Dipartimento di Statistica
"Giuseppe Parenti"

WORKING PAPER 2011/02

The assessment of mixed and intermediate *foci* images using the R software environment

C. Procaccianti, C. Urani,
F. M. Stefanini

Università degli Studi
di Firenze

Dipartimento di Statistica "G. Parenti" – Viale Morgagni 59 – 50134 Firenze - www.ds.unifi.it

The assessment of mixed and intermediate *foci* images using the R software environment.

Procaccianti C.[1], Urani C.[1], Stefanini F.M.[2]

[1]*Dipartimento di Scienze dell'Ambiente e del Territorio (DISAT)*
Università degli Studi di Milano-Bicocca, Milano, Italia

[2]*Dipartimento di Statistica "G.Parenti"*
Università degli Studi di Firenze, Firenze, Italia

Address for correspondence:
Federico M. Stefanini, Dipartimento di Statistica "G.Parenti",
Università degli Studi di Firenze, v.le Morgagni 59,
50134, Firenze (FI), Italia
email: stefanini@ds.unifi.it

March 23, 2011

Abstract

Fast and reliable screening of the carcinogenic potential of a chemical compound can be performed using in vitro methods such as the cell transformation assay (CTA), in which selected cell lines grow under different treatment conditions (with/without chemical under evaluation) and colonies (*foci*) formed at the end of the assay are scored and classified by light microscopy. While *foci* can mostly be divided into three canonic classes (Type I, II and III), often more undefined phenotypes can be spotted, resulting in an uncertain class attribution (mixed and intermediate). Here, we describe the R code developed to calculate a quantitative dissimilarity index and classify mixed or intermediate *foci* by exploiting the quantitative information provided by digital images of *foci* colonies.

Keywords: *Cell transformation assay, cluster analysis, carcinogenic potential, unsupervised learning*

Contents

1 Introduction 3

2 Setting up the R environment 3

3 Preprocessing 4
3.1 Step 1. Building the function 4
3.1.1 Exemple of descriptors extraction 4
3.2 Step 2. Creating the database 6

4 Clustering 8
4.1 Clustering in F1 8
4.1.1 F1 clustering 8
4.1.2 F1 confusion matrix 9
4.2 Clustering in F2 10
4.2.1 Dividing the database 10
4.2.2 MN and T1 clustering 10
4.2.3 T2 and T3 clustering 11
4.2.4 Obtaining the 4x4 confusion matrix 11

5 Merging *tiles* into parent images 12
5.1 Merging in F1 12
5.1.1 Grouping the classification scores in F1 12
5.1.2 Applying the majority of voting criterion 14
5.1.3 F1 parent images confusion matrix 14
5.2 Merging in F2 15
5.2.1 Grouping the classification scores in F2 15
5.2.2 Majority of voting criterion extraction modified 16
5.2.3 F2 parent images confusion matrix 17

6 The quantitative index 18
6.1 Calculating the centers of the canonical classes 18
6.2 Assessing the distances 19
6.3 Quantitative Index of Dissimilarity (QIoD) in graphics 22

1 Introduction

The evaluation of the carcinogenic potential of chemical compounds is of great concern in public health. The 2 year *in vivo* rodent bioassay is the standard for assessment of the carcinogenic potential of chemicals. Though preliminar, fast and reliable screening tests are of great interest (1). *In vitro* methods, such as the cell transformation assay (CTA), are important tools for both research and screening of chemical compounds. Cell transformation assays are based on selected cell lines growing under different treatment conditions (with/without chemical under evaluation). At the end of the assay, the formed colonies (*foci*) are scored and classified by a trained expert by light microscopy. *Foci* are then divided into three classes called Type I, Type II and Type III on the basis of morphological features (e.g. multilayering, polarization, criss-crossing of the cells) (2). While Type I *foci* are composed of non-transformed cells, both Type II and Type III are transformed and induce tumors if inoculated in animals (3). Though in most cases *foci* fall into this well defined categories, often more undefined morphologies are found. In fact, colonies presenteing features intermediate between two classes (intermediate *foci*) and colonies presenting features belonging to two different classes in the same time (mixed *foci*) are both documented, though uncommon. This morphologically undefined colonies result in an uncertain class attribution, which may in turn generate an under/over-estimation of the carcinogenic potential of a chemical compound.
In a previous work (4) we defined a new index of dissimilarity which is calculated on digital *foci* images, and which can be exploited for classifying mixed or intermediate *foci*. Here, we describe the R code developed to calculate this quantitative dissimilarity index.

2 Setting up the R environment

The packages that will be needed for the descriptors extraction and images classification have to be loaded into the R environment. A `Startup` function that will load the "`biOps`", "`fBasics`" and "`cluster`" libraries will be built and then ran.

```
> Startup <- function() {
+     library(biOps)
+     library(fBasics)
+     library(cluster)
+ }
```

```
> Startup()
```

```
 [1] "cluster"    "fBasics"    "timeSeries" "timeDate"   "MASS"
 [6] "biOps"      "stats"      "graphics"   "grDevices"  "utils"
[11] "datasets"   "methods"    "base"
```

3 Preprocessing

In this section, the function needed to extract the image descriptors from an image's gray levels distributions will be described. An example on how to apply this function to an image will also be provided.

3.1 Step 1. Building the function

The `DescrExtr` function is compiled to extract from an image the following descriptors:

- mean
- skewness
- kurtosis
- the 16 central vigintiles (quantiles of m/20 order, spanning from Q15 to Q85)
- the Canny's edge enhancement derived index, calculated as the number of enhanced (edgy) pixels over the image

```
> DescrExtr <- function(immagine) {
+     myQ <- seq(0.15, 0.85, by = 0.05)
+     x <- readTiff(immagine)
+     x <- imgRGB2Grey(x)
+     x <- imgNormalize(x)
+     x <- imgBlur(x)
+     meanX <- mean(c(x))
+     skX <- skewness(c(x))
+     kuX <- kurtosis(c(x))
+     quX <- quantile(x, probs = myQ)
+     canX <- imgCanny(x, 0.5)
+     cnX <- length(which(canX == 0))
+     tmpX <- c(meanX, skX, kuX, quX, cnX)
+     names(tmpX) <- c("Mean", "Skewness", "Kurtosis",
+         "Q15", "Q20", "Q25", "Q30", "Q35", "Q40",
+         "Q45", "Q50", "Q55", "Q60", "Q65", "Q70",
+         "Q75", "Q80", "Q85", "CannyDerived")
+     return(tmpX)
+ }
```

3.1.1 Exemple of descriptors extraction

In this part, the `DescrExtr` function will be applied to a given image *tile* (*Figure 1*).

The 512x512 image shown in *Figure 1* is a *tile* generated from a 1024x1024 image, hereon referred to as parent image (*Figure 2*). We will work on the assumption that maximum 4 tiles will be generated for a single parent image.

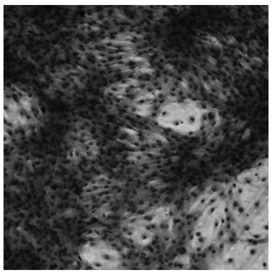

Figure 1: 512x512 *tile* (resized)

In fact, more than 4 512x512 tiles will only represent an over-sampling of the 1024x1024 original image.

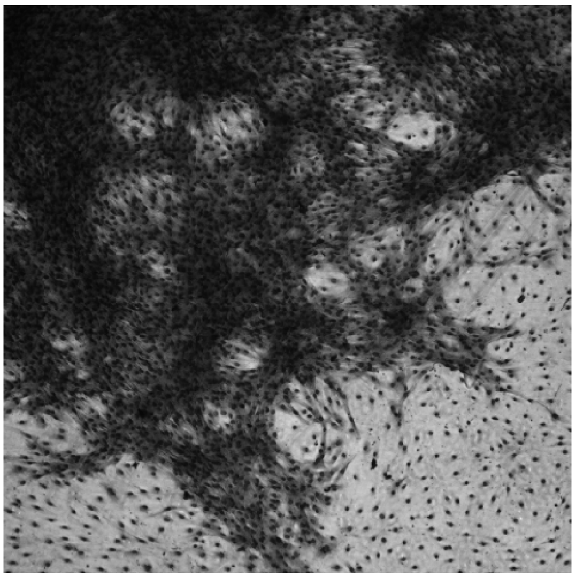

Figure 2: 1024x1024 parent image (resized)

The `DescrExtr` function is used to extract the image's descriptors from the processed *tile* the gray levels distribution (Figure 3).

Finally, the function will build a vector out of the extracted descriptors.

```
> Esempio <- DescrExtr("7[c]05-piastra 3 tipo III luce 7 p_A.tif")

         Mean     Skewness     Kurtosis          Q15
7.111242e+01 1.081049e+00 2.917967e-01 2.200000e+01
          Q20          Q25          Q30          Q35
2.700000e+01 3.100000e+01 3.500000e+01 3.900000e+01
          Q40          Q45          Q50          Q55
4.400000e+01 4.900000e+01 5.500000e+01 6.100000e+01
          Q60          Q65          Q70          Q75
6.800000e+01 7.600000e+01 8.500000e+01 9.700000e+01
```

```
         Q80          Q85 CannyDerived
1.120000e+02 1.320000e+02 2.131200e+04
```

3.2 Step 2. Creating the database

Following the described procedure, we applied the `DescrExtr` function to the image database. All the *tiles* were collected in the project folder. The used database featured:

- 82 parent images of canonical morphology (MD class)
- 20 parent images of unconventional morphology (MU class)
- 196 *tiles* generated from the MD parent images (from number 1 to number 196)
- 52 *tiles* generated from the MU parent images (from number 197 to number 248)

```
> Images <- dir()
> DescrTable <- DescrExtr(Images)
```

A `parent` object was created and the names of the corresponding parent image for each *tile* were stored. This vector was then added to the `DescrTable` table. This step is needed in order to keep the linkage between *tiles* and generating parent images.

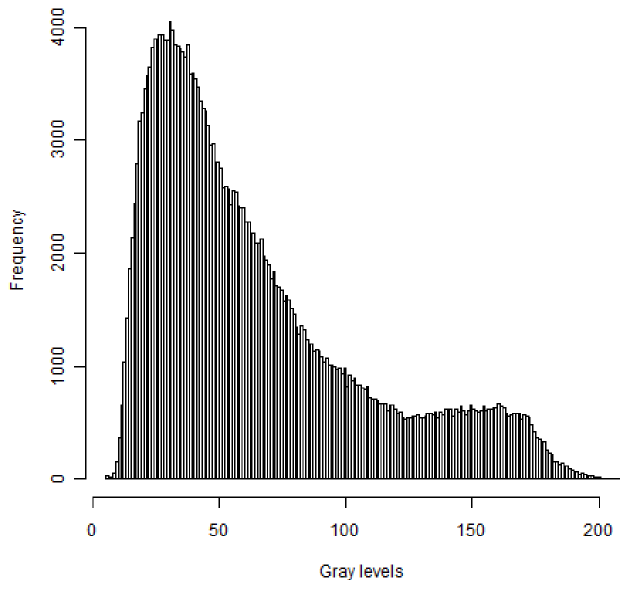

Figure 3: Graphic of the gray levels distribution for the 512x512 *tile*

```
> DescrTable <- cbind(DescrTable, parent)
```

The first 3 records are shown for exemplificative purposes. Row names of the generated table represent the *tiles* names.

```
                      Mean  Skewness Kurtosis Q15 Q20 Q25
1[c]01-MN1_A.tif 193.2694 -1.458412 2.995726 167 175 181
1[c]01-MN1_B.tif 191.8352 -1.665833 3.624617 163 173 180
1[c]01-MN1_C.tif 201.0577 -1.737660 3.632982 172 183 190
                 Q30 Q35 Q40 Q45 Q50 Q55 Q60 Q65 Q70 Q75
1[c]01-MN1_A.tif 186 190 194 197 199 202 205 208 210 214
1[c]01-MN1_B.tif 185 190 193 197 199 202 205 208 211 214
1[c]01-MN1_C.tif 196 200 204 207 210 213 214 217 220 223
                 Q80 Q85 Canny     Parent
1[c]01-MN1_A.tif 216 221 12965 1[c]01-MN1
1[c]01-MN1_B.tif 215 220 15092 1[c]01-MN1
1[c]01-MN1_C.tif 226 229 14170 1[c]01-MN1
```

Finally, all the expert scoring for the parent images were included in the `ExpClass` object. This vector will **exclusively** be used for evaluation of agreement purposes. Therefore, the *unsupervised* classification is ensured. Expert classification scores are present for MD images only.

```
> ExpClass

  [1] MN MN MN MN MN MN MN MN MN MN MN MN MN MN MN MN MN T2
 [19] T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 MN T1
 [37] T1 T1 T1 T1 T1 T1 T1 T2 T2 T2 T2 T2 T3 T3 T3 T3 T3 T3
 [55] T3 T3 T3 T3 T3 T3 T3 T3 T3 T3 T3 T3 T3 MN MN T1 T1 MN
 [73] MN MN MN MN MN MN T1 T1 T1 T2 T2 T2 T2 T2 T2 T2 T3 T3
 [91] T3 T3 T3 T3 T3 T3 T3 T3 T1 T1 T1 T1 T1 T1 MN MN MN MN
[109] T1 T1 T1 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T2
[127] T2 T2 T2 T2 T2 T2 T2 T2 T2 T2 T3 T3 T3 T3 T3 T3 T3 T3
[145] T3 T3 T3 T3 T3 T3 T3 T3 T3 T3 T3 T3 MN MN MN MN T1 T1
[163] T1 T1 T1 T1 T1 T1 MN MN MN MN MN MN MN MN MN MN MN MN
[181] MN MN MN MN MN MN MN T1 T1 T1 T1 T1 T1 T1 T1 T1
Levels: MN T1 T2 T3
```

4 Clustering

4.1 Clustering in F1

The classification scheme applied in this work is based on a hierarchical classification approach as described elsewhere in detail (4-6). Briefly, we developed a two-steps classification model:

- in the first step of classification (hereon, F1) we will divide the *tiles* images in two classes, corresponding to the transformed/untransformed biological division (2,3).

- in the second step of classification (F2) we will further divide the two F1 obtained classes in two sub-classes each, corresponding to the canonical *foci* classification. See **section 4.2** for further details on this classification step.

Both F1 and F2 steps of classification are based on the PAM clustering algorithm (7).

4.1.1 F1 clustering

The PAM algorithm was applied to the `DescrTable` table generated in the Preprocessing step (**section 3**). In this step the dataset will be divided in two classes using a PAM `k` paramenter of 2.

- The C1 (score of 1) class corresponding to normal monolayer or untransformed *foci*(noted as "NT")
- The C2 (score of 2) class corresponding to transformed *foci* (noted as "T")

The Canny's derived index (contained in the column 19) was not exploited in this step of classification. Moreover, the parent images scoring (column 20) was excluded to ensure the unsupervisedness of the classification process.

```
> clus512_F1_k2 <- pam(DescrTable[, -c(19, 20)], k = 2)
```

The obtained `clus512_F1_k2` object contains the classification vector for F1:

```
> as.numeric(clus512_F1_k2$cluster)
```

```
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 2 2 2 2 2 2 2 1 1 1
 [38] 1 1 1 1 1 1 1 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 1 1 1 1 1 1 1
 [75] 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1
[112] 2 2 2 2 1 2 2 2 2 2 1 1 2 2 2 1 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2
[149] 2 2 2 2 1 2 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[186] 1 1 2 1 1 1 1 1 1 1 1 2 1 2 2 2 2 2 2 2 1 1 2 1 1 2 2 1 1 2 2 2 2 2 2 2 2
[223] 2 2 2 2 2 1 2 2 2 1 1 1 1 2 1 1 1 1 1 2 1 2 1 2 2 2
```

4.1.2 F1 confusion matrix

In order to evaluate the performances of the classifier in terms of agreement of judgment with the expert and of sensitivity and specificity, a confusion matrix was built for each step of classification. The matrices featured:

- by **rows**: the expert scoring
- by **columns**: the algorithm scoring
- by **diagonal**: images (*tiles*) where the expert and the algorithm agree on.

The confusion matrix will be built around the images that are of the MD class (*tiles* 1 to 196), while images of mixed and intermediate nature (MU) will be excluded (*tiles* 197 to 248).

```
> confusion512_F1_k2 <- table(ExpClass, clus512_F1_k2$clustering[1:196])
```

```
ExpClass  1  2
      MN 54  0
      T1 37  2
      T2 16 38
      T3  4 45
```

The total number of *tiles* of the MD class scored as C1 is 111, while *tiles* scored as C2 are a total of 85. Moreover, while the automatic classifier is set to divide the images in 2 classes (C1 and C2), the expert scoring is on based on 4 classes (MN and the three canonical *foci* classes). For the sake of comparison of results, we will transform the confusion matrix from a 4x2 matrix in a 2x2, by modifing the `ExpClass` object transforming all the "MN" and "T1" instances in "NT" (untransformed), and the "T2" and "T3" in "T" (transformed).
The obtained `ExpClass_F1` vector is shown.

```
> ExpClass_F1
```

```
  [1] NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT T  T  T  T  T  T  T  T
 [26] T  T  T  T  T  T  T  T  T  NT NT NT NT NT NT NT NT NT T  T  T  T  T  T  T
 [51] T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  NT NT NT NT NT NT NT NT
 [76] NT NT NT NT NT NT T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  NT NT
[101] NT NT NT NT NT NT NT NT NT NT NT T  T  T  T  T  T  T  T  T  T  T  T  T  T
[126] T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T  T
[151] T  T  T  T  T  T  NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT
[176] NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT NT
Levels: NT T
```

The modified confusion matrix will be as follows:

```
> confusions512_F1_k2_2x2 <- table(ExpClass_F1, clus512_F1_k2$clustering[1:196])
```

```
ExpClass_F1  1  2
         NT 91  2
         T  20 83
```

Agreement, sensitivity and specificity values calculated on the 2x2 matrix are reported.

Summary	Value
Agreement	88.78%
Sensitivity	97.85%
Specificity	80.58%

4.2 Clustering in F2

During the F1 step of classification, the *tiles* were divided in 2 classes (C1 and C2), corresponding to the biological classes of transformed and untransformed *foci*. Though biologically highly relevant, this classification is only partial. Therefore, during the F2 step of classification, the two F1 clusters were further divided in two classes each, so to mimick the division of the untransformed images in normal monolayer and Type I, and of the transformed images in Type II and Type III.
In this step of classification, we also exploited the Canny's edge enhancement algorithm derived descriptor.

4.2.1 Dividing the database

The classification vector obtained in F1 (`clus512_F1_k2$clustering`) was used to divide the database in two blocks:

- `DescrTable_C1` containing the C1 scored *tiles* (111 MD, 19 MU)
- `DescrTable_C2` containing the C2 scored *tiles* (85 MD, 33 MU)

```
> estrC1 <- which(clus512_F1_k2$cluster == 1)
> DescrTable_C1 <- DescrTable[estrC1, ]
> DescrTable_C2 <- DescrTable[-estrC1, ]
```

4.2.2 MN and T1 clustering

The PAM algorithm was applied to each of the two blocks separately, each time with `k`=2. First, the `DescrTable_C1` database was processed.

```
> clus512_F2_k2_MNT1 <- pam(DescrTable_C1[, -20],
+     2)
```

The expert scoring for the MD images classified as C1 during F1 was extracted from the `ExpClass` object. The obtained vector was used to build the partial 4x2 confusion matrix on the 111 MD *tiles*. Column names were set as "C'1" and "C'2".

```
> confusioni512_k2_F2_MNT1 <- table(ExpClass_C1,
+     clus512_F2_k2_MNT1$clustering[1:111])
> colnames(confusioni512_k2_F2_MNT1) <- c("C'1",
+     "C'2")
```

```
ExpClass_C1 C'1 C'2
         MN  52   2
         T1   1  36
         T2   2  14
         T3   2   2
```

4.2.3 T2 and T3 clustering

The approach used for MN and T1 classification was repeated to cluster the *tiles* contained in the `DescrTable_C2` table.

```
> DescrTable_C2 <- DescrTable[-estrC1, ]
> clus512_F2_k2_T2T3 <- pam(DescrTable_C2[, -20],
+     2)
```

The expert scoring for the C2 images was extracted from the `ExpClass` object, and the classification matrix built. Column names were set as "C'3" and "C'4".

```
> confusioni512_k2_F2_T2T3 <- table(ExpClass_C2,
+     clus512_F2_k2_T2T3$clustering[1:85])
> colnames(confusioni512_k2_F2_T2T3) <- c("C'3",
+     "C'4")
```

```
ExpClass_C2 C'3 C'4
         MN   0   0
         T1   1   1
         T2  31   7
         T3   4  41
```

4.2.4 Obtaining the 4x4 confusion matrix

Finally, the two partial 4x2 matrices were joined to obtain the complete F2 4x4 confusion matrix, on which agreement, sensitivity and specificity were calculated.

```
   C'1 C'2 C'3 C'4
MN  52   2   0   0
T1   1  36   1   1
T2   2  14  31   7
T3   2   2   4  41
```

Summary	Value
Agreement	81.63%
Sensitivity	82.42%
Specificity	94.01%

5 Merging *tiles* into parent images

5.1 Merging in F1

Following the *tiles* classification, we approached the problem of scoring the parent images. In fact, while the scores we obtained in both F1 and F2 are relative to 512x512 *tiles*, the expert classification we used for comparison is relative to the parent images. Therefore, a majority of voting criterion was applied to assess a parent image's score based on the scores of its derived *tiles*. In case of draw, higher class was chosen.

5.1.1 Grouping the classification scores in F1

A `Voti` function was built to divide the classification *tiles* scores contained in a generic `y` table presenting in column 1 the parent images names and in column 2 the actual scores, grouping them on the basis of the generating parent images number ($n =$ `x`)

```
> Voti <- function(x, y) {
+     matVoti <- matrix(0, x, 4)
+     rownames(matVoti) <- rep(0, x)
+     auxI <- 1
+     auxII <- 1
+     for (aux in 1:nrow(y)) {
+         if (aux < nrow(y)) {
+             if (y[aux, 1] == y[aux + 1, 1]) {
+                 matVoti[auxI, auxII] <- y[aux, 2]
+                 auxII <- auxII + 1
+             }
+             if (y[aux, 1] != y[aux + 1, 1]) {
+                 matVoti[auxI, auxII] <- y[aux, 2]
+                 rownames(matVoti)[auxI] <- y[aux, 1]
+                 auxI <- auxI + 1
+                 auxII <- 1
+             }
+         }
+         if (aux == nrow(y)) {
+             if (y[aux, 1] == y[aux - 1, 1]) {
+                 matVoti[auxI, auxII] <- y[aux, 2]
+                 auxII <- auxII + 1
+                 rownames(matVoti)[auxI] <- y[aux, 1]
+             }
+             if (y[aux, 1] != y[aux - 1, 1]) {
+                 matVoti[auxI, 1] <- y[aux, 2]
+                 rownames(matVoti)[auxI] <- y[aux, 1]
+             }
+         }
+     }
+     matVoti
+ }
```

The built function was then applied to our database. The list of the parent images was acquired from the column 20 of the `DescrTable` object created during **section 3**.

```
                    Mean  Skewness Kurtosis Q15 Q20 Q25 Q30 Q35 Q40 Q45 Q50
1[c]01-MN1_A.tif 193.2694 -1.458412 2.995726 167 175 181 186 190 194 197 199
1[c]01-MN1_B.tif 191.8352 -1.665833 3.624617 163 173 180 185 190 193 197 199
1[c]01-MN1_C.tif 201.0577 -1.737660 3.632982 172 183 190 196 200 204 207 210
                 Q55 Q60 Q65 Q70 Q75 Q80 Q85 Canny     Parent
1[c]01-MN1_A.tif 202 205 208 210 214 216 221 12965 1[c]01-MN1
1[c]01-MN1_B.tif 202 205 208 211 214 215 220 15092 1[c]01-MN1
1[c]01-MN1_C.tif 213 214 217 220 223 226 229 14170 1[c]01-MN1
```

The `clus512_F1_k2$clustering` F1 clustering vector generated in **section 4.1.1** will be exploited in this classification step. For clarity's sake, we remind that a vote of "1" represents untransformed images ("NT"), while a vote of "2" represents transformed images ("T").

```
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 1 1 1 2 2 2 2 2 2 2 2 2 1 1 1
 [38] 1 1 1 1 1 1 1 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1
 [75] 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1
[112] 2 2 2 2 1 2 2 2 2 2 1 1 2 2 2 1 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2
[149] 2 2 2 2 1 2 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[186] 1 1 2 1 1 1 1 1 1 1 1
```

For the 196 MD derived *tiles*, the parent images names (column 1) and the scores of classification (column 2) were united in the `SchedaVoti` table.

```
> SchedaVoti <- cbind(as.character(DescrTable[1:196, 20]), clus512_F1_k2$clustering[1:196]
```

Finally, the `Voti` function was applied to the `SchedaVoti` table.

- x=82 (number of parent images)
- y=`SchedaVoti`
- "0" characters are used as neutral characters when less than 4 *tiles* were extrated from a single parent image

The generated table will feature:

- by **rows** the parent images
- by **columns** the corresponding *tiles* scores

```
> F1_1024x1024 <- Voti(82, SchedaVoti)
```

The first 10 records are shown for exemplificative purposes.

```
                   [,1] [,2] [,3] [,4]
1[c]01-MN1         "1"  "1"  "1"  "1"
1[c]02-MN2         "1"  "1"  "1"  "1"
2[c]02-MN1piastra1 "1"  "1"  "1"  "0"
2[c]03-MN2piastra1 "1"  "1"  "0"  "0"
```

```
2[c]04-MNpiastra1       "1"  "1"  "1"  "1"
2[c]05-Tipo II_bordo_f1 "1"  "1"  "0"  "0"
2[c]07-Tipo II_bordo_f2 "2"  "1"  "0"  "0"
2[c]08-Tipo II_bordo_f3 "1"  "2"  "1"  "0"
2[c]09-Tipo II_bordo_f4 "1"  "1"  "2"  "0"
```

5.1.2 Applying the majority of voting criterion

The `MoV` function was built to extract a vector containing the majority of voting for each parent image from the table generated in **section 4.1.1**. In case of draw (e.g. if 2 "NT" *tiles* and 2 "T" *tiles* were generated from the same parent image), the highest corresponding biological damage class was chosen.

```
> MoV <- function(Scheda_MoV) {
+     mov <- c(NULL)
+     for (aux in 1:nrow(Scheda_MoV)) {
+         L1 <- length(which(Scheda_MoV[aux, ] == 1))
+         L2 <- length(which(Scheda_MoV[aux, ] == 2))
+         ifelse(L2 >= L1, mov <- c(mov, 2), mov <- c(mov, 1))
+     }
+     mov
+ }
```

Finally, the `MoV` function was applied to the `F1_1024x1024` table. The obtained `clus_F1_1024` vector is the clustering vector for parent images in F1.

```
> clus_F1_1024 <- MoV(F1_1024x1024)
```

```
 [1] 1 1 1 1 1 1 2 1 1 2 2 2 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2
[39] 2 2 1 1 1 1 1 1 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 1 1 1 1 1 1
[77] 1 2 1 1 1 1
```

5.1.3 F1 parent images confusion matrix

In order to obtain the confusion matrix, the expert scores for the parent images were copied in an `Exp_parent_F1` vector.

- untransformed images are scored as "**1**"
- transformed images are scored as "**2**"

```
 [1] 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 1 1 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2
[39] 2 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1
[77] 1 1 1 1 1 1
```

Finally, the confusion matrix was obtained, featuring the expert scores (by row) versus the parent images scores obtained through the majority of voting for F1 *tiles* scoring (by column).

```
> confusioni1024_F1_k2_2x2 <- table(Exp_parent_F1, clus_F1_1024)
```

```
             clus_F1_1024
Exp_parent_F1  1  2
            1 35  2
            2  5 40
```

Agreement of classification, sensitivity and specificity are as follows.

Summary	Value
Agreement	91.46%
Sensitivity	94.59%
Specificity	88.89%

5.2 Merging in F2

The procedure described in the previous section was applied to the classification scores obtained in F2 in **section 4.2** in order to obtain the F2 scores for parent images.

5.2.1 Grouping the classification scores in F2

A `DescrTable_Ext` table was built starting from the `DescrTable` table created in **section 3**, by adding a column (column 21) containing the classification scores obtained in F2. Classification scores will be extracted from the `DescrTable_C1` and `DescrTable_C2` tables and modified:

- C'1 images are flagged as 1 (from the `clus512_F2_k2_MNT1$cluster` vector)
- C'2 images are flagged as 2 (from the `clus512_F2_k2_MNT1$cluster` vector)
- C'3 images are flagged as 3 (from the `clus512_F2_k2_T2T3$cluster` vector)
- C'4 images are flagged as 4 (from the `clus512_F2_k2_T2T3$cluster` vector)

```
> DescrTable_Ext <- cbind(DescrTable, rep(0, nrow(DescrTable)))
> colnames(DescrTable_Ext)[21] <- "Voti_F2"
> for (aux in 1:nrow(DescrTable_C1)) {
+     tmp <- which(rownames(DescrTable) == rownames(DescrTable_C1[aux,
+         ]))
+     DescrTable_Ext[tmp, 21] <- as.numeric(clus512_F2_k2_MNT1$cluster[aux])
+ }
> for (aux in 1:nrow(DescrTable_C2)) {
+     tmp <- which(rownames(DescrTable) == rownames(DescrTable_C2[aux,
+         ]))
+     DescrTable_Ext[tmp, 21] <- as.numeric((2 + clus512_F2_k2_T2T3$cluster[aux]))
+ }
```

The obtained classification vector is shown.

```
> DescrTable_Ext[1:196, 21]
```

```
  [1] 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 2 2 3 2 2 3 2 2 2 3 3 3 3 3 3 3 3 1 2 2
 [38] 1 2 2 2 2 2 2 2 3 3 2 4 4 4 4 4 4 4 4 4 4 4 1 4 3 4 4 4 4 2 1 1 2 2 1 1 2
 [75] 1 1 1 1 2 2 2 4 3 3 3 3 3 3 4 4 3 4 4 4 4 4 3 4 2 2 2 2 2 2 1 1 1 1 2 2 2
[112] 4 3 3 3 2 3 3 3 3 3 2 2 3 4 4 1 4 3 3 4 3 1 4 2 3 4 3 4 4 4 4 4 2 4 4 4 4
[149] 4 4 4 4 1 4 4 4 1 1 1 1 3 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[186] 1 1 4 2 2 2 2 2 2 2 2
```

5.2.2 Majority of voting criterion extraction modified

In order to apply the `Voti` function generated in the previous section to divide the scores by parent images, we built a `SchedaVoti_F2` table featuring:

- the parent images names (`DescrTable_Ext` column 20)
- the scores of classification for F2 (`DescrTable_Ext` column 21)

```
> SchedaVoti_F2 <- DescrTable_Ext[1:196, 20:21]
```

The `Voti` function was then applied to the `SchedaVoti_F2` table, and the `F2_1024x1024` table obtained. The first 10 records are shown for exemplificative purposes.

```
> F2_1024x1024 <- Voti(82, SchedaVoti_F2)
```

```
   [,1] [,2] [,3] [,4]
1     1    1    1    1
2     1    2    1    1
4     1    1    1    0
5     1    1    0    0
6     1    1    1    1
7     2    2    0    0
8     3    2    0    0
9     2    3    2    0
10    2    2    3    0
```

The `MoV_F2` function represents a modified version of the `MoV` function, developed in order to extract the majority of voting out of four classes.

```
> MoV_F2 <- function(Scheda_MoV) {
+     mov <- c(NULL)
+     for (aux in 1:nrow(Scheda_MoV)) {
+         L1 <- length(which(Scheda_MoV[aux, ] == 1))
+         L2 <- length(which(Scheda_MoV[aux, ] == 2))
+         L3 <- length(which(Scheda_MoV[aux, ] == 3))
+         L4 <- length(which(Scheda_MoV[aux, ] == 4))
+         L1234 <- c(L1, L2, L3, L4)
+         MaxL1234 <- which(L1234 == max(L1234))
+         if (length(MaxL1234) == 1) {
+             mov <- c(mov, MaxL1234)
+         }
+         if (length(MaxL1234) != 1) {
+             mov <- c(mov, max(MaxL1234))
+         }
+     }
+     mov
+ }
```

The `MoV_F2` function was applied to the `F2_1024x1024` table.

```
> clus_F2_1024 <- MoV_F2(F2_1024x1024)
```

The obtained clustering vector for parent images in F2 is shown.

```
 [1] 1 1 1 1 1 2 3 2 2 3 3 3 1 2 2 2 2 3 4 4 4 4 4 4 4 4 4 1 1 2 1 1 2 2 3 3 4 4
[39] 4 4 2 2 2 2 1 2 3 3 3 3 2 3 4 4 3 4 4 3 4 4 4 4 4 4 4 4 4 1 3 2 2 2 1 1 1 1
[77] 1 4 2 2 2 2
```

5.2.3 F2 parent images confusion matrix

The expert classification for the parent images was copied in the `Exp_parent_F2` object with the following modifications:

- monolayer images were flagged as 1
- Type I *foci* images were flagged as 2
- Type II *foci* images were flagged as 3
- Type III *foci* images were flagged as 4

```
 [1] 1 1 1 1 1 3 3 3 3 3 3 3 1 2 2 2 3 3 4 4 4 4 4 4 4 4 4 1 1 2 1 1 2 2 3 3 4 4
[39] 4 4 2 2 2 2 1 2 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 1 2 2 2 2 1 1 1 1
[77] 1 2 2 2 2 2
```

The confusion matrix for this step of classification was obtained obtained by crossing the expert scores in `Exp_parent_F2` with the parent images scores obtained through the majority of voting for F2 *tiles* scoring contained in the `clus_F2_1024` vector.

```
             clus_F2_1024
Exp_parent_F2  1  2  3  4
            1 17  0  0  0
            2  0 18  1  1
            3  0  5 14  4
            4  0  0  0 22
```

Summary	Value
Agreement	86.58%
Sensitivity	87.72%
Specificity	95.48%

6 The quantitative index

In **section 4** we described the hierarchical classification of the MD *tiles*. In **section 5** we described the merging of the classified tiles to assess a classification score for the generating parent images. We evaluated the performances of each step by means of confusion matrices. In this last section we will describe how a quantitative index suitable for MU scoring was extracted from the MD images classification. In fact, due to their uncanonical morphology, mixed and intermediate (MU) *foci* can sometimes be of hard or doubtful classification (2,3). We suggest the use of the quantitative index to describe how different MU images are from canonical classes, thus indentifying the less dissimilar class (4).

6.1 Calculating the centers of the canonical classes

In order to assess the distance between the MU images and the canonical classes, the coordinates of each class center were calculated.
A `stdTab` function was built to standardize `y` columns in the descriptors table (x).

```
> stdTab <- function(x, y) {
+     tmp <- matrix(0, nrow(x), ncol(x))
+     for (aux in c(y)) {
+         tmpM <- sum(x[, aux])/nrow(x)
+         tmpV <- var(x[, aux])
+         for (auxI in 1:nrow(x)) {
+             tmp[auxI, aux] <- (x[auxI, aux] -
+                 tmpM)/tmpV
+         }
+     }
+     tmp <- cbind(tmp[, y], x[, -y])
+     colnames(tmp) <- colnames(x)
+     rownames(tmp) <- rownames(x)
+     tmp
+ }
```

The described function was applied to the `DescrTable_Ext` table, excluding the columns 20 and 21 due to their non-numerical nature.

```
> Descr_Std <- stdTab(DescrTable_Ext, c(1:19))
```

The first 3 records of the normalized table are shown for exemplificative purposes.

```
                        Mean  Skewness  Kurtosis        Q15
1[c]01-MN1_A.tif 0.03439151 -1.198333 0.4516872 0.03257787
1[c]01-MN1_B.tif 0.03354178 -1.400649 0.5651086 0.03100364
1[c]01-MN1_C.tif 0.03900589 -1.470708 0.5666173 0.03454565
                        Q20        Q25        Q30
1[c]01-MN1_A.tif 0.03051734 0.02912371 0.02822432
1[c]01-MN1_B.tif 0.02977477 0.02876278 0.02786610
1[c]01-MN1_C.tif 0.03348763 0.03237209 0.03180653
```

```
                         Q35        Q40        Q45
1[c]01-MN1_A.tif 0.02751783 0.02713246 0.02665857
1[c]01-MN1_B.tif 0.02751783 0.02676451 0.02665857
1[c]01-MN1_C.tif 0.03112865 0.03081205 0.03045085
                         Q50        Q55        Q60
1[c]01-MN1_A.tif 0.02604615 0.02583097 0.02571289
1[c]01-MN1_B.tif 0.02604615 0.02583097 0.02571289
1[c]01-MN1_C.tif 0.03040288 0.03041956 0.02971506
                         Q65        Q70        Q75
1[c]01-MN1_A.tif 0.02570930 0.02512342 0.02560046
1[c]01-MN1_B.tif 0.02570930 0.02565581 0.02560046
1[c]01-MN1_C.tif 0.03005059 0.03044736 0.03102482
                         Q80        Q85         Canny
1[c]01-MN1_A.tif 0.02449218 0.02473621 -8.013620e-05
1[c]01-MN1_B.tif 0.02379288 0.02391011 -6.694615e-05
1[c]01-MN1_C.tif 0.03148511 0.03134496 -7.266370e-05
                      Parent Voti_F2
1[c]01-MN1_A.tif 1[c]01-MN1       1
1[c]01-MN1_B.tif 1[c]01-MN1       1
1[c]01-MN1_C.tif 1[c]01-MN1       1
```

The *tiles* with the normalized descriptors were divided on the basis of their scores of classification as obtained in F2, thus creating four different tables, one for each cluster group (namely C'1, C'2, C'3 and C'4). For each table, the class center was calculated as the mean values for the 19 featured descriptors. The class center will act as a virtual paragon *tile* for each class, from which to calculate the distance for all the MU tiles.

```
> estr_MD_MN <- which(Descr_Std[1:196, 21] == 1)
> estr_MD_T1 <- which(Descr_Std[1:196, 21] == 2)
> estr_MD_T2 <- which(Descr_Std[1:196, 21] == 3)
> estr_MD_T3 <- which(Descr_Std[1:196, 21] == 4)
> tabMN <- Descr_Std[estr_MD_MN, ]
> tabT1 <- Descr_Std[estr_MD_T1, ]
> tabT2 <- Descr_Std[estr_MD_T2, ]
> tabT3 <- Descr_Std[estr_MD_T3, ]
```

The `ClassCenter` function was built to extract from a table (`x`) the mean value (thus, the class center) for the desired descriptors (`y`).

```
> ClassCenter <- function(x, y) {
+     tmp <- c(NULL)
+     for (aux in y) {
+         tmp <- c(tmp, mean(x[, aux]))
+     }
+     tmp
+ }
```

6.2 Assessing the distances

The four class centers were then united in a comprehensive table.

```
> centerMN <- ClassCenter(tabMN, c(1:19))
> centerT1 <- ClassCenter(tabT1, c(1:19))
> centerT2 <- ClassCenter(tabT2, c(1:19))
> centerT3 <- ClassCenter(tabT3, c(1:19))
> tabCenter <- rbind(centerMN, centerT1, centerT2,
+     centerT3)
> colnames(tabCenter) <- colnames(Descr_Std[, 1:19])
```

```
               Mean   Skewness    Kurtosis          Q15
centerMN  0.02982862 -1.3841773  0.57690088  0.028297083
centerT1  0.01246654 -0.2281752 -0.17059001  0.005524537
centerT2 -0.01632886  0.4061967 -0.25242681 -0.015053341
centerT3 -0.02747331  1.0254952 -0.08899969 -0.019901621
                  Q20          Q25          Q30         Q35
centerMN  0.026869618  0.025805680  0.025063183  0.02449647
centerT1  0.006645006  0.007646646  0.008522186  0.00935677
centerT2 -0.014377366 -0.013847130 -0.013408881 -0.01305369
centerT3 -0.020060952 -0.020434325 -0.020925017 -0.02154504
                Q40         Q45         Q50         Q55
centerMN  0.02409197  0.02377777  0.02359332  0.02344520
centerT1  0.01011099  0.01087848  0.01163374  0.01237422
centerT2 -0.01269901 -0.01238768 -0.01200922 -0.01167723
centerT3 -0.02220397 -0.02296610 -0.02380971 -0.02462643
                Q60         Q65         Q70         Q75
centerMN  0.02330222  0.02322977  0.02320867  0.02314733
centerT1  0.01308875  0.01382882  0.01450511  0.01513122
centerT2 -0.01134427 -0.01105767 -0.01097586 -0.01118140
centerT3 -0.02549860 -0.02635663 -0.02718158 -0.02780674
                Q80         Q85         Canny
centerMN  0.02265193  0.02131588 -8.204021e-05
centerT1  0.01558266  0.01569506  3.690068e-05
centerT2 -0.01214301 -0.01399846  3.528646e-05
centerT3 -0.02751239 -0.02589160 -6.682832e-05
```

A tabMU table was created, by joining the MU *tiles* (52, in our case) and the four class centers. The distances between the *tiles* and the class centers table were calculated in the tabDist table.

```
> tabMU <- rbind(Descr_Std[197:248, 1:19], tabCenter)
> tabDist <- as.matrix((dist(tabMU)))[53:56, -c(53:56)]
> colnames(tabDist) <- parent[197:248]
```

The first 3 records of table tabDist are shown for exemplificative purposes.

```
         2[c]01-Incerto_bordo_h 2[c]01-Incerto_bordo_h
centerMN              1.6554299              1.7447085
centerT1              0.2879688              0.3786066
centerT2              0.3683948              0.2831108
centerT3              1.0036536              0.9211005
         2[c]11-Tipo II_centro1_f
centerMN               1.98940693
```

```
centerT1               0.68120000
centerT2               0.07467855
centerT3               0.59183746
```

A `Distanze` function extracting from an `x` table the mean distances for each of the composing `y` parent MU images from the distances of the derived *tiles* was built.

```
> Distanze <- function(x, y) {
+     mat <- matrix(0, 4, 4)
+     auxI <- 1
+     auxII <- 1
+     vec <- matrix(0, 4, y)
+     colnames(vec) <- rep(0, y)
+     for (aux in 1:ncol(x)) {
+         if (aux != ncol(x)) {
+             if (colnames(x)[aux] == colnames(x)[aux +
+                 1]) {
+                 mat[, auxI] <- as.numeric(x[,
+                   aux])
+                 auxI <- auxI + 1
+             }
+             if (colnames(x)[aux] != colnames(x)[aux +
+                 1]) {
+                 mat[, auxI] <- as.numeric(x[,
+                   aux])
+                 for (auxIII in 1:4) {
+                   vec[auxIII, auxII] <- (sum(mat[auxIII,
+                     ])/length(which(mat[auxIII,
+                     ] != 0)))
+                 }
+                 colnames(vec)[auxII] <- colnames(x)[aux]
+                 auxI <- 1
+                 auxII <- auxII + 1
+                 mat <- matrix(0, 4, 4)
+             }
+         }
+         if (aux == ncol(x)) {
+             if (colnames(x)[aux] == colnames(x)[aux -
+                 1]) {
+                 mat[, auxI] <- as.numeric(x[,
+                   aux])
+                 for (auxIII in 1:4) {
+                   vec[auxIII, auxII] <- (sum(mat[auxIII,
+                     ])/length(which(mat[auxIII,
+                     ] != 0)))
+                 }
+                 colnames(vec)[auxII] <- colnames(x)[aux]
+             }
+             if (colnames(x)[aux] != colnames(x)[aux -
+                 1]) {
```

```
+                 vec[, auxII] <- as.numeric(x[,
+                   aux])
+             }
+             colnames(vec)[auxII] <- colnames(x)[aux]
+         }
+     }
+     vec
+ }
```

The Distanze function was applied to the tabDist table. The obtained distances are the mean distance between a parent image and the four class centers, and by extension, to the canonic classes themselves.

```
> DistMU <- Distanze(tabDist, 20)
```

The first 3 records of the DistMU table are shown for exemplificative purposes.

```
     2[c]01-Incerto_bordo_h 2[c]11-Tipo II_centro1_f
[1,]              1.7000692                1.9254976
[2,]              0.3332877                0.6051854
[3,]              0.3257528                0.0939840
[4,]              0.9623770                0.6716120
     2[c]15-Tipo II-incerto_bordo_h
[1,]                      2.1984268
[2,]                      0.8931625
[3,]                      0.2615870
[4,]                      0.3981819
```

6.3 Quantitative Index of Dissimilarity (QIoD) in graphics

The mean distances between the parent images and the class centers represent a quantitative value of how much each of the MU images is distant (thus, *dissimilar*) from a typical image belonging to one of the canonical classes (monolayer, Type I, Type II, Type III). The smaller the dissimilarity, the closer the image is to that canonical class.
A barchart graphic of the dissimilarity index was built featuring:

- on the **x axis** the ID of the MU parent images (from 1 to 20)
- on the **y axis** the correspoinding distances. Bars represent, from bottom to top in grayshades, classes MN, T1, T2 and T3 respectively.

By definition, the sum of the four distances was set to 1. The QiD function was built to achieve this last feature and applied to the DistMU table.

```
> QiD <- function(x) {
+     mat <- matrix(0, nrow(x), ncol(x))
+     for (aux in 1:ncol(x)) {
+         tmp <- sum(x[, aux])
+         mat[, aux] <- x[, aux]/tmp
+     }
+     mat
+ }
> QiD_MU <- QiD(DistMU)
```

An extract of the obtained table `QiD_MU` is shown.

```
          [,1]       [,2]       [,3]
MN 0.51183984 0.58414278 0.58603488
T1 0.10034292 0.18359654 0.23809043
T2 0.09807439 0.02851215 0.06973128
T3 0.28974285 0.20374853 0.10614341
```

The barchart for the `QiD_MU` table was built with the R basic function `barplot`.

```
> barplot(QiD_MU, names.arg = c(1:20), xlab = "Parent Images",
+     ylab = "Quantitative index of Dissimilarity (QiD)")
```

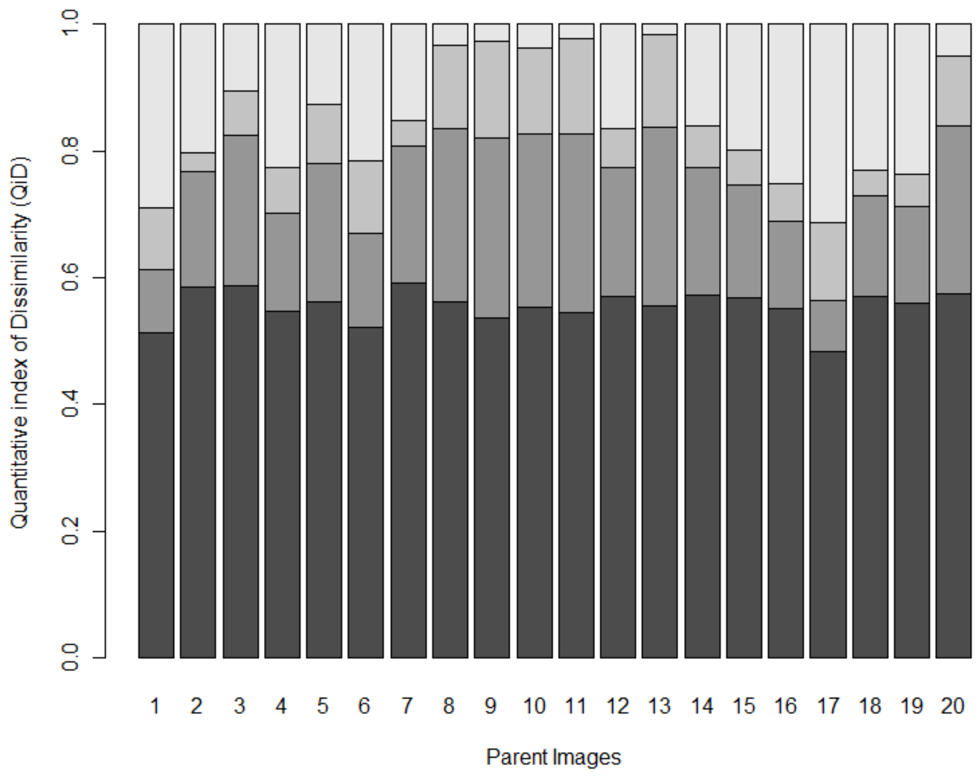

Figure 4: Barchart for the QIoD. In grayshades (from dark to light): MN, T1, T2 and T3

References

[1] OECD, 2007, *Detailed Review Paper on Cell Transformation Assays for Detection of Chemical Carcinogens. No. 31. Series on Testing and Assessment, 164pp. Paris, France: Organisation for Economic Cooperation and Development. Available at: http://www.oecd.org/ (Accessed 07.01.11).*,

[2] Landolph, J.R., 1985, *Chemical transformation in C3H 10T1/2 Cl 8 mouse embryo fibroblasts: Historical background, assessment of the transformation assay, and evolution and optimization of the transformation assay protocol. IARC Scientific Publications 67, 185-201.*,

[3] Reznikoff, C.A., Bertram, J.S., Brankow, D.W. & Heidel berger, C., 1973, *Quantitative and qualitative studies on chemical transformation of cloned C3H mouse embryo cells, sensitive to postconfluence inhibition of cell division. Cancer Research 33, 3239-3249.*,

[4] Procaccianti C., Stefanini F.M., Urani C., 2011, *The Cell Transformation Assay: Toward a Statistical Classification of Mixed and Intermediate Foci Images. ATLA 39, 1-14.*,

[5] Kaufman, L. & Rousseeuw, P.J., 1990, *Finding Groups in Data: An Introduction to Cluster Analysis, 368pp. New York, NY, USA: J. Wiley & Sons.*

[6] Urani, C., Stefanini, F.M., Bussinelli, L., Melchioretto, P. & Crosta, G.F., 2009, *Image analysis and automatic classification of transformed foci. Journal of Microscopy 234, 269-279.*,

[7] Urani, C., Crosta, G. F., Procaccianti, C., Melchioretto, P., Stefanini, F. M., 2010, *Image classifiers for the cell transformation assay: a progress report. Imaging, Manipulation, and Analysis of Biomolecules, Cells, and Tissues VIII. Edited by Farkas, Daniel L.; Nicolau, Dan V.; Leif, Robert C. Proceedings of the SPIE, Volume 7568, pp. 75681F-75681F-11.*,

Copyright © 2011

C. Procaccianti, C. Urani,

F. M. Stefanini