A practical experiment of using speaq package

Trung Nghia VU, Kris Laukens and Dirk Valkenborg

January 5, 2015

1 Introduction

We introduce a novel suite of informatics tools for the quantitative analysis of NMR metabolomic profile data. The core of the processing cascade is a novel peak alignment algorithm, called hierarchical Cluster-based Peak Alignment (CluPA).

The algorithm aligns a target spectrum to the reference spectrum in a top-down fashion by building a hierarchical cluster tree from peak lists of reference and target spectra and then dividing the spectra into smaller segments based on the most distant clusters of the tree. To reduce the computational time to estimate the spectral misalignment, the method makes use of Fast Fourier Transformation (FFT) cross-correlation. Since the method returns a high-quality alignment, we can propose a simple methodology to study the variability of the NMR spectra. For each aligned NMR data point the ratio of the between-group and within-group sum of squares (BW-ratio) is calculated to quantify the difference in variability between and within predefined groups of NMR spectra. This differential analysis is related to the calculation of the F-statistic or a one-way ANOVA, but without distributional assumptions. Statistical inference based on the BW-ratio is achieved by bootstrapping the null distribution from the experimental data.

We are going to introduce step-by-step how CluPA works for a specific dataset, includes

- automatically do alignment
- allow user intervening into the process
- compute BW ratios
- visualize results

For experiments, we used the wine dataset in the experiments of software `icoshift` (ver 1.1.1) at http://www.models.life.ku.dk/icoshift. The wine data from that address needs to be extracted, exported in csv format and named as "wine.csv" for this experiment. If you want to run the peak detection section, MassSpecWavelet package needs to be installed.

2 Data input

```
> # load('wine.RData')
> library(speaq)
> X  <- read.csv("wine.csv", header = FALSE);
> X <- as.matrix(X);
```

3 Peak detection

This section makes use of MassSpecWavelet package to detect peak lists of the dataset.

```
> cat("\n detect peaks....");

 detect peaks....

> startTime <- proc.time();
> peakListTmp <- detectSpecPeaks(X,
+         nDivRange = c(128),
+         scales = seq(1, 16, 2),
+         baselineThresh = 50000,
+         SNR.Th = -1,
+     verbose=FALSE
+ );
> peakList <- peakListTmp; #just a backup for next usage if necessary
> endTime <- proc.time();
> cat("Peak detection time: ", (endTime[3] - startTime[3])/60, " minutes");

Peak detection time:  0.49165  minutes
```

4 Reference finding

Now we are going to find the reference for other spectra align to.

```
> refInd = 0;
> startTime <- proc.time();
> if (refInd == 0){
+     cat("\n Find the spectrum reference...")
+     resFindRef<- findRef(peakList);
+     refInd <- resFindRef$refInd;
+     cat("\n Order of spectrum for reference  \n");
+     for (i in 1:length(resFindRef$orderSpec))
+     {
+         cat(paste(i, ":",resFindRef$orderSpec[i],sep=""), " ");
+         if (i %% 10 == 0) cat("\n")
+     }
```

```
+
+ }

 Find the spectrum reference...
 Order of spectrum for reference
1:35  2:28  3:22  4:2  5:27  6:11  7:31  8:9  9:13  10:3
11:7  12:16  13:23  14:15  15:37  16:39  17:20  18:30  19:14  20:18
21:26  22:29  23:12  24:32  25:4  26:36  27:21  28:8  29:19  30:33
31:40  32:24  33:10  34:5  35:17  36:34  37:38  38:25  39:6  40:1

> endTime <- proc.time();
> cat("\n Finding reference spectrum time: ", (endTime[3] - startTime[3])/60, " minutes");

 Finding reference spectrum time:  0.01545  minutes

> cat("\n The reference is: ", refInd);

 The reference is:  35
```

5 Spectral alignment

This section uses hierarchical Cluster-based Peak Alignment [1] to align spectra togetther. In general, we do alignment for the whole spectra using *dohCluster* function.

```
> # Set maxShift
> maxShift = 50;
> Y <- dohCluster(X,
+                 peakList = peakList,
+                 refInd = refInd,
+                 maxShift  = maxShift,
+                 acceptLostPeak = TRUE, verbose=FALSE);
>
```

In some cases, if users just want to align in speficic segments or want to use different parametter setting for different segments. speaq allows users to do that by intervene into the process. To do that, users need to create a info file in advance. Table 1 is an example.

Table 1: Example of infomation file to customize spectral aligment to segments

begin	end	forAlign	ref	maxShift
3600	4000	0	0	0
5600	6200	1	0	50

Each row contains the following information corresponding to the columns:

- begin: the starting point of the segment.
- end: the end point of the segment.
- forAlign: the segment is aligned (1) or not (0).
- ref: the index of the reference spectrum. If 0, the algorithm will select the reference found by the reference finding step.
- maxShift: the maximum number of points of a shift to left/right.

It is worth to note that only segments with forAlign=1 will be taken into account for spectral alignment.

Below displays an example of the contents of the csv file (Wineinfo.csv) for the table above.

begin,end,forAlign,ref,maxShift
3600,4000,0,0,0
5600,6200,1,0,50

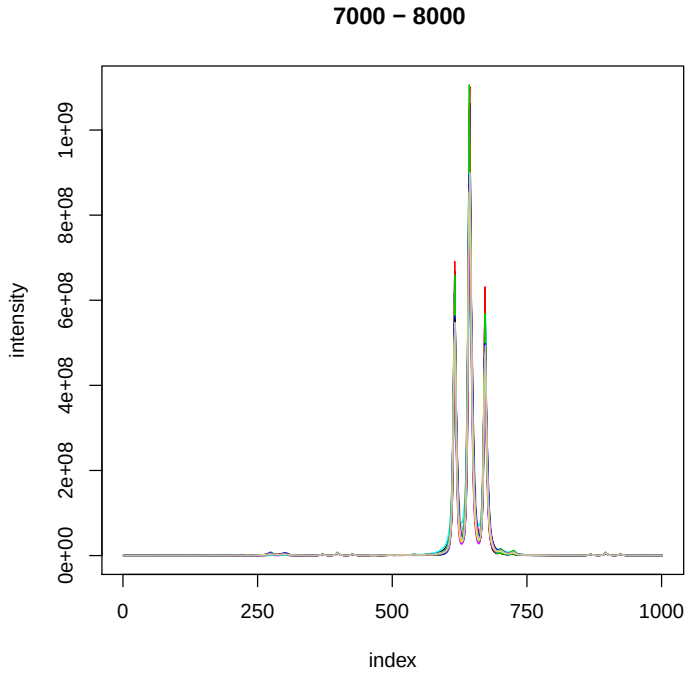
Now, simply run *dohClusterCustommedSegments* with the input from the infomation file.

```
> Yc <- dohClusterCustommedSegments(X,
+                                   peakList = peakList,
+                                   refInd = refInd,
+                                   maxShift  = maxShift,
+                                   acceptLostPeak = TRUE,
+                                   infoFilename = "Wineinfo.csv",
+                                   minSegSize = 128,
+                                   verbose=FALSE)
>
```

6 Spectral plots

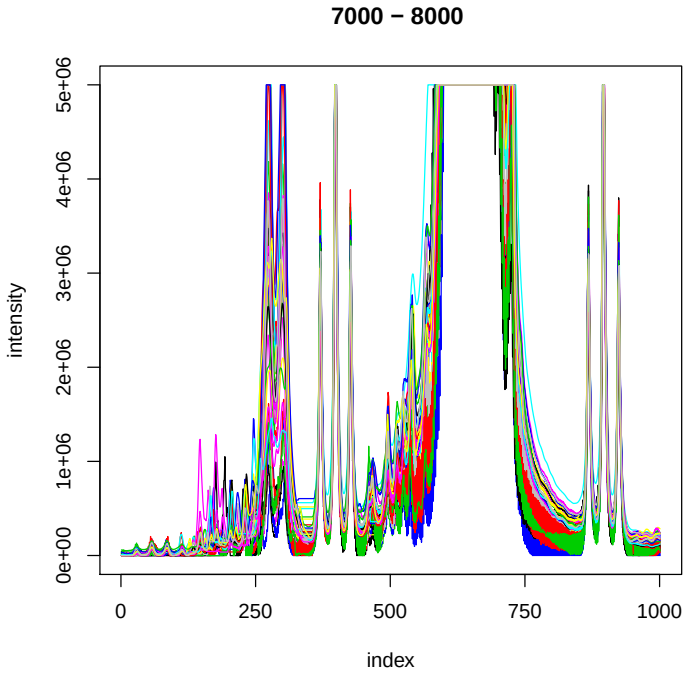
We could draw a segment to see the performance of the alignement.

```
> drawSpec(Y,
+         startP=7000,
+         endP=8000
+         );
```



We could limit the heights of spectra to easily check the alignment.

```
> drawSpec(Y,
+          startP=7000,
+          endP=8000,
+          highBound = 5e+6,
+          lowBound = -1000);
```



7 Quantitative analysis

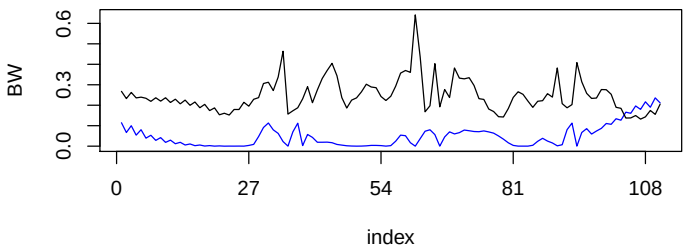
This section presents the quantatative analysis for wine data that was used in our paper [1]. To save time, we just do permutation 100 times to create null distribution.

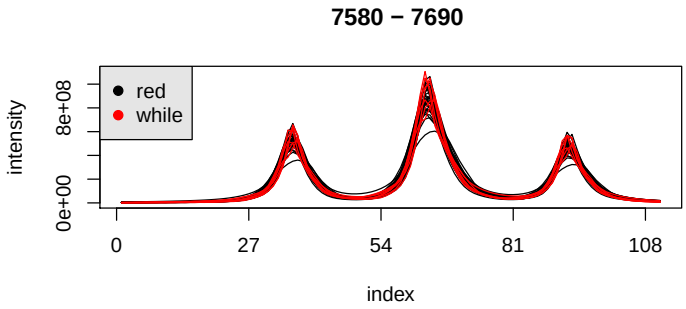
```
> N = 100;
> alpha = 0.05;
> wineLabel <- c("white","red","red","white","red","white","red","rose","red","red","red","w
> wineLabel <- as.factor(wineLabel)
> U1 = Y[which(wineLabel=="white"),]
> U2 = Y[which(wineLabel=="red"),]
> U = rbind(U1, U2)
> Ulabel <- c("while","while","while","while","while","while","while",
+               "red","red","red","red","red","red","red","red","red","red",
+               "red","red","red","red","red","red","red","red","red","red",
+               "red","red","red","red","red","red","red","red","red","red",
+               "red");
> Ulabel <- as.factor(Ulabel);
> # find the BW-statistic
> BW = BWR(U, Ulabel);
```

```
> # create sampled H0 and export to file
> H0 = createNullSampling(U, Ulabel, N = N,verbose=FALSE)
> #compute percentile of alpha
> perc = double(ncol(U));
> alpha_corr = alpha/sum(returnLocalMaxima(U[2,])$pkMax>50000);
> for (i in 1 : length(perc)){
+     perc[i] = quantile(H0[,i],1-alpha_corr, type = 3);
+ }
```

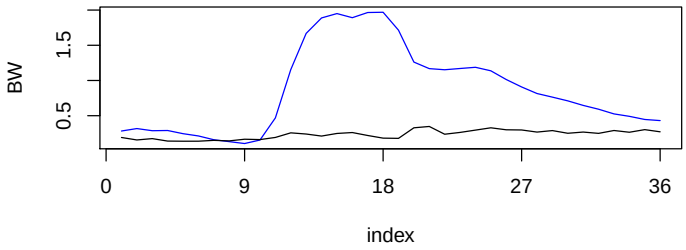
Now, some figures are plotting. Read the publication to understand more about these figures.

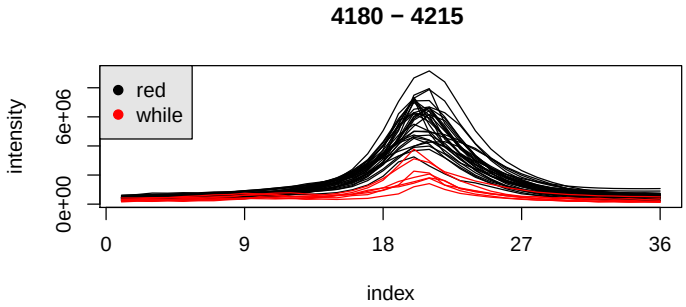
```
> drawBW(BW, perc,U,startP = 7580, endP = 7690, groupLabel = Ulabel) #b
```



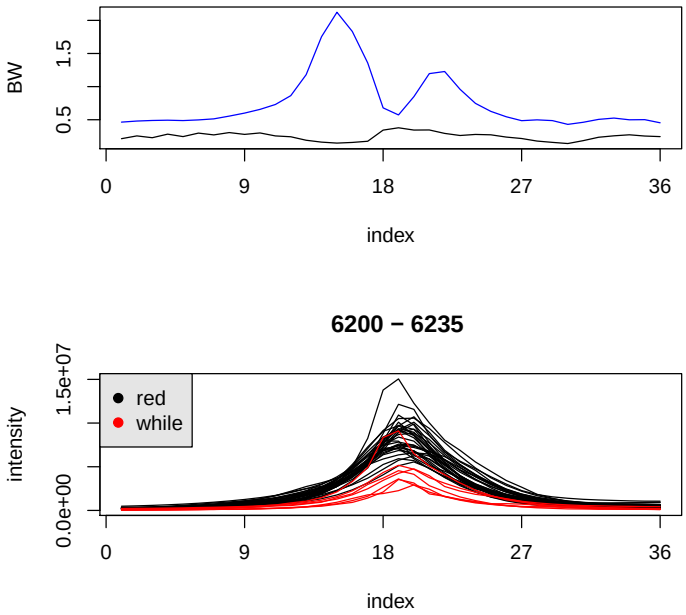



```
> drawBW(BW, perc,U,startP = 4180, endP = 4215, groupLabel = Ulabel) #c
```






```
> drawBW(BW, perc,U,startP = 6200, endP = 6235, groupLabel = Ulabel) #d
```



8 References

1. Vu TN, Valkenborg D, Smets K, Verwaest KA, Dommisse R, Lemie're F, Verschoren A, Goethals B, Laukens K. (2011) An integrated workflow for robust alignment and simplified quantitative analysis of NMR spectrometry data. BMC Bioinformatics. 2011 Oct 20;12:405.