Importing and exporting data in to and out of Cheddar (0.1-631)

Lawrence Hudson

2016-10-10

Contents

1 Introduction

Cheddar's `LoadCommunity` and `SaveCommunity` functions provide a standard data format for community representation. Data are stored in CSV (Comma-Separated Value) files, which are easily edited using standard software and are well supported by R . You should read the 'Community' vignette before reading this one.

Researchers typically use their own bespoke data formats. This means that there are probably as many data formats as there are researchers! It is therefore extremely hard to write a generic 'import my data in to Cheddar' function. Help is at hand! If you have community data that you would like to import in to Cheddar, please contact me (l.hudson@nhm.ac.uk) - I will either provide example data-import R code for you to modify or will write the required R code for you.

2 Importing a single community from CSV data

A Cheddar community is represented by three files, each contains data for a different aspect of the community (Table 1). You can add properties to any aspect of the community simply by adding columns to the relevant CSV file. All properties added to these files are available to Cheddar's plotting and analysis

Aspect	File	Description
Whole-community	`properties.csv`	Contains properties applicable to the community
Nodes	`nodes.csv`	Defines species and associated properties
Trophic Links	`trophic.links.csv`	Optional file that defines the food web

Table 1: Community files

functions. The following sections show how to import a community in to Cheddar using these three files and the `LoadCommunity` function. The data are from a fictitious community named 'Stream 12'.

2.1 properties.csv

This file must contain one row of data only (Table 2). This file must contain the 'title' column. The

title	M.units	N.units
Stream 12	kg	m^-2

Table 2: Example `properties.csv` file

'M.units' and/or 'N.units' must be present if the `nodes.csv` file contains columns called 'M' and/or 'N'. The contents of this file can be accessed using the `CPS` function, which returns a `list`.

2.2 nodes.csv

This file contains one row for every species in the community (Table 3). The 'node' column is the only

node	category	functional.group	M	N
Detritus		detritus		
Fungi		decomposer		
Species 1	producer	producer	3e-13	2e+06
Species 2	producer	producer	1e-13	3e+07
Species 3	producer	producer	7e-11	2e+07
Species 4	invertebrate	detritivore	4e-08	6e+06
Species 5	invertebrate	herbivore	6e-07	6e+05
Species 6	invertebrate	herbivore	1e-07	5e+06
Species 7	invertebrate	predator	9e-05	1e+07
Species 8	vert.ecto	predator	7e-03	2e+03

Table 3: Example `nodes.csv` file

mandatory column; all of the others are optional. The column 'node' must contain node names. An error is raised if any node names are duplicated. Whitespace is stripped from the beginning and end of node names. If provided, columns called 'M' and/or 'N' must represent mean body mass and mean numerical

abundance respectively. All values in 'M' and 'N' must be either empty or greater than 0 and less than infinity. If the columns 'M' and/or 'N' are given then values named 'M.units' and/or 'N.units' must be provided in the `properties.csv` file. The contents of this file can be accessed using the `NPS` function, which returns a `data.frame`.

Many of Cheddar's plot and analysis functions make use of the 'category' node property by default, following previously-used metabolic groupings (Yodzis and Innes, 1992). The 'category' column of `nodes.csv` is optional but, if given, it should contain one of 'producer', 'invertebrate', 'vert.ecto', 'vert.endo' or should be empty. The 'Detritus' and 'Fungi' nodes do not have a metabolic category so have no value for the 'category' column. The 'functional group' column contains a different way of classifying nodes in the community.

2.3 trophic.links.csv

This file contains a row for every resource-consumer trophic interaction in the community (Table 4). Values

resource	consumer
Detritus	Species 4
Detritus	Species 5
Fungi	Species 4
Fungi	Species 6
Species 1	Species 5
Species 1	Species 6
Species 2	Species 4
Species 2	Species 5
Species 2	Species 6
Species 3	Species 4
Species 3	Species 5
Species 3	Species 6
Species 4	Species 7
Species 4	Species 8
Species 5	Species 7
Species 6	Species 8
Species 7	Species 7
Species 7	Species 8

Table 4: Example `trophic.links.csv` file

in 'resource' and 'consumer' should contain node names. An error is raised if any names in 'resource' or 'consumer' are not in the 'node' column of the `nodes.csv` file. Whitespace is stripped from the beginning and end of all values in 'resource' and 'consumer'. Other columns are properties of trophic links. An error is raised if any links appear more than once. The contents of this file can be accessed using the `TLPS` function, which returns a `data.frame`.

2.4 Loading the community

The above files should be in the same directory, say '`c:/Stream12`'.

```
> stream12 <- LoadCommunity('c:/Stream12')
```

Examine the community to make sure that the data have been imported correctly.

```
> stream12

Stream 12 containing 10 nodes and 18 trophic links

> NumberOfNodes(stream12)

[1] 10

> NumberOfTrophicLinks(stream12)

[1] 18
```

Examine each of the three aspects.

```
> # Community properties
> CPS(stream12)

$title
[1] "Stream 12"

$M.units
[1] "kg"

$N.units
[1] "m^-2"

> # Node properties
> NPS(stream12)

               node     category functional.group     M     N
Detritus   Detritus                      detritus    NA    NA
Fungi         Fungi                    decomposer    NA    NA
Species 1 Species 1     producer         producer 3e-13 2e+06
Species 2 Species 2     producer         producer 1e-13 3e+07
Species 3 Species 3     producer         producer 7e-11 2e+07
Species 4 Species 4 invertebrate      detritivore 4e-08 6e+06
Species 5 Species 5 invertebrate        herbivore 6e-07 6e+05
Species 6 Species 6 invertebrate        herbivore 1e-07 5e+06
Species 7 Species 7 invertebrate         predator 9e-05 1e+07
Species 8 Species 8    vert.ecto         predator 7e-03 2e+03

> # Trophic links
> TLPS(stream12)
```

```
    resource  consumer
1   Detritus Species 4
2   Detritus Species 5
3      Fungi Species 4
4      Fungi Species 6
5  Species 1 Species 5
6  Species 1 Species 6
7  Species 2 Species 4
8  Species 2 Species 5
9  Species 2 Species 6
10 Species 3 Species 4
11 Species 3 Species 5
12 Species 3 Species 6
13 Species 4 Species 7
14 Species 4 Species 8
15 Species 5 Species 7
16 Species 6 Species 8
17 Species 7 Species 7
18 Species 7 Species 8
```

`SumBiomassByClass` returns the total biomass in each 'category' node property.

```
> SumBiomassByClass(stream12)

   <unnamed> invertebrate     producer    vert.ecto
          NA   9.0110e+02   1.4036e-03   1.4000e+01
```

You can easily get the total biomass in each 'functional.group'.

```
> SumBiomassByClass(stream12, class='functional.group')

 decomposer detritivore    detritus   herbivore    predator    producer
         NA  2.4000e-01          NA  8.6000e-01  9.1400e+02  1.4036e-03
```

3 Importing a collection of communities

Cheddar's `LoadCollection` and `SaveCollection` functions provide a standard data format for collections of communities. Each community in a collection is stored in its own directory using the CSV format described in Section 2. For example, the following fictitious collection 'Grassland1994' contains three communities: 'Plot 1', 'Plot 2' and 'Plot 3'.

```
Grassland1994
|
| - communities
    |
    | - Plot 1
        |
        | - nodes.csv
        | - properties.csv
        | - trophic.links.csv
    | - Plot 2
        |
        | - nodes.csv
        | - properties.csv
        | - trophic.links.csv
    | - Plot 3
        |
        | - nodes.csv
        | - properties.csv
        | - trophic.links.csv
```

The `LoadCommunity` function loads the collection in to R .

```
> grassland <- LoadCommunity('c:/Grassland1994')

> grassland

A collection of 3 communities

> length(grassland)

[1] 3
```

The 'Collections' vignette explains information community collections in detail.

4 Export

The `SaveCommunity` and `SaveCollection` functions export communities and collections respectively to CSV files. We illustrate how to export Cheddar data to some relevant R packages and to other software.

4.1 igraph

Rather than attempt to implement, test and support the myriad complex graph analyses that have been applied to food webs, we point users to the igraph R package (Csardi and Nepusz, 2006), which is a powerful graph-analysis toolkit. The following function exports a Cheddar community to igraph.

```
> install.packages('igraph')
> library(igraph)
> ToIgraph <- function(community, weight=NULL)
 {
     if(is.null(TLPS(community)))
     {
         stop('The community has no trophic links')
     }
     else
     {
         tlps <- TLPS(community, link.properties=weight)
         if(!is.null(weight))
         {
             tlps$weight <- tlps[,weight]
         }
         return (graph.data.frame(tlps,
                                  vertices=NPS(community),
                                  directed=TRUE))
     }
 }
> data(TL84)
> # Unweighted network
> TL84.ig <- ToIgraph(TL84)
> data(Benguela)
> # Use diet fraction to weight network
> Benguela.ig <- ToIgraph(Benguela, weight='diet.fraction')
```

4.2 NetIndices and foodweb (R forge)

The `PredationMatrix` function can be used to export a Cheddar community to the NetIndices (Kones et al., 2009) and foodweb (Lin et al., 2011) packages, the later available only on R Forge (Theußl and Zeileis, 2009) at the time of writing. The foodweb package depends upon NetIndices.

```
> install.packages("NetIndices")
> install.packages("foodweb", repos="http://R-Forge.R-project.org")
> library(foodweb)    # Loads the dependency NetIndices
```

```
> data(TL84)
> TL84.ni <- PredationMatrix(TL84, weight='diet.fraction')
> # Plot the predation matrix
> foodweb::imageweb(TL84.ni)
> data(Benguela)
> # Use diet fraction to weight network
> Benguela.ni <- PredationMatrix(Benguela, weight='diet.fraction')
> # Compute flow-based trophic level using both Cheddar and NetIndices. Both
> # packages should give the same results
> cbind(ni.tl=NetIndices::TrophInd(Benguela.ni)[,'TL'],
       cheddar.tl=FlowBasedTrophicLevel(Benguela, weight.by='diet.fraction'))
```

4.3 Network 3D

The Network 3D Windows software produces interactive 3D visualisations of food webs and other networks (Yoon et al., 2004). The function below exports a Cheddar community to the '.web' and '.txt' files used by the Network 3D software.

```
> library(cheddar)
> ExportToNetwork3D <- function(community, dir, fname_root=CP(community, 'title'))
 {
     links <- cbind(PredatorID=NodeNameIndices(community, TLPS(community)[,'consumer']),
                    PreyID=NodeNameIndices(community, TLPS(community)[,'resource']))
     write.table(links, file.path(dir, paste(fname_root, '_links.web', sep='')),
                 row.names=FALSE, sep=' ')

     species <- cbind(ID=1:NumberOfNodes(community),
                      CommonName=NP(community, 'node'))

     write.table(links, file.path(dir, paste(fname_root, '_species.txt', sep='')),
                 row.names=FALSE, sep=' ')
 }
> data(TL84)
> ExportToNetwork3D(TL84, 'c:')
```

4.4 foodweb (CRAN)

A second package named foodweb (Perdomo et al., 2012) is available on CRAN. This package also creates 3D graphs of webs.

```
> install.packages("foodweb")
> library(foodweb)
> data(TL84)
> # Write the predation matrix to a csv file in the format required by foodweb.
> write.table(PredationMatrix(TL84), 'TL84.foodweb.csv', row.names=FALSE,
              col.names=FALSE, sep=',')
> # foodweb::analyse.single() creates a file called 'Results-TL84.foodweb.csv'
```

```
> # to the current directory. This file will contain some basic food-web
> # statistics and is required by the foodweb::plotweb() function.
> foodweb::analyse.single('TL84.foodweb.csv')
> foodweb::plotweb(cols=1:7, radii=7:1)           # A 3D plot
```

References

G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006. URL `http://igraph.sf.net`.

J.K. Kones, K. Soetaert, D. van Oevelen, and J. Owino. Are network indices robust indicators of food web functioning? a monte carlo approach. *Ecological Modelling*, 220:370–382, 2009. doi: 10.1016/j.ecolmodel.2008.10.012.

Y. Lin, O. Petchey, K. Soetaert, B.C. Rall, and F. Schneider. *foodweb: Generating food webs*, 2011. URL `http://R-Forge.R-project.org/projects/foodweb/`. R package version 1.0/r14.

G. Perdomo, R. Thompson, and P. Sunnucks. *food web: an open-source program for the visualisation and analysis of compilations of complex food webs.*, 2012.

S. Theußl and A. Zeileis. Collaborative Software Development Using R-Forge. *The R Journal*, 1(1):9–14, May 2009. URL `http://journal.r-project.org/2009-1/RJournal_2009-1_Theussl+Zeileis.pdf`.

P. Yodzis and S. Innes. Body size and resource-consumer dynamics. *The American Naturalist*, 139(6): 1151–1175, 1992. doi: 10.1086/285380.

I. Yoon, R.J. Williams, E. Levine, S. Yoon, J. Dunne, and N.D. Martinez. Webs on the web (wow): 3d visualization of ecological networks on the www for collaborative research and education. In *Proceedings of the IS&T/SPIE Symposium on Electronic Imaging, Visualization and Data Analysis*, volume 5295, pages 124–132. Citeseer, 2004.