-Read Me-

Chronic Wasting Disease Surveillance Expense Cap Software

Cornell Wildlife Health Lab,
Animal Health Diagnostic Center,
College of Veterinary Medicine,
Cornell University
Ithaca, New York, USA

Contacts

Software programmers:
Cara E. Them, *caraethem@gmail.com*
Brenda J. Hanley, *bjh262@cornell.edu*
Corey I. Mitchell, *coreyirene@gmail.com*

Research collaborators:
Md Sohel Ahmed, *msa248@cornell.edu*
Roderick B. Gagne, *rgagne@vet.upenn.edu*

Research supervisor:
Krysten L. Schuler, *ks833@cornell.edu*

Overview

The *Chronic Wasting Disease Surveillance Expense Cap Software* (hereafter, "*Software*") computes the maximal number of samples from white-tailed deer (*Odocoileus virginianus*) that ought to be collected and evaluated for chronic wasting disease (CWD) in each county of a state in each hunting season to maximize the informational return on investment while staying within the total allowable budget. Using an additive approach, the *Software* pinpoints the combination of samples that should be evaluated for two collection categories of deer (hunter harvested and non-hunter harvested) to ensure sampling in the highest risk counties. Inputs to the additive expense cap algorithm in the *Software* broadly include the *Optimization Matrix* and the *Historical Data*, both furnished by the user (i.e., the agency representative of the state of interest). The *Optimization Matrix* includes for each county information such as the costs, benefits, and starting prevalence of CWD in wild herds of deer for each collection category. The *Historical Data* includes for each county information such as details of and results from the sampling strategy enacted by the agency in the prior sampling season. Agencies may further parameterize their algorithm to weight specific management objectives, including monitoring known infections, searching for new infections, and providing detailed information to the public.

The *Software* was initially developed to accommodate the needs of a consortium of state and provincial wildlife agencies that voluntarily opted into the Surveillance Optimization Project for Chronic Wasting Disease (SOP4CWD; CWHL 2022). This release of the *Software* packet includes (redacted) *Optimization Matrices* and (redacted) *Historical Data* from Pennsylvania, US, but the software may be adapted for use in other states and provinces. See the section entitled **Adapting the *Software* to a new state**, below.

User tutorial

Step 0: Download `R` onto your machine. Go to *https://cran.r-project.org/bin/windows/base/old/4.2.1/*. Click on `R-4.2.1-win.exe`. Once downloading is complete, open `R-4.2.1-win.exe` and follow the instructions in the Setup Manager to download `R` onto your computer.

Step 1: Open `Rgui (64-bit)` and verify that you are running the appropriate version of the R Software (R Core Team 2020), which is `R version 4.2.1 (2022-06-23 ucrt) -- "Funny-Looking Kid". Copyright (C) 2022 The R Foundation for Statistical Computing. Platform: x86_64-w64-mingw32/x64 (64-bit).`

Step 2. Install the `renv` package in R. Go to the top menu ribbon, click "`Packages`", then click "`Install package(s)`", select a `CRAN` mirror (any will do) and click "`OK`". In the next window, scroll to "`renv`", then click "`OK`".

Step 3: Create a folder on your computer that will function as your working directory.

3-a: Create a folder, then name it in accordance with your project. This folder will be your working directory.

3-b: Set the path to the working directory in `R` by clicking the R Console, clicking "`File`", clicking "`Change dir…`", then navigating through the file paths to your working directory. Click "`OK`."

3-c: Verify that R has the correct path to your working directory by clicking the R console, typing "`getwd()`", and hitting enter.

Step 4: Prepare the *Software* materials to run on your machine.

4-a: Download, unzip, and save the *Software* contents in your working directory.

4-b: Check that the working directory contains all the files. Type "`dir()`" into the console and check that the files named `_0a_Pennsylvania_Expense_Cap_Spatial_Prep.R`, `_0b_Pennsylvania_Expense_Cap_Cost_Prep.R`, `_0c_Pennsylvania_Expense_Cap_Prevalence_Prep.R`, _0d_Pennsylvania_Expense_Cap_Matrix_Prep.R, _1_Pennsylvania_Expense_Cap_Data_Prep.R, _2_Pennsylvania_Expense_Cap_User_Interface.R, _3_Pennsylvania_Expense_Cap_Algorithm_Two_Categories.R , _4_ Pennsylvania_Expense_Cap_Command_Center.R, All_PennsylvaniaCountySummary2013.txt, All_PennsylvaniaCountySummary2014.txt, All_PennsylvaniaCountySummary2015.txt, All_PennsylvaniaCountySummary2016.txt, All_PennsylvaniaCountySummary2017.txt, All_PennsylvaniaCountySummary2018.txt, All_PennsylvaniaCountySummary2019.txt, All_PennsylvaniaCountySummary2020.txt, All_PennsylvaniaCountySummary2021.txt, All_PennsylvaniaCountySummary2022.txt, Benefits_Early_Detection_Pennsylvania_Mean.csv, Benefits_Monitoring_Pennsylvania_Mean.csv, Cost_Matrix_Pennsylvania.csv, HistoricDataPennsylvania.csv, PennsylvaniaCountySummary2013.txt, PennsylvaniaCountySummary2014.txt, PennsylvaniaCountySummary2015.txt, PennsylvaniaCountySummary2016.txt, PennsylvaniaCountySummary2017.txt, PennsylvaniaCountySummary2018.txt, PennsylvaniaCountySummary2019.txt, PennsylvaniaCountySummary2020.txt, PennsylvaniaCountySummary2021.txt, PennsylvaniaCountySummary2022.txt, progresscoordinates.cvs, a folder named renv, a folder named www, renv.lock, and SampleSize_SRS_Table.txt appear listed on the console. Note: The file named .Rprofile appears in the materials in your working directory, but will not appear in this list on the console.

Step 5: Set up the local environment.

Step 5-a. Open _4_Pennsylvania_Expense_Cap_Command_Center.R in R Gui (64-bit). [Note: The directions that follow will not make sense in RStudio or RVisual]. To open the script, go to the top menu ribbon in R Gui (64-bit), click “File”, “Open script…”, then navigate to _4_Pennsylvania_Expense_Cap_Command_Center.R and click “Open”.

Step 5-b: Click on the window containing the _4_Pennsylvania_Expense_Cap_Command_Center.R command codes. Under the script information, you will find a series of commands to remove extant libraries. Run each of these lines in a series:

```
remove.packages(dplyr)
remove.packages(geodist)
remove.packages(leaflet)
remove.packages(leaflet.extras)
remove.packages(leafpop)
remove.packages(maptools)
remove.packages(raster)
remove.packages(RColorBrewer)
remove.packages(rgdal)
remove.packages(rgeos)
remove.packages(rmapshaper)
remove.packages(sf)
remove.packages(shiny)
remove.packages(shinycssloaders)
remove.packages(shinydashboard)
remove.packages(sp)
remove.packages(stringr)
remove.packages(tigris)
```

To run a line, right-click on the line and click "Run line or selection". Note: These lines of code are meant to clean any extant versions of these packages off your computer so the environment of your current computer cannot interfere with the new environment needed for this script. If a package does not already exist, then running this code will produce an error stating "Error in find.package(pkgs,lib): there is no packages called XXX". This is ideal, as there is no package that could interfere. If a package does already exist, then running this code will produce a message stating "Removing package from XXX." This is also ideal; in that, it removed the package that could interfere.

Step 5-c: Click on the window containing the _4_Pennsylvania_Expense_Cap_Command_Center.R command codes. Hand-enter a "#" to the left of each line, producing this code:

```
#remove.packages(dplyr)
#remove.packages(geodist)
#remove.packages(leaflet)
#remove.packages(leaflet.extras)
#remove.packages(leafpop)
#remove.packages(maptools)
#remove.packages(raster)
#remove.packages(RColorBrewer)
#remove.packages(rgdal)
#remove.packages(rgeos)
#remove.packages(rmapshaper)
#remove.packages(sf)
```

```
#remove.packages(shiny)
#remove.packages(shinycssloaders)
#remove.packages(shinydashboard)
#remove.packages(sp)
#remove.packages(stringr)
#remove.packages(tigris)
```

Step 5-d: Click on the window containing the `_4_Pennsylvania_Expense_Cap_Command_Center.R` command codes. Right-click on the command line that says:
`library(renv)`
and then click "`run line or selection`".

Step 5-e: Click on the window containing the `_4_Pennsylvania_Expense_Cap_Command_Center.R` command codes. Right-click on the command line that says:
`Sys.setenv(RENV_DOWNLOAD_METHOD = "wininet")`
and then click "`run line or selection`".

Step 5-f: Click on the window containing the `_4_Pennsylvania_Expense_Cap_Command_Center.R` command codes. Right-click on the command line that says:
`renv::restore()`
and then click "`run line or selection`". A prompt will appear in the console that asks, "`Would you like to activate this project before restore [Y/n]?`" Type `Y` in the console and hit enter. A second prompt will appear in the console that asks "`Do you want to proceed? [y/N]`". Type `y` in the console and hit enter. Wait until the console prints > on the bottom line, indicating that it is ready to execute additional commands.

Step 5-g: Comment out the previous three lines of code by hand. Directly in the script type "#" to the left of `library(renv)` so the new command line reads `#library(renv)`. Similarly, type "#" to the left of `Sys.setenv(RENV_DOWNLOAD_METHOD = "wininet")` so the new line reads `#Sys.setenv(RENV_DOWNLOAD_METHOD = "wininet")`. Finally, type "#" to the left of `renv::restore()` so the new command line reads `#renv::restore()`.

Step 6: You have two options to create the create the surveillance (testing) data summaries of confirmed CWD-positives for each applicable year for the state of interest.

Option 1: Use the directions in Mitchell et al. (2022) to generate the files for your state entitled:
`(state name)CountySummary2013.txt,`
`(state name)CountySummary2014.txt,`
`(state name)CountySummary2015.txt,`

(state name)CountySummary2016.txt,
(state name)CountySummary2017.txt,
(state name)CountySummary2018.txt,
(state name)CountySummary2019.txt,
(state name)CountySummary2020.txt,
(state name)CountySummary2021.txt, and
(state name)CountySummary2022.txt.
Save them "as is" to your working directory.

Option 2: Hand-enter positive count data into the templates titled
(state name)CountySummary2013.txt,
(state name)CountySummary2014.txt,
(state name)CountySummary2015.txt,
(state name)CountySummary2016.txt,
(state name)CountySummary2017.txt,
(state name)CountySummary2018.txt,
(state name)CountySummary2019.txt,
(state name)CountySummary2020.txt,
(state name)CountySummary2021.txt, and
(state name)CountySummary2022.txt.
Save them "as is" to your working directory. The data in each cell constitutes the total number of tested deer of each age/sex segment confirmed to be CWD positive in each county in the year of the file name. Adult is defined to be >2.5 years; yearling is defined to be 1-2.5 years; and fawn is defined to be <1 year (Chitwood et al. 2015).

Note: To use surveillance summaries from years other than those specified in this packet (2013-2022, 10 total) see Step **v-b** *below.*

Step 7: You have two options to create the testing data summaries for each applicable year for the state of interest.

Option 1: Use the directions in Mitchell et al. (2022) to generate the files for your state entitled:
All_(state name)CountySummary2013.txt,
All_(state name)CountySummary2014.txt,
All_(state name)CountySummary2015.txt,
All_(state name)CountySummary2016.txt,
All_(state name)CountySummary2017.txt,
All_(state name)CountySummary2018.txt,
All_(state name)CountySummary2019.txt,
All_(state name)CountySummary2020.txt,
All_(state name)CountySummary2021.txt, and
All_(state name)CountySummary2022.txt.
Save them "as is" to your working directory.

Option 2: Hand-enter testing count data into the templates entitled
`All_(state name)CountySummary2013.txt`,
`All_(state name)CountySummary2014.txt`,
`All_(state name)CountySummary2015.txt`,
`All_(state name)CountySummary2016.txt`,
`All_(state name)CountySummary2017.txt`,
`All_(state name)CountySummary2018.txt`,
`All_(state name)CountySummary2019.txt`,
`All_(state name)CountySummary2020.txt`,
`All_(state name)CountySummary2021.txt`, and
`All_(state name)CountySummary2022.txt`.
Save them "as is" to your working directory. The data in each cell constitutes the total number of tested deer of each age/sex segment in each county in the year of the file name. Adult is defined to be >2.5 years; yearling is defined to be 1-2.5 years; and fawn is defined to be <1 year (Chitwood et al. 2015).

Note: To use surveillance summaries from years other than those specified in this packet (2013-2022, 10 total) see Step **v-b** *below.*

Step 8: You have two options to create the data files containing the estimated mean values of benefits for Monitoring and Early Detection for the state of interest.

Option 1: Use Them et al. (2023*a*) and (2023*b*) to generate the files entitled `Benefits_Monitoring_(state name)_Mean.txt` and `Benefits_Early_Detection_(state name)_Mean.txt`, then save them "as is" into your working directory.

Option 2: Hand enter the benefit data into the templates entitled `Benefits_Monitoring_(state name)_Mean.txt` and `Benefits_Early_Detection_(state name)_Mean.txt`, then save them "as is" into your working directory. Refer to Them et al. (2023*a*), Them (2023*b*), and Hanley (*in preparation*) for the definition of benefits.

Step 9: Enter your cost data.

If surveillance sampling cost data are available by collection category and county:

Enter the surveillance sampling cost per deer by collection category for each county into the "`Cost_Matrix_(state name).csv`" template.

9-ai. Open the "`Cost_Matrix_(state name).csv`" template.

9-bi. Enter the average cost per deer for hunter harvested samples from each county (or other administrative area) into the Cost_Hunter column.

9-ci. Enter the average cost per deer for other (non-hunter harvested) samples from each county (or other administrative area) into the Cost_Other column.

9-di. Save the "`Cost_Matrix_(state name).csv`" to your working directory.

If surveillance sampling cost data are NOT available by collection category and county:

9-aii. Open the "`_0b_(state name)_Expense_Cap_Cost_Prep.R`" script.

9-bii. Uncomment the code on the lines beginning with `ENTER THE COORDINATES OF THE DIAGNOSTIC LAB` and ending with `END UNCOMMENTING HERE` by manually removing the first # at the beginning of each line or by selecting only those lines and hitting `ctrl + Shift + C` simultaneously.

9-cii. Update the coordinates of the nearest National Animal Health Laboratory Network (NAHLN; APHIS 2021) laboratory that conducts CWD testing for the state of interest on the line:
`labs <- data.frame("Lon" = -75.750810, "Lat" = 39.870860).`

9-dii. Update the Standard Mileage Rate (currently entered based on IRS 2021) on the line `costpermile <- .56` if desired.

9-eii. Update the per deer Diagnostic Cost on the line `costpersample <- 27` based on the selected laboratory.

9-fii. Save the updated script to your working directory.

Step 10: Enter the real surveillance sampling data into the "`HistoricData(state name).csv`" template. Note: You can define your historic sampling data in any way you please so long as it represents a sampling effort that has occurred in the past. For example, your historic data can represent the previous year, the average from five previous years, the maximum sampling data from five previous years, etc.

10-a. Open the "`HistoricData(state name).csv`" template for the state of interest or, if creating a file for a new state not included in this packet, update the County column with a comprehensive list of counties (or other administrative areas) in the new state of interest.

10-b. Retain 9999 in the column labeled Latitude; the computer will fill this column automatically.

10-c. Retain 9999 in the column labeled Longitude; the computer will fill this column automatically.

10-d. Sum the samples from hunter harvested deer for each county (or other administrative area) for the user specified time period and enter that value into the Total_Hunter column. Note: Enter Total_Hunter as a numeric, non-negative integer value (0 ≤ x).

10-e. Sum the samples from other deer (non-hunting) for each county (or other administrative area) for the user specified time period and enter that value into the Total_Other column. Note: Enter Total_Other as a numeric, non-negative integer value (0 ≤ x).

10-f. Sum the samples from hunter harvested and other deer for each county (or other administrative area) for the user specified time period and enter that value into the Total column. Note: Enter Total as a numeric, non-negative integer value (0 ≤ x).

10-g. Save the "`HistoricData(state name).csv`" to your working directory.

Step 11: For the state of interest, run the *Software.*

11-a. Open the file named `_4_(state name)_Expense_Cap_Command_Center.R`.

11-b: In the upper ribbon menu of `R`, click "`Edit`", then click "`Run all`".

Note: From this moment, your script will automatically call the appropriate libraries, then begin running the series of scripts of the Software. Several lines of code will be executed without any further interaction from you. Your script will automatically create the shapefiles, initiate projections, load the data, convert the data, then open the user interface (`UI`). The algorithm will not run until the user then interacts with the interface.

Step 12: Once the `UI` opens, follow the directions on the landing page to interact with the app.

Step 13: Once done interacting with the `UI`, close the tab depicting the UI, stop the `R` code, and close the `R` program.

Similarities of the *Software* to other types of *Optimization Software*

This *Software* assumes the user has two elements of data per county, where the first element represents the cohort of deer that were hunter-harvested, and the second element represents the cohort of deer removed from the population via non-hunting mechanisms (such as targeted culling, clinical suspects, road kills). This *Software* uses an additive ("bottom up") approach as opposed to a combinatorial optimization algorithm.

The n×1 optimization (see Hanley et al. 2023*a*) assumes the user has one element of data per county, where that element represents the cohort of deer that were hunter-harvested. As well, the n×1 optimization utilizes an algorithmic ("top down") approach adapted from a 1-0 knapsack problem (Korte & Vygen 2018).

The n×2 optimization (see Hanley et al. 2023*b*) assumes the user has two elements of data per county, where the first element represents the cohort of deer that were hunter-harvested, and the second element represents the cohort of deer removed from the population via non-hunting mechanisms (such as targeted culling, clinical suspects, road kills, etc.). The n×2 optimization utilizes the algorithmic ("top down") approach adapted from a 1-0 knapsack problem (Korte & Vygen 2018).

The n×12 optimization (see Hanley et al. 2023*c*) assumes the user has 12 elements of data per county, where the first six elements represent the age/sex segments (adult buck, adult doe, yearling buck, yearling doe, fawn buck, fawn doe) of hunter-harvested deer, and the second six elements represent the same age/sex segments for deer removed from the population via non-hunting mechanisms. The n×12 optimization also utilizes the algorithmic ("top down") approach adapted from a 1-0 knapsack problem (Korte & Vygen 2018).

Regardless of the dimensionality (and direction) of the optimization, one can obtain the vectorized data necessary to use this *Software* or the other optimizations using Them et al. (2023*a*), Them et al. (2023*b*), and Hanley (*in preparation*).

The scripts

The *Software* contains eight R scripts:
(1) `_0a_(state name)_Expense_Cap_Spatial_Prep.R` script;
(2) `_0b_(state name)_Expense_Cap_Cost_Prep.R` script;
(3) `_0c_(state name)_Expense_Cap_Prevalence_Prep.R` script;
(4) `_0d_(state name)_Expense_Cap_Matrix_Prep.R` script;
(5) `_1_(state name)_Expense_Cap_Data_Prep.R` script;
(6) `_2_(state name)_Expense_Cap_User_Interface.R` script;
(7) `_3_(state name)_Expense_Cap_Algorithm_Two_Categories.R` script; and
(8) `_4_(state name)_Expense_Cap_Command_Center.R` script.
The `(state name)` in all file names is a placeholder to depict the geographical entity of interest, regardless of whether the entity is a state, province, or otherwise.

Scripts `_0a_(state name)_Expense_Cap_Spatial_Prep.R`, `_0b_(state name)_Expense_Cap_Cost_Prep.R`, `_0c_(state name)_Expense_Cap_Prevalence_Prep.R`, `_0d_(state name)_Expense_Cap_Matrix_Prep.R`, and `_1_(state name)_Expense_Cap_Data_Prep.R` prepare the data, and must be run in sequence.

The `_0a_(state name)_Expense_Cap_Spatial_Prep.R` script creates the geospatial files containing administrative boundaries, as well as related files necessary for executing the user interface. The automatically generated outputs of the `_0a_(state` `name)_Expense_Cap_Spatial_Prep.R` script include the geospatial files and related files autosaved in the structural format needed for immediate upload into the remaining scripts.

The `_0b_(state name)_Expense_Cap_Cost_Prep.R` script provides the option to calculate estimated sampling cost per deer for each county based on distance and diagnostic fee information in the instance when there is no sampling cost information available for a state. The script calculates the distance (miles) from the centroid of each county to the nearest National Animal Health Laboratory Network (NAHLN; APHIS 2021) laboratory that conducts CWD testing, then multiplies the distance by the 2021 Standard Mileage Rate (IRS 2021). The script then adds to the total distance costs, the per deer diagnostic cost to determine CWD status (APHIS 2021), then saves the data in the format needed for immediate upload into the remaining scripts. Cost matters in the software; the difference in sampling cost between sample collection categories (hunter harvested and non-hunter harvested) constitutes the difference between those categories in the algorithm.

The `_0c_(state name)_Expense_Cap_Prevalence_Prep.R` script converts testing summaries from Mitchell et al. (2022) into the standardized prevalence for each collection category based on white-tailed deer vital rates and sex ratios (Chitwood et al. 2015; Verme & Ozoga 1981), then saves the data in the format needed for immediate upload into the remaining scripts. The testing summaries from Mitchell et al. (2022) do not distinguish between hunter harvested and non-hunter harvested samples; therefore, the resulting prevalence in the software will be equivalent for both collection categories.

The script converts the estimated values of benefits for monitoring and early detection (Them et al. 2023*a* and 2023*b*) as well as autosaved outputs from scripts `_0a_(state name)_Expense_Cap_Spatial_Prep.R`, `_0b_(state name)_Expense_Cap_Cost_Prep.R`, and `_0c_(state name)_Expense_Cap_Prevalence_Prep.R`, and surveillance weights (Jennelle et al. 2018) into the format needed for immediate upload into the remaining scripts. The output from the `_0d_(state name)_Expense_Cap_Matrix_Prep.R` script is called the *Optimization Matrix*.

The `_1_(state name)_Expense_Cap_Data_Prep.R` script converts the *Optimization Matrix* into the format necessary for display in the user interface, determines which counties are eligible for each objective (early detection, monitoring, and/or public communications), then saves the outputs in the format needed in the next script.

The `_2_(state name)_Expense_Cap_User_Interface.R` script converts the *Historical Data* into the format necessary for display in the user interface, intakes the autosaved geospatial and optimization files, and launches the user interface of the algorithm. From this interface the user will view (in the case of sample size; Booth et al. *in review*) or initialize their desired management settings.

The `_3_(state name)_Expense_Cap_Algorithm_Two_Categories.R` script contains the algorithm programmed with commands and inputs based on user selected inputs. The `_3_(state name)_Expense_Cap_Algorithm_Two_Categories.R` script is automatically called by the `_2_(state name)_Expense_Cap_User_Interface.R` script once the user enters their settings into the algorithm. Thus, the `_3_(state name)_Expense_Cap_Algorithm_Two_Categories.R` script should not be directly opened nor run by the user.

With one click, the `_4_(state name)_Expense_Cap_Command_Center.R` script of the *Software* runs the appropriate sequence of scripts.

Data inputs

NOTICE. The *Software* will not run until the "9999" data are replaced with real data. Read carefully below.

Data inputs necessary to run the `_0a_(state name)_Expense_Cap_Spatial_Prep.R` script include:

0a-i. Raw geospatial files containing administrative boundaries for the US. *Note:* The Cartographic Boundary and TIGER line files (US Census Bureau 2019; 2020) necessary to create the appropriate files for the US states are procured using command lines directly in the script.

Autosaved outputs from the `_0a_(state name)_Expense_Cap_Spatial_Prep.R` script include:

0a-ii. US shapefiles and zipped folders (for US states only): `tl_2020_us_county` and `cb_2019_us_county_500k`.

0a-iii. The geospatial files for the state of interest: `(state name)_counties.dbf, (state name)_counties.prj, (state name)_counties.shp`, and `(state name)_counties.shx`.

0a-iv. Text files containing the geographic center of the state of interest: `(state name)_Latitude.txt` and `(state name)_Longitude.txt`. Do not modify the names of these files, as they are used as is in later scripts.

If surveillance sampling cost data ARE available by collection category and county:

Data inputs necessary to run the `_0b_(state name)_Expense_Cap_Cost_Prep.R` script include:

0b-i. An n × 3 data frame named "Cost_Matrix_(state name).csv" containing the surveillance sampling cost per deer by collection category for each of the n counties (or other administrative areas) in the state of interest. The names of the counties are listed down the rows in alphabetical order. The three variables (column names) listed from left to right are County, Cost_Hunter, and Cost_Other. *Note: Use these exact header names, capitalization scheme, and order.* County is the current name of the county along with the associated legal/statistical area description and should match the NAMELSAD attribute of the geospatial file for the state of interest. This data is provided for the state included in this packet. Cost_Hunter represents the surveillance sampling cost per hunter harvested deer for that county. Cost_Other represents the surveillance sampling cost per deer that were removed from the population via non-hunting mechanisms, such as roadkill, targeted culling, or other actions overseen by the agency, for that county.

Autosaved outputs from the `_0b_(state name)_Expense_Cap_Cost_Prep.R` script include:

0b-ii. The data files (csv and txt) which contain the n × 2 *Cost Matrix* to sample a single individual from each county (or other administrative area) in the state of interest, where n is the number of counties and 2 is the number of collection types: `Cost_Matrix_(state name).csv` and `Cost_Matrix_(state name).txt`. Do not modify the names of these files, as they are used as is in later scripts.

If surveillance sampling cost data are NOT available by collection category and county:

Data inputs necessary to run the `_0b_(state name)_Expense_Cap_Cost_Prep.R` script include:

0b-i. Geospatial files autosaved during the execution of the `0a` script.

0b-ii. The latitude and longitude coordinates (WGS84) of the nearest laboratory in the National Animal Health Laboratory Network (NAHLN; APHIS 2021) that conducts CWD testing. Note: This data is pre-entered for the state in this packet but must be entered manually for any other state.

0b-iii. The CONUS cost per mile (IRS 2021). This 2021 cost is pre-entered.

0b-iv. The cost per deer to evaluate for CWD at the nearest laboratory in the National Animal Health Laboratory Network (NAHLN; APHIS 2021) that conducts CWD testing. The current cost as of December 2021 is pre-entered for the state in this packet but can be changed manually.

0b-v. The "`Cost_Matrix_(state name).csv`" template described above in **0b-i** with 9999 filled in for the Cost_Hunter and Cost_Other columns.

Autosaved outputs from the `_0b_(state name)_Expense_Cap_Cost_Prep.R` script include:

0b-vi. The data files (csv and txt) which contain the n × 2 *Cost Matrix* to sample a single individual from each county (or other administrative area) in the state of interest, where n is the number of counties and 2 is the number of collection types: `Cost_Matrix_(state name).csv` and `Cost_Matrix_(state` `name).txt`. Do not modify the names of these files, as they are used as is in later scripts.

Data inputs necessary to run the `_0c_(state name)_Expense_Cap_Prevalence_Prep.R` script include:

0c-i. Pre-processed surveillance (testing) data summaries of confirmed CWD-positives by county (or other administrative area) and age/sex segment generated for years 2013-2022, as produced in Mitchell et al. (2022): `(state name)CountySummary2013.txt`, `(state name)CountySummary2014.txt`, `(state name)CountySummary2015.txt`, `(state name)CountySummary2016.txt`, `(state name)CountySummary2017.txt`, `(state name)CountySummary2018.txt`, `(state name)CountySummary2019.txt`, `(state name)CountySummary2020.txt`, `(state name)CountySummary2021.txt`, and `(state name)CountySummary2022.txt`.

0c-ii. Pre-processed surveillance (testing) data summaries of all samples by county (or other administrative area) and age/sex segment generated for years 2013-2022, as produced in Mitchell et al. (2022): `All_(state name)CountySummary2013.txt`, `All_(state name)CountySummary2014.txt`, `All_(state name)CountySummary2015.txt`, `All_(state name)CountySummary2016.txt`, `All_(state name)CountySummary2017.txt`, `All_(state name)CountySummary2018.txt`, `All_(state name)CountySummary2019.txt`, `All_(state name)CountySummary2020.txt`, `All_(state name)CountySummary2021.txt`, and `All_(state name)CountySummary2022.txt`. Note: Only example files have been provided: the real data do not appear in this packet for any state. Run Mitchell et al. (2022) to create real data outputs in the format of these templates.

Autosaved outputs from the `_0c_(state name)_Expense_Cap_Prevalence_Prep.R` script include:

0c-iii. The data files (csv and txt) which contain the n × 2 apparent prevalence for each county (or other administrative area) in the state of interest, where n is the number of counties and 2 is the number of collection types: `Prevalence_Matrix_(state name).csv` and `Prevalence_Matrix_(state name).txt`. Do not modify the names of these files, as they are used as is in later scripts.

Data inputs necessary to run the `_0d_(state name)_Expense_Cap_Matrix_Prep.R` script include:

0d-i. The exact autosaved outputs from the previous scripts: `Cost_Matrix_(state name).txt` and `Prevalence_Matrix_(state name).txt`.

0d-ii. The pre-processed n × 1 estimated mean values of benefits for Monitoring and Early Detection, as produced in Them et al. (2023*a* and 2023*b*), where n is the number of `counties: Benefits_Monitoring_(state name)_Mean.txt` and `Benefits_Early_Detection_(state name)_Mean.txt`. Note: Only example files have been provided; the real data do not appear in this packet for any state. See Them et al. (2023*a* and 2023*b*) to create data outputs or templates.

0d-iii. The pre-processed file named `SampleSize_SRS_Table.txt`, depicting the target sample size under simple random sampling to declare the prevalence of CWD less than or equal to 0.01 (Booth et al. *in preparation*).

Autosaved outputs from the `_0d_(state name)_Expense_Cap_Matrix_Prep.R` script include:

0d-iv. The data files (csv and txt) which contain the n × 10 optimization matrix with the costs, benefits (Monitoring, Early Detection, and Public Communication), and prevalence for both sampling collection types in each of the n counties: `OptimizationMatrix(state name).csv` and `OptimizationMatrix(state name).txt`. The rows represent the n counties (or other administrative areas) of the state of interest. The column headers are cost_hunter, cost_other, benefit_delta_p_hunter, benefit_delta_p_other, benefit_detection_hunter, benefit_detection_other, benefit_communication_hunter, benefit_communication_other, prevalence_hunter, and prevalence_other.

0d-v. The parameter file (txt): `TargetPoints.txt`. Do not modify the names of these files, as they are used as is in later scripts.

Data inputs necessary to run the `_1_(state name)_Expense_Cap_Data_Prep.R` include:

1-i. Geospatial files autosaved during the execution of the `0a` script.

1-ii. An n × 6 data frame named "`HistoricData(state name).csv`" containing historical surveillance sampling data for each of the n counties (or other administrative areas) in the state of interest. The names of the counties are listed down the rows in alphabetical order. The six variables (column names) listed from left to right are County, Latitude, Longitude, Total_Hunter, Total_Other, and Total. Note: Use these exact header names, capitalization scheme, and order. County is the current name of the county along with the associated legal/statistical area description and should match the NAMELSAD attribute of the geospatial file for the state of interest. This data is provided for the state included in this packet. Latitude and Longitude represent the latitude and longitude coordinates of the center of each county (or other administrative area) referenced to the WGS84 datum and if blank, will be filled in by this script. Total_Hunter represents the historic number of samples harvested by hunters from deer in each county for the user specified time period. Total_Other represents the historic number of samples from deer that were removed from the population via non-hunting mechanisms, such as roadkill, targeted culling, or other actions overseen by the agency, in each county for the user specified time period. Total represents the total number of samples from hunter harvested and other (non-hunting mechanisms) deer for each county for the user specified time period. Note: The real data do not appear in the Historical Data for any state in this packet. However, the templates are included. A user must modify the `HistoricData(state name).csv` template to replace the 9999 with real data before this script will run (see Step 8 below).

1-iii. The Optimization Matrix data file autosaved during the execution of the 0d script: `OptimizationMatrix(state name).txt`.

Autosaved outputs from the `_1_(state name)_Expense_Cap_Data_Prep.R` include:

1-iv. Parameter files: `column_names.txt, Cost.txt, EarlyBenefitVector.txt, MonitoringBenefitVector.txt, Names.txt, names_row.txt, PublicCommBenefitVector.txt`, and `StartingPrevalence.txt`. Do not modify the names of these files, as they are used as is in later scripts.

Data inputs necessary to run the `_2_(state name)_Expense_Cap_User_Interface.R` script include:

2-i. Geospatial files autosaved during the execution of the `0a` script.

2-ii. Autosaved files generated during the execution of scripts `0b-0d`.

2-iii. The pre-saved coordinates to inform the progress arrow in the user interface, which displays the user's progression through the steps necessary for use of the algorithm: `progresscoordinates.csv`.

2-iv. Historical data, titled `HistoricData(state name).csv`, described in **1-ii**.

2-v. The `_1_(state name)_Expense_Cap_Data_Prep.R` script.

2-vi. The _3_(state name)_Expense_Cap_Algorithm_Two_Categories.R script.

Autosaved outputs from opening the _2_(state name)_Expense_Cap_User_Interface.R script include:

2-vii. Parameter files: Alpha.txt, Confidence.txt, DESIRED_CHANGE.txt, f.txt, and freedom_from_disease.txt. Do not modify the names of these files, as they are used as is in the interface.

Autosaved outputs from initializing the algorithm internal to the _2_(state name)_Expense_Cap_User_Interface.R script include:

2-viii. Additional parameter files: freedomfromdisease.txt, slider.txt, sliderEarly.txt, sliderMonitoring.txt, StartingVectorEarlyDetectionUP.txt, StartingVectorMonitoringUP.txt, and StartingVectorPublicCommunicationsUP.txt.

2-vix. Output files: OptimalAdjustedBenefitUP.txt, OptimalAdjustedCostUP.txt, OptimalAdjustedRecommendationsMachineReadableUP.txt, OptimalAdjustedRecommendationsRecipeUP.txt, OptimalStrategyEarlyDetectionBenefitUP.txt, OptimalStrategyEarlyDetectionCostUP.txt, OptimalStrategyEarlyDetectionRecipeUP.txt, OptimalStrategyMonitoringBenefitUP.txt, OptimalStrategyMonitoringCostUP.txt, OptimalStrategyMonitoringRecipeUP.txt, OptimalStrategyPublicCommunicationsBenefitUP.txt, OptimalStrategyPublicCommunicationsCostUP.txt, and OptimalStrategyPublicCommunicationsRecipeUP.txt. Do not modify the names of these files, as they are used as is in the interface.

The _3_(state name)_Expense_Cap_Algorithm_Two_Categories.R script is called directly from the user interface and should not be directly opened nor run by the user. Data inputs necessary to run the _3_(state name)_Expense_Cap_Algorithm_Two_Categories.R script include the parameter file outputs created in scripts 1 and 2.

The _4_(state name)_Expense_Cap_Command_Center.R script runs the appropriate sequence and set of scripts for each state with one click. The data inputs necessary to run the _4_(state name)_Expense_Cap_Command_Center.R script include all inputs and autosaved outputs from all previous scripts detailed above.

Instructions to adapt the *Software* to a different state

Step i: Complete **Steps 1-6** above before proceeding.

Step ii: Create a copy of each script in the *Software* for the new state.

ii-a. Make a copy of each script - `0a`, `0b`, `0c`, `0d`, `1`, `2`, `3`, and `4`.

ii-b. Rename the copy to reflect the name of the new state (or province). Note: Name states with spaces WITHOUT the space, nor the country, while maintaining capitalization of each part of the state name.
For example, to add New York, the file name should be "`_1_NewYork_Expense_Cap_Data_Prep.R`".

ii-c. Save the new scripts to the working directory.

Step iii: Update the `_0a_(state name)_Expense_Cap_Spatial_Prep.R` script.

iii-a. Open the `_0a_(state name)_Expense_Cap_Spatial_Prep.R` script created in step **ii-b** and replace the state name on the line `FullStateName <- "Pennsylvania"` with the new state name in quotations. Note: Name states with spaces WITH the spaces in this instance. For example, to add New York, the code should read: `FullStateName <- "New York".`

iii-b. Save and close the updated script.

Step iv: Update the `_0b_(state name)_Expense_Cap_Cost_Prep.R` script.

iv-a. Open `_0b_(state name)_Expense_Cap_Cost_Prep.R` script created in step **ii-b** and click "Edit" then "Replace…", and in the box that pops up, fill in the appropriate fields with the existing state name and the new state name, then click "`Replace All`". Note: Name states with spaces WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, enter `NewYork` in the box.

iv-b. Save and close the updated script.

Step v: Update the `_0c_(state name)_Expense_Cap_Prevalence_Prep.R` script.

v-a. Open `_0c_(state name)_Expense_Cap_Prevalence_Prep.R` script created in step **ii-b** and click "`Edit`" then "`Replace…`", and in the box that pops up, fill in the appropriate fields with the existing state name and the new state name, then click "`Replace All`". Note: Name states with spaces WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, enter `NewYork` in the box.

v-b. To use surveillance summaries from years other than those specified in this packet (2013-2022, 10 total), edit the year values on the line:
`Years <- seq(from = 2013, to = 2022, by = 1)` accordingly.

v-c. Save and close the updated script.

Step vi: Update `_0d_(state name)_Expense_Cap_Matrix_Prep.R` script.

vi-a. Open `_0d_(state name)_Expense_Cap_Matrix_Prep.R` script created in step **ii-b** and click "`Edit`" then "`Replace…`", and in the box that pops up, fill in the appropriate fields with the existing state name and the new state name, then click "`Replace All`". Note: Name states with spaces WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, enter `NewYork` in the box.

vi-b. Save and close the updated script.

Step vii: Update `_1_(state name)_Expense_Cap_Data_Prep.R` script.

vii-a. Open `_1_(state name)_Expense_Cap_Data_Prep.R` script created in step **ii-b** and click "`Edit`" then "`Replace…`", and in the box that pops up, fill in the appropriate fields with the existing state name and the new state name, then click "`Replace All`". Note: Name states with spaces WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, enter `NewYork` in the box.

vii-b. Save and close the updated script.

Step viii: Update `_2_(state name)_Expense_Cap_User_Interface.R` script.

viii-a. Open `_2_(state name)_Expense_Cap_User_Interface.R` script created in step **ii-b** and click "`Edit`" then "`Replace…`", and in the box that pops up, fill in the appropriate fields with the existing state name and the new state name, then click "`Replace All`". Note: Name states with spaces WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, enter `NewYork` in the box.

viii-b. Save and close the updated script.

Step ix: Update the `_3_(state name)_Expense_Cap_Algorithm_Two_Categories.R` script.

ix-a. Open the `_3_(state name)_Expense_Cap_Algorithm_Two_Categories.R script` created in step **ii-b** and click “`Edit`” then “`Replace…`”, and in the box that pops up, fill in the appropriate fields with the existing state name and the new state name, then click “`Replace All`”. Note: Name states with spaces WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, enter `NewYork` in the box.

ix-b. Save and close the updated script.

Step x: Update the `_4_(state name)_Expense_Cap_Command_Center.R` script.

x-a. Open `_4_(state name)_Expense_Cap_Command_Center.R script` created in step **ii-b** and click “`Edit`” then “`Replace…`”, and in the box that pops up, fill in the appropriate fields with the existing state name and the new state name, then click “`Replace All`”. Note: Name states with spaces WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, enter `NewYork` in the box.

x-b. Save and close the updated script.

Step xi: Create a new “`Cost_Matrix_(state name).csv`” file, then add it to the *Software*.

If surveillance sampling cost data are available by collection category and county:

xi-ai. Make a copy of the “`Cost_Matrix_(state name).csv`” of any previously existing state.

xi-bi. Rename the copy to reflect the name of the new state. Note: Name states WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, the file name should be “`Cost_Matrix_NewYork.csv`.”

xi-ci. Follow instructions in **Steps 8-ai** though **8-di** above for the new state.

If surveillance sampling cost data are NOT available by collection category and county:

xi-aii. Follow instructions in **Steps 8-aii** though **8-fii** above for the new state.

Step xii: Create a new “`HistoricData(state name).csv`” file, then add it to the *Software*.

xii-a. Make a copy of the “`HistoricData(state name).csv`” of any previously existing state.

xii-b. Rename the copy to reflect the name of the new state. Note: Name states with spaces WITHOUT the spaces, nor the country, while maintaining capitalization of each part of the state name. For example, to add New York, the file name should be "`HistoricDataNewYork.csv`."

xii-c. Follow instructions in **Step 9** above for the new state.

Step xiii: Run the command code for the new state (i.e., **Steps 10-12** above).

Technical details

The packet includes the environment in which the code was written, as packaged through the `renv` package version 0.16.0 (Ushey 2022). The `renv` folder, all its contents as well as the `renv.lock` file contain a list and versions of all the dependencies of this code. Do not modify these folders.

Acknowledgements

Financial support comes from (1) Michigan Disease Initiative – Optimizing CWD Surveillance: Regional Synthesis of Demographic, Spatial, and Transmission-Risk Factors (2019); (2) Tennessee Wildlife Resources Agency - Modeling Risk of Infection for Individually Harvested Deer & Estimating Prevalence When Sampling is Limited (2020); (3) Michigan Disease Initiative - SOP4CWD Dashboard: A Web Application for Disease Visualization and Data-Driven Decisions (2020); (4) Multistate Conservation Grant Program - Surveillance Optimization Project for Chronic Wasting Disease: Streamlining a Web Application for Disease Visualization and Data-Driven Decisions (2021). We thank J. Peaslee, J. Bloodgood, N. Hollingshead, and A. Kaganer for their helpful comments with the software.

License

Suggested citation

Them, C. E., Hanley, B. J., Mitchell, C. I., Gagne, R. B., Ahmed, M. S., & Schuler, K L. 2023. Chronic wasting disease surveillance expense cap software [Software]. Cornell University Library eCommons Repository. DOI: *https://doi.org/10.7298/r4jy-bp10.*

Citations

Animal and Plant Health Inspection Service [APHIS]. 2021. National Animal Health Laboratory Network. <*www.aphis.usda.gov/animal_health/nahln/downloads/cwd_lab_list.pdf*>. Accessed Dec 2021.

Appelhans, T., & Detsch, F. 2020. leafpop: include tables, images, and graphs in leaflet pop-ups. R package version 0.0.6. DOI: *https://CRAN.R-project.org/package=leafpop.*

Bivand, R., Pebesma, E., & Gomez-Rubio, V. 2013. Applied spatial data analysis with R, 2nd edition. Springer, New York, New York, USA. DOI: *https://asdar-book.org/.*

Bivand, R., Keitt, T., & Rowlingson, B. 2020. rgdal: bindings for the 'geospatial' data abstraction library. R package version 1.5-18. DOI: *https://CRAN.R-project.org/package=rgdal.*

Bivand, R., & Rundel, C. 2020. rgeos: interface to geometry engine - open source ('GEOS'). R package version 0.5-5. DOI: *https://CRAN.R-project.org/package=rgeos.*

Bivand, R., & Lewin-Koh, N. 2022. maptools: tools for handling spatial objects. R package version 1.1-4. DOI: *https://CRAN.R-project.org/package=maptools.*

Booth, J. G., Hanley, B. J., Hodel, F. H., Jennelle, C. S., Guinness, J., Them, C. E., Mitchell, C. I., Ahmed, M. S., & Schuler, K. L. Sample size for estimating disease prevalence in free-ranging wildlife populations: a Bayesian Approach. *In peer review.*

Chang, W., & Borges Ribeiro, B. 2018. shinydashboard: create dashboards with 'Shiny'. R package version 0.7.1. DOI: *https://CRAN.R-project.org/package=shinydashboard.*

Chang, W., Cheng, J., Allaire, J. J., Xie, Y., & McPherson, J. 2020. shiny: web application framework for R. R package version 1.5.0. DOI: *https://CRAN.R-project.org/package=shiny.*

Cheng, B., Karambelkar, B., & Xie, Y. 2019. leaflet: create interactive web maps with the JavaScript 'leaflet' library. R package version 2.0.3. DOI: *https://CRAN.R-project.org/package=leaflet.*

Chitwood, M. C., Lashley, M. A., Kilgo, J. C., Moorman, C. E., & Deperno, C. S. 2015. White-tailed deer population dynamics and adult female survival in the presence of a novel predator. The Journal of Wildlife Management, 79(2), 211–219. DOI: *http://www.jstor.org/stable/43188324.*

Cornell Wildlife Health Laboratory [CWHL]. 2022. Surveillance optimization project for chronic wasting disease [SOP4CWD]. <*https://cwhl.vet.cornell.edu/project/sop4cwd*>. Accessed 2 September 2022.

Hanley, B. J, Mitchell, C. I., Them, C. E., Walter, W. D., Walsh, D. P., Jennelle, C. S., Hollingshead, N. A., Abbott, R. C., Kelly, J. D., Grove, D. M., Williams, D., Christensen, S. A., Ahmed, M. S., Booth, J.G., Guinness, J., Miller, L. A., & Schuler, K. L. 2023*a.* Chronic wasting disease surveillance optimization software (n×1) [Software]. Cornell University Library eCommons Repository. DOI: *https://doi.org/10.7298/c8xm-wx39.*

Hanley, B. J, Mitchell, C. I., Them, C. E., Walter, W. D., Walsh, D. P., Jennelle, C. S., Hollingshead, N. A., Abbott, R. C., Kelly, J. D., Grove, D. M., Williams, D., Christensen, S. A., Ahmed, M. S., Booth, J. G., Guinness, J., Gagne, R. B., DiSalvo, A. R., Fleegle, J. T., Rosenberry, C. S., Miller, L. A., & Schuler, K. L. 2023*b*. Chronic wasting disease surveillance optimization software (n×2) [Software]. Cornell University Library eCommons Repository. DOI: *https://doi.org/10.7298/84md-wg95*.

Hanley, B. J., Mitchell, C. I., Walter, W. D., Walsh, D. P., Jennelle, C. S., Hollingshead, N. A., Abbott, R. C., Kelly, J. D., Grove, D. M., Williams, D., Them, C. E., Ahmed, M. S., Miller, L., & Schuler, K. L. 2023*c*. Chronic wasting disease surveillance optimization software (n×12) (version 2) [Software]. Cornell University Library eCommons Repository. DOI: *https://doi.org/10.7298/p21r-q845.2*.

Hanley, B. J*.*, Them, C. E., Mitchell, C. I., Abbott, R. C., Hollingshead, N. A., Ahmed, M. S., Walsh, D. P., Booth, J. G., Guinness, J., Walter, W. D., Jennelle, C. S., Hodel, F. H., Christensen, S. A*.*, Ballard, J. R., Riggs, A. J., Middaugh, C. R., Cunningham, M., Clemons, B., Sayler, K., Killmaster, C. H., Harms, T. M., Ruden, R. M., Caudell, J., Westrich, M. B., McCallen, E., Casey, C., O'Brien, L. M., Trudeau, J., Straka, K., Stewart, C., Carstensen, M., Hynes, K. P., Ableman, A., Miller, L. A., Cook, M., Meyers, R., Shaw, J., Tonkovich, M. J., Nituch, L., DiSalvo, A. R., Fleegle, J. T., Rosenberry, C. S., Kelly, J. D., Grove, D. M., Kirchgessner, M. S., Storm, D. J., & Schuler, K. L. Risk factors for chronic wasting disease in white-tailed deer using surveillance data from eastern North America. *In preparation.*

Hijmans, R. J. 2020. raster: geographic data analysis and modeling. R package version 3.4-5. DOI: *https://CRAN.R-project.org/package=raster.*

Internal Revenue Service [IRS]. 2021. CONUS mileage rates. <*https://www.irs.gov/tax-professionals/standard-mileage-rates*>. Accessed Dec 2021.

Jennelle, C. S., Walsh, D. P., Samuel, M., Osnas, E., Rolley, R., Langenberg, J., Powers, J., Monello, R., Demarest, E., Gubler, R., & Heisey, D. 2018. Applying a Bayesian weighted surveillance approach to detect chronic wasting disease in white-tailed deer. Journal of Applied Ecology 55(6):1-10. DOI: *https://doi.org/10.1111/1365-2664.13178.*

Karambelkar, B., & Schloerke, B. 2018. leaflet.extras: extra functionality for 'leaflet' package. R package version 1.0.0. DOI: *https://CRAN.R-project.org/package=leaflet.extras*.

Korte, B., & Vygen, J. 2018. Combinatorial optimization: theory and algorithms. 6th edition. Springer Publishing, New York, New York, USA.

Mitchell, C. I., Hanley, B. J., Abbott, R. C., Hollingshead, N. A., Kelly, J. D., Miller, L. A., & Schuler, K. 2022. Wildlife disease positives software [Software]. Cornell University Library eCommons Repository. DOI: *https://doi.org/10.7298/sf7q-8q61.*

Neuwirth, E. 2014. RColorBrewer: colorbrewer palettes. R package version 1.1-2. DOI: *https://CRAN.R-project.org/package=RColorBrewer.*

Padgham, M., & Sumner, M.D. 2021. geodist: fast, dependency-free geodesic distance calculations. R package version 0.0.7. DOI: *https://CRAN.R-project.org/package=geodist.*

Pebesma, E., & Bivand, R. 2005. Classes and methods for spatial data in R. R News 5 (2). DOI: *https://cran.r-project.org/doc/Rnews/.*

Pebesma, E. 2018. Simple features for R: standardized support for spatial vector data. The R Journal, 10 (1), 439-446. DOI: *https://doi.org/10.32614/RJ-2018-009.*

R Core Team. 2020. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. DOI: *www.R-project.org/.*

Sali, A., & Attali, D. 2020. shinycssloaders: add loading animations to a 'shiny' output while it is recalculating. R package version 1.0.0. DOI: *https://CRAN.R-project.org/package=shinycssloaders.*

Teucher, A., & Russell, K. 2020. rmapshaper: client for "mapshaper" for "geospatial" operations. R package version 0.4.4. DOI: *https://CRAN.R-project.org/package=rmapshaper.*

Them, C. E., Mitchell, C. I., Hollingshead, N. A., Abbott, R. C., Hanley, B. J, Ahmed, M. S., Booth, J. G., Jennelle, C. S., Hodel, F. H., Guinness, J., Ballard, J. R., Riggs, A. J., Middaugh, C. R., Cunningham, M., Clemons, B., Sayler, K., Killmaster, C. H., Harms, T. M., Ruden, R. M., Caudell, J., Westrich, M. B., McCallen, E., Casey, C., O'Brien, L. M., Trudeau, J., Straka, K., Stewart, C., Carstensen, M., Hynes, K. P., Ableman, A., Miller, L. A., Cook, M., Meyers, R., Shaw, J., Tonkovich, M. J., Nituch, L., DiSalvo, A. R., Fleegle, J. T., Rosenberry, C. S., Kelly, J. D., Grove, D. M., Kirchgessner, M. S., Storm, D. J., & Schuler, K. L. 2023*a*. Surveillance benefit components for chronic wasting disease in white-tailed deer [Dataset]. Cornell University Library eCommons Repository. DOI: *https://doi.org/10.7298/7txw-2681*.

Them, C. E., Hanley, B. J., Mitchell, C. I., Gagne, R. B., Ahmed, M. S., & Schuler, K. L. 2023*b*. Chronic wasting disease benefit of surveillance matrix software [Software]. Cornell University Library eCommons Repository. DOI: *https://doi.org/10.7298/xc5z-hs96*.

U.S. Census Bureau. 2019. 2019 TIGER/Line Shapefiles (machine-readable data files). DOI: *https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html.*

U.S. Census Bureau. 2020. 2020 TIGER/Line Shapefiles (machine-readable data files). DOI: *https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html*.

Ushey, K. 2022. renv: project environments. R package version 0.16.0. DOI: *https://rstudio.github.io/renv/.*

Verme, L. J., & Ozoga, J. J. 1981. Sex ratio of white-tailed deer and the estrus cycle. The Journal of Wildlife Management, 45(3), 710–715. DOI: *https://doi.org/10.2307/3808704.*

Walker, K. 2020. tigris: load census TIGER/line Shapefiles. R package version 1.0. DOI: *https://CRAN.R-project.org/package=tigris.*

Wickham, H. 2019. stringr: simple, consistent wrappers for common string operations. R package version 1.4.0. DOI: *https://CRAN.R-project.org/package=stringr.*

Wickham, H., Hester, J., & Chang, W. 2020. devtools: tools to make developing R packages easier. R package version 2.3.2. DOI: *https://CRAN.R-project.org/package=devtools.*

Wickham, H., François, R., Henry, L., & Müller, K. 2021. dplyr: a grammar of data manipulation. R package version 1.0.2. DOI: *https://CRAN.R-project.org/package=dplyr.*