Discriminant Fuzzy Pattern to Filter Differentially Expressed Genes

Rodrigo Alvarez-Gonzalez, Daniel Glez-Pena, Fernando Diaz, Florentino Fdez-Riverola

May 3, 2016

Contents

1 Overview

The advent of DNA microarray technology has supplied a large volume of data to many fields like machine learning and data mining. Intelligent support is essential for managing and interpreting this great amount of information. One of the well-known constraints specifically related to microarray data is the large number of genes in comparison with the small number of available experiments. In this context, the ability of design methods capable of overcoming current limitations of state-of-the-art algorithms is crucial to the development of successful applications. In this work we demonstrate how a supervised fuzzy pattern algorithm can be used to perform DNA microarray data reduction over real data [1]. The benefits of our method can be employed to find biologically significant insights relating to meaningful genes in order to improve previous successful techniques.

Classical gene selection methods tend to identify differentially expressed genes from a set of microarray experiments. These genes are expected to be up- or down-regulated between healthy and diseased tissues or, more generally, between different classes. A differentially expressed gene is a gene which has the same expression pattern for all samples of the same class, but different for samples belonging to different classes. The relevance value of a gene depends on its ability to be differentially expressed. However, a non-differentially expressed gene will be considered irrelevant and will be removed from a classification process even though it might well contain information that would improve classification accuracy. One way or another, the selected method has to pursue two main goals: (i) reduce the cost and complexity of the classifier and (ii) improve the accuracy of the model.

These methods rank genes depending on their relevance for discrimination. Then by setting a threshold, one can filter the less relevant genes among those considered. As such, these filtering methods may be seen as particular gene selection methods. An important task in microarray data analysis is therefore to identify genes, which are differentially expressed in this way. Statistical analysis of gene expression data relating to complex diseases is of course not really expected to yield accurate results. A realistic goal is to narrow the field for further analysis, to give geneticists a short list of genes for analysis into which hard-won funds are worth investing.

2 Gene Selection Using Fuzzy Pattern

This work proposes a method for selecting genes which is based on the notion of fuzzy pattern [1,2]. Briefly, given a set of microarrays which are well classified, for each class it can be constructed a fuzzy pattern (FP) from the fuzzy microarray descriptor (FMD) associated to each one of the microarrays. The FMD is a comprehensible description for each gene in terms of one from the

following linguistic labels: 'Low', 'Medium' and 'High'. Therefore, the fuzzy pattern is a prototype of the FMDs belonging to the same class where the membership criterion of each gene to the fuzzy pattern of the class is frequency-based. Obviously, this fact can be of interest, if the set of initial observations are considered of the same class. The pattern's quality of fuzziness is given by the fact that the selected labels come from the linguistic labels defined during the transformation into FMD of an initial observation. Moreover, if a specific label of one feature is very common in all the examples belonging to the same class, this feature is selected to be included in the pattern.

The goal of gene selection in this work is to determine a reduced set of genes, which are useful to classify new cases within one of the known classes [3]. For each class it is possible to compute a fuzzy pattern from the available data. Since each pattern is representative of a collection of microarrays belonging to the same class, we can assume that the genes included in a pattern, are significant to the classification of any novel case within the class associated with that pattern. Now we are interested in those genes that allow us to discriminate the new case from one class with regard to the others. Here we introduce the notion of discriminant fuzzy pattern (DFP) with regard to a collection of fuzzy patterns. A DFP version of a FP only includes those genes that can serve to differentiate it from the rest of the patterns. The algorithm used to compute the DFP version of each FP in a collection of fuzzy patterns is shown in the following pseudo code.

```
procedure discriminantFuzzyPattern (input: ListFP; output: ListDFP) {
  begin
    initialize_DFP: FP <- φ
    for each fuzzy pattern FPi ϵ ListFP do
      initialize_DFP: DFPi <- φ
      for each fuzzy pattern FPj ϵ ListFP and FPi <> FPj do
        for each gen g ϵ getGenes(FPi) do
          if (g ϵ getGenes(FPj)) AND
            (getLabel(FPi, g) <> getLabel(FPj, g)) then
              addMember(DFPi, member(FPi, g))
      add_to_List_of_DFP: add(ListDFP, DFPi)
  end
}
```

As can be observed from the algorithm, the computed DFP for a specific FP is different depending on what other FPs are compared with it. It's not surprising that the genes used to discern a specific class from others (by mean of its DFP) will be different if the set of rival classes also changes.

3 Examples

The DFP package can be used to select significant genes from a microarray experiment. The main function (`discriminantFuzzyPattern`) is parameterized in order to tune the filtering. This algorithm is based on the discretization of float values (gene expression values) stored in an `ExpressionSet` object into labels combining 'Low', 'Medium' and 'High'.

3.1 Quick Start Example

```
> library(DFP)
```

This quick start example will be carried out using the artificial data set `rmadataset`, included in the package.

```
> data(rmadataset)
> # Number of genes in the test set
> length(featureNames(rmadataset))
```

```
[1] 500
```

Once the data is loaded you can execute the algorithm[1], which will work out with the default parameter values.

```
> res <- discriminantFuzzyPattern(rmadataset)
```

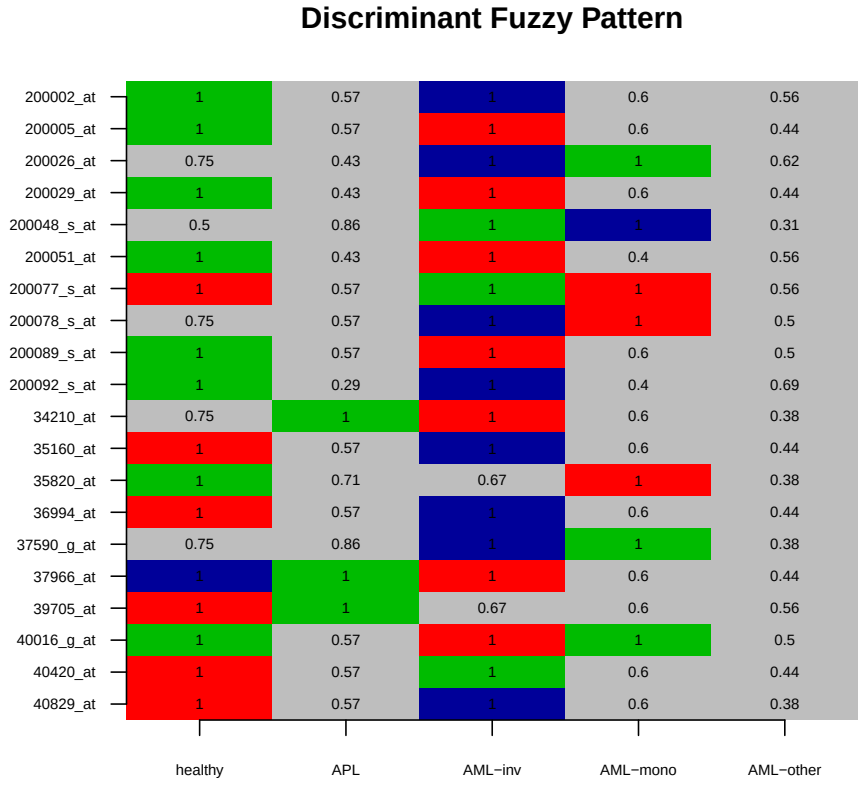
The selected genes can be shown in both text and graphical mode using following function:

```
> plotDiscriminantFuzzyPattern(res$discriminant.fuzzy.pattern)
```

```
            healthy  APL   AML-inv  AML-mono AML-other
200002_at   "Low"    NA    "Medium" NA       NA
200005_at   "Low"    NA    "High"   NA       NA
200026_at   NA       NA    "Medium" "Low"    NA
200029_at   "Low"    NA    "High"   NA       NA
200048_s_at NA       NA    "Low"    "Medium" NA
200051_at   "Low"    NA    "High"   NA       NA
200077_s_at "High"   NA    "Low"    "High"   NA
200078_s_at NA       NA    "Medium" "High"   NA
200089_s_at "Low"    NA    "High"   NA       NA
200092_s_at "Low"    NA    "Medium" NA       NA
34210_at    NA       "Low" "High"   NA       NA
35160_at    "High"   NA    "Medium" NA       NA
35820_at    "Low"    NA    NA       "High"   NA
36994_at    "High"   NA    "Medium" NA       NA
37590_g_at  NA       NA    "Medium" "Low"    NA
37966_at    "Medium" "Low" "High"   NA       NA
39705_at    "High"   "Low" NA       NA       NA
40016_g_at  "Low"    NA    "High"   "Low"    NA
40420_at    "High"   NA    "Low"    NA       NA
40829_at    "High"   NA    "Medium" NA       NA
attr(,"ifs")
            healthy       APL   AML-inv AML-mono AML-other
200002_at      1.00 0.5714286 1.0000000      0.6    0.5625
200005_at      1.00 0.5714286 1.0000000      0.6    0.4375
200026_at      0.75 0.4285714 1.0000000      1.0    0.6250
200029_at      1.00 0.4285714 1.0000000      0.6    0.4375
200048_s_at    0.50 0.8571429 1.0000000      1.0    0.3125
200051_at      1.00 0.4285714 1.0000000      0.4    0.5625
200077_s_at    1.00 0.5714286 1.0000000      1.0    0.5625
200078_s_at    0.75 0.5714286 1.0000000      1.0    0.5000
200089_s_at    1.00 0.5714286 1.0000000      0.6    0.5000
200092_s_at    1.00 0.2857143 1.0000000      0.4    0.6875
34210_at       0.75 1.0000000 1.0000000      0.6    0.3750
35160_at       1.00 0.5714286 1.0000000      0.6    0.4375
35820_at       1.00 0.7142857 0.6666667      1.0    0.3750
36994_at       1.00 0.5714286 1.0000000      0.6    0.4375
37590_g_at     0.75 0.8571429 1.0000000      1.0    0.3750
37966_at       1.00 1.0000000 1.0000000      0.6    0.4375
39705_at       1.00 1.0000000 0.6666667      0.6    0.5625
```

[1] Working with a huge amount of genes (around 20.000) it will take a few minutes

```
40016_g_at      1.00 0.5714286 1.0000000      1.0    0.5000
40420_at        1.00 0.5714286 1.0000000      0.6    0.4375
40829_at        1.00 0.5714286 1.0000000      0.6    0.3750
```



This function plots the Discriminant Fuzzy Pattern of the relevant genes (in rows) for the sample classes (in columns), as well as the impact factor (`ifs`) which determines if a gene belongs to a Fuzzy Pattern in a class (if its value is higher than the `piVal` parameter). In the first table, a `NA` value is shown if the impact factor is lower or equal than the `piVal` parameter, which points out that the corresponding gene does not pertain to the Fuzzy Pattern of the class.

The relevant genes are those which are present in at least two different Fuzzy Patterns with different linguistic labels.

The plotting in graphical mode, shows the liguistic labels as follows:

- 'Low': colour BLUE.
- 'Medium': colour GREEN.
- 'High': colour RED.

3.2 Extended Example

```
> library(DFP)
```

Instead of the ExpressionSet class (rmadataset) which accompanies the package, you can load external data in a predefined CSV format.

First of all, you need to specify the package install directory:

```
> dataDir <- system.file("extdata", package="DFP"); dataDir

[1] "/private/tmp/RtmpCCizpG/Rinst15b0940e73793/DFP/extdata"
```

Secondly, you need to append the file names containing the data and metadata:

```
> fileExprs <- file.path(dataDir, "exprsData.csv"); fileExprs

[1] "/private/tmp/RtmpCCizpG/Rinst15b0940e73793/DFP/extdata/exprsData.csv"

> filePhenodata <- file.path(dataDir, "pData.csv"); filePhenodata

[1] "/private/tmp/RtmpCCizpG/Rinst15b0940e73793/DFP/extdata/pData.csv"
```

Finally, you can use the readCSV function to read from a CSV file into an ExpressionSet with annotated metadata:

```
> rmadataset <- readCSV(fileExprs, filePhenodata); rmadataset

ExpressionSet (storageMode: lockedEnvironment)
assayData: 500 features, 35 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: 0C12_S 0C179_S ... XP16942_S (35 total)
  varLabels: class age sex
  varMetadata: labelDescription
featureData: none
experimentData: use 'experimentData(object)'
Annotation:
```

The parameters you can use to tune the functionality of the algorithm are the following (initialized to the default value):

```
> skipFactor <- 3 # Factor to skip odd values
> zeta <- 0.5 # Threshold used in the membership functions to label the float values with a discre
> piVal <- 0.9 # Percentage of values of a class to determine the fuzzy patterns
> overlapping <- 2 # Determines the number of discrete labels
```

Once the data and parameters are ready, you can execute the algorithm. In order to understand how the algorithm works, the global task is split into the 4 steps it involves:

- STEP 1: Calculate Membership Functions.

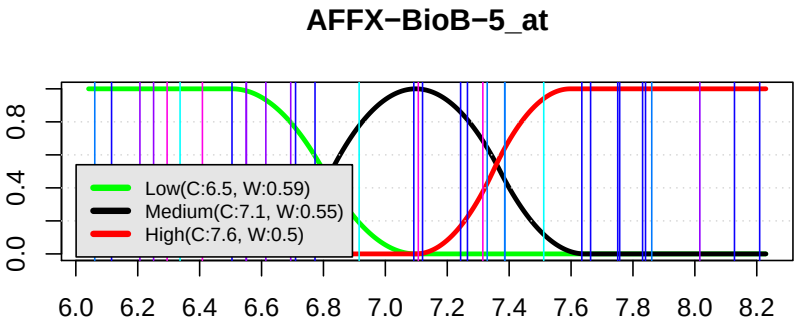
These functions are used in the next step (discretizeExpressionValues) to discretize gene expression data.

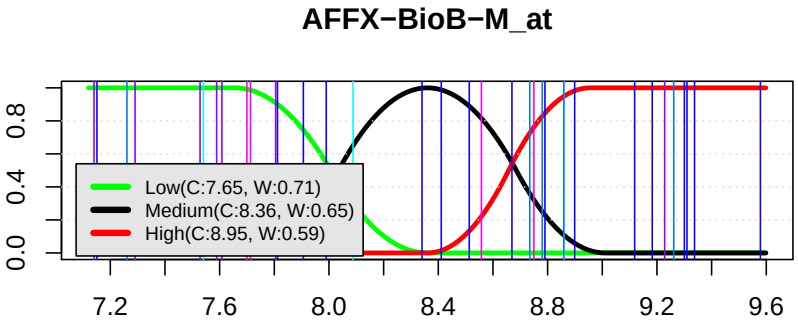
```
> mfs <- calculateMembershipFunctions(rmadataset, skipFactor); mfs[[1]]

$lel
Class Type: LowExpressionLevel
Center: 6.504947
Width: 0.5931398
```

```
$mel
Class Type: MediumExpressionLevel
Center: 7.098087
Width: 0.546313

$hel
Class Type: HighExpressionLevel
Center: 7.597573
Width: 0.4994861
```

```
> plotMembershipFunctions(rmadataset, mfs, featureNames(rmadataset)[1:2])
```

```
              Center(Low) Width(Low) Center(Medium) Width(Medium) Center(High)
AFFX-BioB-5_at        6.50       0.59           7.10          0.55         7.60
AFFX-BioB-M_at        7.65       0.71           8.36          0.65         8.95
              Width(High)
AFFX-BioB-5_at        0.50
AFFX-BioB-M_at        0.59
```





- STEP 2: Discretize Expression Values.
  This function converts the gene expression values into linguistic labels.

```
> dvs <- discretizeExpressionValues(rmadataset, mfs, zeta, overlapping); dvs[1:4,1:10]
```

```
              0C12_S 0C179_S 0C167_S  0C0936_S AP5204_S AP10222_S AP12366_S
AFFX-BioB-5_at "Low"  "Low"   "Medium" "Medium" "Low"    "Low"     "Low"
AFFX-BioB-M_at "Low"  "Low"   "Medium" "High"   "Low"    "Low"     "Low"
AFFX-BioB-3_at "Low"  "Low"   "Medium" "Medium" "Low"    "Low"     "Low"
AFFX-BioC-5_at "Low"  "Low"   "Medium" "Medium" "Low"    "Low"     "Low"
              AP13058_S AP13223_S AP14217_S
AFFX-BioB-5_at "Low"     "Low"     "High"
AFFX-BioB-M_at "Low"     "Low"     "High"
AFFX-BioB-3_at "Low"     "Low"     "High"
AFFX-BioC-5_at "Low"     "Low"     "High"
```

```
> showDiscreteValues(dvs, featureNames(rmadataset)[1:10],c("healthy", "AML-inv"))
```

```
               0C12_S   0C179_S 0C167_S  0C0936_S BP185_S BP355_S  BP7644_S
AFFX-BioB-5_at  "Low"    "Low"   "Medium" "Medium" "Low"   "Medium" "High"
AFFX-BioB-M_at  "Low"    "Low"   "Medium" "High"   "Low"   "Medium" "High"
AFFX-BioB-3_at  "Low"    "Low"   "Medium" "Medium" "Low"   "Medium" "High"
AFFX-BioC-5_at  "Low"    "Low"   "Medium" "Medium" "Low"   "Medium" "High"
AFFX-BioC-3_at  "Low"    "Low"   "Medium" "Medium" "Low"   "Low"    "High"
AFFX-BioDn-5_at "Low"    "Low"   "Medium" "Medium" "Low"   "Medium" "High"
AFFX-BioDn-3_at "Low"    "Low"   "Medium" "Medium" "Low"   "Low"    "High"
AFFX-CreX-5_at  "Low"    "Low"   "High"   "High"   "Low"   "Low"    "Medium"
AFFX-CreX-3_at  "Low"    "Low"   "High"   "High"   "Low"   "Low"    "High"
AFFX-DapX-5_at  "Medium" "Low"   "Medium" "High"   "Low"   "Medium" "Medium"
```

- STEP 3: Calculate Fuzzy Patterns.

This function calculates a Fuzzy Pattern for each category. To do this, a given percentage of the samples belonging to a category must have the same label ('Low', 'Medium' or 'High'). In other case, a NA value is assigned.

```
> fps <- calculateFuzzyPatterns(rmadataset, dvs, piVal, overlapping); fps[1:30,]
```

```
                         healthy APL   AML-inv AML-mono AML-other
AFFX-BioB-5_at           NA      NA    NA      NA       NA
AFFX-BioB-M_at           NA      NA    NA      NA       NA
AFFX-BioB-3_at           NA      NA    NA      NA       NA
AFFX-BioC-5_at           NA      NA    NA      NA       NA
AFFX-BioC-3_at           NA      NA    NA      NA       NA
AFFX-BioDn-5_at          NA      NA    NA      NA       NA
AFFX-BioDn-3_at          NA      NA    NA      NA       NA
AFFX-CreX-5_at           NA      NA    NA      NA       NA
AFFX-CreX-3_at           NA      NA    NA      NA       NA
AFFX-DapX-5_at           NA      NA    NA      NA       NA
AFFX-DapX-M_at           NA      NA    NA      NA       NA
AFFX-DapX-3_at           NA      NA    NA      NA       NA
AFFX-LysX-5_at           NA      NA    NA      NA       NA
AFFX-LysX-M_at           NA      NA    NA      NA       NA
AFFX-LysX-3_at           NA      NA    NA      NA       NA
AFFX-PheX-5_at           NA      NA    NA      NA       NA
AFFX-PheX-M_at           NA      NA    NA      NA       NA
AFFX-PheX-3_at           NA      NA    NA      NA       NA
AFFX-ThrX-5_at           NA      NA    NA      NA       NA
```

```
AFFX-ThrX-M_at               NA      NA    NA      NA       NA
AFFX-ThrX-3_at               NA      NA    NA      NA       NA
AFFX-TrpnX-5_at              NA      NA    NA      NA       NA
AFFX-TrpnX-M_at              NA      NA    NA      NA       NA
AFFX-TrpnX-3_at              NA      NA    NA      NA       NA
AFFX-HUMISGF3A/M97935_5_at   NA      NA    NA      NA       NA
AFFX-HUMISGF3A/M97935_MA_at  NA      NA    NA      NA       NA
AFFX-HUMISGF3A/M97935_MB_at  NA      NA    NA      NA       NA
AFFX-HUMISGF3A/M97935_3_at   NA      NA    NA      NA       NA
AFFX-HUMRGE/M10098_5_at      "Low"   "Low" NA      NA       NA
AFFX-HUMRGE/M10098_M_at      "Low"   NA    NA      NA       NA
```

```
> showFuzzyPatterns(fps, "healthy")[21:50]
```

```
  200021_at   200022_at 200023_s_at   200024_at 200025_s_at   200029_at
     "High"       "Low"       "Low"       "Low"       "Low"       "Low"
200031_s_at 200032_s_at 200036_s_at   200042_at   200043_at   200051_at
      "Low"       "Low"       "Low"       "Low"       "Low"       "Low"
200052_s_at   200054_at 200057_s_at 200060_s_at 200061_s_at 200062_s_at
      "Low"       "Low"       "Low"       "Low"       "Low"       "Low"
200063_s_at   200064_at 200077_s_at 200079_s_at 200080_s_at 200081_s_at
      "Low"       "Low"      "High"       "Low"      "High"       "Low"
200088_x_at 200089_s_at 200091_s_at 200092_s_at 200093_s_at 200094_s_at
      "Low"       "Low"       "Low"       "Low"       "Low"       "Low"
```

- STEP 4: Calculate Discriminant Fuzzy Pattern.

The Discriminant Fuzzy Pattern (DFP) includes those genes present in two or more FPs with different assigned labels.

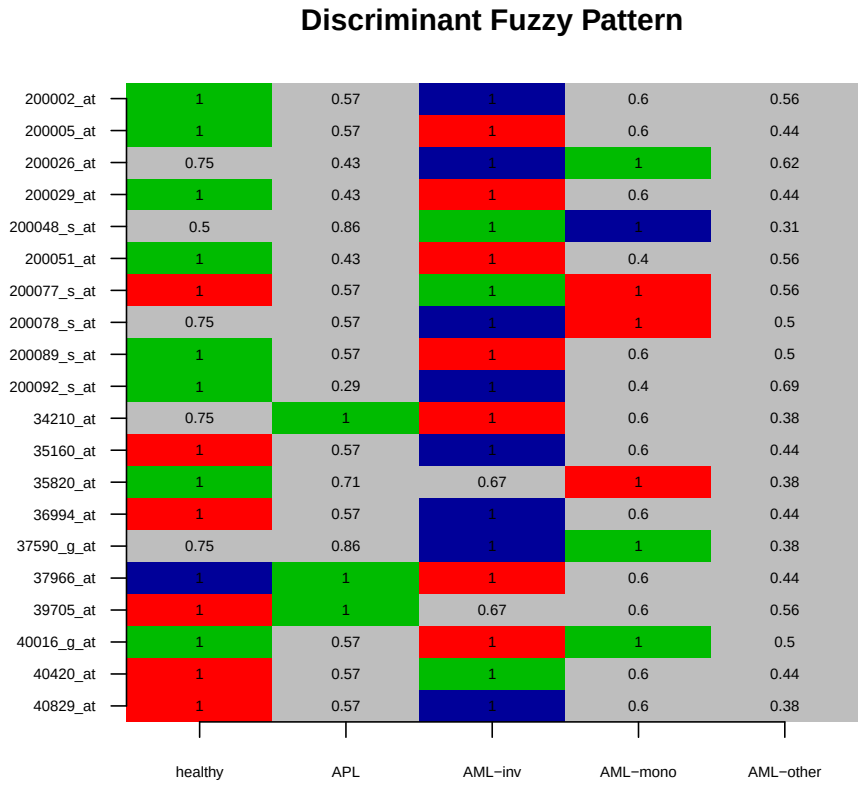
```
> dfps <- calculateDiscriminantFuzzyPattern(rmadataset, fps); dfps[1:5,]
```

```
           healthy APL AML-inv  AML-mono AML-other
200002_at  "Low"   NA  "Medium" NA       NA
200005_at  "Low"   NA  "High"   NA       NA
200026_at  NA      NA  "Medium" "Low"    NA
200029_at  "Low"   NA  "High"   NA       NA
200048_s_at NA     NA  "Low"    "Medium" NA
```

```
> plotDiscriminantFuzzyPattern(dfps, overlapping)
```

```
           healthy  APL   AML-inv  AML-mono AML-other
200002_at  "Low"    NA    "Medium" NA       NA
200005_at  "Low"    NA    "High"   NA       NA
200026_at  NA       NA    "Medium" "Low"    NA
200029_at  "Low"    NA    "High"   NA       NA
200048_s_at NA      NA    "Low"    "Medium" NA
200051_at  "Low"    NA    "High"   NA       NA
200077_s_at "High"  NA    "Low"    "High"   NA
200078_s_at NA      NA    "Medium" "High"   NA
200089_s_at "Low"   NA    "High"   NA       NA
200092_s_at "Low"   NA    "Medium" NA       NA
34210_at    NA      "Low" "High"   NA       NA
35160_at    "High"  NA    "Medium" NA       NA
```

```
35820_at    "Low"    NA    NA       "High"   NA
36994_at    "High"   NA    "Medium" NA       NA
37590_g_at  NA       NA    "Medium" "Low"    NA
37966_at    "Medium" "Low" "High"   NA       NA
39705_at    "High"   "Low" NA       NA       NA
40016_g_at  "Low"    NA    "High"   "Low"    NA
40420_at    "High"   NA    "Low"    NA       NA
40829_at    "High"   NA    "Medium" NA       NA
attr(,"ifs")
            healthy       APL   AML-inv AML-mono AML-other
200002_at      1.00 0.5714286 1.0000000      0.6    0.5625
200005_at      1.00 0.5714286 1.0000000      0.6    0.4375
200026_at      0.75 0.4285714 1.0000000      1.0    0.6250
200029_at      1.00 0.4285714 1.0000000      0.6    0.4375
200048_s_at    0.50 0.8571429 1.0000000      1.0    0.3125
200051_at      1.00 0.4285714 1.0000000      0.4    0.5625
200077_s_at    1.00 0.5714286 1.0000000      1.0    0.5625
200078_s_at    0.75 0.5714286 1.0000000      1.0    0.5000
200089_s_at    1.00 0.5714286 1.0000000      0.6    0.5000
200092_s_at    1.00 0.2857143 1.0000000      0.4    0.6875
34210_at       0.75 1.0000000 1.0000000      0.6    0.3750
35160_at       1.00 0.5714286 1.0000000      0.6    0.4375
35820_at       1.00 0.7142857 0.6666667      1.0    0.3750
36994_at       1.00 0.5714286 1.0000000      0.6    0.4375
37590_g_at     0.75 0.8571429 1.0000000      1.0    0.3750
37966_at       1.00 1.0000000 1.0000000      0.6    0.4375
39705_at       1.00 1.0000000 0.6666667      0.6    0.5625
40016_g_at     1.00 0.5714286 1.0000000      1.0    0.5000
40420_at       1.00 0.5714286 1.0000000      0.6    0.4375
40829_at       1.00 0.5714286 1.0000000      0.6    0.3750
```



This function plots the Discriminant Fuzzy Pattern of the relevant genes (in rows) for the sample classes (in columns), as well as the impact factor (`ifs`) which determines if a gene belongs to a Fuzzy Pattern in a class (if its value is higher than the `piVal` parameter). In the first table, a `NA` value is shown if the impact factor is lower or equal than the `piVal` parameter, which points out that the corresponding gene does not pertain to the Fuzzy Pattern of the class.

The relevant genes are those which are present in at least two different Fuzzy Patterns with different linguistic labels.

The plotting in graphical mode, shows the liguistic labels as follows:

- 'Low': colour BLUE.
- 'Medium': colour GREEN.
- 'High': colour RED.

4 Parameter Settings

- The `skipFactor` parameter can take values in the interval [0,). The default value is 3. Higher values imply that less samples of a gene are considered as odd (skipped in the test).
- The `zeta` parameter is the threshold value which controls the activation of a linguistic label ('Low', 'Medium' or 'High'). It can take values in the interval [0,1].

- The `overlapping` parameter can take the following values:
  1. The algorithm will use the discrete values (labels) 'Low', 'Medium' and 'High'.
  2. The algorithm will use the discrete values (labels) 'Low', 'Low-Medium', 'Medium', 'Medium-High' and 'High'.
  3. The algorithm will use the discrete values (labels) 'Low', 'Low-Medium', 'Low-Medium-High', 'Medium', 'Medium-High' and 'High'.
- The `piVal` parameter controls the degree of exigency for selecting a gene as a member of the pattern, since the higher its value, the fewer the number of genes which make up the puttern. It can take values in the interval [0,1]. The default value is 0.9.

5 Session Information

The version number of R and packages loaded for generating the vignette were:

- R version 3.3.0 RC (2016-04-26 r70550), `x86_64-apple-darwin13.4.0`
- Locale: `C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: Biobase 2.32.0, BiocGenerics 0.18.0, DFP 1.30.0
- Loaded via a namespace (and not attached): tools 3.3.0

6 References

1. F. Diaz, F. Fdez-Riverola, D. Glez-Pena, J. M. Corchado. Using Fuzzy Patterns for Gene Selection and Data Reduction on Microarray Data. 7th International Conference on Intelligent Data Engineering and Automated Learning: IDEAL 2006, (2006) pp. 1095-1102.
2. F. Fdez-Riverola, F. Diaz, J. M. Corchado, J. M. Hernandez, J. San Miguel: Improving Gene Selection in Microarray Data Analysis using Fuzzy Patterns inside a CBR System. Proc. of the ICCBR 2005 Conference, (2005) 23-26.
3. F. Diaz, F. Fdez-Riverola, J. M. Corchado: GENE-CBR: a Case-Based Reasoning Tool for Cancer Diagnosis using Microarray Datasets. Computational Intelligence (2006) 22(3-4):254-268.