CGHcall: Calling aberrations for array CGH tumor profiles.

Sjoerd Vosse and Mark van de Wiel

October 1, 2012

Department of Epidemiology & Biostatistics
VU University Medical Center

mark.vdwiel@vumc.nl

Contents

1 Overview

CGHcall allows users to make an objective and effective classification of their aCGH data into copy number states (loss, normal, gain or amplification). This document provides an overview on the usage of the CGHcall package. For more detailed information on the algorithm and assumptions we refer to the article (van de Wiel et al., 2007) and its supplementary material. As example data we attached the first five samples of the Wilting dataset (Wilting et al., 2006). After filtering and selecting only the autosomal 4709 datapoints remained.

2 Example

In this section we will use CGHcall to call and visualize the aberrations in the dataset described above. First, we load the package and the data:

```
> library(CGHcall)
> data(Wilting)
> Wilting <- make_cghRaw(Wilting)
```

Next, we apply the `preprocess` function which:

- removes data with unknown or invalid position information.
- shrinks the data to `nchrom` chromosomes.
- removes data with more than `maxmiss` % missing values.
- imputes missing values using `impute.knn` from the package `impute` (Troyanskaya et al., 2001).

```
> cghdata <- preprocess(Wilting, maxmiss=30, nchrom=22)
```

```
Changing impute.knn parameter k from 10 to 4 due to small sample size.
```

To be able to compare profiles they need to be normalized. In this package we first provide very basic global median or mode normalization. This function also contains smoothing of outliers as implemented in the DNAcopy package (Venkatraman and Olshen, 2007). Furthermore, when the proportion of tumor cells is not 100% the ratios can be corrected. See the article and the supplementary material for more information on cellularity correction (van de Wiel et al., 2007).

```
> norm.cghdata <- normalize(cghdata, method="median", smoothOutliers=TRUE)
```

```
Applying median normalization ...
Smoothing outliers ...
```

The next step is segmentation of the data. This package only provides a wrapper function that applies the `DNAcopy` algorithm (Venkatraman and Olshen, 2007). It provides extra functionality by allowing to undo splits differently for long and short segments, respectively. In the example below short segments are smaller than clen=10 probes, and for such segments undo.splits is effective when segments are less than undo.SD=3 (sd) apart. For long segments a less stringent criterion holds: undo when less than undo.SD/relSDlong = 3/5 (sd) apart. If, for two consecutive segements, one is short and one is long, splits are undone in the same way as for two consecutive short segments. To save time we will limit our analysis to the first two samples from here on.

```
> norm.cghdata <- norm.cghdata[,1:2]
> seg.cghdata <- segmentData(norm.cghdata, method="DNAcopy",undo.splits="sdundo",undo
+ clen=10, relSDlong=5)

Start data segmentation ..
Analyzing: Sample.1
Analyzing: Sample.2
```

Post-segmentation normalization allows to better set the zero level after segmentation.

```
> postseg.cghdata <- postsegnormalize(seg.cghdata)
```

Now that the data have been normalized and segments have been defined, we need to determine which segments should be classified as double losses, losses, normal, gains or amplifications. Cellularity correction is now provided WITHIN the calling step (as opposed to some earlier of CGHcall)

```
> tumor.prop <- c(0.75, 0.9)
> result <- CGHcall(postseg.cghdata,nclass=5,cellularity=tumor.prop)

EM algorithm started ...
[1] "Total number of segments present in the data: 90"
[1] "Number of segments used for fitting the model: 90"
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 549909 14.7     899071 24.1   792148 21.2
Vcells 599476  4.6    1162592  8.9  1162592  8.9
Calling iteration 1 :
[1] "optim results"
[1] "time: 25"
[1] "minimum: 3748.69878446397"
     j       rl       mudl       musl        mun      mug     mudg      mua
[1,] 2 3732.566 -0.7730131 -0.2898698 0.01374418 0.347191 0.593527 1.057513
          sddl       sdsl        sdn       sdg      sddg       sda
[1,] 0.1935916 0.08495295 0.06150202 0.1134993 0.1140091 0.1183082
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 551068 14.8     899071 24.1   899071 24.1
Vcells 600967  4.6    1162592  8.9  1162592  8.9
Calling iteration 2 :
[1] "optim results"
[1] "time: 18"
```

```
[1] "minimum: 3742.64242208813"
     j       rl       mudl       musl       mun      mug      mudg      mua
[1,] 2 3731.237 -0.7801531 -0.2854698 0.0173235 0.346793 0.5928465 1.056921
          sddl       sdsl        sdn        sdg       sddg       sda
[1,] 0.3283833 0.08301401 0.05550779 0.09518613 0.09570998 0.1552889
EM algorithm done ...
Computing posterior probabilities for all segments ...
Total time: 1 minutes
```

The result of CGHcall needs to be converted to a call object. This can be a large object for large arrays.

```
> result <- ExpandCGHcall(result,postseg.cghdata)

Adjusting segmented data for cellularity ...
Cellularity sample 1 :  0.75
Cellularity sample 2 :  0.9
Adjusting normalized data for cellularity ...
Cellularity sample 1 :  0.75
Cellularity sample 2 :  0.9
[1] 1
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553017 14.8     899071 24.1   899071 24.1
Vcells 630920  4.9    1162592  8.9  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553027 14.8     899071 24.1   899071 24.1
Vcells 648683  5.0    1162592  8.9  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553026 14.8     899071 24.1   899071 24.1
Vcells 648682  5.0    1162592  8.9  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553053 14.8     899071 24.1   899071 24.1
Vcells 677101  5.2    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553091 14.8     899071 24.1   899071 24.1
Vcells 678900  5.2    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553099 14.8     899071 24.1   899071 24.1
Vcells 680679  5.2    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
```

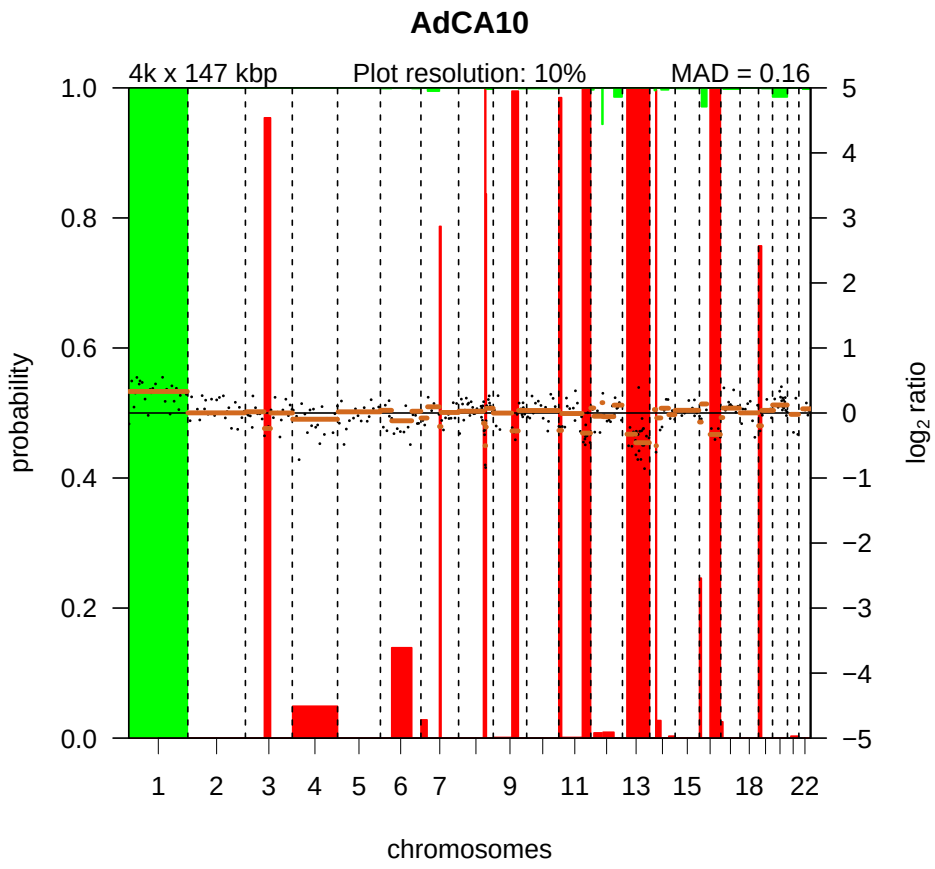
```
Ncells 553107 14.8     899071 24.1   899071 24.1
Vcells 682458  5.3    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553115 14.8     899071 24.1   899071 24.1
Vcells 684237  5.3    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553123 14.8     899071 24.1   899071 24.1
Vcells 686016  5.3    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553127 14.8     899071 24.1   899071 24.1
Vcells 687794  5.3    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 553154 14.8     899071 24.1   899071 24.1
Vcells 703803  5.4    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554034 14.8     899071 24.1   899071 24.1
Vcells 711327  5.5    1300721 10.0  1162592  8.9
[1] 2
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554038 14.8     899071 24.1   899071 24.1
Vcells 729088  5.6    1300721 10.0  1162592  8.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554039 14.8     899071 24.1   899071 24.1
Vcells 729089  5.6    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554038 14.8     899071 24.1   899071 24.1
Vcells 729088  5.6    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554042 14.8     899071 24.1   899071 24.1
Vcells 732641  5.6    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554038 14.8     899071 24.1   899071 24.1
Vcells 729088  5.6    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554046 14.8     899071 24.1   899071 24.1
Vcells 730867  5.6    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554054 14.8     899071 24.1   899071 24.1
Vcells 732646  5.6    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
```

```
Ncells 554062 14.8     899071 24.1   899071 24.1
Vcells 734425  5.7    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554070 14.8     899071 24.1   899071 24.1
Vcells 736204  5.7    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554074 14.8     899071 24.1   899071 24.1
Vcells 737982  5.7    1300721 10.0  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 554101 14.8     899071 24.1   899071 24.1
Vcells 753991  5.8    1445757 11.1  1286761  9.9
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 559160 15.0     899071 24.1   899071 24.1
Vcells 744570  5.7    1445757 11.1  1445400 11.1
FINISHED!
Total time: 0 minutes
```

To visualize the results per profile we use the `plotProfile` function:

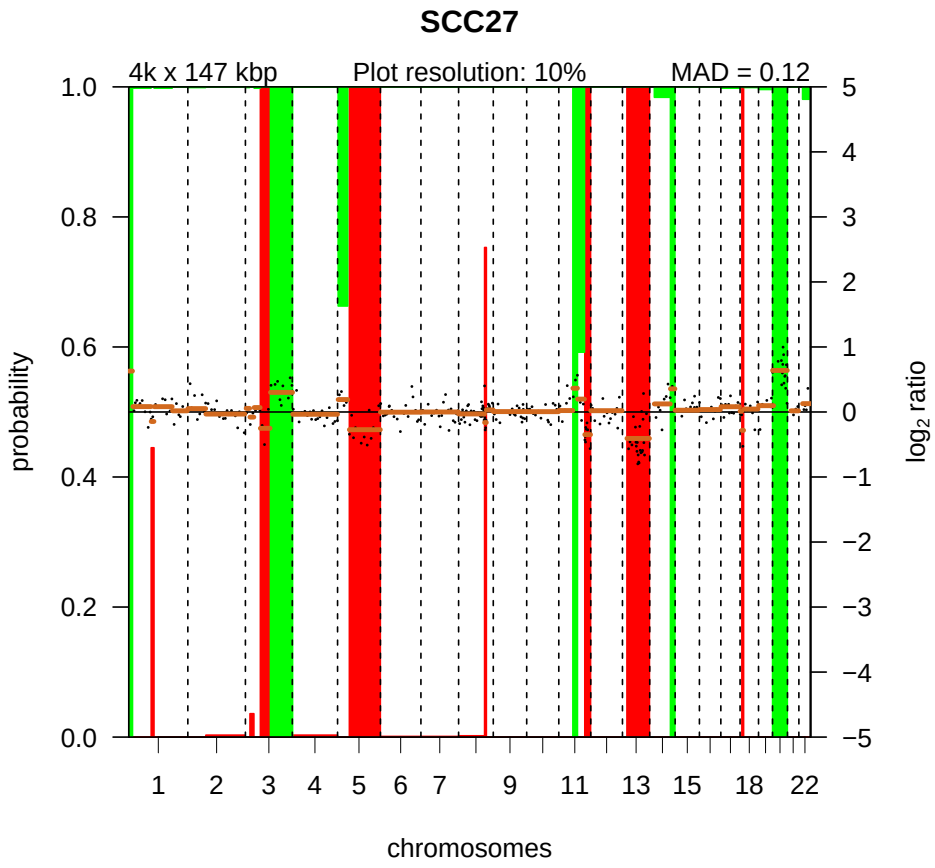
```
> plot(result[,1])

Plotting sample AdCA10
```



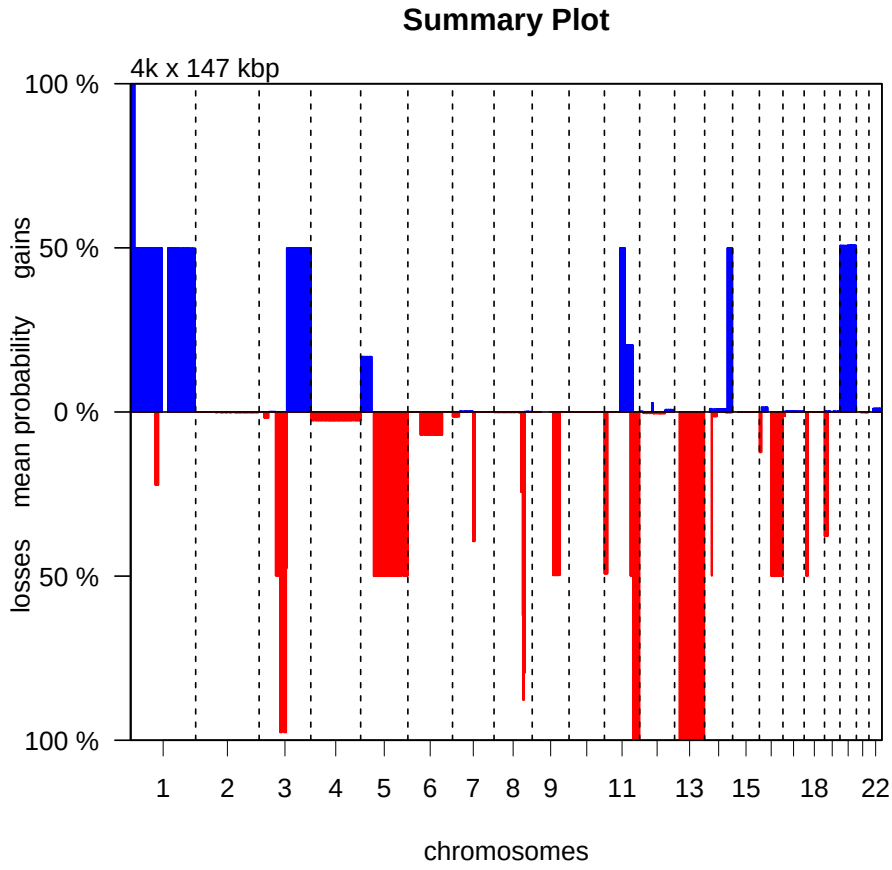
```
> plot(result[,2])

Plotting sample SCC27
```



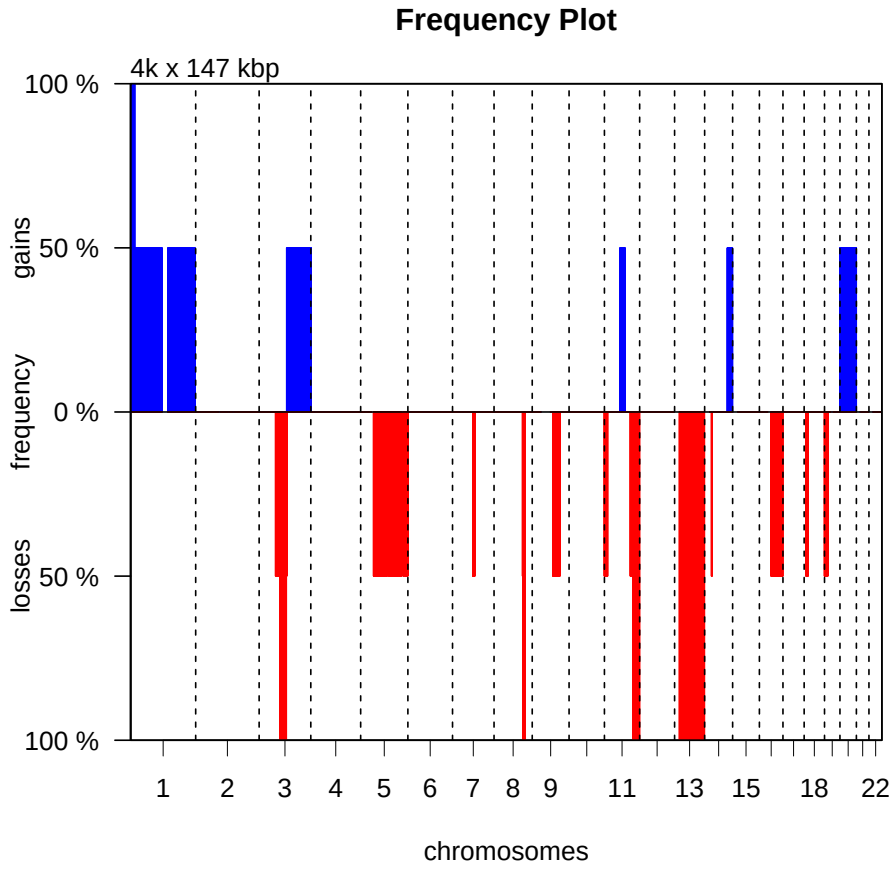
Alternatively, we can create a summary plot of all the samples:

```
> summaryPlot(result)
```



Or a frequency plot::

```
> frequencyPlotCalls(result)
```



References

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. (2001). Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17:520–525.

van de Wiel, M. A., Kim, K. I., Vosse, S. J., van Wieringen, W. N., Wilting, S. M., and Ylstra, B. (2007). CGHcall: calling aberrations for array CGH tumor profiles. *Bioinformatics*, 23:892–894.

Venkatraman, E. S. and Olshen, A. B. (2007). A faster circular binary segmentation algorithm for the analysis of array CGH data. *Bioinformatics*, 23:657–663.

Wilting, S. M., Snijders, P. J. F., Meijer, G. A., Ylstra, B., van den Ijssel, P. R. L. A., Snijders, A. M., Albertson, D. G., Coffa, J., Schouten, J. P., van de Wiel, M. A., Meijer, C. J. L. M., and Steenbergen, R. D. M. (2006). Increased gene copy numbers at chromosome 20q are frequent in both squamous cell carcinomas and adenocarcinomas of the cervix. *J Pathol*, 209:220–230.