GLAD package : Gain and Loss Analysis of DNA

Philippe Hupe[1,2] and Emmanuel Barillot[2]

October 3, 2006

1. UMR 144 CNRS/Institut Curie, Institut Curie, 26, rue d'Ulm, Paris, 75248 cedex 05, France
2. Service Bioinformatique, Institut Curie, 26, rue d'Ulm, Paris, 75248 cedex 05, France
glad@curie.fr http://bioinfo.curie.fr

Contents

1 Overview **1**

2 Data **1**
2.1 Public data set 1
2.2 Bladder cancer data 2

3 GLAD classes **2**
3.1 arrayCGH 2
3.2 profileCGH and profileChr 2

4 Analysis of array CGH profile **3**
4.1 The *glad* function 3
4.2 The *daglad* function 4
4.3 Tuning parameters 6

5 Graphical functions **6**
5.1 Plot of raw array data 6
5.2 Plot of genomic profile 9

1 Overview

This document presents an overview of the `GLAD` package (Gain and Loss Analysis of DNA). This package is devoted to the analysis of Array Comparative Genomic Hybridization (array CGH) (**????**) . The methodology for detecting the breakpoints delimiting altered regions in genomic patterns and assigning a status (normal, gained or lost) to each chromosomal region described in the paper **?** is implemented in this package. Some graphical functions are provided as well.

2 Data

2.1 Public data set

We used the public data set described in **?**. The data correspond to 15 human cell strains with known karyotypes (12 fibroblast cell strains, 2 chorionic villus cell strains, 1 lymploblast cell strain) from the NIGMS Human Genetics Cell Repository (`http://locus.umdnj.edu/nigms`). Each cell

strain has been hybridized with an array CGH of 2276 BAC's, spotted in triplicate. Two array CGH profiles from the data obtained by **?** are available.

2.2 Bladder cancer data

Bladder cancer data from tumors collected at Henri Mondor Hospital (Créteil, France) (**?**) have been hybridized on arrays CGH composed of 2464 BACs (Radvanyi, Pinkel et al., unpublished results). In this data, only the log-ratios are provided and no information about clones is available since the data are not yet published. These data allow only some graphical functionalities to be shown and will be used as a support to illustrate some functions for array normalization (not yes available in the current version of the package).

3 GLAD classes

3.1 arrayCGH

This class stores raw values after images analysis. The object arrayCGH is a list with at least a data.frame named arrayValues and a vector named arrayDesign. The data.frame arrayValues must contain the following fields:

Col Vector of columns coordinates.

Row Vector of rows coordinates.

... Other elements can be added.

The vector arrayDesign is composed of 4 values : c(arrayCol, arrayRow, SpotCol, SpotRow). The array CGH is represented by arrayRow*arrayCol blocs and each bloc is composed of SpotRow*SpotCol spots. N.B. : Col takes the values in 1:arrayRow*SpotRow and Row takes the values in 1:arrayCol*SpotCol

3.2 profileCGH and profileChr

This class stores synthetic values related to each clone available onto the arrayCGH. The object profileChr corresponds to data of only one chromosome. Objects profileCGH and profileChr are composed of a list with the first element profileValues which is a data.frame with the following columns names:

LogRatio Test over Reference log-ratio.

PosOrder The rank position of each clone on the genome.

PosBase The base position of each clone on the genome.

Chromosome Chromosome name.

Clone The name of the corresponding clone.

... Other elements can be added.

LogRatio, Chromosome and PosOrder are compulsory.
To create those objects you can use the function *as.profileCGH*.

4 Analysis of array CGH profile

4.1 The *glad* function

A result of the GLAD methodology on cell line gm13330 (**?**) is presented in **Figure 1**.

```
[1] "Have fun with GLAD"
[1] TRUE
```

```
> data(snijders)
> profileCGH <- as.profileCGH(gm13330)
> res <- glad(profileCGH, mediancenter = FALSE, smoothfunc = "lawsglad",
+     bandwidth = 10, round = 1.5, model = "Gaussian", lkern = "Exponential",
+     qlambda = 0.999, base = FALSE, lambdabreak = 8, lambdacluster = 8,
+     lambdaclusterGen = 40, type = "tricubic", param = c(d = 6),
+     alpha = 0.001, msize = 5, method = "centroid", nmax = 8,
+     verbose = FALSE)
```

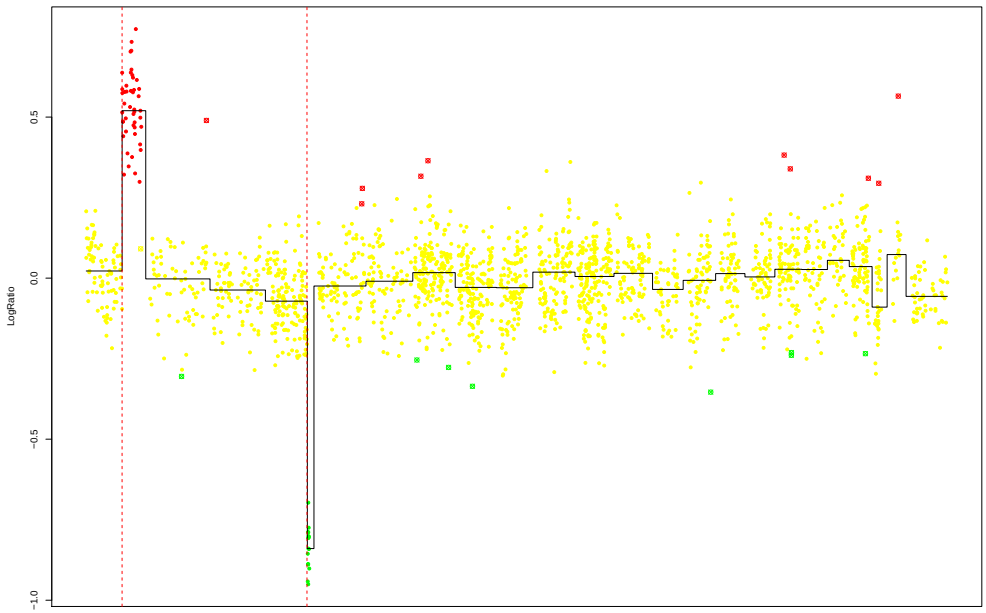

Figure 1: Results of glad on cell line gm13330 (Snijders data).

4.2 The *daglad* function

The algorithm implemented in this function is a slightly modified version of the GLAD algorithm.

```
> data(veltman)
> profileCGH <- as.profileCGH(P9)
> profileCGH <- daglad(profileCGH, mediancenter = FALSE, normalrefcenter = FALSE,
+     genomestep = FALSE, smoothfunc = "lawsglad", lkern = "Exponential",
+     model = "Gaussian", qlambda = 0.999, bandwidth = 10, base = FALSE,
+     round = 1.5, lambdabreak = 8, lambdaclusterGen = 40, param = c(d = 6),
+     alpha = 0.001, msize = 5, method = "centroid", nmin = 1,
+     nmax = 8, amplicon = 1, deletion = -5, deltaN = 0.2, forceGL = c(-0.3,
+         0.3), nbsigma = 3, MinBkpWeight = 0.35, CheckBkpPos = TRUE)

[1] "Smoothing for each Chromosome"
[1] "Optimization of the Breakpoints"
[1] "Check Breakpoints Position"
```

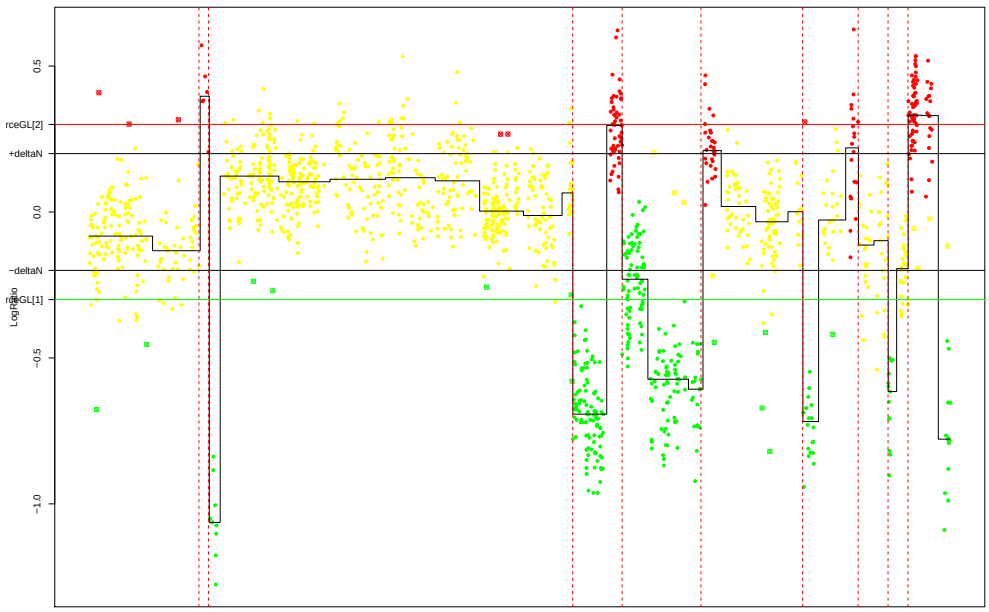

Figure 2: Results of daglad on the patient P9 (Veltman data).

The *daglad* function allows to choose some threshold to help the algorithm to identify the status of the genomic regions. The thresholds are given in the following parameters:

- deltaN
- forceGL
- deletion
- amplicon

Comparing **Figure 2** and **Figure 3** shows the influence of two different sets of parameters.

```
> data(veltman)
> profileCGH <- as.profileCGH(P9)
> profileCGH <- daglad(profileCGH, mediancenter = FALSE, normalrefcenter = FALSE,
+     genomestep = FALSE, smoothfunc = "lawsglad", lkern = "Exponential",
+     model = "Gaussian", qlambda = 0.999, bandwidth = 10, base = FALSE,
+     round = 1.5, lambdabreak = 8, lambdaclusterGen = 40, param = c(d = 6),
+     alpha = 0.001, msize = 5, method = "centroid", nmin = 1,
+     nmax = 8, amplicon = 1, deletion = -5, deltaN = 0.1, forceGL = c(-0.15,
+         0.15), nbsigma = 3, MinBkpWeight = 0.35, CheckBkpPos = TRUE)

[1] "Smoothing for each Chromosome"
[1] "Optimization of the Breakpoints"
[1] "Check Breakpoints Position"
```

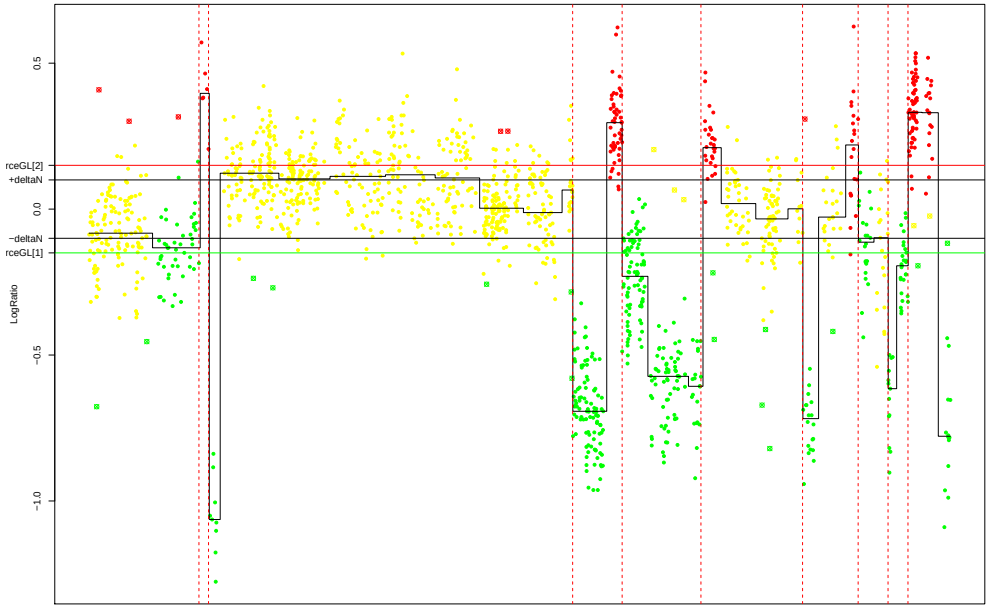

Figure 3: Results of daglad on the patient P9 (Veltman data) - Influence of the thresholds.

The *daglad* function allows a smoothing step over the whole genome (if *genomestep=TRUE*) where all the chromosomes are concatenated together. During this step, the cluster which corresponds to the Normal DNA level is identified: the thresholds used in the function (deltaN, forceGL, amplicon, deletion) are then compared to the median of this cluster.

4.3 Tuning parameters

The most important parameters are:

- *lambdabreak*
- *lambdacluster*
- *lambdaclusterGen*
- *param* $c(d = 6)$

Decreasing those parameters will lead to a higher number of breakpoints identified. For arrays experiments with very small Signal to Noise ratio it is recommended to use a small value of *param* like $d = 3$ or less.

5 Graphical functions

5.1 Plot of raw array data

```
> data(arrayCGH)
> array <- list(arrayValues = array2, arrayDesign = c(4, 4, 21,
+     22))
> class(array) <- "arrayCGH"
```

```
> arrayPlot(array, "Log2Rat", bar = "none")
```

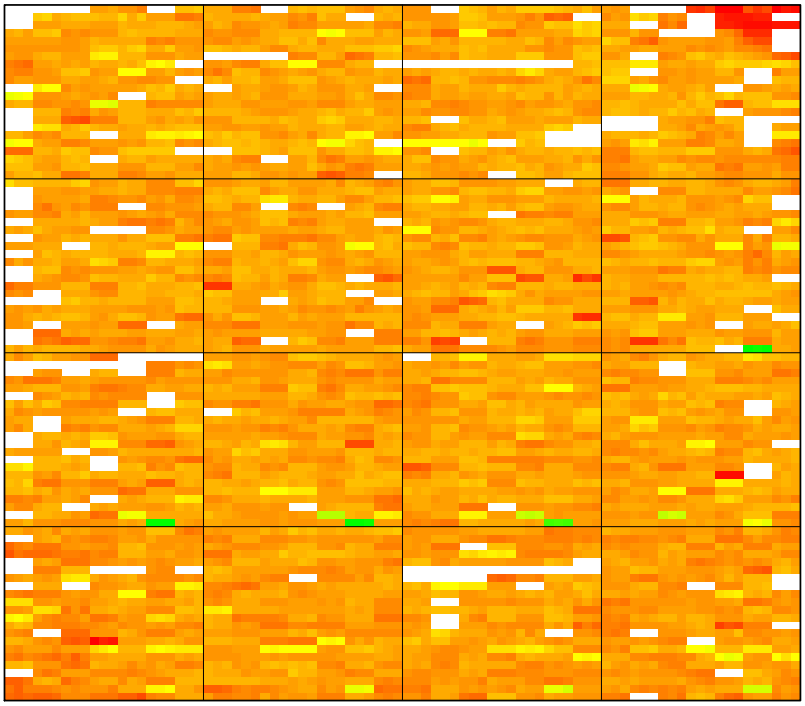

Figure 4: Spatial image of array CGH

```
> arrayPersp(array, "Log2Rat", box = FALSE, theta = 110, phi = 40,
+     bar = FALSE)
```

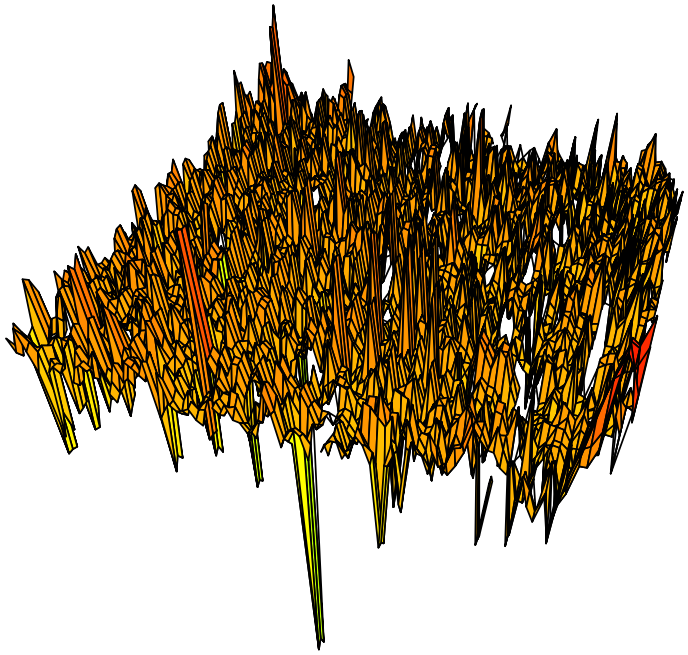

Figure 5: Perspective image of array CGH

5.2 Plot of genomic profile

```
> plotProfile(res, unit = 3, Bkp = TRUE, labels = FALSE, Smoothing = "Smoothing",
+     plotband = FALSE)
```

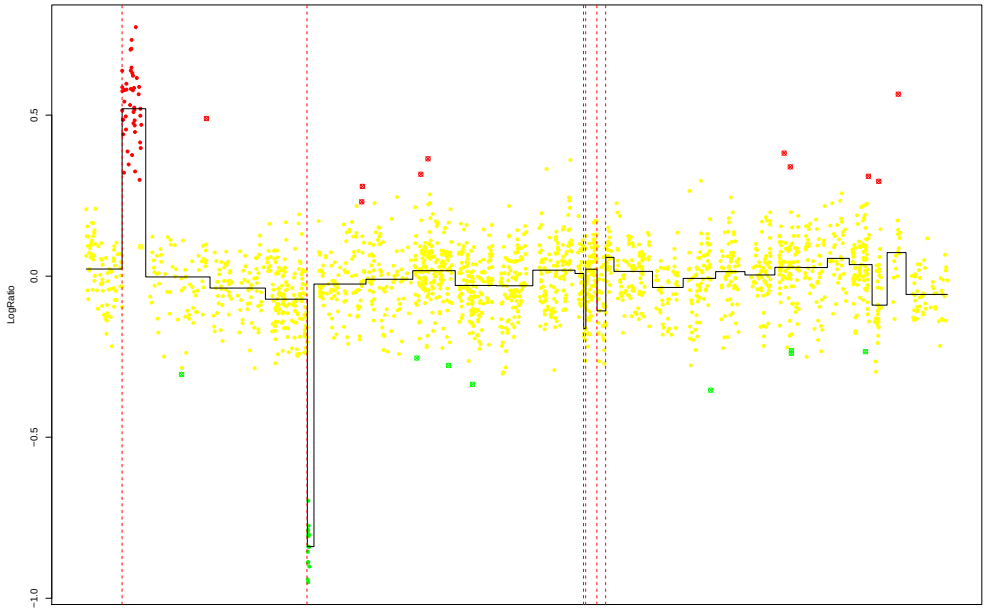

Figure 6: Genomic profile on the whole genome

```
> plotProfile(res, unit = 3, Bkp = TRUE, labels = FALSE, Smoothing = "Smoothing")
```

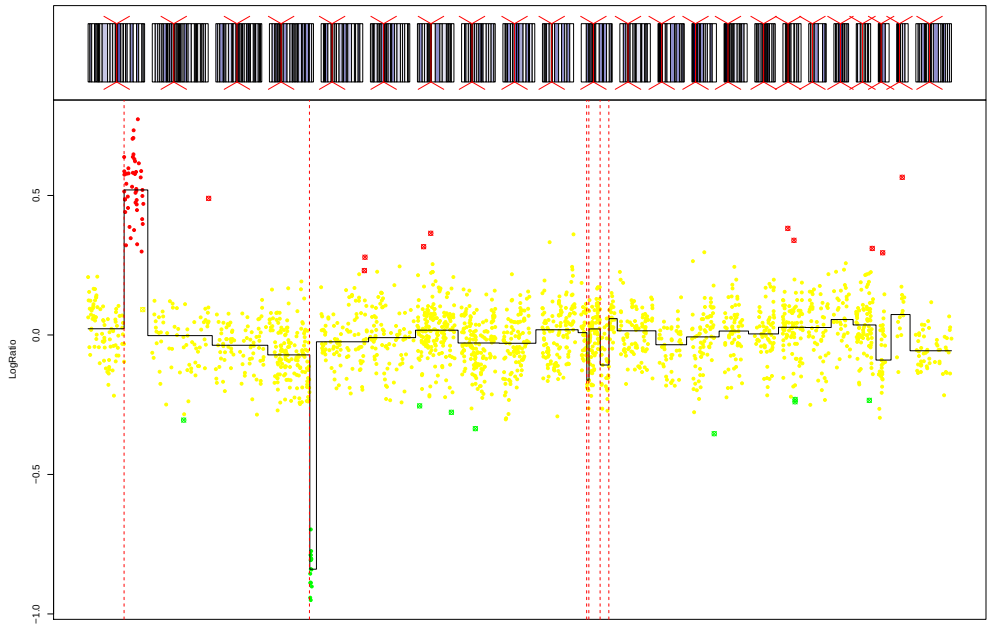

Figure 7: Genomic profile on the whole genome and cytogenetic banding

```
> text <- list(x = c(90000, 2e+05), y = c(0.15, 0.3), labels = c("NORMAL",
+     "GAIN"), cex = 2)
> plotProfile(res, unit = 3, Bkp = TRUE, labels = TRUE, Chromosome = 1,
+     Smoothing = "Smoothing", plotband = FALSE, text = text)
```

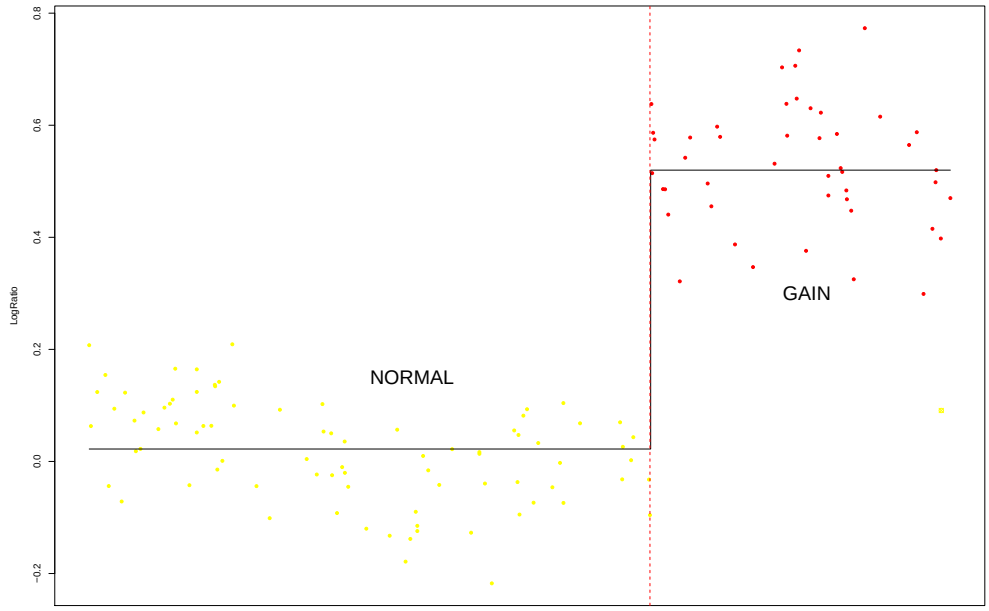

Figure 8: Genomic profile for chromosome 1

```
> text <- list(x = c(90000, 2e+05), y = c(0.15, 0.3), labels = c("NORMAL",
+     "GAIN"), cex = 2)
> plotProfile(res, unit = 3, Bkp = TRUE, labels = TRUE, Chromosome = 1,
+     Smoothing = "Smoothing", text = text, main = "Chromosome 1")
```

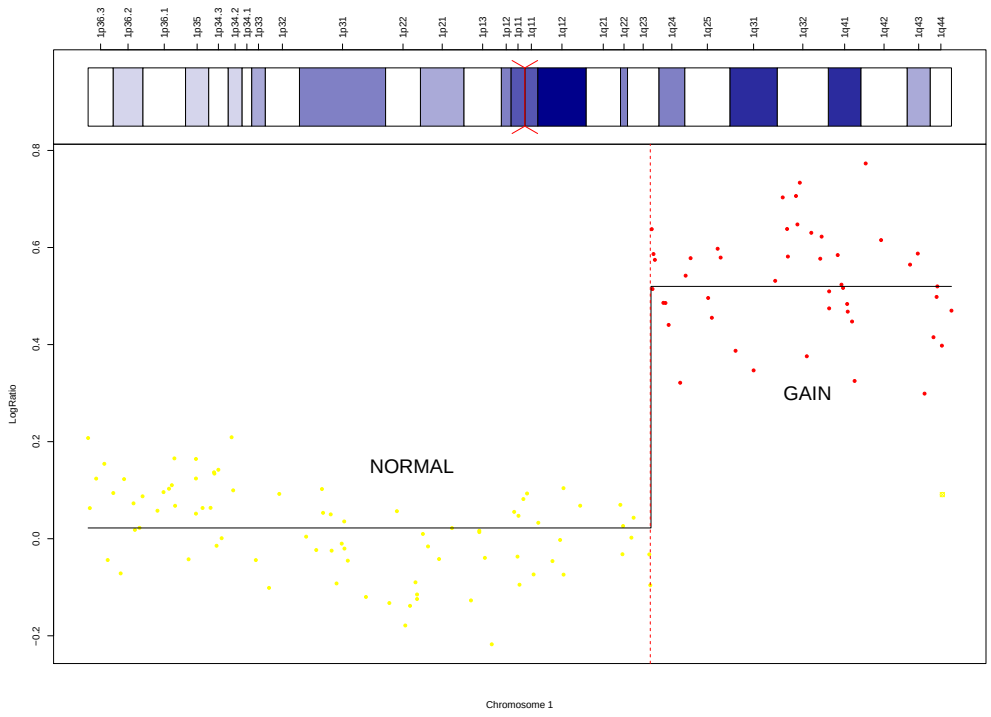

Figure 9: Genomic profile for chromosome 1 and cytogenetic banding with labels