The IntClust Package: Vignette

Marijke Van Moerbeke

July 30, 2018

Contents

1 Introduction

All multi-source clustering methods discussed in Van Moerbeke et al., 2018a and Van Moerbeke et al., 2018b are implemented in the `IntClust` R package. In addition, linear models for microarrays (limma; Smyth, 2004) for the detection of the differential gene expression and functional class scoring (MLP; Raghavan et al., 2012) for pathway analysis are included. Both methods are relevant if one of the high dimensional data sets contains information on gene expression data. This vignette provides a short overview of the capacity of the `IntClust` package for data analysis and visualisation.

2 Methods

The data structure we consider in the examples consist of L different data sets $\mathbf{D}_1, \ldots, \mathbf{D}_L$ of size $n \times m_\ell$ were n is the number of rows and m_ℓ the number of columns in the ℓth data set. Note that we assume that the row dimension is the same in all data matrices. In case that the columns are the common dimension, the data matrices can be transposed. The aim of the analysis is to find robust clusters of rows across all data sources.

The multi-source clustering procedures that are included in the `IntClust` package are presented in Table 1.

Table 1: List of the multi-source clustering methods implemented in the `IntClust` package. The methods are introduced in Van Moerbeke et al., 2018a.

Category	**Method**	**R function**	**Reference**
Direct	ADC	`ADC()`	Fodeh et al. (2013)
Clustering	ADECa	`ADECa()`	Fodeh et al. (2013)
	ADECb	`ADECb()`	Fodeh et al. (2013)
	ADECc	`ADECc()`	Fodeh et al. (2013)
Similarity-based	Weighted	`WeightedClust()`	Perualila-Tan et al. (2016)
approaches	SNF	`SNF()`	Wang et al. (2014)
Graph-based	CSPA	`EnsembleClustering()`	Strehl and Gosh (2002)
approaches	HGPA	`EnsembleClustering()`	Strehl and Gosh (2002)
	MCLA	`EnsembleClustering()`	Strehl and Gosh (2002)
	HBGF	`HBGF()`	Fern and Brodley (2004)
	Balls	`ClusteringAggregation()`	Gionis et al. (2007)
	Aggl.	`ClusteringAggregation()`	Gionis et al. (2007)
	Furthest	`ClusteringAggregation()`	Gionis et al. (2007)
Voting-based	CVAA	`CVAA()`	Saeed et al. (2012)
consensus	W-CVAA	`CVAA()`	Saeed et al. (2014)
approaches	IVC	`ConsensusClustering()`	Nguyen and Caruana (2007)
	IPVC	`ConsensusClustering()`	Nguyen and Caruana (2007)
	IPC	`ConsensusClustering()`	Nguyen and Caruana (2007)
	EA	`EvidenceAccumulation()`	Fred and Jain (2002)
	M-ABC	`M_ABC()`	Amaratunga et al. (2008)
	CTS	`LinkBasedClustering()`	Iam-on and Garrett (2010)
	SRS	`LinkBasedClustering()`	Iam-on and Garrett (2010)
	ASRS	`LinkBasedClustering()`	Iam-on and Garrett (2010)
	CECa	`CECa()`	Fodeh et al. (2013)
	CECb	`CECb()`	Fodeh et al. (2013)
	CECc	`CECc()`	Fodeh et al. (2013)
Hierarchy-based	EHC	`EHC()`	Hossain et al. (2012)
approaches	HEC	`HEC()`	Zheng et al. (2014)

As pointed out in in Van Moerbeke et al., 2018a, the methods either integrate the data sets into a combined data matrix or calculate a distance matrix based on all sources provided as input. Once the integration step is completed, hierarchical clustering with the Ward link is performed.

The resulting clusters of the multi-source methods consist of objects that are expressing similarity in each of the provided data sets. Interest could, for example, be in the clusters that remain stable across methods. This stability indicates a fairly robust (sub)cluster of objects based on multiple sources of data. For the analysis presented in this chapter, the objects represent compounds.

Although the focus is on clustering multiple data sources simultaneously, it is important to in-

vestigate the clustering results of the individual data sources as well. This reveals whether or not the single data sources already show a high degree of resemblance in the formed clusters. Further, if the multi-source clustering procedures are executed, the influence of each data source can be investigated. If a cluster of interest has been chosen, a secondary analysis can be conducted.

3 Application of `IntClust`

We illustrate several functions of the `IntClust` package using the data of the MCF7 cell line. The data sets consist of a 56×350 fingerprint features matrix and a 56×477 target prediction matrix. Both data matrices are binary with rows representing compounds and are included in the `IntClust` package. In addition, a 2434×56 gene expression data matrix is available as well. The package can be installed using the following code.

```
> install.packages("IntClust")
> library(IntClust)
> data(fingerprintMat)
> data(targetMat)
```

3.1 Single source clustering

Depending on the type of the data matrix, several distance measures can be used: `Euclidean` for continuous data and `jaccard` or `tanimoto` for binary data. The argument `clust="agnes"` implies that the implemented method for clustering is agglomerative hierarchical clustering (Hastie et al., 2009). Data normalization can be performed using the argument `normalize=TRUE`. The implemented normalizing methods are: Quantile-Normalization, Fisher-Yates Normalization, standardization and range normalization. For the MCF7 data, normalization is not necessary since both data sets are binary. The complete code for the single source clustering is given below.

```
> MCF7_F <- Cluster(Data=fingerprintMat,type="data",distmeasure="tanimoto",
+           normalize=FALSE,method=NULL,clust="agnes",linkage="flexible",gap=FALSE)
> MCF7_T <- Cluster(Data=targetMat,type="data",distmeasure="tanimoto",
+           normalize=FALSE,method=NULL,clust="agnes",linkage="flexible",gap=FALSE)
```

Two options are available to select the number of clusters. The argument `gap=TRUE` uses the gap statistic (Hastie et al., 2009) for the selection of the number of clusters. A second option to determine the number of clusters is implemented in the function `SelectnrClusters()`. In this case medoid clustering is performed (Struyf et al., 1997) for a sequence of numbers of clusters for each provided data source. The number corresponding with the maximal average silhouette widths over the data sources can be taken as an optimal number of clusters.

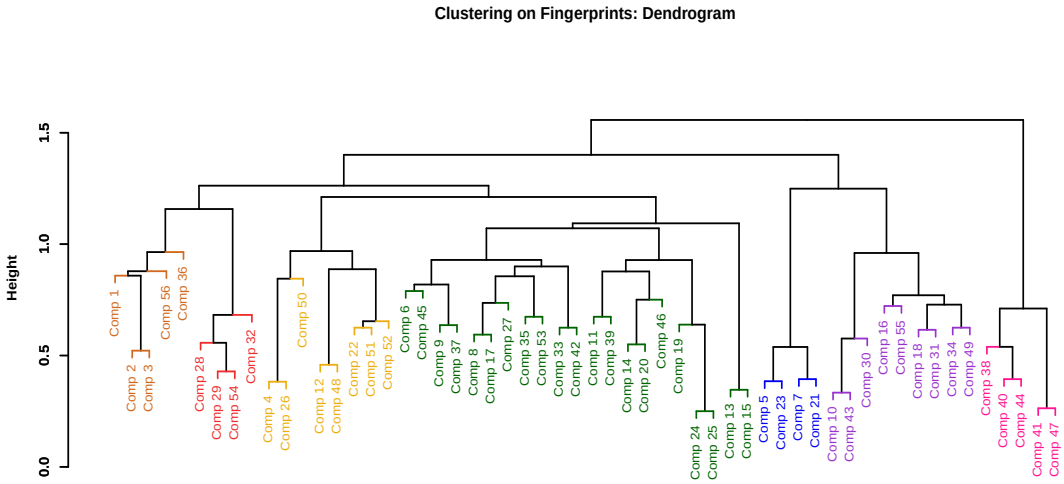
```
> List=list(fingerprintMat,targetMat)
> NrClusters=SelectnrClusters(List=List,type="data",distmeasure=c("tanimoto",
+            "tanimoto"),nrclusters=seq(5,20),normalize=c(FALSE,FALSE),
+             names=c("FP","TP"))
```

In the specific case of these data sources, the average silhouette width will only increase as the number of clusters increases. Therefore, we will rely on the gap statistic and conclude on seven clusters. A dendrogram with a different colour per cluster can be used for visualization in the following way.

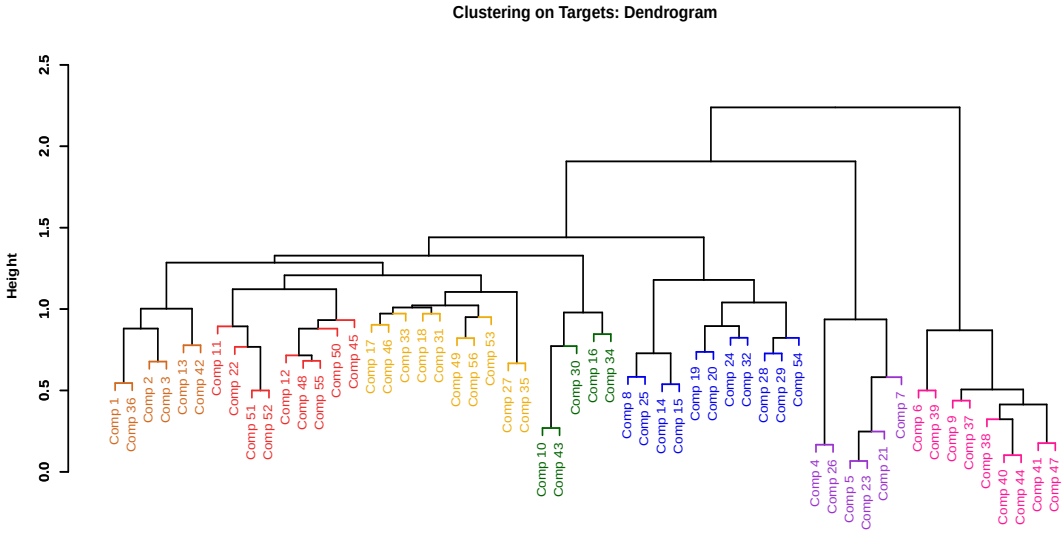
```
> Colours <- ColorPalette(colors=c("chocolate","firebrick2","darkgoldenrod2",
+            "darkgreen","blue2","darkorchid3","deeppink"),ncols=7)
```

The R object `Colours` contains the colour patterns and the function `ClusterPlot()` produces the dendrograms (with seven clusters based on the gap statistic) in Figure 1a and 1b.

```
> ClusterPlot(Data1=MCF7_F,nrclusters=7,cols=Colours,main="Clustering on
+             Fingerprints: Dendrogram",ylim=c(-0.1,1.8))
> ClusterPlot(Data1=MCF7_T,nrclusters=7,cols=Colours,colorComps=NULL,main="Clustering on
+              Targets: Dendrogram",ylim=c(-0.1,2.5))
```



(a) Fingerprint clustering.



(b) Target prediction clustering.

Figure 1: Dendrograms of the individual data clustering results. Panel a: The fingerprint clustering. Panel b: The target prediction clustering.

We can for example investigate the purple cluster shown in Figure 1a. The cluster does not undergo a lot of changes under the influence of the target predictions although the group is split across the blue and yellow clusters.

3.2 Multi-source clustering

Several multi-source clustering procedures, listed in Table 1, have been implemented in the `IntClust` package. We illustrate the Aggregated Data Clustering (ADC), Weighted clustering and Weighting on Membership Clustering (WonM) methods. Multi-source clustering using other methods can be conducted easily using the appropriate function.

3.2.1 Aggregated data clustering (`ADC()`)

Aggregated data clustering can only be applied if all data sources are of the same type. The first step fuses all data matrices into one larger matrix such that only one data matrix remains. Next, clustering is performed on this single matrix.

```
> L=list(fingerprintMat,targetMat)
> MCF7_ADC=ADC(List=L,distmeasure="tanimoto",normalize=FALSE,clust="agnes",
+          linkage="flexible")
```

3.2.2 Weighted clustering (`WeightedClust()`)

The weighted clustering computes a single distance matrix using all data sources. For each data matrix, a distance matrix $\widetilde{\mathbf{DM}}_\ell$ is calculated. The distance matrices are combined in a weighted linear combination $\mathbf{DM}_w$on which clustering is performed. The option `weight=seq(0,1,0.1)` implies that in our setting of two data sets

$$\mathbf{DM}_w = w_1 \cdot \widetilde{\mathbf{DM}}_1 + (1 - w_1) \cdot \widetilde{\mathbf{DM}}_2,$$

for a sequence of weights $w_1 = (0, 0.1, 0.2, \ldots, 0.9, 1)$.

```
> L=list(fingerprintMat,targetMat)
> MCF7_Weighted=WeightedClust(L,type="data",distmeasure=c("tanimoto","tanimoto"),
+               normalize=c(FALSE,FALSE),weight=seq(0,1,0.1),weightclust=0.5,
+               StopRange=FALSE)
```

3.2.3 Weighting on membership clustering (`WonM()`)

Weighting on membership performs hierarchical clustering on each data source separately. The resulting dendrograms are cut, multiple times, into into clusters for a range of numbers of clusters k. Each time, a binary incidence matrix is set up. A value of zero indicates that a pair of objects resides in the same cluster to ensure distances. All incidence matrices are summed over the values of k per data source and the different data sources. On the resulting consensus matrix, hierarchical clustering is performed once again to obtain the final clustering taking into account all information of the data sources.

```
> L=list(fingerprintMat,targetMat)
> MCF7_WonM=WonM(List=L,type="data",distmeasure=c("tanimoto","tanimoto"),
+           normalize=c(FALSE,FALSE),nrclusters=seq(5,25),linkage=
+           c("flexible","flexible"))
```

3.3 Comparison of results

The clusters of the multi-source methods consist of objects that are expected to be similar in each of the individual data sources. Clusters that remain stable over the applied the methods are of interest. If a cluster does not undergo too many changes and is found multiple times, the objects show a similarity with a high confidence. Further, it can be hypothesized that the used data sources are related for the selected clusters. This can provide more insight into the MOA of compounds in drug disovery experiments. One way to visualize the clustering solutions of the multi-source methods is to follow the changes in the solutions obtained for different clustering methods. For the above example, the function `ComparePlot()` was used to produce Figure 2 which presents a comparison across all executed clustering procedures. We notice that the blue, green and pink cluster remain stable over all other methods. Under the influence of the target predictions one compound disappears and is replaced by another. For the weighted procedures, the results for all weights are shown.

Different methods cluster the compounds in a different order and this results in non-corresponding cluster numbers. Therefore, one method is used as a reference (the clustering based on the fingerprints features for the example presented in Figure 2) and the cluster numbers obtained for the other methods are rearranged according to the reference solution. The re-appointing of the cluster numbers is based on finding the cluster that relatively has the most in common with one of the reference clusters and taking over this number. In the `IntClust` package, this is done using the function `MatrixFunction()` in which the rearranging algorithm is partly based on the Gale-Shapley algorithm (Kleinberg and Tardos, 2005). It creates a matrix of which the columns are the compounds in the order of clustering by reference method. The rows are the different methods and the values of the cells are the rearranged cluster numbers which are given different colours in the resulting figure.

```
> L=list(MCF7_F,MCF7_ADC,MCF7_WonM,MCF7_Weighted,MCF7_T)
> N=c("FP","ADC","WonM",paste("Weight",seq(1,0,-0.1),sep=" "),"TP")
> ComparePlot(L,nrclusters=7,cols=Colours,fusionsLog=TRUE,weightclust=FALSE,names=N,
+ margins=c(9.1,4.1,4.1,4.1),plottype="new",location=NULL)
```

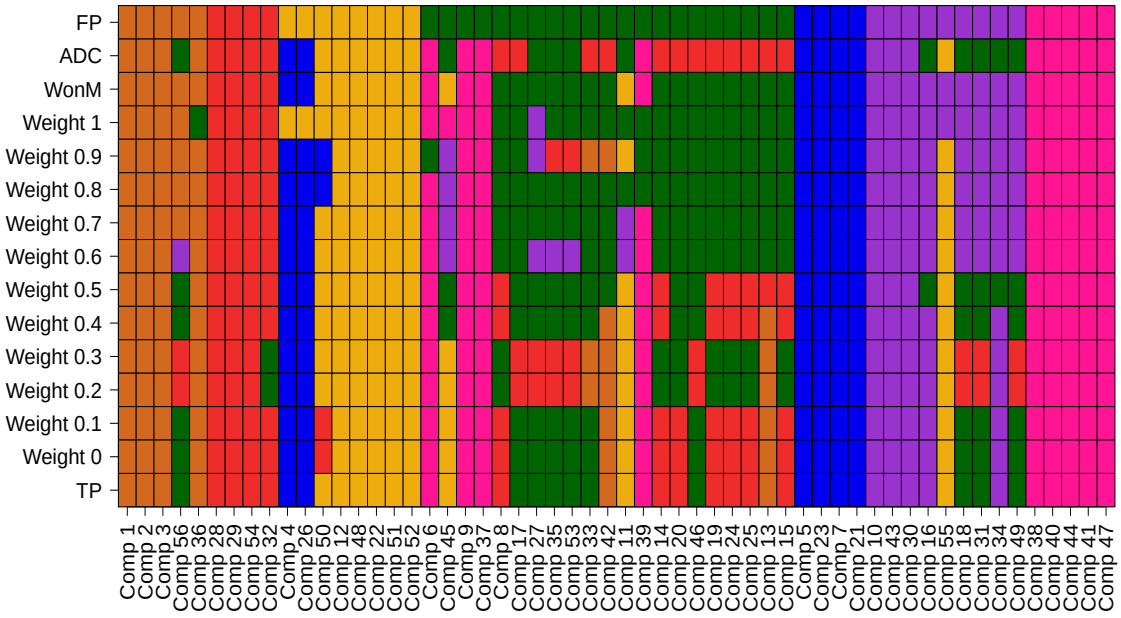


Figure 2: Visualization of the multi-source clustering results of the fingerprint and target prediction data. The first row represents the single source clustering of the fingerprint data and the last row represents the single source clustering of the target prediction data. The single source clustering result of the fingerprint data set is used as a reference clustering. The results of the remaining methods are coloured to this reference.

If a weighted clustering was performed, the `ComparePlott()` function allows us to follow the

membership changes in a chosen cluster with respect to the changing weight. The function `FindCluster()` can be used in order to find the compounds of a cluster.

```
> Comps=FindCluster(List=L,nrclusters=7,select=c(1,6))
> Comps
```

Once the compound subset is found (the R object `Comps` in our example) the function `TrackCluster()` can be used to produce Figure 3 to track the changes in the cluster with respect to the weights.

```
> Tracking=TrackCluster(List=L,Selection=Comps,nrclusters=7,followMaxComps=FALSE,
+                   followClust=TRUE,fusionsLog=TRUE,weightclust=FALSE,
+                   names=N,selectionPlot=FALSE,table=FALSE,legendposy=2.4,
+                   completeSelectionPlot=TRUE,cols=Colours,plottype="sweave",
+                   location=NULL)
```

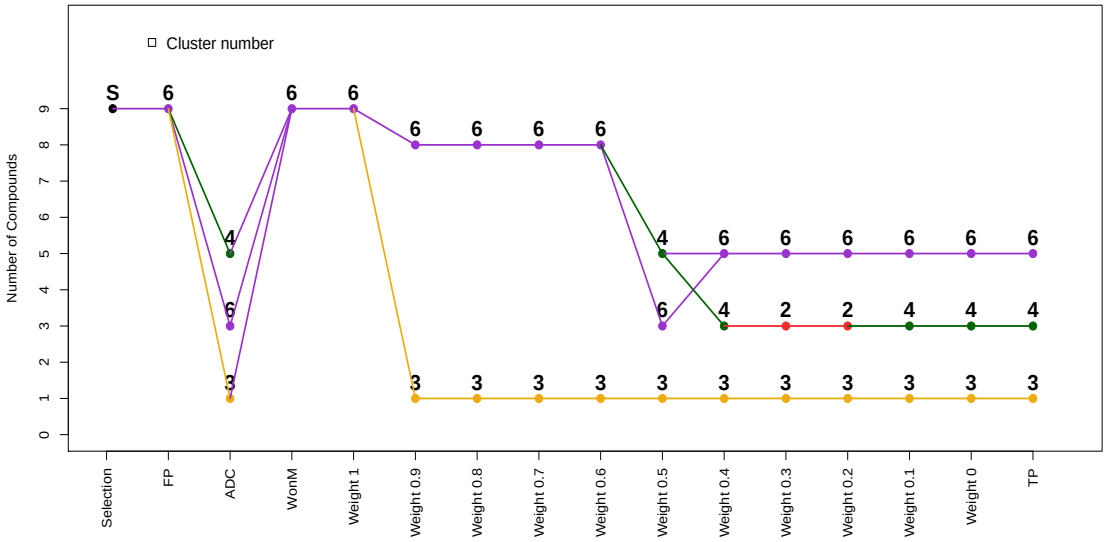


Figure 3: Tracking of the purple cluster across the multi-source clustering methods.

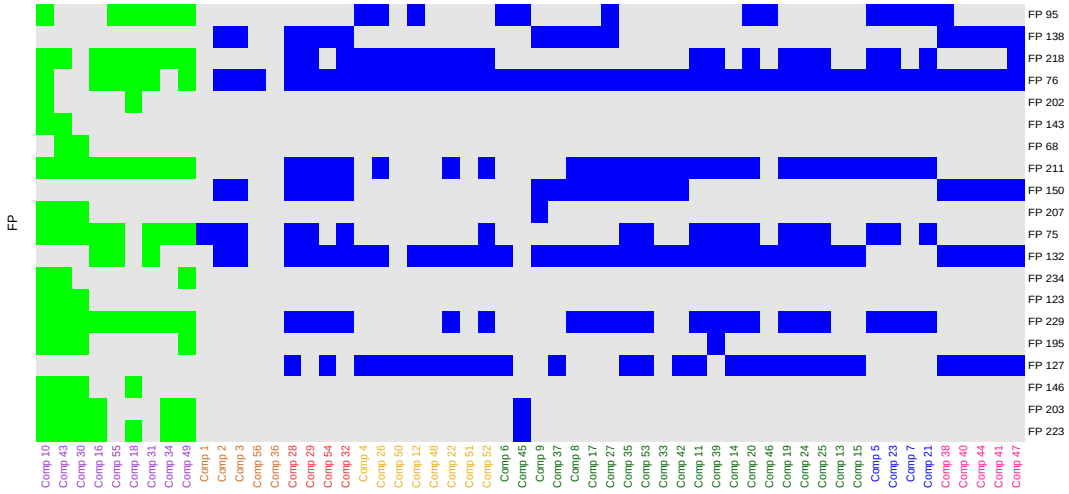
3.4 Characteristic features

In the next stage of the analysis, we can investigate whether there are fingerprints features or target predictions that define a specific cluster using the function `ChooseCluster()`. The function performs a Fisher's exact test (Fisher, 1922) in order to discover discerning features. Note that the function has the option to provide an interactive input as well.

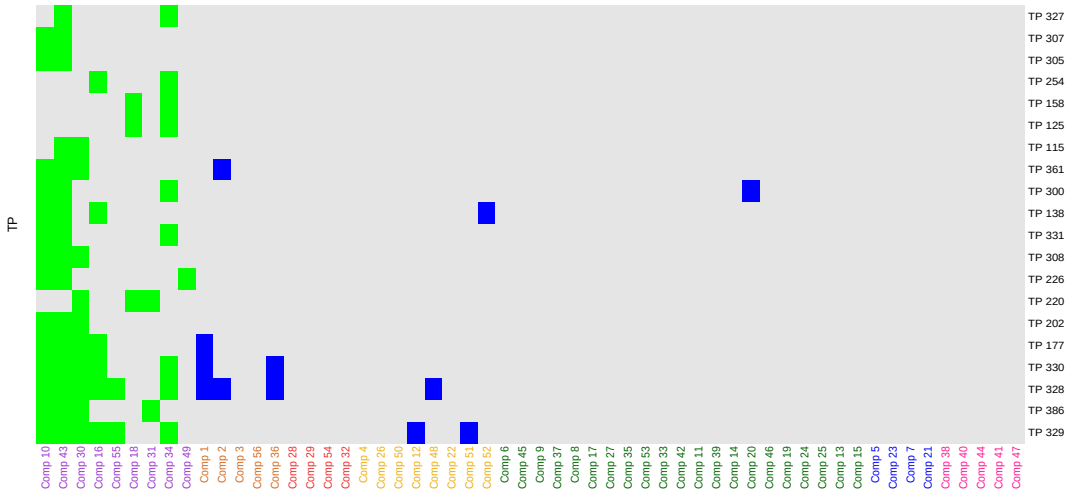
```
> MCF7_Feat=ChooseCluster(Interactive=FALSE,leadCpds=list(Comps),clusterResult=MCF7_F,
+           colorLab=MCF7_F,binData=list(fingerprintMat,targetMat),datanames=
+           c("FP","TP"),topChar = 20,topG = 20)
```

The function `BinFeaturesPlot()` produces the image plot with the top identified features presented in Figure 4.

```
> BinFeaturesPlot_SingleData(leadCpds=Comps,orderLab=MCF7_F,features=MCF7_Feat
+                            $Characteristics$FP$TopFeat$Names,data=fingerprintMat,
+                            colorLab=MCF7_F,nrclusters=7,cols=Colours,name=c("FP"))
> BinFeaturesPlot_SingleData(leadCpds=Comps,orderLab=MCF7_F,features=MCF7_Feat
+                            $Characteristics$TP$TopFeat$Names,data=targetMat,
+                            colorLab=MCF7_F,nrclusters=7,cols=Colours,name=c("TP"))
```



(a) Top 20 fingerprints.



(b) Top 20 target predictions.

Figure 4: The top discriminating targets of the purple cluster by the fingerprint and target prediction data (identified by Fisher's exact test). Columns represent compounds and rows represent target predictions. A hit target is coloured green for the cluster of interest and blue for the other compounds. The labels on the left indicate the data sets while the feature names are indicated on the right. Panel a: Fingerprint features. Panel b: Target prediction features.

4 Exploring connections with external data sets

In addition to the fingerprint features and target prediction data matrices, the MCF7 data consists of a gene expression data matrix for the 56 compounds that was not included in the analysis up to this stage. In this section we explore how gene expression profiles change across the cluster solutions.

4.1 Differential gene expression

The `IntClust` package can be used to detect differentially expressed genes between a chosen cluster and the rest of the compounds using the limma method (Smyth, 2004). The p-values are adjusted to multiple testing using the BH-FDR method (Benjamini and Hochberg, 1995). The option `TopG=10` implies that the top 10 genes will be identified.

```
> data(geneMat)
> MCF7_Genes=DiffGenes(List=NULL,Selection=Comps,geneExpr=geneMat,method="limma",
+             sign=0.05,topG=10)
> Genes=MCF7_Genes$Selection$Genes$TopDE$ID
```

Genes profiles can be plotted with the function `ProfilePlot()`.

```
> ProfilePlot(Genes=Genes[1:5],Comps=Comps,geneExpr=geneMat,raw=FALSE,
+              order=MCF7_F,color=MCF7_F,nrclusters=7,cols=Colours,
+              addLegend=TRUE,margins=c(8.1,4.1,1.1,6.5),plottype="sweave",
+              location=NULL)
```

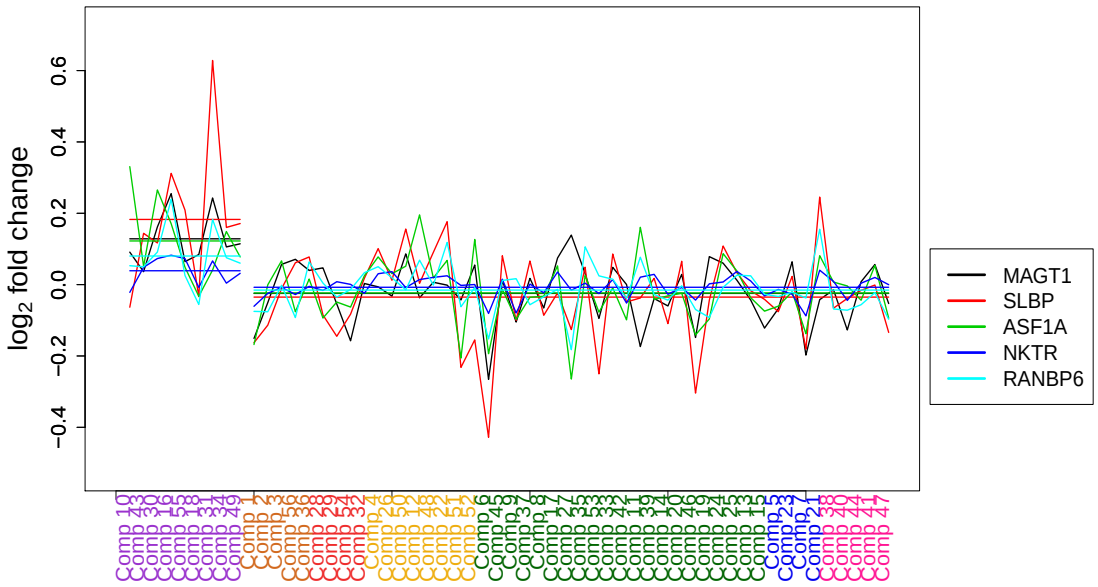


Figure 5: The top 5 genes of the purple cluster as identified by limma.

4.2 Pathway analysis

The final step in the analysis is to allocate the identified genes to a gene set or pathway. If a gene set is enriched the probability to observe significant genes of this gene set by chance is low for the selected cluster. The selected database for pathway analysis is the Gene Ontology (GO) database and the pathway analysis method MLP (Raghavan et al., 2012) is implemented in the `PathwaySelectionIter()` function. We can count how many of the pathways are shared over the different iterations with the `Genseset.intersectSelection()` function. A figure illustrating the discovered pathways, as shown for the example in Figure 6, can be made with the `PlotPathways()` function.

```
> data(GeneInfo)
> data(GS)
> L=list(MCF7_Genes)
> MCF7_Paths=PathwayAnalysis(List=L,Selection=Comps,geneExpr=geneMat,
+             method = c("limma","MLP"),geneInfo=GeneInfo,
+             geneSetSource="GOBP",topP = NULL,topG=NULL,GENESET=GS,
+             sign = 0.05,niter=2)
> PlotPathways(MCF7_Paths$Selection$Pathways)
```

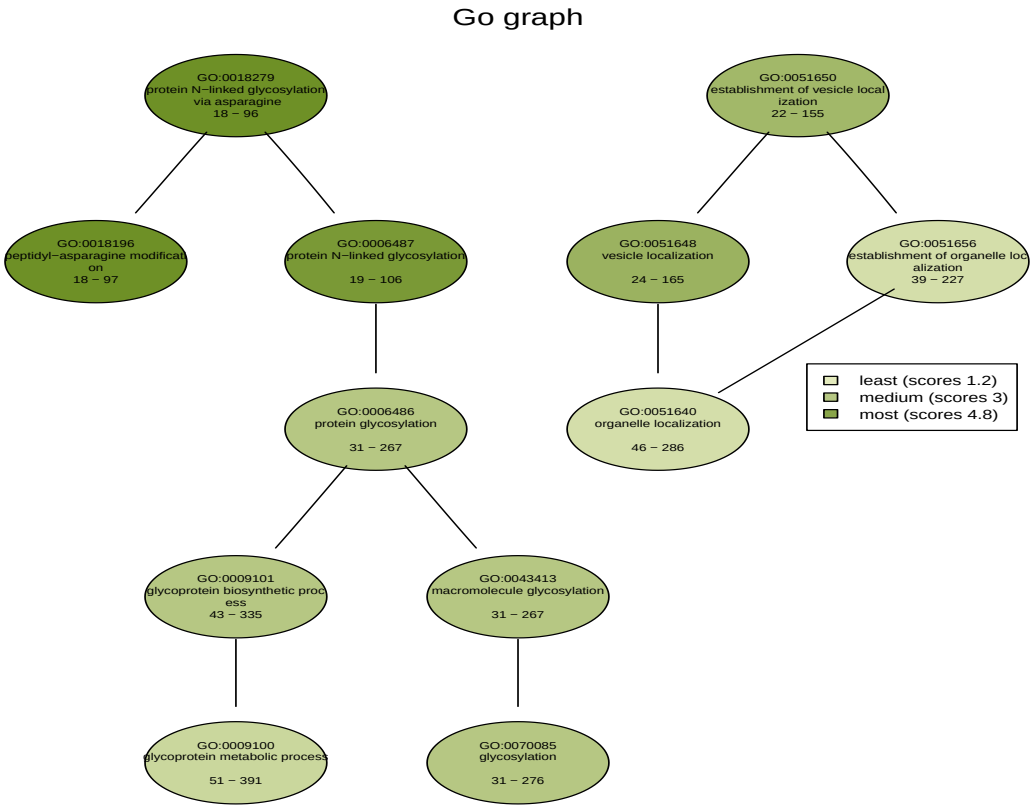


Figure 6: The top pathways annotated to the differentially expressed genes of the purple cluster as determined by the MLP analysis.

5 Software used

- R Under development (unstable) (2018-01-28 r74175), `x86_64-w64-mingw32`
- Locale: `LC_COLLATE=C`, `LC_CTYPE=Dutch_Belgium.1252`, `LC_MONETARY=Dutch_Belgium.1252`, `LC_NUMERIC=C`, `LC_TIME=Dutch_Belgium.1252`
- Running under: `Windows 7 x64 (build 7601) Service Pack 1`
- Matrix products: default
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Loaded via a namespace (and not attached): compiler 3.5.0, tools 3.5.0

References

Amaratunga, D., Cabrera, J., and Kovtun, V. (2008), "Microarray learning with ABC," *Biostatistics*, 9, 128–136.

Benjamini, Y. and Hochberg, Y. (1995), "Controlling the false discovery rate: a practical and powerful approach to multiple testing," *Journal of the Royal Statistical Society. Series B (Methodological),*, 57, 289–300.

Fern, X. Z. and Brodley, C. E. (2004), "Solving cluster ensemble problems by bipartite graph partitioning," in *Proceedings of the 21th International Conference on Machine Learning*.

Fisher, R. A. (1922), "On the Interpretation of X $\times$ 2 from Contingency Tables, and the Calculation of P," *Journal of the Royal Statistical Society*, 85, 87–94.

Fodeh, J., Brandt, C., Luong, B. T., Haddad, A., Schultz, M., Murphy, T., and Krauthammer, M. (2013), "Complementary Ensemble Clustering of Biomedical Data," *Journal of Biomedical Informatics*, 46, 436–443.

Fred, A. L. N. and Jain, A. K. (2002), "Data clustering using evidence accumulation," *International Conference on Pattern Recognition.*, 16, 276– 280 vol.4.

Gionis, A., Mannila, H., and Tsaparas, P. (2007), "Clustering aggregation," *ACM Transactions on Knowledge Discovery from Data*, 1, 4.

Hastie, T., Tibshirani, R., and Friedman, J. (2009), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, chap. Unsupervised Learning, pp. 501–527.

Hossain, M., Bridges, S. M., Wang, Y., and Hodges, J. E. (2012), "An effective ensemble method for hierarchical clustering," in *Proceedings of the Fifth International C* Conference on Computer Science and Software Engineering*, pp. 18–26.

Iam-on, N. and Garrett, S. (2010), "LinkCluE: A MATLAB Package for Link-Based Cluster Ensembles," *Journal of Statistical Software*, 36, 1–36.

Kleinberg, J. and Tardos, E. (2005), *Algorithm Design*, Addison-Wesley.

Nguyen, N. and Caruana, R. (2007), "Consensus clusterings," in *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 607–612.

Perualila-Tan, N., Shkedy, Z., Talloen, W., Goehlmann, H. W. H., Consortium, Q., Van Moerbeke, M., and Kasim, A. (2016), "Weighted-Similarity Based Clustering of Chemical Structure and Bioactivity Data in Early Drug Discovery," *Journal of Bioinformatics and Computational Biology*, 14, 1650018.

Raghavan, N., De Bondt, A., Verbeke, T., and Amaratunga, D. (2012), *Modeling Dose-response Microarray Data in Early Drug Development Experiments Using R*, Springer, chap. Gene Set Analysis as a Means of Facilitating the Interpretation of Microarray Results, pp. 181–192.

Saeed, F., Ahmed, A., and Shamsir, M. S. (2014), "Weighted voting-based consensus clustering for chemical structure databases," *Jounral of computer-aided molecular design*, 28, 675–684.

Saeed, F., Salim, N., and Abdo, A. (2012), "Voting-based consensus clustering for combining multiple clustering of chemical structures," *Journal of Cheminformatics*, 4, 37.

Smyth, G. (2004), "Linear models and empirical Bayes methods for assessing differential expression in microarray experiments," *Statistical Applications in Genetics and Molecular Biology*, 3, No. 1, Article 3.

Strehl, A. and Gosh, J. (2002), "Cluster ensembles - A knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, 3, 583–617.

Struyf, A., Hubert, M., and Rousseeuw, P. (1997), "Clustering in an Object-Oriented Environment," *Journal of Statistical Software*, 1, 1–30.

Van Moerbeke, M., Kasim, A., Amaratunga, D., Cabrera, J., and Shkedy, Z. (2018a), "The IntClust Package: An R Package for Integrated Data Analyis via Clustering," *To be submitted*.

— (2018b), "Working title: Integration of multiple data sets with M-ABC." *To be submitted*.

Wang, B., Mezlini, M. A., Demir, F., Fiume, M., Tu, Z., Brudno, M., Haibe-Kains, B., and Goldenberg, A. (2014), "Similarity Network Fusion for Aggregrating Data Types on a Genomic Scale," *Nature*, 11, 333–337.

Zheng, L., Li, T., and Ding, C. (2014), "A Framework for Hierarchical Ensemble Clustering," *ACM Transactions on Knowledge Discovery from Data*, 9, 9:1–9:23.