Downloading and Visualizing Project Data from the iNaturalist Database

Overview

This tutorial will teach you skills to create and manipulate datasets from iNaturalist. There are four main activities in this guide. The first activity is a general tutorial on how to download data from a project on iNaturalist. The second activity is a tutorial on loading datasets into R. The third activity gives examples of how to explore plant-pollinator interaction data using R. The fourth activity describes how to create a bipartite network with R to visualize plant-pollinator interactions. The example in this exercise uses data from the Calgary Pollinators Project on iNaturalist and looks at plant-pollinator interactions. However, you may adapt this guide for any project on iNaturalist.

1.	Downloading data from iNaturalist	5 – 10 minutes
2.	Loading your dataset into R	5 – 10 minutes
3.	Investigating pollinator-plant interactions with R	15 – 20 minutes
4.	Visualizing interactions: creating network matrices with R	15 – 20 minutes

Objectives

- Create a query, export, and download data from iNaturalist
- Load data into R
- Explore pollinator-plant interactions with R
- Create a heat map, bipartite, and bipartiteD3 network in R

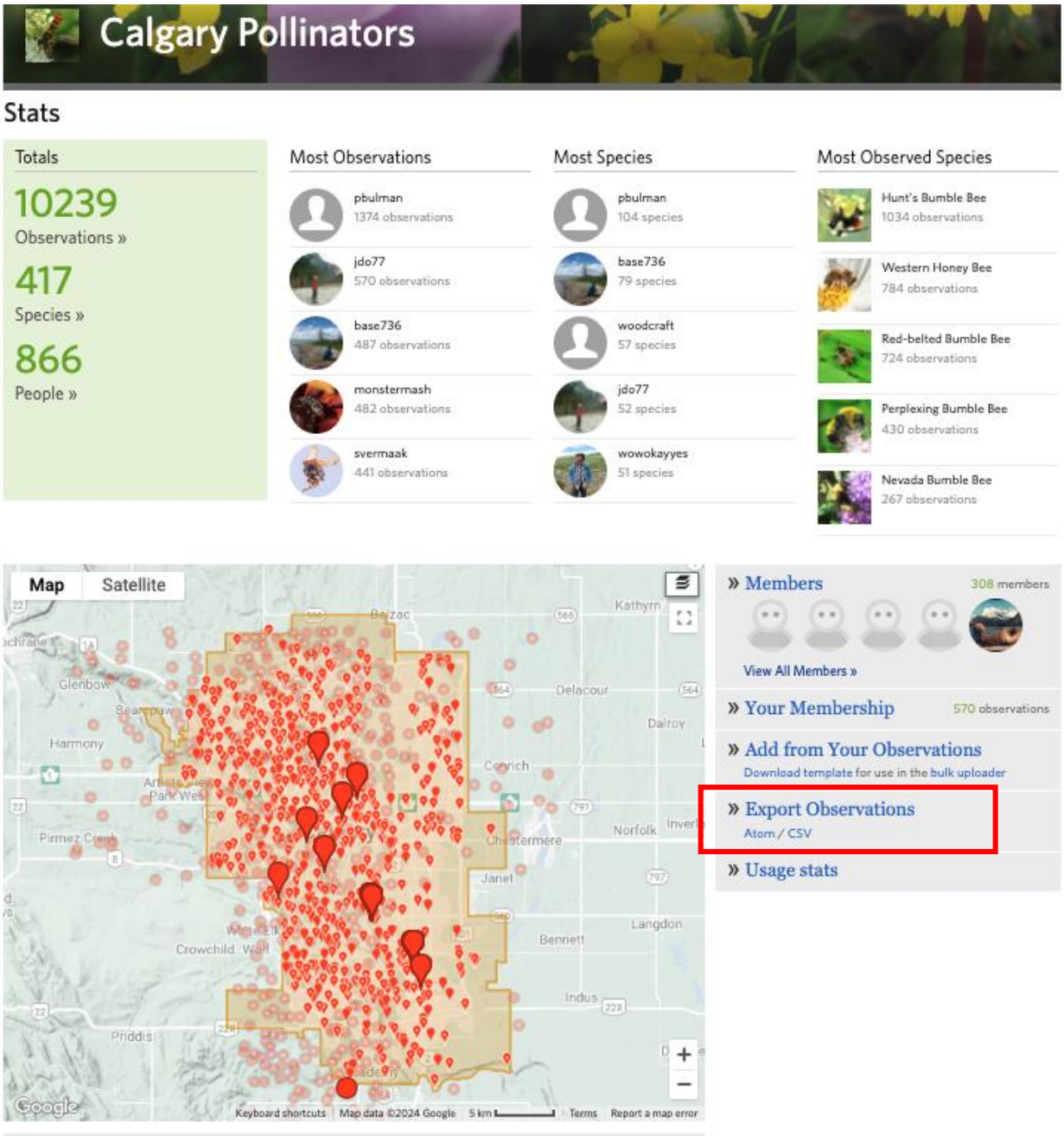
Materials

- Internet browser
- iNaturalist account (optional)
- R / R Studio

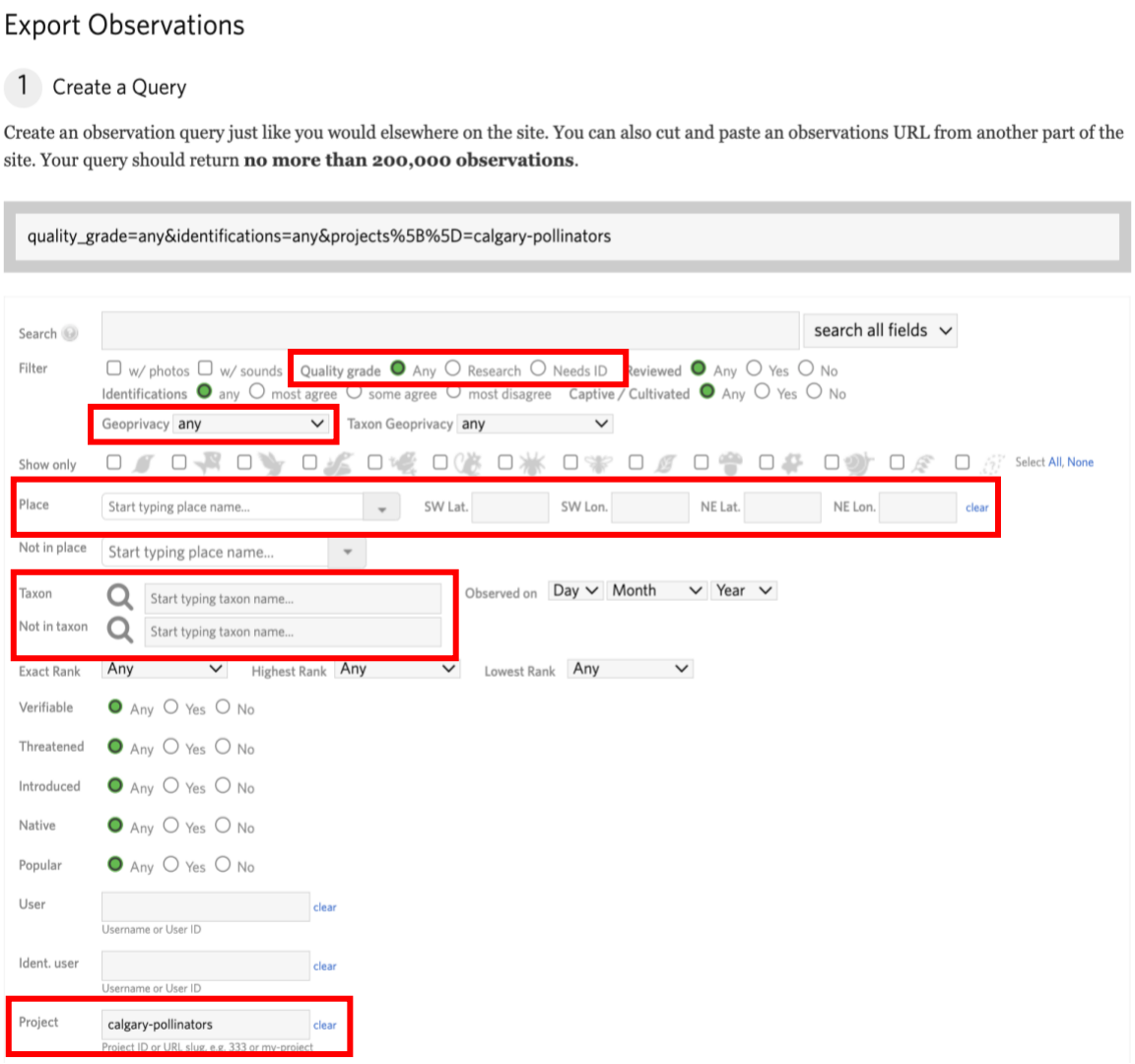
Total activity time **40 – 60 minutes**

1. Downloading data from iNaturalist

Search the Calgary Pollinators Project on iNaturalist and go to the project page. On the homepage, click on “Export Observations”. This will take you to a query where you can fill out specific details to tailor your dataset.



To build your query there are multiple fields that you can use to filter specific data.



Quality Grade (eg. Research Grade or Needs ID) – If you want all the observations included or just Research Grade observations with confirmed IDs.

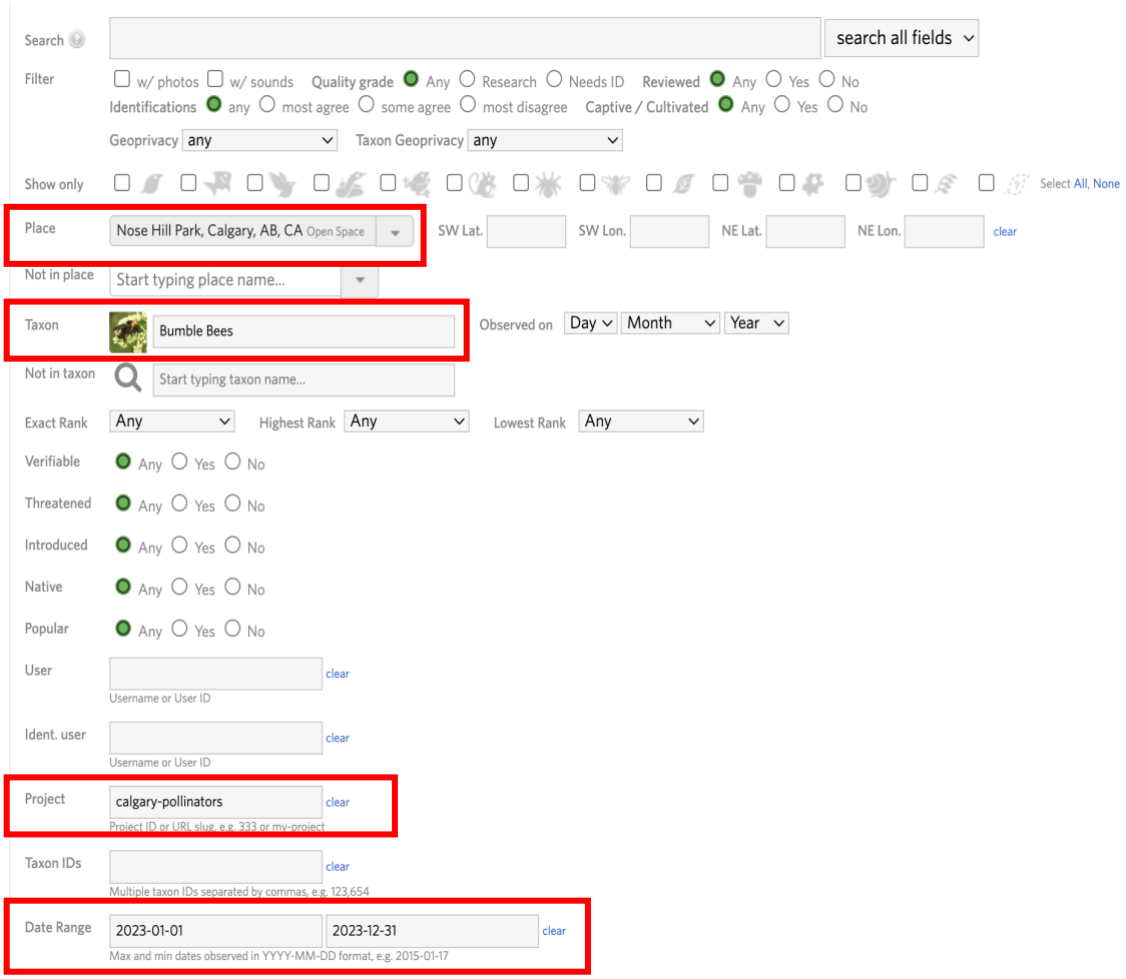
Geoprivacy – For privacy reasons some coordinates of observations can be obscured. If the exact location of observations is not important for your search, then do not worry about this filter function.

Place/Location – Filter by country, province, region/district, town/city, postal code, park, or create a boundary with latitude and longitude coordinates. Alternatively, you can filter to exclude places where you do not want observations to be included.

Taxon – Filter observations by taxonomic group (Order, Family, Genus, or Species, etc.). Alternatively, you can filter to exclude taxonomic groups you do not want to be included.

Project – Filter observations from a specific project.

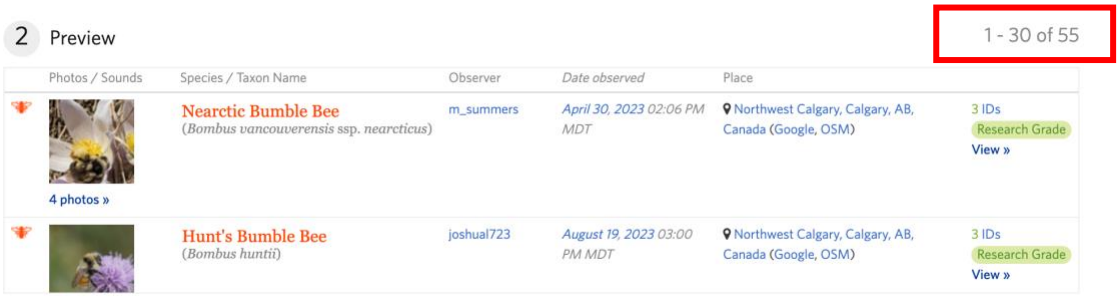
Date Range – Select observations from a specific date range.



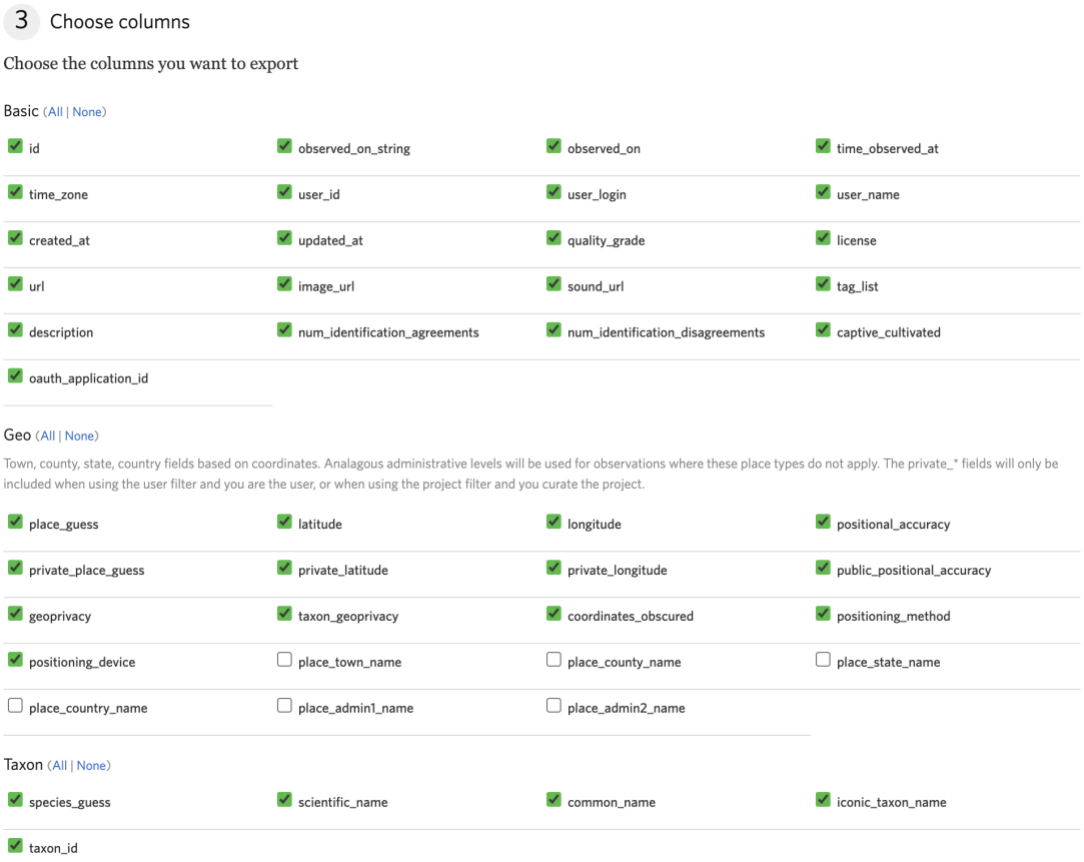
Example 1) After looking through different species' pages, you are interested in learning more about bumble bees (*Bombus*) at Nose Hill Park. You want to determine how many observations were made in 2023.

If you are unsure which area of Calgary you would like to explore plant and pollinator diversity, see Appendix A: Where to Explore Biodiversity for suggestions.

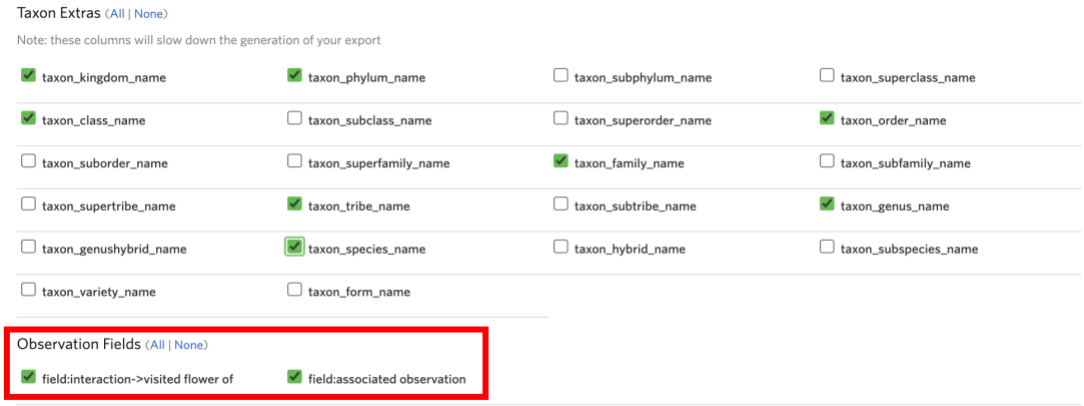
Once you select the parameters for your query, you will be able to see a preview of the observations included. It will show you the picture, species/taxon name, observer, date observed, place, and the number of IDs/quality grade. It will also tell you the total number of observations included in your query.



Next you will need to choose your columns. There are several columns pre-selected by iNaturalist for the data export. You can select/unselect any columns that are not relevant. Data columns are organized by Basic info, Geo info, and Taxon info.



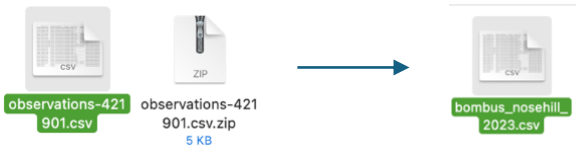
There are extra taxonomic categories you can select if you wish. If your project has extra observation fields like the Calgary Pollinators Project, be sure to select those fields so they are included in your data download. The Observation Field *"Interaction -> Visited flower of"* or similar association field is needed to investigate interactions between plants and their respective pollinators.



When you have selected all the relevant query parameters, create your data export.

Depending on the size of your download, it may take a few minutes. There will be a pop-up that gives you the option to wait or have your download emailed to you once it's complete. Once the export is complete, download the file.

The export will download as a csv zip file. Click the zip file to unzip the csv file of your download. The default download setting is csv format – if you wish to work with your data in Excel, you can right click the file to open with Excel and this will convert the file to xlsx format. We will be working with the data in R which prefers csv format.



Rename your file to describe your data and save it in a folder that is easily accessible so you can load the file into R. I named my file bombus_nosehill_2023 because my data contains information about bumble bees (*Bombus*) found at Nose Hill in 2023. Note: your file name should use underscores instead of spaces if you want to load the file to R.

2. Loading your dataset into R

Open R studio and create a new R script. You may copy and paste any code in this document to use in your own R session. Base R can be to compute basic statistics and analyses, although, there are many pre-built packages that are downloadable and can streamline that data manipulation and analyses processes (R Core Team, 2022). Other packages like bipartite and bipartiteD3 can build, analyze, and visualize networks. Networks allow us to visualize and analyse the connectivity and interactions between trophic levels such as plants and pollinators, and with computer programs like R, we can do these network analyses efficiently on a large scale (Newman, 2010; R Core Team, 2022)

With any R project, the first thing you always want to do is set your global defaults and clear your global environment. This ensures that there are no objects in the background that might alter the output of your code. You may like to reference this R for Beginners tutorial for more guidance on how to use R.

```
rm(list = ls(all = TRUE)) # clear global environment
```

Next you will want to check and set your working directory location. This allows R to check for files in a certain location on your computer. If your working directory is not set to where you would like it, you can change it. Keep your R file and the data you will be using for any project in the same file/directory location. If you are unsure of the location of the folder, right click on the folder and select “get info.” Copy and paste the location of the file into “setwd()”. Save any images or exports from your R script into the same file location. For more information on setting your working directory, visit: Running RStudio and Setting Up Your Working Directory.

If you are using a Mac computer you can usually just copy and paste the location right into “setwd()”, but if you are using a PC you will need to change the direction of your slashes from back () to forward (/) slashes. Ensure your data folder is stored locally on your computer and not in the cloud.

```
getwd() # This will tell you where your working directory is currently set to
```

```
## [1] "/Users/justinedoll/Desktop/"
```

```
# If your current wd location is different than what you want, replace the wd
location by using setwd()

setwd("/Users/justinedoll/Desktop/iNat_Tutorial_Data/") # replace with your
       wd location

dir() # You can also check to see all the files in that specific location
```

```
## [1] "bombus_nosehill_2023.csv"
## [2] "Bombus_NoseHill_R_data.R"
```

Load in the data set that you would like to use. R is an object-based coding language so I will load my data into an object called “Bombus_nosehill” so it is easier to work with. Object names must not have spaces between words, otherwise it can confuse R. Use an underscore between words to avoid this. Any time you store a value or data into an object you will use what’s called a “left hand assignment” which is an arrow that looks like this “<-”. Often when you upload data you may need to manipulate it before you can do any analyses on it. Storing your data in an object makes this easier.

```
Bombus_nosehill <-  read.csv("bombus_nosehill_2023.csv")# replace with your file name

str(Bombus_nosehill) # You can check the structure of your data to see what type of data is contained within each column

colnames(Bombus_nosehill) # You can also check the column names to see all the columns in your data and to ensure that the data uploaded correctly. Normally, data from iNaturalist will upload nicely but it is always good to check
```

The only data we are really interested in is the name of the insect species and the plant that it was visiting. I will be doing the analysis with the columns “scientific_name” that contains the genus and species name for each insect, and “field.interaction..visited.flower.of” that contains the genus and species name for the plant that each insect was visiting.

These names are a bit long - you can rename them to anything you like (shown below). If you wanted to, you could also do this analysis by Class, Order, or Family of insect and the plant depending on your data set and the type of analysis you wish to do.

```
colnames(Bombus_nosehill)[colnames(Bombus_nosehill) == "scientific_name"] <- "insect_name"

colnames(Bombus_nosehill)[colnames(Bombus_nosehill) == "field.interaction..visited.flower.of"] <- "visited_flower"


head(Bombus_nosehill $insect_name) # Using "head" gives you the first few values of data
head(Bombus_nosehill $visited_flower)
```

3. Investigating pollinator-plant interactions with R

3.1. Exploring insect and plant species in your dataset

Here we will go through some basic ways to explore your data set. We will be using the pipe operator (%>%) from the dplyr package within tidyverse to create our matrices and organize our data (Wickham, et al., 2019).

```
# To install any packages that you do not have just delete the pound sign in
front of the code and run it

#install.packages("tidyverse")

# Load the packages

library(tidyverse)
```

First, we are going to create a simplified object with just plant and pollinator data.

```
plants_and_pollinators <- Bombus_nosehill %>% filter(!is.na(insect_name), !is
.na(visited_flower))# Remove any NA values and filter the data by columns of
interest. Store this in a new object.

plants_and_pollinators <- plants_and_pollinators %>% select(insect_name,visit
ed_flower)# Selects just your columns of interest
```

We can view the names of all the insects and plants within the data set and count how many unique species were observed.

```
plants_and_pollinators$insect_name # Print a list of all the insects in your
dataset

plants_and_pollinators$visited_flower # Print a list of all the plants
visited by insects in your dataset


n_distinct(plants_and_pollinators$insect_name)

n_distinct(plants_and_pollinators$visited_flower)

# Number of different insect and plant species included in your dataset


unique(plants_and_pollinators$insect_name)

unique(plants_and_pollinators$visited_flower)

# Unique names of all the insects and plants in your dataset
```

3.2. Exploring plant-pollinator interactions

We can also look at the interactions between the insect and the plant species.

```
Interactions <- plants_and_pollinators %>% group_by(insect_name,visited_flower) %>% summarise(n = n())# Summarizes the number of times each insect visited each type of flower in the plants_and_pollinators data frame. We will use the "Interactions" object again in part 4

print(Interactions) # Print table
```

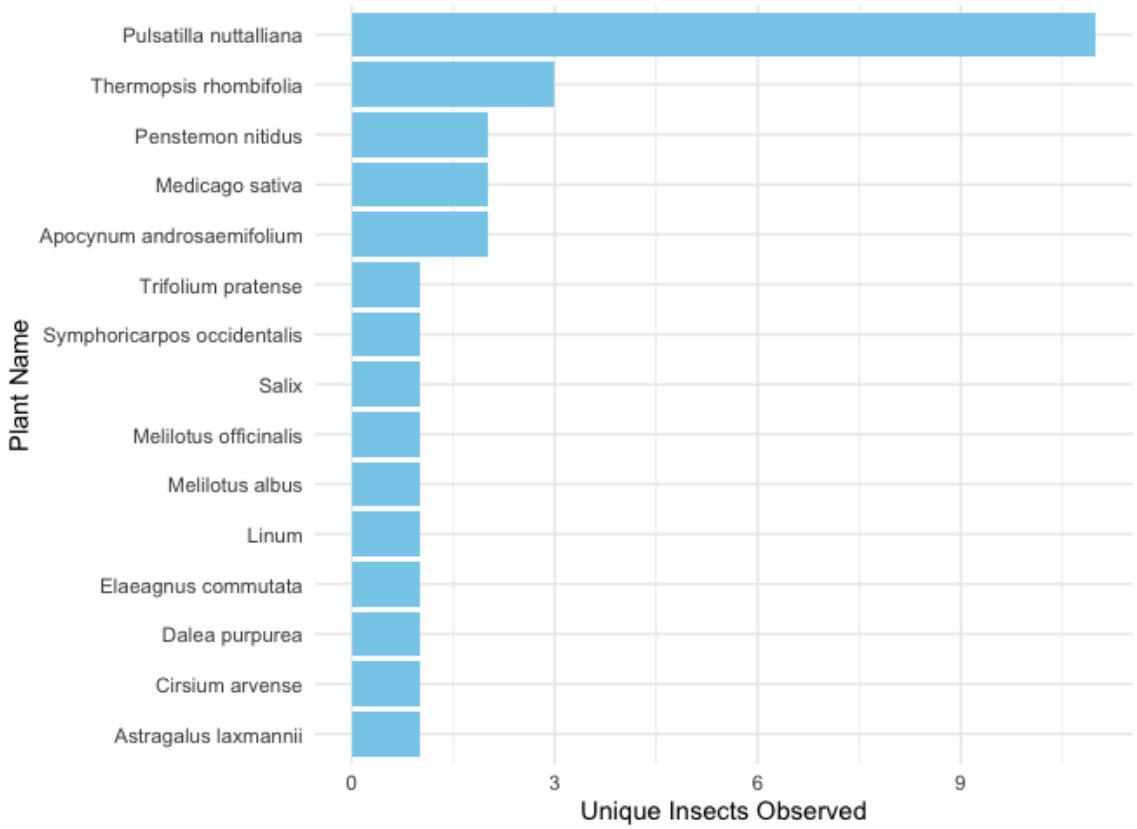
	insect_name	visited_flower	n
1	Bombus	Melilotus albus	2
2	Bombus	Melilotus officinalis	1
3	Bombus	Pulsatilla nuttalliana	3
4	Bombus centralis	Astragalus laxmannii	1
5	Bombus centralis	Medicago sativa	1

Here are the first five rows of the table for the interactions between Bombus species and plant species at Nose Hill in 2023 with a count of how many times that interaction occurs.

We can use the interactions matrix table that we just made to visualize the data. First, we can create a visualization for the most visited plant species (i.e. the plants who received the most visits from different insect species).

```
unique_insect_count <- Interactions %>%
  group_by(visited_flower) %>%
  summarise(unique_insect_count = n_distinct(insect_name)) # Calculate the unique_insect_count for each unique "visited_flower"

sorted_flowers <- unique_insect_count %>%
  arrange(desc(unique_insect_count)) # Sort the data in descending order by unique_insect_count

# Create a bar plot to visualize the data
most_visited_plants <- ggplot(sorted_flowers, aes(x = reorder(visited_flower, unique_insect_count), y = unique_insect_count)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  coord_flip() +  # Flip the coordinates for horizontal bars
```

```
  labs(x = "Plant Name", y = "Unique Insects Observed") +
  theme_minimal() # Use a minimal theme


print(most_visited_plants)
```

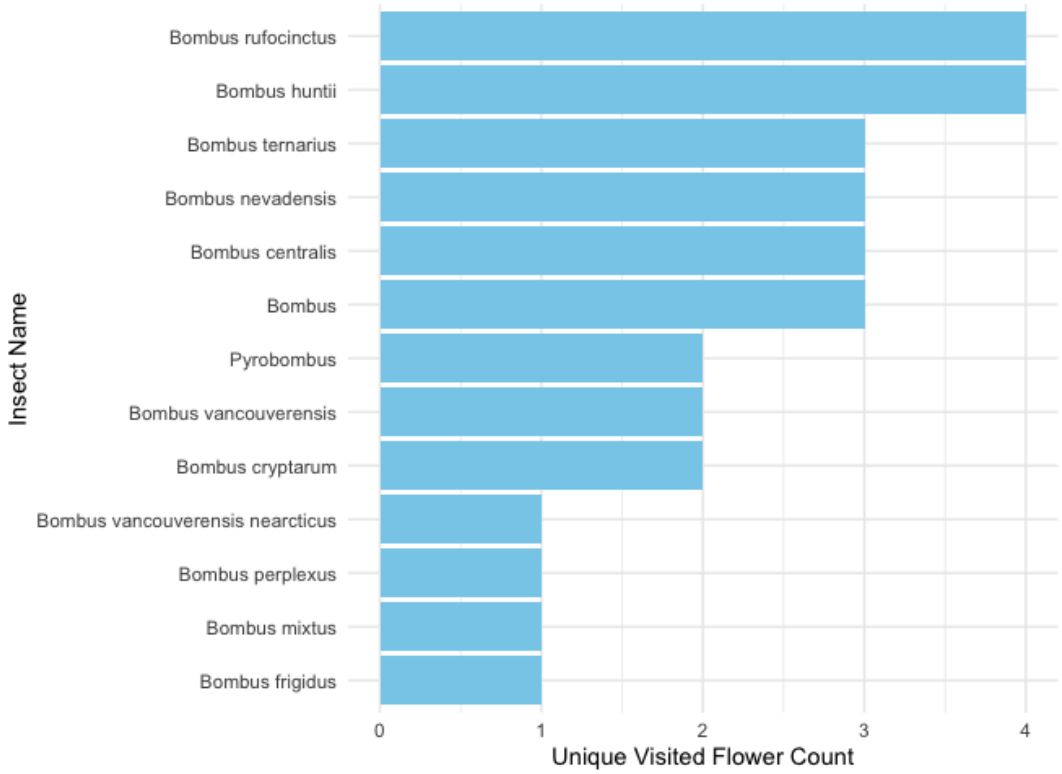


These are the plant species that were most visited by *Bombus* at Nose Hill in 2023. You can see that there were 11 different *Bombus* species that visited Prairie Crocuses (*Pulsatilla nuttalliana*).

We can also create a visualization for the insects that visit the most plant species (i.e. the insects who visited the most different species of plants). This can be helpful for determining generalist pollinator species in larger datasets.

```
unique_visited_flower_count <- Interactions %>%
  summarise(unique_visited_flower_count = n_distinct(visited_flower))
# Calculate the unique_visited_flower_count for each unique "insect_name"


sorted_insects <- unique_visited_flower_count %>%
```

```
  arrange(desc(unique_visited_flower_count)) # Sort the data in descending
order by unique_visited_flower_count


# Create a bar plot to visualize the data

generalist_pollinators <- ggplot(sorted_insects, aes(x = reorder(insect_name,
unique_visited_flower_count), y = unique_visited_flower_count)) +

  geom_bar(stat = "identity", fill = "skyblue") +

  coord_flip() +  # Flip the coordinates for horizontal bars

  labs(x = "Insect Name", y = "Unique Visited Flower Count") +

  theme_minimal() # Use a minimal theme


print(generalist_pollinators)
```

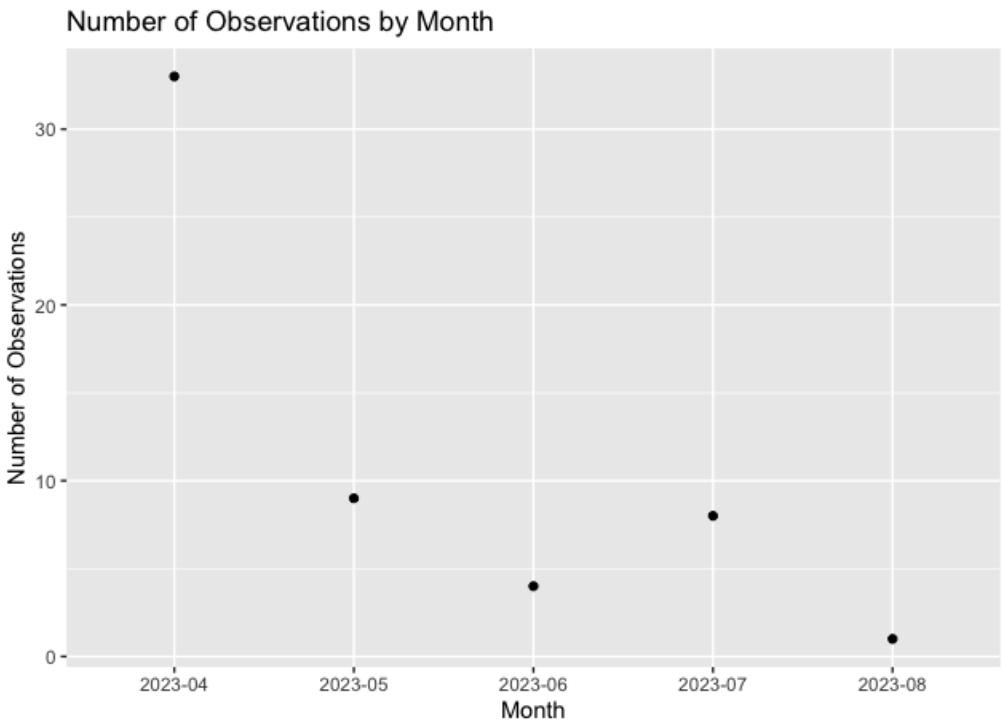


These are the insect species that visited the most plant species at Nose Hill in 2023. You can see that the Red-Belted Bumble Bee (*Bombus rufocinctus*) and Hunt's Bumble Bee (*Bombus huntii*) both visited four different plant species.

3.3. Graphing observations through time

You can create a visualization to see the observations made of *Bombus* at Nose Hill in 2023 through time. Note that this will not represent the actual seasonal distribution of *Bombus* at Nose Hill, it will show when observations of *Bombus* were made over the season.

```
seasonality <- Bombus_nosehill #create a new object

seasonality$observed_on <- as.Date(seasonality$observed_on) # Convert
observed_on to Date format

seasonality$month <- format(seasonality$observed_on, "%Y-%m") # Extract month
from observed date

obs_count_month <- seasonality %>% # Count observations by month
  group_by(month) %>%
  summarise(count = n())

ggplot(obs_count_month, aes(x = month, y = count)) + # Create plot
  geom_line() +
  geom_point() +
  labs(x = "Month", y = "Number of Observations", title = "Number of
Observations by Month")
```

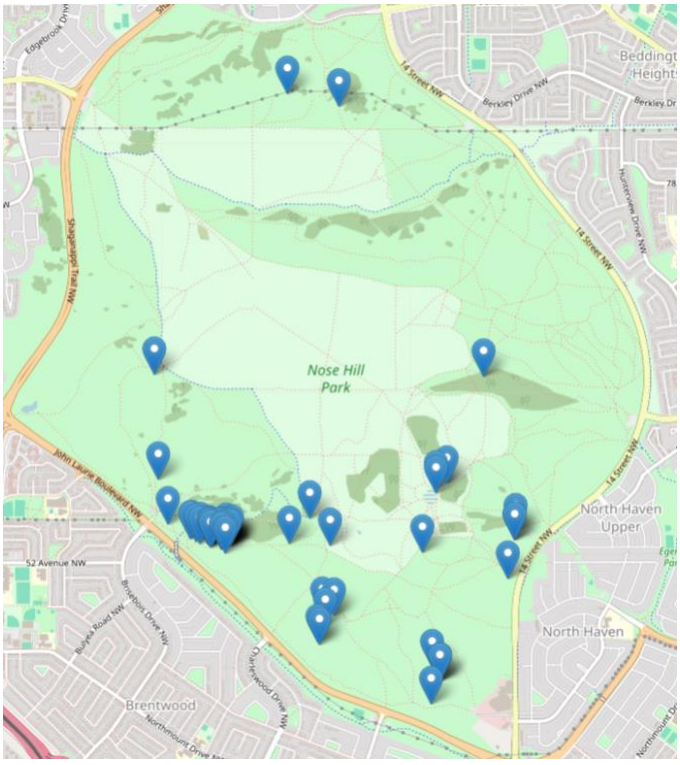


The highest number of Bombus observations at Nose Hill were made at the end of April 2023. This likely coincides with the City Nature Challenge that runs at the end of April every year.

3.4. Mapping observations in space

You can create an interactive map of the observations with the leaflet package.

```
#install.packages("leaflet") # delete pound sign to install package
library(leaflet)


data_filtered <- Bombus_nosehill %>%
  filter(!is.na(latitude) & !is.na(longitude)) # Filter out observations with missing coordinates
map <- leaflet(data = data_filtered) %>% addTiles() %>%
  addMarkers(~longitude, ~latitude, popup = ~place_guess) # Create leaflet map
map # Display the map
```



You can zoom in and out on the map to get a more detailed view. You will see that most observations at Nose Hill are along marked trails.

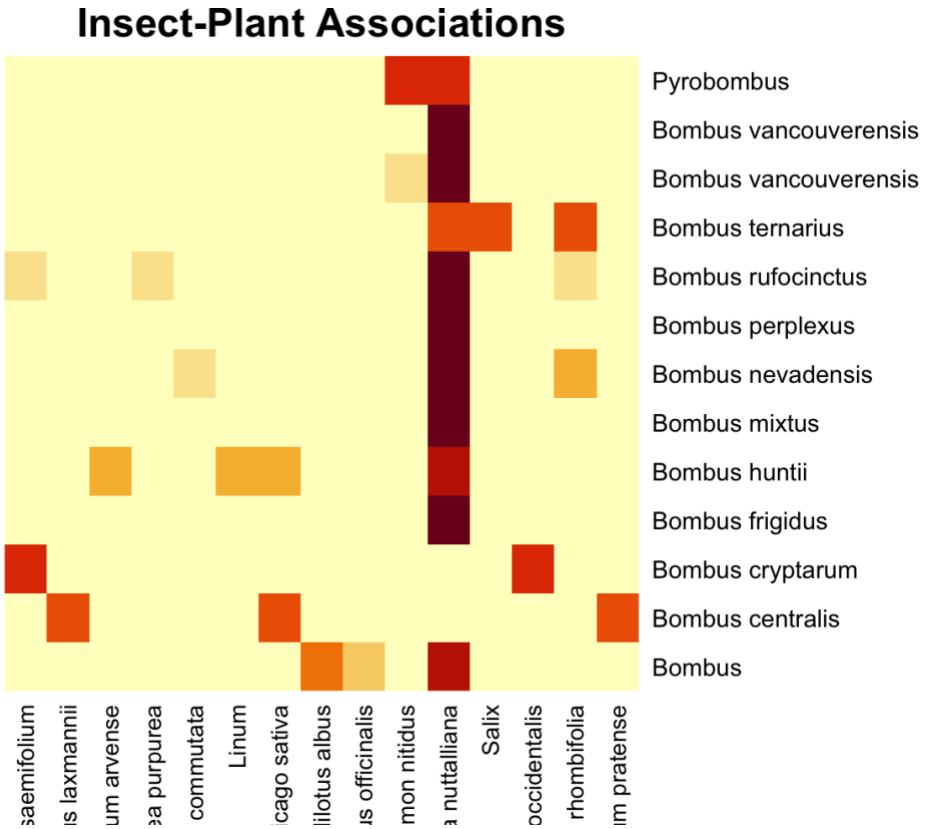
4. Visualizing interactions: creating network matrices with R

4.1. Heat Map Visualization

You can create a simple heat map with your interaction data. You will need to organize your data into a matrix, which you will also need for a bipartite analysis. The heat map creates a relatively low-resolution visualization of plant-pollinator interactions; this works okay for smaller data sets but may not be suitable for large data sets. The scale from yellow to red indicates the relative number of interactions of a specific plant and pollinator interaction from low to high, respectively. The darkest colour signifies the most interactions.

```
# Select your target data to create a species-by-plant matrix
subdata <- data.frame(Insect_species= Bombus_nosehill$insect_name,
                      Plant_species= Bombus_nosehill$visited_flower)
mat <- xtabs(~Insect_species+Plant_species, subdata) # Turn your subdata into
a matrix
heatmap(mat, Rowv=NA, Colv=NA, main="Insect-Plant Associations") # Create a
       heat map
```

Here is what the heat map looks like for Bombus interactions at Nose Hill in 2023.



4.2. Bipartite Analysis

Next, we need to install the packages that you will use to create your plant-pollinator network.

```
# To install any packages that you do not have just delete the pound sign in
front of the code and run it

#install.packages("bipartite")

# Load the package

library(bipartite)
```

We need to organize and select the Bombus_nosehill data so that we can use bipartite. Like creating the heat map, your data will need to be arranged into a matrix.

```
plants_and_pollinators <- Bombus_nosehill %>% filter(!is.na(insect_name), !is
.na(visited_flower))

# Filter the data by your columns of interest. Store this in a new object,
        I have named mine "plants_and_pollinators."

# Remove any NA values so you hopefully won't run into problems down the road


plants_and_pollinators <- plants_and_pollinators %>% select(insect_name,visit
ed_flower)

# Selects just your columns of interest and stores it in the
plants_and_pollinators object


Interactions<- plants_and_pollinators %>% group_by(insect_name,visited_flower
) %>% summarise(n = n())

# This code summarizes the number of times each insect visited each type of
flower in the plants_and_pollinators data frame.
```

This sets up your data to be used in bipartite.

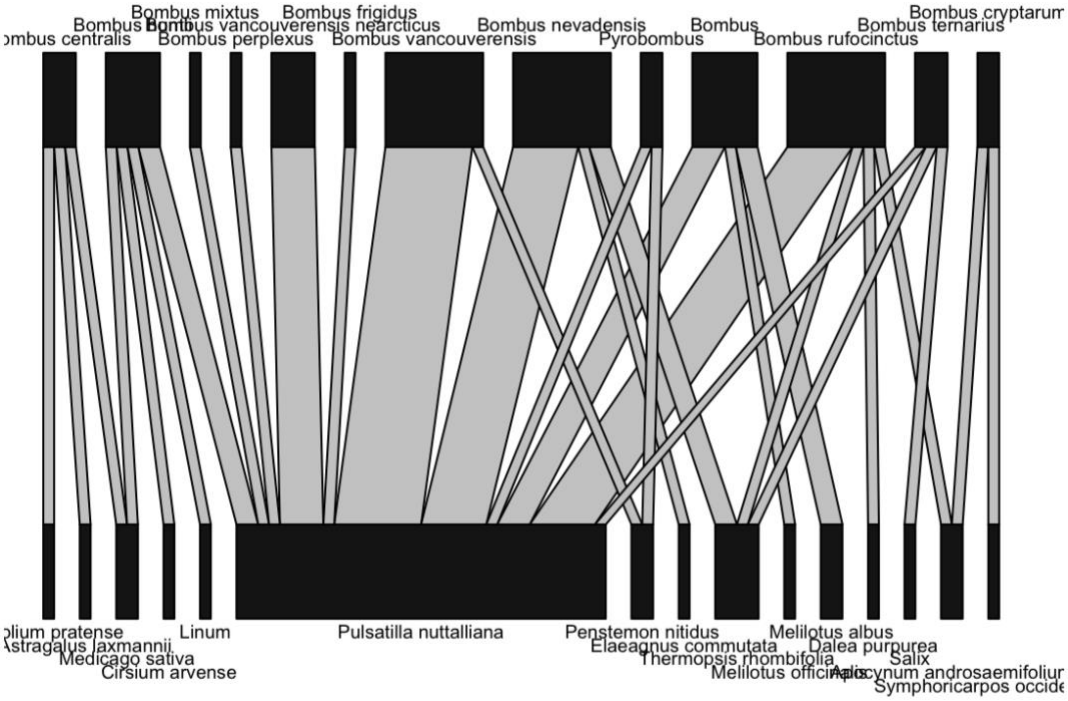
Note: We will use the "Interactions" data again when we create a bipartiteD3 network.

The last step is to reshape the Interaction data into a matrix so we can create a matrix and run the species and network analyses with bipartite.

```
# Reshape the data using pivot_wider() function

interactmatrix <- Interactions %>% pivot_wider(names_from = insect_name, valu
es_from = n, values_fill = 0)


# Convert the "visited_flower" column to row names using column_to_rownames()
function
```

```
interactmatrix <- column_to_rownames(interactmatrix, var = "visited_flower")
plotweb(interactmatrix) #create your plot web
```

Here is what the bipartite interaction matrix looks like for Bombus interactions at Nose Hill in 2023. A line connecting an insect to a plant means that the insect interacts with that plant. The thickness of the line relates to the number of times that insect species was observed interacting with that plant, i.e. the thicker the line, the more interactions have been observed.



4.3. Additional analysis with Bipartite

You can run analyses on your data at both species and network level with bipartite. The species analysis can tell you about degree, species strength, species specificity, closeness, Fisher's alpha index, Shannon diversity of interactions of that species, etc. (Dormann 2020).

```
species_analysis<- specieslevel(interactmatrix) #can get metrics for each of the species
head(species_analysis)
```

Table 1. Species-level metrics, definitions, and how to interpret them (Dormann 2009).

Metric	Definition	Interpretation
Degree	The number of interactions a species has in the network.	A higher degree indicates a species that interacts with many other species, suggesting it is potentially more important for the structure of the network.
Species Strength	The sum of the dependencies of the species that interact with a given species.	A measure of the overall importance of a species within the network. Higher species strength indicates greater influence on the network.
Species Specificity	The extent to which a species interacts with a unique set of partners.	High specificity suggests that the species has specialized interactions, which can be crucial for understanding mutual dependencies in the network.
Closeness	A measure of how close a species is to all other species in the network.	High closeness centrality means a species can quickly interact with all others in the network, indicating a potentially critical role in facilitating interactions.
Fisher's Alpha Index	A diversity index used to quantify the diversity within a species assemblage.	Higher values indicate greater species diversity.
Shannon Diversity	A measure of the diversity of interactions for a species, based on the Shannon entropy.	Higher values indicate a more even distribution of interactions among different partners.

The network analysis can tell you about connectance, web asymmetry, links per species, nestedness, weighted nestedness, weighted NODF, interaction strength asymmetry, specialization asymmetry, linkage density, Fisher alpha, interaction evenness, Shannon diversity, etc. (Dormann 2020). There are also many other analyses included in the bipartite package to choose from.

```
network_analysis <- networklevel(interactmatrix) #can get metrics for the who
le network
head(network_analysis)
```

```
?bipartite #will tell you more about the bipartite package and it's functions
```

Table 2. Network-level metrics, definitions, and how to interpret them (Dormann 2009).

Metric	Definition	Interpretation
Connectance	The proportion of possible interactions that are realized in the network.	Higher connectance suggests a more interconnected network.
Web Asymmetry	The difference in the number of species between the two levels of a bipartite network (e.g., plants and pollinators).	Indicates how balanced the network is. High asymmetry might suggest vulnerability to species loss.
Links per Species	The average number of interactions per species.	Provides insight into the overall connectivity of the network
Nestedness	A measure of how much the interactions of specialist species are subsets of those of more generalist species.	Higher nestedness indicates a more stable and robust network structure.
Weighted Nestedness	Like nestedness but accounts for the frequency of interactions.	Provides a more nuanced view of network stability.
Weighted NODF	Weighted NODF (Nestedness metric based on Overlap and Decreasing Fill) is a metric for nestedness that considers both the presence and the weight of interactions.	Higher values indicate greater nestedness when considering interaction weights.
Interaction Strength Asymmetry	The degree to which interaction strengths are balanced between species pairs.	High asymmetry suggests unequal dependency among species, which can affect network stability.
Specialization Asymmetry	The degree to which specialization is balanced between species pairs.	Indicates whether one species is much more specialized than its interaction partners.
Linkage Density	The average number of links per species.	Reflects how densely the network is connected.
Fisher Alpha	As above, but applied to the entire network.	Indicates overall diversity within the network.

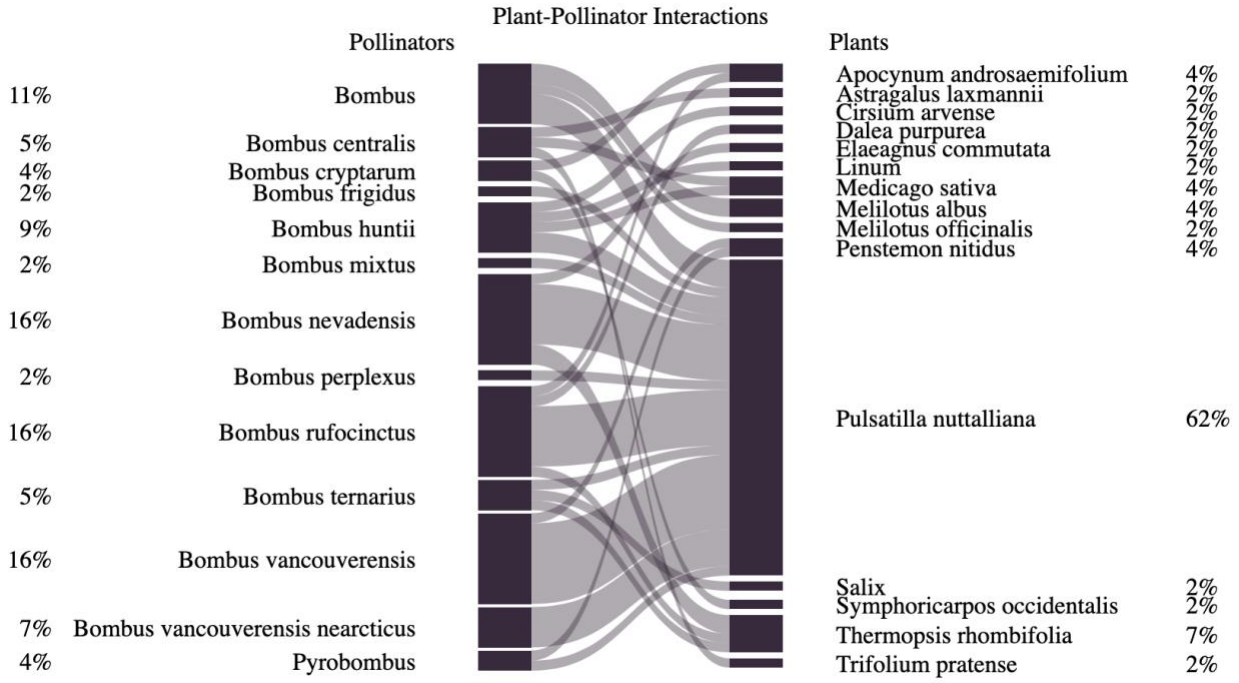
Interaction Evenness	A measure of how evenly interactions are distributed among species.	Higher values indicate a more even distribution, suggesting no single species dominates the interactions.
Shannon Diversity	A measure of the diversity of interactions within the entire network, based on Shannon entropy.	Higher values indicate a more diverse and evenly distributed set of interactions.

4.4. Data Visualization with bipartiteD3

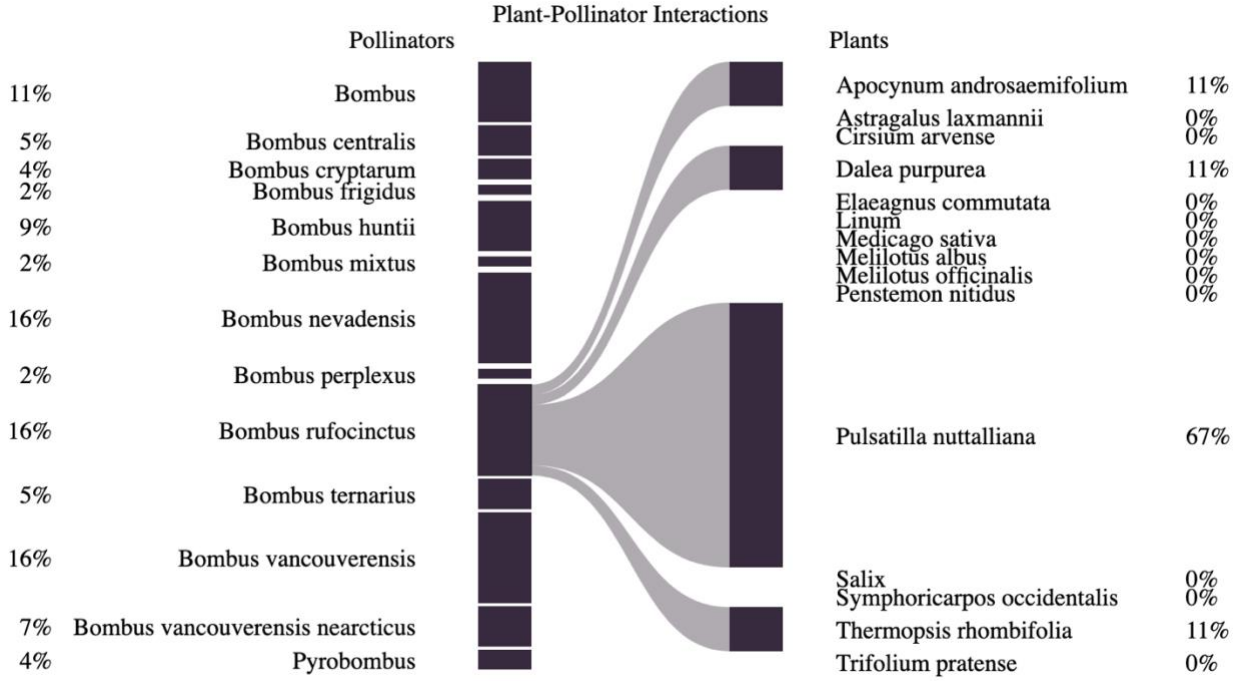
BipartiteD3 that will create a more viewer-friendly visualization. Use the data contained in the object we made earlier, named "Interactions". The Interactions data is a summary of how many times a certain pollinator visited a certain plant.

```
#install.packages("bipartiteD3", dependencies = TRUE) # remove hashtag and
run code to install package and dependencies
library(bipartiteD3) # load package

pretty_network <- Interactions # use Interactions matrix from earlier

pretty_network %>% # Select data, then

  bipartite_D3(. ,  # Opens a webpage with your bipartiteD3 network
               SiteNames = "Plant-Pollinator Interactions", # Creates your
main title
               PrimaryLab = "Pollinators", # Creates left-hand title
               SecondaryLab = "Plants") # Creates right-hand title
```

Here is what the bipartiteD3 interaction matrix looks like for *Bombus* interactions at Nose Hill in 2023. You may need to click on zoom to see the full image.



If you hover your mouse over a specific species the interactive matrix will shift to show you the interactions of that species. This matrix highlights the interactions of *Bombus rufocinctus*.



4.5. Species specific filter bipartiteD3 network code

If your dataset contains many different plant and insect species interactions, you can create a bipartite matrix for the whole network or you can filter the dataset to create a visualization for a species of interest. For example, if your dataset includes all the plant-pollinator interactions from Nose Hill in 2023 you may have a very large network. If you want to focus in on an insect or plant of interest, you can create smaller species-specific networks.

Let's say within the larger dataset I am interested in seeing how many different plants were visited by Red Belted Bumble Bees, (*Bombus rufocinctus*).

```
d3Brufocinctus <- Interactions %>%
  filter(grepl("Bombus rufocinctus",insect_name)) # filter for Bombus rufocin
ctus


bipartite_D3(d3Brufocinctus,
             SiteNames = "Bombus rufocinctus Interactions", # Main title
                PrimaryLab = "Bombus rufocinctus", # Creates left-hand title
                SecondaryLab = "Plants") # Creates right-hand title
```

You could also do this for any plant species you are interested. Let's see how many different pollinators prairie crocus (*Pulsatilla nuttalliana*) supports.

```
d3Pnuttalliana <- Interactions %>%
  filter(grepl("Pulsatilla nuttalliana",visited_flower)) # filter for Pulsati
lla nuttalliana


bipartite_D3(d3Pnuttalliana,
             SiteNames = " Pulsatilla nuttalliana Interactions", # Main title
                PrimaryLab = "Pollinators", # Creates left-hand title
                SecondaryLab = "Pulsatilla nuttalliana") # Creates right-hand
title
```

5. References

Dormann, C.F. (2020). Using bipartite to describe and plot two-mode networks in R. R package version, 4, 1-28.

Dormann, C. F., Fründ, J., Blüthgen, N., & Gruber, B. (2009). Indices, graphs and null models: analyzing bipartite ecological networks.

Newman, M. (2010). Networks: An Introduction, 1st edn. Oxford, online edn, Oxford Academic. https://doi.org/10.1093/acprof:oso/9780199206650.001.0001

R Core Team (2022). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Wickham, H., et al., (2019). Welcome to the Tidyverse. Journal of Open Source Software, 4(43), 1686. https://doi.org/10.21105/joss.01686

6. Appendix A: Where to Explore Calgary Biodiversity

Certified as a Bee City by Bee City Canada in 2018, Calgary is making efforts to support its pollinator habitats. Conservation efforts by The City of Calgary have led to the creation of many naturalized spaces across the city. To learn more about Calgary's commitment to pollinators visit The City of Calgary **Bee a polli-neighbour** page.

There are many areas in Calgary to explore biodiversity. Below is a list of parks and naturalized areas that you may like to explore.

Calgary City Parks

Below is a list of natural/naturalized parks in the city. For more information about each park and for a more comprehensive list of all the parks in Calgary, visit the parks page on The City of Calgary website and click on your park of interest.

12 Mile Coulee
Baker Park
Beaverdam Flats
Bowmont Park
Bowness Park
Bridlewood Wetlands
Carburn Park
Central Memorial Park
Confederation Park
Dale Hodges Park
Edgemont Ravine
Edworthy Park
Elliston Park
Fish Creek Provincial Park
Griffith Woods
Inglewood Bird Sanctuary
Nose Hill Park
North Glenmore Park
Pearce Estate Park
Prince's Island Park
Ralph Klein Park
Reader Rock Garden
Rotary Park
Sandy Beach
South Glenmore Park
St. Patrick's Island Park
Sue Higgins Park
Tom Campbell's Hill
Weaselhead Flats
West Nose Creek

Naturalization Projects

Calgary has several pilot naturalization projects throughout the city. The goal of these areas is to restore natural habitats and plant native wildflowers to foster biodiversity.

16 Avenue N.E.
Bridgeland
Memorial Drive
Coach Hill
Panorama Hills Storm Pond

Canyon Meadows Bee Boulevard

The Canyon Meadows Bee Boulevard was created in 2017 as a joint effort between The City of Calgary Roads and Parks departments, the University of Calgary, Mount Royal University, and the David Suzuki Foundation.

This naturalized greenspace requires little maintenance. The chosen native plants are attractive to pollinators and can withstand harsh Calgary weather. Pollinator habitat, such as a sand bed for ground-nesting bees and houses for bumblebees, have also been installed.

University of Calgary Community Garden

The University of Calgary is a dedicated **Bee Campus** that supports initiatives to improve and maintain pollinator habitats and populations. The community garden plots are one important part of the University's Bee Campus initiative.