

# Package ‘vectorsurvR’

March 5, 2026

**Type** Package

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

**Version** 1.6.3

**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** 7.3.2

**Imports** rstudioapi, jsonlite, kableExtra, knitr, lubridate, stringr, htr2, tidyr, magrittr, DT, sf, scales, ggplot2, rlang, leaflet, base64enc, viridis, leaflet.minicharts

**Suggests** testthat (>= 3.0.0), rmarkdown, devtools, mockery, webshot2, htmlwidgets

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Depends** dplyr, R (>= 2.10)

**NeedsCompilation** no

**Author** Christina De Cesaris [aut],  
Lincoln Wells [aut],  
VectorSurv Staff [aut, cre]

**Maintainer** VectorSurv Staff <[help@vectorsurv.org](mailto:help@vectorsurv.org)>

**Repository** CRAN

**Date/Publication** 2026-03-05 08:00:02 UTC

## Contents

getAbundance . . . . .	2
getAbundanceAnomaly . . . . .	3

getAgency . . . . .	5
getArthroCollections . . . . .	5
getInfectionRate . . . . .	6
getPools . . . . .	7
getPoolsComparisionTable . . . . .	8
getRegions . . . . .	9
getSites . . . . .	9
getSpatialFeatures . . . . .	10
getSpeciesTable . . . . .	10
getToken . . . . .	11
getVectorIndex . . . . .	12
plotInvasiveSums . . . . .	13
plotPoolsMap . . . . .	14
plotSpeciesMap . . . . .	16
plotVectorAbundance . . . . .	17
processAbunAnom . . . . .	18
sample_collections . . . . .	19
sample_pools . . . . .	20
sample_spatial . . . . .	21
<b>Index</b>	<b>22</b>

---

getAbundance	<i>Calculate abundance</i>
--------------	----------------------------

---

## Description

Calculates abundance with option for filtering data and grouping results

## Usage

```
getAbundance(
  collections,
  interval,
  agency = NULL,
  species = NULL,
  trap = NULL,
  sex = "female",
  trapnight_min = 1,
  trapnight_max = NULL,
  separate_by = NULL
)
```

**Arguments**

collections	Collections data retrieved from getArthroCollections()
interval	Calculation interval for abundance, accepts "CollectionDate", "Week", "Biweek", or "Month.
agency	An optional vector for filtering agency by character code
species	Character vector for filtering species. View species in your data 'unique(data\$species_display_name)'. Defaults to all species if no selection
trap	Character vector for filtering trap type by acronym. View trap types in your data 'unique(data\$trap_acronym)'. Defaults to all trap types
sex	Character vector for filtering sex type. View sex options 'unique(data\$sex_type)'. Defaults to "female".
trapnight_min	Minimum trap night restriction for calculation. Default is 1.
trapnight_max	Maximum trap night restriction for calculation. Default is no restriction.
separate_by	Separate/group the calculation by 'trap', 'species', 'agency', 'county', or 'spatial'. Default NULL does not separate.

**Value**

A dataframe of abundance calculations.

**Examples**

```
getAbundance(sample_collections,
             interval = 'Week',
             species = list('Cx pipiens'),
             trap = list('GRVD', 'CO2'),
             sex = list("female"),
             trapnight_min = 1,
             trapnight_max = 5,
             separate_by = "species")
```

---

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**

```
getAbundanceAnomaly(
  collections,
  interval,
  target_year,
  agency = NULL,
  species = NULL,
  trap = NULL,
  sex = "female",
  trapnight_min = 1,
  trapnight_max = NULL,
  separate_by = NULL
)
```

**Arguments**

collections	Collections data retrieved from 'getArthroCollections()'
interval	Calculation interval for abundance, accepts "CollectionDate", "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)
agency	An optional vector for filtering agency by character code
species	Character vector for filtering species. View species in your data 'unique(data\$species_display_name)'. Defaults to all species if no selection
trap	Character vector for filtering trap type by acronym. View trap types in your data 'unique(data\$trap_acronym)'. Defaults to all trap types
sex	Character vector for filtering sex type. View sex options 'unique(data\$sex_type)'. Defaults to "female".
trapnight_min	Minimum trap night restriction for calculation. Default is 1.
trapnight_max	Maximum trap night restriction for calculation. Default is no restriction.
separate_by	Separate/group the calculation by 'trap', 'species', 'agency', 'county', or 'spatial'. Default NULL does not separate.

**Value**

Abundance anomaly calculation

**Examples**

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

---

getAgency	<i>Get Agency data</i>
-----------	------------------------

---

**Description**

Retrieves VectorSurv agency data with shape coordinates converted to sf geometry

**Usage**

```
getAgency(token)
```

**Arguments**

token            Access token retrieved from 'getToken()'

**Value**

An sf object containing agency data with geometry

**Examples**

```
## Not run:  
token <- getToken()  
agency <- getAgency(token)  
  
## End(Not run)
```

---

getArthroCollections	<i>Get arthropod collections data</i>
----------------------	---------------------------------------

---

**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_ids = NULL,  
  spatial_features = NULL,  
  geocoded = FALSE  
)
```

**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_ids	Filter on agency id, default to NULL for all available agencies, otherwise provide a vector of agency ids, such as 'agency_ids = c(55,56)'
spatial_features	Filter data by spatial feature
geocoded	Should city and county be calculated by collection long/lat instead of user input from site inform

**Value**

A dataframe of collections data

**Examples**

```
## Not run:
token = getToken()
collections = getArthroCollections(token, 2021, 2022, 'mosquito', c(55,56), TRUE)
## End(Not run)
```

---

getInfectionRate	<i>Calculate infection rate</i>
------------------	---------------------------------

---

**Description**

'getInfectionRate()' Calculates infection rate from pools data

**Usage**

```
getInfectionRate(
  pools,
  interval,
  target_disease,
  pt_estimate = "bc-mle",
  scale = 1000,
  agency = NULL,
  species = NULL,
  trap = NULL,
  sex = "female",
  separate_by = NULL
)
```

**Arguments**

pools	Pools data retrieved from 'getPools()'
interval	Calculation interval for infection rate, accepts "CollectionDate", "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\$test_target_acronym)'
pt_estimate	The estimation type for infection rate. Options include: "mle", "bc-mle", "mir"
scale	Constant to multiply infection rate by
agency	An optional vector for filtering agency by character code
species	An optional vector for filtering species. 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	An optional vector for filtering trap type by acronym. Trap_acronym is the accepted notation. Run unique(collections\$trap_acronym) to see trap types present in your data. If trap is unspecified, the default NULL will return data for all trap types.
sex	An optional vector for filtering sex type. Accepts 'male', 'female', or 'other'. If sex is unspecified, the default NULL will return data for female sex.
separate_by	Separate/group the calculation by 'trap', 'species' or 'agency'. Default NULL does not separate.

**Value**

Dataframe of infection rate calculation

---

getPools	<i>Get Pools data</i>
----------	-----------------------

---

**Description**

Retrieves VectorSurv pools data for desired year range

**Usage**

```
getPools(token, start_year, end_year, arthropod, agency_ids = NULL)
```

**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'
agency_ids	Filter on agency id, default to NULL for all available agencies, otherwise provide a vector of agency ids

**Value**

Dataframe of pools data

**Examples**

```
## Not run:
token = getToken()
getPools(token, start_year = 2020, end_year = 2021, arthropod = 'tick', 55)
## End(Not run)
```

---

getPoolsComparisionTable

*Get Pools Frequency Table*

---

**Description**

'getPoolsComparisionTable()' 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**

```
getPoolsComparisionTable(pools, interval, target_disease, separate_by = NULL)
```

**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)'
separate_by	Separate/group the calculation by 'trap', 'species' or 'agency'. Default NULL does not separate.

**Value**

Frequency table of for pools data

**Examples**

```
getPoolsComparisionTable(sample_pools,
                          interval = "Biweek",
                          target_disease = "WNV",
                          separate_by = "species")
```

---

getRegions	<i>Get region data</i>
------------	------------------------

---

**Description**

'getRegions()' obtains region data for authorized VectorSurv Gateway accounts.

**Usage**

```
getRegions(token)
```

**Arguments**

token                    A valid access token returned from 'getToken()'

**Value**

A dataframe of region data, used internally to merge spatial information to collections

---

getSites	<i>Get sites data</i>
----------	-----------------------

---

**Description**

'getSites()' obtains site data for authorized VectorSurv Gateway accounts.

**Usage**

```
getSites(token, agency_ids = NULL)
```

**Arguments**

token                    A valid access token returned from 'getToken()'  
agency\_ids                Filter on agency id, default to NULL for all available agencies, otherwise provide a vector of agency ids, such as 'agency\_ids = c(55,56)'

**Value**

A dataframe of site data

**Examples**

```
## Not run:  
token = getToken()  
sites = getSites(token)  
## End(Not run)
```

---

getSpatialFeatures      *Get Spatial Features data*

---

**Description**

Retrieves VectorSurv spatial features data with optional agency filtering

**Usage**

```
getSpatialFeatures(token, agency_ids = NULL)
```

**Arguments**

token                  Access token retrieved from 'getToken()'  
agency\_ids            Optional vector of agency IDs to filter by

**Value**

An sf object containing spatial features data

**Examples**

```
## Not run: token=getToken()  
spatial = getSpatialFeatures(token)  
## End(Not run)
```

---

getSpeciesTable          *Get Table For Species*

---

**Description**

Allows users to create a custom table for vector species. The invasive species count, trap nights, and abundance are calculated by chosen groupings or time intervals. The table has output options based on document format.

**Usage**

```
getSpeciesTable(  
  token,  
  interval = NULL,  
  target_year,  
  cumulative = FALSE,  
  include_abundance = FALSE,  
  species = NULL,  
  trap = NULL,  
  sex = "female",
```

```

    separate_by = NULL,
    output_format = "auto",
    caption = NULL,
    agency_id = NULL
  )

```

### Arguments

token	Authentication token for API access, obtained from 'getToken()'
interval	Time interval for aggregation. One of: "CollectionDate", "Week", "Biweek", "Month", or NULL for year-level summary
target_year	The focal year to highlight in the plot (integer)
cumulative	T/F Adds columns for the cumulative sum count Default to FALSE
include_abundance	T/F Adds column for abundance (non-cumulative only)
species	Character vector for filtering species. View species in your data 'unique(data\$species_display_name)'. Defaults to all species if no selection
trap	Character vector for filtering trap type by acronym. View trap types in your data 'unique(data\$trap_acronym)'. Defaults to all trap types
sex	Character vector for filtering sex type. View sex options 'unique(data\$sex_type)'. Defaults to "female".
separate_by	Adds a column from the data as a grouping variable. Accepts 'site', 'city', 'county', 'agency', 'trap', 'spatial'
output_format	Format to output table display. Accepts 'html', 'pdf', 'word' or 'auto'. Default auto returns the console format dataframe.
caption	Caption for table
agency_id	Agency identifier to filter data if applicable

### Value

A table displaying invasive species interval or/and cumulative counts over time

---

getToken	<i>Get authentication token</i>
----------	---------------------------------

---

### Description

getToken() 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

```
getToken()
```

**Value**

User token

**Examples**

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

---

<code>getVectorIndex</code>	<i>Calculate vector index</i>
-----------------------------	-------------------------------

---

**Description**

`'getVectorIndex()'` Calculates vector index from pools and collections data

**Usage**

```
getVectorIndex(
  collections,
  pools,
  interval,
  target_disease,
  pt_estimate = "bc-mle",
  scale = 1000,
  agency = NULL,
  species = NULL,
  trap = NULL,
  sex = NULL,
  trapnight_min = 1,
  trapnight_max = NULL,
  separate_by = NULL
)
```

**Arguments**

<code>collections</code>	Collections data retrieved from <code>'getArthroCollections()'</code>
<code>pools</code>	Pools data retrieved from <code>'getPools()'</code>
<code>interval</code>	Calculation interval for vector index, accepts "CollectionDate", "Biweek", "Week", and "Month"
<code>target_disease</code>	The disease to calculate infection rate for—i.e. "WNV". Disease acronyms are the accepted input. To see a list of disease acronyms, run <code>'unique(pools\$target_acronym)'</code>
<code>pt_estimate</code>	The estimation type for infection rate. Options include: "mle", "bc-mle", "mir"
<code>scale</code>	Constant to multiply infection rate, default is 1000
<code>agency</code>	Character vector for filtering agency by character code
<code>species</code>	Character vector for filtering species. View species in your data <code>'unique(data\$species_display_name)'</code> . Defaults to all species if no selection

trap	Character vector for filtering trap type by acronym. View trap types in your data <code>unique(data\$trap_acronym)</code> . Defaults to all trap types
sex	Character vector for filtering sex type. View sex options <code>unique(data\$sex_type)</code> . Defaults to "female".
trapnight_min	Minimum trap night restriction for calculation. Default is 1.
trapnight_max	Maximum trap night restriction for calculation. Default is no restriction.
separate_by	Separate/group the calculation by 'trap', 'species' or 'agency'. Default NULL does not separate.

**Value**

Dataframe containing the vector index calculation

**Examples**

```
getVectorIndex(collections=sample_collections,
pools=sample_pools, interval="Month", target_disease = "WNV", pt_estimate="mle")
```

---

plotInvasiveSums      *Plot cumulative sum and interval sums of invasive species*

---

**Description**

This function graphs the cumulative sum of invasive mosquito specie(s) as a line chart over a selected interval. The bar chart shows the interval sum of invasive specie(s). For cumulative sum, a convex slope indicates increased rate of species trappings, concave a decrease in rate, and liner a constant rate.

**Usage**

```
plotInvasiveSums(
  token,
  interval,
  target_year,
  invasive_species,
  agency_ids = NULL
)
```

**Arguments**

token	A valid access token returned from <code>getToken()</code>
interval	Calculation interval for abundance, accepts "CollectionDate", "Week", "Biweek", or "Month.
target_year	Year to plot
invasive_species	Names of invasive species to be plotted, multiple invasive species are entered in vector format i.e. <code>invasive_species = c("Ae aegypti", "Ae albopictus")</code>
agency_ids	An optional vector for filtering agency by id

**Value**

ggplot object

---

plotPoolsMap

*Create Interactive Map of Mosquito Pool Testing Data*

---

**Description**

Generates an interactive Leaflet map visualizing mosquito pool testing results by location, species, and test outcome. The map displays positive and negative pools, monitoring sites, and optional spatial features with filtering capabilities.

**Usage**

```
plotPoolsMap(
  token,
  target_year,
  target_disease = NULL,
  species = NULL,
  trap = NULL,
  agency_ids = NULL,
  time_period = NULL,
  interval = "Month",
  basemap = "Topographic",
  spatial_features = NULL,
  show_agency_boundaries = TRUE,
  html = TRUE,
  height = 400
)
```

**Arguments**

token	Authentication token for API access (character)
target_year	Target surveillance year for data retrieval (numeric)
target_disease	Optional vector of disease acronyms to filter results (character)
species	Optional vector of species names to filter results (character)
trap	Optional vector of trap acronyms to filter results (character)
agency_ids	Optional vector of agency IDs to filter results (character)
time_period	Optional time period to filter results within the year (character)
interval	Time interval for grouping data: "Week", "Biweek", "Month", or "Year" (character, default = "Month")
basemap	Base map type: "Topographic", "Satellite", "Terrain", or "OpenTopoMap" (character, default = "Topographic")

spatial_features	Optional vector of spatial feature names to display (character)
show_agency_boundaries	Whether to display agency boundaries (logical, default = TRUE)
html	Logical. TRUE (default) returns an interactive leaflet widget for HTML output. Set to FALSE when knitting to Word or PDF to return a static screenshot instead.
height	Pixel height of the screenshot when html = FALSE. Default is 400.

### Value

A leaflet map object (html = TRUE) or a static image (html = FALSE).

### Examples

```
## Not run:
# Basic map for a target year
plotPoolsMap(token, target_year = 2023)

# Filter by disease and time period
plotPoolsMap(
  token      = token,
  target_year = 2023,
  target_disease = "WNV",
  interval    = "Month",
  time_period  = "July"
)

# With spatial features and satellite basemap
plotPoolsMap(
  token      = token,
  target_year = 2023,
  agency_ids  = 55,
  spatial_features = c("Zone3", "Downtown"),
  basemap    = "Satellite"
)

# Static output for Word or PDF reports
plotPoolsMap(
  token      = token,
  target_year = 2023,
  target_disease = "WNV",
  html       = FALSE,
  height     = 500
)

## End(Not run)
```

---

plotSpeciesMap

*Create Interactive Map of Mosquito Species Collection Data*


---

### Description

Generates an interactive Leaflet map visualizing mosquito species collection data and monitoring sites from surveillance activities. Displays collection locations colored by species found in that location during the indicated time period, with the option to include spatial features

### Usage

```
plotSpeciesMap(
  token,
  target_year,
  species = NULL,
  trap = NULL,
  agency_ids = NULL,
  interval = NULL,
  time_period = NULL,
  basemap = "Topographic",
  spatial_features = NULL,
  html = TRUE,
  height = 400
)
```

### Arguments

token	Authentication token for API access retrieved from getToken()
target_year	Target surveillance year for data retrieval
species	Optional vector of species names to filter results. View with 'unique(data\$species_display_name)'
trap	Optional vector of trap acronyms to filter results. View available trap types with 'unique(data\$trap_acronym)'
agency_ids	Optional vector of agency IDs to filter results when access to multiple agencies
interval	Time interval to select time period from: "Week", "Biweek", "Month" or NULL. If NULL data is for entire year.
time_period	Optional time period to filter results within the designated interval. Can be numeric (1-12 for months) or character (e.g., "July", "Week 5")
basemap	Base map design: "Topographic", "Satellite", or "Terrain"
spatial_features	Optional vector of spatial feature names to display as polygons. View available features using 'getSpatialFeatures()'
html	T/F Is this map for use in an html document? If not, map will be converted in to a screenshot
height	T/F Relevant for non html files. Sets the height of the map image

**Value**

Returns a map object visualizing mosquito species collection data. Returns NULL if no data is available after filtering.

---

plotVectorAbundance *Plot Vector Abundance Over Time with Comparison to Historical Averages*

---

**Description**

This function creates a time series plot showing vector (mosquito) abundance over multiple years, highlighting a target year in red and optionally showing a historical average line. The plot helps visualize trends and anomalies in vector populations.

**Usage**

```
plotVectorAbundance(
  token,
  interval,
  target_year,
  start_year = NULL,
  end_year = NULL,
  species,
  trap = NULL,
  agency_ids = NULL,
  sex = "female",
  trapnight_min = 1,
  trapnight_max = NULL
)
```

**Arguments**

token	Authentication token for API access, typically obtained from <code>'getToken()'</code>
interval	Time interval for aggregation. One of: "CollectionDate", "Week", "Biweek", "Month"
target_year	The focal year to highlight in the plot
start_year	Starting year for data retrieval. If NULL, defaults to 5 years before target_year
end_year	Ending year for data retrieval (integer). If NULL, defaults to target_year
species	Character vector for filtering species. View species in your data <code>'unique(data\$species_display_name)'</code> . Defaults to all species if no selection
trap	Character vector for filtering trap type by acronym. View trap types in your data <code>'unique(data\$trap_acronym)'</code> . Defaults to all trap types
agency_ids	Agency identifier to filter data if applicable

sex	Character vector for filtering sex type. View sex options ‘unique(data\$sex_type‘). Defaults to "female".
trapnight_min	Minimum number of trap nights to include (numeric). Defaults to 1
trapnight_max	Maximum number of trap nights to include (numeric). If NULL, uses maximum available

### Value

A ggplot object showing vector abundance over time, with the target year highlighted in red and historical average shown in blue

### Examples

```
## Not run:
# Basic usage with default parameters
token <- getToken()
plot <- plotVectorAbundance(
  token = token,
  interval = "Week",
  target_year = 2024,
  species = "Cx pipiens",
  trap = "C02"
)

# Compare multiple years with custom date range
plot <- plotVectorAbundance(
  token = token,
  interval = "Month",
  target_year = 2024,
  start_year = 2020,
  end_year = 2024,
  species = "Ae aegypti",
  trap = c("BG", "CDC"),
  sex = "female",
  trapnight_min = 3
)

## End(Not run)
```

---

processAbunAnom

*Process abundance anomaly*

---

### Description

‘processAbunAnom()‘ processes the output returned from ‘getAbundanceAnomaly()‘ into a long form suitable for plotting using ‘ggplot’

**Usage**

```
processAbunAnom(AbAnomOutput)
```

**Arguments**

AbAnomOutput    output from 'getAbunAnom()'

**Value**

Abundance anomaly output processed into long form, used for plotting functions

---

sample\_collections    *Sample Mosquito Collections Data*

---

**Description**

Sample Mosquito Collections data imitates the essential components of real mosquito collections data

**Usage**

```
sample_collections
```

**Format**

A data frame with 5880 rows and 17 variables:

agency\_code    character Four letter agency code  
 agency\_id    interger Unique agency id number  
 county    character County which site resides  
 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  
 site\_code    integer Identifying code of 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  
 collection\_longitude    numeric longitude of collection  
 collection\_latitude    numeric latitude of collection  
 spatial\_feature    character name of spatial feature to which data belongs  
 multiple\_features    boolean T/F if the point is found within multiple selected spatial features

**Source**

<https://vectorsurv.org/>

---

sample_pools	<i>Sample Pools Data</i>
--------------	--------------------------

---

**Description**

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

**Usage**

sample\_pools

**Format**

A data frame with 3500 rows and 15 variables:

agency\_code character Four letter agency code  
 agency\_id interger Unique agency id number  
 id integer Pool identification number  
 surv\_year integer Surveillance year of pool  
 site\_code integer Identifying code of site  
 collection\_date character The date the trap was picked up for collection  
 sex\_type integer Sex type of collected arthropods  
 num\_count integer Number of arthropods present in collection  
 test\_target\_acronym character The disease being tested for in the pool  
 test\_method\_name character Method used to test pool for disease  
 test\_status\_name character Status of the tested disease, confirmed or negative  
 trap\_acronym character The acronym of the trap placed in the field  
 species\_display\_name character Species name of collected arthropods  
 pool\_longitude numeric longitude of pool  
 pool\_latitude numeric latitude of pool

**Source**

<https://vectorsurv.org/>

---

sample_spatial	<i>Sample Spatial Data</i>
----------------	----------------------------

---

**Description**

Sample Spatial data imitates spatial feature data

**Usage**

```
sample_spatial
```

**Format**

A data frame with 3 rows and 5 variables:

agency character Agency name

agency\_id interger Unique agency id number

id interger ID of spatial feature

name character Name of spatial feature

geometry multipolygon shape geometry spatial feature

**Source**

<https://vectorsurv.org/>

# Index

- \* **abundance**
  - getAbundanceAnomaly, 3
- \* **authentication**
  - getToken, 11
- \* **datasets**
  - sample\_collections, 19
  - sample\_pools, 20
  - sample\_spatial, 21
- \* **infection**
  - getInfectionRate, 6
- \* **pools**
  - getInfectionRate, 6
  - getPools, 7
- \* **rate**
  - getInfectionRate, 6

getAbundance, 2  
getAbundanceAnomaly, 3  
getAgency, 5  
getArthroCollections, 5  
getInfectionRate, 6  
getPools, 7  
getPoolsComparisionTable, 8  
getRegions, 9  
getSites, 9  
getSpatialFeatures, 10  
getSpeciesTable, 10  
getToken, 11  
getVectorIndex, 12

plotInvasiveSums, 13  
plotPoolsMap, 14  
plotSpeciesMap, 16  
plotVectorAbundance, 17  
processAbunAnom, 18

sample\_collections, 19  
sample\_pools, 20  
sample\_spatial, 21