Package 'APCalign'

January 28, 2025

Title Resolving Plant Taxon Names Using the Australian Plant Census

Version 1.1.2

Description The process of resolving taxon names is necessary when working with biodiversity data. 'APCalign' uses the Australian Plant Census (APC) and the Australian Plant Name Index (APNI) to align and update plant taxon names to current, accepted standards. 'APCalign' also supplies information about the established status of plant taxa across different states/territories.

License MIT + file LICENSE

Encoding UTF-8

Language en

LazyData true

Depends R (>= 4.1.0),

Imports readr, purrr, dplyr, stringr, stringi, stringdist, crayon, httr, jsonlite, curl, arrow, rlang

Suggests janitor, tidyr, covr, knitr, rmarkdown, kableExtra, here, testthat (>= 3.0.0)

RoxygenNote 7.3.2

Config/testthat/edition 3

VignetteBuilder knitr

URL https://traitecoevo.github.io/APCalign/, https://github.com/traitecoevo/APCalign

BugReports https://github.com/traitecoevo/APCalign/issues

NeedsCompilation no

Author Daniel Falster [aut, cre, cph] (<https://orcid.org/0000-0002-9814-092X>), Elizabeth Wenk [aut, ctb] (<https://orcid.org/0000-0001-5640-5910>), Will Cornwell [aut, ctb] (<https://orcid.org/0000-0003-4080-4073>), Fonti Kar [aut, ctb] (<https://orcid.org/0000-0002-2760-3974>), Carl Boettiger [ctb] (<https://orcid.org/0000-0002-1642-628X>)

Maintainer Daniel Falster <daniel.falster@unsw.edu.au>

Repository CRAN

Date/Publication 2025-01-28 09:20:02 UTC

Contents

`align_taxa` *Align Australian plant scientific names to the APC or APNI*

Description

For a list of Australian plant names, find taxonomic or scientific name alignments to the APC or APNI through standardizing formatting and fixing spelling errors.

Usage case: Users will run this function if they wish to see the details of the matching algorithms, the many output columns that the matching function compares to as it seeks the best alignment. They may also select this function if they want to adjust the "fuzziness" level for fuzzy matches, options not allowed in create_taxonomic_update_lookup. This function is the first half of create_taxonomic_update_lookup.

Usage

```
align_taxa(
  original_name,
  output = NULL,
  full = FALSE,
  resources = load_taxonomic_resources(),
  quiet = FALSE,
  fuzzy_abs_dist = 3,
  fuzzy_rel_dist = 0.2,
```

```
  fuzzy_matches = TRUE,
  imprecise_fuzzy_matches = FALSE,
  APNI_matches = TRUE,
  identifier = NA_character_
)
```

Arguments

original_name	A list of names to query for taxonomic alignments.
output	(optional) The name of the file to save the results to.
full	Parameter to determine how many columns are output
resources	the taxonomic resources used to align the taxa names. Loading this can be slow, so call load_taxonomic_resources separately to greatly speed this function up and pass the resources in.
quiet	Logical to indicate whether to display messages while aligning taxa.
fuzzy_abs_dist	The number of characters allowed to be different for a fuzzy match.
fuzzy_rel_dist	The proportion of characters allowed to be different for a fuzzy match.
fuzzy_matches	Fuzzy matches are turned on as a default. The relative and absolute distances allowed for fuzzy matches to species and infraspecific taxon names are defined by the parameters fuzzy_abs_dist and fuzzy_rel_dist
imprecise_fuzzy_matches	Imprecise fuzzy matches uses the fuzzy matching function with lenient levels set (absolute distance of 5 characters; relative distance = 0.25). It offers a way to get a wider range of possible names, possibly corresponding to very distant spelling mistakes. This is FALSE as default and all outputs should be checked as it often makes erroneous matches.
APNI_matches	Name matches to the APNI (Australian Plant Names Index) are turned on as a default.
identifier	A dataset, location or other identifier, which defaults to NA.

Details

- This function finds taxonomic alignments in APC or scientific name alignments in APNI.
- It uses the internal function match_taxa to attempt to match input strings to taxon names in the APC/APNI.
- It sequentially searches for matches against more than 20 different string patterns, prioritising exact matches (to accepted names as well as synonyms, orthographic variants) over fuzzy matches.
- It prioritises matches to taxa in the APC over names in the APNI.
- It identifies string patterns in input names that suggest a name can only be aligned to a genus (hybrids that are not in the APC/APNI; graded species; taxa not identified to species), and indicates these names only have a genus-rank match.

Notes:

- If you will be running the function APCalign::create_taxonomic_update_lookup many times, it is best to load the taxonomic resources separately using resources <- load_taxonomic_resources(), then add the argument resources = resources
- The name Banksia cerrata does not align as the fuzzy matching algorithm does not allow the first letter of the genus and species epithet to change.
- With this function you have the option of changing the fuzzy matching parameters. The defaults, with fuzzy matches only allowing changes of 3 (or fewer) characters AND 20% (or less) of characters has been carefully calibrated to catch just about all typos, but very, very rarely mis-align a name. If you wish to introduce less conservative fuzzy matching it is recommended you manually check the aligned names.
- It is recommended that you begin with imprecise_fuzzy_matches = FALSE (the default), as quite a few of the less precise fuzzy matches are likely to be erroneous. This argument should be turned on only if you plan to check all alignments manually.
- The argument identifier allows you to add a fix text string to all genus- and family- level names, such as identifier = "Royal NP" would return "Acacia sp. [Royal NP]".

Value

A tibble with columns that include original_name, aligned_name, taxonomic_dataset, taxon_rank, aligned_reason, alignment_code.

- original_name: the original plant name input.
- aligned_name: the original plant name after the function standardise_names has standardised the syntax of infraspecific taxon designations.
- taxonomic_dataset: the source of the aligned names (APC or APNI).
- taxon_rank: the taxonomic rank of the aligned name.
- aligned_reason: the explanation of a specific taxon name alignment (from an original name to an aligned name).
- alignment_code: a code that accompanies the aligned_reason, indicating the relative sequence of the match during the alignment process.
- cleaned_name: original name with punctuation and infraspecific taxon designation terms standardised by the function standardise_names; streamlines exact matches.
- stripped_name: cleaned name with punctuation and infraspecific taxon designation terms removed by the function strip_names; improves fuzzy matches.
- stripped_name2: cleaned name with punctuation, infraspecific taxon designation terms, and other filler words removed by the function `strip_names_extra`; required for matches to `first two word` and `first three words`.
- trinomial: the first three words in `stripped_name2`, required for matches that ignore all other text in the original_name; improves phrase name matches.
- binomial: the first two words in `stripped_name2`, required for matches that ignore all other text in the original_name; improves phrase name matches.
- genus: the first two words in `cleaned_name`; required for genus-rank matches and reprocessing of genus-rank names.
- fuzzy_match_genus: fuzzy match of genus column to best match among APC-accepted names; required for fuzzy matches of genus-rank names.

- fuzzy_match_genus_synonym: fuzzy match of genus column to best match among APC-synonymous names, only considering different matches to those documented under APC-accepted genera; required for fuzzy matches of genus-rank names.
- fuzzy_match_genus_APNI: fuzzy match of genus column to best match among APNI names, only considering different matches to those documented under APC-accepted and APC-known genera; required for fuzzy matches of genus-rank names.
- fuzzy_match_family: fuzzy match of genus column to best match among APC-accepted family names; required for fuzzy matches of family-rank names.
- fuzzy_match_family_synonym: fuzzy match of genus column to best match among APC-synonymous family names; required for fuzzy matches of family-rank names.
- fuzzy_match_cleaned_APC: fuzzy match of stripped_name to APC-accepted names; created for yet-to-be-aligned names at the match step 05a in the function `match_taxa`.
- fuzzy_match_cleaned_APC_synonym: fuzzy match of stripped_name to APC-synonymous names; created for yet-to-be-aligned names at the match step 05b in the function `match_taxa`.
- fuzzy_match_cleaned_APC_imprecise: imprecise fuzzy match of stripped_name to APC-accepted names; created for yet-to-be-aligned names at the match step 07a in the function `match_taxa`.
- fuzzy_match_cleaned_APC_synonym_imprecise: imprecise fuzzy match of stripped_name to APC-accepted names; created for yet-to-be-aligned names at the match step 07b in the function `match_taxa`.
- fuzzy_match_binomial: fuzzy match of binomial column to best match among APC-accepted names; created for yet-to-be-aligned names at match step 10c in the function `match_taxa`.
- fuzzy_match_binomial_APC_synonym: fuzzy match of binomial column to best match among APC-synonymous names; created for yet-to-be-aligned names at match step 10d in the function `match_taxa`.
- fuzzy_match_trinomial: fuzzy match of trinomial column to best match among APC-accepted names; created for yet-to-be-aligned names at match step 09c in the function `match_taxa`.
- fuzzy_match_trinomial_synonym: fuzzy match of trinomial column to best match among APC-synonymous names; created for yet-to-be-aligned names at match step 09d in the function `match_taxa`.
- fuzzy_match_cleaned_APNI: fuzzy match of stripped_name to APNI names; created for yet-to-be-aligned names at the match step 11a in the function `match_taxa`.
- fuzzy_match_cleaned_APNI_imprecise: imprecise fuzzy match of stripped_name to APNI names; created for yet-to-be-aligned names at the match step 11b in the function `match_taxa`.

See Also

`load_taxonomic_resources`

Other taxonomic alignment functions: `create_taxonomic_update_lookup()`, `update_taxonomy()`

Examples

```
resources <- load_taxonomic_resources()
```

```
# example 1
align_taxa(c("Poa annua", "Abies alba"), resources=resources)

# example 2
input <- c("Banksia serrata", "Banksia serrate", "Banksia cerrata",
"Banksia serrrrata", "Dryandra sp.", "Banksia big red flowers")

aligned_taxa <-
  APCalign::align_taxa(
    original_name = input,
    identifier = "APCalign test",
    full = TRUE,
    resources=resources
  )
```

`create_species_state_origin_matrix`
State level native and introduced origin status

Description

This function uses the taxon distribution data from the APC to determine state level native and introduced origin status.

This function processes the geographic data available in the APC and returns state level native, introduced and more complicated origins status for all taxa.

Usage

```
create_species_state_origin_matrix(resources = load_taxonomic_resources())
```

Arguments

`resources`	the taxonomic resources required to make the summary statistics. Loading this can be slow, so call load_taxonomic_resources separately to greatly speed this function up and pass the resources in.

Value

A tibble with columns representing each state and rows representing each species. The values in each cell represent the origin of the species in that state.

See Also

`load_taxonomic_resources`

Other diversity methods: `native_anywhere_in_australia()`, `state_diversity_counts()`

Examples

```
create_species_state_origin_matrix()
```

`create_taxonomic_update_lookup` *Create a table with the best-possible scientific name match for Australian plant names*

Description

This function takes a list of Australian plant names that need to be reconciled with current taxonomy and generates a lookup table of the best-possible scientific name match for each input name.

Usage case: This is APCalign's core function, merging together the alignment and updating of taxonomy.

Usage

```
create_taxonomic_update_lookup(
  taxa,
  stable_or_current_data = "stable",
  version = default_version(),
  taxonomic_splits = "most_likely_species",
  full = FALSE,
  fuzzy_abs_dist = 3,
  fuzzy_rel_dist = 0.2,
  fuzzy_matches = TRUE,
  APNI_matches = TRUE,
  imprecise_fuzzy_matches = FALSE,
  identifier = NA_character_,
  resources = load_taxonomic_resources(quiet = quiet),
  quiet = FALSE,
  output = NULL
)
```

Arguments

`taxa`	A list of Australian plant species that needs to be reconciled with current taxonomy.
`stable_or_current_data`	either "stable" for a consistent version, or "current" for the leading edge version.
`version`	The version number of the dataset to use.

`taxonomic_splits`
: How to handle one_to_many taxonomic matches. Default is "return_all". The other options are "collapse_to_higher_taxon" and "most_likely_species". most_likely_species defaults to the original_name if that name is accepted by the APC; this will be right for certain species subsets, but make errors in other cases, use with caution.

`full`
: logical for whether the full lookup table is returned or just key columns

`fuzzy_abs_dist`
: The number of characters allowed to be different for a fuzzy match.

`fuzzy_rel_dist`
: The proportion of characters allowed to be different for a fuzzy match.

`fuzzy_matches`
: Fuzzy matches are turned on as a default. The relative and absolute distances allowed for fuzzy matches to species and infraspecific taxon names are defined by the parameters `fuzzy_abs_dist` and `fuzzy_rel_dist`.

`APNI_matches`
: Name matches to the APNI (Australian Plant Names Index) are turned off as a default.

`imprecise_fuzzy_matches`
: Imprecise fuzzy matches uses the fuzzy matching function with lenient levels set (absolute distance of 5 characters; relative distance = 0.25). It offers a way to get a wider range of possible names, possibly corresponding to very distant spelling mistakes. This is FALSE as default and all outputs should be checked as it often makes erroneous matches.

`identifier`
: A dataset, location or other identifier, which defaults to NA.

`resources`
: These are the taxonomic resources used for cleaning, this will default to loading them from a local place on your computer. If this is to be called repeatedly, it's much faster to load the resources using `load_taxonomic_resources` separately and pass the data in.

`quiet`
: Logical to indicate whether to display messages while loading data and aligning taxa.

`output`
: file path to save the output. If this file already exists, this function will check if it's a subset of the species passed in and try to add to this file. This can be useful for large and growing projects.

Details

- It uses first the function `align_taxa`, then the function `update_taxonomy` to achieve the output. The aligned name is plant name that has been aligned to a taxon name in the APC or APNI by the align_taxa function.

Notes:

- If you will be running the function APCalign::create_taxonomic_update_lookup many times, it is best to load the taxonomic resources separately using `resources <- load_taxonomic_resources()`, then add the argument resources = resources
- The name Banksia cerrata does not align as the fuzzy matching algorithm does not allow the first letter of the genus and species epithet to change.
- The argument taxonomic_splits allows you to choose the outcome for updating the names of taxa with ambiguous taxonomic histories; this applies to scientific names that were once attached to a more broadly circumscribed taxon concept, that was then split into several more

narrowly circumscribed taxon concepts, one of which retains the original name. There are three options: most_likely_species returns the name that is retained, with alternative names documented in square brackets; return_all adds additional rows to the output, one for each possible taxon concept; collapse_to_higher_taxon returns the genus with possible names in square brackets.

- The argument identifier allows you to add a fix text string to all genus- and family- level names, such as identifier = "Royal NP" would return `Acacia sp. \[Royal NP]`.

Value

A lookup table containing the accepted and suggested names for each original name input, and additional taxonomic information such as taxon rank, taxonomic status, taxon IDs and genera.

- original_name: the original plant name.
- aligned_name: the input plant name that has been aligned to a taxon name in the APC or APNI by the align_taxa function.
- accepted_name: the APC-accepted plant name, when available.
- suggested_name: the suggested plant name to use. Identical to the accepted_name, when an accepted_name exists; otherwise the the suggested_name is the aligned_name.
- genus: the genus of the accepted (or suggested) name; only APC-accepted genus names are filled in.
- family: the family of the accepted (or suggested) name; only APC-accepted family names are filled in.
- taxon_rank: the taxonomic rank of the suggested (and accepted) name.
- taxonomic_dataset: the source of the suggested (and accepted) names (APC or APNI).
- taxonomic_status: the taxonomic status of the suggested (and accepted) name.
- taxonomic_status_aligned: the taxonomic status of the aligned name, before any taxonomic updates have been applied.
- aligned_reason: the explanation of a specific taxon name alignment (from an original name to an aligned name).
- update_reason: the explanation of a specific taxon name update (from an aligned name to an accepted or suggested name).
- subclass: the subclass of the accepted name.
- taxon_distribution: the distribution of the accepted name; only filled in if an APC accepted_name is available.
- scientific_name_authorship: the authorship information for the accepted (or synonymous) name; available for both APC and APNI names.
- taxon_ID: the unique taxon concept identifier for the accepted_name; only filled in if an APC accepted_name is available.
- taxon_ID_genus: an identifier for the genus; only filled in if an APC-accepted genus name is available.
- scientific_name_ID: an identifier for the nomenclatural (not taxonomic) details of a scientific name; available for both APC and APNI names.
- row_number: the row number of a specific original_name in the input.
- number_of_collapsed_taxa: when taxonomic_splits == "collapse_to_higher_taxon", the number of possible taxon names that have been collapsed.

See Also

load_taxonomic_resources

Other taxonomic alignment functions: align_taxa(), update_taxonomy()

Examples

```
resources <- load_taxonomic_resources()

# example 1
create_taxonomic_update_lookup(c("Eucalyptus regnans",
                                 "Acacia melanoxylon",
                                 "Banksia integrifolia",
                                 "Not a species"),
                                 resources = resources)

# example 2
input <- c("Banksia serrata", "Banksia serrate", "Banksia cerrata",
"Banksea serrata", "Banksia serrrrata", "Dryandra")

create_taxonomic_update_lookup(
    taxa = input,
    identifier = "APCalign test",
    full = TRUE,
    resources = resources
  )

# example 3
taxon_list <-
  readr::read_csv(
  system.file("extdata", "test_taxa.csv", package = "APCalign"),
  show_col_types = FALSE)

create_taxonomic_update_lookup(
    taxa = taxon_list$original_name,
    identifier = taxon_list$notes,
    full = TRUE,
    resources = resources
  )
```

---

default_version *Get the default version for stable data*

Description

This function returns the default version for stable data, which is used when no version is specified.

Usage

```
default_version()
```

Value

A character string representing the default version for stable data.

Examples

```
default_version()
```

`gbif_lite` *GBIF Australian Plant Data*

Description

A subset of plant data from the Global Biodiversity Information Facility

Usage

```
gbif_lite
```

Format

`gbif_lite` A tibble with 129 rows and 7 columns:

species The name of the first or species of scientificname

infraspecificepithet The name of the lowest or terminal infraspecific epithet of the scientificname

taxonrank The taxonomic rank of the most specific name

decimalLongitude Longitude in decimal degrees

decimalLatitude Latitude in decimal degrees

scientificname Scientific Name

verbatimscientificname Scientific name as it appeared in original record

Source

https://www.gbif.org/

get_apc_genus_family_lookup *Lookup Family by Genus from APC*

Description

Retrieve the family name for a given genus using taxonomic data from the Australian Plant Census (APC).

Usage

```
get_apc_genus_family_lookup(genus, resources = load_taxonomic_resources())
```

Arguments

genus	A character vector of genus names for which to retrieve the corresponding family names.
resources	The taxonomic resources required to make the lookup. Loading this can be slow, so call load_taxonomic_resources separately to speed up this function and pass the resources in.

Value

A data frame with two columns: "genus", indicating the genus name, and "family", indicating the corresponding family name from the APC.

See Also

load_taxonomic_resources

Examples

```
get_apc_genus_family_lookup(genus = c("Acacia", "Eucalyptus"))
```

get_versions *Which versions of taxonomic resources are available?*

Description

Which versions of taxonomic resources are available?

Usage

```
get_versions()
```

Value

tibble of dates when APC/APNI resources were downloaded as a Github Release

Examples

```
get_versions()
```

`load_taxonomic_resources` *Load taxonomic reference lists, APC & APNI*

Description

This function loads two taxonomic datasets for Australia's vascular plants, the APC and APNI, into the global environment. It creates several data frames by filtering and selecting data from the loaded lists.

Usage

```
load_taxonomic_resources(
  stable_or_current_data = "stable",
  version = default_version(),
  quiet = FALSE
)
```

Arguments

`stable_or_current_data`	Type of dataset to access. The default is "stable", which loads the dataset from a github archived file. If set to "current", the dataset will be loaded from a URL which is the cutting edge version, but this may change at any time without notice.
`version`	The version number of the dataset to use. Defaults to the default version.
`quiet`	A logical indicating whether to print status of loading to screen. Defaults to FALSE.

Details

- It accesses taxonomic data from a dataset using the provided version number or the default version.
- The output is several dataframes that include subsets of the APC/APNI based on taxon rank and taxonomic status.

Value

The taxonomic resources data loaded into the global environment.

Examples

```
load_taxonomic_resources(stable_or_current_data="stable",
version="2024-10-11")
```

`native_anywhere_in_australia` *Native anywhere in Australia*

Description

This function checks which species from a list is thought to be native anywhere in Australia according to the APC.

Usage

```
native_anywhere_in_australia(species, resources = load_taxonomic_resources())
```

Arguments

`species`	A character string typically representing the binomial for the species.
`resources`	An optional list of taxonomic resources to use for the lookup. If not provided, the function will load default taxonomic resources using the `load_taxonomic_resources()` function.

Details

Important caveats:

- This function will not detect within-Australia introductions, e.g. if a species is from Western Australia and is invasive on the east coast.
- Very recent invasions are unlikely to be documented yet in APC.
- Ideally check spelling and taxonomy updates first via create_taxonomic_update_lookup.
- For the complete matrix of species by states that also represents within-Australia invasions, use create_species_state_origin_matrix.

Value

A tibble with two columns: `species`, which is the same as the unique values of the input `species`, and `native_anywhere_in_aus`, a vector indicating whether each species is native anywhere in Australia, introduced by humans from elsewhere, or unknown with respect to the APC resource.

See Also

Other diversity methods: `create_species_state_origin_matrix()`, `state_diversity_counts()`

Examples

```
native_anywhere_in_australia(c("Eucalyptus globulus","Pinus radiata","Banksis notaspecies"))
```

`standardise_names` *Standardise taxon names*

Description

Standardises taxon names by performing a series of text substitutions to remove common inconsistencies in taxonomic nomenclature.

The function takes a character vector of taxon names as input and returns a character vector of taxon names using standardised taxonomic syntax as output.

Usage

```
standardise_names(taxon_names)
```

Arguments

`taxon_names`	A character vector of taxon names that need to be standardised.

Details

- It removes stray punctuation at the start and end of a character string.
- It standardises unusual characters and symbols to ASCII equivalents.
- It standardises taxon rank abbreviations and qualifiers (subsp., var., f.), as people use many variants of these terms.
- It standardises or removes a few additional filler words used within taxon names (affinis becomes aff.; s.l. and s.s. are removed).

Value

A character vector of standardised taxon names.

Examples

```
standardise_names(c("Quercus suber",
                    "Eucalyptus sp.",
                    "Eucalyptus spp.",
                    "Agave americana var. marginata",
                    "Agave americana v marginata",
                    "Notelaea longifolia forma longifolia",
                    "Notelaea longifolia f longifolia"))
```

standardise_taxon_rank *Standardise taxon ranks*

Description

Standardise taxon ranks from Latin into English.

Usage

```
standardise_taxon_rank(taxon_rank)
```

Arguments

taxon_rank	A character vector of Latin taxon ranks.

Details

The function takes a character vector of Latin taxon ranks as input and returns a character vector of taxon ranks using standardised English terms.

Value

A character vector of English taxon ranks.

Examples

```
standardise_taxon_rank(c("regnum", "kingdom", "classis", "class"))
```

state_diversity_counts *State- and territory-level diversity*

Description

For Australian states and territories, use geographic distribution data from the APC to calculate state-level diversity for native, introduced, and more complicated species origins

Usage

```
state_diversity_counts(state, resources = load_taxonomic_resources())
```

Arguments

state	A character string indicating the Australian state or territory to calculate the diversity for. Possible values are "NSW", "NT", "Qld", "WA", "ChI", "SA", "Vic", "Tas", "ACT", "NI", "LHI", "MI", "HI", "MDI", "CoI", "CSI", and "AR".
resources	the taxonomic resources required to make the summary statistics. loading this can be slow, so call load_taxonomic_resources separately to greatly speed this function up and pass the resources in.

Value

A tibble of diversity counts for the specified state or territory, including native, introduced, and more complicated species origins. The tibble has three columns: "origin" indicating the origin of the species, "state" indicating the Australian state or territory, and "num_species" indicating the number of species for that origin and state.

See Also

load_taxonomic_resources

Other diversity methods: create_species_state_origin_matrix(), native_anywhere_in_australia()

Examples

```
state_diversity_counts(state = "NSW")
```

---

strip_names *Strip taxon names*

Description

Strip taxonomic names of taxon rank abbreviations and qualifiers and special characters

Usage

```
strip_names(taxon_names)
```

Arguments

taxon_names	A character vector of taxonomic names to be stripped.

Details

Given a vector of taxonomic names, this function removes:

- subtaxa designations ("subsp.", "var.", "f.", and "ser")
- special characters (e.g., "-", ".", "(", ")", "?")
- extra whitespace

The resulting vector of names is also converted to lowercase.

Value

A character vector of stripped taxonomic names, with subtaxa designations, special characters, and extra whitespace removed, and all letters converted to lowercase.

Examples

```
strip_names(c("Abies lasiocarpa subsp. lasiocarpa",
              "Quercus kelloggii",
              "Pinus contorta var. latifolia"))
```

---

strip_names_extra *Strip taxon names, extra*

Description

Strip taxonomic names of `sp.` and hybrid symbols. This function assumes that a character function has already been run through `strip_names`.

Usage

```
strip_names_extra(taxon_names)
```

Arguments

`taxon_names`	A character vector of taxonomic names to be stripped.

Details

Given a vector of taxonomic names, this function removes additional filler words (" x " for hybrid taxa, "sp.") not removed by the function `strip_names`

Value

A character vector of stripped taxonomic names, with `sp.` and hybrid symbols removed.

Examples

```
strip_names_extra(c("Abies lasiocarpa subsp. lasiocarpa",
              "Quercus kelloggii",
              "Pinus contorta var. latifolia",
              "Acacia sp.",
              "Lepidium sp. Tanguin Hill (K.R.Newbey 10501)"))
```

update_taxonomy *Update to currently accepted APC name and add APC/APNI name metadata*

Description

For a list of taxon names aligned to the APC, update the name to an accepted taxon concept per the APC and add scientific name and taxon concept metadata to names aligned to either the APC or APNI.

Usage

```
update_taxonomy(
  aligned_data,
  taxonomic_splits = "most_likely_species",
  quiet = TRUE,
  output = NULL,
  resources = load_taxonomic_resources()
)
```

Arguments

aligned_data	A tibble of plant names to update. This table must include 5 columns, original_name, aligned_name, taxon_rank, taxonomic_dataset, and aligned_reason. These columns are created by the function `align_taxa`. The columns `original_name` and `aligned_name` must be in the format of the scientific name, with genus and species, and may contain additional qualifiers such as subspecies or varieties. The names are case insensitive.
taxonomic_splits	Variable that determines what protocol to use to update taxon names that are ambiguous due to taxonomic splits. The three options are: • `most_likely_species`, which returns the species name in use before the split; alternative names are returned in a separate column • `return_all`, which returns all possible names • `collapse_to_higher_taxon`, which declares that an ambiguous name cannot be aligned to an accepted species/infraspecific name and the name is demoted to genus rank
quiet	Logical to indicate whether to display messages while updating taxa.
output	(optional) Name of the file where results are saved. The default is NULL and no file is created. If specified, the output will be saved in a CSV file with the given name.
resources	the taxonomic resources required to make the summary statistics. Loading this can be slow, so call load_taxonomic_resources separately to greatly speed this function up and pass the resources in.

Details

- This function uses the APC to update the taxonomy of names aligned to a taxon concept listed in the APC to the currently accepted name for the taxon concept.
- The aligned_data data frame that is input must contain 5 columns, `original_name`, `aligned_name`, `taxon_rank`, `taxonomic_dataset`, and `aligned_reason`. (These are the columns output by the function `align_taxa`.)
- The aligned name is a plant name that has been aligned to a taxon name in the APC or APNI by the align_taxa function.

Notes:

- As the input for this function is a table with 5 columns (output by align_taxa), this function will only be used when you explicitly want to separate the alignment and updating components of APCalign. This function is the second half of create_taxonomic_update_lookup.

Value

A tibble with updated taxonomy for the specified plant names. The tibble contains the following columns:

- original_name: the original plant name.
- aligned_name: the input plant name that has been aligned to a taxon name in the APC or APNI by the align_taxa function.
- accepted_name: the APC-accepted plant name, when available.
- suggested_name: the suggested plant name to use. Identical to the accepted_name, when an accepted_name exists; otherwise the the suggested_name is the aligned_name.
- genus: the genus of the accepted (or suggested) name; only APC-accepted genus names are filled in.
- family: the family of the accepted (or suggested) name; only APC-accepted family names are filled in.
- taxon_rank: the taxonomic rank of the suggested (and accepted) name.
- taxonomic_dataset: the source of the suggested (and accepted) names (APC or APNI).
- taxonomic_status: the taxonomic status of the suggested (and accepted) name.
- taxonomic_status_aligned: the taxonomic status of the aligned name, before any taxonomic updates have been applied.
- aligned_reason: the explanation of a specific taxon name alignment (from an original name to an aligned name).
- update_reason: the explanation of a specific taxon name update (from an aligned name to an accepted or suggested name).
- subclass: the subclass of the accepted name.
- taxon_distribution: the distribution of the accepted name; only filled in if an APC accepted_name is available.
- scientific_name_authorship: the authorship information for the accepted (or synonymous) name; available for both APC and APNI names.

- taxon_ID: the unique taxon concept identifier for the accepted_name; only filled in if an APC accepted_name is available.
- taxon_ID_genus: an identifier for the genus; only filled in if an APC-accepted genus name is available.
- scientific_name_ID: an identifier for the nomenclatural (not taxonomic) details of a scientific name; available for both APC and APNI names.
- row_number: the row number of a specific original_name in the input.
- number_of_collapsed_taxa: when taxonomic_splits == "collapse_to_higher_taxon", the number of possible taxon names that have been collapsed.

See Also

load_taxonomic_resources

Other taxonomic alignment functions: `align_taxa()`, `create_taxonomic_update_lookup()`

Examples

```
# Update taxonomy for two plant names and print the result

resources <- load_taxonomic_resources()

update_taxonomy(
 dplyr::tibble(
   original_name = c("Dryandra preissii", "Banksia acuminata"),
   aligned_name = c("Dryandra preissii", "Banksia acuminata"),
   taxon_rank = c("species", "species"),
   taxonomic_dataset = c("APC", "APC"),
   aligned_reason = c(NA_character_,
   NA_character_)
 ),
 resources = resources
)
```

Index