Performance Evaluation of Consensus Clustering Algorithms

STAT 598 Progress Report

Derek Chiu

September 1, 2015

Contents

1 Preface

This progress report fulfills the UBC Science Co-op requirement to submit a work term report at the end of every four month period. BC Cancer Agency (BCCA) is a not-for-profit organization that aims to provide care for cancer patients and conduct innovative cancer research. Our department, OvCaRe, is the Ovarian Cancer Research team tasked with studying ovarian cancers of many types. The objective of the project I am working on is to discover a viable classifier for ovarian high-grade serous carcinoma (HGSC). My role is to help devise a clustering algorithm that can partition tumour samples of HGSC into different subtypes without knowledge of the underlying pathological properties of each sample. Instead, only gene expression data will be used in the prediction. The progress report will evaluate the method we are using, consensus clustering, on a publicly available data set as well as some simulated data sets. The final technical report will contain results of our method applied on HGSC data from our own cohort.

2 Introduction

Unsupervised learning is the process of inferring something about a data structure without knowing its true class labels. Cluster analysis is an unsupervised learning method of assigning entities into different groups based on one or more of their attributes. It is unsupervised because we do not know the true partitions of the entities. The goal is to place similar objects together in the same cluster and separate dissimilar objects into different clusters. For example, in genomics studies, we frequently try and cluster patient samples measured on a large number of molecular features.

When we obtain a clustering assignment from an algorithm, we often want to evaluate its performance and validity. Ideally, a good clustering algorithm is able to differentiate entities without knowledge of the true class labels. In addition, we want the algorithm to arrive at a stable clustering result. Some algorithms are sensitive to initial conditions and we do not want the assignments to be dependent on those. Finally, the choice of the number of clusters is not trivial in unsupervised explorations. This will not be a problem in simulations because we do know the true class labels. However, it is important to keep in mind that for real data the number of clusters should be determined using the data structure.

3 Methods

3.1 Clustering Algorithms

There are many clustering algorithms, each approaching the problem in a different way. It is important to note the advantages and limitations of each algorithm. These are some definitions of clustering performance[18]:

- **Compactness**: how close together or tight objects are to each other within a cluster
- **Separation**: how far apart objects are in different clusters
- **Connectivity**: how connected the objects are to their closest neighbours

3.1.1 Hierarchical Clustering

Hierarchical clustering (HC) is popular because of its intuitive representation using dendrograms (trees). More similar objects are joined near the bottom of a dendrogram whereas less similar objects are joined at a higher tree height. First a distance metric is used to compute a distance matrix. Then objects/features are clustered based on a linkage type. The linkage criterion determines the distances among a set of objects/features using the pairwise distances. For example, an average linkage would use the average pairwise distances. On the other hand, single linkage uses the minimum pairwise distance. A dendrogram with all objects/features can be made by recursively linking increasingly larger subsets of observations together.

Single linkage works well for data sets exhibiting connectivity but not compactness. This is because single linkage looks for minimum pairwise distances, which would cluster together neighbouring points. An example of this would be tree rings. The clusters are circles, and objects that are far away can be in the same cluster compared to objects that are actually closer. On the other hand, average linkage works well for data sets exhibiting compactness. This works where the clusters look like non-overlapping blobs.

3.1.2 K-Means and PAM

First, k means (centroids) are randomly initialized in the multidimensional object/feature space that we wish to cluster. The clusters are formed by assigning each object/feature to its closest centroid. The centroids are recalculated based on the cluster memberships and the subsequently updated. This procedure is iterated until the centroids converge.

There are two caveats to note when using k-means. First, sometimes the cluster assignments are unstable because they depend on the random initialization of the centroids. We preferably want to repeat the algorithm many times to see whether the clusters are sensitive to the choice of initial centroid. Secondly, choosing k is not arbitrary. Cross-validation using an appropriate loss function is a popular method for choosing k. Other methods use evaluation indices, some of which we will describe later in the report.

Partitioning Around Medoids (PAM) is very similar to k-means except that we randomly initialize k random data points (medoids). Medoids must be actual data points whereas centroids can be any point in the feature space.

3.1.3 Nonnegative Matrix Factorization (NMF)

Given a non-negative data matrix A, we can factor it into two matrices W and H, which are also non negative. W and H have important properties. Suppose A has genes as rows and samples as columns. If we are interested in clustering samples, then H has a reduced gene space of metagenes that fully explain the samples. Samples are clustered based on the metagene they are most associated with. If we are interested in clustering genes, then W has a reduced sample space of metasamples that fully explain the genes. Genes are clustered based on the metasample they are most associated with.

In gene expression data, it is common to standardize the genes. Doing so would likely disrupt the nonnegativity of A required for NMF. A simple remedy can solve this problem: append the matrix $-A$ to the bottom of A (preserving the same number of columns), and set all negative entries to 0. The computational complexity will have been doubled as a result. NMF takes a long time to run, but studies have shown clustering assignments can be highly stable[1].

3.2 Consensus Clustering

Monti et al.[2] describe consensus clustering as an algorithm but also as a method of combining realizations of other clustering algorithms. Algorithms like k-means and PAM are unstable, as the clustering assignments depend on the initialization of centroids and medoids, respectively. Consensus clustering combines results from repeated runs of a clustering algorithm. Typically, each run uses different random subsamples of the

data to model sampling variability. A final clustering assignment is determined based on agreement across replicates.

The consensus matrix is a significant aspect of consensus clustering. Consider a consensus matrix C. Entry C_{ij} is the proportion of runs that object i and object j were clustered together, out of all runs in which i and j were both selected in subsamples, where $1 \leq i, j \leq N$, N = number of objects. C is symmetric because $C_{ij} = C_{ji}$ (i.e. no order exists). All entries range from 0 to 1 since C_{ij} are proportions. A perfect clustering would consist of a consensus matrix with only 0 (objects never clustered together) or 1 (objects always clustered together). The final step to obtaining the consensus clustering assignment is to use hierarchical clustering on the consensus matrix.

Based on the hierarchical clustering, we can plot a heatmap of the consensus matrix. Repeating this for different values of k (number of clusters) and looking at the corresponding heatmaps, we can determine which value of k provides the most stable clustering assignment. The goal is to have a well-defined diagonal block structure in the heatmap, one block per cluster. Using a performance measure such as PAC (which we will describe) would be a more formal method of assessment that doesn't rely on potentially subjective visualizations.

Consensus clustering attempts to adjust for the randomness introduced by subsampling and the clustering algorithm itself. An extension is sometimes called meta-consensus clustering, where we aggregate results across different algorithms in addition to aggregating across subsamples within an algorithm. For instance, we may combine the consensus clustering assignments from 10 different algorithms to come up with a final clustering. The choice of which algorithms to use is not trivial. We hope to prove that using meta-consensus clustering is superior to using consensus clustering of one algorithm.

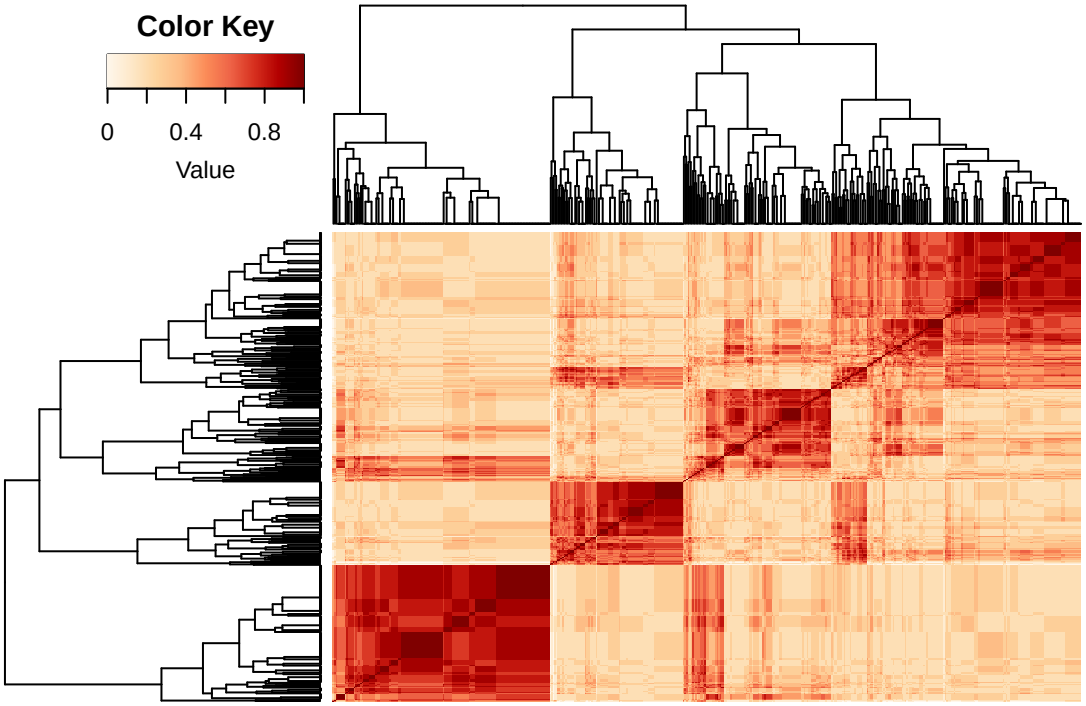
4 Performance Evaluation: TCGA Dataset

A published dataset from TCGA[3] uses 321 genes to cluster 489 HGSC samples into the four subtypes: *mesenchymal*, *immunoreactive*, *differentiated*, and *proliferative*. TCGA used consensus clustering with NMF. In this report, we consider the following clustering algorithms to use in consensus clustering:

- Hierarchical Clustering with Euclidean distance
  - Average Linkage
  - Single Linkage
  - DIvisive ANAlysis (DIANA)
- K-Means
  - Euclidean distance
  - Spearman distance
  - Mutual Information distance
- PAM
  - Euclidean distance
  - Spearman distance
  - Mutual Information distance
- NMF
  - KL divergence
  - Euclidean distance

The number of repetitions used for consensus clustering is 1000. Each subsample (replicate) of the data uses 80% of the total number of features.

The figure below shows the meta-consensus matrix across the 11 algorithms, each of which used consensus clustering.



From the heatmap, we do not see a very high concordance across algorithms. For example, the proportion of cases with at least 0.6 agreement is only 0.229. There is some evidence of a four-class data structure.

The confusion matrix is shown below for the meta consensus classes compared to TCGA's classification. Several metrics are shown for each class.

	TCGA	C1	C2	C4	C5
Meta					
C1		106	23	23	8
C2		2	59	11	4
C4		0	25	91	22
C5		1	0	10	104

	Sensitivity	Specificity	Pos Pred Value	Neg Pred Value	Prevalence	Detection Rate	Detection Prevalence	Balanced Accuracy
Class: C1	0.972	0.858	0.662	0.991	0.223	0.217	0.327	0.915
Class: C2	0.551	0.955	0.776	0.884	0.219	0.121	0.155	0.753
Class: C4	0.674	0.867	0.659	0.875	0.276	0.186	0.282	0.771
Class: C5	0.754	0.969	0.904	0.909	0.282	0.213	0.235	0.861

We hope to use cluster evaluation indices to give more weight to better performing algorithms in the construction of the meta-consensus clusters. Here we outline two main types of evaluation measures: external evaluation indices and internal evaluation indices.

4.1 External Evaluation

External evaluation refers to the situation when we compare our clustering assignments to true class labels, gold standards, or reference labels. In the TCGA analysis, we use their published clustering result as reference labels. The downside of using external evaluation is that the reference classes may not be correctly clustered themselves, and we are treating these as the norm. None the less, we can explore a few metrics to see how our own clustering performance compares.

We expect that our own NMF-based algorithms will perform well on these evaluation indices because the reference classes were clustered using NMF too.

4.1.1 Purity and Entropy

Purity is defined as the sum of the entities of maximal class in each cluster divided by the total number of entities[4]. The equation is:

$$Purity = \frac{1}{n} \sum_{r=1}^{k} \max_{i}(n_r^i)$$

where n is the total number of entities, k is the number of clusters, i is a particular class, and n_r^i is the number of objects classified into class i in cluster r. The larger the purity, the better the clustering accuracy.

Algorithms	Purity
NMF (Div)	0.7669
NMF (Euc)	0.7444
PAM (Spear)	0.7362
KM (Spear)	0.7157
PAM (Euc)	0.6687
HC (Diana)	0.6646
KM (Euc)	0.5971
KM (MI)	0.4376
PAM (MI)	0.4131
HC (A. Euc)	0.2883
HC (S. Euc)	0.2843

Entropy measures the amount of uncertainty in each cluster[4]. The equation is:

$$Entropy = -\frac{1}{n \log q} \sum_{r=1}^{k} \sum_{i=1}^{q} n_r^i \log \frac{n_r^i}{n_r}$$

where n, k, i, and n_r^i are same as above, and q is the number of classes. The smaller the entropy, the less uncertain we are of the cluster membership, and the better the clustering performance.

Algorithms	Entropy
NMF (Div)	0.5101
NMF (Euc)	0.5322
PAM (Spear)	0.5471
KM (Spear)	0.5703
PAM (Euc)	0.6012
HC (Diana)	0.6035

Algorithms	Entropy
KM (Euc)	0.6634
PAM (MI)	0.8759
KM (MI)	0.9107
HC (A. Euc)	0.9814
HC (S. Euc)	0.9891

4.1.2 Kappa Statistics

Cohen's kappa statistic κ measures the level of agreement between two raters[5]. In our case, the raters are different clustering algorithms. The equation is shown below:

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

where p_o is the proportion of entities agreed upon and p_e is the proportion of entities expected to agree by chance.

The weighted κ statistic is as follows and takes into account the off-diagonal elements of the confusion matrix between the two raters[6]. In other words, the disagreements between the raters are weighed in the calculation:

$$\kappa_w = 1 - \frac{\sum_{i=1}^{k}\sum_{j=1}^{k} w_{ij}o_{ij}}{\sum_{i=1}^{k}\sum_{j=1}^{k} w_{ij}e_{ij}}$$

where k is the number of classes, and w_{ij}, o_{ij}, and e_{ij} are the weight, observed, and expected matrices respectively.

Fleiss provided the following rating scheme for κ[7]. Albeit arbitrary and not numerically determined they still serve as a rough guideline.

Rating	Kappa
Poor	Less than 0.40
Fair	Between 0.40 and 0.75
Excellent	Greater than 0.75

The results for the TCGA dataset are shown below. The ratings are based on the weighted κ:

Algorithm	kappa	weighted.kappa	Rating
NMF (Euc)	0.658	0.7912	Excellent
NMF (Div)	0.6887	0.7894	Excellent
Meta	0.6477	0.7826	Excellent
KM (Spear)	0.6215	0.7606	Excellent
KM (Euc)	0.4625	0.741	Fair
PAM (Spear)	0.6501	0.7389	Fair
PAM (Euc)	0.5596	0.7282	Fair
HC (Diana)	0.5539	0.7236	Fair
KM (MI)	0.2521	0.2978	Poor
PAM (MI)	0.2247	0.2188	Poor
HC (A. Euc)	0.0195	0.02409	Poor
HC (S. Euc)	0.004911	0.01442	Poor

4.1.3 Adjusted Rand Index

The Rand Index measures agreement between two classes by counting the number of pairs of objects that are clustered together in two different clustering assignments[8]. Hubert & Araibe propose an adjusted version to account for random chance[9]. The equation for the adjusted Rand index (ARI) is shown below[8,9,10]:

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

where n_{ij}'s are entries in the prediction vs. reference confusion matrix, a_i and b_j are row and column marginal totals respectively, and n is the grand total. The ARI is 1 when the two clusterings are perfect, and 0 when there is no concordance.

Algorithms	ARI
NMF (Div)	0.4799
NMF (Euc)	0.4435
PAM (Spear)	0.427
KM (Spear)	0.4049
PAM (Euc)	0.3434
HC (Diana)	0.3415
KM (Euc)	0.2559
PAM (MI)	0.07465
KM (MI)	0.07369
HC (S. Euc)	-0.0002933
HC (A. Euc)	-0.0008894

4.2 Internal Evaluation

Interval evaluation assesses how well clustered objects are based on the data features they are measured on. Ideally, we want objects in the same cluster to be close together and objects in different clusters to be far from each other. Different definitions of distance define compactness and separation differently. Some evaluation indicies combine these distances into a unitless measure we can use to compare clustering results.

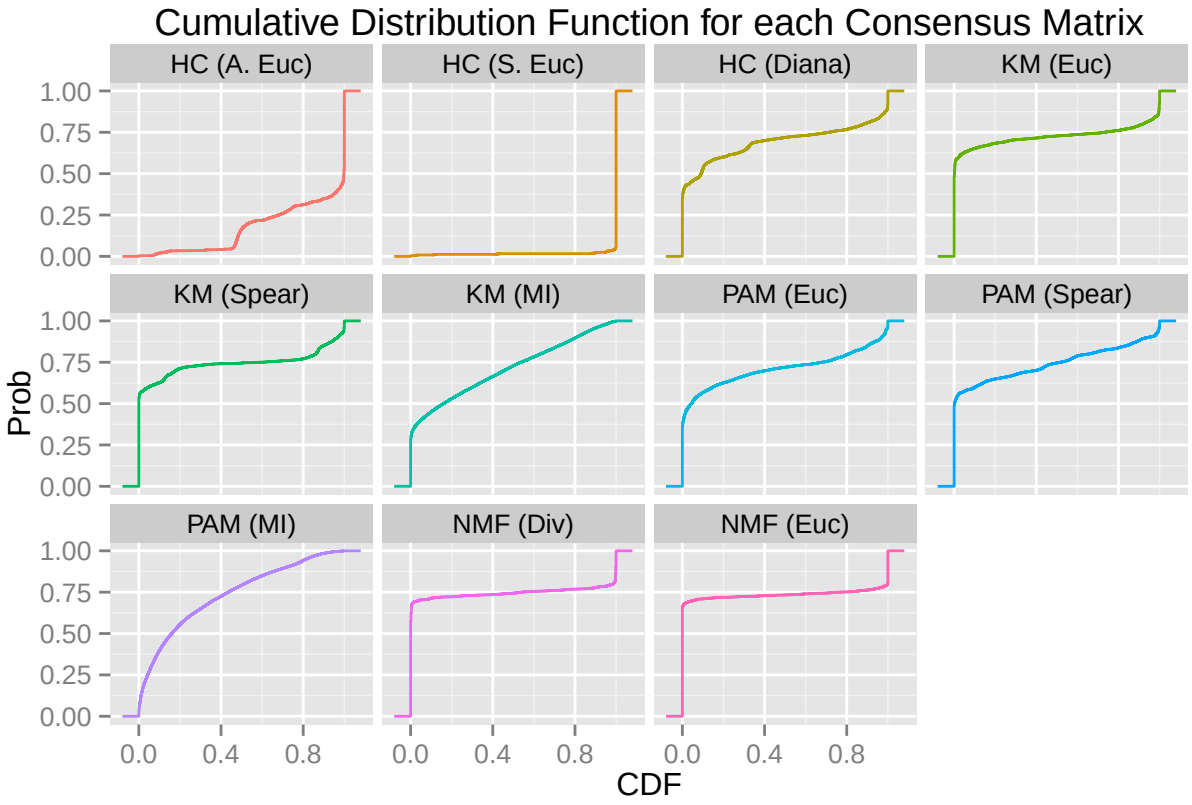
4.2.1 Proportion of Ambiguously Clustered Pairs (PAC)

The notion of PAC is closely related to consensus clustering. Senbabaoglu et al. argue that PAC is a better measure of determining the optimal number of clusters from consensus matrices, compared to other measures such as the Gap statistic, CLEST, etc[11]. The idea is simple: the proportion of ambiguously clustered pairs is the number of entries in the consensus matrix that are not 0 or 1. Recall that the consensus matrix is symmetric, so we only consider the lower (or upper) triangular matrix. In most applications, the definition of ambiguity is less stringent, and any entry greater than p or less than $1 - p$ contributes to PAC, where p is a small proportion (e.g. 0.05).

Although Senbabaoglu et al. initially used PAC to determine the optimal number of clusters, here we are using it to compare different clustering algorithms. None the less, we want the PAC to be as small as possible. A perfect score for PAC would be 0, meaning that all entries in the corresponding consensus matrix are either 0 or 1. The advantage of using this index is that it utilizes results from consensus clustering and thus not biased towards a particular distance metric. However, a major limitation is that PAC only uses stability evidence and doesn't assess accuracy. A clustering algorithm can be consistently wrong and do well for PAC, masking the apparent performance of said algorithm to make correct classifications.

Algorithms	PAC
HC (S. Euc)	0.05077
NMF (Euc)	0.1406
NMF (Div)	0.3267
KM (Spear)	0.425
KM (Euc)	0.4398
PAM (Spear)	0.4748
HC (A. Euc)	0.5232
HC (Diana)	0.5558
PAM (Euc)	0.5984
KM (MI)	0.7167
PAM (MI)	0.9638

A related figure is the cumulative distribution function (CDF) for each consensus matrix. Each panel corresponds to one algorithm. Recall that only the lower triangular entries of each consensus matrix are graphed. The ideal CDF would look like a horizontal line between 0 and 1, and straight at 0 and 1, because this would correspond to PAC = 0.



4.2.2 Davies-Bouldin Index

The Davies-Bouldin Index (DBI) measures the ratio of the within cluster scatter to the between cluster separation[12]. Hence, we want to *minimize* DBI such that objects in the same cluster are not too scattered and objects in different clusters are well separated. The advantage of using DBI to compare algorithms is that it utilizes properties of the data structure, and not true class labels. However, like the PAC, it may not indicate we are making the correct partitions.

Algorithms	DBI
HC (S. Euc)	0.7637
HC (A. Euc)	1.309
NMF (Euc)	1.702
NMF (Div)	1.702
HC (Diana)	1.719
PAM (Spear)	1.72
PAM (Euc)	1.725
KM (Spear)	1.725
KM (Euc)	1.741
PAM (MI)	1.935
KM (MI)	1.972

4.2.3 Dunn Index

The Dunn Index (DI) is a ratio of the minimum intercluster distance to the maximum intracluster distance[13]. In this report, we use the following definitions for the cluster-specific distances:

- Intercluster distance: distance between two farthest points in different clusters
- Intracluster distance: distance between two closest points in the same cluster

As a result, the higher the DI the better the clustering assignment. Similar to the DBI, the DI uses the data itself to determine clustering performance.

Algorithms	DI
HC (S. Euc)	0.4364
HC (A. Euc)	0.3696
KM (Spear)	0.3181
PAM (Spear)	0.3144
NMF (Euc)	0.2888
PAM (Euc)	0.286
HC (Diana)	0.2816
NMF (Div)	0.2766
KM (Euc)	0.2408
KM (MI)	0.2135
PAM (MI)	0.2135

4.2.4 Rousseeuw's Silhouette

Rousseuw's Silhouette internal cluster quality index (RS) measures how well each object is clustered by comparing its dissimilarity with other points in its own cluster to points in its neighbouring cluster[14]. The silhouette index ranges from -1 to 1. If the index is close to 1 then the object is clustered well and if it is close to -1 then the object would be better clustered in the neighbouring cluster.

The average silhouette index measures how well *all* objects are clustered, and is the measure we use here to compare the different algorithms. Just like the individual silhouette indices, we want to maximize the RS as well.

Algorithms	RS
HC (S. Euc)	0.1781

Algorithms	RS
HC (A. Euc)	0.1575
PAM (Euc)	0.1171
HC (Diana)	0.1124
NMF (Div)	0.1111
NMF (Euc)	0.1084
PAM (Spear)	0.1069
KM (Spear)	0.1059
KM (Euc)	0.08707
PAM (MI)	-0.0061
KM (MI)	-0.007504

4.2.5 C-Index

The C-Index (CI) as proposed by Hubert & Levin is a ratio of within-cluster dissimilarities[15]. For the CI, minimizing it indicates the optimal number of clusters. Here we use it to compare algorithms.

Algorithms	CI
KM (Euc)	0.2816
NMF (Div)	0.3023
PAM (Euc)	0.31
HC (Diana)	0.321
NMF (Euc)	0.3369
PAM (MI)	0.3376
PAM (Spear)	0.3489
KM (MI)	0.3529
KM (Spear)	0.356
HC (S. Euc)	0.4535
HC (A. Euc)	0.4795

4.2.6 Gamma Index

The Gamma Index described by Baker & Hubert is a ratio of the difference in the number of discordant comparisons to concordant comparisons, over all comparisons[16]. A high value for GI would thus indicate the clustering assignments have more concordance than discordance. Originally used to determine the optimal number of clusters, here we use it to compare algorithms.

Algorithms	GI
PAM (Euc)	1.754
PAM (Spear)	1.715
KM (Euc)	1.676
NMF (Euc)	1.666
HC (Diana)	1.639
KM (Spear)	1.627
NMF (Div)	1.617
HC (S. Euc)	0.7621
HC (A. Euc)	0.7232
KM (MI)	-3.142
PAM (MI)	-3.358

4.2.7 CH Index

The Calinski-Harabasz pseudo F-statistic index (CHI) measures the ratio of the between-group dispersion matrix to the within-group dispersion matrix, each normalized by their respective degrees of freedom[17]. The CHI is called the pseudo F-statistic because the dfs are similar to the ANOVA F-statistic. A maximal CHI indicates the optimal number of clusters, yet here we use it to compare algorithms.

An advantage of the CHI over other indices is that it does not depend on a distance metric (e.g. Euclidean) in its formulation. Algorithms that use Euclidean distances would be more favoured if that were not the case.

Algorithms	CHI
HC (Diana)	80.42
NMF (Div)	80.36
NMF (Euc)	79.26
KM (Spear)	78.71
PAM (Spear)	77.19
PAM (Euc)	75.71
KM (Euc)	75.38
PAM (MI)	13.23
KM (MI)	7.322
HC (A. Euc)	5.452
HC (S. Euc)	2.342

4.2.8 Connectivity

Connectivity measures how connected the clusters are based on its nearest neighbours[18]. The measure ranges from 0 to infinity and a small value indicates high connectivity.

Algorithms	Conn
HC (S. Euc)	8.93
HC (A. Euc)	21.77
HC (Diana)	252.6
NMF (Euc)	260.8
NMF (Div)	263.8
KM (Spear)	270.7
PAM (Spear)	287.1
PAM (Euc)	297.3
KM (Euc)	329.1
KM (MI)	707.3
PAM (MI)	722.1

4.2.9 Summary

Here is a summary of all the indices for each algorithm, in unsorted order.

Algorithms	PAC	DBI	DI	RS	CI	GI	CHI	Conn
HC (A. Euc)	0.5232	1.309	0.3696	0.1575	0.4795	0.7232	5.452	21.77
HC (Diana)	0.5558	1.719	0.2816	0.1124	0.321	1.639	80.42	252.6
HC (S. Euc)	0.05077	0.7637	0.4364	0.1781	0.4535	0.7621	2.342	8.93
KM (Euc)	0.4398	1.741	0.2408	0.08707	0.2816	1.676	75.38	329.1
KM (MI)	0.7167	1.972	0.2135	-0.007504	0.3529	-3.142	7.322	707.3

Algorithms	PAC	DBI	DI	RS	CI	GI	CHI	Conn
KM (Spear)	0.425	1.725	0.3181	0.1059	0.356	1.627	78.71	270.7
NMF (Div)	0.3267	1.702	0.2766	0.1111	0.3023	1.617	80.36	263.8
NMF (Euc)	0.1406	1.702	0.2888	0.1084	0.3369	1.666	79.26	260.8
PAM (Euc)	0.5984	1.725	0.286	0.1171	0.31	1.754	75.71	297.3
PAM (MI)	0.9638	1.935	0.2135	-0.0061	0.3376	-3.358	13.23	722.1
PAM (Spear)	0.4748	1.72	0.3144	0.1069	0.3489	1.715	77.19	287.1

4.3 Ranked Indices

4.3.1 All Indices

The table below shows the ranking of algorithms for performance on a clustering index, for each index. There is an additional column that shows the proportion of indices where an algorithm was ranked **first or second**.

Algorithms	PAC	DBI	DI	RS	CI	GI	CHI	Conn	Top
HC (S. Euc)	1	1	1	1	10	8	11	1	62.5%
HC (A. Euc)	7	2	2	2	11	9	10	2	50%
NMF (Div)	3	4	8	5	2	7	2	5	25%
HC (Diana)	8	5	7	4	4	5	1	3	12.5%
KM (Euc)	5	9	9	9	1	3	7	9	12.5%
NMF (Euc)	2	3	5	6	5	4	3	4	12.5%
PAM (Euc)	9	7	6	3	3	1	6	8	12.5%
PAM (Spear)	6	6	4	7	7	2	5	7	12.5%
KM (MI)	10	11	10.5	11	8	10	9	10	0%
KM (Spear)	4	8	3	8	9	6	4	6	0%
PAM (MI)	11	10	10.5	10	6	11	8	11	0%

If we were to conduct a weighted meta-consensus clustering, a weight for each algorithm needs to be assigned. One such way of doing so is using the inverse rank sums based on the indices. We can sum the ranks for each algorithm, and assign higher weight for consistently higher ranked methods (1st or 2nd). This is shown in the table below:

Algorithms	Top	Sum	Weight
NMF (Euc)	12.5%	32	12.54%
HC (S. Euc)	62.5%	34	11.8%
NMF (Div)	25%	36	11.15%
HC (Diana)	12.5%	37	10.84%
PAM (Euc)	12.5%	43	9.33%
PAM (Spear)	12.5%	44	9.12%
HC (A. Euc)	50%	45	8.92%
KM (Spear)	0%	48	8.36%
KM (Euc)	12.5%	52	7.72%
PAM (MI)	0%	77.5	5.18%
KM (MI)	0%	79.5	5.05%

4.3.2 PAC and CHI only

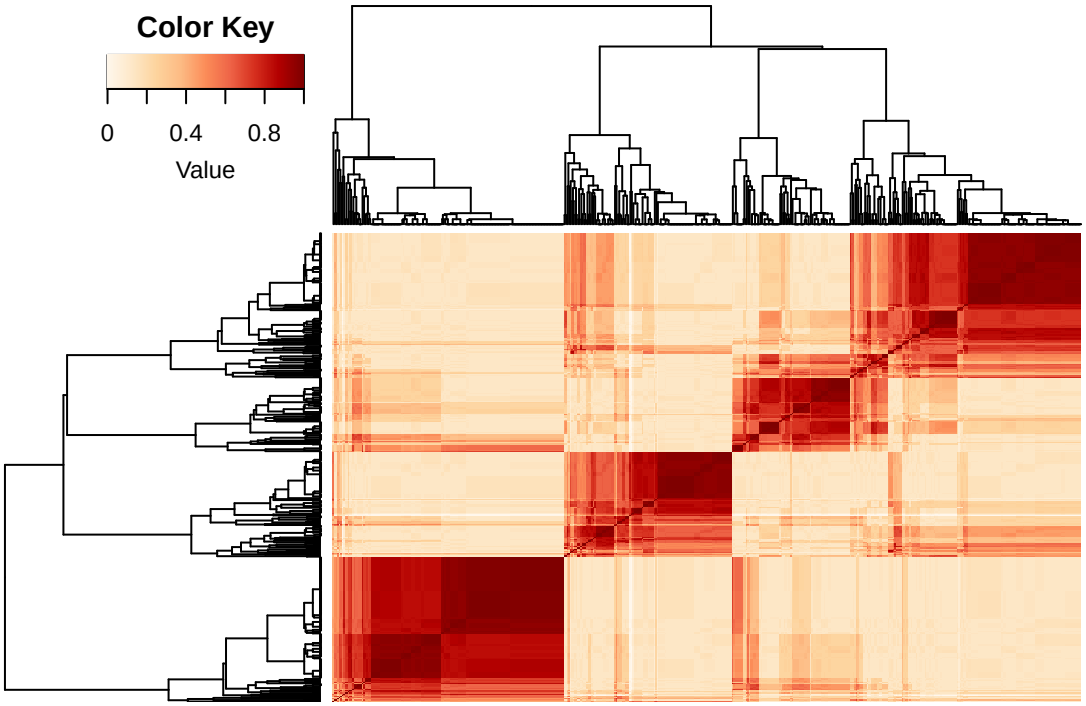
There is a major issue with the way we ranked the indices in the previous section. Upon closer inspection, the algorithm that uses hierarchical clustering with single linkage and Euclidean distance actually performs terrible in the clustering analysis. The table below shows the number of samples clustered into each of the four clusters:

1	2	3	4
486	1	1	1

And yet, the algorithm performs very well for several indices. One suggestion is to consider only PAC and CHI, because they do not depend on a specific distance metric. Also, the weights will be based on the index scores themselves instead of relative ranks. One reason is that since we only have two indices, it doesn't make sense to use ranks and lose information. Secondly, we want to be more precise in how much weight to assign.

Algorithms	PAC	CHI	Weight
NMF (Euc)	0.1406	79.26	0.1432
NMF (Div)	0.3267	80.36	0.128
KM (Spear)	0.425	78.71	0.1181
KM (Euc)	0.4398	75.38	0.1139
PAM (Spear)	0.4748	77.19	0.1125
HC (Diana)	0.5558	80.42	0.1083
PAM (Euc)	0.5984	75.71	0.1005
HC (S. Euc)	0.05077	2.342	0.08408
HC (A. Euc)	0.5232	5.452	0.04595
KM (MI)	0.7167	7.322	0.03085
PAM (MI)	0.9638	13.23	0.01463

Below we show the weighted meta-consensus matrix using the weights from the previous table:



And the corresponding confusion matrix with TCGA's clusters:

	TCGA	C1	C2	C4	C5
Meta					
C1		106	14	23	8
C2		2	64	7	4
C4		0	29	95	27
C5		1	0	10	99

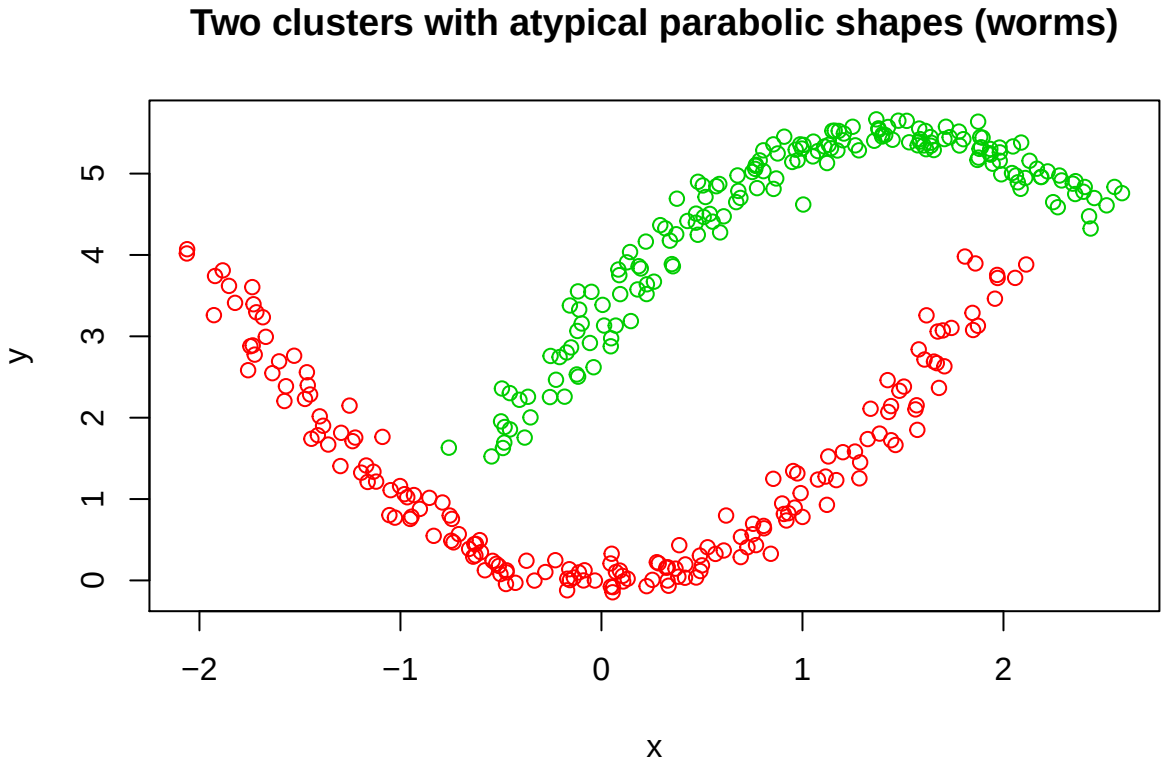
	Sensitivity	Specificity	Pos Pred Value	Neg Pred Value	Prevalence	Detection Rate	Detection Prevalence	Balanced Accuracy
Class: C1	0.972	0.882	0.702	0.991	0.223	0.217	0.309	0.927
Class: C2	0.598	0.966	0.831	0.896	0.219	0.131	0.157	0.782
Class: C4	0.704	0.842	0.629	0.882	0.276	0.194	0.309	0.773
Class: C5	0.717	0.969	0.9	0.897	0.282	0.202	0.225	0.843

5 Simulations

To confirm the clustering results from using the TCGA dataset, we need to try out the algorithms on a few simulated datasets, designed to test the robustness of each method. The `clusterSim` package provides very good built-in examples to use.

5.1 Worms Dataset

The following two-cluster dataset is our first simulation:



The ranked indices and weights tables are shown below:

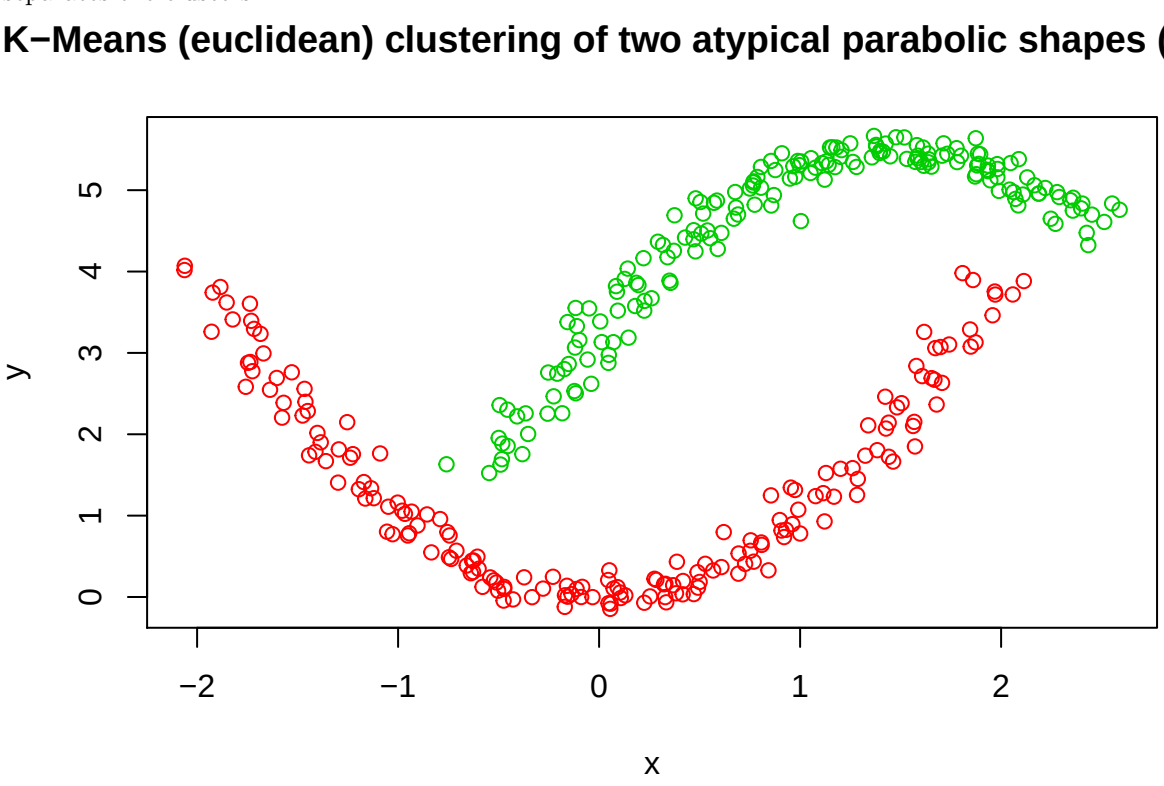
Algorithms	PAC	DBI	DI	RS	CI	GI	CHI	Conn	Top
KM (Euc)	8	1	4	1	1	1	1	5	62.5%
HC (Diana)	7	4	2	3	2	2	3	4	37.5%
NMF (Div)	6	2	10	2	3	3	2	11	37.5%
HC (S. Euc)	3	8	1	7	7	7	7	1	25%
KM (Spear)	1	10	8	8	8	8	8	7	12.5%
PAM (Spear)	1	10	8	8	8	8	8	7	12.5%
PAM (MI)	4	7	6	10	11	10	11	2	12.5%
HC (A. Euc)	11	6	3	6	6	6	6	3	0%
KM (MI)	9	9	7	11	10	11	10	6	0%
PAM (Euc)	10	3	5	4	5	5	4	9	0%
NMF (Euc)	5	5	10	5	4	4	5	10	0%

Algorithms	Top	Sum	Weight
KM (Euc)	62.5%	22	17.39%
HC (Diana)	37.5%	27	14.17%
NMF (Div)	37.5%	39	9.81%
HC (S. Euc)	25%	41	9.33%
PAM (Euc)	0%	45	8.5%
HC (A. Euc)	0%	47	8.14%
NMF (Euc)	0%	48	7.97%
KM (Spear)	12.5%	58	6.6%
PAM (Spear)	12.5%	58	6.6%
PAM (MI)	12.5%	61	6.27%
KM (MI)	0%	73	5.24%

Using hierarchical clustering with Wald's method on the meta-consensus matrix across the ten algorithms, we can obtain meta-consensus classes. The following table shows how the different methods compare, based on the number of matches to the true class labels.

	Matches
HC (S. Euc)	360
KM (Euc)	294
HC (Diana)	291
NMF (Div)	291
Meta	289
NMF (Euc)	287
PAM (Euc)	285
HC (A. Euc)	277
KM (MI)	182
PAM (MI)	181
KM (Spear)	91
PAM (Spear)	91

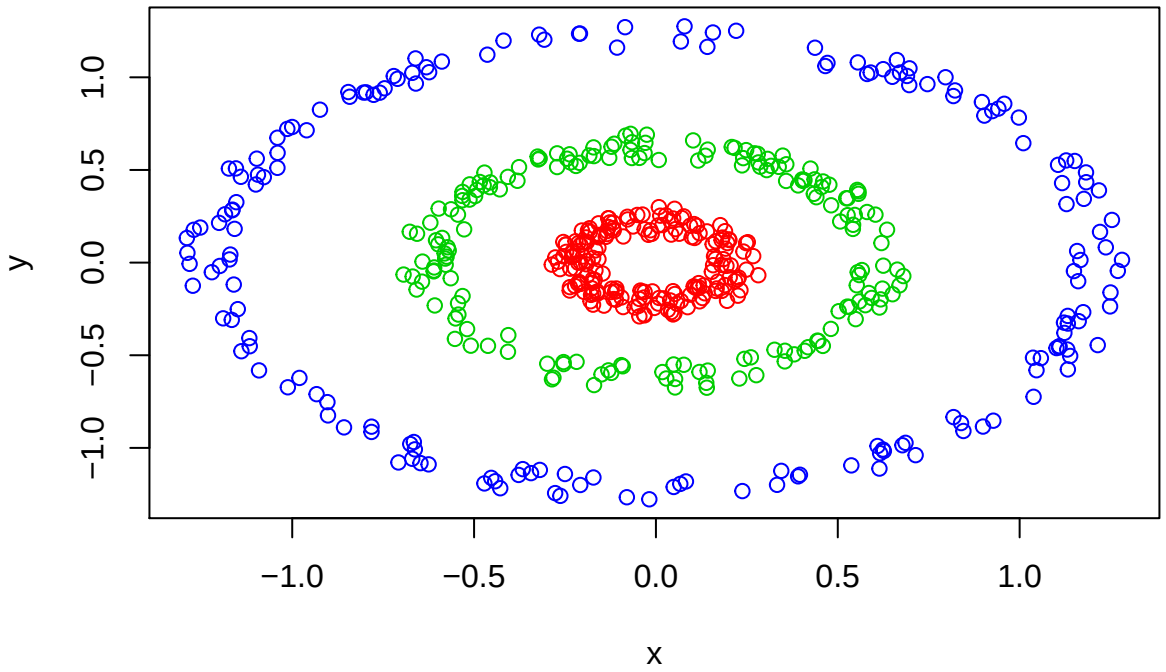
Let's visualize how the best algorithm, hierarchical clustering using single linkage and Euclidean distance, separates the clusters:



It appears that the best algorithm can separate the clusters, yet the indices do not favour such a partition and it appears that the algorithm is performing poorly.

5.2 Rings Dataset

The following three-cluster dataset is our second simulation:

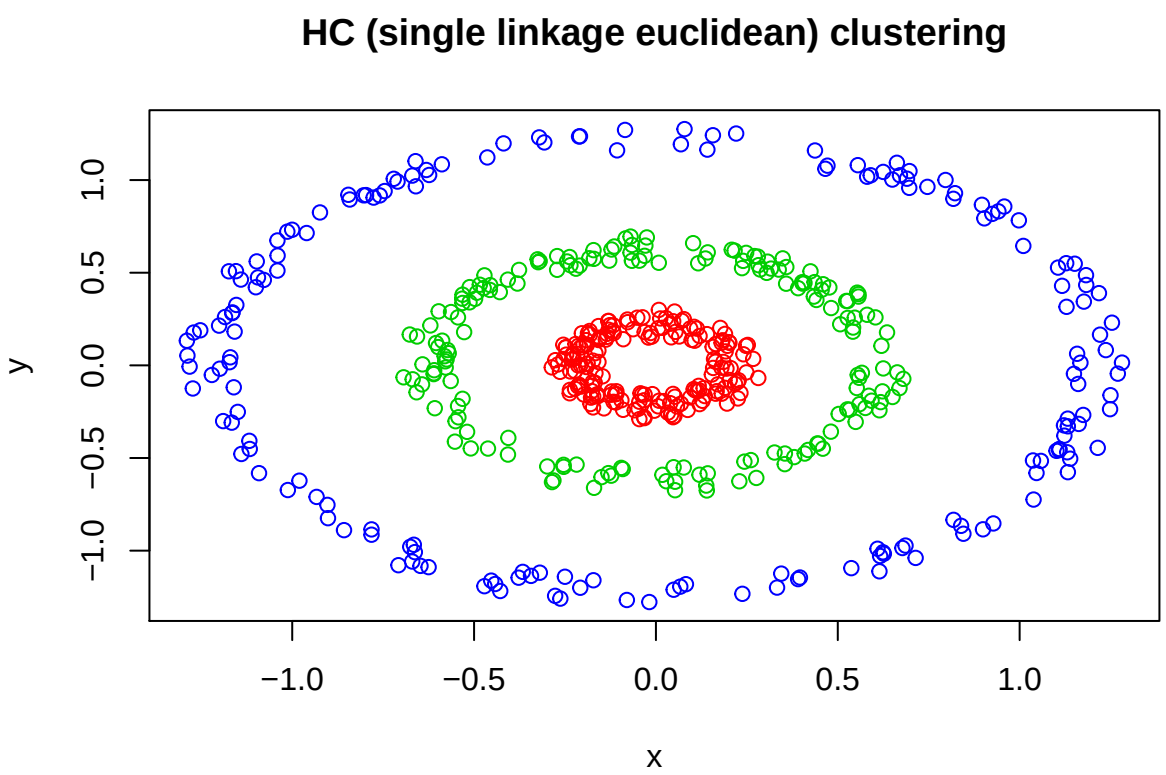


Algorithms	PAC	DBI	DI	RS	CI	GI	CHI	Conn	Top
KM (Euc)	7	7	2	1	1	1	6	2	62.5%
HC (A. Euc)	6	1	3	2	8	9	5	3	25%
HC (S. Euc)	3	8	1	9	9	10	9	1	25%
PAM (MI)	1	2	9	7	5	5	7	9	25%
NMF (Div)	2	4	8	6	7	8	2	6	25%
HC (Diana)	8	5	4	3	2	3	4	5	12.5%
KM (MI)	10	10	7	10	10	2	10	4	12.5%
NMF (Euc)	5	3	5	5	4	7	1	8	12.5%
KM (Spear)	4	9	9	8	6	6	8	10	0%
PAM (Euc)	9	6	6	4	3	4	3	7	0%

Algorithms	Top	Sum	Weight
KM (Euc)	62.5%	27	15.33%
HC (Diana)	12.5%	34	12.17%
HC (A. Euc)	25%	37	11.19%
NMF (Euc)	12.5%	38	10.89%
PAM (Euc)	0%	42	9.85%
NMF (Div)	25%	43	9.62%
PAM (MI)	25%	45	9.2%
HC (S. Euc)	25%	50	8.28%
KM (Spear)	0%	60	6.9%
KM (MI)	12.5%	63	6.57%

	Matches
HC (S. Euc)	540

	Matches
HC (Diana)	323
PAM (Euc)	288
KM (Euc)	270
Meta	240
HC (A. Euc)	211
KM (Spear)	180
NMF (Euc)	180
KM (MI)	176
PAM (MI)	170
NMF (Div)	161



Again, the hierarchical clustering using single linkage and euclidean distance perfectly separates the feature space into the three rings.

5.3 Seeds Dataset

6 References

1. Brunet, J. P., Tamayo, P., Golub, T. R., & Mesirov, J. P. (2004). "Metagenes and molecular pattern discovery using matrix factorization." *Proceedings of the national academy of sciences*, 101 (12), 4164-4169.

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

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

4. Conrad, J. G., Al-Kofahi, K., Zhao, Y., & Karypis, G. (2005, June). "Effective document clustering for large heterogeneous law firm collections." In *Proceedings of the 10th international conference on Artificial intelligence and law* (pp. 177-187). ACM.

5. Cohen, J. (1960). "A coefficient of agreement for nominal scales." *Educational and psychological measurement*, 20 (1), 37-46.

6. Cohen, J. (1968). "Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit." *Psychological bulletin*, 70 (4), 213.

7. Fleiss, J.L. (1981). *Statistical methods for rates and proportions* (2nd ed.). New York: John Wiley.

8. Rand, W. M. (1971). "Objective criteria for the evaluation of clustering methods." *Journal of the American Statistical Association*, 66 (336), 846-850.

9. Hubert, L., & Arabie, P. (1985). "Comparing partitions." *Journal of Classification*, 2 (1), 193-218.

10. Vinh, N. X., Epps, J., & Bailey, J. (2009, June). "Information theoretic measures for clusterings comparison: is a correction for chance necessary?." *In Proceedings of the 26th Annual International Conference on Machine Learning* (pp. 1073-1080). ACM.

11. Senbabaoglu, Y., Michailidis, G., & Li, J. Z. (2014). "Critical limitations of consensus clustering in class discovery." *Scientific reports*, 4.

12. Davies, David L.; Bouldin, Donald W. (1979). "A Cluster Separation Measure." *IEEE Transactions on Pattern Analysis and Machine Intelligence.* PAMI-1 (2): 224–227.

13. Dunn, J. C. (1973). "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters." *Journal of Cybernetics* 3 (3): 32–57.

14. Rousseeuw, P. J. (1987). "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis." *Journal of Computational and Applied Mathematics*, 20, 53-65.

15. Hubert, L. J., & Levin, J. R. (1976). "A general statistical framework for assessing categorical clustering in free recall." *Psychological bulletin*, 83 (6), 1072-1080.

16. Baker, F. B., & Hubert, L. J. (1975). "Measuring the power of hierarchical cluster analysis." *Journal of the American Statistical Association*, 70 (349), 31-38.

17. Calinski, T., & Harabasz, J. (1974). "A dendrite method for cluster analysis." *Communications in Statistics-theory and Methods*, 3 (1), 1-27.

18. Handl, J., Knowles, J., & Kell, D. B. (2005). "Computational cluster validation in post-genomic data analysis." *Bioinformatics*, 21 (15), 3201-3212.