

# Comparing different segmentation schemes

ML Smith, JC Marioni

February 27, 2006

## Overview

**i** Many methods exist for segmenting array CGH data into states containing the same number of copies of the genome. The purpose of this practical is to use graphical means to compare the outputs of segmentation schema that are available as R functions and incorporated within the `snappyCGH` library. Initially, we will give an overview of the theory underpinning each of the segmentation methods, before going on to examine how each method performs in practice.

**👉** Read the following section on segmentation theory before beginning the practical sections on the following page.

## **i** Segmentation Theory

Segmentation schema come in many different flavours. Examples include a homogeneous hidden Markov model (HomHMM) approach, a non-parametric change point method (DNAcopy) and a Gaussian model-based (dynamic programming) scheme, GLAD. Additionally, clustering based approaches, wavelet models and various smoothing schema have also been applied. In this practical we will concentrate on comparing the first three methods. In addition, we will also compare the segmentations obtained using these methods with a new method, BioHMM, that can incorporate biological effects that are not accounted for by other methods.

We note that the methods described below take account of the order of the clones on the genome. This is unlike a threshold based approach where we compare the  $\log_2$  ratio of each clone with a given threshold in order to determine whether it should be classified as gained or lost.

## DNA Copy

DNA copy [4] is a non-parametric method based on a circular binary segmentation algorithm (CBS). The CBS algorithm is itself a modification of a change point method that allows for tertiary (i.e. three or more) splits by joining both ends of the chromosome whilst carrying out the segmentation. By modelling discrete copy number gains/losses, DNACopy splits the chromosome into contiguous regions that have the same copy number. The significance of each of these splits is assessed (internally) using a permutation-based approach - hence, a parametric (model-based) framework is not necessary.

## GLAD

GLAD [2] uses an adaptive weights smoothing (AWS) algorithm to estimate a piecewise constant function that is then used to estimate breakpoints. In essence, GLAD fits a model of the form  $Y_i = \theta(X_i) + \epsilon_i$  where  $\epsilon_i$  are independently and identically distributed as  $N(0, \sigma^2)$  and  $\theta$  is a piecewise constant function. Using a likelihood based approach, the AWS algorithm iteratively finds the maximal region for which  $\theta(X_i)$  is constant for each clone,  $X_j$ .

## HomHMM

This method employs an (unsupervised) HMM based approach [1] to assign clones to underlying (hidden) states with constant copy number. An example of a Markov model would be to predict today's weather based solely upon what the weather was like yesterday. In a HMM, we are trying to predict what the weather will be like today based upon whether a person was carrying an umbrella yesterday. In an aCGH context, the weather is each of the copy number states and whether a person is carrying an umbrella corresponds to the observed  $\log_2$  ratios. A HMM is fitted using a forward-backward process and the number of states is determined using a criterion such as the AIC or BIC.

☞ While all three methods described above take account of the fact that clones are ordered along the genome in the segmentation scheme, they are unable to or do not take account of additional biological information such as the distance between clones or measures of the “quality” associated with particular clones. In order to address this problem, we have developed a new heterogeneous HMM called BioHMM which can take account of additional biological information.

## 📄 BioHMM

BioHMM [3] employs a similar structure to HomHMM with the exception that the probability of a clone being a breakpoint now depends upon biological factors such as the distance of the clone from its neighbours or the quality of the measurement associated with a clone (such a measurement might be obtained using an image analysis software package for example).

☞ We use BioHMM by applying the function `fitBioHMM` to an `MAList`. `fitBioHMM` automatically calculates the distance between clones and uses this as a covariate. Other covariates can be read in as the user wishes. For more information and examples of the application of BioHMM in practice, see the relevant R helpfile or read [3].

## Comparing segmentation results using simulated data

☞ Make sure the current directory for your R session is the `SegComparison` folder. All your analysis will take place in this directory. At any point during your session you may save the workspace so that you can come back to the same session later.

*You can set the current directory by typing `setwd("/path/to/SegComparison")`. Alternatively, in a GUI implementation (the point-and-click kind) use the option in the drop down*

menu to change to a different working directory.

**Exercise 1:** Load the *snapCGH* library (which automatically loads *limma* and other libraries) and read in the `SegComparison.RData` object. Look at the objects in your current R workspace.

```
> library(snapCGH)
> load("SegComp.RData")
> ls()
```

☞ This object contains both unsegmented (`MAList` objects) and the corresponding segmented data (`SegLists`) from three simulated chromosomes. The data used has been simulated using a novel scheme we have developed. We use simulated data since in order to properly compare different segmentation schema we need to know the true location of the breakpoints.

*The comparison of different segmentation schema given below is not thorough, nor does it claim to be. Instead, we aim to provide a broad overview of the different segmentation methods and to illustrate some methods for visualising array CGH that we are developing.*

**Exercise 2:** Generate plots of the unsegmented data for each of the samples.

```
> genomePlot(MA.Sim, array = 4, chrom.to.plot = 14)
> genomePlot(MA.Sim, array = 7, chrom.to.plot = 14)
> genomePlot(MA.Sim, array = 18, chrom.to.plot = 14)
```

☞ The scheme used to simulate the data not only simulates the  $\log_2$  ratios but also the location of clones on a particular chromosome. In the example below, the clone locations have been simulated based upon the length (and centromeric location) of chromosome 14. Twenty different sets of  $\log_2$  ratios have been simulated. In this practical we concentrate on three of these sets, which we will henceforth refer to as arrays 4,7 and 18.

**Exercise 3:** Use plotting functions to compare the output obtained using each of the segmentation schema. Are there any obvious differences between the different methods? Do some breakpoints appear to be harder to detect than other?

📌 On each of the plots we will plot not only the output obtained using each of the segmentation schemes, but we will also overlay the true copy number associated with each of the clones. Recall that we are able to do this because we are using simulated data.

```
> par(mfrow = c(1, 4))
> plotSegmentedGenome(BioHMM.Sim, True.Sim, chrom.to.plot = 14,
+   array = 4, colors = c("blue", "magenta"))
> plotSegmentedGenome(HomHMM.Sim, True.Sim, chrom.to.plot = 14,
+   array = 4, colors = c("blue", "magenta"))
> plotSegmentedGenome(GLAD.Sim, True.Sim, chrom.to.plot = 14, array = 4,
+   colors = c("blue", "magenta"))
> plotSegmentedGenome(DNACopy.Sim, True.Sim, chrom.to.plot = 14,
+   array = 4, colors = c("blue", "magenta"))
```

**Exercise 4:** Repeat the above commands in order to generate plots for the other two arrays we are focusing on (namely arrays 7 and 18).

☞ Save your workspace and change your working directory to Coriell.

## Segmentation methods for Coriell data

📖 If one were using a plot of the whole genome (produced using the `genomePlot` function, one might want to focus in on particular chromosomes of interest. One way of doing this is to re-apply `genomePlot` using the `chrom.to.plot` option. However, this can become rather tiresome. Hence we have developed a new function, `zoomGenome`, that enables the user to zoom in to a chromosome of interest by clicking on it.

**Exercise 5:** Read in the Coriell dataset and apply the `zoomGenome` function to focus in on individual chromosomes that appear to be interesting. This contains an `MAList` and four `SegLists`, one for each of the schema described above. The data is taken from 15 Coriell cell lines and these are thoroughly described in [6].

```
> load("Coriell.Rda")
> zoomGenome(BioHMM.Coriell, array = 7, chrominfo = chrominfo.Mb *
+           1000)
```

*To stop the interaction, click on an “empty” region of the graphics window.*

📖 Similarly, in order to look more closely at a region of interest (either pre or post segmentation) in a particular chromosome one can use the `xlim` option provided within both the `genomePlot` and `plotSegmentedGenome` functions. However, to zoom in using these functions it is necessary to manually change the limits and this is quite inefficient! In order to overcome this we have developed another interactive function (`zoomChromosome`) that enables the user to click on the start and end position of the region they want to zoom into.

**Exercise 6:** Apply the `zoomChromosome` to focus in on a region of interest for both an unsegmented and segmented chromosome.

```
> zoomChromosome(MA.Coriell, array = 1, chrom.to.plot = 1, chrominfo = chrominfo.Mb *
+           1000)
> zoomChromosome(BioHMM.Coriell, array = 1, chrom.to.plot = 1,
+           chrominfo = chrominfo.Mb * 1000)
```

📖 While plots such as those described above are very useful if one is only interested in a particular sample, they are less useful if one wants to compare multiple samples with each other. If one wants to do this an alternative strategy is required. One way in which this could be done is to use the `heatmapGenome` function. This can be applied both pre and post segmentation. If you use an `MAList`, the heatmap will be of the observed  $\log_2$  ratios. Alternatively, if the input to `heatmapGenome` is a `SegList`, the heatmap will use the predicted values for each clone.

**Exercise 7:** Use the `heatmapGenome` function to compare the pre and post segmentation output for the Coriell cell lines when they are segmented using each of the different methods.

```

> heatmapGenome(MA.Coriell)
> heatmapGenome(BioHMM.Coriell)
> heatmapGenome(HomHMM.Coriell)
> heatmapGenome(GLAD.Coriell)
> heatmapGenome(DNAcopy.Coriell)

```

❏ Another way to visualise multiple samples is to use a method that highlights regions where the  $\log_2$  ratios are above/below specified thresholds for all of the samples. We have implemented a function, `plotGainLoss`, that aims to do this. This function is based upon a plot in [5]. It is currently still under heavy development and will probably change considerably in the near future. This current implementation should be considered a beta version of the function.

**Exercise 8:** Apply the `plotGainLoss` function to the Coriell cell lines on both a chromosome and genome-wide basis. Can you see any regions that appear to be consistently gained/lost? What effect does changing the cutoff (threshold) values have?

```

> plotGainLoss(BioHMM.Coriell, up.cutoff = 0.2, down.cutoff = -0.2,
+   chrominfo = chrominfo.Mb * 1000)
> plotGainLoss(BioHMM.Coriell, up.cutoff = 0.2, down.cutoff = -0.2,
+   chrom.to.plot = 9, chrominfo = chrominfo.Mb * 1000)

```

## References

- [1] Fridlyand, J., Snijders, A.M., Pinkel, D., Albertson, D.G., Jain, A.N. (2004) Hidden Markov models approach to the analysis of array CGH data, *Journal of Multivariate Analysis*, **90**, 132-153.
- [2] Hupé, P., Stransky, N., Thiery, J-P., Radvanyi, F., Barillot, E. (2004) Analysis of array CGH data: from signal ratio to gain and loss of DNA regions, *Bioinformatics*, **20**, 3413-3422.
- [3] Marioni, J.C., Thorne, N.P. Tavaré S. (2006) BioHMM: a heterogeneous hidden Markov model for segmenting array CGH data, accepted pending minor revisions.
- [4] Olshen, A.B., Venkatraman, E.S., Lucito, R., Wigler, M. (2004) Circular binary segmentation for the analysis of array-based DNA copy number data, *Biostatistics*, **5**, 557-572.
- [5] Seng, T.J., Ichimura K., Liu, L., Tingby, O., Pearson, D.M., Collins, V.P. (2005) COmplex Chromosome 22 Rearrangements in Astrocytic Tumors Identifies Using Microsatellite and Chromosome 22 Tile Path Array Analysis, *Genes, Chromosomes and Cancer*, **43**, 181-193.
- [6] Snijders, A.M., Nowak, N., Segreaves, R., Blackwood, S., Brown, N., Conroy, N., Hamilton, G., Hindle, A.K., Huey, B., Kimura, K., Law, S., Myambo, K., Palmer, J., Ylstra, B., Yue, J.P., Gray, J.W., Jain, A.N., Pinkel, D., Albertson, D.G. (2001) Assembly of microarrays for genome-wide measurement of DNAcopy number, *Nature Genetics*, **29**, 4281-4286.