Vignette for **HCsnip**: An R Package for semi-supervised adaptive-height snipping of the Hierarchical Clustering tree

Askar Obulkasim

Department of Epidemiology and Biostatistics, VU University Medical Center
P.O. Box 7075, 1007 MB Amsterdam, The Netherlands
askar703@gmail.com

Introduction

This vignette shows the use of `HCsnip` package for extracting clusters from the Hierarchical Clustering (HC) tree in semi-supervised way. Rather than cutting the HC tree at a fixed highest (as existing methods do), it snips the tree at variable heights to extract hidden clusters. Cluster extraction process uses both the data matrix from which HC tree is derived and the available follow-up information for cluster evaluation. Functions for testing the significance of extracted clusters are also given. If two HC trees are presented, which maybe corresponding to the two treatment groups, this package contain functions for optimally assigning new samples to one of the HC trees and testing the significance of group assignment.

The following features are discussed in detail:

- `HCsnipper`: HC tree snipper.
- `measure`: overall partition (clustering) quality evaluation.
- `surv_measure`: overall partition (clustering) quality evaluation using follow-up data.
- `perm_test`: selection of optimal clustering from large number potential candidates and p-value calculation.
- `EnvioPlot`: a tool to visualize sample's molecular entropy.
- `cluster_pred`: semi-supervised clustering on the new set of samples.
- `RSF_eval`: error rate calculation for the clustering on the new set of samples using the Random Survival Forest.
- `TwoHC_assign`: semi-supervised group assignment for new set of samples on the basis of multiple HC trees.
- `TwoHC_perm`: assessment of the group assignment by TwoHC.assign.

Usage and explanation of the out puts of aforementioned package features are illustrated on two example data sets, which are introduced first.

1 Leukemia data set

This data set contains the sub set of gene expression profiles of adult acute myeloid leukemia patients from Bullinger *et al.*, (2004). It Contains 116 samples and 1571 genes. The follow-up data also included. The complete dataset is available at the Gene Expression Omnibus `www.ncbi.nlm.nih.gov/geo/`, accession number GSE425.

```
> library(HCsnip)
> data(BullingerLeukemia)
> names(BullingerLeukemia)
```

The code above loads a list object `BullingerLeukemia` with following components

- `em:` A matrix containing the gene expression with 7128 rows (genes) and 116 columns (samples).
- `surv.time:` A numeric vector contains patients follow-up time of patient's in *em*.
- `status:` A binary vector contains survival status of patients in *em*, 0=alive, 1=dead.

2 Glioblastoma multiforme data set

A subset of latest version of TCGA glioblastoma (GBM) level 3 gene expression data with partial clinical info (The Cancer Genome Atlas Network, 2008). It contains expression data of 120 samples and 3000 genes. Clinical data includes follow-up and type of drugs patients have been administered. The complete dataset is available at the TCGA data portal `https://tcga-data.nci.nih.gov/tcga/`.

```
> library(HCsnip)
> data(TcgaGBM)
> names(TcgaGBM)
```

The code above loads a list object `TcgaGBM` with following components

- `em:` A matrix containing the gene expression with 3000 rows (genes) and 120 columns (samples).
- `surv.time:` A numeric vector contains patients follow-up time of patient's in *em*.
- `status:` A binary vector contains survival status of patients in *em*, 0=alive, 1=dead.
- `drugs:` The types of drugs patients have been administered.

3 HCsnipper: extract complete set of partitions from the HC tree by snipping at variable heights

For a given HC tree, this function snips it at all possible places to extract the complete set of partitions (clusterings) (each one is composed of non-overlapping clusters), which includes parititions in which minimum clusters size smaller than the user specified threshold, under the condition that snipping places are chosen so that only the samples which are neighbours in the leaf node ordering are allowed to form a cluster. This constraint guarantees that sniping does not change the HC tree structure considerably. For example, samples located in far left in the HC tree will not be joined with samples located in far right. Output of this function includs a term "id" which includes the indices the paritions that satified the threshold. The number of partitions return by function depends on the shape of given HC tree. A balanced HC tree results more partitions than the skewed one.

```
> data(BullingerLeukemia)
> attach(BullingerLeukemia)
> cl <- HCsnipper(em[, 1:30], minclus = 5)
> cl <- cl$partitions[cl$id, ]
> ## To check returned partitions size
> table(apply(cl, 1, function(x) length(unique(x))))
```

4 measure: overall partition (clustering) quality assessment using molecular data

This function evaluates clusters in a given partition by user defined evaluation criteria. Numerous cluster quality measuring criteria have been proposed. In this package, we implemented few well known ones. Except for the 'c.index' and the in-group-proportion 'IGP', rest of the criteria come from the function *cluster.stats* in *fpc* package. For latter one, please check the returned arguments of the *cluster.stats* function before you decide which criteria to choose. Note that, the value returned by different criteria has different meaning. For example, the larger the Goodman and Kruskal index (g2) the better. As to G3 (g3), the smaller the better.

```
> a <- apply(cl, 1, function(x) measure(parti = x,
+                          dis = 1 - cor(em[, 1:30])))
```

5 surv_measure: overall partition (clustering) quality assessment using patients follow-up data

This function evaluates a given partition using either *AIC* or *BIC* criteria. In both settings, It fits a Cox model using follow-up data as response and cluster labels in the partition as covariate. The likelihood from the fitted model further used to calculate the modified *AIC* (Liang and Zou, 2008) or *BIC* (Volinsky and Raftery, 2000). See references for more details. Note that, for convenience in later usage, returned value is multiplied by -1 so that large value denotes good quality partition.

```
> b <- apply(cl, 1, function(x) surv_measure(x,
+                       surv.time[1:30], status[1:30]))
```

6 perm_test: optimal partition selection and significance testing

For a given set of partitions (each partition is composed of non-overlapping clusters), this function uses two types of data to evaluate each partition and select the optimal one which has the highest rank in terms of both data type (presumed that *score1* and *score2* were from two different data source). Permutation approach used to calculate the corrected p-value of the selected partition.

When studying association of cluster membership with clinical follow-up, such as survival data, we cannot use the standard testing procedures when our semi-supervised approach has been applied: we would use the follow-up data twice and the resulting p-value is likely to be too small. We avoid this bias by also applying the semi-supervised cluster construction under the null-hypothesis. This null-hypothesis is simply the absence of association between the data type from which *score2* is calculated and the follow-up. Then, our cluster construction in combination with a suitable test statistic is designed to detect associations that can be represented by groups of samples. We adapt the p-value computation as follows.

1. Select an optional partition from given set.
2. Use a suitable test statistic (e.g. log-rank for time-to-event data and chi-square for nominal data) to compute the *conditional* p-value given the resulting clusters: p_{obs}.
3. For $i = 1, \ldots, B$ (e.g. $B = 1000$):
   (a) Randomly permute the follow-up data among the samples and calculate new *score1*. *score2* is fixed.
   (b) Select an optimal partition using new *score1* and *score2* in exactly the same as we did before.
   (c) Conditional on the resulting clusters, compute p-value p_i.
4. Finally, compute the p-value of interest: $p = P(p_{obs} \geq p_i) = (\#i : p_{obs} \geq p_i)/B$.

Here, p satisfies a crucial property of p-values: it is uniformly distributed when the null-hypothesis is true, because then p_{obs} and p_i are exchangeable random variables. The exchangeability is a result from the null-hypothesis and the use of exactly the same procedures to compute p_{obs} and p_i.

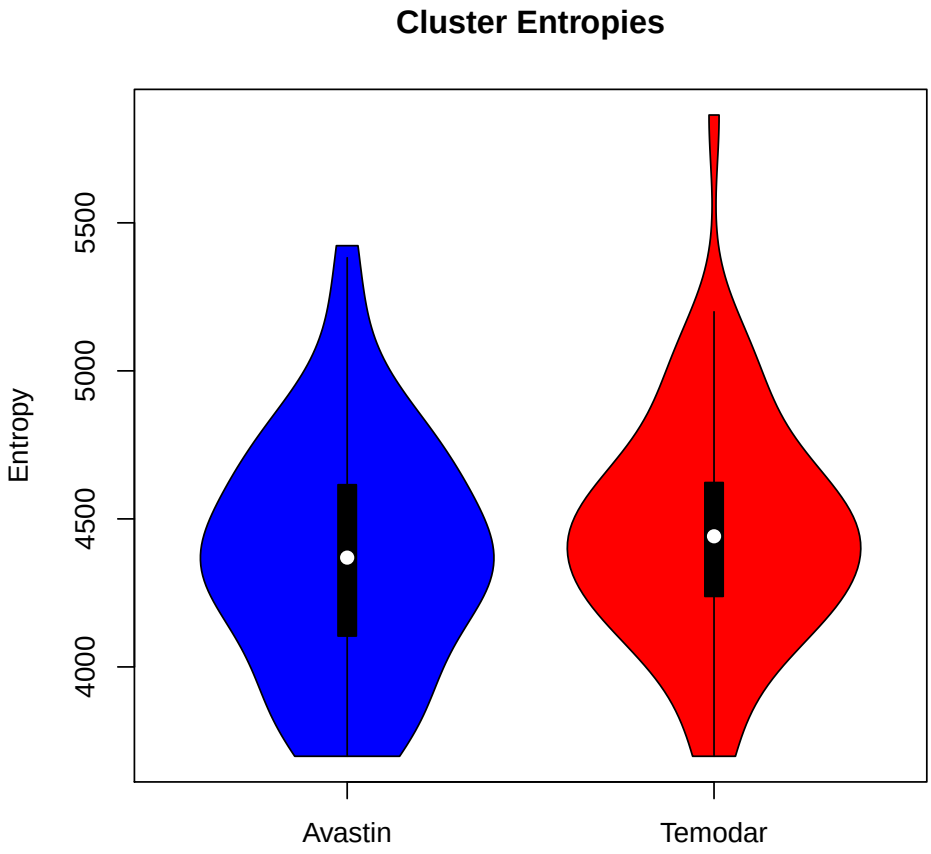
```
> result <- perm_test(cl, surv.time[1:30], status[1:30], score1 = a, score2 = b,
+                         nperm = 10)
```

7 EnvioPlot: cluster visualization using samples molecular entropy

To visually inspect the differences between clusters in terms of sample's molecular profiles, we turn to sample's molecular entropy proposed by van Wieringen,N.W. and van der Vaart,W.A. (2010). They showed that entropy can be used to quantify the overall expression pattern of a cancer sample. High entropy corresponds to high overall expression of genes in the sample.

This function first calculates the entropy for each sample in the given partition, and makes a violin plot for each cluster. If clusters are different in term of their molecular profiles, then one may expect different shaped violin boxes.

```
> data(TcgaGBM)
> em <- TcgaGBM$em
> drugs <- TcgaGBM$drug
> gr <- rep(1, ncol(em))
> gr[drugs == "Temodar"] <- 2
> H <- EnvioPlot(X = em, parti = gr, names = c("Avastin", "Temodar"),
+              col = c("blue", "red"))
```



8 cluster_pred: clustering on the new set

For a given partition (composed of non-overlapping clusters), this function assigns new samples to one of the clusters in the partition. User has two options to assign test set to one of the clusters in the partition. One option is to use the Ward distance. Specifically, an average distance is calculated between a test sample and each cluster in the partition, separately. The test sample is assigned to a cluster for which average distance is the smallest. Follow-up information is not needed for this option.

Second option is to use the Harrel's concordance index (Harrel *et al.*, 1982). For this option both data matrix and follow-up information of samples in the partition are required. Data matrix is used to find the pseudo nearest neighbours (PNN) (Obulkasim *et al.*, 2011) of a test sample, and follow-up information are used to check how much PNN's follow-up info is concordant with samples follow-up in each cluster. The test sample is assigned to a cluster for which average concordance value is the largest. Before selecting either one of the options, we recommend user to check the correlation between the data matrix and follow-up (e.g. by global test). If the correlation is relatively large, we recommend to use *conc* option, and vice versa.

```
> data(BullingerLeukemia)
> attach(BullingerLeukemia)
> par(mfrow = c(1, 2))
> pred <- cluster_pred(X = em[, 1:60], partition = result$best,
+               surv.time = surv.time[1:30], status = status[1:30], te.index = 31:60,
+               te.surv.time = surv.time[31:60], te.status = status[31:60], plot.it = TRUE)
```

```
> H <- EnvioPlot(X = em[, 31:60], parti = pred[["St"]][, 2])
```

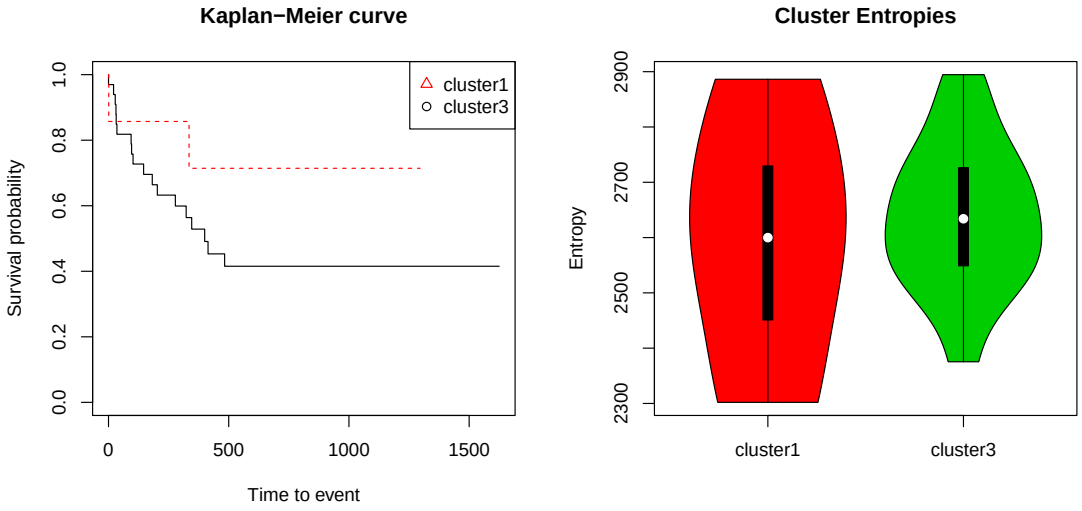


Figure 1: The left figure shows the differences between clusters in the test set in terms of follow-up. The right figure shows the differences in terms of their molecular profiles.

9 RSF_eval: evaluate the clustering on the new set by the Random Survival Forest

This function constructs survival forest (shwaran *et al.*, 2008) using training sample's follow-up as response and cluster labels as covariates. Constructed forest is used to calculate cumulative hazard function (CHF) for each sample in the new set based on its cluster label. CHFs are compared with new sample's actual survival time to calculate the error rate. Error rate is 1 - *C-index* (Harrel's concordance index, 1982). Error rates are between 0 and 1, 1 being the perfect match.

```
> cl <- HCsnipper(em[, 1:40], minclus = 4)
> cl <- cl$partitions[cl$id, ]
> pred <- cluster_pred(X = em[, 1:60], partition = cl[6, ],
+                surv.time = surv.time[1:40], status = status[1:40], te.index = 41:60)
> Err <- RSF_eval(cl[1, ], surv.time[1:40], status[1:40],
+                 pred, surv.time[41:60], status[41:60])
```

10 TwoHC_assign: assign new samples to one of the two HC tree using semi-supervised approach

Say two group of patients (without overlap) treated with two different drugs or the same drugs in different combinations are available. Assume that patients have been administered one of the two different drugs based patient's clinical characteristics and the prior knowledge of doctors. Follow-up information for each group is also available. When a new patients (presume that new patients came from the same population as patients in the two groups) comes in, question will be to which group this patient should be assigned so that he/she will benefit most by the type of treatment this group received.

This function works as follows: first, two independent HC trees will be derived from given data; second, partitions are extracted and the optimal partition is selected for each HC tree. Finally, new patient's data profile (pretend that new patient's follow-up and group label are missing) is compared with each cluster in each optimal partition to calculate average similarity and new samples is assigned to the cluster to which average similarity is highest. The HC tree which contains this cluster will be the group label for this new patient.

```
> data(TcgaGBM)
> attach(TcgaGBM)
> id1 <- which(drugs == "Avastin")
> id2 <- which(drugs == "Temodar")
> twoHC <- TwoHC_assign(X = em[ ,c(id1[1:30], id2[1:30])], index1 = 1:30,
+                 index2 = 31:60, new.X = em[, c(id1[31:60], id2[31:60])],
+                 minclus = 4, surv.time = surv.time[c(id1[1:30], id2[1:30])],
+                 status = status[c(id1[1:30], id2[1:30])])
```

11 TwoHC_perm: assess the quality of group assignment by TwoHC_assign

Significance of group assignment for each patient is calculated as follows: select a patient from the test set, examine the previously found best partition in each HC tree and identify the clusters to which this sample is most similar. Say these two clusters are cluster_1 and cluster_2 of size n_1 and n_2, respectively. Create a binary vector (call it x) of size $n_1 + n_2$ contains n_2 ones and n_1 zeros. Construct a Cox model using follow-up information of samples in cluster_1 and cluster_2 as response, and x as covariate. The absolute value of the estimated group parameter ($\hat{\beta}^i_{\text{obs}}$) in the Cox model that compares the survival times of the other patients in the two competing clusters expresses the predicted gain in survival from the assignment by `TwoHC_assign` with respect to random. The β^i_{obs} will be transformed to $r^i_{\text{obs}} = \exp\left(-|\hat{\beta}^i_{\text{obs}}|\right)$, which quantifies the gain in relative risk in the Cox model. The problem is that this is biased, because `TwoHC_assign` already used $\hat{\beta}^i_{\text{obs}}$. Hence, even when the two groups would be equally good for the molecular profiles in the two competing clusters, we obtain $r^i_{\text{obs}} < 1$. To correct for this bias this function uses a permutation argument. For each new patient it applies *nperm* permutations of the survival data among the two competing clusters. As above it computes r^j_{perm} for each permutation j, which contains the same bias as r^i_{obs}. The relative risk-ratio of the i^{th} new patient is calculated as:

$$rr^i_{\text{obs}} = \frac{\exp\left(-|\hat{\beta}^i_{\text{obs}}|\right)}{Z_i},$$

where

$$Z_i = median\left(r^1_{\text{perm}}, r^2_{\text{perm}}, ..., r^{nperm}_{\text{perm}}\right).$$

rr^i_{obs} quantifies the biased-corrected reduction in relative risk. Function defines a test statistic of the new samples:

$$T_{\text{obs}} = \frac{\frac{1}{n}\sum_{i=1}^{n}\log\left(rr^i_{\text{obs}}\right)}{\text{stdev}\left(\log\left(rr^1_{\text{obs}}, rr^2_{\text{obs}}, \ldots, rr^n_{\text{obs}}\right)\right)},$$

which is just the scaled version of the log-geometric mean (suited well for the ratio scaled data). The permuted version of rr^j_{perm} and T^j_{perm} is defined analogously. Finally, compute the p-value of interest: $p = P(T_{\text{obs}} \geq T^j_{\text{perm}}) = (\#j : T_{\text{obs}} \geq T^j_{\text{perm}})/\text{nperm}$.

```
> result <- TwoHC_perm(twoHC, nperm = 100)
```

References

Obulkasim,A. *et al.* (2013), Semi-supervised adaptive-height snipping of the Hierarchical Clustering tree, submitted.

Bullinger,L. *et al.* (2004), Use of gene-expression profiling to identify prognostic subclasses in adult acute myeloid leukemia, *N Engl J Med.*, **350**, 1605-1616.

The Cancer Genome Atlas Network (2008), Comprehensive genomic characterisation defines human glioblastoma genes and core pathways, *Nature*, **490**, 61-70.

Hennig,C. (2010), fpc: Flexible procedures for clustering, *R package*, `http://CRAN.R-project.org/package=fpc`.

Liang,H. and Zou,G.H. (2008), Improved AIC selection strategy for survival analysis, *Comput Stat Anal.*, **52**, 2538-2548.

Volinsky,T.C. and Raftery,A.E. (2000), Baysian information criteria for censored survival models, *Biometrics*, **56**, 256-262.

Obulkasim,A. *et al.* (2011), Stepwise Classification of Cancer Samples using Clinical and Molecular Data. *BMC Bioinformatics*, **12**, 422-424.

Harrel,F.E. *et al.* (1982), Evaluating the yield of medical tests, *JAMA*, **247**, 2543-2546.

van Wieringen,N.W. and van der Vaart,W.A. (2010), Statistical analysis of the cancer cell's molecular entropy using high-throughput data *Bioinformatics*, **27**, 556-563.

Ishwaran,H. *et al.* (2008) Random survival forest, *Ann. App. Statist.*, **2**, 841-860.

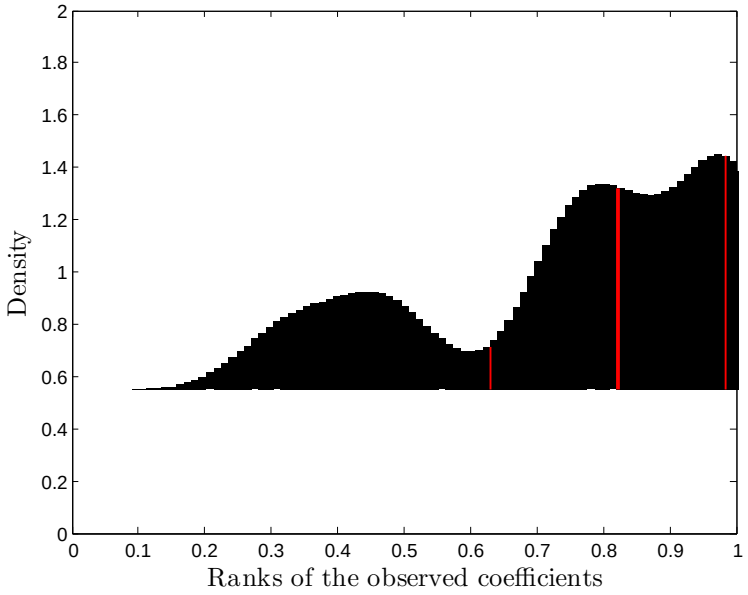


Figure 2: Density of the observed coefficient ($\hat{\beta}_{obs}$) ranks from $n = 20$ samples. We used $nperm = 10000$ permutations. The tick colour bars denotes the 25,50,75 percentiles, respectively.

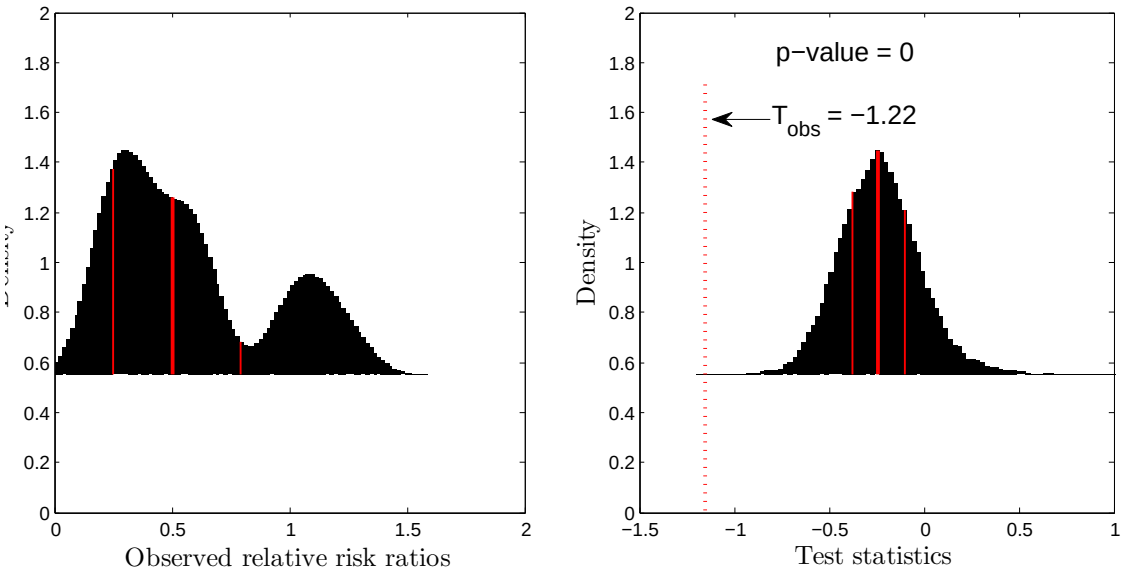


Figure 3: The left figure shows the density of the observed relative risk-ratios (rr_{obs}). The right figure shows the distribution of the test statistics from the null model from permutation (T_{perm}). The tick colour bars denotes the 25,50,75 percentiles, respectively.