saasCNV: Somatic copy number alteration analysis using sequencing and SNP array data

Zhongyang Zhang and Ke Hao

May 12, 2016

saasCNV is a package for the analysis of somactic copy number alterations (SCNAs) of tumor samples using whole genome/exome sequencing (WGS/WES) and SNP array data. It extracts from the sequencing (SNP array) platform two signal dimensions related to SCNA: 1) total read depth (intensity) reflecting total copy number change; 2) allele specific read depth (intensity) reflecting allelic imbalance as a result of differential copy number changes upon the two alleles. The latter also provides valuable clues for the inference of tumor ploidy and purity. It then carries out joint analysis on these two signal dimensions in both segmentation and calling steps. saasCNV also provides visualzation for diagnosis of intermediate data processing and analysis and illustration of final results.

For more information, see the package website:

`http://zhangz05.u.hpc.mssm.edu/saasCNV/`

To test the following scripts, please make sure you have downloaded the following files from `http://zhangz05.u.hpc.mssm.edu/saasCNV/data/` and put them in the current working directory: `WES_example.vcf.gz`, `vcf_table.txt.gz`, `snp_table.txt.gz`, `refGene_hg19.txt.gz` and `GC_1kb_hg19.txt.gz`.

1 Input data

The analysis pipeline begins with VCF file(s). An example vcf file can be found at

```
> library(saasCNV)
> vcf.file <- "WES_example.vcf.gz"
```

The vcf file contains the information of both tumor and matched normal tissues. Following the header of annotions, the first few rows are something like:

```
  #CHROM   POS          ID REF ALT    QUAL                      FILTER
1   chr1 14907 rs79585140   A   G 1650.44  VQSRTrancheSNP99.50to99.90
2   chr1 14930 rs75454623   A   G 2048.44  VQSRTrancheSNP99.50to99.90
3   chr1 15118 rs71252250   A   G   32.69 VQSRTrancheSNP99.90to100.00

1             AC=2;AF=0.500;AN=4;BaseQRankSum=1.098;DB;DP=187;Dels=0.00;FS=10.732;Haplotyp
2 AC=2;AF=0.500;AN=4;BaseQRankSum=-4.662;DB;DP=193;Dels=0.00;FS=7.379;HaplotypeScore=3.368
```

```
3                 AC=2;AF=0.500;AN=4;BaseQRankSum=-3.577;DB;DP=120;Dels=0.00;FS=0.000;Haplo
             FORMAT            WES_0116_Normal               WES_0116_Tumor
1    GT:AD:DP:GQ:PL  0/1:42,43:85:99:768,0,433     0/1:56,46:97:99:911,0,466
2    GT:AD:DP:GQ:PL  0/1:34,48:78:99:916,0,456   0/1:53,58:106:99:1161,0,748
3 GT:AD:DP:FT:GQ:PL 0/1:42,10:51:rd:49:50,0,49 0/1:53,12:64:gq;rd:11:11,0,54
```

We provide a tool `vcf2txt` to retrieve necessary information from vcf file and convert it to a text table,

```
> vcf_table <- vcf2txt(vcf.file=vcf.file, normal.col=9+1, tumor.col=9+2)
```

The `normal.col` and `tumor.col` specify the columns in which the genotype and read depth information of normal and tumor tissues are located in the vcf file. Note that the first 9 columns in vcf file are mandatory, followed by the information for called variants starting from the 10th column. The resulting `vcf_table` can be also directly loaded

```
> vcf_table <- read.delim(file="vcf_table.txt.gz", as.is=TRUE)
> head(vcf_table)
```

```
  CHROM    POS         ID REF ALT    QUAL    MQ Normal.GT Normal.REF.DP
1  chr1 762589 rs71507461   G   C  898.20 37.90       1/1             2
2  chr1 762592 rs71507462   C   G  880.20 37.90       1/1             2
3  chr1 762601 rs71507463   T   C  831.20 37.45       1/1             1
4  chr1 762632 rs61768173   T   A  618.23 37.39       1/1             1
5  chr1 801943  rs7516866   C   T 1551.44 52.03       0/1            23
6  chr1 808631 rs11240779   G   A 1173.37 54.69       0/1            19
  Normal.ALT.DP Tumor.GT Tumor.REF.DP Tumor.ALT.DP
1            19      1/1            0           14
2            19      1/1            0           14
3            20      1/1            0           12
4            16      1/1            0            8
5            35      0/1            4           22
6            13      1/1            3           24
```

The first 6 columans are self-explanatory, where `CHROM` and `POS` are necessary for subsequent analysis. `QUAL` and `MQ` are quality scores for genotyping and reads mapping, which can be used as filters to exclude variants of poor quality. Starting from the 8th column are genotype, reference allele read depth, alternative allele read depth for normal and tumor respectively.

Then we can transform read depth information into log2ratio and log2mBAF that we use for joint segmentation and CNV calling.

```
> seq.data <- cnv.data(vcf=vcf_table, min.chr.probe=100, verbose=TRUE)
```

```
> head(seq.data)
```

```
   chr position   log2ratio   log2mBAF normal.BAF normal.mBAF  tumor.BAF
1 chr1   801943 -1.55042706 0.48768988  0.6034483   0.6034483 0.84615385
2 chr1   808631 -0.63799828 0.58214749  0.4062500   0.5937500 0.88888889
3 chr1   880390 -0.46327511 0.61095771  0.5238095   0.5238095 0.80000000
4 chr1   881627 -1.39288578 0.03533483  0.7000000   0.7000000 0.50000000
```

```
5 chr1   892460 -0.03924883 0.78386657  0.4444444   0.5555556 0.95652174
6 chr1   898852 -0.39288578 0.24100810  0.2142857   0.7857143 0.07142857
  tumor.mBAF
1  0.8461538
2  0.8888889
3  0.8000000
4  0.7173562
5  0.9565217
6  0.9285714
```

2 Joint segmentation

We employ the algorithm developed by (Zhang et al., 2010) to perform joint segmentation on log2ratio and log2mBAF dimensions. The function `joint.segmentation` outputs the starting and ending points of each CNV segment as well as some summary statistics.

```
> seq.segs <- joint.segmentation(data=seq.data, min.snps=10,
+                                global.pval.cutoff=1e-4, max.chpts=30,
+                                verbose=TRUE)
```

```
> head(seq.segs)
```

```
   chr  posStart    posEnd   length chrIdxStart chrIdxEnd numProbe
1 chr1    801943  16731510 15929568           1       228      228
2 chr1  16890428  17275054   384627         229       281       53
3 chr1  17297289  31426815 14129527         282       496      215
4 chr1  31732602  60503594 28770993         497       927      431
5 chr1  60505783 107870899 47365117         928      1193      266
6 chr1 108113856 120455441 12341586        1194      1352      159
  log2ratio.Mean log2ratio.SD log2ratio.Median log2ratio.MAD log2mBAF.Mean
1     -0.5624744    0.4986741       -0.5949604     0.5336560    0.56069308
2     -0.2978404    0.5491663       -0.2897923     0.4047772   -0.02760742
3     -0.5175220    0.5376744       -0.5184167     0.5337303    0.56839298
4     -0.1095851    0.4770595       -0.0880312     0.4519774    0.16125862
5     -0.4161586    0.5397789       -0.4463434     0.4396174    0.16197340
6     -0.1754930    0.4602296       -0.1518777     0.3573186    0.17596483
  log2mBAF.SD log2mBAF.Median log2mBAF.MAD
1   0.2220726      0.58496250    0.1908776
2   0.2565385     -0.06509503    0.2258865
3   0.2423265      0.60145062    0.2264163
4   0.2409466      0.16551790    0.2318938
5   0.2532963      0.16294034    0.2633028
6   0.2591824      0.17508671    0.2806928
```

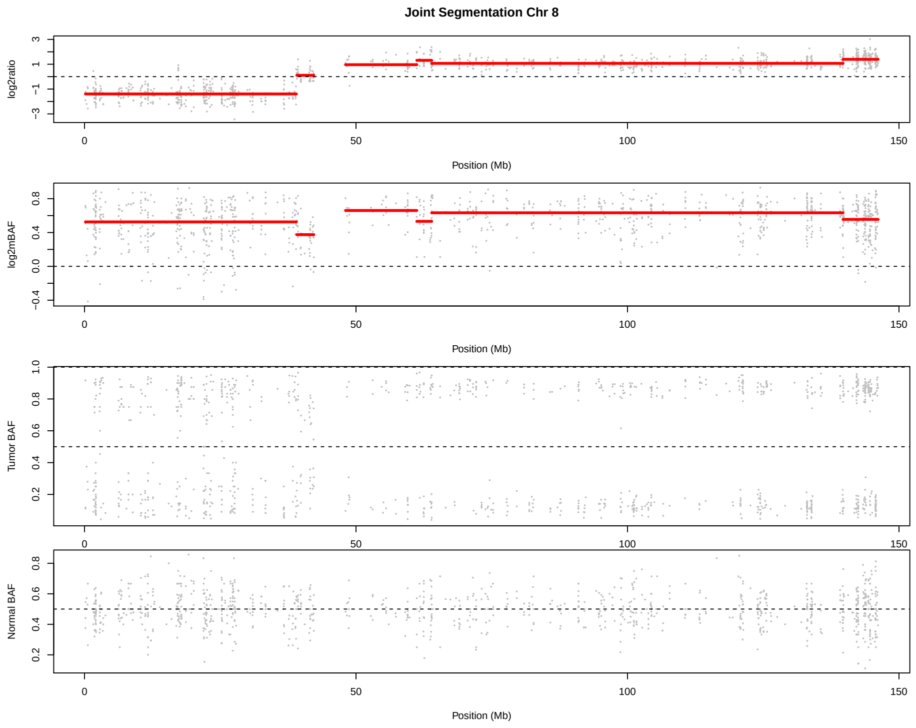
It is an option to merge adjacent segments, for which the median values in either or both dimensions are not substantially different. For WGS and SNP array, it is recommended to do so.

```
> seq.segs.merge <- merging.segments(data=seq.data, segs.stat=seq.segs,
+                                    use.null.data=TRUE,
+                                              N=1000, maxL=2000,
+                                              merge.pvalue.cutoff=0.05, verbose=TRUE)
```

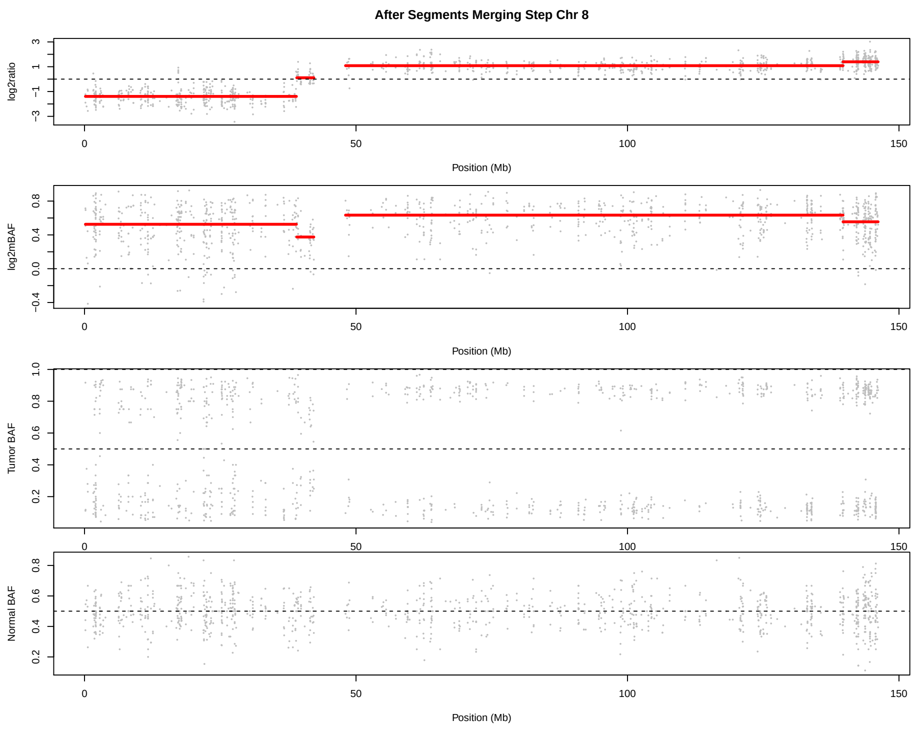
The results from joint segmentation and segments merging can be visualized. This is an example for a chromosome.

```
> data(seq.segs.merge)

> ## joint segmentation
> diagnosis.seg.plot.chr(data=seq.data, segs=seq.segs,
+                          sample.id="Joint Segmentation",
+                          chr=8, cex=0.3)
```



```
> ## merging adjacent segments
> diagnosis.seg.plot.chr(data=seq.data, segs=seq.segs.merge,
+                          sample.id="After Segments Merging Step",
+                          chr=8, cex=0.3)
```



3 CNV calling

Now we can assign SCNA state to each segment directly from joint segmentation or from the results after segments merging step. The baseline adjustment step is incorporated implicitly in the function `cnv.call`.

```
> seq.cnv <- cnv.call(data=seq.data, sample.id="PT116",
+                     segs.stat=seq.segs.merge, maxL=2000, N=1000,
+                     pvalue.cutoff=0.05)
```

```
> head(seq.cnv)
```

```
   chr  posStart    posEnd   length chrIdxStart chrIdxEnd numProbe
1 chr1    801943  16731510 15929568           1       228      228
2 chr1  16890428  17275054   384627         229       281       53
3 chr1  17297289  31426815 14129527         282       496      215
4 chr1  31732602  60503594 28770993         497       927      431
5 chr1  60505783 107870899 47365117         928      1193      266
6 chr1 108113856 120455441 12341586        1194      1352      159
  log2ratio.Mean log2ratio.SD log2ratio.Median log2ratio.MAD log2mBAF.Mean
1     -0.5624744    0.4986741       -0.5949604     0.5336560    0.56069308
2     -0.2978404    0.5491663       -0.2897923     0.4047772   -0.02760742
3     -0.5175220    0.5376744       -0.5184167     0.5337303    0.56839298
4     -0.1095851    0.4770595       -0.0880312     0.4519774    0.16125862
5     -0.4161586    0.5397789       -0.4463434     0.4396174    0.16197340
6     -0.1754930    0.4602296       -0.1518777     0.3573186    0.17596483
```

```
  log2mBAF.SD log2mBAF.Median log2mBAF.MAD Sample_ID remark log2ratio.base.Mean
1   0.2220726      0.58496250    0.1908776     PT116      0           0.1363097
2   0.2565385     -0.06509503    0.2258865     PT116      0           0.1363097
3   0.2423265      0.60145062    0.2264163     PT116      0           0.1363097
4   0.2409466      0.16551790    0.2318938     PT116      0           0.1363097
5   0.2532963      0.16294034    0.2633028     PT116      0           0.1363097
6   0.2591824      0.17508671    0.2806928     PT116      0           0.1363097
  log2ratio.base.Median log2ratio.Sigma log2mBAF.base.Mean log2mBAF.base.Median
1             0.1039458       0.4109074        -0.02477842          -0.02153316
2             0.1039458       0.4109074        -0.02477842          -0.02153316
3             0.1039458       0.4109074        -0.02477842          -0.02153316
4             0.1039458       0.4109074        -0.02477842          -0.02153316
5             0.1039458       0.4109074        -0.02477842          -0.02153316
6             0.1039458       0.4109074        -0.02477842          -0.02153316
  log2mBAF.Sigma log2ratio.Mean.adj log2ratio.Median.adj log2mBAF.Mean.adj
1      0.2104581         -0.6987841           -0.6989062       0.585471500
2      0.2104581         -0.4341501           -0.3937381      -0.002828999
3      0.2104581         -0.6538317           -0.6223625       0.593171401
4      0.2104581         -0.2458948           -0.1919770       0.186037046
5      0.2104581         -0.5524683           -0.5502892       0.186751827
6      0.2104581         -0.3118027           -0.2558235       0.200743256
  log2mBAF.Median.adj log2ratio.p.value log2mBAF.p.value p.value       CNV
1          0.60649566             0.000            0.000   0.000      loss
2         -0.04356187             0.099            0.163   0.091    normal
3          0.62298379             0.016            0.000   0.000      loss
4          0.18705106             0.079            0.000   0.040       LOH
5          0.18447350             0.053            0.000   0.000       LOH
6          0.19661987             0.083            0.000   0.070 undecided
```

A few more columns have been add to `seq.segs.merege`, which summarize the baseline adjusted median log2ratio, log2mBAF, p-values and CNV state for each segment.

Regarding the choise of `pvalue.cutoff`, the study (Zhang and Hao, 2015) provides useful guidance. When the `pvalue.cutoff` varies from 0.001 to 0.05, the sensitivity and specificity are rather stable, ranging around 90%, with smaller p-value favoring relatively higher specificity and lower sensitivity and vice versa. In practice, the users can choose `pvalue.cutoff` within the range from 0.001 to 0.05 depending on their preference for higher sensitivity or specificity.

We also provide an option to add gene annotation to each CNV segment. The RefSeq gene annotation file can be downloaded from UCSC Genome Browser.

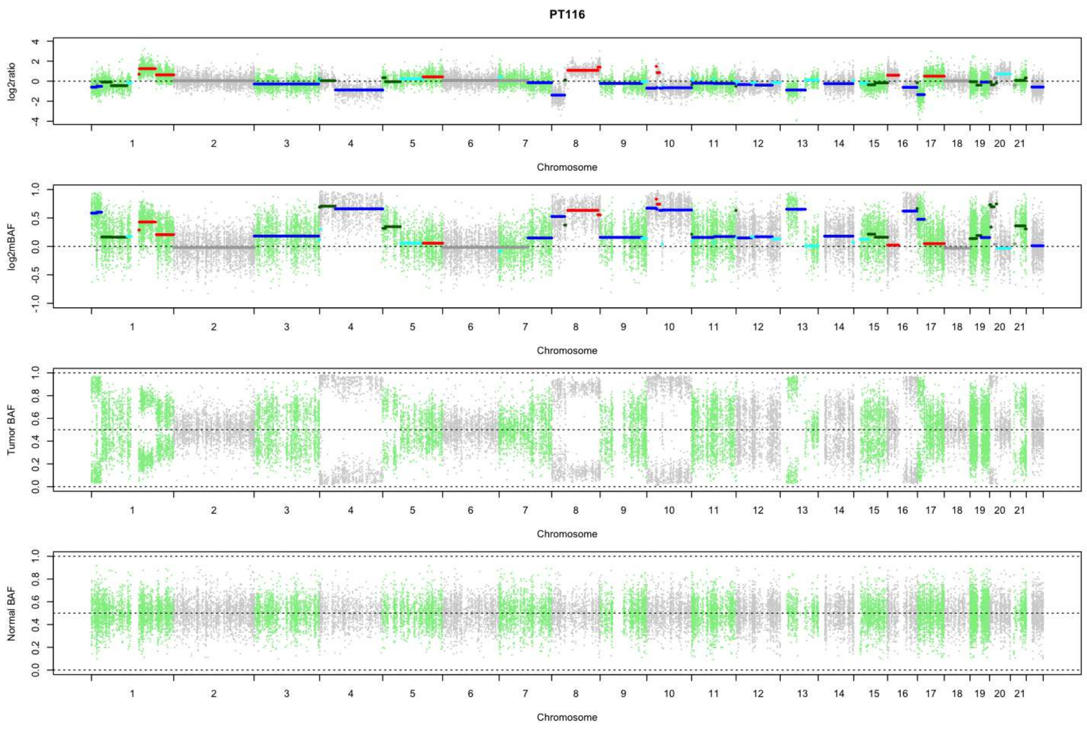
```
> gene.anno.file <- "refGene_hg19.txt.gz"
> gene.anno <- read.delim(file=gene.anno.file, as.is=TRUE, comment.char="")
> seq.cnv.anno <- reannotate.CNV.res(res=seq.cnv, gene=gene.anno, only.CNV=TRUE)
```

4 Visualization of results

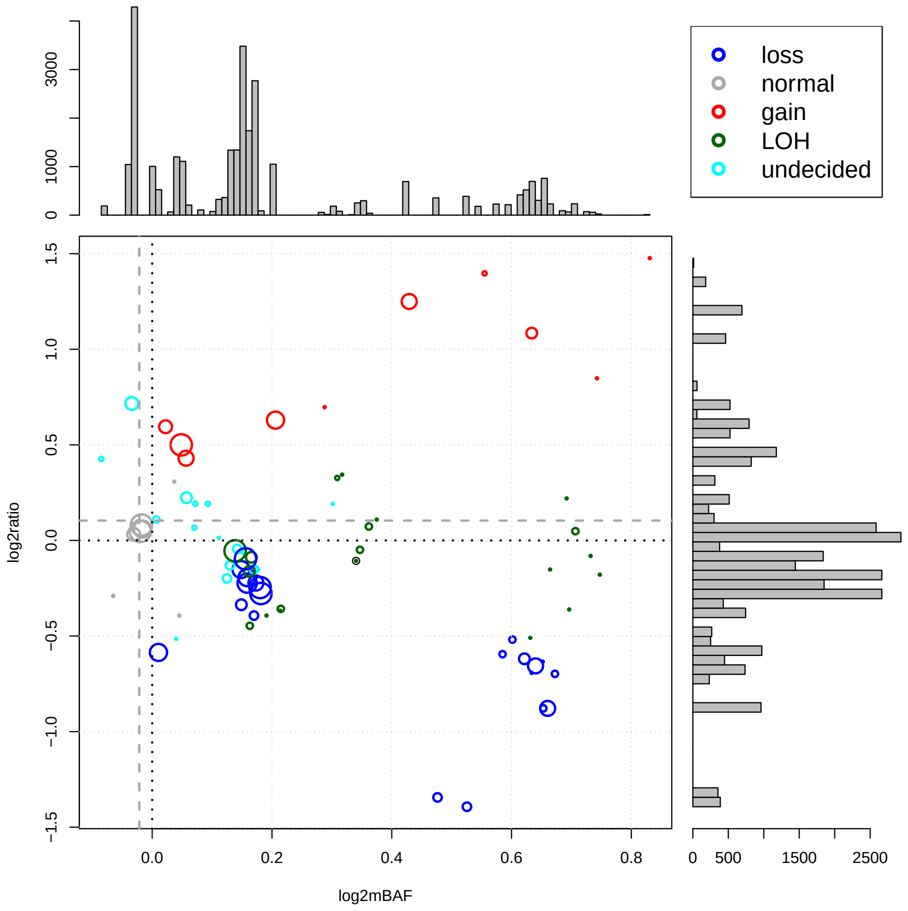
We provide two ways of visualizatio of segmentation and CNV calling results as shown below.

```
> data(seq.cnv)

> ## genome-wide plot
> genome.wide.plot(data=seq.data, segs=seq.cnv,
+                  sample.id="PT116",
+                  chrs=sub("^chr","",unique(seq.cnv$chr)),
+                  cex=0.3)
```



```
> ## cluster plot
> diagnosis.cluster.plot(segs=seq.cnv,
+                        chrs=sub("^chr","",unique(seq.cnv$chr)),
+                        min.snps=10, max.cex=3, ref.num.probe=1000)
```



5 Analysis pipeline

All the above steps are integrate into `NGS.CNV` and can be run altogether. The results, including visualization plots are placed in subdirectories of the output directory `output.dir` as specified by user.

```
> ## NGS pipeline analysis
> vcf_table <- read.delim(file="vcf_table.txt.gz", as.is=TRUE)
> sample.id <- "WES_0116"
> output.dir <- file.path(getwd(), "test_saasCNV")
> NGS.CNV(vcf=vcf_table, output.dir=output.dir, sample.id=sample.id,
+         min.chr.probe=100,
+         min.snps=10,
+         joint.segmentation.pvalue.cutoff=1e-4,
+         max.chpts=30,
+         do.merge=TRUE, use.null.data=TRUE, num.perm=1000, maxL=2000,
+         merge.pvalue.cutoff=0.05,
+         do.cnvcall.on.merge=TRUE,
+         cnvcall.pvalue.cutoff=0.05,
+         do.plot=TRUE, cex=0.3, ref.num.probe=1000,
+          do.gene.anno=TRUE,
+          gene.anno.file="refGene_hg19.txt.gz",
+          seed=123456789,
+          verbose=TRUE)
```

6 SNP array data

The method can be also applied to SNP array data for SCNA analysis with most of the steps being identical to those for NGS data. The input data is slightly different from NGS. Here we provide an example.

```
> snp_table <- read.delim(file="snp_table.txt.gz", as.is=TRUE)
> head(snp_table)

  CHROM     POS         ID REF ALT Normal.GT Normal.LRR Normal.BAF Tumor.GT
1  chr1  768448 rs12562034   G   A       0/1    -0.2072     0.6340      0/0
2  chr1 1005806  rs3934834   C   T       0/0     0.2237     1.0000      0/0
3  chr1 1018704  rs9442372   G   A       0/0    -0.3591     1.0000      0/0
4  chr1 1021415  rs3737728   C   T       0/0    -0.2973     1.0000      0/0
5  chr1 1021695  rs9442398   G   A       0/0     0.4503     0.9993      0/0
6  chr1 1030565  rs6687776   C   T       0/0     0.1752     1.0000      0/0
  Tumor.LRR Tumor.BAF
1   -0.3481    0.9913
2   -0.4176    1.0000
3   -0.6717    1.0000
4   -0.8547    0.9581
5   -0.3396    0.9988
6   -0.2328    0.9983
```

The first 5 columans are the same as NGS data, where `CHROM` and `POS` are necessary for subsequent analysis. Starting from the 6th column are genotype, log R ratio (LRR) and B allele frequency (BAF) for normal and tumor respectively. The information can be extracted from the final report generated by Illumina GenomeStudio.

Then we can transform LRR and BAF information into log2ratio and log2mBAF that we use for joint segmentation and CNV calling.

```
> snp.data <- snp.cnv.data(snp=snp_table, min.chr.probe=100, verbose=TRUE)

> head(snp.data)

   chr position use.in.seg flag log2ratio log2mBAF normal.BAF normal.mBAF
1 chr1   768448          0    0   -0.3481       NA     0.6340       0.634
2 chr1  1005806          0    0   -0.4176       NA     1.0000          NA
3 chr1  1018704          0    0   -0.6717       NA     1.0000          NA
4 chr1  1021415          0    0   -0.8547       NA     1.0000          NA
5 chr1  1021695          0    0   -0.3396       NA     0.9993          NA
6 chr1  1030565          0    0   -0.2328       NA     1.0000          NA
  tumor.BAF tumor.mBAF
1    0.9913         NA
2    1.0000         NA
```

```
3    1.0000         NA
4    0.9581         NA
5    0.9988         NA
6    0.9983         NA
```

The table is basically the same as `seq.data` with two additional columns `use.in.seg` and `flag`. `use.in.seg` indicates whether the probe is to be involved in `joint.segmentation`, `merging.segments`, `cnv.call`, and visualization. `flag` indicates whether there is any issue in the process of converting BAF to mBAF.

As for NGS data analysis, we also integrate all the steps into a function.

```
> ## the pipeline for SNP array analysis
> snp_table <- read.delim(file="snp_table.txt.gz", as.is=TRUE)
> sample.id <- "SNP_0116"
> output.dir <- file.path(getwd(), "test_saasCNV")
> SNP.CNV(snp=snp_table, output.dir=output.dir, sample.id=sample.id,
+         min.chr.probe=100,
+         min.snps=10,
+         joint.segmentation.pvalue.cutoff=1e-4,
+         max.chpts=30,
+         do.merge=TRUE, use.null.data=TRUE, num.perm=1000, maxL=5000,
+         merge.pvalue.cutoff=0.05,
+         do.cnvcall.on.merge=TRUE,
+         cnvcall.pvalue.cutoff=0.05,
+         do.boundary.refine=TRUE,
+         do.plot=TRUE, cex=0.3, ref.num.probe=5000,
+         do.gene.anno=TRUE,
+         gene.anno.file="refGene_hg19.txt.gz",
+         seed=123456789,
+         verbose=TRUE)
```

7 GC content adjustment

When the tumor-normal pair experiment design is properly carried out, the spatial variability in log2ratio signal due to non-uniform GC content distribution and other factors can be effectively neutralized by normalizing tumor data with match normal data. In version 0.3.3 (beta), we provide an optional function `GC.adjust` to adjust for GC content when the log2ratio variability from GC content is not fully neutralized by normal data. In most cases, this step is not necessary. We provide an example file, which summarizes GC content in 1kb window.

```
> gc.file <- "GC_1kb_hg19.txt.gz"
> gc <- read.delim(file = gc.file, as.is=TRUE)
> head(gc)

   chr position   GC
1 chr1    10001 64.6
2 chr1    11001 54.3
```

```
3 chr1    12001 60.0
4 chr1    13001 57.5
5 chr1    14001 58.3
6 chr1    15001 62.0
```

Here is an example to demonstrate how this function works.

```
> ## before GC content adjustment
> data(seq.data)
> head(seq.data)
```

```
   chr position   log2ratio   log2mBAF normal.BAF normal.mBAF  tumor.BAF
1 chr1   801943 -1.55042706 0.48768988  0.6034483   0.6034483 0.84615385
2 chr1   808631 -0.63799828 0.58214749  0.4062500   0.5937500 0.88888889
3 chr1   880390 -0.46327511 0.61095771  0.5238095   0.5238095 0.80000000
4 chr1   881627 -1.39288578 0.03533483  0.7000000   0.7000000 0.50000000
5 chr1   892460 -0.03924883 0.78386657  0.4444444   0.5555556 0.95652174
6 chr1   898852 -0.39288578 0.24100810  0.2142857   0.7857143 0.07142857
  tumor.mBAF
1  0.8461538
2  0.8888889
3  0.8000000
4  0.7173562
5  0.9565217
6  0.9285714
```

```
> ## after GC content adjustment
> seq.data <- GC.adjust(data = seq.data, gc = gc, maxNumDataPoints = 10000)
> head(seq.data)
```

```
      chr position  log2ratio   log2mBAF normal.BAF normal.mBAF  tumor.BAF
2730 chr1   801943 -1.6517163 0.48768988  0.6034483   0.6034483 0.84615385
2732 chr1   808631 -0.6993297 0.58214749  0.4062500   0.5937500 0.88888889
2786 chr1   880390 -0.6854114 0.61095771  0.5238095   0.5238095 0.80000000
2787 chr1   881627 -1.5242841 0.03533483  0.7000000   0.7000000 0.50000000
2788 chr1   892460 -0.2256059 0.78386657  0.4444444   0.5555556 0.95652174
2810 chr1   898852 -0.7533236 0.24100810  0.2142857   0.7857143 0.07142857
     tumor.mBAF   GC log2ratio.woGCAdj
2730  0.8461538 55.5       -1.55042706
2732  0.8888889 52.6       -0.63799828
2786  0.8000000 61.7       -0.46327511
2787  0.7173562 57.3       -1.39288578
2788  0.9565217 60.1       -0.03924883
2810  0.9285714 67.4       -0.39288578
```

After GC content adjustment, the resulting data `seq.data` can be directly fed to downstream analysis shown in previous sections.

8 Manual baseline adjustment

In version 0.3.3 (beta), we add a feature that facilitates users to manually adjust the baseline when automatic adjustment seems to be not correct from a visual check of the diagnosis plot. We embed the function into the functions `merging.segments` and `cnv.call`. For example, after an automatic run of the `NGS.CNV` pipeline, we can manually adjust the baseline by specifying the boundaries within which the "normal" cluster is located and make the SCNA call again.

```
> data(seq.data)
> data(seq.segs.merge)
> seq.cnv <- cnv.call(data=seq.data, sample.id="PT116",
+                      segs.stat=seq.segs.merge, maxL=2000, N=1000,
+                      pvalue.cutoff=0.05,
+                      do.manual.baseline=TRUE,
+                      log2mBAF.left=-0.05, log2mBAF.right=0.05,
+                      log2ratio.bottom=-0.15, log2ratio.up=0.15)
```

References

1. Zhang, Z. and Hao, K. (2015) SAAS-CNV: A joint segmentation approach on aggregated and allele specific signals for the identification of somatic copy number alterations with next-generation sequencing Data. PLoS Computational Biology, 11(11):e1004618.

2. Zhang, N. R., Siegmund, D. O., Ji, H., and Li, J. Z. (2010) Detecting simultaneous changepoints in multiple sequences. Biometrika, 97(3):631-645.