Expert Systems with Applications xxx (2012) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

A visual analytics framework for cluster analysis of DNA microarray data

José A. Castellanos-Garzón [a,*], Carlos Armando García [b], Paulo Novais [c], Fernando Díaz [a]

[a] Department of Computer Science, University of Valladolid, University School of Computer Science, Plaza Santa Eulalia 9-11, 40005 Segovia, Spain
[b] Department of Computer Science and Automatics, University of Salamanca, Faculty of Sciences, Plaza de los Caídos s/n, 37008 Salamanca, Spain
[c] Department of Informatics, Universidade do Minho, Campus of Gualtar, 4710-057 Braga, Portugal

ARTICLE INFO

Keywords:
Data mining
DNA-microarrays
Cluster analysis
Visual analytics
Metric spaces
Boundary points
Surface reconstruction

ABSTRACT

Cluster analysis of DNA microarray data is an important but difficult task in knowledge discovery processes. Many clustering methods are applied to analysis of data for gene expression, but none of them is able to deal with an absolute way with the challenges that this technology raises. Due to this, many applications have been developed for visually representing clustering algorithm results on DNA microarray data, usually providing dendrogram and heat map visualizations. Most of these applications focus only on the above visualizations, and do not offer further visualization components to the validate the clustering methods or to validate one another. This paper proposes using a visual analytics framework in cluster analysis of gene expression data. Additionally, it presents a new method for finding cluster boundaries based on properties of metric spaces. Our approach presents a set of visualization components able to interact with each other; namely, parallel coordinates, cluster boundary genes, 3D cluster surfaces and DNA microarray visualizations as heat maps. Experimental results have shown that our framework can be very useful in the process of more fully understanding DNA microarray data. The software has been implemented in Java, and the framework is publicly available at http://www.analiticavisual.com/jcastellanos/3DVisualCluster/3D-VisualCluster.

1. Introduction

Advances in bioinformatics have resulted in large quantities of biological data, which have been made available in various public databases. Biologists often wish to analyse these databases to understand the relationships within them. Traditional approaches are oriented towards discovering evolutionary relationships between genes or proteins by analysing their sequence and structural similarities (Schroeder, Gilbert, Helden, & Noy, 2001).

Visual data exploration provides a means of verifying criteria or hypotheses formulated on evolutionary processes. However, this may also be accomplished using automatic techniques from statistics or machine learning. Visual data exploration usually allows faster data exploration and often provides better results, especially in cases where automatic algorithms fail (Keim, 2002). Usually, this kind of data exploration involves a three-step process: overview, zooming and filtering, and then accessing details on demand (Maletic, Marcus, & Collard, 2002; Shneiderman, 1996).

A single DNA microarray experiment typically evaluates a large number of DNA sequences (genes, cDNA clones, or expressed sequence tags) under several conditions. These conditions may sometimes be a series from throughout a biological process (e.g., the yeast-cellcycle) or a collection of different tissue samples (e.g., normal versus cancerous tissues). In this research, we focus on cluster analysis (Jain & Dubes, 1998, Jain, Murty, & Flynn, 1999; Kaufman & Rousseeuw, 2005; Pedrycz, 2005) done without considering distinctions between different types of DNA sequences, which are generically called genes. Similarly, the conditions refer uniformly to all kinds of experimental conditions, called samples or simply conditions. On the other hand, our DNA microarray analysis focuses on gene-based clustering analysis, but it is also possible to perform sample-based clustering analysis by computing the transpose matrix of the gene expression matrix (Belacel, Wang, & Cuperlovic-Culf, 2006; Jiang, Tang, & Zhang, 2004).

Because visual data exploration is proving to be very useful in the field of *bioinformatics* (Baldin & Brunak, 1998; Pal, Bandyopadhyay, & Murthy, 2006), then combining it with *data mining* techniques for gene expression data can provide knowledge of processes that take place at the cellular level (Berrar, Dubitzky, & Granzow, 2003; Chan & Kasabov, 2004; Geoffrey, Do, & Ambroise, 2004; Han & Kamber, 2006; Jiang et al., 2004; Olson & Delen, 2008; Speed, 2003). Genes with similar expression levels can be grouped according to cellular functions. This approach may disclose information about the functions of many genes for which no previous information is available (Eisen, Spellman, Brown, & Botstein,

* Corresponding author.
E-mail addresses: jantonio_cu@ieee.org (J.A. Castellanos-Garzón), CarlosGarcia@usal.es (C.A. García), pjon@di.uminho.pt (P. Novais), fdiaz@infor.uva.es (F. Díaz).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.08.038

1998; Tavazoie, Hughes, Campbell, Cho, & Church, 1999). Moreover, visual analytics can be used to discover information about the variety of available clustering algorithms. Biologists often have problems selecting the most appropriate algorithm from a given gene expression data set, since no single algorithm is best in every aspect (Jiang et al., 2004).

This paper introduces visual data exploration for aggregating, summarizing and visualizing information generated during interactive cluster analysis on DNA microarray data (Jain & Dubes, 1998, 1999; Kaufman & Rousseeuw, 2005; Pedrycz, 2005). As a result of our framework, we have developed a prototype tool (called *3D-VisualCluster* or 3D-VC for short), which is able to explore dendrograms, clusterings and clusters interactively with different views (Schroeder et al., 2001; Keim, 2002). This prototype uses *principal component analysis* (PCA, Jolliffe, 2002) to reduce data dimensionality to $\mathbb{R}^3$, so that a first approximation of the data distribution can be analysed on a 3D scatter plot. Furthermore, parallel coordinate visualization (Inselberg & Dimsdale, 1990) and DNA microarray data views are presented using a colour scale corresponding to gene expression levels (heat map). Note that connecting multiple visualizations through interactive linking provides more information than considering the visualization components independently.

Additionally, a new method of computing cluster boundary points is defined using the boundary definition of *metric spaces*. With this method, we can display and analyse the boundary genes of a cluster, 3D cluster surfaces, and 3D representations of reference partitions. Together, these can be seen as a new approach for visually comparing clusterings and reference partitions. Finally, the usefulness of our approach to gene expression research is shown by applying several clustering methods to real data.

To conclude, we explain the structure followed by this paper. Section 2 describes the related work, which deals with the existing tools and components of visualization from DNA microarray data, providing a comparative table of visual components that includes our tool. Furthermore, a background on boundary points has also been given. Section 3 introduces the framework for cluster analysis, which provides two algorithms (boundary point and surface reconstruction algorithm) based on the theory of *metric spaces*. Section 4 explains the 3D-VC tool as a result of our framework and gives a methodology to follow (by a set of tasks) to visually analyse the results of the clustering methods. Section 5 outlines the results and discussion of our framework (by the 3D-VC tool) applied to a case study on a public data set of DNA microarray. Section 6 describes the conclusions of our proposal. Appendices A and B respectively give concepts on metric spaces and the theoretical results of Section 3 on which the framework is based.

2. Related work

Cluster analysis is the technique of finding groupings within data in such a way that these groupings make sense in the context of a particular problem. Issues such as feature selection, missing data imputation, outliers and noise treatment, choosing a suitable metrics and the optimal cluster number are still open challenges. Cluster analysis involves three main stages: pre-processing, cluster analysis and cluster validation. The last stage includes the selection and application of cluster validity techniques, generally divided in internal and external measures (Datta & Datta, 2003; Handl, Knowles, & Kell, 2005; Jiang et al., 2004; Yeung, Haynor, & Ruzzo, 2001). Visual validity can be added as a subsequent step after applying the cluster validity measures, and the use of new and different visualization components may help resolve the above mentioned problems while also validating the statistical measures.

Visual validation approaches applied to DNA microarray data have generated several commercially and publicly available tools for data visualization (Eisen, xxxx; Reich, Ohm, Tamayo, Angelo, & Mesirov, 2004; Saeed et al., 2003; Seo & Shneiderman, 2002; Seo & Shneiderman, 2005; Weber et al., 2009). These tools have the following drawbacks. Most of them do not implement dimensionality reduction to represent data on a 3D scatter plot, and the tools that do have this functionality go no further than showing data-points. On the other hand, these tools implement a small number of pre-fitted clustering methods, which raises problems when validating new methods. Moreover, they do not offer a visual interaction framework that is able to validate clustering results according to a reference partition.

Based on the above discussion, we have designed the main features of our prototype so as to compare it with five of the previously mentioned tools. These visualization components (or features) are listed below with comments about their expected impact on the above mentioned problems in cluster analysis.

1. DNA microarray data dendrogram analysis (MDA). This refers to the ability of the tested tools to incorporate dendrogram analysis, in the sense that the user can select and evaluate different cut-off levels in the dendrogram. This can help the users find the most suitable number of clusters, and offers a global view of the constructed cluster hierarchy and the associated heat map for the available data.
2. Scatter plot analysis (SPA). This is the ability of the tool to implement some type of scatter plot analysis on microarray data. This kind of graph depends on the level chosen in dendrogram analysis and can also be useful for validating the cluster number. Moreover, a scatter plot allows us to detect outliers or noise in the data, as well as to validate the inter-cluster distance used by the clustering methods.
3. Microarray parallel coordinates (MPC). Implementation of parallel coordinate analyses on the clusters represented as points and heat maps. With this kind of representation, a user can validate cluster homogeneity (cluster internal quality) by complementing the dendrogram and scatter plot analysis.
4. Microarray statistical analysis (MSA). Whether the tool integrates other statistical data analyses. This component can provide preparatory data analysis, such as feature, metric and clustering method selection.
5. Microarray data clustering methods (MCM). Whether the tool couples clustering methods. This component allows selection of a clustering method and compares the results with other methods. In this way, the method that best fits the data can be chosen, and thus implicitly includes the selection of data and inter-cluster distance.

Based on these points, Table 1 compares our prototype (3D-VC) with tools from Eisen (xxxx), Saeed et al. (2003), Reich et al. (2004), Seo and Shneiderman (2002), Weber et al. (2009), with visualization items as the rows and tool references as the columns. A check mark (✓) is used to indicate that a tool provides the specified property for that row, and "–" indicates that it does not provide this property. Additionally, the value "b-in" means that the tool implements the corresponding features (clustering methods or statistical measures) as built-in functionalities, and hence cannot be extended without a considerable modification of the software. The value "ext" means the opposite of "b-in", since the tool has the ability of extending its functionality by linking with external software components (for example, through an R language API).

Some of these visual components can be improved by extending their functionality, as in the case of Visual Exploration 3D (Weber et al., 2009) and the proposed 3D-VC, which both implement a 3D

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

Table 1
Comparative table of visualization tools versus existing visualization components. Tools: *Cluster and TreeView of* Eisen (xxxx), *TM4 of* Saeed et al. (2003), *GenCluster 2.0 of* Reich et al. (2004), *HCE 3.5 of* Seo and Shneiderman (2002), *Visual Exploration 3D of* Weber et al. (2009) *and 3D-VisualCluster as 3D-VC.*

Features	Tools					
	Eisen (xxxx)	Saeed et al. (2003)	Reich et al. (2004)	Seo and Shneiderman (2002)	Weber et al. (2009)	3D-VC
DNA microarray data dendrogram analysis (MDA)	✓	✓	✓	✓[a]	–	✓
Scatter plot analysis (SPA)	✓[b]	✓[b]	–	✓	✓[d]	✓[bcde]
Microarray parallel coordinates (MPC)	–	–	–	✓	✓	✓
Microarray statistical analysis (MSA)	b-in	b-in	b-in	b-in	b-in	ext
Microarray data clustering methods (MCM)	b-in	b-in	b-in	b-in	b-in	ext

[a] Dendrogram dynamic query control (DDQC). This lets users eliminate uninteresting clusters from a dendrogram and shows the interesting clusters more clearly (Seo & Shneiderman, 2002).
[b] Microarray dimensionally reduction (MDR). The use of data dimensionality reduction techniques for microarray analysis on a scatter plot.
[c] Cluster boundary point (CBP). The ability of the tool to build the boundary of a cluster. That is, determining the cluster boundary genes, which allows the realization of further analysis without taking into consideration the cluster interior genes.
[d] 3D surface reconstruction (3D-SR). Visualization of clusters or other structures in the form of 3D surfaces as an alternative to the existing visualizations.
[e] 3D reference partition representation (3D-RPR). 3D visualization of the surfaces of a reference partition from the analysed data set, and comparison of the reference partition with the clusters of the dendrogram.

surface reconstruction. 3D-VC also improves the scatter plot analysis by implementing cluster boundary computation and a 3D reference partition representation. 3D-VC avoids implementation of clustering methods and statistical measures as built-in functionalities. Instead, it allows linking to other software components by the corresponding API, and hence is easily extendable to newly developed clustering methods or statistical measures through a general purpose language such as R.

2.1. Cluster boundary

Boundary points are data points that are located at the margin of densely distributed data, and are very useful in data mining applications since they represent a subset of the population that possibly belongs to two or more classes (Xia, Hsu, Lee, & Ooi, 2006). Awareness of these points is also useful in classification tasks, since they can potentially be misclassified (Jain et al., 1999).

Cluster boundary reconstruction is of great importance in DNA microarray data analysis, since:

- the boundary genes of a cluster may be representative of the class that this cluster determines, and so interior genes can be discriminated by those boundary genes (Jiang et al., 2004);
- the above approach may disclose additional knowledge about the functions of many genes and raises hypotheses regarding mechanisms in the transcriptional regulatory network (Dhaeseleer, Wen, Fuhrman, & Somogyi, 1998);
- surface reconstruction bounded by boundary gene-points produces shapes and structures that may be meaningful in the context of cluster analysis from gene expression data (*pattern recognition*), that otherwise would not have been possible (Alon et al., 1999; Duda, Hart, & Stork, 2001).

According to Xia et al. (2006), Korte and Vygen (2003), a boundary point p is an object that meets the following conditions:

(a) it is within a dense region $\mathbb{R}$;
(b) there exist a region $\mathbb{R}'$ near p such that $Density(\mathbb{R}') \gg Density(\mathbb{R})$ or $Density(\mathbb{R}') \ll Density(\mathbb{R})$, where the density of a region ($Density(\mathbb{R})$) measures the relative number of points it contains with respect to its size.

Based on these conditions, (Xia et al., 2006) developed a method that uses the technique of reverse k nearest neighbor (RkNN) (Korn & Muthukrishnan, 2000). Using RkNN on a data set requires the execution of a query for each point in the data set. Thus, this is an expensive task with complexity $O(n^3)$, where n is the size of the data set (Tao, Papadias, & Lian, 2004). On one hand, this is a complex method that is applied to a whole data set rather than of a cluster. On the other hand, although this method performs well, it is intended to separate dense regions from less dense ones, and therefore implicitly performs a clustering task.

Our method is oriented to finding boundary points of a given cluster, and thus the goals differs from those of the previous strategy. That is, these strategies are not comparable since our method assumes that the data has previously been grouped into clusters. Furthermore, our method is based on the boundary definition in terms of theoretical notions from *metric spaces*. This way, boundary points focus on the set of points at the closure of a cluster that do not belong to the interior of the cluster, as will be shown later. Finally, our method runs in $O(k^2)$ time, where k is the size of the cluster, which makes it less complex and more suitable for an interactive framework.

3. Metric-based cluster boundaries

The cluster problem can be described as "finding connected regions in a multi-dimensional space containing a relatively high density of points, separated from other such regions by a region containing a low density of points" (Jain & Dubes, 1998), and assumes that the objects to be clustered are represented as points in the space $\mathbb{R}^d$.

Since $\mathbb{R}^d$ with a defined metric (usually the Euclidean distance) is a *metric space* (Krantz et al., 1964; Simmons, 1963), it can be assumed that the gene expression matrix of a DNA microarray is a subspace of $\mathbb{R}^d$. Starting from these concepts, some important definitions and conditions are given before we present an algorithm which computes the boundary points of a given cluster. This algorithm also improves the runtime of the cluster surface reconstruction algorithm implemented by the proposed visual framework, 3D-VC.

3.1. Boundary points of a cluster

We assume that the gene expression matrix is a bounded metric space as stated in Proposition 1 of the theoretical results (see Appendix B). This allows the domain in which we will work to be defined. Next, the cluster problem is accordingly defined on

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

the bounded metric space. Finally, a proposition about whether a cluster is opened or closed is presented. This proposition is useful for defining the problem of cluster boundary points.

In what follows, the set $\mathfrak{G}_n^d$ denotes a bounded subspace of $\mathbb{R}^d$ with an induced metric ρ, where d is the dimension and n is the number of genes. That is, $\mathfrak{G}_n^d$ with the metric ρ is a bounded discrete metric space, called the *gene metric space*. The methods introduced in this section are based on the definitions and proofs given in the section entitled metric spaces (see Appendix A).

We can now state the cluster problem on $\mathfrak{G}_n^d$, which is defined according to Jain and Dubes (1998) as follows.

Definition 1. *Cluster problem on* $\mathfrak{G}_n^d$.

Let $\mathfrak{G}_n^d$ be a metric space. Then a partition $\mathfrak{C}$ of $\mathfrak{G}_n^d$ is a collection of subsets $\{C_1, C_2, \ldots, C_m\}$ of $\mathfrak{G}_n^d$ satisfying:

$$C_i \cap C_j = \emptyset, \quad \forall i, \quad j \in [1,m], i \neq j, \quad \text{and}$$

$$\mathfrak{G}_n^d = \bigcup_{i=1}^{m} C_i, \quad i \in [1,m].$$

A partition $\mathfrak{C}$ of $\mathfrak{G}_n^d$ for which the above definition holds is called a *clustering*, and the subsets C_i are called clusters. The cluster problem consists of finding a suitable clustering of $\mathfrak{G}_n^d$ satisfying a similarity criteria between genes in $\mathfrak{G}_n^d$, which is represented in most cases as a distance function ρ.

This clustering definition is very important since different clustering definitions could yield distinct results for the cluster properties in $\mathfrak{G}_n^d$. Note that in our case, a cluster C of a clustering in a metric space $\mathfrak{G}_n^d$ is a special type of subset in $\mathfrak{G}_n^d$ where no gene of C belongs to a different cluster of C in the same clustering.

From Definitions 9 and 10 in Appendix A, we can prove that a cluster C of $\mathfrak{G}_n^d$ is a closed set (see Proposition 2 in Appendix B). As a consequence of this result, C is not an open set in $\mathfrak{G}_n^d$ and the frontier of C coincides with its boundary (Definitions 8 and 12). This proposition is very important since if C was an open cluster, then it makes no sense to think about its boundary points.

3.2. Multidimensional algorithm to obtain cluster boundaries

The boundary points of a subset in a d-dimensional space are very important in data mining, since analysing these points can reveal information about the problem being addressed (Jiang et al., 2004; Xia et al., 2006). To compute the boundary of a cluster it is necessary to introduce the concept of an *extreme gene*, which is fundamental in searching for cluster boundary points. An ith extreme gene (or extreme gene) of a cluster is a gene (regarded as a vector in $\mathfrak{G}_n^d$) whose ith component is either greater or less than that for the remaining genes in the cluster. The set of all extreme genes of a cluster C is denoted by Exm$C$ (see Definition 13 in Appendix B). An important result from the above statement is that an extreme gene of a cluster belongs to the boundary of such a cluster (as stated in Proposition 3 in Appendix B). Note that if C is a cluster of $\mathfrak{G}_n^d$ then Exm $C \subseteq$ BdC and $Card(\text{Exm}C) \leqslant 2d$ ($Card$ is the cardinality of Exm$C$), where BdC is the boundary of C.

We now introduce an algorithm called *ClusterBoundary* to find the boundary points of a cluster (note that character % in the algorithm indicates a comment). In contrast with other approaches, our boundary definition focuses on the concept of a boundary in a metric space, namely the set of points in the closure of a cluster that do not belong to the interior of the cluster.

3.2.1. ClusterBoundary algorithm

Input: C a cluster in $\mathfrak{G}_n^d$ and ρ the metric defined on $\mathfrak{G}_n^d$.
Output: BdC, the boundary of the cluster C

1. BdC = $\emptyset$;
2. **while** $C \neq \emptyset$ **do**
3. % Module (I)- Find all ith extreme genes in C.
4. % max.exm, min.exm search for the maximal and minimal ith extreme genes
5. % respectively.
6. Exm$C$ = $\emptyset$;
7. **for all** $i \in [1,d]$ **and** $C \neq \emptyset$ **do**
8. Exm$C$:= Exm$C \bigcup \{g_{\succ}^i := \max.\text{exm}(C,i), g_{\prec}^i := \min.\text{exm}(C \setminus \{g_{\succ}^i\}, i)\}$;
9. $C := C \setminus \{g_{\succ}^i, g_{\prec}^i\}$;
10. **endfor**
11. BdC:=Bd$C \bigcup$ Exm$C$;
12. % Module (II)-Compute the centroid (middle point) of Exm$C$.
13. a:= centroid (Exm$C$);
14. % Module (III)-Compute the mid-points between extreme point pairs except
15. % for $g_{\succ}^i$ and $g_{\prec}^j$ where $i = j$.
16. $P_m := \emptyset$;
17. **for all** $i \in [1,d]$ **do**
18. $P_m := P_m \bigcup \{\text{middle.point}(g_{\succ}^i, g) | g \in \text{Exm}C \setminus \{g_{\succ}^i, g_{\prec}^i\}\}$;
19. $P_m := P_m \bigcup \{\text{middle.point}(g_{\prec}^i, g) | g \in \text{Exm}C \setminus \{g_{\succ}^i, g_{\prec}^i\}\}$;
20. **endfor**
21. % Module (IV)-Compute the radius of a ball with interior points in C as follows:
22. choose either r:=min$\{\rho(a,p) | p \in P_m\}$ or r := mean$\{\rho(a,p) | p \in P_m\}$ or
23. $r := \max\{\rho(a,p) | p \in P_m\}$;
24. % choosing one of the above radiuses determines the type of approximation
25. % to the boundary of the cluster
26. % Remove interior points of the ball with center a and radius r
27. $C := C \backslash N(a,r)$;
28. **endwhile**
29. **end**

The ClusterBoundary algorithm is divided in four fundamental modules, which are explained using an example of a cluster in $\mathbb{R}^3$. Module (I) (lines 3–11) carries out a search for extreme points as shown in Fig. 1a. In this figure, the extreme points of a hypothetical cluster are highlighted in red. At each iteration of the algorithm, the cluster boundary is incrementally built from the extreme points (based on Proposition 3 in Appendix B). Module (II) (lines 12–13) computes the centroid of the extreme points, which will be the centre of the interior point ball as shown in Fig. 1b. Note that the lines drawn between the extreme points form an eighth-sided polygon encloses most of the points. Module (III) (lines 14–21) computes the mid-points between each extreme point-pair, except for the pairs $(g_{\succ}^i, g_{\prec}^j)$ where $i = j$. Fig. 1c shows this, as well as the possible radiuses computed from the centroid to the mid-points. Module (IV) (lines 22–28) determines the radius of the ball with the centre already computed in module (II). The radius can be chosen as either the minimal, the mean or the maximal distance between the centroid and the points in the set of mid-points P_m. The option of choosing different radiuses is related to the strategy for constructing the

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

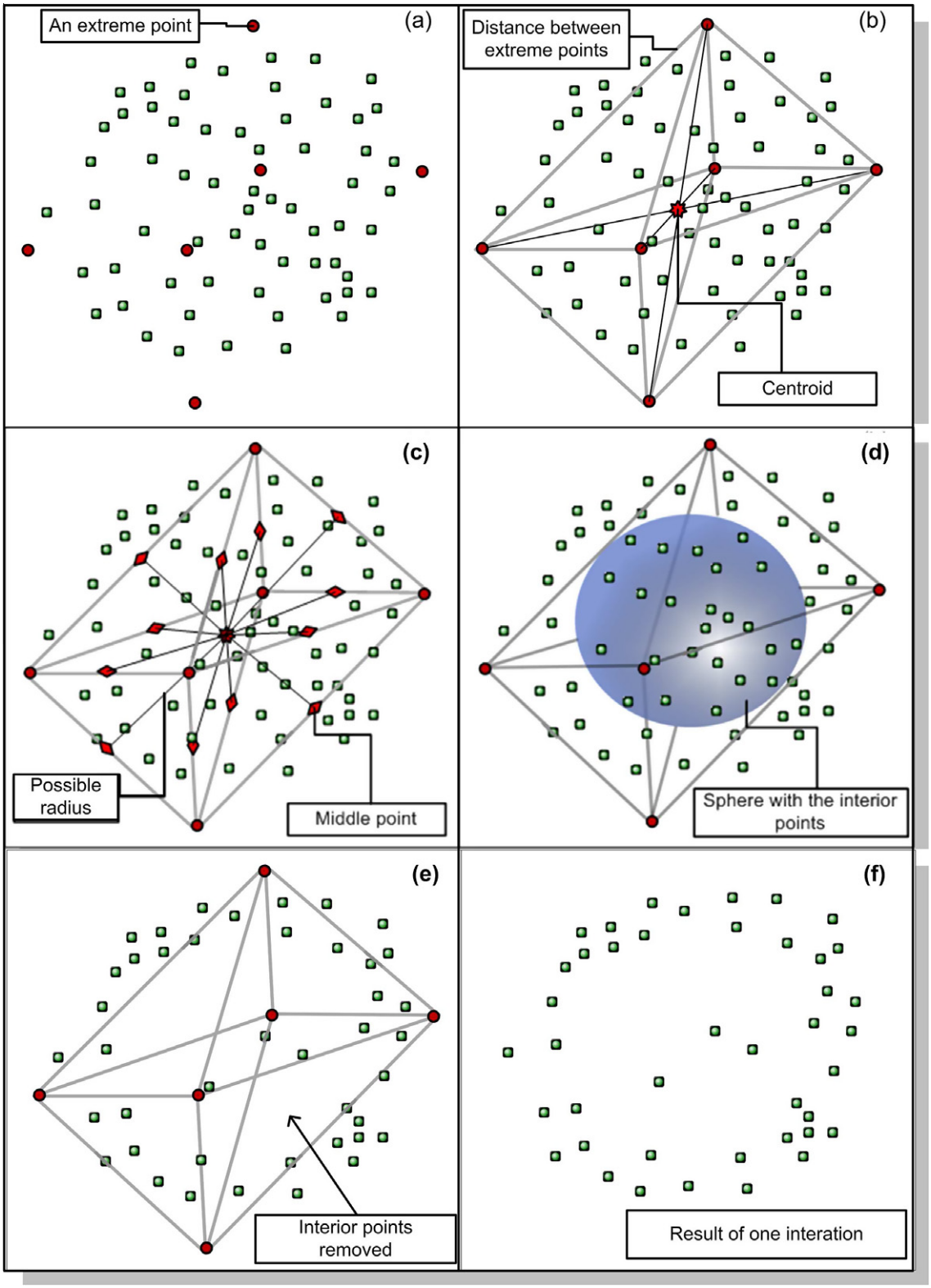

Fig. 1. (a) A 3D gene-point cluster with its extreme points; (b) the centroid of the extreme points and the distances (shown as lines) between the extreme points; (c) the mid-points between the extreme points, and the radiuses computed from the centroid to the mid-points; (d) a sphere (ball) built from the centroid and a chosen radius in Figure c; (e) interior points removed from the sphere; (f) the extreme points from (e) are moved to the set of boundary points, and a new cluster arises.

boundary of the cluster, which can be either conservative (minimal), intermediate (mean) or aggressive (maximal). This strategy depends on how different radiuses can be used to discriminate an increasing number of interior points. The ball (a sphere in $\mathbb{R}^3$) with the chosen radius is shown in Fig. 1d. All interior points of the ball, which are also interior points of the cluster, are removed (see Fig. 1e), whereas the polygon determined by the extreme points is considered to be the current approximation of the boundary.

Fig. 1f shows the result of the algorithm after one iteration, where convergence toward the cluster boundary can be viewed.

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

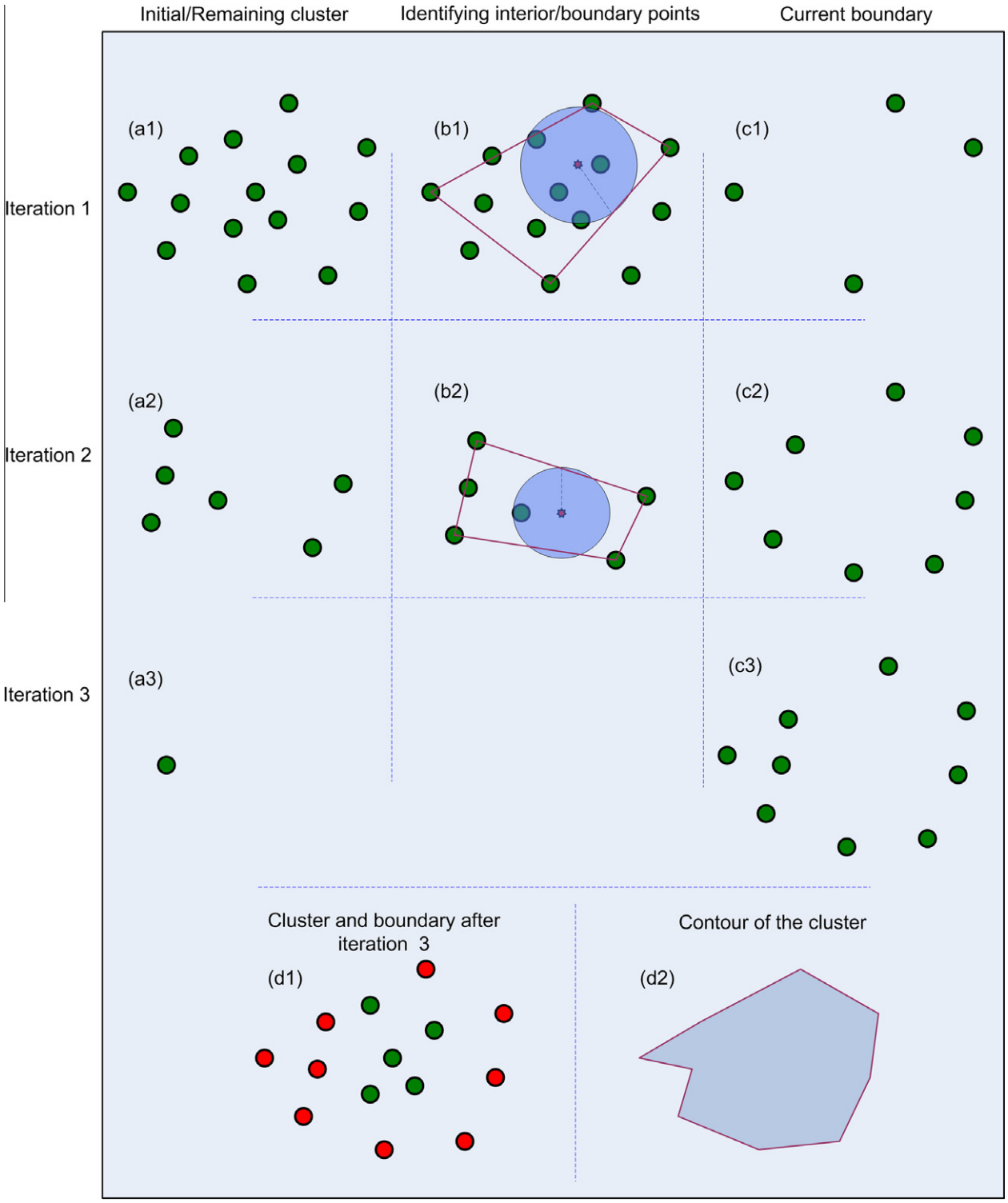

Fig. 2. Steps of the ClusterBoundary Algorithm applied to cluster (a1). The performance of the algorithm using minimum radius is shown under three iterations. Intermediate computations are shown on (b1) and (b2). Remaining cluster and boundary of each iteration are respectively shown on {(a2),(a3)} and {(c1),(c2),(c3)}.

The next iteration takes this new cluster as its input and the whole process is repeated until an empty cluster is obtained. If the size of the input cluster for this algorithm is less than 2*d*, then all the genes are moved to the cluster boundary. A complete execution of the algorithm for a 2D cluster is given in Fig. 2, where it has been shown the fast convergence of the algorithm towards the cluster boundary, which makes this algorithm suitable for an interactive environment. As shown in this figure, the algorithm runs three iterations, where the input cluster for each iteration is shown on views a1, a2 and a3. The process of computing the boundary points is shown on b1 and b2. The boundary computed in each iteration is shown on c1, c2 and c3, where c3 is the output of the algorithm. As the input cluster to iteration 3 just has one gene-point, it is moved to the boundary on c3 and the algorithm ends. Finally, the cluster with its boundary points and its shape is shown on d1 and d2

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

respectively. Note that the algorithm converges towards the boundary of the cluster in a few iterations.

From the performance and the runtime complexity (Aho, Hopcroft, & Ullman, 1983; Gács & Lovász, 1999; Wilf, 1994) of ClusterBoundary, we have that the number of mid-points in P_m is bounded by $2d(d-1)$ (see Proposition 4 in Appendix B). Hence, the runtime of this algorithm in the worst case scenario is $O(k^2)$, where k is the size of the input cluster (for a formal proof see Proposition 5 in Appendix B).

3.3. Surface reconstruction based on boundary points

Surface reconstruction considers the extraction of shape information from a point set. These point sets often contain noise, redundancy and systematic variation arising from the experimental procedure. Therefore, a general approach to reconstructing surfaces is a challenging problem (Hoppe et al., 1994; Heckel, Hamann, & Uva, 1997).

The goal of surface reconstruction methods can be described as follows: *given a set of sample points X assumed to lie on or near an unknown surface U, construct a surface model S approximating U* (Hoppe, DeRose, Duchamp, McDonald, & Stuetzle, 1992, 1994; Marić, Marić, Mijajlović, & Jovanović, 2005).

The proposed algorithm is a modification of the one presented in Berg, Cheong, Kreveld, and Overmars (2008), Boissonnat and Teillaud (2006), which reconstructs convex hulls. In general, since many surfaces are not convex hulls, we provide a version that transforms the basis algorithm to obtain non-convex hulls. As a general schema, the algorithm projects boundary points of a cluster onto the plane, and determines the convex boundary points. Second, the algorithm inserts the remaining non-convex boundary points into the list of convex boundary points. Finally, the algorithm returns an ordered list of boundary points which is used to establish the connectivity of points in a 3D dimensional space.

The first aim of this algorithm is to reconstruct cluster surfaces based on boundary points as a new alternative for cluster visualization. Thus, the input of this algorithm is the cluster boundary found by the ClusterBoundary algorithm. Note that this strategy improves the runtime of our algorithm for surface reconstruction, which is another advantage over the basis algorithm. The second aim is to represent the clusters of a reference partition of a data set as translucent 3D-surfaces, in such a way that genes grouped by a clustering method and also belonging to a cluster of the reference partition can be viewed within the surface of such a cluster in the space. The pseudocode for this algorithm, called BoundaryShape, has been outlined below.

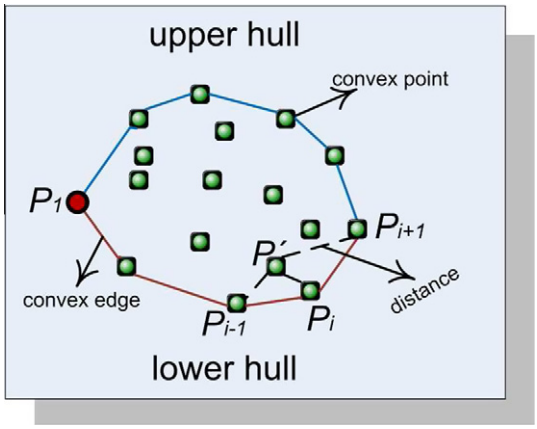

Fig. 3. Convex hull of the boundary points of a cluster.

3.3.1. BoundaryShape algorithm

Input: B, the boundary of a cluster in $\mathfrak{G}_n^d$ and ρ the metric defined on $\mathfrak{G}_n^d$
Output: boundary.idx, index sorted list of genes in B
Require: DPoint function, a generic function to support different criteria for including non-convex points in the boundary
1. $B' :=$ Projection2D (B); % Projects the genes of B onto the plane
2. $B' :=$ Sort$_x(B')$; % Sorts B' by the x coordinate
3. fisrtpos := 1;% Position of the first gene in B'
4. % Initialize the index lists of upper and lower points in B'
5. lupper := llower := NULL;
6. % Find the position of upper convex points
7. **while** there exists (i > firstpos) so that UpperConvexEdge (firstpos, i, B') holds **do**
8. Add (lupper, i); % Adds i to the end of lupper list
9. firstpos := i;
10. **endwhile**
11. % Find the position of the lower convex points
12. fisrtpos := 1; Add (llower, fisrtpos);
13. **while** there exists (i > firstpos) so that LowerConvexEdge (firstpos, i, B') holds **do**
14. Add (llower, i);
15. firstpos := i;
16. **endwhile**
17. Remove the last index of lupper;
18. % Join both index lists of points, inverting lupper
19. boundary.idx := Append (llower, Reverse (lupper));
20. % Take remaining non-convex point indexes out
21. nc := NonconvexIdx (boundary.idx);
22. % Insert each index of nc in boundary.idx suitably
23. **for each** $i \in$ nc **do**
24. % Take the point-index of boundary.idx with the least distance to i out
25. idx := arg$_j$min$\{\rho(i,j) | j \in$ boundary.idx$\}$
26. % Determine whether i is inserted to the left or to the right of idx in boundary.idx
27. % DPoint function returns a value based on a distance criterion for three points
28. **if** DPoint (i, idx, idx - 1) > DPoint (i, idx, idx + 1) **then**
29. Insert (boundary.idx, i, idx + 1)
30. **else**
31. Insert (boundary.idx, i, idx)
32. **endif**
33. **endfor**
34. **end**

The BoundaryShape algorithm is an incremental approach, which has three main parts. Namely, building a list of upper convex point indexes (lines 6–10), a list of lower convex point indexes (lines 11–16), and finally, to insert non-convex point-indexes into the convex indexes list (lines 17–33). So BoundaryShape carries first out some operations such as projecting points onto the $x : y$ plane and then sorts these points by x-coordinate in an ascendent way. Afterwards, it makes a search from left to right (as shown in Fig. 3) for upper convex points and lower convex points (*while* loop, lines 7–10 and 13–16 respectively). Once found these points, they are added to *lupper* and *llower* lists respectively. The UpperConvexEdge and the LowerConvexEdge check whether the edge formed by the vertices *firstpos* and i is an upper or lower convex one, Fig. 3. Therefore, the first *while* loop (lines 7–10) of the

algorithm computes only those convex hull vertices that lie on the upper hull (blue edges in Fig. 3). That is, the part of the convex hull running from the leftmost point P_1 to the rightmost point P_{i+1}, when the points are listed clockwise order as shown in Fig. 3. The second *while* loop (lines 13–16) does the same but for vertices that lie on the lower hull (red edges).

At the end of the algorithm, *boundary.idx* is formed by joining *llower* and the reverse of *lupper*, which allows the convex points to be stored in anti-clockwise order. The set *nc* stores the remaining non-convex points that are finally inserted into *boundary.idx* in the *for* loop (lines 23–33). The value of *idx* in this loop is the point nearest to i (the convex point closest to P' in Fig. 3 is P_i). We then have to decide whether i is inserted to the left or the right of *idx*, which is done using DPoint. DPoint is a generic function that can use a number of distance criteria as shown in Fig. 3. Briefly, the point P' (Fig. 3) can be inserted to the left or right of P_i based on the smaller of the distances between (P',P_{i-1}) and (P',P_{i+1}), the smaller of the distances from P' to the edges (P_i, P_{i-1}) and (P_i, P_{i+1}), the shorter of the paths connecting (P',P_{i-1},P_i) and (P',P_{i+1},P_i), or the smaller of the areas $\Delta\overline{P'P_{i-1}P_i}$ and $\Delta\overline{P'P_{i+1}P_i}$. Note that, in this figure, P' should be inserted to the left of P_i (the area of $\Delta\overline{P'P_{i-1}P_i}$ is less than the area of $\Delta\overline{P'P_{i+1}P_i}$), implying that the edge (P_i,P_{i-1}) is replaced by the edges (P',P_{i-1}) and (P',P_i). This mechanism allows a convex boundary to become non-convex (see also Fig. 2d2).

The information given by the output of the BoundaryShape algorithm is used for the 3D triangulation of the surface that approximates the shape of the cluster, whose boundary was the input to the algorithm.

4. The 3D-VC Framework

In this section, we use DNA microarray data to show the usefulness of 3D-VC in the context of knowledge discovery and visual analytics. 3D-VC can be used for visual validation of the results of clustering methods as an alternative to statistic validity measures (Datta & Datta, 2003; Handl et al., 2005; Jiang et al., 2004; Yeung et al., 2001).The prototype has been written in Java and Java-3D, providing interactive visualizations designed to analyse clustering results, including: microarrays, dendrograms, parallel coordinates and different views of 3D scatter plots.

To show an overview of the 3D-VC tool, a gene expression data set called *cellcycle* given in Yee-Yeung (2001) has been used. The gene expression matrix is composed of 384 genes evaluated under 17 samples, normalized to mean 0 and variance 1. The data set has been classified by Yee-Yeung (2001) into 5 gene clusters to forms a reference partition. Hereinafter, clusters in a reference partition would be called *reference clusters* (in shorthand, r-clusters). Then, to explore the possible clusters of this data set, the *agnes* agglomerative algorithm (Kaufman & Rousseeuw, 2005) has been used. It has been implemented in *R language* (R Development Core Team, 2006), using the packages in Maechler (2005), Chipman and Tibshirani (2006). The Euclidean distance has been used as the metric between data and *mean link* as the inter-cluster distance to generate the

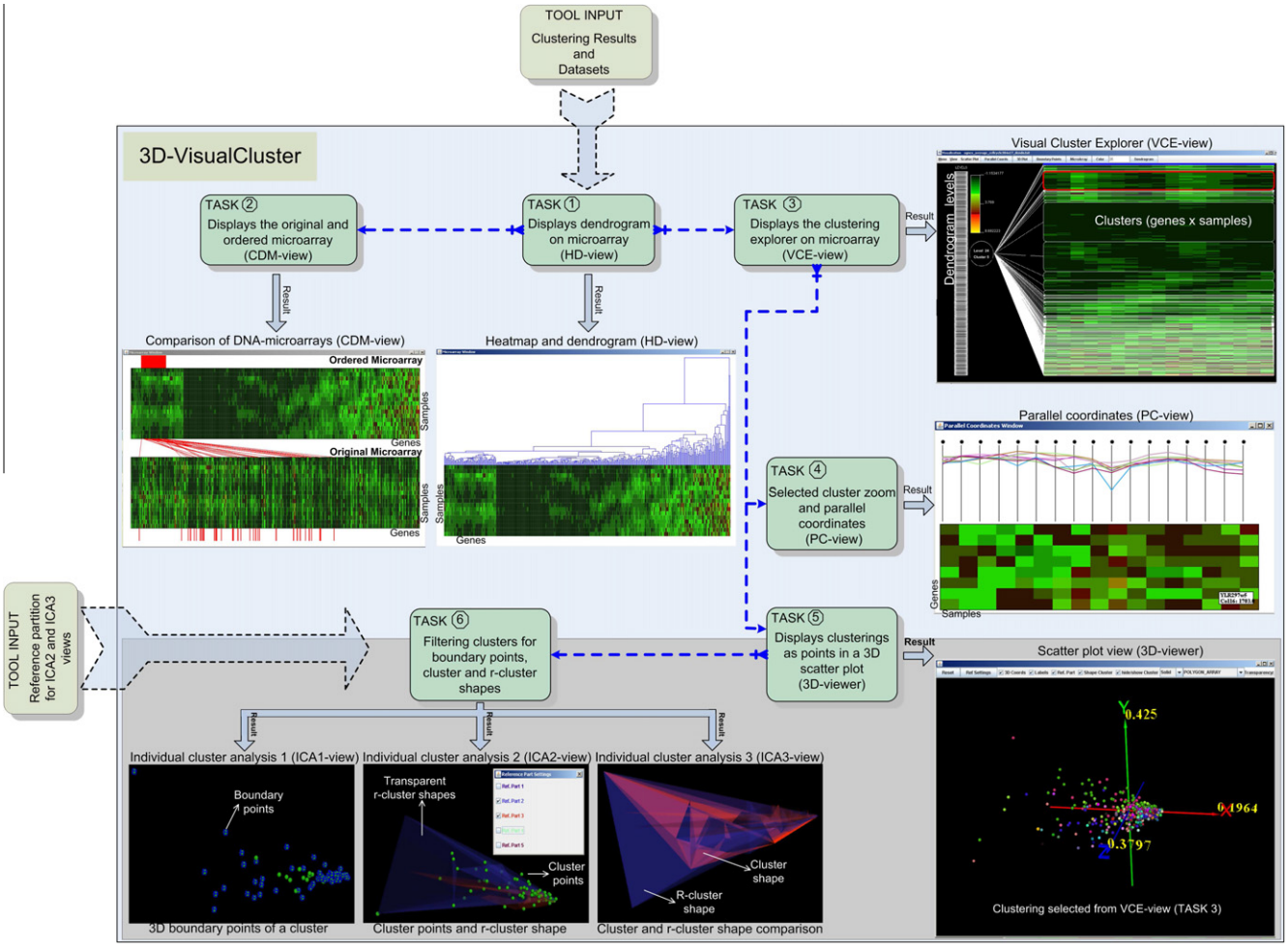

Fig. 4. General view of the tool. There are eight linked views and six tasks of cluster visual analysis showing: microarray, dendrogram and parallel coordinates views at the top of the figure; 3D scatter plot views at the bottom; cluster boundary points, reference partition shapes and cluster shape reconstruction.

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

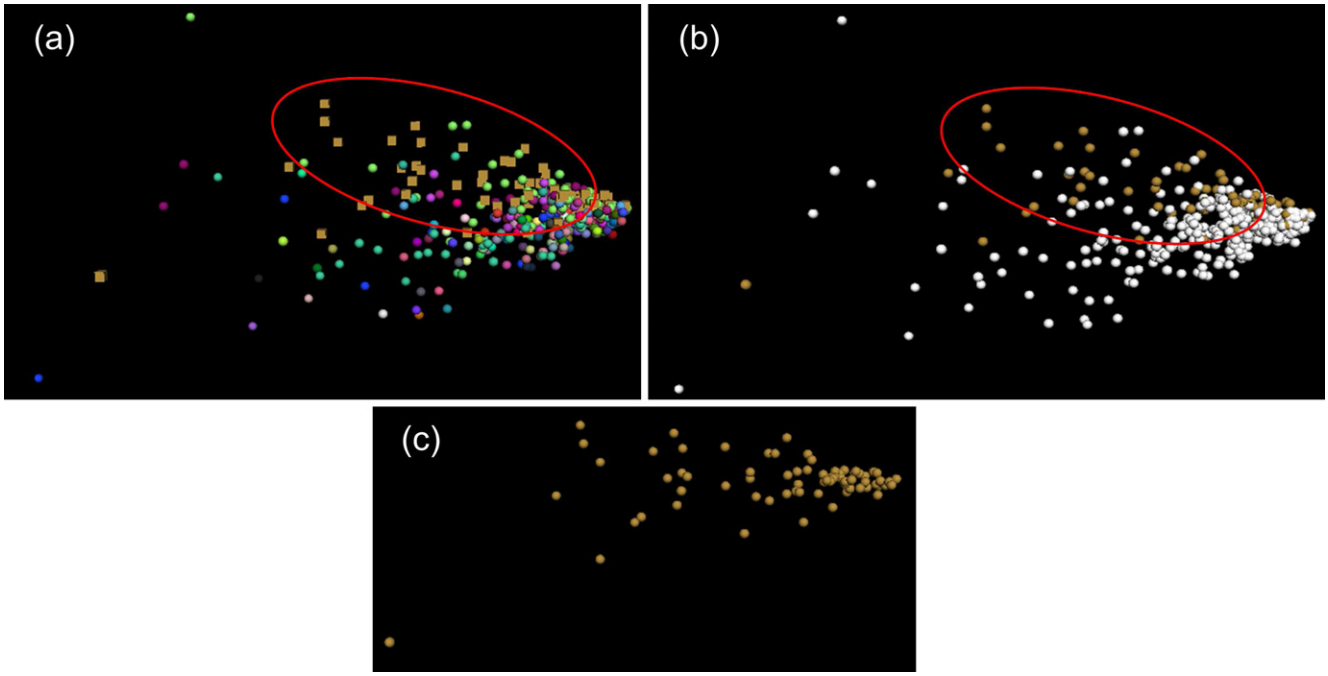

Fig. 5. (a) compares the current cluster by changing the shape of the points (with cubes) vs. remaining points of the data set; (b) compares the current cluster (colored) vs. remaining points of the data set (white circle) and (c) displays only the current cluster. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

output dendrogram. Both, data set and algorithm have just been selected as an example to show the functionality of the tool, so the user can choose any other combination of data sets and algorithms to achieve its purpose.

A general view of our prototype is presented in Fig. 4, which displays a sketch of eight linked views and six tasks of visual cluster analysis. Note that the tasks proposed by 3D-VC in this figure, implicitly states a methodology to follow in the visual analysis and validation of the results of a clustering method. We then explain below, each task of this methodology to describe the 3D-VC tool.

Task 1 in Fig. 4 has as its first goal to read the results of a clustering method applied to a data set of DNA microarray from an external source (in our case, from R language). As a result of the input, this task generates the HD-view, which shows (for this example) the *agnes* dendrogram with its microarray (as a heat map).

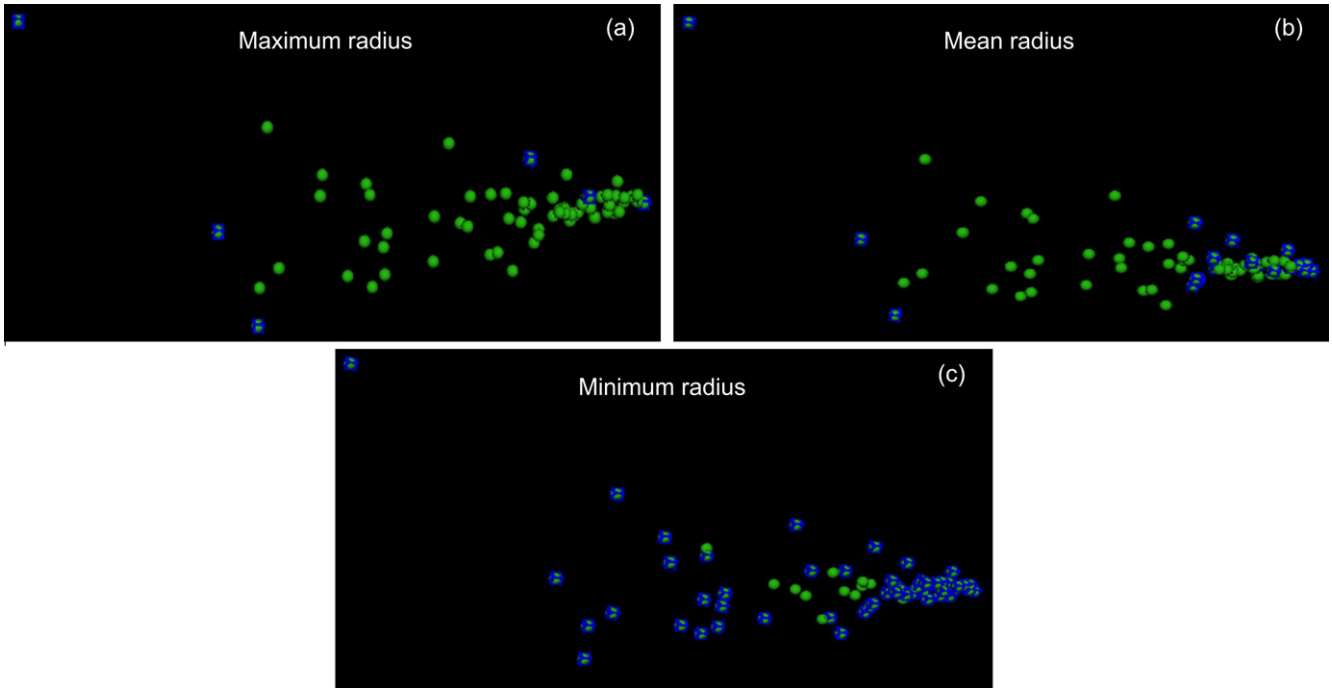

Fig. 6. Shows boundary point options; (a) displays the boundary points computed from maximum radius; (b) boundary points computed from mean radius; and (c) boundary points computed from minimum radius.

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

Table 2
Comparative table of agreement and disagreement of clustering results with respect to the reference partition of *cellcycle*.

Method	Case	#Clusters	Agreement			Disagreement
			JC	RI	ARI	MM
Agnes	A	5	0.20615	0.39175	−0.00455	1.63176
Diana			0.20957	0.38518	−0.00048	1.64056
Eisen			0.22690	0.24387	−0.00153	1.81936
Agnes	B	13	0.16953	0.56706	0.00485	1.37668
Diana			0.13083	0.65473	0.00810	1.22939
Eisen			0.20719	0.39583	−0.00157	1.62622

This view provides an overall analysis of the whole clustering process. The second goal of Task 1 is to generate from the input, two new tasks, 2 and 3, which give way to a new analysis on different views. For its part, Task 2 in Fig. 4 has as a goal to compare the original and the ordered microarray according to the applied clustering method using the CDM-view. Genes in a cluster on the ordered microarray can also be located on the original microarray (through the red lines between both microarrays). This task is optional in the analysis process and is also useful to compare several clustering methods according to the reordering applied by them to the original microarray.

On the order hand, a continuation of Task 1 is Task 3, which has as a goal to locally explore the clusterings (and clusters) of the dendrogram shown by Task 1, through the result given by the VCE-view. Every level (clustering) of the dendrogram can be chosen on the left side in the VCE-view, and the clusters of the chosen level are shown on the right side. Therefore, this view explores in detail each cluster in the dendrogram and moreover, it is a new visual aid for exploring clusters on the microarray.

From Task 3, two possible tasks, Tasks 4 and 5, can be followed. Task 4 has as a goal to carry out a zoom-in of the selected cluster in the VCE-view and show it as a result by including parallel coordinates in the PC-view. Parallel coordinates are added for each sample of genes in the cluster, providing a means of comparison of the cluster quality. Task 5 has as a goal to give a different visualization alternative from the VCE-view. In this case, the clustering selected in the VCE-view (Task 3) has been shown in form of a 3D scatter plot by applying PCA (through the covariance matrix) to reduce the gene space dimensionality to obtain the view of the 3D-viewer. Each cluster in the current clustering has been displayed on the

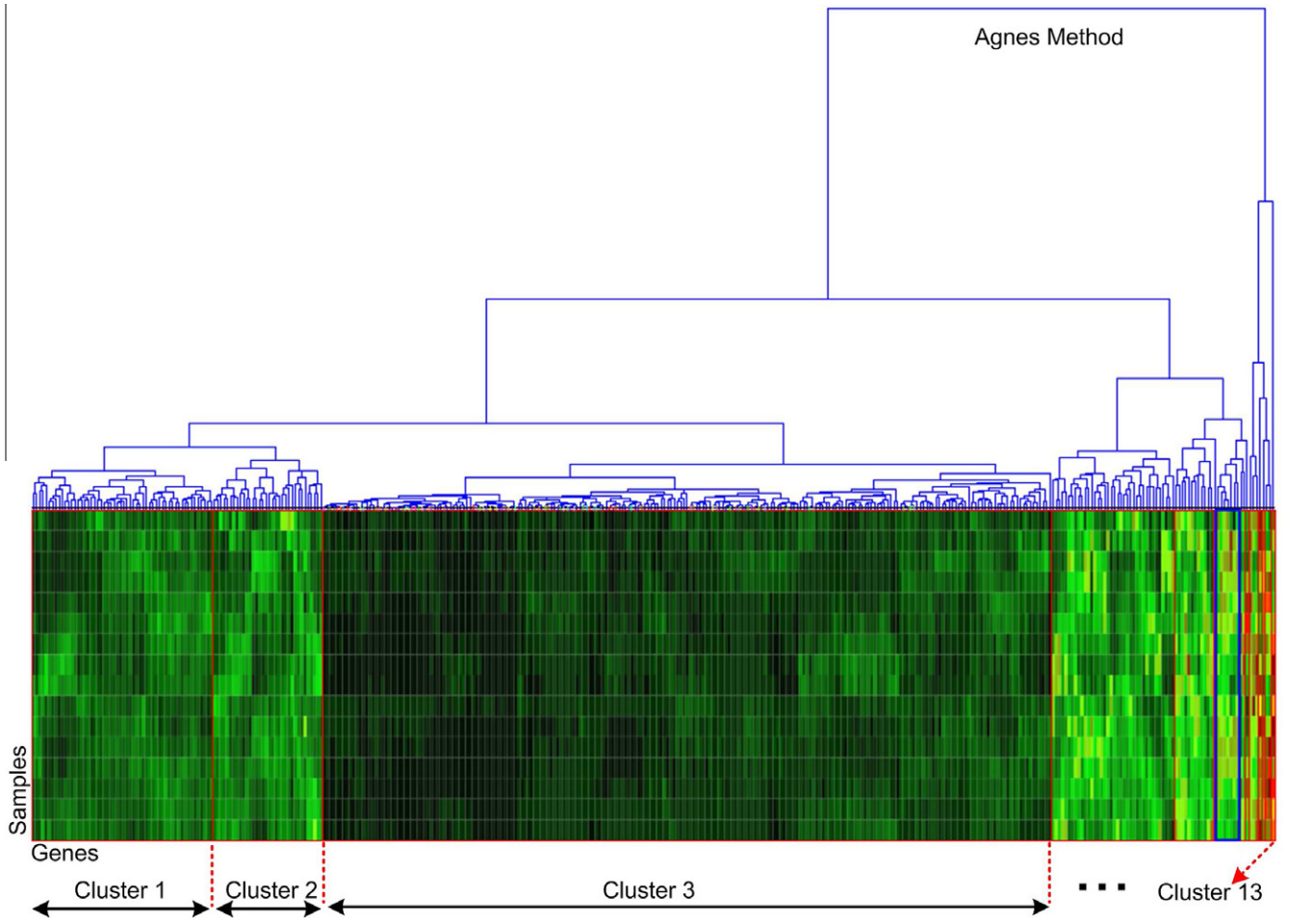

Fig. 7. Dendrogram and microarray of *cellcycle* for *Agnes* method, showing the level of 13 clusters. Clusters are enumerated from left to right.

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

3D-viewer in a different colour. That is, gene-points in the same cluster have the same colour while gene-points in different clusters have different colours. Each cluster on the 3D-viewer can be filtered for separate analysis.

Accordingly to the above, Task 5 generates the last task (task 6) of visual analysis, which has as a goal to filter clusters from the clustering represented on the 3D-viewer for their separately analysis through views ICA1, ICA2 and ICA3. The ICA1-view displays the currently selected cluster with its boundary gene-points, which have been computed by our algorithm ClusterBoundary. Note that before to analyse views ICA2 and ICA3, we need to again read from an external source to the tool, the reference partition of *cellcycle* as shown in Task 6 (Fig. 4). Consequently, in the ICA2-view, we can choose each r-cluster 3D shape (built from our algorithm BoundaryShape) from the reference partition to visually compare it with the current cluster. This way, the r-cluster shape that better match (by visual inspection) with the current cluster can be selected as shows the ICA2-view. Basically, to select the r-cluster shape most similar to the analysed cluster (represented by the cloud of points), we observe the gene-points that fall within the r-cluster translucent shape, the ones on the border of the r-cluster shape and the ones outside of it. On the other hand, the ICA3-view is an alternative with respect to the ICA2-view, where the shape of the analysed cluster is reconstructed (algorithm BoundaryShape) to be compared with the r-cluster shapes. This view provide another way of comparing a cluster with a r-cluster, which allows us to reinforce the assumptions taken into account in the ICA2-view. In our example, the current cluster as both gene-points (ICA2-view) and shape (ICA3-view) visually match with the indicated r-cluster in both views, which means it is a good cluster according to the reference partition. Note that each task in Fig. 4 provides different ways to see and analyse a cluster.

Representing reference partitions from cluster boundary points is one of the main contributions of this paper. This is because there are several statistical indicators for comparing a clustering with a reference partition, but there is no visual approach to validate this comparison. In our framework, each r-cluster of the reference partition is represented by a translucent 3D surface (reconstructed by algorithm ClusterShape), which is generated from the boundary gene-points of each r-cluster using algorithm ClusterBoundary. Thus, the gene-points at the intersection of a cluster with a r-cluster will visually fall within the surface of such a r-cluster (in the space). In that case, one can visually check the degree of agreement between a clustering and a reference partition, or alternatively verify the results of statistical measures. Therefore, it is possible to visually choose the clustering of a dendrogram that best approximates the reference partition.

Other filtering options for the scatter plot can be seen in Fig. 5, where view (a) shows the genes of the current cluster (selected on the VCE-view in Fig. 4) in form of cubes. View (b) shows the same cluster but differentiated by a colour different from the one of the remaining points, and view (c) isolates the cluster. Whereas three types of boundary corresponding to the maximum, mean and minimum radius are displayed in Fig. 6 for the same cluster. Boundary points have been displayed in the form of cubes and have also been computed by algorithm ClusterBoundary. Note that the number of boundary points in Fig. 6 increases from the maximum radius to minimum radius, as different radiuses imply different

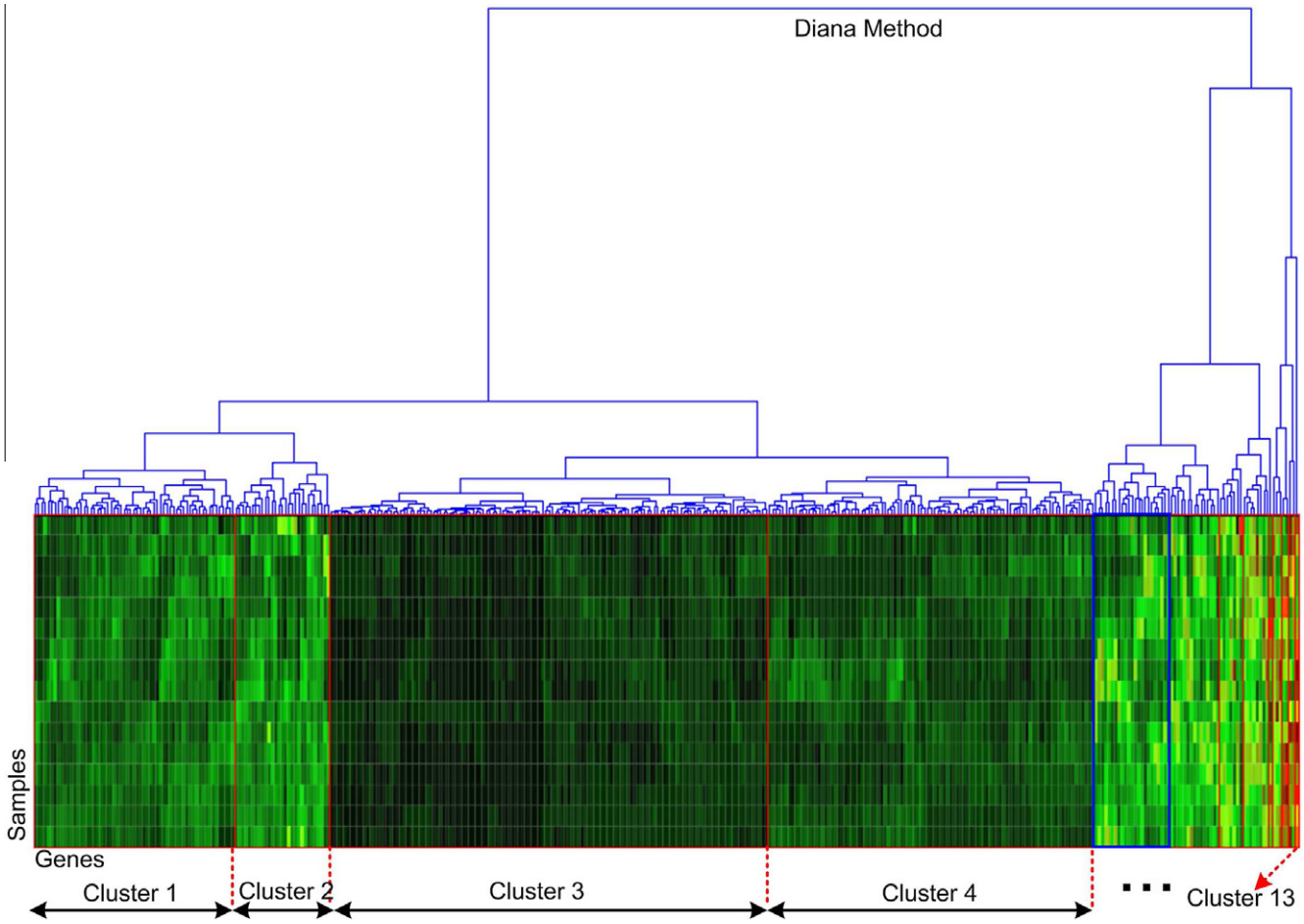

Fig. 8. Dendrogram and microarray of *cellcycle* for the *Diana* method, showing the level of 13 clusters.

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

approximations of the cluster boundary. The radius determines the number of interior points to be removed from a cluster. Thus, the user can choose the type of boundary according to the application. That is, the maximum radius is more suitable to represent 3D surfaces of r-clusters, whereas a smaller radius may be more suitable to represent clusters by their boundary points (or their shapes).

5. A Case Study

This section presents a case study of the *cellcycle* data set (Yee-Yeung, 2001) and its reference partition of 5 r-clusters. The goal of this case study is to show that the visual validation and the numerical validation (cluster validity measures) of the results from the clustering methods are consistent with respect to a reference partition of the given data set. Note that not always visual validation matches the numerical validation, since the numerical validations are based on different assumptions. Then, to reach the previous goal, we first compare the results of three clustering methods with the reference partition for *cellcycle* by mean of statistical indices that measure the similarity degree. Afterwards, we also visually compare the quality of the used statistical indices and show the relationships (numerical and visual) between the reference partition and the clustering results.

To this end, the hierarchical clustering methods *Agnes* and *Diana* from Kaufman and Rousseeuw (2005) and *Eisen* from Eisen et al. (1998), are applied to *cellcycle*. The following statistical indices are used: the Rand index (RI), the Jaccard coefficient (JC), the Minkowski measure (MM) (Halkidi, Batistakis, & Vazirgiannis, 2001; Sokal, 1977), and the Adjusted Rand index (ARI) (Rand, 1971). JC, RI and ARI measure the extent of agreement between two clusterings ($JC \in [0,1]$, $RI \in [0,1]$ and $ARI \leqslant 1$), whereas MM illustrates the proportion of disagreement ($MM \geqslant 0$). When the agreement measure value increases, the agreement grows and when the MM value decreases, the disagreement becomes smaller. All these numerical indices provide evidence that strengthen or weaken the agreement between a clustering and a reference partition, which agrees with the goal of this case study.

Two cases (A and B in Table 2) representing different levels of output dendrogram were chosen to compare each method with the reference partition. Case A (column Case) selects the clustering with 5 clusters (column #Clusters) for each dendrogram, that is, the case where the cluster number in the dendrograms matches that of the reference partition. Case B selects the clustering with 13 clusters for each dendrogram, namely, the clustering that represents a good structural grouping according to the colour intensity levels in the microarray (for example, Fig. 7).The Method column of the table gives the name of the method applied in each case, while the remaining columns contain the values reached for each index in each case. The best values for each index column are underlined.

Note that for Case A, *Agnes* attains the best values for the RI and MM indices, while *Diana* performs best for ARI and *Eisen* for the index JC. This means that *Agnes* has more indices with the best values than the other methods, giving further support to the conclusion that it fits better the reference partition. In Case B, *Diana* gives the best values for the RI, ARI and MM indices, and *Eisen* for JC. Hence, in Case B, *Diana* fits better the reference partition by the

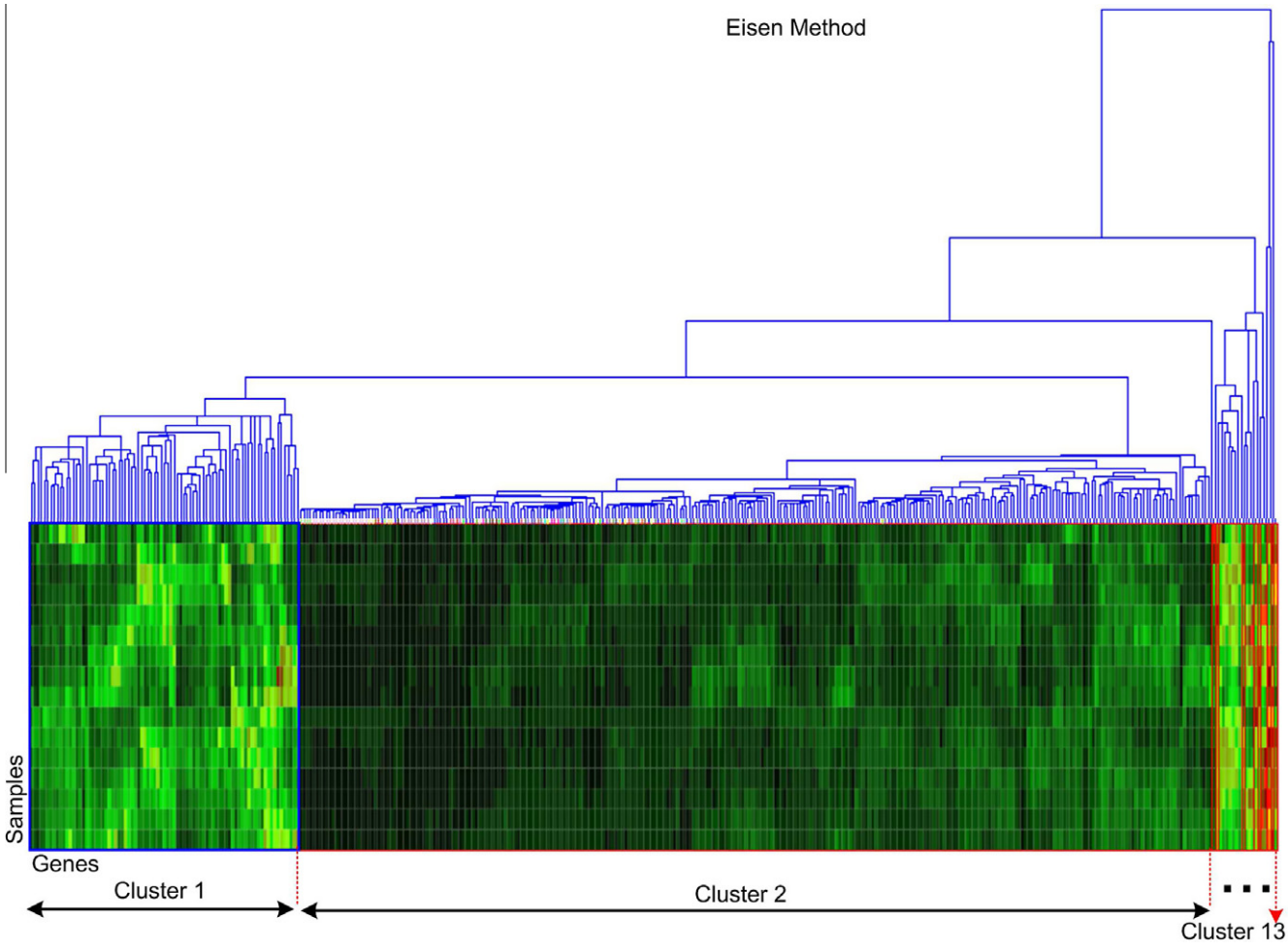

Fig. 9. Dendrogram and microarray of *cellcycle* for *Eisen* method, showing the level of 13 clusters.

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

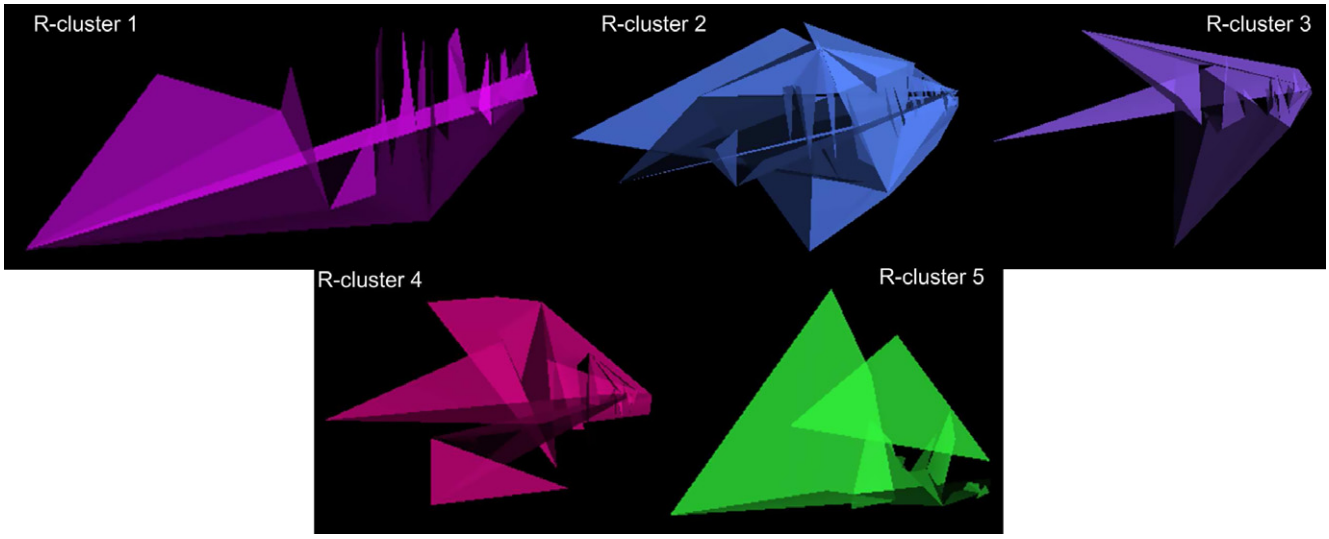

Fig. 10. *cellcycle* reference partition. Each r-cluster has been displayed as a 3D surface.

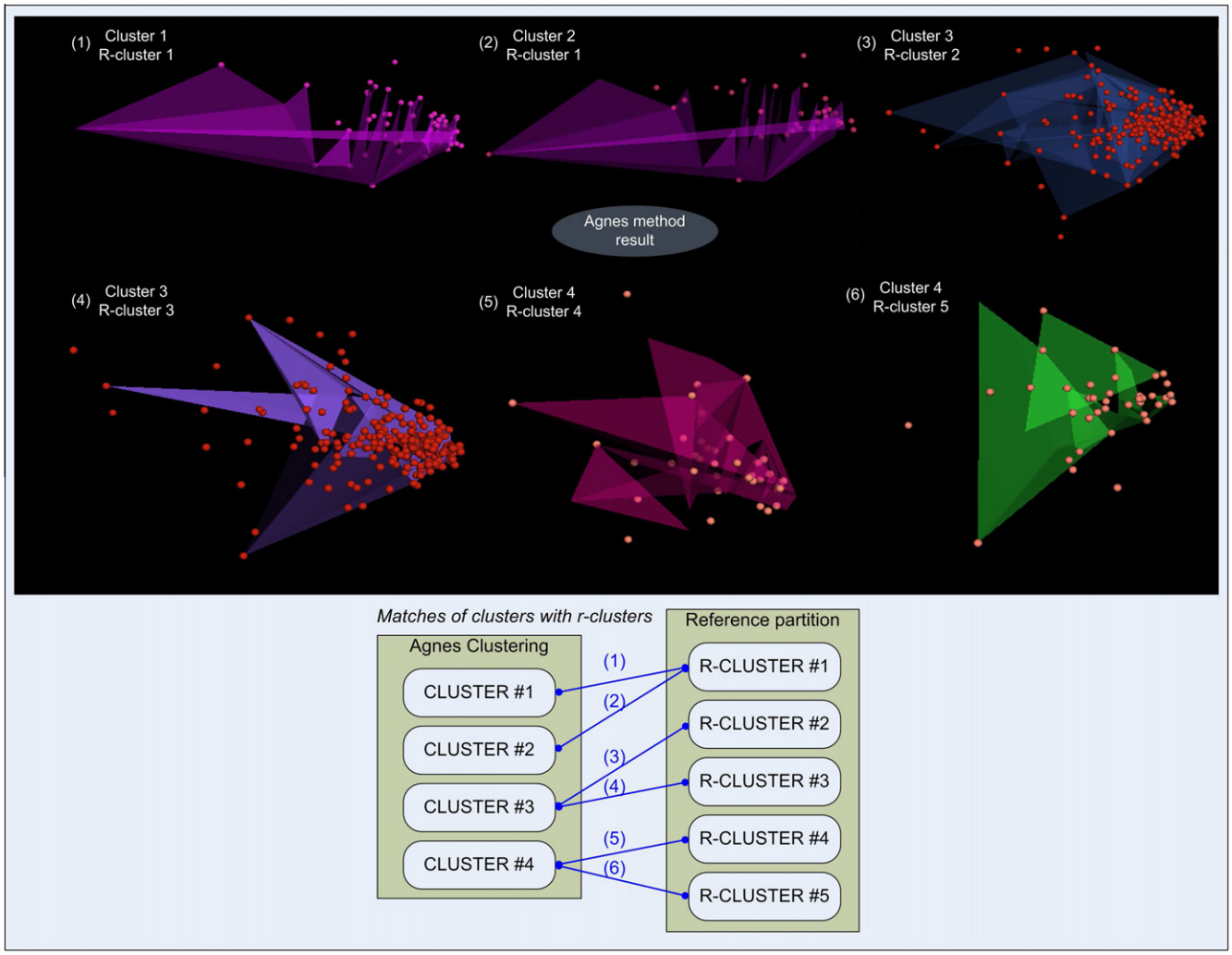

Fig. 11. Shapes of the *cellcycle* reference partition and the clusters that better match with each r-cluster, *Agnes* method.

same conclusion explained for *Agnes* in Case A. In general, except for JC, Case B has better index values than Case A. Thus, the gene distribution with 13 clusters fits better the reference partition than the one with 5 clusters. Finally, since *Diana* in Case B has the higher values on RI, ARI and MM, we can conclude that its result fits better the *cellcycle* reference partition than the others in the whole table.

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

5.1. Visual Check of the Results

As the final part of the case study, we now check that the index results for Case B correspond to the visual representation of the microarray view and with the reference partition of *cellcycle* in the scatter plot view. This allows us a validation process of all the used indices. Firstly, we show the microarray and dendrogram view for each method in Case B in Table 2 by Figs. 7–9, which verify that *Diana* finds a better gene cluster distribution than the other methods. That is, through a visual comparison of the clusters with similar colours, marked on the heat map (red rectangles) in those Figs. 7–9, we can see *Diana* has better division of clusters than *Agnes* and *Eisen*. In contrast, *Eisen* (Fig. 9) shows the worst division of clusters on the heat map.

Secondly, we visually compare the clustering of each method in Case B against the reference partition, aimed at finding the method that yields the clustering most in agreement with the reference partition. A visual representation of the *cellcycle* reference partition is given by Fig. 10, which shows the 3D-surfaces of its five r-clusters, computed from the ClusterShape algorithm. In order to compare the r-clusters with the clusters in Case B in Table 2, a visual inspection has been carried out for the clusters of each method similar to these r-cluster shapes.

Figs. 11–13 show the r-clusters (in the form of 3D-surfaces) overlapped on the clusters (represented as a cloud of gene-points) most similar to them for each method of Case B in Table 2. For *Agnes* in Fig. 11, six visual matches have visually been found for only four clusters with respect to the reference partition. The remaining clusters for *Agnes* are not considered because they are very small and so match with any r-cluster. The process followed to choose the r-cluster shape most similar to a cluster is to first select each r-cluster shape on the cluster and visually analyse that the solid shape of the r-cluster matches the shape of the cloud of gene-points described by the cluster. In addition, we visually check the position of the gene-points of the cluster on the 3D-surface of the r-cluster as explained for ICA2-view in Fig. 4. For *Diana* (Fig. 12), seven matches have been found for only six clusters with respect to the reference partition. For the remaining clusters no matches have been found or the clusters have been very small. For *Eisen* (Fig. 13) six matches have been found for only two clusters. Note that cluster 2 in this figure is repeated four times for different r-clusters. This is because cluster 2 is big and therefore, matches any r-cluster, as also happens for small clusters.

To summarize, we have completed three types of validation for the results of *Agnes, Diana* and *Eisen*. That is, we have validated the results using the indices in Table 2, using the dendrogram and microarray view in Figs. 7–9, and finally by visual comparison with a reference partition in Figs. 11–13. For this experiment, the three tests have given the same results. Namely, in Case B, *Diana* (Table 2) returned the best values for the used indices. Furthermore, *Diana* has shown the best distribution of gene clusters with respect to the microarray and dendrogram visualization, as displayed in Fig. 8, and have also given the best matches (six matches) between the selected clustering and the reference partition for *cellcycle*. On the other hand, *Eisen* achieved the worst results for the three tests (the worst, in the sense that *Eisen* has been less similar to the reference partition than the other methods). In conclusion, we have validated that the *Diana* method has the best performance on *cellcycle* according to its reference partition, and consequently, the clustering that better represent *cellcycle* is that with *13* clusters

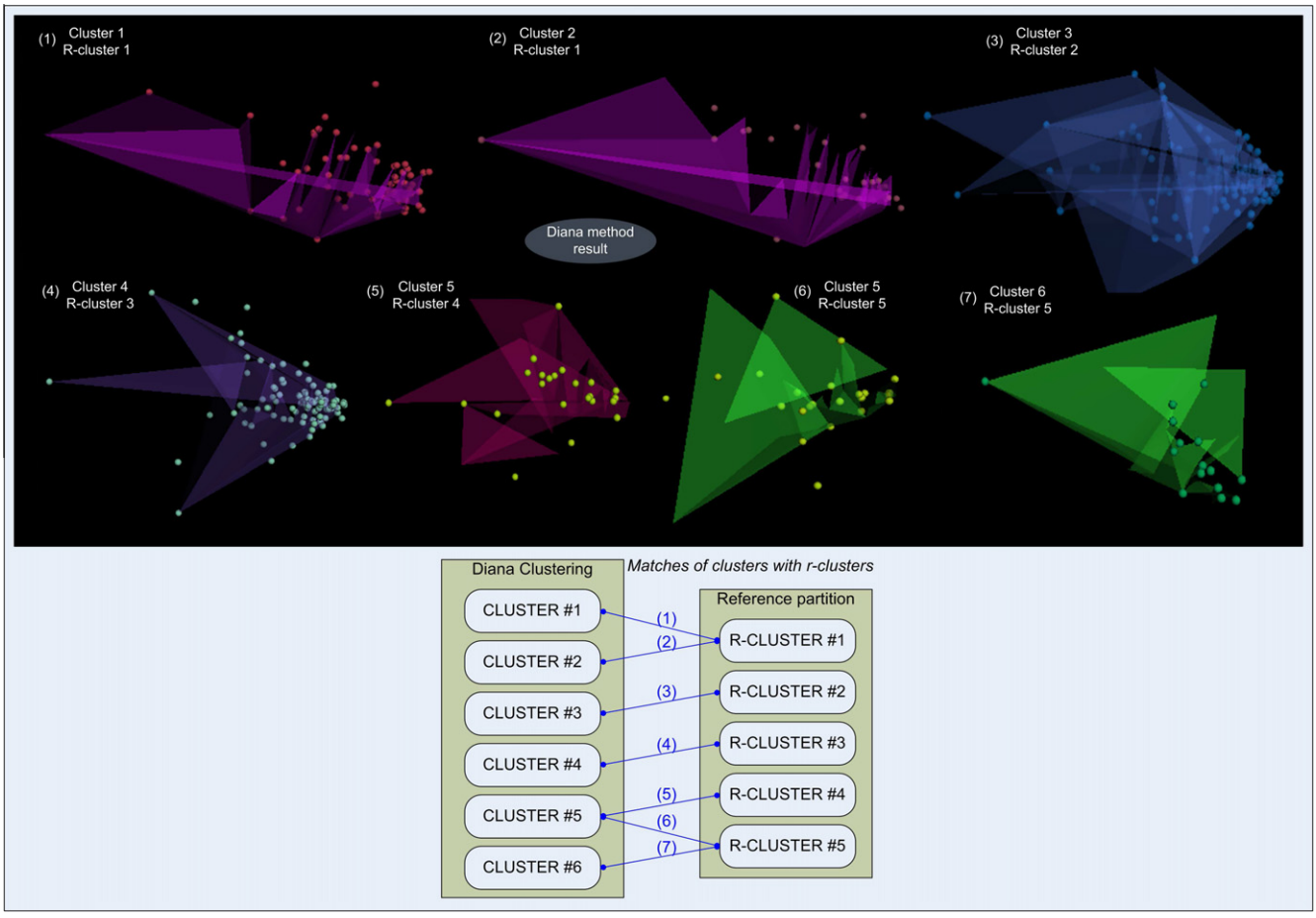

Fig. 12. Shapes of the *cellcycle* reference partition and the clusters that best match with each r-cluster, *Diana* method.

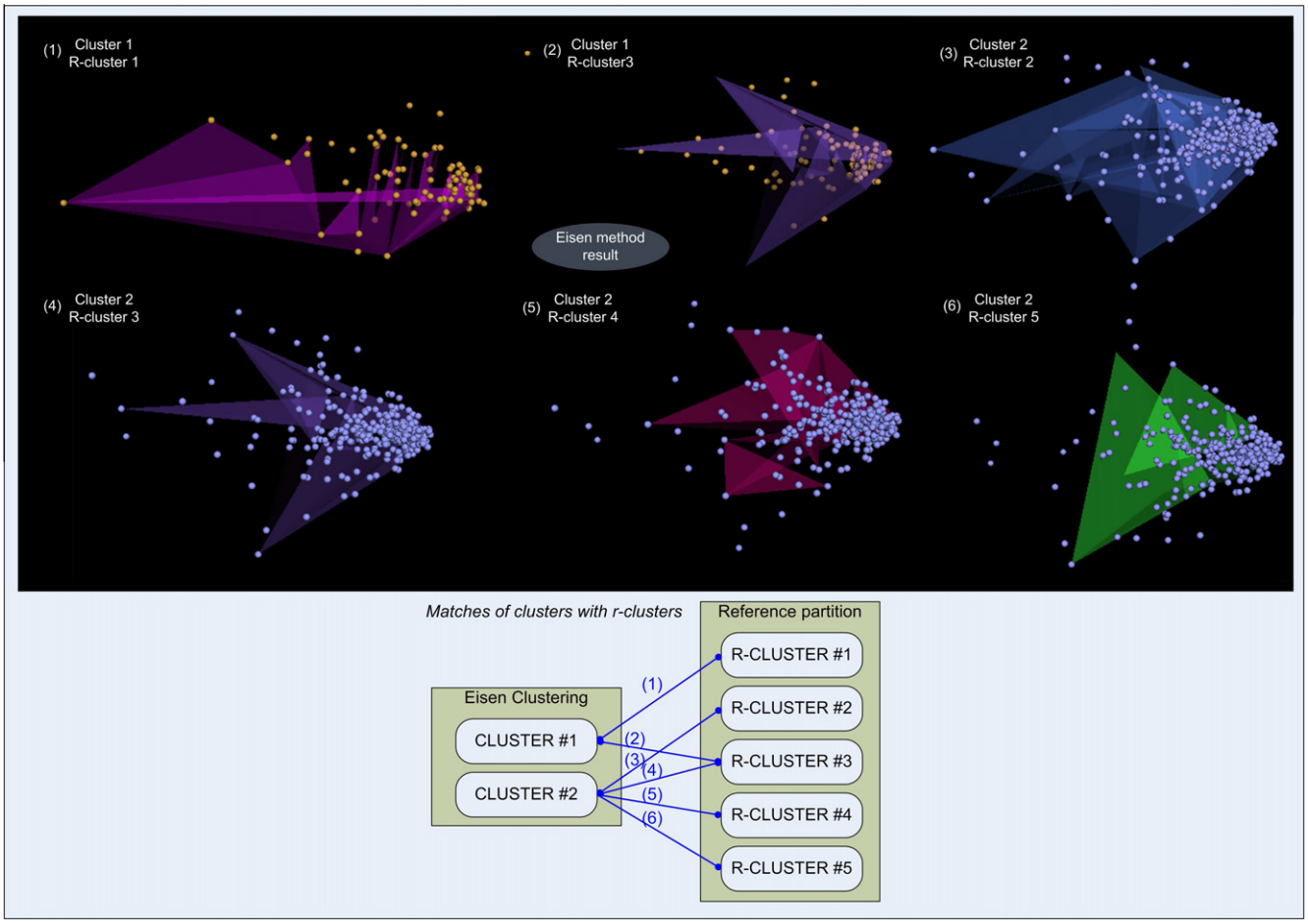

Fig. 13. Shapes of the *cellcycle* reference partition and the clusters that better match with each r-cluster, *Eisen* method.

given by *Diana*. Moreover, the most meaningful clusters in the above clustering were identified according to the reference partition. These clusters are {1,2,3,4,5,6}, as shown in Figs. 8 and 12. Finally, we have that the quantitative and visual analysis agree and additionally, the visual analysis allows us to identify the clusters that match the r-clusters.

6. Conclusions

The main goal of this paper is to provide a visual framework for gene expression data analysis. To do this, we first have developed an approach to DNA microarray data analysis focused on metric spaces, which allows us to use an algorithm to find cluster boundary gene-points. Based on the boundary gene-points, it is possible to approximate the surface of a cluster, as well as represent a reference partition in the space.

These new visualizations are combined with other DNA microarray visualizations, such as heat maps, dendrograms and parallel coordinates, that our framework provides through linked views. This way, the visual analytics process can be achieved. On the other hand, since our prototype is linked using the *R language* (R Development Core Team, 2006), the results of clustering methods implemented in R can be used by 3D-VC. Since R is a programming language widely used in bioinformatics, most clustering methods have been implemented using R in software packages.

Within this approach aimed at cluster analysis for DNA microarray data, we have presented a new method for computing boundary points based on metric spaces. This method is used to render a 3D view of a cluster from its boundary gene-points or surface and a reference partition from the surfaces of its reference clusters, and can thus be used to compare a reference partitions and clusterings from a DNA microarray.

We have decided to combine existing visualizations with the new ideas in our approach, so as to capitalize on the added value gained from interaction between these approaches and thus maximize the benefits to the user. Moreover, the use of 3D-VC states a methodology to follow (through a set of tasks) in the visual analysis process from clustering results. A first prototype of our approach, 3D-VC, has been developed and is available at http://www.analiticavisual.com/jcastellanos/3DVisualCluster/3D-Visual-Cluster. Additional support of 3D-VC such as manuals, examples, images and videos are also publicly available on this website.

7. Authors contributions

JACG designed the proposed framework supervised by FD. JACG and CAG implemented the 3D-VisualCluster tool. JACG wrote the paper, and FD and PN provided the comments and the discussion. All authors read and approved the final manuscript.

8. Competing interests

The authors declare that they have no competing interests.

Acknowledgments

This work has been partially funded by the Spanish Ministry of Science and Innovation, the Plan E from the Spanish Government, the European Union from the ERDF (TIN2009-14057-C03-02).

Appendix A. Metric spaces

Definition 2 (*Metric spaces*). If E is an arbitrary set and $\rho : E \times E \to \mathbb{R}$ a mapping called the distance, then the pair $\langle E, \rho \rangle$ (abbreviated to E) is a metric space if ρ is a metric defined on E. That is, for all points $x, y, z \in E$, the following are satisfied:

1. Positiveness: $\rho(x,y) > 0$ if $x \neq y$, and $\rho(x,x) = 0$.
2. Symmetry: $\rho(x,y) = \rho(y,x)$.
3. Triangle inequality: $\rho(x,z) \leqslant \rho(x,y) + \rho(y,z)$.

Definition 3 (*Bounded metric spaces*). Let E be a metric space consisting with the metric ρ. If there exists a positive number k such that $\rho(x,y) \leqslant k$ for all points $x, y \in E$, then we say that E is a *bounded* metric space.

Definition 4 (*Diameter of a subset of a metric space*). Let A be a subset of a metric space E. Consider the set of non-negative real numbers $\{\rho(x,y) | x \in A, y \in A\}$. If this set is bounded, then it has supremum, denoted $\delta(A)$, which is called the diameter of A.

Definition 5 (*Distance between two subsets*). Let A and B be two subsets of a metric space E. The set of real numbers $\{\rho(x,y) | x \in A, y \in B\}$, is bounded below by zero. Its infimum, denoted by $\rho(A,B)$, is called the distance between A and B.

In the case that A and B can be considered as clusters, other definitions of distance between them can be given (Jain & Dubes, 1998).

Definition 6 (*Subspace of a metric space*). Let E be a metric space with the metric ρ, and let E' be a proper subset of E. Let ρ' be the restriction of ρ to $E' \times E'$, so that $\rho'(x,y) = \rho(x,y)$ for all x and y in E'. Then ρ' is called an *induced metric* and E' with the metric ρ' is called a *subspace of E*.

The gene expression matrix ($m \times n$) of a DNA microarray can be viewed as a subspace of $\mathbb{R}^n$ (see Proposition 1).

Definition 7 (*Balls*). If a is a point in a metric space E with the metric ρ, then the set of all points $x \in E$ such that $\rho(a,x) < r$, where $r > 0$, is called a *ball* (or neighborhood) of centre a and radius r, and is denoted by $N(a,r)$.

Definition 8 (*Open sets*). If A is a subset in a metric space E, the point $a \in A$ is said to be an *interior point* of A if it is the centre of a ball which consists only of points in A. The set of all interior points of A (Int$A$) is called the *interior* of A. If every point of A is interior, then A is called an *open set*.

Definition 9 (*Adherent points*). If A is a subset of a metric space E, the point $a \in E$ is said to be an *adherent point* of A if every ball with centre a contains a point of A.

Definition 10 (*Closure*). If A is a subset of a metric space E, the *closure* of A, denoted by $\overline{A}$, is the set of all adherent points of A. Clearly, $A \subset \overline{A}$. If $A = \overline{A}$, we say that A is *closed*.

Definition 11 (*Exterior points*). A point is said to be an *exterior point* of A if it is an interior point of the complement A^c. The exterior of A (Ext A) is the set of all exterior points of A.

Definition 12. (*Frontier points*). The set $\overline{A} \cap \overline{A^c}$, which is not necessarily empty, is called the *frontier* of A, denoted FrA. The *boundary* of a set A, denoted BdA, is the part of the frontier of A which belongs to A.

Appendix B. Theoretical results

Proposition 1 (*Gene subspace*). *Consider the metric space $\mathbb{R}^d$ and without loose of generality, the Euclidean distance ρ' defined on $\mathbb{R}^d$. Let $X_{n\times d}$ be a gene expression matrix of DNA microarray data, and let $\mathfrak{G}_n^d$ be a set consisting of all row points (genes) in $X_{n\times d}$. Then $\mathfrak{G}_n^d$ is a bounded subspace of $\mathbb{R}^d$ with the induced metric ρ with $\rho(g_x, g_y) = \rho'(g_x, g_y)$, for g_x and g_y genes in $\mathfrak{G}_n^d$.*

Proof. This proposition can be verified from Definitions 2, 3 and 6 in Appendix A. □

Proposition 2. (*Closed cluster*). *If $\mathfrak{G}_n^d$ is a gene metric space with a metric ρ, then every cluster C of $\mathfrak{G}_n^d$ is a closed set (*closed cluster*) in $\mathfrak{G}_n^d$.*

Proof. From Definitions 9 and 10, it is necessary to prove that $C = \overline{C}$. We do this by *reductio ad absurdum*. Suppose that $x \in \mathfrak{G}_n^d$ is an exterior gene of C (see Definition 11) that is also an adherent gene of C. Then $x \in \overline{C}$ and $C \neq \overline{C}$, and moreover, $\forall r \in \mathbb{R}, \exists a \in C, a \in N(x,r)$ (see Definition 7). If, in particular, $r = \rho(x,C)/2$ (see Definition 5), then $\nexists a \in C$ such that $a \in N(x, \rho(x,C)/2)$, and so $x \notin \overline{C}$, which is a contradiction to our previous assumption. Hence, $x \in \overline{C}$, $C = \overline{C}$ and thus C is closed. □

Definition 13 (*Extreme gene*). Let C be a cluster of a gene metric space $\mathfrak{G}_n^d$. A gene $g \in C$ such that $g = (x_1, x_2, \ldots, x_i, \ldots, x_d)$ is said to be an ith *extreme gene* (or an extreme gene) of C, $i \in [1,d]$, if either $x_i \geqslant x_i'$ or $x_i \leqslant x_i' \forall g' = (x_1', x_2', \ldots, x_i', \ldots, x_d') \in C \setminus \{g\}$. An ith extreme gene $g \in C$ is denoted as $g^i_{\succ}$ when $x_i \geqslant x_i'$ or $g^i_{\prec}$ when $x_i < x_i'$ in the above mentioned condition. The set of all extreme genes of C is denoted by Exm$C$.

Proposition 3 (*Extreme genes and cluster boundary*). *If $g = (x_1, x_2, \ldots, x_i, \ldots, x_d)$ is an ith extreme gene of a cluster C of $\mathfrak{G}_n^d$, then $g \in$ BdC.*

Proof. It is enough to prove that $g \in C$ is not an interior gene of C. This is true because in any ball $N(g,r)$ with $r > 0$, we can find a gene $g' = (x_1', x_2', \ldots, x_i', \ldots, x_d')$ with either $x_i' > x_i$ or $x_i' < x_i$ (according to the case) such that g' is an ith extreme gene of $C \cup \{g'\}$ and $\rho(g', g) < r$, $i \in [1,d]$. This implies that $g' \in N(g,r)$ and $g' \notin C$, so g is not an interior gene in C and hence $g \in$ Bd C. □

Proposition 4 (*Cardinality of P_m*). *Since P_m is the set of mid-points, where 2d extreme points are achieved from an iteration of ClusterBoundary, the cardinality of P_m is Card(P_m) = $2d(d-1)$.*

Proof. The number of mid-points computed from extreme points, and that satisfy module (III) of the algorithm, can be expressed as $4\sum_{i=1}^{d-1}(d-i)$. Expanding this expression, the required result is reached. □

Proposition 5 (*Runtime of ClusterBoundary*). *The temporal complexity of ClusterBoundary for computing the cluster boundary in a gene metric space $\mathfrak{G}_n^d$ is $O(k^2)$, where k is the size of the cluster.*

Proof. Module (I) of this algorithm can run in $d \times k$ steps, and each of the other modules can run in d^2 steps. The above results are proven directly for modules (I) and (II), while for modules (III) and (IV) the proof uses Proposition 4. Hence, the order of one iteration of the algorithm is $O(k \times d)$. On the other hand, the number of iterations performed by the algorithm satisfies the inequality $k - (i-1)d \geqslant 0$, where i is the number of iterations. Solving this inequality, we observe that the number of iterations is bounded by $\frac{k}{d} + 1$ and so, the order of ClusterBoundary is $O(k^2)$. □

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038

References

Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (1983). *Data structures and algorithms*. Reading, Massachusetts: Addison-Wesley.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., et al. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences, USA, 96*, 6745–6750.

Baldin, P., & Brunak, S. (1998). *Bioinformatics: The machine learning approach*. Cambridge, MA: MIT Press.

Belacel, N., Wang, Q., & Cuperlovic-Culf, M. (2006). Clustering methods for microarray gene expression data. *OMICS: A Journal of Integrative Biology, 10*(4), 507–531. A special issue on current microarray research.

Berg, M. d., Cheong, O., Kreveld, M. v., & Overmars, M. (2008). *Computational geometry, algorithms and applications* (Third ed.). Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-540-77973-5.

Berrar, D. P., Dubitzky, W., & Granzow, M. (2003). *A practical approach to microarray data analysis*. New York, Boston, Dordrecht, London, Moscow: Kluwer Academic Publishers..

Boissonnat, J. D., & Teillaud, M. (2006). *Mathematics and visualization*. Berlin, Heidelberg: Springer-Verlag.

Chan, Z. S. H., & Kasabov, N. (2004). Gene trajectory clustering with a hybrid genetic algorithm and expectation maximization method. In *IEEE international joint conference on neural networks*, Vol. 3, pp. 1669–1674.

Chipman, H., & Tibshirani, R. (2006). with tsvq code originally from Trevor Hastie, hybridHclust: Hybrid hierarchical clustering, R package version 1.0-1. <http://ace.acadiau.ca/math/chipmanh/hybridHclust>.

Datta, S., & Datta, S. (2003). Comparisons and validation of statistical clustering techniques for microarray gene expression data. *Bioinformatics* (Vol. 19, pp. 459–465). Datta2003.

Dhaeseleer, P., Wen, X., Fuhrman, S., & Somogyi, R. (1998). Mining the gene expression matrix: Inferring gene relationships from large scale gene expression data. *Information Processing in Cells and Tissues*, 203–212.

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern classification* (2nd ed.). New York: Wiley.

Eisen, M., Spellman, T., Brown, P., & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences, USA, 95*, 14863–14868.

Eisen, M. B. (2002). Cluster 2.20 and treeview 1.60, Eisen Lab. <http://rana.lbl.gov/EisenSoftware.htm>.

Gács, P., & Lovász, L. (1999). Complexity of algorithms. In *Lecture notes*. Berlin, Heidelberg: Springer-Verlag.

Geoffrey, J. M., Do, K. A., & Ambroise, C. (2004). *Analyzing microarray gene expression data*. Hoboken, New Jersey: John Wiley & Sons Inc..

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On clustering validation techniques. *Intelligent Information Systems Journal*.

Handl, J., Knowles, J., & Kell, D. B. (2005). *Computational cluster validation in post-genomic data analysis*. Oxford University Press. Vol. 21, pp. 3201–3212.

Han, J., & Kamber, M. (2006). *Data mining: Concepts and techniques*. Elsevier Inc..

Heckel, B., Hamann, B., & Uva, A. E. (1997). Cluster-based generation of hierarchical surface models. In *Scientific visualization conference* (pp. 105–114). IEEE Computer Societyhttp://doi.ieeecomputersociety.org/10.1109/DAGSTUHL.1997.10036 .

Hoppe, H. (1994). Surface reconstruction from unorganized points. Ph.D. thesis, University of Washington, Department of Computer Science and Engineering.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., & Stuetzle, W. (1992). Surface reconstruction from unorganized points. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques* (pp. 71–78), ACM, New York, NY, USA. http://dx.doi.org/10.1145/133994.134011.

Hoppe, H., DeRose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schweitzer, J., & Stuetzle, W. (1994). Piecewise smooth surface reconstruction. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (pp. 295–302) ACM, New York, NY, USA. http://dx.doi.org/10.1145/192161.192233.

Inselberg, A., & Dimsdale, B. (1990). Parallel coordinates: A tool for visualizing multi-dimensional geometry, In VIS '90: *Proceedings of the 1st conference on Visualization '90* (pp. 361–378).

Jain, A. K., & Dubes, R. C. (1998). *Algorithms for clustering data*. Englewood Cliffs, New Jersey: Prentice Hall, p. 07632.

Jain, A. K., Murty, N. M., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys, 31*(3), 264–323.

Jiang, D., Tang, C., & Zhang, A. (2004). Cluster analysis for gene expression data: A survey. *IEEE Transactions on Knowledge and Data Engineering, 16*(11), 1370–1386.

Jolliffe, I. T. (2002). *Principal component analysis*. Springer-Verlag.

Kaufman, L., & Rousseeuw, P. J. (2005). *Finding groups in data. An introduction to clustering analysis*. Hoboken, New Jersey: John Wiley & Sons, Inc..

Keim, D. A. (2002). Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics, 8*, 1–8.

Korn, F., & Muthukrishnan, S. (2000). Influence sets based on reverse nearest neighbor queries. In *Proceedings ACM SIGMOD* (pp. 201–212).

Korte, B., & Vygen, J. (2003). DHC: A density-based hierarchical clustering method for time series gene expression data. In *Proceedings of the third IEEE symposium on bioinformatics and bioengineering (BIBE)*.

Krantz, S. G., Saltman, D., Sallinger, D., & Stern, R. (1964). Metric Spaces, American Mathematical Society, Library of Congress Cataloging-in-Publication Data.

Maechler, M. (2005). Based on S original by P. Rousseeuw, Anja.Struyf@uia.ua.ac.be, Mia.Hubert@uia.ua.ac.be, initial R port by Kurt.Hornik@R-project.org, cluster: Cluster Analysis Extended Rousseeuw et al., R package version 1.10.2.

Maletic, J. I., Marcus, A., & Collard, M. L. (2002). A task oriented view of software visualization. In*IEEE workshop of visualizing software for understanding and analysis (VISSOFT)* Vol. 26, pp. 32–40.

Marić, M., Marić, F., Mijajlović, Z., & Jovanović, B. (2005). Automatic construction of surface model, Tech. rep., School of Mathematics, University of Belgrade, Belgrade, Serbia and Montenegro.

Olson, D. L., & Delen, D. (2008). *Advanced data mining techniques*. Berlin, Heidelberg: Springer-Verlag.

Pal, S. K., Bandyopadhyay, S., & Murthy, C. A. (2006). Evolutionary computation in bioinformatics: A review. *IEEE Transactions on Systems, Man and Cybernerics, Part C, 36*, 601–615.

Pedrycz, W. (2005). *Knowledge-based clustering*. Hoboken, New Jersey: John Wiley & Sons, Inc..

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association, 66*, 846–850.

R Development Core Team. (2006). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0. <http://www.R-project.org>.

Reich, M., Ohm, K., Tamayo, P., Angelo, M., & Mesirov, J. P. (2004). Genecluster 2.0: An advanced toolset for bioarray analysis. *Bioinformatics, 20*(11), 1797–1798.

Saeed, A., Sharov, V., White, J., Li, J., Liang, W., Bhagabati, N., et al. (2003). open-source system for microarray data management and analysis. *Biotechniques, 34*, 374–378.

Schroeder, M., Gilbert, D., Helden, J. v., & Noy, P. (2001). *Approaches to visualization in bioinformatics: From dendrograms to space explorer*. *Information Sciences* (Vol. 139). Elsevier.

Seo, J., & Shneiderman, B. (2002). Interactively exploring hierarchical clustering results. *Computer, 35*(7), 80–86. http://dx.doi.org/10.1109/MC.2002.1016905.

Seo, J., & Shneiderman, B. (2005). A knowledge integration framework for information visualization. In *From integrated publication and information systems to information and knowledge environments*. *LNCS* (Vol. 3379/2005, pp. 207–220). Berlin/Heidelberg: Springer.

Shneiderman, B., (1996). The eyes have it: A task by data type taxonomy for information visualizations, In *IEEE symposium on visual languages*. Vol. 0, p. 336. Available from: <http://doi.ieeecomputersociety.org/10.1109/VL.1996.545307>.

Simmons, G. F. (1963). *Introduction to topology and modern analysis*. McGRAW-HILL Book Company, Inc..

Sokal, R. R. (1977). *Clustering and classification: Background and current directions, classification and clustering*. Academic Press.

Speed, T. (2003). *Statistical analysis of gene expression microarray data*. Chapman & Hall/CRC Press LLC.

Tao, Y., Papadias, D., & Lian, X. (2004). Reverse KNN search in arbitrary dimensionality. In *Proceedings international conference very large data bases* (pp. 744–755).

Tavazoie, S., Hughes, D., Campbell, M. J., Cho, R. J., & Church, G. M. (1999). Systematic determination of genetic network architecture. *Nature Genetics*, 281–285.

Weber, G. H., Rübel, O., Huang, M.-Y., DePace, A. H., Fowlkes, C. C., Keränen, S. V., et al. (2009). Visual exploration of three-dimensional gene expression using physical views and linked abstract views. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 6*, 296–309.

Wilf, H. S. (1994). Internet Edition. <http://www.cscis.upenn.edu/wilf>.

Xia, C., Hsu, W., Lee, M. L., & Ooi, B. C. (2006). Border: Efficient computation of boundary points. *IEEE Transactions on Knowledge and Data Engineering, 18*, 289–303.

Yee-Yeung, K. (2001). Clustering analysis of gene expression data, Ph.D. thesis, University of Washington.

Yeung, K. Y., Haynor, D. R., & Ruzzo, W. L. (2001). Validating clustering for gene expression data. *Bioinformatics* (Vol. 17, pp. 309–318). Oxford University.

Please cite this article in press as: Castellanos-Garzón, J. A., et al. A visual analytics framework for cluster analysis of DNA microarray data. *Expert Systems with Applications* (2012), http://dx.doi.org/10.1016/j.eswa.2012.08.038