Theoretical Computer Science 428 (2012) 58–79

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

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

Algorithmic paradigms for stability-based cluster validity and model selection statistical methods, with applications to microarray data analysis

R. Giancarlo[a,*], F. Utro[b]

[a] University of Palermo, Dipartimento di Matematica ed Informatica, Via Archirafi 34, 90123 Palermo, Italy
[b] IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA

ARTICLE INFO

Article history:
Received 14 July 2011
Received in revised form 17 December 2011
Accepted 10 January 2012
Communicated by M. Crochemore

Keywords:
Algorithms and data structures
General statistics
Analysis of massive datasets
Machine learning
Computational biology
Bioinformatics

ABSTRACT

The advent of high throughput technologies, in particular microarrays, for biological research has revived interest in clustering, resulting in a plethora of new clustering algorithms. However, *model selection*, i.e., the identification of the correct number of clusters in a dataset, has received relatively little attention. Indeed, although central for statistics, its difficulty is also well known. Fortunately, a few novel techniques for model selection, representing a sharp departure from previous ones in statistics, have been proposed and gained prominence for microarray data analysis. Among those, the stability-based methods are the most robust and best performing in terms of prediction, but the slowest in terms of time. It is very unfortunate that as fascinating and classic an area of statistics as model selection, with important practical applications, has received very little attention in terms of algorithmic design and engineering. In this paper, in order to partially fill this gap, we make the following contributions: (A) the first general algorithmic paradigm for stability-based methods for model selection; (B) reductions showing that all of the known methods in this class are an instance of the proposed paradigm; (C) a novel algorithmic paradigm for the class of stability-based methods for cluster validity, i.e., methods assessing how statistically significant is a given clustering solution; (D) a general algorithmic paradigm that describes heuristic and very effective speed-ups known in the literature for stability-based model selection methods.

Since the performance evaluation of model selection algorithms is mainly experimental, we offer, for completeness and without even attempting to be exhaustive, a representative synopsis of known experimental benchmarking results that highlight the ability of stability-based methods for model selection and the computational resources that they require for the task. As a whole, the contributions of this paper generalize in several respects reference methodologies in statistics and show that algorithmic approaches can yield deep methodological insights into this area, in addition to practical computational procedures.

1. Introduction

In the past 15 years, a paradigm shift in Life Sciences research has taken place, thanks to the availability of genomic and proteomic data on an unprecedented scale. Such a revolutionary change has posed new challenges to mathematics,

* Corresponding author.
E-mail addresses: raffaele@math.unipa.it (R. Giancarlo), futro@us.ibm.com (F. Utro).

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.01.024

statistics and computer science, since the conceptual tools proper of those three disciplines are fundamental for the study of biological questions via computational tools. *Microarrays* is one of the technologies that has most affected such a paradigm shift. Indeed, experiments based on it are common practice in biological and medical research to address a wide range of problems, including the classification of tumors [5,7,25,33,55,56,60], where a reliable and precise classification is essential for successful diagnosis and treatment. Microarray experiments, thanks to the monitoring of gene expression levels on a genomic scale, may lead to a more complete understanding of biomolecular variations within tumor classes and therefore to a finer and more reliable "taxonomy" within each class. In turn, as stated in [25,33], the identification of new tumor (sub)classes using gene expression profiles can be seen as an important statistical and machine learning problem whose quest for a solution has resulted, in particular, in revived interest in cluster analysis.

However, the novelty, noisiness and high dimensionality of microarray data provide new challenges even to a classic and well studied area such as clustering. More in general, new methodological and computational challenges are proposed daily triggering, as observed by Mehta et al. [52], a "malthusian growth" of new statistical and computational methods for genomic analysis. Unfortunately, many of them are questionable (see again [52]).

In the classic statistics and data analysis literature, there are two essential aspects of clustering: finding a "good" partition of the datasets and estimating the number of clusters, if any, in a dataset. The former problem is usually solved by the use of a clustering algorithm. For recent reviews on clustering algorithms, in particular for biomedical research, the reader is referred to [8,23]. However, the most fundamental issue is the latter problem, referred to as model selection, which is usually solved with the use of internal/relative measures (defined in Section 2).

Although there is a vast amount of knowledge available in statistics and in the general data analysis literature, e.g., [18,34, 35,38,41,45], gene expression data provide unique challenges to internal validation measures, as already outlined. Despite their potentially important role, both the use of classic internal validation measures and the design of new ones, specific for microarray data, do not seem to have great prominence in bioinformatics, where attention is mostly given to clustering algorithms. The excellent survey by Handl et al. [36] is a big step forward in making the study of those techniques a central part of both research and practice in bioinformatics. It is also worth mentioning that a recent systematic presentation of statistical indices for clustering, with particular attention to microarray data, is given in [57]. It is also somewhat discouraging to have to report that such a beautiful area of statistics, with important applications, has received very little attention both in terms of algorithmic design and engineering. Along with original results, the state of the art regarding those two latter points is described in [32].

Among those measures, the new class of stability-based measures, among others, has gained prominence due to their robustness and predictive power (see [31] and references therein). Although from the theoretic point of view they exhibit shortcomings [9] that seem to be common to other outstanding methods in this area, e.g., the Gap Statistics [66] (`Gap`, for short), they work remarkably well in practice. Unfortunately, on benchmark datasets, they also prove to be very slow methods [31], a state that limits their use to datasets with a small aspect ratio, i.e., number of items to classify and number of experimental conditions per item. For one of those stability-based methods, i.e., Consensus Clustering by Monti et al. [53] (`Consensus`, for short), a speed-up has been proposed [32], i.e., Fast Consensus (`FC`, for short), that substantially extends the dimensionality spectrum in which that particular method can be used.

The stability-based measures proposed in the literature have a common root in earlier work by Breckenridge [18] and Breiman [19], which were concerned with cluster validity and classification boosting, respectively, rather than model selection. Common features of those measures have been highlighted by Valentini [68]. In this paper we propose a general algorithmic paradigm for stability-based measures and show that all the known measures in that class are instances of the paradigm. The paradigm turns out to be so rich that even the reference method `Gap` [66], not perceived to be stability-based, also falls into this class. Moreover, it offers a powerful methodological tool in order to derive novel, and hopefully more effective, stability-based measures. Since stability-based cluster validity methods are essential "subroutines" of stability-based model selection methods, we also provide the first algorithmic paradigm for those former methods. In addition, motivated by the need to obtain fast stability-based model selection methods, we also formalize an algorithmic paradigm from which heuristic speed-ups can be derived.

Overall, our contribution is to show that, generalizing the techniques due to Breckenridge [18], Breiman [19] and Valentini [68], there are unifying algorithmic principles able to describe one of the most promising classes of statistical indices known in the literature. Apart for the methodological and theoretic contribution, the heuristic speed-up paradigm may have a fundamental practical impact.

The remainder of this paper is organized as follows. Section 2 presents a formal statement of the problems we are interested. For the convenience of the reader, Sections 3 and 4 outline essential building blocks of stability-based methods, i.e., data generation procedures and statistical techniques allowing for the establishment of the level of agreement between two clustering solutions. Sections 5, 6 and 8 are devoted to the description of the paradigms mentioned earlier, deferring the presentation of some of their instances to the Appendix. Section 7 shows that `Gap` also falls into the new paradigm formalized in Section 6. Finally, given that the performance of stability-based methods is established experimentally we offer in Section 9, for the convenience of the reader, some experimental results that represent and summarize the state of the art. To this end, we use some well-known microarray benchmark datasets. Finally, the last section offers some conclusions and lines of future investigation.

2. Basics

Consider a set of n items $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$, where σ_i, $1 \le i \le n$, is defined by m numeric values, referred to as features or conditions. That is, each σ_i is an element in a m-dimensional space. In this paper, Σ is represented as a data matrix D, of size $n \times m$, in which the rows correspond to the items and the columns to the condition values.

The aim of cluster analysis is to determine a partition of Σ according to a *similarity/distance* function S, with domain Σ and range $\mathbb{R}_0^+$. Intuitively, the partition should be such that items in the same cluster are similar, while items in different clusters are dissimilar. There are many different ways to formalize such an intuition, depending on the specific objective function one tries to optimize [36]. In what follows, let $P_k = \{p_1, p_2, \ldots, p_k\}$ denote a partition of Σ obtained via a specific clustering algorithm. Each subset p_i, $1 \le i \le k$, is referred to as a *cluster*, and P_k is referred to as a *clustering solution*.

2.1. Problems formulation

Let C_j be a reference classification for Σ consisting of j classes. That is, C_j may either be a partition of Σ into j groups, usually referred to as the *gold standard*, or a division of the universe generating Σ into j categories, usually referred to as *class labels*. An *external index E* is a function that takes as input a reference classification C_j for Σ and P_k and returns a value assessing how close the partition obtained via an algorithm is to the reference classification. It is external because the quality assessment of the partition is established via criteria external to the data, i.e., the reference classification. Notice that it is not required that $j = k$. A brief overview of external indices is provided in Section 4, for the convenience of the reader. An *internal measure I* is a function defined on the set of all possible partitions of Σ and with values in $\mathbb{R}$. It should measure the quality of a partition according to some suitable criteria. It is internal because the quality of the partition is measured according to information contained in the dataset without resorting to external knowledge. For the state of the art on internal measures, the reader is referred to [31,36]. As pointed out in the Introduction, this paper focuses on stability-based internal measures.

The problem of assessing the "quality" of a clustering solution can be stated formally in three meaningful ways [41], that are reported next.

(**Q.1**) Given C_j, P_k and E, measure how far P_k is from C_j, according to E.
(**Q.2**) Given P_k and I, establish whether the value of I computed on P_k is unusual and therefore surprising. That is, significantly small or significantly large.

Notice that the two questions above try to assess the quality of a clustering solution consisting of k clusters, but they give no indication on what the "right number" of clusters is. In the specialistic literature, this problem is also referred to as *model selection*. Technically, one is interested in the following:

(**Q.3**) Given: (Q.3.a) A sequence of clustering solutions $P_1, \ldots, P_s$, obtained for instance via the repeated application of a clustering algorithm A; (Q.3.b) a function R, usually referred to as a *relative index*, that estimates the relative merits of a set of clustering solutions. One is interested in identifying the partition P_{k^*} among the ones given in (Q.3.a) providing the best value of R. In what follows, the optimal number of clusters according to R is referred to as k^*.

2.2. Model selection with stability-based measures: an intuitive description

A "good" clustering algorithm should produce partitions that do not vary much from one sample to another, when data points are repeatedly sampled and clustered. That is, the algorithm must be stable with respect to input randomization. Therefore, the main idea to validate a clustering solution, based on the notion of stability, is to use a measure of the self-consistency of the data, instead of using the classical concepts of isolation and compactness [36,41]. As it will be clear shortly, this framework can be applied to get procedures for both problems (**Q.2**) and (**Q.3**), since procedures addressing problem (**Q.2**) are essential subroutines for those addressing problem (**Q.3**).

In order to obtain a stability-based internal validation method, one needs to specify the following "ingredients":

1. a data generation/pertubation procedure;
2. a similarity measure between partitions;
3. a statistic on clustering stability—it can be used to address problem (**Q.2**);
4. rules on how to select the most reliable clustering(s)—it can be used to address problem (**Q.3**).

Sections 3 and 4 give an outline of the relevant state of the art for Points 1 and 2, respectively. Sections 5 and 6 are dedicated to the presentation of algorithmic paradigms for Points 3 and 4, respectively. They also provide reductions from the paradigms to some known procedures belonging to the "stability class". We anticipate that additional reductions are presented in the Appendix.

3. Data generation/perturbation techniques

In this section we describe the most relevant data generation/pertubation methods used in the scholarly literature. Each of them is a paradigm in itself and therefore we provide only an outline. Formally, a data generation/perturbation method can be seen as a procedure, in what follows referred to as DGP, that takes as input a dataset D along with other parameters and returns a new dataset D' of size $n' \times m'$, with $n' \le n$ and $m' \le m$.

3.1. Subsampling/Bootstrapping

The simplest way to generate a new dataset D' from D is to take random samples from it. Although simple, this approach critically depends on whether the sampling is performed with or without replacement. The first type of method is referred to as *subsampling*. It is widely used in clustering and briefly discussed next.

Formally, a subsampling procedure takes as input a dataset D and a parameter β, $0 < \beta < 1$, and gives as output a percentage β of D, i.e., the dataset D' has size $n' \times m$, with $n' = \lceil \beta n \rceil$. D' is obtained via the extraction of n' items (i.e. rows) from D, which are usually selected uniformly and at random.

The aim of those procedures is to generate a reduced dataset D' that captures the structure (i.e., the right number of clusters) in the original data. Intuitively, the chance to achieve that goal and the time required by the procedures using D' both increase with β. In order to have a good trade-off between the representativeness of D' and the speed of the methods using it, a value of $\beta \in [0.6, 0.9]$ is recommended [10,25,32,53]. Moreover, those procedures do not guarantee that each cluster in D is represented in D' since extracting elements at random does not guarantee that all clusters will be covered by such a selection. Hansen et al. [37] have partially addressed this problem via *proportionate stratified sampling*, although no formal guarantee is given that the entire cluster structure of D is present in D'.

The second method is the well-known *bootstrap* and, although fundamental and of widespread use in statistics [28], it is rarely used in cluster validation as pointed out and discussed in [41,53]. With this technique, the new dataset D' is obtained by sampling, uniformly and at random and *with replacement*, the n rows of D. D' is usually referred to as *bootstrap sample* and it has the same size as D. For ease of notation in what follows, we indicate bootstrap as a subsampling procedure with $\beta = 1$.

3.2. Noise injection

Noise injection is a widely applied perturbation methodology in computer science (see [11,16,46,51,58,70] and reference therein). However, it is not widely applied in clustering. The main idea is to generate D' by adding a random value, i.e., a "perturbation", to each of the elements of D. Perturbations are generated via some random process, i.e., a probability distribution whose parameters can be directly estimated from D. In a study about melanoma patients, Bittner et al. [16] propose to perturb the original dataset by adding Gaussian noise to its elements in order to assess cluster stability. Following up, Wolfinger et al. [70] report that perturbing the data via a Gaussian distribution provides good stability results for the important case of microarray datasets. As for parameter estimation, McShane et al. [51] propose computing the variance of the experiments in each row of D and then using the median of the observed distribution as the variance in the Gaussian distribution.

3.3. Dimensionality reduction methods

The methodology described in this section is, in most cases, the dual of subsampling, since the main idea is to obtain D' by reducing the number of columns of D while trying to preserve its cluster structure. Since each element, i.e. row, of D is a point in m-dimensional space, one has a dimensionality reduction in the data.

Principal Component Analysis (PCA for short) [41] is probably the best known technique used to achieve dimensionality reduction in data analysis. Although of standard use in many tasks, it is not used in conjunction with stability-based internal validation measures because PCA is deterministic in generating D'. We anticipate, however, that the main idea of principal components is used in conjunction with null models, which are described in Section 3.4.

The following three techniques of dimensionality reduction seem to be of use in this area. The first one is rather trivial since it consists of randomly selecting the columns of D [64]. However, the cluster structure of D is unlikely to be preserved and this approach may introduce large distortions into gene expression data, which could result in the introduction of biases into stability-based measures, as reported in [12]. More sensible approaches for dimensionality reduction are matrix factorization and randomized dimensionality reduction techniques reported in the following. This is discussed next, while the interested reader is referred to [67] for a detailed outline of the former. Indeed, non-negative matrix factorization [48,49], which would be the most relevant for internal validation measures, is used together with them mostly as a clustering algorithm rather than as a dimensionality reduction technique, e.g., [20,32].

3.3.1. Randomized dimensionality reduction

The technique consists of the use of a family of transformations that try to preserve the distance with an "ε distortion level" between the elements of D. Intuitively, if two elements are "close" in D, according to some distance function d, they should be "close" in D'. Let f be a transformation from D in D', $f(\sigma_i)$ and $f(\sigma_j)$ be the projections of two elements σ_i and σ_j of D into D'. Let

$$d_f(\sigma_i, \sigma_j) = \frac{d(\sigma_i, \sigma_j)}{d(f(\sigma_i), f(\sigma_j))}.$$

If $d_f = 1$, the distance between the two elements is preserved. When $1-\varepsilon \le d_f(\sigma_i, \sigma_j) \le 1+\varepsilon$, one says that the function f preserves the distance with an "ε distortion level". The Johnson–Lindenstrauss Lemma [42] and *random projections* are the keys to all the randomized dimensionality reduction techniques. Intuitively, for a fixed distortion level ε, the Johnson–Lindenstrauss Lemma gives nearly optimal bounds to the value of m' [6]. Formally:

Lemma 1 (*Johnson–Lindenstrauss [42]*)**.** *For any $0 < \varepsilon < 1$ and any integer n, let m' be a positive integer such that*

$$m' > 4(\varepsilon^2/2 - \varepsilon^2/3)^{-1} \log n.$$

For any set V of n points in $\mathbb{R}^m$, there is a map function $f : \mathbb{R}^m \to \mathbb{R}^{m'}$ such that for all $u, v \in V$

$$(1-\varepsilon)\|u-v\|^2 \leq \|f(u) - f(v)\|^2 \leq (1+\varepsilon)\|u-v\|^2.$$

The interested reader will find two independent simplified versions of the proof of the above Lemma in [22,40] and extensions to other spaces and distances in [4,14,21,43]. It is possible to determine a function f that satisfies the Lemma with high probability, e.g., at least 2/3, in randomized polynomial time [22,40].

Since computing the projection into the new smaller space is a time-consuming task, several heuristic procedures have been proposed in the literature. Some of them are based on sparse projection matrices [3,15], while a more innovative and recent approach has been proposed by Ailon and Chazelle [4] with the addition of the Fast Fourier Transform to the Johnson–Lindenstrauss Lemma.

3.4. Null models

One of the main ingredients in hypothesis testing in statistics is the generation of data based on *null models*, e.g., for clustering, those models formalize the null hypothesis H_0 "no cluster structure in the data" [38,41]. The most relevant that have been proposed in the clustering literature [17,35,41,63] are introduced here. Moreover, we also give an indication of which one is well suited for microarray data analysis [25,66].

Unimodality hypothesis. A new dataset D' is generated as follows: the variables describing the items are randomly selected from a unimodal distribution (e.g. normal). This null model typically is not applied to microarray data, since it gives a high probability of rejection of the null hypothesis. For instance, that happens when the data are sampled from a distribution with a lower kurtosis than the normal distribution, such as the uniform distribution [63]. Fig. 1(b) reports an example of a dataset generated via the unimodality hypothesis.

Random graph hypothesis. The entries of the dissimilarity/distance matrix S are random. That is, one assumes that, in terms of a linear order relation capturing proximity, all the entries of the lower triangular part of S are equally likely, i.e., $S_{i,j} = \frac{1}{[n(n-1)/2]!}$ for $1 \leq i \leq n$ and $1 \leq j \leq i$. This null model is not applied to microarray data, since it does not preserve the distances that may be present among items.

Random label hypothesis. All permutations of the items are equally likely with respect to some characteristic, such as *a priori* class membership. In order to use this model, one needs to specify the *a priori* classification of the data. Each permutation has a probability $\frac{1}{n!}$. In particular, for microarray data, it coincides with the following:

- *Permutational Model* (`Pr` for short): a data matrix is generated by randomly permuting the elements within the rows and/or the columns of D.

In order to properly implement `Pr`, care must be taken in specifying a proper permutation for the data, since some similarity and distance functions are insensitive to permutations of coordinates. That is, although D' is a random permutation of D, it may happen that the distance or similarity among the points in D' is the same as in D, resulting in indistinguishable datasets for clustering algorithms. Some variants of this model have been studied for binary pattern matrices [44,65,69]. Moreover, it may not be suitable for microarray data with very small sample sizes (conditions), since one will not obtain enough observations (data points) to estimate the null model, even if one generates all possible permutations.

Random position hypothesis.. The items can be represented by points that are randomly drawn from a region $\mathcal{R}$ in m-dimensional space. In order to use this model, one needs to specify the region within which the points have to be uniformly distributed. Two instances applied to microarray data [25,31,66] are distinguished:

- *Poisson Model* (`Ps` for short): the region $\mathcal{R}$ is specified from the data. The simplest regions that have been considered are the m-dimensional hypercube and hypersphere enclosing the points specified by the matrix D [35]. Another possibility, in order to make the model more data-dependent, is to choose the convex hull enclosing the points specified by D. Fig. 1(c) reports an example of a dataset D' generated by `Ps`, where the region $\mathcal{R}$ (the box) is obtained from the dataset D reported in Fig. 1(a).
- *Poisson Model Aligned with Principal Components of the Data* (`Pc` for short): Tibshirani et al. [66], following Sarle [63], propose to align the region $\mathcal{R}$ with the principal components of the data matrix D. In detail, assuming that the columns of D have mean zero, let $D = UXV^T$ be its singular value decomposition. Let $\widehat{D} = DV$. One uses $\widehat{D}$ as in `Ps` to obtain a dataset $\widehat{D}'$. Then one back transforms via $D' = \widehat{D}'V^T$ to obtain the new dataset. Fig. 1(d) reports an example of a dataset D' generated by `Pc`, where the region $\mathcal{R}$ (the box) is obtained from the dataset D reported in Fig. 1(a).

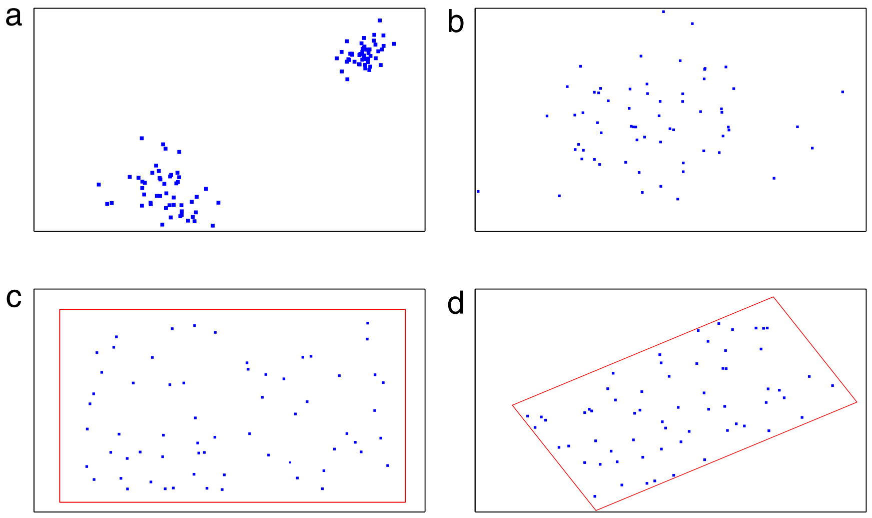

Fig. 1. (a) Dataset; (b) The unimodality hypothesis; (c) Ps; (d) Pc.

It is worth pointing out that, in a multivariate situation, one is not able to choose a generally applicable and useful reference distribution: the geometry of the particular null distribution matters [63,66]. Therefore, in two or more dimensions, and depending on the test statistic, the results can be very sensitive to the region of support of the reference distribution [25,63].

4. Comparing clustering partitions

External indices can be very useful in order to assess the similarity of two different partitions of the data. They are usually defined via a *contingency table* that contains all the basic information needed to quantify the similarity between two partitions. Based on the entries of that table, it is possible to define several external indices such as the Adjusted Rand Index [39], the F-index [59] and the Fowlkes and Mallows Index (FM-Index for short) [30]. It is worth pointing out that, in addition to the entries of the contingency table, the F-Index uses notions from information retrieval, such as precision and recall, in order to evaluate the level of agreement between the two given clustering solutions. The F and the FM-Index both have values in [0, 1], while the Adjusted Rand Index has an expected value of zero, its maximum is one and it can assume negative values [29,71]. The larger the value of each of those indices, the better agreement there is between the two partitions.

The most widespread use of external indices is for benchmarking purposes. Indeed, in order to assess how good is the performance of a clustering algorithm, one can proceed as follows. A dataset with a known *a priori* classification, usually referred to as a *gold standard solution* is given as input to the clustering algorithm. Then, the partition produced by the algorithm is compared with the gold standard solution via an external index, e.g., the Adjusted Rand Index. The higher the value of the index, the better the performance of the algorithm. The interested reader can find a detailed description of the use of external indices in cluster analysis in [25,31].

It is worth pointing out that many stability-based internal validation measures [10,13,18,25] use external indices to establish the similarity between two partitions, one of which is *assumed* to be the gold standard solution for the dataset, the real one being obviously unknown to the measure. Such a putative gold standard solution is obtained via the use of a classifier/clustering algorithm. The intuition behind such a somewhat unusual approach is that the consistency of the partitions produced by the measure with a given clustering algorithm must be assessed via a comparison with a partition produced by another clustering procedure, i.e., a classifier that has been trained only on part of the data. The interested reader will find a more detailed discussion about this point in [25].

5. The stability statistic paradigm

Recall from [41] that a *statistic* is a function of the data capturing useful information about it. In what follows, it is represented by a set S of records. For instance, in its simplest form, a statistic consists of a single real number, while in other cases of interest, it is a one- or two-dimensional array of real numbers.

A statistic assessing cluster stability is, intuitively, a measure of consistency of a clustering solution. The paradigm for the collection of a statistic on cluster stability is best presented as a procedure, reported in Fig. 2. Its input parameters and macro operations are described in abstract form in Figs. 3 and 4, respectively, while its basic steps are described below.

A single iteration of the **while** loop is discussed. The loop is repeated until the condition H is satisfied, i.e., until enough information about the given statistic has been collected. In step 3, a set of perturbed datasets is generated from D_0 by a DGP

```
STABILITY_STATISTIC($D_0, H, \alpha, \beta, \langle C_1, C_2, \ldots, C_t\rangle, k$)
1  $S^k = \emptyset$;
2  while $H$ do
3      $\langle D_1, D_2, \ldots, D_l\rangle \leftarrow \langle \mathrm{DGP}(D_0, \beta), \ldots, \mathrm{DGP}(D_0, \beta)\rangle$;
4      $\langle D_{T,0}, D_{T,1}, \ldots, D_{T,l}, D_{L,0}, D_{L,1}, \ldots, D_{L,l}\rangle \leftarrow$ SPLIT($\langle D_0, D_1, \ldots, D_l\rangle, \alpha$);
5      $\langle G\rangle \leftarrow$ ASSIGN($\langle D_{T,0}, D_{T,1}, \ldots, D_{T,l}\rangle, \langle C_1, C_2, \ldots, C_t\rangle$);
6      $\langle C_{i_1}, C_{i_2}, \ldots, C_{i_q}\rangle \leftarrow$ TRAIN($\langle G\rangle$);
7      $\langle \hat{G}\rangle \leftarrow$ ASSIGN($\langle D_{L,0}, D_{L,1}, \ldots, D_{L,l}\rangle, \langle C_1, C_2, \ldots, C_t\rangle$);
8      $\langle P_1, P_2, \ldots, P_z\rangle \leftarrow$ CLUSTER($\hat{G}, k$);
9      $u \leftarrow$ COLLECT_STATISTIC($\langle P_1, P_2, \ldots, P_z\rangle$);
10     $S^k \leftarrow S^k \bigcup \{u\}$;
11 return $S^k$;
```

Fig. 2. The STABILITY_STATISTIC procedure.

INPUT

- D_0: the input dataset.
- H: a test on the "adequacy" of a statistic S, i.e., it evaluates whether S contains enough information. Note that H could simply be a check as to whether a given number c of iterations has been reached. In what follows, this simple test is denoted as $\hat{H}_c$.
- α: a number in the range [0, 1].
- β: a sampling percentage, used by the DGP procedure (described in Section 3).
- $\langle C_1, C_2, \ldots, C_t\rangle$: a set of procedures, each of which is either a classifier or a clustering algorithm.
- k: it is the number of clusters into which a dataset has to be partitioned.

Fig. 3. List of the input parameters used in the STABILITY_STATISTIC procedure.

procedure. In step 4, D_0 and all the datasets generated in the previous step, are split into a learning and training datasets, according to the input parameter α. The next two steps train a subset of the classifiers on a subset of the training sets. Indeed, with reference to the discussion at the end of Section 4, when this step is performed, the classifiers will later be used to generate putative gold standard solutions against which the partitions of the clustering algorithms will be evaluated. In step 7, the bipartite graph $\hat{G}$ encodes the association between learning datasets and clustering procedures. In step 8, based on the association encoded by $\hat{G}$, the learning datasets are partitioned. Finally, in step 9, a statistic S^k is computed from those partitions and is given as output.

5.1. Instances

There are currently three incarnations of the stability statistic paradigm. The first is `Replicating Analysis`, a ground-breaking method due to Breckenridge [18]. The other two are `BagClust1` and `BagClust2`, due to Dudoit and Fridlyand [26]. In all three cases, the procedures have been proposed to improve a clustering solution for a fixed value of k, rather than to address the model selection problem. It is also worth mentioning that the procedures by Dudoit and Fridlyand [26] can be seen as extensions to clustering algorithms of the technique, proposed by Breiman [19], known in machine learning as *bootstrap aggregation* or *bagging*. `Replicating Analysis` and `BagClust2` play a key role in the design of stability-based measures. Therefore, we present them here, deferring the presentation of `BagClust1` to Appendix A, together with some variants of `Replicating Analysis`. Their presentation is organized as follows: for each of them, the input parameters setup is described first, taking as reference the ones defined in Fig. 3. Then, two reductions are given from the STABILITY_STATISTIC paradigm to the `Replicating Analysis` and `BagClust2` procedures, respectively.
The relevant incarnation of STABILITY_STATISTIC characterizing the method is detailed.

- `Replicating Analysis`.
  - *The input parameters setup*: β is not relevant and the simple test $\hat{H}_1$ is used to allow only one iteration of the **while** loop. Moreover, the set of procedures $\langle C_1, C_2, \ldots, C_t\rangle$ consists of one classifier and one clustering algorithm, referred to as C_1 and C_2, respectively.

MACRO OPERATIONS

- SPLIT: it takes as input a family of datasets $F_1, F_2, \ldots, F_w$ and a real number α in the range [0,1]. The procedure splits each F_i, $1 \leq i \leq w$, into two parts according to α, referred to as *learning* and *training* dataset and denoted with $F_{L,i}$ and $F_{T,i}$, respectively. That is, from each F_i, $\lceil \alpha n_i \rceil$ and $\lfloor (1-\alpha) n_i \rfloor$ rows are selected in order to obtain the corresponding $F_{T,i}$ and $F_{L,i}$, respectively, where n_i is the number of rows of F_i. Each $F_{T,i}$ and $F_{L,i}$ is given as output.
- ASSIGN: it takes as input a family of datasets and a set of procedures, each of which is either a classifier or a clustering algorithm. It returns a finite set of pairs in which the first element is a dataset and the second one is either a classifier or a clustering algorithm. Such an association is encoded via a bipartite graph G, where the nodes in one partition represent the datasets and the nodes in the other partition the procedures. Notice that the graph is not a matching, i.e., the same dataset can be assigned to different procedures and vice versa.
- TRAIN: it takes as input a set of pairs ⟨dataset, classifier⟩, encoded as a bipartite graph, analogous to the one just discussed. For each pair, it gives as output the classifier trained with the corresponding dataset. Notice that the number q of trained classifiers returned as output is equal to the number of edges in the input graph.
- CLUSTER: it takes as input a set of pairs ⟨dataset, classifier/clustering algorithm⟩ and a positive integer k. Again, the set is encoded as a bipartite graph. For each pair, it gives as output a partition into k clusters obtained by the classifier/clustering algorithm on the corresponding input dataset. Notice that the number z of partitions returned as output is equal to the number of edges in the input graph.
- COLLECT_STATISTIC: it takes as input a set of partitions. It returns as output the statistic computed on the input set.

Fig. 4. List of the macro operations used in the STABILITY_STATISTIC procedure.

```
REPLICATING_ANALYSIS($D_0$, $H_c$, $\alpha$, $\langle C_1, C_2 \rangle$, $k$)
1 for $h \leftarrow 1$ to $H_c$ do
2     Split the input dataset $D_0$ into $D_L$ and $D_T$, the learning and training sets, respectively;
3     Train the classifier $C_1$ on $D_T$;
4     Let $P_1$ and $P_2$ be the partitions of $D_L$ into $k$ clusters, obtained with the use of $C_1$ and $C_2$, respectively;
5     Let $e$ be the level of agreement between $P_1$ and $P_2$, as computed by an external index;
6 return $e$;
```

Fig. 5. The `Replicating Analysis` procedure.

- *The reduction from the* STABILITY_STATISTICS *procedure*: step 3 is not performed. In step 4, the SPLIT procedure is applied to D_0 only and it gives as output the training and learning dataset $D_{T,0}$ and $D_{L,0}$, respectively. Then, in steps 5–6, $D_{T,0}$ is used to train the classifier C_1. The learning dataset is used to build a classifier for the data, then to be used to derive "gold standard" partitions of the training set. In steps 7 and 8, two partitions P_1 and P_2 of $D_{L,0}$ are produced, by C_1 and C_2, respectively. Finally, in step 9, the COLLECT_STATISTIC procedure measures the agreement between the two partitions P_1 and P_2 via an external index, in order to assess the stability structure of the dataset.

For the convenience of the reader, as well as for future reference, the `Replicating Analysis` procedure is given in Fig. 5.

- `BagClust2`.
  - *The input parameters setup*: $\hat{H}_c$ is used as a test, for a given number of iterations c, e.g., 20 [26]. The set of procedures $\langle C_1, C_2, \ldots, C_t \rangle$ consists only of one clustering algorithm, $\alpha = 0$ and $\beta = 1$. Moreover, each DGP is an instance of the same subsampling method. Since $\alpha = 0$, the SPLIT procedure gives as output only the learning datasets, which are copies of the corresponding input dataset.
  - *The reduction from the* STABILITY_STATISTICS *procedure*: in step 3, a single DGP procedure is executed to generate D_1. Then, the SPLIT procedure takes as input D_0 and D_1 and it gives as output $D_{L,0} = D_0$ and $D_{L,1} = D_1$. In step 7, the bipartite graph $\hat{G}$ consists of only one node per partition, encoding the dataset D_1 and the clustering procedure, respectively. In step 8, a clustering partition is obtained from it. In step 9, the indicator matrix $I^{(h)}$ and connectivity matrix $M^{(h)}$ are computed. They are defined as follows: $I^{(h)}(p, q) = 1$, if items p and q are both in D_1 and zero

```
BagClust2($D_0$, $H_c$, $\beta$, $\langle C_1 \rangle$, $k$)
1 for $h \leftarrow 0$ to $H_c$ do
2     Generate a subsample $D_1$ from $D_0$;
3     Let $P_1$ be the partition of $D_1$ into $k$ clusters, obtained with the use of $C_1$;
4     Compute the indicator and connectivity matrix;
5 return $\mathcal{M}_{\mathcal{D}}$;
```

Fig. 6. The `BagClust2` procedure.

```
Stability_Measure($D$, $H$, $\alpha$, $\beta$, $\langle C_1, C_2, \dots, C_t \rangle$, $k_{min}$, $k_{max}$)
1 for $k \leftarrow k_{min}$ to $k_{max}$ do
2     $S^k \leftarrow$ Stability_Statistics($D, H, \alpha, \beta, \langle C_1, C_2, \dots, C_t \rangle, k$);
3     $R^k \leftarrow$ Synopsis($S^k$);
4 $k^* \leftarrow$ Significance_Analysis($R^{k_{min}}, \dots, R^{k_{max}}$);
5 return $k^*$;
```

Fig. 7. The Stability_Measure procedure.

Macro Operations

– Synopsis: it takes as input a statistic and returns as output a concise description of it.

– Significance_Analysis: it takes as input all the statistics/information collected, as returned by the Synopsis procedure. It computes the significance level of each statistic. It returns as output, explicitly or implicitly, a prediction about k^*. For instance, an implicit prediction of the value of k^* can be the plot of a histogram or of a curve, as in many methods described in this paper.

Fig. 8. List of the macro operations used in Stability_Measure procedure.

otherwise; $M^{(h)}(p, q) = 1$, if items p and q are in the same cluster and zero otherwise. Let $\mathbb{1}$ be a matrix of size $n \times n$ in which each entry is 1. Finally, in step 10, a dissimilarity matrix $\mathcal{M}_{\mathcal{D}}$ defined as:

$$\mathcal{M}_{\mathcal{D}} = \mathbb{1} - \frac{\sum_h M^{(h)}}{\sum_h I^{(h)}} \tag{1}$$

is computed. The dissimilarity matrix $\mathcal{M}_{\mathcal{D}}$ is then used as input to a clustering procedure in order to obtain a partition. For future use, it is worth pointing out that in an analogous way it is possible to compute a similarity matrix, which we refer to as a *consensus matrix*:

$$\mathcal{M}_{\mathcal{S}} = \frac{\sum_h M^{(h)}}{\sum_h I^{(h)}}. \tag{2}$$

For the convenience of the reader, as well as for future reference, the `BagClust2` procedure is given in Fig. 6.

6. The stability measure paradigm

In this section, the main paradigm of internal stability methods is described. It is best presented as a procedure, reported in Fig. 7. Its macro operations are described in abstract form in Fig. 8, while its basic steps are described below.

For each k in the range $[k_{min}, k_{max}]$, the paradigm collects the statistics S^k computed by the Stability_Statistic procedure. Then a concise description of S^k, denoted R^k, is computed via the Synopsis procedure. Finally, an explicit or implicit prediction of the value of k^* is computed by the Significance_Analysis procedure and is given as output. For the convenience of the reader, it is useful to recall that, in order to predict k^*, many internal validation methods provide only a curve, i.e., implicit information about k^*. Then, via its visual inspection, the user makes a prediction. A compendium of measures that resort to this well-known heuristic is given in [31,38]. On the other hand, for a few measures, e.g., `Clest` [25] and `Gap` [66], there exist theoretically sound automatic methods that identify k^*, i.e., the prediction is explicit. In addition, for some other measures, the prediction is automatic, but based on heuristic geometric observations [31,62]. For most of the stability-based methods, the identification of a theoretically sound automatic method for the explicit prediction of k^* is open and it is not clear that heuristic geometric approaches will yield appreciable results.

```
CLEST_SIGNIFICANCE_ANALYSIS($R^{k_{min}}, \ldots, R^{k_{max}}$)
1  for $i \leftarrow 0$ to $B_0$ do
2      $D^i \leftarrow DGP(D)$;
3      $S^h \leftarrow$ REPLICATING_ANALYSIS($D^i, H_c, \alpha, \langle C_1, C_2 \rangle, k$);
4      $Q_i^k \leftarrow median(S^1, \ldots, S^{B_0})$;
5  for $k \leftarrow k_{min}$ to $k_{max}$ do
6      $t_k^0 \leftarrow$ Compute the average of the $Q_i^k$ values;
7      $p_k \leftarrow$ Compute the p-value of $R^k$;
8      $d_k \leftarrow R^k - t_k^0$;
9  Define a set $K = \{k_{min} \le k \le k_{max} : p_k \le p_{max}$ and $d_k \ge d_{min}\}$;
10 if $K = \emptyset$
11   then $k^* \leftarrow 1$
12   else $k^* \leftarrow \arg\max_{k \in K} d_k$
13 return $k^*$;
```

Fig. 9. Implementation of the SIGNIFICANCE_ANALYSIS procedure proposed by Dudoit and Fridlyand for `Clest`.

```
CLEST($D, H_c, \alpha, \langle C_1, C_2 \rangle, k_{min}, k_{max}$)
1  for $k \leftarrow k_{min}$ to $k_{max}$ do
2      $S_k$ = REPLICATING_ANALYSIS($D, H_c, \alpha, \langle C_1, C_2 \rangle, k$);
3      $R_k \leftarrow median(S_k)$;
4  $k^* \leftarrow$ CLEST_SIGNIFICANCE_ANALYSIS($R^{k_{min}}, \ldots, R^{k_{max}}$);
5  return $k^*$;
```

Fig. 10. The CLEST procedure.

In the remaining part of this section, only two incarnations of the stability measure paradigm are detailed. The other ones proposed in the literature are discussed in Appendix B. For each method, the input parameters setup is described first, again taking as reference the ones defined in Fig. 3. Then, the STABILITY_STATISTIC and the STABILITY_MEASURE procedures are detailed.

6.1. Clest

This method, proposed by Dudoit and Fridlyand [25], generalizes in many aspects `Replicating Analysis` and can be regarded as a clever combination of hypothesis testing and resampling techniques. It estimates k^* by iterating the following: randomly partition the original dataset in a *learning* and *training* set, respectively. The learning set is used to build a classifier $\mathcal{C}$ for the data, which is then used to derive "gold standard" partitions of the training set. That is, the classifier is assumed to be a reliable model for the data. The "gold standard" is then used to assess the quality of the partitions of the training set obtained by a given clustering algorithm.

- *The input parameters setup*: it uses the same input parameters of `Replicating Analysis`, except for the test condition H where, in this case, c iterations of the **while** loop are allowed, for a given integer $c > 1$.
- *The* STABILITY_STATISTICS *procedure*: it corresponds to the one given for `Replicating Analysis`. Therefore, the set S^k of records is a one-dimensional array, in which each entry stores the value of the external index for the corresponding iteration.
- *The* SYNOPSIS *procedure*: it computes R^k as the median of the values stored in S^k.
- *The* SIGNIFICANCE_ANALYSIS *procedure*: it takes as input the collected statistics R^k, $k \in [k_{min}, k_{max}]$. With reference to Fig. 9, the procedure starts collecting B_0 statistics on data generated by the DGP procedure. Indeed, in step 1, a new dataset is generated via a null model. The remaining part of the procedure computes the significance level for each k. The variable p_{max} is a "significance level" threshold and d_{min} is a minimum allowed difference between "computed" and "expected" values. It is worth pointing out that the SIGNIFICANCE_ANALYSIS procedure provides an explicit prediction of k^*.

For the convenience of the reader, the `Clest` procedure is given in Fig. 10.

6.2. Consensus clustering

`Consensus`, by Monti et al. [53], is a reference method in internal validation measures, with a prediction power far better than other established methods [31,53].

```
CONSENSUS($D, H_c, \beta, \langle C_1\rangle, k_{min}, k_{max}$)
1 for $k \leftarrow k_{min}$ to $k_{max}$ do
2     $\mathcal{M}_{\mathcal{S}}{}^{k} = \mathbb{1} - $ BAGCLUST2($D, H_c, \beta, \langle C_1\rangle, k$);
3 Based on the $k_{max} - 1$ consensus matrices, return a prediction for $k^*$;
```

Fig. 11. The CONSENSUS procedure.

- *The input parameters setup*: $\hat{H}_c$ is used as a test, for a given number of iterations $c > 1$, α is not relevant, $\beta \in [0.6, 0.9]$ and the set of procedures $\langle C_1, C_2, \ldots, C_t\rangle$ consists only of one clustering algorithm C_1.
- *The* STABILITY_STATISTICS *procedure*: for a given number of clusters k, a consensus matrix $\mathcal{M}_{\mathcal{S}}{}^{(k)}$ (defined as in Eq. (2)) can be computed based on BagClust2. However, the DGP procedure is a full instance of subsampling, since $\beta < 1$.
- *The* SYNOPSIS *procedure*: it performs a copy of the collected statistic.
- *The* SIGNIFICANCE_ANALYSIS *procedure*: it provides an implicit estimation of k^*, as detailed below.

Based on experimental observations and sound arguments, Monti et al. [53] derive a "rule of thumb" in order to estimate the real number k^* of clusters present in D. Here we limit ourselves to present the key points, since the interested reader can find a full discussion in Monti et al. [53]. Let $\hat{m} = n(n-1)/2$, where n is the number of items to cluster, and let $\{x_1, x_2, \ldots, x_{\hat{m}}\}$ be the list obtained by sorting the entries of the consensus matrix. Moreover, let the empirical cumulative distribution CDF, defined over the range [0, 1], be:

$$CDF(f) = \frac{\sum\limits_{p<q} l\{\mathcal{M}_{\mathcal{S}}(p,q) \le f\}}{\hat{m}}$$

where f is a chosen constant in [0, 1] and l equals one if the condition is true and is zero otherwise.

For a given value of k, i.e., number of clusters, consider the CDF curve obtained by plotting the values of $CDF(x_i)$, $1 \le i \le \hat{m}$, with the use of the corresponding consensus matrix. In an ideal situation in which there are k clusters and the clustering algorithm is so good it can provide a perfect classification, such a curve is bimodal, with peaks at zero and one. Monti et al. [53] observe and validate experimentally that the area under the CDF curves is an increasing function of k. That result has also been confirmed by the experiments in Giancarlo et al. [31]. In particular, for values of $k \le k^*$, that area has a significant increase, while its growth flattens out for $k > k^*$. For instance, with reference to Fig. 12, one sees an increase in the area under the CDFs, for $k = 2, \ldots, 13$. The growth rate of the area is decreasing as a function of k and it flattens out for $k \ge k^* = 3$. The point in which such a growth flattens out can be taken as an indication of k^*. However, operationally, Monti et al. [53] propose a closely associated method, described next. For a given k, the area of the corresponding CDF curve is estimated as follows:

$$A(k) = \sum_{i=2}^{m} [x_i - x_{i-1}] CDF(x_i).$$

Again, $A(k)$ is observed to be an increasing function of k, with the same growth rate as the CDF curves. Now, let

$$\Delta(k) = \begin{cases} A(k) & k = 2, \\ \frac{A(k+1) - A(k)}{A(k)} & k > 2 \end{cases}$$

be the proportion increase of the CDF area as a function of k and as estimated by $A(k)$. Again, Monti et al. [53] observe experimentally that:

(i) For each $k \le k^*$, the area $A(k)$ markedly increases. This results in an analogous pronounced decrease of the Δ curve.
(ii) For $k > k^*$, the area $A(k)$ has no meaningful increases. This results in a stable plot of the Δ curve.

For the convenience of the reader, the Consensus procedure is given in Fig. 11.

7. A special case: the gap statistics

Although Gap [66] is not a stability-based internal validation measure, it can also be derived from the stability measure paradigm introduced in the previous section, as we now show. To this end, we need some definitions. Let $P_k = \{p_1, \ldots, p_k\}$ be a clustering solution with k clusters. Let

$$F_r = \sum_{j \in p_r} \|\sigma_j - \overline{\sigma_r}\|^2 \tag{3}$$

where $\overline{\sigma_r}$ is the centroid of cluster p_r. Then, Within Cluster Sum of Squares (WCSS, for short) is defined as:

$$\mathtt{WCSS}(k) = \sum_{r=1}^{k} F_r. \tag{4}$$

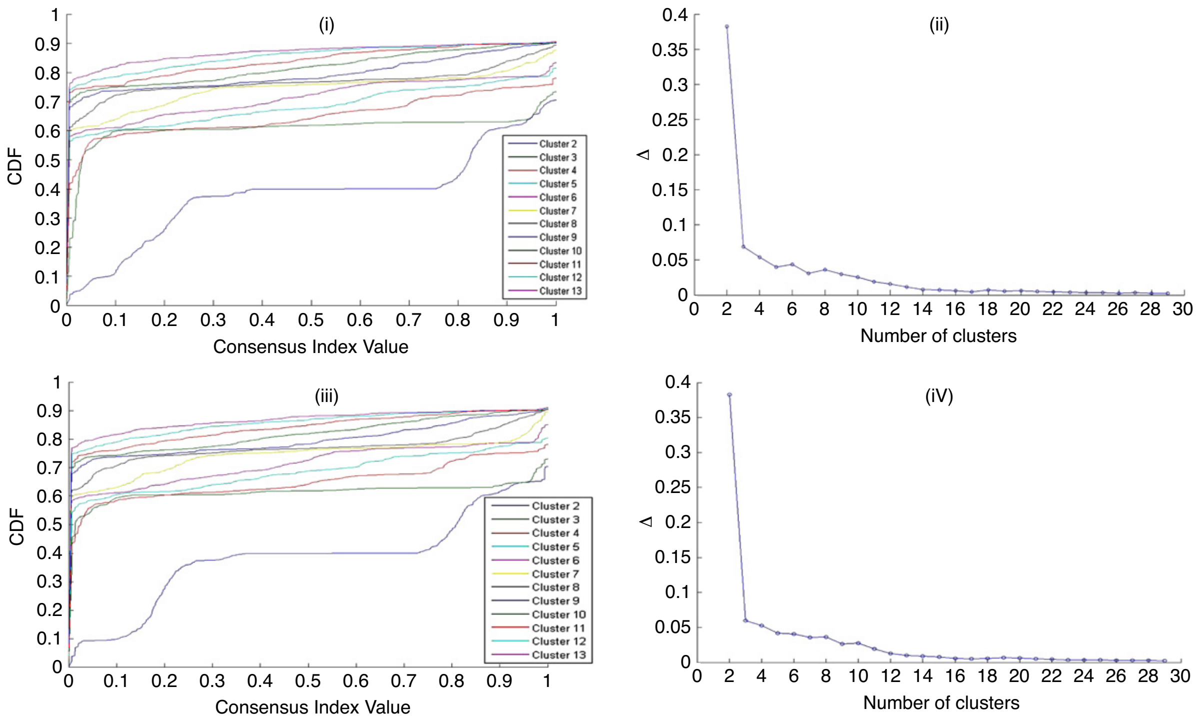

Fig. 12. The experiment is derived with the Leukemia dataset as input, a benchmark dataset for model selection [31], which is known to have a partition in three classes. We have used the well-known K-means clustering algorithm with Average Link initialization [41]. (i) The plot of the CDF curves as a function of k, obtained by `Consensus` with $H = 250$ and $p = 80\%$. For clarity, only the curves for k in $[2, 13]$ are shown. It is evident that there are increasing values of the area under the *CDF* for increasing values of k. The flattening effect in the growth rate of the area is evident for $k \geq k^* = 3$. Therefore, the method estimates the correct number of clusters in the dataset. (ii) The plot of the corresponding Δ curve for k in $[2, 30]$, where the flattening effect indicating k^* is evident for $k \geq k^* = 3$. (iii) The plot of the *CDF* curves, obtained by FC (described in Section 8) with $H = 250$ and $p = 80\%$, in analogy with (i). (iv) The plot of the Δ curve, obtained by FC with $H = 250$ and $p = 80\%$, in analogy with (ii).

It is a measure of cluster compactness and, among its many uses [45], there is also the one of internal validation measure [66]. In particular, `Gap` has been introduced in order to derive from `WCSS` an automatic prediction of k^* that is based on solid statistical ground. The intuition behind the method is brilliantly elegant. Consider the curves in Fig. 13. Curve at the bottom of the figure is the `WCSS` computed with the well known K-means with random initialization [41] on the dataset D with 2 clusters of Fig. 1(a). The curve at the top of the figure is the *average* `WCSS`, computed on ten datasets generated from the original data via the `Ps` null model, again using K-means with random initialization. As is evident from the figure, the curve on the top has a nearly constant slope: an expected behavior on datasets with no cluster structure in them. The vertical lines indicate the gap between the curve obtained from the null model and the one obtained from the input dataset, which supposedly has "cluster structure" in it. Since `WCSS` is expected to decrease sharply up to k^*, on the input dataset, and has a nearly constant slope on the null model datasets, the length of the vertical segments is expected to increase up to k^* and then to decrease. In fact, in the figure, if one takes as the prediction for k^* the first local maximum of the gap values (data not shown), one has $k^* = 2$, the correct number of classes in the dataset.

Normalizing the `WCSS` curves via logs and accounting also for the simulation error, such an intuition can be formalized as follows.

Let $\log(\texttt{WCSS}(k))$ be the statistic S^k used to assess how reliable is a clustering solution with k clusters. The value of that statistic is computed on both the observed data and on data generated by a suitably chosen null model. Then, rather than returning a p-value, the procedure returns the first k for which "the gap" between the observed and the expected statistic is at a local maximum.

- `Gap`.
  - *The input parameters setup*: $\hat{H}_1$, $\alpha = 0$, β is not relevant and the set of clustering procedures $\langle C_1, C_2, \ldots, C_t\rangle$ consists of only one clustering algorithm C_1.
  - *The* Stability_Statistics *procedure*: it gives as output the $\log \texttt{WCSS}(k)$ value, for k in $[k_{min}, k_{max}]$, computed by partitioning D into k clusters, with the use of C_1.
  - *The* Synopsis *procedure*: it return a copy of $\log \texttt{WCSS}(k)$.
  - *The* Significance_Analysis *procedure*: it is a Monte Carlo simulation. Indeed, for a given value of k in $[k_{min}, k_{max}]$, $\log \texttt{WCSS}(k)$ for D is compared to B_0 values of $\log \texttt{WCSS}(k)$, computed by clustering B_0 artificial datasets. In turn,

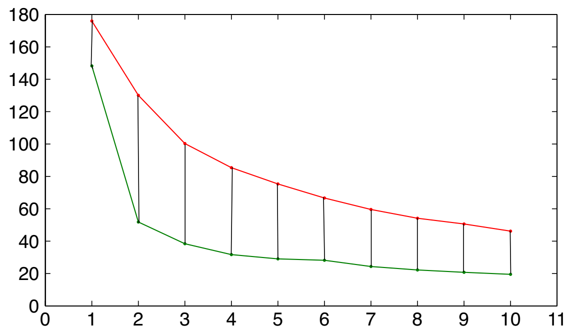

Fig. 13. A geometric interpretation of Gap.

```
GAP_SIGNIFICANCE_ANALYSIS($R^{k_{min}}, \dots, R^{k_{max}}$)
1  for $k \leftarrow k_{min}$ to $k_{max}$ do
2      for $i \leftarrow 1$ to $B_0$ do
3          Compute a new data matrix $D^i$, using the chosen null model;
4          Compute a clustering solution $P_{k,i}$ on $D^i$ using algorithm $C_1$;
5          $Q_i^k \leftarrow$Compute log(WCSS($k$)) on $P_{k,i}$;
6      Compute $Gap(k) = \frac{1}{B_0} \sum_{i=1}^{B_0} Q_i^k - R^k$;
7      Compute the standard deviation $sd(k)$ of the set of numbers $\{R_1^k, \dots, R_{B_0}^k\}$;
8      $s(k) = \left(\sqrt{1 + \frac{1}{B_0}}\right) sd(k)$;
9  $k^*$ is the first value of $k$ such that $Gap(k) \geq Gap(k+1) - s(k+1)$;
10 return $k^*$;
```

Fig. 14. Implementation of the SIGNIFICANCE_ANALYSIS procedure for Gap.

```
GAP($D, C_1, k_{min}, k_{max}$)
1  for $k \leftarrow k_{min}$ to $k_{max}$ do
2      Compute a clustering solution $P_k$ on $D$ using algorithm $C_1$;
3      $R^k \leftarrow$Compute log(WCSS($k$)) on $P_k$;
4  $k^* \leftarrow$ GAP_SIGNIFICANCE_ANALYSIS($R^{k_{min}}, \dots, R^{k_{max}}$);
5  return $k^*$;
```

Fig. 15. The Gap procedure.

each artificial dataset is generated via the DGP procedure which, in this case, is an instance of null models. For the convenience of the reader, the pseudo-code of the procedure is given in Fig. 14. With reference to step 9 of that procedure, it is worth pointing out that the term $s(k+1)$ is a heuristic adjustment that takes into account the Monte Carlo simulation error in the estimation of the expected value of $\log(\texttt{WCSS}(k))$ [66].

For the convenience of the reader, the Gap procedure is given in Fig. 15.

8. Approximations

In this section, we present heuristic speed-ups of the stability-based measures considered in this paper. The need for those speed-ups is evident, as pointed out in [32,47]. The idea is to use algorithms that "approximate" the computations involved in the stability-based measures, in the hope that this will grant a speed-up with no substantial loss in predictive accuracy. It is worth pointing out that the idea of using approximation algorithms in order to speed-up internal validation measures has been introduced in [31,32] and is presented in a homogeneous way in [67]. Although theoretic results assessing the performance guarantee of the heuristic algorithms would be highly desirable, they are not available in the current literature. Such a lack, together with the difficulty of fulfilling it, grants that experimental validation of the heuristics is a good way of assessing their predictive performance. In what follows, we provide a general approximation paradigm of the stability-based internal validation measures. For the convenience of the reader, we describe this general approximation paradigm by first providing a speed-up (denoted FC) of one of the internal validation measure detailed in Section 6, i.e., Consensus, which will allow us to give an intuitive description of the inefficiencies of stability-based methods and of ways to eliminate

```
FC($D$, $H_c$, $\beta$, $\langle C_1 \rangle$, $k_{min}$, $k_{max}$)
1 for $i \leftarrow 1$ to $H_c$ do
2     Generate, via subsampling, a data matrix $D_i$;
3     Compute the indicator matrix $I^{(i)}$;
4     for $k \leftarrow k_{min}$ to $k_{max}$ do
5         Let $P_1$ be the partition of $D_i$ into $k$ clusters, obtained with the use of $C_1$;
6         Based on $P_1$, compute the connectivity matrix $M_k^{(i)}$;
7 for $k \leftarrow k_{min}$ to $k_{max}$ do
8     Compute the consensus matrix $\mathcal{M}_{\mathcal{S}}{}^k$;
9 Based on the $k_{max} - 1$ consensus matrices, return a prediction for $k^*$;
```

Fig. 16. The FC procedure.

at least some of them. Then, based on that intuition, we outline how to derive the approximation paradigm from the general stability paradigm.

We start from a simple observation about the data generation in Consensus. Intuitively, a large number of clustering solutions, each obtained via a sample of the original dataset, seem to be required in order to identify the correct number of clusters. Indeed, each of the $(k_{max} - k_{min} + 1) \times H$ clustering solutions needed is computed from a *different* sample of the input dataset. However, there is no theoretic reason indicating that those clustering solutions must each be generated from distinct samples. It has been also observed that such an approach leads to costly duplications of computations [32], in particular when Consensus is used in conjunction with Hierarchical clustering algorithms [41]. Indeed, the ability to quickly compute a clustering solution with k clusters from one with $k + 1$, typical of these agglomerative clustering methods, cannot be used within Consensus because, for each k, the dataset changes. The same holds true for divisive methods.

FC has been proposed with the goal of avoiding those computational duplications. To this end, it performs first a sampling step to generate a new dataset D_1, which is then used to generate all clustering solutions, for k in the range $[k_{min}, k_{max}]$. In terms of code and with reference to Fig. 6, that implies a modification of BagClust2 as follows. The subsampling step is no longer performed and the procedure must now generate $k_{max} - k_{min} + 1$ connectivity matrices from a single dataset it receives as input. When C_1 is an agglomerative Hierarchical algorithm, the computation of the connectivity matrices can be done incrementally following the same order of generation as the clustering solutions produced by the Hierarchical algorithm. That is, once BagClust2 has been so modified, it becomes possible to interleave the computation of the required connectivity matrices with the level bottom-up construction of the hierarchical tree underlying the clustering algorithms. Specifically, only one dendogram construction is required rather than the repeated and partial construction of dendograms as in the Consensus procedure. Therefore, one uses, in full, the main characteristic of agglomerative clustering algorithms. Again, analogous considerations hold for divisive methods. For the convenience of the reader, the pseudo-code for FC is provided in Fig. 16. It is worth mentioning that the rule of thumb for the determination of k^*, with the use of FC, is identical to the one described for Consensus.

In general, such an interleaving can be accomplished for the Stability_Measure procedure via a simple switch of two main loops, i.e., step 1 in Fig. 7 with step 2 in Fig. 2. The resulting approximation paradigm, formalized by the procedure Fast_Stability_Measure, is given in Fig. 17. The macro operations and inputs are the same used for the Stability_Measure procedure.

9. Experiments

For completeness, we provide here an experimental analysis of the internal stability measures by assessing, for each, their ability to estimate the correct number of clusters in a dataset and their computational time. In particular, we focus our analysis on the following measures: ME (described in Appendix B), Gap, CLEST, Consensus and FC based on the ValWorkBench software library [2] and MOSRAM (also described in Appendix B) based on the mosclust software library [68]. It is worth pointing out that the results reported here are a synopsis of much more extensive benchmarkings that the interested reader can find in [31,32,67]. The experimental setup is described in Section 9.1, while Section 9.2 provides the corresponding results.

9.1. Experimental setup: datasets and parameters

9.1.1. Datasets

We use three datasets, each being a matrix in which each row corresponds to an element to be clustered and each column to an experimental condition. They have been widely used for validation studies [24,25,31,32,67]. For conciseness,

```
FAST_STABILITY_MEASURE($D_0, H, \alpha, \beta, \langle C_1, C_2, \ldots, C_t\rangle, k_{min}, k_{max}$)
1  while $H$ do
2      $\langle D_1, D_2, \ldots, D_l\rangle \leftarrow \langle \mathrm{DGP}(D_0,\beta), \mathrm{DGP}(D_0,\beta), \ldots, \mathrm{DGP}(D_0,\beta)\rangle$;
3      for $k \leftarrow k_{min}$ to $k_{max}$ do
4          $\langle D_{T,0}, D_{T,1}, \ldots, D_{T,l}, D_{L,0}, D_{L,1}, \ldots, D_{L,l}\rangle \leftarrow$ SPLIT($\langle D_0, D_1, \ldots, D_l\rangle, \alpha$);
5          $\langle G\rangle \leftarrow$ ASSIGN($\langle D_{T,0}, D_{T,1}, \ldots, D_{T,l}\rangle, \langle C_1, C_2, \ldots, C_t\rangle$);
6          $\langle C_{i_1}, C_{i_2}, \ldots, C_{i_q}\rangle \leftarrow$ TRAIN($\langle G\rangle$);
7          $\langle \hat{G}\rangle \leftarrow$ ASSIGN($\langle D_{L,0}, D_{L,1}, \ldots, D_{L,l}\rangle, \langle C_1, C_2, \ldots, C_t\rangle$);
8          $\langle P_1, P_2, \ldots, P_z\rangle \leftarrow$ CLUSTER($\hat{G}, k$);
9          $u \leftarrow$ COLLECT_STATISTIC($\langle P_1, P_2, \ldots, P_z\rangle$);
10         $S^k \leftarrow S^k \bigcup \{u\}$;
11 for $k \leftarrow k_{min}$ to $k_{max}$ do
12     $R^k \leftarrow$ SYNOPSIS($S^k$);
13 $k^* \leftarrow$ SIGNIFICANCE_ANALYSIS($R^{k_{min}}, \ldots, R^{k_{max}}$)
14 return $k^*$;
```

Fig. 17. The FAST_STABILITY_MEASURE procedure.

we mention only some relevant facts about them. It is worth recalling from Section 2 that a gold standard for a dataset is a partition of the data into a number of classes known *a priori*. Membership in a class is established by assigning the appropriate class label to each element.

Leukemia: It is a 38×100 data matrix, where each row corresponds to a patient with acute leukemia and each column to a gene. For this dataset, there is a partition into three classes and we take that as the gold standard. The dataset comes from [36] and it has been obtained from the original microarray experiments described in [33].

Lymphoma: It is an 80×100 data matrix, where each row corresponds to a tissue sample and each column to a gene. The dataset comes from the study of Alizadeh et al. [5] on the three most common adult lymphoma tumors. There is a partition into three classes and we take that as the gold standard. The dataset has been obtained from the original microarray experiments as described by Dudoit and Fridlyand [25].

NCI60: It is a 57×200 data matrix, where each row corresponds to a cell line and each column to a gene. This dataset originates from a microarray study in gene expression variation among the sixty cell lines of the National Cancer Institute anti-cancer drug screen [1]. There is a partition of the dataset into eight classes and we take that as the gold standard. The dataset has been obtained from the original microarray experiments as described by Dudoit and Fridlyand [25].

9.1.2. *Input parameters of internal stability measures*

In this section, we describe the input parameters setup, used in our experiments, and with reference to the the ones listed in Fig. 3, with the addition of k_{min}:

- $k_{min} = 2$.
- $k_{max} = 30$ for all methods, except for `Clest`, where $k_{max} = 15$ is used due to the relatively small size of the datasets in relation with the sampling technique of the method.
- $\alpha = 66\%$, where it needs to be used.
- $\beta = 80\%$, where it needs to be used.
- $\langle C_1, C_2, \ldots, C_t\rangle$: we use the Hierarchical Average Link clustering algorithm [41] (Hier-A for short) and diagonal linear discriminant analysis as a classifier [27].
- The test used to assess if enough information about the given statistic has been collected, is H_c for all the measures. In particular, $H_c = 250$ is for `Consensus` and `FC`, $H_c = 100$ is used for `ME` and `MOSRAM`, and $H_c = 20$ is used for `CLEST`. Finally, in the SIGNIFICANCE_ANALYSIS procedures of `CLEST` and `Gap` a `Ps` null model is used as instances of the DGP procedures and $B_0 = 20$.

9.1.3. *Hardware*

All experiments for the assessment of the precision of each measure were performed in part on a 64-bit AMD Athlon 2.2 GHz bi-processor with 1 GB of main memory running Windows Server 2003 and in part on state-of-the-art PCs. All the timing experiments refer to the bi-processor, where one processor per run has been used. We also point out that all the Operating Systems supervising the computations have a 32 bits precision.

Table 1
A summary of the results for each measure on all datasets, with the use of Hier-A as a clustering algorithm. Each cell in the table displays a precision result. That is, the prediction of the number of clusters in a dataset given by a measure. A number in a circle with a black background indicates a prediction in agreement with the number of classes in the dataset; while a number in a circle with a white background indicates a prediction that differs, in absolute value, by 1 from the number of classes in the dataset; a number not in a circle indicates the remaining predictions.

	Precision		
	Leukemia	NCI60	Lymphoma
`FC`	❸	❽	❸
`ME`	1	4	1
`MOSRAM`	②	2	②
`Gap`	④	1	6
`Clest`	❸	3	②
`Consensus`	❸	❽	❸
Gold standard	**3**	**8**	**3**

Table 2
A summary of the timing results, in milliseconds, for each measure on all datasets, with the use of Hier-A as a clustering algorithm.

	Timing		
	Leukemia	NCI60	Lymphoma
`FC`	2.7×10^4	7.0×10^4	6.8×10^4
`ME`	2.3×10^5	7.6×10^5	6.4×10^5
`Gap`	1.4×10^5	6.1×10^5	6.4×10^5
`Consensus`	7.9×10^5	2.0×10^6	1.9×10^6

9.2. Precision results

The results are reported in Table 1. It is worth pointing out that given the ability of `MOSRAM` to detect multi-level structures possibly present in the dataset, only the estimated number of clusters with highest p-value are reported in that table. Although synoptic, those results represent quite well the advantages and shortcomings of the mentioned stability-based measures that have been established by much broader benchmarking experiments [31,32]: `Consensus` and `FC` are the measures providing the most reliable predictions, while the prediction rules of `ME` and `Gap` are really weak.

9.3. Timing Results

The results are reported in Table 2. We do not report the times of `MOSRAM` and `LD01` since they are comparable to those obtained for `ME` and `Consensus`, respectively. Moreover, the time of `CLEST` is not provided since a smaller range has been used to produce clustering solutions for all datasets, due to their size. Again, those results are quite representative of the state-of-the-art [31,32]. `Consensus` is the slowest measure. Although there is a significant difference in terms of prediction, the time performance is practically the same for `ME`, `Gap` and `CLEST`. Moreover, it is worth pointing out that there is at least one order of magnitude difference between `FC` and the other measures.

Accounting also for the precision results, `FC` seems to be the measure of choice, as indicated in [32].

10. Conclusions

A general algorithmic paradigm for stability-based internal validation measures has been introduced and can be seen as a generalization of earlier work by Breckenridge [18], Breiman [19] and Valentini [68]. Moreover, it is also shown that each of the known stability-based measures is an instance of such a novel paradigm. Surprisingly, `Gap` also falls within the new paradigm. Moreover, from this general algorithmic paradigm, it is simple to design new stability-based internal measures combining the building blocks of the measures detailed. Finally, a general algorithmic paradigm is proposed that describes heuristic and very effective speed-ups known in the literature for stability-based model selection methods.

Acknowledgements

Both authors are very thankful to Margaret Gagie for reading a draft version of this paper and by providing many helpful comments that resulted in a better presentation of their ideas.

```
BagClust1($D_0$, $H_c$, $\beta$, $\langle C_1\rangle$, $k$)
1 for $i \leftarrow 1$ to $H_c$ do
2     Generate a bootstrap sample $D_1$ from $D_0$;
3     Let $P_1$ and $P_2$ be the corresponding partitions of $D_0$ and $D_1$ into $k$ clusters, obtained with the use of $C_1$;
4     Permute the elements assigned to the partition $P_2$ so that there is maximum overlap with $P_1$;
5     Let $O_c$ be the number of overlapping elements;
6 return $O_c$;
```

Fig. A.18. The BagClust1 procedure.

Appendix A. Additional instances of the stability statistic paradigm

A.1. BagClust1

- *The input parameters setup*: as in BagClust2.
- *The reduction from the* Stability_Statistics *procedure*: in step 3, a single DGP procedure is executed to generate D_1. Then, the Split procedure takes as input D_0 and D_1 and it gives as output $D_{L,0} = D_0$ and $D_{L,1} = D_1$. The learning dataset is used to build a classifier for the data, then to be used to derive "gold standard" partitions of the training set. In steps 7 and 8, the clustering procedure is applied to both D_0 and D_1, in order to obtain the partitions P_1 and P_2, respectively. The Collect_Statistic procedure permutes the elements assigned to the partition P_2 so that there is the maximum overlap with P_1. Based on observations by Roth et al. (see Appendix A.2.2), a maximum overlap between two partitions can be found by solving a dual optimization problem corresponding to the computation of a minimum weighted perfect matching in bipartite graphs [54]. For each iteration of the **while** loop, the number of overlapping elements are counted and given as output of the method. From that statistic, a new partition is obtained by assigning each element of D_0 to a cluster via a majority vote system. That is, each element is assigned to the cluster for which it has expressed the maximum of number of preferences.

For the convenience of the reader, the BagClust1 procedure is given in Fig. A.18.

A.2. Variants of Replicating Analysis

Variants of Replicating Analysis have been proposed within procedures for model selection. However, they may be of independent interest as cluster validity procedures, although their performance has not been evaluated for that particular application. Therefore, we single them out here. The first one is due to Ben-Hur et al. [10], while the second one is due to Roth et al. [61].

A.2.1. Replicating_ME

- *The input parameters setup*: $\hat{H}_c$ is used as a test, for a given number of iterations c, e.g., 20 [26]. The set of procedures $\langle C_1, C_2, \ldots, C_t\rangle$ consists of one clustering algorithm, $\alpha = 0$ and $\beta = 1$. Moreover, each DGP is an instance of the same subsampling method.
- *The reduction from the* Stability_Statistics *procedure*: in step 3, D_1 and D_2 are generated by two DGP procedures, where each procedure is an instance of subsampling. Since $\alpha = 0$, the Split procedure copies those datasets into the corresponding learning datasets, while steps 5 and 6 are not performed. In step 7, the graph $\hat{G}$, obtained as output of the Assign procedure, encodes two relations: $\langle D_{L,1}, C_1\rangle$ and $\langle D_{L,2}, C_1\rangle$. In step 8, two clustering solutions P_1 and P_2 are obtained from $\langle D_{L,1}, C_1\rangle$ and $\langle D_{L,2}, C_1\rangle$, respectively. The Collect_Statistic procedure computes the level of agreement between the two partitions via an external index, but restricted to the common elements of D_1 and D_2. In step 10, this level of agreement is stored into a one dimensional array S^k. That is, for each iteration of the **while** loop, the value returned by the external index is stored in the corresponding entry of S^k.

For the convenience of the reader, the Replicating_ME procedure is given in Fig. A.19.

A.2.2. Replicating_RLBB02.

- *The input parameters setup*: $\hat{H}_c$ is used as a test, for a given number of iterations $c > 1$, $\alpha = 0.5$, $\beta = 0$ and the set of procedures $\langle C_1, C_2, \ldots, C_t\rangle$ consists of three clustering algorithms C_1, C_2 and C_3. In particular, C_3 produces a random partition of D by first placing k randomly selected items into separate clusters and then by placing the remaining items into the same clusters uniformly and at random.

```
REPLICATING_ ME($D_0$, $H_c$, $\alpha$, $\beta$, $\langle C_1\rangle$, $k$)
1 for $i \leftarrow 1$ to $H_c$ do
2     Generate, via subsampling, two data matrices $D_1$ and $D_2$;
3     Let $P_1$ and $P_2$ be the corresponding partitions of $D_1$ and $D_2$ into $k$ clusters, obtained with the use of $C_1$,
      respectively;
4     Let $S^k(i)$ be the level of agreement between $P_1$ and $P_2$, computed by an external index but restricted to the
      common elements of $D_1$ and $D_2$;
5 return $S^k$;
```

Fig. A.19. The `Replicating_ME` procedure.

```
REPLICATING_RLBB02($D$, $H_c$, $\alpha$, $\beta$, $\langle C_1, C_2, C_3\rangle$, $k$)
1 for $i \leftarrow 1$ to $H_c$ do
2     Split the input dataset $D$ into $D_L$ and $D_T$, the learning and training sets, respectively;
3     Partition $D_T$ into $k$ clusters, with the use of $C_3$, in order to obtain $P_{T,i}$;
4     Partition $D_L$ into $k$ clusters, with the use of $C_1$ and $C_2$, in order to obtain $P_1$ and $P_2$, respectively;
5     Find the correct permutation with the use of the minimum weighted perfect bipartite matching;
6     $u \leftarrow$Compute the number of misclassified elements and normalize it with reference to the random case;
7     $S^k \leftarrow S^k \bigcup \{u\}$;
8 return $S^k$;
```

Fig. A.20. The Replicating_RLBB02 procedure.

– *The reduction from the* STABILITY_STATISTICS *procedure*: step 3 is not performed. In step 4, the SPLIT procedure is applied to D_0 only, and it gives as output two datasets $D_{T,0}$ and $D_{L,0}$ of equal size. In step 6, $D_{T,0}$ is partitioned by C_3 in order to obtain a partition $P_{T,i}$ which, as mentioned later, is used for normalization purposes. In steps 7 and 8, two partitions P_1 and P_2 of $D_{L,0}$ are produced, by C_1 and C_2, respectively. The COLLECT_STATISTIC procedure is the same proposed for `BagClust1`, where, in this case, it minimizes the number of misclassified elements. Indeed, assuming P_1 as the gold standard, COLLECT_STATISTIC gives as output the number of misclassified elements in P_2 computing a minimum weighted perfect bipartite matching [54]. Finally, the number of misclassified elements is normalized with respect to the random case, i.e., $P_{T,i}$.

For the convenience of the reader, the `Replicating_RLBB02` procedure is given in Fig. A.20.

Appendix B. Additional instances of the stability measure paradigm

B.1. Model explorer

This method (`ME` for short), by Ben-Hur et al. [10], is the simplest incarnation of the stability measure paradigm. It can be derived as follows.

– *Input parameters setup*: as in `Consensus`.
– *The* STABILITY_STATISTICS *procedure*: as in `Replicating_Analysis_ME`.
– *The* SYNOPSIS *procedure*: it performs a copy of the collected statistic.
– *The* SIGNIFICANCE_ANALYSIS *procedure*: it provides an implicit estimation of k^*, as we detail next.

For each $k \in [k_{min}, k_{max}]$, the values stored in R^k are histogrammed. Then, the optimal number of clusters k^* is predicted to be the lowest value of k such that the R^k values distribution is close to one and the R^{k+1} value distribution is in a wider range of values. An example of the number of clusters prediction is given in Fig. B.22, where `ME` is computed on the dataset of Fig. 1(a) with the K-means clustering algorithm with a random initialization, for $k \in [2, 5]$.

For the convenience of the reader, the `ME` procedure is given in Fig. B.21.

B.2. MOSRAM

This method, by Bertoni and Valentini [13], is strongly related to `ME`, where the most significant change is in the SIGNIFICANCE_ANALYSIS procedure. Indeed, it estimates automatically k^* and, in addition, it detects multi-level structures possibly present in D, i.e., hierarchical structure - see Fig. B.24 for an example. It can be derived from the stability measure paradigm as follows.

```
ME($D$, $H_c$, $\langle C_1 \rangle$, $k_{min}$, $k_{max}$)
1 for $k \leftarrow k_{min}$ to $k_{max}$ do
2     $S^k \leftarrow$ REPLICATING_ ME($D$, $H_c$, 0, 1, $\langle C_1 \rangle$, $k$);
3     $R^k \leftarrow S^k$;
4 Plot separately the histogram of the values in $R^k$ and return a prediction for $k^*$;
```

Fig. B.21. The ME procedure .

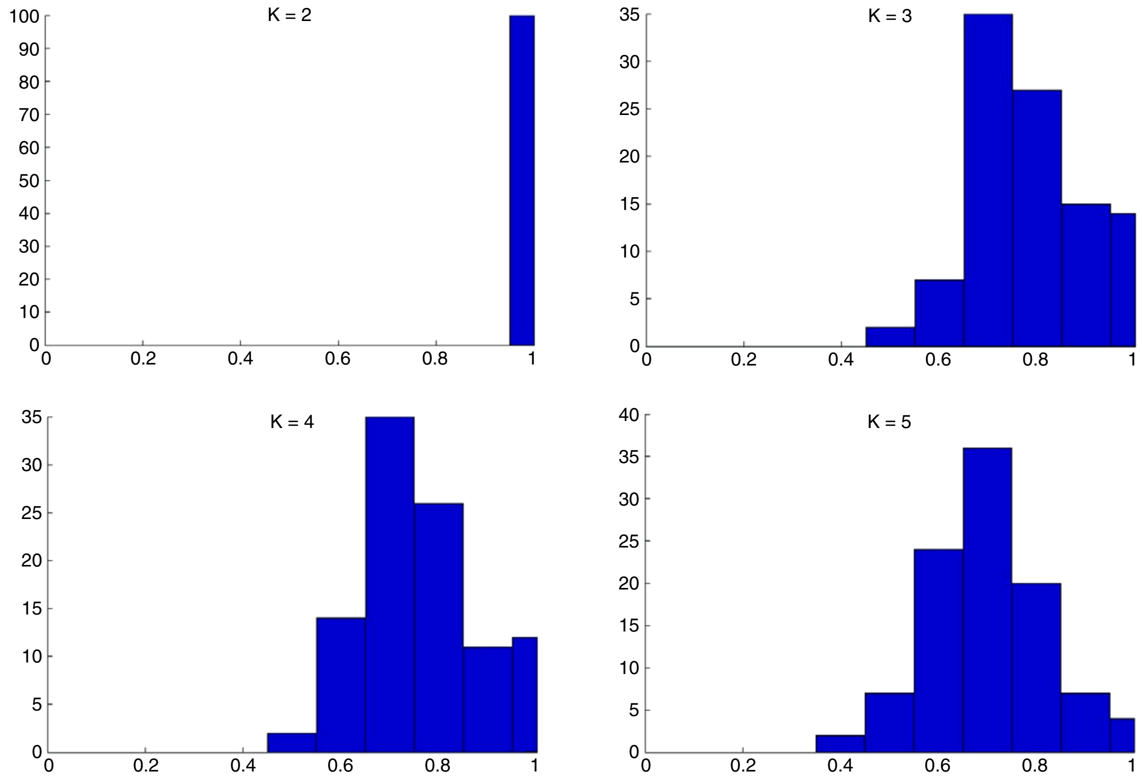

Fig. B.22. The histograms plotting the R^k values distribution for increasing values of k. The prediction of k^* corresponds to the correct number of cluster, i.e., $k^* = 2$.

- *The input parameters setup*: as in `Consensus`.
- *The* STABILITY_STATISTICS *procedure*: it is the same proposed in ME, except that the two DGP procedures performed in step 1 are both an instance of randomized dimensionality reduction methods. Once we have accounted for that difference, we still use the notation ME to refer to this procedure.
- *The* SYNOPSIS *procedure*: it gives as output R^k, the average of the statistics S^k.
- *The* SIGNIFICANCE_ANALYSIS *procedure*: intuitively, if the value of R^k is close to 1, then the clustering solution is stable. Moreover, in order to detect significant and possibly multi-level structures that are simultaneously present in D, a statistical hypothesis test is applied. In particular, a χ^2-based test is used in order to estimate k^*, as follows.

Let $R = \{R^{k_{min}}, \ldots, R^{k_{max}}\}$ and let τ be a significance level. The null hypothesis H_0 considers the set of k-clusterings as equally reliable, while the alternative hypothesis H_1 considers the set of k-clusterings as not equally reliable. When H_0 is rejected at τ significance level, it means that at least one k-clustering significantly differs from the others. The procedure sorts the values in R, and a χ^2-based test is repeated until no significant difference is detected or the only remaining clustering is the top-ranked in R. At each iteration, if a significant difference is detected, the bottom-ranked value is removed from the set R. Therefore, the SIGNIFICANCE_ANALYSIS procedure gives as output the set of the remaining (top sorted) k-clusterings that correspond to the set of estimates of the "true" number of clusters (at τ significance level).

For the convenience of the reader, the *MOSRAM* procedure is given in Fig. B.23.

B.3. Levine and Domany

This method, due to Levine and Domany [50], is strongly related to `BagClust2`.

```
MOSRAM($D$, $H_c$, $\langle C_1\rangle$, $k_{min}$, $k_{max}$)
1 ME($D$, $H_c$, $\langle C_1\rangle$, $k_{min}$, $k_{max}$);
2 Perform an average of the statistcs collected by ME;
3 Perform a $\chi^2$-based test in order to estimate $k^*$;
```

Fig. B.23. The *MOSRAM* procedure.

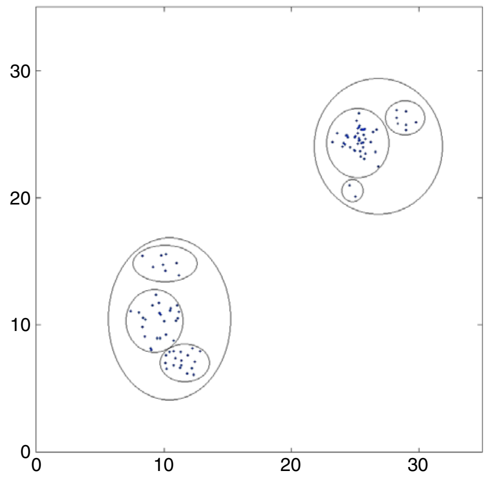

Fig. B.24. An example of hierarchical structures in a dataset.

```
LD01($D$, $H$, $\langle C_1\rangle$, $k_{min}$, $k_{max}$)
1 for $k \leftarrow k_{min}$ to $k_{max}$ do
2     Let $P_1$ be the partition of $D$ into $k$ clusters, obtained with the use of $C_1$;
3     Based on $P_1$, compute the connectivity matrix $S_0^k$;
4     for $i \leftarrow 1$ to $H_c$ do
5         Generate, via subsampling, a data matrix $D_1$;
6         Let $P_1$ be the partition of $D_1$ into $k$ clusters, obtained with the use of $C_1$;
7         Based on $P_1$, compute the connectivity matrix $S_i^k$;
8 Compute $R^k$, as defined in (B.1);
```

Fig. B.25. The Levine and Domany procedure.

- *The input parameters setup*: as in Consensus.
- *The* Stability_Statistics *procedure*: for each iteration, the method computes, as statistic, a connectivity matrix in which each entry is 1, if the two elements are in the same cluster and 0 otherwise. Moreover, the collected statistic S^k is a set of matrices, $S^k = \{S_0^k, \ldots, S_c^k\}$. Matrix S_0^k corresponds to the connectivity matrix for D_0, and matrix S_i^k, for $1 \le i \le c$, corresponds to the connectivity matrix for the dataset D_1, which is generated by the DGP subsampling procedure at the corresponding iteration.
- *The* Synopsis *procedure*: it compares the collected statistic, via the following formula:

$$R^k = \langle\langle \delta_{S_0^k, S_i^k} \rangle\rangle \tag{B.1}$$

  where $\langle\langle \cdot \rangle\rangle$ is a twofold averaging. That is, for each S_i^k, an average is computed over all pairs which are in the same cluster in the original dataset and which have both been selected in the same resampling step. Then, an average for all S_i^k is computed.
- *The* Significance_Analysis *procedure*: it selects as k^* the value of k with the local maximum of R^k, for $k \in [k_{min}, k_{max}]$.

For the convenience of the reader, the procedure proposed by Levine and Domany is given in Fig. B.25.

B.4. Roth et al.

In analogy with `Clest`, this method, by Roth et al. [61], also generalizes `Replicating Analysis`.

```
RLBB02($H_c$, $\langle C_1, C_2, C_3 \rangle$, $D$, $k_{min}$, $k_{max}$)
1 for $k \leftarrow k_{min}$ to $k_{max}$ do
2     $S^k \leftarrow$ Replicating_Analysis_RLBB02($D$, $H_c$, 0.5, 0, $\langle C_1, C_2, C_3 \rangle$, $k$);
3     Compute the average of $S^k$ and compute the expected (in)-stability value;
4 $k^* \leftarrow$ the value of $k$ with the minimum "expected (in)-stability" value;
5 return $k^*$;
```

Fig. B.26. The Roth et al. procedure.

– *The input parameters setup*: $\hat{H}_c$ is used as a test, for a given number of iterations c, $\alpha = 0.5$, β is not relevant and the set of clustering procedures $\langle C_1, C_2, \ldots, C_t \rangle$ consists of three clustering algorithms C_1, C_2 and C_3 .
– *The* Stability_Statistics *procedure*: as in `Replicating_Analysis_RLBB02`.
– *The* Synopsis *procedure*: it computes the average over the assignment cost and it computes the "expected (in)-stability" value defined as the expectation with respect to the two different datasets.
– *The* Significance_Analysis *procedure*: it gives as k^* the value of k with the minimum "expected (in)-stability" value.

For the convenience of the reader, the procedure proposed by Roth et al. is given in Fig. B.26.

References

[1] NCI 60 Cancer Microarray Project. http://genome-www.stanford.edu/NCI60.
[2] Validation Work Bench: Valworkbench web page. http://www.math.unipa.it/~raffaele/valworkbench/.
[3] D. Achlioptas, Database-friendly random projections: Johnson-Lindenstrauss with binary coins, Journal of Computer and System Sciences 66 (2003) 671–687.
[4] N. Ailon, B. Chazelle, Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform, in: STOC '06: Proceedings of the 38th Annual ACM Symposium on Theory of Computing, ACM, 2006, pp. 557–563.
[5] A.A. Alizadeh, M.B. Eisen, R.E. Davis, C. Ma, I.S. Lossos, A. Rosenwald, J.C Boldrick, H. Sabet, T. Tran, X. Yu, J.I Powell, L. Yang, G.E. Marti, T. Moore, J. Hudson Jr, L. Lu, D.B. Lewis, R. Tibshirani, G. Sherlock, W.C. Chan, T.C. Greiner, D.D. Weisenburger, J.O. Armitage, R. Warnke, R. Levy, W. Wilson, M.R. Grever, J.C Byrd, D. Botstein, P.O. Brown, L.M. Staudt, Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling, Nature 403 (2000) 503–511.
[6] N. Alon, Problems and results in extremal combinatorics—II, Discrete Mathematics 308 (2008) 4460–4472.
[7] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, A.J. Levine, 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 of the United States of America 96 (1999) 6745–6750.
[8] B. Andreopoulos, A. An, X. Wang, M. Schroeder, A roadmap of clustering algorithms: finding a match for a biomedical application, Briefings in Bioinformatics 10 (3) (2009) 297–314.
[9] S. Ben-David, U. von Luxburg, D. Pál, A sober look at clustering stability, in: Lecture Notes in Computer Science, vol. 4005, 2006, p. 5.
[10] A. Ben-Hur, A. Elisseeff, I. Guyon, A stability based method for discovering structure in clustering data, in: Seventh Pacific Symposium on Biocomputing, ISCB, 2002, pp. 6–17.
[11] J. Benesty, D. Morgan, M. Sondhi, A better understanding and an improved solution to the problems of stereophonic acoustic echo cancellation, in: ICASSP '97: Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '97, vol. 1, IEEE Computer Society, 1997, p. 303.
[12] A. Bertoni, G. Valentini, Randomized maps for assessing the reliability of patients clusters in DNA microarray data analyses, Artificial Intelligence in Medicine 37 (2006) 85–109.
[13] A. Bertoni, G. Valentini, Model order selection for bio-molecular data clustering, BMC Bioinformatics 8 (2007).
[14] A. Bhattacharya, P. Kar, M. Pal, On low distortion embeddings of statistical distance measures into low dimensional spaces, in: DEXA, 2009, pp. 164–172.
[15] E. Bingham, H. Mannila, Random projection in dimensionality reduction: applications to image and text data, in: KDD '01: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2001, pp. 245–250.
[16] M. Bittner, P. Meltzer, Y. Chen, Y. Jiang, E. Seftor, M. Hendrix, M. Radmacher, R. Simon, Z. Yakhini, A. Ben-Dor, N. Sampas, E. Dougherty, E. Wang, F. Marincola, F, C. Gooden, J. Lueders, A. Glatfelter, P. Pollock, J. Carpten, E. Gillanders, D. Leja, K. Dietrich, C. Beaudry, M. Berens, D. Alberts, V. Sondak, Molecular classification of cutaneous malignant melanoma by gene expression profiling, Nature 406 (2000) 536–540.
[17] H.H. Bock, On some significance tests in cluster analysis, Journal of Classification 2 (1985) 77–108.
[18] J.N. Breckenridge, Replicating cluster analysis: method, consistency, and validity, Multivariate Behavioral Research 24 (2) (1989) 147–161.
[19] L. Breiman, Bagging predictors, Machine Learning 24 (1996) 123–140.
[20] J.-P. Brunet, P. Tamayo, T.R. Golub, J.P. Mesirov, Metagenes and molecular pattern discovery using matrix factorization, Proceedings of the National Academy of Sciences of the United States of America 101 (2004) 4164–4169.
[21] G. Cormode, M. Datar, P. Indyk, S. Muthukrishnan, Comparing data streams using Hamming norms (how to zero in), IEEE Transactions on Knowledge and Data Engineering 15 (2003) 529–540.
[22] S. Dasgupta, A. Gupta, An elementary proof of a theorem of Johnson and Lindenstrauss, Random Structures & Algorithms 22 (2003) 60–65.
[23] P. D'haeseleer, How does gene expression cluster work? Nature Biotechnology 23 (2006) 1499–1501.
[24] V. Di Gesú, R. Giancarlo, G. Lo Bosco, A. Raimondi, D. Scaturro, Genclust: a genetic algorithm for clustering gene expression data, BMC Bioinformatics 6 (2005) 289.
[25] S. Dudoit, J. Fridlyand, A prediction-based resampling method for estimating the number of clusters in a dataset, Genome Biology 3 (2002).
[26] S. Dudoit, J. Fridlyand, Bagging to improve the accuracy of a clustering procedure, Bioinformatics 19 (9) (2003) 1090–1099.
[27] S. Dudoit, J. Fridlyand, T.P. Speed, Comparison of discrimination methods for the classification of tumors using gene expression data, Journal of the American Statistical Association 97 (457) (2002) 77–87.
[28] B. Efron, R.J. Tibshirani, An Introduction to the Bootstrap, Chapman & Hall, London, 1993.
[29] D. Fisher, P. Hoffman, The adjusted rand statistic: a SAS macro, Psychometrika 53 (1988) 417–423.
[30] E.B. Fowlkes, C.L. Mallows, A method for comparing two hierarchical clusterings, Journal of the American Statistical Association 78 (1983) 553–584.
[31] R. Giancarlo, D. Scaturro, F. Utro, Computational cluster validation for microarray data analysis: experimental assessment of clest, consensus clustering, figure of merit, gap statistics and model explorer, BMC Bioinformatics 9 (2008) 462.
[32] R. Giancarlo, F. Utro, Speeding up the Consensus Clustering methodology for microarray data analysis, Algorithms for Molecular Biology 6 (2011) 1.

[33] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeeck, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, E.S. Lander, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 286 (531) (1999) 531–537, 5439.
[34] A.D. Gordon, Clustering algorithms and cluster validation, in: P. Dirschedl, R. Ostermann (Eds.), Computational Statistics, Physica-Verlag, Heidelberg, Germany, 1994, pp. 503–518.
[35] A.D. Gordon, Null models in cluster validation, in: From Data to Knowledge: Theoretical and Practical Aspects of Classification, Springer-Verlag, 1996, pp. 32–44.
[36] J. Handl, J. Knowles, D.B. Kell, Computational cluster validation in post-genomic data analysis, Bioinformatics 21 (15) (2005) 3201–3212.
[37] M.H. Hansen, W.N. Hurwitz, W.G. Madow, Sample Survey Methods and Theory Methods and Applications, vol. 1, Wiley, 1993.
[38] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer, 2003.
[39] L. Hubert, P. Arabie, Comparing partitions, Journal of Classification 2 (1985) 193–218.
[40] P. Indyk, R. Motwani, Approximate nearest neighbors: towards removing the curse of dimensionality, in: STOC '98: Proceedings of the 30th Annual ACM Symposium on Theory of Computing, ACM, 1998, pp. 604–613.
[41] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Englewood Cliffs, 1988.
[42] W.B. Johnson, J. Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space, Contemporary Mathematics 26 (1984) 189–206.
[43] W.B. Johnson, A. Naor, The Johnson-Lindenstrauss lemma almost characterizes Hilbert space, but not quite, in: SODA, 2009, pp. 885–891.
[44] C.W. Harper Jr., Groupings by locality in community ecology and paleoecology: tests of significance, Lethaia 11 (1978) 251–257.
[45] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, Wiley, New York, 1990.
[46] M.K. Kerr, G.A. Churchill, Bootstrapping cluster analysis: assessing the reliability of conclusions from microarray experiments, PNAS 98 (2000) 8961–8965.
[47] J. Kraus, H. Kestler, A highly efficient multi-core algorithm for clustering extremely large datasets, BMC Bioinformatics 11 (2010).
[48] D.D. Lee, H.S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (1999) 788–791.
[49] D.D. Lee, H.S. Seung, Algorithms for non-negative matrix factorization, in: NIPS, 2000, pp. 556–562.
[50] E. Levine, E. Domany, Resampling method for unsupervised estimation of cluster validity, Neural Computation 13 (2001) 2573–2593.
[51] L.M. McShane, M.D. Radmacher, B. Freidlin, R. Yu, M.-C. Li, R. Simon, Methods for assessing reproducibility of clustering patterns observed in analyses of microarray data, Bioinformatics 18 (2002) 1462–1469.
[52] T. Mehta, M. Tanik, D.B. Allison, Towards sound epistemological foundations of statistical methods for high-dimensional biology, Nature Genetics 36 (2004) 943–947.
[53] S. Monti, P. Tamayo, J. Mesirov, T. Golub, Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data, Machine Learning 52 (2003) 91–118.
[54] C.H. Papadimitriou, K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, 1982.
[55] C.M. Perou, S.S. Jeffrey, M. van de Rijn, C.A. Rees, M.B. Eisen, D.T. Ross, A. Pergamenschikov, C.F. Williams, S.X. Zhu, J.C.F. Lee, D. Lashkari, D. Shalon, P.O. Brown, D. Botstein, Distinctive gene expression patterns in human mammary epithelial cells and breast cancers, Proceedings of the National Academy of Sciences of the United States of America 96 (1999) 9212–9217.
[56] J.R. Pollack, C.M. Perou, A.A. Alizadeh, M.B. Eisen, A. Pergamenschikov, C.F. Williams, S.S. Jeffrey, D. Botstein, P.O. Brown, Genome-wide analysis of DNA copy-number changes using cDNA microarrays, Nature Genetics 23 (1999) 41–46.
[57] R. Giancarlo, D. Scaturro, F. Utro, Statistical indices for computational and data driven class discovery in microarray data, in: J.Y. Chen, S. Lonardi (Eds.), Biological Data Mining, CRC Press, San Francisco, USA, 2009, pp. 295–335.
[58] Y. Raviv, N. Intrator, Bootstrapping with noise: An effective regularization technique, Connection Science 8 (1996) 355–372.
[59] C. Van Rijsbergen, Information Retrieval, 2nd edition, Butterworths, London, 1979.
[60] D.T. Ross, U. Scherf, M.B. Eisen, C.M. Perou, P. Spellman, V. Iyer, S.S. Jeffrey, M. van de Rijn, M. Walthama, A. Pergamenschikov, J.C.F. Lee, D. Lashkari, D. Shalon, T.G. Myers, J.N. Weistein, D. Botstein, P.O. Brown, Systematic variation in gene expression patterns in human cancer cell lines, Nature Genetics 24 (2000) 227–235.
[61] V. Roth, T. Lange, M. Braun, J. Buhmann, A resampling approach to cluster validation, in: Proceedings 15th Symposium in Computational Statistics, 2002, pp. 123–128.
[62] S. Salvador, P. Chan, Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms, in: 16th IEEE International Conference on Tools with Artificial Intelligence, 2004, ICTAI 2004, 2004, pp. 576–584.
[63] W.S. Sarle, Cubic clustering criterion, Technical report, SAS, 1983.
[64] M. Smolkin, D. Ghosh, Cluster stability scores for microarray data in cancer studies, BMC Bioinformatics 4 (2003).
[65] R.E. Strauss, Statistical significance of species clusters in association analysis, Ecology 63 (1978) 634–639.
[66] R. Tibshirani, G. Walther, T. Hastie, Estimating the number of clusters in a dataset via the gap statistics, Journal Royal Statistical Society B. 2 (2001) 411–423.
[67] F. Utro, Algorithms for internal validation clustering measures in the Post-genomic era, Doctoral Dissertation, University of Palermo. http://arxiv.org/abs/1102.2915v1, 2011.
[68] G. Valentini, Mosclust: a software library for discovering significant structures in bio-molecular data, Bioinformatics 23 (2007) 387–389.
[69] A. Vassiliou, L. Ignatiades, M. Karydis, Clustering of transect phytoplankton collections with a quick randomization algorithm, Journal of Experimental Marine Biology and Ecology 130 (1989) 135–145.
[70] R.D. Wolfinger, G. Gibson, E.D. Wolfinger, L. Bennet, H. Hamadeh, C.A. Bushel, R.S. Paules, Assessing gene significance from cDNA microarray expression data via mixed models, Journal of Computational Biology (2001) 625–637.
[71] K.Y. Yeung, Cluster analysis of gene expression data, Ph.D. Thesis, University of Washington, 2001.