Package ‘Map2NCBI’

October 12, 2022

Type Package

Title Mapping Markers to the Nearest Genomic Feature

Version 1.4

Date 2020-01-23

Author Lauren L. Hulsman Hanna and David G. Riley

Maintainer Lauren Hanna <Lauren.Hanna@ndsu.edu>

Description Allows the user to generate a list of features (gene, pseudo, RNA, CDS, and/or UTR) directly from NCBI database for any species with a current build available. Option to save downloaded and formatted files is available, and the user can prioritize the feature list based on type and assembly builds present in the current build used. The user can then use the list of features generated or provide a list to map a set of markers (designed for SNP markers with a single base pair position available) to the closest feature based on the map build. This function does require map positions of the markers to be provided and the positions should be based on the build being queried through NCBI.

License GPL (>= 2)

Depends R (>= 3.5), rentrez (>= 1.2)

Encoding UTF-8

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2020-01-24 19:10:02 UTC

R topics documented:

Map2NCBI-package 2
Example10MarkerFile 3
GeneList_BTA1 4
GetGeneList 4
MapMarkers 7

Index **12**

`Map2NCBI-package` *Mapping Markers to the Nearest Genomic Feature*

Description

Allows the user to generate a list of features (gene, pseudo, RNA, CDS, and/or UTR) directly from NCBI database for any species with a build available. Option to save downloaded and formatted files is available, and the user can prioritize the feature list based on type and assembly builds present in the build used. The GetGeneList function can now handle query of the NCBI for genome builds released prior to 2018 as well as the latest build for that species. The user can then use the list of features generated or provide a list to map a set of markers (designed for SNP markers with a single base pair position available) to the closest feature based on the map build. This function does require map positions of the markers to be provided and the positions should be based on the build being queried through NCBI.

Details

Package:	Map2NCBI
Type:	Package
Version:	1.4
Date:	2020-01-23
License:	GPL (>= 2)

This package can be used as a two part process with the `GetGeneList` function followed by the `MapMarkers` function. See individual function documentation for more information.

Author(s)

Lauren L. Hulsman Hanna and David G. Riley

Maintainer: Lauren Hanna <`<Lauren.Hanna@ndsu.edu>`>

References

Hulsman Hanna, L. L., and D. G. Riley. 2014. Mapping genomic markers to closest feature using the R package Map2NCBI. Livest. Sci. 162:59-65. doi:10.1016/j.livsci.2014.01.019

National Center for Biotechnology Information. 2018. Latest assembly version 'README' file, last updated 26 February 2018. Available at: `https://ftp.ncbi.nlm.nih.gov/genomes/refseq/README.txt` (Accessed 23 Jan 2020).

See Also

Functions: `GetGeneList` & `MapMarkers`

Examples

```
#See individual function documentation for applied examples.
```

`Example10MarkerFile` *Example Marker File with 10 Markers on BTA 1*

Description

This file contains marker map placement based on UCD 1.2 build and is meant to be used as an example of the `MapMarkers` function.

Usage

```
data(Example10MarkerFile)
```

Format

The format is: chr "Example10MarkerFile"

Details

Markers are SNP from BovSNP50 assay by Illumina, Inc.(San Diego, CA). These markers were used in a Nellore-Angus crossbred population described in the reference below. Map positions are based on the current Bos taurus assembly.

References

Riley, D.G., Welsh Jr., T.H., Gill, C.A., Hulsman, L.L., Herring, A.D., Riggs, P.K., Sawyer, J.E., Sanders, J.O., 2013. Whole genome association of SNP with newborn calf cannon bone length. Liv. Sci. 155: 186-196. doi:10.1016/j.livsci.2013.05.022

Examples

```
data(Example10MarkerFile)
```

GeneList_BTA1 — *Output from GetGeneList function for BTA 1*

Description

This output is provided to run the `MapMarkers` function example. It was generated using the `GetGeneList` function example and truncated to only include BTA 1 data.

Usage

```
data(GeneList_BTA1)
```

Format

The format is: chr "GeneList_BTA1"

Examples

```
data(GeneList_BTA1)
```

GetGeneList — *A Function to Filter and Save Genomic Features from NCBI (all builds)*

Description

`GetGeneList` allows the user to access the NCBI database for the species specified using the secure ftp site, download feature information as well as filter and save feature information for future use. This update now allows users to specify if the latest assembly build should be used or not using the **rentrez** package. Once the `GetGeneList` function is complete, no other access to NCBI or the internet is required. This function requires user input to determine the feature and class types that will be retained during the filtering process. Note: The requirements for this function have changed slightly due to NCBI ftp site organization changes.

Usage

```
GetGeneList(Species,latest = TRUE, savefiles = TRUE, destfile)
```

Arguments

`Species`	This term designates the species to be used in the function and is dependent on the scientific name. Options: Must include in quotation marks, where the genus and species should be separated by a space (e.g., "Bos taurus").

`latest`	Default is true. This term indicates if the most recent (latest) assembly build for that species should be used to get genomic features for. If set to false, the user will be prompted to idenify the assembly to use. In some species, the same assembly link may be listed more than once (e.g. GCF_000003055.6_Bos_taurus_UMD_3.1.1 vs. GCF_000003055.5_Bos_taurus_UMD_3.1.1). In any case, there is a number that designates one with a higher file number (e.g., "3055.6" vs. "3055.5" for Bos taurus 3.1). Always start with the higher file number for that build as it likely contains the feature table. If this fails, then try the other version. The assembly build should always match the marker map file build.
`savefiles`	Default is true. This term allows you to save the original feature list downloaded from the NCBI database as a text file as well as the filtered feature list produced from the function only if set to TRUE. Options: Must be either TRUE or FALSE.
`destfile`	This is the pathway to the computer location in which files will be saved and must be specified using quotation marks (e.g., `getwd()`).

Details

In running this function, the user will be prompted to enter feedback after the file downloads. Items that will be requested, if multiples are present include 1) primary feature type and 2) primary class type to prioritize filtering the dataset on. In each case, the user can opt to keep all feature and class types. This will mean that duplicate information is available per gene ID. If filtered, all unique gene ID will be returned, where preference is given to the class feature and class types specified. Gene ID without the preferred feature and class types will be queried for their available information and added while still removing duplicates. The file returned contains 20 columns based on the current NCBI file structure. Those column headings and descriptions are provided below in the `Value` section.

Note: While waiting for the function to run, if the user presses "Enter" prematurely, this will result in the function not running correctly and it will have to be started over. Please read instructions carefully.

If `savefiles` = `TRUE`, then both the original file from NCBI and the filtered file the user specified will be saved in the `destfile` location. Once the function has run, the user can choose to either use the information at that time or call it later using the saved file. In either case, the output from the filtered file can be used with marker data to run the `MapMarkers` function that is also a part of this package.

Value

Column headings and descriptions returned to the user from the `GetGeneList` function.

`feature`	The type of feature based on INSDC, which can include GENE, RNA (various types), and CDS.
`class`	Gene features are subdivided into classes according to the gene biotype. ncRNA features are subdivided according to the ncRNA_class. CDS features are subdivided into with_protein and without_protein, depending on whether the CDS feature has a protein accession assigned or not. CDS features marked as without_protein include CDS features for C regions and V/D/J segments of immunoglobulin and similar genes that undergo genomic rearrangement, and pseudogenes.

`assembly`	Accession.version of the assembly.
`assembly_unit`	The name of the assembly unit, such as "Primary Assembly", "ALT_REF_LOCI_1", or "non-nuclear".
`seq_type`	The type of sequence the feature is from. Typically include chromosome, mitochondrion, plasmid, or unplaced scaffold.
`chromosome`	The chromosome the feature is located on, which can include mitochondrial DNA or unknown (blank) if applicable.
`genomic_accession`	The accession.version of that genome the feature is found on.
`start`	The start position of the feature on the chromosome.
`end`	The end position of the feature on the chromosome.
`strand`	The orientation of the feature on the chromosome (can be + or -).
`product_accession`	The accession.version of the product referenced by this feature, if it exists.
`non-redundant_refseq`	For bacteria and archaea assemblies, this column contains the non-redundant WP_ protein accession corresponding to the CDS feature. This may be the same as the previous column for RefSeq genomes annotated directly with WP_ RefSeq proteins, or may be different for genomes annotated with genome-specific protein accessions (e.g. NP_ or YP_ RefSeq proteins) that reference a WP_ RefSeq accession.
`related_accession`	For eukaryotic RefSeq annotations, this is the RefSeq protein accession corresponding to the transcript feature, or the RefSeq transcript accession corresponding to the protein feature.
`name`	For genes, this is the gene description or full name. For RNA, CDS, and some other features, this is the product name.
`symbol`	The gene symbol.
`GeneID`	The corresponding gene ID on the NCBI database the feature is located in.
`locus_tag`	No description available from NCBI. Typically a blank column.
`feature_interval_length`	This is the sum of the lengths of all intervals for the feature (i.e. the length without introns for a joined feature).
`product_length`	This is the length of the product corresponding to the accession.version in product_accession" column. Protein product lengths are in amino acid units and do not include the stop codon which is included in "feature_interval_length" column. Additionally, product_length may differ from feature_interval_length if the product contains sequence differences vs. the genome, as found for some RefSeq transcript and protein products based on mRNA sequences and also for INSDC proteins that are submitted to correct genome discrepancies.
`attributes`	A semi-colon delimited list of a controlled set of qualifiers, if available. The list currently includes: partial, pseudo, pseudogene, ribosomal_slippage, trans_splicing, anticodon=NNN (for tRNAs), old_locus_tag=XXX.

Note

For issues or problems with this function, please contact Lauren Hanna at <Lauren.Hanna@ndsu.edu>.

Author(s)

Lauren L. Hulsman Hanna and David G. Riley

References

Hulsman Hanna, L. L., and D. G. Riley. 2014. Mapping genomic markers to closest feature using the R package Map2NCBI. Livest. Sci. 162:59-65. doi:10.1016/j.livsci.2014.01.019

National Center for Biotechnology Information. 2018. Latest assembly version 'README' file, last updated 26 February 2018. Available at: https://ftp.ncbi.nlm.nih.gov/genomes/refseq/README.txt (Accessed 23 Jan 2020).

See Also

Function: MapMarkers, Package: **rentrez**

Examples

```
#Example 1: Run the following example and, when prompted,
#choose [n],[1],[n],[3] to filter the build and feature
#information. This example is interactive and requires
#user input. Please note that pressing "Enter" prematurely
#can cause the function to not run properly.
## Not run:
GeneList = GetGeneList("Bos taurus",destfile=getwd())

## End(Not run)
```

MapMarkers *Mapping SNP Markers to Closest Genomic Feature*

Description

MapMarkers allows the user to map the supplied DNA markers (primarily designed for SNP markers) to the genomic feature in closest proximity based on the feature list generated using the GetGeneList function or a properly formated feature list (see Values section).

Usage

```
MapMarkers(features, markers, nAut, other = c("X"), savefiles = TRUE, destfile)
```

Arguments

`features`	This is the table or matrix in the current R session that will be used to map the marker list to. If using the `GetGeneList` function, the name given to the output should be supplied here (e.g., "GeneList" from the example provided in the `GetGeneList` documentation file). Note: If using a feature list generated using version 1.1 of this package, please review column names below for changes in format to ensure the function runs properly.
`markers`	This is the table or matrix in the current R session that will provide marker map information to use for the function. See `Values` for format of the marker file needed.
`nAut`	The number of autosomes in the species. This should reflect the total number of autosomes in the species, not the number of autosomes in the marker file.
`other`	The sex chromosomes or other genomic information available (e.g., for eukaryotes this could include mitochondrial DNA). These must be specified inside quotation marks. If sex chromosomes or other genomic information is not provided in the marker file, set other=FALSE.
`savefiles`	Default is TRUE. This term allows you to save the final marker file with genomic feature information in the destfile location as "MappedMarkers.txt" format. Any markers that cannot be mapped due to lack of feature information are saved as "NotMapped.txt". Options: Must be either TRUE or FALSE.
`destfile`	This is the pathway to the folder in which files will be saved and must be specified using quotation marks (e.g., `"C:/Temp/"` or `paste(getwd(),"/",sep="")`).

Details

The `MapMarkers` function processes each chromosome individually to search for features that fall closest to the markers provided based on the map information included. Map positions of the markers must match the assembly being used in the feature list. Once the closest feature has been found, the marker and feature information are saved together and take the format of binding the marker map file (which include at a minimum 3 columns) with the feature list columns provided (20 columns if using the `GetGeneList` function or a minimum of 4 columns if formatting yourself). The function also adds 2 additional columns described in `Value` section to identify the distance the marker is from the feature and a category to group the marker's proximity to the feature by.

Value

1) Format for feature list if not generated using the `GetGeneList` function:

`FeatureName`	The name of the feature provided. Column heading name can be changed, but should be included to identify the feature once the `MapMarkers` function is completed.
`chromosome`	The chromosome in which the genomic feature is located on. The column heading name must be given this name. If including sex chromosomes or other genomic information, label based on letters or abbreviation (e.g., "X").
`start`	The start position of the genomic feature based on the build used. The column heading name must be given this name. This used to be called "chr_start" in version 1.1 of this package.

`end` The end or stop position of the genomic feature based on the build used. The column heading name must be given this name. This used to be called "chr_stop" in version 1.1 of this package.

2) Format for the marker map file:

`Marker` Name of the marker. Be aware of R language and its restrictions. The name of this column heading can be changed to something else.

`chromosome` The chromosome in which the marker is mapped to. The name of this column is required and must be exact. This must be numeric. If including sex chromosomes or other genomic information, assign numbers to each. Number the sex chromosomes or other genomic information in the order that matches the order listed in the other=c() statement (e.g., X and Y chromosomes are labeled 30 and 31, respectively, so other=c("X","Y") to follow that order). The function will automatically align the letter with the correct number as long as they are included in the order specified.

`position` The base pair position of the marker based on the map build used. This build must also match the build in which you generated genomic feature from using the `GetGeneList` function or other method. The name of this column is required and must be exact. This function was designed for SNP markers, but if using other types of markers, you should choose the base pair location that best represents the marker (e.g., center position) and include that in this column.

NOTE: Order of the columns in both files are not necessarily important, but correct column heading names are essential. R programming is case sensitive, so make sure it matches exactly unless otherwise noted. Other columns may be included, but will not be used by the function. Any columns included in this file will be returned with the final marker file after the `MapMarkers` function is completed.

3) Additional columns included in the output file of the `MapMarkers` function:

`Distance` The base pair distance of the marker from the closest feature identified. If the marker is located inside the feature, the distance is set to zero.

`Inside?` The category in which the marker and feature pair fall into. This is based on the distance between the Marker and the closest feature, which is broken into 11 categories described in the next section.

4) Categories that are included in the "Inside?" column:

`Yes,_Inside_Gene`
Marker is located in the closest feature.

`Marker_is_<=_2500_bp_Before_Feature`
The closest feature is located after the marker position and is within 2,500 base pairs (bp).

`Marker_is_<=_2500_bp_After_Feature`
The closest feature is located before the marker position and is within 2,500 bp.

`Marker_is_>_2500_bp_<=5000_bp_Before_Feature`
The closest feature is located before the marker position and is between 2,500 bp and 5,000 bp from the marker.

`Marker_is_>_2500_bp_<=5000_bp_After_Feature`
: The closest feature is located after the marker position and is between 2,500 bp and 5,000 bp from the marker.

`Marker_is_>_5000_bp_<=25000_bp_Before_Feature`
: The closest feature is located before the marker position and is between 5,000 bp and 25,000 bp from the marker.

`Marker_is_>_5000_bp_<=25000_bp_After_Feature`
: The closest feature is located after the marker position and is between 5,000 bp and 25,000 bp from the marker.

`Nearest_feature_is_>_25,000_bp_before_marker`
: The closest feature is located before the marker position and is more than 25,000 bp from the marker.

`Nearest_feature_is_>_25,000_bp_after_marker`
: The closest feature is located after the marker position and is more than 25,000 bp from the marker.

`Nearest_feature_is_>_1_Mb_before_marker`
: The closest feature is located before the marker position and is more than 1,000,000 bp (1 Mb) from the marker.

`Nearest_feature_is_>_1_Mp_after_marker`
: The closest feature is located after the marker position and is more than 1,000,000 bp (1 Mb) from the marker.

Note

For issues or problems with this function, please contact Lauren Hanna at <`Lauren.Hanna@ndsu.edu`>.

Author(s)

Lauren L. Hulsman Hanna and David G. Riley

References

Hulsman Hanna, L. L., and D. G. Riley. 2014. Mapping genomic markers to closest feature using the R package Map2NCBI. Livest. Sci. 162:59-65. doi:10.1016/j.livsci.2014.01.019

See Also

Function: `GetGeneList`

Examples

```
#Example 1: Step 1 includes running "GetGeneList" function.
#As this step is interactive, a dataset from Bos taurus has
#been generated and available to use in the \data folder as
#well as a subset of marker information from BTA 1. Use the
#following code to run this example:

data(GeneList_BTA1)
data(Example1@MarkerFile)
```

```
Example1 = MapMarkers(GeneList_BTA1, Example10MarkerFile,
    nAut=29,other="X",savefiles = FALSE)

#Note, this example will not save the output to the working
#directory, but will return the information to "Example1"
#variable.
```

Index

∗ **datasets**
    Example10MarkerFile, 3
    GeneList_BTA1, 4
∗ **package**
    Map2NCBI-package, 2

Example10MarkerFile, 3

GeneList_BTA1, 4
GetGeneList, *2*, *4*, 4, *7*, *8*, *10*

Map2NCBI *(*Map2NCBI-package*)*, 2
Map2NCBI-package, 2
MapMarkers, *2–5*, *7*, 7

rentrez, *4*, *7*