Significant distinct branches of hierarchical trees:

A framework for statistical analysis and applications to biological data

A Dissertation Presented

by

Guoli Sun

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

December 2014

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

Stony Brook University

The Graduate School

Guoli Sun

We, the dissertation committee for the above candidate for the

Doctor of Philosophy degree, hereby recommend

acceptance of this dissertation.

Alexander Krasnitz – Dissertation Advisor
Assistant Professor, Simons Center for Quantitative Biology,
Cold Spring Harbor Laboratory

Stephen Finch - Chairperson of Defense
Professor, Department of Applied Mathematics and Statistics,
Stony Brook University

Wei Zhu – Dissertation Co-Advisor
Professor, Department of Applied Mathematics and Statistics,
Stony Brook University

Seungtai Yoon – Committee Member
Research Assistant Professor,
Cold Spring Harbor Laboratory

This dissertation is accepted by the Graduate School

Charles Taber
Dean of the Graduate School

Abstract of the Dissertation

Significant distinct branches of hierarchical trees:

A framework for statistical analysis and applications to biological data

by

Guoli Sun

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

2014

One of the most common goals of hierarchical clustering is finding those branches of a tree that form quantifiably distinct data subtypes. Achieving this goal in a statistically meaningful way requires (a) a measure of distinctness of a branch and (b) a test to determine the significance of the observed measure, applicable to all branches and across multiple scales of dissimilarity.

We formulate a method termed Tree Branches Evaluated Statistically for Tightness (TBEST) for identifying significantly distinct tree branches in hierarchical clusters. For each branch of the tree a measure of distinctness, or tightness, is defined as a rational function of heights, both of the branch and of its parent. A statistical procedure is then developed to determine the significance of the observed values of tightness. We test TBEST as a tool for tree-based data partitioning by applying it to five benchmark datasets, one of them synthetic and the other four each from a different area of biology. With each of the five datasets, there is a well-defined partition of the data into classes. In all test cases TBEST performs on par with or better than the existing techniques.

One dataset uses Cores Of Recurrent Events (CORE) to select features. CORE was developed with my participation in the course of this work. An R language

implementation of the method is available from the Comprehensive R Archive Network: cran.r-project.org/web/packages/CORE/index.html.

Based on our benchmark analysis, TBEST is a tool of choice for detection of significantly distinct branches in hierarchical trees grown from biological data. An R language implementation of the method is available from the Comprehensive R Archive Network: cran.r-project.org/web/packages/TBEST/index.html.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my deepest thanks to my advisor, Professor Alexander Krasnitz, for supporting me during these past five years. I am grateful to Alex for his support and encouragement, and also his extensive knowledge in cancer genomics, computer science, physics and statistics. He provides very useful guidance and keen insights whenever I have difficulties during my research. Also, thanks to him, I can have the opportunity to perform statistical analysis in the cutting-edge quantitative biology lab. It is my great honor to work at his lab in Cold Spring Harbor Laboratory. With regards to my research at Cold Spring Harbor Laboratory, I also thank Professor Michael Wigler for discussions on research work at early stages, Peter Andrews and Todd Heywood who helped me on technical issues in implementation, and all other staffs from Wigler Lab. I would like to take this opportunity to thank Martin Akerman, Joan Alexander, Timour Baslan for generously sharing their data with us, and to Theo A. Knijnenburg for generously providing software. This work was supported by the National Institutes of Health grantNIH/1UO1CA168409-01 and by grant 125217 from the Simons Foundation.

I would like to thank Professor Stephen Finch, Professor Wei Zhu, and Professor Seungtai Yoon for being on my dissertation committee.

Last but not least, I would like to express my gratitude to my beloved family, for their unconditional support.

Chapter 1 Introduction

This dissertation presents a method for identifying distinct substructures of data based on hierarchical clustering. Hierarchical clustering, as the name suggests, builds a hierarchy of clusters, i.e. groups of observations. Hierarchical clustering has a number of useful properties. First of all, hierarchical structure with N-1 clusters is derived from N observations. It does not need a number of clusters specified in advance and provides a tree-like organization of the data. Each cluster is combined with, or split from the rest of the tree based on a quantitative measure called dissimilarity. Secondly, hierarchical clustering lends itself easily to visualization of a hierarchical tree with labels for observations.

Taking advantage of the second property, most commonly, application of hierarchical clustering consists of visual examination, and intuitive identification of sub-trees that appear clearly distinct from the rest of the tree. Obviously, results of such qualitative analysis and conclusions from it can be observer-dependent. Quantifying the interpretation of hierarchical trees and introducing mathematically and statistically well-defined criteria for distinctness of sub-trees would therefore be highly beneficial and is the focus of this work.

A method was designed in the course of this work for identifying statistically distinct subsets of hierarchically clustered data. Termed Tree Branches Evaluated Statistically for Tightness, or TBEST in the following, the performance of the method was thoroughly studied in comparison with existing methods for the same or similar purposes, and found to be superior in its ability to reproduce known meaningful partitions of biological data. The detailed description of TBEST in the following is an expansion of its briefer description given in our recent publication (Sun and Krasnitz 2014).

One of the data sets used to evaluate TBEST was generated using a novel statistical tool for the analysis of interval data. The tool, termed Cores Of Recurrent Events (CORE), was developed with my participation in the course of this work. Its brief description here follows our publication (Krasnitz, Sun et al. 2013).

This dissertation has the following structure. Introductory Chapter 1 provides background on clustering analysis (1.1), existing methods designed to find distinct branches (1.2) and measures to evaluate clustering (1.3). Chapter 2 addresses data preprocessing techniques and discusses several options of preparing data before clustering. In Chapter 3, we introduce the method TBEST. The performance of this method is studied in comparison to existing methods on data sets from a variety of biological various origins in Chapter 4. Implementation of TBEST is discussed in Chapter 5. Chapter 6 is devoted to discussion and conclusions.

Appendix includes: A time complexity and performance analysis, Figure S1, Figure S2 and Table S1. Brief introduction of content in these four materials: 1) a comparison of time complexity and performance for TBEST, SC, SLB and DTC, 2) Figure S1, an 11-panel figure illustrating null distribution of tightness, 3) Figure S2, a comparison of empirical p-value estimates for tightness to EVT-based estimates and 4) Table S1, detailing the properties of the Simulated6 dataset.

1.1 Clustering analysis

Clustering is otherwise known as unsupervised learning. Division of learning methods into supervised and unsupervised ones is based on the availability and existence of response variables, also called class labels.

To distinguish clustering from classification, a general problem of classification is posed as follows: Given the predictor variables/features X, and a categorical response variable Y, what is the relationship between X and Y? A simple binary classification example is, knowing which patients have heart disease (1) or not (0), and this is the response variable Y, fit a model $\delta(Y|X)$ that predicts the occurrence of heart disease for variables/features X such as blood pressure, age, etc. Examples of $\delta(Y|X)$ include logit in logistic regression or majority vote for K-nearest neighbors (KNN) (Altman 1992). This set of problems belongs to supervised learning.

Clustering, on the other hand, finds subsets of data based on similarities between observations in the absence of known class labels Y. Quantitative measures of similarity among observations depend on their properties X and are discussed in the following. Subsets generated by this procedure are called clusters. Clustering often helps to learn meaningful class labels from the data. For example, to be discussed in greater detail in the following, similarities among patterns of somatic mutations in the genomes of individual cells can be used to discover the clonal structure of the population from which the sample of cells is drawn. Another example is segmentation of online customer pool according to the similarity among patterns X of the customer online These clusters can be used to predict behavior of future customers.

The most common clustering approaches are centroid based clustering (1.1.1) and hierarchical clustering (1.1.2). Instead of similarity among observations, centroid based clustering use similarity between observation and the so-called "centroid", defined in 1.1.1.

1.1.1 Centroid based clustering

For these clustering method the number of clusters needs to be specified in advance. The algorithms then seek a partition into clusters that maximizes the within-cluster similarity. Two representative examples of this set of algorithms are *K*-means and Partition Around Medoids (PAM). These can be briefly described as follows.

In *K*-means method, given n observations $X = \{X_1, X_2, \dots X_n\}$, each of p real-valued variables, a partition S of X with given number of K parts, $S = \{S_1, S_2, \dots S_K\}$ is found by minimizing the objective function

$$\mathrm{F} = \sum_{i=1}^{K} \sum_{X_j \in s_i} \left\| X_j - \mu_i \right\|_2^2$$

where μ_i is the mean vector of observations in S_i, with dimension p. Note mean vector μ_i is centroid, and Euclidean norm measures similarity between observation and centroid. In the commonly used Lloyd's iterative algorithm the K centroids are found for the given cluster assignments, followed by reassignment of each observation to the nearest centroid. The procedure is repeated until the assignment no longer changes (Lloyd 2006). There is no guarantee that the absolute minimum of F will be found, and K-means may

fail, for example, by choosing a centroid at the middle point between two obvious clusters.

PAM, or *K*-medoids (Kaufman 1990), is designed as a more robust centroid based clustering algorithm. Instead of choosing mean as the centroid of a cluster μ_i, PAM chooses median among the observations. This choice is known as a medoid. To make the optimal choice of medoids, PAM compares the objective function for the current medoids with that for randomly selected non-medoids. In each iteration, PAM has a trade-off of computing time. It exhaustively compares current medoid with non-medoids in $O(n(n-K)K)$, while *K*-means only uses $O(nK)$.

Although there are other variations of *K*-means, such as *K*-medians (Jain 1988, Bradley 1997), and Sparse *K*-means (Witten 2010), the requirement to specify the number *K* in advance remains an inevitable limitation of centroid based clustering algorithms.

1.1.2 Hierarchical clustering

Hierarchical clustering builds a hierarchy of groups of data based on quantitative similarity measurements. The measure that hierarchical clustering uses is namely Dissimilarity and Linkage (1.1.2.1). Unlike centroid based clustering, there is no need to specify the number of clusters. A hierarchical structure is built while samples are merged or split into clusters:

Agglomerative: A bottom-up approach. All individual observations are listed on the bottom. The first cluster contains the pair of individual observations that has least dissimilarity. One more pair of observations is merged at each step until every observation is combined into one cluster at top.

Divisive: A top-down approach. All individual observations are listed in one cluster at the beginning. Splits are performed recursively until every observation is in its singleton cluster at bottom.

Agglomerative hierarchical clustering is more widely used and is more time efficient then divisive hierarchical clustering. We focus on agglomerative hierarchical clustering, and refer to it as HC in short from now on.

1.1.2.1 Agglomerative Rule

Unlike centroid-based methods, HC does not require the number of clusters to be pre-specified. Another advantage of HC is it generates a hierarchical tree structure. This agglomerative hierarchical tree, also known as dendrogram, grows from the bottom-up with chosen dissimilarity measures. The dendrogram provides visual picture of how items are merged into clusters. This section provides the definition and example of HC dissimilarity measures, and how to use a connectivity matrix/dissimilarity matrix to grow a hierarchical tree.

Dissimilarity:

Definition: The dissimilarity metric, or dissimilarity is defined between any two observations X_i and X_j, $i \neq j$, $i,j \in [1,n]$. Commonly used choices are:

Euclidean distance $\left\|X_i - X_j\right\|_2$

Manhattan distance $\left\|X_i - X_j\right\|_1$

maximum distance $\left\|X_i - X_j\right\|_\infty$

cosine dissimilarity (1-uncentered Pearson's Correlation) $1 - \frac{\langle X_i, X_j \rangle}{\|X_i\|_2 \left\|X_j\right\|_2}$

1 – Pearson's Correlation $1 - \frac{\sum_{m=1}^{p}(X_{im} - \overline{X_i})(X_{jm} - \overline{X_j})}{\sqrt{\sum_{m=1}^{p}(X_{im} - \overline{X_i})^2}\sqrt{\sum_{m=1}^{p}(X_{jm} - \overline{X_j})^2}}$

There are many other dissimilarity metrics, such as 1 – Kendall's Correlation, 1 – Spearman's Correlation. Measurements of correlation coefficient have range from -1 to 1, the correlation-based dissimilarity metrics therefore have range from 0 to 2.

Dissimilarity metric defined in HC need not be a distance function. Correlation-based dissimilarity metric may violate the triangle inequality, i.e. $g(x, y) + g(y, z) \geq + g(y, z$ where x, y and z are observations with p dimensions.

An example is given below using Kendall's correlation. The dissimilarity metric is 1 – Kendall's Correlation,

$$\tau = \begin{cases} 1 - Tau_a & no\ ties \\ 1 - Tau_b & otherwise \end{cases}$$

$$Tau_a = \frac{number\ of\ concordant\ pairs - number\ of\ discordant\ pairs}{\frac{1}{2}n(n-1)}$$

$$Tau_b = \frac{number\ of\ concordant\ pairs - number\ of\ discordant\ pairs}{\sqrt{[\frac{1}{2}n(n-1) - n_1][\frac{1}{2}n(n-1) - n_2]}}$$

A concordant pair of observations means the order of inequality is consistent through p dimensions. Otherwise, the pair is discordant. While in Tau_b, n_1 and n_2 are numbers of pairs tied in each observation. Given three observations with rankings, O = (1, 1, 1, 2, 3), P = (2, 2, 2, 1, 1), Q = (1, 2, 1, 2, 3). $\tau(PQ) = 1.722$, $\tau(OQ) = 0.198$, $\tau(OP) = 1.926$,

$$\tau(PQ) + \tau(OQ) < \tau(OP)$$

This dissimilarity metric is not a distance function as it fails to satisfy the triangle inequality.

Linkage:

Definition: Linkage is defined as dissimilarity measure between an individual observation and a cluster, or a pair of clusters. Let two clusters, C and S, each contain individual observations, indexed, respectively by c and s. With this notation, commonly used choices of linkage are:

complete linkage (furthest neighbor) $\max\{Dissimilarity(s,c): \forall\ s \in S, c \in C\}$

single linkage (nearest neighbor) $\min\{Dissimilarity(s,c): \forall\ s \in S, c \in C\}$

average linkage $\frac{\sum_{s\in S}\sum_{c\in C} Dissimilarity(s,c)}{\#\ of\ objects\ in\ S * \#\ of\ objects\ in\ C}$

Ward's linkage (minimal variance)(Ward 1963)

cost(C∪S)-cost(C)-cost(S)

Cost function cost(S) is $\sum_{s\in S} Dissimilarity(s, Centroid_S)$. This is the variance if using square of Euclidean distance as dissimilarity metric and arithmetic mean as centroid. Ward's linkage is used to find the cluster with minimal merged cost along the hierarchy.

centroid linkage $\|Centroid_C - Centroid_S\|$

Centroid here means the center of a cluster, which has been introduced in 1.1.1. The mean vector (1.1.1) of observations in a cluster is the most popular choice of centroid.

median linkage $\|Median_C - Median_S\|$

Instead of using arithmetic mean of a cluster in centroid linkage, it uses median of observations in corresponding cluster.

Agglomeration:

With chosen dissimilarity, one n by n connectivity matrix, or dissimilarity matrix D is generated, n being the number of observations. Item D_{ij} is the dissimilarity value between a pair of ith and jth individual objects. The following algorithm explains how D is updated each step a cluster is merged.

Algorithm:

Compute dissimilarity matrix for the initial $n(n\text{-}1)/2$ pairs of observations

ith step along the hierarchy, i=1,2,…n-1,

1. While dimension of dissimilarity matrix D >1, search the pair of objects which have smallest value in D, and this pair is merged to cluster i
2. With ith cluster and other n-1-i objects, dissimilarity values between ith cluster and other n-1-i objects are computed, and dissimilarity Matrix D is updated with dimension n-i by n-i

Repeat step 1 and 2 until n-1th cluster is classified, i.e. all observations are clustered into one cluster

With agglomeration rule in mind, here we provide some tips on choosing dissimilarity measures. Monotonically increasing dissimilarity is not guaranteed in centroid method, and median linkage. Inversions may be observed. Therefore we do not recommend using linkage (centroid linkage, median linkage) that brings on inversions, as it spoils the hierarchical structure. We recommend average linkage, complete linkage and Ward's linkage. Ward's linkage is used to find the cluster with minimal merged cost along the hierarchy.

Dendrogram:

Dendrogram provides a visual rendition of HC as a binary tree. Each leaf corresponds to an individual observation, and the tree structure is dictated by the agglomeration algorithm (1.1.2.1). Specifically, each internal node of the tree corresponds to a merger of two objects and is displayed at a value of the vertical coordinate equal to the dissimilarity of the two objects being merged. This value is known as height. By convention, all leaves are displayed at zero height and in the horizontal order that makes a planar display

of the tree possible. Figure 1 below shows a dendrogram grown from Golub's Leukemia dataset (Golub, Slonim et al. 1999).

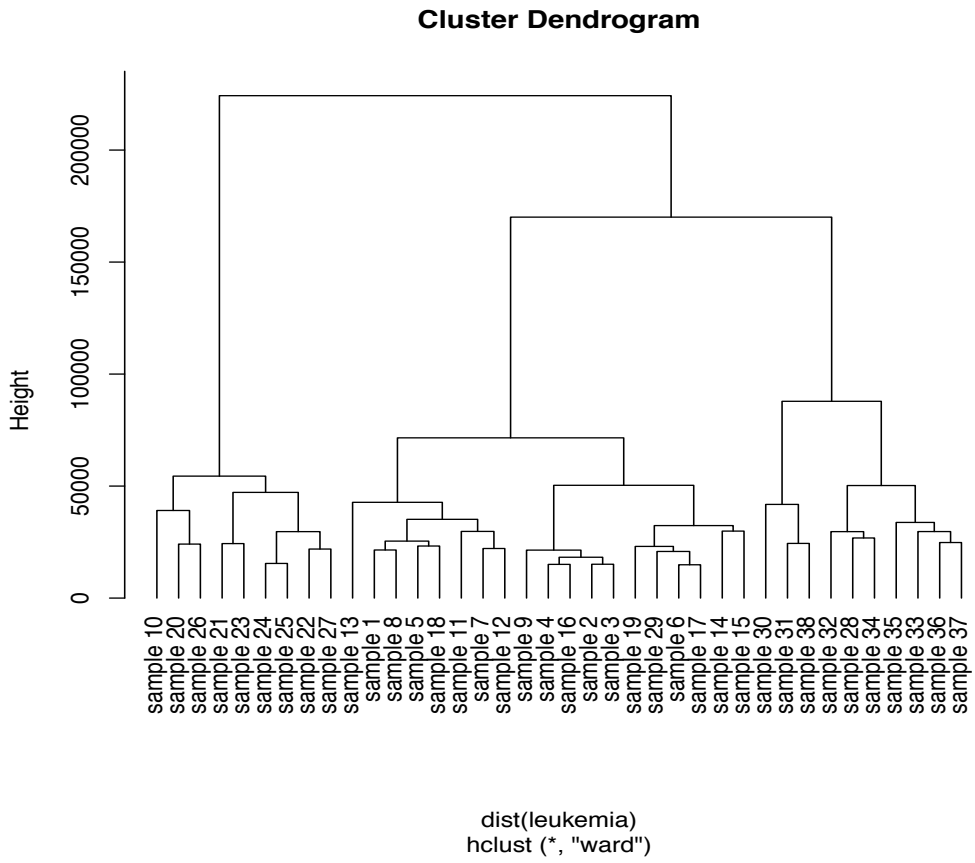


Figure 1 Dendrogram of HC, using 38 samples from Golub's Leukemia dataset.
This dendrogram is grown using Ward's linkage and Euclidean dissimilarity. The first cluster is merged by "sample 6" and "sample 17" with smallest non-zero height. *i*th cluster has monotonically increasing height.

1.2 Existing methods for finding distinct branches

So far we have introduced Clustering Analysis (1.1). Without prior knowledge of class labels, HC groups observations based on given dissimilarity and linkage, and produce dendrogram i.e. a visualization of hierarchy of clusters, such as Figure 2. HC is now widely used in partitioning data and finding distinct cluster of data. Given a hierarchical tree produced by HC, a method needs to be designed to serve two purposes:

1). Determine whether a branch is distinct from others

2). Check if candidate distinct branches form a partition of the data

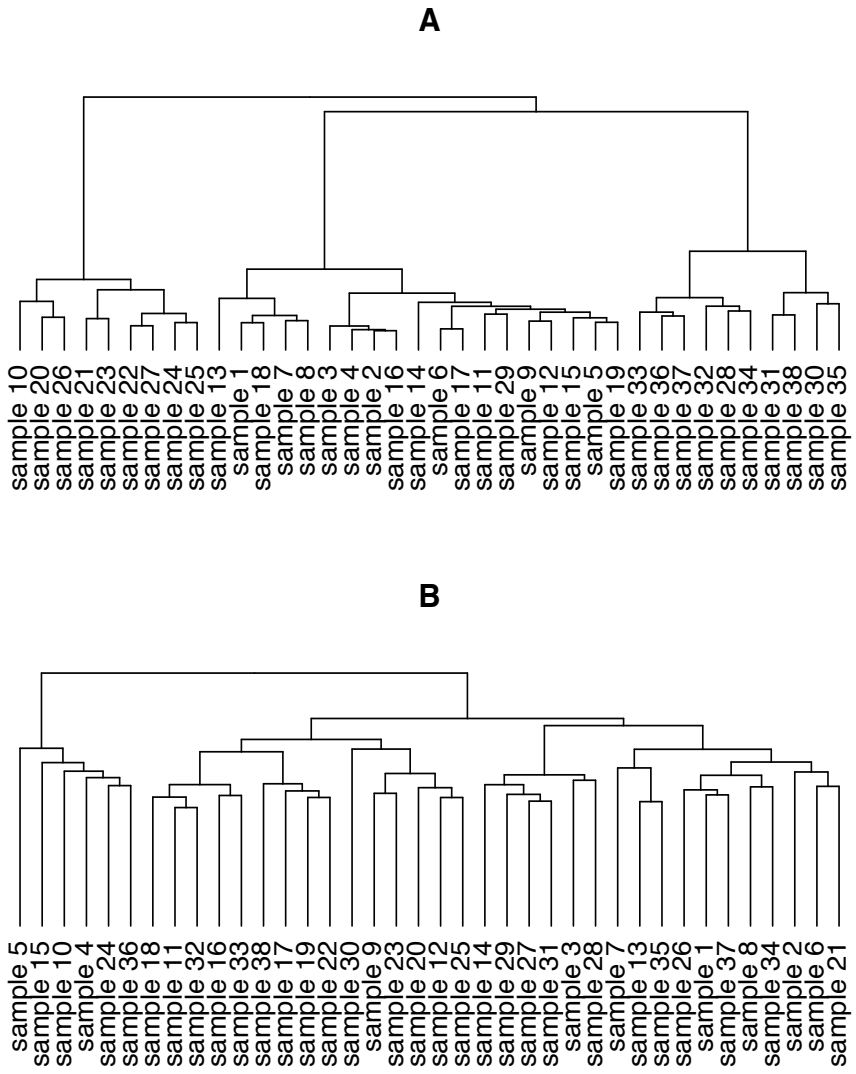


Figure 2 A comparison of two dendrograms.
Dendrograms are grown from HC with Leukemia data set. A). This dendrogram is based on real leukemia expression data. B). This dendrogram is based on randomized leukemia expression data. Both hierarchical trees are grown with Euclidean dissimilarity and Ward's linkage.

Identification of distinct branches of hierarchical trees by practitioners in biological sciences is predominantly qualitative and intuitive and rarely goes beyond visual inspection of the dendrogram, sometimes along with the image of the data matrix. Such identification is observer-dependent, and any two biologists may disagree on the result. For example, in Figure 2 A), the leftmost cluster containing nine observations is distinct from the rest of the tree, so is the middle cluster with four observations, "sample

3", "sample 4", "sample 2" and "sample 16".

With a second look in Figure 2 B), the leftmost cluster containing six observations also appears distinct from others. However, this is actually a branch from randomized data. This reveals the importance of examining if a distinct branch is really useful, or rather, not spurious. Statistical tests can be used to test if a branch is clustered by chance, with statistics defined on distinctness. Here we reformulate properties of a suitable method:

1). A measurement of distinctness of branches

2). A statistical test to find statistically meaningful branches

3). Ability to find a partition of data into distinct branches

Next we present existing methods. These methods fall into two categories, heuristic methods and statistically supported methods, following key property 3). Whether other two properties are satisfied is illustrated in each method. Generalization of properties of existing methods is given in 1.2.3.

1.2.1 Heuristic methods

Two existing methods fall into a heuristic category, tree-cut and dynamic tree cut. They neither have a quantitative measurement of distinctness of branches, nor perform a statistical test on truly usefulness of branches. However, they guarantee a partition of data when obtaining clusters with setting up certain parameters. The parameters, such as cutting height, number of clusters or minimal cluster size, that the results depend on, make the detection of distinct clusters less trustworthy.

1.2.1.1 Tree-cut

Tree-cut, as the name suggests, it takes advantage of the hierarchical structure in a dendrogram, cutting from certain height will leave a partition of clusters. For example, a cut-off at height 100000 results in a partition of three clusters in Figure 1. Instead of height, one can ask for a partition with certain number of clusters. However, the Tree-cut method alone has no evaluation of distinctness of the partition it produced. This makes choosing a height or a number of clusters without confidence. There are cases it performs even worse than intuitive observation. Being a classical approach to get a partition, we still acknowledge it to be a candidate of compared methods in Chapter 4. The function "cutree" in R is the tool we used in analysis. We call this method TC, in the following.

1.2.1.2 Dynamic Tree Cut

Dynamic Tree Cut (Langfelder, Zhang et al. 2008), or DTC in the following, is a more sophisticated recipe wherein the tree is generally partitioned into branches of unequal heights. The approach we are using is called "Dynamic Hybrid".

This is a bottom-up algorithm relying on dendrogram and dissimilarity matrix. It is called "hybrid" because it finds a partition of clusters with a combination of HC and PAM.

Algorithm of DTC, bottom up assignment:

Step 1. *Detection of clusters*

This step needs user-specified minimum cluster size N_0, together with the other three parameters h_{max}, g_{min} and d_{max}. Among the n-1 clusters in given dendrogram, a cluster is considered qualified if it satisfies all four criteria: 1). contains at least N_0 individual objects 2). joining height is at most h_{max} 3). gap, defined as the difference between joining height and mean of pairwise dissimilarity within cluster, is at least g_{min} 4). mean of pairwise dissimilarity within cluster is at most d_{max}.

Step 2. *Assignment of unlabeled objects to nearest clusters*

Here "unlabeled objects" refer to those objects within clusters that fail to pass criterion 2), 3) and 4). Unlabeled objects are assigned to clusters that are qualified in Step 1, based on PAM-like method. Note that objects in clusters which fail to meet criterion 1) will not be examined.

With PAM (1.1.1)-like procedure, *Dynamic Hybrid* improves detection of outlying members of clusters. The method provides built-in mechanisms to select optimal parameters, except user-supplied minimum cluster size. This is problematic because various settings of this cluster size may result in different partitions, which will be shown in the method validation session of Chapter 4. In addition, the results are sensitive to the choice of minimal cluster size.

1.2.2 Statistically supported methods

There are two existing methods that fall into this category, sigclust and sum of the branch lengths below. Methods have been developed, using statistical approaches to evaluate distinctness of clusters. These methods do not only satisfy property 1) and

property 2), but also satisfy property 3), since they examine branches from top down. However, top-down methods may not be able to detect nested statistically significant branches, as explained in the following.

1.2.2.1 SigClust

Statistical Significance of Clustering (SigClust), or SC in the following, is a parametric method designed to assess the significance of a binary partition of data. The method is valid for High Dimension Low Sample Size (HDLSS) data (Liu, Hayes et al. 2008). A measure of separation between the two parts is quantified as 2-means cluster index (CI).

$$CI_k \equiv \frac{\sum_{j\in k_1}\left\|X_j - \bar{X}^{k_1}\right\|^2 + \sum_{j\in k_2}\left\|X_j - \bar{X}^{k_2}\right\|^2}{\sum_{j\in k}\left\|X_j - \bar{X}\right\|^2}$$

Here X is a set of observations, labeled by index j. Each observation can be formulated as a data point from a Euclidean space of dimension p, i.e. number of variables. k is the set of indices of observations X, which is split into two disjoint subsets of indices k_1 and k_2. $\bar{X}^{k_1}$, $\bar{X}^{k_2}$ and $\bar{X}$ are the mean vectors (recall K-means in 1.1.1) of the two subsets and of the entire set of observations, respectively. CI, then, is the ratio of the sum of the two within-part variances to the variance of the entire set. The larger the CI is, the more evident a binary partition is within X. With CI as a test statistic, a null hypothesis is tested that X is sampled from a single multivariate normal distribution. Under the null hypothesis, a multivariate normal distribution is simulated, with each variable from a normal distribution of mean zero and standard deviation equaling to singular values of X itself. To test the null hypothesis, a number of random samples, each with same number of observations as X, are drawn from simulated normal distribution. CI is computed for each of these random samples, and forms an empirical distribution. P-value can be calculated from probability of obtaining test statistic CI_k at least as likely as observed in empirical distribution of CI under null hypothesis. The null hypothesis is rejected if p-value is less than some given significant level.

SC can be used in combination with many clustering methods, by testing 2-means assignment of one cluster at a time. In application to HC, SC is used in a top-down fashion. It starts with examining the split at the root node, and proceeds from a parent cluster to its children clusters, only if the two-way split at the parent cluster has been

found significant, i.e. when null hypothesis has been rejected. This top-down algorithm would not be able to detect significant clusters whose ancestor in the hierarchy is not significant. More importantly, SC compares clusters to samples from a single multivariate normal distribution and therefore is inherently parametric. Further, an underlying assumption of SC is that data items are points in a Euclidean space.

1.2.2.2 SLB

Unlike SC, Munneke et al (Munneke, Schlauch et al. 2005) proposed a measure of statistical distinctness of clusters, without making model assumptions about data distribution. This method is designed specifically for hierarchical structure produced by HC. The test statistic of a two-way split is defined as sum of the branch lengths below, or SLB in the following. Given a parent cluster with height D_0, and its child cluster 1 with height D_1 and child cluster 2 with height D_2, SLB computes the sum of difference of dissimilarity between child cluster and the last join.

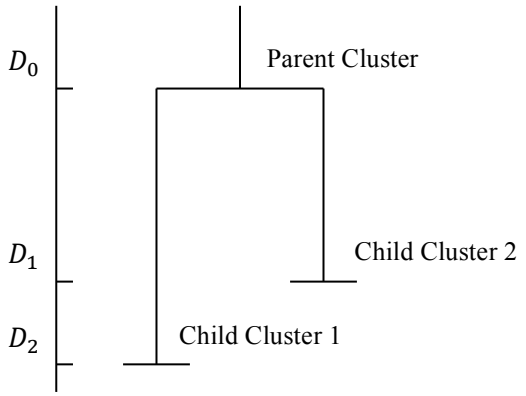


Figure 3 Illustration of definition on branch lengths
For each cluster, a left child cluster and a right child cluster exist. D_0, D_1 and D_2 are the corresponding dissimilarity value, i.e. height of cluster on dendrogram.

According to Figure 3, $SLB \equiv (D_0 - D_1) + (D_0 - D_2)$. This statistic utilizes the distance of parent to children clusters as its measurement of distinctness. SLB depends linearly on height of parent cluster, and those of two children clusters. Cluster with very large SLB, i.e. children far away, is more likely to contain distinct subgroup structures; a cluster with very small SLB, is not likely to be separated from children, and less likely to have a

hidden distinct subgroup. Randomized data sets are generated without parametric assumptions, and are used to obtain a null distribution of SLB. The null hypothesis assumes a random permutation represents an outcome that is as likely to have been observed as the original data. Empirical *p*-value can be calculated from probability of obtaining SLB from original branch at least as likely as observed in null distribution of SLB. Null hypothesis is rejected if SLB from original set is predominantly large, and statistically speaking, the obtained empirical *p*-value is smaller than some chosen significance level. Like SC, hypothesis testing is performed in a top-down fashion. Hypothesis testing starts from the root node, and proceeds from a parent cluster to its children clusters, only if the two-way split at the parent cluster has been found significant, i.e. when null hypothesis has been rejected. This examination stops when null hypothesis cannot be rejected.

Although SLB has a big advantage over SC because it gets rid of parametric assumptions, it is still implemented as a top-down algorithm, and the definition of statistic, the linear relationship with difference of node height, can be further improved.

1.2.3 Summary of existing methods

The key properties of all four published methods are summarized in Table 1. Although SC and SLB are more advanced than TC and DTC in employing statistical tests in determining branch distinctness, they suffer from limitations of top-down examination and SC is inherently parametric. This summary makes it clear that there is an unmet need for a statistically supported, non-parametric statistical method for assessing the distinctness of all internal branches in a tree. The performance of DTC, SC and SLB is further discussed in Chapter 4.

Table 1 Properties of four existing methods

Method	Order of examining the tree	Non-parametric	Significance estimated
TC	one-time horizontal cut	-	No
DTC	bottom up	-	No
SC	top down	No	Yes
SLB	top down	Yes	Yes

1.3 Measures of quality of partitions

To assess the performance of identified distinct braches, a partition composed of these candidate branches is compared to a reference, which is a pre-determined partition of class labels, also called a "truth" partition. Measures known as external cluster evaluation tools are used to quantify how close a computed partition is to the truth partition. These measures include Rand Index, Purity and Entropy, F-measure and V-measure.

1.3.1 Rand Index (RI)

Rand index, or RI (Rand 1971), in the following, quantifies the quality of partition by computing the proportion of correctly assigned pairs of objects. Given a computed partition and the truth partition, there are two types of correct decisions: a). A pair of individual objects found in one cluster also comes from same cluster in truth partition b). A pair of individual objects found in different clusters also comes from different clusters in truth partition. The pair of observations is denoted as γ_{ij}, while *i* and *j* are indices of corresponding observations, and there are *n* observations in total.

$RI = \sum_{i<j}^{n} \gamma_{ij} / \binom{n}{2}$, where $\gamma_{ij} = \begin{cases} 1 & correct\ decision \\ 0 & otherwise \end{cases}$

RI is the ratio of number of pairs which meet correct decisions. It is between 0 and 1. Large RI suggests clustering results not far from truth.

1.3.1.1 corrected-for-chance Rand Index (cRI)

When we evaluate performance of clustering, we would expect it performs at least better than random assignment. One problem with RI is it lacks a comparison with expected score from random partitions.

Hubert and Arabie (Hubert and Arabie 1985) proposed corrected-for-chance Rand Index, or cRI in the following, because of the way expected index is calculated. We generalize the problem as comparing two partitions of data, the truth partition $U = \{U_1, \dots, U_R\}$ with R classes and a computed partition $V = \{V_1, \dots, V_K\}$ with K clusters. An R by K contingency table can be built based on the agreement with assignment of objects.

Table 2 The contingency table formed by two partitions

U/V	V_1	V_2	⋯	V_K	Sums
U_1	n_{11}*	n_{12}	⋯	n_{1K}	a_1
U_2	n_{21}	n_{22}	⋯	n_{2K}	a_2
⋮	⋮	⋮	⋱	⋮	⋮
U_R	n_{R1}	n_{R2}	⋯	n_{RK}	a_R
Sums	b_1	b_2	⋯	b_K	n

*n_{ij} denotes the number of objects that are in both U_i and V_j. a_i and b_j denote marginal sums of n_{ij}.

According to the above contingency table, cRI is defined as:

$$cRI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}$$

, where the numerator is the difference between the number of common pairs and the expected index from model of random selection, under the assumption of generalized hypergeometric distribution. Unlike RI, cRI can be negative. Positive cRI suggests a better quality of clustering results over clustering by chance. cRI is also bounded above by 1, and the higher the better.

1.3.2 Purity and Entropy

Two commonly used external evaluation measures are Purity and Entropy(Zhao 2001). Suppose we use the same notation of U, V, R, K and a as in Table 2, n_{rk} is the number of observations in cluster k that belong to class r. Purity and Entropy of partition V are defined as:

$$Purity = \frac{1}{n}\sum_{k=1}^{K} \max_r(n_{rk})$$

$$Entropy = \sum_{k=1}^{K} \frac{a_k}{n}\left(-\frac{1}{\log R}\sum_{r=1}^{R}\frac{n_{rk}}{a_k}\log\frac{n_{rk}}{a_k}\right)$$

This method evaluates whether all elements in cluster k come from class r, but does not evaluates whether all members of class r are gathered into cluster k. An extreme example is $K>>R$, but each cluster is pure, i.e. with items from one single class.

1.3.3 F-measure

Another frequently used measure is F-measure(Rijsbergen 1979). It evaluates accuracy of clustering based on Precision (P) and Recall (R). Suppose we use the same notation of *U*, *V*, *R*, *K, a* and *b* as in Table 2, n_{rk} is the number of observations in cluster *k* that belong to class *r.* For a given class *r*, and a cluster *k*,

$$P(r,k) = \frac{n_{rk}}{|b_k|}$$

$$R(r,k) = \frac{n_{rk}}{|a_r|}$$

Traditional F-measure is defined as,

$$F(r,k) = \frac{2 * R(r,k) * P(r,k)}{R(r,k) + P(r,k)}$$

A more generalized form is a weighted form F_β, where β is a nonnegative real value,

$$F_\beta(r,k) = \frac{(1+\beta^2) * R(r,k) * P(r,k)}{R(r,k) + \beta^2 * P(r,k)}$$

The traditional F-measure according to this weighted formula is also called F-1 score. Above statistics are used to evaluate specified cluster. The overall F-measure for partition *V* is (Fung 2003),

$$\sum_r \frac{a_r}{n} \max_k \{F(r,k)\}$$

F-measure relies on the assignment of each cluster to a class.

1.3.4 V-measure

Rosenberg (Rosenberg 2007) introduced two quality criteria for a partition: homogeneity and completeness. To check homogeneity, examine class distribution within each cluster, determine how close it is to a single class. To check completeness, examine cluster assignment within each class, determine how close it is to a single cluster. Clustering with high homogeneity, tends to have low completeness, and may be far from truth partition.

According to Rosenberg (Rosenberg 2007), small "unmatched" clusters are not measured at all in the calculation of F-measure. He proposed V-measure to improve this drawback, and maintain a higher accuracy. Suppose we use the same notation of *U*, *V*, *R*,

K, a and *b* as in Table 2, n_{rk} is the number of observations in cluster *k* that belong to class *r*.

Homogeneity (h)

$$h = 1 - \frac{H(U|V)}{H(U)},$$

$H(U|V)$ is 0 with perfect homogeneity, members in each cluster come from a single class.

$$H(U|V) = -\sum_{k=1}^{K}\sum_{r=1}^{R}\frac{n_{rk}}{n}\log\frac{n_{rk}}{\sum_{r=1}^{R} n_{rk}}$$

$$H(U) = -\sum_{r=1}^{R}\frac{\sum_{k=1}^{K} n_{rk}}{n}\log\frac{\sum_{k=1}^{K} n_{rk}}{n}$$

Completeness (c)

$$c = 1 - \frac{H(V|U)}{H(V)},$$

$H(V|U)$ is 0 with perfect completeness, members of each class are gathered in one cluster.

$$H(V|U) = -\sum_{r=1}^{R}\sum_{k=1}^{K}\frac{n_{rk}}{n}\log\frac{n_{rk}}{\sum_{k=1}^{K} n_{rk}}$$

$$H(V) = -\sum_{k=1}^{K}\frac{\sum_{r=1}^{R} n_{rk}}{n}\log\frac{\sum_{r=1}^{R} n_{rk}}{n}$$

Similar to F_β, V-measure also has a generalized weighted form, β is a non-negative real value,

$$V_\beta = \frac{(1+\beta)*h*c}{(\beta*h)+c}.$$

With $\beta > 1$, completeness is weighted more; with $\beta < 1$, homogeneity is weighted more.

1.4 Summary of Chapter 1

In chapter 1, we provide a general introduction on

- Clustering analysis, agglomeration rule of hierarchical clustering
- Terms such as dissimilarity, linkage and dendrogram in agglomerative hierarchical clustering

- Existing methods of finding a partition with distinct clusters, DTC, SC and SLB
- Measures of quality of partitions, cRI and V-measure

Given this background, a nonparametric statistical method designed to assess distinctness of all internal branches within a hierarchical tree is in high demand. Performance of this method can be further validated, 1) with a comparison of existing methods, 2) and estimated by measures such as cRI and V-measure, using datasets with known truth partitions.

Chapter 2 Dataset Overview

There are five benchmark datasets used for validation in this dissertation. One is synthetic named Simulated6 (2.1), generated to simulate a set of gene expression profiles. The remaining four datasets, namely Leukemia, T10, Organelles and Chondrosarcoma (2.2-2.5) share two common features: they originate in biological experiments and in each case there is an independently known, biologically meaningful partition of observations into types. We call this known partition "truth", and the corresponding types the true types, henceforth. Data origin and assignment of true types in each dataset can be found in the corresponding sections.

This chapter also discusses how variables in each dataset are prepared from real-world raw data. Preprocessing methods including feature selection using our original method CORE (2.3.1-2.3.2) and normalization (2.4). We also include methods on handling missing data in 2.6.

2.1 Synthetic dataset Simulated6

This synthetic data set is named Simulated6. It has 60 observations, and 600 variables in simulation of gene expressions (Monti, Tamayo et al. 2003). The true partition of the data is into six subtypes (namely class 1, class 2, class 3, class 4, class 5 and class 6), with the sizes of 8, 12, 10, 15, 5, and 10. Each subtype is marked with 50 simulated unique up-regulated genes. Each of these first 300 genes has highest differential expression and lowest variation within its own subtype. The next 300 genes are simulated as background genes, sampled from same distribution across all observations.

Simulated expression levels of observations within each unique class are shown in Figure 4. Boundaries of unique up-regulated genes for each true type are marked by dashed vertical lines. Each true type has varying expression magnitudes. One observation belonging to class 1 (up left graph in Figure 4) shares up-regulated genes for both class 1 and class 2. Genes with indices 51 to 100 have higher expressions than the first 50 genes.

Identification of this observation and clusters detected on this data set are further discussed in 4.1.

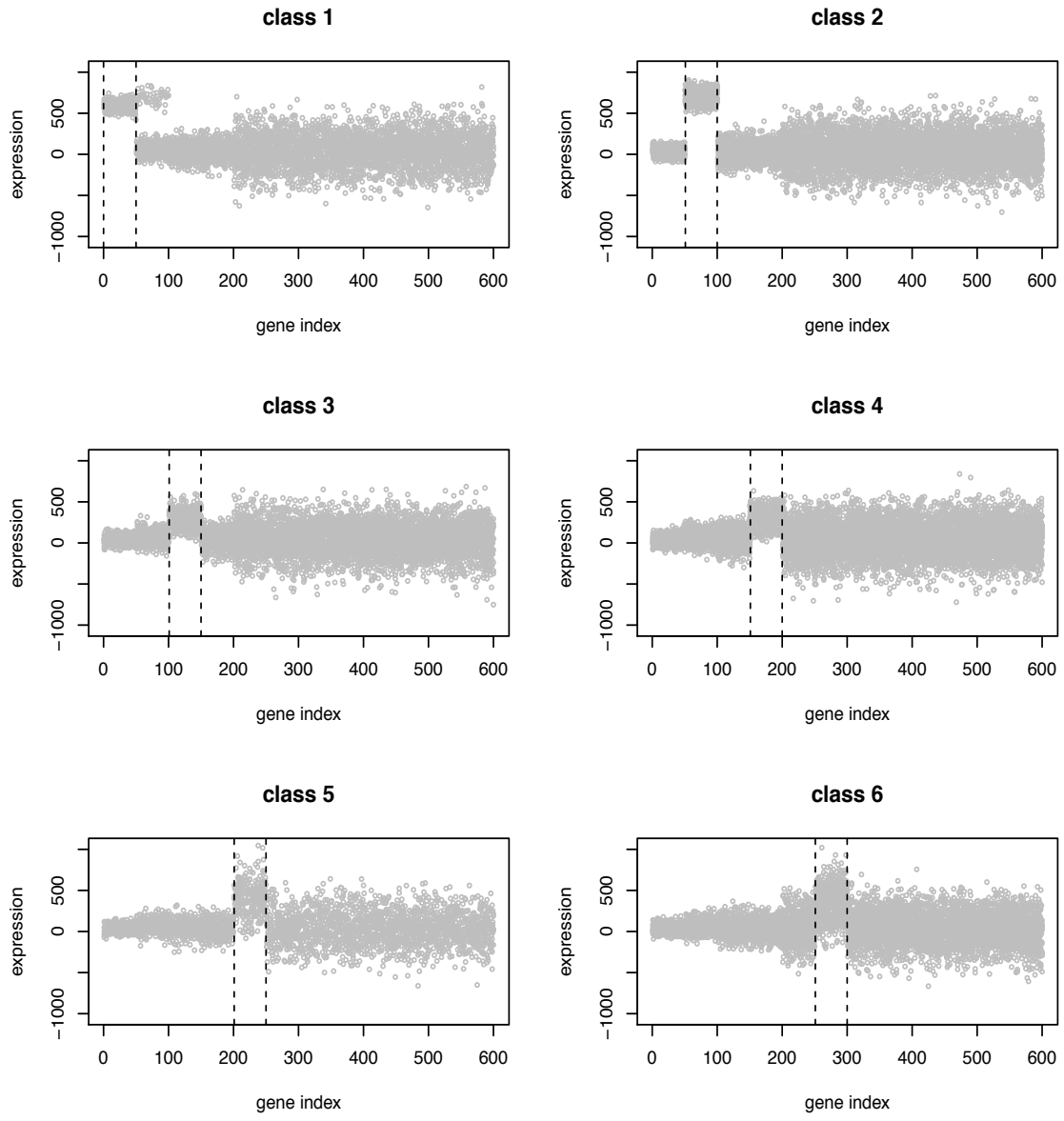


Figure 4 Scatter plot of simulated gene expressions in each ground truth subtype.
Each subtype has unique 50 most up-regulated genes. Last 300 genes are treated as noisy background genes.

Technically, distribution of each block of 50 variables over 60 observations is simulated from a normal distribution. While values of variables "up-regulated" for certain

class are represented by a significant positive shift. The last added 300 variables are coming from a normal distribution, consistent for all 60 observations. Properties of simulated gene expressions can be found in Table S1 of Appendix.

2.2 Leukemia dataset

This data set has been used to plot Figure 1 and Figure 2 in Chapter 1. It comes from a well-known cancer class discovery and prediction paper published by Golub (Golub, Slonim et al. 1999). There were 38 bone marrow samples obtained from acute leukemia patients at the time of diagnosis. The truth is a partition of patient cases into those of acute myeloid leukemia (AML, 11 cases) and of acute lymphoblastic leukemia (ALL), and a further partition of the ALL subset into the B-cell lineage (ALL_B, 19 cases) and the T-cell lineage (ALL_T, 8 cases) types. These are most important known distinctions in acute leukemia, in terms of both biology and clinical treatment. Microarray data was produced by Affymetrix, which contained probes of 6817 genes. The Leukemia dataset we are using is published by Monti (Monti, Tamayo et al. 2003). The 999 most up-regulated genes are chosen. Altogether this dataset has 38 observations/rows and 999 variables/columns, with each data entry as a numeric expression value.

2.3 T10 dataset and CORE

This data set tumor T10, contains 100 single cells that came from a primary breast tumor (Navin, Kendall et al. 2011). The true partition in this case is four-way, with the subsets differing from each other by ploidy as determined by cell sorting. Among 100 individual cells, specifically, there are 47 cases with Diploid and Pseudo-diploid (D+P), 24 cases with Hypo-diploid (H), 25 cases with Aneuploid A (AA) and 4 cases with Aneuploid B (AB).

Raw data has a large number of genomic interval events covering the entire genome. We have developed a feature selection method to select a small set of recurrent gain or loss fundamental genomic intervals. Next we introduce our original feature selection method, Cores Of Recurrent Events, also known as CORE (Krasnitz, Sun et al. 2013). Description of CORE in 2.3.1 and 2.3.2 follows our publication.

2.3.1 Methodology of CORE

DNA copy number analysis yields a set of copy number profiles, one per sample, describing the amplifications and deletions within the genome of the tumor of each

patient. CORE algorithm is the solution to finding regions of the genome that are significantly recurrent in large collections of copy number profiles.

Example of the problem

Copy number profiles presumably arise rather randomly throughout the genome of an unstable cell, but are selected for retention in the successful tumor clones at least in part by the presence of cancer genes, oncogenes in the amplified regions, and tumor suppressors in the deleted regions. The profiles can be further reduced to a set of intervals, regions of the genome where the amplifications or deletions took place. We refer to this data-reduction step as "slicing". Some of the intervals may contain the oncogenes and tumor suppressors that provided selective advantage, and some intervals are present by chance. Intervals of the first class will, in some sense, share recurrent elements, and intervals of the second class will not. Sets of genomic intervals that explain many of the observed intervals, for example because they contain cancer genes, are what we call cores. There are various types of explanation. A putative core might explain an interval if the interval contains the core. Alternatively, a core might explain an interval if they significantly overlap. Any number of quantitative relations between core and interval can be postulated to accommodate a variety of biological notions. In the end, one wishes to have a minimal set of cores that "best" explain the data, and that can be subject to some form of statistical testing for significance. We refer to this process as CORE.

Formulation of the general case

The input into CORE is a set of N intervals d_j, $j = 1, \ldots, N$ of a given type (for example, amplification or deletion events) derived from the observations. The domain Δ in which these observed intervals reside depends on the origin of the data. For data originating from genome-wide analysis, Δ consists of multiple disjoint intervals of the real line, each representing a chromosome. The objective of CORE is to find an optimal explanation of the intervals, the solution of a problem formulated as follows.

For an observed interval d_j and an explanatory interval s in Δ, we define an "explanation" of d_j by s as a function $E(d_j, s)$ with values in [0,1]. The specific functional form of $E(d_j, s)$ is dictated by biological considerations. For example, a useful form of $E(d_j, s)$ that reflects the degree of overlap of the two intervals is the Jaccard index:

$$E(d_j, s) = \frac{|d_j \cap s|}{|d_j \cup s|} \qquad (2.1)$$

In this case, s explains d_j completely if and only if the two coincide and not at all if the two are disjoint. However, a specific form for E is not required for a general formulation of the method. We also refer to $E(d_j, s)$ as an association measure. In the following, we use $P(d_j, s) \equiv 1 - E(d_j, s)$, the portion of d_j that s leaves unexplained.

Next, to generalize this concept to a set of explanatory intervals $S = \{s_1, s_2, \ldots, s_K\}$, we define the portion $P(d_j, s)$ of d_j left unexplained by S as:

$$P(d_j, S) = \prod_{k=1}^{K} P(d_j, s_k) \quad (2.2)$$

Finally, to generalize even further, we write the unexplained portion of the entire observed interval sets $D = \{d_1, d_2, \ldots, d_N\}$, the equation above can be further generalized by summation over all events,

$$P(D, S) = \sum_{j=1}^{N} P(d_j, S) = \sum_{j=1}^{N} \prod_{k=1}^{K} P(d_j, s_k) \quad (2.3)$$

For a fixed number of explanatory intervals K, we seek to minimize $P(D, S)$ over all possible sets S of K explaining intervals. Any such solution set of explaining intervals, $C_K = \{c_1, c_2, \ldots, c_K\}$ will be called "optimal" and the individual elements cores. Note that we have not so far specified the appropriate number K of cores to be sought. This question is addressed later when we consider the statistical assessment of cores.

Forms of explanation

The computational complexity of the minimization problem depends on the form of explanation. From now on, we consider important restricted cases of explanation in which $P(D, S)$ cannot attain a minimum unless each boundary of the cores s_k coincides with that of one of the observed intervals. With this proviso, minimization of $P(D, S)$ requires considering only a finite set of explaining intervals, namely those bound by $O(N^2)$ pair-wise combinations of the boundaries of the N events. Consequently, the quantities $P_{jk} = P(d_j, s_k)$ form a finite matrix of N rows and $O(N^2)$ columns, and the problem amounts to a choice of K columns such that $P(D, S)$ is minimized—that is, the minimizaton becomes a combinatorial problem.

To permit such minimization by a finite search, it is sufficient for $P(D, S)$ to be concave or linear as a function of either boundary position of s_k for all k, in any interval between adjacent event boundaries in D. In particular, this condition is satisfied for the following three special forms of association measures, $E(d, s)$: (first) $E(d, s) = 1$ if $s \subseteq d$

and otherwise $E(d, s) = 0$; (second) the Jaccard index $J(d, s)$ raised to a power $P \geq 1$; (third) $E(d, s) = f(|s|/|d|)$, where f is any strictly convex or linear function on the interval [0,1] with a range contained in [0,1] when $s \subseteq d$ and otherwise $E(d, s) = 0$.

These three forms of explanation capture different aspects of recurrence. The first form is especially simple and is designed to seek the genomic positions with the highest possible combined event count. However, this form of explanation ignores the degree of overlap among events explained by a given s and emphasizes regions where events overlap. The ability to detect clustering of broad events is thus reduced, especially when the broad events contain regions of narrow events that can be recurrent. On the other hand, the second and third explanation forms favor explanatory intervals at the intersection of multiple events with approximately coincident boundaries. Each core will therefore tend to be representative of a large number of similar genomic lesions.

Minimization of the unexplained portion

The minimization problem defined by the first form of explanation as defined above is an instance of the p-coverage location problem, exactly solvable by dynamic programming in $O(KN^2)$ time, making this form of explanation computationally advantageous. To our knowledge, however, no general algorithm with execution time polynomial in K has been found for the exact minimization problem as posed in Eq. 2.3, even if $P(\mathrm{D}, \mathrm{S})$ permits combinatorial minimization. In the absence of such a solution, we offer an iterative greedy procedure for finding cores that has a polynomial time complexity.

We initialize at $i = 0$ by setting $c_0 = \emptyset, P(d_j, c_0) = 1$ for all j. Then, at the i-th iteration, $c_i = argmin_s \sum_j P(d_j, c_{i-1}) P(d_j, s)$ is found, and C_i is formed by adding c_i to C_{i-1}. To continue the iteration efficiently, $P(d_j, C_i)$ is stored for each j, computed as in Eq. 2.1 above: $P(d_j, C_i) = P(d_j, C_{i-1}) P(d_j, c_i)$. The execution time of an individual iteration is independent of i, and the total execution time is proportional to K. Moreover, with any of the three explanatory forms, only a finite number of explanatory intervals need be searched at each iteration, and the greedy solution must search no more than $O(N^2)$ candidate explaining intervals. As the unexplained portion is a sum over N terms, the execution time is not greater than $O(KN^3)$. We will consider only greedy solutions for the remainder of this work.

Note that the Eqs. 2.2 and 2.3 can be generalized by the inclusion of weights for each event. In particular, the *i*-th minimization step of the greedy procedure may be interpreted as finding a single optimal core for the observed interval set *D*, but with each event d_j of *D* assigned a weight $W_{j,i-1} = P(d_j, C_{i-1})$, namely the portion of d_j left unexplained by previous cores. We view the set of intervals with their weights $P(d_j, C_{i-1})$ as the remaining unexplained data after the *i*-th iteration. This interpretation is used next in assessing the statistical significance of a new core.

Statistical criteria for depth

We tackle now a way to determine the depth of analysis, the lowest number of intervals that give a sufficient explanation of the data. Such a determination is made by seeking the lowest value for *K* such that the remaining unexplained data no longer display an unexpected amount of recurrence—that is, there is no new interval with a surprising amount of explanatory power. To determine this, we use a score, the amount of explanation gained from unexplained data by adding a new core, and compare this score to the scores obtained after the randomization of the unexplained data.

The total explanation provided by the core set C_K is $N - P(D, C_K)$. The gain in explanation from the *K*-th core is then $G_K = P(D, C_{K-1}) - P(D, C_K)$. For an exact solution of the problem, it is generally not true that C_K is obtained by adding one core to C_{K-1}. However, this is an intrinsic property for our greedy solution to the problem, so for the greedy case we can define the score of the optimal interval, c_K, as:

$$G_K = \sum_j W_{j,K-1} E(d_j, c_K) = \max_s \sum_j W_{j,K-1} E(d_j, s) \qquad (2.4)$$

We seek to evaluate the statistical significance of this score, judging thereby the significance of the core itself. Significance is determined by testing the null hypothesis that the *K*-th observed score is not improbably high in the set of weighted events with the event randomly placed in the genome.

More specifically, we sample from the null distribution of the score. After *m* iterations of CORE, we generate multiple independent trials. In each trial, each event d_j is transformed into an event d'_j by a random placement, while its weight $W_{j,m-1}$ is left unchanged. We then estimate the probability of a value G_m or larger would be drawn from the distribution of $G'_m = \max_s \sum_j W_{j,m-1} E(d'_j, s)$ generated from the multiple trials.

Typically we perform 1,000 trials. If $M + 1$ is the smallest m for which the null hypothesis cannot be rejected, the first M cores are retained.

Because events occur on chromosomes, and the events can themselves be large, on the order of the size of chromosomes, we must modify the above random translation scheme. The human chromosomes have broadly varying lengths, and a large event on chromosome 1, for example, cannot be translated to chromosome 21, restricting drastically our ability to randomize its placement. Therefore, when the observed interval data are randomly placed onto human chromosomes, we consider not the absolute length of an event but its length relative to the length of the chromosome on which it occurs.

2.3.2 Quantitative analysis using CORE

Breast Cancer Data

For analysis of individual tumor subpopulations, we use single-cell copy number data we previously described for human breast cancer tumors T10. The data consist of bin counts of sequence reads, segmented, and then converted to integer copy number segments. A total of 50,009 bins cover the entire genome, laid out in the usual order of chromosomes: 1, … , 22, X, Y.

Processing Breast Cancer Data

To use CORE, we must first extract interval events from segmented copy number profiles. The method of transforming each profile into a set of intervals differs for single-cell data and for mixed-cell population data. In both cases, we use a process we call slicing. We then find the significant cores, and create an incidence table.

To slice profiles from single cells, we determine the median ploidy for each cell, defined as the median of integer copy numbers for all bins. Segments above the median ploidy are considered amplified, and those below deleted. There is no restriction on the segment lengths, and these range from the shortest detectable by the segmentation algorithm to an entire chromosome. For each integer value of copy number except the median ploidy, we determine a unique set of largest intervals that can be placed without disruption into the profile. In essence, this procedure is a simplified version of the ziggurat deconstruction algorithm. Note that the information about the degree of copy number change caused by an amplification or a deletion event is lost in this

transformation. The input into CORE, separately for amplifications and deletions, is formed by pooling the intervals, with start and end positions specified as bin numbers.

Incidence Table for Profiles and Cores

In the case of DNA copy number analysis discussed in the following, the input set of intervals is formed by copy number events (gains or losses), each originating in one of multiple copy number profiles. Each profile represents a biological entity such as a tissue sample or a cell. Having derived K cores from this joint input set, we construct an incidence table T that quantifies how well each core performs in each profile. The incidence table is thus an $L \times K$ matrix, L being the number of profiles and K the number of cores. Each of its matrix elements, T_{lk}, is computed as the maximum over all of the intervals in profile l of the explanations by core k. In other words, T_{lk} is the explanation of the best fit of core k to profile l. It follows from this definition that all matrix elements of T are in the [0, 1] range.

Availability of Software

An implementation of CORE as an R package is available upon request and includes tools for computing core positions and scores and for assessing the statistical significance of scores, with a choice among measures of association as described here. In addition, R software is available for the analysis of integer copy number data, both upstream and downstream of CORE, including the slicing procedure and the derivation of the incidence table that we used to examine the subpopulation structure of breast tumors. R code for generating a simulated event configuration for arbitrary R, I, Λ, NR, σ, NB will also be provided upon request. More details in CORE analysis can be found in our corresponding publication (Krasnitz, Sun et al. 2013).

CORE analysis of this set yields 354 cores, 172 amplification cores and 182 deletion cores at $P = 0.05$ level of significance. T10 dataset uses incidence table computed from CORE analysis. As we mentioned above, each of its matrix elements is computed as the maximum over all of the intervals in a cell of the explanations by a core, with range [0, 1]. T10 data matrix has 100 rows and 354 columns. Rows of the data matrix correspond each to a cell, the columns correspond each to a core, specified by the sign of variation (amplification or deletion). An additional array has the endpoint chromosome positions of the region.

2.4 Organelles dataset and normalization

Organs and organelles represent core biological systems in mammals. In a global survey of organelle protein expression in mouse, protein content of four major organellar compartments was examined (Kislinger, Cox et al. 2006). A comprehensive proteomic profiling was done on assessing organellar enrichment. Data collected is protein expression of four subcellular compartments (true types), Nuclei, Cytosol, Microsomes and Mitochondria, independently on healthy laboratory adult mouse brain, heart, kidney, liver, lung and embryonic placenta. Altogether, there are 24 observations, each with 4768 proteins expressions. The expression levels used are cumulative spectral counts of proteins in each organelle.

Cluster 3.0 (Eisen 1998) was used to preprocess data and obtain a hierarchical tree structure in Kislinger's paper. Although the authors provide no detailed description of how the data were prepared, the following two-step normalization allows us to reproduce the original hierarchical tree.

1). Log-transformation of all data. This step comes before normalization. Log-transformation is widely used in processing DNA microarray data, because results of such experiments come out as fluorescent ratios. Preprocessing of expression levels in this case uses log base 2.

2). Normalization on proteins. This step changes magnitude of data within each protein, by multiplying a factor, so that sum of the squares of expression levels within each protein is one. All values are greater than 0 and less than one after this normalization.

Unlike CORE in 2.3.1, this two-step normalization does not reduce dimension of predictor variables. It makes sure that 4768 protein expression levels are on the same scale, which is a prerequisite of growing a reasonable hierarchical tree structure. With log transformation, any cumulative spectral counts with values zero become missing values, and pairwise deletion is used when computing dissimilarity of a pair of proteins.

2.5 Chondrosarcoma dataset

This data set comes from a study on using flow cytometry data to classify conventional central chondrosarcoma (Diaz-Romero, Romeo et al. 2010). It contains 34 cells, collecting from four types of chondrosarcoma. All cells were cultured in monolayer

under the same conditions and analyzed by flow cytometry for the level of expression, based on mean fluorescence intensity (MFI) ratios, of 11 surface markers. The truth is a four-way partition, with three parts corresponding each to a different tissue of origin and the fourth part formed by cells from tumor cell lines. Specifically, these cells are 10 human articular chondrocytes (HAC), 10 mesenchymal stem cells (MSC), 6 fibroblasts (FIB), and 8 tumor cell lines (TCL). Prior to clustering, observations/cells were examined, and two cells, one from HAC and one from MSC were removed using multivariate outlier detection(Marchette DJ 2003). In summary, Chondrosarcoma dataset has 32 cells/rows and 11 surface markers/columns.

2.6 Handling missing values

Observations with missing features are common. This section talks about how to deal with missing data before clustering. With existence of missing data, the first thing is to determine whether the occurrences are at random, and will not distort the observed data. Detailed exploration of missing data can be found in Little and Rubin's book (Little 2002). Denote X_{obs} as observed entries (without missing data) in predictor matrix X. R is formed as an indicator matrix with ij-th value as 1 when X_{ij} is missing and 0 otherwise.

Data is missing at random (MAR) if distribution of R depends on Z only through Z_{obs}: $P(R|X,\theta) = P(R|X_{obs},\theta)$. Here θ are any parameters in the distribution of R. This is equally meaning missing data indicators can be explained by observations with full information. "At random" here suggests occurrence of missing data is conditional on observed data. It is not strictly the definition of random event. Also equality of probability is difficult to be validated with an unsupervised predictor matrix.

Data is missing completely at random (MCAR) if distribution of R doesn't depend on missing or observed data: $P(R|X,\theta) = P(R|\theta)$. This is a stronger assumption. Most methods dealing with missing data rely on assumption of MCAR.

Approach to cleaning missing data: 1). Discard observations with any missing values. 2). Rely on learning algorithm if its input has tolerance on missing data. 3). Impute all missing data.

The first approach (list-wise deletion) is okay if sample size is large and proportion of missing data is small. The second approach depends on the algorithm. For example, implementation of some dissimilarity functions allow missing values, certain

dimension containing missing values are omitted (pair-wise deletion) when computing pairwise dissimilarity. HC can tolerate missing values using pair-wise deletion, meaning if there are p features in a pair of observations, and altogether q dimensions out of p have missing values, dissimilarity can be calculated on p - q dimensions with chosen dissimilarity.

The third approach, imputation is popular, not limited by assumption or algorithm. Most common imputation is substituting missing values with mean or median with nonzero values for that feature. Of course there are more advanced and sophisticated imputation methods, such as Soft-Impute (Mazumder 2010).

2.7 Summary of Chapter 2

This chapter provides overview on five datasets, with their biological origins, their observations, variables and true subtypes. The essential properties of these datasets are summarized in Table 3. These datasets are further used in Chapter 4 as benchmark cases. Partition of true types in each data set is deemed as ground truth, and later employed as reference of optimal partition in validation of performance.

Table 3 Properties of five benchmark datasets

Dataset	Origin	Number of leaves	Number of variables	True number of classes
Simulated6	Simulation of gene expression	60	600	6
Leukemia	mRNA levels from microarray analysis	38	999	3
T10	DNA copy number analysis, sequencing	100	354	4
Organelles	Proteomic analysis, using mass spectrometry	24	4768	4
Chondrosarcoma	Flow cytometry analysis of surface markers from fluorescence intensity	32	11	4

All five datasets are public and are available with their corresponding publications. All except Organelles can be directly used for clustering. Organelles dataset needs to be normalized using the two-step procedure mentioned in its corresponding part 2.4.

This chapter also introduces our published method CORE as feature selection

approach for dataset T10. Summary of CORE follows our publication. CORE is a general approach to inference from interval data. Given a collection of observed events and a geometric association measure between events and explanatory intervals, CORE finds a given number of explanatory cores that maximizes the explanation. When the association measure is drawn from three broad varieties outlined in the text, for example the Jaccard index, we find a greedy solution with algorithmic complexity $O(KN^3)$, where N is the number of events and K is the number of cores. We believe our formulation of the problem is "natural" in the sense that it captures the manner in which a human observer seeks to find fundamental intervals behind a set of recurrent events in the presence of noisy events and boundaries.

Chapter 3 Methodology

Nowadays, hierarchical clustering is more widely used as a method of partitioning data and of identifying meaningful data subsets. Quantifying the interpretation of hierarchical trees and introducing mathematically and statistically well-defined criteria for distinctness of sub-trees would therefore be highly beneficial and is the focus of this thesis. This chapter presents methodology of Tree Branches Evaluated Statistically for Tightness, or TBEST in the following (Sun and Krasnitz 2014). In 3.1, we use a simple example to motivate the proposed measure of distinctness of branches/clusters. In 3.2, a sampling procedure is discussed for examining randomizations and null distributions. In 3.3, statistical tests are employed to examine tightness of branches, and *p*-value estimation can be calculated using two approaches, empirical and EVT-based.

Consider a set of objects with pair-wise relations given by a dissimilarity matrix. With a linkage rule, a hierarchical tree can be grown for the set. We will only consider inversion-free linkage rules here. The tree is specified, in addition to its branching structure, by the heights of its nodes. The height of the node quantifies the dissimilarity within the data subset defined by the node. We wish to construct, for each node of the tree, a measure of how distinct the data subset corresponding to the node is from the remainder of the data set. Next a one-dimensional example is given to explain how statistic $S(n)$ measures tightness of branches.

3.1 A simple example

The special case of the objects being points in a Euclidean space, with the dissimilarities defined as distances between the points, may be used for guidance in this construction. The node height then quantifies the linear extent of the data subset defined by the node. Accordingly, it has been proposed (Munneke, Schlauch et al. 2005) to make the measure of distinctness of a node n linear in the difference in heights between a parent $P(n)$ of n and that of n itself.

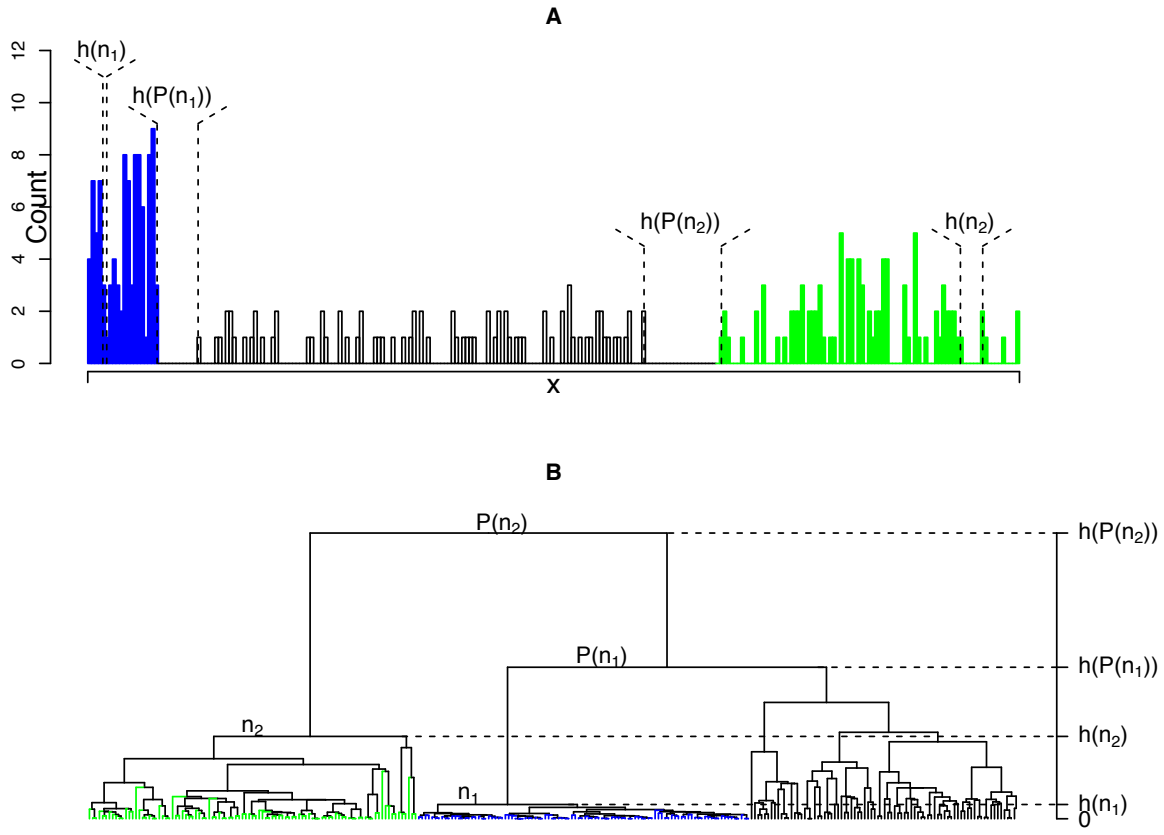


Figure 5 Illustration of the definition of tightness.
The data consist of 280 points in one dimension, drawn from a normal mixture with the components $N(0.5,0.4^2)$ (blue), $N(11,1^2)$ (green) and $N(5,2^2)$; (black). **A)** A histogram of the input data. **B)** A hierarchical tree of the input data, grown using the absolute difference of the data values as the dissimilarity measure, and single linkage. Thus, the node heights shown in **(B)** are equal to the corresponding gaps in the data, as indicated in **(A)**. Nodes n1 and n2 are approximately equally tight.

An example of a one-dimensional dataset, tabulated in Supplemental File and shown in Figure 5, illustrates a difficulty with such construction. Both the subsets shown in blue and in green are clearly distinct from the rest of the data, but the difference in heights between the blue node and its parent is not as great as that between the green node and its parent. Thus, based on the parent to child difference in heights, one would conclude, counter-intuitively, that the blue subset is not nearly as distinct as the green subset. A measure in better agreement with intuition is the relative difference of heights:

$$S(n) \equiv \frac{h(P(n)) - h(n)}{h(P(n))} \quad (3.1)$$

, where $h(n)$ is the height of node n. In the following we refer to $S(n)$ as the tightness of node n. In the absence of inversions, the tightness of any node is a number between 0 and 1. In particular, $S(n) = 1$ identically if n is a leaf.

The two subsets highlighted in Figure 5 are nearly equally tight by this measure, despite the disparity in their heights.

3.2 Randomizations and null distribution

To enable statistical analysis of tightness, a null distribution of $S(n)$ is required, for making comparisons with the observed $S(n)$. This null distribution is obtained by randomizing the dataset from which trees are grown.

How such randomization is to be performed depends on the type of the data and on the broader context of the study and cannot be specified in general. For example, if the data matrix represents gene expression, with genes as rows and observations as columns, it may be appropriate to randomize the data by permuting values independently within each row. However, in other situations a more restrictive randomization should be adopted. For example, the elements of a binary data matrix may represent the mutation status at a set of genomic positions (rows) in a collection of genomes (columns). The investigator may wish to randomize the data while preserving both the site mutation frequencies (row sums) and the overall mutation burden within each genome (column sums).

Randomization methods, or data permutation methods on synthetic data and four real-world benchmark cases are listed in Table 4 below. Simulated6 is the synthetic data set. It was introduced in 2.2 and is used, together with four real-world cases, to validate performance of TBEST. Note that in Table 4 randomization of T10 data is different from

other three. Besides cores detected using the CORE algorithm (2.1.2), this data set also includes chromosome information of each core. There are multiple instances of strong geometric overlap between cores. As a result, the corresponding columns in the data matrix exhibit strong pairwise correlations, positive for cores of equal sign (both gains or both losses), and negative for cores of opposite signs. Consistent with these geometric constraints, the null distribution in this case is generated as follows: the data matrix is divided into sub-matrices by the chromosome number (1,2,...,22,X), and rows are permuted independently within each sub-matrix.

Table 4 Data permutation methods for benchmark cases

Dataset	Data permutation Method
Simulated6	Independently for each coordinate (column)
Leukemia	Independently for each gene (column)
T10	Independently for each chromosome; identically for all cores (columns) in a chromosome
Organelles	Independently for each protein (column)
Chondrosarcoma	Independently for each surface marker (column)

3.2.1 Distribution of tightness

As Figure 6 and Appendix Figure S1 illustrate, the shapes of these distributions generally depend on the number of leaves and, in most cases examined, the peak of the distribution occurs at higher tightness for smaller number of leaves. The identity $S(n) = 1$ for single-leaf nodes is consistent with this observation. We therefore conclude that, for a given observed value of tightness, the appropriate null distribution should be sampled by repeated randomization of the data, growing a tree for each randomization, selecting among its nodes the ones with the numbers of leaves matching the observation, and determining the tightness of these nodes.

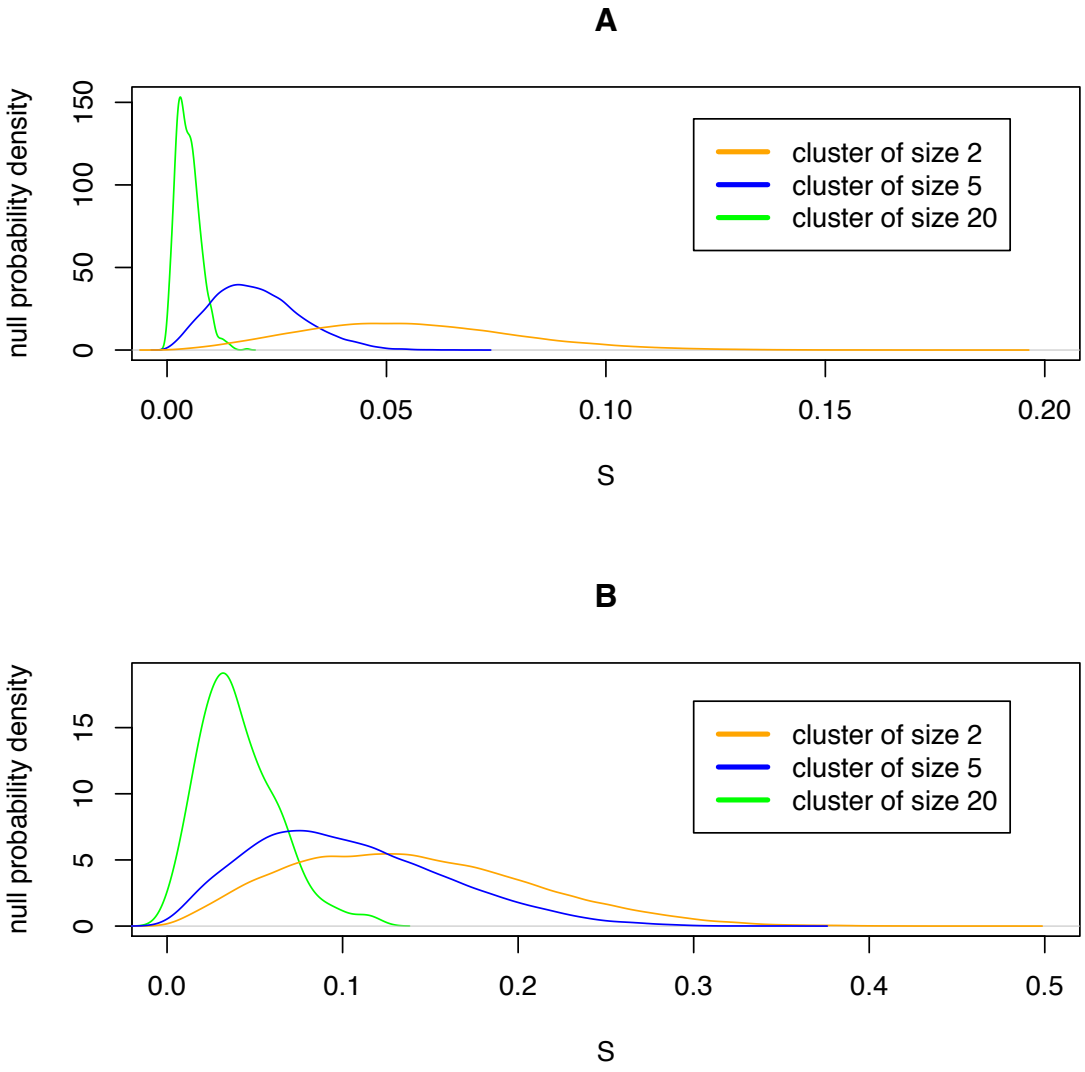


Figure 6 Null distribution of tightness.
The null distribution of node tightness *S* depends on the number of leaves. The empirical probability density distributions for the Simulated6 set with (1 - Pearson correlation) dissimilarity – average linkage combination (**A**) and for the Organelles set with (1 - Pearson correlation) dissimilarity – Ward linkage combination (**B**) are shown, for three different values of the number of leaves in each case. Each plot is based on 5000 randomizations of the respective data set.

So far, with a hierarchical tree structure, $S(n)$ can be calculated for all the internal nodes, excluding the root, since the latter has no parent. With a number of trees grown from randomized data, null distribution of $S(n)$ is obtained.

However, it is not guaranteed that, in any tree grown from randomized data, there will be a unique node with a number of leaves exactly equal to that of the observed node. To resolve this difficulty conservatively, we adopt the following procedure. If, for a given data randomization, the tree contains nodes with the number of leaves exactly as observed, the highest S(n) computed for these nodes is added to the sample. Otherwise we consider all the nodes with the number of leaves nearest the observed one from above and all those with the number of leaves nearest the observed one from below, and add to the sample the highest S(n) of any of these nodes. Note that, since $S(n) = 1$ for all single leaves, the latter can never be found significantly tight, and the analysis as described is only valid for internal nodes.

3.3 Compute statistical significance

With the sampling procedure specified, tests for statistical significance of tightness can be conducted for all the internal nodes of the observed tree excluding the root. The number of tests is therefore two less than the number of leaves. Due to this multiplicity of tests, higher levels of significance are required for rejection of the null hypotheses for trees with larger numbers of leaves. A straightforward way to handle this requirement would be to increase the size of the sample from the null distribution by performing more randomizations. In this thesis, largest number of leaves among five benchmark cases is 100, and validation in Chapter 4 is done with conservative empirical approach. However, for trees with large numbers of leaves this simple-minded approach may be rendered impractical by computational cost. Therefore, we offer an alternative approach to improve time efficiency.

3.3.1 Extreme value theory based estimation

Instead of a purely empirical approach, higher levels of significance may be assessed by using extreme-value theory (EVT) to approximate the tail of the null distribution, thereby permitting considerable economy of computational effort (Knijnenburg, Wessels et al. 2009). To estimate the p-value, a test statistic of one branch $S(n)$ and a number of null statistics, represented as set S_0, from randomized data are

needed. Unlike the empirical approach, number of randomizations does not rely on total number of observations, and can therefore be much less than that of empirical approach. The set of extreme (very large or very small) null statistics that forms the tail of null distribution can be modeled with Generalized Pareto Distribution (Pickands 1975). Algorithm of estimating *p*-values simply builds on the condition that $S(n)$ is in the tail of distribution or not. The number of null statistics inside set S_0 that are greater than or equal to $S(n)$ is M. M_0 is a threshold suggesting number of null statistics in the tail.

Algorithm:

if $M \geq M_0$,

estimate *p*-value using empirical cumulative distribution function

else,

estimate *p*-value using Generalized Pareto Distribution

Parameters estimations in Generalized Pareto Distribution are obtained by maximum likelihood or methods of moments. Goodness of fit is tested, and *p*-value is calculated from approximated tail distribution.

We have used this EVT-based method alongside the more costly purely empirical computation of significance in our benchmark studies reported in Chapter 4, and found the two approaches to be in good agreement, as shown in Appendix: Figure S2.

3.3.2 Correction for multiple hypotheses testing

Computing the probability of test statistic of one branch among null distribution of statistic gives one *p*-value. Test statistic of this branch is not likely to appear in null distribution of statistic when *p*-value is less then given significance level, and this branch is statistically significant. Computing the probability of test statistic of all internal branches simultaneously among null distribution of statistic falls into the field of multiple hypotheses. Family wise error rate (FWER) is the probability of making at least one Type I error when performing multiple hypothesis tests. Here we perform FWER correction equivalently for each empirical *p*-value. The *p*-values displayed in the following were computed by applying a multiple-hypotheses correction of the form $p = 1 - (1 - p_e)^{N-2}$, where p_e is the empirical *p*-value and N is the number of leaves. Note that the number of hypotheses tested is $N - 2$, i.e. the number of internal nodes.

The false discovery rate (FDR) is designed to control false positives among the set of rejected hypothesis. *q*-value of an individual hypothesis test is the minimum FDR at which the test may be called significant. One approach is to directly estimate *q*-values rather than fixing a level at which to control the FDR. FDR correction of empirical *p*-values is implemented in R package.

We use TBEST in the following way to identify most detailed significant partitions of the data into branches of a given hierarchical tree. We define a partition to be significant with a threshold α if (a) every part is a branch and (b) if for every part at least one of the children of its parent node is tight with the *p*-value $p < \alpha$. Among the significant partitions with a threshold α we find the most detailed, i.e., the one with the highest number of parts. In case of a nested distinct branch with its parent node and counterpart of its parent node being non-significant, TBEST can identify this branch as a statistically tight branch. But there may not exist a most detailed significant partition judged by criteria a) and b).

3.4 Comparison of TBEST and existing methods

Our method TBEST shares features with the existing approaches. Recall 1.2, here we compare TBEST with DTC, SC and SLB.

Like SC and SLB, TBEST employs statistical analysis to identify significantly distinct branches of a hierarchical tree. Similarly to DTC and SLB, it uses tree node heights to assess the distinctness of a tree branch. At the same time, TBEST differs from the existing designs in several aspects, two of which are critical.

- First, unlike DTC, SC and SLB, it examines all the tree nodes simultaneously for distinctness.
- Secondly, unlike SLB, it combines node heights non-linearly to construct a statistic for distinctness that is better able to handle a tree in which distinct branches of approximately equal numbers of leaves occur at different heights.

The key properties of all four methods are summarized in Table 5. The third column suggests whether the algorithm needs extra criterion to detect tight branches, given the dataset and its hierarchical tree structure. Recall from 1.2 that results of DTC depend on minimal node size and SC relies on assumption of normal distribution.

Table 5 Properties of TBEST and three existing methods

Method	Order of examining the tree	Non-parametric	Significance estimated
TBEST	all internal nodes in parallel	Yes	Yes
DTC	bottom up	No	No
SC	top down	No	Yes
SLB	top down	Yes	Yes

3.5 Summary of Chapter 3

This chapter presents methodology of TBEST. In this chapter, we have reviewed,

- Statistic of tightness in Eq. 3.1, which computes relative difference of height. A simple example shows this definition is in better agreement with intuition.
- Appropriate null distribution should be sampled by repeated randomization of the data, growing a tree for each randomization, selecting among its nodes the ones with the numbers of leaves matching the observation, and determining the tightness of these nodes.
- Two approaches to estimate statistical significance: empirical approach and EVT-based approach. Empirical *p*-values are corrected by multiple hypothesis correction.

Chapter 4 Validation

With the methodology of TBEST presented in Chapter 3, this chapter discusses the performance of TBEST, with a comparison to the three existing methods, DTC, SC and SLB (1.2). Detection of distinct branches in five benchmark datasets is examined and validated in 4.1-4.5.

To better judge the performance of TBEST in comparison to the other three algorithms, we considered, for each dataset, more than one combination of dissimilarity and linkage methods used for hierarchical clustering. These combinations, together with permutation methods are shown in Table 6. With the exception of the third benchmark case, randomization of the input data, as required for both TBEST and SLB, consisted of randomly permuting the observed values, independently for each variable. Reasoning for this is mentioned in 3.2.

Table 6 Combinations of datasets, dissimilarity, linkage and randomization methods, used for testing TBEST

Dataset	Dissimilarity	Linkage	Randomization Method
Simulated6	Euclidean	complete	Independently for each coordinate (column)
	(1 - Pearson correlation)	average	
Leukemia	Euclidean	Ward	Independently for each gene (column)
	(1 - Pearson correlation)	average	
T10	Euclidean	Ward	Independently for each chromosome; identically for all cores (columns) in a chromosome
	(1 - Pearson correlation)	average	
Organelles	(1 - Pearson correlation)	Ward	Independently for each protein (column)
	(1 - Pearson correlation)	average	
Chondrosarcoma	(1 - Spearman correlation)	Ward	Independently for each surface marker (column)
	(1 - Kendall correlation)	average	
	Manhattan	Ward	

The degree of agreement between a computed partition of the data and the truth is quantified in terms of the corrected-for-chance Rand index, or cRI (1.3.1.1). It should be noted that the subsets of the data identified as distinct by TBEST and the other three techniques by necessity correspond each to a branch of a tree. This, however, is not necessarily the case for the true types, some of which do not correspond to a single branch. As a result, a perfect match between any computed partition and the truth may not be possible, and the maximal attainable value of cRI may be below 1. For this reason, to evaluate the performance of TBEST and the published methods across benchmark datasets, we also identify, for each tree considered, a partition into branches that best matches the truth and determine cRI between that partition and the computed partitions for each of the methods.

In each of the cases in the following we studied how the most detailed significant partition found by TBEST, and its correspondence to the truth, vary with the significance threshold α. In an analogous fashion, we analyzed the detailed partitions generated by SLB and SC. For DTC, which is not a statistically supported method, we examined the properties of the most detailed partition as a function of the minimal allowed number of leaves in each part.

4.1 Simulated6

The data are a sample of size 60 in 600 dimensions (Monti, Tamayo et al. 2003). The true partition of the data is into six subtypes. Each of the 600 variables represents a simulation of a gene expression. Properties of this dataset are summarized in more detail in Table S1 of Appendix.

The comparison between the four algorithms is displayed graphically in Figure 7. For both combinations of dissimilarity and linkage only TBEST and DTC match the truth exactly, while the other two methods either fail to partition the set or do so incompletely. We note that the Euclidean dissimilarity – complete linkage combination results in a particularly challenging tree (A in Figure 7), which cannot be partitioned correctly by a static cut.

The most left color bars in Figure 7 represent assignment of leaves. With class 1 in blue, and class 2 in red, the exceptional observation is obvious in the top color bar "Truth" of A and D in Figure 7. This observation is combined with observations from

class 2, using HC of both dissimilarity linkage combinations. TBEST has the broadest range of significance levels that matches ground-truth, with any $\alpha \geq 0.016$ in the first case (A, B, C in Figure 7), and any $\alpha \geq 0.001$ in the second case (D, E, F in Figure7). While in the first case, if significance level is set less than 0.016, a partition of five branches are found, corresponding to red, blue, green, yellow and the rest branches in A, and results in cRI around 0.78 in B. Given more extreme significance level, none of orange branch, purple branch and their parent branch are considered statistically tight.

Table 7 records scores of clustering quality compared to the partition best matches true subtypes. cRI and V measure are both calculated for partitions found at a given level of significance (0.001 and 0.05). For both dissimilarity linkage combinations, TBEST outperforms SC and SLB, with the highest clustering scores of finding a partition equal or closest to the optimal partition.

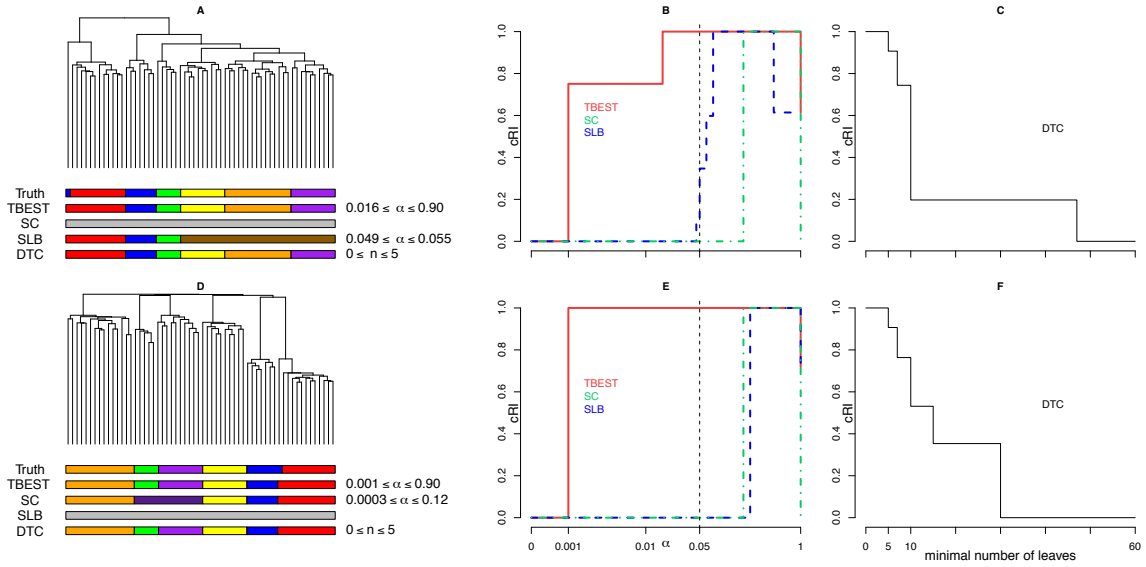


Figure 7 TBEST compared to published methods for Simulated6
Performance comparison of TBEST and the three published methods in Simulated6 dataset for the Euclidean dissimilarity – complete linkage combination (top) and for the (1 - Pearson correlation) dissimilarity – average linkage (bottom). For each combination the left portion (**A** or **D**) shows the corresponding dendrogram, under which then true partition and the partition best matching the truth for each of the methods are shown as color bars. In the middle portion (**B** or **E**), the relative cRI of the computed partition is plotted against the required level of significance α for each of the significance-based methods. The customary α = 0.05 threshold of significance is shown by a dashed vertical. In the right portion (**C** or **F**), the relative cRI of the computed partition is plotted against the minimal allowed number of leaves for DTC.

Table 7 Quality of partition in Simulated6*

Dissimilarity and linkage	α	TBEST	SC	SLB
Euclidean dissimilarity complete linkage	0.001	0.75(0.91)	0(-)	0(-)
	0.05	1(1)	0(-)	0.35(0.66)
(1 - Pearson correlation) dissimilarity average linkage	0.001	1(1)	0(-)	0(-)
	0.05	1(1)	0(-)	0(-)

* Values of α in second column are significance levels. Quality of partitions is recorded as cRI (V measure), from column three to column five.

4.2 Leukemia

The original Leukemia dataset (Golub, Slonim et al. 1999) contained mRNA level values for 6817 genes; this number was reduced to 999 by feature selection (Monti, Tamayo et al. 2003). Performance of TBEST is compared with that of the other three methods in Figure 8. For the Ward linkage, two of the significance-based methods, SC and TBEST, attain the highest possible value of the cRI. However, SC only does so with low significance ($\alpha > 0.33$), while TBEST achieves it best performance with high significance ($\alpha \approx 2\times10^{-3}$) and maintains performance close to optimal in a wide range of *p*-values. The performance of SLB in this case is similar to that of TBEST, but SLB does not attain the optimum. With the average linkage, TBEST outperforms both SC and SLB throughout the entire range of thresholds considered and attains optimal performance at high significance.

In both cases the performance of DTC is highly sensitive to the minimal allowed size of a branch, especially so for the Ward linkage, where this algorithm attains top performance for sizes between 6 and 10, but performs substantially below the optimum outside this range.

To show quality of partitions found by significance-based methods, cRI and V measure are both used at given significance level (0.001 and 0.05) in Table 8 below. From this table, TBEST outperforms SC and SLB, with the highest clustering scores of finding a partition equal or closest to the optimal partition. In both Ward and average linkages, with a lower significance (0.05), the close-to-optimal partition is found with a further split of AML subtype (branch with green bar in dendrograms, Figure 8). This suggests interesting substructures. Given a lower significance level, a statistically tight branch with observations labeled “AML_14”, “AML_16”, “AML_3” and “AML_7” is detected with both dissimilarity linkage combinations.

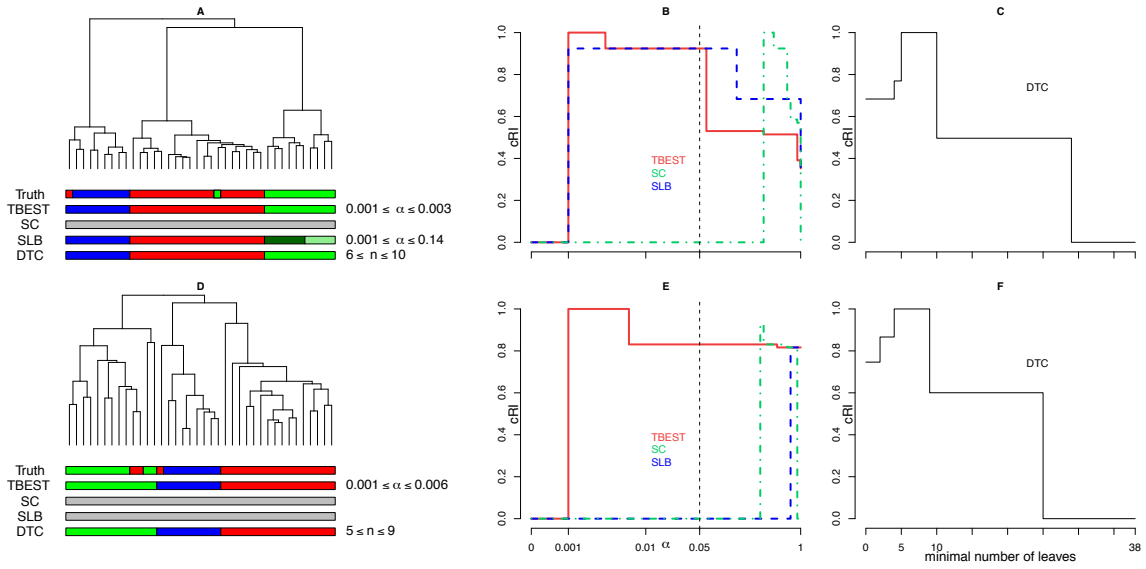


Figure 8 TBEST compared to published methods for Leukemia.
Performance comparison of TBEST and the three published methods in Leukemia dataset for the Euclidean dissimilarity – Ward linkage combination (top) and for the (1 - Pearson correlation) dissimilarity – average linkage (bottom). For each combination the left portion (**A** or **D**) shows the corresponding dendrogram, under which then true partition and the partition best matching the truth for each of the methods are shown as color bars. In the middle portion (**B** or **E**), the relative cRI of the computed partition is plotted against the required level of significance α for each of the significance-based methods. The customary $\alpha = 0.05$ threshold of significance is shown by a dashed vertical. In the right portion (**C** or **F**), the relative cRI of the computed partition is plotted against the minimal allowed number of leaves for DTC.

Table 8 Quality of partition in Leukemia*

Dissimilarity and linkage	α	TBEST	SC	SLB
Euclidean dissimilarity Ward linkage	0.001	1(1)	0(-)	0.91(0.92)
	0.05	0.92(0.92)	0(-)	0.91(0.92)
(1 - Pearson correlation) dissimilarity average linkage	0.001	1(1)	0(-)	0(-)
	0.05	0.83(0.86)	0(-)	0(-)

* Values of α in second column are significance levels. Quality of partitions is recorded as cRI (V measure), from column three to column five.

4.3 T10

The third benchmark dataset originates from DNA copy number analysis of 100 individual cells harvested from a breast tumor (Navin, Kendall et al. 2011). The true partition in this case is four-way, with the subsets differing from each other by ploidy as determined by cell sorting. Randomization method of this dataset is introduced in 3.1.2, it is done independently for each chromosome because strong geometric overlap between cores.

The results are illustrated in Figure 9. For the Euclidean dissimilarity - Ward linkage combination only TBEST and SLB identify the true partition, with TBEST succeeding in a broader range of significance level α. For the (1 – Pearson correlation) dissimilarity - average linkage combination TBEST outperforms the other two significance-based algorithms and matches the truth perfectly in a broad range of significance level α. Table 9 presents cRI and V measure estimated at significance level $\alpha = 0.001$ and $\alpha = 0.05$. Results of findings with TBEST obtain the highest scores with both dissimilarity and linkage cases, matching a partition of four subtypes exactly.

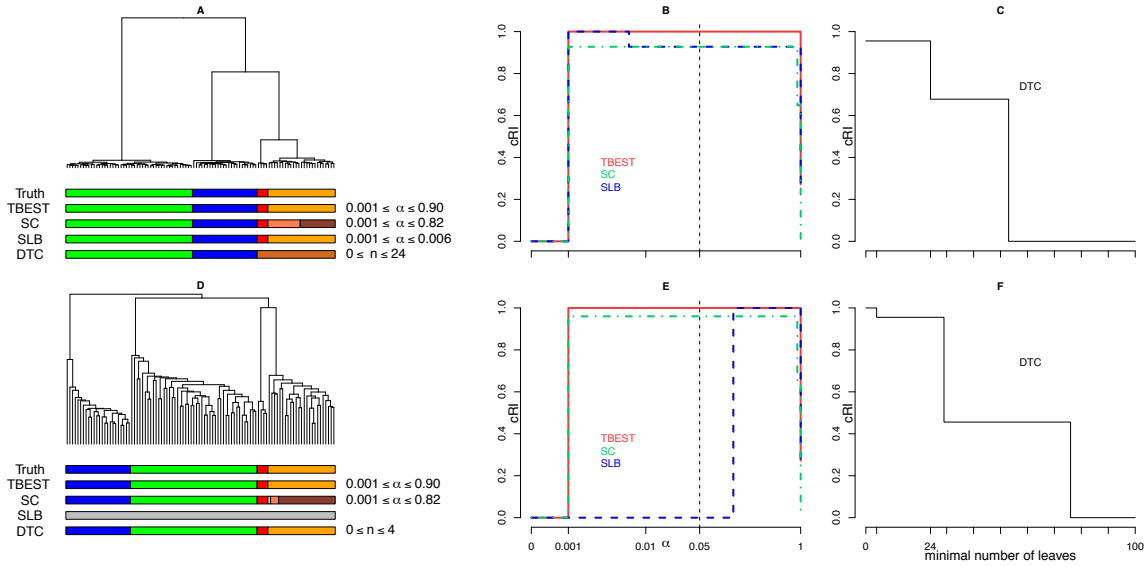


Figure 9 TBEST compared to published methods for T10.
Performance comparison of TBEST and the three published methods in T10 dataset for the Euclidean dissimilarity – Ward linkage combination (top) and for the (1 - Pearson correlation) dissimilarity – average linkage (bottom). For each combination the left portion (**A** or **D**) shows the corresponding dendrogram, under which then true partition and the partition best matching the truth for each of the methods are shown as color bars. In the middle portion (**B** or **E**), the relative cRI of the computed partition is plotted against the required level of significance α for each of the significance-based methods. The customary $\alpha = 0.05$ threshold of significance is shown by a dashed vertical. In the right portion (**C** or **F**), the relative cRI of the computed partition is plotted against the minimal allowed number of leaves for DTC.

Table 9 Quality of partition in T10*

Dissimilarity and linkage	α	TBEST	SC	SLB
Euclidean dissimilarity Ward linkage	0.001	1(1)	0.93(0.93)	1(1)
	0.05	1(1)	0.93(0.93)	0.96(0.95)
(1 - Pearson correlation) dissimilarity average linkage	0.001	1(1)	0.96(0.95)	0(-)
	0.05	1(1)	0.96(0.95)	0(-)

* Values of α in second column are significance levels. Quality of partitions is recorded as cRI (V measure), from column three to column five.

4.4 Organelles

Next, we consider a dataset derived from proteomic analysis of the content of four cellular compartments in each of six mouse tissues. The analysis is based on 4768 protein level readings (Kislinger, Cox et al. 2006).

The true partition of the data is by the cellular compartment, and the two hierarchical clustering methods considered here both have the branch structure organized by the compartment label, to a good approximation. Of the three significance-based methods compared, only TBEST reproduces the truth to the maximal extent possible for both combinations of dissimilarity and linkage, and it does so stably in the broadest range of the levels of significance (Figure 10).

DTC achieves top performance for the (1 - Pearson correlation) dissimilarity – Ward linkage combination if its minimal allowed number of leaves does not exceed that of the smallest compartment-associated branch of the tree. However, this property is lost for the (1 - Pearson correlation) dissimilarity – average combination where an additional cluster with two leaves is identified by DTC if the minimal number of leaves is set at or below 2.

B and E in Figure 10 are plots of cRI on computed partition with various significance levels α. Table 10 provides cRI and V measure of computed partition on significance-based methods, at significance level 0.001 and 0.05. Findings of clustering with TBEST obtain the highest scores with both dissimilarity and linkage cases.

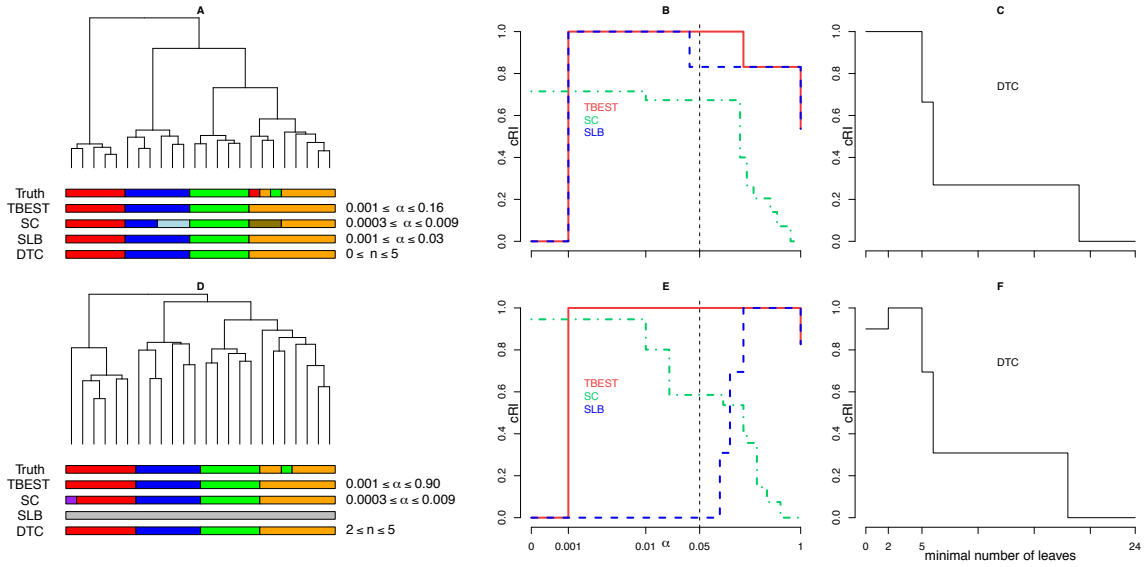


Figure 10 TBEST compared to published methods for Organelles.
Performance comparison of TBEST and the three published methods in Organelles dataset for the (1 - Pearson correlation) dissimilarity – Ward linkage combination (top) and for the (1 - Pearson correlation) dissimilarity – average linkage (bottom). For each combination the left portion (**A** or **D**) shows the corresponding dendrogram, under which then true partition and the partition best matching the truth for each of the methods are shown as color bars. In the middle portion (**B** or **E**), the relative cRI of the computed partition is plotted against the required level of significance α for each of the significance-based methods. The customary $\alpha = 0.05$ threshold of significance is shown by a dashed vertical. In the right portion (**C** or **F**), the relative cRI of the computed partition is plotted against the minimal allowed number of leaves for DTC.

Table 10 Quality of partition in Organelles*

Dissimilarity and linkage	α	TBEST	SC	SLB
(1 - Pearson correlation) dissimilarity Ward linkage	0.001	1(1)	0.71(0.87)	1(1)
	0.05	1(1)	0.63(0.84)	0.83(0.93)
(1 - Pearson correlation) dissimilarity average linkage	0.001	1(1)	0.96(0.96)	0(-)
	0.05	1(1)	0.55(0.80)	0(-)

* Values of α in second column are significance levels. Quality of partitions is recorded as cRI (V measure), from column three to column five.

4.5 Chondrosarcoma

Finally, we discuss the performance of the four methods on a dataset generated by flow cytometry analysis of cells harvested from human tissues and cell lines. Among 34 samples, two samples were identified as multivariate outliers and removed before clustering (Diaz-Romero, Romeo et al. 2010). The truth is a four-way partition, with three parts corresponding each to a different tissue of origin and the fourth part formed by cells from tumor cell lines.

We have identified three combinations of dissimilarity and linkage for which the tree structure is fully consistent with the true partition and performed comparative analysis for all three, as shown in Figure 11. For two of these combinations ((1 - Spearman correlation) dissimilarity – Ward linkage and (1 - Kendall correlation) dissimilarity – average linkage) partition by TBEST matches the truth in a range of acceptable levels of significance. SLB does so for the first and, in a narrow range of significance thresholds, for the third combination. SC fails to match the truth. Note the data dimension in this case is 11, and it is smaller than 32, the number of observations. This dataset is therefore outside the range of applicability of SC. For Manhattan dissimilarity – Ward linkage TBEST also matches the truth, albeit at low significance (α = 0.1). DTC performs well for the first and third combinations, but only matches the truth in a restricted range of numbers of leaves in the second case.

cRI and V measure of computed partition in significance-based methods are listed in Table 11. In the third dissimilarity and linkage combination, although TBEST does not find a most detailed partition composed from significant branches, neither SC nor SLB finds the optimal partition at significance level 0.001 and 0.05.

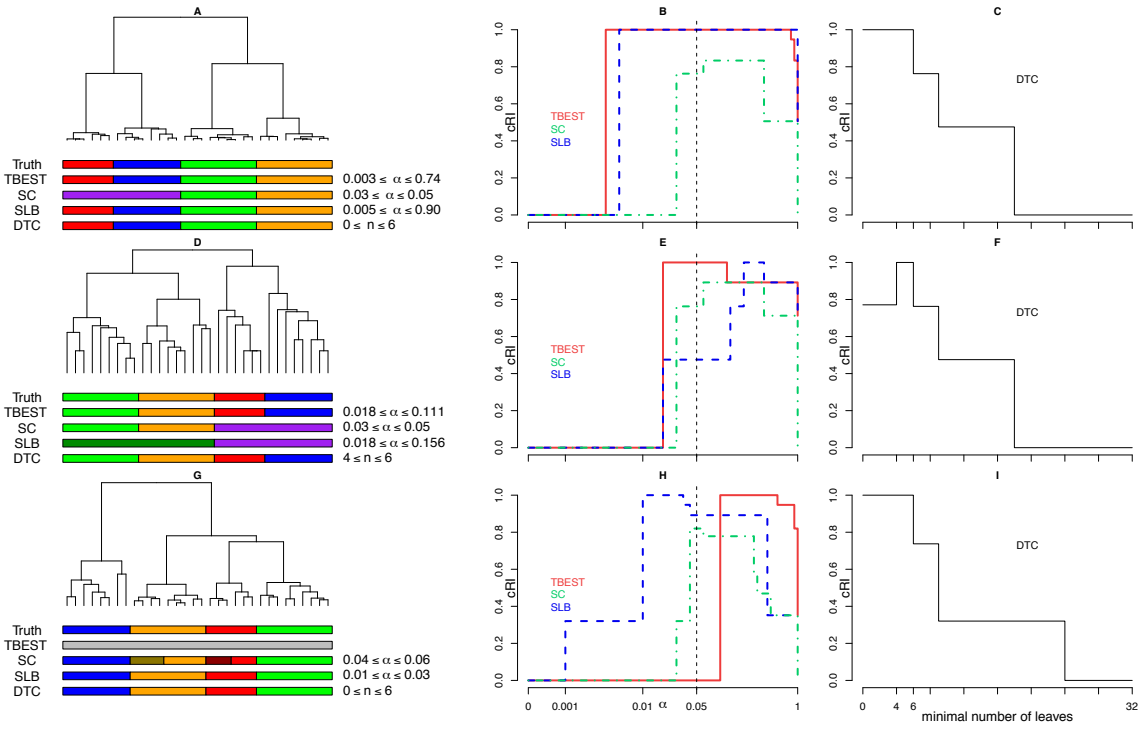


Figure 11 TBEST compared to published methods for Chondrosarcoma.
Performance comparison of TBEST and the three published methods in Chondrosarcoma dataset for the (1 - Spearman correlation) dissimilarity – Ward linkage combination (top), (1 - Kendall correlation) dissimilarity – average linkage combination (middle), and Manhattan dissimilarity – Ward linkage (bottom). For each combination the left portion (**A**, **D** or **G**) shows the corresponding dendrogram, under which then true partition and the partition best matching the truth for each of the methods are shown as color bars. In the middle portion (**B**, **E** or **H**), the relative cRI of the computed partition is plotted against the required level of significance α for each of the significance-based methods. The customary $\alpha = 0.05$ threshold of significance is shown by a dashed vertical. In the right portion (**C**, **F** or **I**), the relative cRI of the computed partition is plotted against the minimal allowed number of leaves for DTC.

Table 11 Quality of partition in Chondrosarcoma*

Dissimilarity and linkage	α	TBEST	SC	SLB
(1 - Spearman correlation) dissimilarity Ward linkage	0.001	0(-)	0(-)	0(-)
	0.05	1(1)	0.76(0.88)	1(1)
(1 - Kendall correlation) dissimilarity average linkage	0.001	0(-)	0(-)	0(-)
	0.05	1(1)	0.76(0.88)	0.48(0.67)
Manhattan dissimilarity Ward linkage	0.001	0(-)	0(-)	0.32(0.58)
	0.05	0(-)	0.82(0.89)	0.89(0.94)

***** Values of α in second column are significance levels. Quality of partitions is recorded as cRI (V measure), from column three to column five.

4.6 Summary of Chapter 4

In this chapter, performance of TBEST is validated in comparison with existing methods SC, SLB and DTC on five benchmark datasets (one synthetic and others from various biological origins), based on at least two combinations of dissimilarity and linkage. Furthermore, quality of partition found by TBEST and other significance-base methods is evaluated across a broad range of significance levels, using cRI and V measure. Validation result from each dataset is associated with one figure panel and one table.

There are eleven combinations of dissimilarity and linkage in total. Being a heuristic method, DTC is limited from its dependence on minimal number of leaves in a branch. It fails to find the optimal partition in the first combination of dataset T10, and it has unstable performance over four out of the other ten combinations. Each significance-based method is compared with TBEST at a broad range of significance levels. With any significance level less than or equal to 0.05, TBEST performs on par with or better than other significance-based methods in ten out of eleven combinations.

Chapter 5 Implementation

TBEST has been implemented, and published with open-source availability, in an R package named "TBEST" on CRAN website (Sun and Krasnitz 2013). If interested, users can simply install it and try TBEST out. The package website has link to manuals, including details on how to use functions, what results can be expected. This chapter offers an introduction and walk-through tutorial of using TBEST as a pipeline to detect tight branches, and find a most detailed partition.

5.1 Introduction to R package "TBEST"

This R package implements the methodology of Tree Branches Evaluated Statistically for Tightness (TBEST). Foreseeing the time-consuming issue of randomizations, randomizations are run in paralleled scripts. More user-friendly features in this package are:

1. Input of customized dissimilarity and randomization function
2. Two approaches to choose in *p*-value estimation
3. Visualization of tight branches and partition if exists

Functions used in TBEST pipeline are SigTree, PartitionTree, plot.best and LeafContent. SigTree estimats the tightness of branches, using randomizations and hypothesis testing. This is the main function that implements TBEST. It outputs statistics and *p*-values of branches. PartitionTree uses the output of SigTree, by examining tightness of branches, to search for the existence of a partition of candidate branches. plot.best is a visualization function that provides dendrogram with statistical significance on candidate branches, with annotated branch number. Interesting branches can be further studied by LeafContent, as it lists labels of observations within certain branch.

To run data through the above pipeline, all users need to do is to prepare data as a data matrix, or dataframe, with row corresponding to observations and column corresponding to variables. Techniques mentioned in Chapter 2 can be applied to ensure data quality before clustering.

An example of using TBEST is given in 5.2. Running time of TBEST is recorded in supplementary report of Appendix.

5.2 Example of using TBEST in R

In this section, we use a subset of benchmark data set, Leukemia, to give a tutorial over TBEST pipeline. More details can be found at the package website (Sun and Krasnitz 2013). After TBEST package in your R is installed, and loaded, users can simply type the same code in R console to reproduce this pipeline.

5.2.1 SigTree

This is the main function, and should be used in the first place to obtain estimates of tightness on branches. Instructions of functions in R, can always be shown with typing a question mark before the name of function, such as "?SigTree".

```
> data(leukemia)
> mytable<-SigTree(myinput=data.matrix(leukemia),mystat="fldc",
+ mymethod="ward",mymetric="euclidean",rand.fun="shuffle.column",
+ distrib="Rparallel",njobs=10,Ptail=TRUE)
> class(mytable)
[1] "best"
> names(mytable)
[1] "Call"       "data"       "indextable"
```

Figure 12 Usage of function SigTree

This tutorial uses a subset of Leukemia data set as input data, which has been included in the package. Usage of "SigTree" is shown from second to fourth command lines. Statistic of tightness addressed in Chapter 3 is named "fldc", there are alternative statistics can be chosen, see 5.1 and package website for more details. Combination of linkage and dissimilarity in HC is chosen within "mymethod" and "mymetric", correspondingly. With randomization function "rand.fun" equaling to "shuffle.column", randomization in this case is done by sampling expressions independently for each gene (column). Distributed computing option "distrib" is set to multi-core processing as "Rparallel", and "njobs" suggesting the number of workers. Ptail is an argument of logical values. If Ptail is TRUE (default), the Generalized Pareto Distribution is used to approximate the tail of the null distribution for each of the chosen measures (3.3.1). Otherwise, empirical p-values are computed directly from the corresponding samples.

Output of this function is an object of class "best". Details on values inside this

object can also be acquired by "?best". There are three items inside object "mytable", shown above. The third one "indextable" is a matrix containing hierarchical structures, number of leaves under branches, and estimates of tightness on branches, i.e. *p*-values on branches, the main objective of TBEST.

5.2.2 PartitionTree

This function finds the most detailed partition of a hierarchical tree into tight branches, given a level of significance for tightness.

```
> mypartition<-PartitionTree(x=mytable,siglevel=0.001,statname="fldc",
+         sigtype="raw")
> class(mypartition)
[1] "partition"
> names(mypartition)
[1] "Call"      "best"      "sigvalue"  "partition"
> mypartition$partition[1:4,]
                ID index
1 ALL_19769_B.cell    34
2 ALL_23953_B.cell    34
3 ALL_28373_B.cell    34
4  ALL_9335_B.cell    34
> length(unique(mypartition$partition[,2]))
[1] 3
```

Figure 13 Usage of function PartitionTree

While the first argument "x" being the object of class "best", produced by function "SigTree", arguments also include significance level "siglevel", i.e. threshold of significance for tightness of branches, the measure of tightness "statname", and how significance level should be interpreted "sigtype". Here "raw" means significance level used in hypothesis testing is directly 0.001, other options are "corrected" as correction of multiple hypothesis testing (3.2.2) and "fdr" as threshold of false discovery rate.

Output of this function is an object of class "partition". Details on values inside this object can also be acquired by "?partition". There are four items inside object "mypartition", shown above. The fourth one "partition" is a matrix containing two columns, labels of observations and which part, in terms of branch number among 1 to *n*-1, this observation belongs to. From this, we can further obtain number of parts and plot these using function below.

5.2.3 plot.best

This is a visualization function. It plots dendrogram and provides significance estimates of tight branches, or those that form a most detailed partition.

```
> ?plot.best
> plot(x=mytable,mystat="fldc",siglevel=0.001,sigtype="raw",hang=-1)
> plot(x=mytable,mystat="fldc",partition=mypartition,hang=-1)
```

Figure 14 Usage of function plot.best

"plot.best" is a method for class "best", it can be used as "plot" in short with input from class "best". First usage of this function is to visualize all tight branches, given a significance level. Second usage is to visualize a partition if exists. There are extra graphical arguments, such as logical values of showing labels of observations, logical values of showing number of branches, from 1 to *n*-1. *P*-values on branches are shown by multiplying a scale of 100, because of limitations on visualizing multi-digits. Exact estimates of tightness can be obtained from object "best".

Figure 15 corresponds to the second line of codes in Figure 14. This visualization needs only the output of SigTree. It shows that only branch 33, 34 and 7 are statistically tight, with *p*-values less than 0.001. Argument "hang" here is used to organize all leaves at height zero.

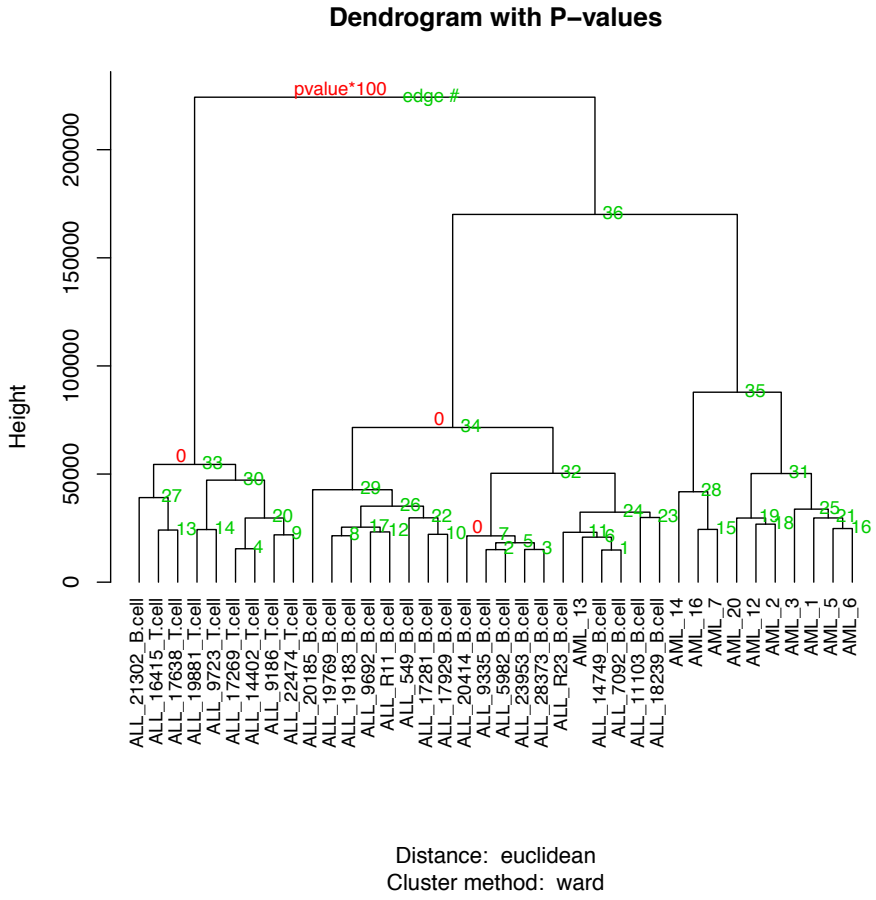


Figure 15 Visualization of statistically significant branches, produced by function plot.best

Figure 16 corresponds to the third line of codes in Figure 14. This visualization needs both the output of SigTree and the output of PartitionTree. It shows that branch 33, 34 and 35 form a most detailed partition of data. Note that branch 35 is not statistically significant, shown in Figure 15. However, branch 33 and 34 are statistically significant, tight in themselves, and far away from the rest branch. Relatively, we consider observations under branch 35 are also far away from those in the other two branches. More details on obtaining a partition are in 3.3.2. Therefore, a partition of three parts exists and their *p*-values (multiplying by 100) are shown on branches.

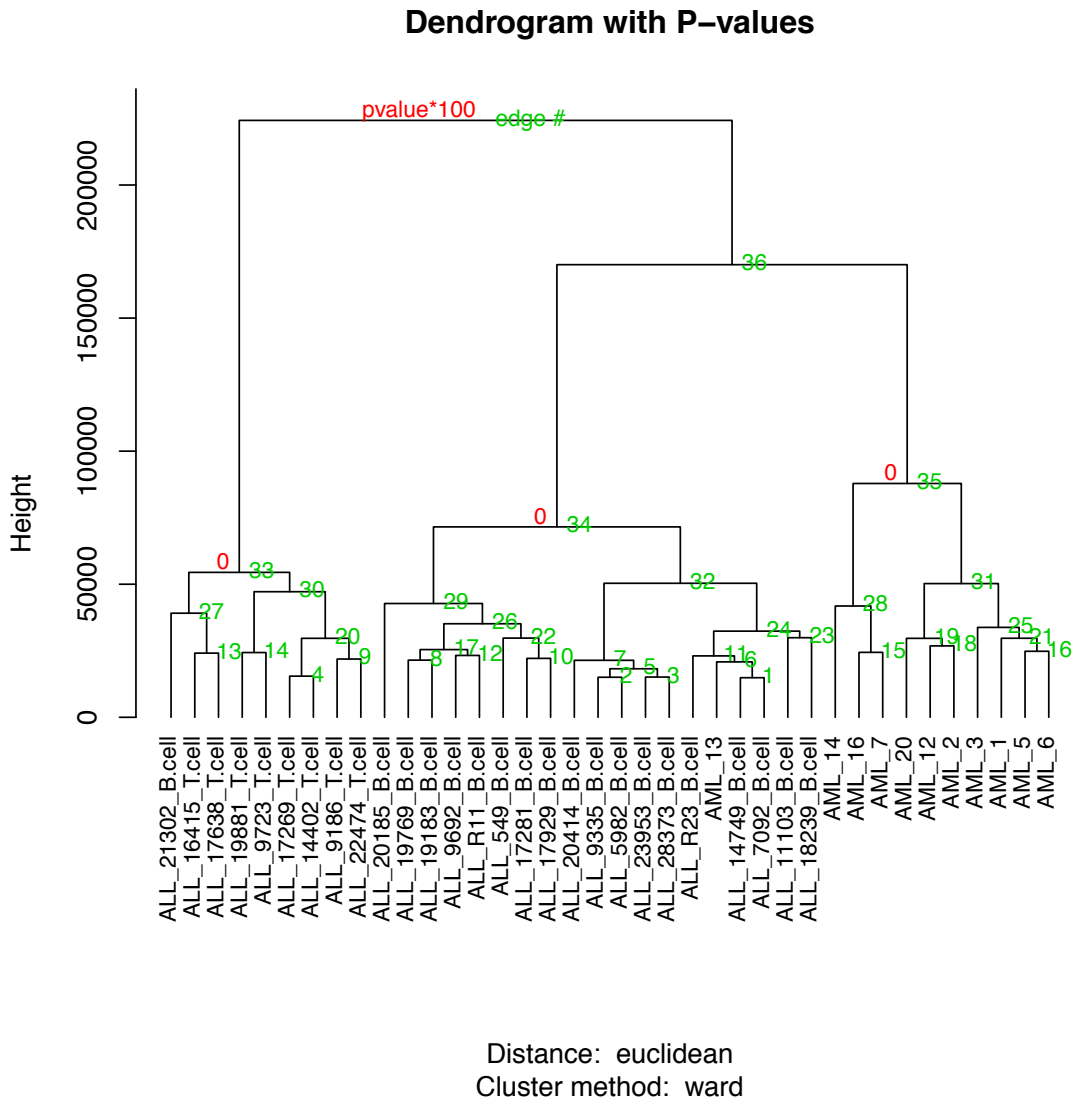


Figure 16 Visualization of three-part partition, and estimates of tightness on each part, produced by function plot.best.

5.2.4 LeafContent

As the name suggests, this function finds names of leaves belonging to given branches of a hierarchical tree. With a large number of observations, labels of leaves in Figure 15 and Figure 16 may not be able to be recognized, or to be shown by user's choice. Therefore a function to list labels of observations under interesting branches is designed. Input "myinput" is not restricted to object of class "best" or "partition", but

also class “hclust”, hierarchical structure produced by HC. “mynode” is an integer vector of the numbers of branches whose leaf content is desired.

```
> ?LeafContent
> LeafContent(myinput=mytable,mynode=c(1,28))
$`branch 1`
[1] "ALL_14749_B.cell" "ALL_7092_B.cell"

$`branch 28`
[1] "AML_14" "AML_16" "AML_7"
```

Figure 17 Usage of function LeafContent.

In above example, labels of observations under branch 1 and 28 are listed. The branch number is always positive, between 1 and 28. A singleton, however, is always a negative number, represented in R. If we use “mynode = c(-1, -28)” instead, this outputs label of the first individual observation and that of the 28th individual observation.

5.3 Summary of Chapter 5

This chapter serves as an introduction and also a tutorial on how to use TBEST. As a maintainer of this open-source package, I am responsible for keeping TBEST updated. Knijnenburg (Knijnenburg, Wessels et al. 2009) generously provided Matlab codes on their EVT-based *p*-value estimation. The most updated version of TBEST will include implementation of SLB as a test statistic and Bootstrapping as a test of cluster stability. More detailed and updated manuals can always be found at the package website.

Chapter 6 Discussion and Conclusion

As our test results demonstrate, the performance of TBEST as a tool for data partitioning is equal or superior to that of similar published methods in a variety of biology-related settings. This is true in particular for datasets with underlying tree-like organization, such as sets of genomic profiles of individual cancer cells, of the same type as our third benchmark case above. In a work presently in progress we are applying TBEST systematically to a number of datasets of a similar nature. But TBEST also performs well on datasets with no underlying hierarchical structure, such as Simulated6 or Leukemia above. In total, TBEST was able to recover the true partition of the data on par with or better than the published methods in ten out of eleven test cases considered here. We further note that in all but one case considered the optimal partition of the data by TBEST also was the most significant nontrivial partition. This was not the case for the other significance-based methods included in the comparison.

Now let us have a look at the three properties we mentioned in 1.2. There are no doubts that TBEST satisfies property 1) and 2). Statistic $S(n)$ is exactly a measurement on tightness of branch n. Statistical tests are conducted to find statistically significant branches, with distribution of null statistics obtained from randomized data. However, TBEST cannot guarantee that a partition into tight branches exists. The last paragraph in 3.2.3 suggests a partition to be significant with a threshold α if (a) every part is a branch and (b) if for every part at least one of the children of its parent node is tight with the p-value $p < \alpha$. In case of a nested distinct branch with its parent node and counterpart of its parent node being non-significant, TBEST can identify this branch as a statistically tight branch. But there may not exist a most detailed significant partition judged by criteria a) and b).

TBEST can both be applied and formulated more broadly. The applicability of TBEST is not limited to data partitioning that has been our focus here. TBEST can be

used for finding all significantly distinct branches of a hierarchical tree, regardless of whether these form a full partition.

6.1 Alternative statistics

Further, alternatives to the test statistic of Equation 1 can easily be devised, For example, for any non-leaf node *n* we can introduce

$$\sigma_1(n) \equiv \frac{h(n)-\frac{1}{2}[h(c_1(n))+h(c_2(n))]}{h(n)} \qquad (2)$$

, where $c_1(n)$, $c_2(n)$ are the two children of *n*.

Similarly, we can design alternative statistic for any non-leaf node *n*

$$\sigma_2(n) \equiv \frac{h(P(n))-\frac{1}{2}[h(n)-h'(n)]}{h(n)} \qquad (3)$$

, where $h'(n)$ is the height of node *n*'s sibling node.

These alternative statistics share the property that tightness of any non-leaf node is a number between 0 and 1. While this dissertation focused on validation of better or on par performance of TBEST using statistic proposed in Chapter 3, an implementation of TBEST as an R language package provides a number of these alternative options, both for the definition of tightness and for annotation of significantly distinct branches (Sun and Krasnitz 2013).

6.2 Cluster stability

Finally, we note that tightness of tree branches is complementary to another important notion in clustering, namely, cluster stability under re-sampling of the input data. The latter property can be analyzed in a number of ways, such as bootstrap analysis of trees (Felsenstein 1985, Efron, Halloran et al. 1996, Shimodaira 2002) or methods not directly related to trees (Dudoit and Fridlyand 2002, Monti, Tamayo et al. 2003). Existing work provides examples where both distinctness and stability under resampling are prerequisites of a meaningful partition (Cancer Genome Atlas Research Network 2011).

Previously in Chapter 4, most detailed significant partitions found by TBEST are validated composed of biological meaningful subtypes. Stability of these branches is examined, with bootstrapping sampling. The algorithm is straightforward,

1. For a tight cluster detected by TBEST, obtain the labels of observations in it, as set *C*.

2. Generate M bootstrapping samples, and grow hierarchical tree on each bootstrapping sample. Note that each sample is generated by sampling variables with replacement.
3. In each hierarchical tree, find the cluster C^* with similar size and most same labels of observations. Calculate stability score

$$B^* = \text{size of } C \cap C^* / \; size\; of\; C$$

4. Compute the average stability score over M bootstrapping samples

$$B = \frac{1}{M}\sum B^*$$

This score measures how likely occurrence of this cluster is in a number of bootstrapping samples, which in this sense suggests cluster stability. Each table below lists stability score of statistically significant branches. Score is computed with M = 1000 bootstrapping samples. An additional parameter r shown in the table has two values 0.5 and 1.0. It represents the proportion of variables sampled in each bootstrapping sample. We see, from the tables, cluster stability is higher in bootstrapping samples with original size of variables (r = 1) and those with half size of variables (r = 0.5). On the other aspects, changing the size of variables reveals meaningful distribution of variables and examines existence of dominant redundant variables.

Table 12 Cluster stability in Simulated6*

	Euclidean dissimilarity, complete linkage		(1 - Pearson correlation) dissimilarity, average linkage	
	r = 0.5	r = 1.0	r = 0.5	r = 1.0
cluster (class 1)	0.99	0.99	0.99	0.99
cluster (class 2)	0.98	0.98	0.99	0.99
cluster (class 3)	0.92	0.99	0.98	0.99
cluster (class 4)	0.95	0.99	0.99	0.99
cluster (class 5)	0.97	0.99	0.99	0.99
cluster (class 6)	0.96	0.99	0.99	0.99

* Label of dominant true subtype is shown in parentheses with corresponding cluster

Table 13 Cluster stability in Leukemia*

	Euclidean dissimilarity, Ward linkage		(1 - Pearson correlation) dissimilarity, average linkage	
	r = 0.5	r = 1.0	r = 0.5	r = 1.0
cluster (ALLT)	0.90	0.94	0.94	0.95
cluster (ALLB)	0.85	0.88	0.91	0.94
cluster (AML)	0.69	0.72	0.80	0.83

* Label of dominant true subtype is shown in parentheses with corresponding cluster

Table 14 Cluster stability in T10*

	Euclidean dissimilarity, Ward linkage		(1 - Pearson correlation) dissimilarity, average linkage	
	r = 0.5	r = 1.0	r = 0.5	r = 1.0
cluster (D+P)	0.99	0.99	0.99	1
cluster (H)	0.99	0.99	1	1
cluster (AA)	0.99	1	1	1
cluster (AB)	1	1	1	1

* Label of dominant true subtype is shown in parentheses with corresponding cluster

Table 15 Cluster stability in Organelles*

	(1 - Pearson correlation) dissimilarity, Ward linkage		(1 - Pearson correlation) dissimilarity, average linkage	
	r = 0.5	r = 1.0	r = 0.5	r = 1.0
cluster (Nuclei)	0.99	0.99	0.95	0.96
cluster (Cytosol)	0.98	0.99	0.99	0.99
cluster (Microsomes)	0.89	0.94	0.94	0.98
cluster (Mitochondria)	0.88	0.92	0.87	0.90

* Label of dominant true subtype is shown in parentheses with corresponding cluster

Table 16 Cluster stability in Chondrosarcoma*

	(1 - Spearman correlation) dissimilarity, Ward linkage		(1 - Kendall correlation) dissimilarity, average linkage		Manhattan dissimilarity, Ward linkage	
	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0
cluster (FIB)	0.60	0.76	0.61	0.76	0.63	0.75
cluster (HAC)	0.65	0.78	0.67	0.80	0.62	0.71
cluster (MSC)	0.60	0.72	0.60	0.73	0.65	0.70
cluster (TCL)	0.66	0.82	0.66	0.82	0.68	0.78

* Label of dominant true subtype is shown in parentheses with corresponding cluster

Most stability scores in Table 12 to Table 15 are well above 0.9, except cluster with AML subtype, last row in Table 13. This is consistent with interesting findings by

TBEST on benchmark case Leukemia (4.2), where substructures of AML subtype are revealed with lower significance. Clusters in Chondrosarcoma dataset have less stability (Table 16). The reason for this may be the number of variables is one third of the number of observations. Cluster is less stable when constructing hierarchical structures with variables that insufficiently explain observations. On the other hand, with $r = 1$, clusters under last combination in Table 16 appear most unstable, and this is the exact case when TBEST discovers the optimal partition with lowest significance.

6.3 Summary of Chapter 6

In this dissertation, our method TBEST performs equal to or superior to similar published methods in finding meaningful partition on a variety of biology-related settings. It has a broad application not only on finding partitions of datasets but also on detecting statistical significance on internal branches. We have further discussed alternative statistics and cluster stability. Alternative statistics have been implemented in R package with a number of options (Sun and Krasnitz 2013). Comparison of performance within this statistic family among various datasets is beyond the scope of this work, but should be an interesting aspect to look into in the future. Cluster stability has been incorporated to latest TBEST implementation (Sun and Krasnitz 2013). Cluster stability, together with statistical evaluation on tightness of cluster, provides more insights on significance of internal branches in a hierarchical tree, and thus can detect biologically meaningful partition and clusters with more confidence.

Bibliography

Altman, N. S. (1992). "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression." The American Statistician **46**(3): 175-185.

Bradley, P. S. (1997). "Clustering via Concave Minimization."

Cancer Genome Atlas Research Network (2011). "Integrated genomic analyses of ovarian carcinoma." Nature **474**(7353): 609-615.

Diaz-Romero, J., S. Romeo, J. V. Bovee, P. C. Hogendoorn, P. F. Heini and P. Mainil-Varlet (2010). "Hierarchical clustering of flow cytometry data for the study of conventional central chondrosarcoma." J Cell Physiol **225**(2): 601-611.

Dudoit, S. and J. Fridlyand (2002). "A prediction-based resampling method for estimating the number of clusters in a dataset." Genome Biol **3**(7): RESEARCH0036.

Efron, B., E. Halloran and S. Holmes (1996). "Bootstrap confidence levels for phylogenetic trees." Proc Natl Acad Sci U S A **93**(14): 7085-7090.

Eisen, M. B., Spellman, P.T., Brown, P.O., and Botstein, D. (1998). "Cluster analysis and display of genome-wide expression patterns." Proc. Natl. Acad. Sci. **95**(25): 14863-14868.

Felsenstein, J. (1985). "Confidence limits on phylogenies: An approach using the bootstrap." Society for the Study of Evolution **39**: 783-791.

Fung, B. C.M., K. Wang and M. Ester (2003). Hierarchical Document Clustering Using Frequent Itemsets. N PROC. SIAM INTERNATIONAL CONFERENCE ON DATA MINING

Golub, T. R., D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield and E. S. Lander (1999). "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring." Science **286**(5439): 531-537.

Hubert, L. and P. Arabie (1985). "Comparing Partitions." Journal of Classification **2**(2-3): 193-218.

Jain, A. K. (1988). Algorithms for Clustering Data, Prentice-Hall, Inc.

Kaufman, L. and P. J. Rousseeuw (1990). Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley and Sons.

Kislinger, T., B. Cox, A. Kannan, C. Chung, P. Hu, A. Ignatchenko, M. S. Scott, A. O. Gramolini, Q. Morris, M. T. Hallett, J. Rossant, T. R. Hughes, B. Frey and A. Emili

(2006). "Global survey of organ and organelle protein expression in mouse: combined proteomic and transcriptomic profiling." Cell **125**(1): 173-186.

Knijnenburg, T. A., L. F. A. Wessels, M. J. T. Reinders and I. Shmulevich (2009). "Fewer permutations, more accurate P-values." Bioinformatics **25**(12): I161-I168.

Krasnitz, A., G. Sun, P. Andrews and M. Wigler (2013). "Target inference from collections of genomic intervals." Proc Natl Acad Sci U S A **110**(25): E2271-2278.

Langfelder, P., B. Zhang and S. Horvath (2008). "Defining clusters from a hierarchical cluster tree: the Dynamic Tree Cut package for R." Bioinformatics **24**(5): 719-720.

Little, R. J. A. and D. B. Rubin (2002). Statistical Analysis with Missing Data.

Liu, Y., D. N. Hayes, A. Nobel and J. S. Marron (2008). "Statistical Significance of Clustering for High-Dimension, Low-Sample Size Data." Journal of the American Statistical Association **103**(483): 1281-1293.

Lloyd, S. (2006). "Least squares quantization in PCM." IEEE Trans. Inf. Theor. **28**(2): 129-137.

Marchette DJ, S. J. (2003). "Using data images for outlier detection." Comput Stat Data Anal **43**(4): 541-552.

Mazumder, R., T. Hastie and R. Tibshirani (2010). "Spectral Regularization Algorithms for Learning Large Incomplete Matrices." The Journal of Machine Learning Research **11**: 2287-2322.

Monti, S., P. Tamayo, J. Mesirov and T. Golub (2003). "Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data." Machine Learning **52**(1-2): 91-118.

Munneke, B., K. A. Schlauch, K. L. Simonsen, W. D. Beavis and R. W. Doerge (2005). "Adding confidence to gene expression clustering." Genetics **170**(4): 2003-2011.

Navin, N., J. Kendall, J. Troge, P. Andrews, L. Rodgers, J. McIndoo, K. Cook, A. Stepansky, D. Levy, D. Esposito, L. Muthuswamy, A. Krasnitz, W. R. McCombie, J. Hicks and M. Wigler (2011). "Tumour evolution inferred by single-cell sequencing." Nature **472**(7341): 90-94.

Pickands, J. (1975). "Statistical-Inference Using Extreme Order Statistics." Annals of Statistics **3**(1): 119-131.

Rand, W. M. (1971). "Objective Criteria for Evaluation of Clustering Methods." Journal of the American Statistical Association **66**(336): 846-&.

Rijsbergen, C. J. V. (1979). Information Retrieval, Butterworth-Heinemann.

Rosenberg, A. and J. Hirschberg (2007). V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure. Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning.

Shimodaira, H. (2002). "An approximately unbiased test of phylogenetic tree selection." Syst Biol **51**(3): 492-508.

Sun, G. and A. Krasnitz (2013). "TBEST: Tree branches evaluated statistically for tightness." The Comprehensive R Archive Network: http://cran.r-project.org/web/packages/TBEST/index.html.

Sun, G. and A. Krasnitz (2014). "Significant distinct branches of hierarchical trees: a framework for statistical analysis and applications to biological data." BMC Genomics **15**: 1000.

Ward, J. H., Jr. (1963). "Hierarchical Grouping to Optimize an Objective Function." Journal of the American Statistical Association **58**(301): 236-244.

Witten, D. M. and R. Tibshirani (2010). "A framework for feature selection in clustering." Journal of the American Statistical Association **105**(490): 713-726.

Zhao, Y and G. Karypis (2001). Criterion functions for document clustering: experiments and analysis University of Minnesota.

Appendix

Comparative analysis of time complexity and performance

1. Time Complexity

Time complexities of TBEST and of the three existing methods are listed in Table S1.1. These depend on the number of randomizations m, number of observations n and number of variables d. For the two top-down methods, SC and SLB, the complexities as stated correspond to the worst-case scenario wherein all internal nodes of the tree are examined.

Table S1.1 Time complexities of TBEST and of the published methods

TBEST	$O(mn^2d) + O(mn^3)$
SC	$O(mn^2d) + O(n^3d)$
SLB	$O(mn^3d) + O(mn^4)$
DTC	$O(n^2d)$

TBEST requires, for each randomization, to compute a dissimilarity matrix at a cost $O(n^2d)$ and to grow a hierarchical tree at a cost $O(n^3)$. Other computational costs, such as computing the tightness, are sub-dominant to these two.

The complexity of SC was computed under the worst-case assumption that one of the daughters at each internal node of the tree is a single leaf. With this assumption, computing statistic on n-1 hierarchies with each simulation, from 1 to m, needs $O(mn^2d)$. For each branch, from 1 to n-1, SC computes variance-covariance matrix [2], which takes $O(n^3d)$ in total. Other computational costs, such as computing the eigenvalues of the variance-covariance matrix, are sub-dominant to these two.

SLB performs randomization for each internal node being examined and requires computation of dissimilarity matrix and hierarchical clustering for each such randomization [3]. The necessity of performing these operations separately for each internal node explains the additional factor of n in the complexity of SLB compared to that of TBEST.

DTC does not perform statistical assessment of partitions, and its complexity is independent of m. The complexity as stated refers to the worst-case scenario, wherein the minimal allowed number of leaves on a branch is one. The dominant term in the complexity estimate comes from executing step 2 of the dynamic hybrid algorithm [1].

2. Performance

Here we report execution times of TBEST and of the other three methods for all combinations of datasets, dissimilarities and linkages studied in this work. These are reported in five tables, one per dataset.

The following computing resource was used:

MacBook Air
Processor Name: Intel Core i5
Processor Speed: 1.3 GHz
Number of Processors: 1
Total Number of Cores: 2
L2 Cache (per Core): 256 KB
L3 Cache: 3 MB
Memory: 4 GB

For TBEST, SC and SLB, 5000 randomizations were performed. For SC and DTC the packages sigclust and dynamicTreeCut were used, respectively [4-5]. For TBEST and SLB the R language package TBEST was invoked [6]. All these packages are publicly available from the Comprehensive R Archive Network (CRAN). The TBEST package facilitates parallel execution, and both cores of the processor were employed.

Table S2.1 Simulated6

Method / Combination	Euclidean dissimilarity complete linkage	(1 - Pearson correlation) dissimilarity average linkage
TBEST	74.00s	123.00s
SC	78.10s	78.10s
SLB	232.83s	122.76s
DTC	0.05s	0.04s

Table S2.2 Leukemia

Method / Combination	Euclidean dissimilarity Ward linkage	(1 - Pearson correlation) dissimilarity average linkage
TBEST	92.36s	118.83s
SC	92.47s	92.47s
SLB	400.42s	129.02s
DTC	0.03s	0.02s

Table S2.3 T10

Method / Combination	Euclidean dissimilarity Ward linkage	(1 - Pearson correlation) dissimilarity average linkage
TBEST	376.92s	446.75s
SC	297.65s	265.15s
SLB	1063.80s	448.84s
DTC	0.07s	0.06s

Table S2.4 Organelles

Method / Combination	(1 - Pearson correlation) dissimilarity Ward linkage	(1 - Pearson correlation) dissimilarity average linkage
TBEST	320.38s	320.22s
SC	2135.00s	2125.91s
SLB	1382.38s	326.29s
DTC	0.01s	0.01s

Table S2.5 Chondrosarcoma

Method / Combination	(1 - Spearman correlation) dissimilarity Ward linkage	(1 - Kendall correlation) dissimilarity average linkage	Manhattan dissimilarity Ward linkage
TBEST	117.84s	121.13s	5.50s
SC	30.01s	17.09s	54.99s
SLB	232.01s	194.40s	24.30s
DTC	0.03s	0.04s	0.02s

References

1. Langfelder P, Zhang B, Horvath S: **Defining clusters from a hierarchical cluster tree: the Dynamic Tree Cut package for R**. *Bioinformatics* 2008, **24**(5):719-720.

2. Liu Y, Hayes DN, Nobel A, Marron JS: **Statistical Significance of Clustering for High-Dimension, Low-Sample Size Data**. *Journal of the American Statistical Association* 2008, **103**(483):1281-1293.

3. Munneke B, Schlauch KA, Simonsen KL, Beavis WD, Doerge RW: **Adding confidence to gene expression clustering**. *Genetics* 2005, **170**(4):2003-2011.

4. Langfelder P, Zhang B, Horvath S: **dynamicTreeCut: Methods for detection of clusters in hierarchical clustering dendrograms**. *The Comprehensive R Archive Network*: http://cran.r-project.org/web/packages/dynamicTreeCut/index.html.

5. Huang H, Liu Y, Marron JS: **sigclust: Statistical Significance of Clustering**. *The Comprehensive R Archive Network:* http://cran.r-project.org/web/packages/sigclust/index.html.

6. Sun G, Krasnitz A: **TBEST: Tree branches evaluated statistically for tightness**. *The Comprehensive R Archive Network*: http://cran.r-project.org/web/packages/TBEST/index.html.

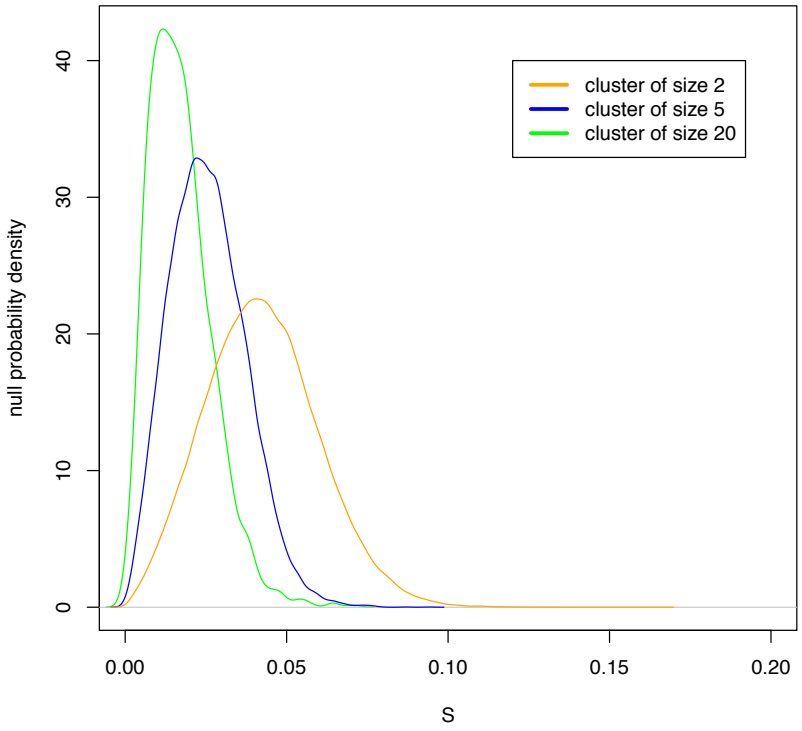
Figure S1

The null distribution of node tightness *S* depends on the number of leaves.

This dependence is illustrated for all the benchmarks and dissimilarity – linkage combinations analyzed. In each case the distributions of *S* are shown for nodes with 2, 5 and 20 leaves. Each plot is based on 5000 randomizations of the respective data set.

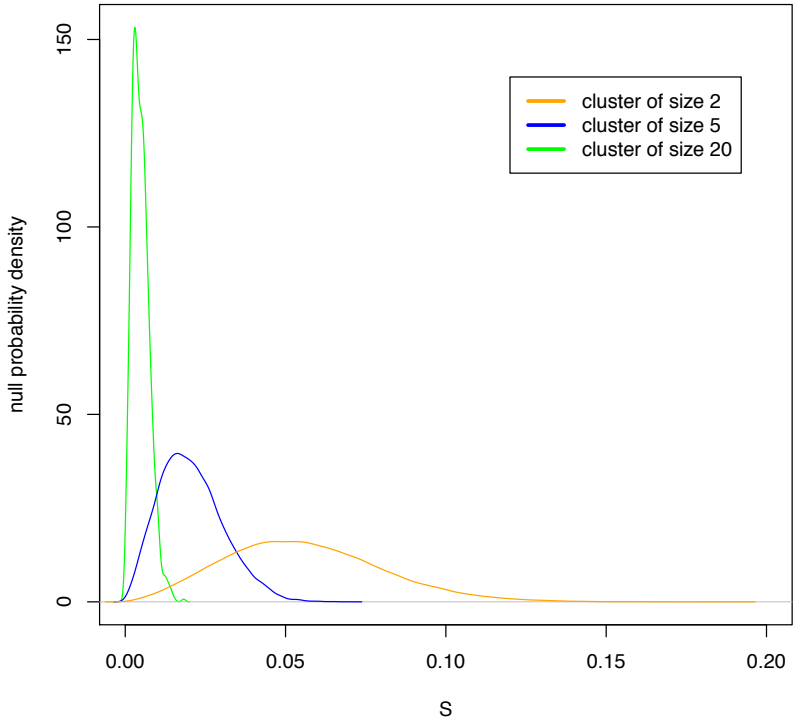
A: Simulated6

Euclidean dissimilarity – complete linkage combination



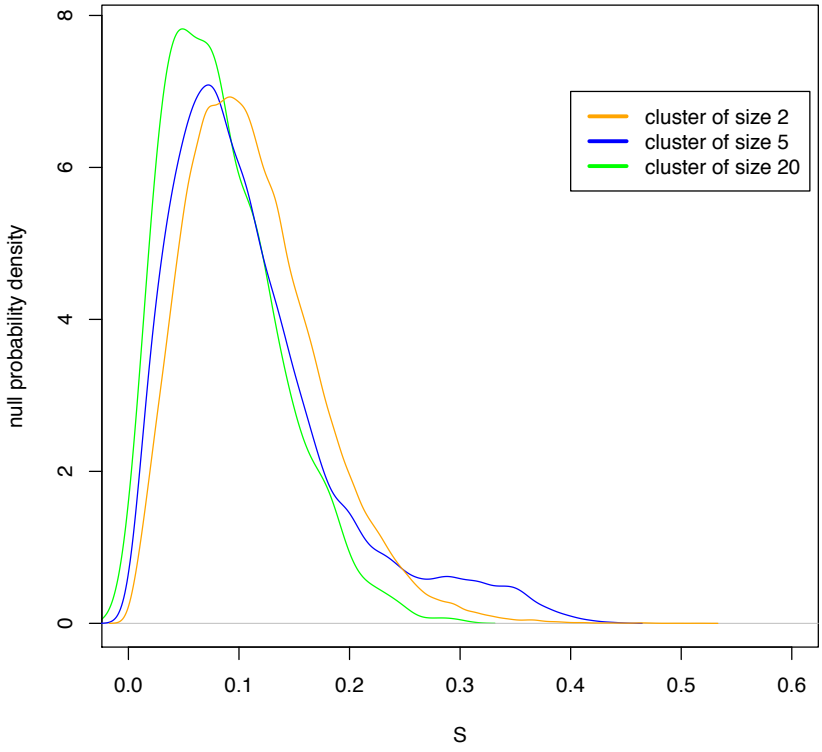
B: Simulated6

(1 - Pearson correlation) dissimilarity – average linkage combination



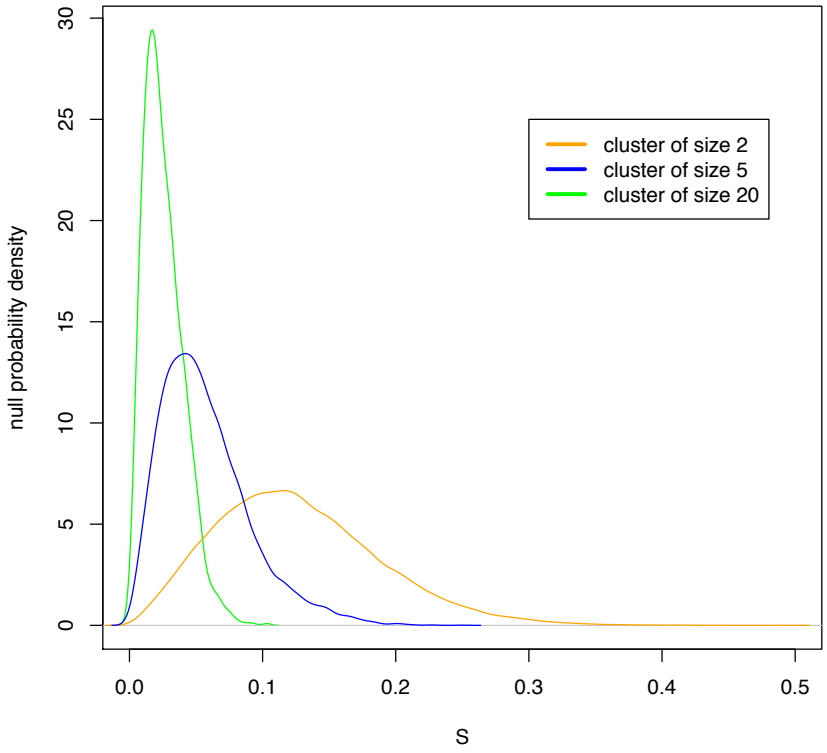
C: Leukemia

Euclidean dissimilarity – Ward linkage combination



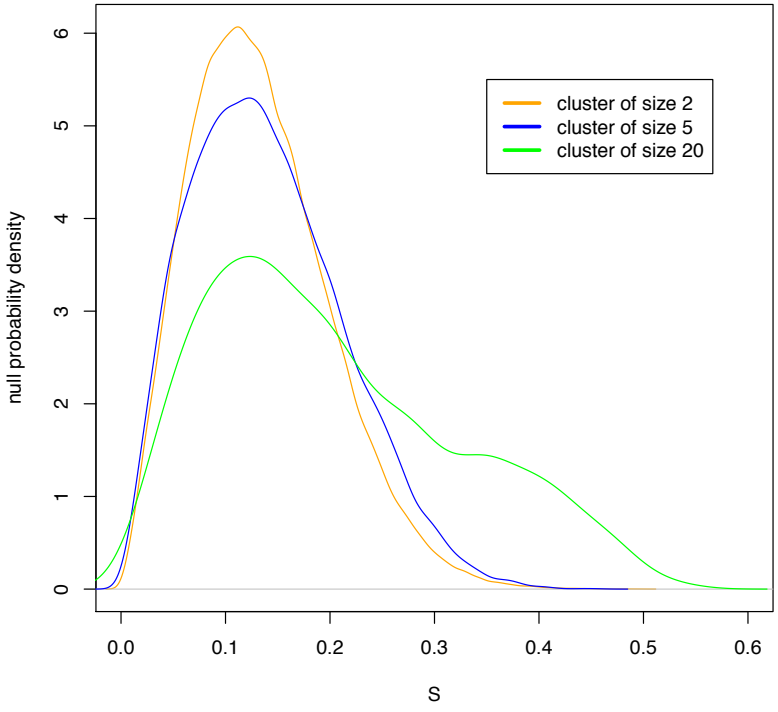
D: Leukemia

(1 - Pearson correlation) dissimilarity – average linkage combination



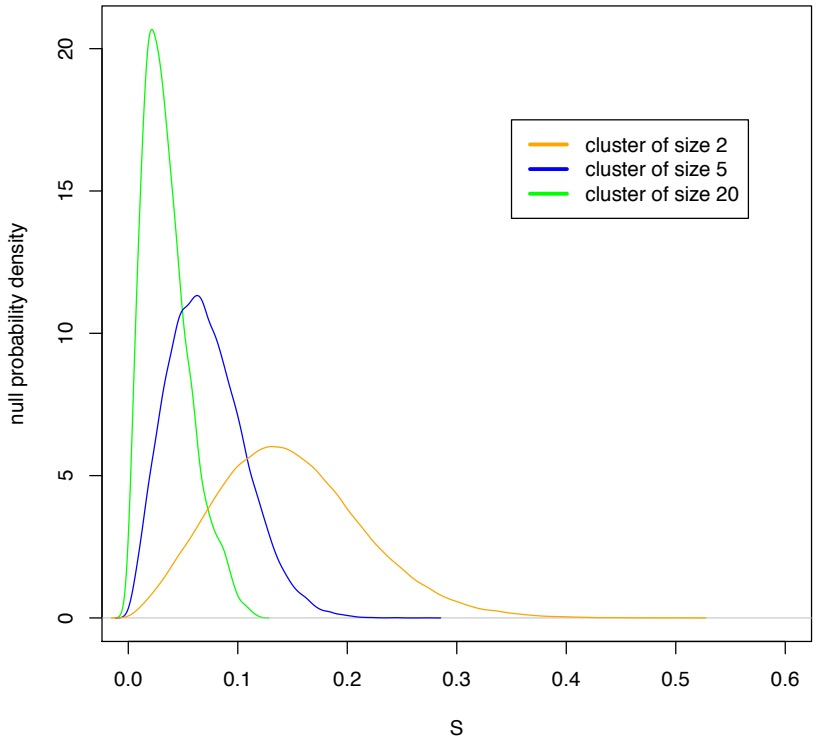
E: T10

Euclidean dissimilarity – Ward linkage combination



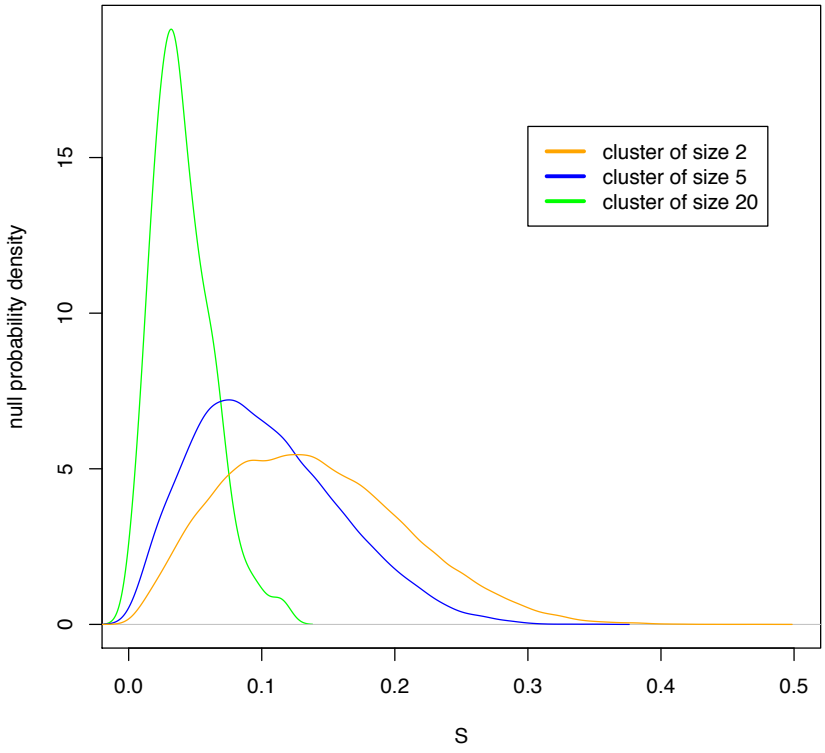
F: T10

(1 - Pearson correlation) dissimilarity – average linkage combination



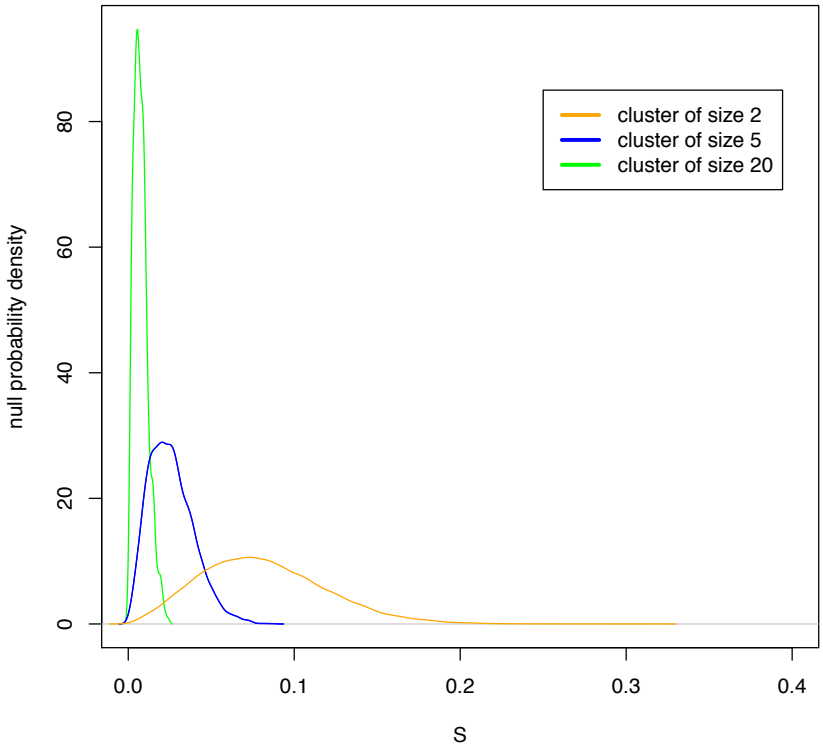
G: Organelles

(1 - Pearson correlation) dissimilarity – Ward linkage combination



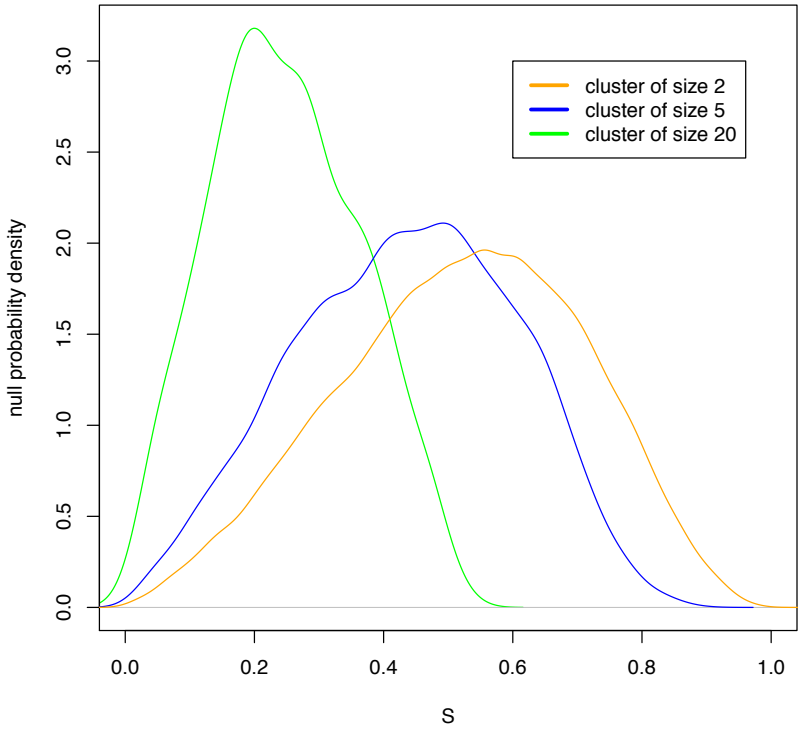
H: Organelles

(1 - Pearson correlation) dissimilarity – average linkage combination



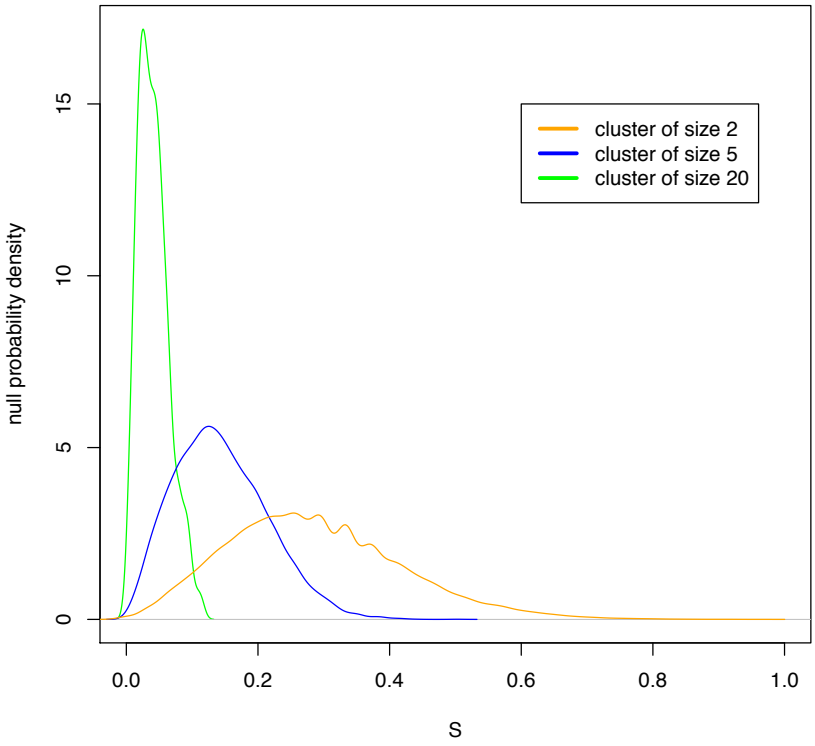
I: Chondrosarcoma

(1 - Spearman correlation) dissimilarity – Ward linkage combination



J: Chondrosarcoma

(1 - Kendall correlation) dissimilarity – average linkage combination



K: Chondrosarcoma

Manhattan dissimilarity – Ward linkage combination

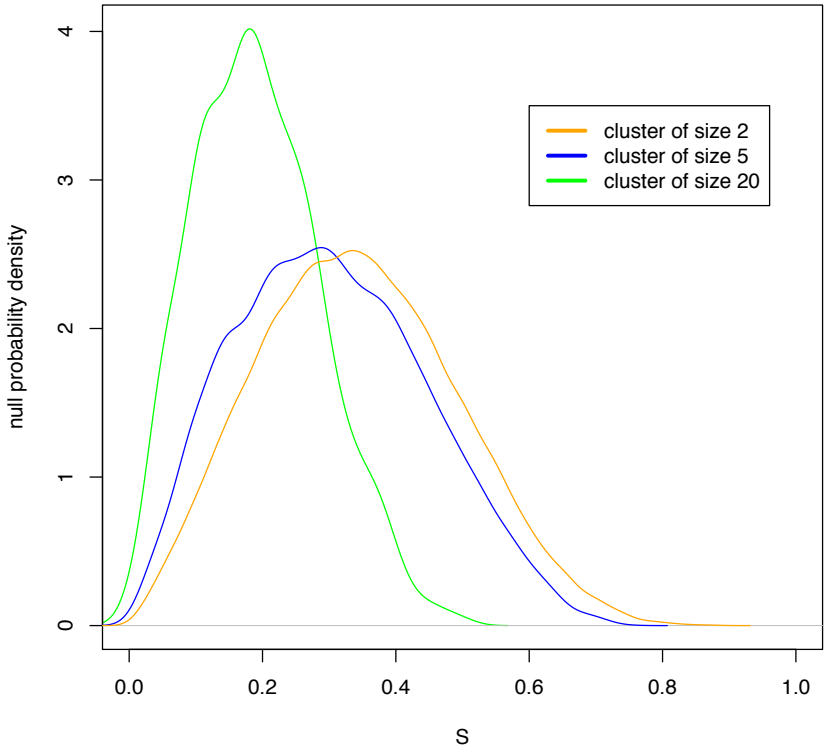


Figure S2

Empirical *p*-value estimates for tightness compared to EVT-based estimates.
Combined results for all tree nodes in all benchmark studies are shown. For each benchmark the combinations of dissimilarity and linkage are enumerated in the same order as they appear in Table 2. Displayed are the values corrected for hypothesis multiplicity (*cf* the Methods section). Empirical estimates are based on $1000 \times N$ randomizations each, N being the number of leaves. EVT estimates are based on 1000 randomizations each. If the empirical p-value estimate based on these 1000 randomization is large, the EVT algorithm defaults to this estimate. The corresponding points are shown by empty symbols of the appropriate shape and color. The diagonal dashed line indicates the identity. The vertical dashed line indicates the minimal multiplicity-corected empirical p-value $[1 - (1 - p_e)^{N-2}] / (n_r + 2)$, where N is the number of leaves and n_r is the number of randomizatons.

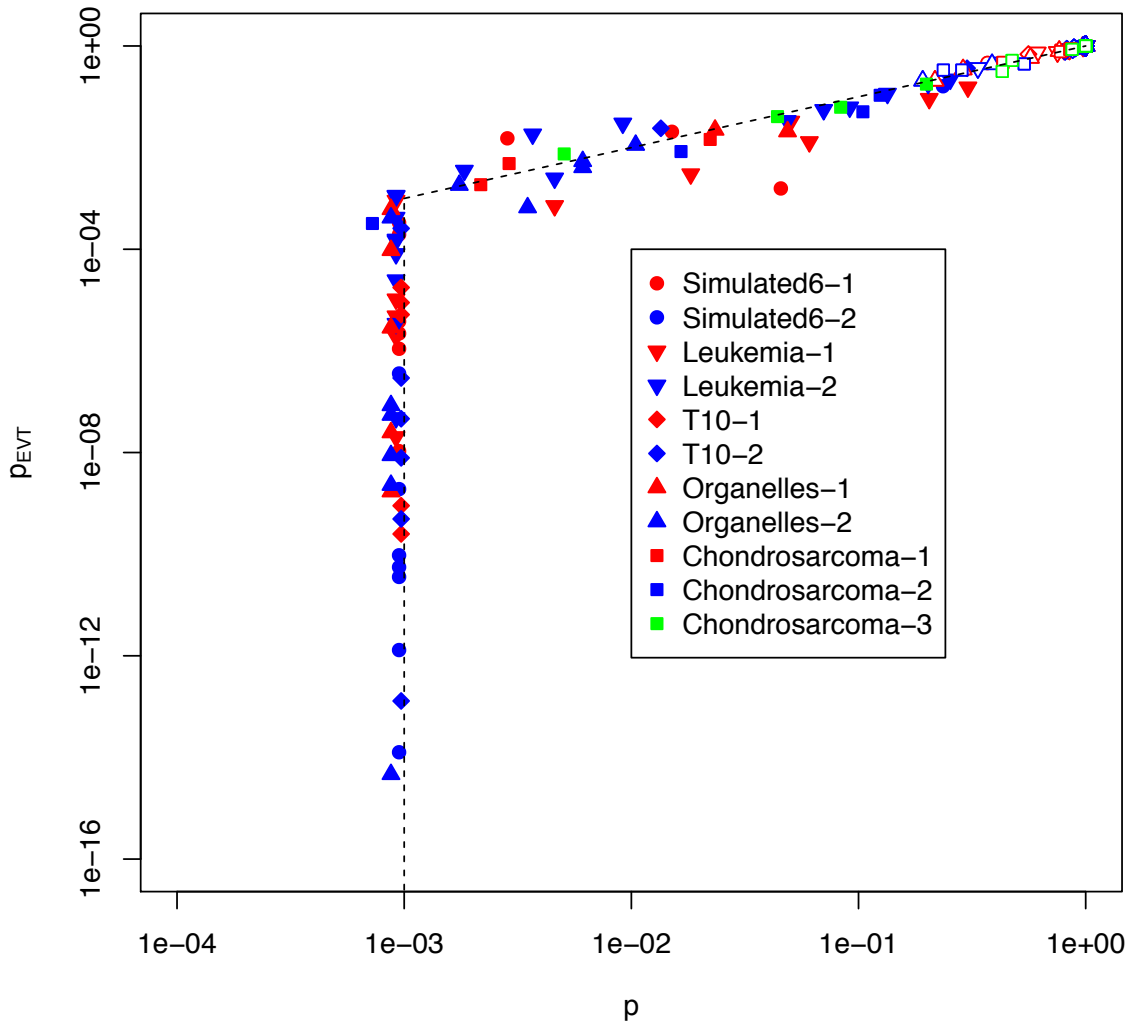


Table S1 Properties of Simulated6 dataset*

genes	1-50	51-100	101-150	151-200	201-250	251-300	301-600
observations with up-regulated genes	1-8	9-20	21-30	31-45	46-50	51-60	
mean of up-regulated expressions	594	699	296	296	401	344	
regular mean	40	69	40	39	37	39	38
standard deviation	50	75	100	101	200	203	200

* Among the first 300 genes, each block of 50 genes comes from a normal distribution with parameters as tabulated, except for those observations within the class that these genes are up-regulated. Within each block the regular and the up-regulated distributions differ in the mean but have equal standard deviations. The values for the last 300 genes are drawn from the same normal distribution for all observations.