Prepare input data for CINdex

1 Introduction

Genomic instability is known to be a fundamental trait in the development of tumors; and most human tumors exhibit this instability in structural and numerical alterations: deletions, amplifications, inversions or even losses and gains of whole chromosomes or chromosomes arms.

To mathematically and quantitatively describe these alternations we first locate their genomic positions and measure their ranges. Such algorithms are referred to as **segmentation algorithms**. Bioconductor has several copy number segmentation algorithms including ("Copynumber" 2015), ("Fastseg" 2015), ("Vega" 2015), ("SMAP" 2015), ("BiomvRCNS" 2015). There are many copy number segmentation algorithms outside of Bioconductor as well, examples are Fused Margin Regression (FMR)(2010) and Circular Binary Segmentation (CBS)(2004).

Segmentation results are typically have information about the start position and end position in the genome, and the segment value. The algorithms typically covers chromosomes 1 to 22 without any gaps, sometimes sex chromosomes are also included.

2 Preperation of input data for CINdex

2.1 Segment data

The CINdex package can accept output from ANY segmentation algorithm, as long as the data are in the form of a GRangesList object.

Note: The segmentation algorithms will use a **probe annotation file** (that will contain location of the probes), and a **genome reference** file to generate segmentation results. User must note the name and versions of these files, as the same files and versions are needed for CIN analysis.

The segment data used in this example was obtained by applying the Fused Margin Regression (FMR) algorithm to raw copy number and SNP data from Affymetrix SNP 6.0 platform that was on the hg18 human reference genome.

The segment information is stored in form of a GRangesList, with one list element for each sample.

```
#source("http://bioconductor.org/biocLite.R")

library(GenomicRanges)
library(AnnotationHub)
library(pd.genomewidesnp.6)
library(rtracklayer)
library(biovizBase) #needed for stain information
library(CINdex)
library(IRanges)
```

```
#Load example segment data into the workspace.
data("grl.data")
#Examining the class of the object - GRangesList
class(grl.data)
```

```
## [1] "GRangesList"
## attr(,"package")
```

```
## [1] "GenomicRanges"
```

```
#Print first few rows
head(grl.data)
```

```
## GRangesList object of length 6:
## $s1
## GRanges object with 1318 ranges and 1 metadata column:
##          seqnames         ranges strand |            value
##             <Rle>      <IRanges>  <Rle> |        <numeric>
##      [1]        1   [   1,  576]      * | 2.03029610115085
##      [2]        1   [ 577, 1112]      * | 1.94531972410956
##      [3]        1   [1113, 1511]      * |  1.9298187324386
##      [4]        1   [1512, 2113]      * | 1.86864791799073
##      [5]        1   [2114, 2573]      * |   1.938821696368
##      ...      ...            ...    ... .              ...
##   [1314]       22 [19081, 19153]      * | 2.00131227897477
##   [1315]       22 [19154, 20059]      * | 1.95002882899646
##   [1316]       22 [20060, 21926]      * |  2.0095238076147
##   [1317]       22 [21927, 23554]      * | 1.95224808000525
##   [1318]       22 [23555, 24465]      * | 2.01209765172696
##
## ...
## <5 more elements>
## -------
## seqinfo: 22 sequences from an unspecified genome; no seqlengths
```

```
#The names of the list items in the GRangesList
names(grl.data)
```

```
##  [1] "s1"  "s2"  "s3"  "s4"  "s5"  "s6"  "s7"  "s8"  "s9"  "s10"
```

```
# NOTE - The names of the list items in 'grl.data' must match the
# sample names in the clinical data input 'clin.crc'

#Extracting segment information for the sample named "s4" shown as a GRanges object
grl.data[["s4"]]
```

```
## GRanges object with 3392 ranges and 1 metadata column:
##          seqnames         ranges strand |            value
##             <Rle>      <IRanges>  <Rle> |        <numeric>
##      [1]        1   [   1,  552]      * | 1.96193560386905
##      [2]        1   [ 553, 1258]      * | 1.83514425263746
##      [3]        1   [1259, 3011]      * | 1.94194538663918
##      [4]        1   [3012, 3849]      * | 1.89644527376197
##      [5]        1   [3850, 6095]      * | 1.83595753622325
##      ...      ...            ...    ... .              ...
##   [3388]       22 [23734, 23874]      * | 2.03987006989916
##   [3389]       22 [23875, 23936]      * | 1.76662964626522
##   [3390]       22 [23937, 24255]      * | 2.09791747070747
##   [3391]       22 [24256, 24352]      * | 1.72595759143167
##   [3392]       22 [24353, 24465]      * | 1.95292180925646
##   -------
##   seqinfo: 22 sequences from an unspecified genome; no seqlengths
```

```
#You can see that each row in the GRanges object is a segment. The "value" columns shows the
#copy number value for that segment.
```

The object to input into the package: `grl.data`

NOTE: At this time, the CINdex package can only accept segmentation data where probes are in the autosomes (Chromosome 1 - 22). Please remove segment data in the X, Y and mitochondrial chromosomes before input to CINdex.

2.2 Probe annotation file

Use the same platform annotation file used for the segmentation algorithm. The probe annotation file can be obtained in several ways:

2.2.1 Method 1 - Directly from Bioconductor

As an example, we show how to get probe annotation information from Affymetrix SNP 6.0 platform (on hg19 reference genome)

```
#connect to the underlying SQLite database that is part of the pd.genomewidesnp.6 package
con <- db(pd.genomewidesnp.6)
# get the copy number probes
cnv <- dbGetQuery(con, "select man_fsetid, chrom, chrom_start, chrom_stop,
                  strand from featureSetCNV;")
head(cnv, n =3) #print first few rows
```

```
##   man_fsetid chrom chrom_start chrom_stop strand
## 1  CN_477984     1       62140      62165      0
## 2  CN_473963     1       61723      61748      0
## 3  CN_473964     1       61796      61821      0
```

```
#get the SNP probes
snp <- dbGetQuery(con, "select man_fsetid, dbsnp_rs_id, chrom, physical_pos,
                  strand from featureSet;")
head(snp, n=3)
```

```
##      man_fsetid dbsnp_rs_id chrom physical_pos strand
## 1 SNP_A-2131660   rs2887286     1      1156131      0
## 2 SNP_A-1967418   rs1496555     1      2234251      0
## 3 SNP_A-1969580  rs41477744     1      2329564      0
```

Now that we have obtained the probe information, we convert it into a GRanges object

```
#function to convert Copy number data into GRanges object
convert.to.gr.cnv <- function(cnv2) {
    cnv2$chrom <- paste0("chr", cnv2$chrom)
    # subset out SNPs with missing location
    cnv2 <- cnv2[!(is.na(cnv2$chrom) | is.na(cnv2$chrom_start)),]
    # convert strand info to +,-,*
    cnv2$strand[is.na(cnv2$strand)] <- 2
    cnv2$strand <- cnv2$strand + 1
    cnv2$strand <- c("+", "-", "*")[cnv2$strand]
    #convert into GRanges object
    cnv2.gr <- GRanges(cnv2$chrom, IRanges(cnv2$chrom_start,cnv2$chrom_stop),
                       cnv2$strand, ID = cnv2$man_fsetid)
    return(cnv2.gr)
}
```

```
#function to convert SNP data into GRanges object
convert.to.gr.snp <- function(snp2) {

    # make chromosomes the same as a chain file
    snp2$chrom <- paste0("chr", snp2$chrom)
    # subset out SNPs with missing location
    snp2 <- snp2[!(is.na(snp2$chrom) | is.na(snp2$physical_pos)),]
    # convert strand info to +,-,*
    snp2$strand[is.na(snp2$strand)] <- 2
    snp2$strand <- snp2$strand + 1
    snp2$strand <- c("+", "-", "*")[snp2$strand]
    snp2.gr <- GRanges(snp2$chrom, IRanges(snp2$physical_pos,snp2$physical_pos),
                       snp2$strand, ID = snp2$man_fsetid,
                       dbsnp = snp2$dbsnp_rs_id)
    return(snp2.gr)
}
# convert this copy number data from into a GRanges object
cnv.gr <- convert.to.gr.cnv(cnv2 = cnv)
head(cnv.gr, n=3)
```

```
## GRanges object with 3 ranges and 1 metadata column:
##       seqnames         ranges strand |          ID
##          <Rle>      <IRanges>  <Rle> | <character>
##   [1]     chr1 [62140, 62165]      + |   CN_477984
##   [2]     chr1 [61723, 61748]      + |   CN_473963
##   [3]     chr1 [61796, 61821]      + |   CN_473964
##   -------
##   seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

```
# convert this SNP data from into a GRanges object
snp.gr <- convert.to.gr.snp(snp2 = snp)
head(snp.gr, n=3)
```

```
## GRanges object with 3 ranges and 2 metadata columns:
##       seqnames             ranges strand |            ID       dbsnp
##          <Rle>          <IRanges>  <Rle> |   <character> <character>
##   [1]     chr1 [1156131, 1156131]      + | SNP_A-2131660   rs2887286
##   [2]     chr1 [2234251, 2234251]      + | SNP_A-1967418   rs1496555
##   [3]     chr1 [2329564, 2329564]      + | SNP_A-1969580  rs41477744
##   -------
##   seqinfo: 25 sequences from an unspecified genome; no seqlengths
```

Retain only those probes that are located in autosomes

```
#subset only those probes that are in autosomes
snpgr.19.auto <- subset(snp.gr, seqnames(snp.gr) %in% c("chr1",
                                                        "chr2", "chr3","chr4",
                                                        "chr5", "chr6", "chr7",
                                                        "chr8", "chr9", "chr10",
                                                        "chr11", "chr12", "chr13",
                                                        "chr14", "chr15", "chr16",
                                                        "chr17", "chr18","chr19",
                                                        "chr20", "chr21", "chr22"))

#subset only those probes that are in autosomes
cnvgr.19.auto <- subset(cnv.gr, seqnames(cnv.gr) %in% c("chr1",
```

```
                                                    "chr2", "chr3","chr4",
                                                    "chr5", "chr6", "chr7",
                                                    "chr8", "chr9", "chr10",
                                                    "chr11", "chr12", "chr13",
                                                    "chr14", "chr15", "chr16",
                                                    "chr17", "chr18","chr19",
                                                    "chr20", "chr21", "chr22"))

#This gives a total of 1756096 probes (copy number and SNP) on this Affymetrix chip
```

The objects to input into the package : `cnvgr.19.auto` and `snpgr.19.auto`

2.2.2 Method 2 - Get file from Bioconductor (hg19) and convert into required format (hg18)

The example segment data we have is from Affymetrix SNP 6.0 platform (on hg18 reference genome). To get this, we first download the probe annotation on hg19 platform, and convert the data into the hg18 version.

```
con <- db(pd.genomewidesnp.6)
cnv2 <- dbGetQuery(con, "select man_fsetid, chrom, chrom_start,
                   chrom_stop, strand from featureSetCNV;")
snp2 <- dbGetQuery(con, "select man_fsetid, dbsnp_rs_id, chrom,
                   physical_pos, strand from featureSet;")
```

Now that we have obtained the probe information, we convert it into a GRanges object using the functions shown above

```
# convert this copy number data into a GRanges object
cnv2gr <- convert.to.gr.cnv(cnv2 = cnv2)
head(cnv2gr, n=3)
```

```
## GRanges object with 3 ranges and 1 metadata column:
##       seqnames         ranges strand |          ID
##          <Rle>      <IRanges>  <Rle> | <character>
##   [1]     chr1 [62140, 62165]      + |   CN_477984
##   [2]     chr1 [61723, 61748]      + |   CN_473963
##   [3]     chr1 [61796, 61821]      + |   CN_473964
##   -------
##   seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

```
# convert this SNP data into a GRanges object
snp2gr <- convert.to.gr.snp(snp2 = snp2)
head(snp2gr, n=3)
```

```
## GRanges object with 3 ranges and 2 metadata columns:
##       seqnames             ranges strand |            ID       dbsnp
##          <Rle>          <IRanges>  <Rle> |   <character> <character>
##   [1]     chr1 [1156131, 1156131]      + | SNP_A-2131660   rs2887286
##   [2]     chr1 [2234251, 2234251]      + | SNP_A-1967418   rs1496555
##   [3]     chr1 [2329564, 2329564]      + | SNP_A-1969580  rs41477744
##   -------
##   seqinfo: 25 sequences from an unspecified genome; no seqlengths
```

Since our segment data was performed on the hg18 reference genome version, we need our annotations also to be on this reference version. We can use "liftOver" method for this.

```
download.file("http://hgdownload.cse.ucsc.edu/gbdb/hg19/liftOver/hg19ToHg18.over.chain.gz",
              "hg19ToHg18.over.chain.gz")
system("gzip -d hg19ToHg18.over.chain.gz") ## have to decompress
```

```
## now use liftOver from rtracklayer, using the hg19ToHg18.over.chain from UCSC
chain <- import.chain("hg19ToHg18.over.chain")
snpgr.18 <- unlist(liftOver(snp2gr, chain))
```

```
## Discarding unchained sequences: chrMT
```

```
head(snpgr.18, n=3)
```

```
## GRanges object with 3 ranges and 2 metadata columns:
##       seqnames             ranges strand |            ID       dbsnp
##          <Rle>          <IRanges>  <Rle> |   <character> <character>
##   [1]     chr1 [1145994, 1145994]      + | SNP_A-2131660   rs2887286
##   [2]     chr1 [2224111, 2224111]      + | SNP_A-1967418   rs1496555
##   [3]     chr1 [2319424, 2319424]      + | SNP_A-1969580  rs41477744
##   -------
##   seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

```
cnvgr.18 <- unlist(liftOver(cnv2gr, chain))
head(cnvgr.18, n=3)
```

```
## GRanges object with 3 ranges and 1 metadata column:
##       seqnames         ranges strand |          ID
##          <Rle>      <IRanges>  <Rle> | <character>
##   [1]     chr1 [52003, 52028]      + |   CN_477984
##   [2]     chr1 [51586, 51611]      + |   CN_473963
##   [3]     chr1 [51659, 51684]      + |   CN_473964
##   -------
##   seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

```
#subset only those probes that are in autosomes
snpgr.18.auto <- subset(snpgr.18, seqnames(snpgr.18) %in% c("chr1",
                                                            "chr2", "chr3","chr4",
                                                            "chr5", "chr6", "chr7",
                                                            "chr8", "chr9", "chr10",
                                                            "chr11", "chr12", "chr13",
                                                            "chr14", "chr15", "chr16",
                                                            "chr17", "chr18","chr19",
                                                            "chr20", "chr21", "chr22"))

#subset only those probes that are in autosomes
cnvgr.18.auto <- subset(cnvgr.18, seqnames(cnvgr.18) %in% c("chr1",
                                                            "chr2", "chr3","chr4",
                                                            "chr5", "chr6", "chr7",
                                                            "chr8", "chr9", "chr10",
                                                            "chr11", "chr12", "chr13",
                                                            "chr14", "chr15", "chr16",
                                                            "chr17", "chr18","chr19",
                                                            "chr20", "chr21", "chr22"))

#This gives us a total of 1756029 probes (about 1.7 million)

#Save these objects for future
```

```
#save(cnvgr.18.auto, file = "cnvgr.18.auto.RData")
#save(snpgr.18.auto, file = "snpgr.18.auto.RData")
```

The objects to input into the package: `cnvgr.18.auto` and `snpgr.18.auto`

2.2.3 Method 3 - Get annotation file from vendor website, and format it into a GRanges object

In case an annotation file is not available through Bioconductor, one could download it from the vendor web site, and format the file as required. We have outlined the steps in brief.

- download Annotation file from vendor website
- format file with following information
  - All copy number probes:
    - Extract probe name, chromosome number, chromosome start , chromosome end location
    - Format as GRanges object
  - All SNP probes:
    - include probe name, chromosome number and physical location
    - Format as GRanges object

2.3 Reference genome

As mentioned previously, the segment data used in this tutorial was done on the Human reference genome hg18. So we need to download this file. Note that this file must include both **cytoband** information and **stain** information. We use the `biovizBase` package for this.

```
hg18.ucsctrack <- getIdeogram("hg18", cytoband = TRUE)
```

```
## [1] "hg18"
```

```
head(hg18.ucsctrack, n=3)
```

```
## GRanges object with 3 ranges and 2 metadata columns:
##       seqnames             ranges strand |     name gieStain
##          <Rle>          <IRanges>  <Rle> | <factor> <factor>
##   [1]     chr1 [      1, 2300000]      * |   p36.33     gneg
##   [2]     chr1 [2300000, 5300000]      * |   p36.32   gpos25
##   [3]     chr1 [5300000, 7100000]      * |   p36.31     gneg
##   -------
##   seqinfo: 24 sequences from an unspecified genome
```

```
#The user must ensure that the input object is a GRanges object

#Save this object for future use
#save(hg18.ucsctrack, file = "hg18.ucsctrack.RData")

#NOTE - To get the reference file for hg19, use the code below
#hg19IdeogramCyto <- getIdeogram("hg19", cytoband = TRUE)
```

The objects to input into the package: `hg18.ucsctrack`

2.4 Clinical data

The CINdex package allows users to compare the Chromosome CIN and Cytoband CIN values across two groups of patients - a typical use case in translational research studies.

The clinical data input used in the tutorial must have a matrix with two columns. The first column must have the sample ids `"Sample"`, and the second column must have the group labels `"Label"`.

The example dataset consisits of 10 colon cancer patients, of which 5 had relapse (return of cancer to tumor site) and the rest did not relapse. This example dataset is part of the complete dataset used in our published paper (2013), and can be accessed via G-DOC Plus https://gdoc.georgetown.edu(2013).

Load the example clinical data into the workspace.

```
data("clin.crc")

# checking the class of the object
class(clin.crc)
```

```
## [1] "matrix"
```

```
# checking the structure
str(clin.crc)
```

```
##  chr [1:10, 1:2] "s1" "s2" "s3" "s4" "s5" "s6" "s7" ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:2] "Sample" "Label"
```

```
#Let us examine the first five rows of this object
head(clin.crc,5)
```

```
##      Sample Label
## [1,] "s1"   "relapse"
## [2,] "s2"   "relapse"
## [3,] "s3"   "relapse"
## [4,] "s4"   "relapse"
## [5,] "s5"   "relapse"
```

```
# Look at sample names
clin.crc[,1]
```

```
##  [1] "s1"  "s2"  "s3"  "s4"  "s5"  "s6"  "s7"  "s8"  "s9"  "s10"
```

Before you input your own clinical data into the CINdex package, ensure to format your data in this way. *Object to input into CINdex:* `clin.crc`

NOTE - The names of the list items in the GRangesList `grl.data` must match the sample names in the clinical data input i.e `names(grl.data)` must be same as `clin.crc[,1]`

2.5 Gene annotation file

Our CINdex package allows users to compare cytoband CIN values between two groups for patients (control vs case), a typical use case in translational research. Once we get the list of differentially changed cytobands, it would be interesting to see which genes fall in these cytoband regions.

To be able to use this function, a CDS gene annotation file is required. We show an example of how this file can be created using `TxDb` objects from UCSC.

```
##biocLite("TxDb.Hsapiens.UCSC.hg18.knownGene")
##biocLite("Homo.sapiens")
library(TxDb.Hsapiens.UCSC.hg18.knownGene)
library(Homo.sapiens)
```

```
# We will continue to use hg18 gene annotations in this tutorial.
TxDb(Homo.sapiens) <- TxDb.Hsapiens.UCSC.hg18.knownGene
z <- select(Homo.sapiens, keys(Homo.sapiens, "ENTREZID"),
            c("CDSID","CDSCHROM","CDSSTRAND","CDSSTART","CDSEND","SYMBOL"), "ENTREZID")
```

```
## 'select()' returned 1:many mapping between keys and columns
```

```
z1 <- na.omit(object = z) #remove NA values

# extracting only the columns we want as a matrix
geneAnno <- cbind(z1$CDSCHROM, z1$CDSSTRAND,  z1$CDSSTART, z1$CDSEND, z1$SYMBOL)
colnames(geneAnno) <- c("chrom","strand", "cdsStart", "cdsEnd", "GeneName")

#So this gene annotation file looks like this
head(geneAnno, n=3)
```

```
##      chrom   strand cdsStart   cdsEnd     GeneName
## [1,] "chr19" "-"    "63556582" "63556615" "A1BG"
## [2,] "chr19" "-"    "63556470" "63556505" "A1BG"
## [3,] "chr19" "-"    "63556106" "63556375" "A1BG"
```

```
# Examining the class and structure of this oject
class(geneAnno)
```

```
## [1] "matrix"
```

```
str(geneAnno)
```

```
##  chr [1:219244, 1:5] "chr19" "chr19" "chr19" "chr19" "chr19" ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:5] "chrom" "strand" "cdsStart" "cdsEnd" ...
```

```
#Save this object for future use
#save(geneAnno, file = "geneAnno.RData")
```

Before you input your own clinical data into the CINdex package, ensure to format your data in this way. *Object to input into CINdex:* `geneAnno`

References

"BiomvRCNS." 2015. Accessed December 3. http://bioconductor.org/packages/release/bioc/html/biomvRCNS.html.

"Copynumber." 2015. Accessed December 3. http://bioconductor.org/packages/release/bioc/html/copynumber.html.

"Fastseg." 2015. Accessed December 3. http://bioconductor.org/packages/release/bioc/html/fastseg.html.

Feng, Yuanjian, Guoqiang Yu, Tian Li Wang, Ie Ming Shih, and Yue Wang. 2010. "Analysing Dna Copy Number Changes Using Fused Margin Regression." *International Journal of Functional Informatics and*

Personalised Medicine 3, 3-15.

Madhavan, Subha, Yuriy Gusev, Michael Harris, David M Tanenbaum, Robinder Gauba, Krithika Bhuvaneshwar, Andrew Shinohara, et al. 2013. "G-Doc: A Systems Medicine Platform for Personalized Oncology." *Neoplasia* 13(9).

Madhavan, Subha, Yuriy Gusev, Thanemozhi G Natarajan, Lei Song, Krithika Bhuvaneshwar, Robinder Gauba, Abhishek Pandey, et al. 2013. "Genome-Wide Multi-Omics Profiling of Colorectal Cancer Identifies Immune Determinants Strongly Associated with Relapse." *Frontiers in Genetics* 4.

Olshen, Adam B, ES Venkatraman, Robert Lucito, and Michael Wigler. 2004. "Circular Binary Segmentation for the Analysis of Array-Based Dna Copy Number Data." *Biostatistics* 5(4).

"SMAP." 2015. Accessed December 3. http://bioconductor.org/packages/release/bioc/html/SMAP.html.

"Vega." 2015. Accessed December 3. http://bioconductor.org/packages/release/bioc/html/Vega.html.