Visualizing genomic rearrangements using the R package *ReConPlot*

Jose Espejo Valle-Inclán and Isidro Cortes-Ciriano*

12/09/2023

Complex genomic rearrangements in cancer

Complex genome aberrations are pervasive across diverse cancer types, and are often implicated in tumour development and therapeutic resistance. The study of genomic aberrations in cancer has been facilitated by the advent of next generation sequencing. Changes in the structure and number of chromosomes is indicated by the presence of somatic copy number profiles and structural variants (SVs).

Visualization of SVs and copy number aberrations is often required to identify the genomic instability mechanisms underpinning the observed patterns and to interpret the functional consequences of genomic alterations, such as the loss of tumour suppressors or the amplification of oncogenes. Currently, there exist several types of visualizations to study genomic rearrangements. These include Circos plots, in which the genome is displayed as a circle and somatic alterations are shown in concentric circular tracks, and 2-dimensional genomic rearrangement profiles, in which the copy number profile and SVs mapping to one or multiple chromosomes are displayed side by side. Whereas no single visualization can encompass the complexity of complex genomic aberrations observed in cancer, genomic rearrangement profiles are very useful to interpret cancer genomics data and report the results of genomics analysis in publications. Yet, libraries for the visualization of genomic rearrangement profiles are lacking, which hampers research and the reproducibility of scientific results.

ReConPlot

Here, we present the R package *ReConPlot*, which provides functionalities for the visualization of copy number profiles and SVs. The package relies on public packages, including the popular ggplot2, thus allowing further customization of the plots.

To visualize genomic rearrangement profiles, *ReConPlot* requires structural variation and copy number data in the commonly used bed format. For SVs, *ReConPlot* uses breakend notation to facilitate compatibility with any SV caller conforming with best practices for reporting SVs in vcf format. While the primary focus of *ReConPlot* is the visualization of complex somatic copy number and SVs in the context of cancer, the functionalities provided can be used to visualize genomics data from non-cancer samples, such as somatic aberrations detected in normal tissues, single-cell whole-genome data or others. At present *ReConPlot* supports the build of the human reference genome GRCh37, GRCh38 and T2T-CHM13, and mouse reference genomes mm10 and mm39.

Examples

In the next section we provide examples of the functionalities of *ReConPlot* and provide best practices to generate high-quality plots. First, we load the package.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------------------------------------------
## v dplyr     1.1.0     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.1     v tibble    3.1.8
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ReConPlot)
```

Next, we load test data provided with the package, which corresponds to a human glioblastoma.

```
data(test_data)
ls()
```

```
## [1] "cn_data"  "snv_data" "sv_data"
```

The information required for the copy number data is:

```
print(head(cn_data))
```

```
##    chr    start      end copyNumber minorAlleleCopyNumber
## 1 chr1        1  9631965          2                     1
## 2 chr1  9631966  9631966          3                     1
## 3 chr1  9631967 11239516          2                     1
## 4 chr1 11239517 11239533          3                     1
## 5 chr1 11239534 22578082          2                     1
## 6 chr1 22578083 27086500          2                     1
```

and for the structural variants:

```
print(head(sv_data))
```

```
##     sample chr1      pos1  chr2      pos2 filter homlen homseq inslen strands
## 318   Test chr1 203476803  chr1 204967317   PASS      .      .      .      -+
## 319   Test chr1 203476817  chr1 204967172   PASS      .      .      .      +-
## 320   Test chr1 203507815  chr9  37245409   PASS      .      .      .      +-
## 321   Test chr1 203509239 chr12  69072120   PASS      3    GTA      .      ++
## 322   Test chr1 203509728 chr12  69071885   PASS      .      .      .      -+
## 323   Test chr1 203509991  chr1 203522717   PASS      1      A      .      +-
```

The main function of the package is *ReConPlot*. This function generates a rearrangement plot for the genomic regions defined above in the *chr_selection* data.frame. In addition to SV and copy number information, it is necessary to indicate which genomic regions are to be displayed. This is defined using a data.frame indicating the chromosomes, as well as the start and end positions for each chromosome. See below for an example. If

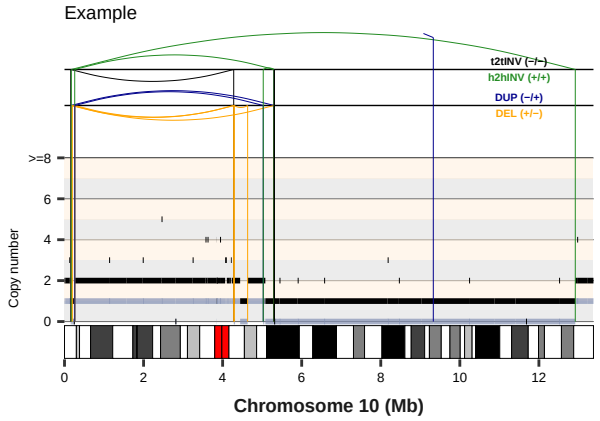
the entire chromosome is to be displayed, the start position needs to be set to 0, and the end position can be set to a value higher than the lenght of human chromosomes, e.g. 250Mbp (the size of chromosome 1 is ~249Mbp).

```
chrs=c("chr10")
chr_selection = data.frame(
  chr=chrs,
  start=rep(0 ,length(chrs)),
  end=rep (250000000, length(chrs))
)
```

Once the regions of interest is defined, we can generate the plot. For example, we can plot the genomic rearrangement profile for chromosome 10:

```
plot = ReConPlot(sv_data,
cn_data,
chr_selection=chr_selection,
legend_SV_types=T,
pos_SVtype_description=115000000,
scale_separation_SV_type_labels=1/23,
title="Example")

print(plot)
```



In this example, the total and minor copy number data are shown in black and red, respectively, on top of a representation of the Giemsa stain annotation as reported in the UCSC Genome Browser. The SVs are shown as arcs following the notation established by the Pan-Cancer Analysis of Whole Genomes Project (PCAWG). The colour arbitrarily assigned to each SV type is shown in the legend on the left-hand side (see below for options to modify graphical parameters). SVs are classified into 4 categories depending on the read orientiation at the breakpoints: deletion-like SVs (+-), duplication-like SVs (-+), and inversions (++ and −).

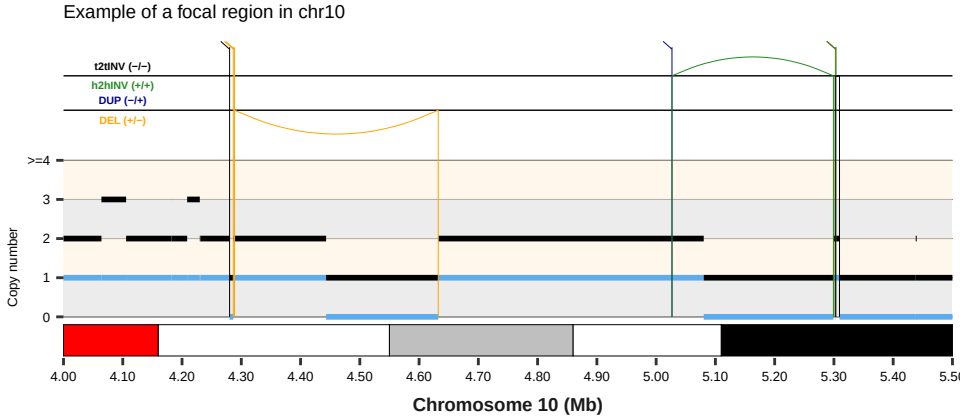
In our experience, a width of 6cm per chromosome, and a height of 5 cm per chromosome are adequate to display rearrangement plots and save them for publications.

If we wanted to visualize a more focal region of chromosome 10, we could do so by specifying the region of interest and then applying the function:

```
chrs=c("chr10")
chr_selection = data.frame(
  chr=chrs,
  start=rep(40000000 ,length(chrs)),
  end=rep (55000000, length(chrs))
)

plot = ReConPlot( sv_data,
cn_data,
chr_selection=chr_selection,
scale_ticks=1000000,
color_minor_cn="steelblue2",
size_title = 7,
legend_SV_types=T,
pos_SVtype_description=1000000,
scale_separation_SV_type_labels=1/18,
max.cn = 4,
curvature_intrachr_SVs = -0.12,
title="Example of a focal region in chr10")

print(plot)
```



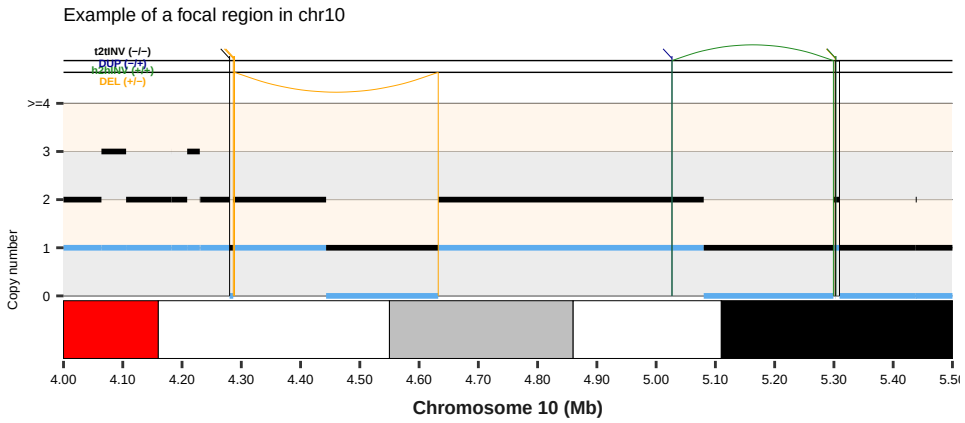
We can also change the relative span of the SV and copy number plots using the argument *scaling_cn_SVs*, which defaults to 1/6. The dimensions of the Giemsa stain track can be modified using the parameters *upper_limit_karyotype* and *karyotype_rel_size*.

```
chrs=c("chr10")
chr_selection = data.frame(
  chr=chrs,
  start=rep(40000000 ,length(chrs)),
  end=rep (55000000, length(chrs))
)

plot = ReConPlot( sv_data,
cn_data,
chr_selection=chr_selection,
scale_ticks=1000000,
color_minor_cn="steelblue2",
```

```
size_title = 7,
legend_SV_types=T,
scaling_cn_SVs = 1/18,
upper_limit_karyotype = -0.1,
karyotype_rel_size = 0.3,
pos_SVtype_description=1000000,
scale_separation_SV_type_labels=1/23,
curvature_intrachr_SVs = -0.1,
max.cn = 4,
title="Example of a focal region in chr10")

print(plot)
```



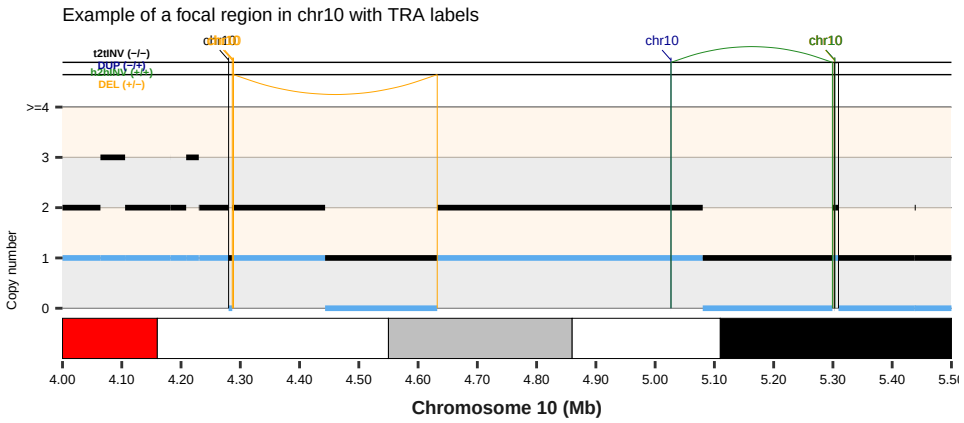
Interchromosomal translocations or SVs with one breakpoint falling outside the *chr_selection* viewpoint can be labelled with the destination chromosome with the parameter *label_interchr_SV=TRUE*. This setting is useful for simple SVs, but not recommended for genomic areas with complex genome rearrangements involving multiple translocations in a small areas.

```
chrs=c("chr10")
chr_selection = data.frame(
  chr=chrs,
  start=rep(40000000 ,length(chrs)),
  end=rep (55000000, length(chrs))
)

plot = ReConPlot( sv_data,
cn_data,
chr_selection=chr_selection,
scale_ticks=1000000,
color_minor_cn="steelblue2",
size_title = 7,
legend_SV_types=T,
scaling_cn_SVs = 1/18,
pos_SVtype_description=1000000,
scale_separation_SV_type_labels=1/23,
curvature_intrachr_SVs = -0.1,
max.cn = 4,
title="Example of a focal region in chr10 with TRA labels",
```

```
label_interchr_SV=TRUE)

print(plot)
```



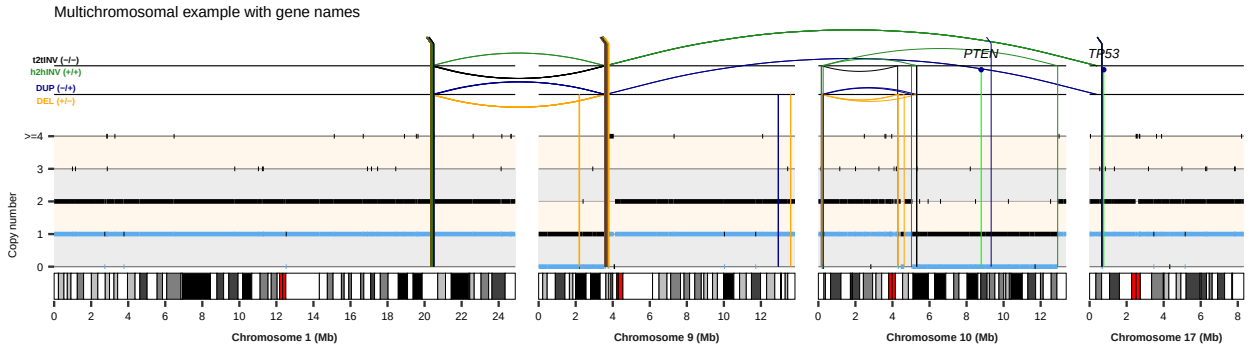
We can also plot several chromosomes. In this case, interchromosomal SVs involving the set of chromosomes selected are also displayed. In our experience, a width of 19cm for multichromosomal plots gives good results. Note that this is the maximum width for figures for most publishing groups.

The plot above was generated using default parameter values. However, it is possible to modify some graphical parameters. For example, we can change the colour of the minor copy number segments using the "color_minor_cn" argument. We can input a list of genes using the argument "genes" to display the genomic locations of relevant genes, such as TP53 below. Other available options are shown below and described in the package documentation.

```
chrs=c('chr1',"chr9","chr10","chr17")
chr_selection = data.frame(
  chr=chrs, #factor(chrs,levels=chrs),
  start=rep(0 ,length(chrs)),
  end=rep (250000000, length(chrs))
)
```

```
plot = ReConPlot( sv_data,
cn_data,
chr_selection=chr_selection,
genes=c("MDM2","MYC","TP53","CDK4","EGFR","PTEN"),
title="Multichromosomal example with gene names",
size_gene_label = 2,
pos_SVtype_description=1000000,
color_minor_cn="steelblue2",
size_title = 7,
size_chr_labels=5,
legend_SV_types=T,
scale_separation_SV_type_labels=1/23,
max.cn = 4,
scale_ticks=20000000)

print(plot)
```

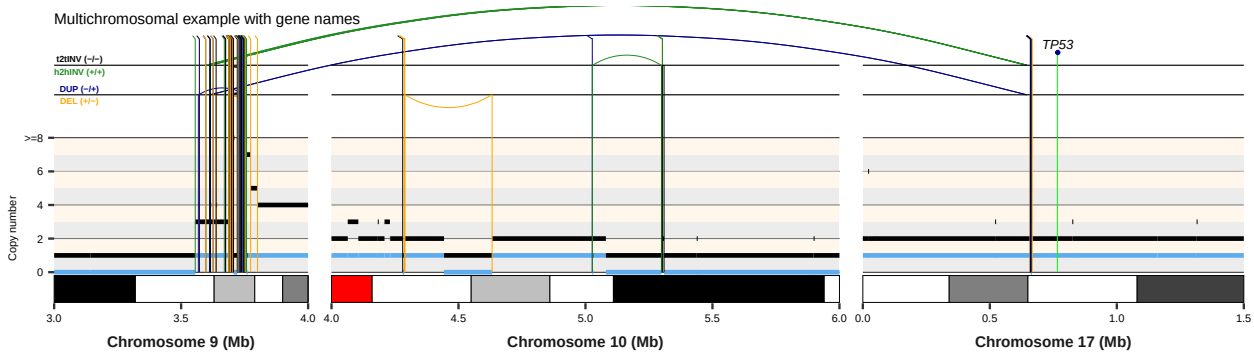


We can focus on specific regions of interest across multiple chromosomes:

```
chrs=c("chr9","chr10","chr17")
chr_selection = data.frame(
  chr=chrs,
  start=c(30, 40, 0) * 1000000,
  end=c(40, 60, 15) * 1000000
)
```

```
plot = ReConPlot( sv_data,
cn_data,
chr_selection=chr_selection,
genes=c("MDM2","MYC","TP53","CDK4","EGFR"),
title="Multichromosomal example with gene names",
size_gene_label = 2,
pos_SVtype_description=1000000,
color_minor_cn="steelblue2",
scale_separation_SV_type_labels=1/23,
size_title = 7,
size_chr_labels=7,
legend_SV_types=T,
curvature_interchr_SVs = -0.08,
max.cn = 8,
scale_ticks=5000000)

print(plot)
```

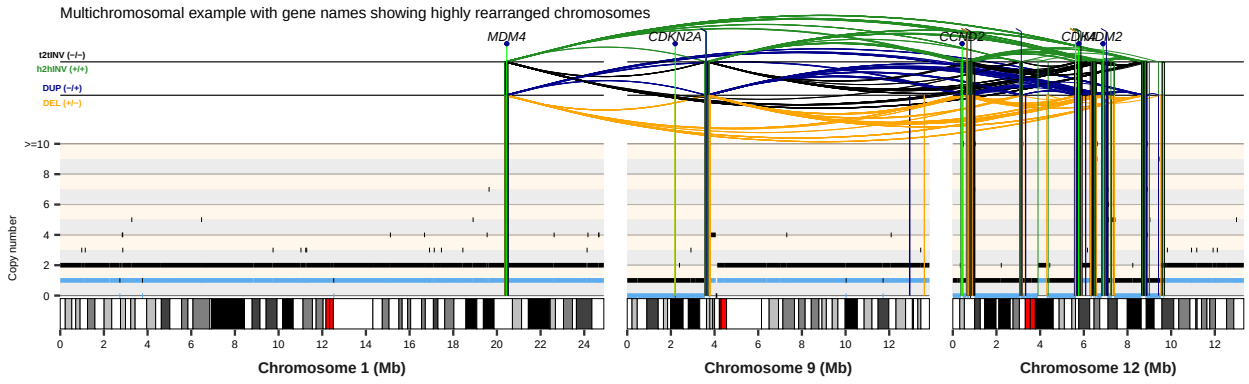


The examples above display chromosomes with a relatively low number of SVs. However, it is possible to visualize more complex patterns, such as those detected in chromosome 12 in this sample:

```
chrs=c('chr1',"chr9","chr12")
chr_selection = data.frame(
  chr=chrs,
  start=rep(0 ,length(chrs)),
  end=rep (250000000, length(chrs))
)
```

```
plot = ReConPlot( sv_data,
cn_data,
chr_selection=chr_selection,
genes=c("MDM2","MDM4","MYC","CDK4","EGFR","CDKN2A","CCND2"),
title="Multichromosomal example with gene names showing highly rearranged chromosomes",
size_gene_label = 2,
pos_SVtype_description=1000000,
color_minor_cn="steelblue2",
size_title = 7,
size_chr_labels=7,
legend_SV_types=T,
scale_separation_SV_type_labels=1/23,
max.cn = 10,
scale_ticks=20000000)

print(plot)
```



Finally, an annotation plot can be added below the ReCon plot, by providing a dataframe with the columns "chr", "pos" and "y" and the parameter *custom_annotation*. Only the points within the *chr_selection* point of view will be displayed. "y" can represent a scale of user definition. In the example below, we annotate the plot with SNVs where "y" represents the allele frequency. Please note the other parameters controlling different visual aspects of the annotation plot.

```
head(snv_data)
```

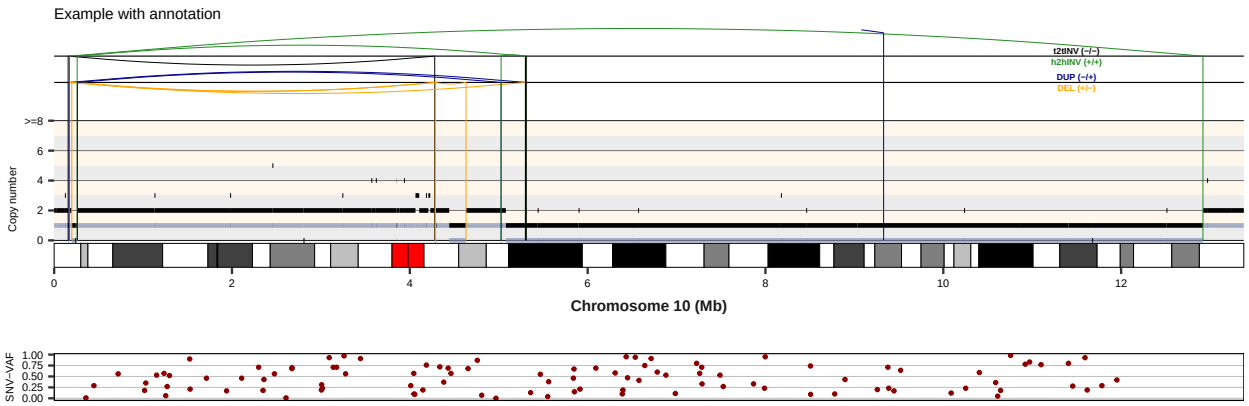
```
##     chr      pos    y
## 1 chr10  3578551 0.01
## 2 chr10  4477330 0.29
```

```
## 3 chr10  7204511 0.56
## 4 chr10 10177400 0.18
## 5 chr10 10312319 0.35
## 6 chr10 11527581 0.53
```

```
chrs=c("chr10")
chr_selection = data.frame(
  chr=chrs,
  start=rep(0 ,length(chrs)),
  end=rep (250000000, length(chrs))
)

plot = ReConPlot(sv_data,
cn_data,
chr_selection=chr_selection,
legend_SV_types=T,
pos_SVtype_description=115000000,
scale_separation_SV_type_labels=1/23,
title="Example with annotation",
curvature_intrachr_SVs=-.05,
custom_annotation=snv_data,
ann_dot_col="darkred",
ann_dot_size=.5,
ann_y_title="SNV-VAF",
#Label for y-axis
ann_rel_size=.4
#Relative size to main plot
)

print(plot + theme(plot.margin=margin(t=1, unit = "cm")))
```



The ReCon plots are ggplot objects and can be modified after generation as such. They can be easily saved to a PDF file. In our experience, the following dimensions work well for publication-quality figures and written reports. If using an annotation plot in combination with the ReCon plot, the height might need to be increased.

```
ggsave(filename = "example_ReConPlot.pdf", plot = p, width = 19, height = 5, units = "cm")
```

Contact

If you have any comments or suggestions please raise an issue or contact us: Jose Espejo: jespejo@ebi.ac.uk Isidro Cortes-Ciriano: icortes@ebi.ac.uk