dev\_records\_as\_sf

N.M. Tarr

knitr::opts\_chunk$set(  
 echo = TRUE,  
 warning = FALSE,  
 message = FALSE)

# Purpose

This document details a function that creates a spatial representation of checklists or observations. SHOULD A MAX EFFORT DISTANCE PARAMETER BE ADDED?

records\_as\_sf\_dev <- function(records\_df, kind, method, fill\_na\_km = 0.1) {  
 # Create new simple features (spatial data frame) of records (checklists or   
 # observations) Output can be plotted or used as input for other functions.  
 #   
 # Description:   
 # Checklist records often need to be assigned geometries for visualization  
 # and spatial analyses, and different methods could be used. Checklists  
 # can be represented as points or polygons, and polygons could be drawn as   
 # buffers around the checklist coordinates (circles) or buffers drawn around  
 # checklist tracks. Buffer length is meant to represent locational  
 # uncertainty (spatial precision) and can be approximated in different ways.  
 # Stationary or short lists should likely be buffered 100 m or more to   
 # at least partially account for area surveyed. Null effort\_distance\_km   
 # values are filled with zero, which assumes those records are stationary   
 # counts.  
 #  
 # Parameters:  
 # records\_df -- data frame of records with latitude, longitude,   
 # checklists\_id or sampling\_event\_identifier, atlas\_block, protocol\_type,  
 # and effort\_distance\_km columns.  
 # kind -- "checklists" or "observations" to identify what type of records are  
 # in the data frame. Individual species data will be observations.  
 # method -- how to represent each record spatially. Options are "points",  
 # "point-radius", and "buffered-tracks".  
 # fill\_na\_km -- NA values may exist in some records, which precludes creating  
 # a point-radius polygon for them. Enter a km distance to use as a   
 # replacement. This argument is inconsequential for the point method.  
 #   
 # Results:  
 # A spatial (simple features) data frame with columns for checklist\_id or   
 # sampling\_event\_identifier, atlas\_block, protocol\_type,   
 # effort\_distance\_km, latitude, longitude and observation\_count and  
 # breeding\_code if the kind is observations.  
 library(sf)  
   
 if (kind == "checklists"){  
 records\_df <- records\_df %>%  
 select(checklist\_id, atlas\_block, protocol\_type, effort\_distance\_km,  
 latitude, longitude)  
 } else {  
 records\_df <- records\_df %>%  
 select(sampling\_event\_identifier, atlas\_block, protocol\_type,   
 effort\_distance\_km, latitude, longitude, observation\_count,   
 breeding\_code)  
 }  
   
 # Make spatial frame  
 checklists\_sf <- records\_df %>%  
 st\_as\_sf(coords=c("longitude", "latitude"), crs=4326) %>%  
 st\_transform(6542)  
   
 # Apply method  
 if (method == "points") {  
 checklists\_sf <- checklists\_sf  
 }  
   
 if (method == "point-radius") {  
 checklists\_sf <- checklists\_sf %>%  
 # Buffer coordinates  
 replace\_na(list(effort\_distance\_km=fill\_na\_km)) %>%  
 mutate(buffer\_length = (effort\_distance\_km)\*1000) %>%  
 mutate(footprint = st\_buffer(geometry, buffer\_length)) %>%  
 select(-c(geometry)) %>%  
 mutate(geometry = footprint) %>%  
 st\_set\_geometry("geometry")  
 }  
   
 if (method == "buffer-tracks") {  
 print("This method is currently unavailable until we get checklist tracks.")  
 }  
   
 return(checklists\_sf)  
}

# Usage

This demo requires the tidyverse packages.

library(tidyverse)

Load the NCBA functions because this function relies upon the output from some of them.

setwd("~/Code/NCBA/resources")  
source("ncba\_functions.R")  
config <- "~/Documents/NCBA/Scripts/ncba\_config.R"

Set the working directory to somewhere outside of the NCBA repository so that results are not saved in the repository.

setwd("~/Documents/NCBA/species/")

Use the NCBA connection function to connect to the Atlas Cache (the mongodb).

# connect to a specific collection (table)  
connection <- connect\_ncba\_db(ncba\_config = config, database = "ebd\_mgmt",   
 collection = "ebd")

ncba\_functions has a function that returns a spatial data frame of NC counties in the 6542 CRS. Get that for use later in maps.

counties <- counties\_NC()

## Observations (species-level records)

Identify a species to investigate.

species <- "Willow Flycatcher"

Retrieve the records for the species from the Atlas Cache

time1 <- proc.time()  
  
# execute a query  
query <- str\_interp('{"OBSERVATIONS.COMMON\_NAME":"${species}"}')  
  
nc\_data <- connection$find(query) %>%  
 unnest(cols = (c(OBSERVATIONS))) %>% # Expand observations  
 filter(COMMON\_NAME == species)  
  
# format columns to the standard analysis format (ebd format)  
records <- to\_ebd\_format(nc\_data, drop=FALSE)  
  
# Calculate processing time  
mongotime <- proc.time() - time1  
  
# Print number of records returned  
print(paste("Records returned:", nrow(records)))

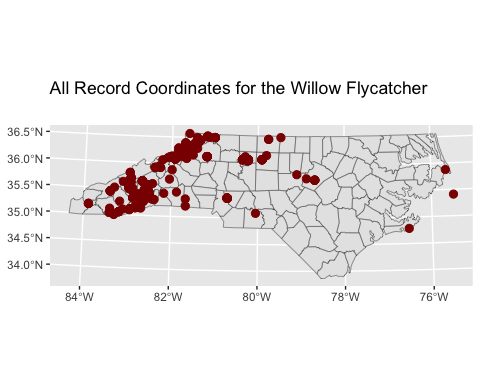
## [1] "Records returned: 1661"

print(paste("Runtime: ", mongotime[["elapsed"]]))

## [1] "Runtime: 2.015"

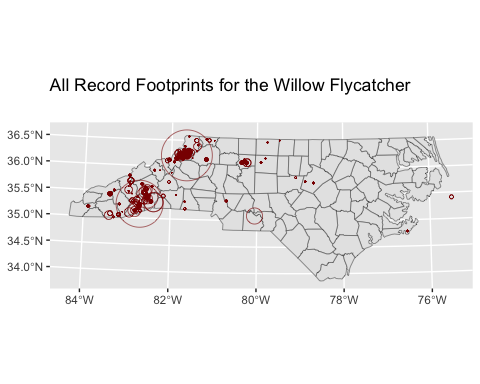
Plot the coordinates

coords <- records\_as\_sf\_dev(records, kind="observations", method="points")  
  
# Make a simple map  
ggplot(data = counties) +  
 geom\_sf() +  
 geom\_sf(data = coords, size = 4, shape = 20,   
 crs = st\_crs(6542), colour = "darkred") +   
 ggtitle(paste("All Record Coordinates for the", species), subtitle = "")



Plot the footprints.

footprints <- records\_as\_sf\_dev(records, kind="observations",   
 method="point-radius")  
  
# Make a simple map  
ggplot(data = counties) +  
 geom\_sf() +  
 geom\_sf(data = footprints, size = 4, shape = 20,   
 crs = st\_crs(6542), colour = "darkred", fill = NA) +   
 ggtitle(paste("All Record Footprints for the", species), subtitle = "")



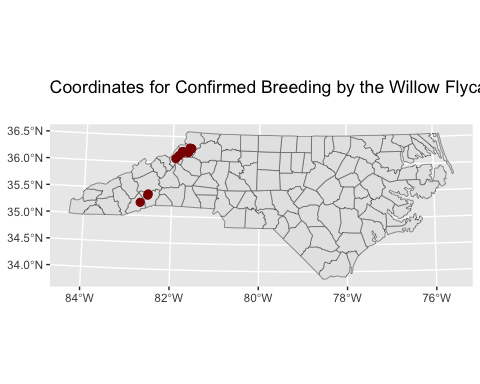
Preview the result as a data frame

print(head(footprints))

## Simple feature collection with 6 features and 7 fields  
## Active geometry column: geometry  
## Geometry type: POLYGON  
## Dimension: XY  
## Bounding box: xmin: 359390.7 ymin: 272443.5 xmax: 374352.8 ymax: 278881.5  
## Projected CRS: NAD83(2011) / North Carolina  
## # A tibble: 6 × 9  
## sampling\_event\_identifier atlas\_block protocol\_type effort\_distance\_km  
## <chr> <chr> <chr> <dbl>  
## 1 S86189566 "36081B7NE" Traveling 0.636  
## 2 S86671035 "" Traveling 0.987  
## 3 S86650256 "36081B6CE" Traveling 0.37   
## 4 S86493005 "" Traveling 3.22   
## 5 S86167367 "36081B7NE" Traveling 0.577  
## 6 S86187424 "" Traveling 0.636  
## # ℹ 5 more variables: observation\_count <int>, breeding\_code <chr>,  
## # buffer\_length <dbl>, footprint <POLYGON [m]>, geometry <POLYGON [m]>

Plot just the records that confirm breeding

# Get our list of confirmation codes from ncba\_functions  
conf\_codes <- breeding\_codes()["confirmed"][[1]]  
  
# Select only those records that confirm  
confirmed <- records %>% filter(breeding\_code %in% conf\_codes)  
  
# Make it spatial  
confirmed\_sf <- records\_as\_sf\_dev(confirmed, kind="observations",   
 method="points")  
  
# Make a simple map  
ggplot(data = counties) +  
 geom\_sf() +  
 geom\_sf(data = confirmed\_sf, size = 4, shape = 20,   
 crs = st\_crs(6542), colour = "darkred") +   
 ggtitle(paste("Coordinates for Confirmed Breeding by the", species), subtitle = "")



## Checklists

Retrieve a dataframe of checklists. Note that this use depends on connection code from above.

all\_checklists <- get\_all\_checklists(ncba\_config=config, drop\_ncba\_col=TRUE)

Plot the coordinates for all of the checklists from the Atlas Cache.

# Make the data frame spatial  
footprints <- records\_as\_sf\_dev(all\_checklists, kind="checklists",   
 method="points")  
  
# Make a simple map  
ggplot(data = counties) +  
 geom\_sf() +  
 geom\_sf(data = footprints, size = 4, shape = 3, crs = st\_crs(6542),  
 colour = "darkred") +   
 ggtitle("All NC Atlas Cache Checklist Coordinates", subtitle = "")

