Investigating the Fossil Record of the Soricidae Family

Jeffrey Lee

Abstract

The Soricidae family consists of small mice-like mammals commonly known as shrews. They are abundant throughout several continents, primarily living in the grasslands. However, this abundance is not reflected in the current fossil record. The objective of this project is to investigate and analyze the fossil record. By doing so, this project will be able to determine the limits of the current fossil record. Analysis of the fossil record datasets will be done in Python while visual representation of the fossil record will be done using RStudio.

Introduction

The Soricidae family is an extremely diverse family of small mammals made up of shrews or shrew mouses. The Soricidae family can be further broken down into five subfamilies: the Heterosoricinae subfamily, the Crocidurinae subfamily, the Limnoecinae subfamily, the Soricinae subfamily, and the Allosoricinae subfamily (Repenning 1967). Of these subfamilies, only the Soricinae and the Crocidurinae subfamilies are extant. The Soricidae family is an extremely old family with the oldest Soricidae species existing in the middle of the Eocene Epoch in North and Central America (Minwer-Barakat *et al.* 2010). Despite the extinction of three out of the five subfamilies, there still exists about 335 extant Soricidae species (Quérouil *et al.* 2001) that can be found throughout the grasslands that make up Eurasia, Africa, and both Americas (Minwer-Barakat *et al.* 2010).

Due to the fact that shrews exhibit extremely high within-habitat diversity, the Soricidae family plays an important role in research into the interactions and interspecific competition of small mammals (Kirkland Jr 1991). Additionally, the consistent size, physiology, and behavior patterns between different species of the Soricidae family provides key insights into the mechanisms of small mammal interactions (Kirkland Jr 1991). It should be noted that while this consistency proves to be useful for researchers, it has led to some problems when analyzing the fossil record. The various fossils of the Soricidae family proves to be extremely difficult to sort into their respective subfamilies and species due to the lack of difference in the external morphology of different species (Dubey *et al.* 2008). This proves to be particularly troublesome due to the fact that taxonomy of the Soricidae family is primarily classified and analyzed using morphology featurues (Willows-Munro & Matthee 2011). This issue is further compounded by the relative lack of fossils in the fossil record (Dubey *et al.* 2008). As a result, it is important to look at the Soricidae family's current fossil record to determine its accuracy and usefulness for further research.

Materials and Methods

Downloading and Cleaning the Fossil Record Data

The fossil record for the Soricidae family is required to analyze the extinction and diversification rates. This data was downloaded from the Paleobiology Database as a comma separated value (csv) file. The file was examined and cleaned in the shell terminal using the following commands:

```
mv ~/Downloads/pbdb_data.csv ~/Desktop/EEB-177/eeb-177-final-project/Soricidae_data.csv
# Checked the number of species
tail -n +19 Soricidae_data.csv | cut -d "," -f 6 | sort |uniq | wc -l
# Removed the header
tail -n +18 Soricidae_data.csv > Soricidae_data_full.csv
```

The first line of code moved the csv file to the correct folder while also renaming it to reflect that the csv file contained data for the Soricidae family. The second line of code was run to verify that the data had the appropriate number of species to work with. The third line of code was used to remove the header which was unimportant for this project.

Looking at the Fossil Record

One of the first things that was looked at was the age of the fossils in the fossil record. This was achieved by finding the difference between the first and last occurance of each species in the fossil record. This was achieved by utilizing a for loop to run through the csv file and extract the first and last occurance in the fossil record from the appropriate columns. The first occurance was assigned to the variable max_age and the last occurance was assigned to min_age. By finding the difference between the two variables, the age/duration of each fossil was determined in terms of millions of years ago. This value was then appended to an empty dictionary that had the species' name as they key and the duration of the fossil record as the value.

```
def calculate_age(filename):
    Soricidae_dict = {}
    # Creates an empty dictionary
    Soricidae_species = open(filename, "r", encoding = "ISO-8859-15")
    # Will open the data file
    Soricidae_records = Soricidae_species.readlines()[1:]
    for line in Soricidae_records:
        line = line.replace('\"', '')
        # Removes the extra quotes
        record_elements = line.split(",")
        # Denotes that each record element is separated by a file
        species = record_elements[5]
        # Making species be the keys and taking the names from column 5
```

```
        max_age = float(record_elements[15])
        # Assigning max age to column 15
        min_age = float(record_elements[16])
        # Assigning min age to column 16
        age = max_age - min_age
        # Calculating the age of the fossil
        Soricidae_dict[species] = age
        # Assigning the age as the values in the dictionary
    return Soricidae_dict

calculate_age("Soricidae_data_full.csv")
```

The next two functions created dictionaries that have the Soricidae species as the key with either the first occurance in the fossil record or the last fossil occurance in the fossil record as the value. This was done in the same way as the first function; a for loop was utilized to parse the data files for the specific column associated with either the first or last occurance and that value was added to an empty dictionary. The dictionary containing the species as the key and the first occurance in the fossil record was called 'fo_dict' and the dictionary containing the species as the key and the last occurance in the fossil record was called 'lo_dict'.

```
# Create a dictionary with the species as the key and the first occurances as the valu
def FirstOccurance(filename):
    firstOccurance_dict = {}
    # Create an empty dictionary
    species_data = open(filename, "r", encoding = "ISO-8859-15")
    # Open file
    firstOccurance_records = species_data.readlines()[19:]
    # Read the file
    for line in firstOccurance_records:
        line = line.replace('\"', '')
        firstOccurance_elements = line.split(",")
        species = firstOccurance_elements[5]
        # Populate the species with the data in column 6
        start_year = float(firstOccurance_elements[15])
        # Populate the start year with data from column 16
        firstOccurance_dict[species] = start_year
        # Assign species as keys and start year as values
    return firstOccurance_dict

fo_dict = FirstOccurance("Soricidae_data_full.csv")
```

```
# Create a dictionary with the species as the key and the last occurances as the value
def LastOccurance(filename):
    lastOccurance_dict = {}
    # Crete an empty dictionary
```

```
    rspecies_data = open(filename, "r", encoding = "ISO-8859-15")
    # Open file
    lastOccurance_records = rspecies_data.readlines()[19:]
    # Read the file
    for line in lastOccurance_records:
        line = line.replace('\"', '')
        lastOccurance_elements = line.split(",")
        species = lastOccurance_elements[5]
        # Populate the species with the data in column 6
        end_year = float(lastOccurance_elements[16])
        # Populate the end year with data from column 17
        lastOccurance_dict[species] = end_year
        # Assign species as keys and end year as values
    return lastOccurance_dict

lo_dict = LastOccurance("Soricidae_data_full.csv")
```

Next, utilizing a Python module called collections, defaultdict was called upon to create a dictionary that took the two dictionaries created in the previous two functions ('fo_dict' and 'lo_dict') and combined them into a new dictionary called 'combined_dict'. This new dictionary has the species as the key and both the first and the last occurance in the fossil record as the values. This dictionary could be utilized to verify the duration of each species in the fossil record that was found in the 'calculate_age' function in the first code. Additionally, this combined dictionary could be used to verify the accuracy of the graph that would be later created.

```
# Create a combined dictionary in which each key (species) has both first and last occ
from collections import defaultdict

combined_dict = defaultdict(list)
# Create lists for values within dictionary

for d in (fo_dict, lo_dict):
    for key, value in d.items():
        combined_dict[key].append(value)
        # Append both values as a list for each key in a new combined dictionary

combined_dict
```

The next step in our investigation of the fossil record was to create a graph visualizing the duration of the fossils. The graph was created in RStudio (code shown later). However, before the creation of the graph is possible, Python was utilized to create a new csv file containing just the species name and the first and last occurance in the fossil record. This was done by two for loops. The first for loop calculated the mean age of each species and appended it to a dictionary. The second for loop created a new csv file that contained the

genus, species, minimum age, and maximum age. This new csv file was used for the graph.

```
## Open the file

infile = "Soricidae_data_full.csv"
with open(infile, 'r') as ff:
    Soricidae_recs = ff.readlines()[1:]
```

```
# Make dictionary
from collections import defaultdict

species_range = defaultdict(list)

for line in Soricidae_recs:
    species = line.split(",")[5]
    minage = line.split('","')[14]
    maxage = line.split('","')[15]

    mean_age = (float(minage) + float(maxage)) / 2

    # Add species as keys and mean ages as elements to the list of ages
    species_range[species].append(mean_age)
```

```
output = open("Soricidae-ranges.csv", "w")

for key in species_range.keys():
    ages = species_range[key]
    minage = min(ages)
    maxage = max(ages)
    genus = key.split(" ")[0]
    species = key
    outline = "{},{},{},{}\n".format(genus, species, minage, maxage)
    output.write(outline)
```

The graph was made using the ggplot2 package in RStudio. The first step was to read in the new csv file created in the previous Python code. This can be seen below.

```
# Read in the file
library(ggplot2)
setwd("/home/eeb177-student/Desktop/EEB-177/eeb-174-final-project/project-deadline-6/")

Soricidae <- read.csv("/home/eeb177-student/Desktop/EEB-177/eeb-174-final-project/final-

names(Soricidae) <- c("genus", "species", "minage", "maxage")
head(Soricidae)
```

The code to make the graph itself was made using ggplot and can be seen below. The graph

was made with the x-axis set as “Millions of Years Ago” and the y-axis as “Species”. As seen in the comments of the code, the other lines of code were primarily used to clean up and adjust the aesthetics of the graph such as changing text sizes to make the graph readable.

```
Soricidae_occ <- ggplot(Soricidae, aes( species, ymin = maxage, ymax=minage, colour = ge
Soricidae_occ <- Soricidae_occ + geom_linerange()
# Remove the legend
Soricidae_occ <- Soricidae_occ + theme(legend.position="none")
# Flip axis
Soricidae_occ <- Soricidae_occ + coord_flip()
# Change text size
Soricidae_occ <- Soricidae_occ +  theme(axis.text.y = element_text(size=1.75))
# Remove tick marks on y-axis
Soricidae_occ <- Soricidae_occ + theme(axis.ticks.y=element_blank())
# Remove excess white space
Soricidae_occ <- Soricidae_occ + scale_y_continuous(limits=c(0, 40), expand = c(0, 0), b
# Add title and labels
Soricidae_occ <- Soricidae_occ + labs(title = "Soricidae Fossil Occurrences", x = "Speci
Soricidae_occ
```

The final graph was then saved as pdf using the following code:

```
ggsave(filename = "Soricidae-occ.pdf", plot = Soricidae_occ)
```

After analyzing the age of the fossil record, the location of each fossil in the fossil record was analyzed. First, a dictionary was created for the latitude (lat_dict) and the longitude (long_dict) for each fossil. These dictionaries were populated using a for loop to go through the csv file to extract the appropriate columns - column 30 contained the longitudes and column 31 contained the latitudes. In both dictionaries, the species was the key while the latitude or the longitude was the value. These two functions are extremely similar to the functions used to create ‘lo_dict’ and ‘fo_dict’.

```
# Make a dictionary for the species and longitudes
def Longitudes(filename):
    long_dict = {}
    # Create an empty dictionary
    species_data = open(filename, "r", encoding = "ISO-8859-15")
    # Open file
    long_records = species_data.readlines()[1:]
    # Read the file
    for line in long_records:
        line = line.replace('\"', '')
        long_elements = line.split(",")
        species = long_elements[5]
        # Populate the species with the data in column 6
        long = float(long_elements[29])
        # Populate the longitudes with data from column 30
```

```
        long_dict[species] = long
        # Assign species as keys and start year as values
    return long_dict

long_dict = Longitudes("Soricidae_data_full.csv")
```

```
# Create a dictionary of latitudes
def Latitudes(filename):
    lat_dict = {}
    # Crete an empty dictionary S
    rspecies_data = open(filename, "r", encoding = "ISO-8859-15")
    # Open file
    lat_records = rspecies_data.readlines()[1:]
    # Read the file
    for line in lat_records:
        line = line.replace('\"', '')
        lat_elements = line.split(",")
        species = lat_elements[5]
        # Populate the species with the data in column 6
        lat = float(lat_elements[30])
        # Populate the latitudes with data from column 31
        lat_dict[species] = lat
        # Assign species as keys and latitudes as values
    return lat_dict

lat_dict = Latitudes("Soricidae_data_full.csv")
```

Much like the first and last occurance dictionaries, the latitude and longitude dictionaries were combined into a master dictionary. This new combined dictionary has the species as the key and the latitude and the longitude as the value. This master dictionary could be used to verify the accuracy of the fossil location map created later.

```
# Create a combined dictionary in which each key (species) has both latitudes and long
from collections import defaultdict

combined_dict = defaultdict(list)
# Create lists for values within dictionary

for d in (long_dict, lat_dict):
    for key, value in d.items():
        combined_dict[key].append(value)
        # Append both values as a list for each key in a new combined dictionary

combined_dict
```

The next step in looking at the location of the fossils was to create a map that plotted the locations of the fossils. In order to create a map in RStudio, a cleaned csv file containing only the latitude and longitudes of the fossils was created in the shell. The shell command is listed below.

```
# Cut the columns for latitude and longitude and create a new csv file with just those
cut -d "," -f 30-31 Soricidae_data_full.csv > latlong.csv
```

In RStudio, a world map was populatd with the appropriate latitudes and longitudes for each fossil. This was made possible by utilizing ggplot2 and maps. First the newly created csv file containing the latitudes and longitudes was read in.

```
Soricidae <- read.csv("/home/eeb177-student/Desktop/EEB-177/eeb-174-final-project/final-
```

Next, the code calls upon an empty world map, the fill color of the map was removed to create just a map with an outline of the countries. This was done for visibility purposes. The map was then populated with the Soricidae fossil locations with the x-axis as longitude and the y-axis as latitude. The map was then saved as a pdf file.

```
library(ggplot2)
library(maps)

world_map <- map_data("world")
plot <- ggplot() + coord_fixed() + xlab("") + ylab("")
empty_world_plot <- plot + geom_polygon(data=world_map, aes(x=long, y=lat, group=group),

filled_in_world_plot <- empty_world_plot + geom_point(data = Soricidae,aes(x = lng, y =
filled_in_world_plot <- filled_in_world_plot + labs(title = "Soricidae Fossil Records",
filled_in_world_plot <- filled_in_world_plot + theme(plot.title = element_text(hjust = (

filled_in_world_plot

ggsave(filename = "Fossil_Record_Map.pdf", plot = filled_in_world_plot)
```

A map showing the fossil locations of current extant species in the fossil record was created in the same way. The extant species' location csv (extant.csv) was manually created by referring to the two combined dictionaries created in the aforementioned code (the two combined dictionaries).

```
library(ggplot2)
library(maps)

world_map <- map_data("world")

plot <- ggplot() + coord_fixed() + xlab("") + ylab("")

empty_world_plot2 <- plot + geom_polygon(data=world_map, aes(x=long, y=lat, group=group)
```

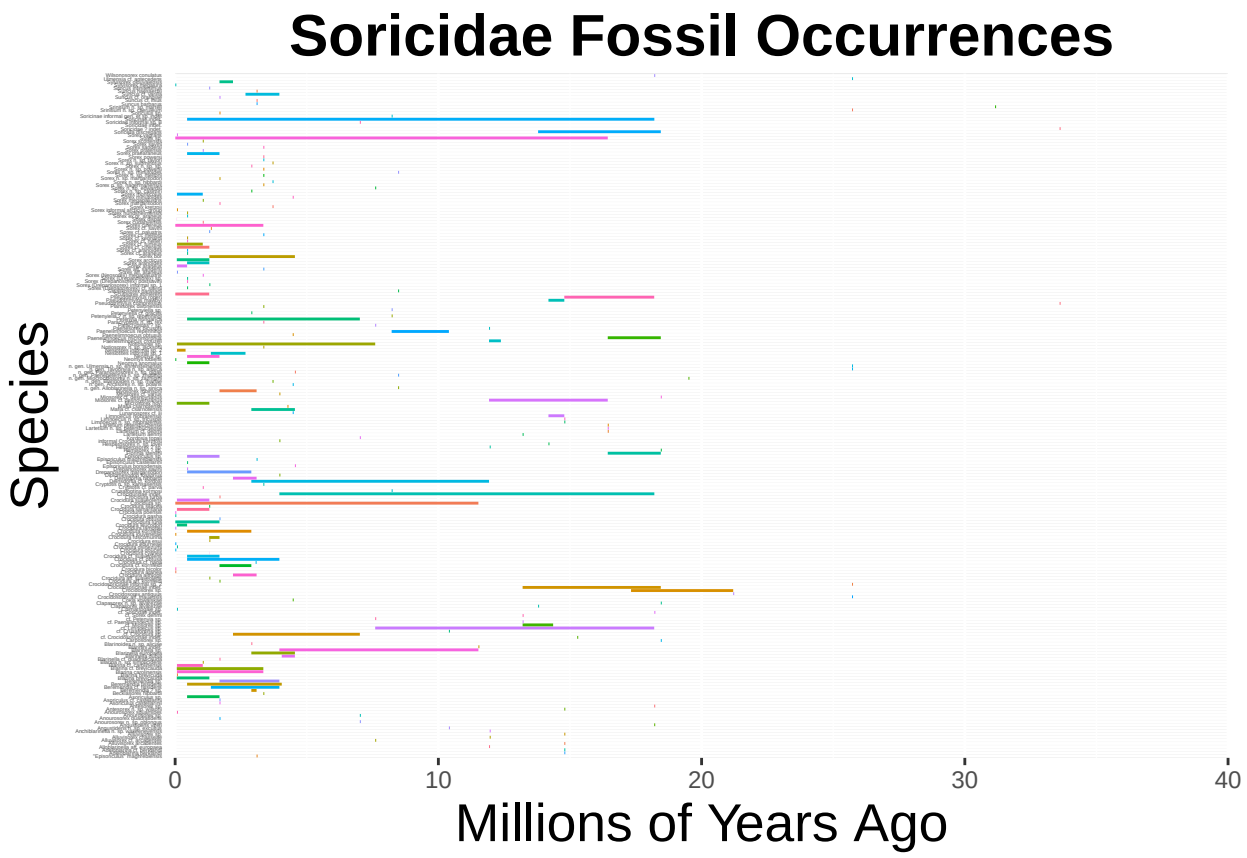
```
Soricidae <- read.csv("/home/eeb177-student/Desktop/EEB-177/eeb-174-final-project/final-

filled_in_world_plot2 <- empty_world_plot2 + geom_point(data = Soricidae,aes(x = lng, y
filled_in_world_plot2 <- filled_in_world_plot2 + labs(title = "Soricidae Fossil Records"
filled_in_world_plot2 <- filled_in_world_plot2 + theme(plot.title = element_text(hjust =

filled_in_world_plot2

ggsave(filename = "Fossil_Record_Map_Extant.pdf", plot = filled_in_world_plot2)
```

Results and Discussion

Fossil Occurrences Within the Family

The graph of the timeline of the fossil record provides a clear visualization of the number species throughout time. According to the fossil record, the number of extant species is greatly outnumbered by the number of extinct species.

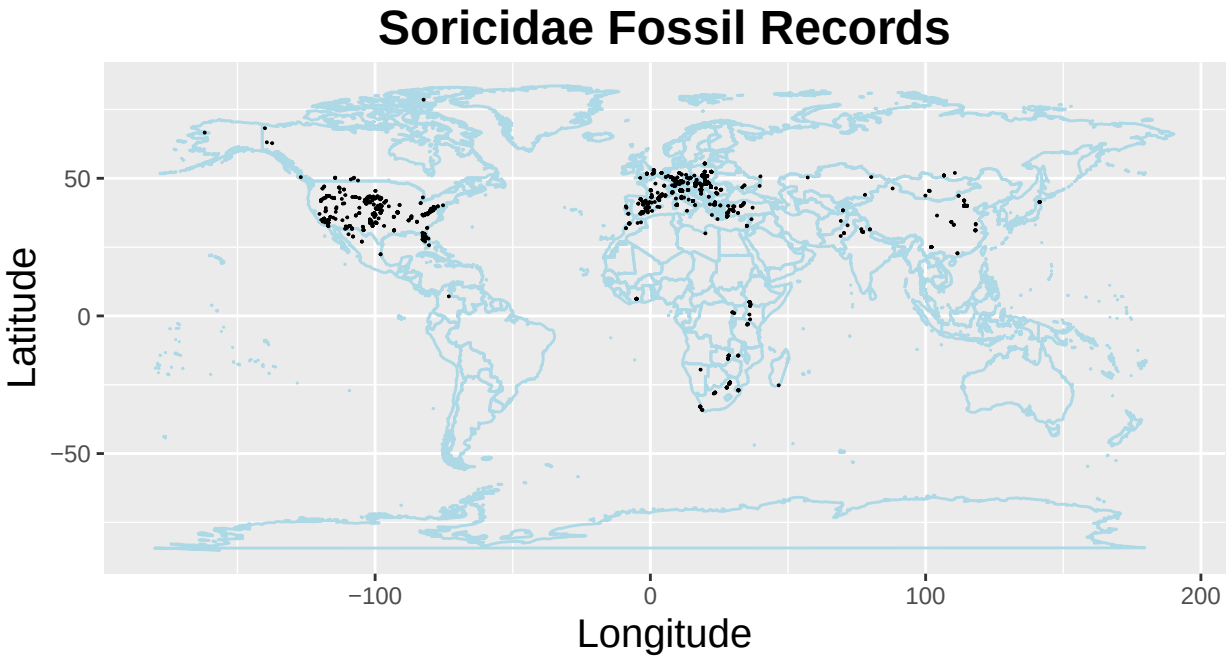
```
## Warning: Removed 1 rows containing missing values (geom_linerange).
```



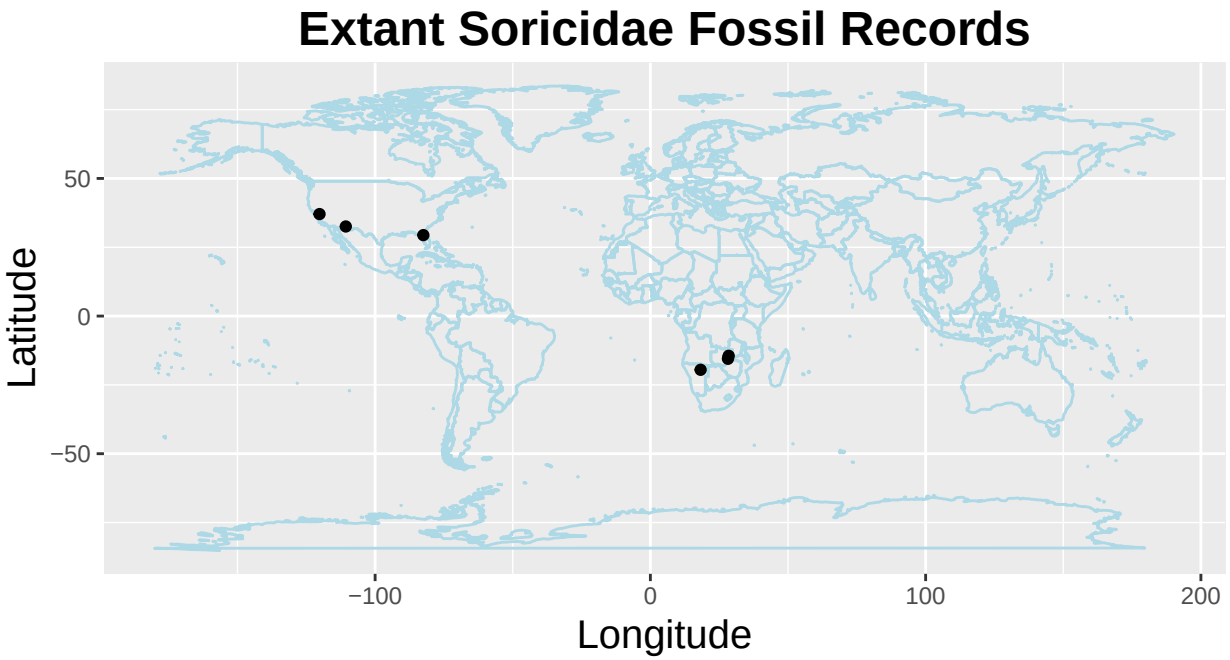
According to this graph, there are about six extant speices in the Soricidae family. Furthermore, this graph only shows about ~30 species in the Soricidae family in the fossil record. According to extensive literature, we know that this is not true. More than anything else, this graph shows that the fossil record is not reflective of the reality of the Soricidae family. This may be due to the fact that the Soricidae family fossils prove to be extremely hard to classify.

Fossil Occurances Throughout the World

The folllowing is the map of the entire fossil record throughout the world.



The following is the map of the extant species found in the fossil record throughout the world.




```
## Saving 6.5 x 4.5 in image
```

The fossil record map shows the distribution of the Soricidae family spans throughout North America, Africa, Europe, and parts of Asia. This data is consistent with literature that documents the habitat of Soricidae species (Minwer-Barakat *et al.* 2010). However, looking at the extant species fossil record, inconsitences can be found. There are upwards of 335 documented extant species of the Soricidae family (Quérouil *et al.* 2001); however, the fossil record only shows six.

Both the time of the fossil record and the map of the fossil record show inconsistencies with modern-day knowledge of the Soricidae family. This is reflective of the fact that there remains a relative lack of a Soricidae fossil record (Dubey *et al.* 2008). Even among the fossils that researchers do have, there remains the difficulty in classifying the fossils found. Despite these problems, the collective fossil record map does provide a consistent distribution of the Soricidae family across the world.

Conclusion

When compared with modern-day research, the fossil record is revealed to have many flaws. Many of these flaws have been documented throughout this paper. The most prominent issue is the lack of a complete fossil record (Dubey *et al.* 2008). This could be due to several reasons. Due to their plesiomorphic morphology, the Soricidae family proves hard to classify (Willows-Munro & Matthee 2011). This could potentially have caused some Soricidae family fossils being classified as other families. Additionally, due to their relatively small size (body sizes of only a few inches), the Soricidae family may not fossilize as well as other mammals. One potential area for future research may be to move away from utilizing external morphology as an analysis technique; instead, mitochondrial DNA can potentially be used to

classify and sort through the fossil record. Though this will not increase the current fossil record, this provides an avenue for better classification of the fossils that we do have. This could ultimately lead to an increase in the accuracy of the fossil record for future research.

Github Link

https://github.com/Jeffrey-Lee1797/eeb-174-final-project

References

1.Dubey, S., Salamin, N., Ruedi, M., Barrière, P., Colyn, M. & Vogel, P. (2008). Biogeographic origin and radiation of the old world crocidurine shrews (mammalia: Soricidae) inferred from mitochondrial and nuclear genes. *Molecular Phylogenetics and Evolution*, 48, 953–963.

2.Kirkland Jr, G.L. (1991). Competition and coexistence in shrews (insectivora: Soricidae). *The biology of the Soricidae. Special Publication of the Museum of Southwestern Biology, University of New Mexico, Albuquerque*, 15–22.

3.Minwer-Barakat, R., García-Alix, A., Suárez, E.M. & Freudenthal, M. (2010). Soricidae (soricomorpha, mammalia) from the pliocene of tollo de chiclana (guadix basin, southern spain). *Journal of Vertebrate Paleontology*, 30, 535–546.

4.Quérouil, S., Hutterer, R., Barrière, P., Colyn, M., Peterhans, J.C.K. & Verheyen, E. (2001). Phylogeny and evolution of african shrews (mammalia: Soricidae) inferred from 16s rRNA sequences. *Molecular phylogenetics and evolution*, 20, 185–195.

5.Repenning, C.A. (1967). *Subfamilies and genera of the soricidae*. US Govt. Print. Off.,

6.Willows-Munro, S. & Matthee, C. (2011). Exploring the diversity and molecular evolution of shrews (family soricidae) using mtDNA cytochrome b data. *African Zoology*, 46, 246–262.