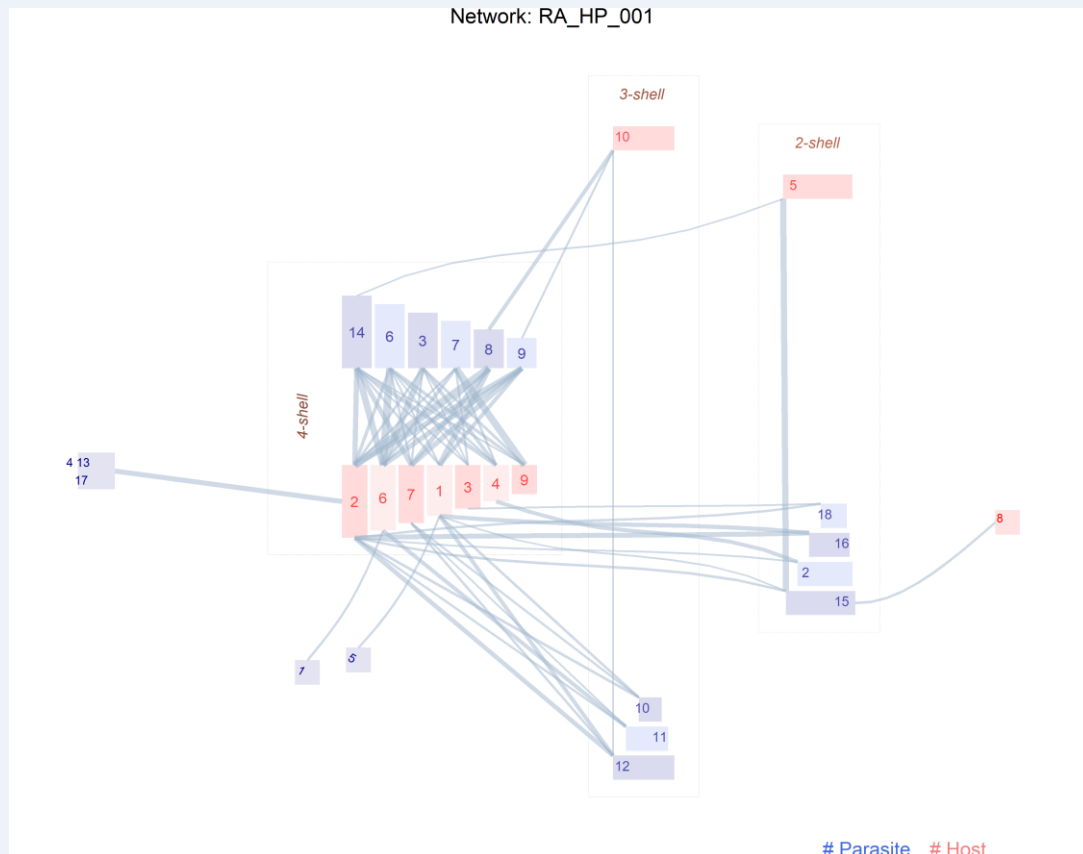


BipartGraph



User Guide

R2.0 May 2025

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: All copies or substantial portions of the Software must include the above copyright and permission notice.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES, OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT, OR OTHERWISE, ARISING FROM, OUT OF, OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Contents

1	Introduction.....	4
2	Install BipartGraph	5
2.1	Fast-track: Install and use BipartGraph with Docker	5
2.2	Requirements for native installation	6
2.3	First step: Download the software	7
2.4	Second step: install R.....	8
2.5	Third step: setting up the application.....	10
2.6	Hacks for installation on Linux.....	11
2.7	Running <i>BipartGraph</i>	12
3	Data management	13
4	The ziggurat plot.....	15
4.1	Explore the interactive ziggurat	18
4.2	Modifying the look and feel of the ziggurat plot	19
4.3	Printable ziggurat	26
5	The bipartite plot.....	29
6	The matrix plot	32
7	The polar plot.....	34
	Useful hacks	36
	References	36

1 Introduction

BipartGraph is an interactive application to visualize bipartite ecological networks using the *k-core* decomposition method [1,2].

The most common visualization of this kind of network is the bipartite graph. It underlines the existence of two communities, but links among species are difficult to distinguish even with fewer nodes. When the size of the network grows, the graph becomes a *hairball*, an expression common in visualization for cluttered plots (Fig. 1).

Identifying the central core requires graphical intuition, and guessing the centrality of species is almost impossible. As a result, it is uncommon to find plots of networks with more than a dozen species per guild in the research literature, although typical mutualistic networks are larger. This problem is not due to the user's lack of skills; it is a limitation of the graph itself.

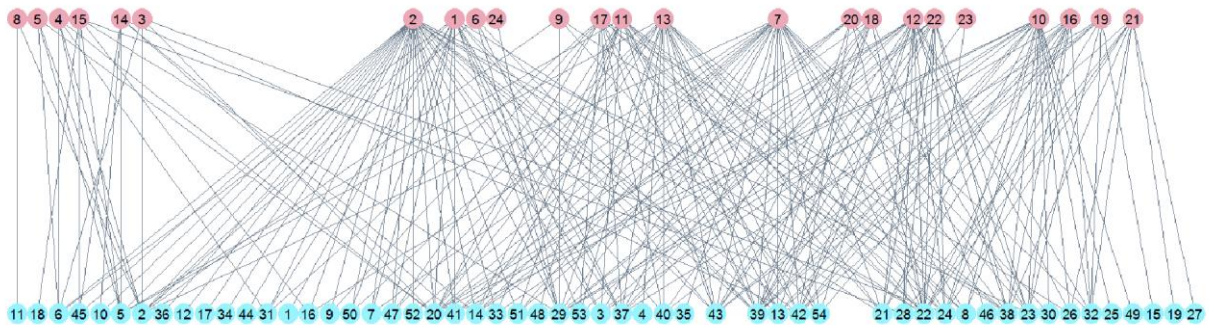


Figure 1: Bipartite graph of a mutualistic community with 78 species [3].

The *k-core* decomposition offers a natural way to group species with similar connectivity, enabling a more powerful spatial distribution.

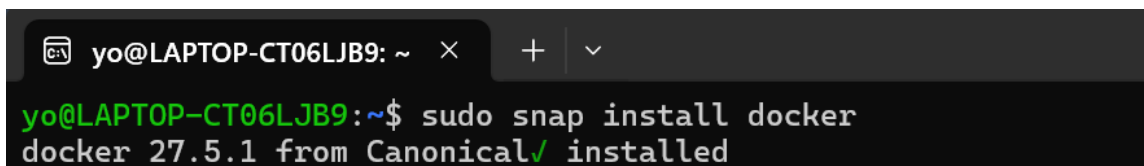
You do not need to be an expert in graph analysis to enjoy this program; install it and start playing with the different options.

2 Install BipartGraph

BipartGraph is an interactive application developed in R. Whether you are familiar with R or have never worked with it or any other programming language, this document guides you through the installation procedure step by step.

2.1 Fast-track: Install and use BipartGraph with Docker

Installing a Docker image of BipartGraph is the easiest way to run the application. First, you need to install Docker. The provider documentation is excellent: <https://docs.docker.com/engine/install/ubuntu/> (or any other Linux distro). If you are using Windows, you can use the Windows Subsystem for Linux (WSL) by following the instructions at <https://learn.microsoft.com/en-us/windows/wsl/install>. From the command line, install Docker by typing: `sudo snap install docker`

A terminal window with a dark background. The prompt is 'yo@LAPTOP-CT06LJB9: ~'. The command 'sudo snap install docker' has been executed, and the output is 'docker 27.5.1 from Canonical✓ installed'.

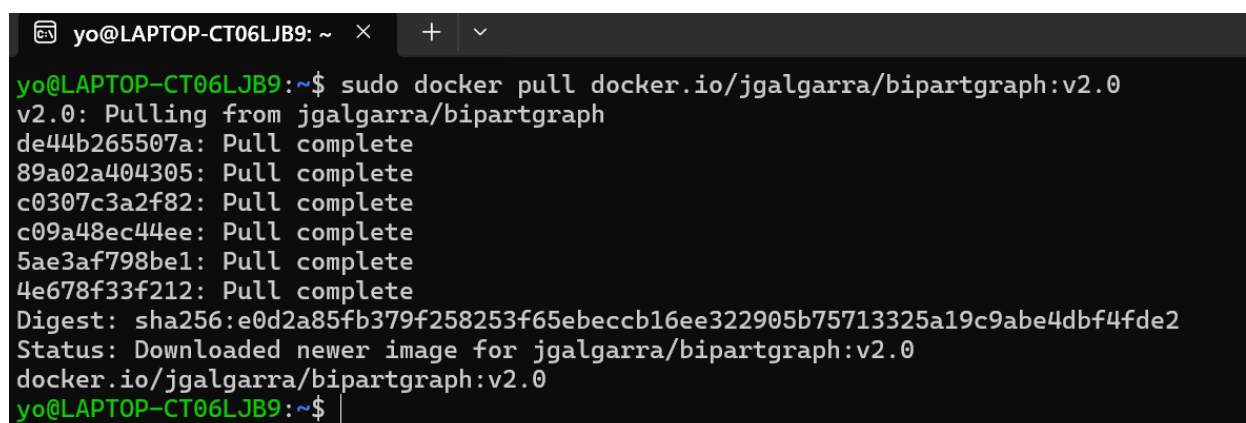
```
yo@LAPTOP-CT06LJB9: ~$ sudo snap install docker
docker 27.5.1 from Canonical✓ installed
```

If you work with Mac, then install Docker Desktop:

<https://docs.docker.com/desktop/setup/install/mac-install/>

Once you have installed Docker, download the BipartGraph image from your command line, no matter if you are using Windows, Linux, or Mac:

```
sudo Docker pull docker.io/jgalgarra/bipartgraph:v2.0
```

A terminal window with a dark background. The prompt is 'yo@LAPTOP-CT06LJB9: ~'. The command 'sudo docker pull docker.io/jgalgarra/bipartgraph:v2.0' has been executed. The output shows the image being pulled from the registry, with various layer hashes and their status (Pull complete). The digest is also shown, followed by a status message indicating a newer image was downloaded.

```
yo@LAPTOP-CT06LJB9: ~$ sudo docker pull docker.io/jgalgarra/bipartgraph:v2.0
v2.0: Pulling from jgalgarra/bipartgraph
de44b265507a: Pull complete
89a02a404305: Pull complete
c0307c3a2f82: Pull complete
c09a48ec44ee: Pull complete
5ae3af798be1: Pull complete
4e678f33f212: Pull complete
Digest: sha256:e0d2a85fb379f258253f65ebeccb16ee322905b75713325a19c9abe4dbf4fde2
Status: Downloaded newer image for jgalgarra/bipartgraph:v2.0
docker.io/jgalgarra/bipartgraph:v2.0
yo@LAPTOP-CT06LJB9: ~$
```

BipartGraph is now ready to run. The following command launches the application:

```
yo@LAPTOP-CT06LJB9: ~ × + v
yo@LAPTOP-CT06LJB9:~$ sudo docker run -p 127.0.0.1:8080:8080 docker.io/jgalgarra/bipartgraph:v2.0 &
[1] 2999
yo@LAPTOP-CT06LJB9:~$ |
```

You can avoid typing it each time you launch the application, saving it as a shell script file; you may call it `launch_bipartgraph.sh`

```
sudo docker run -p 127.0.0.1:8080:8080 docker.io/jgalgarra/bipartgraph:v2.0
# Uncomment to open your browser automatically
# firefox 127.0.0.1
```

If you run Docker on your WSL subsystem and want to connect from your Windows browser, or if you installed the Docker image on a server machine and want to connect from a client, then launch your browser and type <http://127.0.0.1:8080> to find BipartGraph up and running.

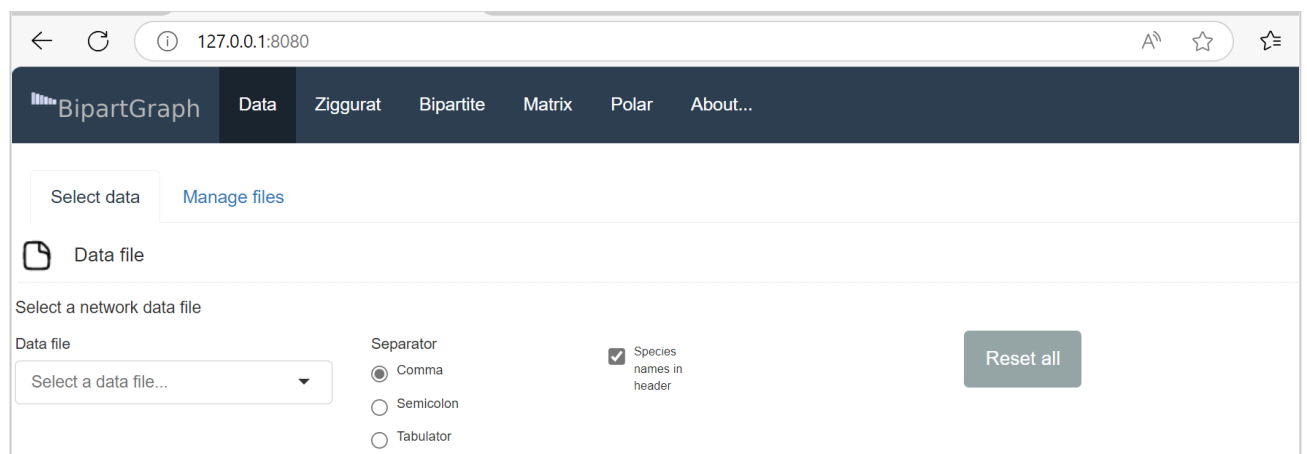


Figure 2: *BipartGraph* user interface

If you are working with Mac, then copy this code into the `launch_bipartgraph.sh` file:

```
docker run -p 127.0.0.1:8080:8080 docker.io/jgalgarra/bipartgraph:v2.0
open -a "Google Chrome" http://127.0.0.1:8080
```

Skip the following sections and go straight to **¡Error! No se encuentra el origen d e la referencia.**

2.2 Requirements for native installation

There are only two requirements to install BipartGraph: a working R environment and any web browser (Firefox, Chrome, or *Safari*, if possible). We recommend at least

8 GB of RAM in your machine. R is an open-source project; you will find the download and installation instructions for different operating systems at <https://cran.r-project.org/>

2.3 First step: Download the software

Bipartgraph is available as open-source software under the MIT license on GitHub. Open your browser and type : <https://github.com/jgalgarra/bipartgraph>

Notice the green button, **Clone or download**, that provides two ways to get the software.

(Fig. 3).

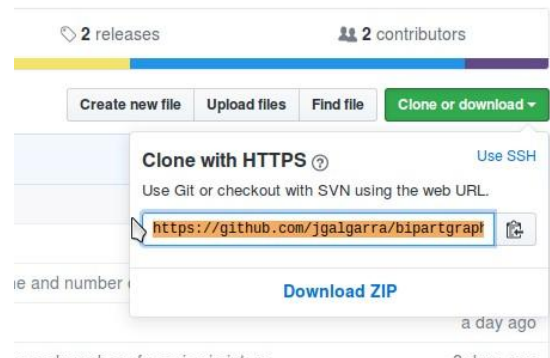


Figure 3: BipartGraph repository at github.com

The first one uses git, the control version tool (<https://git-scm.com/>). If you know how git works, clone the jgalgarra/bipartgraph repository.

```
> git clone https://github.com/jgalgarra/bipartgraph
```

A new directory called bipartgraph appears. We recommend this method because you can update your local software anytime with the `git pull` command.

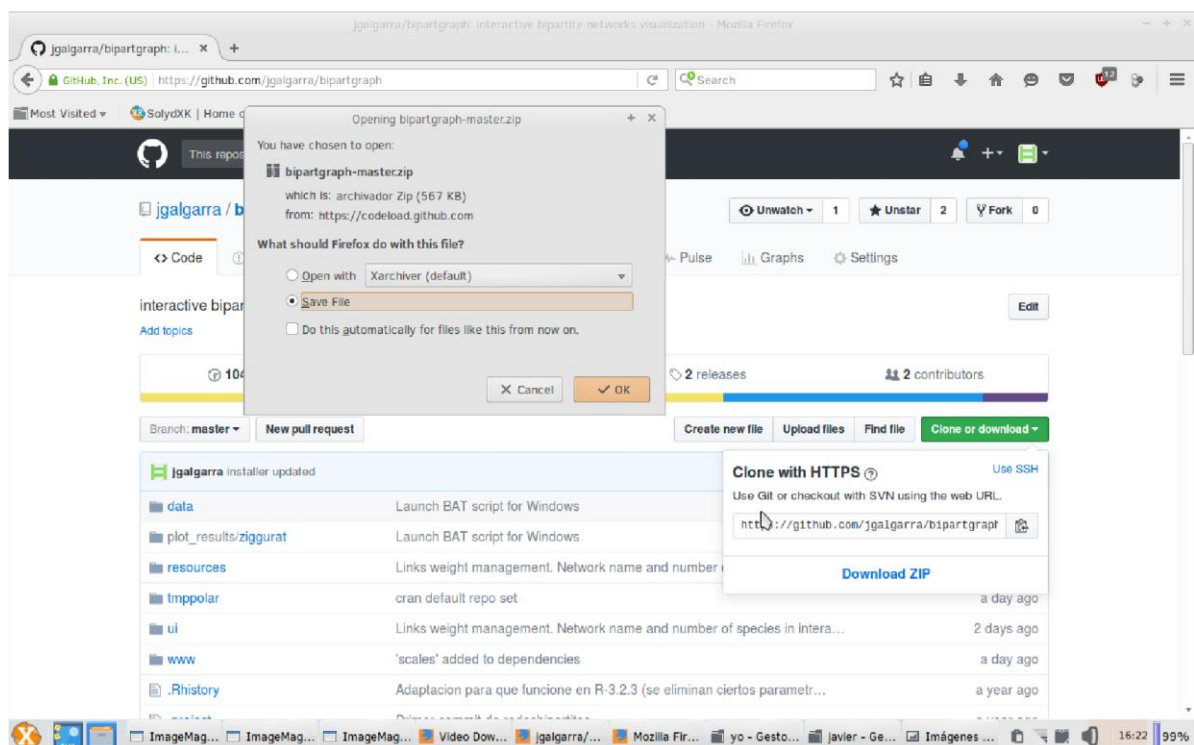


Figure 4: Downloading the BipartGraph ZIP file

What if you are not a software developer and do not want to install `git`? Download the ZIP file instead (fig. 4), uncompress it, and you will see a new directory called `bipartgraph-master` with the same contents as the original repository.

2.4 Second step: install R

You need R 4.3.0 or later. If you have never installed R, the last available release from the download page will be fine.

A) Windows

Figure 5 shows the installer program. Please pay attention, the installation program will suggest a path. Please do not change it unless you know what you are doing. For instance, in Windows systems, volume `C:` is usually restricted to users with administrator privileges. Let the R installer choose the best location for your user. Please write the installation folder's name somewhere (a notepad is a good choice) because you need it for the next step.

Once you install R, check that it is ready to run. What does this mean? If you type `R` in the command line, the application starts. If you are a Windows user, it is probable that you get this disappointing message: `R is not recognized as an internal or external command`.

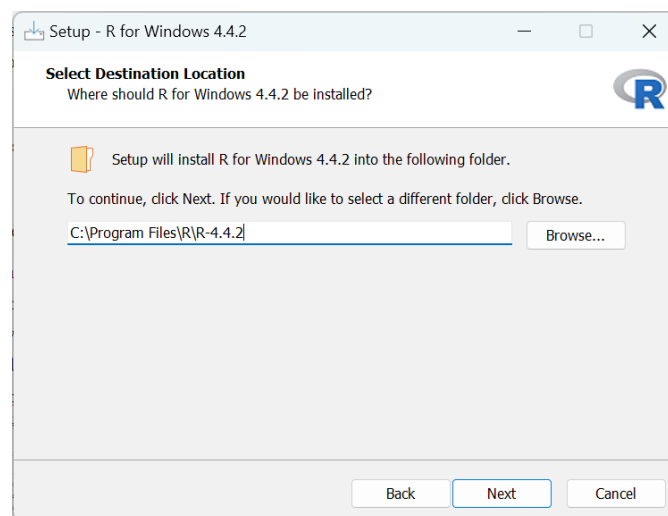


Figure 5: Installer program for Windows. The installation folder's name is essential when setting the PATH variable in Windows.

There are two methods to overcome this problem in Windows:

1. **Simple and safe.** Go to the directory where you installed `bipartgraph`. Select the file called `global`, right-click it, select “Open with” and then “Choose another app” (the option name may be slightly different; we captured this screen in Windows 11). Navigate to the installation folder that you wrote in your Notepad. Inside this folder, click on the `bin` folder and then select `Rscript`.

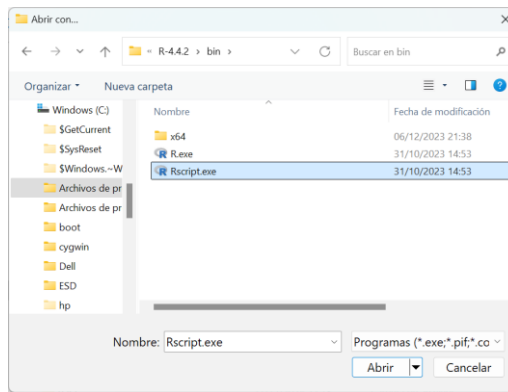


Figure 6: Choose Rscript.exe to run the .R files

After you perform this action, the script's name displays the R icon.

2. Include the R bin folder in the path variable. In the example of Figure 5, it is `C:\Program Files\R\R-4.4.2\bin`. You must add `\bin` to the name of the installation folder you copied in Notepad. To modify the value of the variable `Path` in Windows, we suggest this helpful page; the procedure depends on the release

<http://www.dowdandassociates.com/blog/content/howto-set-an-environmentvariable-in-windows-gui/>

B) UX-like systems (including Linux and MacOS)

For Debian-based and Fedora distributions, there is a detailed set of instructions if you find problems installing R 4.3 or later. Check section 2.6 and then come back.

Figure 7 is a screen capture of your terminal after launching R.

```
yo@LAPTOP-CT06LJB9: ~  
yo@LAPTOP-CT06LJB9:~$ R  
  
R version 4.3.3 (2024-02-29) -- "Angel Food Cake"  
Copyright (C) 2024 The R Foundation for Statistical Computing  
Platform: x86_64-pc-linux-gnu (64-bit)  
  
R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.  
  
R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.  
  
Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.  
  
>
```

Figure 7: Launching R from the command shell. The release number (4.3.3 in this example) appears in the first row of information.

2.5 Third step: setting up the application

Once you have downloaded and uncompressed the ZIP file from `github.com` or performed a `git clone` of the `bipartgraph` repository, the software is ready for the final setup. Move to the `bipartgraph` directory (`bipartgraph-master` if installed from the ZIP).

A) Windows

Double-click on `install_bipartgraph.R`. If you have set up R in your Path variable, you may also install from the Windows command line or PowerShell by typing:
`Rscript install_bipartgraph.R`

B) UX like

Type in the terminal:

```
Rscript install_bipartgraph.R
```

The installation procedure is very fast in Windows because all packages are precompiled. It may take a while in Linux because your machine will need to compile several packages. Please be patient and do not close the terminal until the procedure finishes.

Workaround

Depending on your user privileges, `install_bipartgraph` may fail in Linux if the current user has no permission to write in the package directory. If so and only if so, type instead:

```
sudo Rscript install_bipartgraph.R
```

C) Final Check

Once the script finishes, the procedure is done. Check that everything was correctly installed. Open a new terminal (or command line), and go to the `bipartgraph` directory or `bipartgraph-master` if you installed it from the ZIP file. Double-click the `plot_test.R` (Windows) icon or type in the command line (UX like): `Rscript plot_test.R`

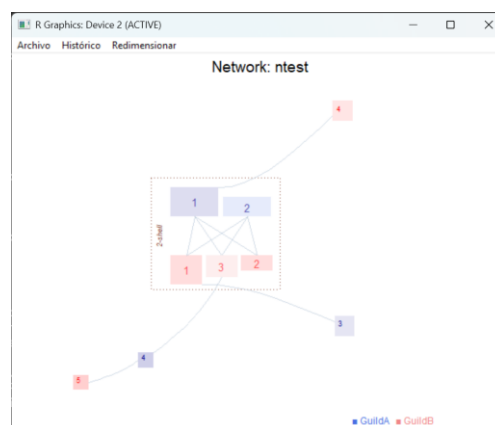


Figure 8: Result after running the script `plot_test.R`

Can you see Figure 8? If so, the installation ended successfully.

2.6 Hacks for installation on Linux

This section explains overcoming common problems when installing Bipartgraph on a Debian-based or Fedora Linux distribution. With minor modifications, it is also valid for any other distro.

You need a gcc compiler on your machine. If you are a developer and have it, kcorebip will install smoothly. Otherwise, you will get a message telling you that Rtools is unavailable. In that case, follow the instructions on the following page to install Rtools:

<https://cran.r-project.org/bin/windows/Rtools/>

Run the script for Linux dependencies, choosing your distro:

```
> sudo sh < linux_debian_dependencies
```

or

```
> sudo sh < linux_fedora_dependencies
```

This script downloads and installs the R runtime environment and the libraries to compile dependencies.

Download <https://github.com/jgalgarra/bipartgraph/archive/refs/heads/master.zip>, then unzip and place the bipartgraph-master folder in any path of your user home. Move to that folder or `git clone` the repository and go to the "bipartgraph-master" folder.

```
> Rscript install_bipartgraph.R
```

The installation procedure may last around half an hour because of the compilation of dependencies. If you get... "This is the end, my friend. Installation completed", everything went OK :-)

You can check that the installation is fine with the following command:

```
> Rscript plot_test.R
```

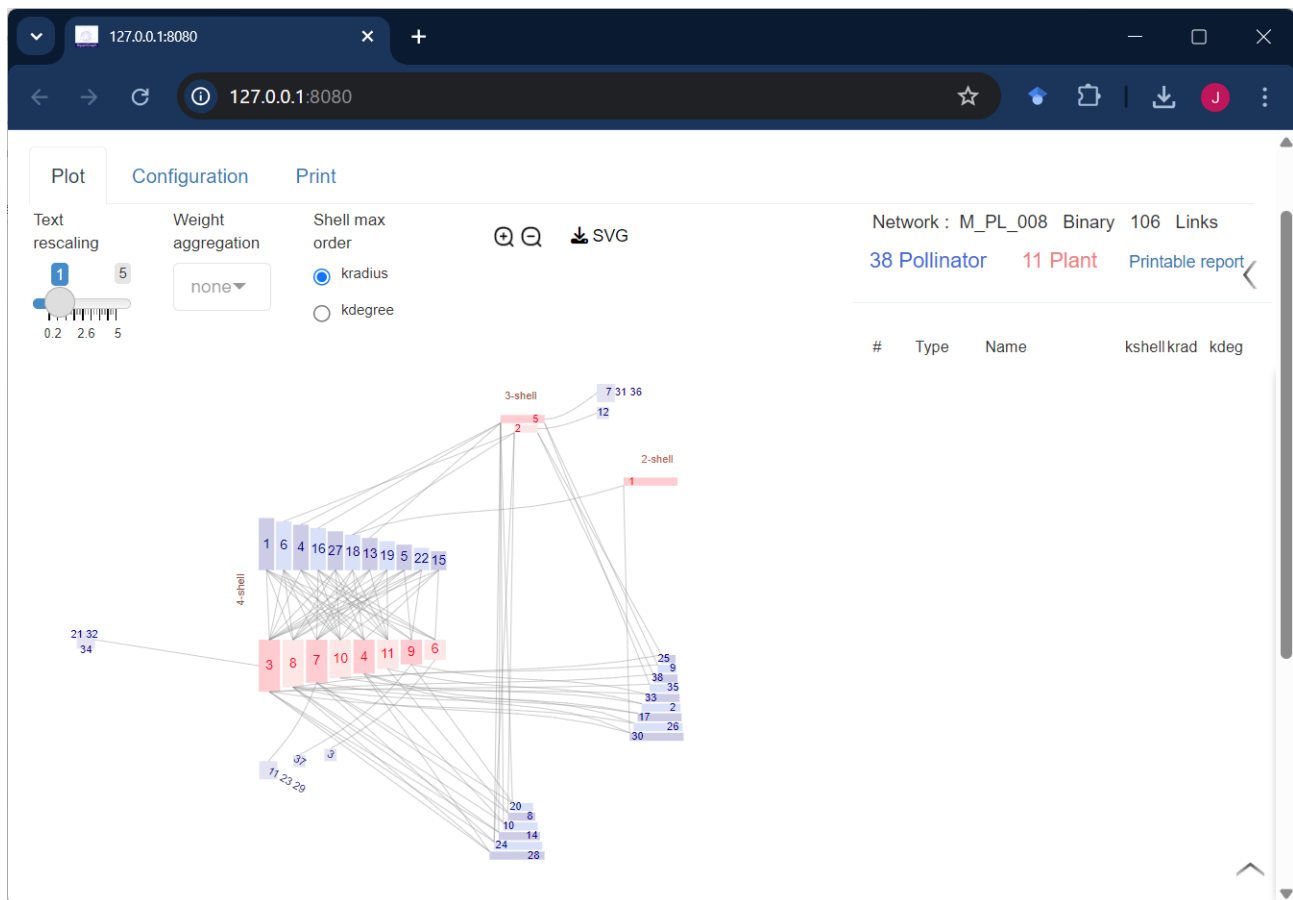
2.7 Running *BipartGraph*

The application is now ready to run. If you are running *BipartGraph* with Docker, please read section **¡Error! No se encuentra el origen de la referencia..** Otherwise, launch the script by double-clicking on `bipartgraph.R` icon (Windows) or type in the command line (UX like):

```
Rscript bipartgraph.R
```

Your default web browser will open a new tab with a local URL. If everything went right, you have reached the landing page.

Click on **Data** (main menu), then click on the **Data file** box and select `M_PL_008.csv`. Click on the **Interactive Ziggurat** tab (main menu) to see the network plot of Figure 9.



3 Data management

BipartGraph is a tool for plotting bipartite ecological communities, but you may use it with any bipartite network. This guide uses examples of mutualism and parasitism that are well documented in the Web of Life database [4].

	A	B	C	D	E	F	G	H	I	J
1	Anastoechus latifrons	Echium wildpretii	Pimpanella cumbrae	Pteroccephalus lasiospermus	Mentha longifolia	Erysimum scoparium	Spartocytisus supranubius	Tolpis webbia	Argyranthemum teneriffae	Scrophularia
2	1	1	0	1	0	1	1	1	0	0
3	Anthophora alluaudi	1	0	0	1	1	1	0	0	0
4	Apis mellifera	1	1	1	0	1	1	0	0	1
5	Eudynesus reflexus	0	1	0	1	0	1	1	0	0
6	Geron hesperidion	0	1	1	0	1	0	1	1	1
7	Ernstalis tenax	1	1	1	1	0	0	1	0	0
8	Megachile canariensis	1	0	1	0	1	1	1	0	0
9	Anthrax anthrax	1	1	1	0	0	0	0	0	0
10	Eucera gracilipes	1	0	0	0	1	1	0	0	0
11	Hylaeus canariensis	1	1	0	0	0	1	1	0	0
12	Lasioslossum viride	1	1	1	0	0	0	0	0	0
13	Limnaemyia soror	1	1	0	1	0	0	0	0	1
14	Cephalodromia sp1 M_PL_008	0	1	0	0	0	0	0	1	1
15	Cyclotryus webbians	0	0	1	1	1	0	0	0	0
16	Estheria simonyi	1	0	1	1	0	0	0	0	0
17	Lasioslossum acitrons	0	1	0	1	0	0	0	0	0
18	Melecta curvispina	1	0	1	0	1	0	0	0	0
19	Osmia canariensis	0	0	1	0	1	0	0	0	0
20	Andrena vollesstoni	0	0	0	0	0	1	1	0	0
21	Colletes dimidiatus	0	0	0	1	0	0	1	0	0
22	Gasterosternus sp1 M_PL_008	0	1	1	0	0	0	0	0	0
23	Lucilia sericata	0	0	1	0	0	0	0	0	1
24	Macroglossum stellatarum	1	0	0	0	1	0	0	0	0
25	Scaeva albomaculata	0	0	1	0	0	0	0	0	0
26	Stomophina lunata	1	0	0	1	0	0	0	0	0
27	Undentified sp1 M_PL_008	0	0	0	0	0	0	0	0	1
28	Anthidium manicatum	0	0	0	0	0	0	0	0	0
29	Bibio elmoi	0	0	0	1	0	0	0	0	0
30	Dermasothus gracile	0	1	0	0	0	0	0	0	0
31	Drosophila sp1 M_PL_008	0	0	0	0	0	0	0	0	1
32	Lasioslossum chalcodes	1	0	0	0	0	0	0	0	0
33	Leptochilus eatoni	0	1	0	0	0	0	0	0	0
34	Nyctia lugubris	0	1	0	0	0	0	0	0	0
35	Peleteria ruficornis	0	0	0	1	0	0	0	0	0
36	Phylloscopus collybita	1	0	0	0	0	0	0	0	0
37	Serinus canarius	1	0	0	0	0	0	0	0	0

Figure 10: Interaction matrix of network M_PL_008 [5]

Data are stored as .csv files. Fields are separated by commas, semicolons, or tabs. The first row stores the binomial names of guild A species, and the first column those of guild B. Remember that the application waits for that information when you upload your data. If you do not have it, or it does not make sense in your environment, type the numerical values and uncheck the option Name species in the header when you select to analyze your network. BiparthGraph will label species following the order in the input file. If the interaction matrix is binary, the cell of *species A* m ; *species B* n will be set to 1. If it is weighted to a positive real, 0 means that there is no link. We provide examples with the default installation software. To upload your data, click on the Data tab of the main menu and then on Manage Files.

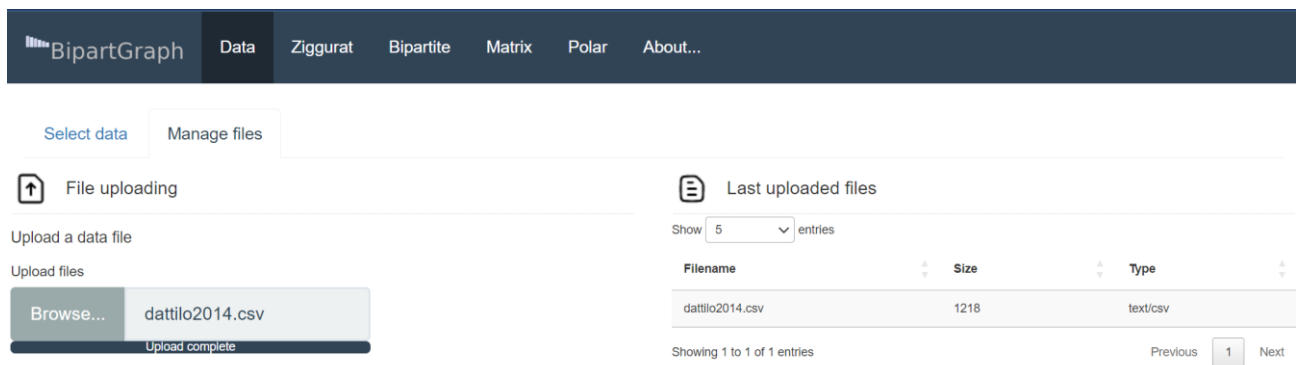


Figure 11: Uploading a file

Click again on **Select Data** and choose any file. The table shows the interaction matrix. You can change Guild A and Guild B labels; *BipartGraph* will remember the new ones next time you open the same file.

The **Network Analysis** button creates a .csv file with the individual *k-magnitudes* of all nodes. With **Reset all**, the session restarts, and all configuration parameters are set to their default values.

The **Species names** button downloads just the species names of a network; this list may be convenient for the caption of a LaTeX image, for instance.

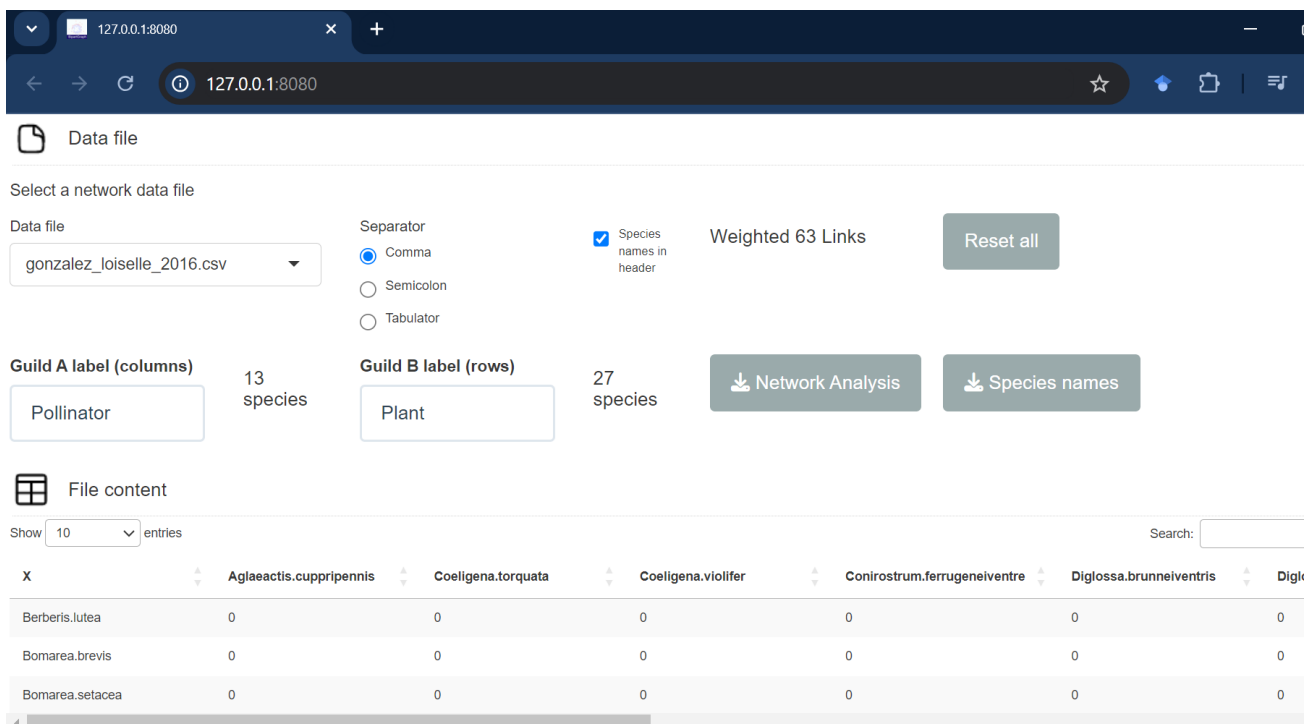


Figure 12: Network data

4 The ziggurat plot

The ziggurat graph is an original kind of visualization. The idea is splitting species into sets by their *k-shell* numbers [2]. If you are not interested in *k-magnitudes*, knowing that the *k-shell* number is a connectivity measure is enough. Groups of nodes that share the same *k-shell* number are depicted as miniature ziggurats.

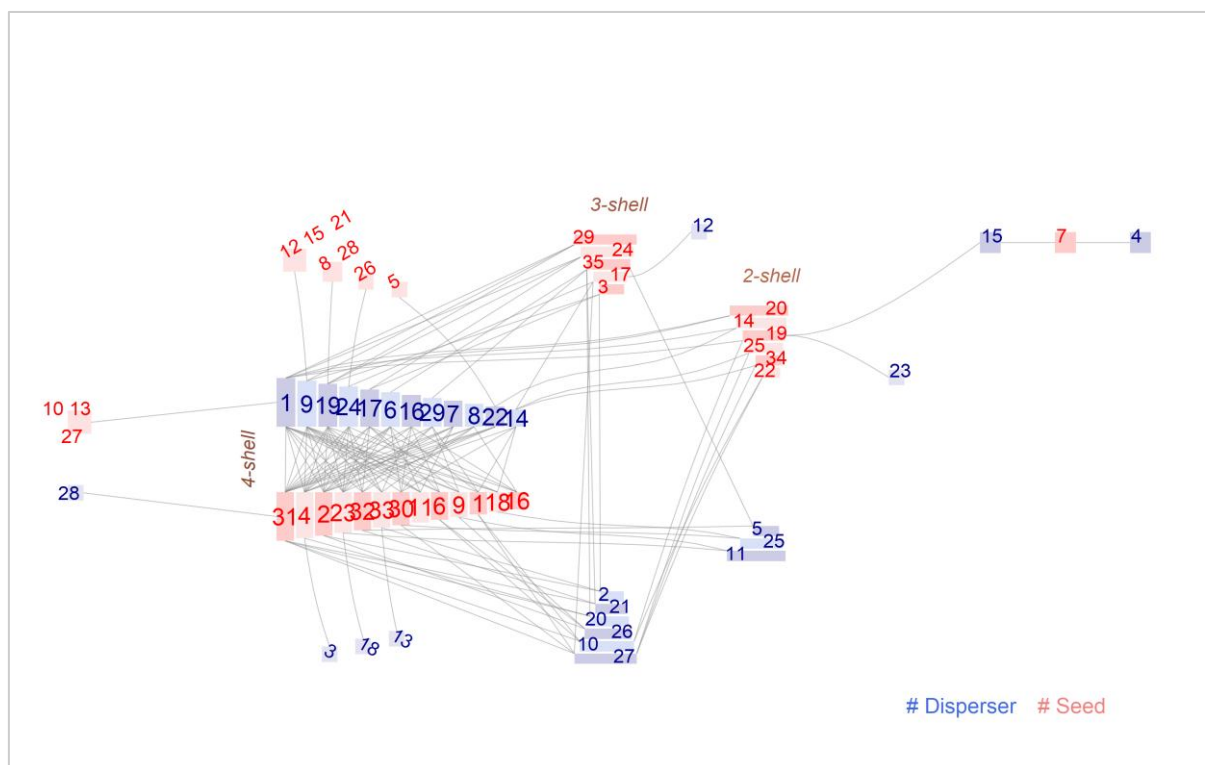


Figure 13: Ziggurat plot of an eating fruit birds network in Brazil, M_SD_012 [4]

Although networks are quite different in size and connectivity, they share building units that will help you to get a clean glimpse of the structure (fig. 14). The building blocks are **nodes** and **links**. Bipartite networks have two kinds of nodes that we conventionally call **guilds**. In this example, there are frugivore species of birds, playing the role of seed *dispersers* (guild A), and plant species that yield those *seeds* (guild B). If two species interact, there is a link between them; for instance, bird 18 feeds on the fruits of *plant* 23. The bird species, in turn, benefits the plant species by fostering its reproduction through seed dispersal. Links among nodes of the same guild are not possible in bipartite networks.

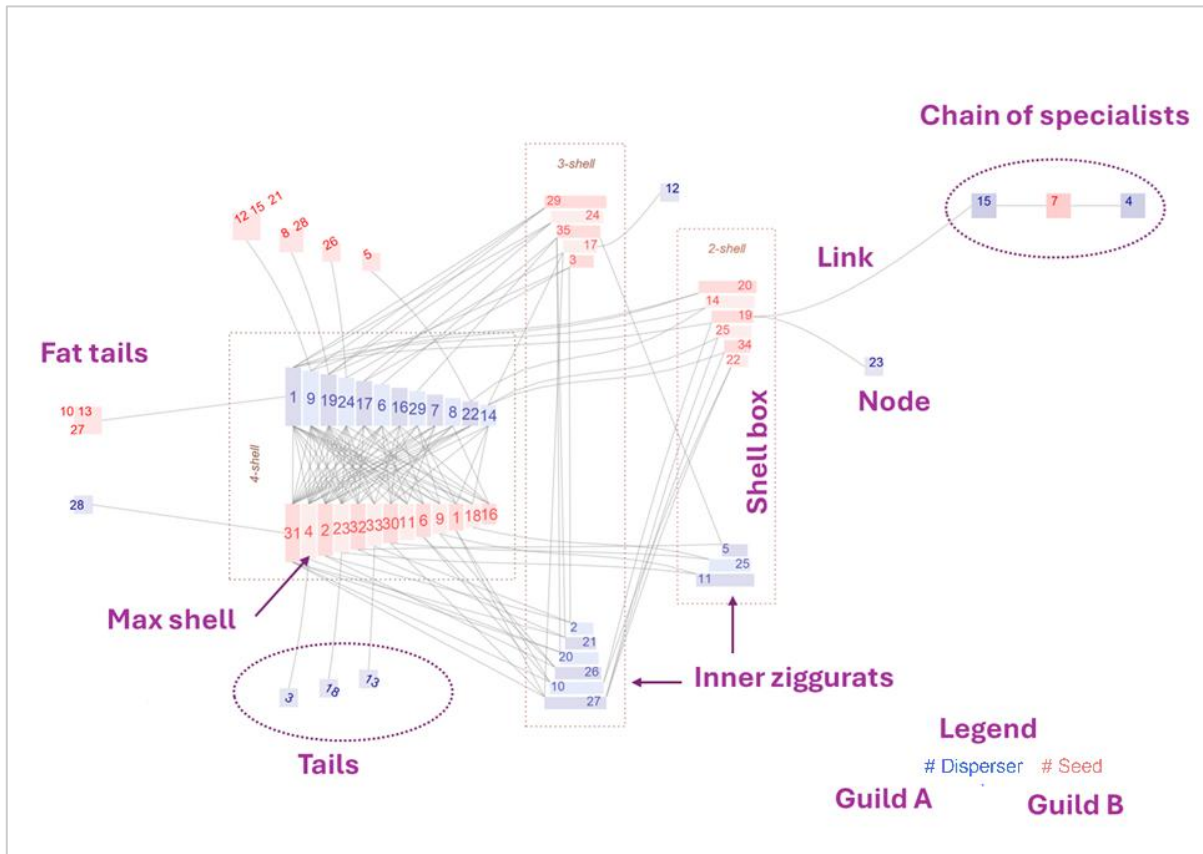


Figure 14: Elements of the ziggurat plot

Links have no label and share a standard color. By default, nodes are filled with the color of their guild, with two slightly different tones to help visual perception. Each species is numbered by its order within the input file. So, plant 1 is the species recorded in column 1, and bird 5 is in row 5.

Node size does not convey any information in the ziggurat plot; shapes make it more readable. Although species of the same guild and shell appear stacked in the ziggurat, they do not share any other property. Plants 2, 10, 20, 26, and 27 belong to the 3-shell; this is the only fact relevant for this plot.

The **legend** informs about the name of each guild; the left element is always *guild A*. Ziggurats may optionally appear enclosed by dotted rectangles that we call **shell boxes** with a box label. The shell with the highest *k-index* (4 in the example) is the **max shell**. It is located at the center of the plot. Shells of lower indexes (3 and 2) are arranged following an almond-shaped distribution. We call them **inner ziggurats**.

Nodes of *1-shell* are scattered around the plot. They may be part of two structures: tails and chains of specialists. A tail is just one species of *1-shell* linked to one species of a higher shell of the opposite guild. For instance, *bird 5* and *plant 3* are tails.

When multiple nodes of a *1-shell* are tied to the same species of any of the ziggurats, they are clustered to reduce the number of lines. Species *Plant 12, 15, and 21* are linked to the generalist *disperser 9*.

Chains of specialists are less common than tails. They are sets of species of *1-shell* linked to species of *1-shell* of the opposite guild. *Bird 15–plant 7–bird 4* is an example of a chain of specialists.

With this organization, we have a clear view of structure and interconnections. The central, almond-shaped area leaves a vast space for the links, so they do not overlap the boxes. Other structural details are easy to catch with this visualization. It is also clear why *bird species 4* and *plant 7* are in a dangerous position. They depend on another specialist, *bird 15*, with only one link to one *2-shell* species. If this species gets extinct in that environment, the chain of specialists disconnects from the network giant component.

Sometimes, field scientists record species that are not linked to the giant component. These **outsider** species are plotted as a subnetwork under the ziggurat plot (Fig. 15).

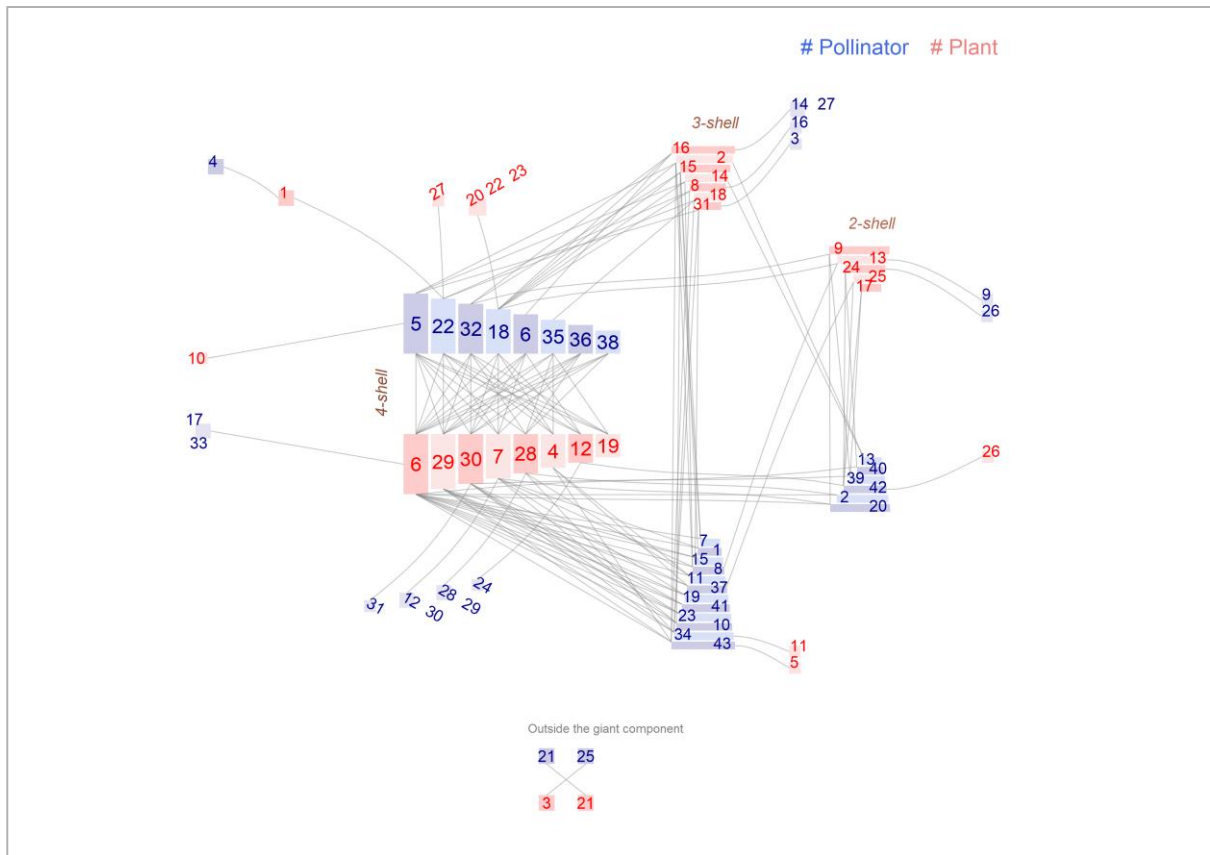


Figure 15: Plant – pollinator community M_PL_041 in Syndicate, Dominica Island [6]

4.1 Explore the interactive ziggurat

Go to Data, select M_PL_008.csv, and then click on Interactive Ziggurat. The plot is displayed on the left, inside the Plot panel.

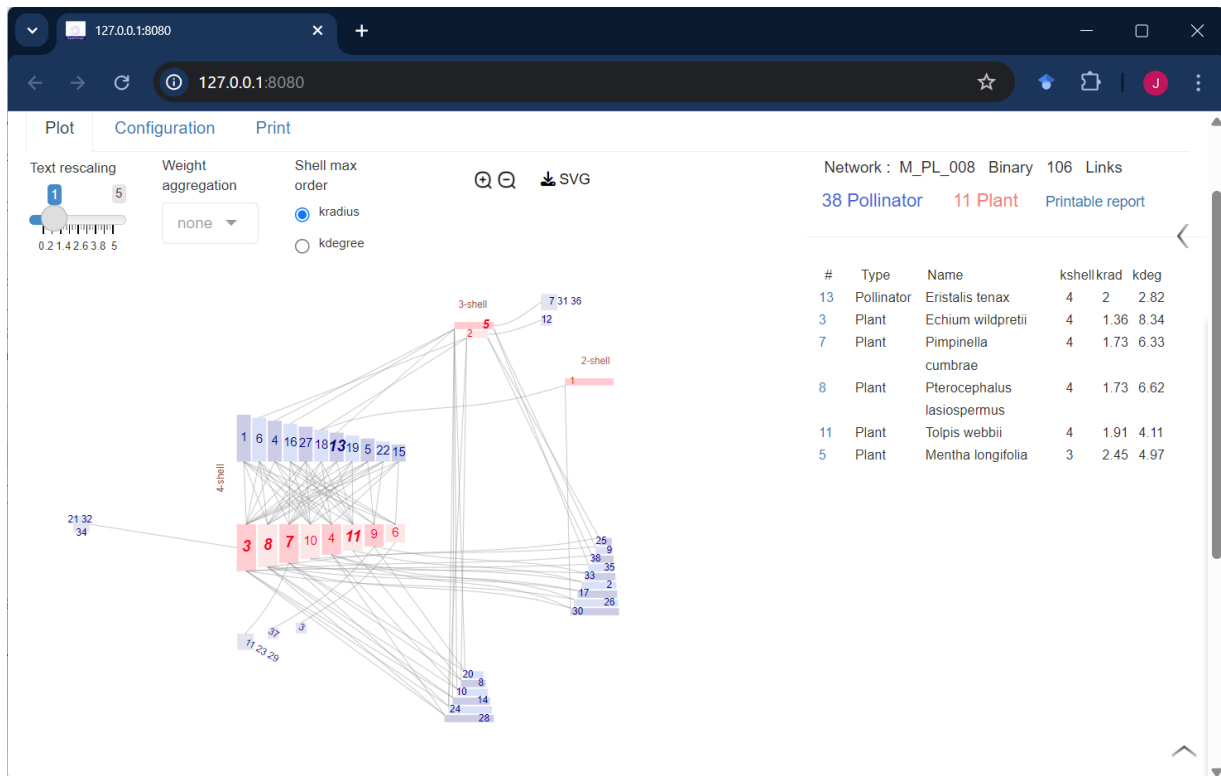


Figure 16: Interactive Ziggurat

When you pass the mouse over each species box, a tooltip shows its binomial name and *k* magnitudes.

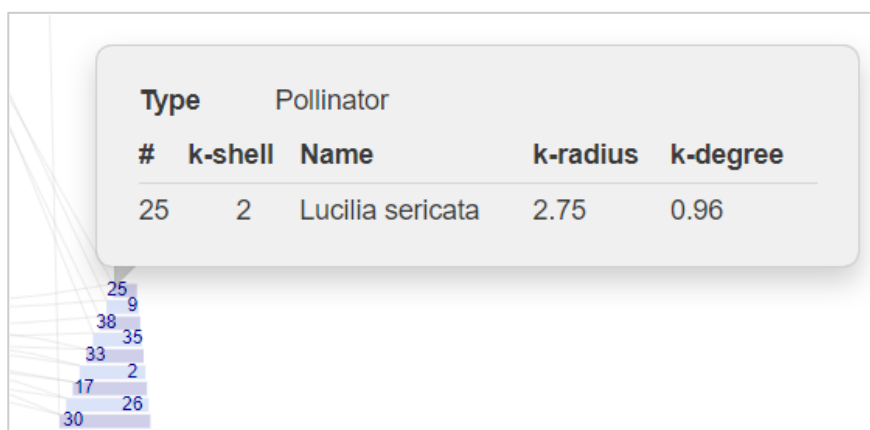


Figure 17: Species tooltip

If you click on one of these boxes, the links to other species are highlighted as dashed lines, and the labels of those neighbors look larger.

On the right panel, you may see the information table of the selected species (*disperser 13* and all its partners in Fig. 16). Clicking on the species number in the table, the application

opens the Wikipedia information about that species. You will get an error if the binomial name is not written correctly in the data file. Sometimes, Wikipedia may redirect your request to a disambiguation page.

4.2 Modifying the look and feel of the ziggurat plot

The elements of a ziggurat are configurable for the sake of clarity and beauty. Let's start with the simple ones, *links*. Please select the file RA_HP_001.csv, a host-parasite network, with 18 host species, 10 parasite species, and 61 links. If it is the first time you load it, change the Guild A label to Host and the Guild B label to Parasite in the Data panel. *BipartGraph* will remember this setting next time you load this network file. Click Interactive Ziggurat on the main menu, and you will get a fancy default visualization of this small network (fig. 18).

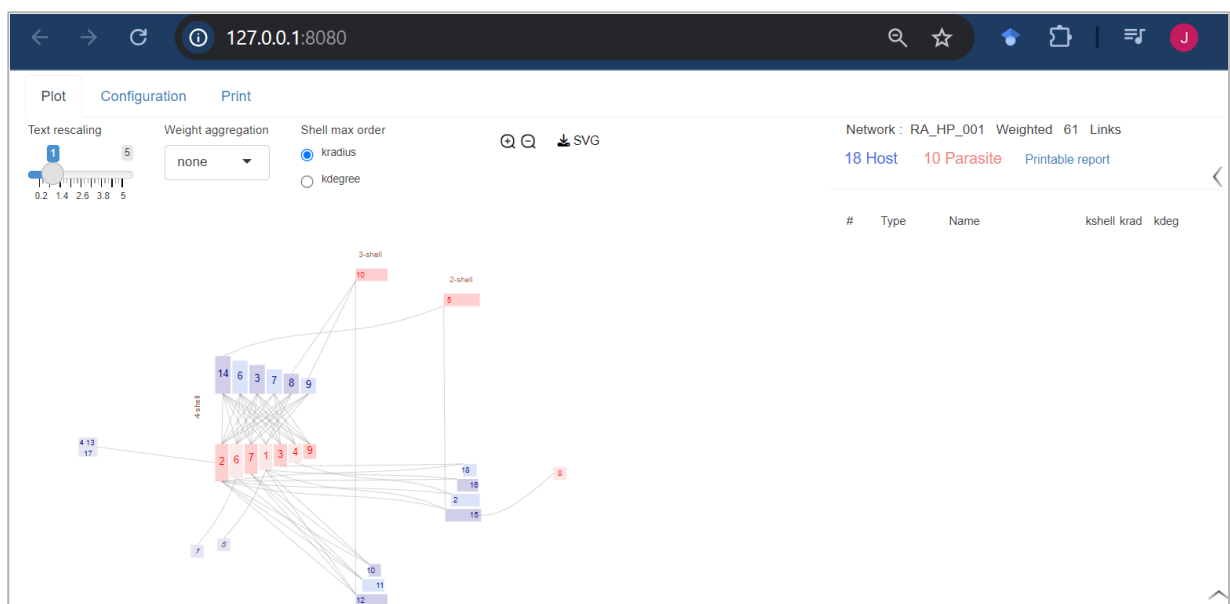


Figure 18: Host-parasite network in Azharia [7].

You may display all species names at any time by clicking on the upper right (or lower right) corner arrows to show/hide a vertical (horizontal) slider (fig. 26).

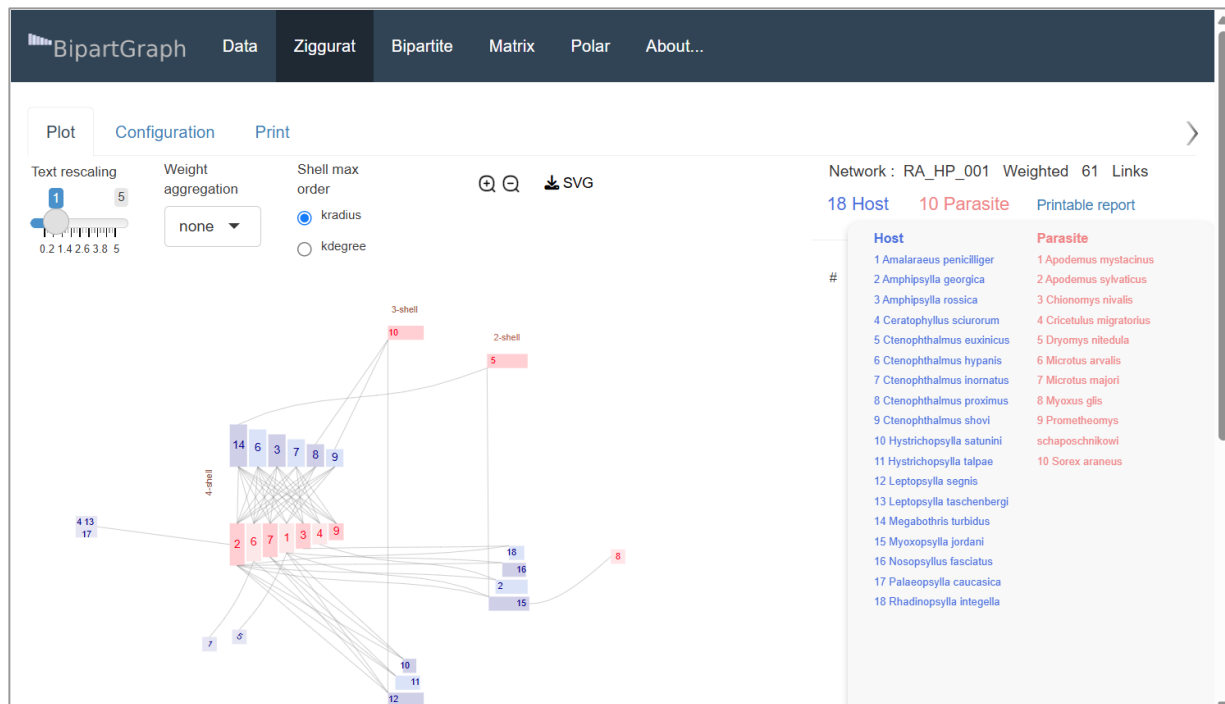


Figure 19: Vertical slider with species names.

Click on **Configuration** tab and the first thing you see are the link controls. There are 5 of them (fig. 20).

- **Show links.** This box is checked by default. If you uncheck it, links will hide. This may be useful when working with a huge network, because link processing is the most time-consuming task. You may decide to hide them while you configure other elements.
- **Use spline.** Links come in two flavors: straight lines (parasite 7—host 6) and splines, smoothly curved lines (parasite—host 10). If you uncheck this box, every link will be straight, and the plot will look more rude.
- **Spline points.** The number of points of each spline; the bigger, the smoother and more time-consuming. The default value is 10; do not increase it very much unless you produce a high-quality plot for a journal.
- **Link width.** If the network is binary (there are only 1's and 0's in the interaction matrix) all links have the same width. If weighted, Weight aggregation lets you choose to make their width proportional to the interaction strength (\ln or \log_{10} of these values) or ignore weight (no) and plot them as if the network were binary.
- **Color and Transparency** define how the links are filled. The transparency ranges from 0.1 (almost invisible) to 1 (opaque color).

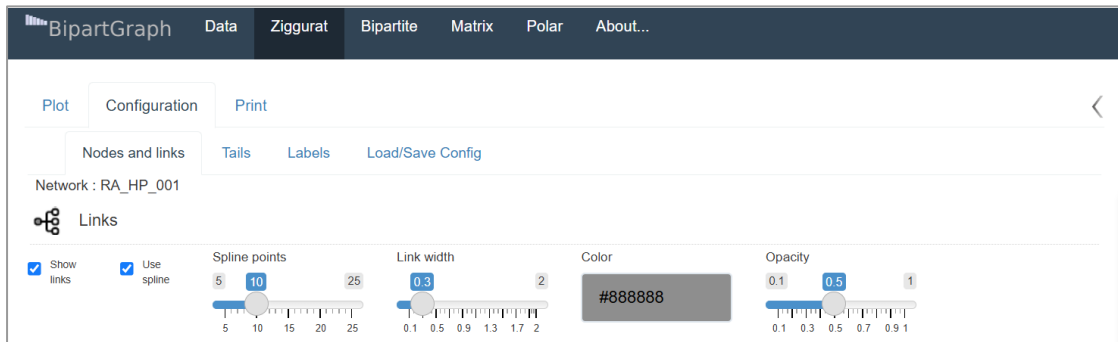


Figure 20: Link configuration control.

Let's go back to Figure 18 to make some adjustments. First, uncheck the **Use spline** box (the **Spline points** control has no meaning if you use straight line links). You can make the width proportional to the interaction strength. Select **ln** in **Weight aggregation**, the width of each link will be proportional to the natural logarithm of its strength (the value recorded in the interaction matrix). Click again on **Interactive Ziggurat / Configuration**.

To test the effect of other configuration parameters, increase the **Link width** slider up to 0.5, move the **Transparency** control to 0.2 and pick a green tone color **#087007** (you can write it inside the **Color** box).

That's all, now it's time to see how the ziggurat changes. Click on the **Plot** tab and the software will plot again the network (fig. 21). Perhaps it will not win a plot beauty contest, colors pop up your eyes, and straight links make it less pleasant than the default figure. 18. On the other hand, you can detect that *host 7* is more prone to have parasites of *species 6* than of *species 10*.

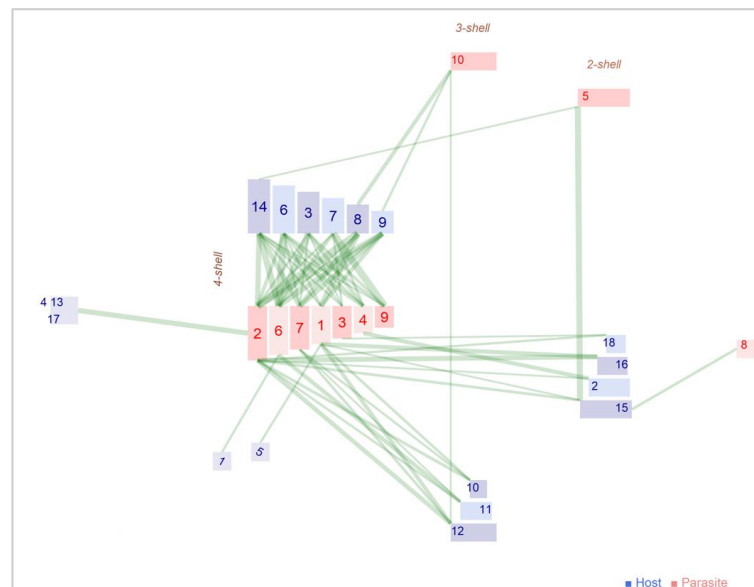


Figure 21: Host - parasite network in Azharia [7], with modified link properties.

Now that we have learned how to modify **links**, let's move on to more exciting elements of the graph. The network of figure 15 is larger. There are some details that need improvement,

for instance the height of the nodes of the *inner ziggurats*, they look smashed. Go to Configuration, move the Nodes height slider to 1.5 and click on Plot again.

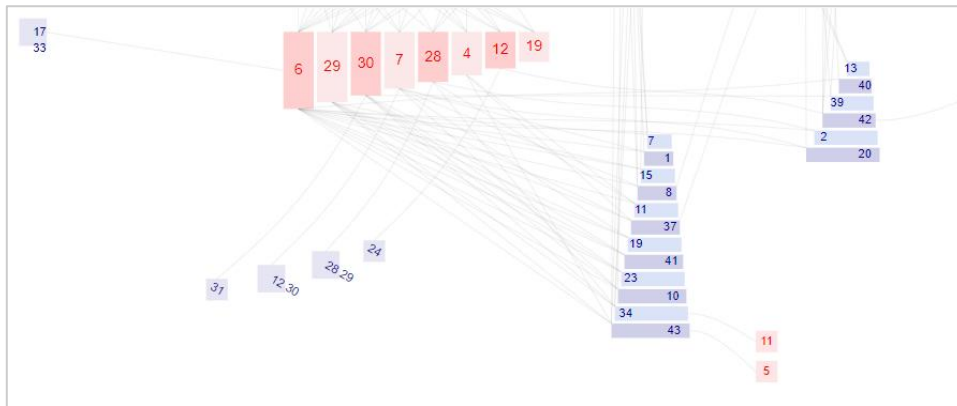


Figure 22: Detail of plant – pollinator community M_PL_041 in Syndicate, Dominica Island [6]

Boxes are fancier now, but links from 2-shell towards 4-shell overcross the ziggurat of pollinators of 3-shell. We can move it upwards or downwards with the Inner ziggurats vertical displacement control. The ziggurat troubling you is the 3-shell of guild A (if you do not know which guild you are dealing with, the color of the legend will tell you).

Please, move the 3-shell A slider to -0.1 (roughly move it down 10% of the original position). The *chain of specialists* in the upper left end of Figure 15 is a bit long. Click the Tails tab and reduce the distance.

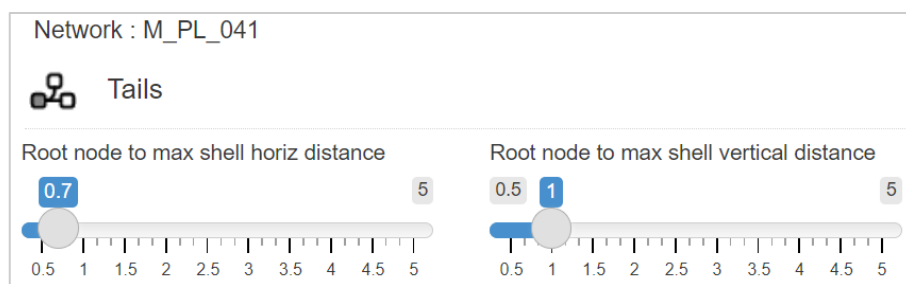


Figure 23: Tail display control sliders

Finally, move the *outsiders*, clicking on the Nodes and links tab:

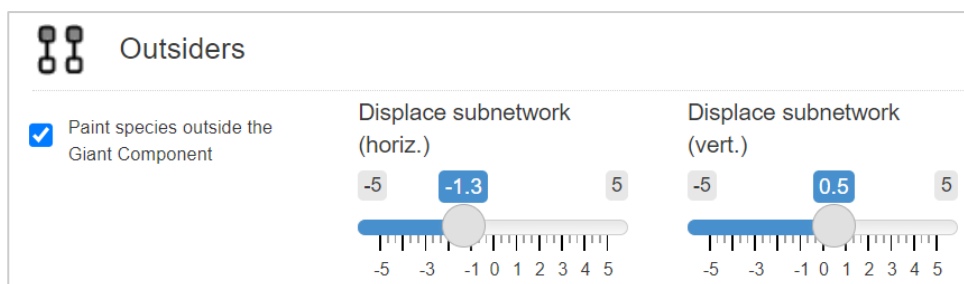


Figure 24: Outsider species display control sliders

After you apply all these changes, you get the ziggurat plot in Figure 25. Of course, you could keep on tuning parameters.

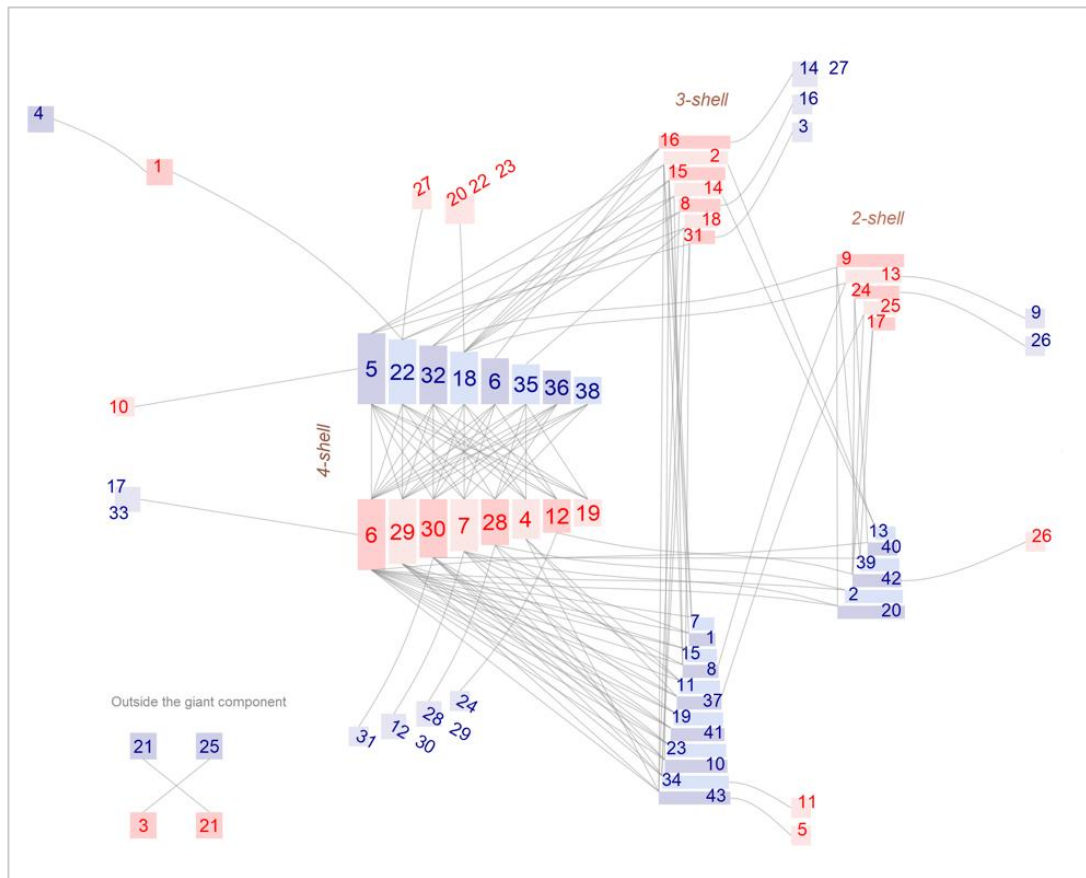


Figure 25: Ziggurat graph of a plant-pollinator community in Syndicate, Dominica Island [5] after visual adjustments.

For instance, outsiders could be disturbing, so you may hide or reduce their area. This network is weighted; try changing link properties as in the previous example and see how the plot tells a different story.

A new example will show how to deal with other elements of the ziggurat. Go to **Data**, and before selecting a new file, push the **Reset all** button. This action restores the default values of all the controls. Now, select the `M_PL_012.csv` file and display the interactive ziggurat.

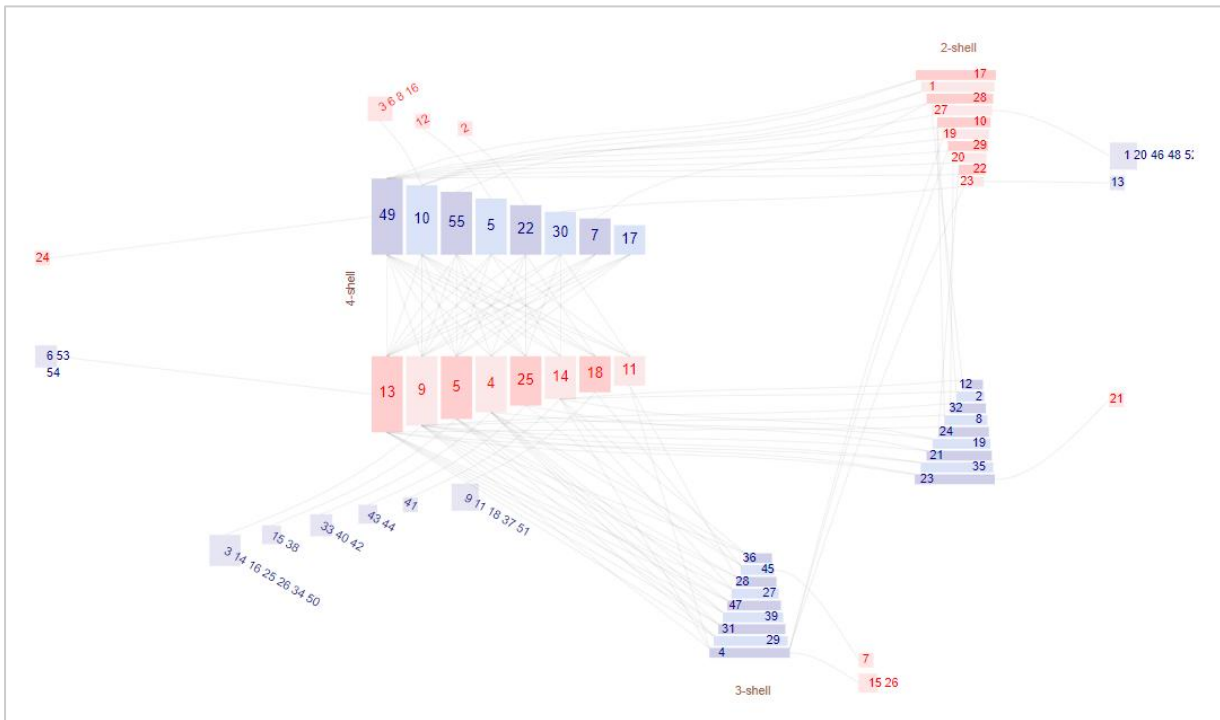
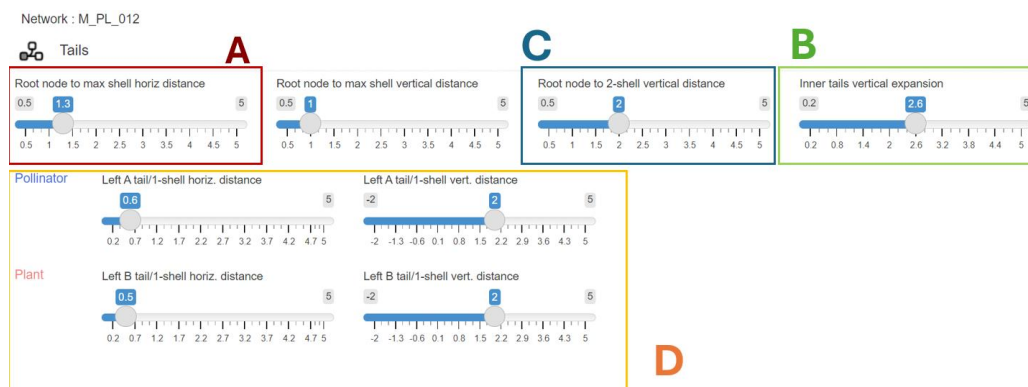


Figure 26: Ziggurat graph of a plant-pollinator in Garajonay (Spain), recorded by Jens Olesen. Unpublished.

This community has many species in *1-shell*, grouped in *tails*. Pay attention to the tails that are above and below the *max shell*. The graph is overcrowded, and it would be nice to make additional room for them, expanding their horizontal distance (A). Then, we increase the vertical distance among tails connected to inner ziggurats (B), like plants 7, 15, and 26. The third step is widening the gap between the two tails connected to plant 2-shell (C). The last modification pulls up/down and right the left tails, those connected to the leftmost species of the *max shell* (D).



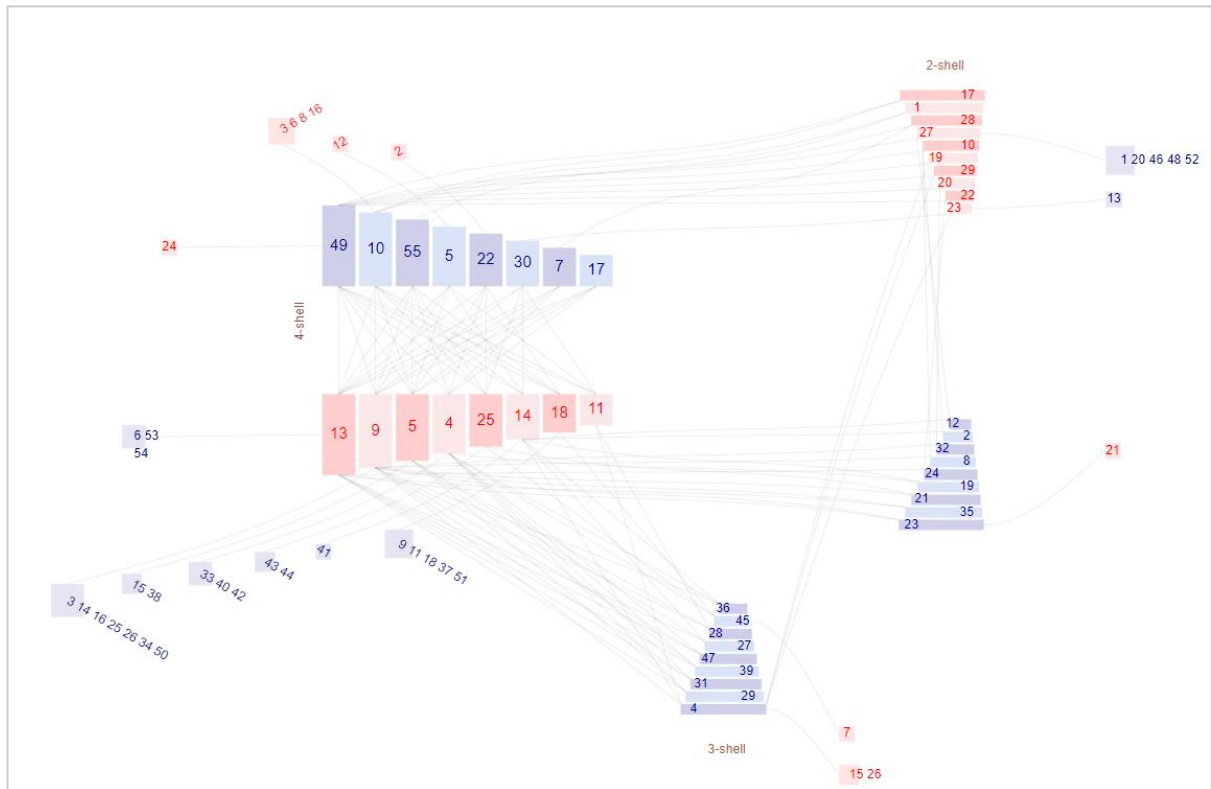
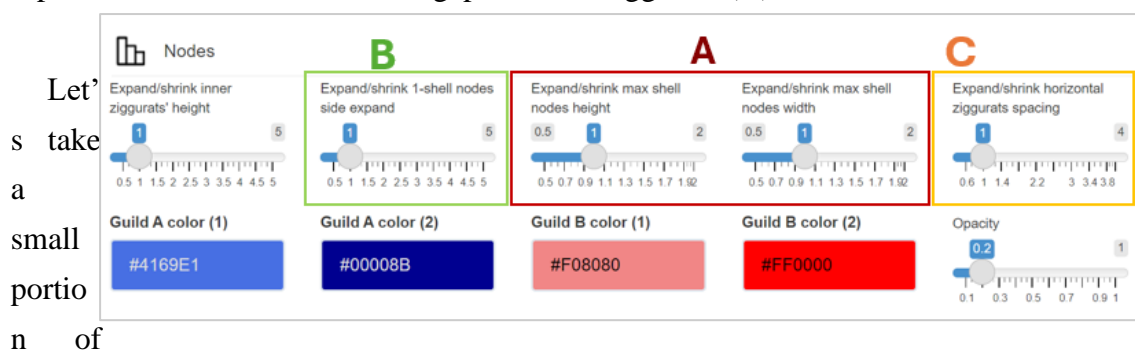


Figure 27: Ziggurat graph of a plant-pollinator in Garajonay (Spain), after visual adjustments.

The dimensions of the *max shell* triangle are controlled with two sliders: Max shell height expand and Max shell width expand (**A**). The area of all boxes of *1-shell* may be increased by the factor of the 1-shell node expansion (**B**). There are four color pickers for guild nodes. Please use them with care. *BipartGraph* will remember your choice next time you load the same network. Transparency works as explained above when talking about *links*. The horizontal inner shells' separation widens the horizontal gap between ziggurats (**C**).



Let's take a small portion of network M_PL_031 to show the behavior of *specialists'* controls (Fig. 21). They expand or contract these chains. The distance of the root node is modified by the First node's horizontal/vertical distance to the network if tied to nodes in any ziggurat except a 2-shell, which has its controls.

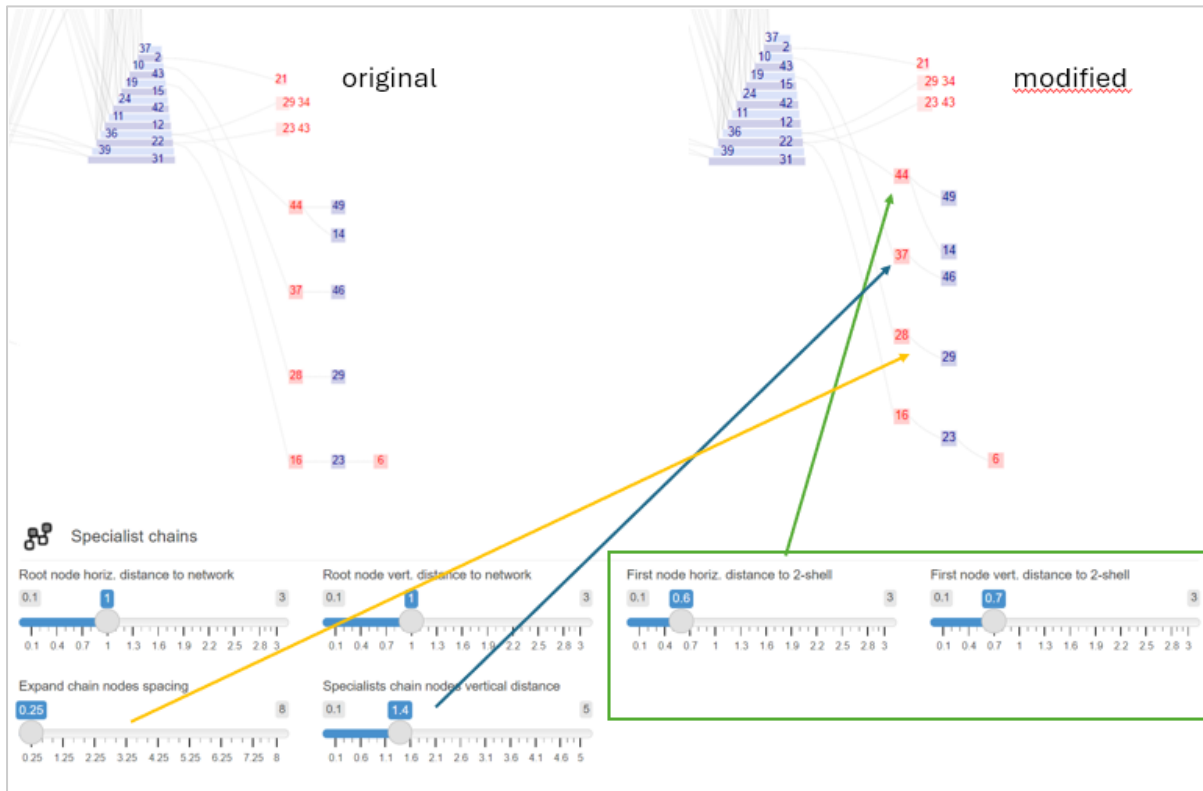


Figure 28: Changing the configuration of the chains of specialists.

Expand chain boxes separation controls the horizontal distance among nodes of any chain, and the Specialists chain nodes' vertical distance is the vertical gap.

By default, each guild is filled with two different tones of color, but you can pick just one by checking the `One color per guild` box.

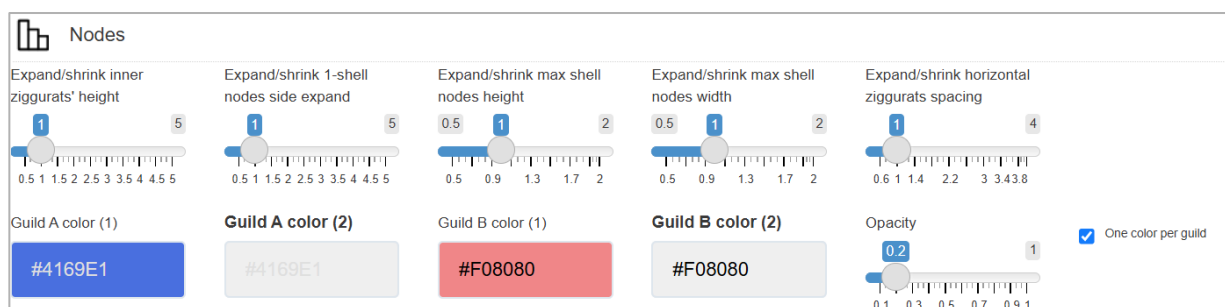


Figure 29: Choosing one color per guild.

4.3 Printable ziggurat

Once you have a clean ziggurat, it is quite possible that you need to produce a high-quality plot for your research. The Print tab of the main menu will do that for you.

Select again the network `M_PL_012.csv` and visualize the default Interactive Ziggurat. Now, go to **Print** and click the button **Plot Download**. Your browser will download a file called `M_PL_012-ziggurat.png` (fig. 30). Compare it with the interactive plot (fig. 26). It is

almost identical but has minor visual differences. In the printed plot, the name of the network and legend can be displayed, and perhaps the sizes of labels and legend are slightly different.

The interactive and printed plots are built with two different technologies: the first one is an SVG object, and the second is a `ggplot2` object. These details are not essential for you as an application's end user, but sizes may differ depending on your screen's resolution and the installed fonts. In the **Plot** view, you can rescale all labels at once. The default value or rescaling is 1; if you increase it, the size of the labels plot will grow compared to the printed ones. It may be helpful for final adjustments.

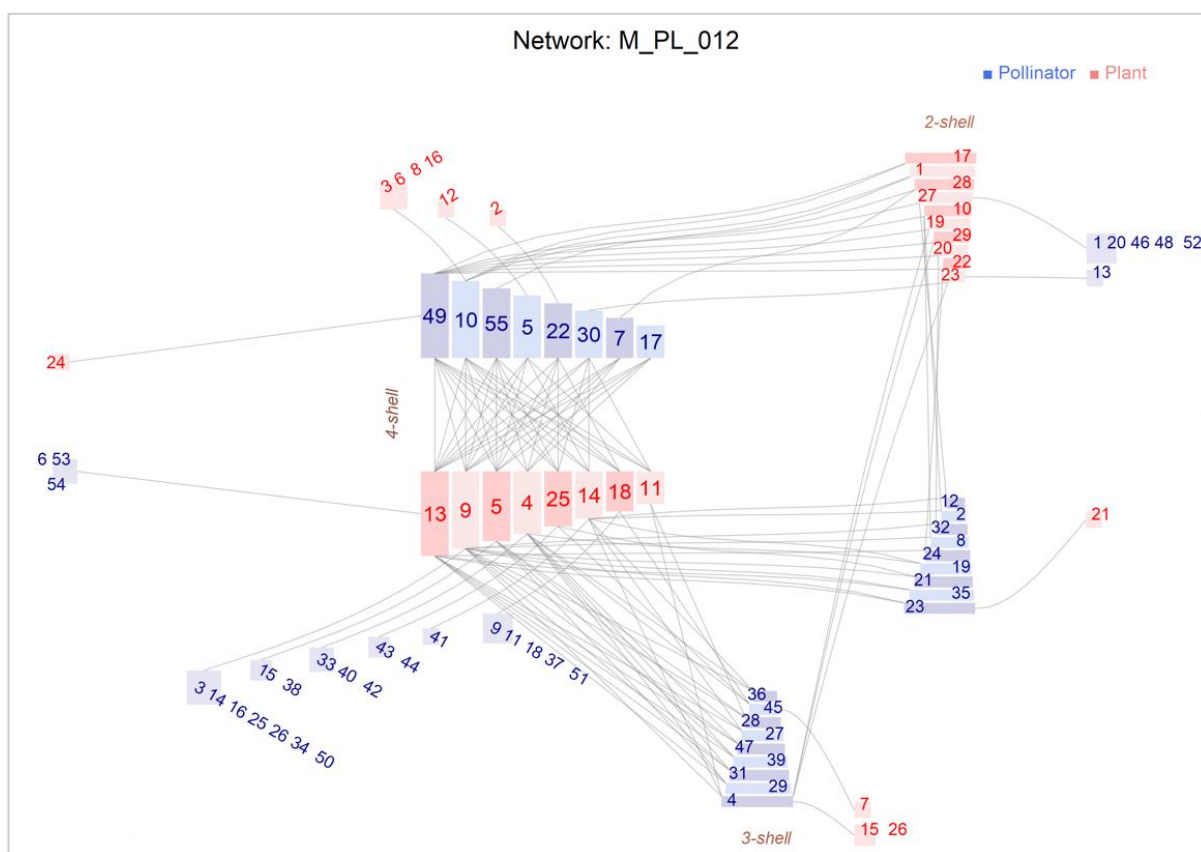


Figure 30: Printed Ziggurat graph of a plant-pollinator in Garajonay (Spain).

Go back to the **Print** pane. The options are pretty straightforward. You may change the orientation (Landscape), resolution, title, and legend options. The Aspect Ratio is an exceptional control that only affects the printed version. If it is bigger than 1, the graph is stretched in the vertical direction; otherwise, it is the opposite. You can choose four file formats for the printed plot: `png`, `jpg`, `eps`, `tiff`, `svg`, and `pdf`.

When you get a nice printed ziggurat, you may want to save the display parameters configuration for reproducibility. This is what the **Download generating code** button does. Click on it to see the complete invocation of the R `ziggurat_graph()` function. The meaning of each invocation parameter is explained in the `kcorebip` user manual. This package was installed on your computer during the process, and the guide explains how to run the code:

https://github.com/jgalgarra/kcorebip/blob/master/inst/doc/kcorebip_man.pdf

If you are not a programmer but want to get the printed ziggurat again using the code, please follow these steps. Create a directory called `MyRCode` where you want to go and copy the file `M_PL_012-ziggurat-code.txt` that the application generated. Create a folder called `data` inside that directory and copy there the input data files, for instance, the `M_PL_012.csv` that you used as an example and lives in `biparthgraph/data`.

Create the folder `plot_results` inside `MyRCode`, and inside `plot_results`, the folder `ziggurat`. Please copy, paste, and save this code in a `genziggurat.R` file inside the `MyRCode` directory.

```
library(kcorebip)
source("M_PL_012-ziggurat-code.txt")
```

Your new directory should look like this: with the file `M_PL_012.csv` in the `data` folder and an empty `ziggurat` folder inside `plot_results`.

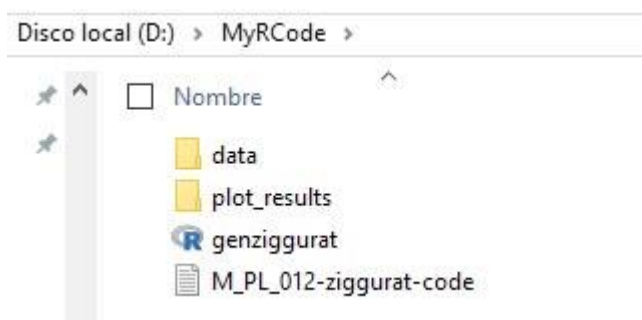


Figure 31: Downloading a ziggurat generating code.

Go to the command line and run

```
> Rscript genziggurat.R
```

When the script finishes, the `M_PL_012_ziggurat.png` file is available inside `plot_results/ziggurat`.

If you are not a developer but want to save the tuned parameters for the future, go to the **Config/Load Save Config** pane. Use the **Save plot configuration** to save the values in a text file. You can read them later with the **Browse** dialogue to choose any configuration file. Checking **Show contents** will show the values of each parameter stored in that particular file.

5 The bipartite plot

The bipartite plot is the conventional way to depict a bipartite network. Nodes of both guilds are aligned in two parallel lines (horizontal or vertical is a matter of convenience), ordered by degree. A straight line between two nodes of different guilds means that there is a link, and its width may be proportional to the strength of the interaction. The bipartite plot becomes a *hairball* as the number of nodes and links increases. There are some minor mitigation strategies to overcome this problem. This package offers the *legacy* style for bipartite plots, following the traditional layout but distributing the nodes of the smallest guild evenly. This solution makes the diagram more readable.

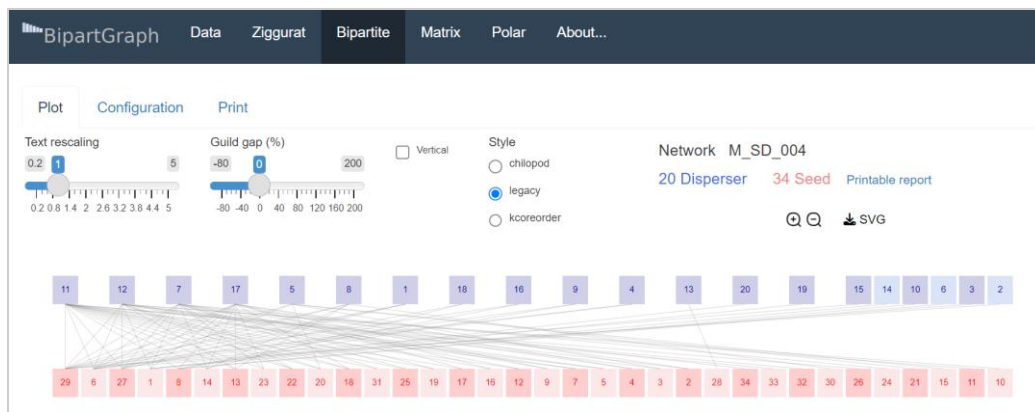


Figure 32: Legacy bipartite plot of disperser network M_SD_004

There are some details that you can deduce at a glance. The more generalist species lie on the plot's left side, and the pair *disperser 13/plant 28* is disconnected from the giant component of the network. We can get further visual information by tuning input parameters.

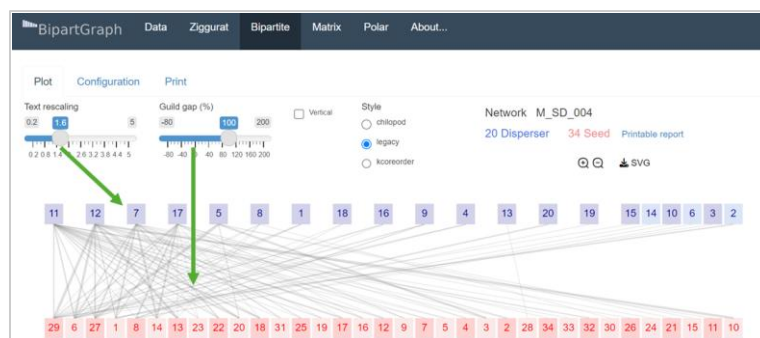


Figure 33: Legacy bipartite plot

The *kcorder* style reorders nodes according to their *k-shell* numbers (Fig. 34). It is easy to spot the different *k-shells*, with species of 4-shell being the most interconnected. On their left, we find 1-shell, on their right, 3-shell, 2-shell, and finally, species disconnected from the giant component. The only difference between *legacy* and *kcorder* styles is the order of the species, but information is still concentrated in the two lines of nodes.

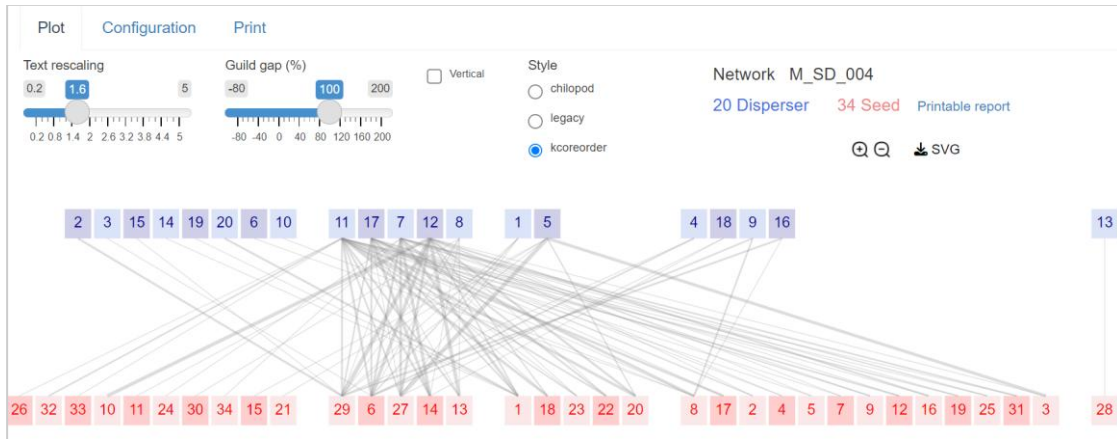


Figure 34: *kcorder* bipartite plot of disperser network M_SD_004

What if we could keep the whole idea of a bipartite plot but spread part of the visual information in two dimensions? That is the idea behind the third style, the *chilopod plot*. The name refers to the resemblance of this plot to a centipede. Nodes of 1-shell that we call tails in the ziggurat plot are moved away from the guild lines, and appear as short appendices. We apply the same grouping idea in ziggurat's fat tails: nodes with just one link tied to the same species of a higher shell are plotted together (Fig. 35).

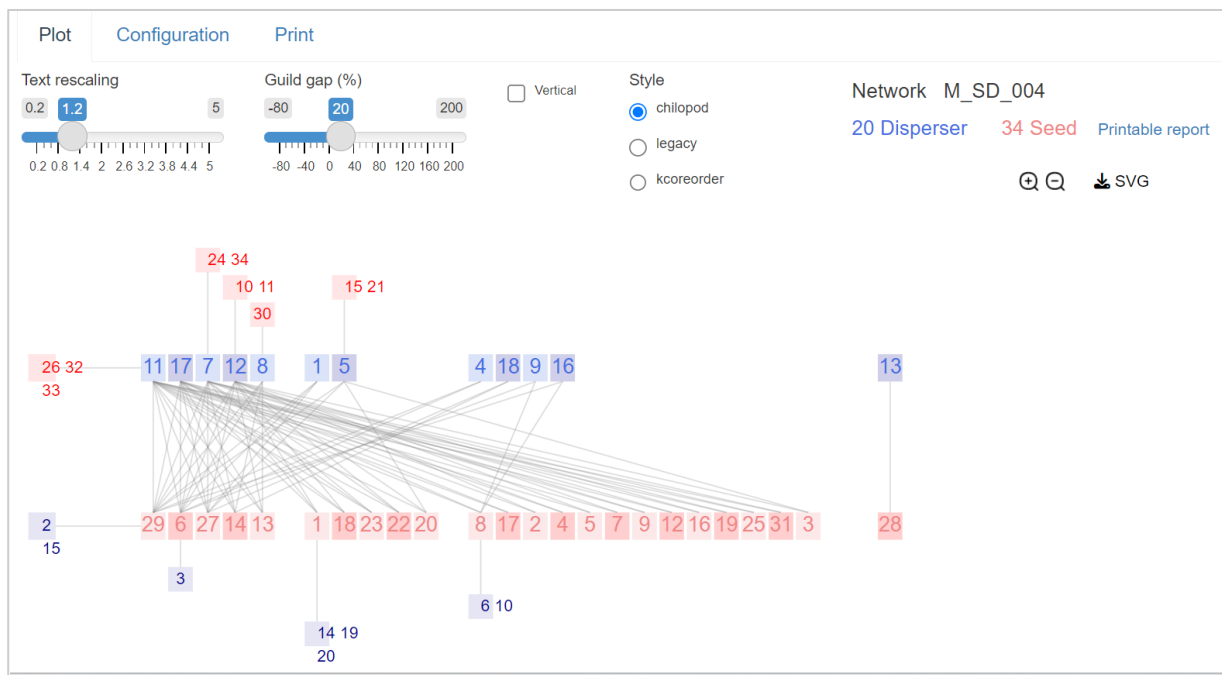


Figure 35: *chilopod* bipartite plot of disperser network M_SD_004

The bipartite plot offers the same customization and code generation options as the ziggurat plot.

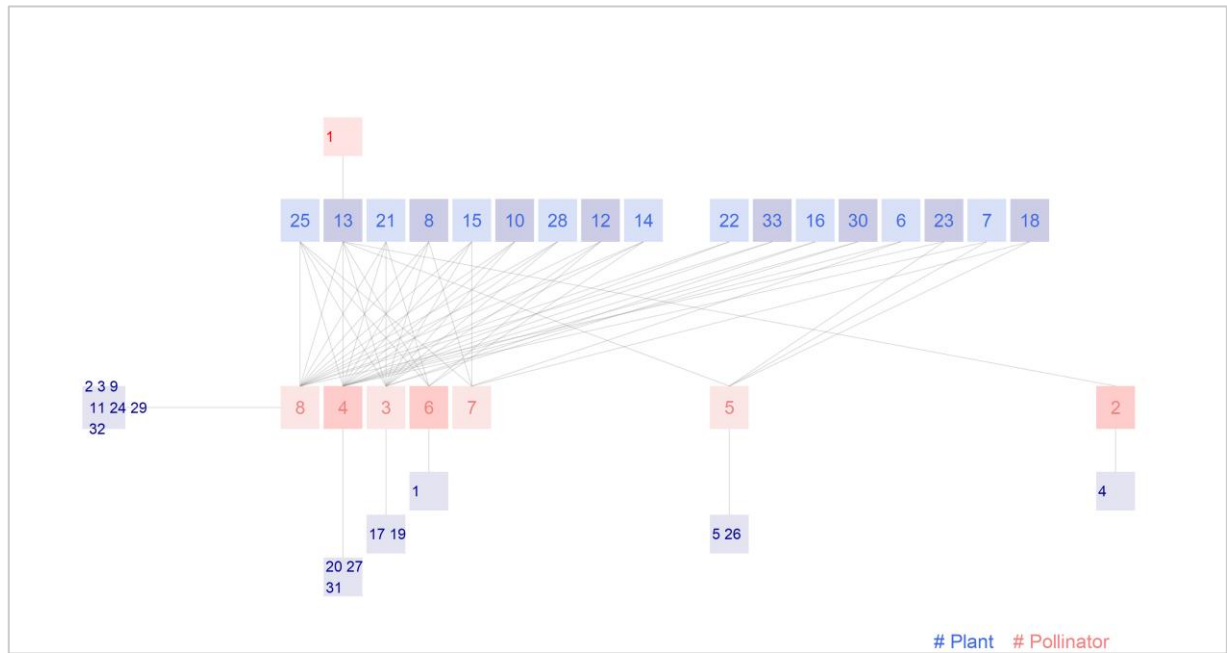


Figure 36: *Chilopod* bipartite plot of a mutualistic network in Sicily [9]

6 The matrix plot

The Matrix Plot is another traditional way of plotting bipartite networks. It is just the depiction of the interaction matrix, with species of guild A as columns and species of guild B as rows. For binary matrices, the cell i, j is filled with a solid color if there is a link between species i of guild A and species j of guild B. The interaction strength is usually encoded as a color gradient for weighted networks. Nodes are sorted by their degree in descending order. This plot is simple and helpful for spotting spatial properties, such as nestedness, but it is hard to discover chains of specialists.

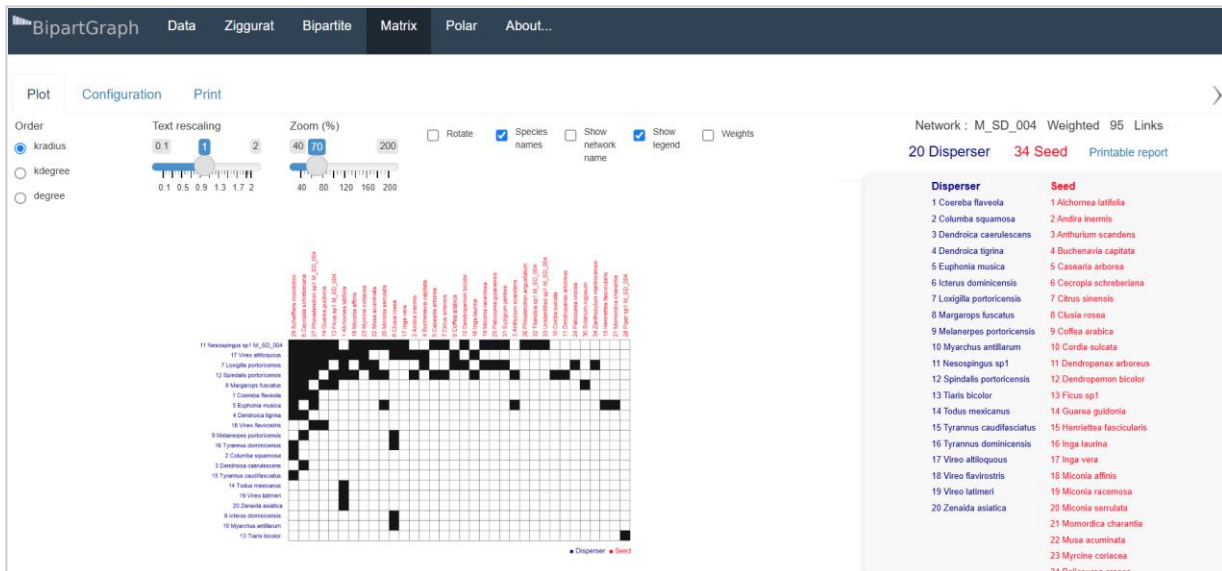
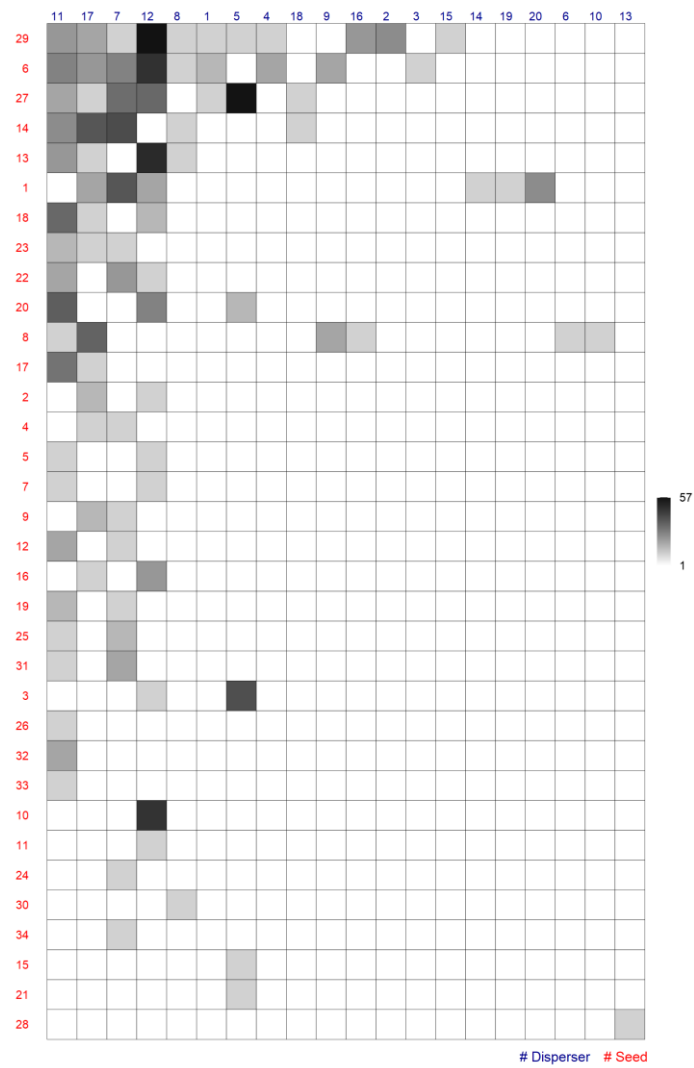


Figure 37: Matrix plot of disperser network M_SD_004

As in this example, the function allows you to flip the matrix to show it in landscape layout for convenience. Besides the traditional order by degree, BipartGraph offers order by *kradius* and *kdegree*. Species names may be removed, and this feature is of special interest when the network is extensive.

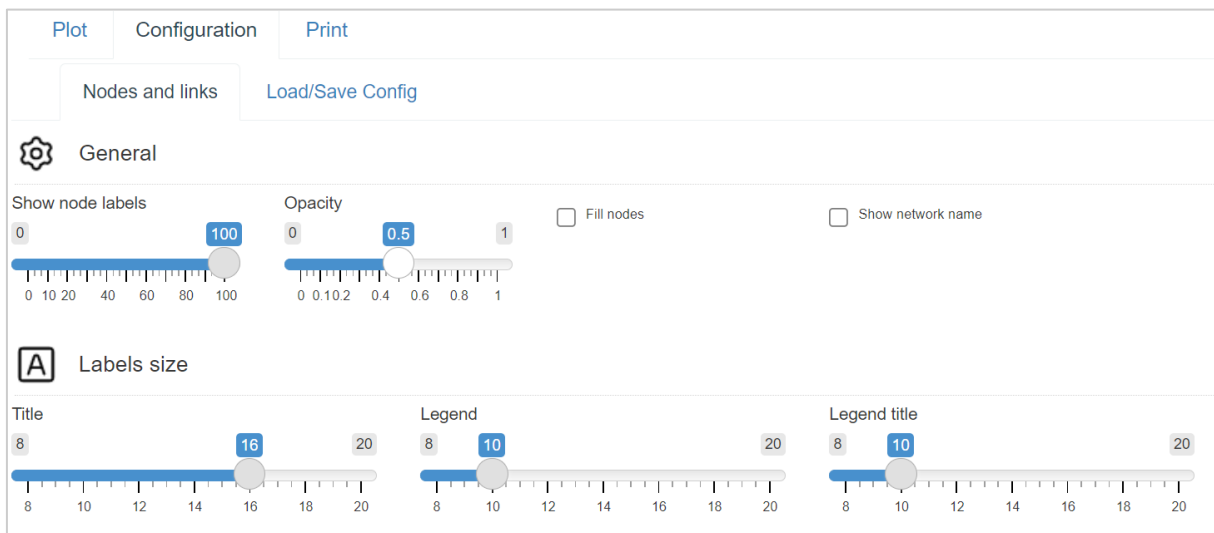
Figure 38: Rotated matrix plot of disperser network M_SD_004



7 The polar plot

The goal of the polar graph is to show species centrality and network compactness. It provides a quick overview of network distribution. It was inspired by the fingerprint-like graph, developed by Alvarez-Hamelin et al. [8] to plot extensive k -decomposed networks.

Species are plotted at k -radius from the center. The visualization algorithm assigns angles to reduce overlapping, with the condition that each guild lies inside one of the half planes. The area of each node is proportional to its k -degree, and color represents the k -shell. This visualization does not include links. The user may add the histograms of k -magnitudes, a handy option because they convey a wealth of structural information. Polar graphs are handy when comparing different networks side by side, even if they are of very different sizes. The configuration of a polar plot is relatively easy compared to the configuration of the ziggurat.



The algorithm sets the color. You may choose the level of transparency and whether you want to fill the pixels. The pixels box controls the size of the image on the screen and may help you adapt it to your equipment.

If you want to add the node number, use **Show node labels**. It tells the application how many node labels to show, from highest degree to lowest. If the network is big, the labels may spoil the plot. You can choose to fill the nodes and the level of opacity.

The **Labels size** control plays the same role as for the ziggurat plot.

Figure 39: Polar graph of a plant-pollinator in Garajonay (Spain), including labels and histograms. Compare with Fig. 26



Once the plot is displayed, you may download it or its generating code, just like the ziggurat plot. A significant difference is that the polar plot shown in the browser and the printed version are the same object, so screen resolution does not matter.

Useful hacks

In your `bipartgraph` directory, there is a file called `CONFIG.txt` with the following contents:

```
LANGUAGE;PORT;LabelA;LabelB;ColorGuildA1;ColorGuildA2;ColorGuildB1;ColorGuildB2;Wiki  
pediaSubdomain
```

```
"en";8080;"Plant";"Pollinator";"#4169E1";"#00008B";"#F08080";"#FF0000";"en"
```

You may change the default language, application TCP port, labels of guilds, and colors. Please make a copy before editing it. If you accidentally delete this file, the application starts with these default values, but you cannot modify them.

References

- [1] García-Algarra J, Pastor JM, Irioido JM, Galeano J., 2017. Ranking of critical species to preserve the functionality of mutualistic networks using the k-core decomposition. *PeerJ* 5:e3321 <https://doi.org/10.7717/peerj.3321>
- [2] García-Algarra J, Pastor JM, Mouronte ML, Galeano J, 2018. A structural approach to disentangle the visualization of bipartite biological networks. *Complexity*, 2018. <https://www.hindawi.com/journals/complexity/2018/6204947/>
- [3] Burkle, Laura A.; Marlin, John C.; Knight, Tiffany M. Plant–pollinator interactions over 120 years: loss of species, co-occurrence, and function, 2013. *Science*, vol. 339, no 6127, p. 1611–1615.
- [4] Bascompte, Jordi. Disentangling the web of life, 2009. *Science*, vol. 325, no 5939, p. 416–419.
- [5] Galetti, M. and Pizo, MA., 1996. Fruit-eating birds in a forest fragment in southeastern Brazil. *Ararajuba*, vol. 4, no. 2, p. 71–79.
- [6] Ingverser, T. T., 2006. *Plant–pollinator interactions on Jamaica and Dominica: The centrality, asymmetry and modularity of networks*. Master's thesis, University of Aarhus, Denmark.
- [7] Hadfield, Jarrod D., et al. 2013. A tale of two phylogenies: comparative analyses of ecological interactions. *The American Naturalist*, 2013, vol. 183, no 2, p. 174–187
- [8] Alvarez-Hamelin, J. Ignacio, et al. Large scale networks fingerprinting and visualization using the k-core decomposition. *Advances in neural information processing systems*, 2006, vol. 18, p. 41.
- [9] Polidori, C., Barletti, B. R., Quaranta, M., Ferrari, A., & De la Rúa, P. (2025). Bombus–plant interactions defined by bipartite network analysis in an underexplored Mediterranean island (Sicily). *Apidologie*, 56(1), 26.