Zheng, T., Thorne, W., Zaiane, O. R. et Goebel, R. (2007). Visual data mining of web navigational data. *In Information Visualization, 2007. IV'07. 11th International Conference*, pages 649–656. IEEE.

[Chernoff, 1973] Chernoff, H. (1973). The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68(342): pp. 361–368.

[Cho et Lee, 2002] Cho, S. B. et Lee, J. Y. (2002). A human-oriented image retrieval system using interactive genetic algorithm. *Trans. Sys. Man Cyber. Part A*, 32(3):452–458.

[Cleveland, 1979] Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74(368):829–836.

[Cleveland et McGill, 1984] Cleveland, W. S. et McGill, R. (1984). Graphical perception : Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 79(387):531–554.

[Collberg *et al.*, 2003] Collberg, C., Kobourov, S., Nagra, J., Pitts, J. et Wampler, K. (2003). A system for graph-based visualization of the evolution of software. *In In Proceedings of the 2003 ACM symposium on Software visualization*, pages 77–86. ACM Press.

[Darwin, 1859] Darwin, C. (1859). *On The Origin of Species*. John Murray.

[Dash *et al.*, 1997] Dash, M., Liu, H. et Yao, J. (1997). Dimensionality reduction of unsupervised data. *In Tools with Artificial Intelligence, 1997. Proceedings., Ninth IEEE International Conference on*, pages 532–539.

[Dawkins, 1986] Dawkins, R. (1986). *The Blind Watchmaker*. Norton, San Mateo.

[De Jong et Spears, 1992] De Jong, K. A. et Spears, W. M. (1992). A formal analysis of the role of multi-point crossover in genetic algorithms. *Annals of Mathematics and Artificial Intelligence*, 5:1–26. 10.1007/BF01530777.

[Derthick *et al.*, 1997] Derthick, M., Roth, S. F. et Kolojejchick, J. (1997). Coordinating declarative queries with a direct manipulation data exploration environment. *In Proceedings of the 1997 IEEE Symposium on Information Visualization (InfoVis '97)*, INFOVIS '97, pages 65–72, Washington, DC, USA. IEEE Computer Society.

[Dienes, 2012] Dienes, I. (2012). A meta study of 26 " how much information " studies : Sine qua nons and solutions. *International Journal Of Communication*, 6:874–906.

[Dos Santos et K., 2004] Dos Santos, S. et K., B. (2004). Gaining understanding of multivariate and multidimensional data through visualization. *Computers & Graphics*, 28(3):311–325.

[Dyer, 1990] Dyer, D. (1990). A dataflow toolkit for visualization. *Computer Graphics and Applications, IEEE*, 10(4):60–69.

[Eick, 2000] Eick, S. G. (2000). Visualizing multi-dimensional data. *SIGGRAPH Comput. Graph.*, 34(1):61–67.

[Erbacher et Grinstein, 1994] Erbacher, R. et Grinstein, G. (1994). Issues in the development of 3d icons. *Visualization in Scientific Computing*, pages 109–123.

[Fayyad *et al.*, 2002] Fayyad, U., Wierse, A. et Grinstein, G. (2002). *Info Visualization in Data Mining*. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science & Tech.

[Fekete et Plaisant, 2002] Fekete, J.-D. et Plaisant, C. (2002). Interactive information visualization of a million items. *In Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, INFOVIS '02, pages 117–124, Washington, DC, USA. IEEE Computer Society.

[Fekete *et al.*, 2008] Fekete, J.-D., Wijk, J. J., Stasko, J. T. et North, C. (2008). Information visualization. chapitre The Value of Information Visualization, pages 1–18. Springer-Verlag, Berlin, Heidelberg.

[Ferreira de Oliveira et Levkowitz, 2003] Ferreira de Oliveira, M. C. et Levkowitz, H. (2003). From visual data exploration to visual data mining : a survey. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):378–394.

[Finger et Genesereth, 1985] Finger, J. J. et Genesereth, M. R. (1985). Residue : a deductive approach to design synthesis. Rapport technique STAN-CS-85-1035, Stanford University, Department of Computer Science, Stanford, CA, USA.

[Fisher, 1936] Fisher, R. (1936). The use of multiple measurements in taxonomic problems. *Annals of Human Genetics*, 7(2):179–188.

[Fujishiro *et al.*, 2000] Fujishiro, I., Furuhata, R., Ichikawa, Y. et Takeshima, Y. (2000). Gadget/iv : A taxonomic approach to semi-automatic design of information visualization applications using modular visualization environment. *In Proceedings of the IEEE Symposium on Information Vizualization 2000*, INFOVIS '00, pages 77–83, Washington, DC, USA. IEEE Computer Society.

[Fujishiro *et al.*, 1997] Fujishiro, I., Takeshima, Y., Ichikawa, Y. et Nakamura, K. (1997). Gadget : goal-oriented application design guidance for modular visualization environments. *In Visualization '97., Proceedings*, pages 245–252.

[Gilbert, 1958] Gilbert, E. W. (1958). Pioneer maps of health and disease in england. *The Geographical Journal*, 124(2):pp. 172–183.

[Gnanamgari, 1981] Gnanamgari, S. (1981). *Information presentation through default displays*. Thèse de doctorat, Philadelphia, PA, USA.

[Goldberg, 1989] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st édition.

[Goldberg, 1991] Goldberg, D. E. (1991). Real-coded genetic algorithms, virtual alphabets, and blocking. *Complex Systems*, 5(5):139–167.

[Goldberg et Deb, 1990] Goldberg, D. E. et Deb, K. (1990). A comparative analysis of selection schemes used in genetic algorithms. *In* Rawlins, G. J. E., éditeur : *FOGA*, pages 69–93. Morgan Kaufmann.

[Gong *et al.*, 2009] Gong, D., Yao, X. et Yuan, J. (2009). Interactive genetic algorithms with individual fitness not assigned by human. *J. UCS*, 15(13):2446–2462.

[González et Kobsa, 2003] González, V. et Kobsa, A. (2003). Benefits of information visualization systems for administrative data analysts. *In Proceedings of the Seventh International Conference on Information Visualization*, IV '03, pages 331–336, Washington, DC, USA. IEEE Computer Society.

[Grinstein *et al.*, 2001] Grinstein, G., Trutschl, M. et Cvek, U. (2001). High-dimensional visualizations.

[Guyon, 2006] Guyon, I. (2006). *Feature extraction : foundations and applications*, volume 207. Springer.

[Haber et Mcnabb, 1990] Haber, R. B. et Mcnabb, D. A. (1990). Visualization idioms : a conceptual model for scientific visualization systems. *In Visualization in Scientific Computing*, pages 74–93. IEEE Computer Society Press.

[Haj-Rachid *et al.*, 2010] Haj-Rachid, M., Bloch, C., Ramdane-Cherif, W. et Chatonnay, P. (2010). Différentes opérateurs évolutionnaires de permutation : sélections, croisements et mutations.

[Hauser *et al.*, 2002] Hauser, H., Ledermann, F. et Doleisch, H. (2002). Angular brushing of extended parallel coordinates. *In Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, INFOVIS '02, pages 127–130, Washington, DC, USA. IEEE Computer Society.

[Hayashida et Takagi, 2000] Hayashida, N. et Takagi, H. (2000). Visualized iec : Interactive evolutionary computation with multidimensional data visualization. *In Industrial Electronics Society, 2000. IECON 2000. 26th Annual Confjerence of the IEEE*, volume 4, pages 2738–2743. IEEE.

[Hayashida et Takagi, 2002] Hayashida, N. et Takagi, H. (2002). Acceleration of ec convergence with landscape visualization and human intervention. *Applied Soft Computing*, 1(4):245–256.

[Healey *et al.*, 2008] Healey, C., Kocherlakota, S., Rao, V., Mehta, R. et St Amant, R. (2008). Visual perception and mixed-initiative interaction for assisted visualization design. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):396–411.

[Healey, 1996] Healey, C. G. (1996). Choosing effective colours for data visualization. *In Proceedings of the 7th conference on Visualization '96*, VIS '96, pages 263–270, Los Alamitos, CA, USA. IEEE Computer Society Press.

[Healey *et al.*, 1999] Healey, C. G., Amant, R. S. et Elhaddad, M. S. (1999). Via : A perceptual visualization assistant. *In In 28th Workshop on Advanced Imagery Pattern Recognition (AIPR-99)*, pages 2–11.

[Healey et Enns, 1998] Healey, C. G. et Enns, J. T. (1998). Building perceptual textures to visualize multidimensional datasets. *In Proceedings of the conference on Visualization '98*, VIS '98, pages 111–118, Los Alamitos, CA, USA. IEEE Computer Society Press.

[Healey et Enns, 1999] Healey, C. G. et Enns, J. T. (1999). Large datasets at a glance : Combining textures and colors in scientific visualization. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):145–167.

[Hilbert et López, 2011] Hilbert, M. et López, P. (2011). The world's technological capacity to store, communicate, and compute information. *Science (New York, N.Y.)*, 332(6025):60–5.

[Hoffman, 1977] Hoffman, P. (1977). *Table visualizations : a formal model and its applications*. Thèse de doctorat, UNIVERSITY OF MASSACHUSETTS.

[Hoffman et Grinstein, 2002] Hoffman, P. et Grinstein, G. (2002). A survey of visualizations for high-dimensional data mining. *Fayyad, U., Grinstein, GG, Wierse, A.(eds.). Information Visualization in Data Mining and Knowledge Discovery. Morgan Kaufmann Publishers, San Francisco*, pages 47–82.

[Holland, 1975] Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, USA.

[Horowitz, 1994] Horowitz, D. (1994). Generating rhythms with genetic algorithms. *In Proceedings of the International Computer Music Conference*, pages 142–142. INTERNATIONAL COMPUTER MUSIC ACCOCIATION.

[Hsu et Huang, 2005] Hsu, F. C. et Huang, P. (2005). Providing an appropriate search space to solve the fatigue problem in interactive evolutionary computation. *New Gen. Comput.*, 23(2):115–127.

[Inselberg et Dimsdale, 1990] Inselberg, A. et Dimsdale, B. (1990). Parallel coordinates : a tool for visualizing multi-dimensional geometry. *In Proceedings of the 1st conference on Visualization '90*, VIS '90, pages 361–378, Los Alamitos, CA, USA. IEEE Computer Society Press.

[Iwasaki *et al.*, 2000] Iwasaki, T., Kimura, A., Todoroki, Y., Hirose, Y., Takagi, H. et Takeda, T. (2000). Interactive virtual aquarium (1st report). *5th Annual Conf. of the Virtual Reality Society of Japan*, pages 141–144.

[Jakša et Takagi, 2003] Jakša, R. et Takagi, H. (2003). Image filter design with interactive evolutionary computation. *Proc. of the IEEE International Conference on Computational Cybernetics (ICCC2003), ISBN*, 963(7154):175.

[Janssen, 2001] Janssen, R. (2001). *Computational Image Quality*. Spie Press Monograph. SPIE Press.

[Kamalian *et al.*, 2004] Kamalian, R., Takagi, H. et Agogino, A. (2004). Optimized design of mems by evolutionary multi-objective optimization with interactive evolutionary computation. *In Genetic and Evolutionary Computation–GECCO 2004*, pages 1030–1041. Springer.

[Kamohara *et al.*, 1997] Kamohara, S., Takagi, H. et Takeda, T. (1997). Control rule acquisition for an arm wrestling robot. *In Systems, Man, and Cybernetics, 1997.'Computational Cybernetics and Simulation'., 1997 IEEE International Conference on*, volume 5, pages 4227–4231. IEEE.

[Keim *et al.*, 2008] Keim, D., Andrienko, G., Fekete, J. D., Görg, C., Kohlhammer, J. et Melançon, G. (2008). Visual analytics : Definition, process, and challenges. *In* Kerren, A., Stasko, J., Fekete, J.-D. et North, C., éditeurs : *Information Visualization*, volume 4950 de *Lecture Notes in Computer Science*, pages 154–175. Springer Berlin / Heidelberg.

[Keim *et al.*, 2010] Keim, D., Kohlhammer, J., Ellis, G. et Mannsmann, F. (2010). *Mastering the Information Age. Solving Problems with Visual Analytics*. Eurographics Association.

[Keim et Kriegel, 1996] Keim, D. A. et Kriegel, H.-P. (1996). Visualization techniques for mining large databases : A comparison. *IEEE Trans. on Knowl. and Data Eng.*, 8(6):923–938.

[Keim *et al.*, 2006] Keim, D. A., Mansmann, F., Schneidewind, J. et Ziegler, H. (2006). Challenges in visual data analysis. *In Proceedings of the conference on Information Visualization*, IV '06, pages 9–16, Washington, DC, USA. IEEE Computer Society.

[Keim *et al.*, 2002] Keim, D. A., Müller, W. et Schumann, H. (2002). Visual data mining.

[Keller et Keller, 1993] Keller, P. et Keller, M. (1993). *Visual cues : practical data visualization*. IEEE Computer Society Press.

[Kim et Cho, 2000] Kim, H.-S. et Cho, S.-B. (2000). Application of interactive genetic algorithm to fashion design. *Engineering Applications of Artificial Intelligence*, 13(6): 635–644.

[Kimani *et al.*, 2008] Kimani, S., Catarci, T. et Santucci, G. (2008). A visual data mining environment. *In* Simoff, S., Bohlen, M. et Mazeika, A., éditeurs : *Visual Data Mining*, volume 4404 de *Lecture Notes in Computer Science*, pages 331–366. Springer Berlin / Heidelberg.

[Knight, 2001] Knight, C. (2001). Visualisation effectiveness. *In 2001 International Conference on Imaging Science, Systems, and Technology (CISST 2001)*, numéro Cisst, pages 249–254. Citeseer.

[Kohlhammer *et al.*, 2011] Kohlhammer, J., Keim, D., Pohl, M., Santucci, G. et Andrienko, G. (2011). Solving problems with visual analytics. *Procedia Computer Science*, 7(0):117–120.

[Korf, 1990] Korf, R. E. (1990). Real-time heuristic search. *Artif. Intell.*, 42(2-3):189–211.

[Lai et Chen, 2011] Lai, C. C. et Chen, Y. C. (2011). A user-oriented image retrieval system based on interactive genetic algorithm. *Instrumentation and Measurement, IEEE Transactions on*, 60(10):3318–3325.

[Lange *et al.*, 1995] Lange, S., Schumann, H., Müller, W. et Krömker, D. (1995). Problem-oriented visualisation of multi-dimensional data sets. pages 1–15. Singapore : World Scientific, c1995.

[Lesk, 1997] Lesk, M. (1997). How much information is there in the world ?

[Likert, 1932] Likert, R. (1932). A technique for the measurement of attitudes. *Archives of Psychology*, 22(140):1–55.

[Llorà *et al.*, 2005] Llorà, X., Sastry, K., Goldberg, D. E., Gupta, A. et Lakshmi, L. (2005). Combating user fatigue in igas : partial ordering, support vector machines, and synthetic fitness. *In Proceedings of the 2005 conference on Genetic and evolutionary computation*, GECCO '05, pages 1363–1370, New York, NY, USA. ACM.

[Lund *et al.*, 1998] Lund, H., Miglino, O., Pagliarini, L., Billard, A. et Ijspeert, A. (1998). Evolutionary robotics-a children's game. *In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 154 –158.

[Mackinlay, 1986] Mackinlay, J. (1986). Automating the design of graphical presentations of relational information. *ACM Trans. Graph.*, 5(2):110–141.

[Mahfoud, 1994] Mahfoud, S. M. (1994). Population sizing for sharing methods. *In In Foundations of Genetic Algorithms 3*. Morgan Kaufmann.

[Man *et al.*, 1996] Man, K., Tang, K. et Kwong, S. (1996). Genetic algorithms : concepts and applications [in engineering design]. *Industrial Electronics, IEEE Transactions on*, 43(5):519–534.

[Mazza, 2009] Mazza, R. (2009). *Introduction to Information Visualization*. Springer Publishing Company, Incorporated, 1 édition.

[Miceli, 1994] Miceli, K. D. (1994). Mdv : An intelligent system for multidisciplinary visualization.

[Michalewicz, 1996] Michalewicz, Z. (1996). *Genetic algorithms + data structures = evolution programs*. Springer-Verlag, London, UK, UK.

[Nakamura *et al.*, 1995] Nakamura, K., Fujishiro, I. et Ichikawa, Y. (1995). A knowledge-based system for visualization network design. *In Proc. Intl. Conf. on Virtual System and Multimedia '95*, pages 118–123, Gifu, Japan.

[Niggemann, 2001] Niggemann, O. (2001). *Visual Data Mining of Graph-Based Data*. Phd thesis, University of Paderborn.

[Ocagne, 1885] Ocagne, M. (1885). *Coordonnées parallèles & axiales : méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèles*. Gauthier-Villars.

[Ohsaki *et al.*, 1998] Ohsaki, M., Takagi, H. et Ohya, K. (1998). An input method using discrete fitness values for interactive ga. *J. Intell. Fuzzy Syst.*, 6(1):131–145.

[Peng *et al.*, 2004] Peng, W., Ward, M. et Rundensteiner, E. (2004). Clutter reduction in multi-dimensional data visualization using dimension reordering. *In Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 89–96.

[Pickett et Grinstein, 1988] Pickett, R. M. et Grinstein, G. G. (1988). Iconographic displays for visualizing multidimensional data. *In Systems, Man, and Cybernetics, 1988. Proceedings of the 1988 IEEE International Conference on*, volume 1, pages 514–519.

[Pirolli et Rao, 1996] Pirolli, P. et Rao, R. (1996). Table lens as a tool for making sense of data. *In Proceedings of the workshop on Advanced visual interfaces*, AVI '96, pages 67–80, New York, NY, USA. ACM.

[R. Albertoni et Hauska, 2003] R. Albertoni, A. Bertone, U. D. M. M. et Hauska, H. (2003). Knowledge extraction by visual data mining of metadata in site planning. *In Proceedings of SCANGIS03*, pages 119–130.

[Robertson, 1991] Robertson, P. K. (1991). A methodology for choosing data representations. *IEEE Comput. Graph. Appl.*, 11(3):56–67.

[Rossi, 2006] Rossi, F. (2006). Visual data mining and machine learning. *In ESANN*, pages 251–264.

[Salisbury, 2001] Salisbury, L. D. P. (2001). *Automatic visual display design and creation*. Thèse de doctorat. AAI3022892.

[Schaffer *et al.*, 1989] Schaffer, J. D., Caruana, R. A., Eshelman, L. J. et Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. *In Proceedings of the third international conference on Genetic algorithms*, pages 51–60, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Schulz *et al.*, 2006] Schulz, H.-J., Nocke, T. et Schumann, H. (2006). A framework for visual data mining of structures. *In Proceedings of the 29th Australasian Computer Science Conference - Volume 48*, ACSC '06, pages 157–166, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

[Senay et Ignatius, 1992] Senay, H. et Ignatius, E. (1992). Vista : A knowledge based system for scientific data visualization. Rapport technique.

[Senay et Ignatius, 1994] Senay, H. et Ignatius, E. (1994). A knowledge-based system for visualization design. *IEEE Computer Graphics and Applications*, 14(6):36–47.

[Shneiderman, 1986] Shneiderman, B. (1986). *Designing the user interface : strategies for effective human-computer interaction.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[Shneiderman, 1992] Shneiderman, B. (1992). Tree visualization with tree-maps : 2-d space-filling approach. *ACM Trans. Graph.*, 11(1):92–99.

[Shneiderman, 1996] Shneiderman, B. (1996). The eyes have it : A task by data type taxonomy for information visualizations. *In Proceedings of the 1996 IEEE Symposium on Visual Languages*, VL '96, pages 336–343, Washington, DC, USA. IEEE Computer Society.

[Shneiderman, 2008] Shneiderman, B. (2008). Extreme visualization : squeezing a billion records into a million pixels. *In SIGMOD Conference*, pages 3–12.

[Simoff *et al.*, 2008] Simoff, S. J., Böhlen, M. H. et Mazeika, A. (2008). Visual data mining. chapitre Visual Data Mining : An Introduction and Overview, pages 1–12. Springer-Verlag, Berlin, Heidelberg.

[Sims, 1991] Sims, K. (1991). Artificial evolution for computer graphics. *SIGGRAPH Comput. Graph.*, 25(4):319–328.

[Sims, 1992] Sims, K. (1992). Interactive evolution of dynamical systems. *Toward a Practice of Autonomous Systems : Proceedings of the First European Conference on Artifical Life*, pages 171–178.

[Smith, 1991] Smith, J. R. (1991). Designing biomorphs with an interactive genetic algorithm. *In Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 535–538.

[Smith, 1997] Smith, R. E. (1997). *Handbook of Evolutionary Computation.* Oxford University Press., Bristol, UK, UK, 1st édition.

[Soriano et Gendreau, 1997] Soriano, P. et Gendreau, M. (1997). Fondements et applications des méthodes de recherche avec tabous. *RAIRO. Recherche opérationnelle*, 31(2):133–159.

[Starkweather *et al.*, 1991] Starkweather, T., Mcdaniel, S., Whitley, D., Mathias, K., Whitley, D. et Dept, M. E. (1991). A comparison of genetic sequencing operators. *In Proceedings of the fourth International Conference on Genetic Algorithms*, pages 69–76. Morgan Kaufmann.

[Stolte *et al.*, 2002] Stolte, C., Tang, D. et Hanrahan, P. (2002). Polaris : a system for query, analysis, and visualization of multidimensional relational databases. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):52–65.

[Sywerda, 1989] Sywerda, G. (1989). Uniform crossover in genetic algorithms. *In Proceedings of the third international conference on Genetic algorithms*, pages 2–9, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Takagi, 1998a] Takagi, H. (1998a). Interactive evolutionary computation-cooperation of computational intelligence and human kansei. *Proceedings of the 5th International Conference on Soft Computing and Information/Intelligent Systems*, 1:41–50.

[Takagi, 1998b] Takagi, H. (1998b). Interactive evolutionary computation : System optimization based on human subjective evaluation. *IEEE Int. Conf. on Intelligent Engineering Systems (INES'98)*, pages 17–19.

[Takagi, 2000] Takagi, H. (2000). Active user intervention in an ec search. *5th Joint Conf. on Information Sciences (JCIS2000)*, pages 995–998.

[Takagi, 2001] Takagi, H. (2001). Interactive evolutionary computation : fusion of the capabilities of ec optimization and human evaluation. *Proceedings of the IEEE*, 89(9): 1275–1296.

[Takagi, 2003] Takagi, H. (2003). Humanized computational intelligence with interactive evolutionary computation. *Computational Intelligence : The Experts Speak, edited by DB Fogel and CJ Robinson, Wiley*, pages 207–218.

[Takagi et Kishi, 1999] Takagi, H. et Kishi, K. (1999). On-line knowledge embedding for an interactive ec-based montage system. *In Knowledge-Based Intelligent Information Engineering Systems, 1999. Third International Conference*, pages 280–283. IEEE.

[Takagi et Ohsaki, 1999] Takagi, H. et Ohsaki, M. (1999). Iec-based hearing aid fitting. *In Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on*, volume 3, pages 657–662.

[Takagi et Ohya, 1996] Takagi, H. et Ohya, K. (1996). Discrete fitness values for improving the human interface in an interactive ga. *In Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pages 109–112.

[Terano et Inada, 2003] Terano, T. et Inada, M. (2003). Data mining from clinical data using interactive evolutionary computation. *In* Ghosh, A. et Tsutsui, S., éditeurs : *Advances in Evolutionary Computing*, Natural Computing Series, pages 847–861. Springer Berlin Heidelberg.

[Thomas et Cook, 2005] Thomas, J. et Cook, K. (2005). *Illuminating the Path : Research and Development Agenda for Visual Analytics*. IEEE Computer Society Press.

[Tokui et Iba, 2000] Tokui, N. et Iba, H. (2000). Music composition with interactive evolutionary computation. *Communication*, 17(2):215–226.

[Tufte, 1983] Tufte, E. (1983). *The visual display of quantitative information*. Graphics Press.

[Tufte, 2001] Tufte, E. (2001). *The visual display of quantitative information*. Graphics Press, Cheshire, CT.

[Unemi, 1998] Unemi, T. (1998). A design of multi-field user interface for simulated breeding. *In The Third Asian Fuzzy System Symposium*, Kyungnam University, Masan, Korea.

[Unemi, 2000] Unemi, T. (2000). SBART 2.4 : an IEC tool for creating 2D images, movies and collage. *In Proceedings of 2000 Genetic and Evolutionary Computational Conference workshop program*, Las Vegas, Nevada.

[Upson *et al.*, 1989] Upson, C., Faulhaber, T.A., J., Kamins, D., Laidlaw, D., Schlegel, D., Vroom, J., Gurwitz, R. et van Dam, A. (1989). The application visualization system : a computational environment for scientific visualization. *Computer Graphics and Applications, IEEE*, 9(4):30–42.

[Venturini *et al.*, 1997] Venturini, G., Slimane, M., Morin, F. et Asselin de Beauville, J.-P. (1997). On using interactive genetic algorithms for knowledge discovery in databases. *In* Back, T., éditeur : *Genetic Algorithms : Proceedings of the Seventh International Conference*, pages 696–703, Michigan State University, East Lansing, MI, USA. Morgan Kaufmann.

[Wang, 2007] Wang, L. H. (2007). A comparison of three fitness prediction strategies for interactive genetic algorithms. *J. Inf. Sci. Eng.*, 23(2):605–616.

[Watanabe et Takagi, 1995] Watanabe, T. et Takagi, H. (1995). Recovering system of the distorted speech using interactive genetic algorithms. *In Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on*, volume 1, pages 684 –689 vol.1.

[Wegman, 2003] Wegman, E. J. (2003). Visual data mining. *Statistics in medicine*, 22(9): 1383–1397.

[Wehrend et Lewis, 1990] Wehrend, S. et Lewis, C. (1990). A problem-oriented classification of visualization techniques. *In Proceedings of the 1st conference on Visualization '90*, VIS '90, pages 139–143, Los Alamitos, CA, USA. IEEE Computer Society Press.

[Weicker, 2000] Weicker, K. (2000). An analysis of dynamic severity and population size. *In Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, PPSN VI, pages 159–168, London, UK, UK. Springer-Verlag.

[Wilkins, 2003] Wilkins, B. (2003). *MELD : a pattern supported methodology for visualisation design*. Thèse de doctorat.

[Wong, 1999] Wong, P. C. (1999). Guest editor's introduction : Visual data mining. *IEEE Computer Graphics and Applications*, 19(5):20–21.

[Wright, 1991] Wright, A. (1991). Genetic algorithms for real parameter optimization. *Foundations of genetic algorithms*, 1:205–218.

## Résumé :

Nous nous intéressons dans cette thèse au problème de l'automatisation du processus de choix et de paramétrage des visualisations en fouille visuelle de données. Pour résoudre ce problème, nous avons développé un assistant utilisateur *"VizAssist"* dont l'objectif principal est de guider les utilisateurs (experts ou novices) durant le processus d'exploration et d'analyse de leur ensemble de données.

Nous illustrons, l'approche sur laquelle s'appuie VizAssit pour guider les utilisateurs dans le choix et le paramétrage des visualisations. VizAssist propose un processus en deux étapes. La première étape consiste à recueillir les objectifs annoncés par l'utilisateur ainsi que la description de son jeu de données à visualiser, pour lui proposer un sous ensemble de visualisations candidates pour le représenter. Dans cette phase, VizAssist suggère différents appariements entre la base de données à visualiser et les visualisations qu'il gère. La seconde étape permet d'affiner les différents paramétrages suggérés par le système. Dans cette phase, VizAssist utilise un algorithme génétique interactif qui a pour apport de permettre aux utilisateurs d'évaluer et d'ajuster visuellement ces paramétrages. Nous présentons enfin les résultats de l'évaluation utilisateur que nous avons réalisé ainsi que les apports de notre outil à accomplir quelques tâches de fouille de données.

## Mots clés :

Assistant Utilisateur, Fouille Visuelle de Données, Interaction Visuelle, Interaction Homme-Machine, Algorithme Génétique Interactif.

## Abstract :

In this thesis, we deal with the problem of automating the process of choosing an appropriate visualization and its parameters in the context of visual data mining. To solve this problem, we developed a user assistant *"VizAssist"* which mainly assist users (experts and novices) during the process of exploration and analysis of their dataset.

We illustrate the approach used by VizAssit to help users in the visualization selection and parameterization process. VizAssist proposes a process based on two steps. In the first step, VizAssist collects the user's objectives and the description of his dataset, and then proposes a subset of candidate visualizations to represent them. In this step, VizAssist suggests a different mapping between the database for representation and the set of visualizations it manages. The second step allows user to adjust the different mappings suggested by the system. In this step, VizAssist uses an interactive genetic algorithm to allow users to visually evaluate and adjust such mappings. We present finally the results that we have obtained during the user evaluation that we performed and the contributions of our tool to accomplish some tasks of data mining.

## Keywords :

User Assistant, Visual Data Mining, Visual Interaction, Human-Computer Interaction, Interactive Genetic Algorithm.