R4RNA: A R package for RNA visualization and analysis

Volodymyr Tsybulskyi, Mohamed Mounir, Daniel Lai,Irmtraud M. Meyer

August 13, 2021

Contents

1 R4RNA

The R4RNA package aims to be a general framework for the analysis of RNA secondary structure and comparative analysis in R, the language so chosen due to its native support for publication-quality graphics, and portability across all major operating systems, and interactive power with large datasets.

To demonstrate the ease of creating complex arc diagrams, a short example is as follows.

1.1 Reading Input

Currently, supported input formats include dot-bracket, connect, bpseq, and a custom "helix" format. Below, we read in a structure predicted by TRANSAT, the known structure obtained form the RFAM database.

```
> library(R4RNA)
> message("TRANSAT prediction in helix format")
> transat_file <- system.file("extdata", "helix.txt", package = "R4RNA")
> transat <- readHelix(transat_file)
> message("RFAM structure in dot bracket format")
> known_file <- system.file("extdata", "vienna.txt", package = "R4RNA")
> known <- readVienna(known_file)
> message("Work with basepairs instead of helices for more flexibility")
> message("Breaks all helices into helices of length 1")
> transat <- expandHelix(transat)
> known <- expandHelix(known)
```

For multiple entities you can use just updated "helix" format. Below, we read predicted and known structures.

```
> #read fasta
> fasta_file.mltp <- system.file("extdata", "fasta.mltp.txt", package = "R4RNA")
> fasta.mltp <- readBStringSet(fasta_file.mltp)
> #read helix
> transat.file.mltp <- system.file("extdata", "transat.helix.mltp.txt" ,package = "R4RNA")
> helix.transat.mltp <- readHelixMltp(transat.file.mltp)
> known.file.mltp <- system.file("extdata", "known.helix.mltp.txt", package = "R4RNA")
> helix.known.mltp <- readHelixMltp(known.file.mltp)
> is.helix.mltp(helix.transat.mltp)
```

```
[1] TRUE
```

```
> helix.transat.exp <- expandHelixMltp(helix.transat.mltp)
> helix.known.exp <- expandHelixMltp(helix.known.mltp)
> is.helix.mltp(helix.transat.exp)
```

```
[1] TRUE
```

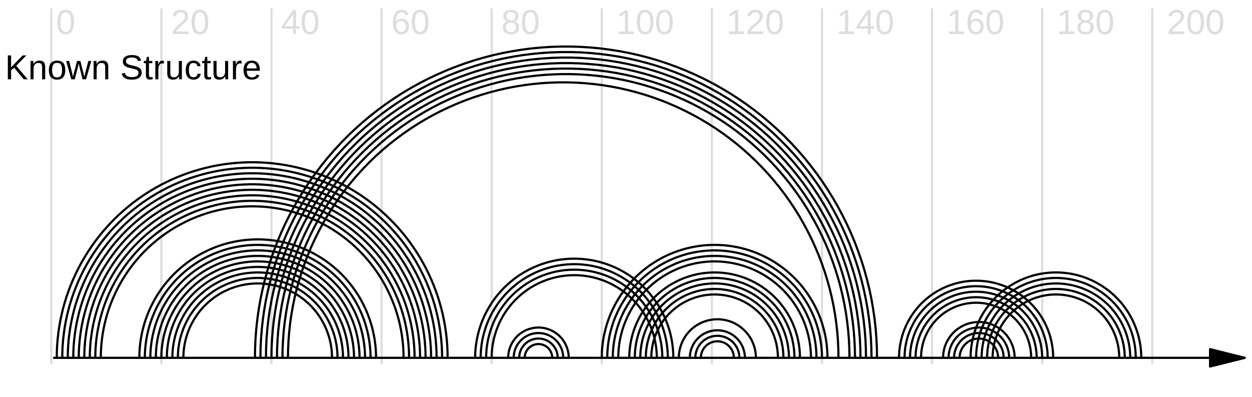
```
> is.helix.mltp(helix.known.exp)
```

```
[1] TRUE
```

1.2 Basic Arc Diagram

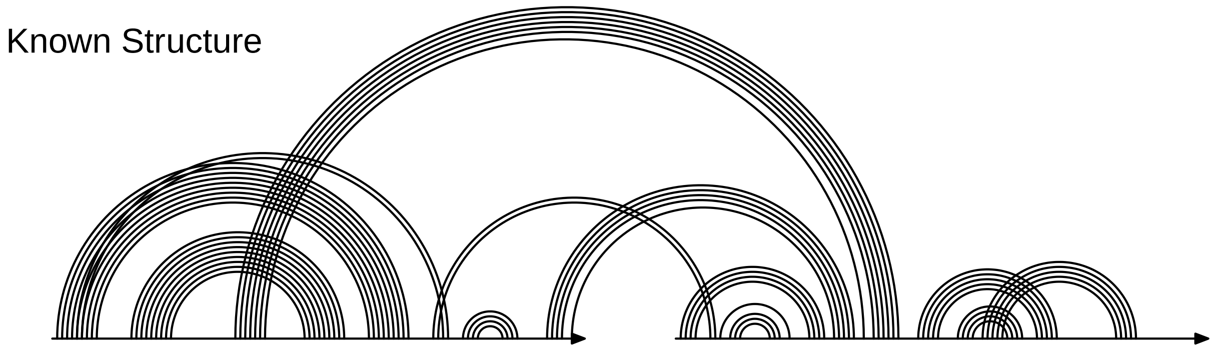
The standard arc diagram, where the nucleotide sequence is the horizontal line running left to right from 5' to 3' at the bottom of the diagram. Any two bases that base-pair in a secondary structure are connect with an arc.

```
> plotHelix(known, line = TRUE, arrow = TRUE)
> mtext("Known Structure", side = 3, line = -2, adj = 0)
```



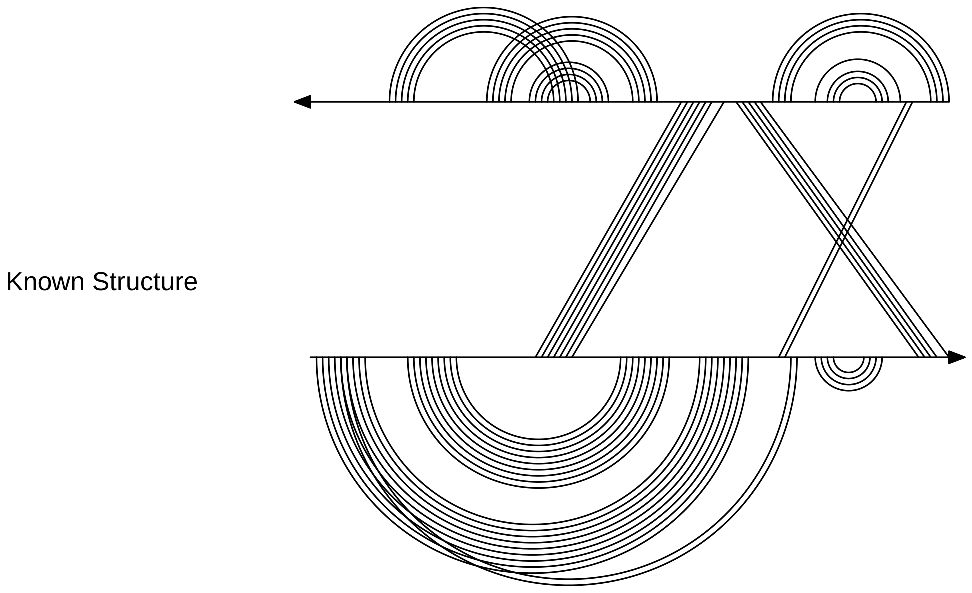
Plot for multiple entities. Trans interactions are given by length equal 1. There is Single line and Double Line Modes. Single line:

```
> plotHelixMltpSingleLine(copy(helix.known.exp),scale = FALSE)
> mtext("Known Structure", side = 3, line = -2, adj = 0)
```



Double line:

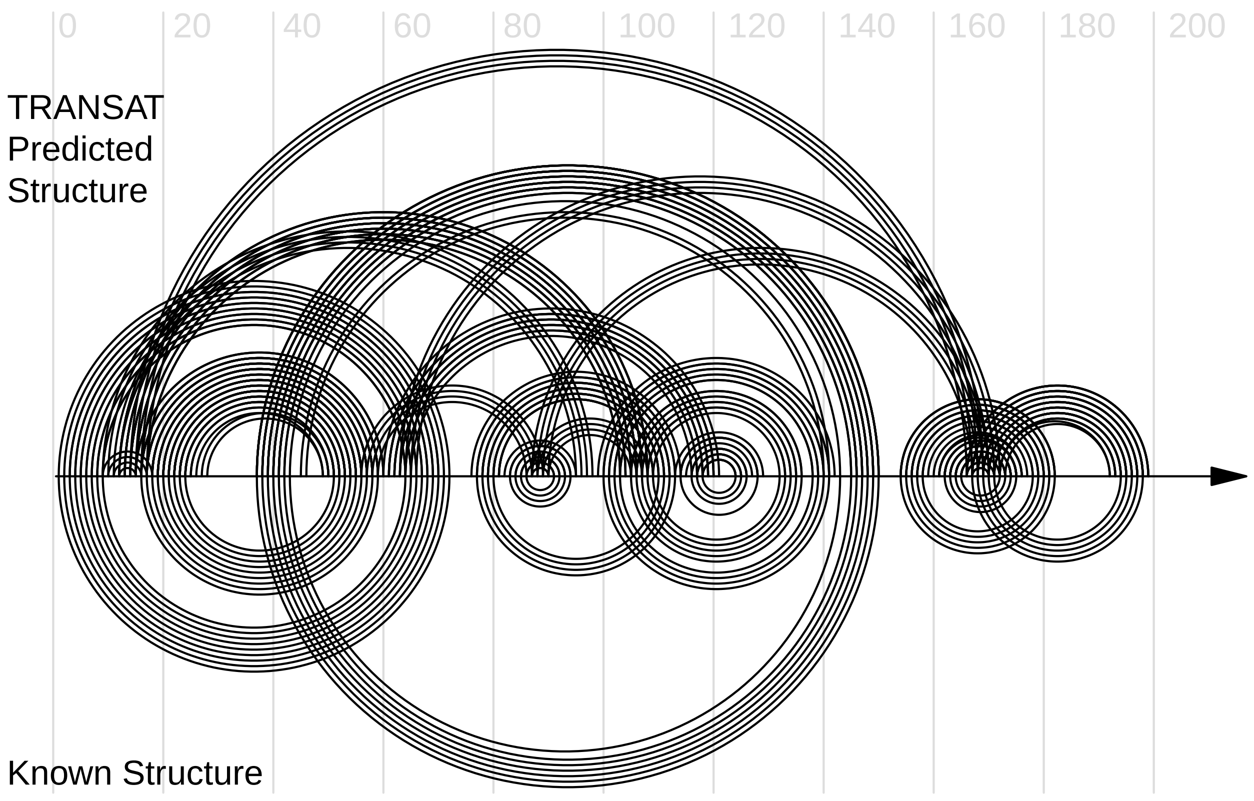
```
> plotHelixMltpDoubleLine(copy(helix.known.exp),scale = FALSE)
> mtext("Known Structure", side = 3, line = -10, adj = 0)
```



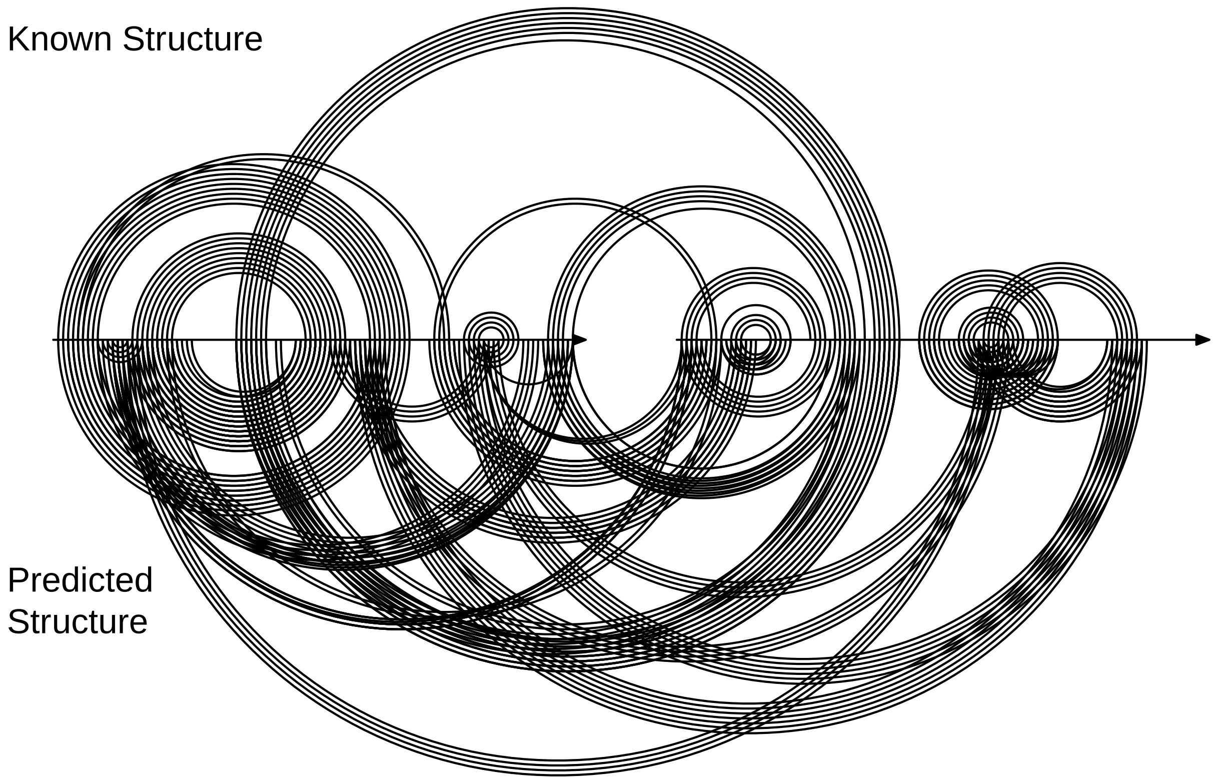
1.3 Multiple Structures

Two structures for the same sequence can be visualized simultaneously, allowing one to compare and contrast the two structures.

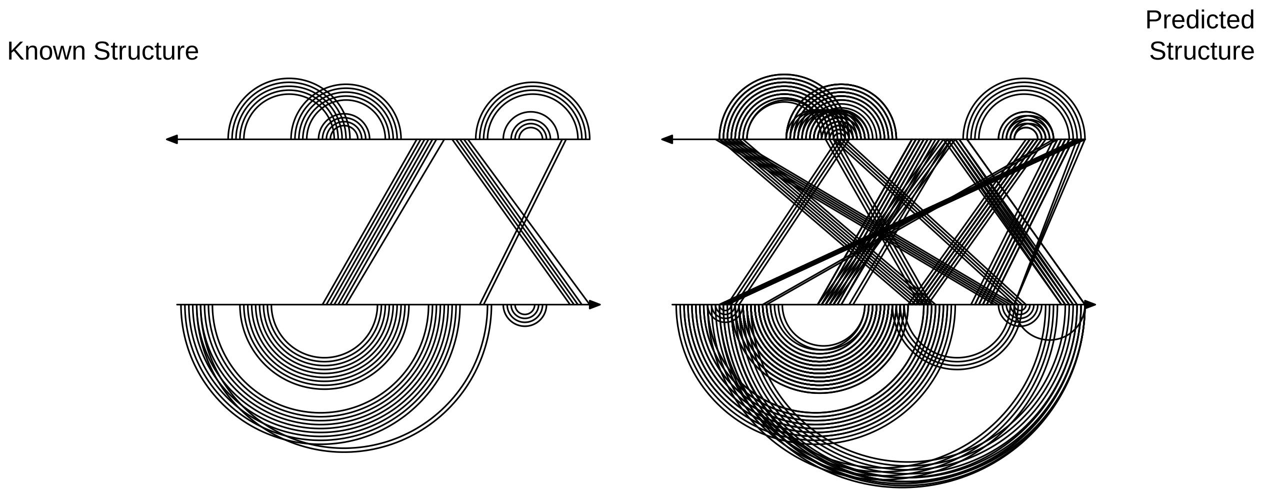
```
> plotDoubleHelix(transat, known, line = TRUE, arrow = TRUE)
> mtext("TRANSAT\nPredicted\nStructure", side = 3, line = -5, adj = 0)
> mtext("Known Structure", side = 1, line = -2, adj = 0)
```



```
> plotDoubleHelixMltpSingleLine(helix.known.exp,helix.transat.exp,scale = FALSE)
> mtext("Predicted\nStructure", side = 1, line = -5, adj = 0)
> mtext("Known Structure", side = 3, line = -2, adj = 0)
```



```
> plotDoubleHelixMltpDoubleLine(helix.known.exp,helix.transat.exp,
+                               stable = TRUE,scale = FALSE)
> mtext("Predicted\nStructure", side = 3, line = -5, adj = 1)
> mtext("Known Structure", side = 3, line = -5, adj = 0)
```



1.4 Filtering Helices

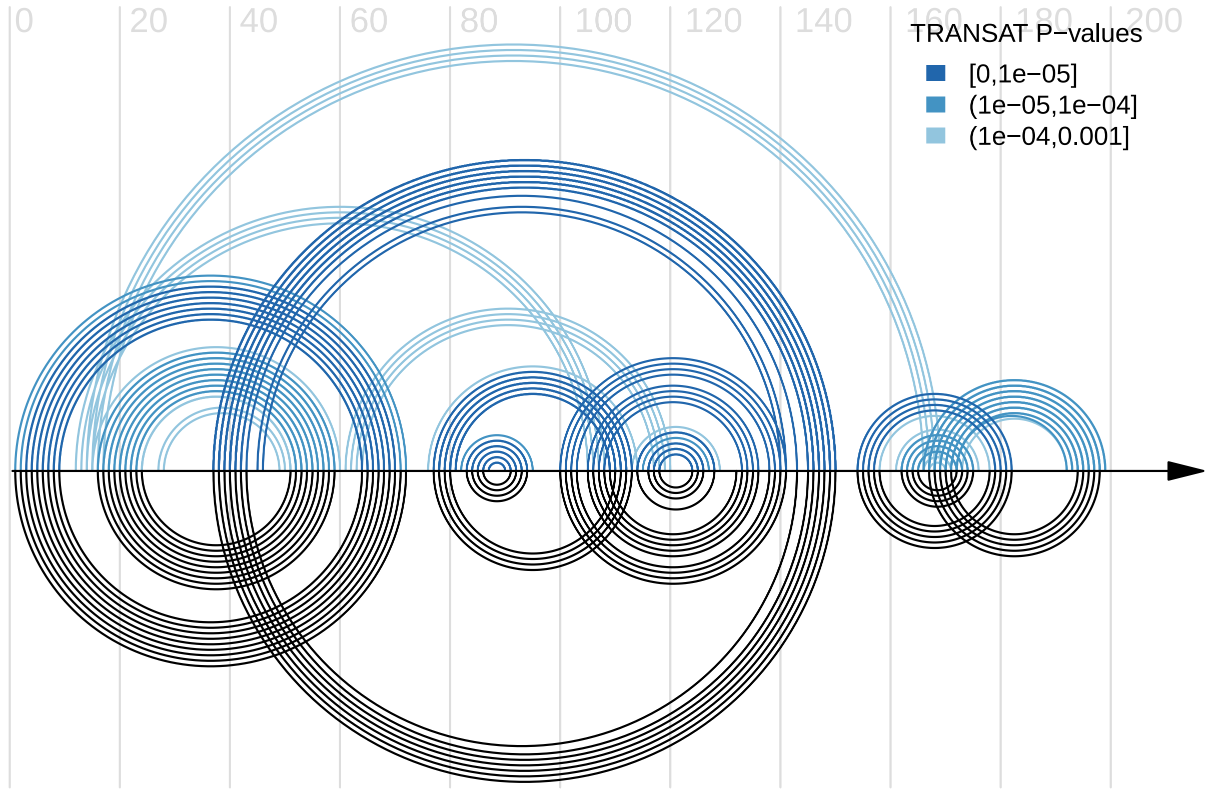
Base-pairs can be associated with a value, such as energy stability or statistical probability, and we can easily filter out basepairs according to such rules.

```
> message("Filter out helices above a certain p-value")
> transat <- transat[which(transat$value <= 1e-3), ]
```

1.5 Colouring Structures

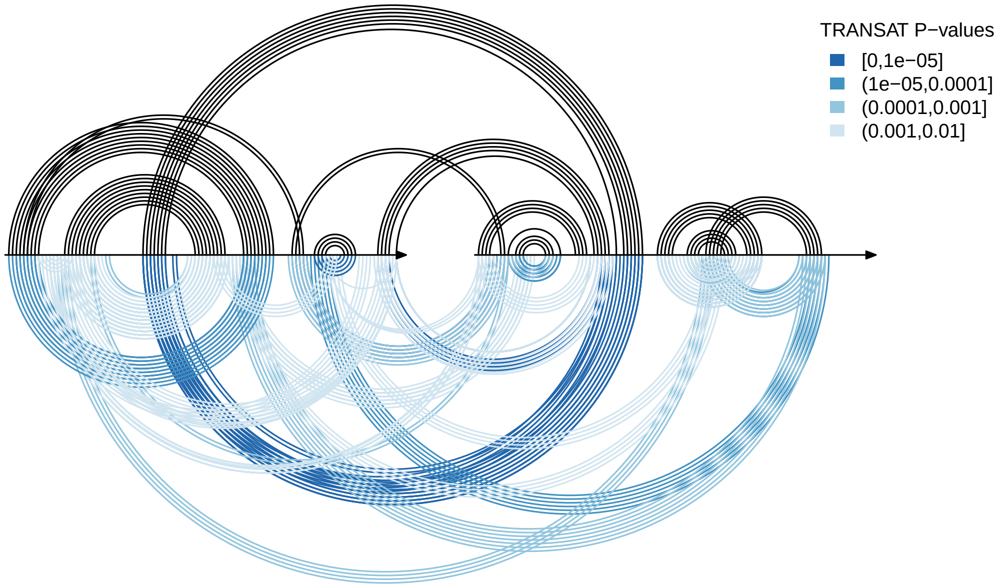
We can also assign colour to the structure according to base-pairs values.

```
> message("Assign colour to basepairs according to p-value")
> transat$col <- col <- colourByValue(transat, log = TRUE)
> message("Coloured encoded in 'col' column of transat structure")
> plotDoubleHelix(transat, known, line = TRUE, arrow = TRUE)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+     inset = 0.05, bty = "n", border = NA, cex = 0.75, title = "TRANSAT P-values")
```

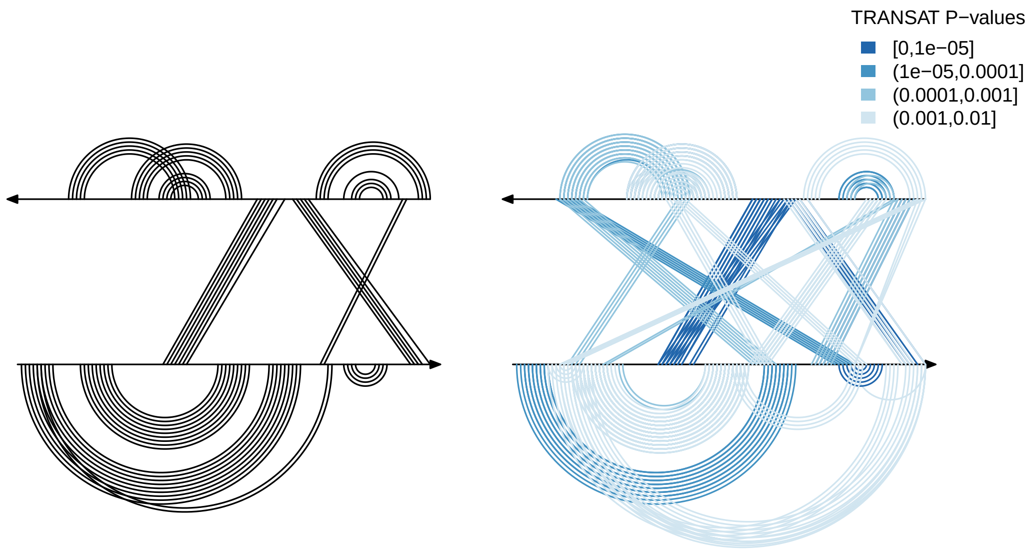


Same can be done to multiple entitites.

```
> helix.transat.exp <- helix.transat.exp[,1:6]
> helix.transat.exp$col <- col <- colourByValueMltp(helix.transat.exp, log = TRUE)
> message("Coloured encoded in 'col' column of transat structure")
> plotDoubleHelixMltpSingleLine(helix.known.exp,helix.transat.exp,scale = FALSE)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+          inset = 0.05, bty = "n", border = NA, cex = 0.75, title = "TRANSAT P-values")
```



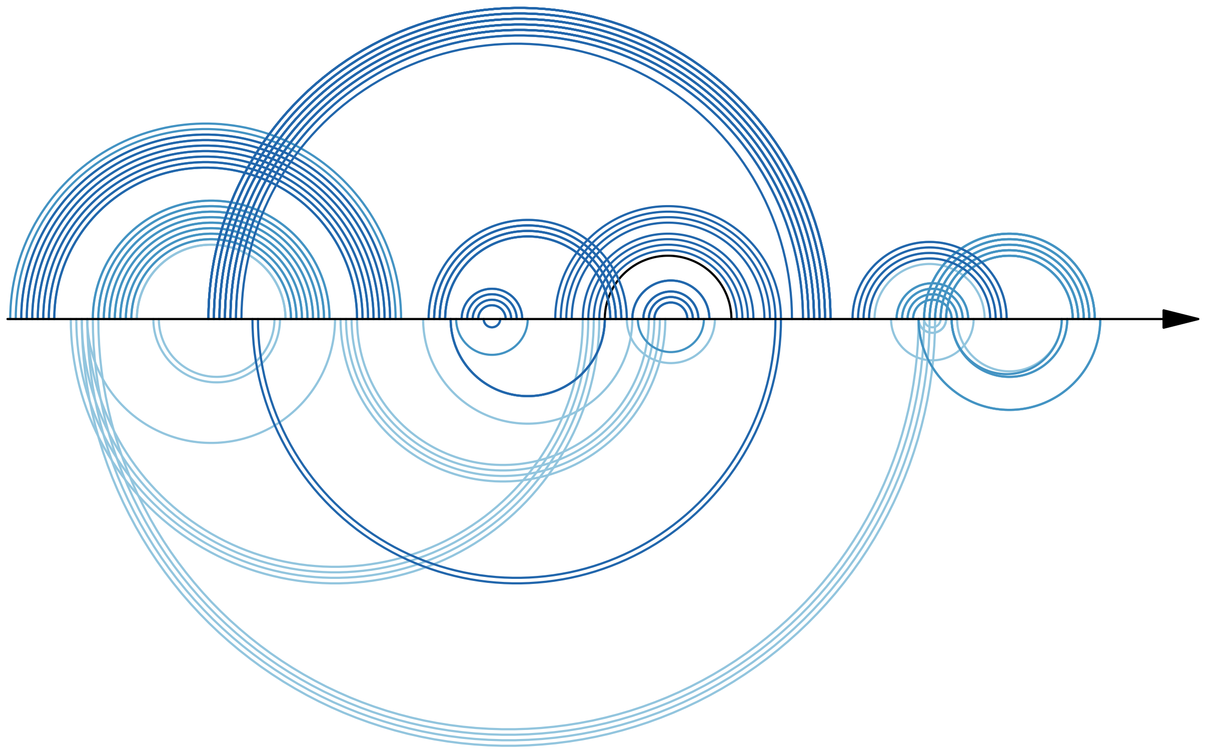
```
> helix.transat.exp <- helix.transat.exp[,1:6]
> helix.transat.exp$col <- col <- colourByValueMltp(helix.transat.exp, log = TRUE)
> message("Coloured encoded in 'col' column of transat structure")
> plotDoubleHelixMltpDoubleLine(helix.known.exp,helix.transat.exp,scale = FALSE,stable = TRUE)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+          inset = 0.05, bty = "n", border = NA, cex = 0.75, title = "TRANSAT P-values")
```



1.6 Overlapping Multiple Structures

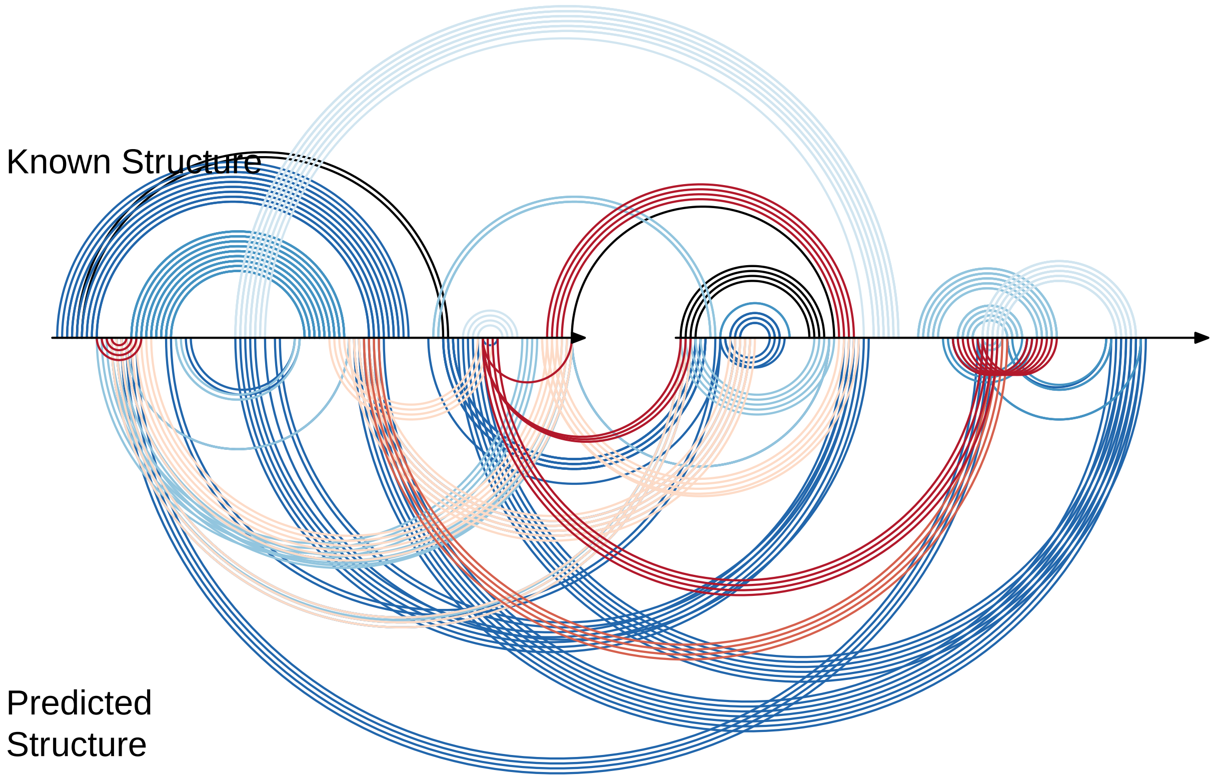
A neat way of visualizing the concordance between two structure is an overlapping structure diagram, which we can use to overlap the predicted TRANSAT structure and the known RFAM structure. Predicted basepairs that exist in the known structure are drawn above the line, and those predicted that are not known to exist are drawn below. Those known but unpredicted are shown in black above the line.

```
> plotOverlapHelix(transat, known, line = TRUE, arrow = TRUE, scale = FALSE)
```

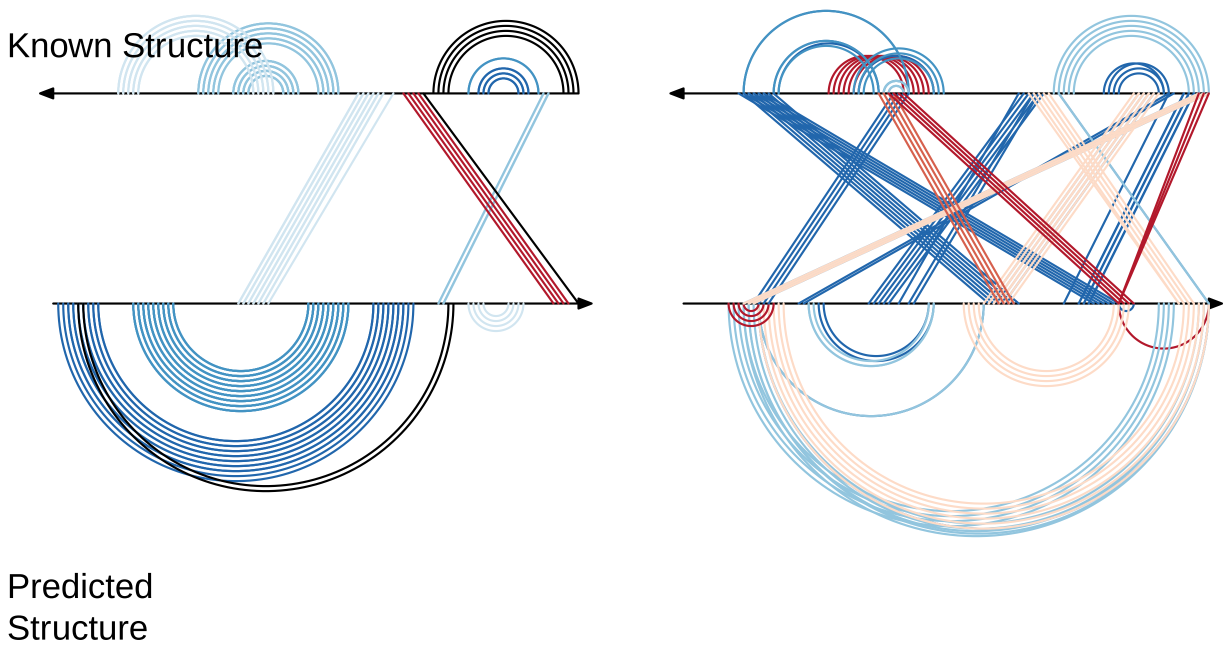


Same can be done to multiple entitites.

```
> plotComparisonHelixMltpSingleLine(helix1 = helix.known.exp[,1:6],
+                                   helix2 = helix.transat.exp[,1:6],
+                                   scale = FALSE)
> mtext("Known Structure", side = 3, line = -5, adj = 0)
> mtext("Predicted\nStructure", side = 1, line = -2, adj = 0)
```



```
> plotComparisonHelixMltpDoubleLine(helix1 = helix.known.exp[,1:6],
+                                   helix2 = helix.transat.exp[,1:6],
+                                   scale = FALSE)
> mtext("Known Structure", side = 3, line = -5, adj = 0)
> mtext("Predicted\nStructure", side = 1, line = -2, adj = 0)
```

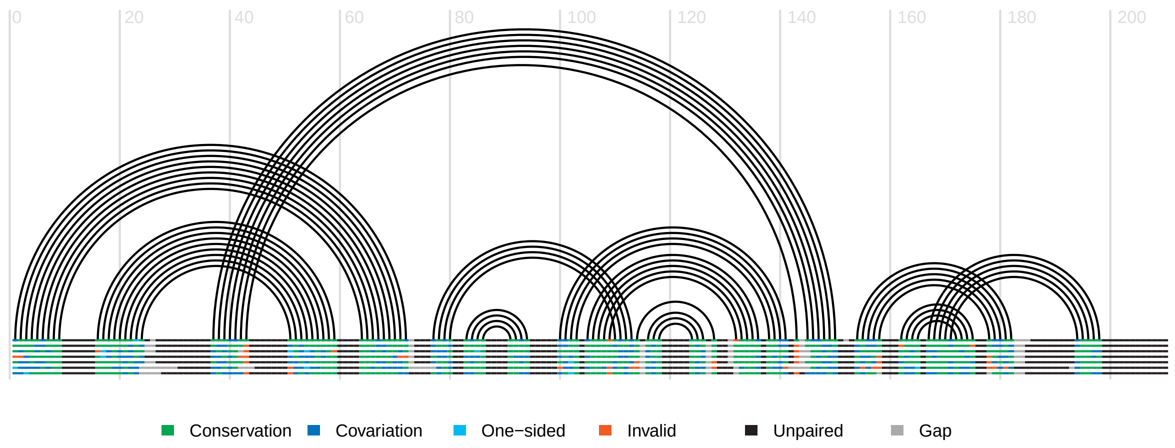


1.7 Visualizing Multiple Sequence Alignments

In addition to visualizing the structure alone, we can also visualize a secondary structure along with aligned nucleotide sequences. In the following, we will read in a multiple sequence alignment obtained from RFAM, and visualize the known structure on top of it.

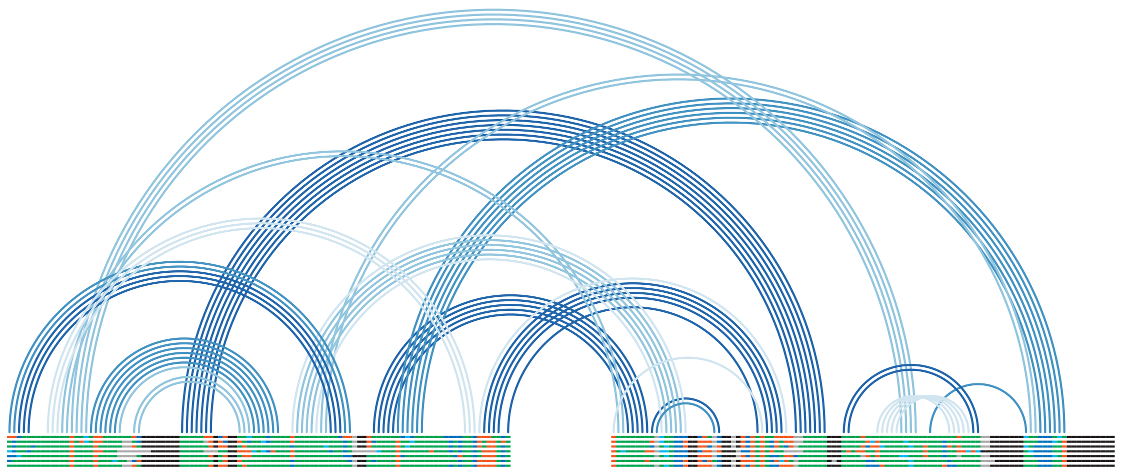
We can also annotate the alignment colours according to their agreement with the known structure. If a sequence can form as basepair as dictated by the structure, the basepair is coloured green, else red. For green basepairs, if a mutation has occured, but basepairing potential is retained, it is coloured in blue (dark for mutations in both bases, light for single-sided mutation). Unpaired bases are in black and gaps are in grey.

```
> message("Multiple sequence alignment of interest")
> library(Biostrings)
> fasta_file <- system.file("extdata", "fasta.txt", package = "R4RNA")
> fasta <- as.character(readBStringSet(fasta_file))
> message("Plot covariance in alignment")
> plotCovariance(fasta, known, cex = 0.5)
```

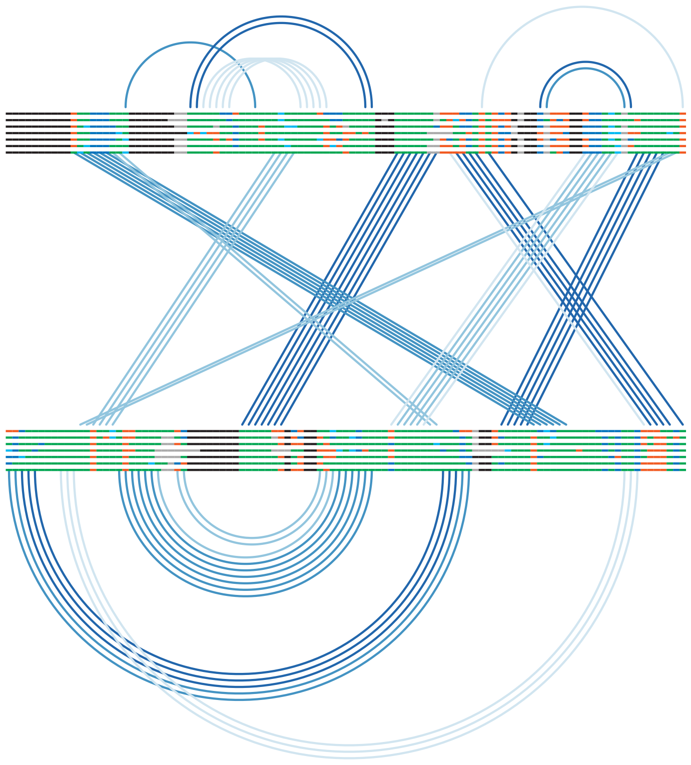


Same can be done to multiple entitites.

```
> message("Multiple sequence alignment of interest")
> fasta_file.mltp <- system.file("extdata", "fasta.mltp.txt", package = "R4RNA")
> fasta.mltp <- readBStringSet(fasta_file.mltp)
> message("Plot covariance in alignment")
> plotCovarianceMltpSingleLine(helix = helix.transat.exp[,1:7],
+                              msa = fasta.mltp,grid = FALSE,
+                              scale = FALSE)
```



```
> message("Multiple sequence alignment of interest")
> fasta_file.mltp <- system.file("extdata", "fasta.mltp.txt", package = "R4RNA")
> fasta.mltp <- readBStringSet(fasta_file.mltp)
> message("Plot covariance in alignment")
> plotCovarianceMltpDoubleLine(helix = helix.transat.exp[,1:7],
+                              msa = fasta.mltp,grid = FALSE,
+                              legend = FALSE,scale = FALSE)
```

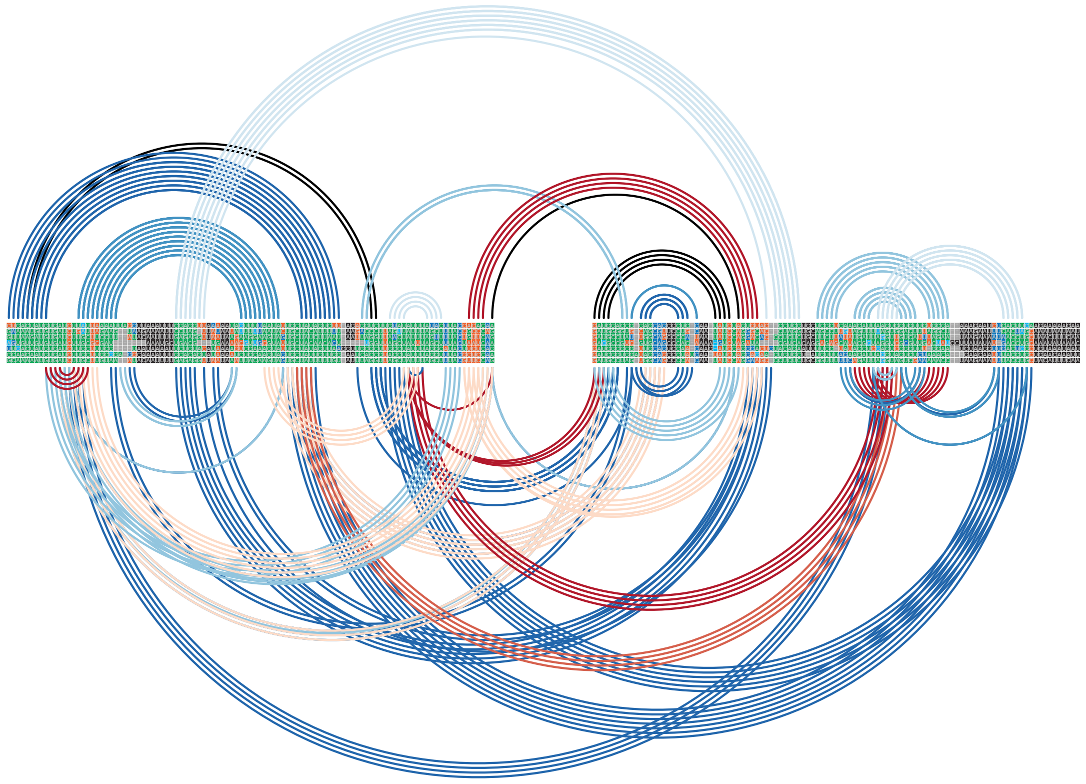


1.8 Overlapping Multiple Structures with Multiple Sequence Alignments

Overlapping Multiple Structures can be presented with msa and have msa annotation. There is 3 functions for it. They follow same rules as usual comparison does. Predicted basepairs that exist in the known structure are drawn above the line, and those predicted that are not known to exist are drawn below. Those known but unpredicted are shown in black above the line.

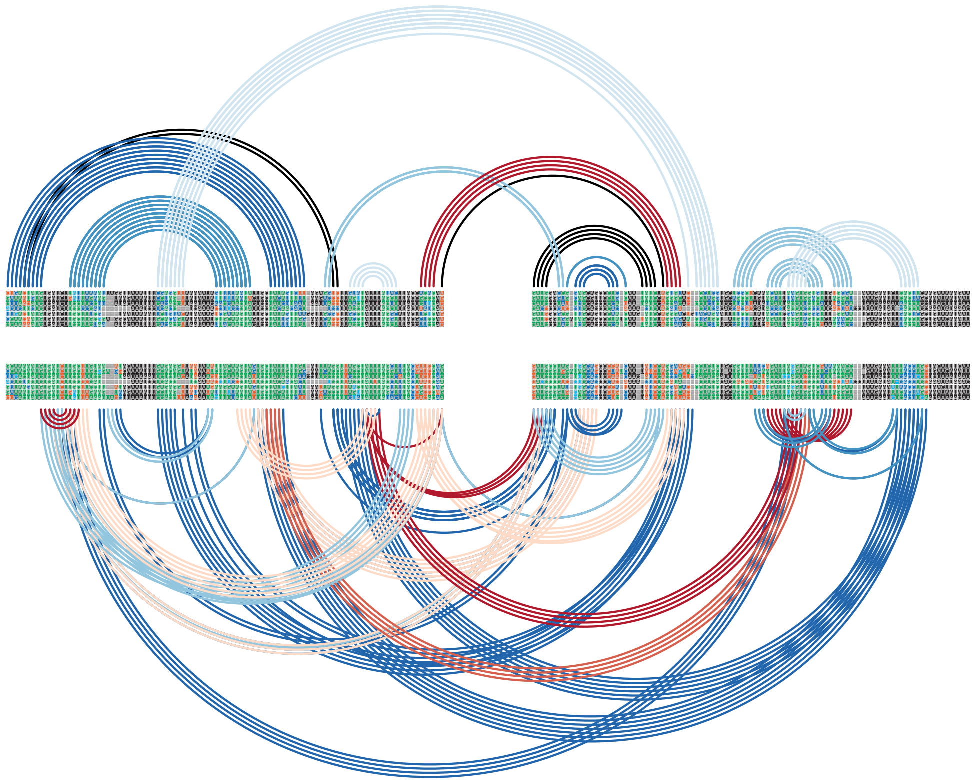
Single Line mode where upper and bottom part shares same msa visualization.

```
> plotCovarianceComparisonMltpSingleLine(msa = fasta.mltp,
+                                        helix1 = helix.known.exp[,1:6],
+                                        helix2 = helix.transat.exp[,1:6],
+                                        scale = FALSE,legend = FALSE,
+                                        species = 0)
>
```



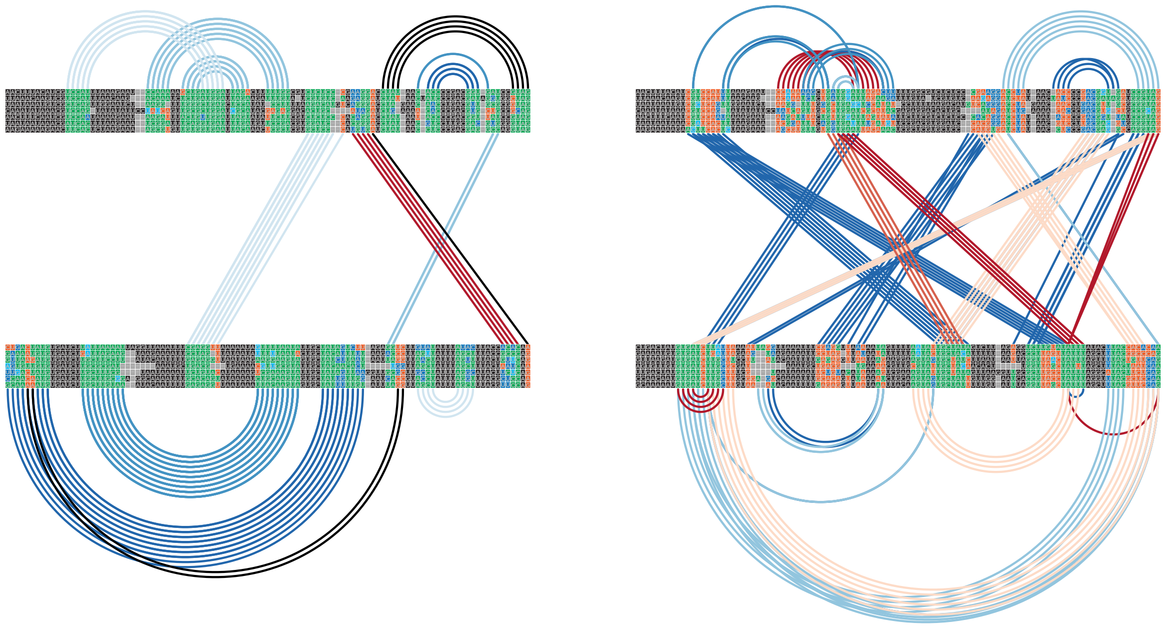
Single Line mode where upper and bottom part have their own msa visualization.

```
> plotDoubleCovarianceComparisonMltpSingleLine(msa = fasta.mltp,
+                                              helix1 = helix.known.exp[,1:6],
+                                              helix2 = helix.transat.exp[,1:6],
+                                              scale = FALSE,legend = FALSE,
+                                              species = 0,dist.y.between = 5 )
```



Double Line mode where predicted basepairs that exist in the known structure are drawn on left side, and those predicted that are not known to exist are drawn on right side. Those known but unpredicted are shown in black on left side.

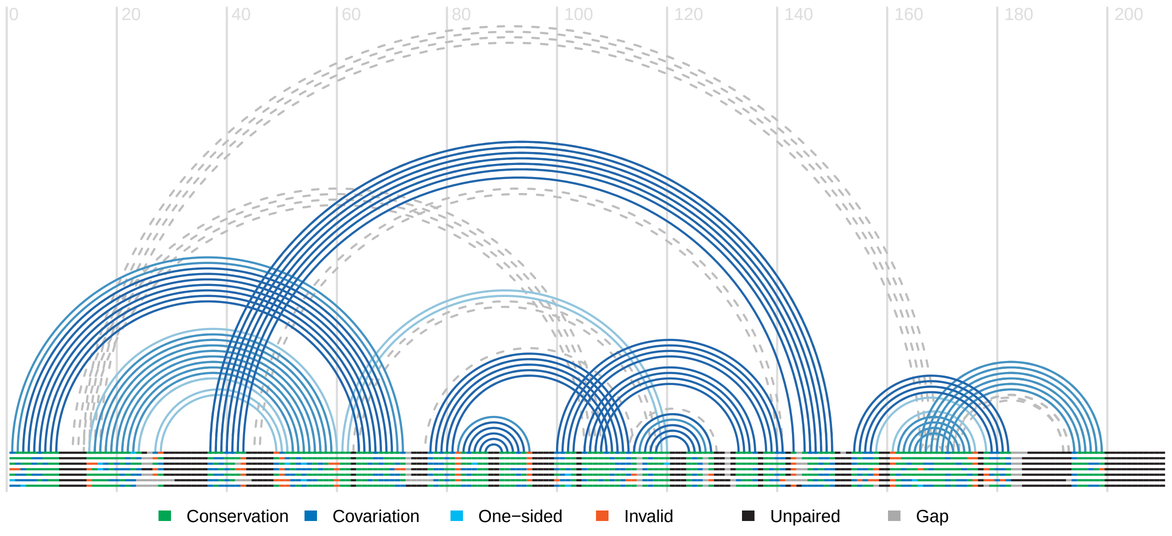
```
> plotCovarianceComparisonMltpDoubleLine(msa = fasta.mltp,
+                                        helix1 = helix.known.exp[,1:6],
+                                        helix2 = helix.transat.exp[,1:6],
+                                        scale = FALSE,legend = FALSE,
+                                        species = 0)
```



1.9 Multiple Sequence Alignements with Annotated Arcs

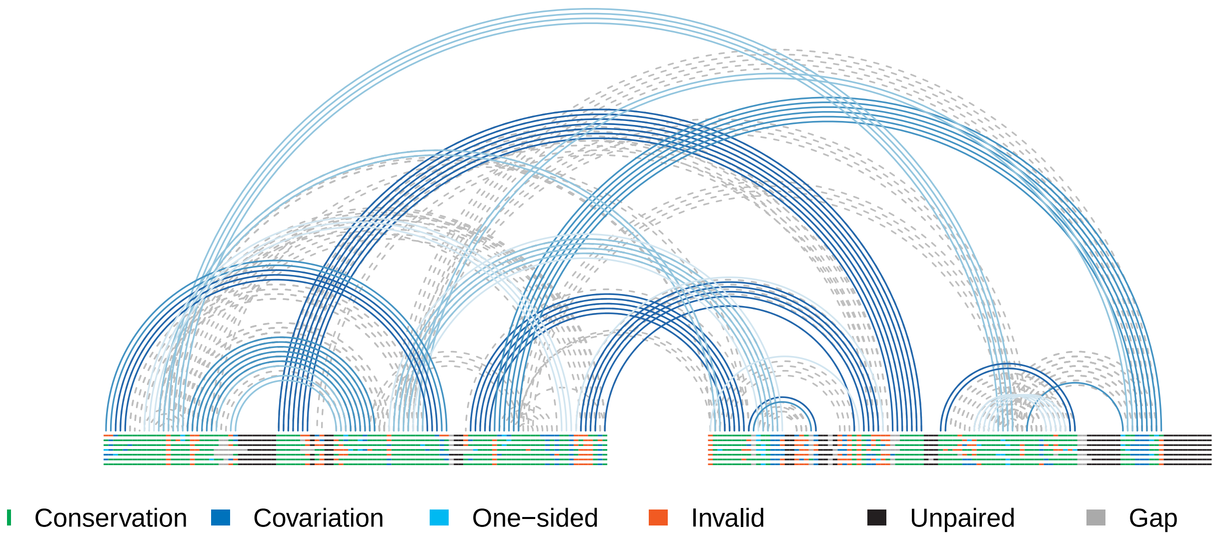
Arcs can be coloured as usual. It should be noted that structures with conflicting basepairs (arcs sharing a base) cannot be visualized properly on a multiple sequence alignment, and are typically filtered out (*e.g.* drawn in grey here).

```
> plotCovariance(fasta, transat, cex = 0.5, conflict.col = "grey")
```

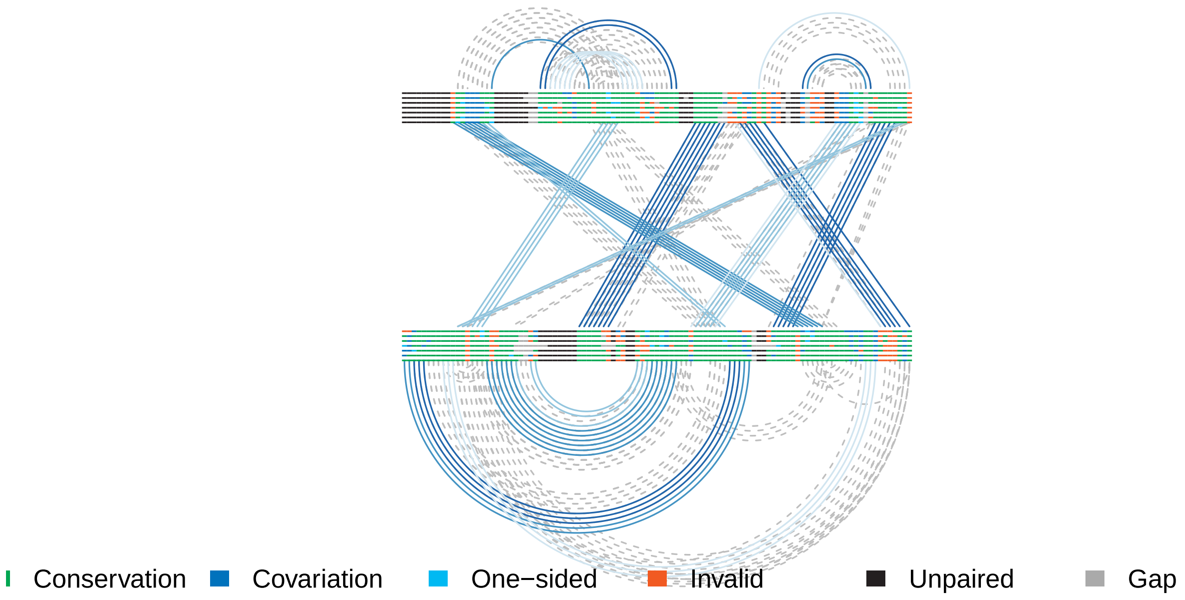


Same can be done to multiple entitites.

```
> message("Multiple sequence alignment of interest")
> fasta_file.mltp <- system.file("extdata", "fasta.mltp.txt", package = "R4RNA")
> fasta.mltp <- readBStringSet(fasta_file.mltp)
> message("Plot covariance in alignment")
> plotCovarianceMltpSingleLine(helix = helix.transat.exp[,1:7],msa = fasta.mltp,
+                              conflict.col = "grey",legend = TRUE,grid = FALSE,scale = FALSE)
```



```
> message("Multiple sequence alignment of interest")
> fasta_file.mltp <- system.file("extdata", "fasta.mltp.txt", package = "R4RNA")
> fasta.mltp <- readBStringSet(fasta_file.mltp)
> message("Plot covariance in alignment")
> plotCovarianceMltpDoubleLine(helix = helix.transat.exp[,1:7],msa = fasta.mltp,
+                              conflict.col = "grey",grid = FALSE,scale = FALSE)
```

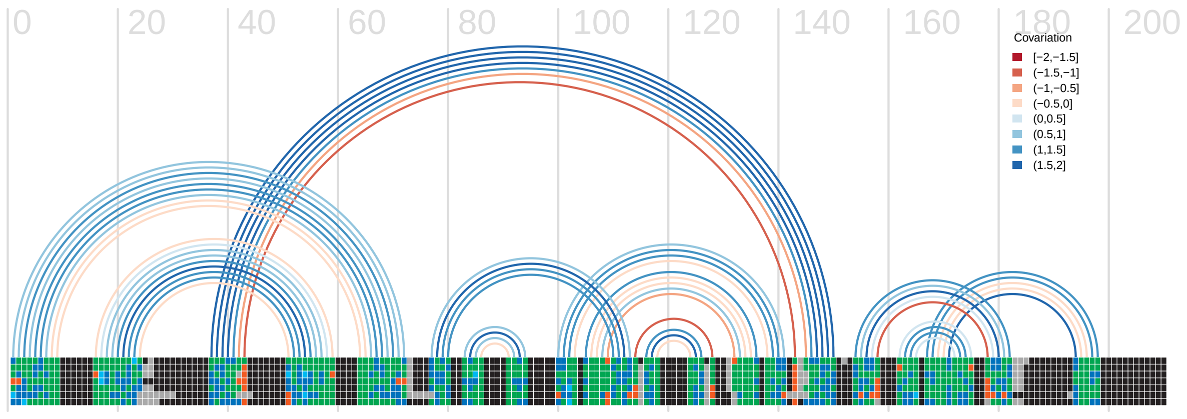


1.10 Additional Colouring Methods

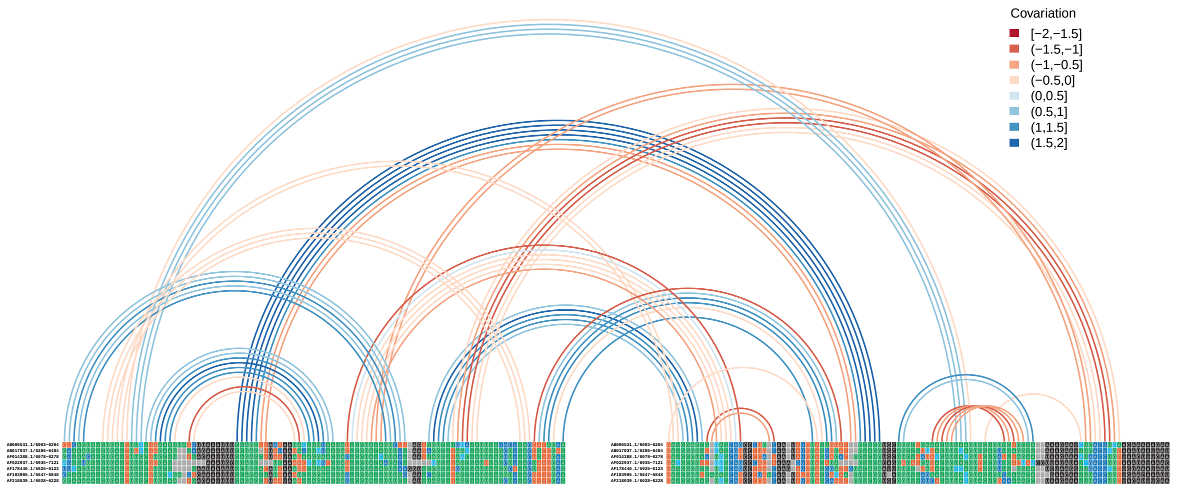
Various other methods of colour arcs exist, along with many options to control appearances:

1.10.1 Colour By Covariation (with alignment as blocks)

```
> col <- colourByCovariation(known, fasta, get = TRUE)
> plotCovariance(fasta, col, grid = TRUE, legend = FALSE)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+     inset = 0.1, bty = "n", border = NA, cex = 0.37, title = "Covariation")
```

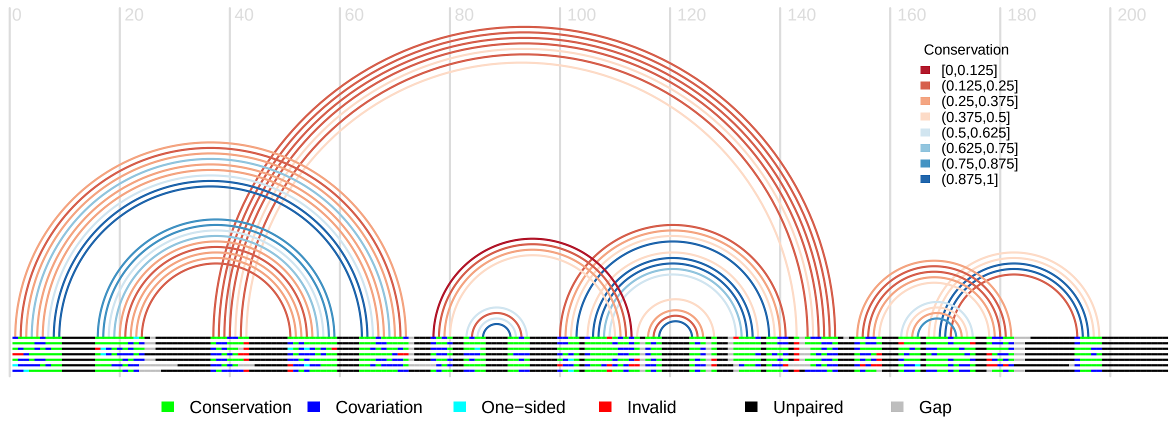


```
> col <- colourByCovariationMltp(helix = helix.transat.exp,msa = fasta.mltp,get = TRUE)
> plotCovarianceMltpSingleLine(helix = col,msa = fasta.mltp,grid = TRUE,scale = FALSE)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+        inset = 0.1, bty = "n", border = NA, cex = 0.5, title = "Covariation")
```

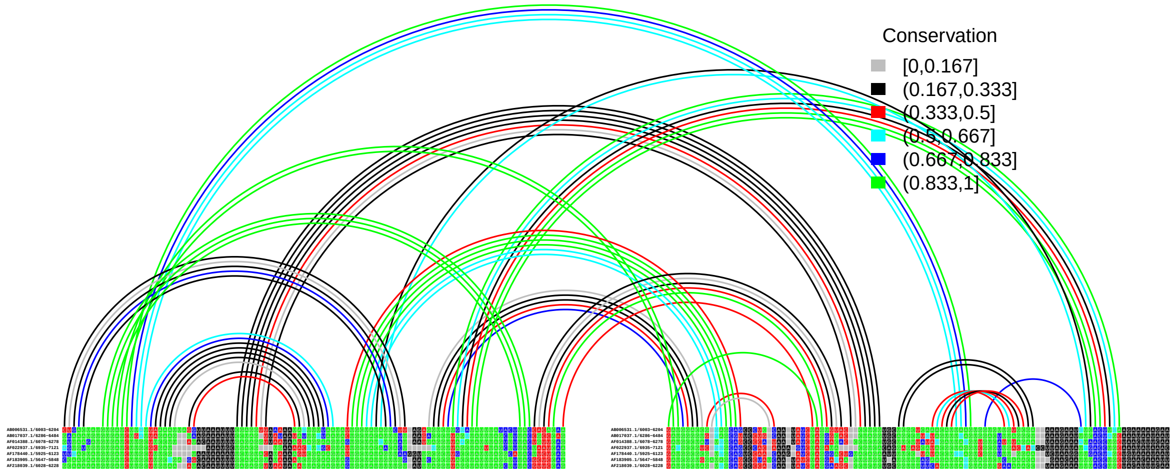


1.10.2 Colour By Conservation (with custom alignment colours)

```
> custom_colours <- c("green", "blue", "cyan", "red", "black", "grey")
> plotCovariance(fasta, col <- colourByConservation(known, fasta, get = TRUE),
+     palette = custom_colours, cex = 0.5)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+     inset = 0.15, bty = "n", border = NA, cex = 0.75, title = "Conservation")
```

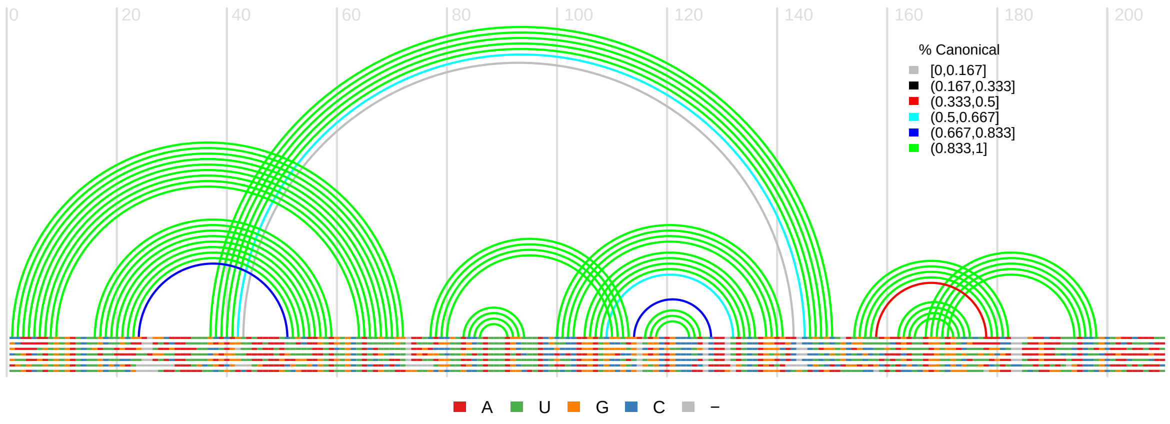


```
> custom_colours <- c("green", "blue", "cyan", "red", "black", "grey")
> plotCovarianceMltpSingleLine(msa = fasta.mltp,
+                              helix = col <- colourByConservationMltp(helix = helix.transat.exp,
+                                                                      msa = fasta.mltp,get = TRUE,
+                                                                      cols = custom_colours),
+                              palette = custom_colours,scale = FALSE)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+        inset = 0.15, bty = "n", border = NA, cex = 0.75, title = "Conservation")
```

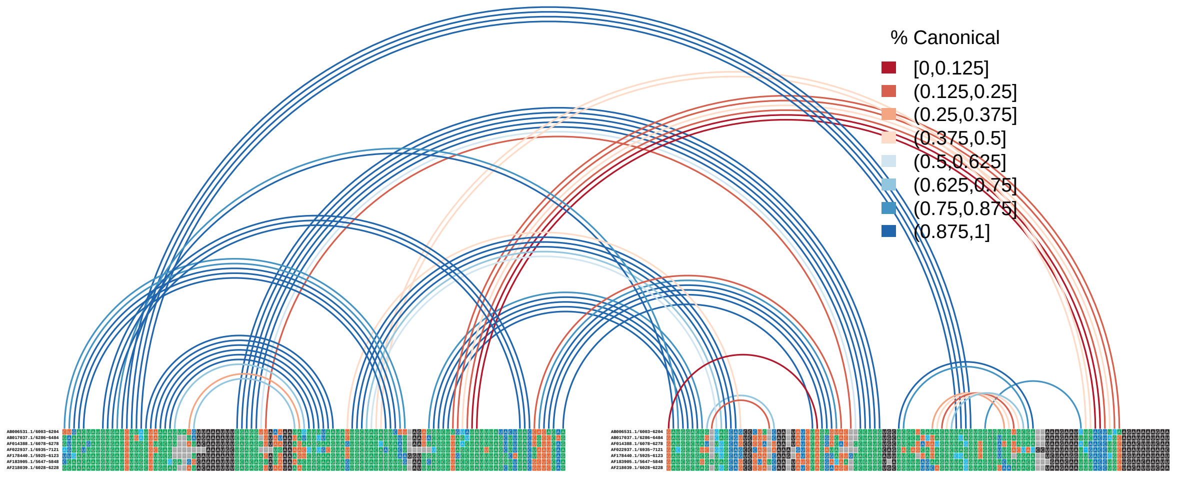


1.10.3 Colour By Percentage Canonical Basepairs (with custom arc colours)

```
> col <- colourByCanonical(known, fasta, custom_colours, get = TRUE)
> plotCovariance(fasta, col, base.colour = TRUE, cex = 0.5)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+     inset = 0.15, bty = "n", border = NA, cex = 0.75, title = "% Canonical")
```

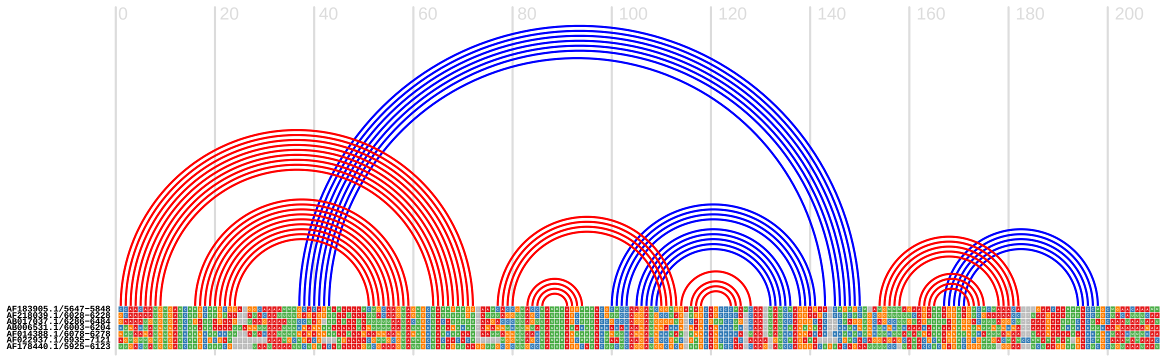


```
> col <- colourByCanonicalMltp(helix = helix.transat.exp,msa = fasta.mltp,get = TRUE)
> plotCovarianceMltpSingleLine(helix = col,msa = fasta.mltp,scale = FALSE)
> legend("topright", legend = attr(col, "legend"), fill = attr(col, "fill"),
+         inset = 0.15, bty = "n", border = NA, cex = 0.75, title = "% Canonical")
```

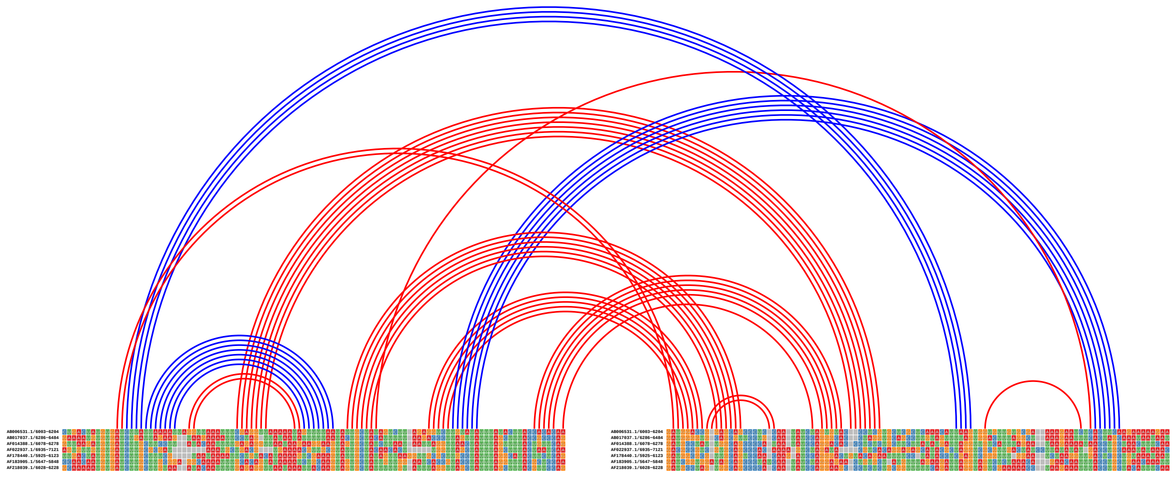


1.10.4 Colour Pseudoknots (with CLUSTALX-style alignment)

```
> col <- colourByUnknottedGroups(known, c("red", "blue"), get = TRUE)
> plotCovariance(fasta, col, base.colour = TRUE, legend = FALSE,
+                species = 23, grid = TRUE, text = TRUE,
+                text.cex = 0.2, cex = 0.5)
```



```
> col <- colourByUnknottedGroupsMltp(helix = helix.transat.exp,cols = c("red","blue"),get = TRUE)
> plotCovarianceMltpSingleLine(helix = col,msa = fasta.mltp,base.colour = TRUE,scale = FALSE)
```



1.11 Working with HiC data

HiC triangular format file obtained with straw

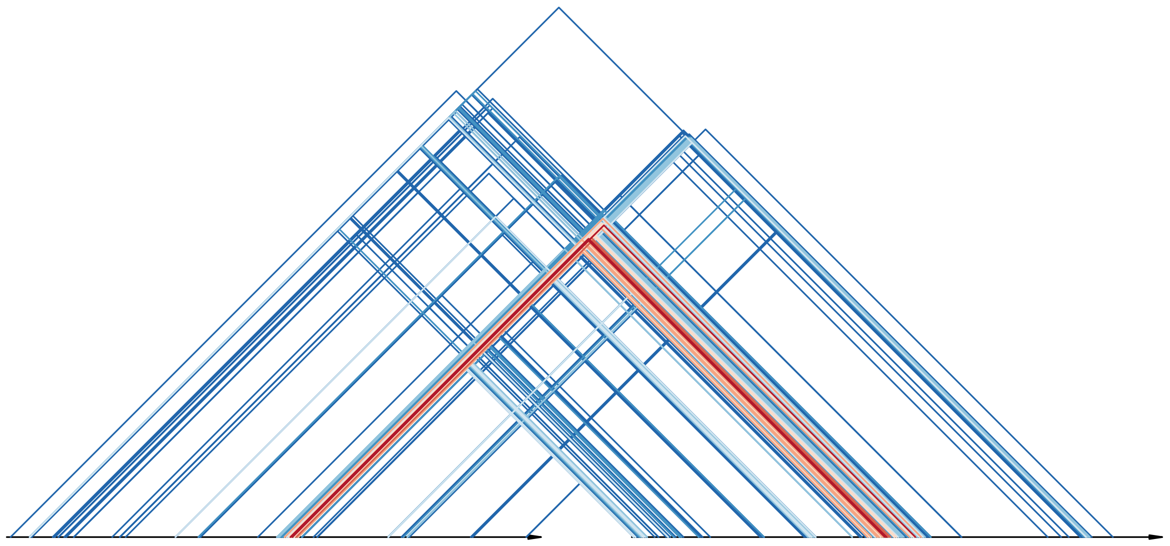
```
> hic_file <- system.file("extdata", "GSE63525.chr10.chr11.500kb.txt", package = "R4RNA")
> helix_hic <- StrawToHelix(file = hic_file, chr1 = "chr10",chr2 = "chr11",scale = 500000)
```

Trimm low values of strength interactions. After update attribute for new generated helix file. And colour by value (interaction strength).

```
> helix_hic.trimmed <- helix_hic[value > 40]
> attr(helix_hic.trimmed,"length") <- attr(helix_hic,"length")
> helix_hic.trimmed <- colourByValueMltp(helix_hic.trimmed,get = TRUE)
```

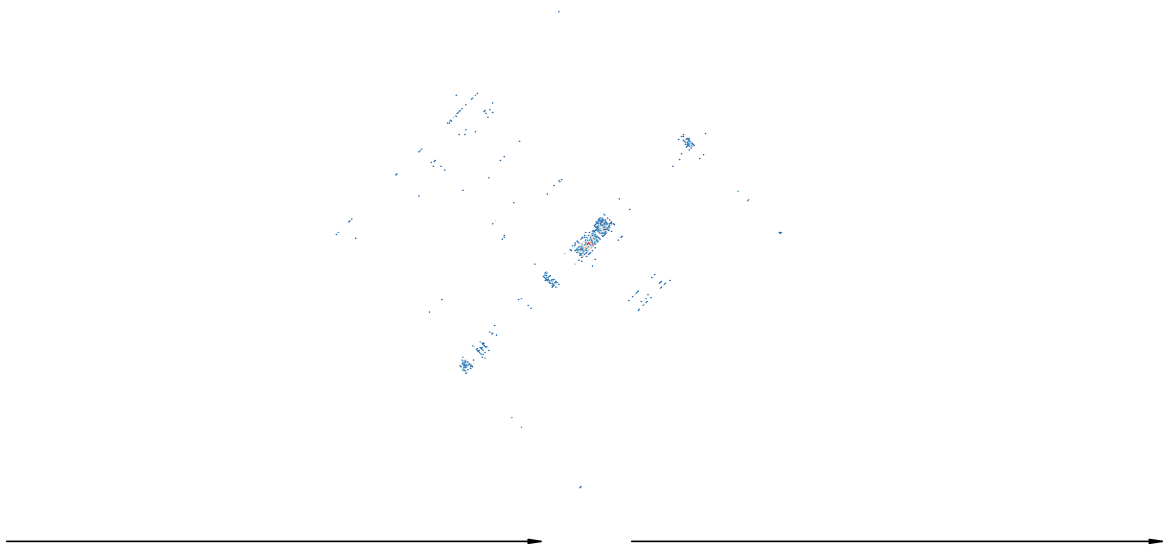
Plot figure from filtered data

```
> plotHelixMltpSingleLine(helix = helix_hic.trimmed,shape = "triangle",scale = FALSE)
```



Plot figure from filtered data in heatmap way

```
> plotHelixMltpSingleLine(helix = helix_hic.trimmed,shape = "heatmap",scale = FALSE)
```



2 Session Information

The version number of R and packages loaded for generating the vignette were:

- R version 4.1.1 (2021-08-10), `x86_64-w64-mingw32`
- Locale: `LC_COLLATE=C`, `LC_CTYPE=Russian_Russia.1251`, `LC_MONETARY=Russian_Russia.1251`, `LC_NUMERIC=C`, `LC_TIME=Russian_Russia.1251`

- Running under: `Windows 10 x64 (build 19043)`
- Matrix products: default
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils
- Other packages: BiocGenerics 0.38.0, Biostrings 2.60.2, GenomeInfoDb 1.28.1, IRanges 2.26.0, R4RNA 2.0.8, S4Vectors 0.30.0, XVector 0.32.0, data.table 1.14.0
- Loaded via a namespace (and not attached): GenomeInfoDbData 1.2.6, RCurl 1.98-1.3, bitops 1.0-7, compiler 4.1.1, crayon 1.4.1, knitr 1.33, rstudioapi 0.13, tools 4.1.1, xfun 0.25, zlibbioc 1.38.0