DNAcopy: A Package for Analyzing DNA Copy Data

Venkatraman E. Seshan[1] and Adam B. Olshen[2]

October 17, 2016

[1]Department of Epidemiology and Biostatistics
Memorial Sloan-Kettering Cancer Center
seshanv@mskcc.org

[2]Department of Epidemiology and Biostatistics
University of California, San Francisco
olshena@biostat.ucsf.edu

Contents

1 Overview

This document presents an overview of the `DNAcopy` package. This package is for analyzing array DNA copy number data, which is usually (but not always) called array Comparative Genomic Hybridization (array CGH) data (**???**). It implements our methodology for finding change-points in these data (**?**), which are points after which the (log) test over reference ratios have changed location. Our model is that the change-points correspond to positions where the underlying DNA copy number has changed. Therefore, change-points can be used to identify regions of gained and lost copy number. We also provide a function for making relevant plots of these data.

2 Data

We selected a subset of the data set presented in **?**. We are calling this data set `coriell`. The data correspond to two array CGH studies of fibroblast cell strains. In particular, we chose the studies **GM05296** and **GM13330**. After selecting only the mapped data from chromosomes 1-22 and X, there are 2271 data points. There is accompanying spectral karyotype data (not included), which can serve as a gold standard. The data can be found at
`http://www.nature.com/ng/journal/v29/n3/suppinfo/ng754_S1.html`

3 An Example

Here we perform an analysis on the **GM05296** array CGH study described above.

```
> library(DNAcopy)

> data(coriell)
```

Before segmentation the data needs to be made into a CNA object.

```
> CNA.object <- CNA(cbind(coriell$Coriell.05296),
+                   coriell$Chromosome,coriell$Position,
+                   data.type="logratio",sampleid="c05296")
```

We generally recommend smoothing single point outliers before analysis. It is a good idea to check that the smoothing is proper for a particular data set.

```
> smoothed.CNA.object <- smooth.CNA(CNA.object)
```

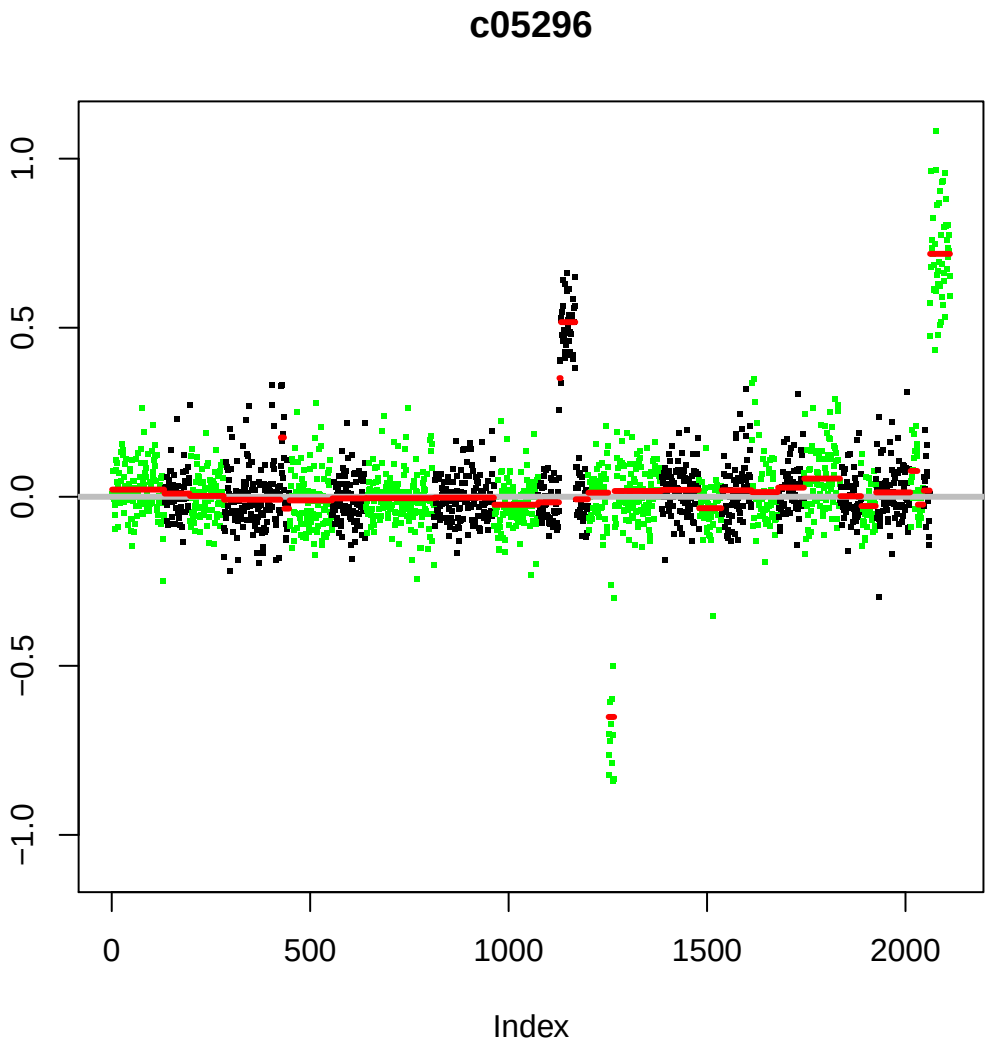
After smoothing, if necessary, the segmentation is run. Here the default parameters are used. A brief discussion of parameters that can be adjusted is in the Tips section.

```
> segment.smoothed.CNA.object <- segment(smoothed.CNA.object, verbose=1)

Analyzing: c05296
```

There are a number of plots that can be made. The first is ordering the data by chromosome and map positons. The red lines correspond to mean values in segments. Note that the points are in alternate colors to indicate different chromosomes.

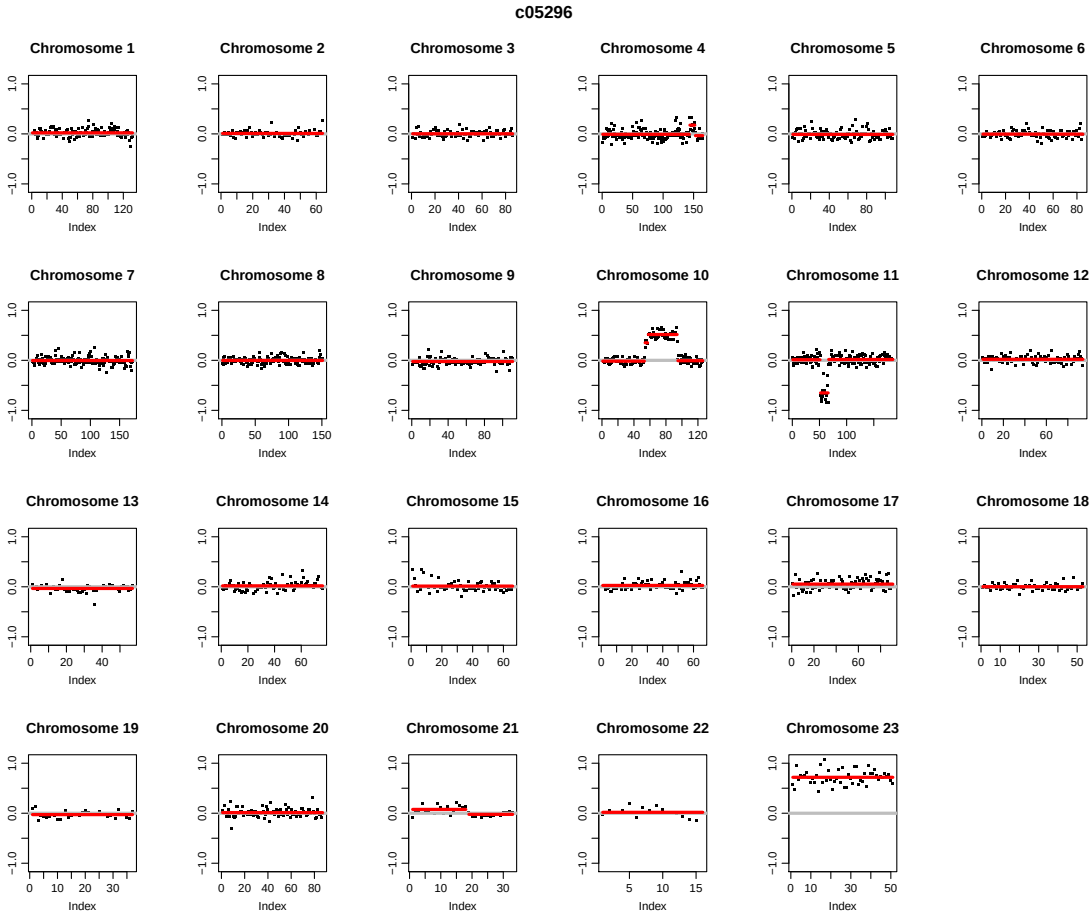
```
> plot(segment.smoothed.CNA.object, plot.type="w")
```



Another possibility is to plot by chromosome within a study.

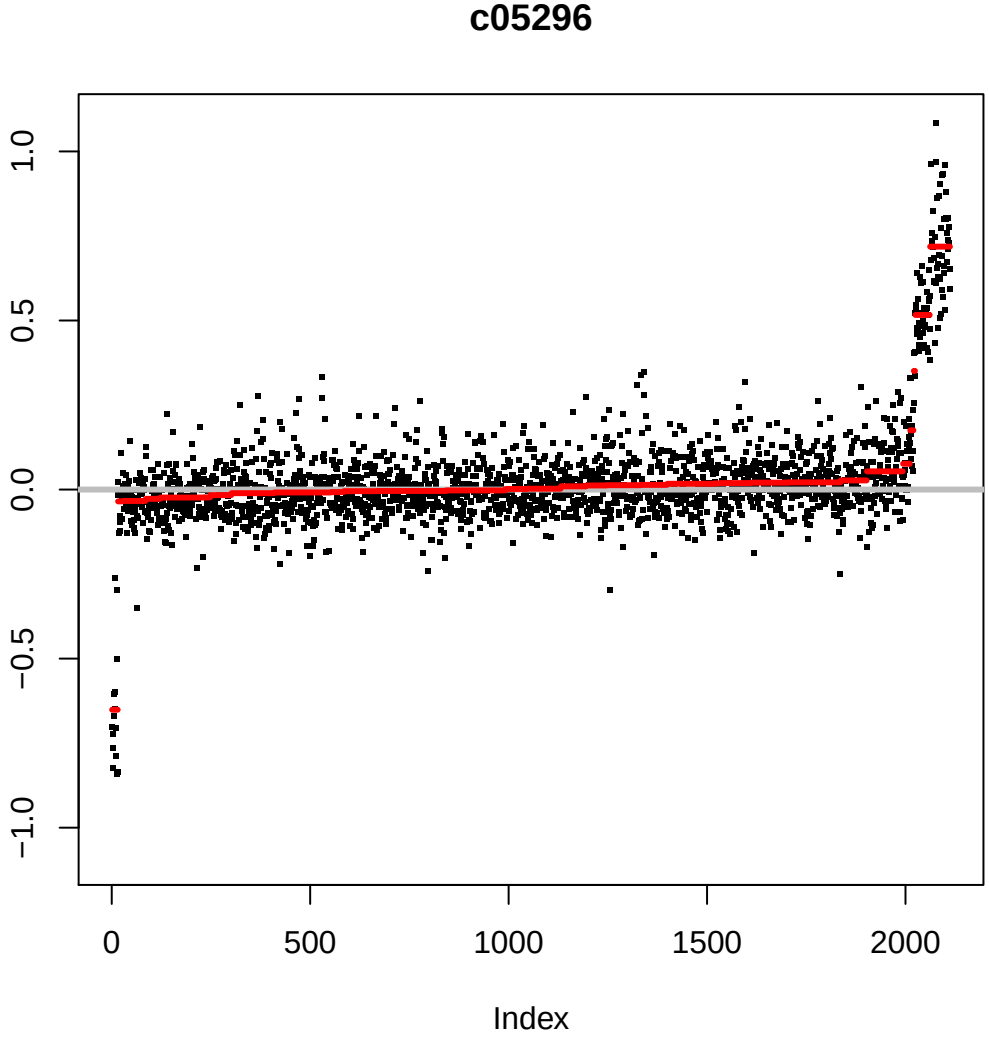
```
> plot(segment.smoothed.CNA.object, plot.type="s")
```

```
Setting multi-figure configuration
```



If there are multiple studies, one could plot by chromosome across studies using the option `plot.type='c'`. A final plot orders the segment by their chromosome means. One can take the plateaus in this plot to determine what the mean values should be for calling segments gains or losses. In this case, maybe 0.4 for gains and -0.6 for losses. For most data, these plateaus are much closer to zero. The next generation of this software will have automatic methods for calling gains and losses.

```
> plot(segment.smoothed.CNA.object, plot.type="p")
```



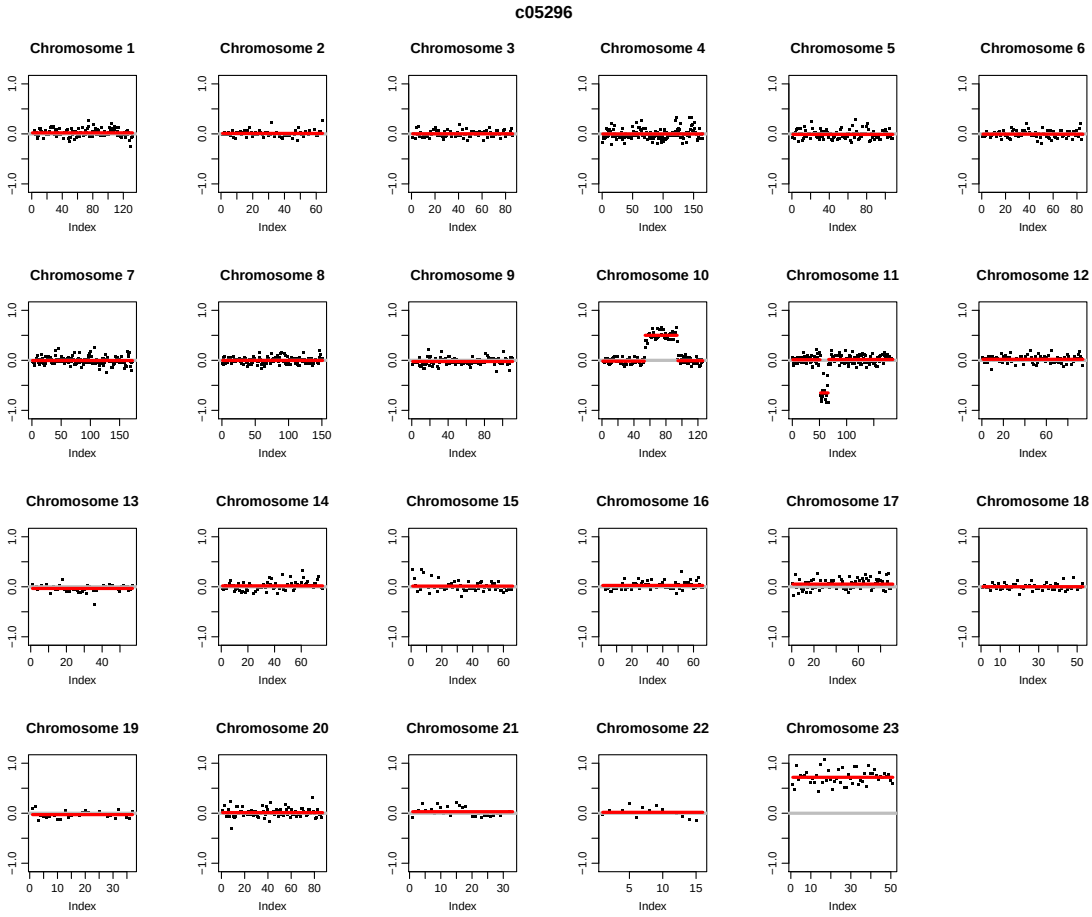
Change-points are often found due to local trends in the data. An undo method is needed to get rid of unnecessary change-points. Below all splits that are not at least three SDs apart are removed. The following plot shows that all splits not corresponding to the gold standard results have been removed.

```
> sdundo.CNA.object <- segment(smoothed.CNA.object,
+                              undo.splits="sdundo",
+                              undo.SD=3,verbose=1)

Analyzing: c05296

> plot(sdundo.CNA.object,plot.type="s")

Setting multi-figure configuration
```



4 Tips

A function that may be of interest that has not been mentioned is `subset.CNA`. It allows for subsetting of a CNA object by chromosome and sample so that segmentation does not have to be run on a whole data set. Similarly, `subset.DNAcopy` allows subsetting of DNAcopy objects, which contain the output of segmentation.

The original default segmentation algorithm, because it was based on permutation, took $O(N^2)$ computations, where N is the number of markers on a chromosome. The new default algorithm is much faster. It includes a hybrid approach to compute the p-value for segmenting based partly on permutation and partly on a Gaussian approximation (available in all versions after 1.2.0) and a stopping rule (available in all versions after 1.5.0) to declare change when there is a strong evidence for its presence (`?`). We no longer recommend using overlapping windows for larger data sets. It is still possible to run the full permutations analysis using the option `p.method='perm'`. If the new algorithm is still too slow, one can reduce the number of permutations in the hybrid method using the parameter `nperm` (default is 10,000). However, the lower `alpha` (the significance level for the test to accept change-points) is, the more permutations that are needed. The stopping boundary needs to be computed for any choice of `nperm` and `alpha` which is not the default which is done

automatically within the function `segment` or can be done externally using the function `getbdry` and passed on to `segment`.