The OmicCircos usages by examples

Ying Hu and Chunhua Yan

October 13, 2014

Contents

1 Introduction

The OmicCircos is to generate high-quality circular plots for visualizing variations in genomic data. The data can be gene or chromosome position-based values for mutation, copy number variation, expression, and methylation. OmicCircos is capable of displaying variations in scatterplot, line, and text label. The relationships between genomic features can be presented in polygon and curve. By utilizing the statistical and graphic functions in R/Bioconductor environment, OmicCircos is also able to draw boxplot, histogram, and heatmap from multiple sample data.

In this vignette, we will introduce the package plotting functions using simulation data sets and TCGA gene expression and copy umber variation (cnv) data sets (http://www.cancergenome.nih.gov/). A quick way to load the vignette examples is:

```
1 vignette("OmicCircos")
```

2 Input file formats

There are four input data files in the package: segment data, mapping data, linking data and linking polygon data.

2.1 segment data

The first input file `segment data` lays out the foundation for a circular graph. Column 1 should be the segment or chromosome name. Columns 2 and 3 are the start and end points of the segment. Columns 4 and 5 are optional and used as segment or other segment name and description. The package comes with the segment data for human (hg18 and hg19) or mouse (mm9 and mm10). Let's start by loading the package

```
options(stringsAsFactors = FALSE);
library(OmicCircos);
# load the hg18 segment data
data(UCSC.hg18.chr);
# display the first six rows of the data
head(UCSC.hg18.chr);
```

```
## Loading required package:  GenomicRanges
## Loading required package:  BiocGenerics
## Loading required package:  parallel
##
## Attaching package:  'BiocGenerics'
##
## The following objects are masked from 'package:parallel':
##
##    clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##    clusterExport, clusterMap, parApply, parCapply, parLapply,
##    parLapplyLB, parRapply, parSapply, parSapplyLB
##
## The following object is masked from 'package:stats':
##
##    xtabs
##
## The following objects are masked from 'package:base':
##
##    Filter, Find, Map, Position, Reduce, anyDuplicated, append,
##    as.data.frame, as.vector, cbind, colnames, do.call,
##    duplicated, eval, evalq, get, intersect, is.unsorted, lapply,
##    mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##    pmin.int, rank, rbind, rep.int, rownames, sapply, setdiff,
##    sort, table, tapply, union, unique, unlist, unsplit
##
## Loading required package:  S4Vectors
## Loading required package:  stats4
## Loading required package:  IRanges
## Loading required package:  GenomeInfoDb

##   chrom chromStart chromEnd   name gieStain
## 1  chr1          0  2300000 p36.33     gneg
## 2  chr1    2300000  5300000 p36.32   gpos25
## 3  chr1    5300000  7100000 p36.31     gneg
```

```
## 4  chr1    7100000  9200000 p36.23   gpos25
## 5  chr1    9200000 12600000 p36.22     gneg
## 6  chr1   12600000 16100000 p36.21   gpos50
```

2.2 mapping data

The second input file `mapping data` is an R data frame which includes values to be drawn in the graph. Columns 1 and 2 are segment name and position respectively. And the third column is optional which can be the value or name. In the following example, the third column is the gene symbol.

```
options(stringsAsFactors = FALSE);
# load the OmicCircos-package
library(OmicCircos);
# load the gene expression data sets as the mapping data
data(TCGA.BC.gene.exp.2k.60);
# display the first six rows and the first six columns of the data
head(TCGA.BC.gene.exp.2k.60[,c(1:6)]);
```

```
##   chr        po     NAME TCGA.A1.A0SK.01A TCGA.A1.A0SO.01A
## 1   1  939245.5    ISG15           -3.618           -2.286
## 2   1 2533140.5    MMEL1           -2.832           -3.093
## 3   1 6446321.0 TNFRSF25            2.559           -0.660
## 4   1 7832974.5     UTS2            1.708           -0.726
## 5   1 7912164.5  TNFRSF9           -0.189           -0.768
## 6   1 9989253.0     RBP7            0.817           -0.463
##   TCGA.A2.A04W.01A
## 1           -0.998
## 2            1.037
## 3           -0.705
## 4           -0.102
## 5            1.020
## 6           -1.472
```

2.3 linking data

The third file `linking data` is for linking two points. Columns 1 and 4 are segment names for the two anchor points. Columns 2 and 5 are the point positions on the segments. Columns 3 and 6 are the names of the two points. Column 7 is optional and could be used for the link type description.

```
options(stringsAsFactors = FALSE);
# load the OmicCircos-package
library(OmicCircos);
# load the gene fusion data sets as the linked data
data(TCGA.BC.fus);
# display the first six rows and the first six coloumns of the data
head(TCGA.BC.fus[,c(1:6)]);
```

```
##   chr1      po1    gene1 chr2       po2    gene2
## 1    2 63456333    WDPCP   10  37493749 ANKRD30A
```

```
## 2   18 14563374   PARD6G   21  14995400    POTED
## 3   10 37521495 ANKRD30A    3  49282645   CCDC36
## 4   10 37521495 ANKRD30A    7 100177212    LRCH4
## 5   18 18539803    ROCK1   18    112551   PARD6G
## 6   12  4618159  C12orf4   18   1514414   PARD6G
```

2.4 linking polygon data

The last input file `linking polygon data` is for linking two sub-segments. In the data frame, columns 1, 2 and 3 are name, start and end points for first segment and columns 4, 5 and 6 are for the second segment in the same order. Here is an example by homological analysis between human and mouse. One row is one of the homological segments between the species.

```
options(stringsAsFactors = FALSE);
# load the OmicCircos-package
library(OmicCircos);
# load the genome comparison data sets as the linked data
data(UCSC.hs_cyto_mm);
# display the first six rows of the data
head(UCSC.hs_cyto_mm);
```

```
##   hs.chr hs.start   hs.end mm.chr  mm.start    mm.end
## 1      1   168564 19853547      4 155606060 137469648
## 2      1 19999359 19853547     14 119903262 124902244
## 3      1 20114146 42595152      4 137324970 117524887
## 4      1 42702806 42595152      2 156994785 117524887
## 5      1 42707481 43332813      4 117433768 116843734
## 6      1 43355885 43332813      7 100327829 116843734
```

3 The package functions

There are three main functions in the package: sim.circos, seg.engle.po and circos.

3.1 sim.circos

The sim.circos function is a simulation function that is used to generate the four input files. In the following example, there are 10 segments, 10 individuals, 10 links and 10 link.pgs. For each segment, the range of the point number is from 20 to 50. The values will be generated by rnorm(1) + i. The i is the ordinal number of the segments. The values are increased by the segment order.

```
options(stringsAsFactors = FALSE);
# load the OmicCircos-package
library(OmicCircos);
# set up the initial parameters
seg.num     ← 10;
ind.num     ← 20;
seg.po      ← c(20:50);
link.num    ← 10;
link.pg.num ← 10;
# run sim.circos function
sim.out     ← sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num, link.pg=
    link.pg.num);
# display the data set names
names(sim.out)
# display the segment data
head(sim.out$seg.frame[,c(1:3)])
```

```
##   seg.name seg.Start seg.End
## 1     chr1         0       1
## 2     chr1         1       2
## 3     chr1         2       3
## 4     chr1         3       4
## 5     chr1         4       5
## 6     chr1         5       6
```

```
# display the mapping data
head(sim.out$seg.mapping[,c(1:5)])
```

```
##   seg.name seg.po name1  name2  name3
## 1     chr1      1 1.784  0.447 -0.363
## 2     chr1      2 0.786  0.263  0.947
## 3     chr1      3 2.289  1.389  2.725
## 4     chr1      4 0.406  3.139 -0.086
## 5     chr1      5 1.028 -0.358 -0.365
## 6     chr1      6 2.785  1.799 -0.164
```

```
# display the linking data
head(sim.out$seg.link)
```

```
##    seg1 po1 name1 seg2 po2 name2 name3
## 1  chr4  18    n1 chr6  15    n1    n1
## 2  chr7  25    n2 chr4   3    n2    n2
## 3  chr7   6    n3 chr3  29    n3    n3
## 4 chr10   6    n4 chr9   0    n4    n4
## 5  chr6  11    n5 chr9  34    n5    n5
## 6  chr2  16    n6 chr1  20    n6    n6
```

```
# display the linking polygon data
head(sim.out$seg.link.pg)
```

```
##   seg1 start1 end1 seg2 start2 end2
## 1 chr2     14    2 chr7      8   30
## 2 chr2     14   10 chr7     12   13
## 3 chr9     19   36 chr5     23   24
## 4 chr6     16    0 chr3     29   31
## 5 chr6      8   25 chr8     16   38
## 6 chr4     22   37 chr9     19   17
```

3.2 segAnglePo

The segAnglePo function converts the segment pointer positions into angle values and returns a data frame.

```
options(stringsAsFactors = FALSE);
# load the OmicCircos-package
library(OmicCircos);
# set up the initial parameters
seg.num     ← 10;
ind.num     ← 20;
seg.po      ← c(20:50);
link.num    ← 10;
link.pg.num ← 10;
# run sim.circos function
sim.out     ← sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num, link.pg=
    link.pg.num);
# To get the segment data
seg.f    ← sim.out$seg.frame;
# rename the segment names
seg.name ← paste("chr", 1:seg.num, sep="");
# run segAnglePo function
db       ← segAnglePo(seg.f, seg=seg.name);
db[,2]   ← round(as.numeric(db[,2]), 3);
db[,3]   ← round(as.numeric(db[,3]), 3);
# To display the result
db
```

```
##       seg.name angle.start angle.end seg.sum.start seg.sum.end seg.start
##  [1,] "chr1"   "270"       "296.077" "0"           "26"        "0"
##  [2,] "chr2"   "298.077"   "319.139" "26"          "47"        "0"
##  [3,] "chr3"   "321.139"   "356.242" "47"          "82"        "0"
```

```
##  [4,] "chr4"   "358.242"   "397.357" "82"          "121"       "0"
##  [5,] "chr5"   "399.357"   "449.504" "121"         "171"       "0"
##  [6,] "chr6"   "451.504"   "478.584" "171"         "198"       "0"
##  [7,] "chr7"   "480.584"   "503.652" "198"         "221"       "0"
##  [8,] "chr8"   "505.652"   "536.743" "221"         "252"       "0"
##  [9,] "chr9"   "538.743"   "579.864" "252"         "293"       "0"
## [10,] "chr10"  "581.864"   "628"     "293"         "339"       "0"
##       seg.end
##  [1,] "26"
##  [2,] "21"
##  [3,] "35"
##  [4,] "39"
##  [5,] "50"
##  [6,] "27"
##  [7,] "23"
##  [8,] "31"
##  [9,] "41"
## [10,] "46"
```

In the above example, there are 10 segments, one segment per row. Column 1 is segment name. Columns 2, 3 are the start and end angles of the segment. Column 4 and 5 are the accumulative start and end points. Column 6 and 7 are the start and end point for the segment. The plotting is clockwise and the start pointer is at 12 oclock (270 degree) only.

3.3 circos

The circos is the main function to draw different shapes of the circle. The shapes can be scatterplots, lines and polygons. The function supports multiple samples to draw boxplot, histogram, multiple lines and heatmap. The detail usages of the function are in next sections.

4 Plotting parameters

4.1 basic plotting

The input data sets were generated by sim.circos function.

```
options(stringsAsFactors = FALSE);
library(OmicCircos);
options(stringsAsFactors = FALSE);
set.seed(1234);

# initial
seg.num     ← 10;
ind.num     ← 20;
seg.po      ← c(20:50);
link.num    ← 10;
link.pg.num ← 10;

sim.out ← sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num,
  link.pg=link.pg.num);

seg.f     ← sim.out$seg.frame;
seg.v     ← sim.out$seg.mapping;
link.v    ← sim.out$seg.link
link.pg.v ← sim.out$seg.link.pg
seg.num   ← length(unique(seg.f[,1]));

# name segment (option)
seg.name ← paste("chr", 1:seg.num, sep="");
db       ← segAnglePo(seg.f, seg=seg.name);
# set transparent colors
colors   ← rainbow(seg.num, alpha=0.5);
```

To create square figure and get perfect circle is by the same values in width, height and in the same values in the numbers of lines of the margin.

```
par(mar=c(2, 2, 2, 2));
#
plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
#
circos( R=400, cir=db, type="chr",  col=colors, print.chr.lab=T, W=4, scale=T);
circos(R=360, cir=db, W=40, mapping=seg.v, col.v=3, type="l",   B=T, col=colors[1],
    lwd=0.1, scale=T);
circos(R=320, cir=db, W=40, mapping=seg.v, col.v=3, type="ls",  B=F, col=colors[3],
    lwd=0.1, scale=T);
circos(R=280, cir=db, W=40, mapping=seg.v, col.v=3, type="lh",  B=T, col=colors[7],
    lwd=0.1, scale=T);
circos(R=240, cir=db, W=40, mapping=seg.v, col.v=19, type="ml",  B=F, col=colors, lwd
    =0.1, scale=T);
circos(R=200, cir=db, W=40, mapping=seg.v, col.v=19, type="ml2", B=T, col=colors, lwd
    =0.1);
circos(R=160, cir=db, W=40, mapping=seg.v, col.v=19, type="ml3", B=F, cutoff=5, lwd=0
    .1);
circos(R=150, cir=db, W=40, mapping=link.v, type="link", lwd=2, col=colors);
circos(R=150, cir=db, W=40, mapping=link.pg.v, type="link.pg", lwd=2, col=colors);
```

Figure 1

In figure 2, from outside to inside, the first track is the boxplot by the samples from column 8 (col.v=8) to the last column sample with the scale. Track 2 is the histogram (in horizontal) for multiple samples. Track 3 is the scatter plots for multiple samples. Tracks 4, 5 and 6 are the histogram (in vertical), scatter plot and vertical line by one sample without the scale.

```
options(stringsAsFactors = FALSE);
library(OmicCircos);
set.seed(1234);

## initial
seg.num     ← 10;
ind.num     ← 20;
seg.po      ← c(20:50);
link.num    ← 10;
link.pg.num ← 10;

sim.out ← sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num,
  link.pg=link.pg.num);

seg.f     ← sim.out$seg.frame;
seg.v     ← sim.out$seg.mapping;
link.v    ← sim.out$seg.link
link.pg.v ← sim.out$seg.link.pg
seg.num   ← length(unique(seg.f[,1]));

## select segments
seg.name ← paste("chr", 1:seg.num, sep="");
db       ← segAnglePo(seg.f, seg=seg.name);

colors   ← rainbow(seg.num, alpha=0.5);
```

```
par(mar=c(2, 2, 2, 2));

plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
circos(R=400, type="chr", cir=db, col=colors, print.chr.lab=T, W=4, scale=T);
circos(R=360, cir=db, W=40, mapping=seg.v, col.v=8, type="box",  B=T, col=colors[1],
    lwd=0.1, scale=T);
circos(R=320, cir=db, W=40, mapping=seg.v, col.v=8, type="hist", B=T, col=colors[3],
    lwd=0.1, scale=T);
circos(R=280, cir=db, W=40, mapping=seg.v, col.v=8, type="ms",   B=T, col=colors[7],
    lwd=0.1, scale=T);
circos(R=240, cir=db, W=40, mapping=seg.v, col.v=3, type="h",  B=F,  col=colors[2],
    lwd=0.1);
circos(R=200, cir=db, W=40, mapping=seg.v, col.v=3, type="s", B=T, col=colors, lwd=0.1
    );
circos(R=160, cir=db, W=40, mapping=seg.v, col.v=3, type="b", B=F, col=colors, lwd=0.1
    );
circos(R=150, cir=db, W=40, mapping=link.v, type="link", lwd=2, col=colors);
circos(R=150, cir=db, W=40, mapping=link.pg.v, type="link.pg", lwd=2, col=colors);
```

In figure 3, from outside to inside, track 1 is the three lines for quantile values by the samples from column 8 (col.v=8) with the scale. The middle line is for the median, the outside line is for 90% (or 75% if using type=<93>quant75<94>) and the inside line is for 1-90%. Track 2 is the circle points with the center=median and radium=variance. Track 3 is the circle plot with the center equal to the mean and scaled value (for example, the range from 0 to 3). Tracks 4 is the heatmap.Track 5 is the circle plot with the center=median and radium=standard deviation. Track 6 is the 95%

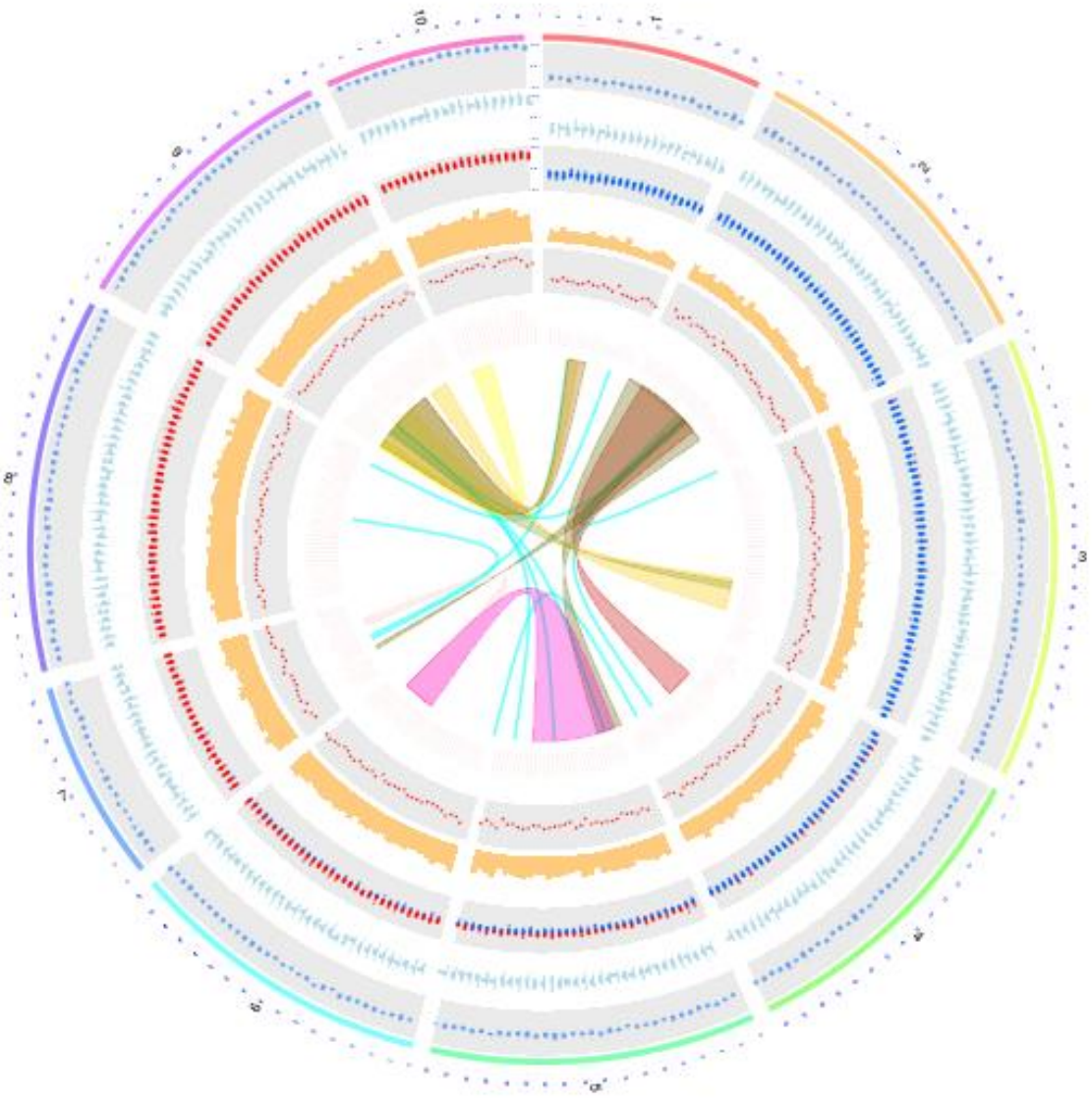

Figure 2

confidence interval of the samples.

```
options(stringsAsFactors = FALSE);
library(OmicCircos);
set.seed(1234);

## initial
seg.num      ← 10;
ind.num      ← 20;
seg.po       ← c(20:50);
link.num     ← 10;
link.pg.num ← 10;

sim.out ← sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num,
  link.pg=link.pg.num);

seg.f      ← sim.out$seg.frame;
seg.v      ← sim.out$seg.mapping;
link.v     ← sim.out$seg.link
link.pg.v ← sim.out$seg.link.pg
seg.num    ← length(unique(seg.f[,1]));

##
seg.name ← paste("chr", 1:seg.num, sep="");
db       ← segAnglePo(seg.f, seg=seg.name);

colors   ← rainbow(seg.num, alpha=0.5);
```

```
par(mar=c(2, 2, 2, 2));
plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");

circos(R=400, type="chr", cir=db, col=colors, print.chr.lab=T, W=4, scale=T);
circos(R=360, cir=db, W=40, mapping=seg.v, col.v=8, type="quant90", B=F, col=colors,
    lwd=0.1, scale=T);
circos(R=320, cir=db, W=40, mapping=seg.v, col.v=3, type="sv", B=T, col=colors[7], lwd
    =0.1, scale=T);
circos(R=280, cir=db, W=40, mapping=seg.v, col.v=3, type="ss", B=F, col=colors[3], lwd
    =0.1, scale=T);
circos(R=240, cir=db, W=40, mapping=seg.v, col.v=8, type="heatmap", lwd=3);
circos(R=200, cir=db, W=40, mapping=seg.v, col.v=3, type="s.sd", B=F, col=colors[4],
    lwd=0.1);
circos(R=160, cir=db, W=40, mapping=seg.v, col.v=3, type="ci95", B=T, col=colors[4],
    lwd=0.1);
circos(R=150, cir=db, W=40, mapping=link.v, type="link", lwd=2, col=colors);
circos(R=150, cir=db, W=40, mapping=link.pg.v, type="link.pg", lwd=2, col=colors);

the.col1=rainbow(10, alpha=0.3)[3];
highlight ← c(160, 410, 6, 2, 6, 10, the.col1, the.col1);
circos(R=110, cir=db, W=40, mapping=highlight, type="hl", lwd=2);

the.col1=rainbow(10, alpha=0.01)[3];
the.col2=rainbow(10, alpha=0.8)[1];
highlight ← c(160, 410, 6, 12, 7, 10, the.col1, the.col2);
circos(R=110, cir=db, W=40, mapping=highlight, type="hl", lwd=2);
```

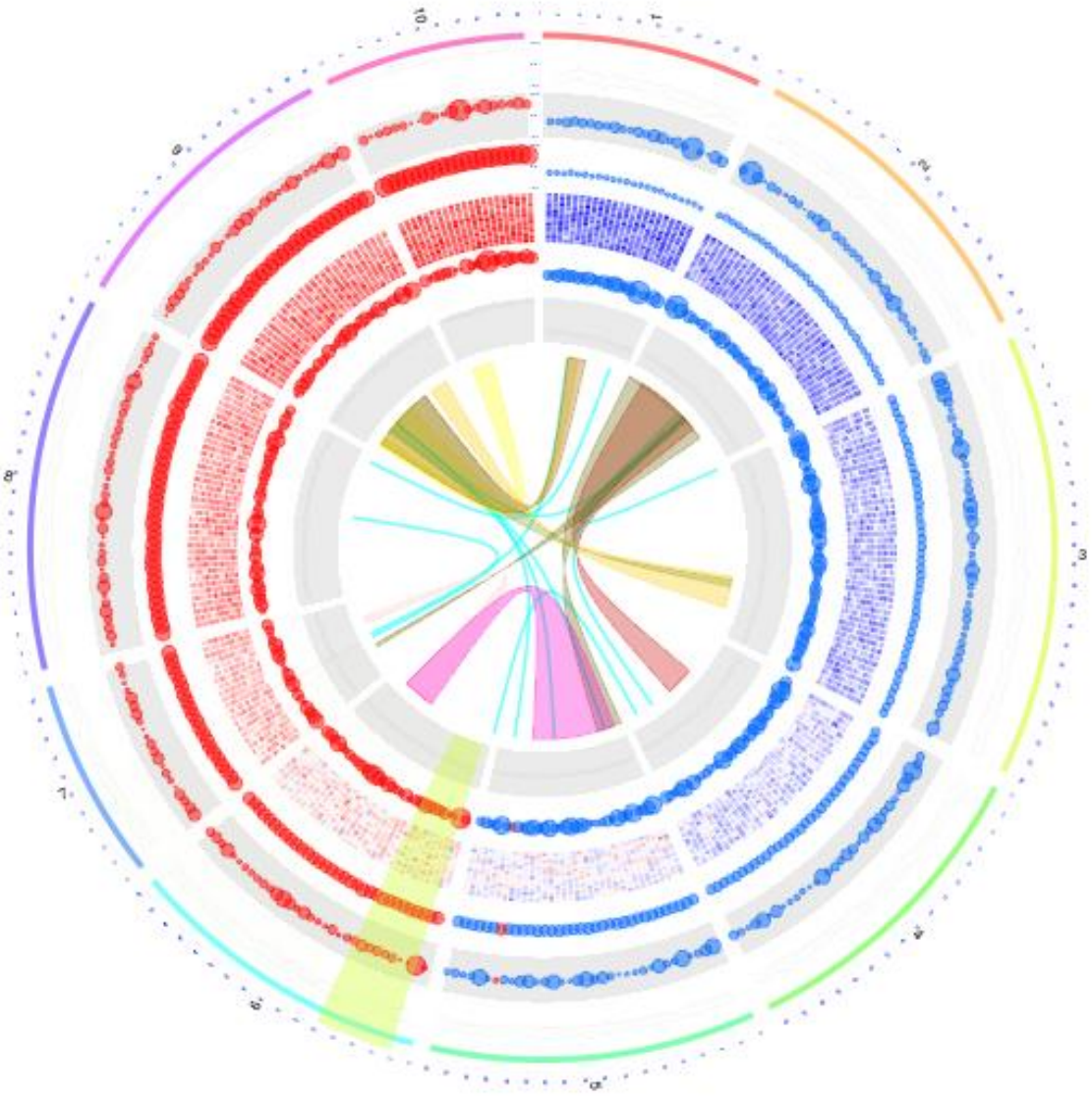

Figure 3

4.2 lables

An example of adding outside label.

```
options(stringsAsFactors = FALSE);
library(OmicCircos);

data("TCGA.PAM50_genefu_hg18");
data("TCGA.BC.fus");
data("TCGA.BC.cnv.2k.60");
data("TCGA.BC.gene.exp.2k.60");
data("TCGA.BC.sample60");
data("TCGA.BC_Her2_cnv_exp");

pvalue ← -1 * log10(TCGA.BC_Her2_cnv_exp[,5]);
pvalue ← cbind(TCGA.BC_Her2_cnv_exp[,c(1:3)], pvalue);

Her2.i ← which(TCGA.BC.sample60[,2] == "Her2");
Her2.n ← TCGA.BC.sample60[Her2.i,1];

Her2.j ← which(colnames(TCGA.BC.cnv.2k.60) %in% Her2.n);
cnv    ← TCGA.BC.cnv.2k.60[,c(1:3,Her2.j)];
cnv.m  ← cnv[,c(4:ncol(cnv))];
cnv.m[cnv.m >  2] ← 2;
cnv.m[cnv.m < -2] ← -2;
cnv ← cbind(cnv[,1:3], cnv.m);

Her2.j   ← which(colnames(TCGA.BC.gene.exp.2k.60) %in% Her2.n);
gene.exp ← TCGA.BC.gene.exp.2k.60[,c(1:3,Her2.j)];
colors ← rainbow(10, alpha=0.5);
```

```
par(mar=c(2, 2, 2, 2));
plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
circos(R=300, type="chr", cir="hg18", print.chr.lab=F, W=4);
circos(R=310, cir="hg18", W=20, mapping=TCGA.PAM50_genefu_hg18, type="label", side="
    out", col="black");
circos(R=250, cir="hg18", W=50, mapping=cnv, col.v=4, type="ml3", B=F, col=colors[7],
    cutoff=0, scale=T);
circos(R=200, cir="hg18", W=50, mapping=gene.exp, col.v=4, type="ml3", B=T, col=colors
    [3], cutoff=0, scale=T);
circos(R=140, cir="hg18", W=50, mapping=pvalue, col.v=4, type="l", B=F, col=colors[1],
     scale=T);
circos(R=132, cir="hg18", W=50, mapping=TCGA.BC.fus, type="link", lwd=2);
```

This is an example of the inside label.

```
par(mar=c(2, 2, 2, 2));
plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
circos(R=300, type="chr", cir="hg18", col=T, print.chr.lab=F, W=4);
circos(R=290, cir="hg18", W=20, mapping=TCGA.PAM50_genefu_hg18, type="label", side="in
    ", col="blue");
circos(R=310, cir="hg18", W=50, mapping=cnv, col.v=4, type="ml3", B=T, col=colors[7],
    cutoff=0, scale=T);
circos(R=150, cir="hg18", W=50, mapping=gene.exp, col.v=4, type="ml3", B=T, col=colors
    [3], cutoff=0, scale=T);
circos(R=90,  cir="hg18", W=50, mapping=pvalue, col.v=4, type="l", B=F, col=colors[1],
     scale=T);
```

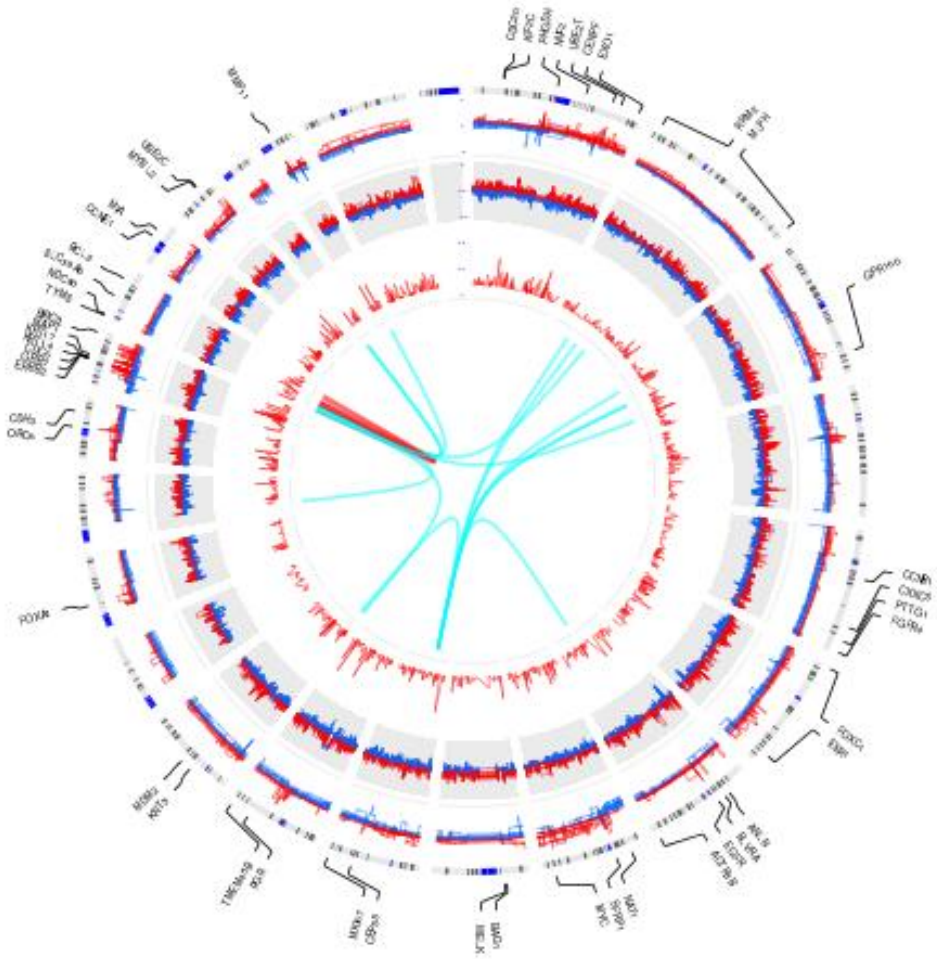

Figure 4

```
circos(R=82, cir="hg18", W=50, mapping=TCGA.BC.fus, type="link", lwd=2);
```

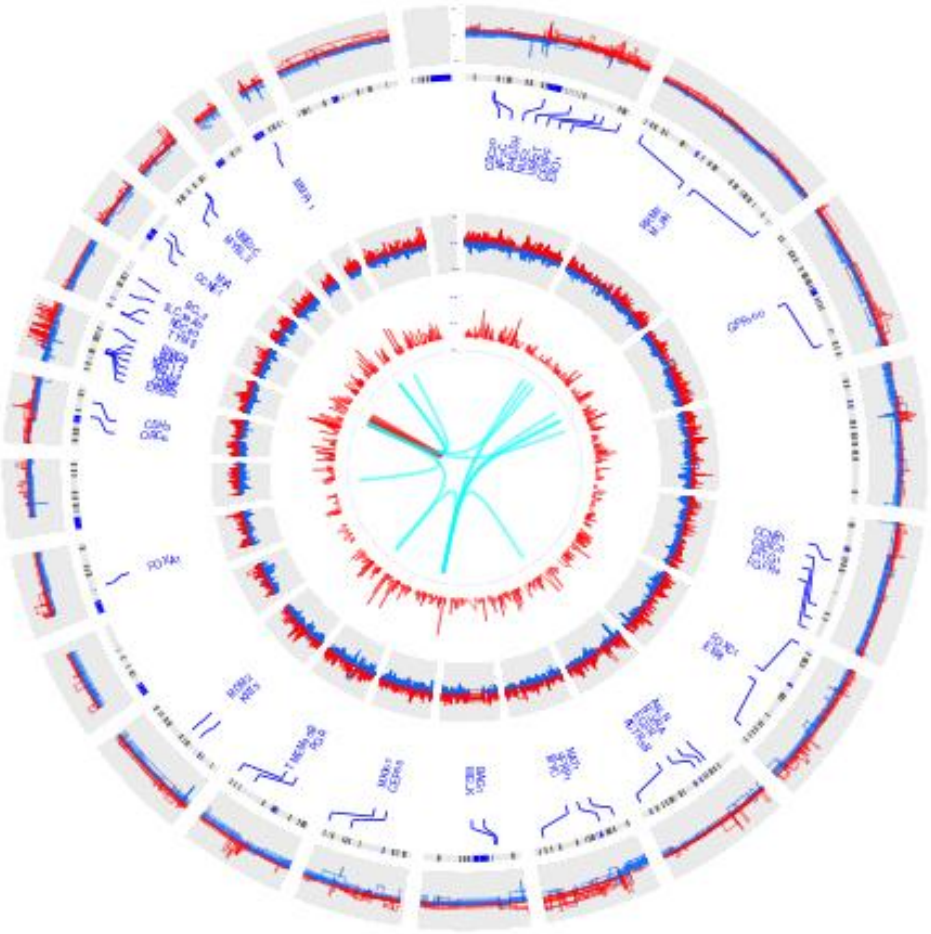

Figure 5

4.3 cluster and heatmap legend

An example of the clustered heatmap with the cluster structure and the color bar.

```
options(stringsAsFactors = FALSE);
library(OmicCircos);

data("TCGA.PAM50_genefu_hg18");
data("TCGA.BC.fus");
data("TCGA.BC.cnv.2k.60");
data("TCGA.BC.gene.exp.2k.60");
data("TCGA.BC.sample60");
data("TCGA.BC_Her2_cnv_exp");

pvalue ← −1 * log10(TCGA.BC_Her2_cnv_exp[,5]);
pvalue ← cbind(TCGA.BC_Her2_cnv_exp[,c(1:3)], pvalue);

Her2.i ← which(TCGA.BC.sample60[,2] == "Her2");
Her2.n ← TCGA.BC.sample60[Her2.i,1];

Her2.j ← which(colnames(TCGA.BC.cnv.2k.60) %in% Her2.n);
cnv    ← TCGA.BC.cnv.2k.60[,c(1:3,Her2.j)];
cnv.m  ← cnv[,c(4:ncol(cnv))];
cnv.m[cnv.m >  2] ← 2;
cnv.m[cnv.m < −2] ← −2;
cnv ← cbind(cnv[,1:3], cnv.m);

Her2.j   ← which(colnames(TCGA.BC.gene.exp.2k.60) %in% Her2.n);
gene.exp ← TCGA.BC.gene.exp.2k.60[,c(1:3,Her2.j)];

colors ← rainbow(10, alpha=0.5);
```

```
par(mar=c(2, 2, 2, 2));

plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");

circos(R=400, cir="hg18", W=4,   type="chr", print.chr.lab=T, scale=T);
circos(R=400, cir="hg18", W=4,   type="chr", print.chr.lab=T, scale=T);
circos(R=300, cir="hg18", W=100, mapping=gene.exp,  col.v=4,  type="heatmap2", cluster
    =T, col.bar=T, lwd=0.01);
circos(R=220, cir="hg18", W=80,  mapping=cnv,   col.v=4,   type="ml3", B=F, lwd=1,
    cutoff=0);
circos(R=140, cir="hg18", W=80,  mapping=pvalue,  col.v=4,    type="l",   B=T, lwd=1,
    col=colors[1]);
circos(R=130, cir="hg18", W=10,  mapping=TCGA.BC.fus, type="link", lwd=2);
```

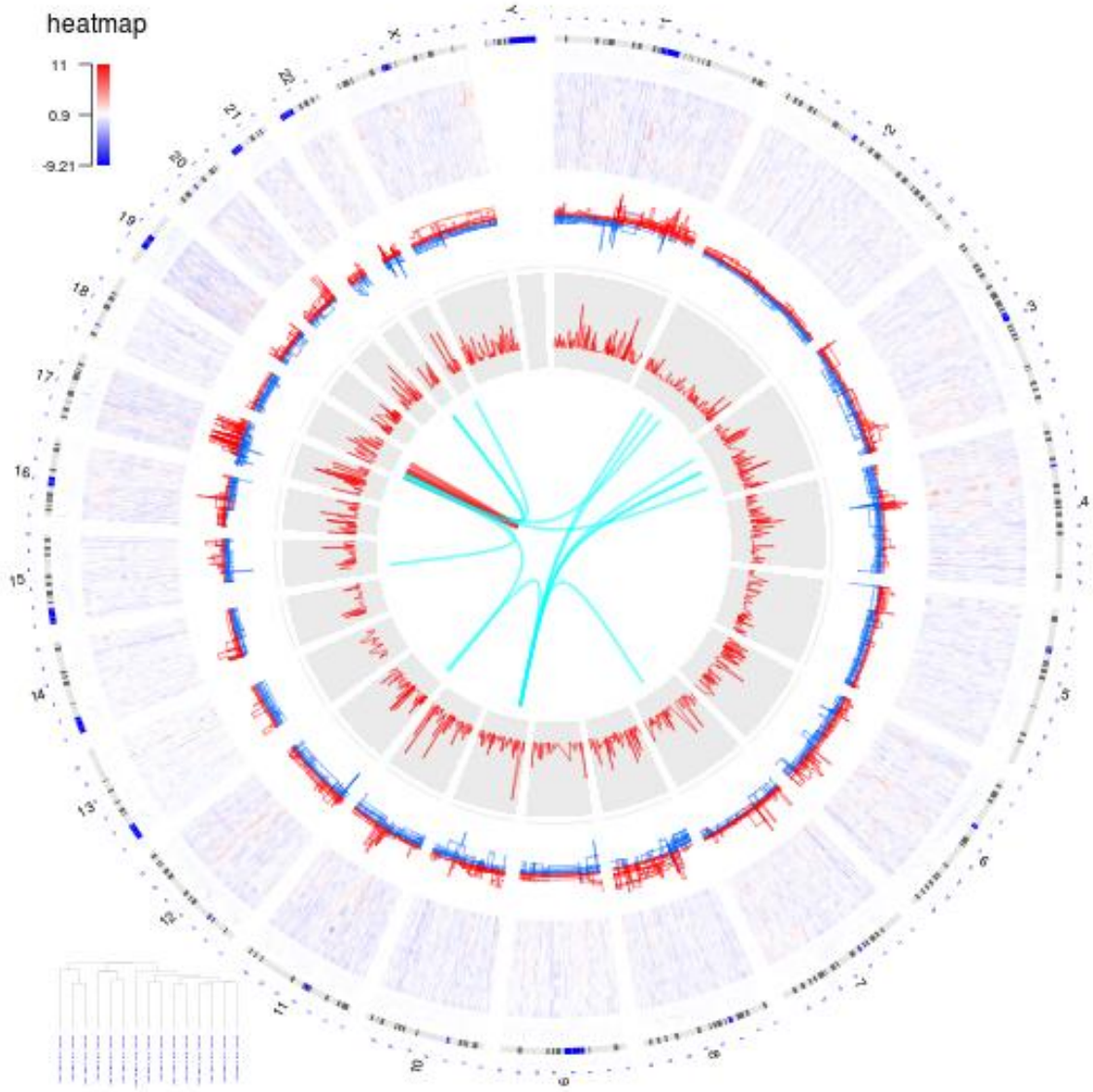

Figure 6

4.4 traditional plotting and OmicCircos

```
options(stringsAsFactors = FALSE);
library(OmicCircos);

data("TCGA.BC.fus");
data("TCGA.BC.cnv.2k.60");
data("TCGA.BC.gene.exp.2k.60");
data("TCGA.BC.sample60");

## gene expression data for PCA
exp.m   ← TCGA.BC.gene.exp.2k.60[,c(4:ncol(TCGA.BC.gene.exp.2k.60))];
cnv     ← TCGA.BC.cnv.2k.60;
type.n  ← unique(TCGA.BC.sample60[,2]);
colors  ← rainbow(length(type.n), alpha=0.5);

## sub-type colors
pca.col ← rep(NA, nrow(TCGA.BC.sample60));
for (i in 1:length(type.n)){
  n   ← type.n[i];
  n.i ← which(TCGA.BC.sample60[,2] == n);
  n.n ← TCGA.BC.sample60[n.i,1];
  g.i ← which(colnames(exp.m) %in% n.n);
  pca.col[g.i] ← colors[i];
}

## run PCA
exp.m   ← na.omit(exp.m);
pca.out ← prcomp(t(exp.m), scale = TRUE);

## subtype cnv
cnv.i ← c();
for (i in 1:length(type.n)){
  n     ← type.n[i];
  n.i   ← which(TCGA.BC.sample60[,2] == n);
  n.n   ← TCGA.BC.sample60[n.i,1];
  cnv.i ← which(colnames(cnv) %in% n.n);
}
```

```
## PCA is plotting.
plot(pca.out$x[,1]*5, pca.out$x[,2]*5, pch=19, col=pca.col, main="",
     cex=2, xlab="PC1", ylab="PC2", ylim=c(-200, 460), xlim=c(-200,460));
legend(200,0, c("Basal","Her2","LumA","LumB"), pch=19, col=colors[c(2,4,1,3)], cex=1,
     title ="Gene Expression (PCA)", box.col="white");

## It is going to plot the circos.
circos(xc=280, yc=280, R=168, cir="hg18", W=4,     type="chr", print.chr.lab=T);
R.v ← 135;
for (i in 1:length(type.n)){
  n     ← type.n[i];
  n.i   ← which(TCGA.BC.sample60[,2] == n);
  n.n   ← TCGA.BC.sample60[n.i,1];
  cnv.i ← which(colnames(cnv) %in% n.n);
  cnv.v ← cnv[,cnv.i];
  cnv.v[cnv.v > 2] ← 2;
```

```
  cnv.v[cnv.v < -2] <- -2;
  cnv.m <- cbind(cnv[,c(1:3)], cnv.v);
  circos(xc=280, yc=280, R=R.v, cir="hg18", W=34, mapping=cnv.m, col.v=4, type="ml3",
       B=F, lwd=0.5, cutoff=0);
  R.v <- R.v - 25;
}

legend(-80,460, c("1 Basal", "2 Her2", "3 LumA", "4 LumB", "(center)"), cex=1,
      title ="CNV (OmicCircos)", box.col="white");
```

It is an example, PCA plotting is at the center of the circos.

```
plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");

legend(680,800, c("Basal","Her2","LumA","LumB"), pch=19, col=colors[c(2,4,1,3)], cex=0
    .5,
      title ="Gene Expression (PCA)", box.col="white");

legend(5,800, c("1 Basal", "2 Her2", "3 LumA", "4 LumB", "(center)"), cex=0.5,
      title ="CNV (OmicCircos)", box.col="white");

circos(xc=400, yc=400, R=390, cir="hg18", W=4, type="chr", print.chr.lab=T, scale=T);
R.v <- 330;
for (i in 1:length(type.n)){
  n     <- type.n[i];
  n.i   <- which(TCGA.BC.sample60[,2] == n);
  n.n   <- TCGA.BC.sample60[n.i,1];
  cnv.i <- which(colnames(cnv) %in% n.n);
  cnv.v <- cnv[,cnv.i];
  cnv.v[cnv.v > 2]  <- 2;
  cnv.v[cnv.v < -2] <- -2;
  cnv.m <- cbind(cnv[,c(1:3)], cnv.v);
  circos(xc=400, yc=400, R=R.v, cir="hg18", W=60, mapping=cnv.m, col.v=4, type="ml3",
       B=F, lwd=1, cutoff=0, scale=T);
  R.v <- R.v - 60;
}

points(pca.out$x[,1]*3.6+400, pca.out$x[,2]*3.6+400, pch=19, col=pca.col, cex=2);
```

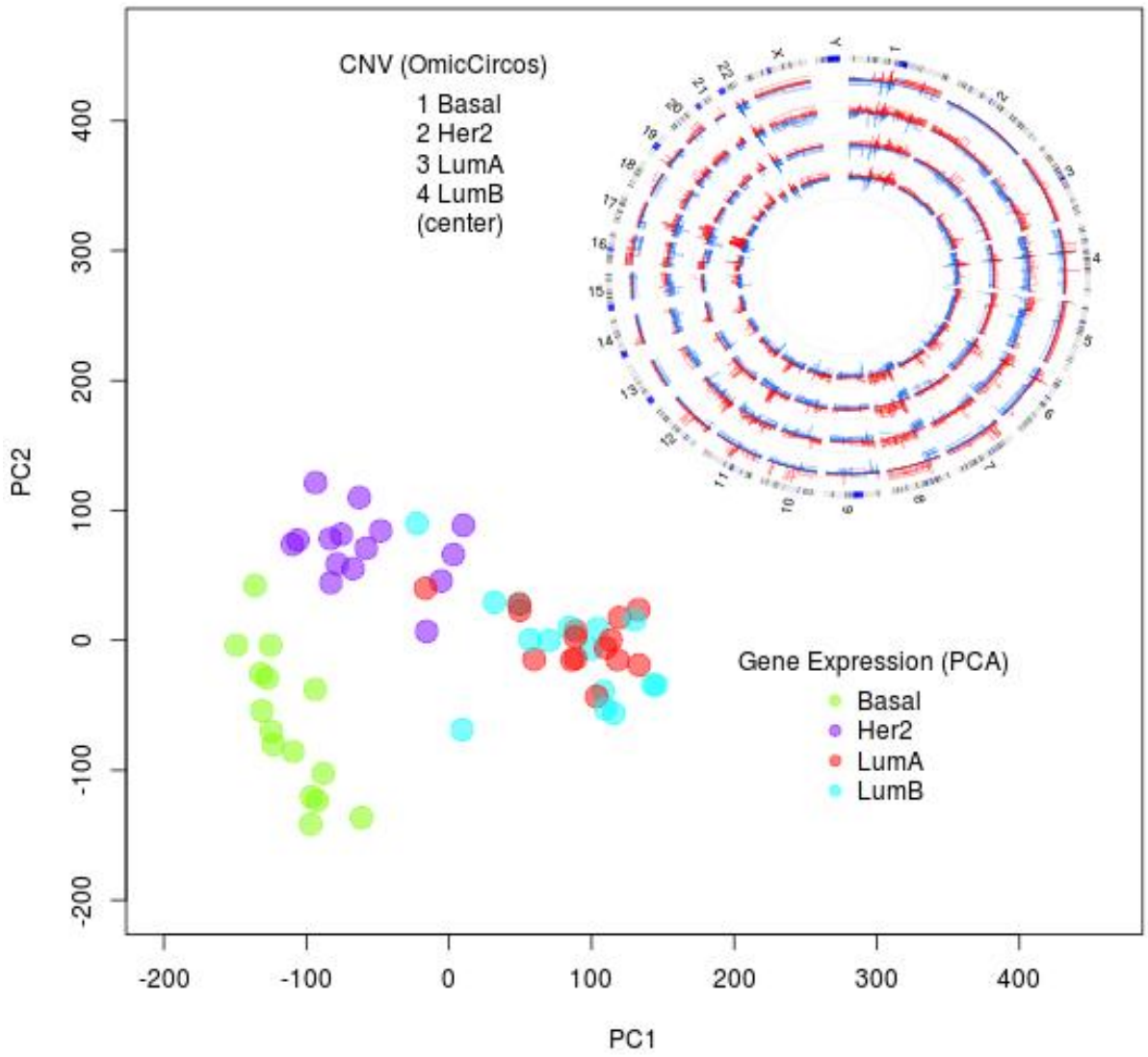
CNV (OmicCircos)
1 Basal
2 Her2
3 LumA
4 LumB
(center)
Gene Expression (PCA)
Basal
Her2
LumA
LumB
PC1
PC2
-200
-100
0
100
200
300
400

Figure 7

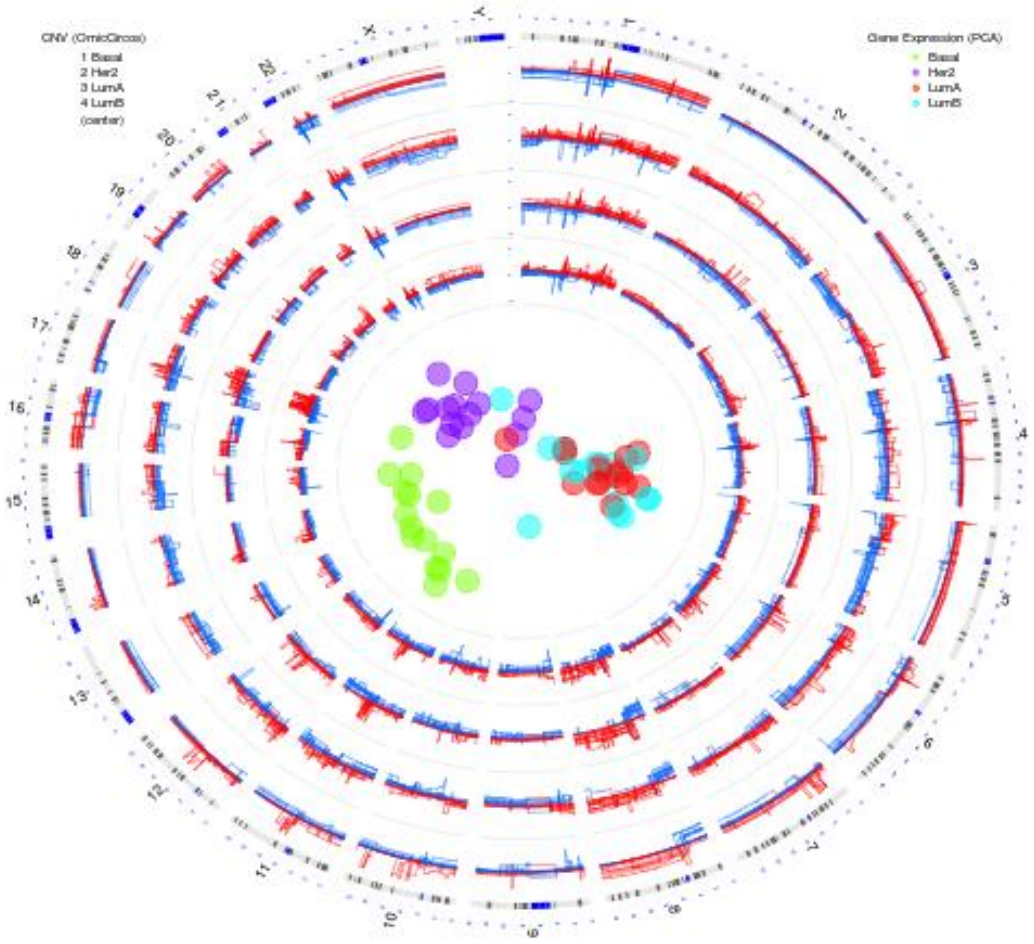
CNV (OmicCircos)
1 Basal
2 Her2
3 LumA
4 LumB
(center)
Gene Expression (PCA)
Basal
Her2
LumA
LumB

Figure 8

4.5 zoom

```
options(stringsAsFactors = FALSE);
library(OmicCircos);

data("TCGA.PAM50_genefu_hg18");
data("TCGA.BC.fus");
data("TCGA.BC.cnv.2k.60");
data("TCGA.BC.gene.exp.2k.60");
data("TCGA.BC.sample60");
data("TCGA.BC_Her2_cnv_exp");
data("TCGA.PAM50_genefu_hg18");

pvalue ← −1 * log10(TCGA.BC_Her2_cnv_exp[,5]);
pvalue ← cbind(TCGA.BC_Her2_cnv_exp[,c(1:3)], pvalue);

Her2.i ← which(TCGA.BC.sample60[,2] == "Her2");
Her2.n ← TCGA.BC.sample60[Her2.i,1];

Her2.j ← which(colnames(TCGA.BC.cnv.2k.60) %in% Her2.n);
cnv    ← TCGA.BC.cnv.2k.60[,c(1:3,Her2.j)];
cnv.m  ← cnv[,c(4:ncol(cnv))];
cnv.m[cnv.m >  2] ← 2;
cnv.m[cnv.m < −2] ← −2;
cnv ← cbind(cnv[,1:3], cnv.m);

gene.exp ← TCGA.BC.gene.exp.2k.60[,c(1:3,Her2.j)];

colors ← rainbow(10, alpha=0.5);
```

```
par(mar=c(2, 2, 2, 2));

plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
# In figure 7, the chromosome 1 to chromosome 22 are going to be plotted from the
    angle 0 (12 Oclock)
# to 180 degree (6 Oclock).
zoom ← c(1, 22, 939245.5, 154143883, 0, 180);
circos(R=400, cir="hg18", W=4,   type="chr", print.chr.lab=T, scale=T, zoom=zoom);
circos(R=300, cir="hg18", W=100, mapping=gene.exp, col.v=4,  type="heatmap2", cluster=
    T, col.bar=T, col.bar.po = "bottomright", lwd=0.01, zoom=zoom);
circos(R=220, cir="hg18", W=80,  mapping=cnv,      col.v=4,  type="ml3", B=F, lwd=1,
    cutoff=0, zoom=zoom);
circos(R=140, cir="hg18", W=80,  mapping=pvalue,   col.v=4,  type="l",   B=T, lwd=1,
    col=colors[1], zoom=zoom);
circos(R=130, cir="hg18", W=10,  mapping=TCGA.BC.fus, type="link", lwd=2, zoom=zoom);

# zoom in links by using the hightlight functions
# highlight
the.col1=rainbow(10, alpha=0.5)[1];

# The highline region is radium from 140 to 400 and from position 282412.5 to
    133770314.5 in chromosome 11.
highlight ← c(140, 400, 11, 282412.5, 11, 133770314.5, the.col1, the.col1);
circos(R=110, cir="hg18", W=40, mapping=highlight, type="hl", lwd=2, zoom=zoom);
the.col2=rainbow(10, alpha=0.5)[6];
```

```
highlight ← c(140, 400, 17, 739525, 17, 78385909, the.col2 , the.col2 );
circos(R=110, cir="hg18", W=40, mapping=highlight , type="hl", lwd=2, zoom=zoom);
## highlight link
highlight.link1 ← c(400, 400, 140, 376.8544 , 384.0021 , 450, 540.5);
circos(cir="hg18", mapping=highlight.link1 , type="highlight.link", col=the.col1 , lwd
    =1);
highlight.link2 ← c(400, 400, 140, 419.1154 , 423.3032 , 543, 627);
circos(cir="hg18", mapping=highlight.link2 , type="highlight.link", col=the.col2 , lwd
    =1);

# The chromosome 11 region is going plotting from 180 (6 Oclock) to 270 degree (9
    Oclock).
zoom ← c(11, 11, 282412.5 , 133770314.5 , 180, 270);
circos(R=400, cir="hg18", W=4,   type="chr", print.chr.lab=T, scale=T, zoom=zoom);
circos(R=300, cir="hg18", W=100, mapping=gene.exp , col.v=4,  type="heatmap2", cluster=
    T, lwd=0.01 , zoom=zoom);
circos(R=220, cir="hg18", W=80,  mapping=cnv ,      col.v=4,  type="ml3", B=F, lwd=1,
    cutoff=0, zoom=zoom);
circos(R=140, cir="hg18", W=80,  mapping=pvalue ,   col.v=4,  type="l",   B=T, lwd=1,
    col=colors[1], zoom=zoom);

# The chromosome 17 region is going plotting from 180 (6 Oclock) to 270 degree (9
    Oclock).

gene.names ← c("ERBB2","CDC6");
PAM50.17   ← which(TCGA.PAM50_genefu_hg18[,3]==gene.names);
TCGA.PAM50 ← TCGA.PAM50_genefu_hg18[PAM50.17,];

# zoom in chromosome 17
zoom ← c(17, 17, 739525, 78385909, 274, 356);
circos(R=400, cir="hg18", W=4,   type="chr", print.chr.lab=T, scale=T, zoom=zoom);
circos(R=300, cir="hg18", W=100, mapping=gene.exp , col.v=4,  type="heatmap2", cluster=
    T, lwd=0.01 , zoom=zoom);
circos(R=220, cir="hg18", W=80,  mapping=cnv ,      col.v=4,  type="ml3", B=F, lwd=1,
    cutoff=0, zoom=zoom);
circos(R=140, cir="hg18", W=80,  mapping=pvalue ,   col.v=4,  type="l",   B=T, lwd=1,
    col=colors[1], zoom=zoom);
circos(R=410, cir="hg18", W=40,  mapping=TCGA.PAM50, type="label", side="out", col="
    blue", zoom=zoom);
```

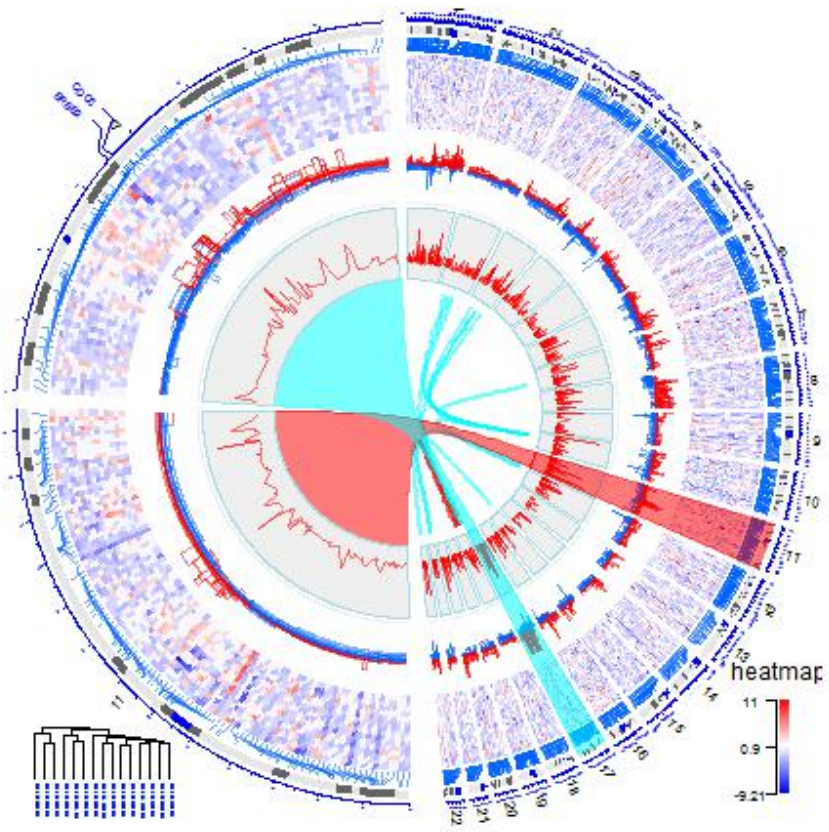

Figure 9