Fric et al. Data formatting

Vaughn Shirey & Elise Larsen

Current version 7-Dec-2020; initiated Feb-2020

*

Data Import and Formatting

This code reads the occurrence data file provided by Fric et al. (2020) and formats it to be ready for analysis. It uses the data.csv and ele13419-suo-0003-tables2.xlsx files from the fric_supplements folder, which have been acquired from Fric et al. (2020) supplements.

This code outputs the following files to the data folder: fric_data_header_1.txt, fic_data_header_2.txt, name_changes.csv, occurrences.RData

```
# load libraries
library(tidyverse)
library(ggplot2)
library(readxl)
library(lubridate)
```

*

###Data Input
The data.csv file was downloaded from https://doi.org/10.6084/m9.figshare.9946934 (https://doi.org/10.6084/m9.figshare.9946934) (https://figshare.com/articles/Phenology_responses_of_temperate_butterflies_-_Supplementary_data/9946934 (https://figshare.com/articles/Phenology_responses_of_temperate_butterflies_-_Supplementary_data/9946934))

This cvs datafile contains the occurrence data used in Fric et al. (2020), which they downloaded from gbif. The file includes separate data tables for each dataset, which have been concatenated into one file. These data tables have the same fields but are not formatted as a single data table; individual datasets were all written into one data file, including headers and row indices in each dataset. This first set of code reformats the data & writes formatted data files.

```
all.data <- readLines("fric_supplements/data.csv")

#identify header rows
all.header.rows<-grep("decimalLongitude", all.data)

#check headers for consistency
uniqueheaders<-unique(all.data[all.header.rows])

# 2 versions!

# 2 versions! -> Get row numbers for "header 1"
header.rows1<-grep(uniqueheaders[1], all.data)
#Get row numbers for "header 2"
header.rows2<-setdiff(all.header.rows, header.rows1)

#Create row identifiers:
#0 is a header row, 1 is format 1 data, 2 is format 2 data
j<-rep(0,length(all.data))
for (i in all.header.rows) {
  #set index to the next header if it's not the last header; otherwise set to end of datafile + 1
  if(i<max(all.header.rows)) {
    next_index<-min(all.header.rows[all.header.rows>i])
  }else { next_index<-length(all.data)+1 }

  #for data between header rows, set row index
  j[(i+1):(next_index-1)]<-ifelse(i%in%header.rows1,1,2)
}

#need to add a row index to the header text for new data files
newheader1<-paste('"row.index\",' ,uniqueheaders[1], sep="")
newheader2<-paste('"row.index\",' ,uniqueheaders[2], sep="")

#write data files for each header
formatteddatafile1<-file("data/fric_data_header_1.txt")
writeLines(c(newheader1,all.data[which(j==1)]), formatteddatafile1)
close(formatteddatafile1)

formatteddatafile2<-file("data/fric_data_header_2.txt")
writeLines(c(newheader2,all.data[which(j==2)]), formatteddatafile2)
close(formatteddatafile2)
```

```
rm(list=ls())

#read back in the formatted data
data1<-read_csv("data/fric_data_header_1.txt")
```

```
## Parsed with column specification:
## cols(
##   row.index = col_double(),
##   name = col_character(),
##   decimalLongitude = col_double(),
##   decimalLatitude = col_double(),
##   year = col_double(),
##   month = col_double(),
##   country = col_character(),
##   day = col_double(),
##   SuccDay = col_double(),
##   rndLat = col_double(),
##   alt = col_double()
## )
```

```
data2<-read_csv("data/fric_data_header_2.txt")
```

```
## Parsed with column specification:
## cols(
##   row.index = col_double(),
##   name = col_character(),
##   decimalLongitude = col_double(),
##   decimalLatitude = col_double(),
##   year = col_double(),
##   month = col_double(),
##   day = col_double(),
##   country = col_character(),
##   SuccDay = col_double(),
##   rndLat = col_double(),
##   alt = col_double()
## )
```

```
paste( nrow(data1), "records in format 1;", nrow(data2), "records in format 2")
```

```
## [1] "49243 records in format 1; 233201 records in format 2"
```

```
alldata<-bind_rows(data1,data2)
rm(data1,data2)
```

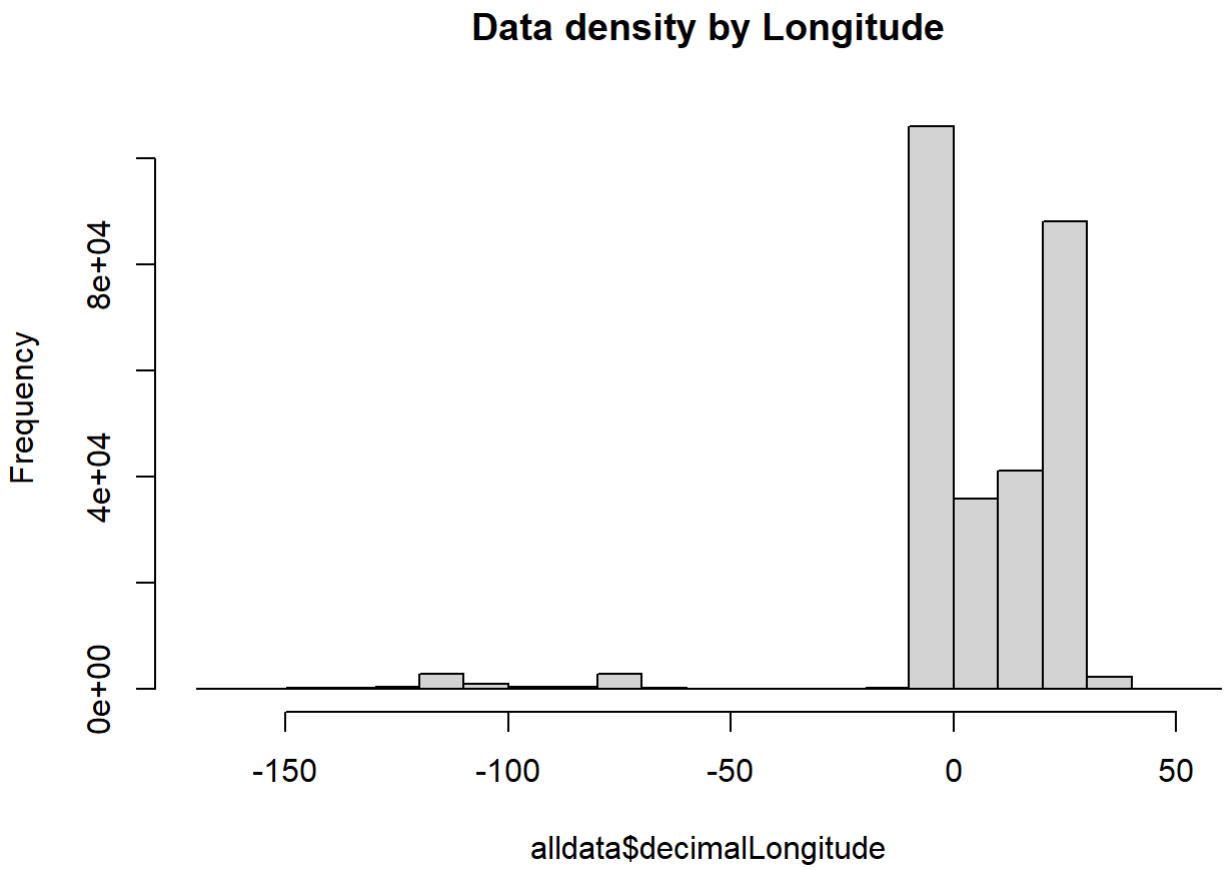
Data exploration 1: species names and regions

Now we assign regions to occurrence data and reconcile species names that don't match between the data file and results files provided in the original supplement.

```
summary(alldata)
```

```
##    row.index         name           decimalLongitude   decimalLatitude
##  Min.   :    1   Length:282444      Min.   :-162.559   Min.   : 5.787
##  1st Qu.: 2367   Class :character   1st Qu.:  -2.782   1st Qu.:52.781
##  Median : 7006   Mode  :character   Median :   9.398   Median :55.628
##  Mean   :14816                      Mean   :   6.298   Mean   :56.267
##  3rd Qu.:20210                      3rd Qu.:  23.573   3rd Qu.:60.624
##  Max.   :85273                      Max.   :  59.333   Max.   :71.216
##
##       year          month          country               day
##  Min.   :1616   Min.   : 1.000   Length:282444      Min.   : 1.00
##  1st Qu.:1992   1st Qu.: 6.000   Class :character   1st Qu.: 9.00
##  Median :2002   Median : 7.000   Mode  :character   Median :16.00
##  Mean   :1996   Mean   : 6.517                      Mean   :16.15
##  3rd Qu.:2009   3rd Qu.: 7.000                      3rd Qu.:24.00
##  Max.   :2015   Max.   :12.000                      Max.   :31.00
##  NA's   :58
##     SuccDay          rndLat           alt
##  Min.   :  2.0   Min.   : 6.00   Min.   :-2666.74
##  1st Qu.:163.0   1st Qu.:53.00   1st Qu.:   23.21
##  Median :186.0   Median :56.00   Median :   64.33
##  Mean   :181.6   Mean   :56.21   Mean   :  114.25
##  3rd Qu.:202.0   3rd Qu.:61.00   3rd Qu.:  111.09
##  Max.   :361.0   Max.   :71.00   Max.   : 4305.17
##
```

```
##Fric et al identifies datasets by region (N. America, Europe), but the data file does not include this information. We lab
el data by region using longitude:
## visualize data density by longitude
hist(alldata$decimalLongitude, main="Data density by Longitude")
```




```
#We label everything East of -40 as Europe, the rest as N. America
alldata<-alldata %>%
  mutate(region=ifelse(decimalLongitude>=(-40),"Europe","N. America"))

#We expect 100 species names, based on the manuscript.
length(unique(alldata$name))
```

```
## [1] 108
```

```
#What are the names in the dataset?
datanames<-sort(unique(alldata$name))
data.gs<-strsplit(datanames," ")
data.names <-as.data.frame(cbind(datanames,matrix(unlist(strsplit(datanames," ")),ncol=2,byrow=T)))
names(data.names)<-c("data.name","genus","spep")

#Which of these names shows up in the results?
result.names<-unique(na.omit(read_excel("fric_supplements/ele13419-sup-0003-tables2.xlsx", sheet="~latitude", range="A3:A11
3"))$Species)
resultnames<-(strsplit(result.names, " "))
result.names<-tibble(name=character(),genus=character(),spep=character())
for(i in 1:length(resultnames)) {
  genus<-paste(resultnames[[i]][1])
  spep<-paste(resultnames[[i]][2])
  name<-paste(genus,spep,sep=" ")
  temp.names<-tibble(name=as.character(name),genus=as.character(genus),spep=as.character(spep))
  result.names<-bind_rows(result.names,temp.names)
}
#which names match
which(data.names$data.name%in%result.names$name)
```

```
##  [1]   1   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
## [20]  21  22  23  25  26  27  28  29  31  32  33  34  35  36  37  38  39  40  41
## [39]  42  44  45  46  47  48  49  50  51  53  54  55  57  58  60  61  62  64  65
## [58]  67  68  69  70  72  73  74  75  76  78  79  80  81  82  83  84  85  86  87
## [77]  88  89  90  91  92  93  94  97  98 100 101 102 103 105 106 107
```

```
#not matched
names1<-data.names[which(!data.names$data.name%in%result.names$name),]
names2<-result.names[which(!result.names$name%in%data.names$data.name),]
names1$result.name<-NA

#First let's try fuzzy matching
for (i in 1:nrow(names1)) {
    if(length(agrep(names1$data.name[i], names2$name, ignore.case = TRUE, value = TRUE, max.distance = 0.1))>0) {
      names1$result.name[i]<-agrep(names1$data.name[i], names2$name, ignore.case = TRUE, value = TRUE, max.distance = 0.2)
    }
}
#names1 #Looks good

#now let's match on specific epithets
which(names2$spep%in%names1$spep[is.na(names1$result.name)])
```

```
## [1] 2 5 7 8
```

```
names1$result.name[which(names1$spep%in%names2$spep)]<-names2$name[match(names1$spep[which(names1$spep%in%names2$spep)],name
s2$spep)]
names1 #Looks good
```

```
##               data.name         genus       spep         result.name
## 2      Agriades optilete      Agriades   optilete Vacciniina optilete
## 24        Boloria selene       Boloria     selene                <NA>
## 30     Callophrys polios    Callophrys     polios    Callophrys polia
## 43       Cupido amyntula        Cupido   amyntula                <NA>
## 52         Erynnis tages       Erynnis      tages                <NA>
## 56    Euphydryas aurinia    Euphydryas    aurinia                <NA>
## 59     Fabriciana adippe    Fabriciana     adippe     Argynnis adippe
## 63  Incisalia augustinus     Incisalia augustinus                <NA>
## 66        Lethe eurydice         Lethe   eurydice  Satyrodes eurydice
## 71        Lycaeides idas     Lycaeides       idas                <NA>
## 77       Maculinea arion     Maculinea      arion                <NA>
## 95  Phyciodes campestris     Phyciodes campestris                <NA>
## 96      Phyciodes tharos     Phyciodes     tharos                <NA>
## 99    Plebejus saepiolus      Plebejus  saepiolus  Icaricia saepiolus
## 104  Scolitantides orion Scolitantides      orion                <NA>
## 108   Thymelicus lineola    Thymelicus    lineola Thymelicus lineolus
```

```
print("The species names in the results that are not present in the data are:")
```

```
## [1] "The species names in the results that are not present in the data are:"
```

```
names2$name[!names2$name%in%names1$result.name]
```

```
## [1] "Phyciodes cocyta"    "Phyciodes pratensis"
```

```
#GBIF considers Phyciodes cocyta a synonym of Phyciodes tharos (https://www.gbif.org/species/1918971)
#GBIF considers Phyciodes pratensis a synonym of Phyciodes campestris (https://www.gbif.org/fr/species/1918960)
names1$result.name[names1$data.name=="Phyciodes tharos"]<-"Phyciodes cocyta"
names1$result.name[names1$data.name=="Phyciodes campestris"]<-"Phyciodes pratensis"

#Now we can match data specific epithets to other results specific epithets
shared.spep<-result.names$spep[which(result.names$spep%in%names1$spep[is.na(names1$result.name)])]

names1$result.name[which(names1$spep%in%shared.spep)]<-result.names$name[which(result.names$spep%in%shared.spep)]

names1
```

```
##                data.name         genus       spep           result.name
## 2      Agriades optilete      Agriades   optilete   Vacciniina optilete
## 24        Boloria selene       Boloria     selene                  <NA>
## 30     Callophrys polios    Callophrys     polios      Callophrys polia
## 43       Cupido amyntula        Cupido   amyntula                  <NA>
## 52         Erynnis tages       Erynnis      tages                  <NA>
## 56    Euphydryas aurinia    Euphydryas    aurinia                  <NA>
## 59     Fabriciana adippe    Fabriciana     adippe       Argynnis adippe
## 63  Incisalia augustinus     Incisalia augustinus Callophrys augustinus
## 66        Lethe eurydice         Lethe   eurydice    Satyrodes eurydice
## 71        Lycaeides idas     Lycaeides       idas         Plebejus idas
## 77       Maculinea arion     Maculinea      arion       Phengaris arion
## 95  Phyciodes campestris     Phyciodes campestris   Phyciodes pratensis
## 96      Phyciodes tharos     Phyciodes     tharos      Phyciodes cocyta
## 99    Plebejus saepiolus      Plebejus  saepiolus    Icaricia saepiolus
## 104  Scolitantides orion Scolitantides      orion                  <NA>
## 108   Thymelicus lineola    Thymelicus    lineola   Thymelicus lineolus
```

```
#It is unclear if any other species names in the data contribute to the results.
#Euphydryas aurinia is removed by Fric et al.
names1$result.name[names1$data.name=="Euphydryas aurinia"]<-""
#This leaves four species names, which we will not address.

write.csv(names1, file="data/name_changes.csv")
# this file can now be used for correcting names in the main file

for(namei in 1:nrow(names1)) {
  alldata$name[alldata$name==names1$data.name[namei]]<-names1$result.name[namei]
}
```

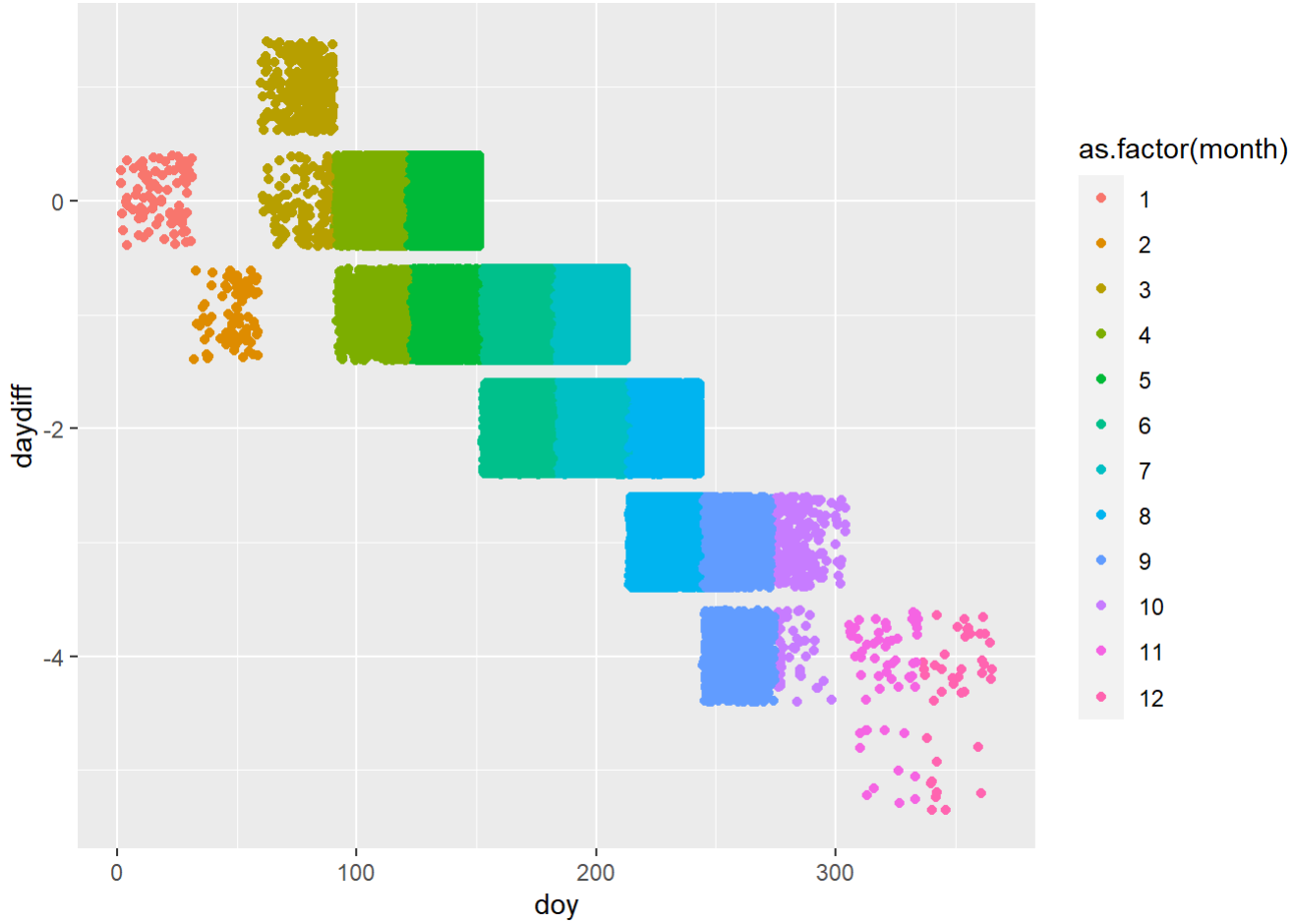
New code 11/25/2020: day of year reconciliation

Until today, we had assumed that the “SuccDay” values were a consistent index for day of year. However, we had not documented our initial spot-checking of altitudes. While identifying GBIF records for documented spot-checking, we found some inconsistencies in the SuccDay value. Here we identify how “SuccDay” was calculated.

```
#DOES SUCCDAY MATCH DOY?
checkdays<-na.omit(alldata) %>%
  mutate(doy=yday(as.Date(paste(year,month,day, sep="-"), "%Y-%m-%d")),
         daydiff=SuccDay-doy, fricday=(month-1)*30+day) %>%
  select(name,day,month,year,SuccDay,doy,daydiff,fricday)
#summary(checkdays)
table(checkdays$fricday-checkdays$SuccDay)
```

```
##
##      0
## 264889
```

```
ggplot(data=checkdays, aes(y=daydiff, x=doy, color=as.factor(month))) + geom_jitter()
```




```
#we'd prefer to use calendar day
alldata<-alldata %>%mutate(doy=yday(as.Date(paste(year,month,day, sep="-"),"%Y-%m-%d")))
rm(checkdays)

##Save formatted occurrence data file
save(alldata, file="data/occurrences.RData")
```

End of File.