Package ‘vectorsurvR’

July 2, 2024

**Type**  Package

**Title**  Data Access and Analytical Tools for 'VectorSurv' Users

**Version**  0.2.0

**Description**  Allows registered 'VectorSurv' <https://vectorsurv.org/> users access to data through the 'VectorSurv API' <https://api.vectorsurv.org/>. Additionally provides functions for analysis and visualization.

**License**  GPL-3

**Encoding**  UTF-8

**LazyData**  true

**RoxygenNote**  7.3.1

**Imports**  rstudioapi, dplyr, jsonlite, kableExtra, knitr, lubridate, stringr, plotly, ggplot2, httr, tidyR, magrittr, DT

**Suggests**  testthat (>= 3.0.0), rmarkdown, devtools,

**VignetteBuilder**  knitr

**Config/testthat/edition**  3

**Depends**  R (>= 2.10)

**NeedsCompilation**  no

**Author**  Christina De Cesaris [aut, cre]

**Maintainer**  Christina De Cesaris <cmdecesaris@ucdavis.edu>

**Repository**  CRAN

**Date/Publication**  2024-07-01 22:50:01 UTC

**Contents**

getAbundance ......................................................... 2
getAbundanceAnomaly ................................................. 3
getArthroCollections ................................................ 4
getInfectionRate .................................................... 5
getPools ............................................................... 6
getPoolsComparisionTable .......................................... 7
**getAbundance**

**getAbundance**

Calculates abundance

**Usage**

```r
getAbundance(
  collections,
  interval,
  species_list = NULL,
  trap_list = NULL,
  species_separate = FALSE
)
```

**Arguments**

- **collections**: Collections data retrieved from `getArthroCollections()`
- **interval**: Calculation interval for abundance, accepts “collection_date”, “Biweek”, “Week”, and “Month.”
- **species_list**: Species filter for calculating abundance. Species_display_name is the accepted notation. To see a list of species present in your data run `unique(collections$species_display_name)`. If species is unspecified, the default NULL will return data for all species in data.
- **trap_list**: Trap filter for calculating abundance. Trap_acronym is the accepted notation. Run `unique(collections$trap_acronym)` to see trap types present in your data. If trap_list is unspecified, the default NULL will return data for all trap types.
- **species_separate**: Should the species in species_list have abundance calculated separately? Setting to FALSE calculates the combined abundance. The same result can be performed by calculating on one species at the time.

**Value**

A dataframe of abundance values grouped by interval and filtered by parameters
**getAbundanceAnomaly**

**Examples**

```r
getAbundance(sample_collections,
interval = 'Week',
species_list = list('Cx pipiens'),
trap_list = list('GRVD', 'CO2'),
species_separate = FALSE)
```

---

**getAbundanceAnomaly**  
*Get Abundance Anomaly*

**Description**

`getAbundanceAnomaly(...)` requires at least five years prior to the target_year of arthro collections data to calculate for the specified parameters. The function uses the methods of the Gateway Abundance Anomaly calculator, and will not work if there is fewer than five years of data present.

**Usage**

```r
getAbundanceAnomaly(
collections,
interval,
target_year,
species_list = NULL,
trap_list = NULL,
species_separate = FALSE)
```

**Arguments**

- **collections**
  Collections data retrieved from `getArthroCollections()`

- **interval**
  Calculation interval for abundance, accepts "collection_date", "Biweek", "Week", and "Month"

- **target_year**
  Year to calculate analysis on. Collections data must have a year range of at least (target_year - 5, target_year)

- **species_list**
  Species filter for calculating abundance. Species_display_name is the accepted notation. To see a list of species present in your data run `unique(collections$species_display_name)`.
  If species is unspecified, the default NULL will return data for all species in data

- **trap_list**
  Trap filter for calculating abundance. Trap_acronym is the accepted notation. Run `unique(collections$trap_acronym)` to see trap types present in your data. If trap_list is unspecified, the default NULL will return data for all trap types

- **species_separate**
  Should the species in species_list have abundance calculated separately? Setting to FALSE calculates the combined abundance. The same result can be performed by calculating on one species at the time description
getArthroCollections

Value

Abundance anomaly calculation

Examples

getAbundanceAnomaly(sample_collections,"Biweek",target_year=2020, species_list="Cx pipiens")

getArthroCollections  Get arthropod collections data

Description

‘getArthroCollections()’ obtains collections data on a year range [start_year, end_year] for authorized VectorSurv Gateway accounts.

Usage

getArthroCollections(token, start_year, end_year, arthropod, agency_id = NULL)

Arguments

token                        A valid access token returned from ‘getToken()’
start_year                   Start year of data
end_year                     End year of data
arthropod                    Specify arthropod type from: 'mosquito', 'tick'
agency_id                    Filter on agency id, default to NULL for all available agencies, otherwise specify a single agency by code

Value

A dataframe of collections data specific to users account

Examples

## Not run:
token = getToken()
collections = getArthroCollections(token, 2021, 2022, 'mosquito')
## End(Not run)
**getInfectionRate**  

**Get Infection Rate**

**Description**

`getInfectionRate()` requires at least five years prior to the target_year of arthro collections data to calculate for the specified parameters. The function uses the methods of the Gateway Abundance Anomaly calculator, and will not work if there is fewer than five years of data present.

**Usage**

```r
getInfectionRate(
  pools,
  interval,
  target_disease,
  pt_estimate,
  scale = 1000,
  species_list = NULL,
  trap_list = NULL
)
```

**Arguments**

- **pools**: Pools data retrieved from `getPools()`
- **interval**: Calculation interval for infection rate, accepts “collection_date”, “Biweek”, “Week”, and “Month”
- **target_disease**: The disease to calculate infection rate for—i.e. “WNV”. Disease acronyms are the accepted input. To see a list of disease acronyms, run `unique(pools$target_acronym)`
- **pt_estimate**: The estimation type for infection rate. Options include: “mle”, “bc-mle”, “mir”
- **scale**: Constant to multiply infection rate by
- **species_list**: Species filter for calculating infection rate. `species_display_name` is the accepted notation. To see a list of species present in your data run `unique(pools$species_display_name)`. If species is unspecified, the default ‘NULL’ will return data for all species in data.
- **trap_list**: Trap filter for calculating infection rate. `Trap_acronym` is the accepted notation. Run `unique(pools$Trap_acronym)` to see trap types present in your data. If trap_list is unspecified, the default ‘NULL’ will return data for all trap types.

**Value**

Infection rate calculation
getPools

Get Pools data

Description

Retrieves VectorSurv pools data for desired year range

Usage

getPools(token, start_year, end_year, arthropod)

Arguments

token access token retrieved from 'getToken'
start_year Beginning of year range
end_year End of year range
arthropod Specify arthropod type from: 'mosquito', 'tick', 'nontick'

Value

Dataframe of pools data

Examples

## Not run:
token = getToken()
getPools(token, start_year = 2020, end_year = 2021, arthropod = 'tick')
## End(Not run)
Description

‘getPoolsComparisonTable()’ produces a frequency table for positive, negative, and pending pools counts by year and species. The more years present in the data, the larger the table.

Usage

getPoolsComparisonTable(
    pools,  # Pools data retrieved from ‘getPools()’
    interval,  # Calculation interval for comparison table, accepts “collection_date”, “Biweek”, “Week”, and “Month”
    target_disease,  # The disease to calculate infection rate for–i.e. “WNV”. Disease acronyms are the accepted input. To see a list of disease acronyms, run ‘unique(pools$target_acronym)’
    species_separate = FALSE  # Should the pools comparison be split by species of each pool. Default is FALSE
)

Arguments

pools  # Pools data retrieved from ‘getPools()’
interval  # Calculation interval for comparison table, accepts “collection_date”, “Biweek”, “Week”, and “Month”
target_disease  # The disease to calculate infection rate for–i.e. “WNV”. Disease acronyms are the accepted input. To see a list of disease acronyms, run ‘unique(pools$target_acronym)’
species_separate  # Should the pools comparison be split by species of each pool. Default is FALSE

Value

Frequency table of for pools data

Examples

getPoolsComparisonTable(sample_pools,
    interval = "Biweek",
    target_disease = "WNV",
    species_separate = TRUE)
getToken

**Get Token**

**Description**

ggetToken() returns a token needed to run getArthroCollections() and getPools(). Prints agencies associated with account credentials. The function prompts users for a VectorSurv account credentials.

**Usage**

ggetToken()

**Value**

User token

**Examples**

```r
## Not run: token = getToken()
```

getVectorIndex

**Get Vector Index**

**Description**

ggetVectorIndex() requires at least five years prior to the target_year of arthro collections data to calculate for the specified parameters. The function uses the methods of the Gateway Abundance Anomaly calculator, and will not work if there is fewer than five years of data present.

**Usage**

ggetVectorIndex(
collections,
pools,
interval,
target_disease,
pt_estimate,
scale = 1000,
species_list = NULL,
trap_list = NULL
)
```
**plotInfectionRate**

**Arguments**

- **collections**: Collections data retrieved from `getArthroCollections()`
- **pools**: Pools data retrieved from `getPools()`
- **interval**: Calculation interval for vector index, accepts “collection_date”, “Biweek”, “Week”, and “Month”
- **target_disease**: The disease to calculate infection rate for—i.e. “WNV”. Disease acronyms are the accepted input. To see a list of disease acronyms, run `unique(pools$target_acronym)`
- **pt_estimate**: The estimation type for infection rate. Options include: “mle”, “bc-mle”, “mir”
- **scale**: Constant to multiply infection rate
- **species_list**: Species filter for calculating abundance. Species_display_name is the accepted notation. To see a list of species present in your data run `unique(pools$species_display_name)` If species is unspecified, the default ‘NULL’ will return data for all species in data.
- **trap_list**: Trap filter for calculating abundance. Trap_acronym is the accepted notation. Run `unique(pools$trap_acronym)` to see trap types present in your data. If trap_list is unspecified, the default ‘NULL’ will return data for all trap types.

**Value**

Vector index calculation

**Examples**

```r
getVectorIndex(sample_collections, sample_pools, "Month", "WNV", "mle")
```

---

**plotInfectionRate**  
*Plot Infection Rate*

**Description**

`plotInfectionRate()` plots the output returned from `getInfectionRate()` with confidence intervals using ggplot

**Usage**

```
plotInfectionRate(InfRtOutput, year)
```

**Arguments**

- **InfRtOutput**: Output from returned `getInfectionRate()`
- **year**: Year to plot infection rate on

**Value**

ggplot object
Examples

```r
IR = getInfectionRate(sample_pools,
  interval = "Week",
  target_disease = "WNV",
  pt_estimate = "mle", species_list = c("Cx pipiens"),
  trap_list = c("CO2","GRVD")
)
plotInfectionRate(InfRtOutput = IR, year = 2017)
```

Description

`processAbunAnom()` processes the output returned from `getAbundanceAnomaly()` into a long form suitable for plotting in 'ggplot'

Usage

```r
processAbunAnom(AbAnomOutput)
```

Arguments

- `AbAnomOutput`: output from `getAbunAnom()`

Value

Abundance anomaly output processed into long form

Examples

```r
AbAnOut = getAbundanceAnomaly(sample_collections,
  interval = "Biweek",
  target_year = 2020,
  species_list = c("Cx tarsalis", "Cx pipiens"),
  species_separate = TRUE)
AbAnOut_L = processAbunAnom(AbAnOut)
```
**sample_collections**

**Sample Arthropod Collections Data**

**Description**

Sample Arthropod Collections data imitates the essential components of real collections data.

**Usage**

`sample_collections`

**Format**

A data frame with 200 rows and 10 variables:

- `collection_id` (double): Collection identification number.
- `collection_date` (character): The date the trap was picked up for collection.
- `num_trap` (integer): The number of unique traps in operation at one site.
- `surv_year` (double): Surveillance year of collection.
- `trap_nights` (integer): The number of nights a trap was in the field.
- `trap_problem_bit` (logical): If there was an issue with the trap.
- `num_count` (integer): Number of arthropods present in collection.
- `sex_type` (character): Sex of collected arthropods.
- `species_display_name` (character): Species name of collected arthropods.
- `trap_acronym` (character): The acronym of the trap placed in the field.

**Source**

https://vectorsurv.org/

---

**sample_pools**

**Sample Pools Data**

**Description**

Sample Pools data imitates the essential components of real pools data needed for calculations.

**Usage**

`sample_pools`
Format

A data frame with 82644 rows and 10 variables:

- **pool_id**: integer Pool identification number
- **surv_year**: integer Surveillance year of pool
- **collection_date**: character The date the trap was picked up for collection
- **sex**: integer Sex of collected arthropods
- **num_count**: integer Number of arthropods present in collection
- **target_acronym**: character The disease being tested for in the pool
- **status_name**: character Status of the tested disease
- **trap_acronym**: character The acronym of the trap placed in the field
- **species_display_name**: character Species name of collected arthropods

Source

https://vectorsurv.org/
Index

* abundance
  getAbundanceAnomaly, 3
* authentication
  getToken, 8
* datasets
  sample_collections, 11
  sample_pools, 11
* infection
  getInfectionRate, 5
* pools
  getInfectionRate, 5
  getPools, 6
* rate
  getInfectionRate, 5
  getAbundance, 2
  getAbundanceAnomaly, 3
  getArthroCollections, 4
  getInfectionRate, 5
  getPools, 6
  getPoolsComparisonTable, 7
  getToken, 8
  getVectorIndex, 8
  plotInfectionRate, 9
  processAbunAnom, 10

sample_collections, 11
sample_pools, 11