circlize implements and enhances circular visualization in R

Zuguang Gu[1,2], Lei Gu[1], Roland Eils[1,2,3], Matthias Schlesner[1] and Benedikt Brors[1,2,*]

[1]Division of Theoretical Bioinformatics, German Cancer Research Center (DKFZ), [2]Heidelberg Center for Personalized Oncology (DKFZ-HIPO) and [3]Department for Bioinformatics and Functional Genomics, Institute for Pharmacy and Molecular Biotechnology (IPMB) and BioQuant Center, Heidelberg University, 69120 Heidelberg, Germany

ABSTRACT

Summary: Circular layout is an efficient way for the visualization of huge amounts of genomic information. Here we present the circlize package, which provides an implementation of circular layout generation in R as well as an enhancement of available software. The flexibility of this package is based on the usage of low-level graphics functions such that self-defined high-level graphics can be easily implemented by users for specific purposes. Together with the seamless connection between the powerful computational and visual environment in R, circlize gives users more convenience and freedom to design figures for better understanding genomic patterns behind multi-dimensional data.

Availability and implementation: circlize is available at the Comprehensive R Archive Network (CRAN): http://cran.r-project.org/web/packages/circlize/

Contact: b.brors@dkfz.de

1 INTRODUCTION

As genomic information becomes more and more quickly accessible nowadays, a good solution for efficient visualization is urgently needed to better understand and decipher current genomic data. There are several ways to explore such high-dimensional genomic information, including genomic coordinates-based visualization or network views (Schroeder et al., 2013). Among them, circular layout is an advantageous approach reflected in three aspects: first, it elegantly represents information with long axes or a large amount of categories (e.g. data along several chromosomes); second, it intuitively shows data with multiple tracks focusing on the same object (e.g. multidimensional description on a same chromosome); third, it easily demonstrates relations between elements such as genomic rearrangements.

Circos is a pioneer tool widely used for circular layout representations implemented in Perl (Krzywinski et al., 2009). It greatly enhances the visualization of scientific results. Thus, plots with circular layout are normally named as 'circos plot'. However, limitations of Circos exist. Users need to combine data and graphical settings into configuration files, which means that the format of these files and a large number of parameters need to be remembered and set, making it tedious to configure

and debug. As a result, learning curves of Circos tend to be steep. On the other hand, users can only draw limited types of graphics, and new types of graphics are determined only by the software.

R is an ideal environment for statistical analysis and visualization. Currently, there are three R packages that aim to support circular layout, namely RCircos (Zhang et al., 2013), OmicCircos (Hu et al., 2014) and ggbio (Yin et al., 2012). Nevertheless, they are restricted to limited types of graphics, lack of flexibility and not extendable to other fields besides genomic analysis. Thus, there is still room for better use of the capabilities of R to improve the circular layout implementation.

Here, we present an R package, circlize, which efficiently and flexibly visualizes genomic data in a circular layout. The power of the package is based on the implementation of basic low-level graphics functions (e.g. drawing points and lines). Therefore, it is flexible to customize new types of graphics without difficulty. In addition, with the seamless connection between data analysis and visualization in R, automatic procedures for generation of circular designs can be easily achieved, which enhances the applicability of circlize in genomic analysis as well as in other fields. Finally, with the generality and simplicity of the package, circlize provides a basis on which high-level packages focusing on specific interests can be built.

2 IMPLEMENTATION

To map graphics onto the circle, there exist transformations from several coordinate systems. First, there is a data coordinate system in which ranges for x-axes and y-axes are the ranges of original data. Second, there is a polar coordinate system in which these coordinates are mapped onto a circle. Finally, there is a canvas coordinate system in which graphics are really drawn on the graphical device. circlize first transforms coordinates from data coordinate system to polar coordinate system and finally transforms into canvas coordinate system. circlize does all the transformations internally so that users only need to design graphs in the data coordinate system.

A circular layout is composed of sectors and tracks. For data in different categories, they are allocated into different sectors in which the width of each sector corresponds to the range of data in each category. For multiple measurements of the same category, they are represented as stacked tracks from outside of the circle to the inside. The intersection of a sector and a track is called a cell, which is the basic unit in a circular layout. It is an imaginary plotting region for data points in a certain category. Cells are independent from each other, and they can be either

---

*To whom correspondence should be addressed.

plotted together in a track as a batch mode or separately in a cell-by-cell fashion.

2.1 Low-level graphics functions

Generally, figures are composed from basic graphical elements, such as points, lines and regions. Our package, *circlize*, implements all these low-level functions (e.g. `circos.points`, `circos.lines`) to plot graphics in a circular layout so that more complicated graphics can be easily generated by different combinations of low-level graphics functions. These functions expect data points measured in the data coordinate system. The usage for the functions is almost the same as the functions in the traditional graphics system, which demonstrates the simple and user-friendly features of *circlize*.

2.2 Enhancement

Theoretically, based on the low-level graphics functions, users are able to draw most kinds of graphics (including basic graphics such as scatter plot and more complicated graphics) in the circular layout. Therefore, types of graphics are not restricted by the software itself and users can create circular graphics for their own purposes.

Under R's elegant statistical environment, data transformation can be first applied before visualization that enables *circlize* to support a large number of new types of graphics. For example, in Supplementary Figure S1A, smoothing by loess method is performed, and then smoothed lines as well as error regions are added to each cell implemented by `circos.lines` and `circos.polygon`. In Supplementary Figure S1E, clustering is performed to determine the order of elements in which heatmaps are drawn by `circos.rect`, and dendrograms are implemented by `circos.lines`.

With R's powerful graphics engine, more complicated graphics can be easily generated. For example, in Supplementary Figure S1B, two circos plots are drawn on the same graphical device represented as a nested circular layout. Such type of graphic is useful because it can zoom in a specific region on the outer circle. In Supplementary Figure S1D, two parts of circos plots are drawn together so that direct comparisons on the same genomic regions between two experimental conditions become achievable. For more detailed description of higher level techniques, users may refer to the main vignette of the package.

2.3 Genome graphics

The widest use of circular layouts today is to display genomic information. *circlize* particularly implements functions that facilitate genomic data visualization. The functions are built on general circos graphics functions and expect *BED*-like format of input data. Among them, there are low-level graphics functions such as `circos.genomicPoints` that add basic graphics and high-level functions such as `circos.initializeWithIdeogram` that initialize and organize genome circos plot.

Additionally, *circlize* provides customized functions `circos.genomicDensity` and `circos.genomicRainfall` for better visualizing genome-wide distributions of genomic features. For examples of basic and complex genome graphics, please refer to Supplementary Figure S1.

2.4 Examples

The process for drawing circular layouts in *circlize* is simple. It follows the sequence of (i) initializing the layout using `circos.initialize` or `circos.initializeWithIdeogram` to allocate different categories into sectors; (ii) creating plotting regions for the new track by `circos.trackPlotRegion` or `circos.genomicTrackPlotRegion`, and adding basic graphics with low-level graphics functions; (iii) repeating the second step to draw multiple tracks; (iv) finally calling `circos.clear` to do cleanings.

A typical chunk of code for drawing a circular layout would look like (pseudo code):

```
circos.initialize(factors, xlim)
circos.trackPlotRegion(factors, x.data, y.data,
  panel.fun = function(x, y) {
    circos.points(x, y, pch = 16, cex = 0.5)
    circos.lines(x, y, lwd = 0.5, type = "h")
})
circos.clear()
```

3 CONCLUSION

circlize provides a general and flexible solution for the circular layout. It provides basic low-level graphics functions, so that more complicated graphics can be easily implemented by users. Together with the R environment, the flexibility and enhancement make the package powerful in visualizing and deciphering genomic information, as well as in other related areas.

Conflict of Interest: none declared.

REFERENCES

Hu,Y. *et al.* (2014) OmicCircos: a simple-to-use R package for the circular visualization of multidimensional omics data. *Cancer Inform.*, **13**, 13–20.

Krzywinski,M. *et al.* (2009) Circos: an information aesthetic for comparative genomics. *Genome Res.*, **19**, 1639–1645.

Schroeder,M.P. *et al.* (2013) Visualizing multidimensional cancer genomics data. *Genome Med.*, **5**, 9.

Yin,T. *et al.* (2012) ggbio: an R package for extending the grammar of graphics for genomic data. *Genome Biol.*, **13**, R77.

Zhang,H. *et al.* (2013) RCircos: an R package for Circos 2D track plots. *BMC Bioinformatics*, **14**, 244.