

An Introduction and User Guide for mcaGUI

Last Modified 3/22/2011

Package Maintainer: Wade K. Copeland (wade@kingcopeland.com)

Authors:

Wade K. Copeland
Vandhana Krishnan
Daniel Beck
Matt Settles
James Foster
Kyu-Chul Cho
Mitch Day
Roxana Hickey
Ursel M.E. Schütte
Xia Zhou
Chris Williams
Larry J. Forney
Zaid Abdo

Additional Input and Suggestions:

John Verzani - Author of Original PMG Source Code

Table of Contents

Introduction	3
Definitions	3
Install Instructions.....	6
Description of the Data.....	7
Overview of mcaGUI layout	12
Methods for Reading in Data	13
OTUbase Reader.....	13
Proportion or Count Table Reader	16
Example Workflow	20
Abundance Tables	21
Histograms	23
Summary Statistics	26
Richness and Evenness Calculation.....	28
Clustering	30
Principle Components Analysis	32
References	35

Introduction

The Analysis of Microbial Community data stems from a larger project known as the Human Microbiome Project (HMP). One of the key aims of the HMP is to characterize and understand the microbial communities of different sites within the human body. These include for example the female urogenital tracts, oral cavity, naso-pharyngeal tract, and skin (5). The research presented in this paper furthers this effort by creating a graphic user interface (GUI) called mcaGUI that allows researchers to easily read in, analyze, and create graphical/empirical representations for microbial community data. The GUI uses an already developed R- GUI called “Poor Man’s GUI (PMG)” as the base layout (6). Additional functionality was added by using custom made and preprogrammed functions for graphic and empirical analysis. By using mcaGUI researchers are now able to read in data generated from 454 sequencing technology directly or pre-made abundance tables. With data read in an assortment graphical tools ranging from histograms to cluster diagrams give the user the ability to inspect data further. Researchers are also able to use analyses specific to studying microbial communities, such as richness and diversity estimation. The R-Programming environment provides a steep learning curve for those who wish to analyze microbial community data; mcaGUI provides a bridge that will enable accessibility to the power of the R-Programming environment for biologists studying microbial communities.

Definitions

Some definitions are needed to describe specific parts of the GUI that a non-programmer or non-biologist will not be familiar with. Table 1 outlines this terminology.

Table 1: Definitions

Term	Definition
Argument	An argument is input used to run a command. For example, to create a

	<p>histogram the argument “breaks” can be used to set the number of breaks in the histogram.</p>
Dialogue Window or Box	<p>A dialogue window or box is used to input user information. In the case of mcaGUI a dialogue window will allow users to specify arguments needed to perform analysis.</p>
Data Frame	<p>A data frame is similar to an Excel table for R. With a data frame users can conveniently access and manipulate data.</p>
Evenness	<p>A basic feature of biological communities is the distribution of abundance among species. There are many aspects of this distribution that can be measured, but the simplest feature is evenness. A community in which each species present is equally abundant has high evenness; a community in which the species differ widely in abundance has low evenness.</p>
Fluent R User	<p>The fluent R user is someone who knows the R-programming language and does not require a graphical user interface for R.</p>
Graphical User Interface (GUI)	<p>User interface based on graphics (icons, pictures and menus) instead of</p>

	text; uses a mouse as well as a keyboard as an input device (3).
Human Microbiome Project (HMP)	One of the key aims of the Human Microbiome Project (HMP) is to characterize and understand the microbial communities of different sites within the human body. These include for example the female urogenital tracts, oral cavity, naso-pharyngeal tract, and skin (5).
Operating System (OS)	An Operating System is a computer program that manages the resources of a computer. It accepts keyboard or mouse inputs from users and displays the results of the actions and allows the user to run applications, or communicate with other computers via networked connections (1).
Object	An object holds information in the R-Programming environment. OTUbase creates the OTUset <i>object</i> , which stores information about data that is read in.
Richness	This is an estimate of the number of species in a given area (9).
R-Package	The R-programming environment is completely open source and free to

	the public as a powerful statistical programming language. People create extensions to the capability of R by creating packages that perform specific tasks. For example, mcaGUI is an R- Package that creates a graphical user interface (GUI) to analyze microbial community data.
Variable	Similar to an object, a variable stores information such as a number or string, For example the variable X might store the number three.

Install Instructions

Currently mcaGUI can be installed on multiple operating systems (OS). This section discusses how to install mcaGUI on Windows OS, Machintosh OS and the Ubuntu distribution of Linux.

A. WINDOWS AND UBUNTU LINUX INSTALLATION INSTRUCTIONS

1. Download and install the most recent version of R for the Windows OS at <http://www.r-project.org/> (at the time of writing R-2.12.0).
2. Download and install GTK from <http://downloads.sourceforge.net/gtk-win/gtk2-runtime-2.22.0-2010-10-21-ash.exe>.
3. Open the R program on your computer and type:


```
source("http://bioconductor.org/biocLite.R")
biocLite("mcaGUI")
```

B. MACHINTOSH OSX INSTALLATION INSTRUCTIONS

1. Download and install the most recent version of R for the Macintosh OS at <http://www.r-project.org/> (at the time of writing R-2.12.0).
2. Open the R program on your computer.
3. Download and install GTK from http://r.research.att.com/libs/GTK_2.18.5-X11.pkg.
4. Open the most recent version of R (at the time of writing R-2.12.0)
5. Back in the R window type:

```
source("http://bioconductor.org/biocLite.R")  
biocLite("mcaGUI")
```

C. INSTALL INSTRUCTIONS FOR "mcaGUI" R-PACKAGE UPDATES

1. Open the most recent version of R (at the time of writing R-2.12.0)
2. In the R command window type:

```
source("http://bioconductor.org/biocLite.R")  
biocLite("mcaGUI")
```

Description of the Data

Sogin 2006 Data Set

To illustrate the use of this package, examples will be based on the data obtained by Sogin et. al in 2006 (15). The reason for using this data set is two-fold. First, it is already published data used as a reference in other programs such as MOTHUR, and the R-Packages OTUbase and Vegan. Second, research presented in this paper uses 454 sequencing technology. For the sake of brevity we refer to this data set as "Sogin 2006" or "Sogin 2006 data" for the remainder of this document.

The Sogin 2006 data can be accessed at any time by going to the R Library folder or directory and navigating to (.../mcaGUI/extdata/Sogin_2006).

Data Generated From MOTUR or Similar Programs

The GUI uses several types of data generated from other programs such as MOTUR (11), DOTHUR (9), Sons (12), Treeclimber (13), s-libshuff (10) and Unifrac (7). These programs are built for analyzing data generated from 454 sequencing, but are not part of the R-programming environment (and thus cannot take advantage of the unique functionality). For information on how the data below was generated from MOTUR visit

http://www.mothur.org/wiki/Sogin_data_analysis.

OTU file - This file has the ending *.list* and contains sequences that cluster together to form an operational taxonomic unit (OTU). *Table 2* shows part of this file from the Sogin 2006 data. The first column is the distance measure (ie. Manhattan, Chao,... ect.) used when clustering. The second column is the total number of clusters and the remaining columns are the sequences that were clustered together.

Table 2: Example of the OTU file from Sogin 2006.

unique	466	D4WT9DQ13H7U2V	D4WT9DQ02A6U4N	D4WT9DQ16JWOQH	D4WT9DQ07D7G6D
0.01	461	D4WT9DQ13H7U2V	D4WT9DQ02A6U4N	D4WT9DQ16JWOQH	D4WT9DQ07D7G6D
0.02	423	D4WT9DQ13H8UR7,D4WT9I	D4WT9DQ16JWOQH,D4WT9DC	D4WT9DQ07D7G6D	D4WT9DQ09FV1II
0.03	399	D4WT9DQ13H8UR7,D4WT9I	D4WT9DQ16JWOQH,D4WT9DC	D4WT9DQ07D7G6D	D4WT9DQ09FV1II
0.04	396	D4WT9DQ13H8UR7,D4WT9I	D4WT9DQ16JWOQH,D4WT9DC	D4WT9DQ07D7G6D	D4WT9DQ09FV1II
0.05	378	D4WT9DQ06DQCHX,D4WT9I	D4WT9DQ16JWOQH,D4WT9DC	D4WT9DQ07D7G6D	D4WT9DQ09FV1II
0.06	360	D4WT9DQ16JREN2,D4WT9I	D4WT9DQ07D7G6D,D4WT9DQ	D4WT9DQ09FV1II	D4WT9DQ15JGG55
0.07	349	D4WT9DQ13H9N96,D4WT9I	D4WT9DQ07D7G6D,D4WT9DQ	D4WT9DQ09FV1II	D4WT9DQ15JGG55
0.08	335	D4WT9DQ13H9N96,D4WT9I	D4WT9DQ07D7G6D,D4WT9DQ	D4WT9DQ09FV1II	D27LU0R16JRNQG,D4V
0.09	330	D4WT9DQ08ET23T,D4WT9I	D4WT9DQ07D7G6D,D4WT9DQ	D4WT9DQ09FV1II	D27LU0R16JRNQG,D4V
0.1	324	D4WT9DQ08ET23T,D4WT9I	D4WT9DQ07D7G6D,D4WT9DQ	D4WT9DQ09FV1II	D27LU0R16JRNQG,D4V

SAMPLE file – This file has the ending *.groups*. In the process used by MOTUR or similar programs, sequences are separated into groups based on clustering and that information is stored here. This file has two columns: The first column contains the sequence names and the second column contains the groups they were assigned to. *Table 3* shows part of this file from the Sogin 2006 data.

Table 3: Partial output of sample data file from Sogin 2006.

D27LU0R16JQ6Y5	FS396
D27LU0R13H56WA	FS396
D27LU0R01AKRMM	FS312
D27LU0R07D3VKK	FS312
D4WT9DQ16JSRXV	138
D4WT9DQ10F8CWU	112R
D27LU0R12HK6IG	FS396
D27LU0R16J0ADB	FS396
D4WT9DQ01AT1SC	FS312
D4WT9DQ15JFMAD	138
D27LU0R01AINLQ	FS312
D4WT9DQ14IRRYQ	137
D4WT9DQ01ARGSB	FS312
D27LU0R08EWXPJ	FS312
D27LU0R09FJFBP	FS396

FASTA file – This file has the ending *.fasta*. This is a formatted text file that contains either nucleotide sequences or peptide sequences. The base pairs are represented by single letter codes on the first line and the second line contains the actual sequence. *Table 4* shows part of this file from the Sogin 2006 data.

Table 4: Partial output of the .fasta file from Sogin 2006.

>D4WT9DQ06DVGFR
TGCCTTTGACATCCTCGGAACGGTCCGGAACGGACTGGTGCCTTCGGGAACCGAGAGAC
>D4WT9DQ05C6YNI
TGGACTTGACATGTTAGTGTAACCCGATGAAAGTCGGGCCCTCTGCGAGCTTGCTCAAAGACACTATCAC
>D4WT9DQ12HNQY2
CGGGCTTGAAGTGCAAGCGACAACCCATGAAAGTGGATTTCCGCAAGGACGCTTGTAG
>D4WT9DQ01AP0UQ
TGGTCTTGACATCCCAGGAATCTTAGAAAATAAGAGAGTGCCTCATTAGAGGAGCCTGGTGAC
>D4WT9DQ09FLPTJ
AGGACTTGACATCCAGAGAAGTCCGGCAGAGATGCCTTGGTGCCTTCGGGAAGTCTGTGAC
>D27LU0R02A82DK
ATCCCTTGACATCCTGCGAACTTTCTAGAGATAGATTGGTGCCTTCGGAAACGCAGTGAC
>D4WT9DQ04COZYL
TAGCCTTGACATACTACAGAATTTGACAGAGATGTTGGAGTGCCTTCGGGAGCTGTAATAC
>D27LU0R14ITQ9P
TAGCCTTGACATACTACAGAATTTGACAGAGATGTTGGAGTGCCTTCGGGAGCTGTAATAC
>D27LU0R10GCZQJ
TAGCCTTGACATACTACAGAATTTGACAGAGATGTTGAAGTGCCTTCGGGAGCTGTAATAC

QUALITY file - This file has the ending *.qual*. It is generated by the 454 sequencer and contains the quality of each sequence read.

SAMPLE Meta-Data file – This file is usually a *.txt* file and contains meta-data such as the sample ID, site, investigator, ... ect. *Table 5* shows a sample file for the Sogin 2006 data.

Table 5: Sample Meta-Data file from Sogin 2006.

# Sample_ID: Sample ID						
# Site: Site						
# Lat_N: Latitude N (degrees)						
# Long_W: Longitude W (degrees)						
# Depth: Depth (m)						
# Temperature: Temperature (C)						
# Cells: Cells per ml of water						
Sample_ID	Site	Lat_N	Long_W	Depth	Temperature	Cells
53R	Labrador seawater	58.3	âˆˆ29.133	1,400	3.5	6.4 Å– 104
55R	Oxygen minimum	58.3	âˆˆ29.133	500	7.1	1.8 Å– 105
112R	Lower deep water	50.4	âˆˆ25.000	4,121	2.3	3.9 Å– 104
115R	Oxygen minimum	50.4	âˆˆ25.000	550	7	1.5 Å– 105
137	Labrador seawater	60.9	âˆˆ38.516	1,710	3	3.3 Å– 104
138	Labrador seawater	60.9	âˆˆ38.516	710	3.5	5.2 Å– 104
FS312	Bag City	45.916	âˆˆ129.983	1,529	31.2	1.2 Å– 105
FS396	Marker 52	45.943	âˆˆ129.985	1,537	24.4	1.6 Å– 105

FEATURE (OR ASSIGNMENT) DATA file – If the data understudy were classified using the RDP classifier then this file will have the ending *.rep.fix.tax*. Otherwise this file will have a name similar to assignmentADF. This file contains all sequences along with their assignment to the Phylum, Order, Family and Genus levels. *Table 6* shows part of the assignment/feature file used in Sogin 2006.

Table 6: Partial output of the feature data file from Sogin 2006.

D4WT9DQ13H7U2V	Bacteria	domain	1	Proteobacteria	phylum	0.99	Alphaproteobacteria	class	0.99	Rickettsiales	order	0.99	SAR11	family
D4WT9DQ16IWOQH	Bacteria	domain	0.9	Proteobacteria	phylum	0.48	Alphaproteobacteria	class	0.27	Rickettsiales	order	0.06	SAR11	family
D4WT9DQ07D7G6D	Bacteria	domain	0.99	Firmicutes	phylum	0.48	Bacilli	class	0.22	Bacillales	order	0.15	Bacillaceae	family
D4WT9DQ09FV1II	Bacteria	domain	0.94	Proteobacteria	phylum	0.68	Alphaproteobacteria	class	0.61	Rickettsiales	order	0.61	SAR11	family
D4WT9DQ15JGG55	Bacteria	domain	1	Proteobacteria	phylum	0.92	Alphaproteobacteria	class	0.87	Rickettsiales	order	0.85	SAR11	family
D4WT9DQ08EZ20A	Bacteria	domain	0.98	Proteobacteria	phylum	0.57	Alphaproteobacteria	class	0.53	Rickettsiales	order	0.48	SAR11	family
D4WT9DQ14IP5ZS	Bacteria	domain	0.99	Proteobacteria	phylum	0.87	Alphaproteobacteria	class	0.8	Rickettsiales	order	0.63	SAR11	family
D4WT9DQ11GYABG	Bacteria	domain	1	Proteobacteria	phylum	0.72	Alphaproteobacteria	class	0.59	Rickettsiales	order	0.52	SAR11	family
D4WT9DQ11GVRYO	Bacteria	domain	1	Proteobacteria	phylum	0.76	Alphaproteobacteria	class	0.68	Rickettsiales	order	0.55	SAR11	family
D4WT9DQ09FMFMM	Bacteria	domain	1	Proteobacteria	phylum	0.83	Alphaproteobacteria	class	0.77	Rickettsiales	order	0.68	SAR11	family
D4WT9DQ15I8CDI	Bacteria	domain	1	Proteobacteria	phylum	0.99	Alphaproteobacteria	class	0.99	Rickettsiales	order	0.96	SAR11	family
D4WT9DQ12HIKGU	Bacteria	domain	1	Proteobacteria	phylum	0.94	Alphaproteobacteria	class	0.93	Rickettsiales	order	0.91	SAR11	family

Abundance Tables - Table 7 shows an example of what an abundance table looks like from the Sogin 2006 data. The columns are from the sample meta-data and the rows are OTU names (usually in terms of Phylum, Class or Order). Abundance tables are usually generated by using the sample-meta data and assignment/feature data.

Table 7: Abundance table generated using the Sogin 2006 data.

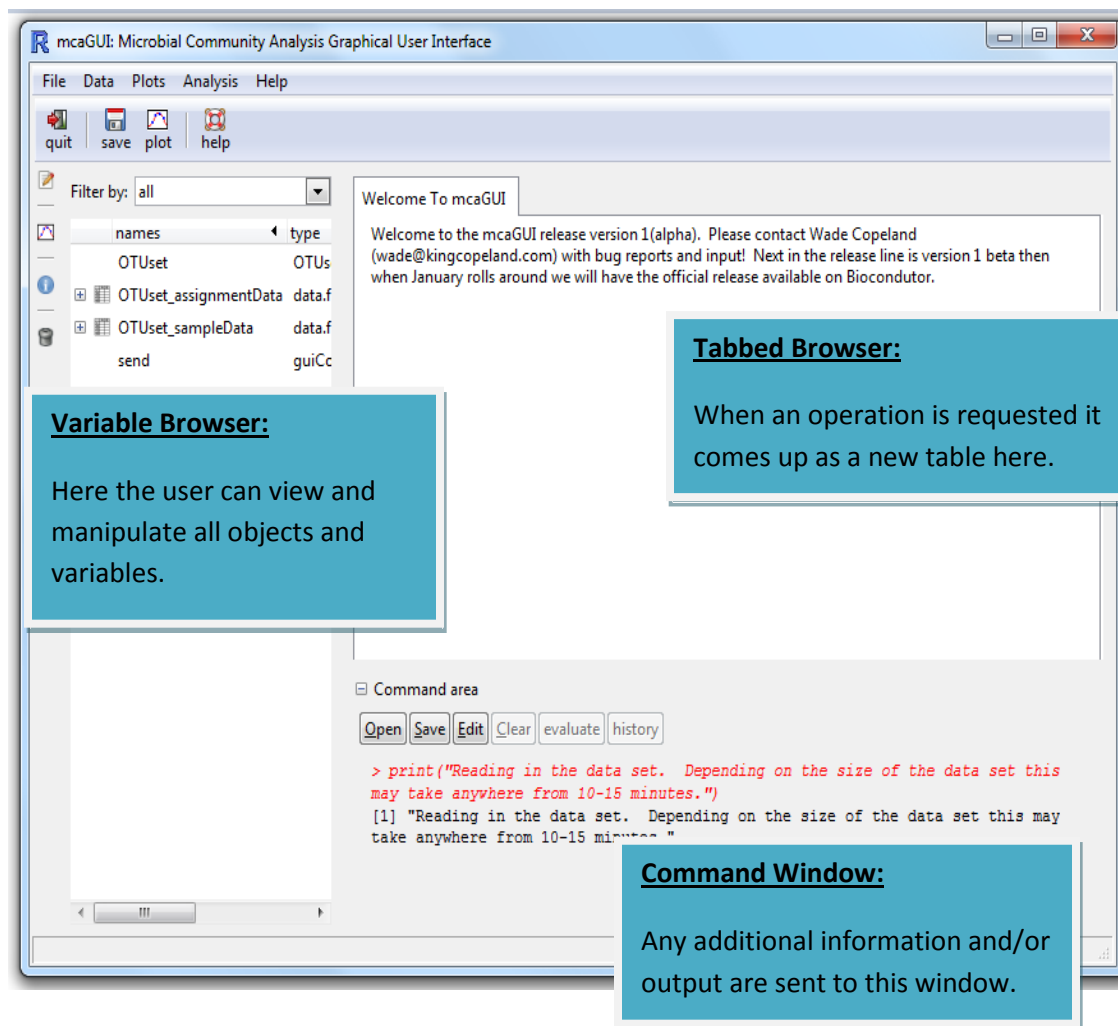
	Lower deep water	Oxygen minimum	Labrador seawater	Labrador seawater
Acidobacteria	0.013333333	0.013333333	0.015151515	0.015384615
Actinobacteria	0.173333333	0.146666667	0.121212121	0.123076923
Aquificae	0	0	0	0
Bacteroidetes	0.026666667	0.066666667	0.015151515	0.015384615
Caldiserica	0	0	0	0.015384615
Chlamydiae	0	0	0	0
Chlorobi	0	0	0	0
Chloroflexi	0.066666667	0.013333333	0.015151515	0.015384615

Overview of the mcaGUI layout

Layout of the mcaGUI is divided into three parts as shown in *Figure 1*. The left side of the screen contains the variable browser. This window shows all created items that can be manipulated such as variables or objects. On the top right is the tabbed browser. For most operations in mcaGUI a new tab will be opened where the user can input some specific

arguments. On the bottom right hand side of the screen is the command window. This window provides information relevant to the performed analysis.

Figure 1: Basic layout of mcaGUI.



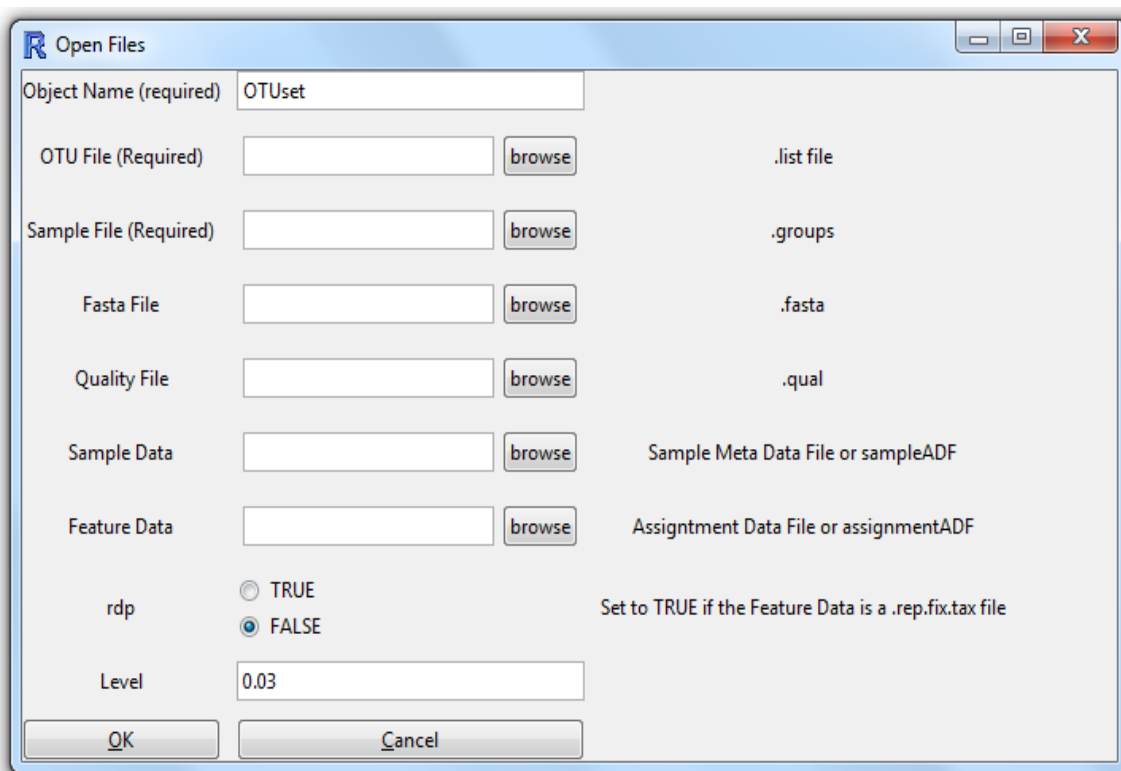
Methods for Reading in Data

There are currently two primary ways for users to read in their data. The first is using the OTUbase reader. This allows the user maximum access to information needed to customize and view their data. The second is reading an abundance tables directly. This option is more limited in how the abundance table can be customized, but requires fewer files to read.

Reading in the Data using the OTUbase Reader

OTUbase reader is the portion of the GUI that reads in data generated from programs such as MOTHUR (11). In the GUI window go to **file -> Read OTUbase** (this notation means to first navigate to the file menu and then click “Read OTUbase”). *Figure 2* shows what the dialogue window looks like after the previous steps are followed:

Figure 2: OTUbase reader dialogue window.



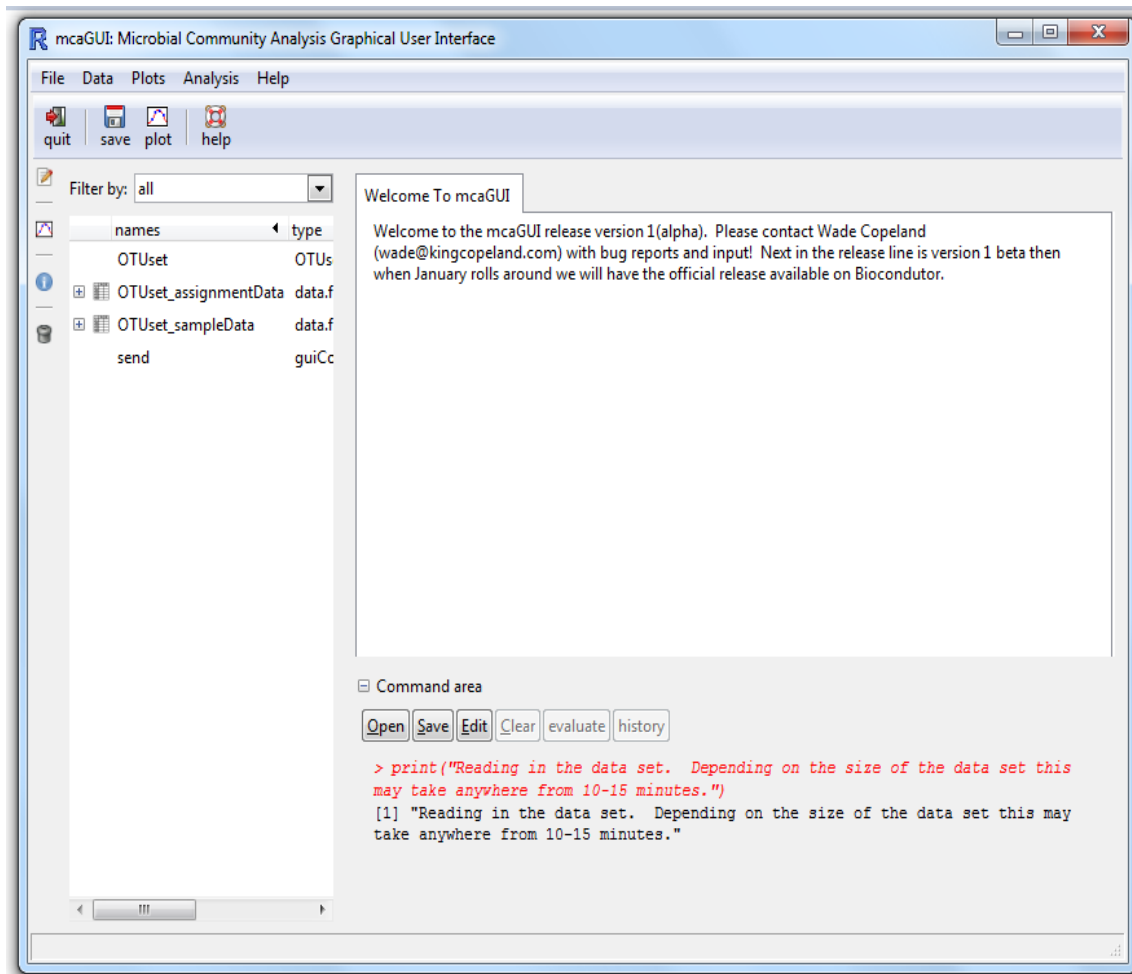
The “Object Name” field names to the object being read in and can be set to any value. For the purposes of this demonstration, the default value “OTUset” is used. *Figure 3* shows the filled in dialogue box after the Sogin 2006 files are browsed to (.../mcaGUI/extdata/Sogin_2006). Since the feature data has the ending .rep.fix.tax, RDP is set to TRUE. The default value of level is used in this case but any level specified in the OTU file can be used. *Warning: The files must be located in the same directory or folder.*

Figure 3: Filled in values using the Sogin 2006 data.

Object Name (required)	OTUset	
OTU File (Required)	006\sogin.unique.filter.fn.list	browse .list file
Sample File (Required)	ata\Sogin_2006\sogin.groups	browse .groups
Fasta File	tdata\Sogin_2006\sogin.fasta	browse .fasta
Quality File		browse .qual
Sample Data	in_2006\sample_metadata.txt	browse Sample Meta Data File or sampleADF
Feature Data	\\Sogin_2006\sogin.rep.fix.tax	browse Assignment Data File or assignmentADF
rdp	<input checked="" type="radio"/> TRUE <input type="radio"/> FALSE	Set to TRUE if the Feature Data is a .rep.fix.tax file
Level	0.03	

After the needed files shown in *Figure 3* have been filled in press **OK**. Check the mcaGUI command window to verify that the data has been read in successfully or if any errors have occurred. Once the data is read in use the variable browser to view the newly created object. In the mcaGUI window go to the variable browser on the left hand side of the screen and click **filter by -> All** as shown in *figure 4*.

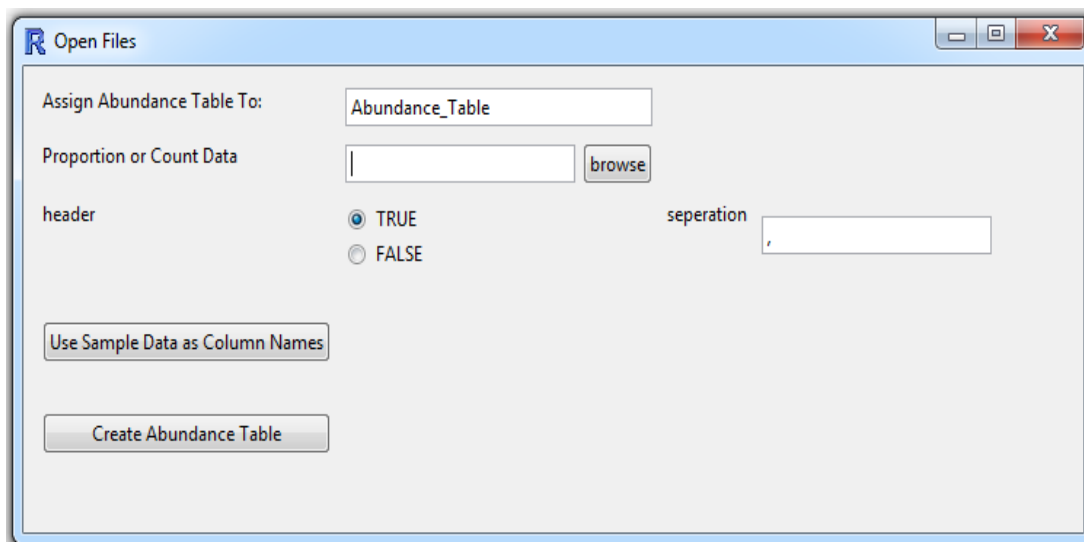
Figure 4: Selecting “filter by -> all” allows the user to view the newly created OTUset object.



Reading in the Data using the Proportion or Count Reader

In the case that a user does not have all the files needed to use the OTUbase reader there is the option to read in abundance tables directly. To start loading the data, go to **File -> Read Proportion or Count Data**. Figure 5 shows the dialogue window after doing this.

Figure 5: Dialogue box for read in proportion or count data.

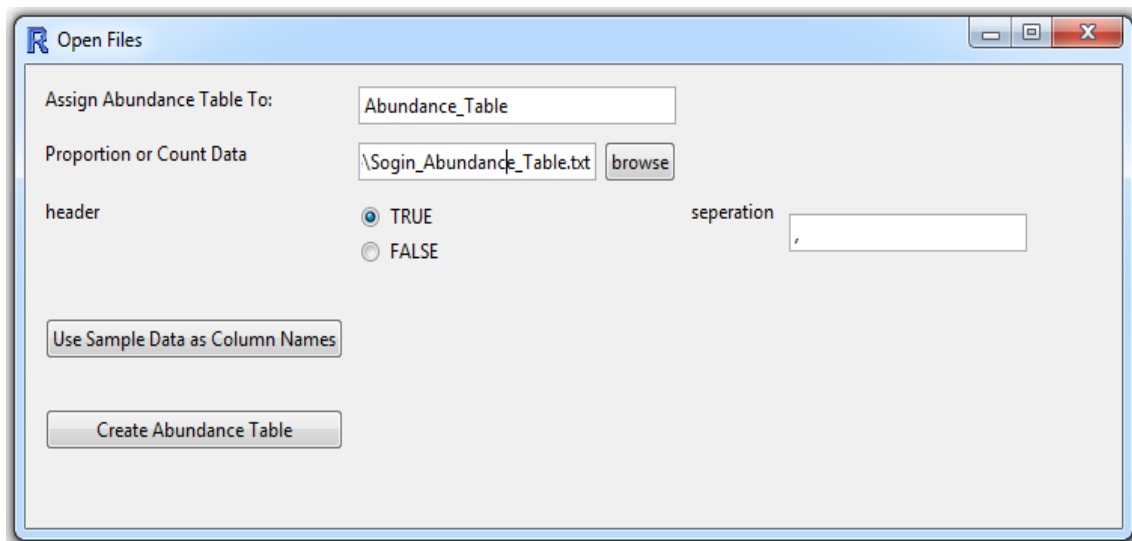


The screenshot shows a standard Windows-style dialog box titled "Open Files" with the R logo in the top-left corner. The dialog contains the following elements:

- Assign Abundance Table To:** A text input field containing the text "Abundance_Table".
- Proportion or Count Data:** An empty text input field followed by a "browse" button.
- header:** Two radio buttons, "TRUE" (which is selected) and "FALSE".
- seperation:** A text input field containing a comma character ",".
- Use Sample Data as Column Names:** A button.
- Create Abundance Table:** A button.

The first argument is "Assign Abundance Table To:". The default value is already filled in but can be changed to any value. The first file needed to be read in is the abundance table. For this example I will use *Sogin_Abundance_Data.txt* (.../mcaGUI/extdata/Sogin_2006). Figure 6 shows what the dialogue window looks like after this file has been navigated to.

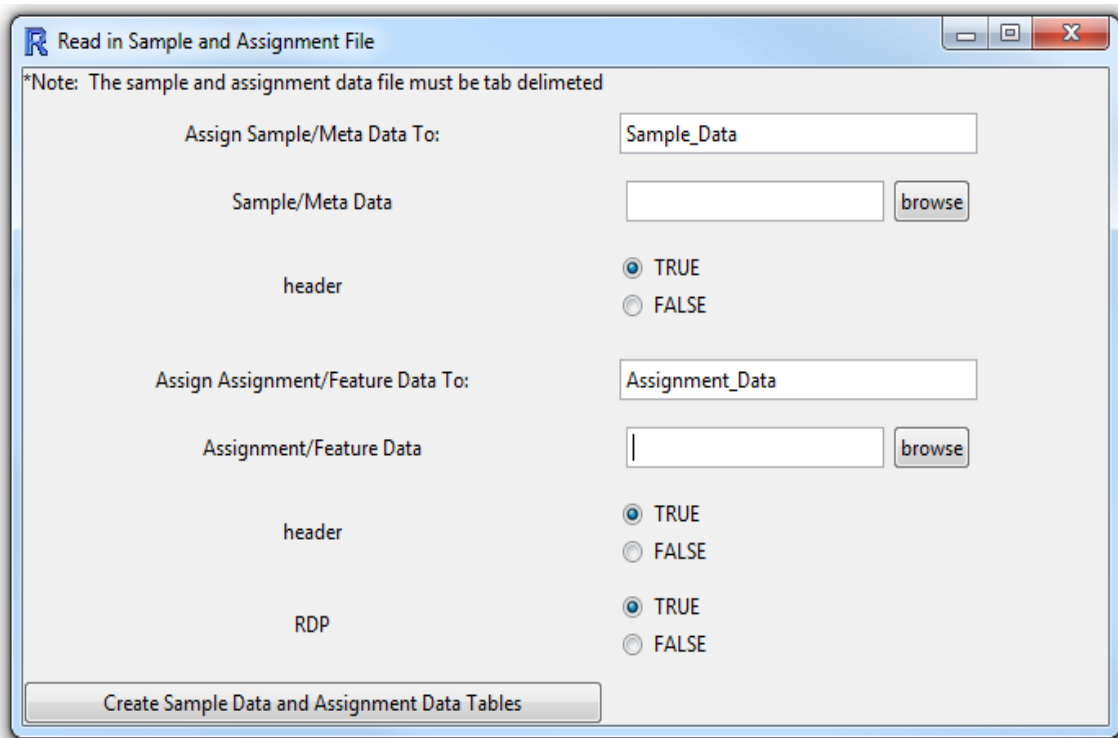
Figure 6: Proportion or count reader dialogue box with filled in file browser box for the abundance table.



This screenshot is identical to Figure 5, but the "Proportion or Count Data" text box now contains the file path "\\Sogin_Abundance_Table.txt".

In *Figure 6* a few additional options can also be selected. Header true means that the first row of the file contains variable names and not data. Separation in this case tells the reader that the abundance table file being read in is delimited with commas. The user can also specify tab delimited with `\t`. At this point the abundance table can be read directly by clicking the **Create Abundance Table** button. There is one additional button called **Use Sample Data as Column Names**. *Figure 7* shows an image of the dialogue box are clicking this button.

Figure 7: Dialogue box after clicking the “Use Sample Data as Column Names” button.



In this window there are two files that can be read in. The first is the sample meta-data and the second is the assignment/feature data. One or both of these files can be read in. *Figure 8* shows the window with all the values filled in for this example.

Figure 8: Update sample meta-data dialogue box with filled in values.

Read in Sample and Assignment File

*Note: The sample and assignment data file must be tab delimited

Assign Sample/Meta Data To: Sample_Data

Sample/Meta Data in_2006\sample_metadata.txt browse

header TRUE FALSE

Assign Assignment/Feature Data To: Assignment_Data

Assignment/Feature Data \\Sogin_2006\sogin.rep.fix.tax browse

header TRUE FALSE

RDP TRUE FALSE

Create Sample Data and Assignment Data Tables

Figure 8 also shows some additional options. Header defined to be true as before implies that the first row of the file is variable names. RDP defined to be true tells the reader that the assignment/feature data was created using an RDP classifier. If the assignment/feature data is not RDP set this option to false. Once these files are read in they create data frames in the variable browser. After clicking the **Create Sample Data and Assignment Data Tables** button the original dialogue box is brought back up with additional options. Figure 9 shows the updated dialogue box for reading in the abundance table.

Figure 9: Updated dialogue box after clicking the “Create Sample Data and Assignment Data Tables” button.

With the sample meta-data read in, click the **Use Sample Data as Column Names** button to specify a column to use from the sample meta-data. *Warning: The column names of the original abundance table must be in the same order as the rows of the sample meta-data.* After selecting a column to use, click the **Create Abundance Table** button.

Example Workflow

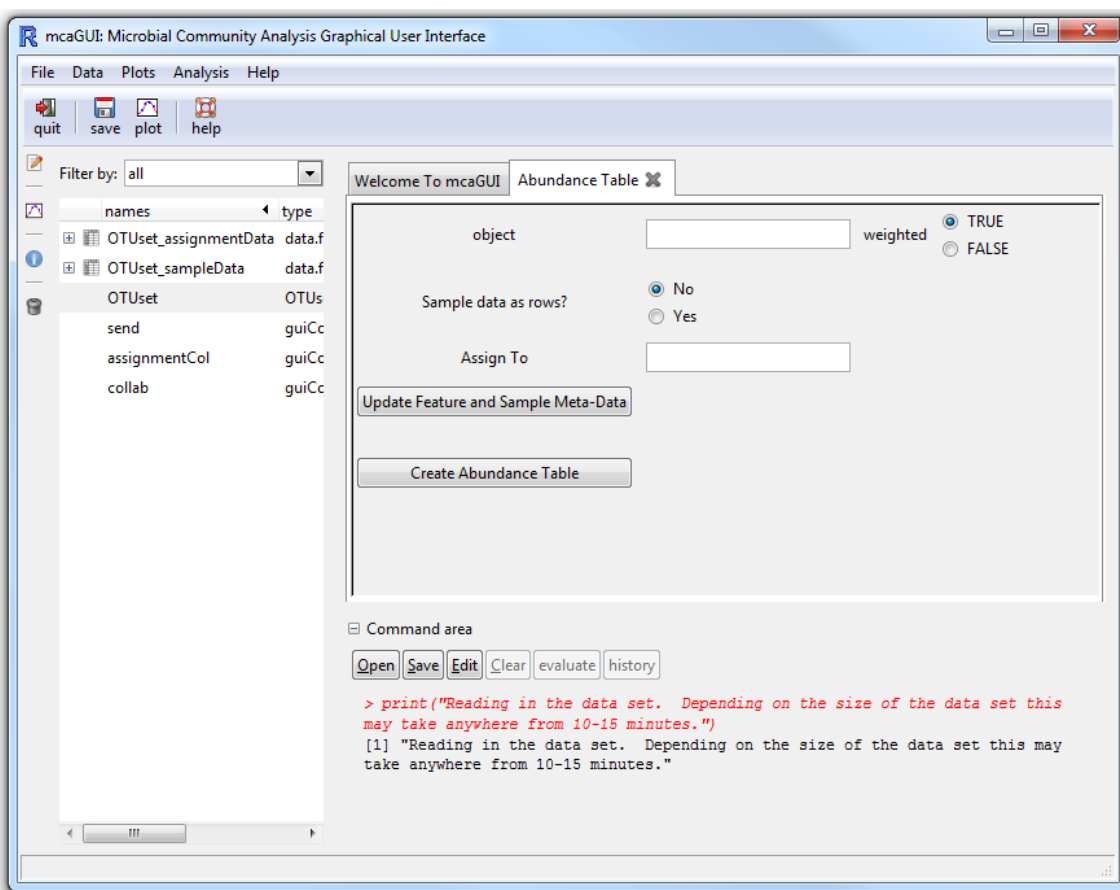
After the data is read in using either OTUbase reader or the proportion or count table reader, there are multiple workflows available for analyzing data. Below is an outline of the example discussed in this section.

1. Create an Abundance Table as either count or proportion data.
2. Use a histogram to view the distribution of the data.
3. Obtain basic summary statistics.
4. Use the count abundance table to calculate richness and evenness.
5. Perform a cluster analysis.
6. Perform PCA analysis and view the results in both 2 and 3 dimensional space.

1. Create Custom Abundance Tables (OTUbase reader only).

Creating an abundance table from the data is the first and most important step if OTUbase reader was used to read in the data. To create abundance tables go to **data -> Abundance Table**. Figure 10 shows the resulting dialogue window.

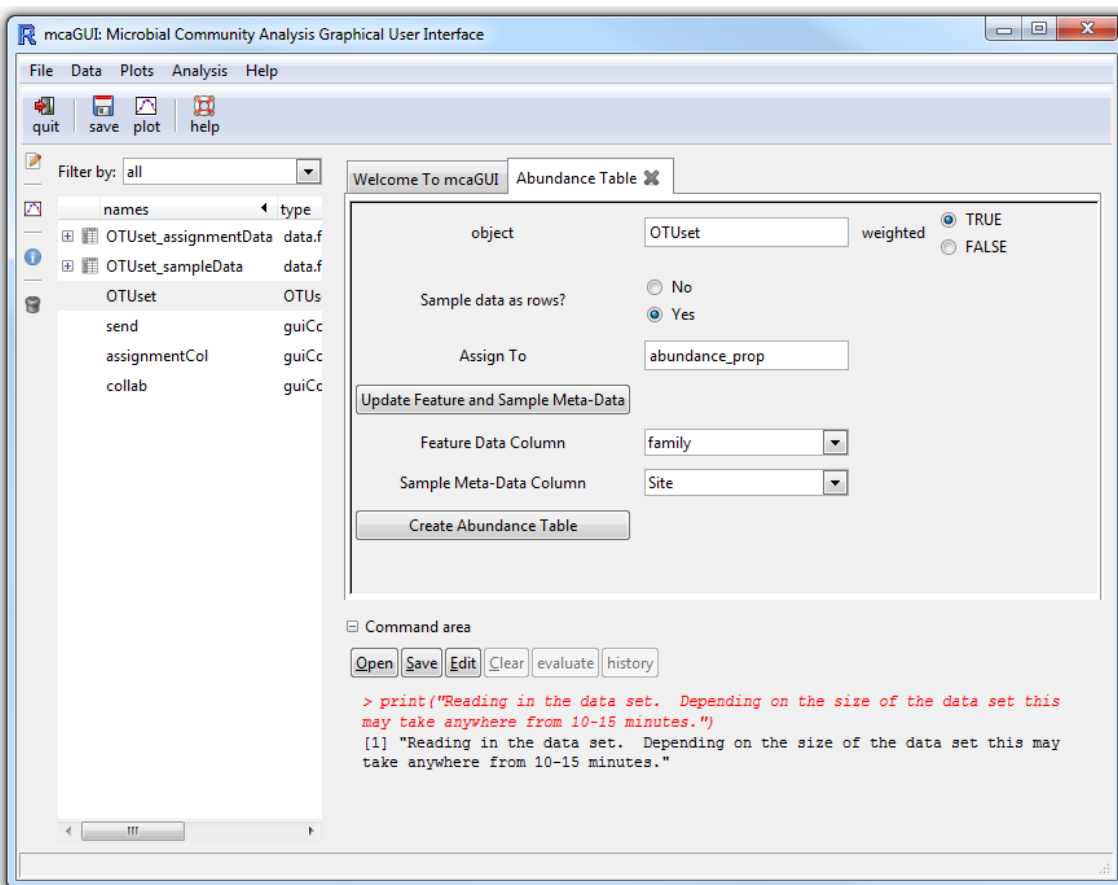
Figure 10: Dialogue box for creating an abundance table.



There are four items that need to be described in the resulting dialogue box. The first is "object"; this is the name specified when reading in the data using the OTUbase reader (in this example it is called "OTUset"). To put the object name in the object field the user can click and drag "OTUset" from the variable browser to the space provided or type it in. The next argument specifies if the abundance table should be weighted. If true, the abundance table will be proportions; if false it will be count data. Specifying sample data as rows will put the sample

meta-data (such as Sample ID) on the rows. If false it will put sample meta-data as the columns. The “assignto” argument specifies the name of the new abundance table. Since the sample meta-data and assignment data were read in click the **Update Feature and Sample Meta Data** button. This option allows the user to customize their abundance table by using a row and column from the assignment/feature data and sample-meta data. *Figure 11* shows what the dialogue box looks like after being filled in.

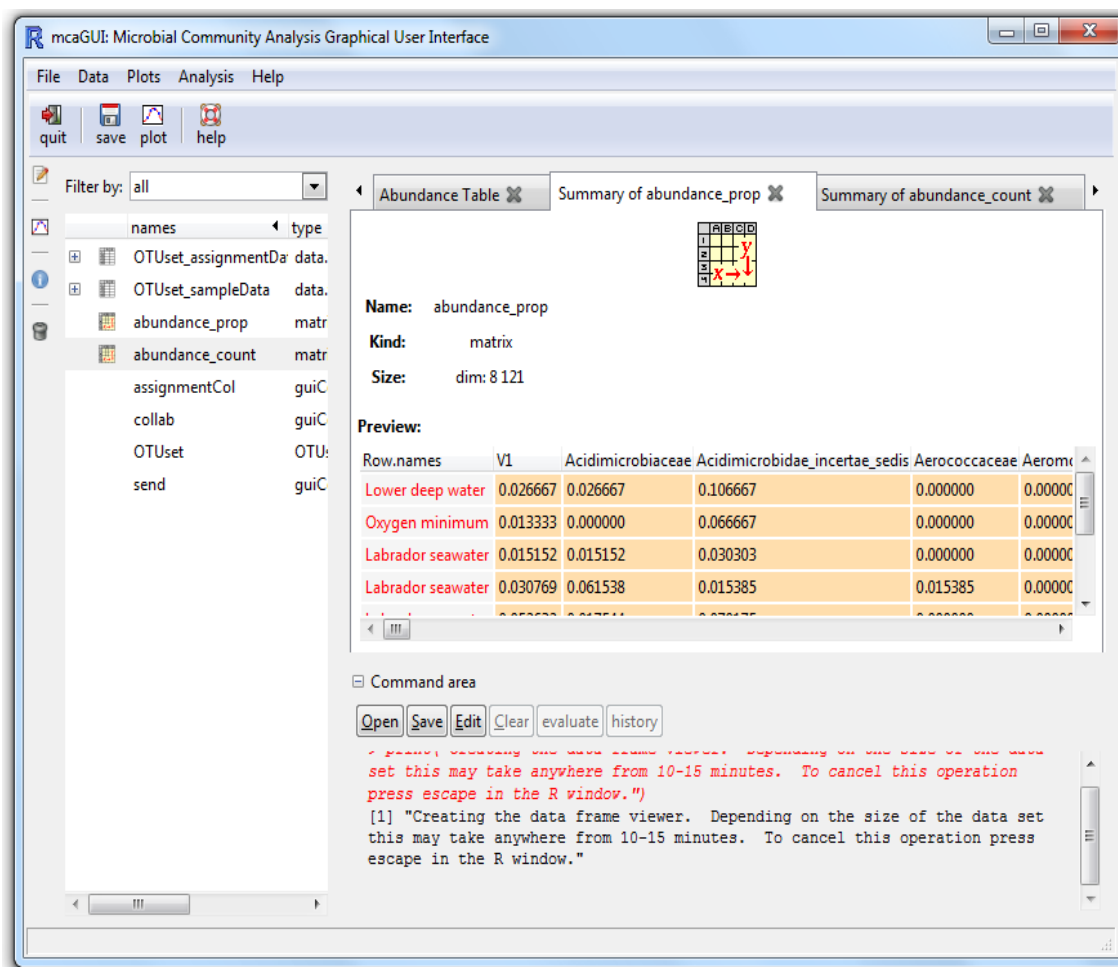
Figure 11: Filled dialogue boxes for reading in an abundance table.



For this demonstration the new abundance table is named “abundance_prop”. Following the same steps create one more abundance table called “abundance_count” where weighted is set to false. If “abundance_prop” or “abundance_count” is double clicked it creates a table in the tabbed browser for viewing.

Note that sometimes the variable browser does not refresh as soon as a new variable is created. As a workaround the user can go to **filter by -> data sets and models** and then back to **filter by -> all**. Figure 12 shows what mcaGUI looks like after all these steps are completed for the Sogin 2006 data.

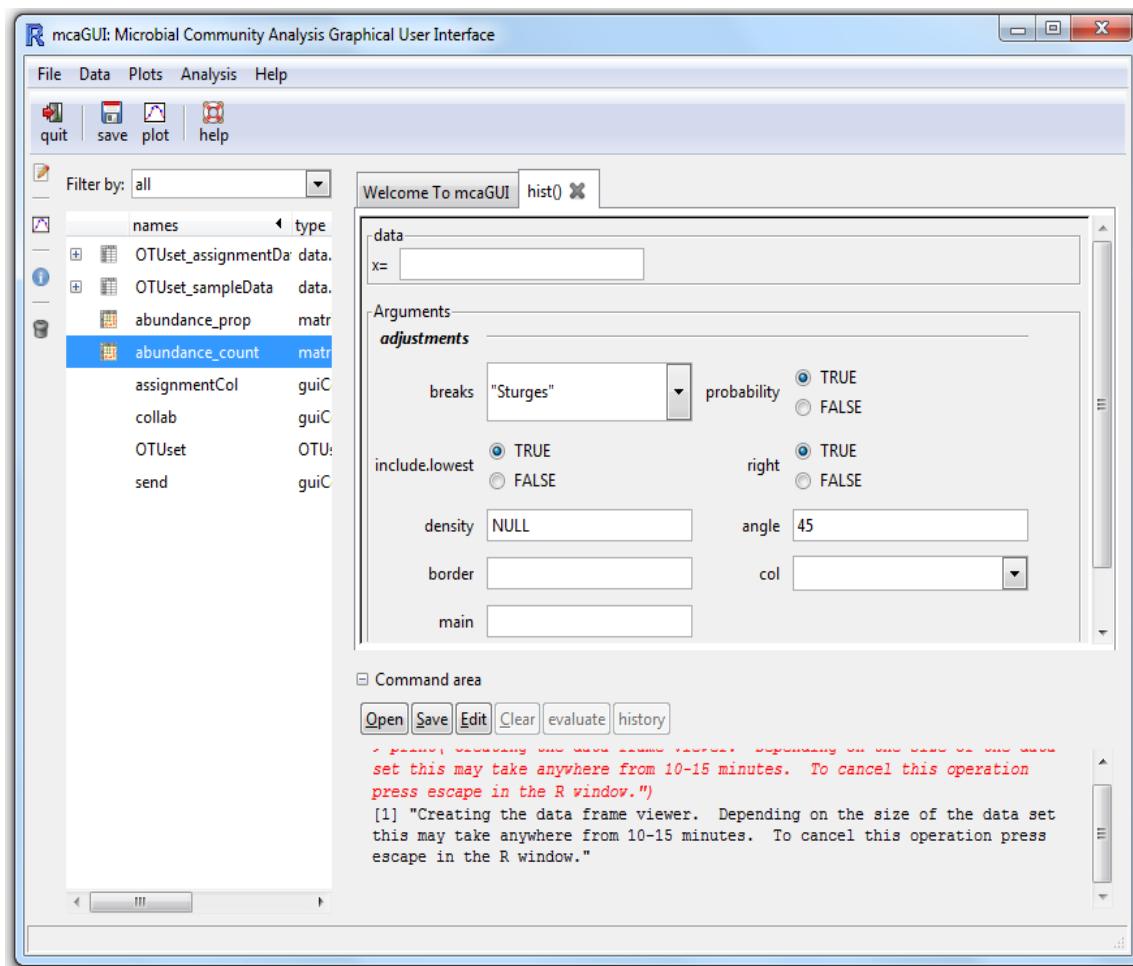
Figure 12: Abundance tables for both counts and proportions have been created. The user can view them in the tabbed browser by double clicking them.



2. Using a histogram to view the distribution of the data.

Creating a histogram is relatively straight forward. With an abundance table created. Go to **plots -> univariate -> histogram**. Figure 13 shows the resulting dialogue box.

Figure 13: Histogram dialogue box.



In order to create the histogram drag the abundance table into the $x=$ field. After doing this there are many arguments that can be specified. Note that the title under the ***main*** field must be in quotes. Figure 14 a and Figure 14 b show what the window looks like after all the arguments are filled in, along with the generated histogram.

Figure 14 a: Arguments filled in to create a histogram.

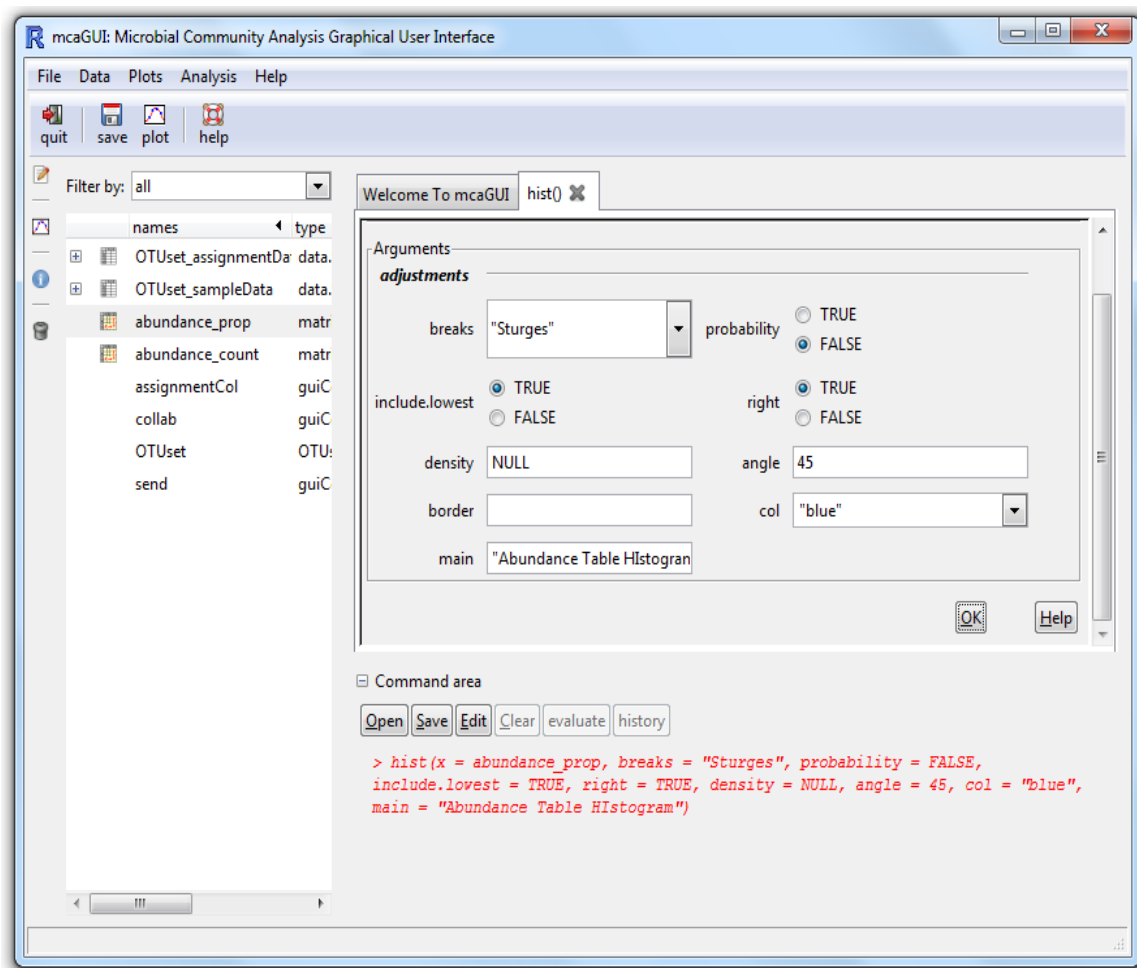
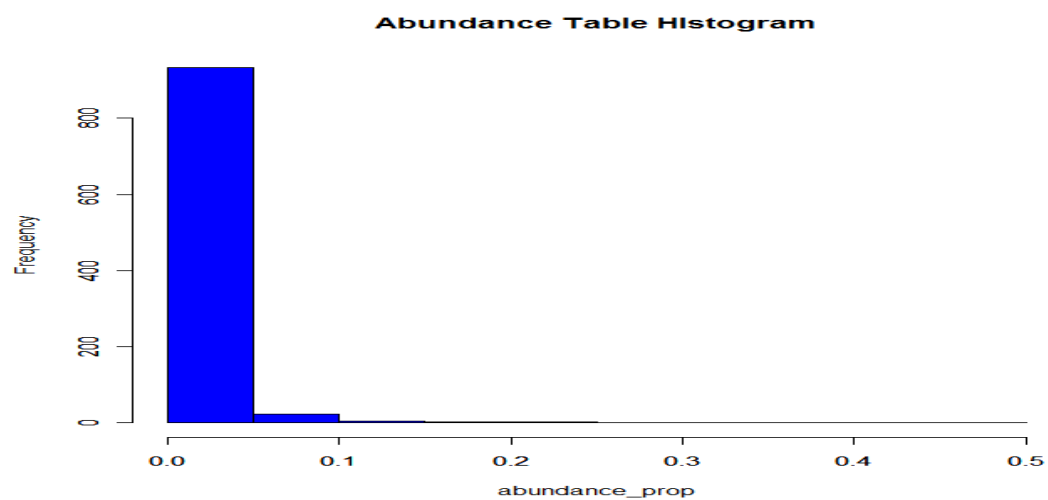


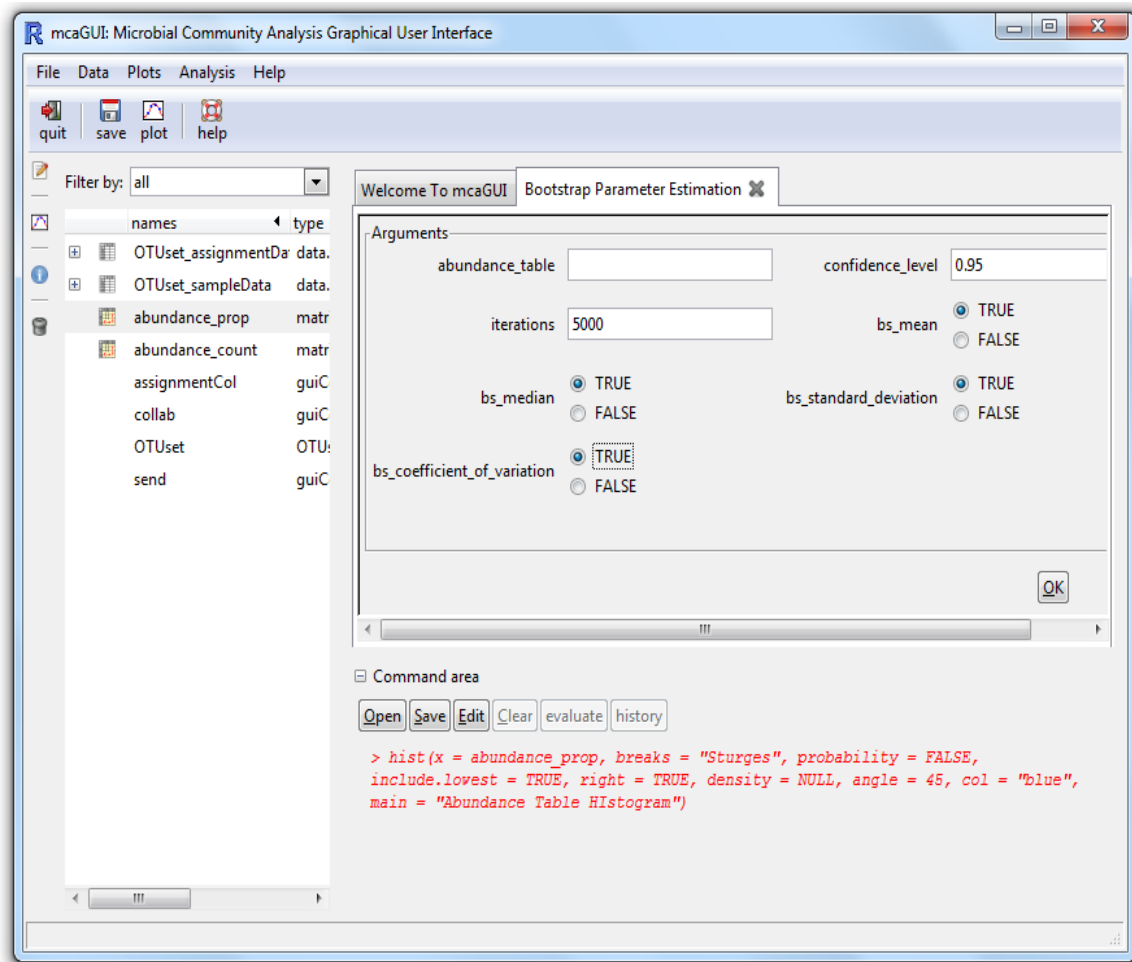
Figure 14 b: Histogram of created from the abundance table.



3. Obtaining basic summary statistics.

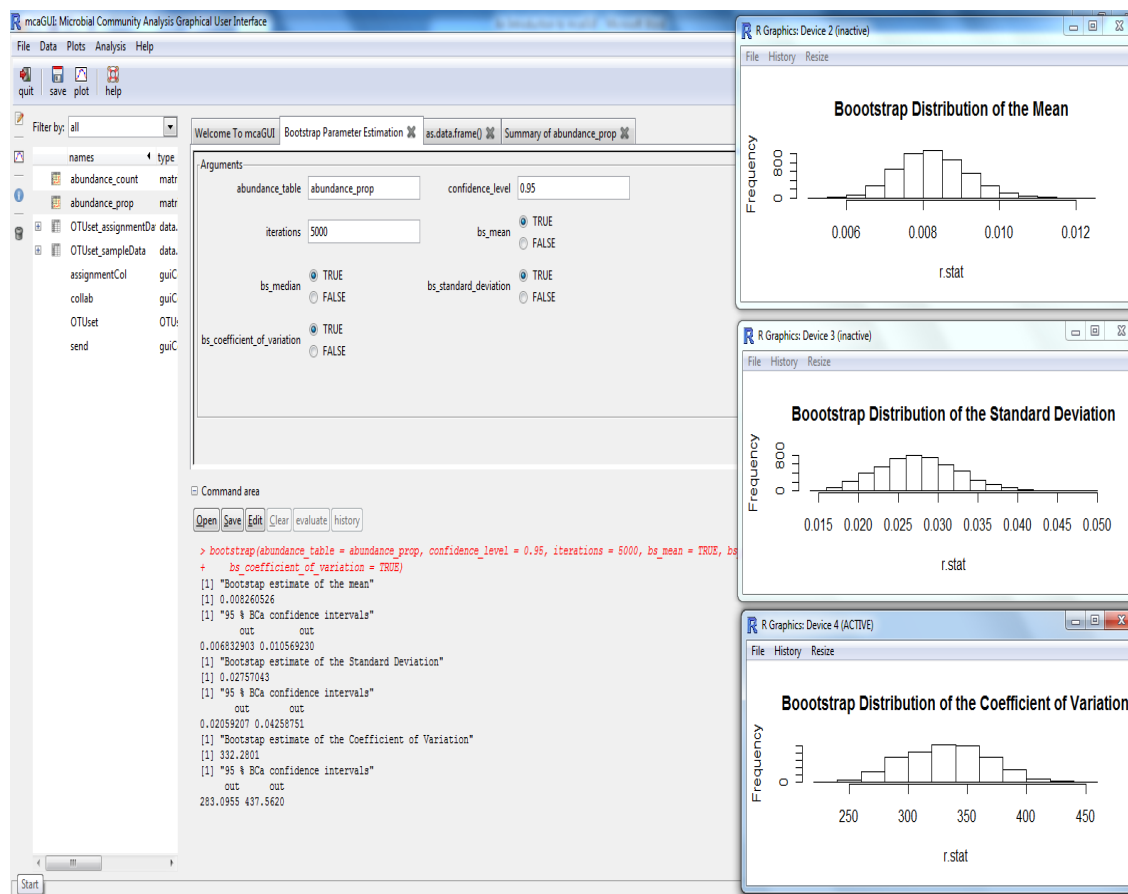
Summary statistics can be obtained a variety of ways. The user can navigate to **data -> univariate summary statistics** to get the mean, median or mode. The user can also go to **data -> univariate summaries -> summary** to get confidence intervals with the summary statistics. Obtaining summary statistics in this way though, gives parametric confidence intervals that depend on the model being normally distributed. Given that the histogram in *figure 14 b* has a very heavy right tail skew, confidence intervals dependent upon normal theory would be inappropriate. Go to **data -> Nonparametric Bootstrap Summaries** for a non-parametric alternative. *Figure 15* shows the dialogue box that comes up after the previous steps are followed.

Figure 15: Bootstrap parameter estimation dialogue box.



Bootstrap parameter estimation takes multiple arguments. The first is the abundance table, or parts of an abundance table that might be of interest. The user can also specify the confidence level and the number of iterations (ie. The number of bootstrap samples taken). Finally the user can specify what statistics to estimate along with plots for the distribution for each. *Figure 16* shows the information obtained from computing summary statistics for “abundance_prop”.

Figure 16: Bootstrap parameter estimation output.



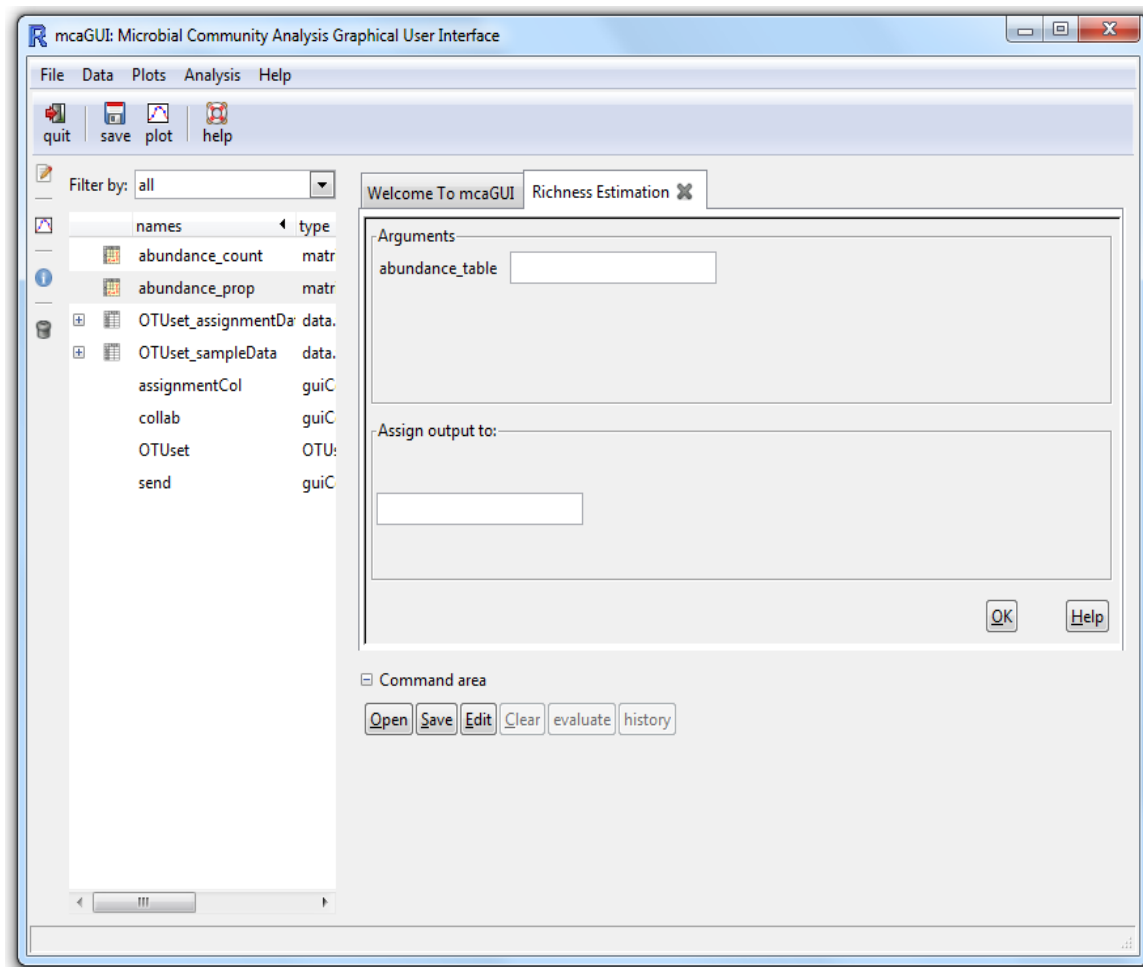
Notice that three plots are generated along with confidence intervals for each of the statistics. While it might be of interest to see these statistics for the entire data set it might also be of interest to see them for specific variables.

To obtain summary statistics for the first row of “abundance_prop” type in the **abundance_table** argument “abundance_prob[1,]”. Similarly to obtain summary statistics first column write “abundance_prop[, 1]”. In general to obtain summary statistics for any row or column in the abundance table use “abundance_table[row,]” or “abundance_table[, column]”.

4. Using count abundance tables to calculate richness and evenness.

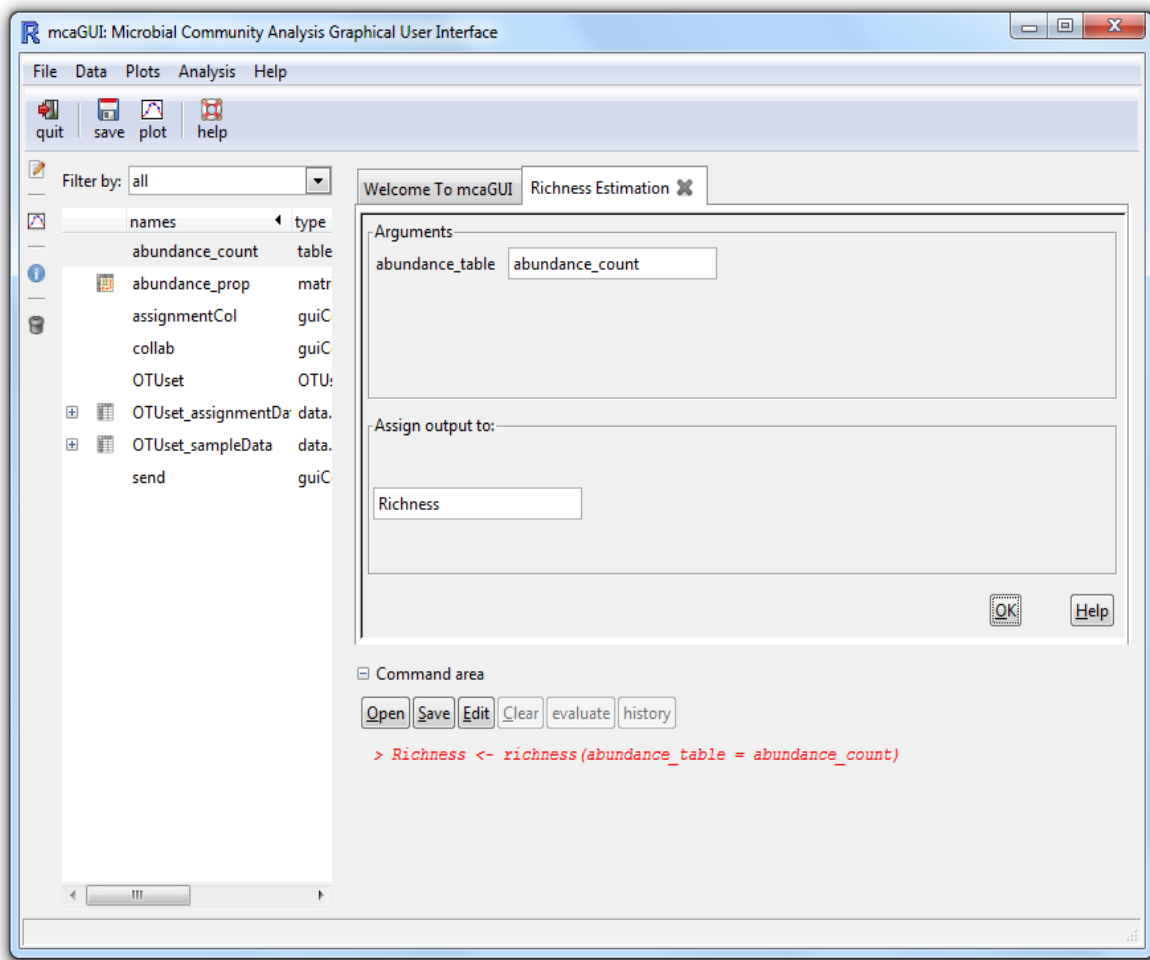
Go to **analysis -> richness**. Figure 17 shows the dialogue box that is opened after this is done.

Figure 17: Richness estimation dialogue box.



To get richness estimates for each sample drag and drop “abundance_count” into the **abundance_table** field. Note for richness estimates the abundance table must be count data since richness is not defined for proportions. *Figure 18* shows the filled in values.

Figure 18: Filled in values for richness estimation.

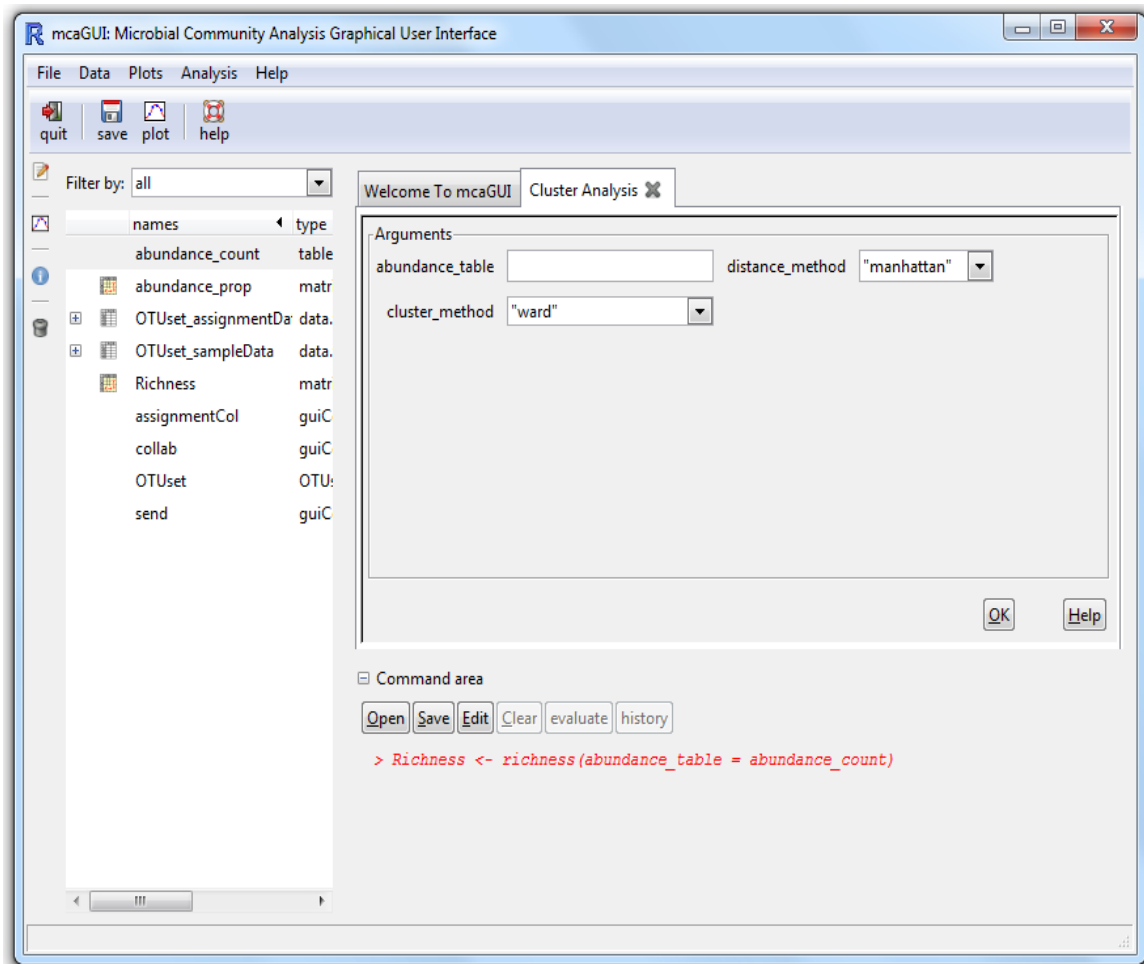


Press the **OK** button to perform the analysis. After this is done the variable browser will again need to be refreshed using the method described in creating abundance tables. Double click the new object, in this case Richness, to view it. An error will be returned if “abundance_prop” is used. This is because richness and evenness estimation are only defined for integer values (ie. count data). The process for estimating evenness is exactly the same as above so is omitted.

5. Perform a cluster analysis.

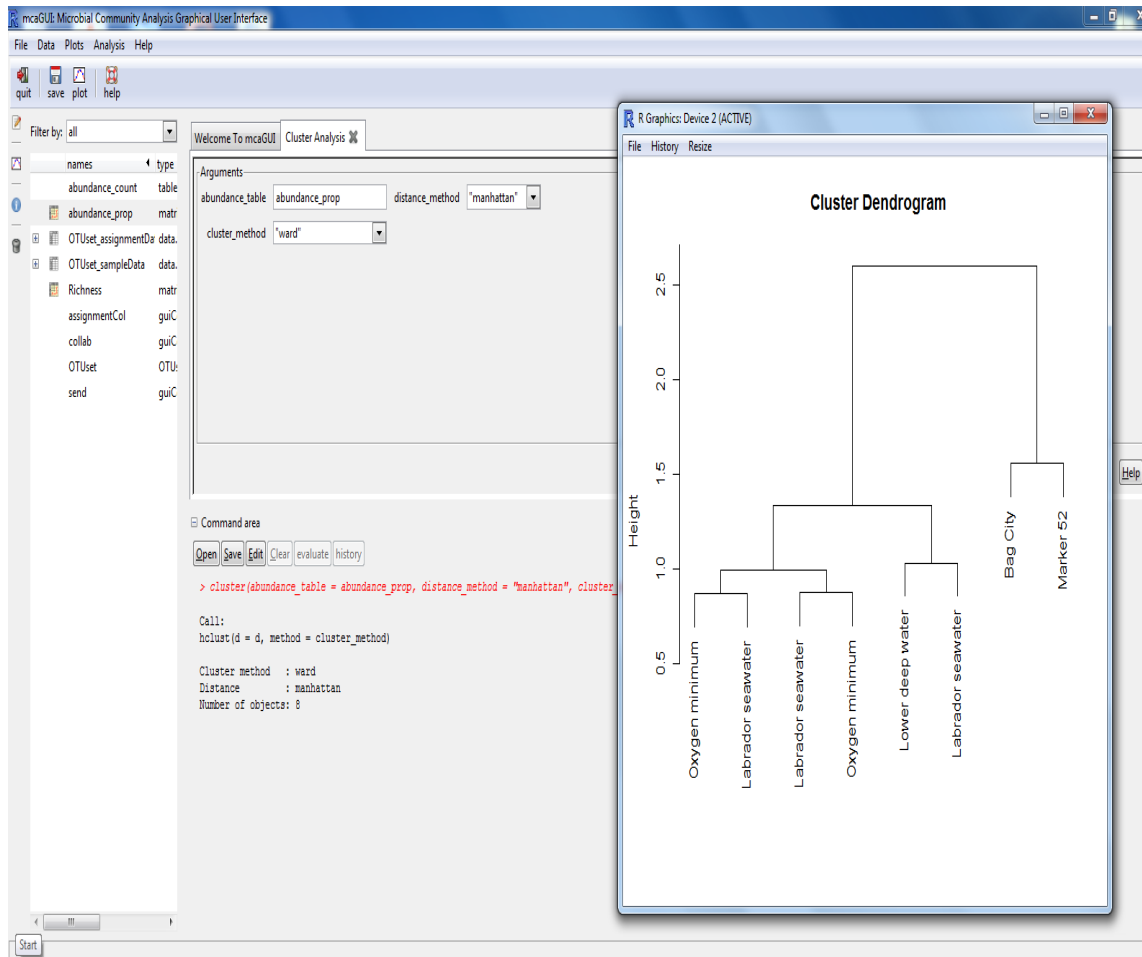
Cluster analysis can be performed as follows. Go to **analysis -> multivariate -> cluster analysis**. Figure 19 shows the dialogue box created after this is done.

Figure 19: Cluster analysis dialogue box.



There are a few additional arguments required for clustering. The first is the abundance table to perform cluster analysis on. The second two arguments identify the distance method and clustering method. *Figure 20* shows the filled in values and resulting output.

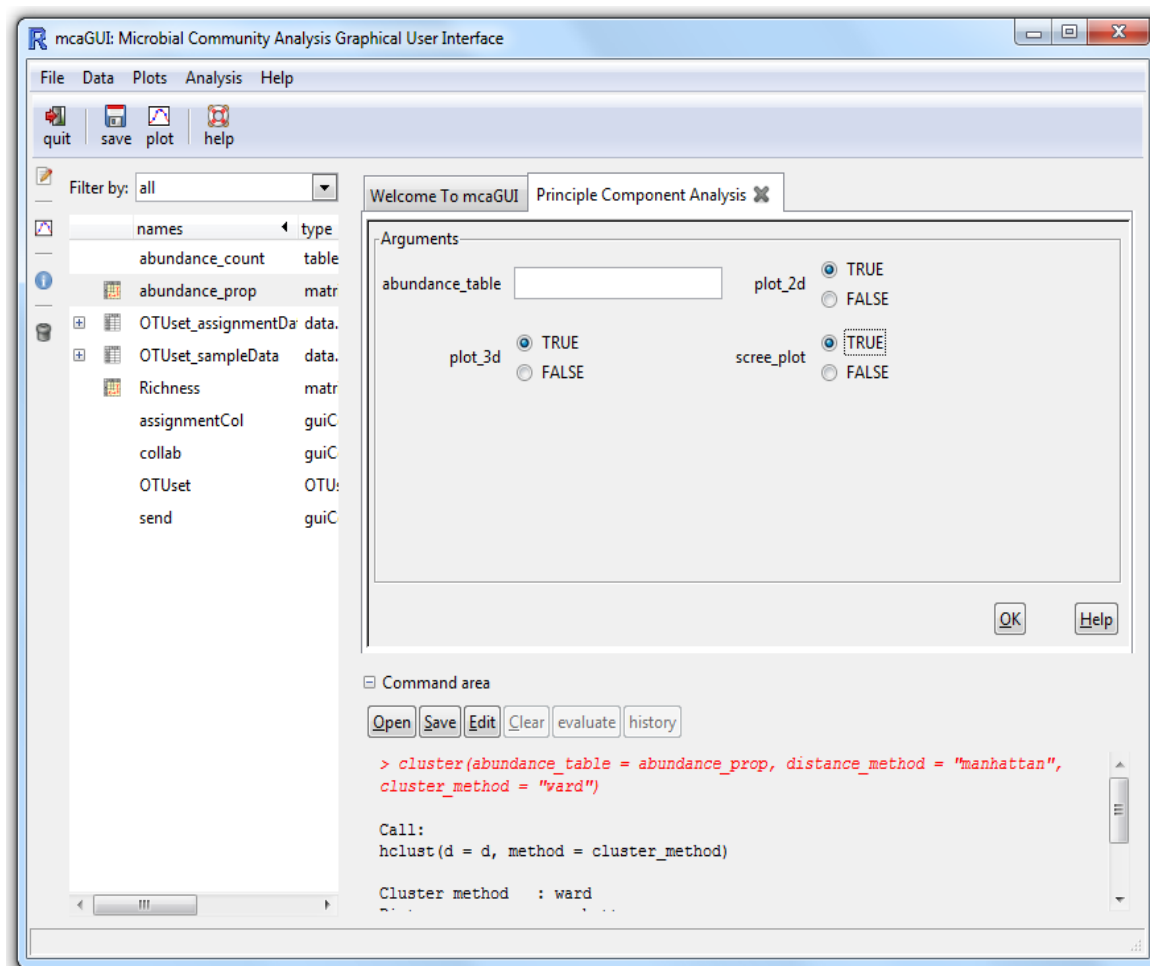
Figure 20: Cluster analysis output.



6. Perform PCA analysis in both 2 and 3 dimensional space.

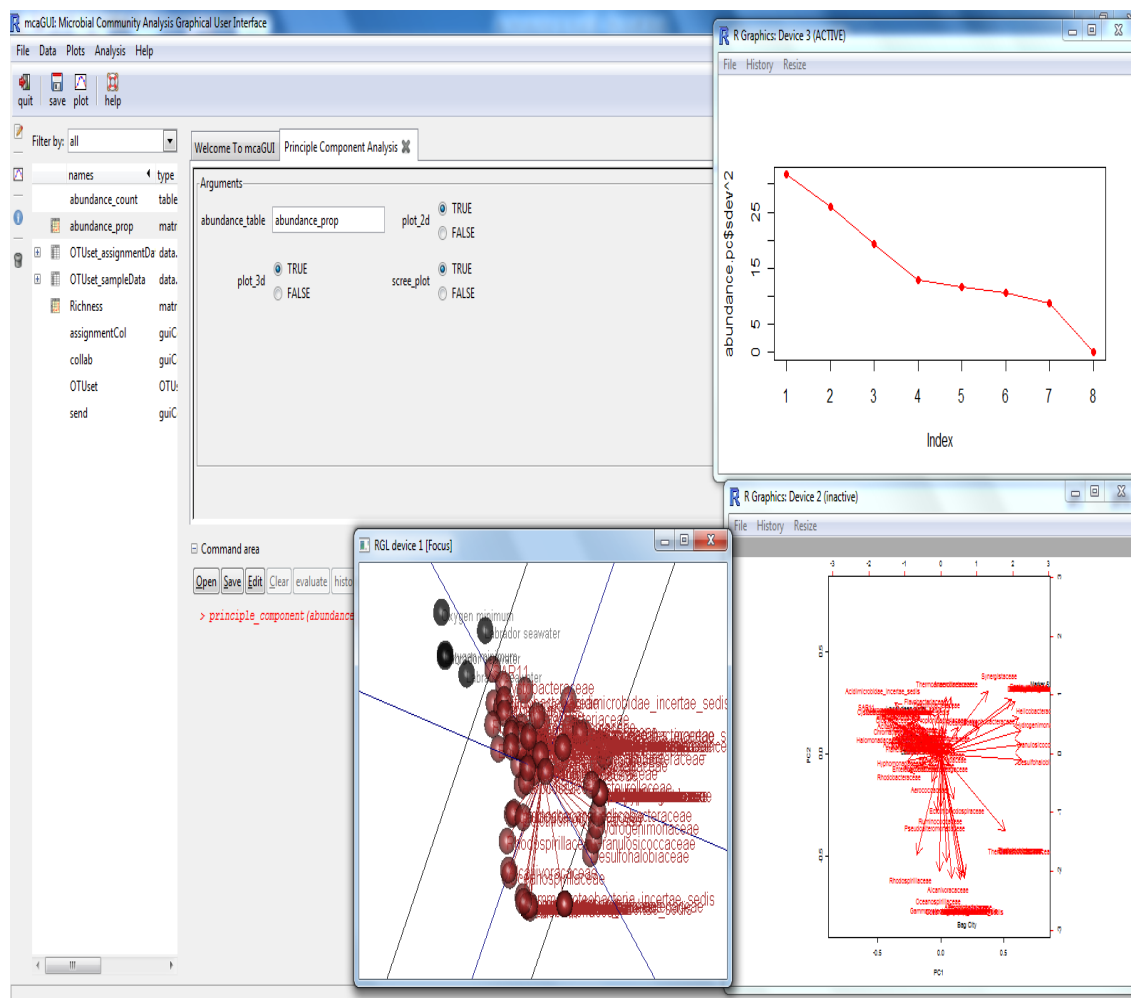
Principle Components Analysis (PCA) is a commonly used tool for understanding multivariate data structures. In order to run principle components analysis, go to **Analysis -> Multivariate -> Principle Component Analysis (PCA)**. Below is the resulting dialogue box.

Figure 21: Dialogue box for Principle Component Analysis.



As in the previous examples an abundance table needs to be provided. There are also several options that can be selected. The “plot_2d” option makes a 2D plot in the PCA1 and PCA2 directions. The “plot_3d” option creates a 3D plot in the PCA1, PCA2 and PCA3 directions. The last option is a Scree plot. This type of plot helps in identifying the PCA directions that account for the most variation in the data. If the 3D plot is specified the user can further explore the space by moving the mouse and using the mouse wheel to zoom in and out. *Figure 22* shows the filled in values and output generated from the PCA analysis.

Figure 22: Output from PCA analysis.



References

1. Definition of Operating System. (2011). Retrieved December January 5, 2011 from <http://cplus.about.com/od/introductiontoprogramming/g/opsystemdef.htm>
2. FASTA format. (2010). Retrieved December 5, 2010 from http://en.wikipedia.org/wiki/FASTA_format
3. Graphical User Interface. (2011). Retrieved December January 19, 2011 from <http://www.thefreedictionary.com/graphical+user+interface>
4. Group File. (2010). Retrieved December 5, 2010 from http://www.mothur.org/wiki/Group_file
5. Human Microbiome Project (HMP) Program Initiatives. (2008). Retrieved February 28, 2010, from <http://nihroadmap.nih.gov/hmp/initiatives.asp>
6. John Verzani with contributions by Yvonnick Noel (2010). pmg: Poor Man's GUI. R package version 0.9-42. <http://CRAN.R-project.org/package=pmg>
7. Lozupone, Hamady & Knight, "UniFrac - An Online Tool for Comparing Microbial Community Diversity in a Phylogenetic Context.", BMC Bioinformatics 2006, 7:371 - [PubMed](#)
8. R Development Core Team (2010). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org>
9. Schloss, PD & Handelsman, J. (2005). Introducing DOTUR, a computer program for defining operational taxonomic units and estimating species richness. Applied and Environmental Microbiology. 71:1501-6.
10. Schloss, PD, Larget, BR, & Handelsman J. (2004). Integration of microbial ecology and statistics: a test to compare gene libraries. Applied and Environmental Microbiology. 70:5485-92.
11. Schloss, P.D., et al., Introducing mothur: Open-source, platform-independent, community-supported software for describing and comparing microbial communities. Appl Environ Microbiol, 2009. 75(23):7537-41.

12. Schloss, PD & Handelsman, J. (2006) Introducing SONS, a tool for OTU-based comparisons of membership and structure between microbial communities. *Applied and Environmental Microbiology*. 72:6773-9.
 13. Schloss, PD & Handelsman, J. (2006). Introducing TreeClimber, a test to compare community structures. *Applied and Environmental Microbiology*. 72: 2379-84.
 14. Sogin Data Analysis. (2009). Retrieved December 5, 2010 from http://www.mothur.org/wiki/Sogin_data_analysis

 15. Sogin, M. L., Morrison G. H., Huber, J. A., Welch, D. M., Huse, S. M., Neal, P. R, Arrieta, J. M. & Herndl, G. J. "*Microbial diversity in the deep sea and the underexplored 'rare biosphere'*", Published online before print July 31, 2006, doi: 10.1073/pnas.0605127103 PNAS August 8, 2006 vol. 103 no. 32 12115-12120
 16. Species Evenness. (2010). Retrieved December January 19, 2011 from http://en.wikipedia.org/wiki/Species_evenness
 17. Species richness. (2011). Retrieved December January 19, 2011 from http://en.wikipedia.org/wiki/Species_richness
-