Kern *et al. BMC Bioinformatics* (2017) 18:406
DOI 10.1186/s12859-017-1813-7

BMC Bioinformatics

SOFTWARE Open Access

Interactive visual exploration and refinement of cluster assignments

Michael Kern[1,2], Alexander Lex[1*], Nils Gehlenborg[3] and Chris R. Johnson[1]

Abstract

Background: With ever-increasing amounts of data produced in biology research, scientists are in need of efficient data analysis methods. Cluster analysis, combined with visualization of the results, is one such method that can be used to make sense of large data volumes. At the same time, cluster analysis is known to be imperfect and depends on the choice of algorithms, parameters, and distance measures. Most clustering algorithms don't properly account for ambiguity in the source data, as records are often assigned to discrete clusters, even if an assignment is unclear. While there are metrics and visualization techniques that allow analysts to compare clusterings or to judge cluster quality, there is no comprehensive method that allows analysts to evaluate, compare, and refine cluster assignments based on the source data, derived scores, and contextual data.

Results: In this paper, we introduce a method that explicitly visualizes the quality of cluster assignments, allows comparisons of clustering results and enables analysts to manually curate and refine cluster assignments. Our methods are applicable to matrix data clustered with partitional, hierarchical, and fuzzy clustering algorithms. Furthermore, we enable analysts to explore clustering results in context of other data, for example, to observe whether a clustering of genomic data results in a meaningful differentiation in phenotypes.

Conclusions: Our methods are integrated into Caleydo StratomeX, a popular, web-based, disease subtype analysis tool. We show in a usage scenario that our approach can reveal ambiguities in cluster assignments and produce improved clusterings that better differentiate genotypes and phenotypes.

Keywords: Cluster analysis, Visualization, Biology visualization, Omics data

Background

Rapid improvement of data acquisition technologies and the fast growth of data collections in the biological sciences increase the need for advanced analysis methods and tools to extract meaningful information from the data. Cluster analysis is a method that can help make sense of large data and has played an important role in data mining for many years. Its purpose is to divide large datasets into meaningful subsets (clusters) of elements. The clusters then can be used for aggregation, ordering, or, in biology, to describe samples in terms of subtypes and to derive biomarkers. Clustering is ubiquitous in biological data analysis and applied to gene expression, copy number, and epigenetic data, as well as biological networks or text documents, to name just a few application areas.

A cluster is a group of similar items, where similarity is based on comparing data items using a measure of similarity. Cluster analysis is part of the standard toolbox for biology researchers, and there is a myriad of different algorithms designed for various purposes and with differing strengths and weaknesses. For example, clustering can be used to identify functionally related genes based on gene expression, or to categorize samples into disease subtypes. Since Eisen et al. [1] introduced cluster analysis for gene expression in 1998, it has been widely used to classify both, genes and samples in a variety of biological datasets [2–5].

However, while clustering is useful, it is not always simple to use. Scientists have to deal with several challenges: the choice of an algorithm for a particular dataset, the parameters for these algorithms (e.g., the number of

*Correspondence: alex@sci.utah.edu
[1]Scientific Computing and Imaging Institute, University of Utah, 72 Sout Central Campus Drive, 84112 Salt Lake City, USA
Full list of author information is available at the end of the article

expected clusters), and the choice of a suitable similarity metric. All of these choices depend on the dataset and on the goals of the analysis. Also, methods generally suitable for a dataset can be sensitive to noise and outliers in the data and produce poor results for a high number of dimensions.

Several (semi)automated cluster validation, optimization, and evaluation techniques have been introduced to address the basic challenges of clustering and to determine the amount of concordance among certain outcomes (e.g., [6–8]). These methods try to examine the robustness of clustering results and guess the actual number of clusters. This task is often accompanied by visualizations of these measures by histograms or line graphs. Consensus clustering [9] addresses the task of detecting the number of clusters and attaining confidence in cluster assignments. It applies clustering algorithms to multiple perturbed subsamples of datasets and computes a consensus and correlation matrix from these results to measure concordance among them, and explores the stability of different techniques. These matrices are plotted both as histograms and two-dimensional graphs to assist scientists in the examination process.

Although cluster validation is a useful method to examine clustering algorithms it does not guarantee to reconstruct the actual or desired number of clusters from each data type. In particular, cluster validation is not able to compensate weaknesses of cluster algorithms to create an appropriate solution if the clustering algorithm is not suitable for a given dataset.

While knowledge about clustering algorithms and their strengths and weaknesses, as well as automated validation methods are helpful in picking a good initial configuration, trying out various algorithms and parametrizations is critical in the analysis process. For that reason, scientists usually conduct multiple runs of clustering algorithms with different parameters and compare the varying results while examining the concordance or discordance among them.

In this paper we introduce methods to evaluate and compare clustering results. We focus on revealing specificity or ambiguity of cluster assignments and embed our contributions in StratomeX [10, 11], a framework for stratification and disease subtype analysis that is also well suited to cluster comparison. Furthermore, we enable analysts to manually refine clusters and the underlying cluster assignments to improve ambiguous clusters. They can transfer entities to better fit clusters, merge similar clusters, and exclude groups of elements assumed to be outliers. An important aspect of this interactive process is that these operations can be informed by considering data that was not used to run the clustering: when considering cluster refinements, we can immediately show the impact on, for example, average patient survival.

In our tool, users are able to conduct multiple runs of clustering algorithms with full control over parametrization and examine both conspicuous patterns in heatmaps and quantify the quality and confidence of cluster assignments simultaneously. Our measures of cluster fit are independent from the underlying stratification/clustering technique and allow investigators to set thresholds to classify parts of a cluster as either reliable, uncertain, or a bad fit. We apply our methods to matrices of genomic datasets, which covers a large and important class of datasets and clustering applications.

We evaluate our tool based on a usage scenario with gene expression data from The Cancer Genome Atlas and demonstrate how visual inspection and manual refinement can be used to identify new clusters.

In the following we briefly introduce clustering algorithms and their properties, as well as StratomeX, the framework we used and extended for this this research, and other, relevant related work.

Cluster analysis

Clustering algorithms assign data to groups of similar elements. The two most common classes of algorithms are partitional and hierarchical clustering algorithms [12]; less frequently used are probabilistic or fuzzy clustering algorithms.

Partitional algorithms decompose data into non-overlapping partitions that optimize a distance function, for example by reducing the sum of squared error metric with respect to Euclidean distance. Based on that, they either attempt to iteratively create a user-specified number of clusters, like in k-Means [13] or they utilize advanced methods to guess the number of clusters implicitly, such as Affinity Propagation [14].

In contrast to that, **hierarchical clustering** algorithms generate a tree of similar records by either merging smaller clusters into larger ones (agglomerative approach) or splitting groups into smaller clusters (divisive). In the resulting binary tree, commonly represented with a dendrogram, each leaf node represents a record, each inner node represents a cluster as the union of its children. Inner nodes commonly also store a measure of similarity among their children. By cutting the tree at a threshold, we are able to obtain discrete clusters from the similarity tree.

These approaches use a deterministic cluster assignment, i.e., elements are assigned exclusively to one cluster and are not in other clusters. In contrast, **fuzzy clustering** uses a probabilistic assignment approach and allows entities to belong to multiple clusters. The degree of membership is described by weights, with values between 0 (no membership at all) and 1 (unique membership to one cluster). These weights, which are commonly called probabilities, capture the likelihood of an element belonging

to a certain partition. A prominent example algorithm is Fuzzy c-Means [15].

Clustering algorithms make use of a measure of similarity or dissimilarity between pairs of elements. They aim to maximize pair-wise similarity or minimize pair-wise dissimilarity by using either geometrical distances or correlation measures. A popular way to define similarity is a measure of geometric distance based on, for example, squared Euclidean or Manhattan distance. These measures work well for "spherical" and "isolated" groups in the data [16] but are less well suited for other shapes and overlapping clusters. More sophisticated methods measure the cross-correlation or statistical relationship between two vectors. They compute correlation coefficients that denote the type of concordance and dependence among pairs of elements. The coefficients range from -1 (opposite or negative correlation) to 1 (perfect or positive correlation), whereas zero values denote that there is no relationship between two elements. The most commonly used coefficient in that context is the Pearson product-moment correlation coefficient that measures the linear relationship by means of the covariance of two variables. Spearman's rank correlation coefficient is another approach to estimate concordance similar to Pearson's but uses ranks or scores for data to compute covariances.

The choice of distance measure has an important impact on the clustering results, as it drives an algorithm's determination of similarity between elements. At the same time, we can also use distance measures to identify the fit of an element to a cluster, by, for example, measuring the distance of an element to the cluster centroid. In doing so, we do not necessarily need to use the same measure that was used for the clustering in the first place. In our technique, we visualize this information for all elements in a cluster, to communicate the quality of fit to a cluster.

StratomeX

StratomeX is a visual analysis tool for the analysis of correlations of stratifications [10, 11]. This is especially important when investigating disease subtypes that are believed to have a genomic underpinning. Originally developed as a desktop software tool, it has since been ported to a web-based client-server system [17]. Figure 1 shows an example of the latest version of StratomeX. By integrating our methods into StratomeX, we can also consider the relationships of clusters to other datasets, including clinical data, mutations, and copy number alteration of individual genes.

StratomeX visualizes stratifications of samples (patients) as rows (records) based on various attributes, such as clinical variables like gender or tumor staging, bins of numerical vectors, such as binned values of copy number alterations, or clusters of matrices/heat maps. Within these heat maps, the columns correspond to e.g., differentially expressed genes. StratomeX combines the visual metaphor used in parallel sets [18], with visualizations of the underlying data [19]. Each dataset is shown as a column. A header block at the top shows the distribution of the whole dataset, while groups of patients are shown as blocks in the columns. Relationships between blocks are visualized by ribbons whose thickness represents the number of patients shared across two bricks. This method can be used to visualize relationships between groupings and clusterings of different data, but can equally be used to compare multiple clusterings of the same dataset.

StratomeX also integrates the visualization of "dependent data" by using the stratification of a neighboring column for a different dataset. This is commonly used to visualize survival data in Kaplan-Meier plots for a particular stratification, or to visualize expression of a patient cluster in a particular biological pathway.

Related work

There are several tools to analyze clustering results and assess the quality of clustering algorithms. A common approach to evaluate clustering results is to visualize the underlying data: heatmaps [1], for example, enable users to judge how consistent a pattern is within a cluster for high-dimensional data.

Seo at el. [20] introduced the hierarchical clustering explorer (HCE) to visualize hierarchical clustering results. It combines several visualization techniques such as scattergrams, histograms, heatmaps and dendrogram views. In addition to that, it supports dynamic partitioning of clusters by cutting the dendrogram interactively. HCE also enables the comparison of different clustering results while showing the relationship among two clusters with connecting links. Mayday [21, 22] is a similar tool that, in contrast to HCE, provides a wide variety of clustering options.

CComViz [23] is a cluster comparison application that uses the parallel sets technique to compare clustering results on the same data, and hence is related to the original StratomeX. In contrast to our proposed technique it does not allow for internal evaluation, cluster refinement, or the visualization of cluster fits.

Lex et al. [24] introduced Matchmaker, a method that enables both, comparisons of clustering algorithms, and clustering and visualization of homogeneous subsets, with the intention of producing better clustering results. Matchmaker uses a hybrid heatmap and a parallel sets or parallel coordinates layout to show relationships between columns, similar to StratomeX. VisBricks [19] is an extension of this idea and provides multiform visualization for the data represented by clusters: users can choose which visualization technique to use for which cluster.

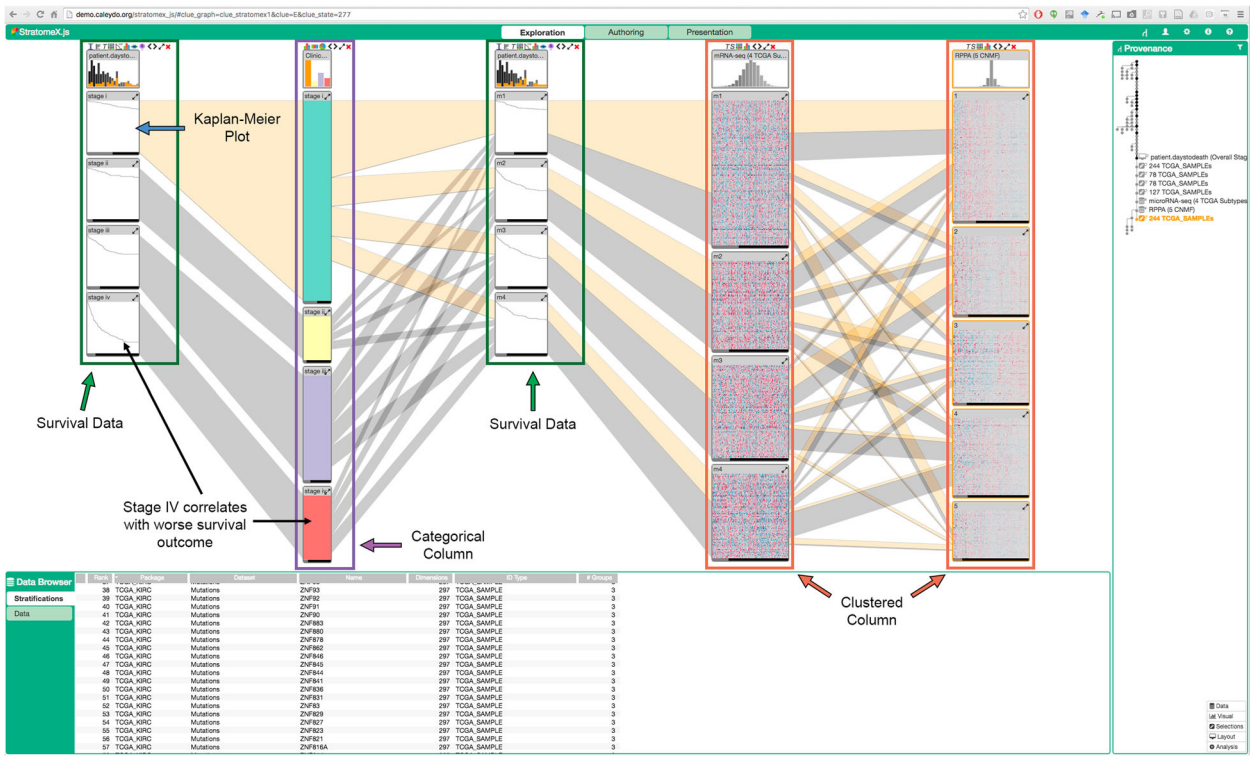

Fig. 1 Screenshot of Caleydo StratomeX, which forms the basis of the technique introduced in this paper showing data from the TCGA Kidney Renal Clear Cell Carcinoma dataset [4]. Each column represents a dataset, which can either be categorical, like in the second column from the left which shows tumor staging, or based on the clustering of a high-dimensional dataset, like the two columns on the right, showing mRNA-seq and RPPA data, respectively. The blocks in the columns represent groups of records, where matrices are visualized as heat maps, categories with colors, and clinical data as Kaplan-Meier plots. The columns showing Kaplan-Meier plots are "dependent columns", i.e., they use the same stratification as a neighboring column. The Kaplan-Meier plots show survival times from patients. The first column shows survival data stratified by tumor staging, where, as expected, higher tumor stages correlate with worse outcomes

In contrast to these techniques, Domino [25] provides a completely flexible arrangement of data subsets that can be used to create a wide range of visual representations, including the Matchmaker representation. It is, however, less suitable for cluster evaluation and comparison.

A tool that addresses the interactive exploration of fuzzy clustering in combination with biclustering results is FURBY [26]. It uses a force-directed node-link layout, representing clusters as nodes and the relationship between them as links. The distance between nodes encodes the (approximate) similarity of two nodes. FURBY also allows users to refine or improve fuzzy clusterings by choosing a threshold that transforms fuzzy clusters into discrete ones.

Tools such as ClustVis [27] and Clustrophile [28] take a more traditional approach to cluster visualization by using scatterplots based on dimensionality reduction (e.g., using PCA) and/or heat maps to visualize clustering results. While these tools are well suited to evaluate a particular clustering result, they are less powerful with regards to comparison between clusterings.

A tool that is more closely related to our work is XCluSim [29]. It focuses on visual exploration and validation of different clustering algorithms and the concordance or disconcordance among them. It combines several small sub-views to form a multiview layout for cluster evaluation. It contains dendrogram and force-directed graph views to show concordance among different clustering results and uses colors to represent clusters, without showing the underlying data. It offers a parallel sets view where each row represents one clustering result and thick dark ribbons depict which groups are stable, i.e., consistent throughout all clustering results. In contrast to XCluSim, our method integrates cluster metrics with the data more closely and can also bring in other, related data sources, to evaluate clusters. Also, XCluSim does not support cluster refinement.

Table 1 provides a comparison between these most closely related tools and our technique.

Our methods are also related to silhouette plots, which visualize the tightness and separation of the elements in a cluster [30]. Silhouette plots, however, work best for

Table 1 Comparison of our technique to the most important existing tools with respect to basic data-processing and visualization features, clustering options, cluster visualization features, and software properties

General features	This work	StratomeX [10, 11]	CComViz [23]	XCluSim [29]	ClustVis [27]
Support of multiple genomic data types	✓	✓	✓	✓	✓
Integration of contextual data	✓	✓	✗	✗	✗
Import of custom datasets	✓	✓	✓	✓	✓
Preprocessing of datasets	✗	✗	✗	✓	✓
Interactive plots / Linking and brushing	✓	✓	✗	✓	✗
Customizable plots	✓	✓	✗	✓	✓
Automated removal of visual clutter	✓	✓	✓	✓	✗
Clustering features					
Dynamic application of clustering	✓	✓	✗	✓	✓
Interactive cluster refinement	✓	✗	✗	✗	✗
Hard/partitional clustering	✓	✗	✗	✓	✓
Hierarchical clustering	✓	✗	✗	✓	✗
Fuzzy clustering	✓	✗	✗	✗	✗
Density Based Clustering	✗	✗	✗	✓	✗
Cluster visualization					
Visualize multiple stratifications	✓	✓	✓	✓	✗
Detailed view of clusters / stratifications	✓	✓	✗	✓	✗
Visualization of cluster fits	✓	✗	✗	✗	✗
PCA plots	✗	✗	✗	✗	✓
Cluster results comparison / evaluation	✓	✓	✓	✓	✗
Stable cluster analysis	✓	✗	✓	✓	✗
Software properties					
Web-based software / tool	✓	✗	✗	✗	✓
Open source	✓	✓	✗	✗	✓
Actively developed	✓	✗	✗	✓	✓

The most important features for our technique are highlighted in bold. Note that our technique does not support preprocessing, density based clustering, and PCA plots, but otherwise is the most comprehensive tool. Feature groups and important features are shown in bold

geometric distances and clearly separated and spherical clusters, whereas our approach is more flexible in terms of supporting a variety of different measures of cluster fit. Also, silhouette plots are typically static, however, we could conceivably integrate the metrics used for silhouette plots in our approach. iGPSe [31], for example, is a system similar to StratomeX that integrates silhouette plots.

Implementation

Requirements

Based on our experience in designing multiple tools for visualizing clustered biomolecular data [10, 11, 19, 24, 25, 32], conversations with bioinformaticians, and a literature review, we elicited a list of requirements that a tool for the analysis of clustered matrices from the biomolecular domain should address.

R I: Provide representative algorithms with control over parametrization. A good cluster analysis tool should enable investigators to flexibly run various clustering algorithms on the data. Users should have control over all parameters and should be able to choose from various similarity metrics.

R II: Work with discrete, hierarchical and probabilistic cluster assignments. Visualization tools that deal with the analysis of cluster assignments should be able to work with all important types of clustering, namely discrete/partitional, hierarchical, and fuzzy clustering. The visualization of hierarchical and fuzzy clusterings is usually more challenging: to deal with hierarchical clusterings a tool needs to enable dendrogram cuts, and to address the properties of fuzzy clusterings, it must support the analysis of ambiguous and/or redundant assignments.

R III: Enable comparison of cluster assignments. Given the ability to run multiple clustering algorithms, it is essential to enable the comparison of the clustering results. This will allow analysts to judge similarities and differences between algorithms, parametrizations, and similarity measures. It will also enable them to identify stable clusters, i.e., those that are robust to changes in parameters and algorithms.

R IV: Visualize fit of records to their cluster. For the assessment of confidence in cluster assignments, a tool should show the quality of cluster assignments for its records and the overall quality for the cluster. This enables analysts to judge whether a record is a good fit to a cluster or whether it's an outlier or a bad fit.

R V: Visualize fit of records to other clusters. Clustering algorithms commonly don't find the perfect fit for a record. Hence, it is useful to enable analysts to investigate if particular records are good fits for other clusters, or whether they are very specific to their assigned clusters. This allows users to consider whether records should be moved to other clusters, whether a group of records should be split off into a separate cluster, and more generally, to evaluate whether the number of clusters in a clustering result is correct.

R VI: Enable refinement of clusters. To enable the improvement of clusters, users should be able to interactively modify clusters. This includes shifting of elements to better fitting clusters based on similarity, merging clusters considered to be similar, and excluding non-fitting groups from individual groups or the whole dataset.

R VII: Visualize context for clusters. It is important to explore evidence for clusters in other data sources. In molecular biology applications in particular, datasets rarely stand alone but are connected to a wealth of other (meta)data. Judging clusters based on effects in other data sources can indicate practical relevance of a clustering, or can reveal dependencies between data sets and hence is important for validation and interpretation of the results.

Based on these requirements, our tool extends StratomeX with new clustering features for cluster evaluation and cluster improvement. Table 1 illustrates how our tool differs from existing clustering tools by comparing their set of features with our work.

Design

We designed our methods to address the aforementioned requirements while taking into account usability and good visualization design practices. Our design was influenced by our decision to integrate the methods into Caleydo StratomeX as StratomeX is a well-established tool for subtype analysis. A prototype of our methods is available at http://caleydo.org/publications/2017_bmc_clustering/. Please also refer to the Additional file 1: supplementary video for an introduction and to observe the interaction.

We developed a model workflow for the analysis and refinement of clustered data, illustrated in Fig. 2. This workflow is made up of four core components: (1) running a clustering algorithm, (2) visual exploration of the results, (3) manual refinement of the clustering results, and (4) interpretation of the results.

1. **Cluster creation.** Investigators start by choosing a dataset and either applying clustering algorithms with desired parametrization or selecting existing, precomputed clustering results. The clustered dataset is added to potentially already existing datasets and clusterings.
2. **Visual exploration.** Once a dataset and clustering are chosen, analysts explore the consistency of clusters and/or compare the results to other clustering outcomes to discover patterns, outliers or ambiguities. If there are not confident about the quality of the result, or want to see an alternative clustering, they can return to step 1 and create new clusters by adjusting the parameters or selecting a different algorithm.
3. **Manual refinement.** If analysts detect records that are ambiguous, they can manually improve clusters to create better stratifications in a process that iterates between refinement and exploration. The refinement process includes splitting, merging and removing of clusters.
4. **Result interpretation.** Once clusters are found to be of reasonable quality, the analysts can proceed to interpret the results. In the case of disease subtype

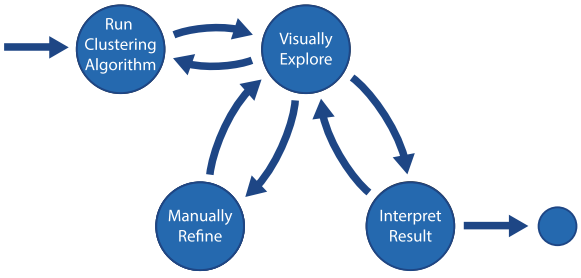

Fig. 2 The workflow for evaluating and refining cluster assignments: (1) running clustering algorithms, (2) visual exploration of clustering results by investigating cluster quality and comparing cluster results (3) manual refinement and improvement of unreliable clusters and (4) final interpretation of the improved results considering contextual data

analysis with StratomeX, they can assess the clinical relevance of subtypes, or explore relationships to other genomic datasets, confounding factors, etc. Of course, supplemental data can also inform the exploration and refinement steps.

We now introduce a set of techniques to address our proposed requirements within this workflow.

Creating clusters

Users are able to create clusters by selecting a dataset from a data browser window and choosing an algorithm and its configuration (see Fig. 3). In our prototype, we provide a selection of algorithms commonly in bioinformatics, including k-Means, (agglomerative) hierarchical clustering, Affinity Propagation, and Fuzzy c-Means. Each tab represents one clustering technique with corresponding parameters, such as the number of clusters for k-Means, the linkage method for hierarchical clustering, or the fuzziness factor for Fuzzy c-Means, addressing **R I**. Each execution of a clustering algorithm adds a new column to StratomeX, so that multiple alternative results can be easily compared.

Cluster evaluation

In our application, there are two components that enable analysts to evaluate cluster assignments: (1) the display of the underlying data in heatmaps or other visualizations and (2) the visualizations of cluster fit alongside the heatmap, as illustrated in Fig. 4. The cluster fit data is either a measure of similarity of each record to the cluster centroid, or, if fuzzy clustering is used, the measure of probability that a record belongs to a cluster. Combining heatmaps and distance data allows users to relate patterns or conspicuous groups in the heatmap to their measure of fit.

To evaluate the fit of each record to its cluster (R IV), we use a **distance view** shown right next to the heatmap (orange in Fig. 4). It displays a bar-chart showing the distances of each record to the cluster centroid. Each bar is aligned with the rows in the heatmap and thus represents the distance or correlation value of the corresponding record to the cluster mean. The length of a bar encodes the distance, meaning that short bars indicate well fitting records while long bars indicate records that are a poor fit. In the case of cross-correlation, long bars represent records with high concordance whereas small bars indicate a disconcordance among them. While the absolute values of distances are typically not relevant for judging the fit of elements to the cluster, we show them on mouse-over in a tool-tip. The heatmaps and distance views are automatically sorted from best to worst fit which makes identifying the overall quality of a cluster easy. In addition to that, we globally scale the length of each bar according to its distance measure, so that the largest bar represents the maximal computed distance measure across all distance views. Note that the distance measure used for the distance view does not have to be the one that was used for clustering. Figure 5 shows a montage of different distance measures for the same cluster in distance views. Notice that while some trends are consistent across many measures, this is not the case for all measures and all patterns, illustrating the strong influence of the choice of a similarity measure.

Related to cluster fit is the question about the specificity of a record to a cluster (R V). It is conceivable that a record is a fit for multiple clusters, or that it would be

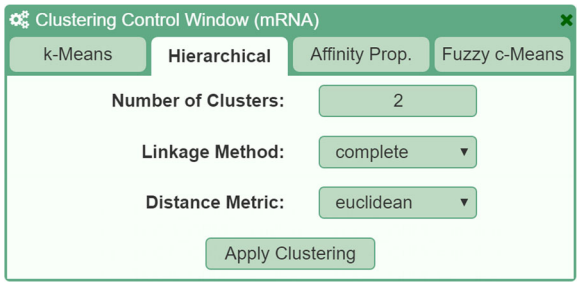

Fig. 3 Example of the control window to apply clustering algorithms on data. Different algorithms are accessible using tabs. Within the tabs, the algorithm can be configured using algorithm-specific parameters and general distance metrics

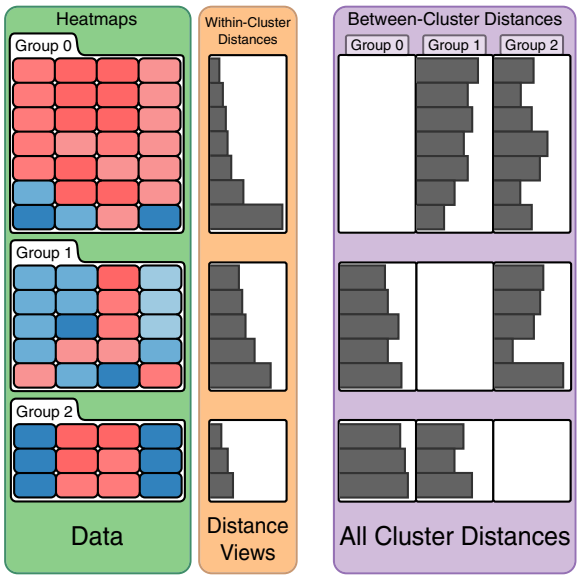

Fig. 4 Illustration of heatmaps, within-cluster, and between-cluster distance views. The heat maps (green, left) show the raw data grouped by a clustering algorithm. The within-cluster distance view shows the quality of fit of each record to its cluster (orange, middle). The between-cluster distance view shows the quality of fit of each record to each other cluster (violet, right). This enables analysts to spot whether a record would also fit to another cluster

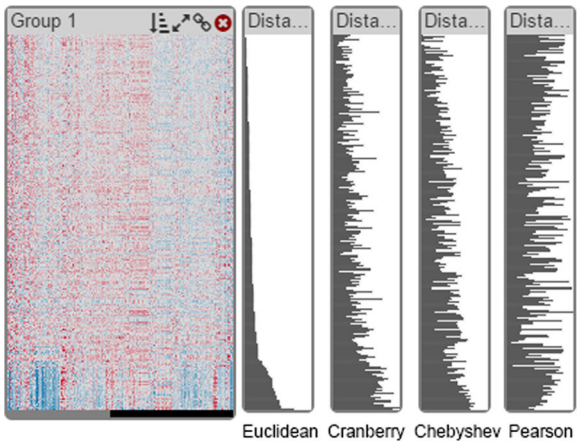

Fig. 5 A montage of distance views showing different distance metrics for the same cluster. From left to right: Euclidean distance, Cranberry distance, Chebyshev distance, and Pearson correlation. Note that long bars for Pearson correlation indicate high similarity. This illustrates that different distance metrics are likely to produce different results

a better fit to another cluster. To convey this, we compute the distances of each record to all other cluster centroids and visualize it in a **matrix of distances** to the right of the within-cluster distance view (violet in Fig. 4). In doing so, we keep the row associations intact. We do not display the within-cluster distances in the matrix, which results in empty cells along the diagonal. This view helps analysts to investigate ambiguous records and supports them in judging whether the number of clusters is correct: if a lot of records have high distances to all clusters, maybe they should belong to a separate cluster. On demand, the heatmaps can also be sorted by any column in the between-cluster distance matrix. As an alternative to the bar charts, we also provide a grayscale heat map for between-cluster distances (see Fig. 6), which scales better when the algorithm produced many clusters.

Visualizing probabilities for fuzzy clustering Since our tool also supports fuzzy clustering (R II) we provide a **probability view**, similar to the distance view, to show the degree of membership of each record to all clusters. In the probability view, the bars show the probability of a record belonging to a current cluster, which means that long bars always indicate a good fit. As each record has a certain probability to belong to each cluster, we use a threshold above which a record is displayed as a member of a cluster. Records can consequently occur in multiple clusters. Records that are assigned to multiple clusters are highlighted in purple, as shown in Fig. 7, whereas unique records are shown in green. As for distance views, we also show probabilities of each record belonging to each cluster in a matrix, as shown in Fig. 7 on the right.

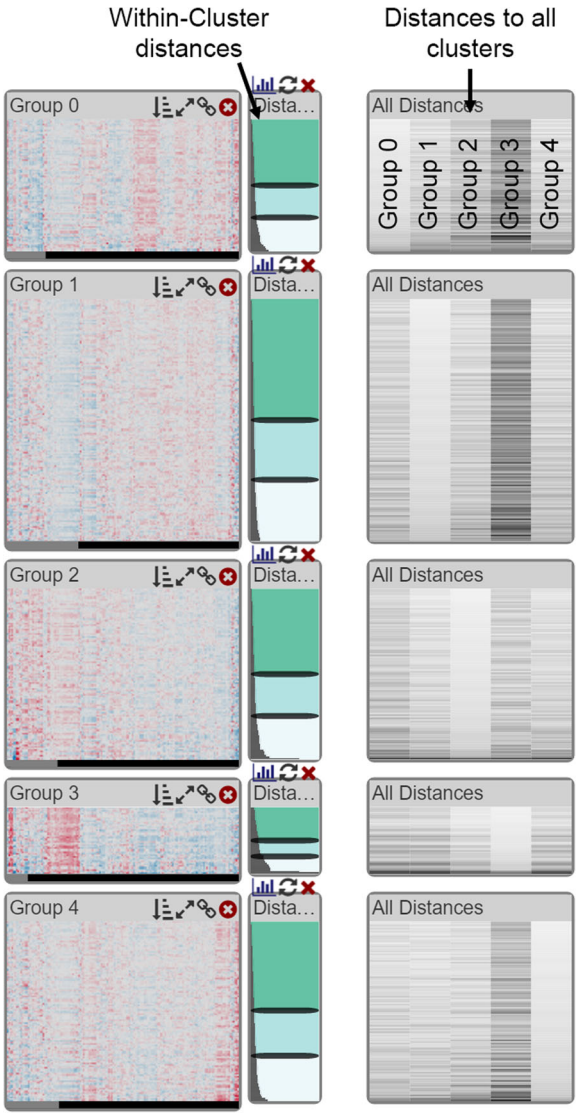

Fig. 6 Example of five clusters, shown in heat maps. Next to the heat maps, small bar charts show the within-cluster distances which enables an analyst to evaluate the fit of individual elements to the cluster. The records are sorted by fit, hence the worst fitting records are shown at the bottom of each cluster. The grayscale heat map on the right shows the distance of each record to each other cluster, i.e., the first column shows the fit to the first cluster, the second column shows the fit to the second cluster, etc. Columns that correspond to the within-cluster distances are empty

Cluster refinement

Once scientists have explored the cluster assignments, the next step is to improve the cluster assignments if necessary (R VI).

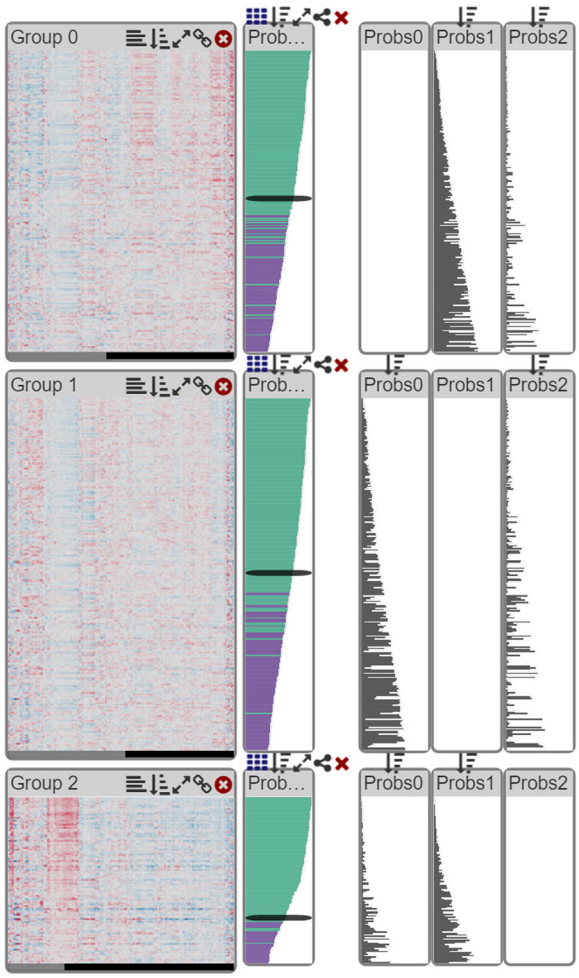

Fig. 7 Example of three clusters produced by fuzzy clustering, shown in heatmaps. The probabilities of each patient belonging to their cluster are shown to their right. Green bars represent elements unique to the cluster while purple indicates elements belonging to more clusters. The between-cluster probabilities are displayed on the right

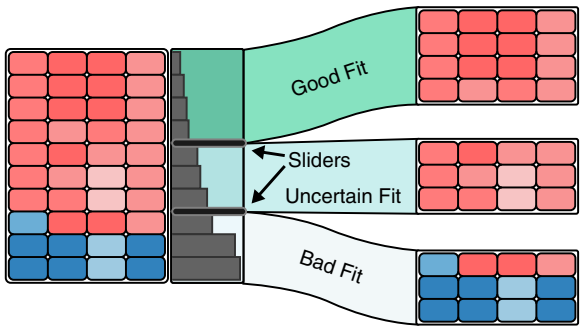

Fig. 8 Example of a cluster being split into three different subsets. The dark green region at the top corresponds to record that fit reliably to the cluster, the light-green group in the middle corresponds to records that are uncertain with respect to cluster fit, the white group at the bottom corresponds to records that do not fit well with the cluster. The black sliders on top of the bar charts can be used to manually adjust these regions. The split clusters are shown as a separate column on the right

Splitting clusters Not all elements assigned to a cluster fit equally well. It is not uncommon that a group of elements within a cluster is visibly different from the rest, and the clusters would be of higher quality if it were split off. To support splitting of clusters, we extended StratomeX to enable analysts to define ambiguous regions in a cluster. The distance views contain adjustable sliders that enable analysts to select up to three regions to classify records into good, ambiguous, and bad fit (the green, light-green, and bright regions in Fig. 8). By default, the sliders are set to the second and third quartile of the within-cluster distance distribution. Based on these definitions, analysts can split the cluster, which extracts the blocks into a separate column in StratomeX, as illustrated in Fig. 8). This new column is treated like a dataset in its own right, such that the distance views show the distances to the new centroids. However, these splits are not static: it is possible to dynamically adjust both sliders and hence the corresponding cluster subsets. In the context of fuzzy clustering, clusters can also be split based on probabilities.

Splitting only based on distances, however, does not guarantee that the resulting groups are as homogeneous as they could be: all they have in common is a certain distance range from the original centroid, yet these distances could be in opposite "directions". To improve the homogeneity of split clusters, we can dynamically shift the elements between the clusters, so that the elements are in the cluster that is closest to them using an approach similar to the k-Means algorithm. Shifting is based on the same similarity metric that was used to produce the original stratification.

Merging and exclusion Our application also has the option to merge clusters. Especially when several clusters are split first, it is likely that some of the new clusters exhibit a similar pattern, and that their distances also indicate that they could belong together. This problem of too many clusters for the data can be addressed using a merge operation. We also support cluster exclusion since there might be groups or individual records that are outliers and shouldn't belong to any cluster.

Integration with StratomeX

The original StratomeX technique already enables cluster comparison R III through the columns and ribbons approach. It also is instrumental in bringing in contextual information for clusters R VII, as mentioned before. This can, for example, be used to asses the impact of refined

clusterings on phenotypes. Figure 9 shows the impact of a cluster split on survival data, for example.

Technical realization

Our methods are fully integrated with the web-version of Caleydo StratomeX. The software version is based on Phovea [33], an open source visualization platform targeting biomedical data. It is based on a client-server architecture with a server runtime in Python using the Flask framework and a client runtime in JavaScript and Typescript. Phovea supports the development of client-side and server-side plugins to enhance web tools in a modular manner. The clustering algorithms and distance computation used in this work are implemented as server-side Phovea plugins in Python using the SciPy and NumPy libraries. The front end, including the distance and matrix views, is implemented as a client-side Phovea plugin and uses D3 [34] to dynamically create the plots. The source code is released under the BSD license and is available at http://caleydo.org/publications/2017_bmc_clustering/.

Results

A common question in clustering is how to determine the appropriate number of clusters in the data. While there are algorithmic approaches, such as the cophenetic correlation coefficient [35], to estimate the number of clusters, visual inspection is often the initial step in confirming that a clustering algorithm has separated the elements appropriately. In this usage scenario we use our approach toinspect and refine a clustering result provided by an external clustering algorithm and to confirm our results with an integrated clustering algorithm.

We obtained mRNA gene expression data from the glioblastoma multiforme cohort of The Cancer Genome Atlas study [2] as well as clustering results generated using a consensus non-negative matrix factorization (CNMF) [36]. Verhaak et al. [2] reported four expression-derived subtypes in glioblastoma, which motivated us to review the automatically generated, uncurated, CNMF clustering results with 4 clusters. Visual inspection indicates that clusters named Group 0 and Group 1 contain patients that appear to have expression profiles that are very different from the other patients (see Fig. 10c). Using the within-cluster distance visualization and sorting the patients in those clusters according to the within-cluster distance reveals that the expression patterns are indeed very different and that the within-cluster distances for those patients are also notably larger than for the other patients. Resorting the clusters by between-cluster distances to the other 3 clusters, respectively, shows that these patients are also different from the patients in the other clusters (see Fig. 10).

Manual cluster refinement Using the sliders in the within-cluster distance visualization and the cluster splitting function we separated aforementioned patients from the clusters named Group 0 and Group 1. Because their profiles are very similar, we merged them into a single cluster using the cluster merging function (see Fig. 10e). The expression profiles in the resulting new cluster look

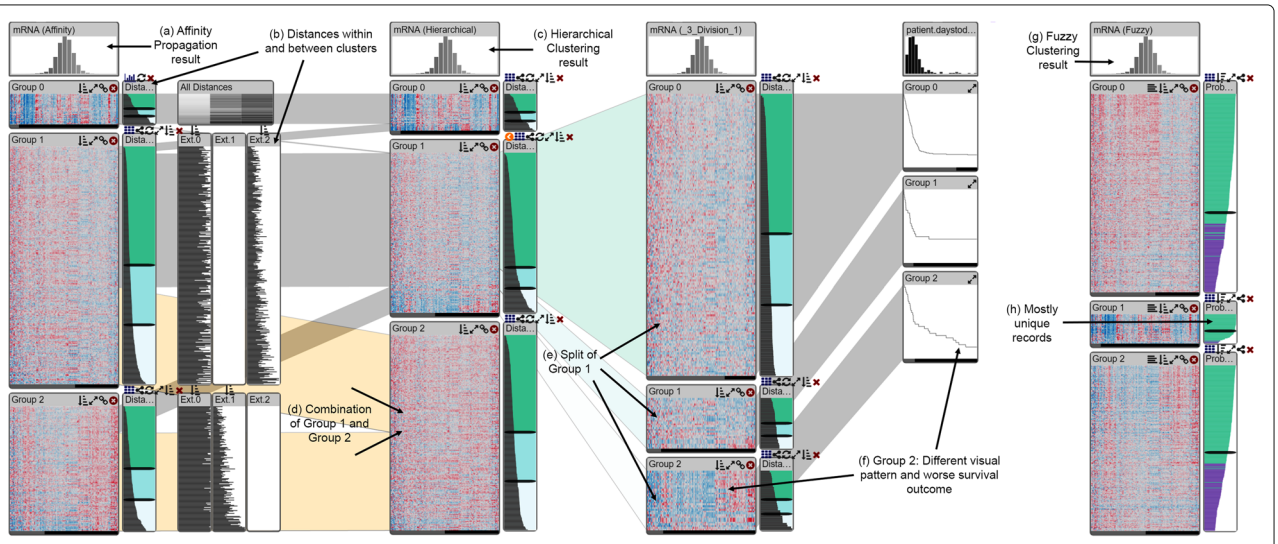

Fig. 9 Overview of the improved StratomeX. **a** The first column is stratified into three groups using affinity propagation. **b** Distances between all clusters are shown. **c** The second column shows the same data but is clustered differently using a hierarchical algorithm. **d** Notice that Group 2 in the second column is a combination of parts of Group 1 and Group 2 of the first column. **e Manual cluster refinement:** The second block (Group 1) of the second column is split, and we see clearly that the patterns in the block at the bottom is quite different from the others. **f** This block also exhibits a different phenotype: the Kaplan-Meier plot shows worse outcomes for this block. **g** The rightmost column shows the same dataset clustered with a fuzzy algorithm. **h** Notice that the second cluster contains mostly unique records (most bars are green), while the other two clusters share about a third of their records (violet)

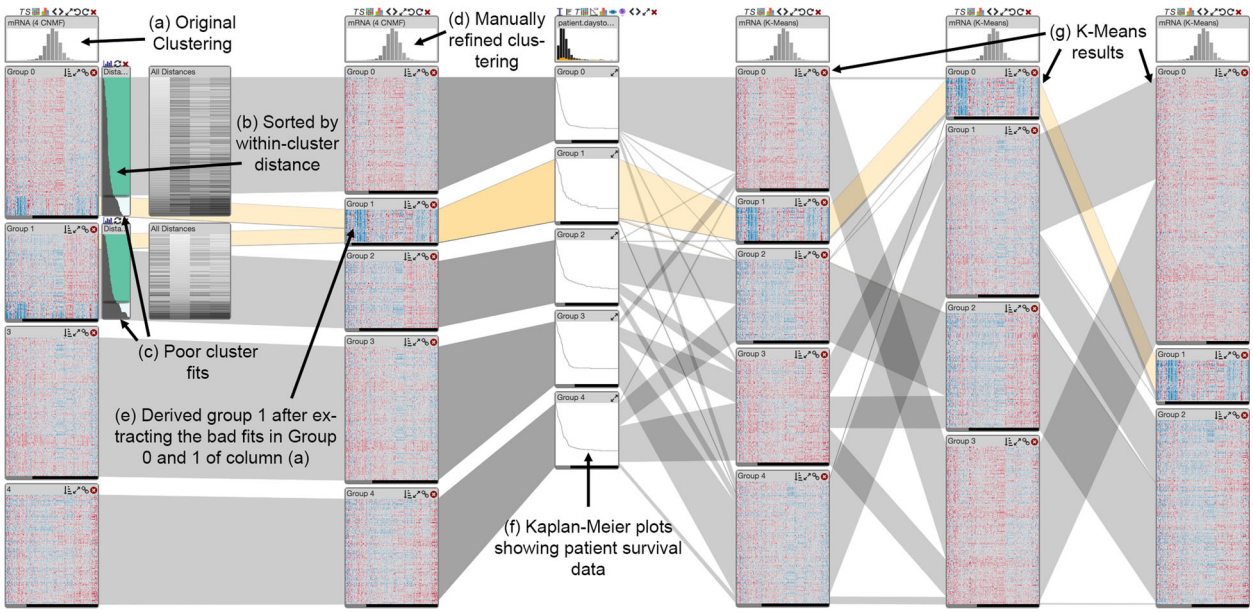

Fig. 10 Results of manual cluster refinement and comparison with additional data types and clustering results. **a** The original clustering is shown in the leftmost column. **b** The records in the clusters are sorted by within-cluster distances, **c** the bottom sections of the clusters contain the expression profiles with poor cluster fit. **d** The second column from the left shows the manually refined clustering with 5 clusters. **e** Here, poorly fitting records from two groups of the original clustering were split off and subsequently merged. **f** The Kaplan-Meier plots show patient survival for each of the five clusters in the second column. **g** The three rightmost columns are k-Means clustering results computed with the software. Notice that the manually derived cluster **e** consistently appears in the k-means clustering results

homogeneous and are visibly different from the expression profiles in the other four clusters. We examined patient survival times (Fig. 10f) across the five clusters and did not observe any notable differences in the new cluster. Since the web-based prototype of StratomeX is currently still lacking the guided exploration features of the original standalone application [11], we were unable to identify a meaningful correlation between the new cluster and mutation and copy number calls or to identify significantly overlapping clusters in other data types.

However, we also compared the five clusters derived from the original four-cluster CNMF result with other clustering results computed on the same gene expression matrix (Fig. 10g) and found, for example, that three-, four-, and five-cluster k-Means clustering results using Euclidean distance and the k-means algorithm include almost exactly the same cluster that we identified in the CNMF clustering results using visual inspection and manual refinement.

Discussion

Our methods are limited by the inherent limitations of StratomeX: when working with a large number of clusters, ribbons between the individual columns can result in clutter. We observe that 10-15 clusters can be used without too much clutter. Also, the number of columns is limited to about ten on typical displays. In terms of computational scalability, we found that even the computationally complex clustering algorithms such as affinity propagation execute almost interactively for a dataset with about 500×1500 entries, and complete within one to two minutes for a genomic dataset with about $500 \times 12,000$ entries on our *t2.micro Amazon EC2 instance* with 1 CPU and 1 GB memory. We find that the performance of our technique is in line with or superior to related techniques (see Table 1).

Our implementation currently cannot appropriately compare columns clustered with fuzzy algorithms, as the ribbons connecting the columns assume that every row exists only once. We plan on addressing this limitation in the future, either by allowing overlapping ribbons, or by using a separate visualization optimized to visualize set overlaps, such as UpSet [37].

Conclusions

Clustering is an important yet inherently imperfect process. In this paper we have introduced methods to evaluate and refine clustering results for the application to matrix data, as it is commonly used in molecular biology. In contrast to previous approaches, we combine visualization of the data directly with visualization of cluster quality and enable the comparison of multiple clustering results. We also allow interactive refinement of clusters while associating the updated clusters with

contextual data, which allows analysts to judge clusters not only by the data used for clustering, but also based on effects observable in related datasets. We argue that our tool is thus the most comprehensive technique to refine, create, evaluate, compare, and contextualize clustering results.

In the future, we plan on adding additional clustering algorithms, as different algorithms have complementary strengths and weaknesses, and explore the possibility of using distributed clustering algorithms to scale to even bigger datasets. Also, density based clustering algorithms [38], which treat outliers separately would be valuable to integrate and would mandate an extension of our visualization method. We also plan on addressing cases with large numbers of clusters, a current limitation of our approach, which, however, will likely require a different visualization approach.

We plan on enabling analysts to cluster individual blocks, i.e., to run a clustering algorithm on the subset of records that were previously assigned to a cluster. This approach could be used to identify groups of outliers in clusters, which could then be split off and re-integrated with other clusters.

Finally, we will extend our work to datasets that are not in matrix form. This will require novel visual representations, as there is no equivalent to the well-defined borders of cluster blocks when clustering graphs or textual data.

Availability and requirements

Project name: Caleydo StratomeX
Project home page: http://caleydo.org/publications/2017_bmc_clustering/
Operating system(s): web-based
Programming language: Python, JavaScript
Other requirements: none
License: The 3-Clause BSD License

Additional file

Additional file 1: We provide a Supplementary Video showing an interactive demonstration of the technique. (MP4 44,339 kb)

Abbreviations

GBM: Glioblastoma multiforme; TCGA: The Cancer Genome Atlas

Acknowledgements

We thank Samuel Gratzl for his help with the implementation.

Funding

This work was funded in part by the US National Institutes of Health (U01 CA198935, P41 GM103545-17, R00 HG007583) and supported by a fellowship of the FITweltweit program of the German Academic Exchange Service (DAAD). The funding agencies played no role in the design or the conclusion of this study.

Availability of data and materials

The source code is released under the BSD license and is available at http://caleydo.org/publications/2017_bmc_clustering/. We also host a prototype that is accessible through that link. The data we use to demonstrate the technique has been downloaded from the NIH GDC Data Portal at https://portal.gdc.cancer.gov/.

Authors' contributions

MK implemented the software and contributed to the design of the study and the write-up. AL, NG, and CRJ contributed to the design of the study and the writing. All authors read and approved the final manuscript.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Author details

[1]Scientific Computing and Imaging Institute, University of Utah, 72 Sout Central Campus Drive, 84112 Salt Lake City, USA. [2]Department of Informatics, Technical University of Munich, 85747 Garching bei München, Germany. [3]Department of Biomedical Informatics, Harvard Medical School, 02115 Boston, USA.

Received: 7 April 2017 Accepted: 29 August 2017
Published online: 12 September 2017

References

1. Eisen MB, Spellman PT, Brown PO, Botstein D. Cluster analysis and display of genome-wide expression patterns. Proc Natl Acad Sci USA. 1998;95(25):14863–8.
2. Verhaak RGW, et al. Integrated Genomic Analysis Identifies Clinically Relevant Subtypes of Glioblastoma Characterized by Abnormalities in PDGFRA, IDH1, EGFR, and NF1. Cancer Cell. 2010;17(1):98–110.
3. The Cancer Genome Atlas Research Network. Comprehensive molecular portraits of human breast tumours. Nature. 2012;490(7418):61–70.
4. The Cancer Genome Atlas Research Network. Comprehensive molecular characterization of clear cell renal cell carcinoma. Nature. 2013;499(7456):43–9.
5. The Cancer Genome Atlas Research Network. Genomic Classification of Cutaneous Melanoma. Cell. 2015;161(7):1681–96.
6. Halkidi M, Batistakis Y, Vazirgiannis M. On Clustering Validation Techniques. J Intell Inf Syst. 2001;17(2-3):107–45.
7. Bolshakova N, Azuaje F. Cluster validation techniques for genome expression data. Signal Proc. 2003;83(4):825–33.
8. Famili AF, Liu G, Liu Z. Evaluation and optimization of clustering in gene expression data analysis. Bioinformatics. 2004;20(10):1535–45.
9. Monti S, Tamayo P, Mesirov J, Golub T. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. Mach Learn. 2003;52(1-2):91–118.
10. Lex A, Streit M, Schulz HJ, Partl C, Schmalstieg D, Park PJ, Gehlenborg N. StratomeX: Visual Analysis of Large-Scale Heterogeneous Genomics Data for Cancer Subtype Characterization. Comput Graph Forum (EuroVis '12). 2012;31(3):1175–84.
11. Streit M, Lex A, Gratzl S, Partl C, Schmalstieg D, Pfister H, Park PJ, Gehlenborg N. Guided visual exploration of genomic stratifications in cancer. Nat Methods. 2014;11(9):884–5.
12. Jain AK, Murty MN, Flynn PJ. Data clustering: a review. ACM Comput Surv. 1999;31(3):264–323.
13. Macqueen JB. Some methods for classification and analysis of multivariate observations. In: In 5-Th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1. Berkeley, California, USA: University of California Press; 1967. p. 281–97.
14. Frey BJJ, Dueck D. Clustering by Passing Messages Between Data Points. Science. 2007;315(5814):972–6.

15. Bezdek JC, Ehrlich R, Full W. FCM: The fuzzy c-means clustering algorithm. Comput Geosci. 1984;10(2):191–203.
16. Mao J, Jain AK. A self-organizing network for hyperellipsoidal clustering (HEC). Neural Netw, IEEE Trans. 1996;7(1):16–29.
17. Gratzl S, Lex A, Gehlenborg N, Cosgrove N, Streit M. From Visual Exploration to Storytelling and Back Again. Comput Graph Forum. 2016;35(3):491–500.
18. Kosara R, Bendix F, Hauser H. Parallel Sets: Interactive Exploration and Visual Analysis of Categorical Data. IEEE Trans Vis Comput Graph. 2006;12(4):558–68.
19. Lex A, Schulz HJ, Streit M, Partl C, Schmalstieg D. VisBricks: Multiform Visualization of Large, Inhomogeneous Data. IEEE Trans Vis Comput Graph (InfoVis '11). 2011;17(12):2291–300.
20. Seo J, Shneiderman B. Interactively exploring hierarchical clustering results [gene identification]. Computer. 2002;35(7):80–6.
21. Gehlenborg N, Dietzsch J, Nieselt K. A framework for visualization of microarray data and integrated meta information. Inf Vis. 2005;4(3): 164–75.
22. Dietzsch J, Gehlenborg N, Nieselt K. Mayday–a microarray data analysis workbench. Bioinformatics. 2006;22(8):1010–2.
23. Zhou J, Konecni S, Grinstein G. Visually comparing multiple partitions of data with applications to clustering. In: Proceedings Volume 7243, visualization and data analysis 2009. San Jose: SPIE; 2009. p. 72430J. doi:10.1117/12.810093.
24. Lex A, Streit M, Partl C, Kashofer K, Schmalstieg D. Comparative Analysis of Multidimensional, Quantitative Data. IEEE Trans Vis Comput Graph (InfoVis '10). 2010;16(6):1027–35.
25. Gratzl S, Gehlenborg N, Lex A, Pfister H, Streit M. Domino: Extracting, Comparing, and Manipulating Subsets across Multiple Tabular Datasets. IEEE Trans Vis Comput Graph (InfoVis '14). 2014;20(12):2023–32.
26. Streit M, Gratzl S, Gillhofer M, Mayr A, Mitterecker A, Hochreiter S. Furby: Fuzzy Force-Directed Bicluster Visualization. BMC Bioinforma. 2014;15(Suppl 6):4.
27. Metsalu T, Vilo J. ClustVis: A web tool for visualizing clustering of multivariate data using Principal Component Analysis and heatmap. Nucleic Acids Res. 2015;43(Web Server issue):566–70.
28. Demiralp C. Clustrophile: A Tool for Visual Clustering Analysis. In: KDD 2016 workshop on Interactive Data Exploration and Analytics (IDEA'16) August 14th, 2016, San Francisco, CA, USA; 2016.
29. L'Yi S, Ko B, Shin D, Cho YJ, Lee J, Kim B, Seo J. XCluSim: A visual analytics tool for interactively comparing multiple clustering results of bioinformatics data. BMC Bioinforma. 2015;16(11):1–15.
30. Rousseeuw PJ. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math. 1987;20:53–65.
31. Ding H, Wang C, Huang K, Machiraju R. iGPSe: A visual analytic system for integrative genomic based cancer patient stratification. BMC Bioinforma. 2014;15(1):203.
32. Turkay C, Lex A, Streit M, Pfister H, Hauser H. Characterizing Cancer Subtypes Using Dual Analysis in Caleydo StratomeX. IEEE Comput Graph Appl. 2014;34(2):38–47.
33. Gratzl S, Gehlenborg N, Lex A, Strobelt H, Partl C, Streit M. Caleydo Web: An Integrated Visual Analysis Platform for Biomedical Data. In: Poster Compendium of the IEEE Conference on Information Visualization (InfoVis '15). Chicago, IL, USA: IEEE; 2015.
34. Bostock M, Ogievetsky V, Heer J. D3: Data-Driven Documents. IEEE Trans Vis Comput Graph. 2011;17(12):2301–9.
35. Sokal RR, Rohlf FJ. The Comparison of Dendrograms by Objective Methods. Taxon. 1962;11(2):33.
36. Broad Institute TCGA Genome Data Analysis Center. Clustering of mRNA Expression: Consensus NMF. 2013. doi:10.7908/C16W983Z.
37. Lex A, Gehlenborg N, Strobelt H, Vuillemot R, Pfister H. UpSet: Visualization of Intersecting Sets. IEEE Trans Vis Comput Graph (InfoVis '14). 2014;20(12):1983–92.
38. Ester M, Kriegel HP, Sander J, Xu X, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: The second international conference on Knowledge Discovery and Data Mining (KDD-96) August 2–4, 1996, Portland, Oregon. Association for the Advancement of Artificial Intelligence; 1996. p. 226–31.