Riparian Plot Guide

JTLovell

10/25/2021

1. Run GENESPACE

See the GENESPACE overview for pipeline details. The full run through syntenic block construction is here:

```
if (!requireNamespace("devtools", quietly = TRUE))
    install.packages("devtools")
if (!requireNamespace("GENESPACE", quietly = TRUE))
    devtools::install_github("jtlovell/GENESPACE", upgrade = F)


if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
if (!requireNamespace("Biostrings", quietly = TRUE))
  BiocManager::install("Biostrings")
if (!requireNamespace("rtracklayer", quietly = TRUE))
   BiocManager::install("rtracklayer")


library(GENESPACE)
runwd <- file.path("~/Desktop/testGenespace")

make_exampleDataDir(writeDir = runwd)

gpar <- init_genespace(
  genomeIDs = c("human","chimp","rhesus"),
  speciesIDs = c("human","chimp","rhesus"),
  versionIDs = c("human","chimp","rhesus"),
  ploidy = rep(1,3),
  diamondMode = "fast",
  orthofinderMethod = "fast",
  wd = runwd,
  nCores = 4,
  minPepLen = 50,
  gffString = "gff",
  pepString = "pep",
  path2orthofinder = "orthofinder",
  path2mcscanx = "~/MCScanX",
  rawGenomeDir = file.path(runwd, "rawGenomes"))

parse_annotations(
  gsParam = gpar,
  gffEntryType = "gene",
```

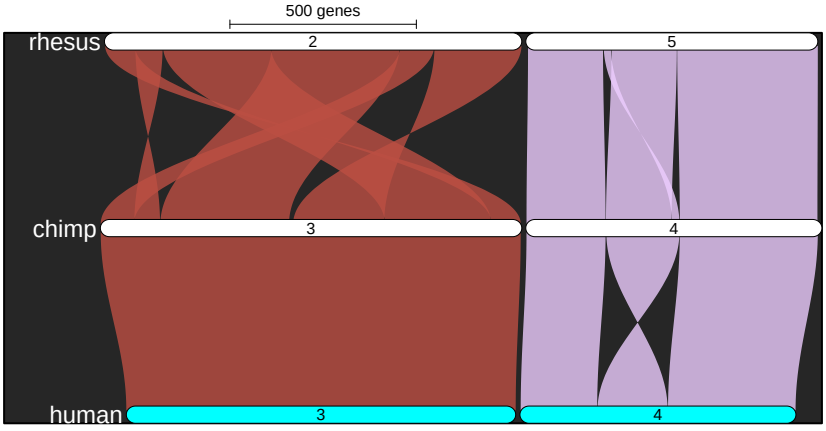
```
  gffIdColumn = "locus",
  gffStripText = "locus=",
  headerEntryIndex = 1,
  headerSep = " ",
  headerStripText = "locus=")

gpar <- run_orthofinder(
  gsParam = gpar)

gpar <- synteny(gsParam = gpar)
```

2. Basic riparian functionality

2.1 Default specification Color chromosomes by the reference genome (and highlight the reference chromosomes in light yellow). Return the source data to construct the plots (for publications etc.). If you want to save the figure to file, see `?pdf`, `?png` or other graphic device writing functions.

```
ripSourceData <- plot_riparian(
  gpar,
  returnSourceData = T,
  highlightRef = "cyan")
```

```
##  Loading the gff ... Done!
##  Mapping genes against human chromosomes ... Done!
##  Projecting linear coordinate system ... Done!
##  Generating block coordinates ... Done!
##  Rendering plot ...
```



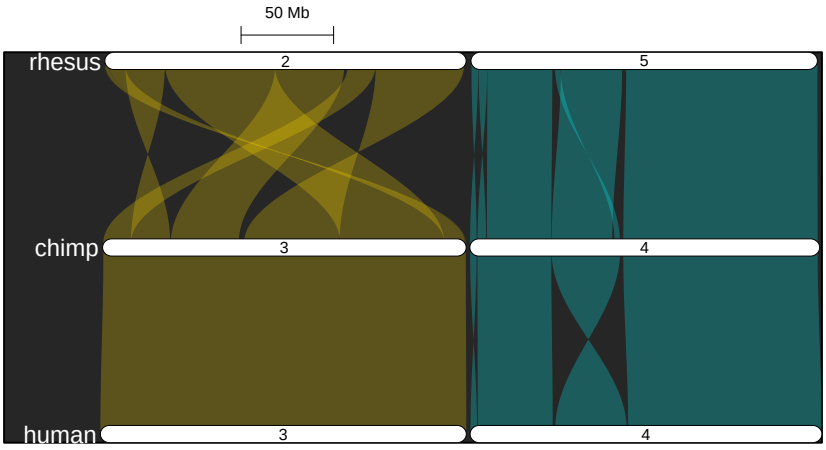
Chromosomes scaled by gene rank order

```
## Done!
```

2.2 change the input data Use physical positioninstead of gene rank order (`useOrder = F`) and plot the small syntenic blocks instead of the large syntenic regions (`plotRegions = F`). Also illustrating here that you can manually specify the colors (`colByChrscolByChrs`) and transparency (`braidAlpha`) of the syntenic polygons. If the number of colors specified is the same as the number of reference chrs, the colors will be respected exactly. Otherwise, the colors are used to make a ramp palette.

```
plot_riparian(
  gpar,
```

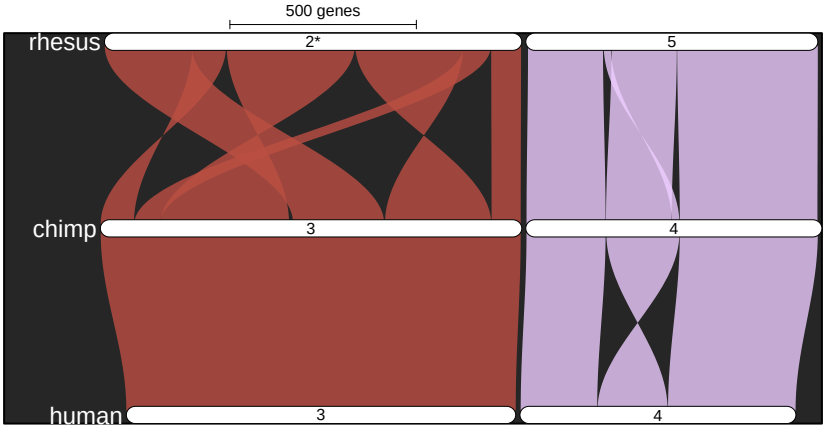
```
  plotRegions = FALSE,
  useOrder = FALSE,
  colByChrs = c("gold", "cyan"),
  braidAlpha = .25,
  verbose = FALSE)
```



Chromosomes scaled by physical position

2.3 Invert a chromosome that is more syntenic if flipped Often some chromosomes are more syntenic if flipped. Here, we can specify which chromosomes to flip. The chromosome ID is then flagged with an "*".

```
invertThisGenomeChr <- data.table(genome = "rhesus", chr = "2")
plot_riparian(
  gpar,
  invertTheseChrs = invertThisGenomeChr,
  verbose = FALSE)
```

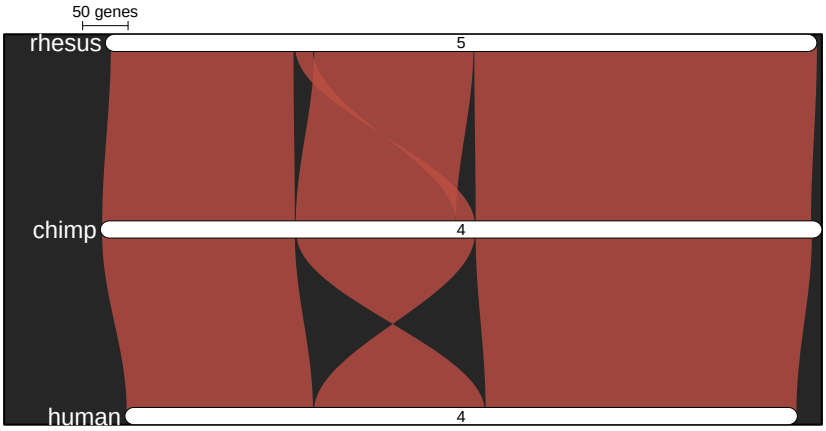


Chromosomes scaled by gene rank order

3. Using riparian plots to highlight specific regions

3.1 only plot specific chromosomes Often, the user wants to study a specific region, we can use riparian to just zoom in on specific chromosome. In this case, the chromosome has to be in the reference genome ... "human", here.

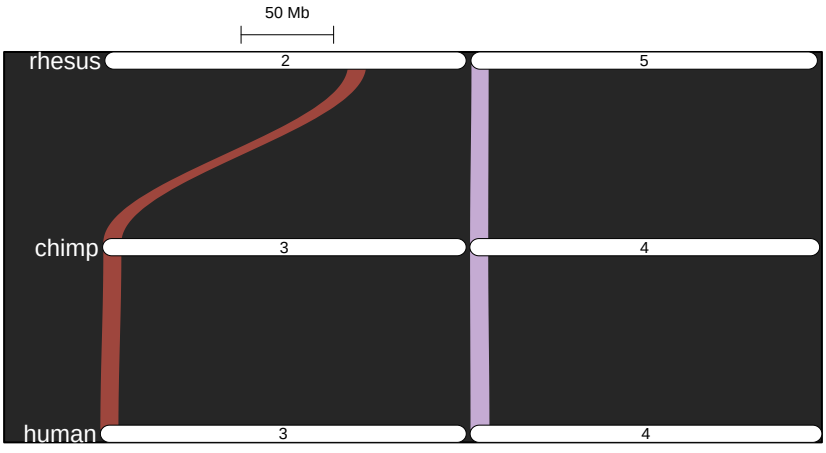
```
plot_riparian(
  gpar,
  onlyTheseChrs = "4",
  verbose = FALSE)
```



Chromosomes scaled by gene rank order

3.2 only plot specific regions We can also zoom into just a couple regions. Here, the regions are specified by the physical position (bp), so we are best served setting useOrder = F. This is a nice way to combine multiple plots with multiple colors . . . make a full plot, then a separate plot for each region and overlay them in a vector graphics editor.

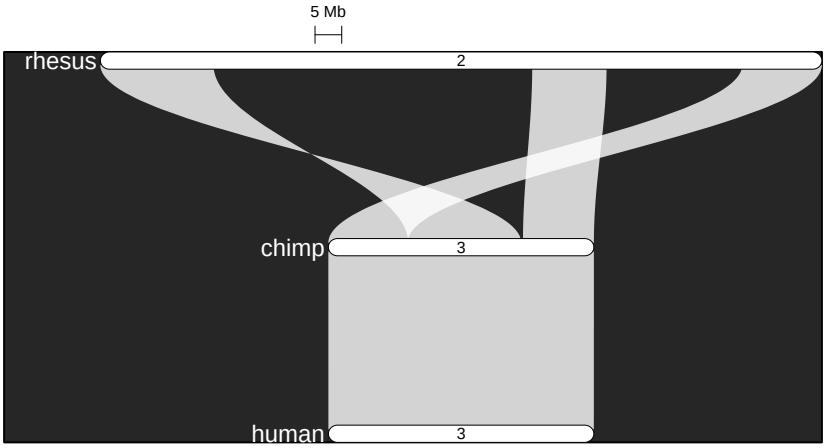
```
regs <- data.table(
  genome = c("human", "rhesus"),
  chr = c(3, 5),
  start = c(0, 0),
  end = c(1e7, 1e7))
plot_riparian(
  gpar,
  useOrder = FALSE,
  onlyTheseRegions = regs,
  verbose = FALSE)
```



Chromosomes scaled by physical position

we can also zoom in on just the highlighted regions.

```
regs <- data.table(
  genome = "human",
  chr = 3,
  start = 0,
  end = 5e7)
plot_riparian(
  gpar,
  useOrder = FALSE,
  onlyTheseRegions = regs,
  excludeChrOutOfRegion = TRUE,
  colByChrs = "white",
  verbose = FALSE)
```

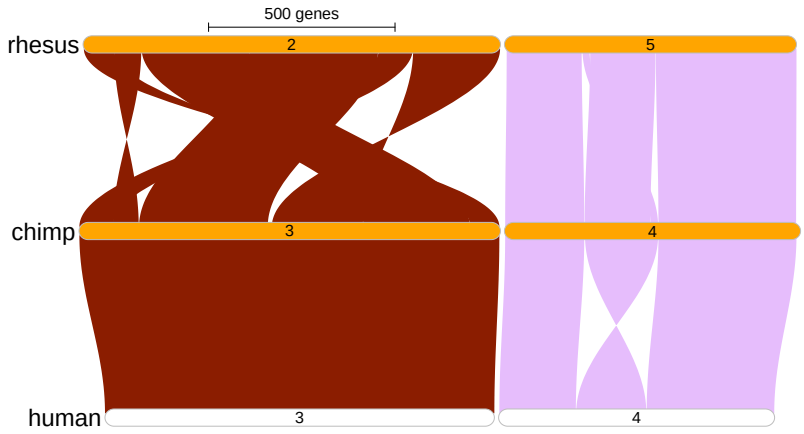


Chromosomes scaled by physical position

4 Adjust the general appearence

4.1 change the background and chr colors Use a white background, orange chrs and completely opaque braids.

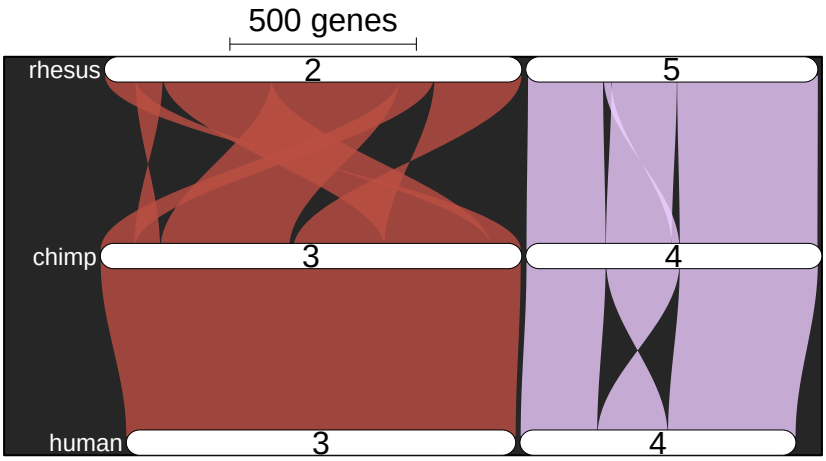
```
plot_riparian(
  gpar,
  blackBg = FALSE,
  chrFill = "orange",
  chrBorder = "grey",
  braidAlpha = 1,
  verbose = FALSE)
```



Chromosomes scaled by gene rank order

4.2 change the size of chrs Increase the size of the chr links, but decrease the buffer size around the chrs

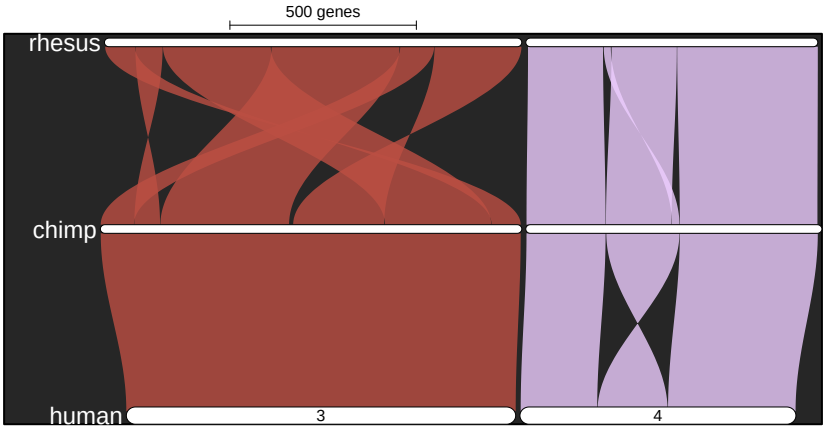
```
plot_riparian(
  gpar,
  chrLabCex = 1,
  chrRectBuffer = 1.1,
  verbose = FALSE)
```



Chromosomes scaled by gene rank order

4.3 only label one genome In some cases, where there are lots of genomes or several genomes have tons of chromosomes, we might want to just label one or two genome chrs

```
plot_riparian(
  gpar,
  labelTheseGenomes = "human",
  verbose = F)
```



Chromosomes scaled by gene rank order

There is more functionality, but this is most of it.