The HybRIDS User Manual

Ben J. Ward

2014-12-07

Contents

This manual is provided as a vignette to the HybRIDS package for R. It can be acessed from an R session with with following:

```
vignette("HybRIDS_user_manual")
```

Installing HybRIDS

To install HybRIDS from the GitHub repository executing the following code - snippet into an R session should get HybRIDS installed with minimum difficulty:

```
install.packages("devtools", dep=T)
library(devtools)
install_github("Ward9250/HybRIDS", build_vignettes=TRUE)
```

This installs a package by R community contributor Hadley Wickham which allows you to install R packages from their GitHub Repositories with minimum fuss.

Now you have the latest version of HybRIDS installed in your R library. By default the above command installs the master branch of the HybRIDS repository. This is the "safest" option by which we mean it is the version we are by far most confident in to contain no errors or bugs. Anything new that is written for HybRIDS first is written in it's own branch, before being pulled to the devel branch. When we are then sure this is stable it is merged into the master branch.

You may wish to use some of the devel features before they are merged into the master branch. In this case you would replace the last line of code in the box above with the following:

```
install_github("Ward9250/HybRIDS", ref="devel")
```

Starting HybRIDS and Loading Data

To get started fire up the R console and enter:

```
library(HybRIDS)
```

```
##
## TACTTTGTACCTAAGTATGCATTACGTTACGTTAGTAGCTGGACCTAGTAAATCGGA
## ,--.  ,--.          ,--.   ,------. ,--.,------.   ,---.
## |  '--'  |,--. ,--.|  |-. |  .--. '|  ||  .-.  \ '   .-'
## |  .--.  | \  '  / | .-. '|  '--'.'|  ||  |  \  :`.  `-.
## |  |  |  |  \    '  | `-' ||  |\  \ |  ||  '--'  /.-'    |
## `--'  `--'.-'  /    `---' `--' '--'`--'`-------' `-----'
##           `---'
## ATGAAACATGGATTCATACG - Version 1.0 - CGACCTGGATCATTTAGCCT
##
##
## Hybridisation, Recombination and Introgression Detection
##                     and Dating Package.
##
## -----------------=========****=========------------------
## Cite: TBD
## Licence: GPL (Like R and most packages).
## http://ward9250.github.io/HybRIDS/
## -----------------=========****=========------------------
```

To get started using HybRIDS to analyse a set of sequences, you first have to create a new HybRIDS object.

The HybRIDS object will contain all data, options, and will carry out the analysis. So everything you do in HybRIDS is done by interacting with this object. By providing a file-path to a FASTA format sequence file when the new object is created the DNA data will also be loaded and prepared automatically. Base positions of the alignment which are non-informative, or positions with 'N's will be excluded.

```
MyAnalysis <- HybRIDS$new("~/Dropbox/MySequences.fas")
```

With the above line a new HybRIDS object would be created and assigned to a variable "MyAnalysis".

The HybRIDS package has been loaded, and data has been read in and is now contained in a HybRIDS object, which also contains all analysis settings and steps.

You can also create a new HybRIDS object from sequences you already have loaded into R as a `DNAbin` object (the DNA type used in the `ape` package). The HybRIDS package comes with an example datasets that is used in this user manual. It can be accessed with the `data` command after loading the HybRIDS package.

```
data(MySequences)
MyAnalysis <- HybRIDS$new(MySequences)
```

```
## Class of input is DNAbin from ape package.
## Looking for duplicates (sequences with p_distances of 0)...
## Done...
## Subsetting the informative segregating sites...
## Finished DNA input.
## Generating triplets to find recombination in sequences, between partitions.
## Initializing new triplets data.
```

Viewing and editing analysis settings

When scripting with a HybRIDS object, you execute several methods that make the HybRIDS object perform several steps of the analysis. Each analysis step is performed according to its settings. Before we walk through executing the analysis steps in detail, it's important to know how to view and alter the settings of the steps.

Viewing HybRIDS current settings

To view the settings and state of your current HybRIDS object, you just need to enter the name of the variable you used to assign the HybRIDS object, in the case of this example, the HybRIDS object created was assigned to the name `MyAnalysis`

```
MyAnalysis
```

```
## HybRIDS object:
##
## DNA Sequence Information:
## ------------------------
## An alignment of 10 sequences.
##
## Full length of alignment: 400000
## Excluding non-informative sites: 33043
##
## Sequence names:
## 1: Seq1
## 2: Seq2
## 3: Seq3
## 4: Seq4
## 5: Seq5
```

```
## 6: Seq6
## 7: Seq7
## 8: Seq8
## 9: Seq9
## 10: Seq10
##
##
## Settings for Sequence Scan Combinations:
## ----------------------------------------
## Triplet Generation Method (Method): 1
##
## Sequences are organized according to the following groups:
##
##
## Distance Threshold for excluding triplets with
##  too similar sequences (DistanceThreshold): 0.01
##
## How many sequences from the same partition are
##  allowed in a triplet (PartitionStrictness): 2
##
## A total of 120 triplets will be compared.
##
##
## Settings for sliding window scan of recombination signal:
## ---------------------------------------------------------
## Size of the sliding window in base pairs (WindowSize): 100
##
## Number of base pairs to move the sliding window on
##  each iteration of the scan (StepSize): 1
##
##
## Settings for detecting blocks from recombination signal:
## --------------------------------------------------------
## Manual sequence similarity thresholds (ManualThresholds): 90
##
## Automatically decide on thresholds (AutoThresholds): TRUE
##
## Fall back to manual thresholds (ManualFallback): TRUE
##
## Standard deviation divisor (SDStringency): 2
##
##
## Settings for testing and dating recombination blocks:
## -----------------------------------------------------
## Assumed substitution rate for dating (MutationRate): 1e-08
##
## Critical alpha value for significance testing (PValue): 0.005
##
## Apply bonferroni correction to critical alpha (BonfCorrection): TRUE
##
## Keep and date blocks that fail the alpha (DateAnyway): FALSE
##
## Assumed mutation model for dating (MutationCorrection): JC69
##
```

```
## Plotting Settings are not shown because of the number of settings.
## use showParameters('Plotting') to view them.
```

A text summary printed like that above which shows the settings for each analysis step.

Alternatively, to view the settings of only a single step, the `showParameters` method can be used:

```
MyAnalysis$showParameters("TripletGeneration")
```

```
## Settings for Sequence Scan Combinations:
## ----------------------------------------
## Triplet Generation Method (Method): 1
##
## Sequences are organized according to the following groups:
##
##
## Distance Threshold for excluding triplets with
##  too similar sequences (DistanceThreshold): 0.01
##
## How many sequences from the same partition are
##  allowed in a triplet (PartitionStrictness): 2
##
## A total of 120 triplets will be compared.
```

Changing the analysis settings

The settings of each analysis step are altered in a consistent way, using the same method of the HybRIDS object.

To change the settings for any given step of the HybRIDS analysis, you use the `setParameters` method.

This method accepts first a word that dictates which analysis step it is you want to change:

1. `TripletGeneration` - Settings for Sequence Scan Combinations.
2. `SSAnalysis` - Settings for the sequence similarity scan step.
3. `BlockDetection` - Settings for detecting recombinant blocks from sequence similarity scan data generated from the sequence similarity scan step.
4. `BlockDating` - Settings for calculating the significance values and divergence time estimates, for recombinant blocks detected in the `BlockDetection` step.
5. `Plotting` - Settings for generating figures of the recombination signal in one or more triplets.

In addition the method is then provided a series of name, value pairs where a name is one of the settings, and the value is the value to change it to.

To use an example, we know from the text summary printed for the HybRIDS object above, that the `TripletGeneration` step has several settings: `Method`, `DistanceThreshold`, and `PartitionStrictness`.

(Don't worry about what these settings do right now, they are described more fully in the next manual sections - although the nice text summary gives you a clue e.g. `DistanceThreshold` is a *Distance Threshold for eliminating triplets with too similar sequences*)

Let's say we wanted to increace the `DistanceThreshold` parameters from 0.01 to 0.1. We can do that by calling the `setParameters` method:

```
MyAnalysis$setParameters("TripletGeneration", DistanceThreshold = 0.1)
```

```
## Generating triplets to find recombination in sequences, between partitions.
## Deleting all triplets data.
## Initializing new triplets data.
```

Multiple settings could be edited too, for example:

```
MyAnalysis$setParameters("TripletGeneration", DistanceThreshold = 0.1, PartitionStrictness = 1)
```

To summarize: Each step of a HybRIDS analysis has settings associated with them, but the settings of each step are all easily changed by calling the `setParameters` method of the HybRIDS object.

In the next sections, each step of the HybRIDS analysis, and how to execute them, will be described.

Specifying sequence triplets

The first thing you need to do after creating the HybRIDS object for your alignment file, is inform the HybRIDS object which triplets for your sequences you would like to analyze for evidence of recombination.

The following methods of generating triplets are provided denoted by the integers 1, 2 and 3, and are selected by changeing the `Method` parameter:

1 (DEFAULT) - HybRIDS will analyze sequence triplets based on a set of specified groups (defined by the `Groups` parameter, such that triplets analyzed will be made up of sequences to try and find recombination events between the groups.

A group is any given subset of sequences from your input file. For example, the example anaysis has 10 DNA sequences. The first 3 sequences, might be from one population, the next three from a second population, and the remaining sequences from a third population.

If you wanted to find evidence of recombination between these populations. You would want to analyze triplets made of e.g. The first sequence, the fourth sequence, and the seventh sequence, or e.g. the second sequence, the fourth sequence, and the ninth sequence. With this default method HybRIDS will figure out all the possible triplets that need to be analyzed that might find evidence of recombination between the populations or Groups.

If this method of triplet generation is set, as it is by default, and no groups are provided, the HybRIDS object will analyze every possible combination of three of your provided sequences.

2 HybRIDS will analyze sequences triplet based on how similar the sequences that make the triplet are to one another. Every possible triplet of your provided sequences will be analyzed, providing that all the pairwise distances in a triplet are above a set threshold, and this threshold is defined by the `DistanceThreshold` parameter.

3 HybRIDS will analyze triplet based on sequence similarity as in method 2, however instead of using the `DistanceThreshold` parameter, a threshold is automatically chosen based on the distribution of pairwise distances for your provided alignment.

Now you know what this step does and how each of the settings for this step affect it, the example demonstrates how to appropriately edit the settings of this step.

In this example, there is an alignment of 10 sequences loaded in the HybRIDS object, and a researcer has collected the sequences from three populations/locations. Sequences one to three are from one location, sequences four to six are from the second location, and sequences 7 to 10 are from the final population or location.

First the sequences are loaded into a HybRIDS object as demonstrated in previous sections and the summary is printed to the console:

```
library(HybRIDS)
MyAnalysis <- HybRIDS$new("~/Dropbox/MySequences.fas")
```

```
## File to be read is expected to be FASTA format...
## Reading in sequence file...
## Looking for duplicates (sequences with p_distances of 0)...
## Done...
## Subsetting the informative segregating sites...
## Finished DNA input.
## Generating triplets to find recombination in sequences, between partitions.
## Initializing new triplets data.
```

```
MyAnalysis$showParameters("TripletGeneration")
```

```
## Settings for Sequence Scan Combinations:
## ----------------------------------------
## Triplet Generation Method (Method): 1
##
## Sequences are organized according to the following groups:
##
##
## Distance Threshold for excluding triplets with
##  too similar sequences (DistanceThreshold): 0.01
##
## How many sequences from the same partition are
##  allowed in a triplet (PartitionStrictness): 2
##
## A total of 120 triplets will be compared.
```

We can see from the print-out that 120 triplets will be analyzed - this is because we have speified no groups but are using method 1. As a result, ever possible triplet that could be generated for these sequences will be analyzed.

Now the settings can be changed to add the groups between which we wish to find recombination, and edit the `PartitionStrictness` so as only one sequence from each group is allowed in a triplet.

In order to specify a group, you simply make a vector of the sequence names for each group, and then combine each one into a list:

```
# Let's make our groups:

FirstGroup <- c("Seq1", "Seq2", "Seq3")
SecondGroup <- c("Seq4", "Seq5", "Seq6")
ThirdGroup <- c("Seq7", "Seq8", "Seq9", "Seq10")
myGroups <- list(FirstGroup, SecondGroup, ThirdGroup) # Put the groups in a list.
```

```
# Let's edit the Triplet Generation settings
MyAnalysis$setParameters("TripletGeneration", Groups = myGroups, PartitionStrictness = 1L)
```

```
## Generating triplets to find recombination in sequences, between partitions.
## Deleting all triplets data.
## Initializing new triplets data.
```

```
# All done, now let's print the settings again:
MyAnalysis$showParameters("TripletGeneration")
```

```
## Settings for Sequence Scan Combinations:
## ----------------------------------------
## Triplet Generation Method (Method): 1
##
## Sequences are organized according to the following groups:
## c("Seq1", "Seq2", "Seq3"),
## c("Seq4", "Seq5", "Seq6"),
## c("Seq7", "Seq8", "Seq9", "Seq10")
##
## Distance Threshold for excluding triplets with
##  too similar sequences (DistanceThreshold): 0.01
##
## How many sequences from the same partition are
##  allowed in a triplet (PartitionStrictness): 1
##
## A total of 36 triplets will be compared.
```

Note in the above example, that `1L` is used and not `1` when setting the `PartitionStrictness` as `1L` specifies the value is a whole number integer and not the floating point `1.0`.

We see after altering these settings, less triplets will be analyzed now, and each one will contain only one sequence from each group.

It is also possible to combine method 1 with either method 2 or 3, thereby only generating triplets that contain A). One sequence from each group, and B). Sequences that are all sufficiently diverged.

```
MyAnalysis$setParameters("TripletGeneration", Method = c(1L, 2L),
                         Groups = myGroups, PartitionStrictness = 1L)
```

```
## Generating triplets to find recombination in sequences, between partitions.
## Deleting all triplets data.
## Initializing new triplets data.
```

Now that you have instructed the HybRIDS object on which triplets will be analyzed, the recombination signal in each triplet can be scanned. This is explained in the next section.

Scanning triplets for recombination signal

This step will use a sliding window to measure pairwise sequence similarity across the informative sites of sequence triplets. It is started by using the `analyzeSS()` method of the HybRIDS object. The parameters of the step are:

WindowSize The size in bases of the Sliding Window. The default size is 100.

StepSize The number of bases that the sliding window moves across sequences. The default size is 1.

The `analyzeSS` method also accepts an argument which tells it which triplets (of the set you have defined in the `TripletGeneration` step) are to be scanned with the current `SSAnalysis` settings.

This argument can be set as `"NOT.SCANNED"` (default) in which case, every triplet of the set defined by the `TripletGeneration` parameters will be scanned for recombination signal with the current `SSAnalysis` settings, if it has not previously already been scanned. This argument can also be set to `"ALL"`, in which case all the triplets defined by the `TripletGeneration` settings will be scanned for recombination signal.

If the argument is not one of these two arguments, it can be a list of vectors that each contain three sequence names.

For example just two triplets could be scanned:

```
tripletsToScan <- list(c("Seq1", "Seq4", "Seq7"), c("Seq1", "Seq4", "Seq8"))

MyAnalysis$analyzeSS(tripletsToScan)
```

```
## Scanning sequence similarity for triplet Seq1, Seq4, Seq7
## Checking the sliding window parameters...
## Making all the window frames...
## Scanning Now!
## Scanning sequence similarity for triplet Seq1, Seq4, Seq8
## Checking the sliding window parameters...
## Making all the window frames...
## Scanning Now!
## Finished Sequence Similarity Analysis.
```

It is important to note that you can't enter a triplet of names which either contains names which are not names of your sequences, or is not allowed by the `TripletGeneration` settings. This is demonstrated in the example below. Recall from the previous section that the sequences "Seq1", "Seq2", and "Seq3" were defined as being in the same group, and because of the settings, only triplets that compared sequences from different groups were set. Therefore the following will not analyze any triplets, as the selected triplet does not exist.

```
# This triplet is not allowed, because earlier the TripletGeneration
# settings were set so as only triplets which try to find recombination regions between
tripletsToScan <- list(c("Seq1", "Seq2", "Seq3"))
MyAnalysis$analyzeSS(tripletsToScan)
```

```
## Finished Sequence Similarity Analysis.
```

Detection of recombination blocks in signal

Having scanned for recombination signal for the desired sequence triplets, the `BlockDetection` step is executed using the `findBlocks` method.

This step identifies regions of recombination in a given sequence triplet by examining the data from the sliding window scans in the previous `SSAnalysis` step. If there are any regions in which two sequences meet a certain sequence similarity threshold, they are treated as potentially recombinant.

The parameters for this step are:

ManualThresholds A vector of numbers between 0 and 100.

Each is a percenage sequence similarity threshold.

These thresholds are set by the user, and can be used to find recombination,

or used as a fallback if HybRIDS fails to automatically decide on some thresholds.

AutoThresholds A logical argument (`TRUE`, or `FALSE`), if `TRUE` (default) then during execution of the `findBlocks` method,

thresholds will be decided on automatically. If `FALSE` then the `ManualThresholds` will be used.

ManualFallback A logical argument (`TRUE`, or `FALSE`), if `TRUE` (default), then if automatic thresholds cannot be found, the

`ManualThresholds` will be used instead.

SDstringency A numeric value, the higher it is, the closer automatically detected thresholds are allowed to be to the mean

sequence similarity across the sequences. It is usually fine to leave this as 2.

Like the `analyzeSS` method in the previous section, the `findBlocks` must be given an argument that dictates which triplets of the set defined by the `TripletGeneration` parameters will have blocks found from their scan data. By default this is set to `"NOT.SEARCHED"` which will find blocks for triplets which have not already had blocks identified from their scan data. This argument can also be set to `"ALL"` or a list of vectors that each contain three sequence names.

The below example finds blocks in the two triplets that were scanned for recombination signal in the previous section:

```
tripletsToSearch <- list(c("Seq1", "Seq4", "Seq7"), c("Seq1", "Seq4", "Seq8"))
MyAnalysis$findBlocks(tripletsToSearch)
```

```
## Using the autodetect thresholds method...
## Deciding on suitable thresholds...
## Now beginning Block Search...
## Using the autodetect thresholds method...
## Deciding on suitable thresholds...
## Now beginning Block Search...
## Finished finding potential blocks for all triplet selections.
```

Note that, as with the `analyzeSS` method, you cannot specify a triplet which was not specified by the `TripletGeneration` settings:

```
tripletsToSearch <- list(c("Seq1", "Seq2", "Seq3"))
MyAnalysis$findBlocks(tripletsToSearch)
```

```
## Finished finding potential blocks for all triplet selections.
```

You also cannot search for blocks for triplets which have not been analyzed by `analyzeSS`:

```
tripletsToSearch <- list(c("Seq1", "Seq4", "Seq9"))
MyAnalysis$findBlocks(tripletsToSearch)
```

```
## Finished finding potential blocks for all triplet selections.
```

```
# A triplet must be scanned for recombination signal first
MyAnalysis$analyzeSS(tripletsToSearch)
```

```
## Finished Sequence Similarity Analysis.
```

```
MyAnalysis$findBlocks(tripletsToSearch)
```

```
## Finished finding potential blocks for all triplet selections.
```

If you want to get a Data Frame containing a table of the detected blocks in some triplets, use the `tabulateDetectedBlocks` method.

This method needs to be given a set of sequence names to set the triplets that will be tabulated:

```
tripletsToTabulate <- list(c("Seq1", "Seq4", "Seq7"), c("Seq1", "Seq4", "Seq8"))
MyAnalysis$tabulateDetectedBlocks(tripletsToTabulate)
```

```
## Tabulating blocks for the triplet Seq1:Seq4:Seq7
## Tabulating blocks for the triplet Seq1:Seq4:Seq8
```

```
##           Triplet Sequence_Pair Sequence_Similarity_Threshold
## 1  Seq1:Seq4:Seq7     Seq1:Seq4                            65
## 2  Seq1:Seq4:Seq7     Seq1:Seq7                            85
## 3  Seq1:Seq4:Seq7     Seq1:Seq7                            68
## 4  Seq1:Seq4:Seq7     Seq1:Seq7                            68
## 5  Seq1:Seq4:Seq7     Seq4:Seq7                            68
## 6  Seq1:Seq4:Seq8     Seq1:Seq4                            72
## 7  Seq1:Seq4:Seq8     Seq1:Seq8                            84
## 8  Seq1:Seq4:Seq8     Seq1:Seq8                            68
## 9  Seq1:Seq4:Seq8     Seq1:Seq8                            68
## 10 Seq1:Seq4:Seq8     Seq4:Seq8                            66
##    First_BP_Position Last_BP_Position Approximate_Length_BP Number_of_SNPs
## 1             287454           295896                  8443             NA
## 2             283984           287383                  3400             NA
## 3             283957           283980                    24             NA
## 4             287386           287407                    22             NA
## 5             269143           280337                 11195             NA
## 6             287463           295890                  8428             NA
## 7             283980           287386                  3407             NA
## 8             283957           283978                    22             NA
## 9             287389           287408                    20             NA
## 10            269088           280341                 11254             NA
##    Corrected_Number_of_SNPs p=0.05_Age p=0.5_Age p=0.95_Age P_Value
## 1                        NA         NA        NA         NA      NA
## 2                        NA         NA        NA         NA      NA
## 3                        NA         NA        NA         NA      NA
## 4                        NA         NA        NA         NA      NA
## 5                        NA         NA        NA         NA      NA
## 6                        NA         NA        NA         NA      NA
## 7                        NA         NA        NA         NA      NA
## 8                        NA         NA        NA         NA      NA
## 9                        NA         NA        NA         NA      NA
```

```
## 10                       NA         NA         NA         NA      NA
##    P_Thresh
## 1        NA
## 2        NA
## 3        NA
## 4        NA
## 5        NA
## 6        NA
## 7        NA
## 8        NA
## 9        NA
## 10       NA
```

Note that some columns are NA because they are produced from the next step, which is discussed in the next section.

Testing and dating detected blocks

Having searched for blocks in the desired sequence triplets, the `BlockDating` step is executed using the `dateBlocks` method.

This step assigns a significance value to blocks based on the size of a block, the number of mutations observed, and the binomial probability distribution. Afterwards, 95%CI regions for the divergence time of the blocks are calculated, assuming a mutational model and a strict molecular clock.

The parameters for this step are:

MutationRate The substitution rate assumed when estimating the ages of blocks. By default it is 10e-9.

PValue The critical alpha value when testing the significance values of blocks. By default is is 0.05.

BonfCorrection Logical (`TRUE` / `FALSE`) when `TRUE` (default) the `PValue` will be adjusted with a Bonferroni correction.

DateAnyway Logical (`TRUE` / `FALSE`) when `TRUE` (`FALSE` by default), all detected blocks will be kept and dated.

When `FALSE`, blocks that do not pass the critical alpha when testing for significance are not kept in the results

and are not dated.

MutationCorrection Set to one of "raw", "TS", "TV", "JC69", "K80", "F81", "K81", "F84", "BH87", "T92", "TN93", "GG95".

Base frequencies are calculated from the recombinant block if they are required by the model.

As was the case with the `analyzeSS` and `findBlocks` methods, `dateBlocks` needs a first argument that dictates which triplets of the set defined by the `TripletGeneration` parameters will have their blocks tested and dated. By default this is `"NOT.DATED"`, which means all triplets with blocks detected, that have not been tested or dated, will have their detected blocks tested and dated witht the current settings for the step. This can be set to `"ALL"`, or, as previously, can be set as a list of vectors that each contain three sequence names.

The following example finds blocks in the two triplets that were scanned for recombination signal and had blocks identified in the examples of the previous two sections:

```
tripletsToDate <- list(c("Seq1", "Seq4", "Seq7"), c("Seq1", "Seq4", "Seq8"))
MyAnalysis$dateBlocks(tripletsToDate)
```

```
## Now dating blocks
## Now dating blocks
```

The blocks can then be tabulated again, the columns of the table that were previously `NA` are now filled with values:

```
MyAnalysis$tabulateDetectedBlocks(tripletsToDate)
```

```
## Tabulating blocks for the triplet Seq1:Seq4:Seq7
## Tabulating blocks for the triplet Seq1:Seq4:Seq8

##          Triplet Sequence_Pair Sequence_Similarity_Threshold
## 1 Seq1:Seq4:Seq7     Seq1:Seq4                            65
## 2 Seq1:Seq4:Seq7     Seq1:Seq7                            85
## 3 Seq1:Seq4:Seq7     Seq4:Seq7                            68
## 4 Seq1:Seq4:Seq8     Seq1:Seq4                            72
## 5 Seq1:Seq4:Seq8     Seq1:Seq8                            84
## 6 Seq1:Seq4:Seq8     Seq4:Seq8                            66
##   First_BP_Position Last_BP_Position Approximate_Length_BP Number_of_SNPs
## 1            287454           295896                  8443             56
## 2            283984           287383                  3400             21
## 3            269143           280337                 11195             71
## 4            287463           295890                  8428             55
## 5            283980           287386                  3407             22
## 6            269088           280341                 11254             75
##   Corrected_Number_of_SNPs p=0.05_Age p=0.5_Age p=0.95_Age      P_Value
## 1                       56     414430    335424     268846 3.877419e-63
## 2                       21     443886    317432     219006 5.136120e-39
## 3                       71     386630    319937     261874 1.491423e-65
## 4                       55     410144    330115     263346 1.120252e-63
## 5                       22     461617    331466     230160 1.525899e-38
## 6                       75     402225    335793     276109 1.935040e-63
##   P_Thresh
## 1    0.005
## 2    0.005
## 3    0.005
## 4    0.005
## 5    0.005
## 6    0.005
```

Plotting the recombination signal of a triplet

To generate a graphic of the recombination signal for a given set of triplets, the `plotTriplet` method is used. This method also takes a list of vectors, each containing 3 sequence names that specifies the triplets to plot. The method returns a list containing ggplot objects of each triplet chosen to be visualized.

The parameters for plotting are as follows:

What: This value can either be "Bars", "Lines" or an R vector of both. This determines which plots HybRIDS will plot. HybRIDS produces two class of plot of a given triplet's SS similarity data (often on one canvas), a lines plot, and a coloured heatmap-like plot.

PlotTitle: Boolean, set wether to include a title with the plot.

CombinedTitle: Boolean, set whether to only have one overall title for a Lines and Bars plot on the same canvas.

TitleSize: Integer value, sets the font size of the text in the plot title.

TitleFace: A character value "bold" by default, can be changed to a face compatible with ggplot2, see documentation of ggplot2 for more details.

TitleColour: A string denoting a colour "black" by default. Can be changed to any colour notation compatible with ggplot2, see the documentation or examples of ggplot2 for more details.

XLabels: Boolean, set whether to include the value labels of the x-axis in the plots.

YLabels: Boolean, set whether to include the value labels of the y-axis in the plots.

XTitle: Boolean, set whether to include the title of the x-axis in the plots.

XTitleFontSize: Integer value, sets the font size of the x-axis title in plots.

XTitleColour: A character string denoting a colour, "black" by default. Sets the colour of the x-axis title font. Can be changed to any colour notation compatible with ggplot2, see the documentation or examples of ggplot2 for more details.

XLabelSize: Integer value, sets the font size of the x-axis labels.

XLabelColour: A character string denoting a colour, "black" by default. Sets the colour of the x-axis value labels. Can be changed to any colour notation compatible with ggplot2, see the documentation or examples of ggplot2 for more details.

YTitle: Boolean, set whether to include the title of the y-axis in the plots.

YTitleFontSize: Integer value, sets the font size of the y-axis title in plots.

YTitleColour: A character string denoting a colour, "black" by default. Sets the colour of the y-axis title font. Can be changed to any colour notation compatible with ggplot2, see the documentation or examples of ggplot2 for more details.

YLabelSize: Integer value, sets the font size of the y-axis labels.

YLabelColour: A character string denoting a colour, "black" by default. Sets the colour of the x-axis value labels. Can be changed to any colour notation compatible with ggplot2, see the documentation or examples of ggplot2 for more details.

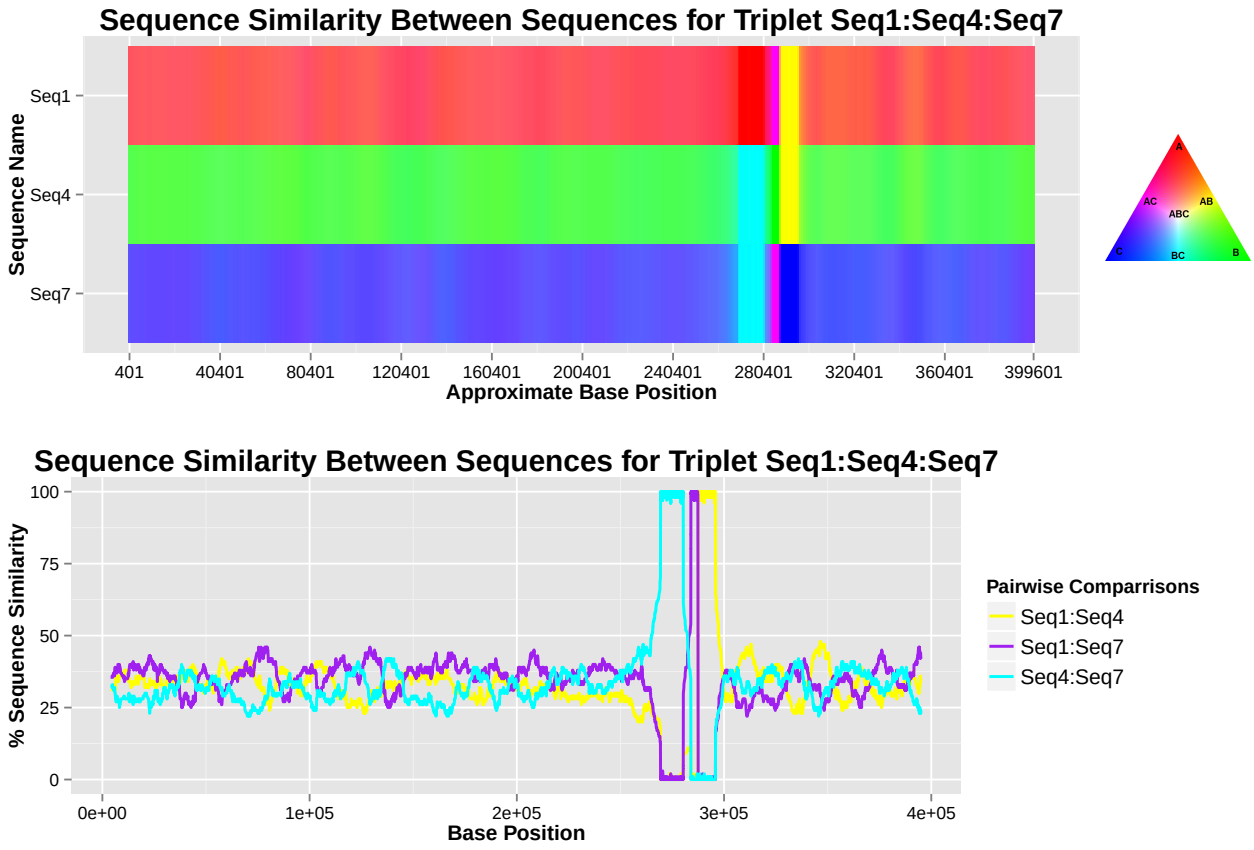
Legends: Boolean value, set's whether to include the legends of plots. TRUE by default.

LegendFontSize: An integer value, sets the legend font size.

MosaicScale: An integer value, 500 by default. This affects how the Bars plot is drawn. With a higher value, a greater resolution is achieved but more and finer sequence similarity data is required. This is achieved with sequences with high numbers of informative sites and by adjusting the WindowSize and StepSize settings for the SSAnalysis step.

HybRIDS will print a message to the console if problems arise during plot drawing due to the MosaicScale settings.

```
MyAnalysis$plotTriplets(c("Seq1", "Seq4", "Seq7"))[[1]]
```



Testing known blocks

It is possible to tell HybRIDS to assume a given region between your two sequences is recombinant, and then test them for significance and estimate their date.

This is done using the `addUserBlock` method, providing the two sequence names, the base position the block starts at, and the base position the block ends at.

User defined blocks can then be tabulated with the `tabulateUserBlocks` method, and they can be tested for significance and dated according to the `BlockDating` step settings, with the `dateUserBlocks` method.

```
MyAnalysis$addUserBlock(c("Seq1", "Seq4"), firstbp = 287463, lastbp = 295890)
MyAnalysis$tabulateUserBlocks()
```

```
##           FirstBP LastBP ApproxBpLength SNPs CorrectedSNPs P_Value
## Seq1:Seq4  287463 295890           8428   NA            NA      NA
##           P_Threshold fiveAge fiftyAge ninetyFiveAge
## Seq1:Seq4          NA      NA       NA            NA
```

```
MyAnalysis$dateUserBlocks()
MyAnalysis$tabulateUserBlocks()
```

```
##           FirstBP LastBP ApproxBpLength SNPs CorrectedSNPs      P_Value
## Seq1:Seq4  287463 295890           8428   55            55 1.120252e-63
##           P_Threshold fiveAge fiftyAge ninetyFiveAge
## Seq1:Seq4       0.005  410144   330115        263346
```

To clear the user added blocks, use the `clearUserBlocks` method:

```
MyAnalysis$clearUserBlocks(c("Seq1", "Seq4"))
MyAnalysis$tabulateUserBlocks()
```

```
##  [1] FirstBP        LastBP         ApproxBpLength SNPs
##  [5] CorrectedSNPs  P_Value        P_Threshold    fiveAge
##  [9] fiftyAge       ninetyFiveAge
## <0 rows> (or 0-length row.names)
```

The HybRIDSapp

We provide a Shiny based webapp that allows and R session with HybRIDS to be run as a webapp - providing either a web-based graphical user interface on the user's local machine, or a means of a centalized platform.

It can be obtained from its GitHub repository by using git on the terminal:

```
git clone https://github.com/Ward9250/HybRIDSapp.git
```

Once it has been downloaded, you can open the project file in RStudio and hit the Run App button to run the app on your machine. You will need the `shiny` and `shinyBS` packages installed to R.

There are multiple options and avenues for deploying a Shiny based web app. See the Shiny webpage and documentation for details on these.

The settings and steps of the analysis are exactly the same as those presented in this manual. The web-app provides a way for people to use HybRIDS in an exporatory manner if they are not proficient at coding, or want to change the settings of the analysis frequently.

The HybRIDS package was designed to be easy to use in an exploratory way on the R console, or in a script for batch processing. The 'reactive' design of Shiny and the web-app is particularily suited for exploratory analyses: Whenever a setting is altered the analysis is updated and the results are displayed to the user in real-time automatically.

The HybRIDSapp is much newer than the package and some bugs may be likely and the design of the app may change.

Problems, Bugs, and assistance

If you have any trouble using the HybRIDS package, or find an error or bug you are unfamiliar with, don't hesitate to contact code author and repo maintainer Ben J. Ward (Ward9250) at b.ward@uea.ac.uk, or visit the GitHub Repository Ward9250/HybRIDS and file a bug report or message there.
