Tutortial Documentation

2023-10-17

Introduction

Source files, directory

```
#To get current directory
getwd()
```

```
## [1] "C:/Users/Christina/Desktop/R-tutorial"
```

```
#'VS_functions.R' must be in the same directory as the .rmd files which access its functions, you can s

#Set below to the location of the downloaded files on your machine
setwd("/Users/Christina/Desktop/R-tutorial")

#loads relevant packages and functions
source("VS_functions.R")
```

Retriving Data

getArthroCollections(. . .)

Description

getArthroCollections(. . .) obtains collections data on a year range (start_year, end_year) and agency_id. It prompts the user for their Gateway username and password before retrieving the associated data. Agency id is the number associated with your agency through Vectorsurv. If you have access to multiple agencies, id can be used to specific what data you wish to retrieve. You can only retrieve data from agencies linked to your Gateway account.

Usage

getArthroCollections(start_year, end_year, agency_code)

Arguments

- start_year: Beginning of year range
- end_year: End of year range
- agency_code: Agency ID number

```
#Example
collections = getArthroCollections(2022,2023, 55)
```

getPools(. . .)

Description

getPools(. . .) similar to getArthroCollections() obtains pools on a year range (start_year, end_year) and agency_id. It prompts the user for their Gateway username and password before retrieving the associated data. getPools() retrieve data for both mosquito and tick pools.

Usage

getPools(start_year, end_year, agency_code)

Arguments

- start_year: Beginning of year range
- end_year: End of year range
- agency_code: Agency ID number

```
#Example
pools = getPools(2022,2023, 55)
```

Write Data to file

write.csv(. . .)

read.csv(. . .)

Basic subseting and filtering

```
#Subset using column names or index
colnames(collections)
```

```
##  [1] "collection_id"            "collection_num"
##  [3] "collection_date"          "collection_date_date_only"
##  [5] "comments"                 "identified_by"
##  [7] "num_trap"                 "site"
##  [9] "surv_year"                "trap_nights"
## [11] "trap_problem_bit"         "user"
## [13] "add_date"                 "deactive_date"
## [15] "updated"                  "id"
## [17] "num_count"                "sex_id"
## [19] "sex_type"                 "sex_name"
## [21] "species_id"               "species_full_name"
## [23] "species_display_name"     "agency_id"
## [25] "agency_code"              "agency_name"
## [27] "trap_id"                  "trap_acronym"
## [29] "trap_name"                "trap_presence"
```

```
collections[c("collection_date","species_display_name","num_count")]
```

```
## # A tibble: 48,058 x 3
##    collection_date          species_display_name num_count
```

```
##    <chr>                     <chr>                    <int>
##  1 2023-10-18T07:00:00.000Z Ae melanimon                 1
##  2 2023-10-18T07:00:00.000Z An freeborni                 2
##  3 2023-10-18T07:00:00.000Z Cx pipiens                  17
##  4 2023-10-18T07:00:00.000Z An freeborni                 1
##  5 2023-10-18T07:00:00.000Z Cx pipiens                   1
##  6 2023-10-18T07:00:00.000Z Cx pipiens                  15
##  7 2023-10-18T07:00:00.000Z Cx tarsalis                  1
##  8 2023-10-18T07:00:00.000Z Cx tarsalis                  1
##  9 2023-10-18T07:00:00.000Z Ae aegypti                   7
## 10 2023-10-18T07:00:00.000Z Ae aegypti                  14
## # i 48,048 more rows
```

```
#We can filter using column names


filter(collections,trap_nights==1)
```

```
## # A tibble: 33,773 x 30
##    collection_id collection_num collection_date  collection_date_date~1 comments
##            <int>          <int> <chr>            <lgl>                  <chr>
##  1       3031264           8282 2023-10-18T07:0~ TRUE                   "Week 6~
##  2       3031264           8282 2023-10-18T07:0~ TRUE                   "Week 6~
##  3       3031264           8282 2023-10-18T07:0~ TRUE                   "Week 6~
##  4       3031280           8283 2023-10-18T07:0~ TRUE                   "Week 6~
##  5       3031280           8283 2023-10-18T07:0~ TRUE                   "Week 6~
##  6       3031280           8283 2023-10-18T07:0~ TRUE                   "Week 6~
##  7       3031280           8283 2023-10-18T07:0~ TRUE                   "Week 6~
##  8       3031280           8283 2023-10-18T07:0~ TRUE                   "Week 6~
##  9       3031280           8283 2023-10-18T07:0~ TRUE                   "Week 6~
## 10       3031280           8283 2023-10-18T07:0~ TRUE                   "Week 6~
## # i 33,763 more rows
## # i abbreviated name: 1: collection_date_date_only
## # i 25 more variables: identified_by <chr>, num_trap <int>, site <int>,
## #   surv_year <int>, trap_nights <int>, trap_problem_bit <lgl>, user <int>,
## #   add_date <chr>, deactive_date <lgl>, updated <chr>, id <int>,
## #   num_count <int>, sex_id <int>, sex_type <chr>, sex_name <chr>,
## #   species_id <int>, species_full_name <chr>, species_display_name <chr>, ...
```

Calculations

Abundance

getAbundance(. . .)

Description

getAbundance(. . .) uses any amount of arthro collections data to calculate the abundance for the specified parameters. The function calculates using the methods of the Gateway Abundance calculator.

Usage

getAbundance(collections,interval, species_list = NULL, trap_list = NULL, species_seperate = 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$species_display_name). If species is unspecified, the default NULL will return data for all species in data.
- trap_list: Trap filter for calculating abundance. Trap_acronym is the is the accepted notation. Run unique(collections$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_seperate: 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.

```
getAbundance(collections, interval = "Biweek", species_list = c("Cx tarsalis", "Cx pipiens"), trap_list
```

```
##    EPIYEAR Biweek Count Trap_Events Abundance
## 1     2023     10   882          65     13.57
## 2     2023     11  3254         142     22.92
## 3     2023     12  4395         153     28.73
## 4     2023     13 15803         182     86.83
## 5     2023     14 24939         226    110.35
## 6     2023     15 24113         217    111.12
## 7     2023     16 19062         255     74.75
## 8     2023     17 12865         226     56.92
## 9     2023     18 10088         213     47.36
## 10    2023     19  7161         211     33.94
## 11    2023     20  5934         211     28.12
## 12    2023     21  2518         123     20.47
## 13    2022      4     9           3      3.00
## 14    2022      9  1358         126     10.78
## 15    2022     10  1202         133      9.04
## 16    2022     11  1969         145     13.58
## 17    2022     12  3503         159     22.03
## 18    2022     13  5630         159     35.41
## 19    2022     14 10444         154     67.82
## 20    2022     15  9722         178     54.62
## 21    2022     16  7949         186     42.74
## 22    2022     17  6501         180     36.12
## 23    2022     18  6038         166     36.37
## 24    2022     19  3798         163     23.30
## 25    2022     20  1869         120     15.57
## 26    2022     21  1189          84     14.15
```

Abundance Anomaly (comparison to 5 year average)

getAbundanceAnomaly()

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, species_list = NULL, trap_list = NULL, species_seperate = 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$species_display_name). If species is unspecified, the default NULL will return data for all species in data.
- trap_list: Trap filter for calculating abundance. Trap_acronym is the is the accepted notation. Run unique(collections$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_seperate: 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.

```
collections = getArthroCollections(2018,2023, 55)

getAbundanceAnomaly(collections, interval = "Biweek",target_year = 2023, species_list = c("Cx tarsalis"
```

```
##    Biweek EPIYEAR Count Trap_Events Abundance Five_Year_Avg  Delta
## 1      10    2023   882          65     13.57        12.926   4.98
## 2      11    2023  3254         142     22.92        19.666  16.55
## 3      12    2023  4395         153     28.73        37.988 -24.37
## 4      13    2023 15803         182     86.83        54.496  59.33
## 5      14    2023 24939         226    110.35        81.972  34.62
## 6      15    2023 24113         217    111.12        75.588  47.01
## 7      16    2023 19062         255     74.75        78.528  -4.81
## 8      17    2023 12865         226     56.92        66.406 -14.28
## 9      18    2023 10088         213     47.36        61.704 -23.25
## 10     19    2023  7161         211     33.94        51.736 -34.40
## 11     20    2023  5934         211     28.12        32.970 -14.71
## 12     21    2023  2518         123     20.47        20.082   1.93
```

Infection Rate

getInfectionRate()

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

getInfectionRate(pools,interval, target_year, target_disease,pt_estimate, species_list = c(NULL), trap_list = c(NULL))

Arguments

- pools: Pools data retrieved from getPools(...)

- interval: Calculation interval for abundance, accepts “collection_date”,“Biweek”,“Week”, and “Month.
- target_year: Year to calculate infection rate for. This year must be present in the data.
- 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$target_acronym).
- pt_estimate: The estimation type for infection rate. Options include: “mle”,“bc-”mle”, “mir.”
- species_list: Species filter for calculating abundance. Species_display_name is the accepted notation. To see a list of species present in your data run unique(pools$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 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.

```
IR = getInfectionRate(pools, interval = "Week", target_year = 2023, target_disease = "WNV", pt_estimate
IR
```

```
##    Year Week Disease Point_Estimate   Lower_CI  Upper_CI
## 1  2023   20     WNV      0.0000000 0.00000000  4.617179
## 2  2023   21     WNV      0.0000000 0.00000000  4.119261
## 3  2023   22     WNV      0.0000000 0.00000000  3.156551
## 4  2023   23     WNV      0.5727378 0.03289081  2.738134
## 5  2023   24     WNV      0.0000000 0.00000000  1.747176
## 6  2023   25     WNV      0.4806543 0.02755773  2.306985
## 7  2023   26     WNV      3.3477400 1.86750670  5.545080
## 8  2023   27     WNV      5.7178780 3.09721593  9.675510
## 9  2023   28     WNV      6.7309218 4.23336858 10.151171
## 10 2023   29     WNV     11.2854003 7.75018423 15.810330
## 11 2023   30     WNV     14.7000473 9.47311042 21.664153
## 12 2023   31     WNV      9.2984320 5.96219856 13.772354
## 13 2023   32     WNV     12.4179889 7.25814203 19.769851
## 14 2023   33     WNV     13.7153647 8.36519124 21.080100
## 15 2023   34     WNV     12.9772471 8.09345084 19.628633
## 16 2023   35     WNV      8.6523803 3.54402123 17.698274
## 17 2023   36     WNV      8.1714178 4.17883093 14.386721
## 18 2023   37     WNV      3.4384086 1.27892410  7.512439
## 19 2023   38     WNV      2.7297894 0.72200878  7.245097
## 20 2023   39     WNV      0.0000000 0.00000000  2.223982
## 21 2023   40     WNV      1.8502043 0.33078143  5.950046
## 22 2023   41     WNV      1.0256410 0.05906777  4.851477
## 23 2023   42     WNV      0.0000000 0.00000000 20.603165
```

Vector Index

getVectorIndex()

```
#TODO
```

Tables

getPoolsComparisionTable()

Description

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

Usage

getPoolsComparisionTable(pools,target_disease, species_seperate=F)

Arguments

- pools: Pools data retrieved from getPools(...)
- 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_seperate: Should the pools comparison be split by species of each pool. Default is FALSE.

```
getPoolsComparisionTable(pools, "WNV", species_seperate = T)
```

```
## # A tibble: 7 x 6
## # Groups:   surv_year, species_display_name [7]
##   surv_year species_display_name Negative Confirmed Total PercentPositive
##       <int> <chr>                   <int>     <int> <int>           <dbl>
## 1      2022 An franciscanus             1         0     1            0
## 2      2022 Cx pipiens               3164        38  3202            1.19
## 3      2022 Cx stigmatosoma             2         0     2            0
## 4      2022 Cx tarsalis              2070        69  2139            3.23
## 5      2023 An freeborni                1         0     1            0
## 6      2023 Cx pipiens               3787       240  4027            5.96
## 7      2023 Cx tarsalis              3406       419  3825           11.0
```

Charts

When plotting with libraries in R, it is easiest when the data is prepared in long form.

ProcessAbunAnom()

Description

ProcessAbunAnom() processes the output returned from getAbundanceAnomaly() into a long form suitable for plotting in ggplot.

Usage

ProcessAbunAnom(AbAnomOutput)

Arguments

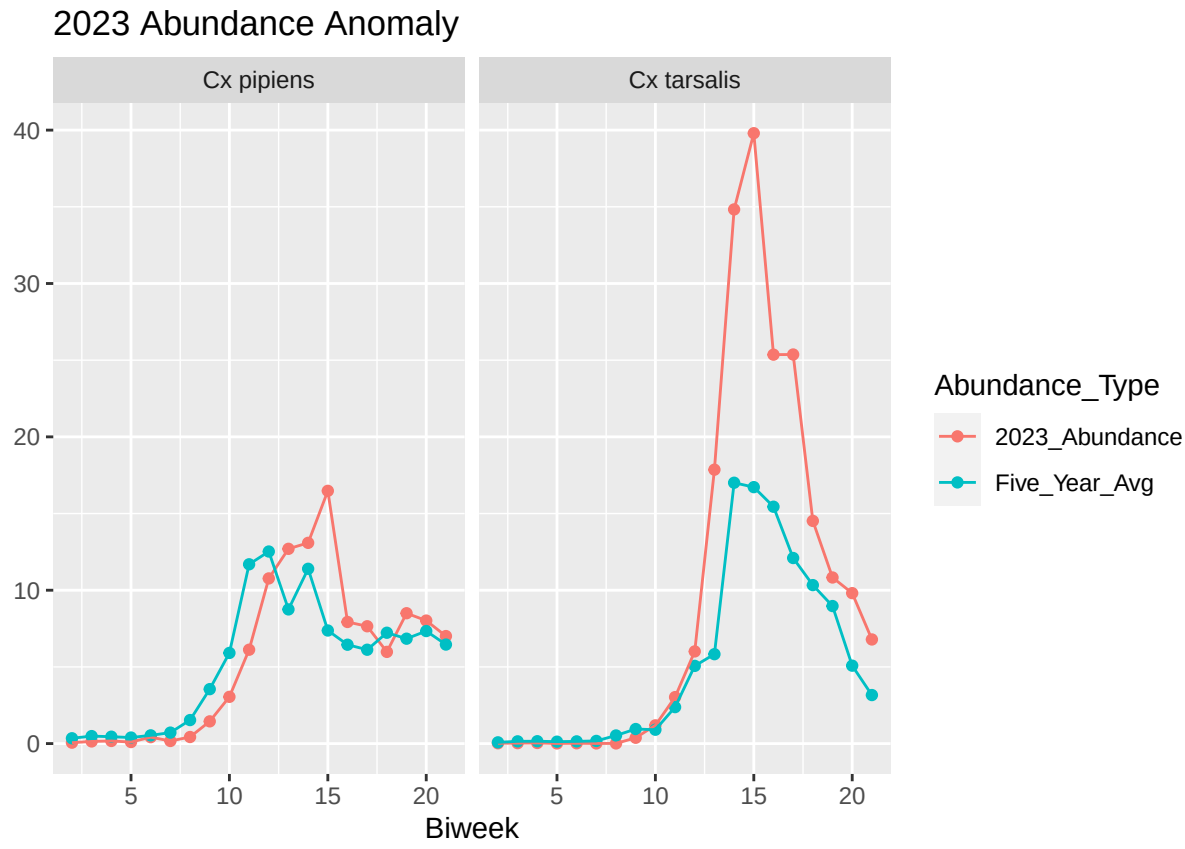
- AbAnomOutput: Output from returned getAbundanceAnomaly()

```
AbAnOut = getAbundanceAnomaly(collections, interval = "Biweek",target_year = 2023, species_list = c("Cx


AbAnOut_L = ProcessAbunAnom(AbAnOut)

#Example using ggplot
AbAnOut_L %>%  filter(Abundance_Type %in% c("2023_Abundance",
                                            "Five_Year_Avg"))%>%
```

```
ggplot(aes(x=Biweek,
           y= Abundance_Calculation,
           color = Abundance_Type)) +
geom_point()+
geom_line() +
facet_wrap(~species_display_name)+
labs(title="2023 Abundance Anomaly",y = "")
```



plotInfectionRate()

Description

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

Usage

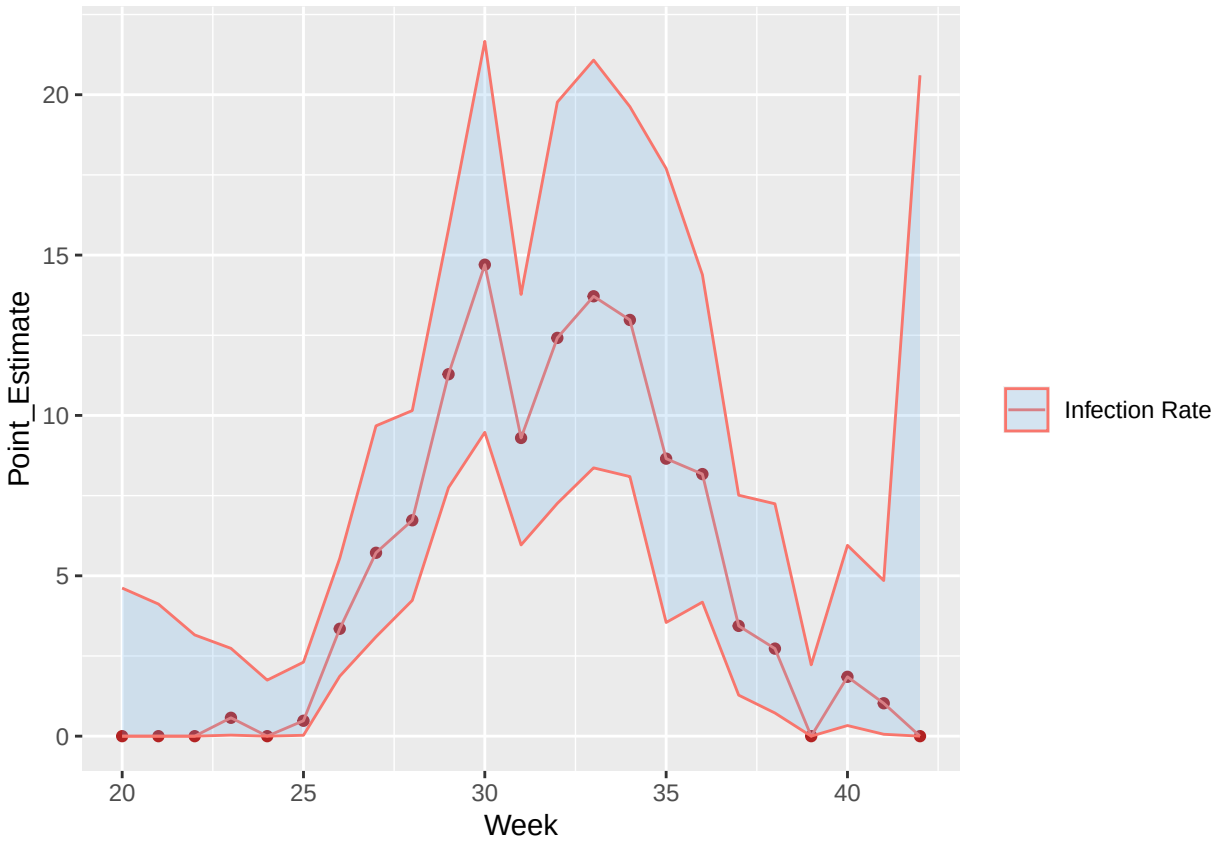
plotInfectionRate(InfRtOutput)

Arguments

- InfRtOutput: Output from returned getInfectionRate()

```
IR = getInfectionRate(pools, interval = "Week", target_year = 2023, target_disease = "WNV", pt_estimate

plotInfectionRate(InfRtOutput = IR)
```


Point_Estimate
20
15
10
5
0
20
25
30
35
40
Week
Infection Rate