

Contents

Manual Reference – sbv	1
INSTALLATION	1
INTRODUCTION	2
PIE	6
GGPLOT2	8
BUBBLE	22
RPLOT	23
MAPLOT	24
MANHATTAN	26
FREQ	27
VENN	29
LASV	32
TREE	33
TAXTREE	40
HEATMAP	41
KARYO	42
HCGD	48
SDU	51
PPI *	53
ACCESSORY	54
Symbol reference list	54

Manual Reference – sbv

INSTALLATION

1. Requirement

To run sbv, you need perl. The sbv software was written and tested by perl 5.10.1.

After perl was installed, you'll need to install the following perl modules to run sbv.

- 1) SVG (v2.59 or later)
- 2) Config::General (v2.50 or later)
- 3) Math::Round
- 4) Math::Cephes
- 5) PostScript::Font (v1.10.02 or later)
- 6) Readonly
- 7) BioPerl (v1.6.922 or later, optional, just for read phylogenetic tree file in command 'tree' and 'heatmap')

8) Algorithm::Cluster (v1.52 or later)

2. Installation

To install sbv, you just need to download sbv.tar.gz and decompressed it. The folder you installed sbv is marked as \$SBV_HOME, and can be omitted to describe the file path at below of this article. Like '\$SBV_HOME/bin/sbv.pl' equals as 'bin/sbv.pl'.

3. The context of sbv

When you download and unpack sbv, you'll have a directory structure as follows.

SBV => the sbv home directory (\$SBV_HOME)

bin => the main executable perl script of sbv is in this directory

data => some test and example data files are in this directory

doc => the manual document is in this directory

etc => the configuration files live here, include sbv global configuration files, such as colors.conf and styles.conf

example => the example configuration files of some type figure are in this directory

fonts => The afm fonts files used by sbv.

Images => the additional images used in figures can be put in this directory, like background images.

Js => the js file you need in the figures can be put in this directory

lib => Code libraries

src => some tools which can help you to use sbv

tmp => the temporary files are put in this directory

INTRODUCTION

1. Quick start

Use the command as follow to run sbv quickly, just for impatient people

perl sbv.pl <figure type> [-conf *.conf] [options] [input data file]

1.1 Figure type

The type of the figures, like 'pie', 'ggplot2', 'histogram', 'venn', 'tree', 'karyo' and so on, whose will be described particularly at the below chapters respectively.

1.2 Options

--strict	run sbv strict, output the warning information in STDERR
--quiet	run sbv quiet, don't output the log information
--man	output the detailed help information to screen
--help	output the simple help information to screen
--conf	the configuration file for sbv
--pattern	the style pattern of the figure
--out	the out file key name

Note: if --conf is not set, the default value will be set as the figure type pastes the suffix

'conf', for example the 'pie' type, the default value will be 'pie.conf'. Then this program will search this name in those folders in order,

- 1) The current folder.
- 2) \$SBV_HOME/data
- 3) \$SBV_HOME/etc
- 4) \$SBV_HOME/example

This file search rule is also fit for input data file and other files defined in the configuration file.

1.3 Input data file

Be optional, the input data file for some figure type, you can set it in configuration file also.

2. Configuration file

Configuration files are parsed using Config::General module. All pertinent features are described below, but for those so inclined, refer to this [module's man page](#) to learn about the details of the syntax and parsing of these files.

Settings are defined in configuration files using the format

variable = value

Note: Please ensure that all your configuration blocks are correctly terminated with an appropriate close block tag. Like

<pie>

...

</pie> # ← if this is missing, an error will result.

The attributes in configuration files falls into three categories.

2.1 Basic attributes, which are defined outer of the root blocks

width, height => define the size of the figure, default is 600 x 600.

margin => the margin of 4 figure boundary, default all are 20. You can defined it as follow,

margin = 20 , all the 4 boundary margin are 20.

margin = 20 40 20 40, the top, right, bottom, left boundary margin are 20, 40, 20, 40 respectively.

background => the background color, default 'none'

border => the border of the 4 boundary, default is 0 (0 means not draw, 1 means draw).

You can defined it as follow,

border = 0, all 4 direction border is deletion

border = 0100, 4 chars defined the border of top, right, bottom, left respectively.

hspace => the space of horizontal, default is 4.

vspace => the space of vertical, default is 4.

dir => the output file directory, default is the current directory ('.').

file => the output file name (sbv will add suffix 'svg' auto), default is the figure type

command, for example the 'pie' figure type, the default out file is 'pie.svg'.

2.2 Public attributes, which can be defined in all block

x, y, width, height => defined the region of the block, default is same with the parent block. The value can be relative (use the decimal which less than 1).

margin, background, border => the definition is same with the basic attributes

2.3 Private personal attributes, for each figure type, which will be described in next sections below.

3. Colors

The colors attributes defined format is like below, the separator between two colors must be space.

background = eee

fill = red blue rgb(0,200,0) chr1

color = hsv(0,1,1) hsv(0,1,0)

Many type colors format, which is displayed as follow, are supported by sbv.

1) RGB, use the format as follow to define

color = rgb(0,200,0)

2) HSV, use the format as follow to define

color = hsv(0,1,1)

3) The color of hex, the '#' is omitted, like

97ff10, 9f0 (=99ff00)

4) The color name you defined in the block of <colors>

Like general color name, 'red', 'blue', 'yellow' etc.

Besides you can defined the color name as you want, for example 'chr1', 'chr2', 'A', 'B' etc.

But you need to set those in the block of <colors>. The file 'etc/colors.conf' contains all color name defined by sbv. You just need to import those by add the command as follow to your configuration files.

<<include etc/colors.conf>>

About the colors name definition you can refer to [Circos web sites](#) for more information.

4. Fonts

The attributes of fonts contains

1) font-size: the unit default is px, you can set pt also.

2) font-weight: 'normal' or 'bold'

3) font-style: 'normal' or 'italic' ('oblique')

4) font-family: refer to 'fonts/README.txt' to get more information

5) fill: the color the font

You can change the font attributes in the 'styles.conf' file, or you can set the font theme in configuration files for text like this,

title_theme = size:14;family:arial;weight:bold;angle:0;

5. Styles

<styles> block is used to defined the **css** style of svg figure like as follow.

```
<styles>
<text>
font-size = 14px
font-family = Helvetica
font-weight = normal
font-style = normal
fill = black
</text>
<text>
class = title
font-size = 22px
font-family = Arial
font-weight = bold
font-style = italic
fill = black
</text>
</styles>
```

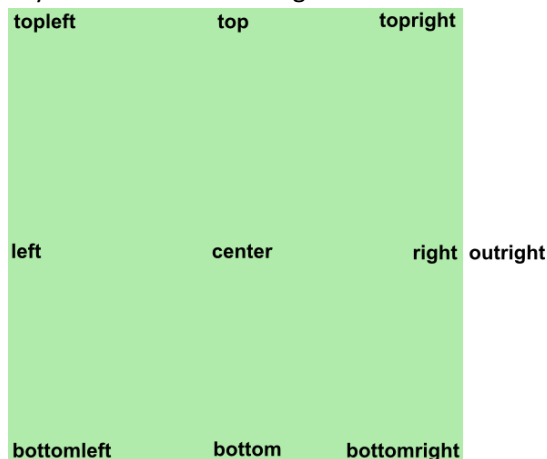
This <styles> block will decide the default or specific class svg element style attributes. About 'css', you can refer to [w3school](http://www.w3school.com) for more information.

6. Legend

This block is used to define the legend on the figure like file 'etc/legend.conf'

The attributes contains

- 1) background, border, margin
- 2) pos => the location of the legend, you can set it as coordinate value like '(x,y)' or keyword. Default is 'outright'.



- 3) title => the title of legend, optional
- 4) title_pos => the legend title location, Default is 'top'.
- 5) title_theme
- 6) title_hjust
- 7) title_vjust

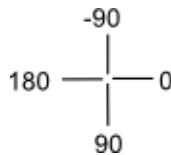
- 8) label => set the labels string, needed.
- 9) label_show => show labels or not, Boolean type, default is yes.
- 10) label_pos => the label text location. Default is 'right'.
- 11) label_theme
- 12) label_hjust
- 13) label_vjust
- 14) ncol or nrow => set the symbols number of each column or row. Default is ncol = 1.
- 15) byrow => draw symbols by row order or not, Boolean type, default yes.
- 16) reverse => draw the symbols in reverse order.
- 17) shape => the symbol shape id, default is 1 (circle).
- 18) color => the stroke color, default is 'black'
- 19) fill => the symbol fill color, default is 'none'
- 20) width => the symbol width, default is 20
- 21) height => the symbol height, default is 20
- 22) hspace => the legend horizontal space, default is the sbv 'hspace' value
- 23) vspace => the legend vertical space, default is the sbv 'vspace' value

7. Axis

The two blocks, <xaxis> and <yaxis>, define the xaxis and yaxis information respectively, just used in Direct coordinate system figures, like 'ggplot2', 'rplot', 'freq' etc.

The attributes contains

- 1) angle => the axis angle between with three o'clock direction at clockwise. Default is 0.



- 2) bone => display the axis bone line or not, Boolean type.
- 3) tail => add the arrow to the axis tail, default null.
- 4) tick => set the axis tick values
- 5) side => set the tick side, 'left' or 'right', default is right.
- 6) start => set the axis start tick position, 0-1 unit, default is 0.2
- 7) show_tick_line => display the tick line or not, default yes.
- 8) size => set the main tick size, default is 8.
- 9) show_tick_label => show tick label or not, default yes.
- 10) ticktext => reset the tick text to replace the default tick text created by tick values.
- 11) multiple => show the tick value multiplied by this value, default is 1.
- 12) unit_label => paste this value after each tick label, default is null.
- 13) translate => translate the tick labels by 0-1 unit, default is 0.
- 14) theme => the tick label theme, default null.

PIE

With this figure type, you can draw a pie or ring chart.

1. Quick start

Use this command to draw a pie chart quickly as follow. This command will use the default pie configuration file 'example/pie.conf' to draw your pie figure with the input file set in command.

perl sbv.pl pie [options] <input file>

2. Input data file

The input data file for pie figure contains two or three fields as follow, the last field is optional.

<object> <value> [attributes]

The separator between fields must be <tab>. The attributes of pie object are defined at format as follow,

attr1=val1;attr2=val2;.....

The detail attributes information will be described in section 3 (configuration).

3. Configuration

You can change your pie style or context by revising the pie configuration, and then use the command as follow to draw your personal pie figure.

perl sbv.pl pie -conf your_pie.conf [options] [input file]

You can set the attributes as follow. These attributes must be contained in block <pie>.

1) Public attributes

2) Private pie figure attributes but can't be set in input data file

file => input data file, you can set it in command arguments also

title => the title of the pie figure, optional

filter => the filter used in pie figure(now just support '01'), default not use filter.

show_legend => show legend at the right of pie graph or not (Boolean type), default no

legend => the legend block used to defined the legend, effective while show_legend is true

3) Private pie figure attributes and can be set in input data file

color => the colors used, default is rainbow colors.

r0 => the inner arc radius, default is 0.

r1 => the outer arc radius, optional, can't overflow the figure bounds.

raise_r => the radius of the object deviate from the circle center, default is 0.

show_label => show the label of object in the figure or not (Boolean type), default yes

show_val => show the value of object after label or not (Boolean type), default yes, effective while show_label is true.

show_percentage => show the percentage value of object after label or not (Boolean type), default no, effective while show_label is true.

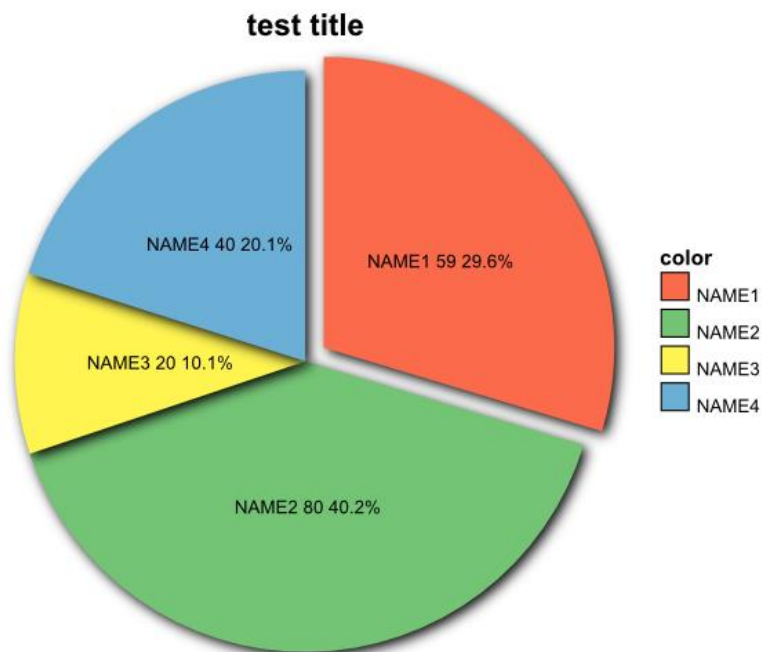
show_label_orientation => the orientation of the label in the fan of object, 'inner' or 'outer', default is 'inner'.

show_label_links => show the link line of label or not, default no, effective while show_label_orientation is 'outer'.

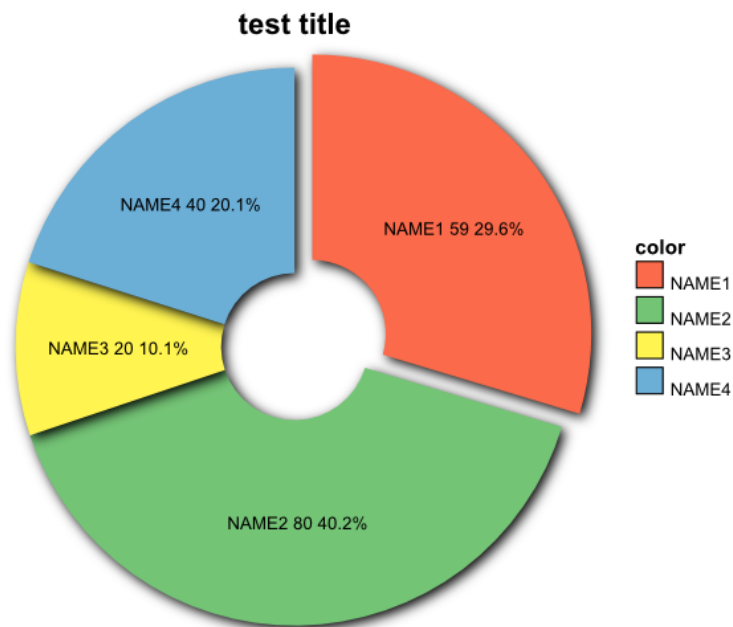
label_links_length => the link line length, default is 10

4. Example

The pie configuration example file is 'example/pie.conf', the figure created by this configuration file is displayed as follow.



If set the $r0 = 50$, will generated a ring chart like below.



GGPLOT2

Ggplot2 is data visualization package for statistical programming language R. In sbv, ggplot2 figure type can do some simple function of ggplot2 package in R, like draw scatterplot, histogram,

boxplot, ribbon etc. which will be described as follow. You can refer to [the official website of ggplot2](#) to get more information.

1. Quick start

Use the command as follow to draw a ggplot2 file,

perl sbv.pl ggplot2 [options] <input data file>

2. Input data file

The input data file is a table format file.

3. Configuration

The attributes must be put in block <ggplot2>.

file => the input data file

header => the input data file contains header description or not (Boolean type), default no.

rownames => the input data file contains row names (first field) or not (Boolean type), default no.

xlab => x axis title, default no xlab

ylab => y axis title, default no ylab

title => the title of the figure, default no title

xlim => set the axis limitation, format like: min,max[,step]

ylim => set the yaxis limitation, format like: min,max[,step]

col => the stroke color used in ggplot2, default is "black"

fill => the fill color used in ggplot2, default is "none"

shape => the symbol shape id used in ggplot2, default is 1.

xnames => the x axis ticks is string or not, default no.

<xaxis> => the axis block used to define the x axis

<yaxis> => the axis block used to define the y axis

<legend> => the legend block used to define the legend

<eval> block, the main command of ggplot2, which will described detailed as follow.

4. eval

The ggplot2 command is defined in this block using the format as follow

<eval>

1 = lines()

2 = points()

...

</eval>

The commands are separated by line feed, and start with "number id =", than sbv will run those commands according to the number ids order.

The general arguments used in ggplot2 command

x: the x coordinate vector or a field name of the data, default is the first filed of the data.

y: the y coordinate vector or a field name of the data, default is the second field of the data.

col: the stroke color vector or a field name of the data, can be defined like below.

lines(col=>"V3") filed name.

lines(col=>["red","green"]) colors vector pointer

lines(col=>"red green") colors string separated by space

fill: the fill color vector or a field name of the data, can be defined like 'col'

shape: the shape symbol id vector or a field name of the data

stroke_width: the stroke width, default is 1.

group: the field name of the data, used to divide the data into groups.

width: the percentage of unit width, 0-1, group is false, default is 0.6, otherwise is 0.8.

The ggplot2 commands include,

4.1 points

4.1.1 arguments

Add points on the figure. The arguments are displayed as follow.

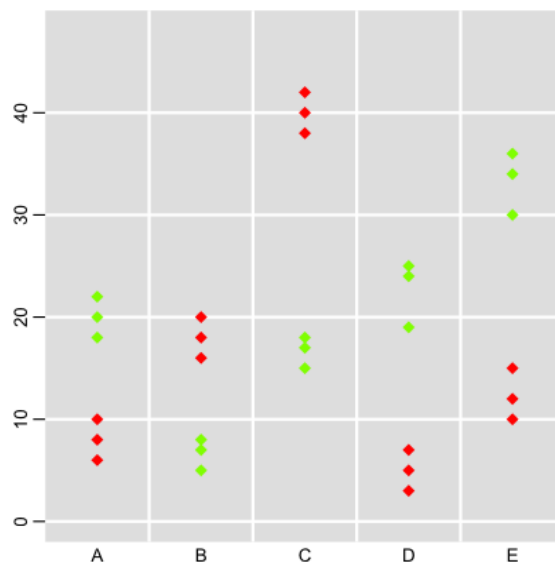
x, y, fill, col, shape, stroke_width, size, group

size: the symbol radius size, default is 5.

4.1.2 Example

file = ggplot2/bar.test.dat

```
1 = points(x=>'V1',y=>'V2',fill=>'V4',group=>'V4',shape=>['2'],size=>4);
```



4.2 lines

4.2.1 arguments

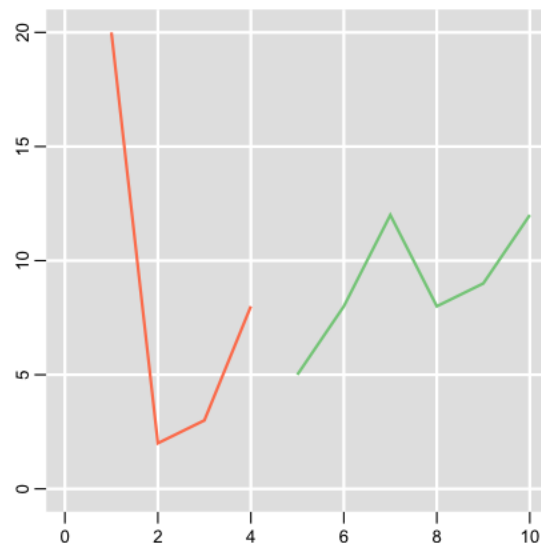
Add polyline on the figure. The arguments are displayed as follow.

x, y, col, group, stroke_width, stroke_dashes

4.2.2 Example

file = ggplot2/plot.test.dat

```
1 = lines(x=>'V1',y=>'V2',group=>'V4',stroke_width=>2)
```



4.3 density

draw density polyline

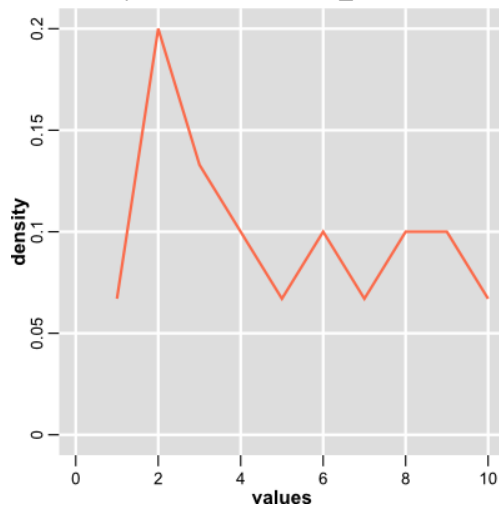
4.3.1 arguments

x, col, stroke_width, stroke_dashes

4.3.2 example

file = ggplot2/density.test.dat

1 = density(col=>"red",stroke_width=>2)



4.4 abline, hline and vline

4.4.1 abline

Add a straight line on the figure. The arguments are displayed as follow.
The first two arguments are the slope value and y axis intercept value.

4.4.2 hline

Add a horizontal straight line on the figure.

The first argument is the y axis intercept value vector.

4.4.3 vline

Add a vertical straight line on the figure.

The first argument is the x axis intercept value vector.

4.4.4 Arguments

col, stroke_width, stroke_dashes

4.4.5 Example

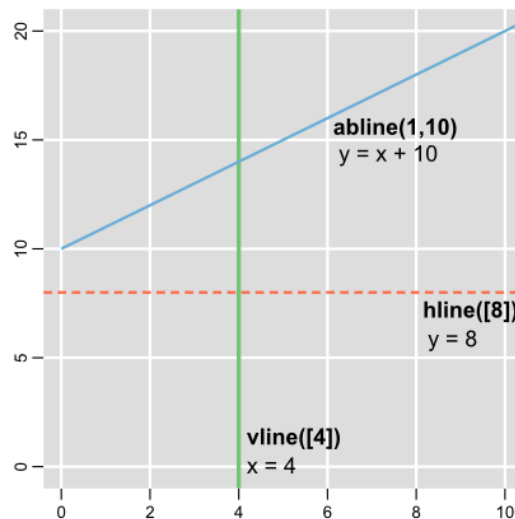
file = ggplot2/plot.test.dat

0 = lines(col=>'none')

1 = hline([8],col=>"red",stroke_width=>2,stroke_dashes=>"6 4")

2 = vline([4],col=>"green",stroke_width=>3)

3 = abline(1,10,col=>"blue",stroke_width=>2)



4.5 bar

4.5.1 arguments

add rectangles on the figure. The arguments are displayed as follow.

x, y, col, fill, group, stat, width, size, pileup, val

size: effective with 'group', the zoom size of the width, 0-1, default is 1.

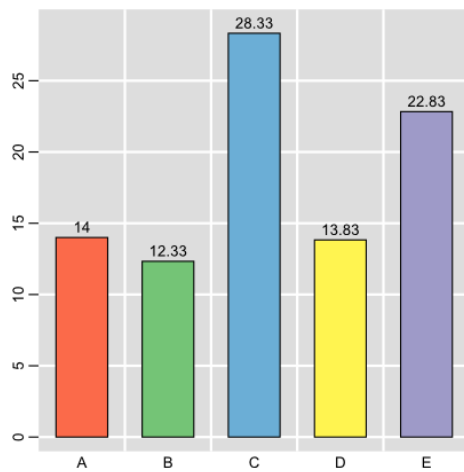
pileup: effective with 'group', pileup the bars or not, Boolean type, default no.

val: add the val at the top of the bar.

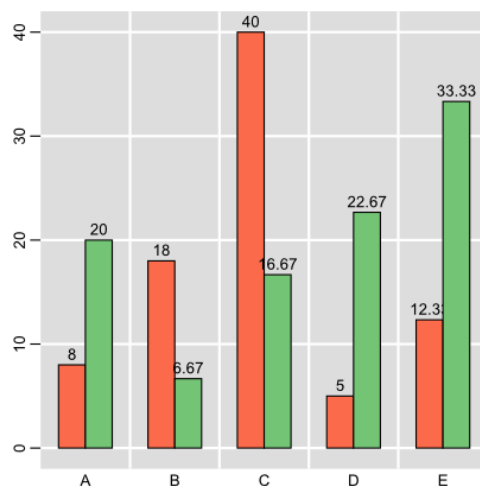
4.5.2 Example

file = ggplot2/bar.test.dat

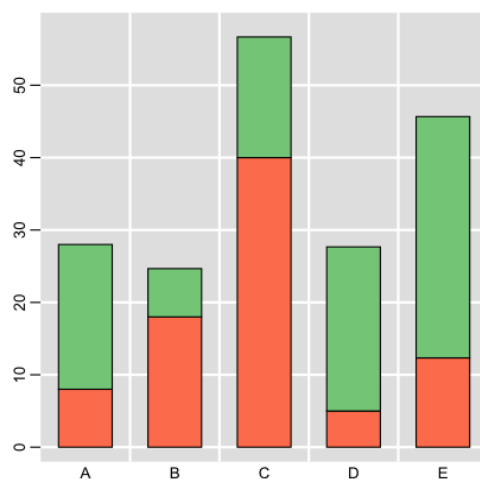
1 = bar(stat=>'mean',val=>1,width=>0.6)



1 = bar(stat=>'mean',val=>1,width=>0.6,group=>'V4')

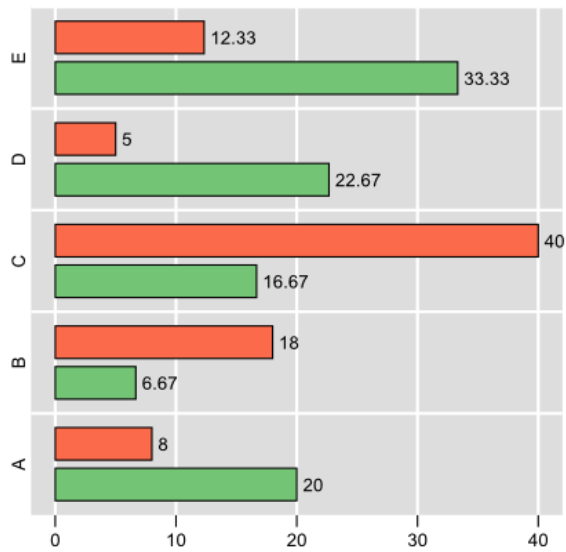


1 = bar(stat=>'mean',width=>0.6,group=>'V4',pileup=>1)



flip = yes

1 = bar(stat=>'mean',width=>0.8,size=>0.8,group=>'V4',pileup=>0,val=>1)



4.6 Boxplot

4.6.1 arguments

add boxplot on the figure. The arguments are displayed as follow.

x, y, fill, col, group, width, size

size: effective with 'group', the zoom size of the width, 0-1, default is 0.9.

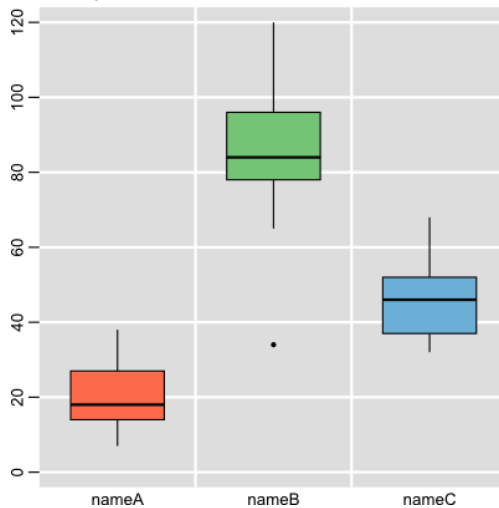
4.6.2 example

file = boxplot.test.dat

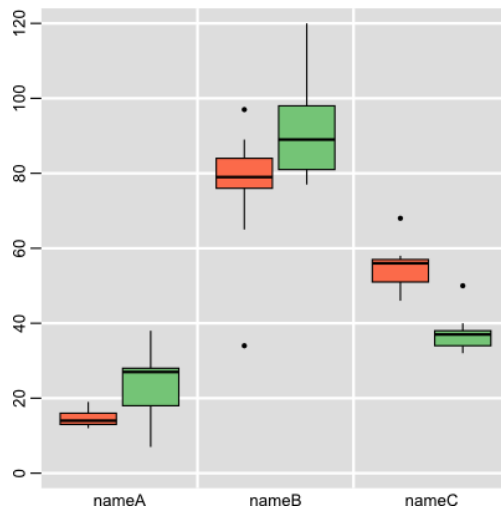
fill = red green blue

col = black

1 = boxplot()



1 = boxplot(group=>'V3')



4.7 text

4.7.1 arguments

x, y, z, fill, group, weight, size, style, family, angle, hjust, vjust

z: the text label, a field name of the data or a label array, default is the third field of the data.

weight: 'font-weight' text style attribute, 'normal' or 'bold', default is 'normal'.

style: 'font-style' text style attribute, 'normal' or 'italic', default is 'normal'.

size: 'font-size' text style attribute, default is 12px.

family: 'font-family' text style attribute, default is 'Arial'.

angle: the text rotate angle, default is 0.

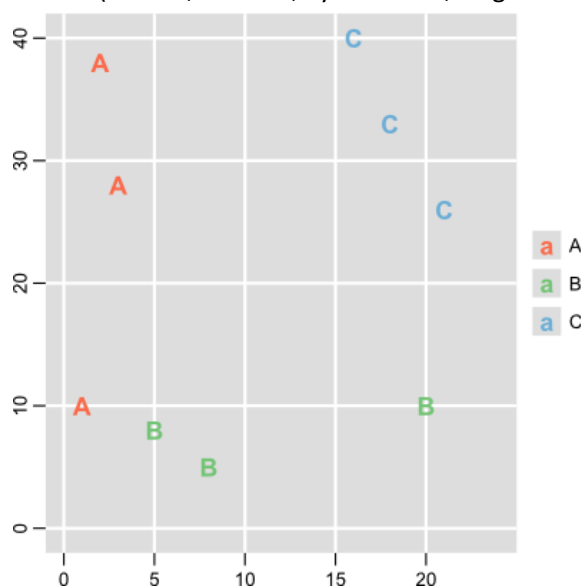
hjust: text horizontal translation percentage, 0-1, default is 0.5.

vjust: text vertical translation percentage, 0-1, default is 0.5.

4.7.2 example

file = ggplot2/text.test.dat

1 = text(z=>'V3',size=>20,style=>'italic',weight=>'bold',fill=>'red')



```

1 = points(fill=>'red',size=>3)
2 = text(x=>[2],y=>[38],z=>["default"])
3 = text(x=>[1],y=>[10],z=>["angle=45"],angle=>45)
4 = text(x=>[3],y=>[28],z=>["angle=45;hjust=0;vjust=0"],angle=>45,hjust=>0,vjust=>0)
5 = text(x=>[5],y=>[8],z=>["hjust=0;vjust=0"],hjust=>0,vjust=>0)
6 = text(x=>[8],y=>[5],z=>["size=16"],size=>16)
7 = text(x=>[20],y=>[10],z=>["weight=bold,style=italic"],weight=>"bold",style=>"italic")
8 = text(x=>[16],y=>[40],z=>["family=Book Antiqua"],family=>"Book Antiqua")
9 = text(x=>[18],y=>[33],z=>["fill=blue"],fill=>"blue")

```



4.8 ribbon/area

4.8.1 arguments

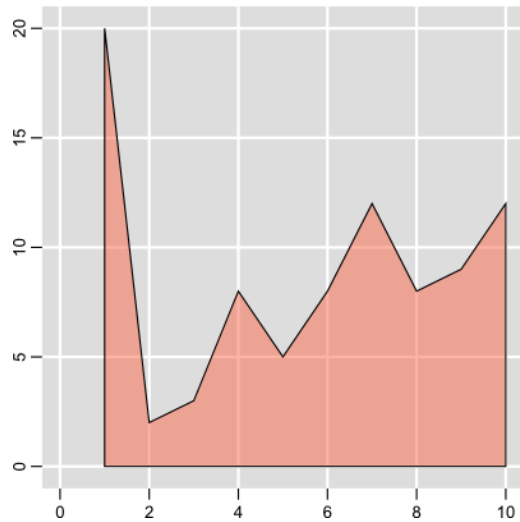
x, y, col, fill, group, ymin, opacity

opacity: the path fill opacity, default is 0.5

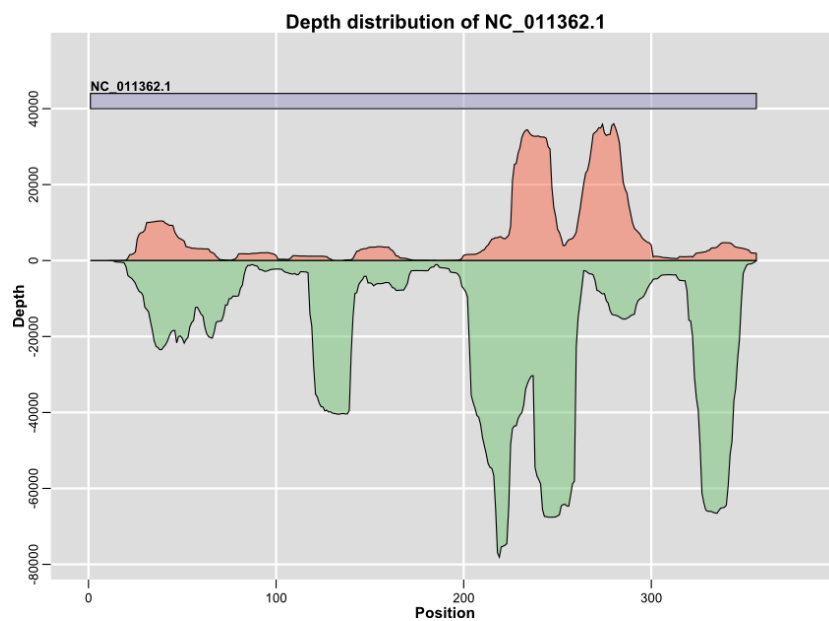
4.8.2 example

file = plot.test.dat

```
1 = ribbon(fill=>"red",col=>"black")
```

```
file = ggplot2/temp.depth.dat
<eval>
1 = ribbon(x=>'V1',y=>'V2',fill=>'red');
2 = ribbon(x=>'V1',y=>'V3',fill=>'green');
3 = polygon(x=>[1,356,356,1],y=>[44000,44000,40000,40000],fill=>'purple');
4 = text(x=>[1],y=>[46000],z=>['NC_011362.1'],hjust=>0,weight=>'bold');
</eval>
<yaxis>
tick = -80000 60000 20000
</yaxis>
xlab = Position
ylab = Depth
title = Depth distribution of NC_011362.1
```



4.9 polygon

4.9.1 arguments

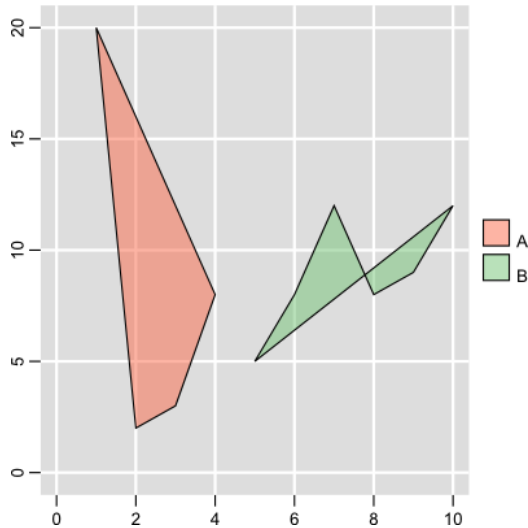
x, y, col, fill, opacity

opacity: the path fill opacity, default is 0.5

4.9.2 example

file = ggplot2/plot.test.dat

```
1 = polygon(group=>'V4',fill=>"red green")
```



4.10 range

4.10.1 arguments

type, x, y, group, col, fill, width, size, stat, legend

the range type: linerange, pointrange, crossbar, errbar. Those four types can be used directly.

fill: just effective with crossbar

width: just effective with crossbar and errbar

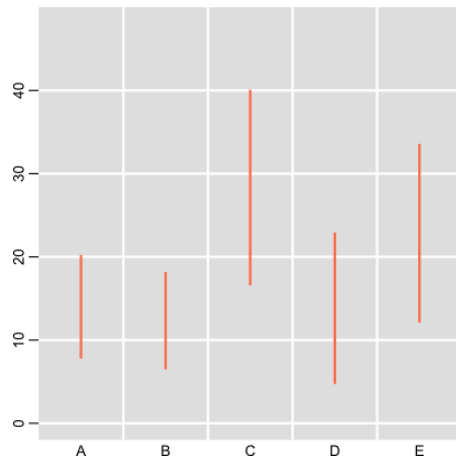
stat: the method to stat the range of values, 'sem' or 'normal'. 'normal' range is the min, mean and max. 'sem' is the standard error of the mean. Default is sem.

legend: show the range legend or not, default no.

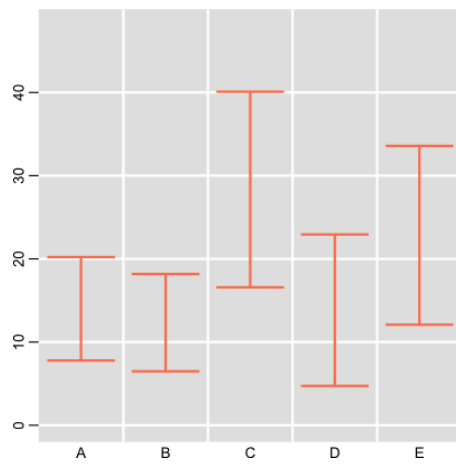
4.10.2 example

file = ggplot2/bar.test.dat

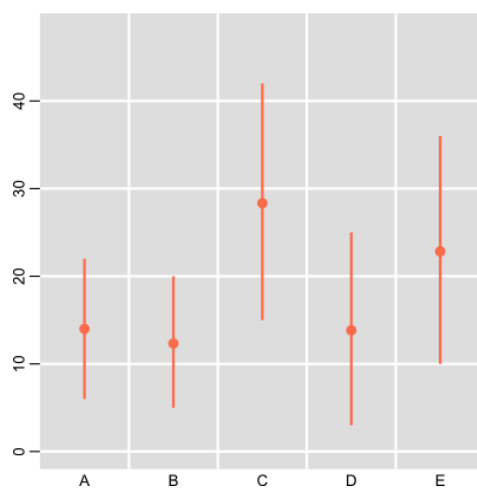
```
range('linerange',stroke_width=>2,col=>"red") or linerange(stroke_width=>2,col=>"red")
```



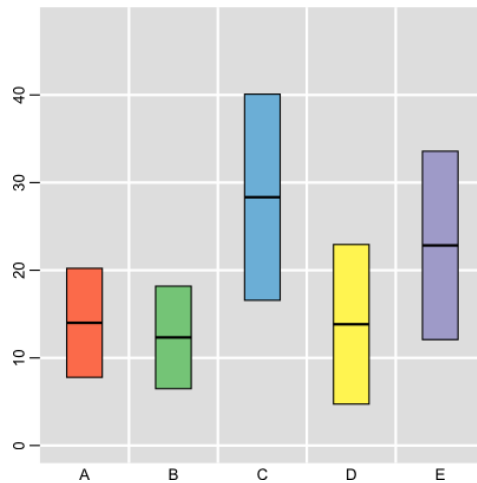
`range('errbar',col=>"red",width=>0.8)` or `errbar(col=>"red",width=>0.8)`



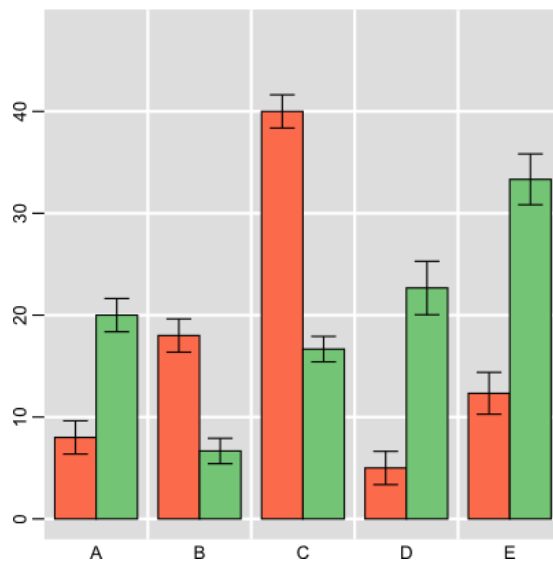
`range('pointrange',col=>"red",stat=>'normal')` or `pointrange(col=>"red",stat=>'normal')`



`range('crossbar',col=>'black',width=>0.4)` or `crossbar(col=>"black",width=>0.4)`



```
1 = bar(group=>'V4',pileup=>0)
2 = errbar(group=>'V4',size=>0.6,stroke_width=>1)
```



4.11 bubble

4.11.1 arguments

x, y, z, group, col, fill, zoom

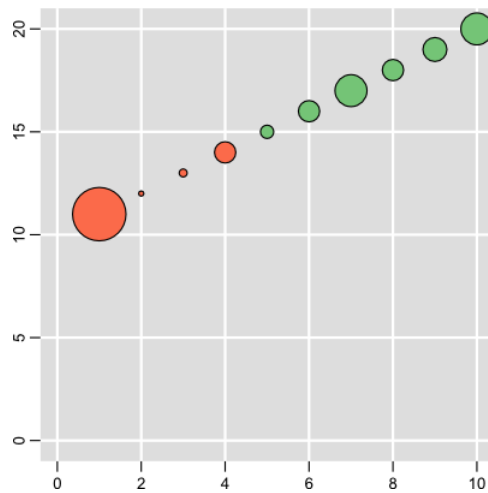
z: the radius size. A vector or a field name of the data, default is the third field of the data.

zoom: the z value zoom size, default is 1.

4.11.2 example

file = plot.test.dat

```
1 = bubble(x=>'V1',y=>'V3',z=>'V2',group=>'V4',zoom=>1)
```



4.12 xlab, ylab, title, labs

The style of the title and axis sub title is defined in the styles.*.conf file by text class 'title' and 'subtitle' respectively.

4.12.1 xlab

add x axis title on the figure.

xlab("x axis title")

4.12.2 ylab

add y axis title on the figure

ylab("y axis title")

4.12.3 title

add title on the figure

title("title");

4.12.4 labs

set the x axis title, y axis title or figure title on the figure.

labs(x=>"x axis title", y=>"y axis title", title=>"title");

4.12.5 example

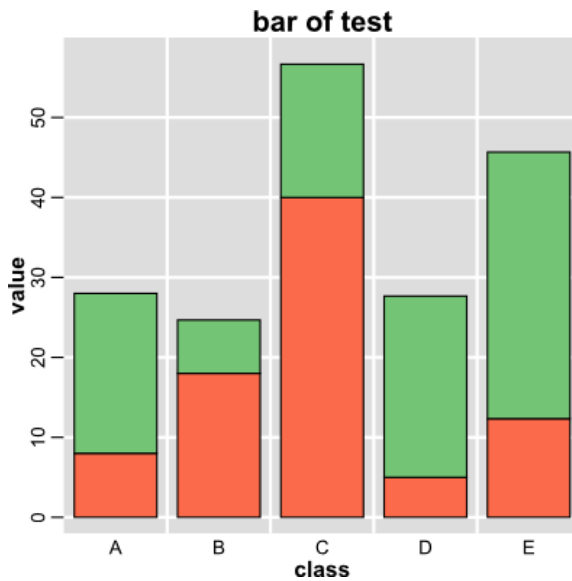
file = bar.test.dat

1 = bar(stat=>'mean',width=>0.8, group=>'V4',pileup=>1)

2 = xlab("class")

3 = ylab("value")

4 = title("bar of test");



4.13 aes

set the arguments of the ggplot2

`aes(col=>"xxx",fill=>"xxx",.....);`

4.14 read

load a new data file

`read("file_name",header=>0,rownames=>0)`

4.15 setXlim, setYlim

set the xaxis and yaxis limit, the first and second values are the minimum and maximum value restrictively, and are needed, the third value is the step/window value, and is optional.

`setXim(1,10,20)`

`setYlim(20,100)`

BUBBLE

A [bubble chart](#) is a type of chart that displays three dimensions of data. You can refer to Wikipedia for more details. Use this figure type you can draw a simple bubble chart, for the complex bubble chart (like bubble with different stroke colors and fill colors), you can turn to 'ggplot2' figure type.

1. Quick start

Use the command below you can draw a bubble chart, this command will use the default configuration 'example/bubble.conf' and the input file to draw your bubble chart.

`perl sbv.pl [options] <input data file>`

2. Input data file

The input data file for bubble contains at least tree fields. The first field is for x axis, the second field is for y axis, the third field decides the radius size of the bubble circle.

`<v1> <v2><v3>`

3. Configuration

The attributes must be put in block <bubble>.

The private personal attributes of 'bubble' contains:

fill => the circle points fill colors, default is 'black'.

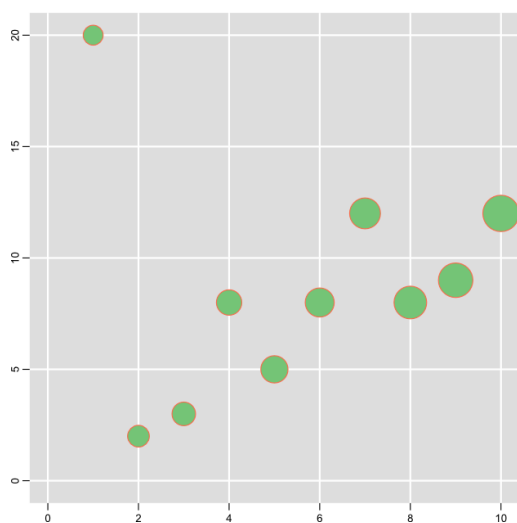
col => the circle points stroke color, default is 'black'.

z_val_multiple => the third value multiple size, this will decides the radius size of the circle points, default 1.

xab, ylab, title => the tree attributes is same as the 'ggplot2', default all null.

4. Example

The default configuration file is 'example/bubble.conf', this figure created by this file is displayed as follow.



RPLOT

This figure type 'rplot' is used to draw a chart named "range plot", which made up by three parts,

- The 'linerange': a line displayed the minimum and the maximum value of a group data. The stroke color is 'black' and can't be changed.
- The 'points': a point displayed the mean value of one group data, no stroke color, but the fill color can be set.
- The 'lines': a fold line connect all points of b. of one file. The stroke color used the same color with the points.

The figure is designed for displaying the core and pan genes of many genomes.

1. Quick start

The command below can help you to draw your range plot with the default configuration file 'exmpale/rplot.conf' and the input data files.

perl sbv.pl rplot [options] <input data file>s

2. Input data files

The figure type can have multi input files, the file contains format like this:

<genomes number><gene num1> <gene num2> <gene numN>

Contains two parts, the first field is the number of genomes, other fields are the number of genes stat result. The data must be sorted by the first field in ascending order.

3. Configuration

The attributes must be put in block <rplot>.

The private personal attributes of 'rplot' contains:

col => the colors of the points fill and the fold line stroke, default is 'red green blue', **An input file corresponds to a color.**

size => the radius size of the circle points, default is 3.

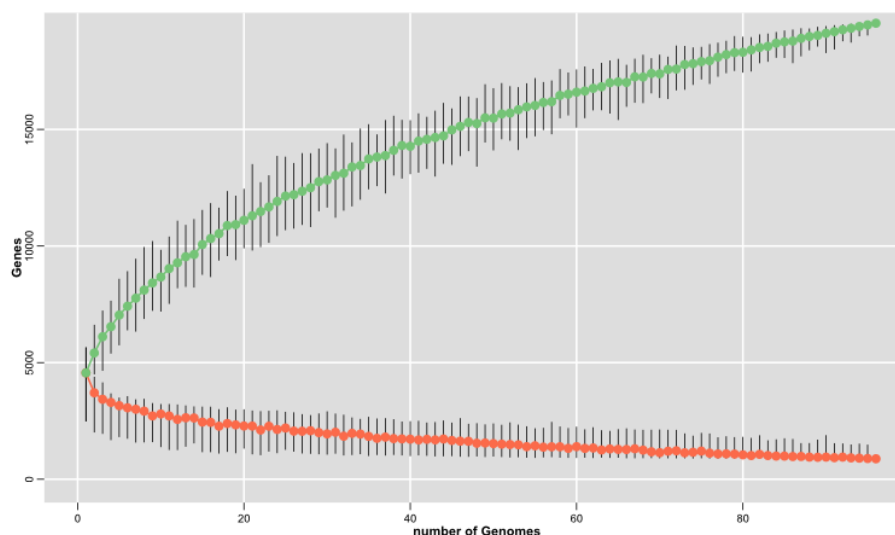
xlab => x axis title, default is 'number of Genomes'

ylab => y axis title, default is 'Genes'.

title=> the figure title, default no title.

4. Example

The example configuration file is 'example/rplot.conf', the figure created by this file is displayed below.



MAPLOT

An [MA plot](#) is an application of a Bland–Altman plot for visual representation of two channel DNA microarray gene expression data which has been transformed onto the M (log ratios) and A (mean average) scale. You can refer to Wikipedia for more details.

1. Quick start

Use the command as follow to draw a MA plot, this command will use the default manhattan configuration file 'example/maplot.conf' and the input data file to draw your MA plot.

perl sbv.pl maplot [options] <input file>

2. Input data file

The input data file for MA plot is the gene expression value of different samples. The format is like this.

Header: <GeneID> <Sample1> <Sample2> ... <Sample n> [Different Expression]

Main: <Gene1> <Exp1> <Exp2> ... <Exp n> <no/up/down>

.....

Note:

- 1) To draw the MA plot, sbv just need two samples gene expression value, so you need pick two samples by the attributes of 'ev1' and 'ev2' in configuration file.
- 2) The header and gene id information are both optional, you can set them by the logical type attributes of 'header' and 'rownames' in configuration file respectively.
- 3) If you have the information of different expression gens, you can add it to the input file, sbv will draw the genes in MA plot with different colors.
- 4) If you want to add fold lines to MA plot to help people understand this figure, the <foldline> block in configuration file can do it.

3. Configuration

Besides the basic attributes, other attributes in the configuration file of MA plot must be put in block <maplot>. The attributes of a MA plot are displayed as follow.

- 1) Public attributes

- 2) Private personal attributes

file => the input data file

header => the input data file contains header description or not (Boolean type), default no.

rownames => the input data file contains row names (first field) or not (Boolean type), default no.

ev1, ev2 => the field values used to draw MA plot, default is the first two field (the row names field is omitted).

deg => the field defined the gene different expression information, default no.

size => the points size (radius), default is 3

fill => the points fill colors, default is 'black red green'

xlab => x axis title, default is 'A'

ylab => y axis title, default is 'M'

title => title of the MA plot, default no title

<hline> => block, defined the horizontal split lines in MA plot, default no hline

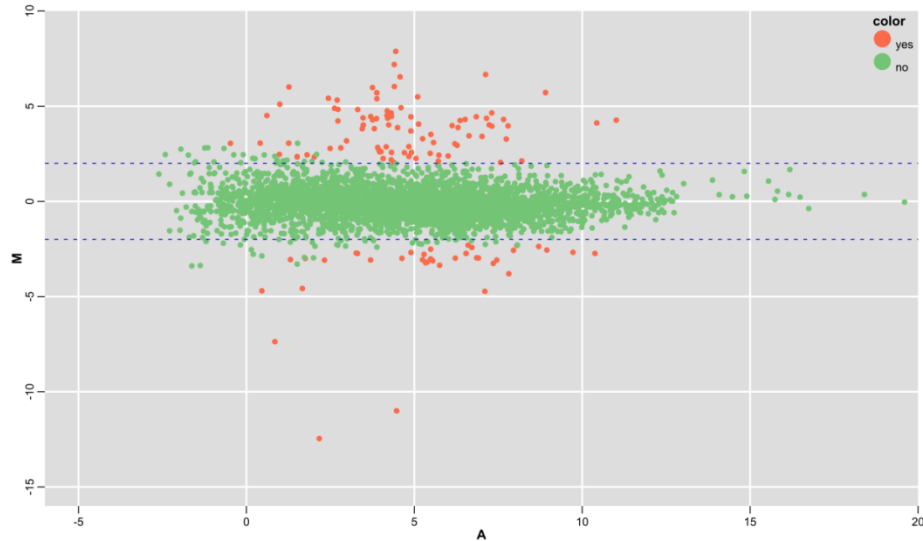
fold => the fold value, default is 1

stroke_color, stroke_width, stroke_dasharray => define the styles of the lines, default values are 'red', 1 and '3 3' respectively.

</hline>

4. Example

The example of MA plot configuration file is 'example/maplot.conf'. The picture below was created by this configuration file. The blue horizontal auxiliary lines are the split line.



MANHATTAN

A [Manhattan plot](#) is a type of scatter plot, usually used to display data with a large number of data-points - many of non-zero amplitude, and with a distribution of higher-magnitude values, for instance in genome-wide association studies (GWAS). You can refer to Wikipedia for more details.

1. Quick start

Use the command as follow to draw a manhattan plot, this command will use the default manhattan configuration file 'example/manhattan.conf' and the GWAS result file and **the chromosomes length list file, which defined in the configuration file**, to draw your manhattan plot.

perl sbv.pl manhattan [options] <input file>

2. Input data file

There are two input data files, GWAS result file and the chromosomes length file.

The GWAS result file contains 4 fields,

<SNP name> <chromosome id> <position> <p-value>

The chromosomes length file just has 2 fields,

<chromosome id> <chromosome length>

3. Configuration

Besides the basic attributes, other attributes in the configuration file of manhattan plot must be put in block <manhattan>. The attributes of a manhattan plot are displayed as follow.

3) Public attributes

4) Private personal attributes

file => the GWAS result file

length => the chromosomes length list file, the default is rice's chromosome length file.

fill => the points fill colors of different chromosomes, default is 'red green'.

size => the points size (radius), default is 1.5

title => the main title of the manhattan plot. Default no title.

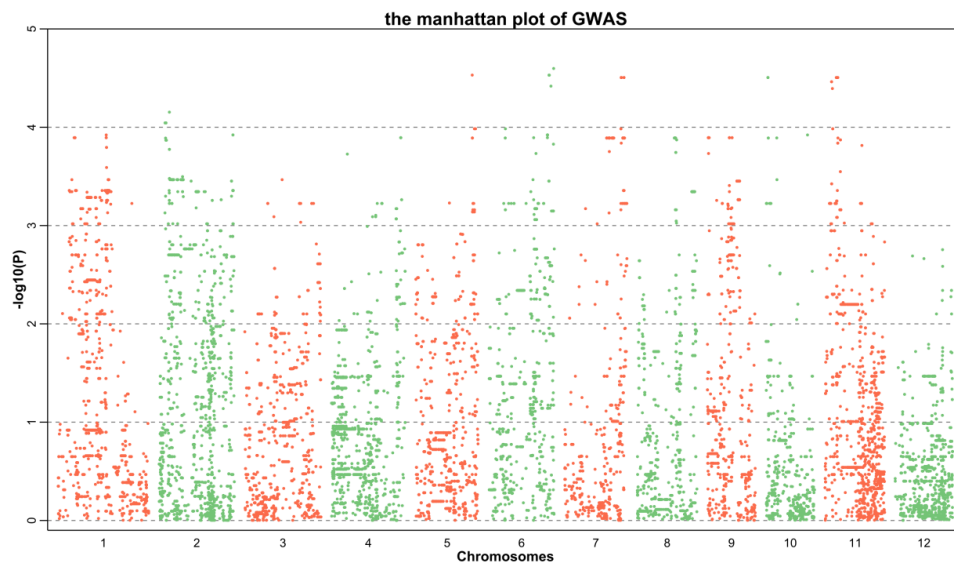
xlab => the x axis title, default is Chromosomes.

ylab => the y axis title, default is -log10(P).

chr_spacing => the spacing length between two chromosomes used to separate chromosomes from each other, default is 0u ('u' means 'bp', 1000000u=1Mbp)

4. Example

The example of manhattan configuration file is 'example/manhattan.conf'. The picture below was created by this configuration file.



FREQ

Use this figure type, you can draw a figure about the amino acid/nucleotide acid frequency in multi aligned protein/dna sequence at each site.

1. Quick start command

perl sbv.pl freq [options] <input file>

2. Input data file

The input sequence data file has two format, plain and fasta. Both format needs the multi sequence must be aligned.

The 'plain' format displayed as follow.

ATGC

GCTA

CCGT

[The 'fasta' format](#) displayed as follow, more details, you can refer to Wikipedia.

>seq1

ATGC
>seq2
TGCA
>seq3
CATC

3. Configuration

The attributes must be put in block '<freq>'.

This private personal attributes of 'freq' contains:

file => the input sequence file

format => the input file formation, plain or fasta, default is plain.

type => the sequence type, dna or protein, default is dna.

start => the sequence start position you want to draw the frequency, default is 1.

end => the sequence end position you want to draw the frequency, default all sequence.

xnames => the x axis ticks is string or not, default yes.

xlab => x axis title, default is 'position'

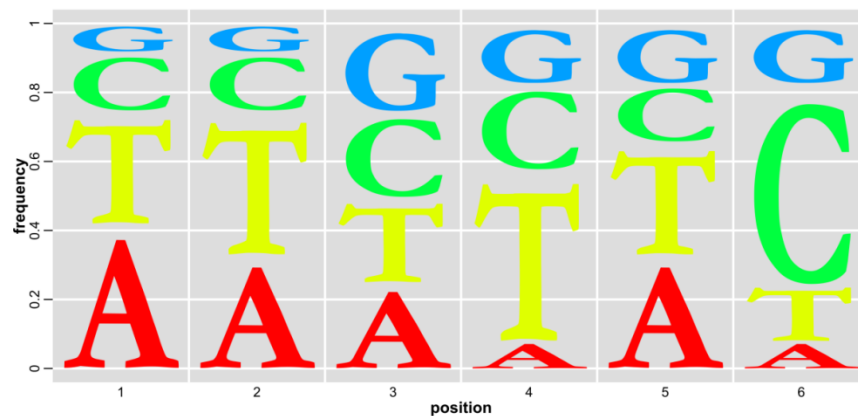
ylab => y axis title, default is 'frequency'

4. Example

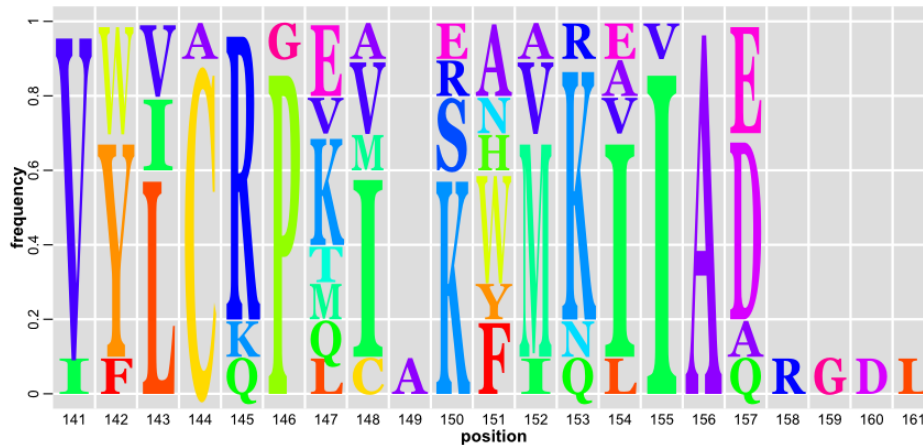
Example configuration file: "exmpale/freq.conf"

Example figure:

Dna sequence map



Protein sequence map



VENN

A [venn diagram](#) is a diagram that shows all possible logical relations between a finite collection of sets. You can refer to Wikipedia for more details.

With this figure type, you can draw a venn diagram with 2-5 sets.

1. Quick start

Use the command as follow to draw a venn diagram, this command will use the default venn configuration file 'example/venn.conf' and the input data file in arguments to draw your venn diagram.

perl sbv.pl venn [options] <input file>

2. Input data file

This file contains no more than five rows and more than one field. The format like this,

<Object> <val1>,<val2> ...

The separator between object and values must be <tab>, the separator between values can be <tab> or comma.

3. Configuration

Besides the basic attributes, other attributes in the configuration file of venn diagram are must be in block <venn>. The attributes of a venn diagram are displayed as follow.

1) Public attributes

2) Private personal attributes

file => input data file

fill => the fill colors used in venn diagram, you can set the colors you want, or you can select the color styles, 'CMG_Lee', 'rainbow', 'none'. 'CMG_Lee' is the styles used in the venn diagram in Wikipedia, 'rainbow' is used the rainbow colors generated by the soft auto, 'none' means not set the fill colors. Default is 'CMG_Lee'.

show_label => show the object label or not (Boolean type), default yes.

show_logical_label => show the logical label or not (Boolean type), default no.

print_stat_info => print the stat result to STDERR/file or not (Boolean type), default no.

stat_file => print the stat result to which file, default is STDERR, effective with

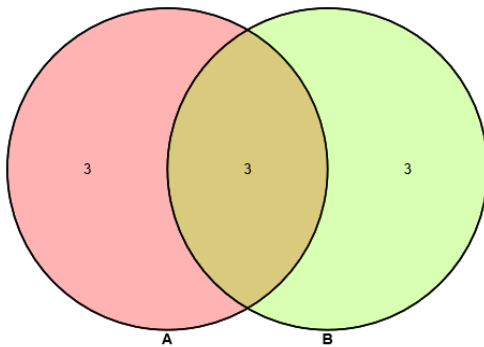
print_stat_info as true.

model => set the venn figure diagram, now just for 2 or 5 sets venn diagram. For 2 sets venn diagram, it can be "horizontal" or "vertical", default is "horizontal". For 5 sets it can be "ellipse" or "Branko" which will draw 5 ellipses, otherwise it will draw 5 pear-shaped figure, default will draw 5 pear-shaped figure.

4. Example

The venn configuration example file is 'example/pie.conf', the 2-5 sets venn figures are displayed as follow respectively (the fill color is rainbow).

For 2 sets

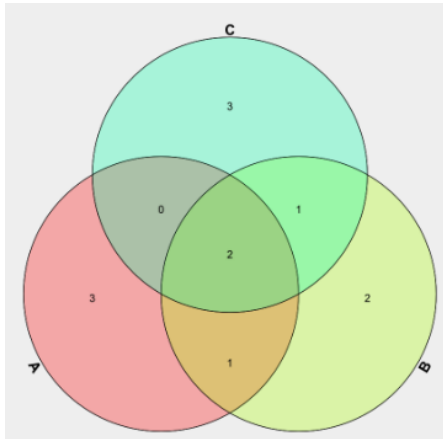


model = "horizontal" or model = "h" (default)

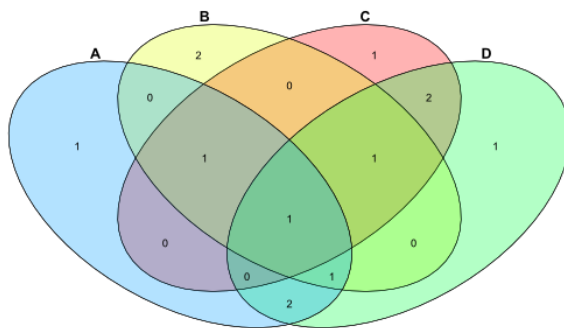


model = "vertical" or model = "v"

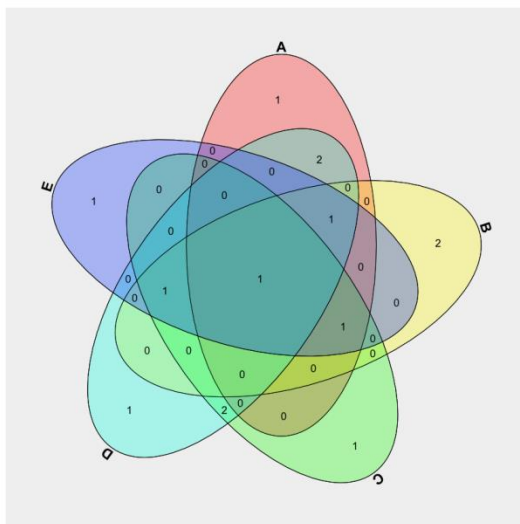
For 3 sets



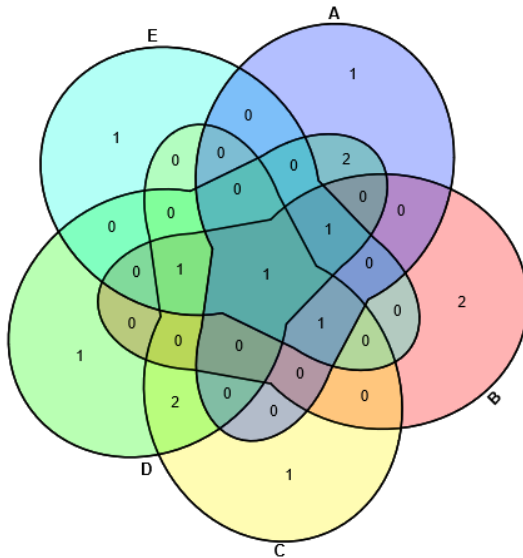
For 4 sets



For 5 sets



model = "ellipse" or model = "Branko"



model = "pear" (default)

LASV

Lasv is the abbreviation of Local Alignment Search Visualization. Lasv chart can help you to observe the alignment result. With this figure type, you can draw two type lasv char, 'simple' or 'detail' which will be described below.

1. Quick start

perl sbv.pl lasv [options] <input data file>

The default configuration file is 'example/lasv.conf'

2. Input data file

This file is a tabular file, contains 9 field, showed as below,

**<query_name> <query_length> <query_start> <query_end> <subject_name>
<subject_length> <subject_start> <subject_end> <identity>**

Note: There is a script ('src/ searchDeal.pl') which can be used to turn other format, like blast, to this data format ('sbv'). The detail usage information is in the document of this script.

3. Configuration

The attributes of lasv is contained by block 'lasv'. The personal attributes of lasv are displayed as follow.

file => the input data file

key => the key bar is 'query' or 'subject', default is 'query'

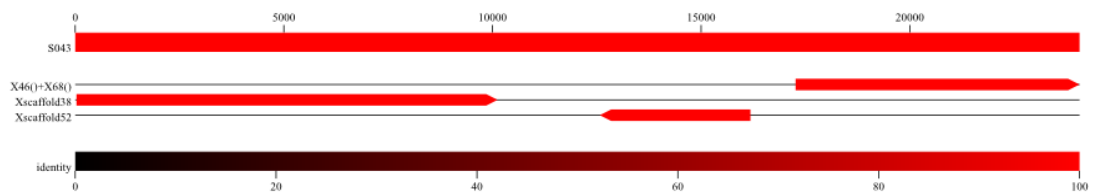
graph => the lasv chart type, 'simple' or 'detail', default is 'simple'

draw_first => just draw the first object lasv chart, this attributes is useful when you just want to take a look at this type figure.

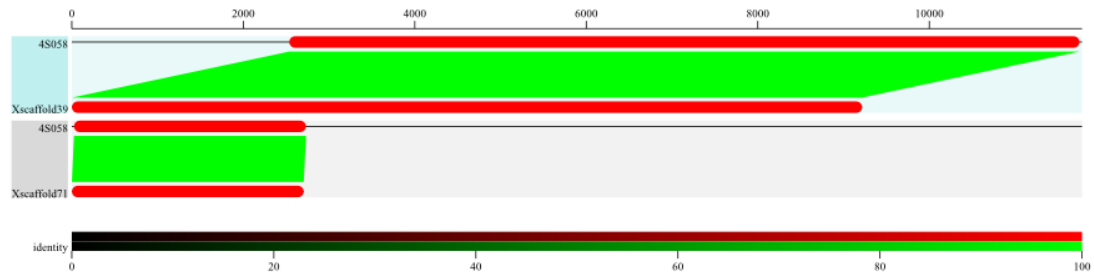
4. Example

The lasv configuration example file is 'example/lasv.conf', the simple and detail lasv chart are displayed as follow.

Simple lasv chart



Detail lasv chart



TREE

A [phylogenetic tree](#) or evolutionary tree is a branching diagram or "tree" showing the inferred evolutionary relationships among various biological species or other entities based upon similarities and differences in their physical and/or genetic characteristics. The taxa joined together in the tree are implied to have descended from a common ancestor. You can refer to Wikipedia for more details.

With this figure type 'tree', you can draw 3 type tree maps, 'normal tree', 'circular tree' and 'unrooted tree'. You can add your own stat data to the tree figure, like 'boxplot', barplot, binary plot, domain etc. You can define your labels text and the colors of the clade, range and label. The more details will be described by the configuration file.

Note: this figure type is designed like [iTOL](#).

1. Quick start

perl sbv.pl tree [-conf tree.conf] [options] <input file>

The default configuration file is 'example/tree.conf'.

The default configuration file is a demo for normal tree,

'example/circular_tree.conf' is a demo for circular tree,

'example/unrooted_tree.conf' is a demo for unrooted tree.

2. Input data file

In sbv, the input tree file will be read by BioPerl, so you need to install BioPerl module. The supported tree format are displayed as follow, you can refer to [Bio::TreeIO](#) for more details.

newick	Newick tree format
nexus	Nexus tree format
nhx	NHX tree format

svggraph	SVG graphical representation of tree
tabtree	ASCII text representation of tree
lintree	lintree output format

3. Configuration

The attributes must be put in block <tree>.

The private personal attributes are described below.

file => the input tree file

format => the input tree file format, default is nhx.

model => the tree figure type, 'normal', 'circular' and 'unrooted', default is 'normal'.

3.1. Attributes exist in all 3 type trees

unit => set the evolutionary distance unit, default is 0.01

align => align the branch or not, (Boolean type), default yes (extend the branch lines with dotted line).

ignore_branch_length => ignore the branch length or not (Boolean type), default not.

show_branch_length => show the branch length or not (Boolean type), default not.

tree_width => fix the tree width, in this case the label size of leaves will be adjust auto.

3.2. Attribute just exists in circular and normal trees.

outgroup => reset the root for the tree.

scale_no_offset => the scale with no offset in the figure, default offset.

3.3. Attributes just exist in circular and unrooted trees.

angle => set the sum angle for the tree, default is 360.

rotation => the rotation angle for the tree, default is 0 (the 0 o clock direction).

3.4. Attribute just exists in circular tree.

radius => set the start radius far from the center point, default is 0.

3.5. <bootstrap> block

The bootstrap value of the tree can be displayed in one of the three ways which are displayed as follow and the attributes must be put in block <bootstrap>.

a. text => show the bootstrap value as text or not, (Boolean type), default yes.

threshold => set the bootstrap threshold.

b. symbol => show the bootstrap value by symbol, defined like below

symbol = 50,60,* 60,70,@ 70,80,\# 80,100,\$

c. color => show the bootstrap value by branch colors, define like below

color = 50,60,0ff 60,70,0f0 70,80,00f 80,100,f00

3.6. <definition> block

This block is used to redefine the label text, label text colors, range colors, clade colors and HGT.

leaf => redefine the leaves label text, the value is a file which defines the leaves label texts. The file format is like below.

<node id> <label> [theme=]***

One line per leaf, two or three fields per line, tab separated.

First file defines the leaf id.

Second file defines the label you want to show in the figure.

Third file defines the leaf label text font style, is optional.

color => defining color ranges, clade and label colors. The value of this attribute is a file which defines the color information as follow.

<node_id> <type> <color> [label]

Three or four fields per line, tab separated.

Node_id can be either leaf ID or internal node ID.

Type should be either **range**, **label** or **clade**.

Label is optional, and just for color range.

cover => the color range covers full clades ('full') or labels only ('label'), default is 'full'.

hgt => displaying horizontal gene transfers (HGTs) on the tree. The value of this attribute is a file which defines the HGTs information like as follow.

<source_node_id> <destination_node_id> <color>

One line per HGT, three fields per line, tab separated.

Both **source_node_id** and **destination_node_id** can be either leaf IDs or internal node IDs.

3.7. <datasets> block

Use this block, you can add different datasets on the tree. At the same time, you can add many datasets on the tree, but each dataset should be in a separated plain text file. All fields should be using the tab as separator. First file on each line should be a node ID (or an ID pair), followed by the actual data value(s). Each dataset is put in block <dataset>.

This block is just for circular tree and normal tree.

The attributes of datasets, which can be inherited by each dataset, are displayed as follow.

type => the type of the dataset, must be defined for each dataset.

color => the colors used in dataset, default is '#f00'.

width => the width of the dataset chart, default is 20.

height => the height of the dataset chart, 0-1, default is 1 (all space of each leaf).

show_axis => show the axis ticks or not. (Boolean type), default yes.

The attributes of all dataset types:

All attributes of datasets, (type, color, width, height, show_axis)

file => the dataset file

show => show this dataset or not, (Boolean type), default yes.

The other attributes of data will be described separately for each dataset type.

'The tree' figure type can display the following types of data.

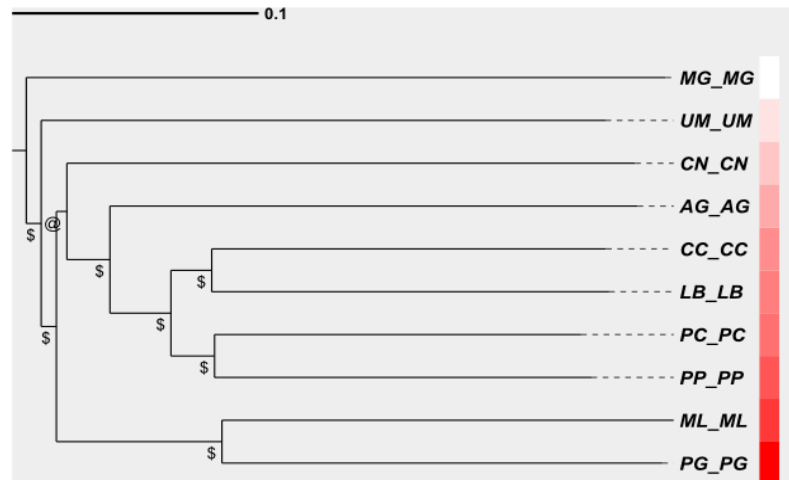
3.7.1. marker

gradient => use the gradient color to fill the marker rectangle or not, default not.

The dataset file of marker has two fields.

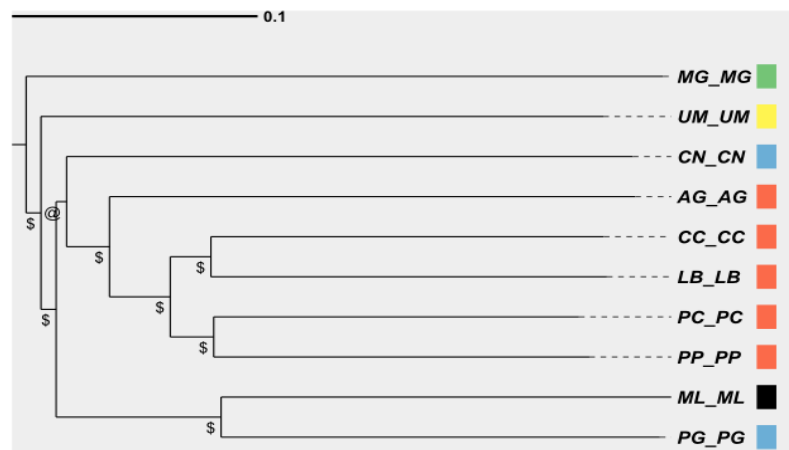
If gradient is true, the second field is a number value

<node id> <number>



If gradient is false, the second field is a color value

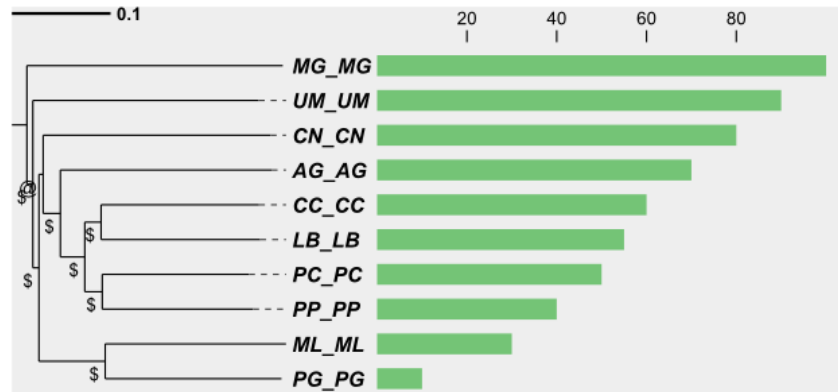
<node id> <color>



3.7.2. simple_bar

the dataset file of 'simple_bar' has two fields, the second field is a number value.

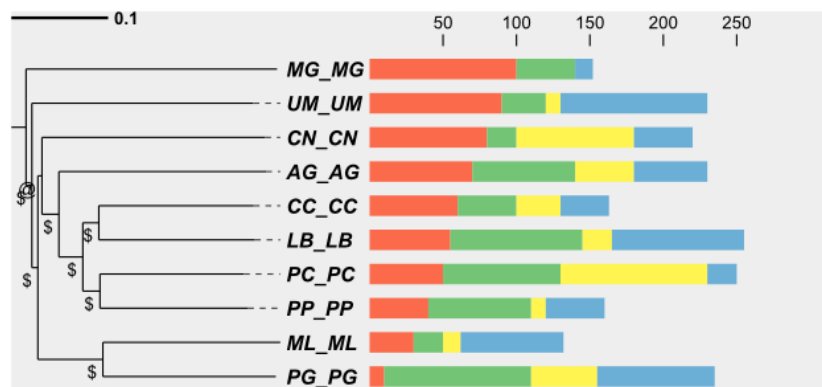
<node_id> <value>



3.7.3. bar

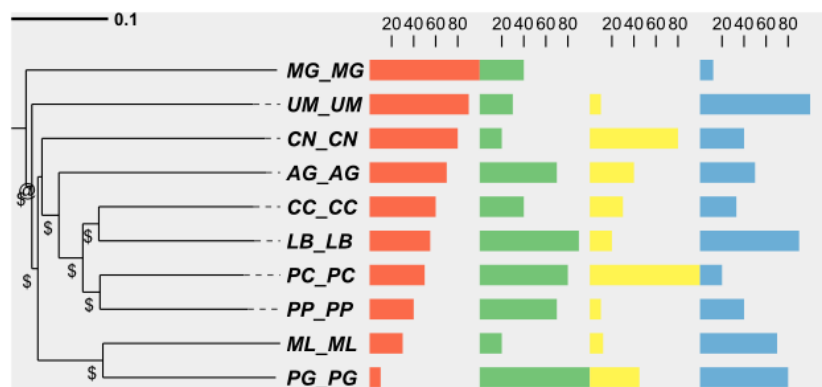
The dataset file of 'bar' has at least two fields, the second to last fields are number values.

`<node_id> <val1> <val2> ...`



3.7.4. multi_bar

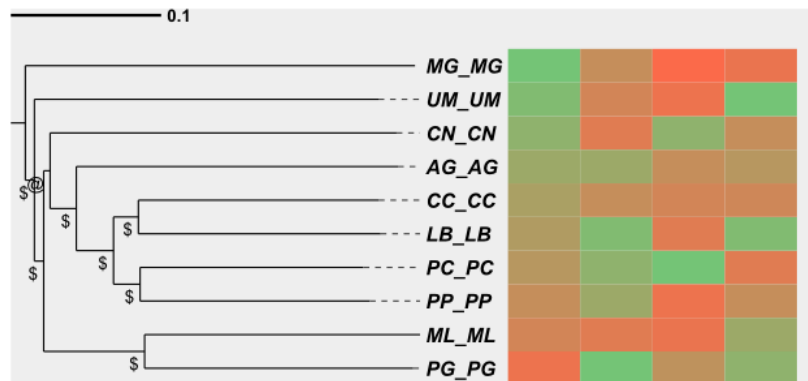
The dataset file of 'multi_bar' is same as the 'bar'. But the group data are aligned separately.



3.7.5. heatmap

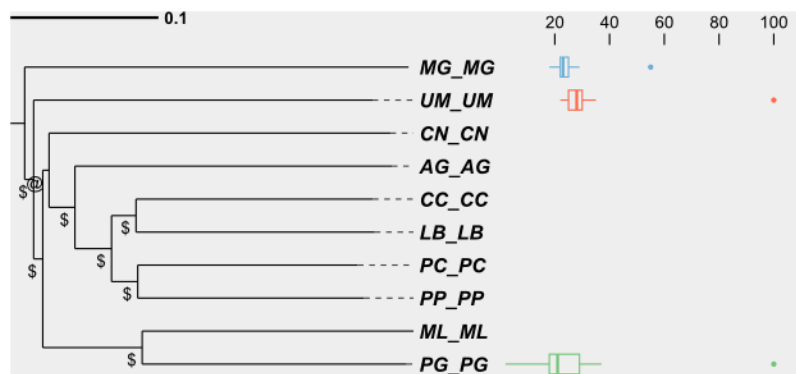
The format of dataset file is same as the 'bar'. The start color and the stop color are both defined by 'color' attribute.

color = strat_color stop_color



3.7.6. boxplot

The dataset of 'boxplot' has two or more fields, and each line could have different fields.

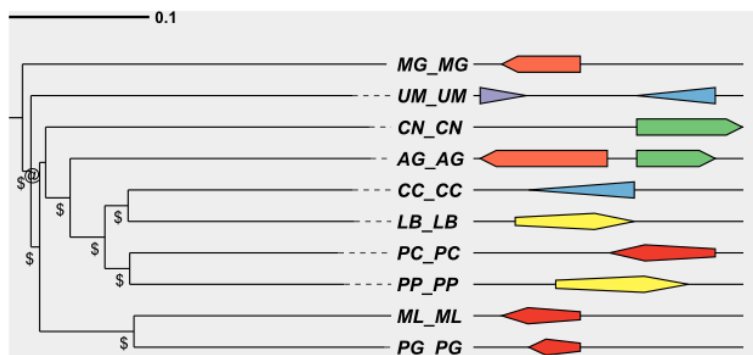


3.7.7. modify

format => the format of the modify, 'domain' or 'symbol'

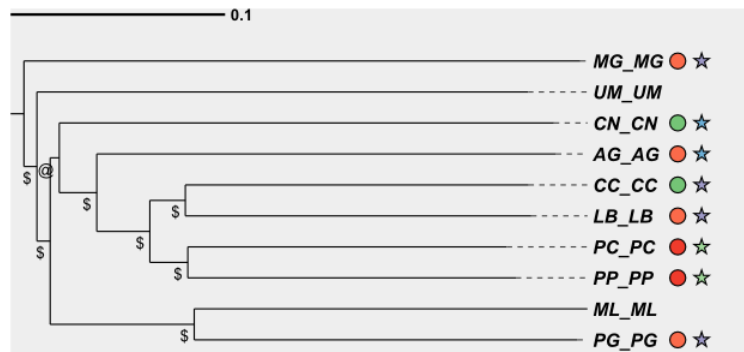
The domain modify file format

<node_id> <length> <start> <end> <symbol_id> <stroke_color> <fill>



The symbol modify file format

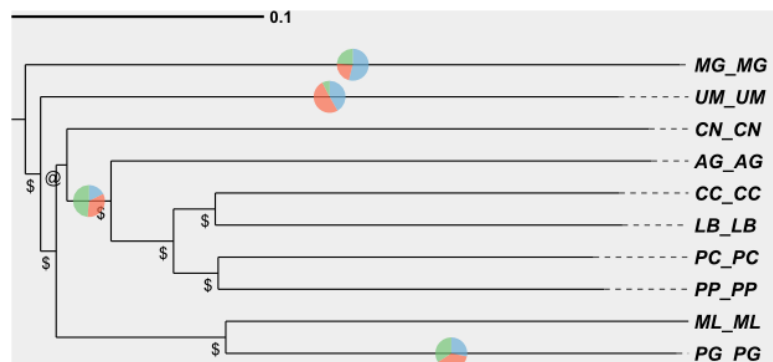
`<node_id> <symbol_id> <stroke_color> <fill>`



Note: For setting the symbol ID, you can refer to [the accessory](#) for more information.

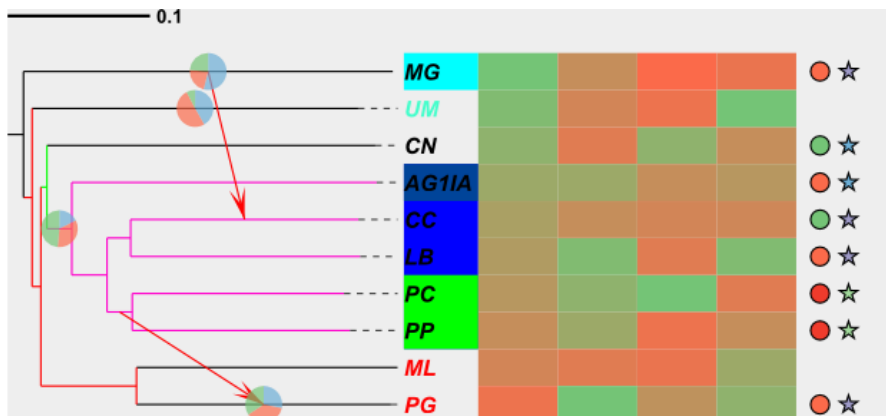
3.7.8. pie

The dataset format is same as the boxplot.

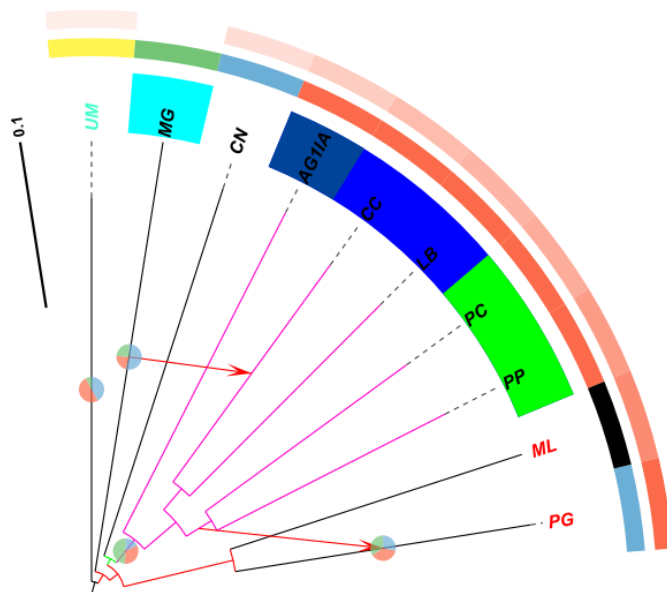


4. Example

Demo of normal tree

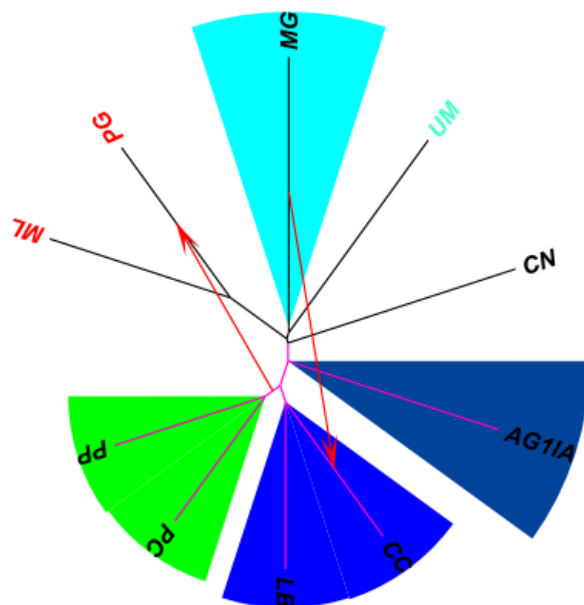


Demo of circular tree



Demo of unrooted tree

— 0.1



TAXTREE

Use this figure type, you can draw a taxonomy tree and a pie figure on each nodes.

1 Quick start command

perl sbv.pl taxtree [-conf taxtree.conf] [options] <input file>

The default configuration file is 'example/taxtree.conf'

2 Input Data files

There are two input file which can be created by script 'src/tax2nhx.pl', one is a tree file, another is a stat information tabular file for adding pie figure on each nodes.

The raw file which can be deal by 'src/tax2nhx.pl' contains header information and at least 3 fields, displayed as follow. The first filed must be the taxonomy information which separated by ';'; the last field name must be fixed as "total_percent" whose value will be displayed under the taxonomy name; others are all samples values which will be used to draw pie figure, and its header names will be displayed in legend region of the taxonomy tree figure. The file 'data/taxtree/all.tsv' is an example.

<taxonomy> <sample1> <sample2> ... <sampleN> <total_percent>

Or you can prepare the two input files by yourself, like 'data/taxtree/temp.nhx' and 'data/taxtree/temp.percent', but we suggest you to use the script to create them.

3 Configuration

The attributes must be put in block <taxtree>.

The private personal attributes are described below.

file => The input tree file

percent => The input stat tabular file

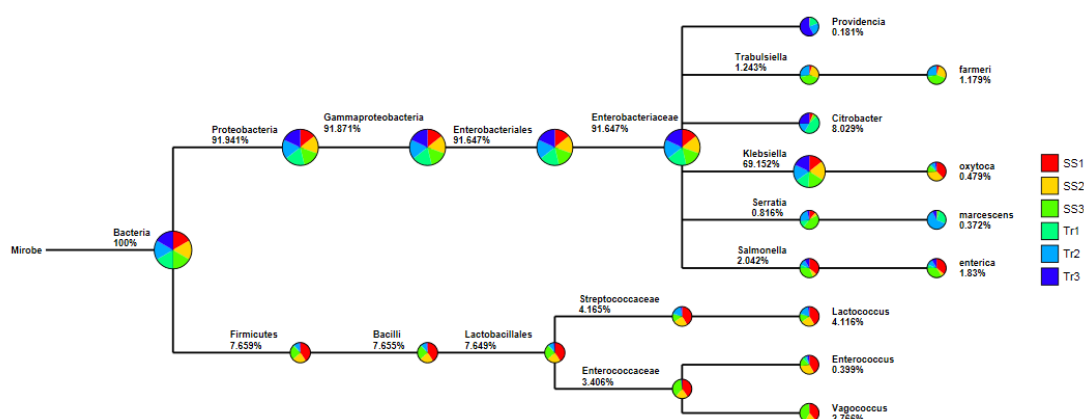
radius => The maximum radius of pie.

colors => The sample colors definition file, contains two fields. **<sample_name><color>**

If you want to change the font styles, you can define yourself styles configuration file follow "styles/styles.taxtree.conf".

If you want to change the legend location and styles, you can define yourself legend configuration file follow "example/legend_taxtree.conf".

4 Example



HEATMAP

Use this figure type, you can draw a complex heatmap figure which can dress up the two Hierarchical clustering trees, and can add many elements after the heatmap.

1. Quick start command

perl sbv.pl heatmap [-conf heatmap.conf] [options] <input file>

The default configuration file is 'example/heatmap.conf'

2. Input Data file

The input file is a tabular file which contains the expression information of many samples, the row names are the gene IDs, the column names are the samples names. You can refer to the example file '\$SBV_HOME/data/heatmap/test.fpkm'.

3. Configuration

The attributes must be put in block <heatmap>.

The private personal attributes are described below.

file => the input expression matrix tabular file.

header => the header line is column name or not, default is 1.

rownames => the rownames is defined in file or not, default is 1.

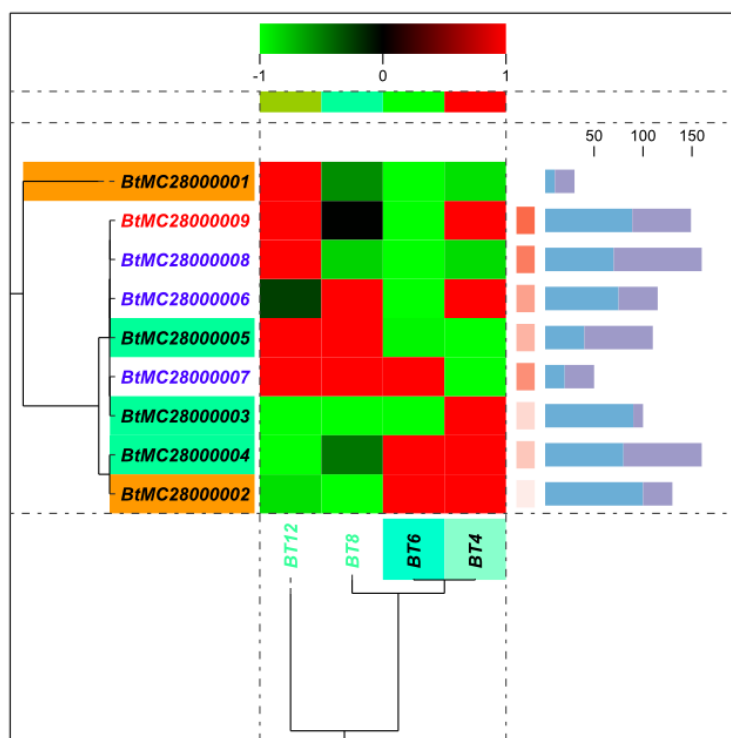
show_split_line => show the split line or not, default is 1.

colors => the heatmap color range, default is 'f00 000 0f0'.

scale => normalize the matrix data, 'none', 'row', 'column' or 'all', default is 'none'

<horizontal> and <vertical block> block, these two block is used to dress up the tree and add elements on the tree which is the same as <tree> block in 'tree' figure type.

4. Example



KARYO

Use this figure type, you can draw a karyotype figure. A karyotype figure can display in two model, 'normal' and 'circular'.

1. Quick start command

perl sbv.pl karyo [-conf karyo.conf] [options] <input file>

the default configuration file is 'example/karyo.conf'

The default configuration file is a demo for normal karyo figure,
'example/circular_karyo.conf' is a demo for circular karyo figure,

2. Input data file

The input data file contains at least five or six fields as follow, the last field is optional.

<chr id> <label> <start pos> <end pos> <color> [attributes]

The separator between fields must be <tab>. The attributes of chromosome are defined at format as follow,

attr1=val1;attr2=val2;...

The attributes contains:

radius => the chromosome radius

thickness => the chromosome thickness

3. Configuration

The attributes must be put in block <tree>.

The private personal attributes are described below .

file => the input karyotype data file.

model => the karyo figure type, 'normal', 'vertical' or 'circular', default is 'normal'. In the 'normal' model, the chromosome ideograms are horizontal. In the 'vertical' model, the chromosome ideograms are vertical, which is generated by rotated 90 degree with normal model image.

start => the chromosome start position, default is 0.5r, just for 'normal' and 'vertical' model.

rotation => the rotation angle of karyo figure, default is 0 (the 0 o clock direction), just for 'circular' model.

radius => the default radius for chromosomes, just for 'circular' model, must be defined.

3.1 <ideogram> block

Set the chromosome ideograms in the image.

You can refer to [CIRCOS ideogram tutorials](#) for more detail information.

3.1.1 show => show chromosomes, ticks and chromosome labels or not, default is yes.

3.1.2 thickness => the chromosome ideograms thickness

3.1.3 show_chromosomes_default => show all chromosomes in the image

3.1.4 chromosomes => filtering the chromosomes showed in the image, effective with turn off the 'show_chromosomes_default'.

3.1.5 chromosomes_order => set the chromosomes displayed order.

3.1.6 chromosomes_reverse => set the reverse show chromosomes.

3.1.7 chromosomes_color => fill the chromosomes color or not, default yes.

3.1.8 chromosomes_rounded_ends => the bar ends is rounded or not, just for 'normal' and 'vertical' model, default is yes.

3.1.9 show_label => show chromosome labels or not, default is yes.

3.1.10 label_with_tag => show the label with tag or by id, default is tag.

3.1.11 `label_parallel` => the chromosome label parallels ideogram or not, default is yes.

3.2 `<ticks>` block and `<tick>` blocks

Set the chromosome ticks in the image, which can contain any number of `<tick>` blocks, each defining ticks with a different spacing.

3.2.1 `spacing` => the distance between the ticks in this set, must be defined.

3.2.2 `chromosomes` => optional list of ideograms on which ticks are drawn or are suppressed

3.2.3 `orientation` => set the tick orientation, 'up' and 'down' is optional for 'normal' and 'vertical' model, default is 'up'. 'inner' and 'outer' is optional for 'circular' model, default is 'outer'.

3.2.4 `offset` => set the offset distance between ticks and ideogram, default is 0.

3.2.5 `show_true_ticks` => show the raw true ticks or relative ticks started with 0, default is show the raw true ticks.

3.2.6 `label_multiplier` => the label multiplier is the constant used to multiply the tick value to obtain the tick label. default is 1.

3.2.7 `unit_label` => paste this value after each tick label, default is null.

3.2.8 `thickness` => the thickness of tick line, default is 1.

3.2.9 `size` => the length of tick line, default is 8.

3.2.10 `show_label` => show tick labels or not, default no.

3.2.11 `label_theme` => the tick label text style theme.

3.3 `<highlights>` block and `<highlight>` blocks

3.3.1 `file` => highlights input data file, must be set.

3.3.2 `stroke_width` => highlight graph stroke width, default is 1.

3.3.3 `ideogram` => the highlights is on chromosome ideograms or not, default is no.

3.3.4 `loc0/loc1` => the up and down location of the plot track, for 'normal' and 'vertical' model.

3.3.5 `r0/r1` => the inner and outer radius of the plot track, for 'circular' model

3.3.6 `shape` => the highlight graph symbol shape, refer to 'Symbol reference list' to pick your shape, just for 'normal' and 'vertical' model, default is 0.

3.3.7 `fill` => the highlight fill color.

3.3.8 `color` => the highlight stroke color.

3.4 `<links>` block and `<link>` blocks

3.4.1 `file` => the link input data file, must be set.

3.4.2 `fill` => the link fill color, default is '#000'

3.4.3 `color` => the link stroke color, default is '#000'

3.4.4 `opacity` => the link fill opacity, default is 1.

3.4.5 `stroke_width` => the link stroke width, default is 1.

3.4.6 `loc0/loc1` => the up and down location of the plot track, for 'normal' and 'vertical' model.

3.4.7 `r0/r1` => the inner and outer radius of the plot track, for 'circular' model

3.5 <plots> block and <plot> blocks

3.5.1 type

3.5.1.1 line

min => the minimum value of the data

max => the maximum value of the data

color => the line stroke color

stroke_width => the line stroke width

show_tick_line => show the tick line, default not

show_tick_label => show the tick label, default not

3.5.1.2 scatter

min => the minimum value of the data

max => the maximum value of the data

shape => the scatter points shape.

radius => the scatter points radius size

color => the scatter points fill color

stroke_width => the scatter points stroke width

show_tick_line => show the tick line, default not

show_tick_label => show the tick label, default not

3.5.1.3 histogram

min => the minimum value of the data

max => the maximum value of the data

fill => the histogram bar fill color

color => stroke color

stroke_width => stroke width

show_tick_line => show the tick line, default not

show_tick_label => show the tick label, default not

3.5.1.4 heatmap

fill0 => the start fill color

fill1 => the end fill color

color => stroke color

stroke_width => stroke width

3.5.1.5 text

theme => text style theme

show_links => show text label links or not, default yes

link_length => the link line length size

link_color => the link line color, default is '#000'

link_thickness => the link line stroke width, default is 1.

fill => the text label fill color, default is '#000'

ideogram_highlights => the corresponding chromosome ideogram region
draw highlight line or not, default no.

snuggle_layer => the maximum snuggle text layer size, default is 1.

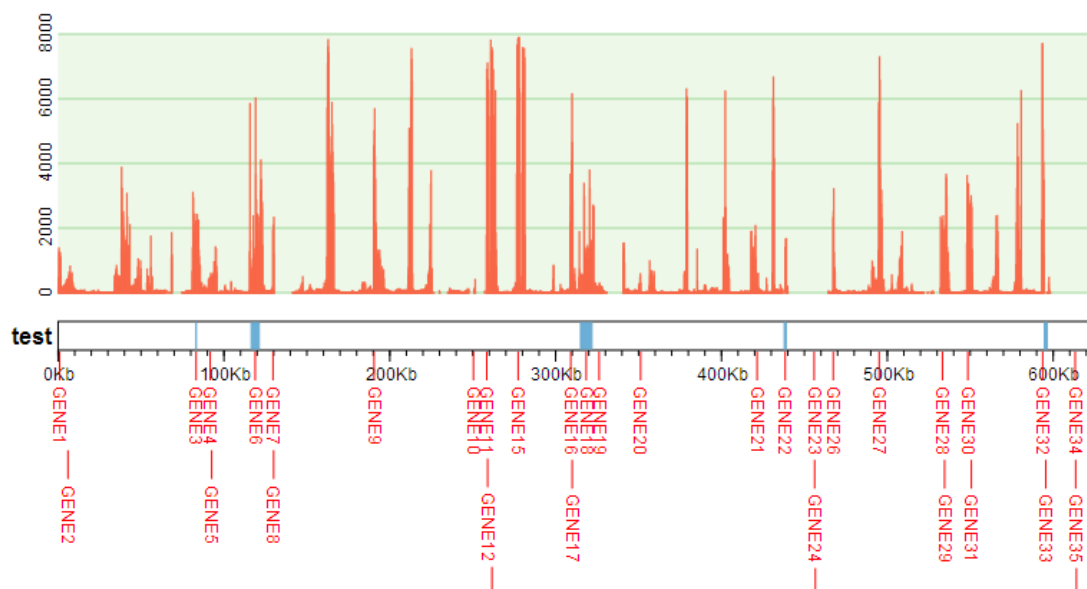
3.5.2 file => the plot input data file

3.5.3 loc0/loc1 => the up and down location of the plot track, for 'normal' and 'vertical' model.

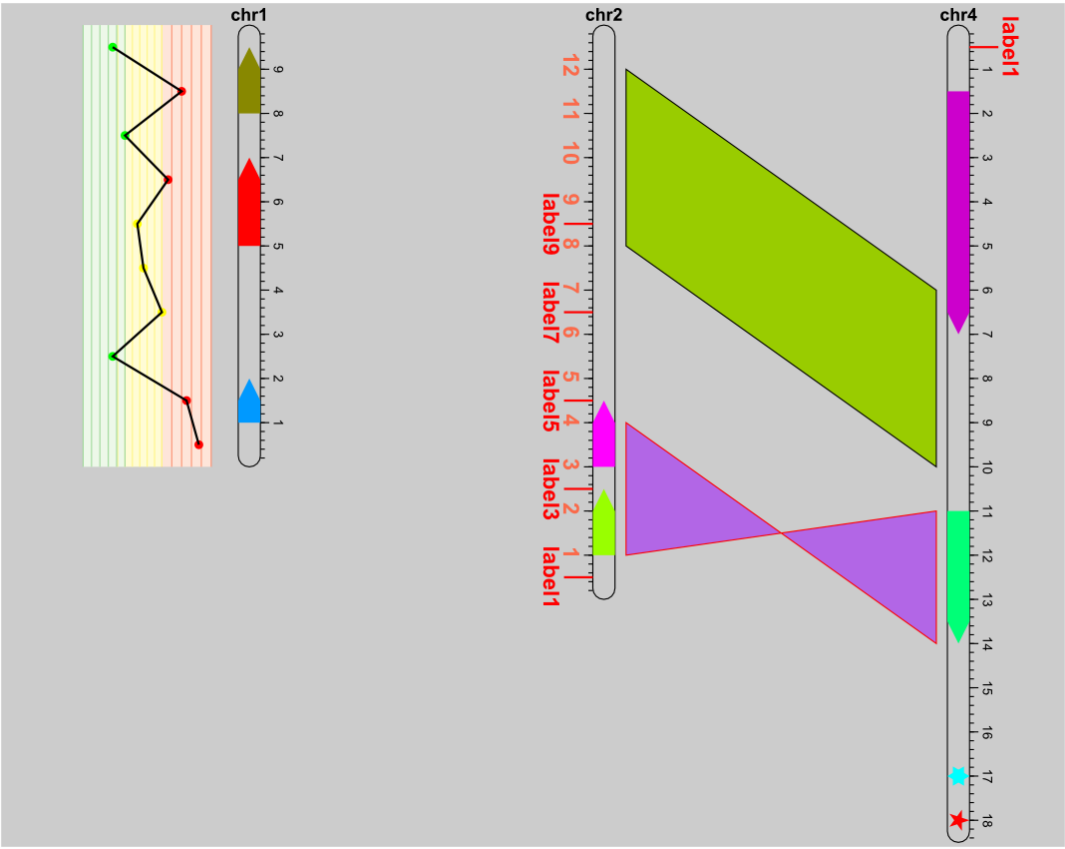
- 3.5.4 $r0/r1 \Rightarrow$ the inner and outer radius of the plot track, for 'circular' model
- 3.5.5 $z \Rightarrow$ the order of the plot to add on the image
- 3.5.6 `<backgrounds>` block and `<background>` blocks
 - $y0 \Rightarrow$ the start position of the plot set, default is 0.
 - $y1 \Rightarrow$ the end position of the plot set, default is 1.
 - $color \Rightarrow$ the background color
- 3.5.7 `<axes>` block and `<axis>` blocks
 - $y0 \Rightarrow$ the start position of the plot set, default is 0.
 - $y1 \Rightarrow$ the end position of the plot set, default is 1.
 - $spacing \Rightarrow$ the axis spacing size
 - $color \Rightarrow$ the axis line color

4. Example

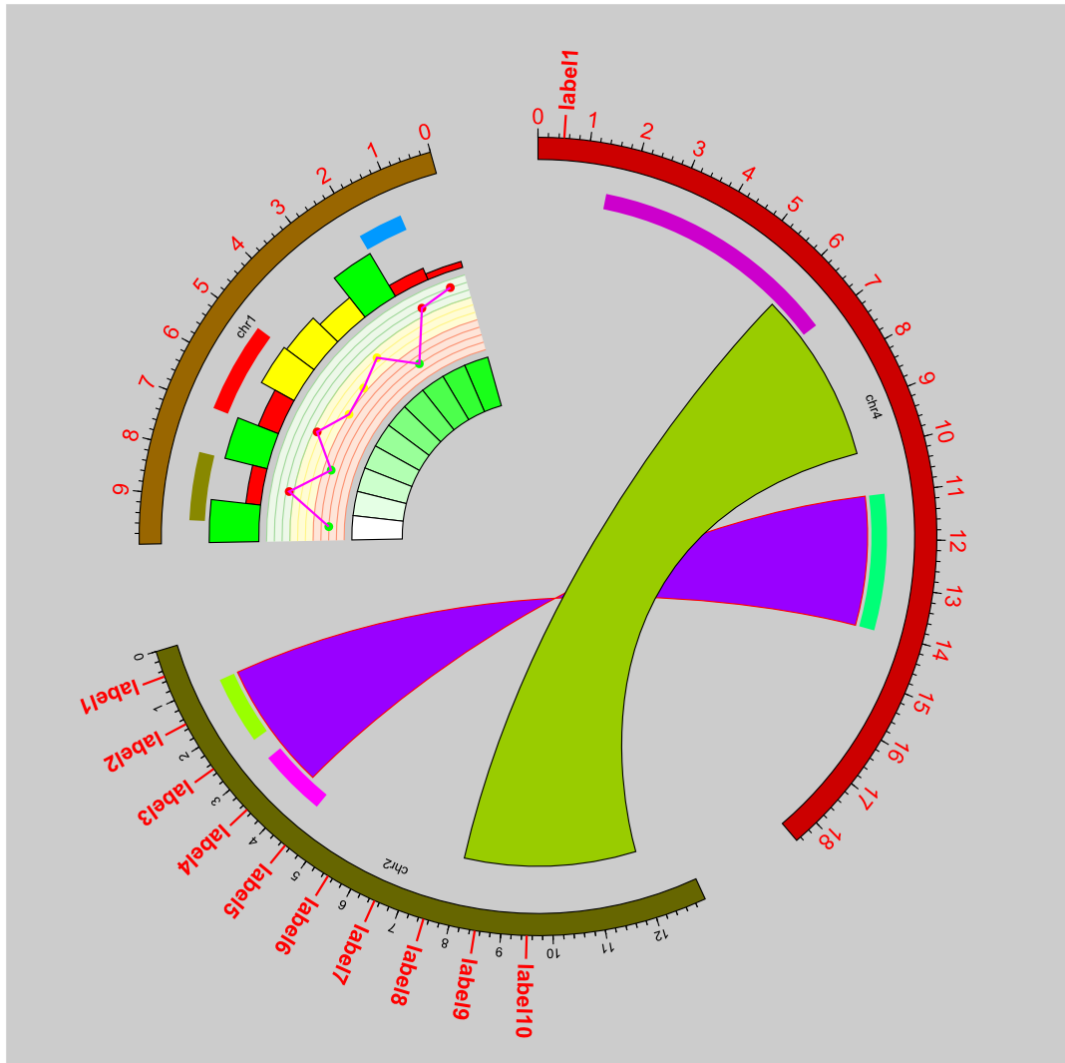
Demo of normal karyo figure



Demo of vertical karyo figure



Demo of circular karyo figure



HCGD

Use this figure type, you can draw a human chromosomes G-banding diagram (hcgd) in three models (Normal, NCBI, Ensembl), or draw other organism chromosomes location diagram in normal model, similar as the karyo figure.

1. Quick start command

perl sbv.pl hcgd [-conf hcgd.conf] [options] <input file>

The default configuration file is 'example/hcgd.conf'

2. Input data file

The default input file is "data/hcgd/cytoBand.txt" which downloaded from UCSC, contains 5 fields, displayed as below.

<chr_id> <start> <end> <banding_name> <color>

3. Configuration

The attributes must be put in block '<hcgd>'.

The private personal attributes are described below.

file => The input banding file.

model => the figure model, 'normal', 'NCBI' or 'Ensembl', just for human. Default is 'normal'.

offset => the chromosome offset position, default is 0.5r.

chromosomes_order => set the chromosomes displayed order

chromosomes_show =>

chr_rounded_ratio => chromosome bar rounded ratio, rounded radius / thickness, must be in 0-0.5

col_chr_number => chromosome number each row

row_chr_spacing => the spacing between rows

dense => show the chrs dense, means chrs in the second or later rows will not aligned.

thickness => the chromosome bar thickness

label_theme => the chromosome label text theme

show_ticks => show ticks of chromosomes.

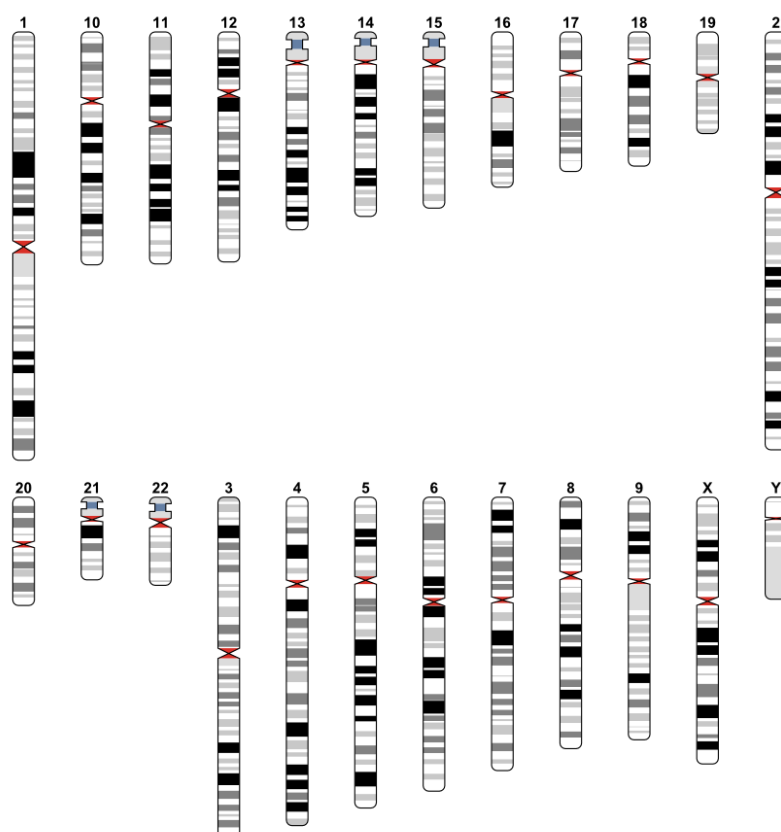
3.1 <ticks> and <tick> block => refer to 'karyo' type

3.2 <highlights> and <highlight> block => refer to 'karyo' type

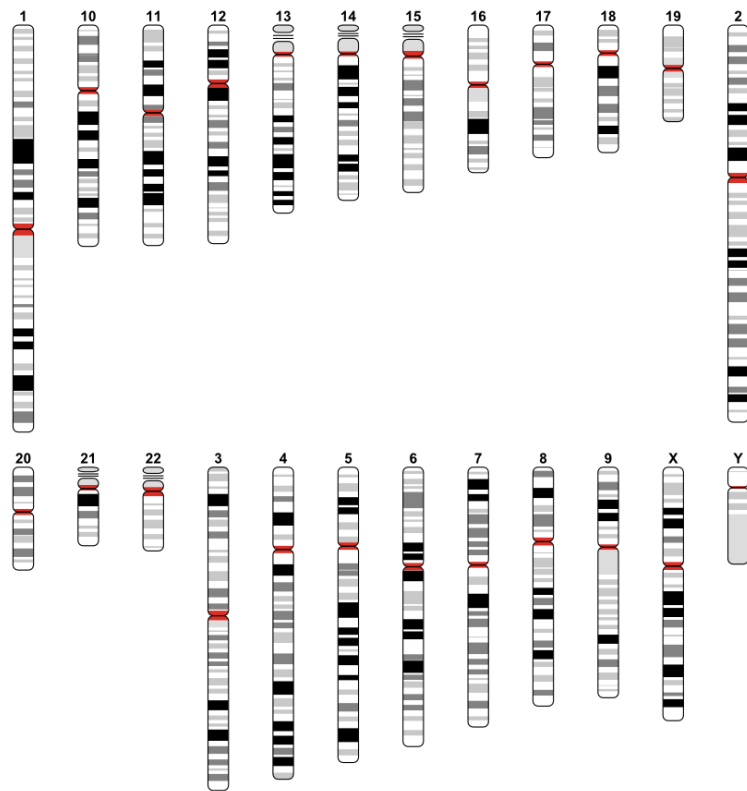
3.3 <plots> and <plot> block => refer to 'karyo' type

4. Example

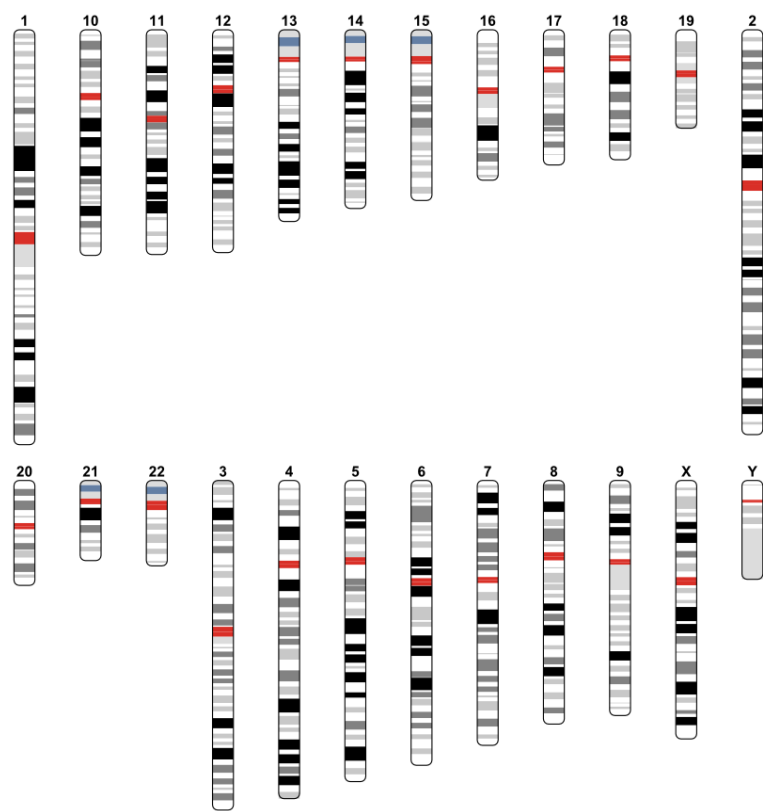
Ensembl model



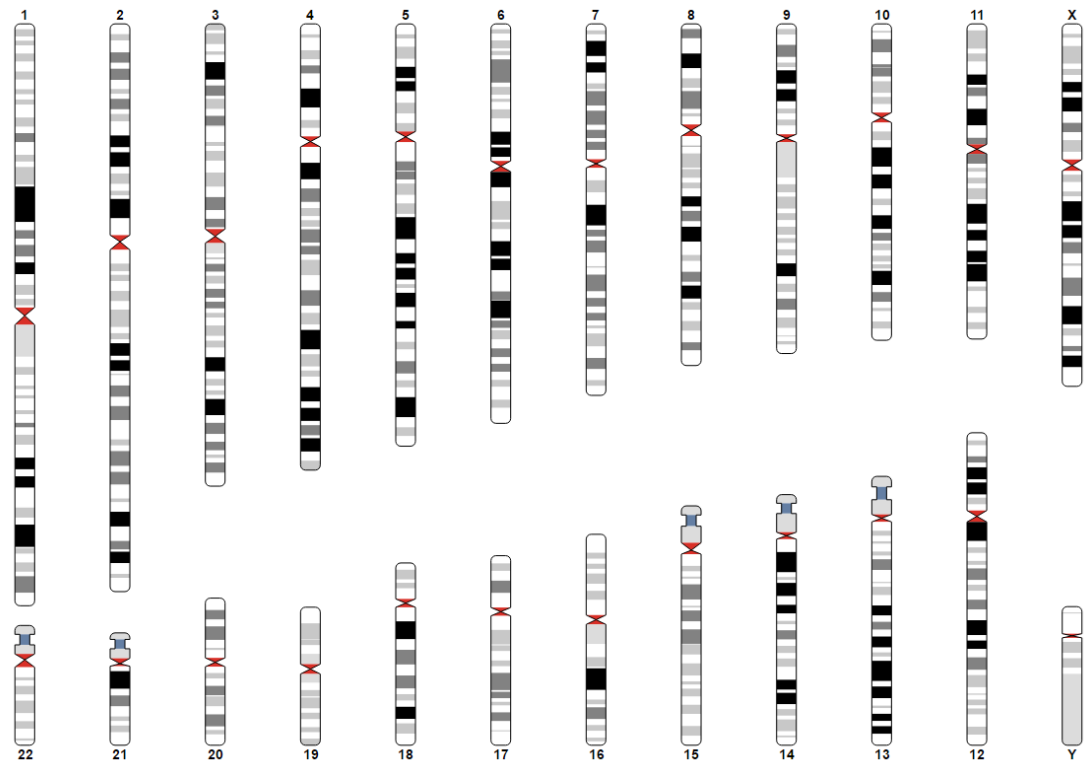
NCBI model



Normal model



Ensembl model with dense set



SDU

Use this figure type, you can draw a “sequence dress up figure.

1. Quick start command

perl sbv.pl sdu [-conf karyo.conf] [options] <input file>

The default configuration file is ‘example/sdu.conf’

2. Input data file

The input sequence data must be saved as ‘**fasta**’ format, and just one sequence in it.

3. Configuration

The attributes must be put in block ‘<sdu>’.

The private personal attributes are described below.

file => the input sequence file, must be defined

nohead => hide the header sequence bar, default is ‘no’

label_theme => the sequence name and length label theme

thickness => the sequence bar thickness, default is 20

header_color => the sequence bar color, default is ‘none’

spacing => the spacing between the sequence bar and its text, default is 60

num => each line sequence base number, default is 80

subnum => each sub block sequence base number, default is 10

line_spacing => the sequence text line spacing, default is 8

theme => the font style of the sequence text, default is "family:Courier New"

3.1. <decorates> block

Use this block you can dress up the sequence in in which style you want. At the same time you can define many styles which must be separated put in block <decorate>.

The attributes of all dataset types:

file => the input file which defined the decorate context, this file cileontains at least 2 fields, the third field is about the attributes and is optional.

<start> <end> [attr1=val1;attr2=val2;...]

Note: if you want to define the font theme, you can use '|' to separate the attributes of theme.

type => the dress up type, 'symbol', 'style' and 'highlight'

'symbol' is used to define the symbol highlight in sequence bar of header, has attributes as follow.

color => the stroke color of the symbol, default is 'none'

fill => the fill color of the symbol, default is 'none'

stroke_width => the stroke width of the symbol, default is 1

shape => the shape of the symbol, default is 0

'style' is used to define the sequence label(text) style, has attributes as follow.

theme => the theme of the label, default is 'fill:red'

underline => add underline or not at the bottom of the label, default is 'no'

'highlight' is used to define the sequence label highlight, has attributes as follow.

fill => the fill color of the highlight, default is 'none'

color => the stroke color of the highlight, default is 'none'

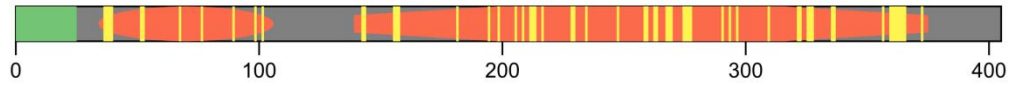
stroke_width => the stroke width of the highlight, default is 1

opacity => the opacity of the highlight, default is 1

4. Example

The sdu configuration example file is 'example/sdu.conf', the sequence dress up chart is displayed as follow.

name: AG1IA_07795
length: 405































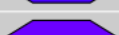
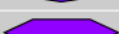

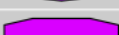
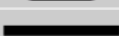
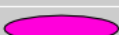
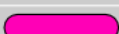




```
1 MRGFTSYLVA ALAALSLLGE NVYATAIPRA EG KRYIVLL KEETNRRTHM AWVDGQRAKT
61 TAGSLSVLSV TSSYNALNGY TAHLSDDRVA QLSKSPDVAM VVEDQRCGY RGLKQTDAPW
121 GISRVSRKTK LRRGSKVDKL NYVFERKPSG AGVDVYVLDT GVNTQHVDFG GRAGWGATFG
181 PYNDTDGHGH GTHIAGTIAG KRFGVAKAAS IIAVKVLGDD NSGWNSDLVA GINWATORAM
241 STQRPSVISI SIGSPGDAAV DAAVTRAVAL GVHVVVAAGN SNRDAKDFSP ARVPDVITVG
301 ATNIQDGRWV TSASVGSNYG PIVDVFAPGQ DITSAWIGSD TATKRLTGTS MATPHVAGLV
361 AYLLAVEGRR TPANMMTRIK QLAPDRLLSG IPPGTPNEII WNGG
```

PPI *

ACCESSORY

Symbol reference list

Code	Shape	Example
0	rectangle(rect)	
1	circle	
2	diamond	
3	up pointing regular triangle	
4	low pointing regular triangle	
5	cross	
6	error	
7	cross and error	
8	rect and cross	
9	rect and error	
10	email	
11	circle and cross	
12	circle and error	
13	diamond and cross	
14	3 and 4	
15	five points star	
16	six points star	
17	horizontal line	
18	horizontal line and a point	
19	boxplot	
20	vertical line	
21	vertical line and a point	
22	box bar	
23	err bar	
24	left pointing pencil	
25	right pointing pencil	
26	right pointing triangle	
27	left pointing triangle	
28	right pointing pentagram	
29	left pointing pentagram	
30	up pointing pentagram	
31	down pointing pentagram	
32	horizontal hexagon	
33	vertical hexagon	
34	octagon	
35	1/3 height rect	
36	ellipse	
37	rounded rect	
40	left pointing arrow	
41	right pointing arrow	