

The OmicCircos usages by examples

Ying Hu and Chunhua Yan

October 14, 2013

Contents

1	Introduction	2
2	Input file formats	3
2.1	segment data	3
2.2	mapping data	4
2.3	linking data	4
2.4	linking polygon data	5
3	The package functions	6
3.1	sim.circos	6
3.2	segAnglePo	7
3.3	circos	8
4	Plotting parameters	9
4.1	basic plotting	9
4.2	lables	15
4.3	cluster and heatmap legend	19
4.4	traditional plotting and OmicCircos	21
4.5	zoom	25

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 number 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

```
1 options(stringsAsFactors = FALSE);
2 library(OmicCircos);
3 # load the hg18 segment data
4 data(UCSC.hg18.chr);
5 # display the first six rows of the data
6 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, 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
##
## Loading required package: IRanges
## Loading required package: XVector
##
##   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.

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

##	chr	po	NAME	TCGA.A1.A0SK.01A	TCGA.A1.A0SO.01A	TCGA.A2.A04W.01A
## 1	1	939246	ISG15	-3.618	-2.286	-0.998
## 2	1	2533140	MMEL1	-2.832	-3.093	1.037
## 3	1	6446321	TNFRSF25	2.559	-0.660	-0.705
## 4	1	7832974	UTS2	1.708	-0.726	-0.102
## 5	1	7912164	TNFRSF9	-0.189	-0.768	1.020
## 6	1	9989253	RBP7	0.817	-0.463	-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.

```
1 options(stringsAsFactors = FALSE);
2 # load the OmicCircos-package
3 library(OmicCircos);
4 # load the gene fusion data sets as the linked data
5 data(TCGA.BC.fus);
6 # display the first six rows and the first six columns of the data
7 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.

```
1 options(stringsAsFactors = FALSE);
2 # load the OmicCircos-package
3 library(OmicCircos);
4 # load the genome comparison data sets as the linked data
5 data(UCSC.hs_cyto_mm);
6 # display the first six rows of the data
7 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 $\text{rnorm}(1) + i$. The i is the ordinal number of the segments. The values are increased by the segment order.

```
1 options(stringsAsFactors = FALSE);
2 # load the OmicCircos-package
3 library(OmicCircos);
4 # set up the initial parameters
5 seg.num ← 10;
6 ind.num ← 20;
7 seg.po ← c(20:50);
8 link.num ← 10;
9 link.pg.num ← 10;
10 # run sim.circos function
11 sim.out ← sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num, link.pg=
    link.pg.num);
12 # display the data set names
13 names(sim.out)
14 # display the segment data
15 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
```

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

```
##   seg.name seg.po  name1  name2  name3
## 1   chr1     1 -1.491  0.159  0.607
## 2   chr1     2  1.052  1.734 -0.218
## 3   chr1     3 -0.413  2.264  0.842
## 4   chr1     4  0.335  1.127  1.646
## 5   chr1     5  1.951 -0.141  0.853
## 6   chr1     6  0.833  1.768 -0.203
```

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

```
##      seg1 po1 name1 seg2 po2 name2 name3
## 1 chr9  21   n1 chr9  16   n1   n1
## 2 chr2  18   n2 chr5  33   n2   n2
## 3 chr5  31   n3 chr1  20   n3   n3
## 4 chr6   5   n4 chr4  34   n4   n4
## 5 chr7  27   n5 chr1  43   n5   n5
## 6 chr3  27   n6 chr7   9   n6   n6
```

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

```
##      seg1 start1 end1 seg2 start2 end2
## 1 chr10    14   13 chr1    40   19
## 2 chr3     12   23 chr8    16    9
## 3 chr9     14    5 chr4    20   13
## 4 chr9     11    8 chr1     0    6
## 5 chr3      6   38 chr6    37   28
## 6 chr5     28   15 chr1    16    7
```

3.2 segAnglePo

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

```
1 options(stringsAsFactors = FALSE);
2 # load the OmicCircos-package
3 library(OmicCircos);
4 # set up the initial parameters
5 seg.num    <- 10;
6 ind.num    <- 20;
7 seg.po     <- c(20:50);
8 link.num   <- 10;
9 link.pg.num <- 10;
10 # run sim.circos function
11 sim.out    <- sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num, link.pg=
    link.pg.num);
12 # To get the segment data
13 seg.f      <- sim.out$seg.frame;
14 # rename the segment names
15 seg.name   <- paste("chr", 1:seg.num, sep="");
16 # run segAnglePo function
17 db         <- segAnglePo(seg.f, seg=seg.name);
18 db[,2]     <- round(as.numeric(db[,2]), 3);
19 db[,3]     <- round(as.numeric(db[,3]), 3);
20 # To display the result
21 db
```

```
##      seg.name angle.start angle.end seg.sum.start seg.sum.end seg.start
## [1,] "chr1"    "270"      "308.857" "0"          "40"         "0"
## [2,] "chr2"    "310.857"  "353.6"    "40"         "84"         "0"
## [3,] "chr3"    "355.6"    "394.457" "84"         "124"        "0"
```

```

## [4,] "chr4" "396.457" "441.143" "124" "170" "0"
## [5,] "chr5" "443.143" "472.286" "170" "200" "0"
## [6,] "chr6" "474.286" "498.571" "200" "225" "0"
## [7,] "chr7" "500.571" "528.743" "225" "254" "0"
## [8,] "chr8" "530.743" "577.371" "254" "302" "0"
## [9,] "chr9" "579.371" "599.771" "302" "323" "0"
## [10,] "chr10" "601.771" "628" "323" "350" "0"
## seg.end
## [1,] "40"
## [2,] "44"
## [3,] "40"
## [4,] "46"
## [5,] "30"
## [6,] "25"
## [7,] "29"
## [8,] "48"
## [9,] "21"
## [10,] "27"

```

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 o'clock (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.

```
1 options(stringsAsFactors = FALSE);
2 library(OmicCircos);
3 options(stringsAsFactors = FALSE);
4 set.seed(1234);
5
6 # initial
7 seg.num ← 10;
8 ind.num ← 20;
9 seg.po ← c(20:50);
10 link.num ← 10;
11 link.pg.num ← 10;
12
13 sim.out ← sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num,
14 link.pg=link.pg.num);
15
16 seg.f ← sim.out$seg.frame;
17 seg.v ← sim.out$seg.mapping;
18 link.v ← sim.out$seg.link
19 link.pg.v ← sim.out$seg.link.pg
20 seg.num ← length(unique(seg.f[,1]));
21
22 # name segment (option)
23 seg.name ← paste("chr", 1:seg.num, sep="");
24 db ← segAnglePo(seg.f, seg=seg.name);
25 # set transparent colors
26 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.

```
1 par(mar=c(2, 2, 2, 2));
2 #
3 plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
4 #
5 circos(R=400, cir=db, type="chr", col=colors, print.chr.lab=T, W=4, scale=T);
6 circos(R=360, cir=db, W=40, mapping=seg.v, col.v=3, type="l", B=T, col=colors[1],
7 lwd=0.1, scale=T);
8 circos(R=320, cir=db, W=40, mapping=seg.v, col.v=3, type="ls", B=F, col=colors[3],
9 lwd=0.1, scale=T);
10 circos(R=280, cir=db, W=40, mapping=seg.v, col.v=3, type="lh", B=T, col=colors[7],
11 lwd=0.1, scale=T);
12 circos(R=240, cir=db, W=40, mapping=seg.v, col.v=19, type="ml", B=F, col=colors, lwd
13 =0.1, scale=T);
14 circos(R=200, cir=db, W=40, mapping=seg.v, col.v=19, type="ml2", B=T, col=colors, lwd
15 =0.1);
16 circos(R=160, cir=db, W=40, mapping=seg.v, col.v=19, type="ml3", B=F, cutoff=5, lwd=0
17 .1);
18 circos(R=150, cir=db, W=40, mapping=link.v, type="link", lwd=2, col=colors);
19 circos(R=150, cir=db, W=40, mapping=link.pg.v, type="link.pg", lwd=2, col=colors);
```

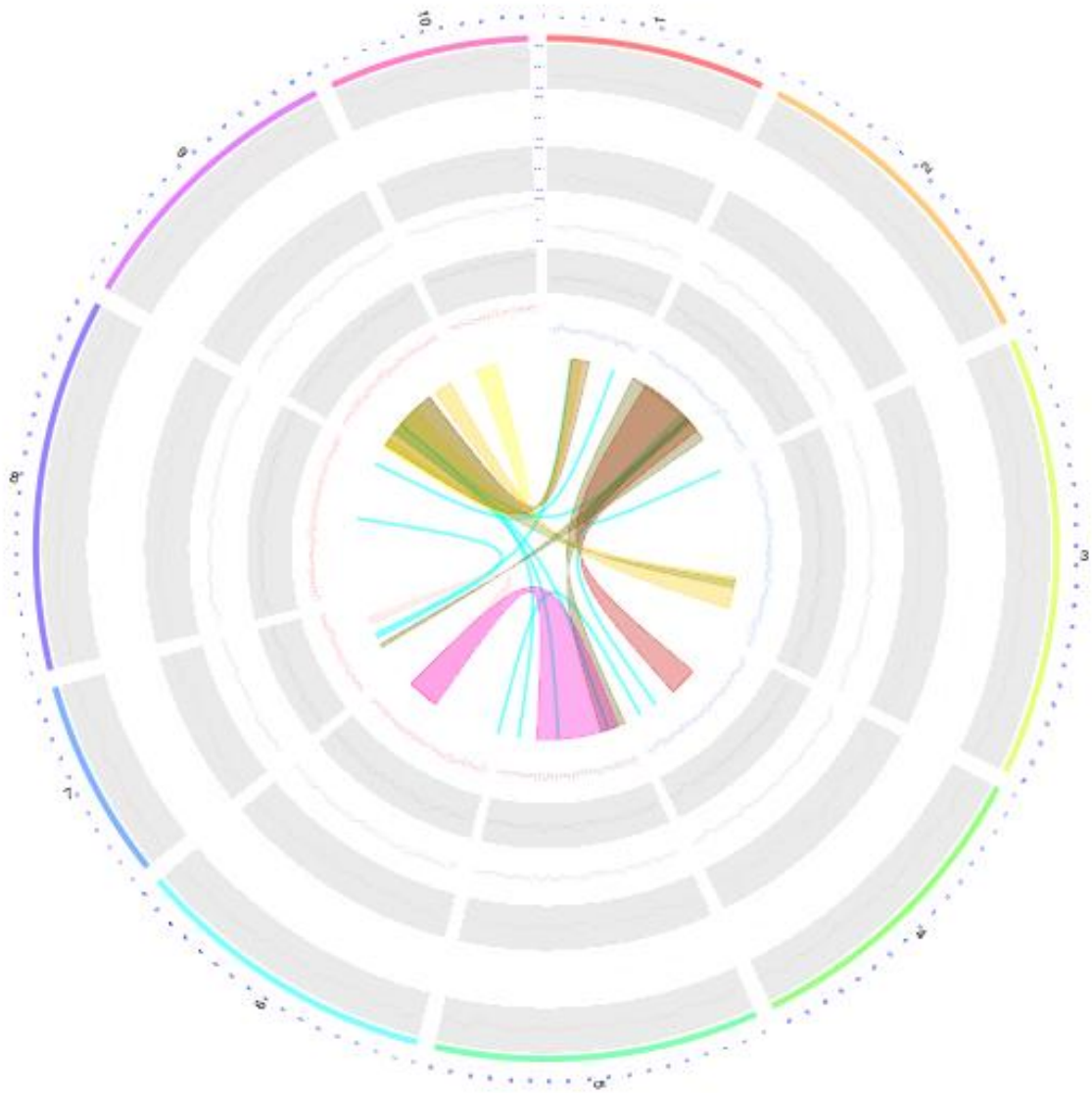


Figure 1

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.

```

1 options(stringsAsFactors = FALSE);
2 library(OmicCircos);
3 set.seed(1234);
4
5 ## initial
6 seg.num    <- 10;
7 ind.num    <- 20;
8 seg.po     <- c(20:50);
9 link.num   <- 10;
10 link.pg.num <- 10;
11
12 sim.out <- sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num,
13   link.pg=link.pg.num);
14
15 seg.f     <- sim.out$seg.frame;
16 seg.v     <- sim.out$seg.mapping;
17 link.v    <- sim.out$seg.link
18 link.pg.v <- sim.out$seg.link.pg
19 seg.num   <- length(unique(seg.f[,1]));
20
21 ## select segments
22 seg.name <- paste("chr", 1:seg.num, sep="");
23 db       <- segAnglePo(seg.f, seg=seg.name);
24
25 colors   <- rainbow(seg.num, alpha=0.5);

```

```

1 par(mar=c(2, 2, 2, 2));
2
3 plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
4 circos(R=400, type="chr", cir=db, col=colors, print.chr.lab=T, W=4, scale=T);
5 circos(R=360, cir=db, W=40, mapping=seg.v, col.v=8, type="box", B=T, col=colors[1],
6   lwd=0.1, scale=T);
7 circos(R=320, cir=db, W=40, mapping=seg.v, col.v=8, type="hist", B=T, col=colors[3],
8   lwd=0.1, scale=T);
9 circos(R=280, cir=db, W=40, mapping=seg.v, col.v=8, type="ms", B=T, col=colors[7],
10  lwd=0.1, scale=T);
11 circos(R=240, cir=db, W=40, mapping=seg.v, col.v=3, type="h", B=F, col=colors[2],
12  lwd=0.1);
13 circos(R=200, cir=db, W=40, mapping=seg.v, col.v=3, type="s", B=T, col=colors, lwd=0.1
14  );
15 circos(R=160, cir=db, W=40, mapping=seg.v, col.v=3, type="b", B=F, col=colors, lwd=0.1
16  );
17 circos(R=150, cir=db, W=40, mapping=link.v, type="link", lwd=2, col=colors);
18 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 radius=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 and 5 are the heatmap. Track 6 is the circle plot with the center=median and radius=standard deviation. Track 7 is the 95%

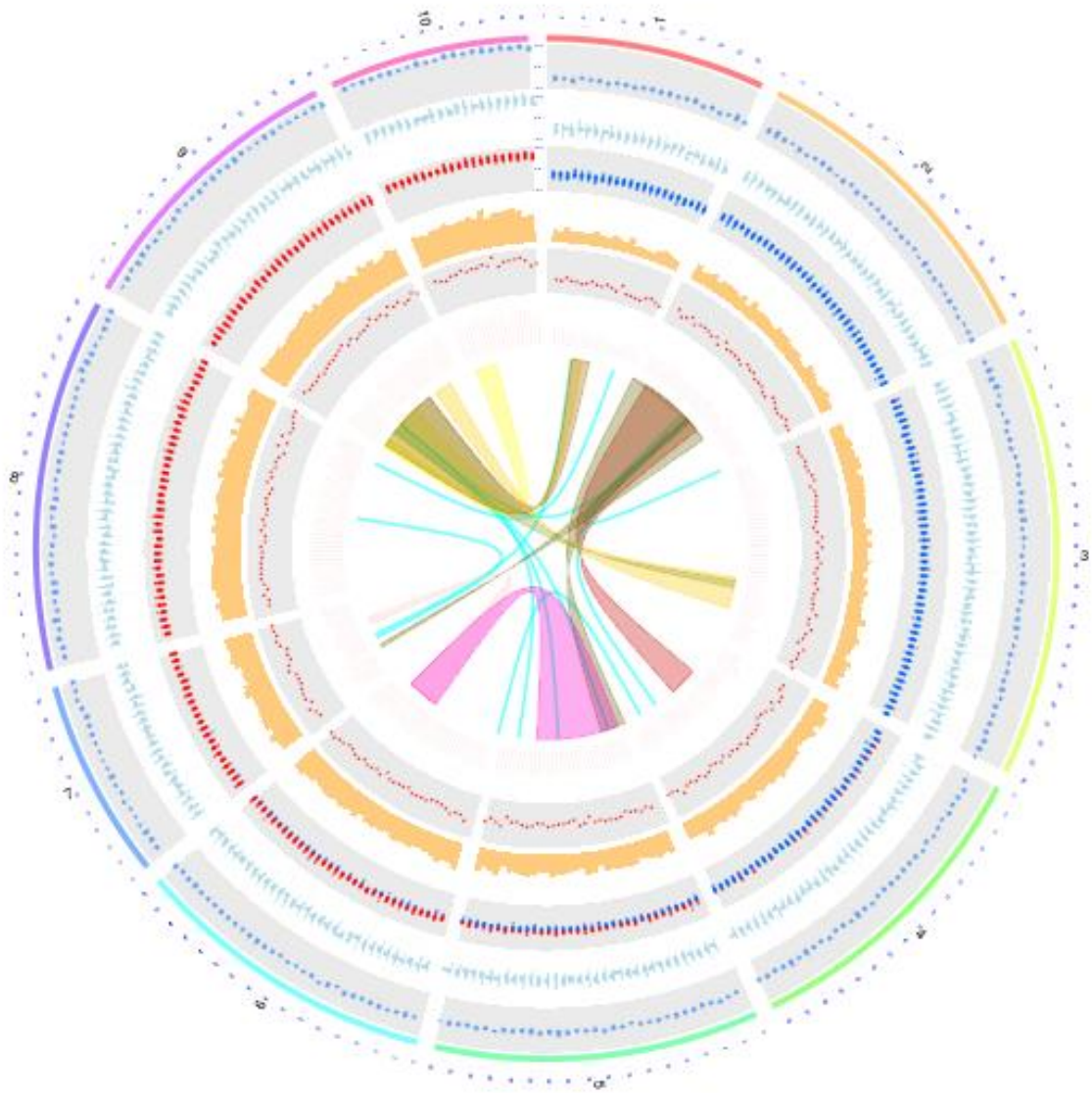


Figure 2

confidence interval of the samples.

```
1 options(stringsAsFactors = FALSE);
2 library(OmicCircos);
3 set.seed(1234);
4
5 ## initial
6 seg.num ← 10;
7 ind.num ← 20;
8 seg.po ← c(20:50);
9 link.num ← 10;
10 link.pg.num ← 10;
11
12 sim.out ← sim.circos(seg=seg.num, po=seg.po, ind=ind.num, link=link.num,
13   link.pg=link.pg.num);
14
15 seg.f ← sim.out$seg.frame;
16 seg.v ← sim.out$seg.mapping;
17 link.v ← sim.out$seg.link
18 link.pg.v ← sim.out$seg.link.pg
19 seg.num ← length(unique(seg.f[,1]));
20
21 ##
22 seg.name ← paste("chr", 1:seg.num, sep="");
23 db ← segAnglePo(seg.f, seg=seg.name);
24
25 colors ← rainbow(seg.num, alpha=0.5);

1 par(mar=c(2, 2, 2, 2));
2 plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
3
4 circos(R=400, type="chr", cir=db, col=colors, print.chr.lab=T, W=4, scale=T);
5 circos(R=360, cir=db, W=40, mapping=seg.v, col.v=8, type="quant90", B=F, col=colors,
6   lwd=0.1, scale=T);
7 circos(R=320, cir=db, W=40, mapping=seg.v, col.v=3, type="sv", B=T, col=colors[7], lwd
8   =0.1, scale=T);
9 circos(R=280, cir=db, W=40, mapping=seg.v, col.v=3, type="ss", B=F, col=colors[3], lwd
10  =0.1, scale=T);
11 circos(R=240, cir=db, W=40, mapping=seg.v, col.v=8, type="heatmap", lwd=3);
12 circos(R=200, cir=db, W=40, mapping=seg.v, col.v=3, type="s.sd", B=F, col=colors[4],
13  lwd=0.1);
14 circos(R=160, cir=db, W=40, mapping=seg.v, col.v=3, type="ci95", B=T, col=colors[4],
15  lwd=0.1);
16 circos(R=150, cir=db, W=40, mapping=link.v, type="link", lwd=2, col=colors);
17 circos(R=150, cir=db, W=40, mapping=link.pg.v, type="link.pg", lwd=2, col=colors);
18
19 the.col1=rainbow(10, alpha=0.3)[3];
20 highlight ← c(160, 410, 6, 2, 6, 10, the.col1, the.col1);
21 circos(R=110, cir=db, W=40, mapping=highlight, type="hl", lwd=2);
22
23 the.col1=rainbow(10, alpha=0.01)[3];
24 the.col2=rainbow(10, alpha=0.8)[1];
25 highlight ← c(160, 410, 6, 12, 7, 10, the.col1, the.col2);
26 circos(R=110, cir=db, W=40, mapping=highlight, type="hl", lwd=2);
```

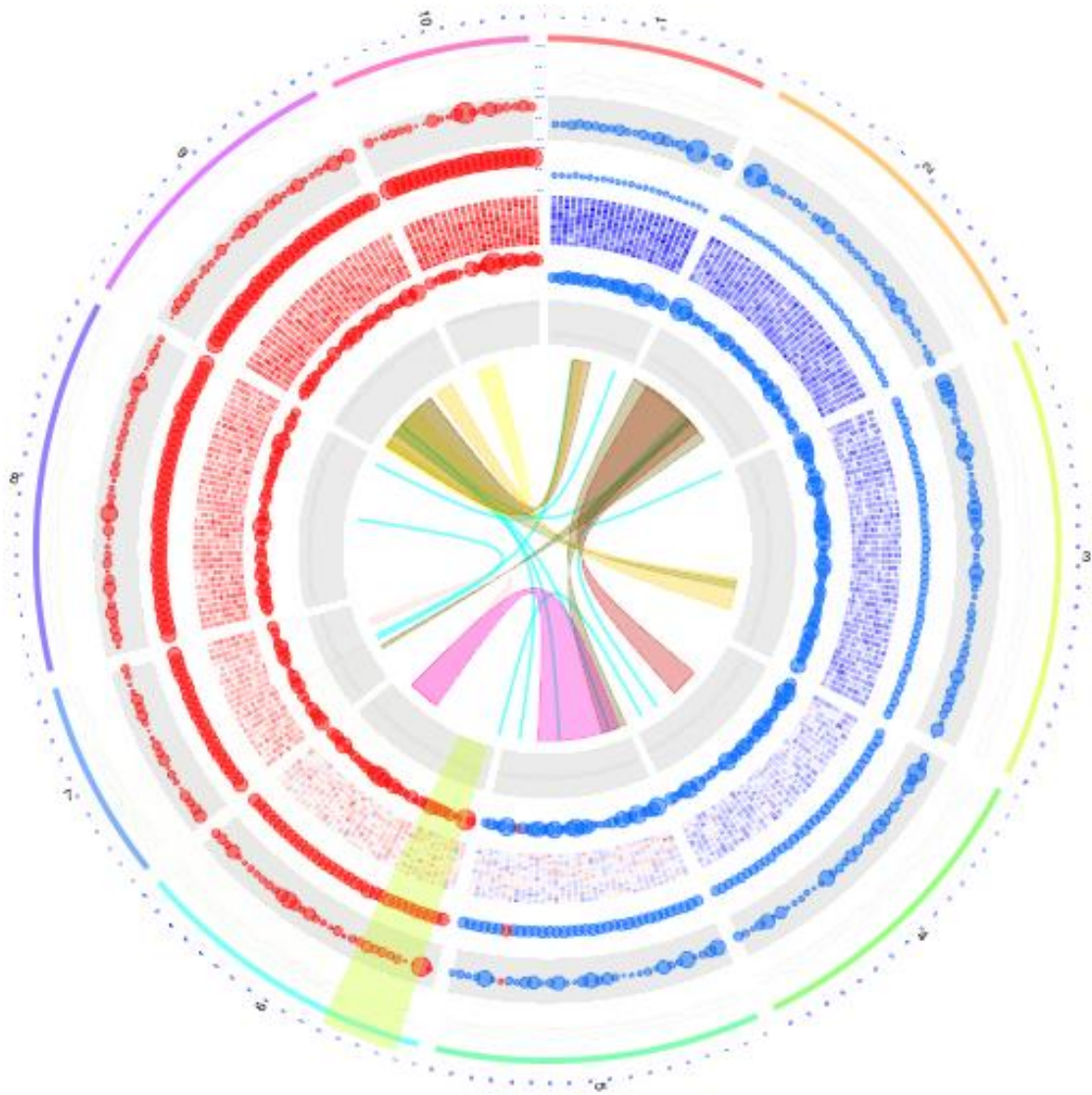


Figure 3

4.2 lables

An example of adding outside label.

```
1 options(stringsAsFactors = FALSE);
2 library(OmicCircos);
3
4 data("TCGA.PAM50_genefu_hg18");
5 data("TCGA.BC.fus");
6 data("TCGA.BC.cnv.2k.60");
7 data("TCGA.BC.gene.exp.2k.60");
8 data("TCGA.BC.sample60");
9 data("TCGA.BC_Her2_cnv_exp");
10
11 pvalue <- -1 * log10(TCGA.BC_Her2_cnv_exp[,5]);
12 pvalue <- cbind(TCGA.BC_Her2_cnv_exp[,c(1:3)], pvalue);
13
14 Her2.i <- which(TCGA.BC.sample60[,2] == "Her2");
15 Her2.n <- TCGA.BC.sample60[Her2.i,1];
16
17 Her2.j <- which(colnames(TCGA.BC.cnv.2k.60) %in% Her2.n);
18 cnv <- TCGA.BC.cnv.2k.60[,c(1:3, Her2.j)];
19 cnv.m <- cnv[,c(4:ncol(cnv))];
20 cnv.m[cnv.m > 2] <- 2;
21 cnv.m[cnv.m < -2] <- -2;
22 cnv <- cbind(cnv[,1:3], cnv.m);
23
24 Her2.j <- which(colnames(TCGA.BC.gene.exp.2k.60) %in% Her2.n);
25 gene.exp <- TCGA.BC.gene.exp.2k.60[,c(1:3, Her2.j)];
26 colors <- rainbow(10, alpha=0.5);
```

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

This is an example of the inside label.

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

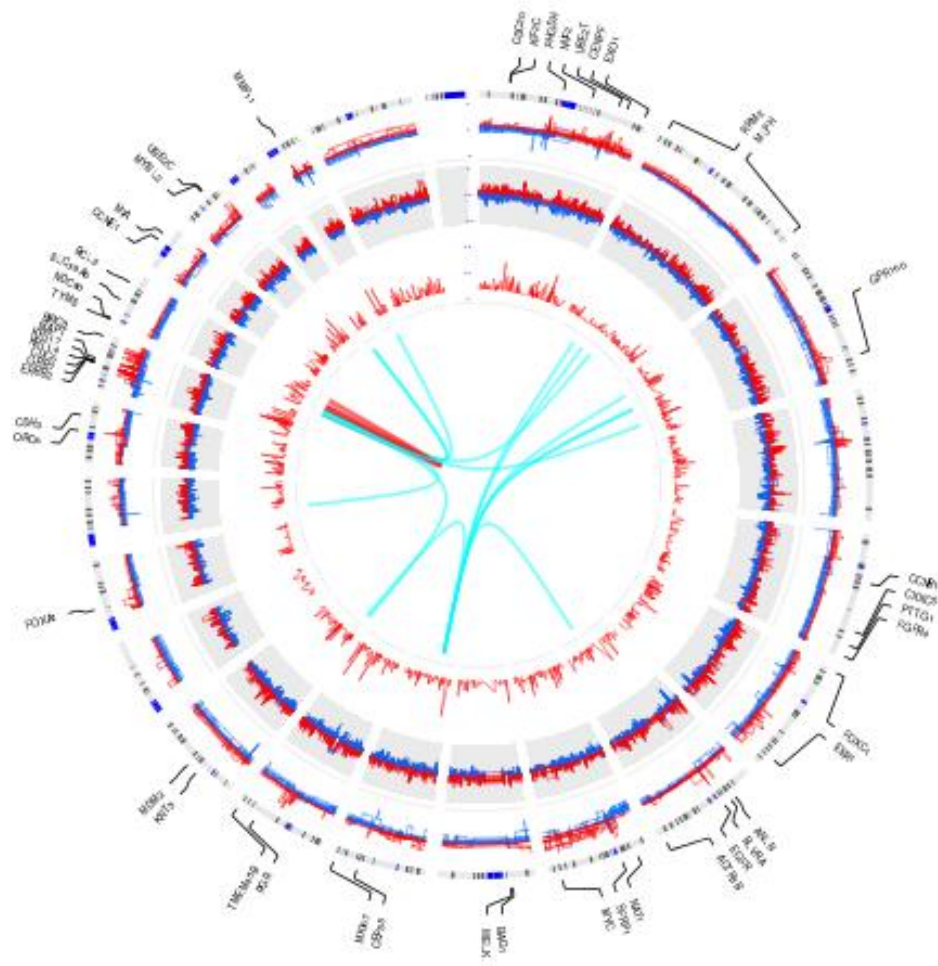


Figure 4


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


4.3 cluster and heatmap legend

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

```
1 options(stringsAsFactors = FALSE);
2 library(OmicCircos);
3
4 data("TCGA.PAM50_genefu_hg18");
5 data("TCGA.BC.fus");
6 data("TCGA.BC.cnv.2k.60");
7 data("TCGA.BC.gene.exp.2k.60");
8 data("TCGA.BC.sample60");
9 data("TCGA.BC_Her2_cnv_exp");
10
11 pvalue <- -1 * log10(TCGA.BC_Her2_cnv_exp[,5]);
12 pvalue <- cbind(TCGA.BC_Her2_cnv_exp[,c(1:3)], pvalue);
13
14 Her2.i <- which(TCGA.BC.sample60[,2] == "Her2");
15 Her2.n <- TCGA.BC.sample60[Her2.i,1];
16
17 Her2.j <- which(colnames(TCGA.BC.cnv.2k.60) %in% Her2.n);
18 cnv <- TCGA.BC.cnv.2k.60[,c(1:3, Her2.j)];
19 cnv.m <- cnv[,c(4:ncol(cnv))];
20 cnv.m[cnv.m > 2] <- 2;
21 cnv.m[cnv.m < -2] <- -2;
22 cnv <- cbind(cnv[,1:3], cnv.m);
23
24 Her2.j <- which(colnames(TCGA.BC.gene.exp.2k.60) %in% Her2.n);
25 gene.exp <- TCGA.BC.gene.exp.2k.60[,c(1:3, Her2.j)];
26
27 colors <- rainbow(10, alpha=0.5);

1 par(mar=c(2, 2, 2, 2));
2
3 plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
4
5 circos(R=400, cir="hg18", W=4, type="chr", print.chr.lab=T, scale=T);
6 circos(R=400, cir="hg18", W=4, type="chr", print.chr.lab=T, scale=T);
7 circos(R=300, cir="hg18", W=100, mapping=gene.exp, col.v=4, type="heatmap2", cluster
  =T, col.bar=T, lwd=0.01);
8 circos(R=220, cir="hg18", W=80, mapping=cnv, col.v=4, type="ml3", B=F, lwd=1,
  cutoff=0);
9 circos(R=140, cir="hg18", W=80, mapping=pvalue, col.v=4, type="l", B=T, lwd=1,
  col=colors[1]);
10 circos(R=130, cir="hg18", W=10, mapping=TCGA.BC.fus, type="link", lwd=2);
```

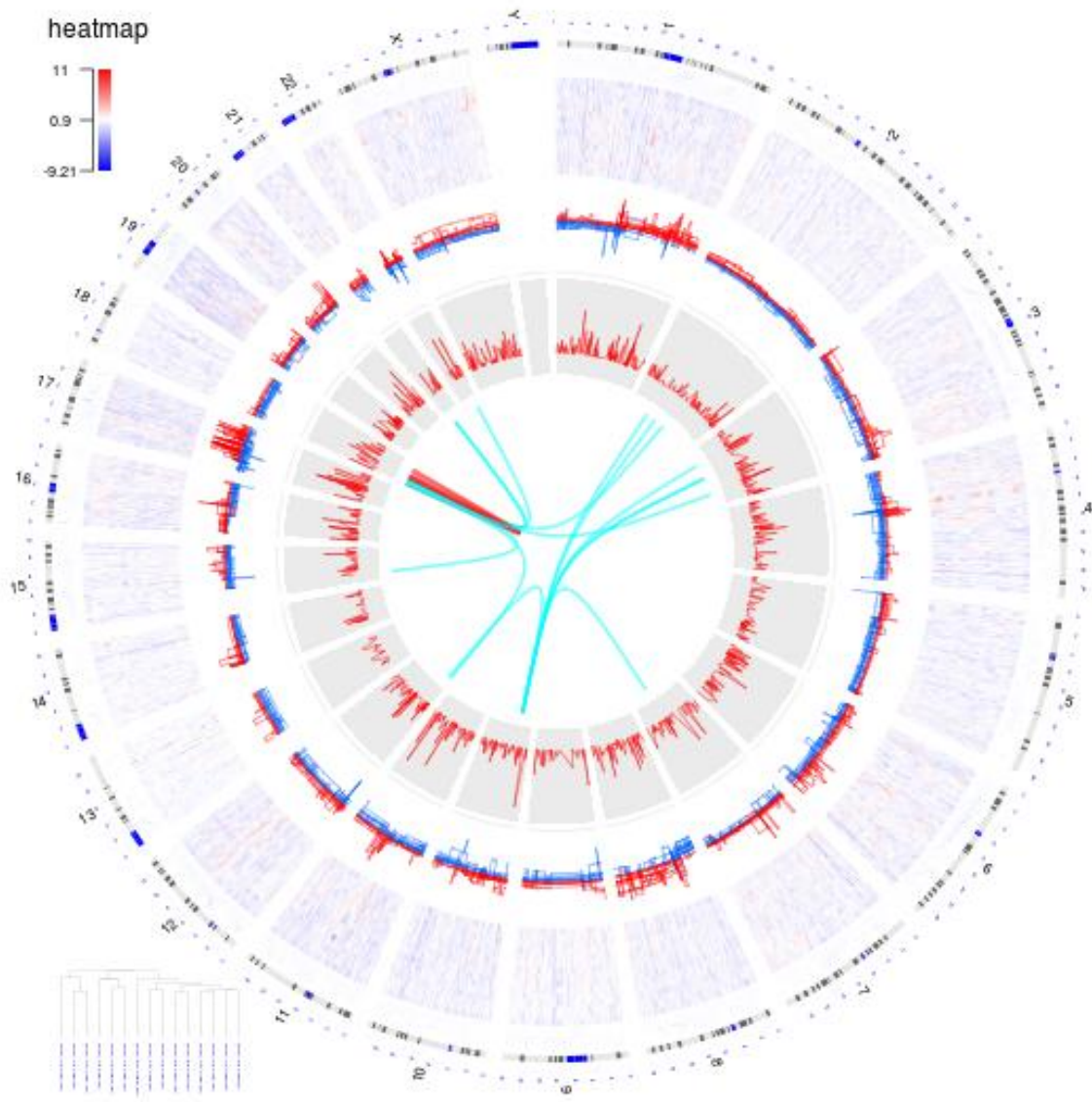


Figure 6

4.4 traditional plotting and OmicCircos

```
1 options(stringsAsFactors = FALSE);
2 library(OmicCircos);
3
4 data("TCGA.BC.fus");
5 data("TCGA.BC.cnv.2k.60");
6 data("TCGA.BC.gene.exp.2k.60");
7 data("TCGA.BC.sample60");
8
9 ## gene expression data for PCA
10 exp.m ← TCGA.BC.gene.exp.2k.60[,c(4:ncol(TCGA.BC.gene.exp.2k.60))];
11 cnv ← TCGA.BC.cnv.2k.60;
12 type.n ← unique(TCGA.BC.sample60[,2]);
13 colors ← rainbow(length(type.n), alpha=0.5);
14
15 ## sub-type colors
16 pca.col ← rep(NA, nrow(TCGA.BC.sample60));
17 for (i in 1:length(type.n)){
18   n ← type.n[i];
19   n.i ← which(TCGA.BC.sample60[,2] == n);
20   n.n ← TCGA.BC.sample60[n.i,1];
21   g.i ← which(colnames(exp.m) %in% n.n);
22   pca.col[g.i] ← colors[i];
23 }
24
25 ## run PCA
26 exp.m ← na.omit(exp.m);
27 pca.out ← prcomp(t(exp.m), scale = TRUE);
28
29 ## subtype cnv
30 cnv.i ← c();
31 for (i in 1:length(type.n)){
32   n ← type.n[i];
33   n.i ← which(TCGA.BC.sample60[,2] == n);
34   n.n ← TCGA.BC.sample60[n.i,1];
35   cnv.i ← which(colnames(cnv) %in% n.n);
36 }

1 ## PCA is plotting.
2 plot(pca.out$x[,1]*5, pca.out$x[,2]*5, pch=19, col=pca.col, main="",
3       cex=2, xlab="PC1", ylab="PC2", ylim=c(-200, 460), xlim=c(-200,460));
4 legend(200,0, c("Basal","Her2","LumA","LumB"), pch=19, col=colors[c(2,4,1,3)], cex=1,
5        title = "Gene Expression (PCA)", box.col="white");
6
7 ## It is going to plot the circos.
8 circos(xc=280, yc=280, R=168, cir="hg18", W=4, type="chr", print.chr.lab=T);
9 R.v ← 135;
10 for (i in 1:length(type.n)){
11   n ← type.n[i];
12   n.i ← which(TCGA.BC.sample60[,2] == n);
13   n.n ← TCGA.BC.sample60[n.i,1];
14   cnv.i ← which(colnames(cnv) %in% n.n);
15   cnv.v ← cnv[,cnv.i];
16   cnv.v[cnv.v > 2] ← 2;
```

```

17   cnv.v[cnv.v < -2] ← -2;
18   cnv.m ← cbind(cnv[,c(1:3)], cnv.v);
19   circos(xc=280, yc=280, R=R.v, cir="hg18", W=34, mapping=cnv.m, col.v=4, type="ml3",
20         B=F, lwd=0.5, cutoff=0);
21   R.v ← R.v - 25;
22 }
23 legend(-80,460, c("1 Basal", "2 Her2", "3 LumA", "4 LumB", "(center)"), cex=1,
24       title ="CNV (OmicCircos)", box.col="white");

```

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

```

1 plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
2
3 legend(680,800, c("Basal","Her2","LumA","LumB"), pch=19, col=colors[c(2,4,1,3)], cex=0
4     .5,
5     title ="Gene Expression (PCA)", box.col="white");
6 legend(5,800, c("1 Basal", "2 Her2", "3 LumA", "4 LumB", "(center)"), cex=0.5,
7     title ="CNV (OmicCircos)", box.col="white");
8
9 circos(xc=400, yc=400, R=390, cir="hg18", W=4, type="chr", print.chr.lab=T, scale=T);
10 R.v ← 330;
11 for (i in 1:length(type.n)){
12   n ← type.n[i];
13   n.i ← which(TCGA.BC.sample60[,2] == n);
14   n.n ← TCGA.BC.sample60[n.i,1];
15   cnv.i ← which(colnames(cnv) %in% n.n);
16   cnv.v ← cnv[,cnv.i];
17   cnv.v[cnv.v > 2] ← 2;
18   cnv.v[cnv.v < -2] ← -2;
19   cnv.m ← cbind(cnv[,c(1:3)], cnv.v);
20   circos(xc=400, yc=400, R=R.v, cir="hg18", W=60, mapping=cnv.m, col.v=4, type="ml3",
21         B=F, lwd=1, cutoff=0, scale=T);
22   R.v ← R.v - 60;
23 }
24 points(pca.out$x[,1]*3.6+400, pca.out$x[,2]*3.6+400, pch=19, col=pca.col, cex=2);

```

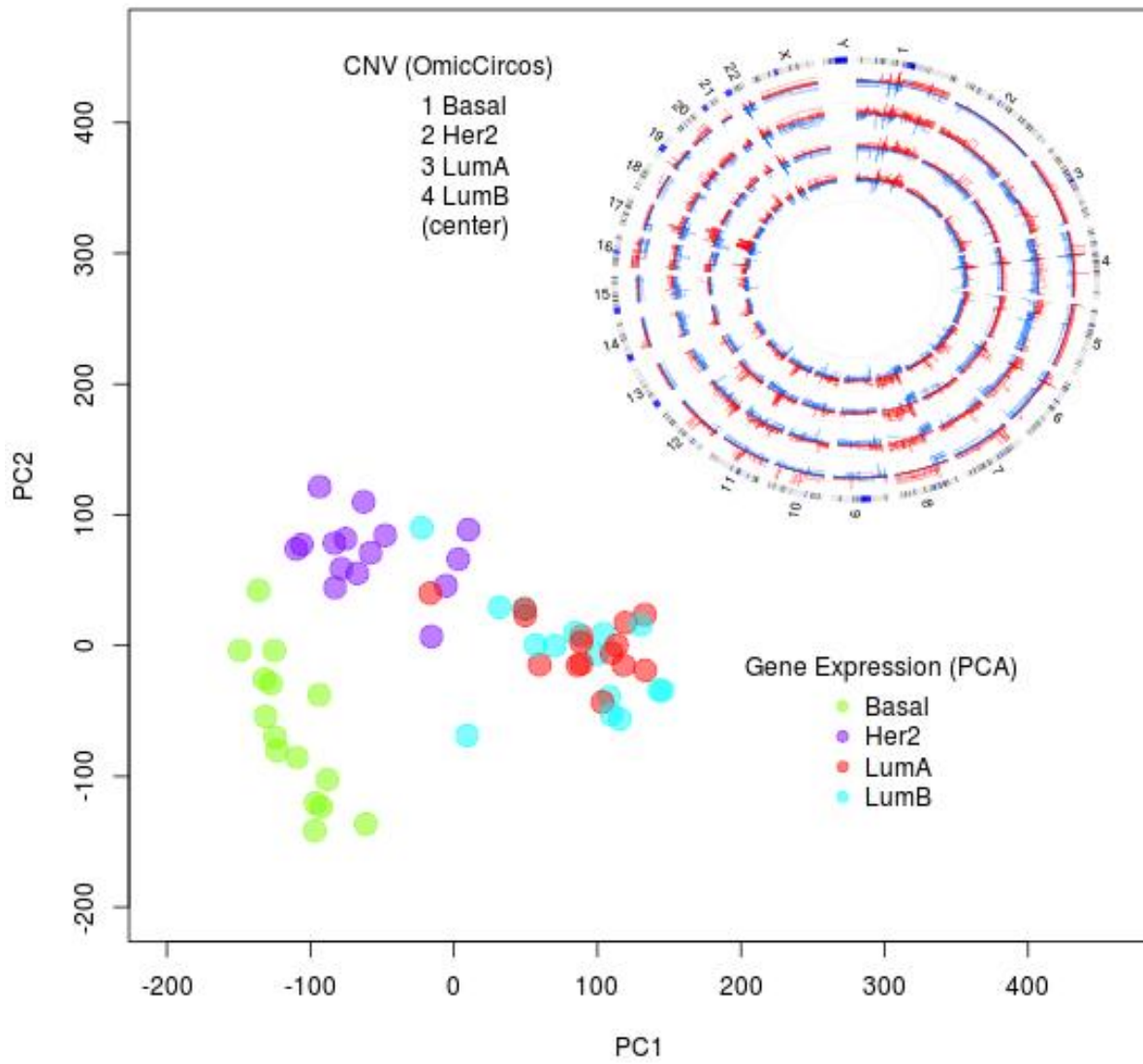


Figure 7

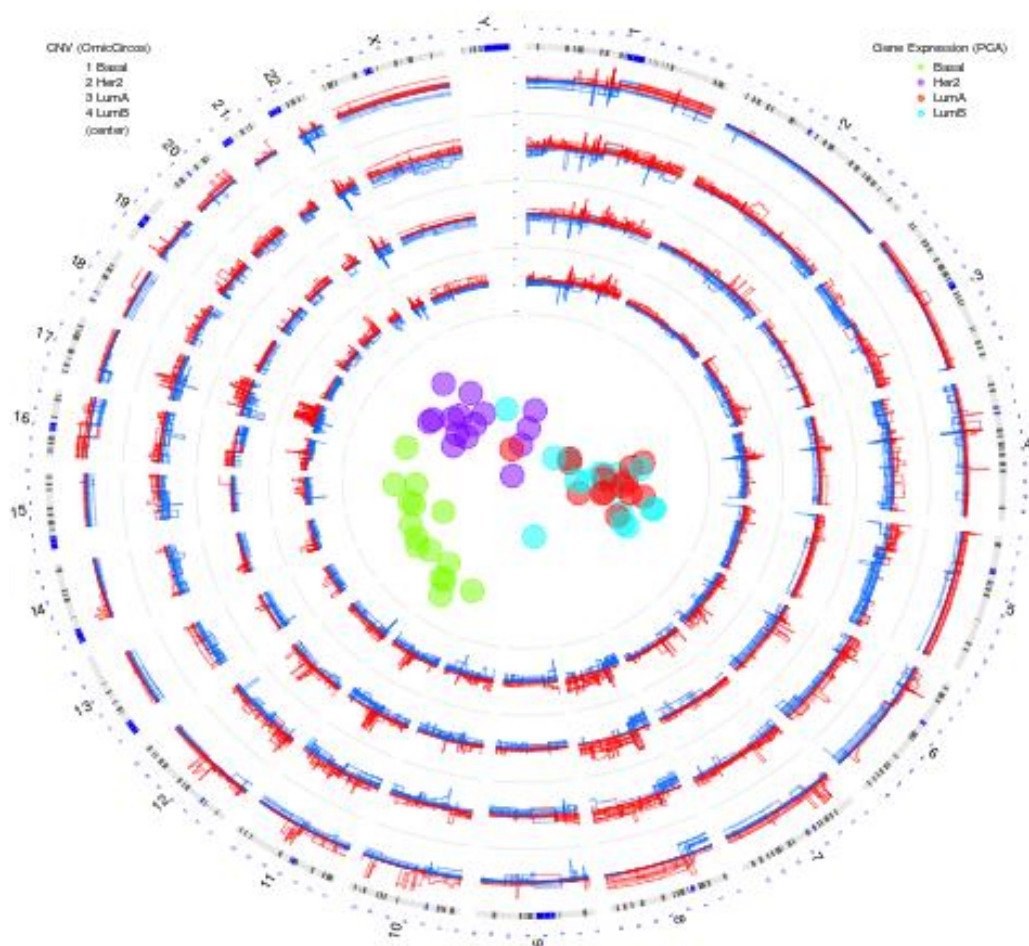


Figure 8

4.5 zoom

```
1 options(stringsAsFactors = FALSE);
2 library(OmicCircos);
3
4 data("TCGA.PAM50_genefu_hg18");
5 data("TCGA.BC.fus");
6 data("TCGA.BC.cnv.2k.60");
7 data("TCGA.BC.gene.exp.2k.60");
8 data("TCGA.BC.sample60");
9 data("TCGA.BC_Her2_cnv_exp");
10 data("TCGA.PAM50_genefu_hg18");
11
12 pvalue <- -1 * log10(TCGA.BC_Her2_cnv_exp[,5]);
13 pvalue <- cbind(TCGA.BC_Her2_cnv_exp[,c(1:3)], pvalue);
14
15 Her2.i <- which(TCGA.BC.sample60[,2] == "Her2");
16 Her2.n <- TCGA.BC.sample60[Her2.i,1];
17
18 Her2.j <- which(colnames(TCGA.BC.cnv.2k.60) %in% Her2.n);
19 cnv <- TCGA.BC.cnv.2k.60[,c(1:3,Her2.j)];
20 cnv.m <- cnv[,c(4:ncol(cnv))];
21 cnv.m[cnv.m > 2] <- 2;
22 cnv.m[cnv.m < -2] <- -2;
23 cnv <- cbind(cnv[,1:3], cnv.m);
24
25 gene.exp <- TCGA.BC.gene.exp.2k.60[,c(1:3,Her2.j)];
26
27 colors <- rainbow(10, alpha=0.5);

1 par(mar=c(2, 2, 2, 2));
2
3 plot(c(1,800), c(1,800), type="n", axes=F, xlab="", ylab="", main="");
4 # In figure 7, the chromosome 1 to chromosome 22 are going to be plotted from the
5 # angle 0 (12 O'clock)
6 # to 180 degree (6 O'clock).
7 zoom <- c(1, 22, 939245.5, 154143883, 0, 180);
8 circos(R=400, cir="hg18", W=4, type="chr", print.chr.lab=T, scale=T, zoom=zoom);
9 circos(R=300, cir="hg18", W=100, mapping=gene.exp, col.v=4, type="heatmap2", cluster=
10 T, col.bar=T, col.bar.po = "bottomright", lwd=0.01, zoom=zoom);
11 circos(R=220, cir="hg18", W=80, mapping=cnv, col.v=4, type="ml3", B=F, lwd=1,
12 cutoff=0, zoom=zoom);
13 circos(R=140, cir="hg18", W=80, mapping=pvalue, col.v=4, type="l", B=T, lwd=1,
14 col=colors[1], zoom=zoom);
15 circos(R=130, cir="hg18", W=10, mapping=TCGA.BC.fus, type="link", lwd=2, zoom=zoom);
16
17 # zoom in links by using the highlight functions
18 # highlight
19 the.col1=rainbow(10, alpha=0.5)[1];
20
21 # The highline region is radium from 140 to 400 and from position 282412.5 to
22 # 133770314.5 in chromosome 11.
23 highlight <- c(140, 400, 11, 282412.5, 11, 133770314.5, the.col1, the.col1);
24 circos(R=110, cir="hg18", W=40, mapping=highlight, type="hl", lwd=2, zoom=zoom);
25 the.col2=rainbow(10, alpha=0.5)[6];
```

```

21 highlight ← c(140, 400, 17, 739525, 17, 78385909, the.col2, the.col2);
22 circos(R=110, cir="hg18", W=40, mapping=highlight, type="hl", lwd=2, zoom=zoom);
23 ## highlight link
24 highlight.link1 ← c(400, 400, 140, 376.8544, 384.0021, 450, 540.5);
25 circos(cir="hg18", mapping=highlight.link1, type="highlight.link", col=the.col1, lwd
    =1);
26 highlight.link2 ← c(400, 400, 140, 419.1154, 423.3032, 543, 627);
27 circos(cir="hg18", mapping=highlight.link2, type="highlight.link", col=the.col2, lwd
    =1);
28
29 # The chromosome 11 region is going plotting from 180 (6 O'clock) to 270 degree (9
    O'clock).
30 zoom ← c(11, 11, 282412.5, 133770314.5, 180, 270);
31 circos(R=400, cir="hg18", W=4, type="chr", print.chr.lab=T, scale=T, zoom=zoom);
32 circos(R=300, cir="hg18", W=100, mapping=gene.exp, col.v=4, type="heatmap2", cluster=
    T, lwd=0.01, zoom=zoom);
33 circos(R=220, cir="hg18", W=80, mapping=cnv, col.v=4, type="ml3", B=F, lwd=1,
    cutoff=0, zoom=zoom);
34 circos(R=140, cir="hg18", W=80, mapping=pvalue, col.v=4, type="l", B=T, lwd=1,
    col=colors[1], zoom=zoom);
35
36 # The chromosome 17 region is going plotting from 180 (6 O'clock) to 270 degree (9
    O'clock).
37
38 gene.names ← c("ERBB2", "CDC6");
39 PAM50.17 ← which(TCGA.PAM50_genefu_hg18[,3]==gene.names);
40 TCGA.PAM50 ← TCGA.PAM50_genefu_hg18[PAM50.17,];
41
42 # zoom in chromosome 17
43 zoom ← c(17, 17, 739525, 78385909, 274, 356);
44 circos(R=400, cir="hg18", W=4, type="chr", print.chr.lab=T, scale=T, zoom=zoom);
45 circos(R=300, cir="hg18", W=100, mapping=gene.exp, col.v=4, type="heatmap2", cluster=
    T, lwd=0.01, zoom=zoom);
46 circos(R=220, cir="hg18", W=80, mapping=cnv, col.v=4, type="ml3", B=F, lwd=1,
    cutoff=0, zoom=zoom);
47 circos(R=140, cir="hg18", W=80, mapping=pvalue, col.v=4, type="l", B=T, lwd=1,
    col=colors[1], zoom=zoom);
48 circos(R=410, cir="hg18", W=40, mapping=TCGA.PAM50, type="label", side="out", col="
    blue", zoom=zoom);

```

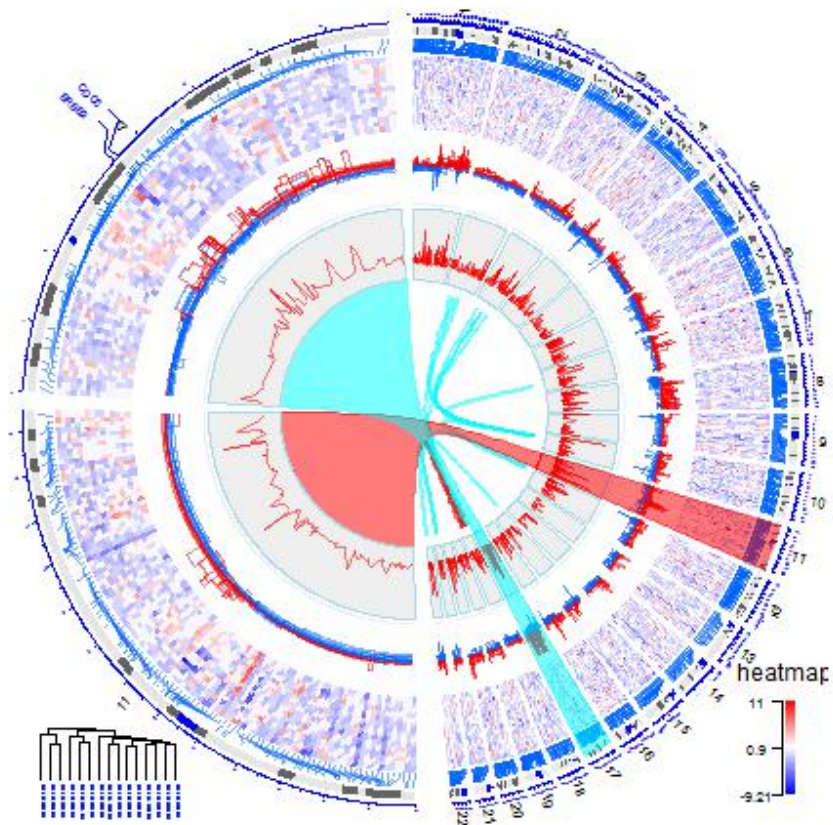


Figure 9