KC-SMART Vignette

Jorma de Ronde, Christiaan Klijn

April 15, 2025

1 Introduction

In this Vignette we demonstrate the usage of the `KCsmart` R package. This R implementation is based on the original implementation coded in Matlab and first published in January 2008 [1].

KC-Sor finding recurrent gains and losses from a set of tumor samples measured on an array Comparative Genome Hybridization (aCGH) platform. Instead of single tumor aberration calling, and subsequent minimal common region definition, KC-SMART takes an approach based on the continuous raw log2 values

Briefly: we apply Gaussian locally weighted regression to the summed totals positive and negative log2 ratios separately. This result is corrected for uneven probe spaceing as present on the given array. Peaks in the resulting normalized KC score are tested against a randomly permuted background. Regions of significant recurrent gains and losses are determined using the resulting, multiple testing corrected, significance threshold.

2 Basic Functionality

After installing the `KCsmart` package via Bioconductor, load the KCsmart code into R.

```
> library(KCsmart)
```

Load some sample aCGH data. We supply an example dataset of an old 3000 probe BAC clone experiment. KC-SMART has no problem running on modern 300k+ probe datasets, but this dataset is supplied to keep downloading times at a minimum. This dataset can be a simple dataframe, `DNAcopy` object or a `CGHbase` object, and not contain any missing values. If the data contains missing values, we advise the user to either impute the missing values using the mean of the two neighboring probes or to discard probes containing missing values.

```
> data(hsSampleData)
```

This data is the simplest form available to use in KCSMART. As can be observed using:

```
> str(hsSampleData)
```

```
'data.frame':        3268 obs. of  22 variables:
 $ chrom    : chr  "1" "1" "1" "1" ...
 $ maploc   : num  200000 200050 3284807 4542451 7103443 ...
 $ sample 1 : num  1.326 0.0368 0.4335 1.8879 1.2002 ...
 $ sample 2 : num  1.1155 -0.0478 0.641 0.456 0.831 ...
 $ sample 3 : num  0.4204 0.6486 -0.3638 -0.2573 0.0953 ...
 $ sample 4 : num  0.06313 -0.00108 0.13552 0.57361 -0.09144 ...
 $ sample 5 : num  0.816 0.843 0.893 1.275 0.686 ...
 $ sample 6 : num  0.471 1.262 0.4 0.987 0.725 ...
 $ sample 7 : num  1.296 0.471 0.614 1.007 1.041 ...
 $ sample 8 : num  0.835 0.522 0.643 0.572 0.953 ...
 $ sample 9 : num  1.98 -0.529 0.808 -0.165 1.668 ...
 $ sample 10: num  0.668 0.313 1.083 0.7 1.782 ...
 $ sample 11: num  0.338 -0.118 0.205 1.402 0.299 ...
 $ sample 12: num  0.688 -0.187 0.593 0.941 0.625 ...
 $ sample 13: num  0.4127 0.5349 0.9524 0.9948 0.0954 ...
 $ sample 14: num  0.938 0.254 0.599 0.615 0.466 ...
 $ sample 15: num  0.0481 -0.141 0.9038 0.5821 0.2582 ...
 $ sample 16: num  0.4 -0.306 0.198 0.652 0.589 ...
 $ sample 17: num  1.178 0.72 0.731 1.192 0.287 ...
 $ sample 18: num  0.2878 0.5926 1.3355 0.6192 -0.0581 ...
 $ sample 19: num  0.738 0.489 0.739 1.075 -0.109 ...
 $ sample 20: num  1.552 0.439 0.264 0.841 0.218 ...
```

The user can easily model their data in this format, basically only requiring a column named `chrom` containing the chromosome on which the probe is located and a column named `maploc` containing the chromosomal location of the probe. Alternatively, `DNAcopy`, `CGHraw` and `CGHbase` objects can be used as direct input for the `calcSpm` function discussed below.

KC-SMART uses mirroring of the probes at the ends of chromosomes and near centromeres to prevent signal decay by the convolution. The locations of the centromeres and the lengths of the chromosomes are therefore necessary information. Hard-coded information about the human and mouse genome is supplied, but the user can easily provide the coordinates themselves in R. The 'mirrorLocs' object is a list with vectors containing the start, centromere (optional) and end of each chromosome as the list elements. Additionally it should contain an attribute 'chromNames' listing the chromosome names of each respective list element.

To load the presupplied information use:

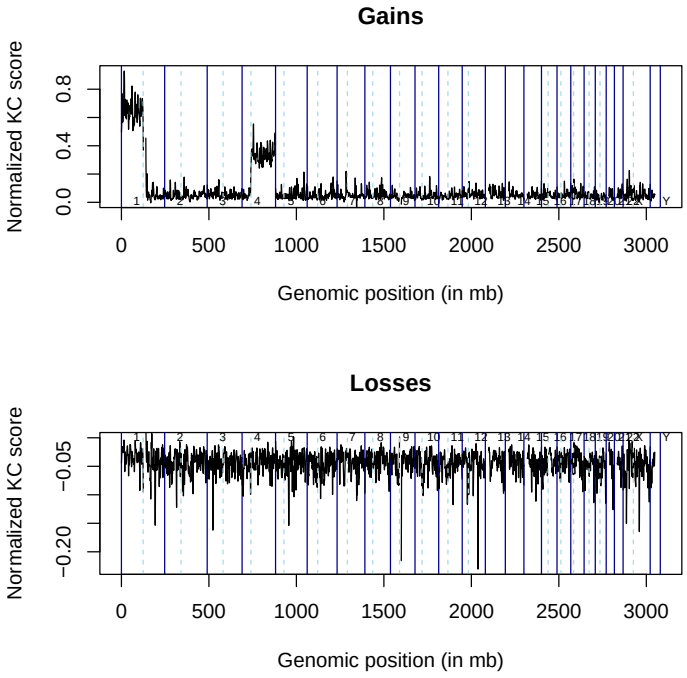
```
> data(hsMirrorLocs)
```

For an analysis in mouse `data(mmMirrorLocs)` is used. Using the `calcSpm` command the convolution is performed at the given parameters and stored in a samplepoint matrix. Here we perform the convolution at two different kernel-widths (first the default sigma = 1Mb, and a second 4Mb sigma). We apply the default parameters to run the convolution.

```
> spm1mb <- calcSpm(hsSampleData, hsMirrorLocs)
> spm4mb <- calcSpm(hsSampleData, hsMirrorLocs, sigma=4000000)
```

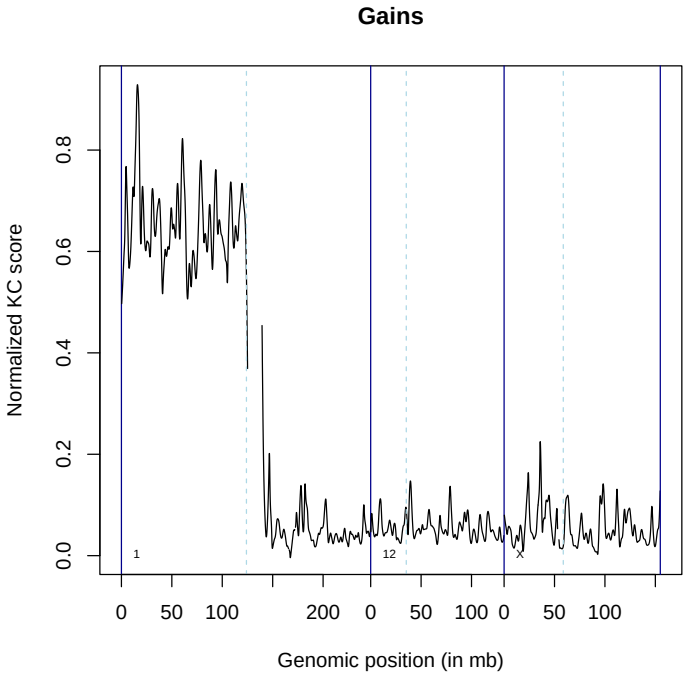
Now we plot the chromosome-wide view of the convolution for both the gains and the losses for the 1Mb kernelwidth analysis. Since the samplepoint matrix we just created is an S4 object we can just apply the plot command to it to visualize the results.

```
> plot(spm1mb)
```



It is possible to select only particular chromosomes in the plot command, and to only display the gains or only the losses. We will now visualize the KCsmart output for gains of chromosomes 1, 12 and X.

```
> plot(spm1mb, chromosomes=c(1, 12, "X"), type="g")
```



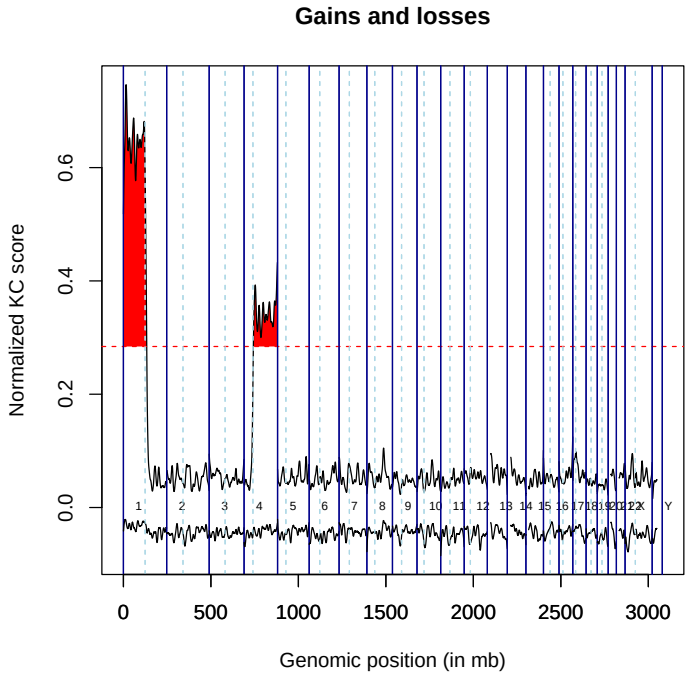
As can be seen, in this particular (synthetic) dataset chromosome 1 shows a large recurrent gain compared to other chromosomes. How significant is this gain? We can determine a significance level based on permutation to determine this. In this example we will permute the data 10 times and determine a threshold at $p < 0.05$. We recommend 1000 permutations if you want to do a definitive analysis of a dataset. Since our dataset is quite low resolution we will use the 4Mb kernel size.

```
> sigLevel4mb <- findSigLevelTrad(hsSampleData, spm4mb, n=10, p=0.05)

[1] "Calculating alpha =  0.05 significance cut-off"
[1] "Found  170  pos peaks and  174  neg peaks in observed sample point matrix"
[1] "Calculating Mirror Positions"
[1] "Starting permutations .."

At iteration 1 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 2 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 3 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"
```

```
At iteration 4 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 5 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 6 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 7 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 8 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 9 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 10 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"
```

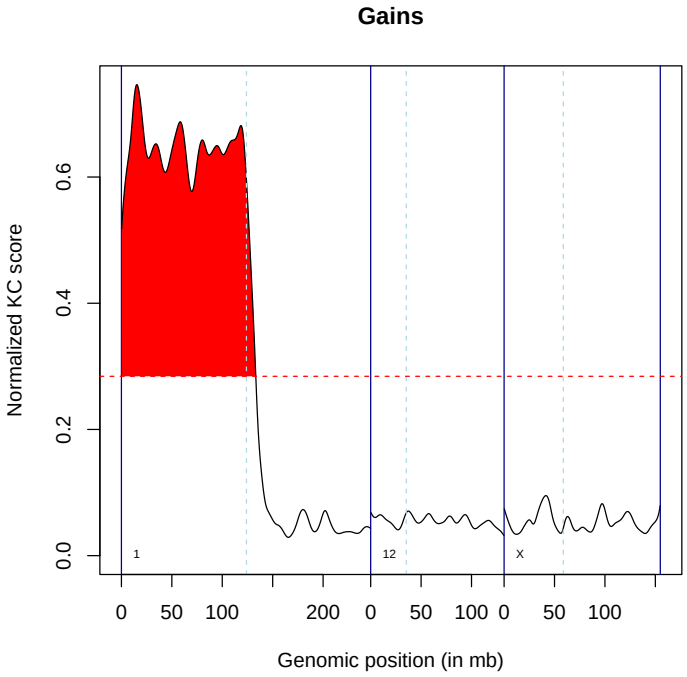
We can now plot the significance level we just calculated in the genomewide view of the KCsmart result using the `sigLevels=` parameter of the `plot` function. This time, we'll use the `type=1` parameter. This will plot the gains and losses in a single frame.

```
> plot(spm4mb, sigLevels=sigLevel4mb, type=1)
```



Let's take a closer look at the gains of chromosome 1, 12 and X again, now with the added significance level.

```
> plot(spm4mb, chromosomes=c(1, 12, "X"), type="g", sigLevels=sigLevel4mb)
```



We've also implemented a less stringently corrected version of the permutation algorithm that was shown above. This approach doesn't calculate the Bonferroni corrected significance level, but the False Discovery Rate (FDR). The command to find the FDR for this particular dataset would be `FDR1mb <- findSigLevelFdr(hsSampleData, spm1mb, n=10, fdrTarget = 0.05)`. In general the FDR is less conservative, and caution must be taken when using this as a threshold.kile

Now we have our significance level we would like to get usable information back about which regions are found to be significantly abberrant and which probes of the original dataset are contained in these regions. Using the `getSigSegments` function we can input a samplepoint matrix and our calculated significance level to get back the relevant information.

```
> sigRegions4mb <- getSigSegments(spm4mb,sigLevel4mb)
```

Now let's have a look at the information contained in `sigRegions1mb`, it is again an S4 object and will display its contents if you just type its name.

```
> sigRegions4mb
```

```
Significantly gained and lost segments
Cutoff value was 0.284240536554693 for gains and -0.152970410488382 for losses
A total of 2 gains and 0 losses were detected

Use 'write.table' to write the file to disk
```

You could write the infomation contained in `sigRegions1mb` to a file using `write.table(sigRegions1mb, file='sig.txt')`. You can also access the

information in R using subsetting. For example, if you wanted to find the probenames of the probes contained in the first significantly gained region you can get them like this:

```
> sigRegions4mb@gains[[1]]$probes
```

```
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
 [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
 [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
 [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
 [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
[109] 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
[127] 127 128 129 130
```

Using `str(sig.regions.1mb)` you can find out more about the way the significant regions are stored in the sigRegions object.

3 Multi-scale analysis

One of the powerful features of KC-SMART is the ability to perform multi-scale analysis. The algorithm can be run using different kernel widths, where small kernel widths will allow you to detect small regions of aberration and vice versa using large kernel widths large regions of aberration can be detected. Remember our convolution step? We calculated the convolution at two sigma's: 1Mb and 4Mb. We will now calculate a significance level for the 1Mb samplepoint matrix and plot the results of both kernelwidths in a scale space figure.

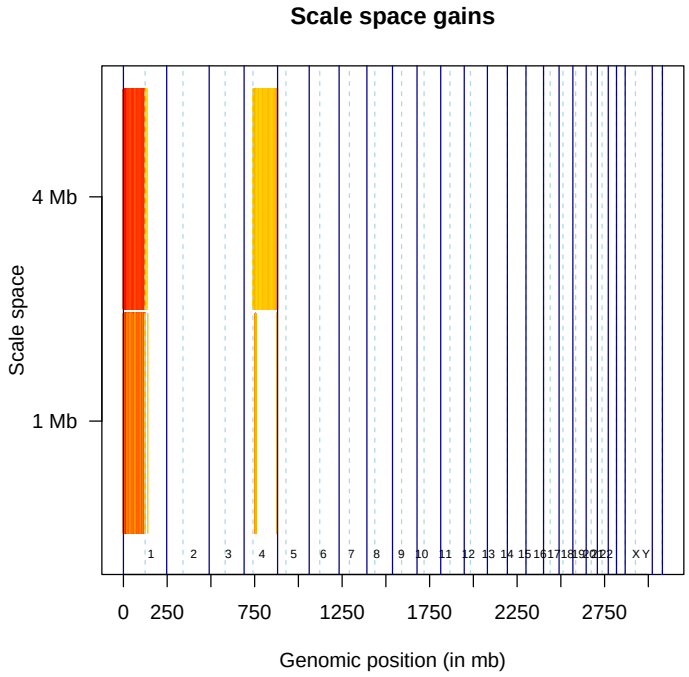
```
> sigLevel1mb <- findSigLevelTrad(hsSampleData, spm1mb, n=10)
```

```
[1] "Calculating alpha =  0.05 significance cut-off"
[1] "Found  580  pos peaks and  600  neg peaks in observed sample point matrix"
[1] "Calculating Mirror Positions"
[1] "Starting permutations .."

At iteration 1 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 2 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 3 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 4 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"
```

```
At iteration 5 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 6 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 7 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 8 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 9 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"

At iteration 10 of 10[1] "Permuting"
[1] "Combining"
[1] "Returning"
```

Now you can plot the scalespace of the two different kernel widths using `plotScaleSpace`. A scale space can show you small and large significant regions in one plot, as well as give you insight within a single significant region. Darker red means more significant. As an option you can plot either the gains or the losses or both. When plotting both, two x11 devices will be opened. Using the 'chromosomes' parameter the chromosomes to be plotted can be set.

```
> plotScaleSpace(list(spm1mb, spm4mb), list(sigLevel1mb, sigLevel4mb), type='g')
```



4 Comparative KC-SMART

Using KC-SMART you can also find regions of significant different copy number change between two specific groups of samples. The resulting regions of significant difference are not required to be significantly recurrent in the entire tumor set. We employ kernel smoothing on a single tumor basis, followed by either a permutation based significance analysis using the signal to noise ratio of the sample point matrices or the use of the siggenes package to determine significance.

The main function of the comparative version of KC-SMART is `calcSpmCollection`, this function requires the data to be in the same form as needed by `calcSpm`, but it also requires a class vector. This is a vector of ones and zeros, with as many elements as there are samples in the dataset. The ones denote one class, and the zeros the other.

Our sample data contains an amplification on chromosome 4 that is specific to a subset of the samples, namely the last 10 samples. We will now calculate the significant difference between the first 10 samples and the last 10 samples in the example dataset using comparative KC-SMART

First we'll want to calculate a sample point matrix collection, the starting point for a comparative analysis. The class vector we use is just 10 zeros followed by 10 ones. Note that this procedure can take a long time if you have a large dataset! Make sure you do it only once. If you have multiple subgroups within a dataset you can define alternative class vectors later in the analysis. You can perform this analysis again on different scales, here we use a sigma of 1 Mb.

```
> spmc1mb <- calcSpmCollection(hsSampleData, hsMirrorLocs, cl=c(rep(0,10),rep(1,10)), sigm

[1] "Mirror locations looking fine"
Processing sample 1 / 20
Processing sample 2 / 20
Processing sample 3 / 20
Processing sample 4 / 20
Processing sample 5 / 20
Processing sample 6 / 20
Processing sample 7 / 20
Processing sample 8 / 20
Processing sample 9 / 20
Processing sample 10 / 20
Processing sample 11 / 20
Processing sample 12 / 20
Processing sample 13 / 20
Processing sample 14 / 20
Processing sample 15 / 20
Processing sample 16 / 20
Processing sample 17 / 20
Processing sample 18 / 20
Processing sample 19 / 20
Processing sample 20 / 20
```

We get a few warnings that some sample points are listed as `NA`. These sample points are the mirrored sample points that fall outside the chromosome boundaries, so they will not be used in the analysis. The next step is to calculate the significance of the diffences between the subgroups. We use the `compareSpmCollection` command for this. Using a parameter called `method` we define the method to determine significance. The option `"perm"` will use class label permutation to define an empirical null-distribution against which the actual differences are measured. The `"siggenes"` option will utilise the siggenes package to determine the significance, this option is both faster and uses less memory.

```
> spmc1mb.sig <- compareSpmCollection(spmc1mb, nperms=3, method=c("siggenes"))
> spmc1mb.sig

Comparison of 20 samples (10 vs. 10)
Using siggenes to find significant regions
```

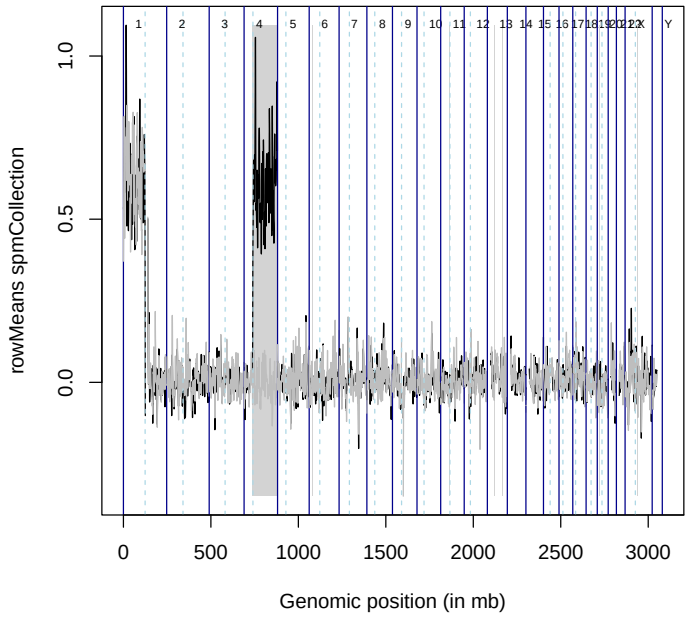
Now we can use the `getSigRegionsCompKC` command to extract the significant regions from the original data `spmc1mb` using the significance information contained in `spmc1mb.sig`.

```
> spmc1mb.sig.regions <- getSigRegionsCompKC(spmc1mb.sig)
> spmc1mb.sig.regions

Significantly different regions using siggenes
delta value was 1.614894 , FDR = 0.01
  startrow endrow chromosome startposition endposition
```

```
1     14770  17195          4       48800001    170050001
2     17207  17604          4      170650001    190500001
3     21641  21662          6       20250001     21300001
4     37261  37266         11       49750001     50000001
5     42375  42403         13       38700001     40100001
6     43300  43323         13       84950001     86100001
7     54439  54457         20       13700001     14600001
8     58787  58809          X       72150001     73250001
Use 'write.table' to write the file to disk
```

The information contained in `spmc1mb.sig.regions` variable can be saved to disk using the `write.table` command if further processing in other programs is required. An overview of the comparative analysis can also be plotted using the `plot` function.

```
> plot(spmc1mb.sig, sigRegions=spmc1mb.sig.regions)
```



5 References

1. Klijn *et. al.* Identification of cancer genes using a statistical framework for multiexperiment analysis of nondiscretized array CGH data. *Nucleic Acids Research.* **36** No. 2, e13.

2. de Ronde *et. al.* KC-SMARTR: An R package for detection of statistically significant aberrations in multi-experiment aCGH data. *BMC Research Notes.* **11** No. 3, 298.