CGHregions: Dimension reduction for array CGH data with minimal information loss.

Mark van de Wiel and Sjoerd Vosse

May 3, 2016

Department of Pathology
VU University Medical Center

mark.vdwiel@vumc.nl

Contents

1 Overview

CGHregions allows users to reduce the dimensionality of array CGH data to facilitate downstream analysis. CGHregions takes as input array CGH data (log2-ratios) that have been segmented (i.e., split into chromosomal segments of similar log2-ratios) and called (i.e., a copy number assigned to each segment) on a per-sample basis and adjusts the segmentation so that break-points that are in similar locations across multiple samples are set to be in identical locations. Segmented and called data can be obtained by using the CGHcall package. The resulting dimensionality reduction facilitates downstream analysis in a variety of ways (e.g., reduces severity of multiple hypothesis testing, facilitates clustering and visualization, reduces computer memory requirements).
This document provides an overview on the usage of the CGHregions package. For more detailed information on the algorithm and assumptions we refer to the CGHregions article (**?**). As example data we attached the first five samples of the Wilting dataset (**?**), called with CGHcall (**?**).

2 Example

In this section we will use CGHregions to define regions of minimal information loss and visualize the results. First, we load the package and the data. The data have been segmented and called using the CGHcall package.

```
> library(CGHregions)
> data(WiltingCalled)
> WiltingCalled

cghCall (storageMode: lockedEnvironment)
assayData: 3552 features, 5 samples
  element names: calls, copynumber, probgain, probloss, probnorm, segmented
protocolData: none
phenoData: none
featureData
  featureNames: RP11-465B22 RP4-785P20 ... CTB-99K24 (3552 total)
  fvarLabels: Chromosome Start End
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

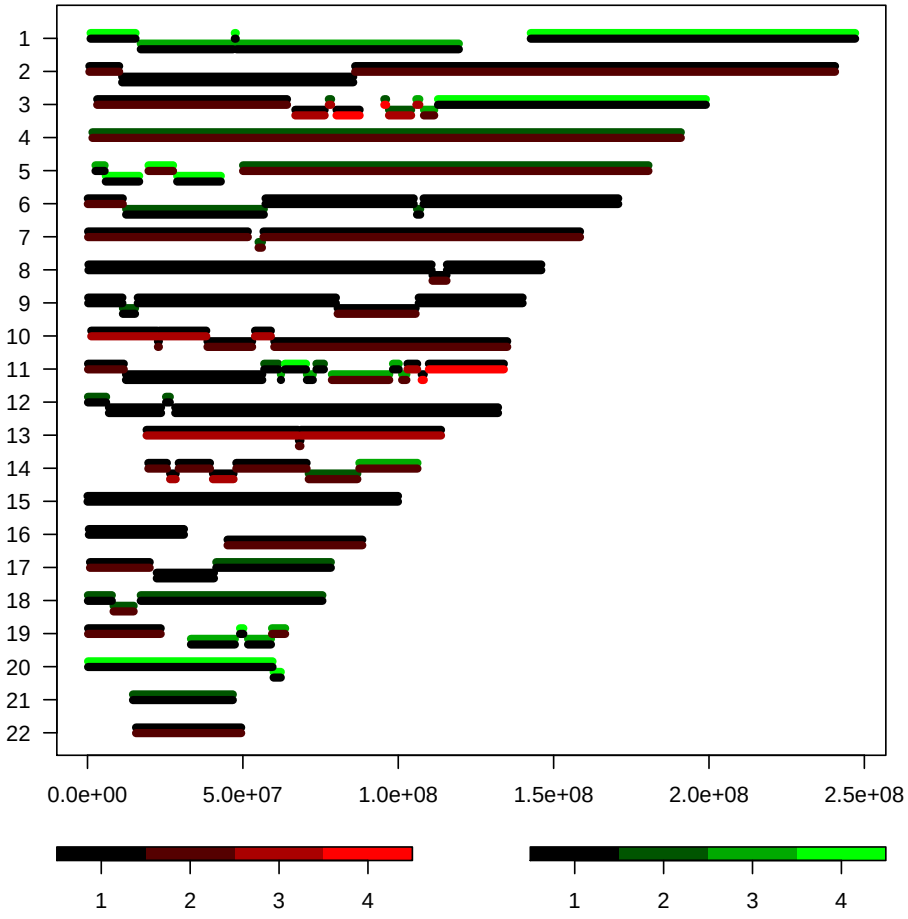
Next, we apply the `CGHregions` function which returns an object of class cghRegions. For the maximum information loss we allow a value of 0.01.

```
> regions <- CGHregions(WiltingCalled, averror=0.01)

                  Samples
 1.000000000  0.009145129 10.000000000
            Samples
2.0000000 0.0500994 8.0000000
[1] "Tuning on small data set finished...started with entire data set"
    Samples     Samples
 1.00000000  0.01245421 41.00000000  2.00000000
Samples Samples
      0       0      90
[1] "c = 0, nr of regions: 90"
[1] "Finished with entire data set."

> regions
```

```
cghRegions (storageMode: lockedEnvironment)
assayData: 90 features, 5 samples
  element names: regions
protocolData: none
phenoData: none
featureData
  featureNames: 1 2 ... 90 (90 total)
  fvarLabels: Chromosome Start ... AveDist (5 total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

As we can see, the algorithm has returned an object with 90 regions. To visualize the results we use the `plot` method which creates a plot displaying chromosomes on the Y-axis and base pair position on the X-axis. A new region is displayed by a slight jump with respect to the previous region. Each region is displayed as a bi-colored segment, the lower and upper part of which correspond to the proportions pl and pg of samples with a loss (red) or gain (green), respectively. The color coding is displayed as well: 1: pl (pg) $< 10\%$; 2: 10% = pl (pg) $< 30\%$; 3:30% = pl (pg) $< 50\%$; 4: pl (pg) = 50%.

```
> plot(regions)
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
0.0e+00
5.0e+07
1.0e+08
1.5e+08
2.0e+08
2.5e+08
1
2
3
4
1
2
3
4