Vignette: enaR

S.R. Borrett and M.K. Lau

July 2, 2014

Contents

1 Introduction

This package is a collection of functions to implement Ecological Network Analysis (ENA), which is a family of algorithms for investigating the structure and function of ecosystems modeled as networks of thermodynamically conserved energy–matter exchanges. The package brings together multiple ENA algorithms from several approaches into one common software framework that is readily available and extensible. The package builds on the *network* data structure for R developed by Butts (2008*a*). In addition to being able to perform several types of ENA with a single package, users can also make use of network analysis tools built into the *network* package, the *sna* (social network analysis) package (Butts, 2008*b*), and other components of what is now called *statnet* (Handcock et al., 2008).

This vignette illustrates how to use the *enaR* package to perform ENA. It is not meant to be a detailed guide to ENA, but we provide some references to the primary literature for those wishing to learn more about the techniques.

2 Background

Before describing how to use this package, we provide a brief background of ENA. Users may find this helpful as several software design decisions were predicated on the history and current state of the field.

The ENA methodology is an application and extension of economic Input–Output Analysis (Leontief, 1936, 1966) that was first introduced into ecology by Hannon (1973). Two major schools have developed in ENA. The first is based on Dr. Robert E. Ulanowicz's work with a strong focus on trophic dynamics and a use of information theory (Ulanowicz, 1986, 1997, 2004). The second school has an environment focus and is built on the environ concept introduced by Dr. Bernard C. Patten (Fath and Patten, 1999; Patten, 1978; Patten et al., 1976). Patten's approach has been collectively referred to separately as *Network Environ Analysis*. At the core the two approaches are very similar; however, they make some different starting assumptions and follow independent yet braided development tracks. One example difference that has historically inhibited collaboration and applications is that the two schools orient their analytical matrices in different ways. The Ulanowicz school orients their matrices as flows from rows-to-columns, which is the most common orientation in the broader field of network science (e.g., Brandes and Erlebach, 2005). In contrast, the Patten School has historically oriented their matricies from column-to-row. Recent research has started to bring the work of the two schools back together (e.g., Scharler and Fath, 2009); we hope this software contributes to this. Borrett et al. (2012) provides an entry level overview of the field.

Disparate software packages have been created to support ENA. Ulanowicz first developed and distributed the DOS based NETWRK4 code, which is still available. Recently some of these algorithms were reimplemented in an Microsoft Excel based WAND package (Allesina and Bondavalli, 2004). Some of these methods have also been encoded in the popular Ecopath with Ecosim software that assists with model construction (Christensen and Walters, 2004). Fath and Borrett (2006) published NEA.m, a MATLAB©function that collected the Patten School's algorithms together into one set of code. One objective for this R package is to begin to bring together these different algorithms into a single accessible and extensible package. The primary ENA algorithms included in this package are summarized in Table 1 and a plot of the network of functions for the package can be found in Figure 1.

Table 1: Primary Ecological Network Analysis algorithms in *enaR*.

Analysis	**Function Name**	**School**
Structure	`enaStructure`	foundational, Patten
Flow	`enaFlow`	foundational, Patten
Ascendency	`enaAscendency`	Ulanowicz
Storage	`enaStorage`	Patten
Utility	`enaUtility`	Patten
Mixed Trophic Impacts	`enaMTI`	Ulanowicz
Control	`enaControl`	Patten
Environ	`enaEnviron`	Patten

3 Data Input: General

In this section we describe the data necessary for the Ecological Network Analysis and show how to build the central network data object in R that contains the model data for subsequent analysis. To start, we assume you have installed the enaR package, and then loaded the library as follows:

```
> library(enaR)
```

3.1 Model Data

ENA is applied to a network model of energy–matter exchanges among system components. The system is modeled as a set of n compartments or nodes that represent species, species-complexes (i.e., trophic guilds or functional groups), or non-living components of the system in which energy–matter is stored. Nodes are connected by L observed fluxes, termed directed edges or links. This analysis requires an estimate of the energy–matter flowing from node i to j over a given period, $\mathbf{F}_{n\times n} = [f_{ij}]$, $i, j = 1, 2, \ldots, n$. These fluxes can be generated by any process such as feeding (like a food web), excretion, and death. As ecosystems are thermodynamically open, there must also be energy–matter inputs into the system $\mathbf{z}_{1\times n} = [z_i]$, and output losses from the system $\mathbf{y}_{1\times n} = [y_i]$. While the Patten School treats all outputs the same, the Ulanowicz School typically partitions outputs into respiration $\mathbf{r}_{1\times n} = [r_i]$ and export $\mathbf{e}_{1\times n} = [e_i]$ to account for differences in energetic quality. Note that $y_i = r_i + e_i, \forall i$. Some analyses also require the amount of energy–matter stored in each node (e.g., biomass), $\mathbf{X}_{1\times n} = [x_i]$. The final required information is a categorization of each node as living or not, which is essential for algorithms from the Ulanowicz School. For our implementation, we have created a logical vector $\mathbf{Living}_{1\times n}$ that indicates whether the i^{th} node is living (TRUE) or not (FALSE). Together, the model data $\mathcal{M}$ can be summarized as $\mathcal{M} = \{\mathbf{F}, \mathbf{z}, \mathbf{e}, \mathbf{r}, \mathbf{X}, \mathbf{Living}\}$.

Notice the row-to-column orientation of $\mathbf{F}$. This is consistent with the Ulanowicz School of network analysis, as well as the orientation commonly used in Social Network Analysis and used in the *statnet* packages. However, this is the opposite orientation typically used in the Patten School of analysis that conceptually builds from a system of differential equations and thus uses the column-to-row orientation common in this area of mathematics. Even though the difference is only a matrix transpose, this single difference may be the source of much confusion in the literature and frustration on the part of users. We have selected to use row-to-column orientation for our primary data structure, as it is the dominant form across network analytics as evidenced by it use in the

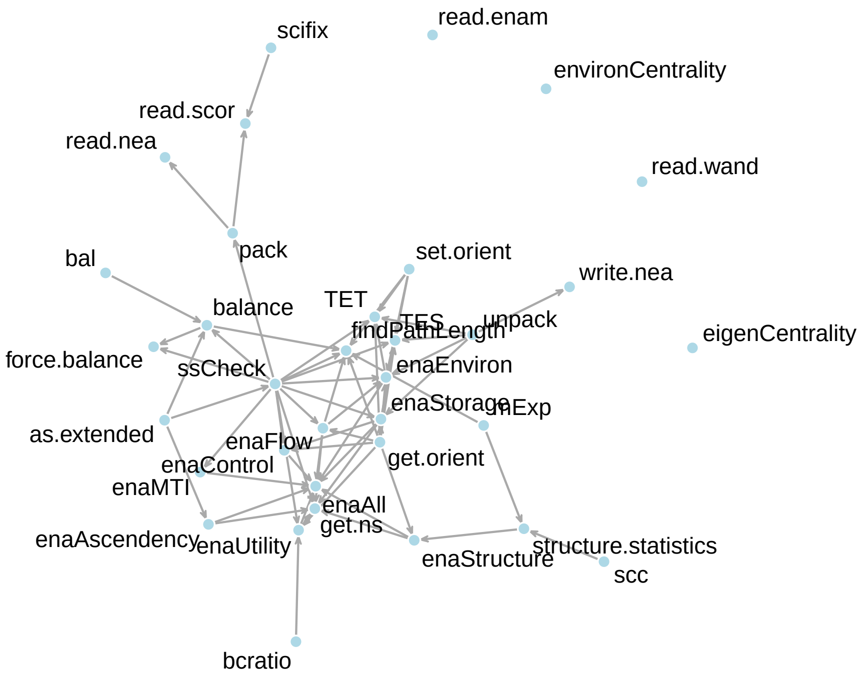

Figure 1: A plot of the *enaR* function relationships. Edges point *from* a function that provides information *to* the function that receives that information.

statnet packages. The package algorithms also return the results in the row-to-column orientation by default; however, we have built in functionality with the functions `get.orient` and `set.orient` that allows users to return output in the Patten School row-to-column orientation (see Section 6.10 for details).

3.2 Network Data Class

The *enaR* package stores the model data in the **network** class defined in the *network* package (see Butts, 2008*a*, for details). Again, the primary network object components are:

- F = flow matrix oriented row-to-column
- z = inputs
- r = respiration
- e = exports
- y = respiration+exports
- X = biomass or storage values
- Living = logical vector indicating if the node is living (TRUE) or non-living (FALSE)

3.3 Building a Network Object

Users can assemble the necessary data elements described in Section 3.1 and then use the `pack` function to create the network data object. Here is an example of doing this with hypothetical data.

```
> # generate the flow matrix
> flow.mat <- array(abs(rnorm(100,4,2))*sample(c(0,1),100,replace=TRUE),
+                    dim=c(4,4))
> # name the nodes
> rownames(flow.mat) <- colnames(flow.mat) <- paste('node',(1:nrow(flow.mat)),sep='')
> # generate the inputs
> inputs <- runif(nrow(flow.mat),0,4)
> # generate the exports
> exports <- inputs
> # pack
> fake.model <- pack(flow=flow.mat,
+                     input=inputs,
+                     export=exports,
+                     living=TRUE)

[1] "respiration" "storage"

> # model
> fake.model

 Network attributes:
  vertices = 4
  directed = TRUE
  hyper = FALSE
  loops = FALSE
  multiple = FALSE
  bipartite = FALSE
  flow:
     node1           node2            node3           node4
 Min.   :0.000   Min.   :0.0000   Min.   :0.000   Min.   :0.000
 1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.000
 Median :1.967   Median :0.0000   Median :0.000   Median :1.282
 Mean   :2.219   Mean   :0.4644   Mean   :1.118   Mean   :1.615
 3rd Qu.:4.186   3rd Qu.:0.4644   3rd Qu.:1.118   3rd Qu.:2.897
 Max.   :4.939   Max.   :1.8575   Max.   :4.474   Max.   :3.895
  balanced = FALSE
  total edges= 3
    missing edges= 0
    non-missing edges= 3

 Vertex attribute names:
    export input living output respiration storage vertex.names

 Edge attribute names:
    flow
```

Unfortunately, the attributes() function does not clearly identify the network data objects we are using.

```
> attributes(fake.model)

$names
[1] "mel" "gal" "val" "iel" "oel"

$class
[1] "network"
```

However, individual components can be extracted from the data object using the form specified in the *network* package. For example, we can pull out node of vertex attributes as follows:

```
> fake.model%v%'output'

[1] NA NA NA NA

> fake.model%v%'input'

[1] 1.44664313 0.03584714 2.13542654 0.01644835

> fake.model%v%'living'

[1] TRUE TRUE TRUE TRUE
```

For convenience, we have defined the flow matrix as a network based characteristic and it can be extracted as:

```
> fake.model%n%'flow'

         node1   node2    node3    node4
node1 4.939150 0.00000 4.473502 0.000000
node2 0.000000 1.85745 0.000000 0.000000
node3 3.934966 0.00000 0.000000 3.894596
node4 0.000000 0.00000 0.000000 2.564117
```

There are times that it is useful to extract all of the ecosystem model data elements from the network data object. This can be accomplished using the `unpack` function. The `unpack` output is as follows:

```
> unpack(fake.model)

$F
         node1   node2    node3    node4
node1 4.939150 0.00000 4.473502 0.000000
node2 0.000000 1.85745 0.000000 0.000000
node3 3.934966 0.00000 0.000000 3.894596
node4 0.000000 0.00000 0.000000 2.564117

$z
[1] 1.44664313 0.03584714 2.13542654 0.01644835
```

```
$r
[1] 0 0 0 0

$e
[1] 1.44664313 0.03584714 2.13542654 0.01644835

$y
[1] NA NA NA NA

$X
[1] NA NA NA NA

$Living
[1] TRUE TRUE TRUE TRUE
```

Note that we did not specify the storage values. In these instances `pack` produces `NA` values. Although the package is designed to help users navigate missing data issues be sure to check that you are providing the appropriate input for a given function. For more information, see the help file for the function in question.

3.4 Balancing for Steady-State

Many of the ENA functions assume that the network model is at steady-state (node inputs equal node outputs). Thus, this package has functions for (1) checking to see if the assumption is met and (2) automatically balancing the model so that input equal outputs.

To determine if the model is balanced and then balance it if necessary:

```
> ## --- Check to see if the model is balanced ---#
> ssCheck(fake.model)

[1] FALSE

> ## --- To BALANCE a model if needed --- #
> fake.model <- balance(fake.model,method="AVG2")

[1] AVG2

> ## --- To FORCE BALANCE a model if needed --- #
> fake.model <- force.balance(fake.model)
```

The automated balancing routines are based on those presented in Allesina and Bondavalli (2003). These authors compare alternative balancing algorithms and further discuss the implications of using automated procedures. Caution is warranted when using these techniques, as they indiscriminately alter the model flow rates.

4 Data Input: Reading Common Data File Formats

Several software packages exist in the literature for running ENA. For convenience, we have written functions to read in a few of the more common data formats used by these software.

SCOR

The `read.scor` function reads in data stored in the SCOR format specified by Ulanowicz and Kay (1991) that is the input to the NETWRK4 programs. This function can be run as follows.

```
> scor.model <- readLines('http://people.uncw.edu/borretts/data/oyster.dat')
> m <- read.scor(scor.model,from.file=FALSE)
```

This constructs the network data object from the SCOR file that stores the ecosystem model data for an oyster reef model (Dame and Patten, 1981). The individual model elements are

```
> unpack(m)

$F
                  Filter Feeders Microbiota Meiofauna Deposit Feeders
Filter Feeders                 0     0.0000    0.0000          0.0000
Microbiota                     0     0.0000    1.2060          1.2060
Meiofauna                      0     0.0000    0.0000          0.6609
Deposit Feeders                0     0.0000    0.0000          0.0000
Predators                      0     0.0000    0.0000          0.0000
Deposited Detritus             0     8.1721    7.2745          0.6431
                  Predators Deposited Detritus
Filter Feeders       0.5135            15.7910
Microbiota           0.0000             0.0000
Meiofauna            0.0000             4.2403
Deposit Feeders      0.1721             1.9076
Predators            0.0000             0.3262
Deposited Detritus   0.0000             0.0000

$z
[1] 41.47  0.00  0.00  0.00  0.00  0.00

$r
[1] 25.1650  5.7600  3.5794  0.4303  0.3594  6.1759

$e
[1] 0 0 0 0 0 0

$y
[1] 25.1650  5.7600  3.5794  0.4303  0.3594  6.1759

$X
[1] 2000.0000    2.4121   24.1210   16.2740   69.2370 1000.0000

$Living
[1]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
```

This same data is stored as a network data object that is distributed with this package, which can be accessed as:

```
> data(oyster)
> m <- oyster
```

WAND

In part to make ENA more accessible to biologists, Allesina and Bondavalli (2004) recoded some of Ulanowicz's NETWRK4 algorithms into a Microsoft Excel based tool called WAND. For this tool, the model data is stored as a separate Excel file with two worksheets. The first contains many of the node attributes and the second contains the flow matrix. The `read.wand` function will create an R network data object from a WAND model file. An example WAND file can be found at http://people.uncw.edu/borretts/data/MDmar02_WAND.xls.

```
>   m <- read.wand('./MDmar02_WAND.xls')
```

This code creates a network data object for *enaR* from the WAND formatted Mdloti ecosystem model data (Scharler, 2012). This data is courtesy of U.M. Scharler.

ENAM

Another commonly used data format stores the necessary model data in a csv or Excel formatted file. We include an example Excel file of the Mdloti estuary stored in this form ("MDMAR02.xlsx", courtesy of U. M. Scharler). This format has not been described technically in the literature nor has it been named. We refer to it as ENAM as it is the ENA model data stored primarily as a square matrix with several preliminary rows that include meta-data, the number of nodes, and number of living nodes (similar to SCOR). The data format is generally similar in concept, if not exact form, to the data system matrix used as the input to the NEA.m function (Fath and Borrett, 2006). However, the ENAM format includes information on whether nodes are living and partitions output into respiration and exports.

Using an example data file, http://people.uncw.edu/borretts/data/MDMAR02.xlsx, this data format can be read into the *enaR* package as:

```
>   m <- read.enam('./MDMAR02.xlsx')
```

The current read.enam function assumes the data are stored on the first worksheet of an Excel file. In the future, we expect to expand this function's capabilities to read the data from a CSV file.

NEA

For their Matlab function to perform network environ analysis (Patten School), Fath and Borrett (2006) packaged the model flows, inputs, outputs, and storage values into what they called a system matrix

$$\mathbf{S} = \begin{bmatrix} \mathbf{F} & \vec{z} & \vec{X} \\ \vec{y} & 0 & 0 \end{bmatrix}_{(n+1)\times(n+2)}. \quad (1)$$

Flows in the system matrix are oriented from column to row.

The *enaR* function `read.nea` reads in data with this format stored as a comma separated value file. The function `write.nea()` will write any network model to a CSV file with this format.

While convenient, this data format does not enable inclusion of the full range of model information included in the *enaR* network data object. This format does not partition outputs into exports and respiration values, nor does it identify the node labels are their living status. This missing information will prevent the use of some *enaR* functions.

Here is an example of using these functions:

```
> data(oyster)
> # write oyster reef model to a csv file
> write.nea(oyster, file.name="oyster.csv")
> # read in oyster reef model data from NEA.m formatted CSV file
> m <- read.nea("oyster.csv")
>
> # Again, this model object does NOT contain all
> # of the information in the "oyster" data object.
```

5 Network Visualization

The *enaR* package uses the *network* package plot tools. Here is one example of how to plot a network model. The figure scaling may need to be adjusted depending on computer and devices. Also note that the graph only shows internal system flows.

Figure 2 (left) is a very simple example of to plot a graph of the oyster reef model accomplished with default settings.

```
> data(oyster)  # load data
> m <- oyster
> set.seed(2)    # set random seed to control plot
> plot(m)       # plot network data object (uses plot.network)
```

We can use the excellent graphics capabilities of R to make fancier plot of the same data (Fig. 2(right)).

```
> # set colors to use
> my.col=c("red","yellow",
+    rgb(204,204,153,maxColorValue=255),
+    "grey22")
> F=m%n%'flow'                      # extract flow information for later use.
> f=which(F!=0, arr.ind=T)           # get indices of positive flows
> opar <- par(las=1,bg=my.col[4],xpd=TRUE,mai=c(1.02, 0.62, 0.82, 0.42))
> set.seed(2)                        # each time the plot is called, the
>                                    # layout orientation changes.  setting
>                                    # the seed ensures a consistent
>                                    # orientation each time the plot
>                                    # function is called.
> plot(m,
+       vertex.cex=log(m%v%'storage'), # scale nodes with storage
+       label= m%v%'vertex.names',     # add node labels
+       boxed.labels=FALSE,
+       label.cex=0.65,
```

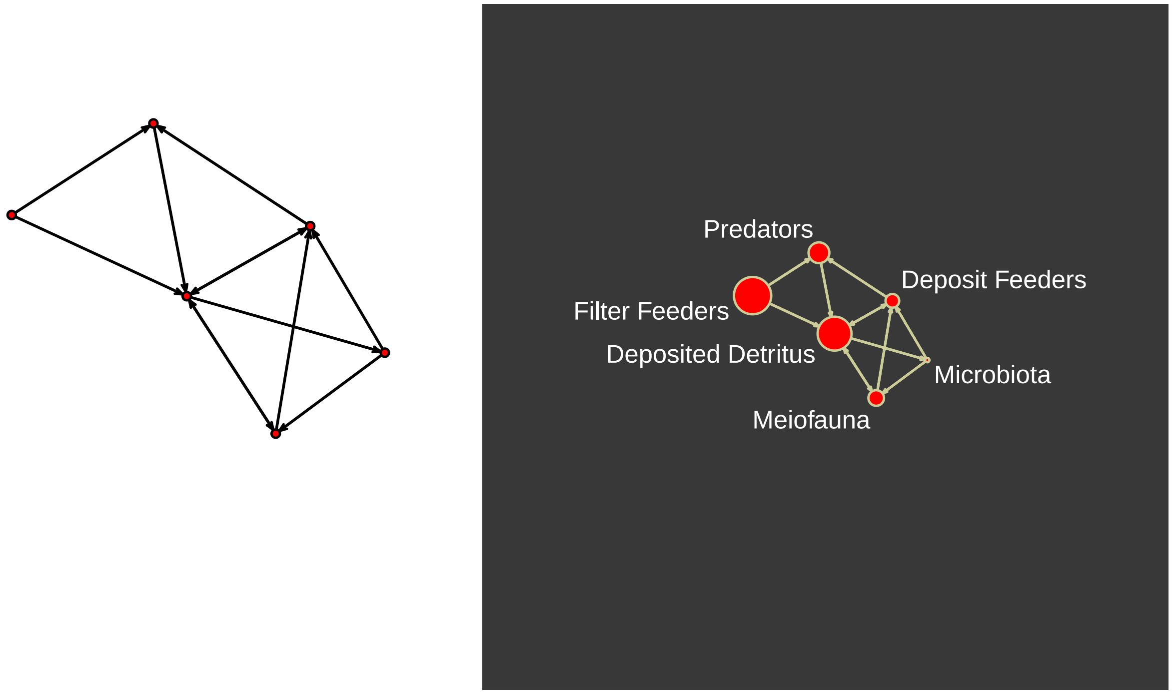

Figure 2: Simple (left) and fancy (right) plot of the Oyster network model (Dame and Patten 1981).

```
+        vertex.sides=45,   # to make rounded
+        edge.lwd=log10(abs(F[f])),     # scale arrows to flow magnitude
+        edge.col=my.col[3],
+        vertex.col=my.col[1],
+        label.col="white",
+        vertex.border = my.col[3],
+        vertex.lty = 1,
+        xlim=c(-4,1),ylim=c(-2,-2))
> rm(opar)               # remove changes to the plotting parameters
```

6 Single Model Analysis

In practice, ENA is applied to a single model. Here, we walk through an example of applying multiple ENA algorithms to the oyster reef model (Dame and Patten, 1981). The main ENA algorithms encoded in *enaR* are summarized in Table 1.

Again, in this package results are reported in the row-to-column orientation by default – including the algorithms from the Patten school. Please see Section 6.10 for how to change this default if needed.

6.1 Structural Network Analysis

Structural network analysis is common to many types of network analysis. The structural analyses applied here are based on those presented in NEA.m (Fath and Borrett, 2006) following the Patten School. Output of the enaStructure function is summarized in Table 2

Table 2: Resultant matrices and network statistics returned by the `enaStructure` function in *enaR*.

Label	Description
Matrices	
A	$n \times n$ adjacency matrix
Network statistics	
n	number of nodes
L	number of directed edges
C	connectance ($C = L/n^2$); the proportion of possible directed edges connected.
LD	Link Density (L/n)
ppr	estimated rate of pathway proliferation (Borrett and Patten, 2003)
lam1A	dominant eigenvalue of A ($lambda_1(\mathbf{A})$), which is the asymptotic rate of pathway proliferation (Borrett et al., 2007)
mlam1A	multiplicity of the dominant eigenvalue (number of times repeated)
rho	damping ratio, an indicator of how quickly $[a_{ij}]^{(m)}/[a_{ij}]^{(m-1)}$ goes to $lam_1(\mathbf{A})$ (Caswell, 2001, , p. 95)
R	distance of $lam_1(\mathbf{A})$ from the bulk of the eigen spectrum (Farkas et al., 2001)
d	difference between dominant eigenvalue and link density (expected value for random graph)
no.scc	number of strongly connected components (SCC)
no.scc.big	number of SCC with more than one node
pscc	fraction of network nodes included in a big SCC

```
> St <- enaStructure(m)
> attributes(St)

$names
[1] "A"  "ns"

> St$ns

     n  L         C LD      ppr    lam1A mlam1A      rho         R
[1,] 6 12 0.3333333  2 2.147899 2.147899      1 2.147899 0.4655712
            d no.scc no.scc.big      pscc
[1,] 0.147899      2          1 0.8333333
```

The structural network statistics show that the oyster reef model has 6 nodes, a pathway proliferation rate of 2.14, and that the model is comprised of two strongly connected components but that only one has more than one node.

6.2 Flow Analysis

Flow analysis or throughflow analysis is one of the core ENA analyses for both the Ulanowicz and Patten Schools (Fath and Borrett, 2006; Fath and Patten, 1999; Schramski et al., 2011). The *enaR* implementation `enaFlow` mostly follows the NEA.m function, with small updates (e.g. calculating the ratio of indirect-to-direct flows Borrett and Freeze, 2011; Borrett et al., 2011). Results returned by `enaFlow` are summarized in Table 3.

Here, we extract the flow statistics and then isolate and remove the output-oriented direct flow intensity matrix **G** matrix. Recall that ENA is partially derived from Input–Output analysis; the input and output orientations provide different information about the system. We also show the input-oriented integral flow matrix $\mathbf{N}'$.

Table 3: Matrices and network statistics returned by the `enaFlow` function in *enaR*.

***enaR* label**	**Description**
Matrices	
T	$n \times 1$ vector of node throughflows (M $L^{-2 \text{ or } -3}$ T^{-1})
G	output-oriented direct throughflow intensity matrix
GP	input-oriented direct throughflow intensity matrix
N	output-oriented integral throughflow intensity matrix
NP	input-oriented integral throughflow intensity matrix
Network statistics	
Input	Total input boundary flow
TST	Total System ThroughFLOW
TSTp	Total System ThroughPUT
APL	Average Path Length (Finn, 1976)
FCI	Finn Cycling Index (Finn, 1980)
BFI	Boundary Flow Intensity, $Boundary/TST$
DFI	Direct Flow Intensity, $Direct/TST$
IFI	Indirect Flow Intensity, $Indirect/TST$ (Borrett et al., 2006)
ID.F	Ratio of Indirect to Direct Flow Borrett and Freeze (2011); Borrett et al. (2011)
ID.F.I	input oriented ratio of indirect to direct flow intensity (as in Fath and Borrett, 2006)
IF.F.O	output oriented ratio of indirect to direct flow intensity (as in Fath and Borrett, 2006)
HMG.F.I	input oriented network homogenization to direct flow intensity
HMG.F.O	output oriented network homogenization to direct flow intensity
AMP.F.I	input oriented network amplification
AMP.F.O	output oriented network amplification
mode0.F	Boundary Flow
mode1.F	Internal First Passage Flow
mode2.F	Cycled Flow
mode3.F	Dissipative Equivalent to mode1.F
mode4.F	Dissipative Equivalent to mode0.F

```
>   F <- enaFlow(m)
> attributes(F)

$names
[1] "T"  "G"  "GP" "N"  "NP" "ns"

> F$ns

     Boundary      TST     TSTp       APL       FCI       BFI       DFI
[1,]    41.47 83.5833 125.0533 2.015512 0.1101686 0.4961517 0.1950689
           IFI      ID.F    ID.F.I    ID.F.O     HMG.I     HMG.O AMP.I AMP.O
[1,] 0.3087794 1.582925 1.716607 1.534181 2.051826 1.891638     3     1
     mode0.F  mode1.F  mode2.F  mode3.F mode4.F
[1,]   41.47 32.90504 9.208256 32.90504   41.47

> G <- F$G # output-oriented direct flow matrix
> rm(G)
> F$NP     # input-oriented integral flow matrix

                     Filter Feeders Microbiota Meiofauna Deposit Feeders
Filter Feeders                    1  1.0000000 1.0000000       1.0000000
```

```
Microbiota                         0  1.1018630 0.2440716       0.6197856
Meiofauna                          0  0.2971032 1.2971032       0.5604100
Deposit Feeders                    0  0.1240688 0.1240688       1.1240688
Predators                          0  0.0203426 0.0203426       0.0203426
Deposited Detritus                 0  1.3885039 1.3885039       1.3885039
                   Predators Deposited Detritus
Filter Feeders     1.0000000          1.0000000
Microbiota         0.1555792          0.1018630
Meiofauna          0.1406747          0.2971032
Deposit Feeders    0.2821649          0.1240688
Predators          1.0051064          0.0203426
Deposited Detritus 0.3485436          1.3885039
```

Note: you can use the attach function to have access to the objects nested within an object. Since some objects may conflict in name, it's best to detach an object once it's not in use.

```
> attach(F)

The following object is masked from package:base:

    T

> G

                   Filter Feeders Microbiota Meiofauna Deposit Feeders
Filter Feeders                  0  0.0000000 0.0000000      0.00000000
Microbiota                      0  0.0000000 0.1475753      0.14757529
Meiofauna                       0  0.0000000 0.0000000      0.07793173
Deposit Feeders                 0  0.0000000 0.0000000      0.00000000
Predators                       0  0.0000000 0.0000000      0.00000000
Deposited Detritus              0  0.3670363 0.3267221      0.02888377
                    Predators Deposited Detritus
Filter Feeders     0.01238245          0.3807813
Microbiota         0.00000000          0.0000000
Meiofauna          0.00000000          0.5000059
Deposit Feeders    0.06856574          0.7600000
Predators          0.00000000          0.4757876
Deposited Detritus 0.00000000          0.0000000

> detach(F)
```

Matrix powers – raising a matrix to a power is not a native operation in R. Thus, the *enaR* package includes a function `mExp` to facilitate this matrix operation commonly used in ENA.

```
> mExp(F$G,2)

                   Filter Feeders Microbiota  Meiofauna Deposit Feeders
Filter Feeders                  0  0.1397606 0.12440966      0.01099840
Microbiota                      0  0.0000000 0.00000000      0.01150080
```

Table 4: Graph-level network statistics returned by the *enaR* `enaAscendency` function (see Ulanowicz, 1986, 1997, for interpretations).

Label	Description
AMI	average mutual information (bits)
ASC	ascendency, AMI $\times$ TSTp
OH	overhead
CAP	capacity
ASC.CAP	ascendency-to-capacity ratio (dimensionless)
OH.CAP	overhead-to-capacity ratio (dimensionless)

```
Meiofauna                     0  0.1835203 0.16336297      0.01444205
Deposit Feeders               0  0.2789476 0.24830879      0.02195166
Predators                     0  0.1746313 0.15545033      0.01374254
Deposited Detritus            0  0.0000000 0.05416549      0.07962750
                     Predators Deposited Detritus
Filter Feeders     0.000000000        0.005891414
Microbiota         0.010118608        0.185945731
Meiofauna          0.005343446        0.059228112
Deposit Feeders    0.000000000        0.032622730
Predators          0.000000000        0.000000000
Deposited Detritus 0.001980437        0.185314635
```

6.3 Ascendency

A key contribution of the Ulanowicz School to ENA is Ascendency concept and the development of several information based indices (Ulanowicz, 1986, 1997). This analysis is based on all of the flows in the system and does not assume the modeled system is at steady-state. The `enaAscendency` function returns several of these information based measures (Table 4). This is run as follows:

```
>   enaAscendency(oyster)

          AMI      ASC       OH      CAP   ASC.CAP    OH.CAP
[1,] 1.330211 166.3473 211.0979 377.4452 0.4407191 0.5592809
```

6.4 Storage Analysis

Storage ENA was developed in the Patten School. It is similar to flow ENA, but divides the flows by storage (e.g., biomass) instead of throughflow. See Fath and Patten (1999) and Schramski et al. (2011) for an overview of this method. Output of this function is summarized in Table 5, and this is an example of its implementation.

```
>   S <- enaStorage(m)
> attributes(S)

$names
 [1] "X"  "C"  "P"  "S"  "Q"  "CP" "PP" "SP" "QP" "dt" "ns"
```

Table 5: Matrices and graph-level network statistics returned by the *enaR* `enaStorage` function.

Label	**Description**
Matrices	
X	$n \times 1$ vector of storage values [M L^{-2}]
C	$n \times n$ donor-storage normalized output-oriented direct flow intensity matrix (T^{-1})
P	$n \times n$ storage-normalized output-oriented direct flow matrix (dimensionless)
S	$n \times n$ donor-storage normalized output-oriented integral flow intensity matrix (T^{-1})
Q	$n \times n$ output-oriented integral flow intensity matrix (dimensionless)
CP	$n \times n$ recipient-storage normalized input-oriented direct flow intensity matrix (T^{-1})
PP	$n \times n$ storage-normalized input-oriented direct flow matrix (dimensionless)
SP	$n \times n$ donor-storage normalized input-oriented integral flow intensity matrix (T^{-1})
QP	$n \times n$ input-oriented integral flow intensity matrix (dimensionless)
dt	discrete time step
Network statistics	
TSS	Total System Storage
CIS	Storage Cycling Index
BSI	Boundary Storage Intensity
DSI	Direct Storage Intensity
ISI	Indirect Storage Intensity
ID.S	Ratio of Indirect-to-Direct storage (realized)
ID.S.I	storage-based input-oriented indirect-to-direct ratio (as in Fath and Borrett, 2006)
ID.S.O	storage-based input-oriented indirect-to-direct ratio (as in Fath and Borrett, 2006)
HMG.S.I	input-oriented storage network homogenization
HMG.S.O	output-oriented storage network homogenization
AMP.S.I	input-oriented storage network amplification
AMP.S.O	output-oriented storage network amplification
mode0.S	Storage from Boundary Flow
mode1.S	Storage from Internal First Passage Flow
mode2.S	Storage from Cycled Flow
mode3.S	Dissipative Equivalent to mode1.S
mode4.S	Dissipative Equivalent to mode0.S

```
> S$ns

          TSS       CIS         BSI         DSI       ISI     ID.S
[1,] 3112.044 0.9940252 0.003331412 0.003320932 0.9933477 299.1171
      ID.S.I   ID.S.O  HMG.S.O HMG.S.I NAS NASP mode0.S  mode1.S
[1,] 454.227 294.1527 1.115985 1.38251  20   21 10.3675 8.226261
     mode2.S  mode3.S mode4.S
[1,] 3093.45 8.226261 10.3675
```

6.5 Utility Analysis

Utility analysis describes the relationship between node pairs in the ecosystem model when considering both direct and indirect interactions. It developed in the Patten School (Fath and Patten, 1999; Patten, 1991) and is similar to yet distinct from the Ulanowicz School mixed trophic impacts analysis (Ulanowicz and Puccia, 1990). Utility analysis can be conducted from both the flow and storage perspectives, so the "type" argument needs to be set to suit the users needs. This is

Table 6: Matrices and graph-level network statistics returned by the *enaR* `Utility` function.

Label	Description
Matrices	
$D_{n\times n}$	throughflow-normalized direct utility intensity (dimensionless)
$U_{n\times n}$	integral flow utility (dimensionless)
$Y_{n\times n}$	integral flow utility scaled by original throughflow (M $L^{-2 \text{ or } -3}$ T^{-1})
$DS_{n\times n}$	storage-normalized direct utility intensity (dimensionless)
$US_{n\times n}$	integral storage utility (dimensionless)
$YS_{n\times n}$	integral storage utility scaled by original throughflow (M $L^{-2 \text{ or } -3}$ T^{-1})
Network Statistics	
lam1D	dominant eigenvalue of D
synergism.F	benefit-cost ratio or network synergism (flow)
mutualism.F	positive to negative interaction ratio or network mutualism (flow)
lam1DS	dominant eigenvalue of DS
synergism.S	benefit-cost ratio or network synergism (storage)
mutualism.S	positive to negative interaction ratio or network mutualism (storage)

again implemented as in NEA.m. Table 6 summarizes the function output for the flow and storage versions. These analyses are executed as:

```
> UF <- enaUtility(m,eigen.check=TRUE,type="flow")
> US <- enaUtility(m,eigen.check=TRUE,type="storage")
> attributes(UF)

$names
[1] "D"  "U"  "Y"  "ns"
```

Please note the function argument "eigen.check=TRUE". For this analysis to work, the power series of the direct utility matrices must converge, which is only true if the dominant eigenvalue of the direct utility matrix is less than 1. The function default prevents the analysis from being performed if this condition is not met. Users that wish to perform the analysis anyway can set "eigen.check=FALSE". Care should be used when doing this, as the meaning of the underlying mathematics is uncertain.

6.6 Environ Analysis

Environ Analysis finds the n *unit* input and output environs for the model (Fath and Patten, 1999; Patten, 1978). These unit environs are returned by the *environ* function as in NEA.m. They indicate the flow activity in each subnetwork generated by pulling a unit out of a node (input environs) or pushing a unit into a node (output environ). These unit environs can be converted into "realized" environs by multiplying each by the relevant observed input or output (Borrett and Freeze, 2011).

```
> E <- enaEnviron(m)
> attributes(E)

$names
[1] "input"  "output"
```

```
> E$output[1]

$`Filter Feeders`
                   Filter Feeders Microbiota   Meiofauna
Filter Feeders                 -1  0.0000000  0.00000000
Microbiota                      0 -0.1970605  0.02908126
Meiofauna                       0  0.0000000 -0.20449723
Deposit Feeders                 0  0.0000000  0.00000000
Predators                       0  0.0000000  0.00000000
Deposited Detritus              0  0.1970605  0.17541596
z                               1  0.0000000  0.00000000
                   Deposit Feeders    Predators Deposited Detritus
Filter Feeders          0.00000000  0.012382445        0.380781288
Microbiota              0.02908126  0.000000000        0.000000000
Meiofauna               0.01593682  0.000000000        0.102249819
Deposit Feeders        -0.06052568  0.004149988        0.045999518
Predators               0.00000000 -0.016532433        0.007865927
Deposited Detritus      0.01550760  0.000000000       -0.536896552
z                       0.00000000  0.000000000        0.000000000
                             y
Filter Feeders     0.606836267
Microbiota         0.138897999
Meiofauna          0.086310586
Deposit Feeders    0.010376176
Predators          0.008666506
Deposited Detritus 0.148912467
z                  0.000000000
```

The TET function returns vectors of the unit and realized input and output total environ throughflow. The realized total environ throughflow is an environ based partition of the total system throughflow (TST).

```
> tet <- TET(m)
> show(tet)

$realized.input
[1] 25.165000 22.647638 14.582798  2.028052  1.053786 18.107007

$realized.output
[1] 83.5833  0.0000  0.0000  0.0000  0.0000  0.0000

$unit.input
[1] 1.000000 3.931882 4.074090 4.713111 2.932069 2.931882

$unit.output
[1] 2.015512 1.836089 2.540670 3.124836 2.234317 2.594261
```

The TES functions returns the both the realized and unit total environ storage for the input and output environs. Again, the realized TES is a partition of the total system storage (TSS).

```
> tes <- TES(m)
> show(tes)

$realized.input
    Filter Feeders         Microbiota          Meiofauna
        2000.00000            2.41209           24.12171
   Deposit Feeders          Predators Deposited Detritus
          16.27440           69.23803         1000.03118

$realized.output
[1] 3112.044    0.000    0.000    0.000    0.000    0.000

$unit.input
    Filter Feeders         Microbiota          Meiofauna
       289.3658066          0.6561948          7.3735209
   Deposit Feeders          Predators Deposited Detritus
        11.5308112        109.7205293        265.1036470

$unit.output
    Filter Feeders         Microbiota          Meiofauna
          75.04326           16.06273           41.03146
   Deposit Feeders          Predators Deposited Detritus
          65.81279          132.44451           66.11575
```

6.7 Control Analysis

Control analysis is implemented as in the original NEA.m function. Recent updates to control analysis (e.g., Schramski et al., 2006, 2007) still need to be included.

```
> C <- enaControl(m)
> attributes(C)

$names
[1] "CN" "CQ" "CR" "CD" "sc"
```

6.8 Mixed Trophic Impacts

Mixed Trophic Impacts is a popular analysis from the Ulanowicz School of ENA (Ulanowicz and Puccia, 1990). The `enaMTI` function generates comparable results to the calculations in Ulanowicz and Puccia (1990). These are implemented as follows; Table 7 summarizes the function output.

```
>                                         #conduct mixed trophic impacts
> mti <- enaMTI(oyster)
> attributes(mti)

$names
[1] "G"  "FP" "Q"  "M"

>                                         #shows the total impact matrix
> mti$M
```

Table 7: Matricies returned by the *enaR* `enaMTI` function, which are based on (Ulanowicz and Puccia, 1990).

Label	**Description**
Matrices	
$G_{n\times n}$	positive effect of prey on its predator
$F_{n\times n}$	negative impact of the predator on its prey
$Q_{n\times n}$	direct net impact of one node on another
$M_{n\times n}$	total impact of i on j (direct and indirect)

```
[1] NA
```

In this case, the power series of the direct trophic impacts matrix does not converge (dominant eigenvalue is greater than one). Thus, the function returns the mti\$M = NA. Like with Utility analysis, however, we can use the eigen.check argument to do the calculation despite the mathematical problem.

```
>   mti <- enaMTI(oyster,eigen.check=FALSE)
> attributes(mti)

$names
[1] "G"  "FP" "Q"  "M"

> mti$M  # shows the total impact matrix

                   Filter Feeders  Microbiota    Meiofauna
Filter Feeders      -0.0250635283  0.16956382  0.431493557
Microbiota          -0.0015848556 -0.30675078 -0.182458391
Meiofauna           -0.0001241781 -0.47413204 -0.070959618
Deposit Feeders     -0.0069255188 -0.26769125 -0.007062628
Predators           -0.0301817448  0.02000515 -0.004028911
Deposited Detritus  -0.0034657973  0.21795628  0.612654910
                   Deposit Feeders    Predators Deposited Detritus
Filter Feeders          0.26144106  0.795834137        0.516016759
Microbiota              0.20520368  0.050323410       -0.295378609
Meiofauna               0.01607831  0.003942987       -0.001592286
Deposit Feeders        -0.10329881  0.219903765        0.177109591
Predators              -0.07586335 -0.041648786       -0.019939324
Deposited Detritus      0.44874394  0.110048344       -0.251366300
```

6.9 Other Analyses

There are a number of additional tools in the package. Here we highlight a couple of them.

A quick way to get a list of all of the global network statistics reported in Structure, Flow, Ascendency, Storage, and Utility analysis is to use the `get.ns` function.

```
> ns <- get.ns(m)
> str(ns)    # examine the structure of ns
```

```
'data.frame':	1 obs. of  62 variables:
 $ n          : num 6
 $ L          : num 12
 $ C          : num 0.333
 $ LD         : num 2
 $ ppr        : num 2.15
 $ lam1A      : num 2.15
 $ mlam1A     : num 1
 $ rho        : num 2.15
 $ R          : num 0.466
 $ d          : num 0.148
 $ no.scc     : num 2
 $ no.scc.big : num 1
 $ pscc       : num 0.833
 $ Boundary   : num 41.5
 $ TST        : num 83.6
 $ TSTp       : num 125
 $ APL        : num 2.02
 $ FCI        : num 0.11
 $ BFI        : num 0.496
 $ DFI        : num 0.195
 $ IFI        : num 0.309
 $ ID.F       : num 1.58
 $ ID.F.I     : num 1.72
 $ ID.F.O     : num 1.53
 $ HMG.I      : num 2.05
 $ HMG.O      : num 1.89
 $ AMP.I      : num 3
 $ AMP.O      : num 1
 $ mode0.F    : num 41.5
 $ mode1.F    : num 32.9
 $ mode2.F    : num 9.21
 $ mode3.F    : num 32.9
 $ mode4.F    : num 41.5
 $ AMI        : num 1.33
 $ ASC        : num 166
 $ OH         : num 211
 $ CAP        : num 377
 $ ASC.CAP    : num 0.441
 $ OH.CAP     : num 0.559
 $ TSS        : num 3112
 $ CIS        : num 0.994
 $ BSI        : num 0.00333
 $ DSI        : num 0.00332
 $ ISI        : num 0.993
 $ ID.S       : num 299
 $ ID.S.I     : num 454
 $ ID.S.O     : num 294
```

```
 $ HMG.S.O    : num 1.12
 $ HMG.S.I    : num 1.38
 $ NAS        : num 20
 $ NASP       : num 21
 $ mode0.S    : num 10.4
 $ mode1.S    : num 8.23
 $ mode2.S    : num 3093
 $ mode3.S    : num 8.23
 $ mode4.S    : num 10.4
 $ lam1D      : num 0.899
 $ synergism.F: num 4.92
 $ mutualism.F: num 2.27
 $ lam1DS     : num 0.302
 $ synergism.S: num 13.1
 $ mutualism.S: num 2.6
```

It is also possible to instantly return all of the main ENA output with `enaAll`:

```
> oyster.ena <- enaAll(oyster)
> names(oyster.ena)
```

```
[1] "ascendency" "control"    "environ"    "flow"       "mti"
[6] "storage"    "structure"  "utility"
```

Centrality analysis is a large topic in network science. Fann and Borrett (2012) introduced an environ based centrality and contrasted it with the more commonly used eigenvector centrality. Both of these centralities can be calculated in *enaR* as follows:

```
> F <- enaFlow(oyster)
> ec <- environCentrality(F$N)
> show(ec)
```

```
$ECin
    Filter Feeders         Microbiota          Meiofauna
         0.1404961          0.1279889          0.1771034
   Deposit Feeders          Predators Deposited Detritus
         0.2178241          0.1557484          0.1808391

$ECout
    Filter Feeders         Microbiota          Meiofauna
        0.06970737         0.19108709         0.20595483
   Deposit Feeders          Predators Deposited Detritus
        0.12350944         0.07903903         0.33070223

$AEC
    Filter Feeders         Microbiota          Meiofauna
         0.1051017          0.1595380          0.1915291
   Deposit Feeders          Predators Deposited Detritus
         0.1706668          0.1173937          0.2557707
```

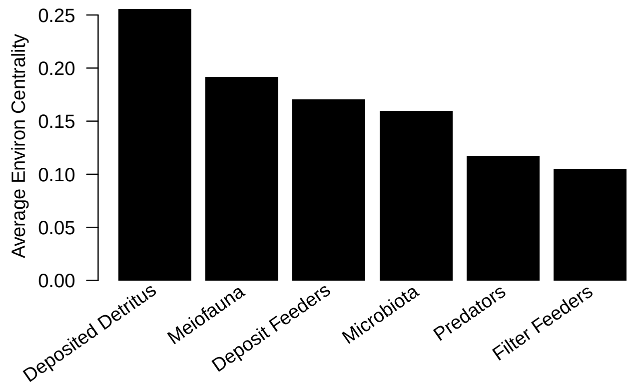

Figure 3: Bar plot of the Oyster Reef model Average Environ Centralities.

```
> eigenCentrality(F$G)

$EVCin
[1] 0.1207568 0.1093625 0.1876329 0.2518905 0.1470501 0.1833072

$EVCout
[1] 0.00000000 0.23325048 0.26566843 0.11130122 0.01286707 0.37691280

$AEVC
[1] 0.06037842 0.17130647 0.22665067 0.18159586 0.07995858 0.28011000
```

These centrality values have been normalized to sum to one.

Figure 4 shows one way to visualize the Average Environ Centralities.

```
> # set plotting parameters
> opar <- par(las=1,mar=c(7,5,1,1),xpd=TRUE,bg="white")
> # find centrality order
> o <- order(ec$AEC,decreasing=TRUE)
> bp <- barplot(ec$AEC[o],     # create barplot
+               names.arg=NA,
+               ylab="Average Environ Centrality",
+               col="black",border=NA)
> text(bp,-0.008,                 # add labels
+      labels=names(ec$AEC)[o],
+      srt=35,adj=1,cex=1)
> rm(opar)  # remove the plotting parameters
```

6.10 Output Orientation

To facilitate package use by the existing ENA community, some of which use the column-to-row orientation (e.g. the Patten School), we have created orientation functions that enable the user to set the expected output orientation for functions written in a particular "school" of analysis. Thus, functions from either school will receive network models with the standard row-to-column, but will return output with flow matrices oriented in the column-to-row orientation when appropriate (i.e. Patten school functions) and return them in that same orientation.

Here is an example of how to use the model orientation functions to re-orient the output from `enaFlow`:

```
> ###Check the current orientation
> get.orient()

[1] "rc"

> ###enaFlow output in row-column
> flow.rc <- enaFlow(oyster)$G
> ###Set the global orientation to school
> set.orient('school')
> ###Check that it worked
> get.orient()

[1] "school"

> ###enaFlow output in column-row
> flow.cr <- enaFlow(oyster)$G
> ###Check. Outputs should be transposed from each other.
> all(flow.rc == flow.cr)

[1] FALSE

> all(flow.rc == t(flow.cr))

[1] TRUE

> ###Now change back to the default orientation ('rc')
> set.orient('rc')
>
```

7 Model Library

The *enaR* package includes a library of 100 empirically based ecosystem models. There are two general classes of ecosystem models. First, there are 58 of the models are trophically-based models with food webs at their core (Tables 8). Second, there are 42 models are focused on biogeochemical cycling in ecosystems (Table 9). Christian et al. (1996), Baird et al. (2008), and Borrett et al. (2010) have previously suggested this model class distinction. In summary, these models were originally published for a number of different types of ecosystems, though predominantly aquatic, by a number of author teams. Models in the library range in size from 4 nodes to 125 nodes with connectance values ranging from 7% to 45%.

This collection of models overlaps with other data sets. For example, twenty-seven of the models (47%) are included in the set of models compiled and distributed by Dr. Ulanowicz (http://www.cbl.umces.edu/ ulan/ntwk/network.html). All 50 of the models analyzed by Borrett and Salas (2010) and Salas and Borrett (2011) and the 45 models analyzed in Borrett (2013) are included in this model library.

The trophic models are grouped as the `troModels` object and the biogeochemically-based models are available as the `bgcModels` object. Both data objects return a list of the model network objects. To use these models simply use the R *base* `data` function. This will load the models into the working memory as a named list of network objects:

```
> ### Import the model sets
> data(bgcModels)
> data(troModels)
> ### Check the first few model names
> head(names(bgcModels))

[1] "Hubbard Brook (Ca)(Waide)"     "Hardwood Forest, NH (Ca)"
[3] "Duglas Fir Forest, WA (Ca)"    "Duglas Fir Forest, WA (K)"
[5] "Puerto Rican Rain Forest (Ca)" "Puerto Rican Rain Forest (K)"

> head(names(troModels))

[1] "Marine Coprophagy (oyster)" "Lake Findley "
[3] "Mirror Lake"                "Lake Wingra"
[5] "Marion Lake"                "Cone Springs"

> ### Isolate a single model
> x <- troModels[[1]]
> x <- troModels$"Marine Coprophagy (oyster)"
> ### Check out the model
> summary(x)

Network attributes:
  vertices = 4
  directed = TRUE
  hyper = FALSE
  loops = FALSE
  multiple = FALSE
  bipartite = FALSE
  flow:
     SHRIMP  BENTHIC ORGANISMS SHRIMP FECES & BACTERIA
 Min.   :0   Min.   : 0.00     Min.   : 0.000
 1st Qu.:0   1st Qu.: 0.00     1st Qu.: 0.000
 Median :0   Median : 7.65     Median : 0.000
 Mean   :0   Mean   :17.05     Mean   : 5.475
 3rd Qu.:0   3rd Qu.:24.70     3rd Qu.: 5.475
 Max.   :0   Max.   :52.90     Max.   :21.900
 BENTHIC FECES & BACTERIA
 Min.   : 0.0
 1st Qu.: 0.0
 Median : 0.0
 Mean   :19.9
 3rd Qu.:19.9
 Max.   :79.6
```

```
  balanced = TRUE
 total edges = 4
   missing edges = 0
   non-missing edges = 4
 density = 0.3333333

Vertex attributes:

 export:
   logical valued attribute
   attribute summary:
   Mode    NA's
logical       4

 input:
   numeric valued attribute
   attribute summary:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00    0.00   62.05   94.90  157.00  255.50

 living:
   logical valued attribute
   attribute summary:
   Mode   FALSE    TRUE    NA's
logical       2       2       0

 output:
   numeric valued attribute
   attribute summary:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   6.60   21.67   64.45   94.90  137.70  244.10

 respiration:
   numeric valued attribute
   attribute summary:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   6.60   21.67   64.45   94.90  137.70  244.10

 storage:
   numeric valued attribute
   attribute summary:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      1       1       1       1       1       1
  vertex.names:
   character valued attribute
   4 valid vertex names

Edge attributes:
```

```
 flow:
   numeric valued attribute
   attribute summary:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      0       0       0       0       0       0

Network adjacency matrix:
                         SHRIMP BENTHIC ORGANISMS
SHRIMP                        0                 0
BENTHIC ORGANISMS             0                 0
SHRIMP FECES & BACTERIA       0                 1
BENTHIC FECES & BACTERIA      0                 1
                         SHRIMP FECES & BACTERIA
SHRIMP                                         1
BENTHIC ORGANISMS                              0
SHRIMP FECES & BACTERIA                        0
BENTHIC FECES & BACTERIA                       0
                         BENTHIC FECES & BACTERIA
SHRIMP                                          0
BENTHIC ORGANISMS                               1
SHRIMP FECES & BACTERIA                         0
BENTHIC FECES & BACTERIA                        0
```

8 Multi-Model Analyses (Batch Processing)

While many investigators analyze single models, much of ENA is used to compare ecosystem models (e.g., Baird et al., 1991, 1995; Christian and Thomas, 2003; Whipple et al., 2007). Investigators have also analyzed large set of models to determine the generality of hypothesized ecosystem properties (e.g., Borrett and Salas, 2010; Christensen, 1995; Salas and Borrett, 2011). For both of these applications, investigators need to analyze multiple models. One advantage of the *enaR* R package is that it simplifies this batch processing. Here we illustrate how to batch analyze a selection of models.

Our first step is to read in the model data for a set of trophic models:

```
> data(troModels)
```

Now that we have the raw data loaded, we can start to manipulate it. The first step is to balance the models and then we can run the flow analysis. We are using the `lapply` function to apply the analysis across the list of models stored in model.list.

```
> # balance models as necessary
> m.list <- lapply(troModels[1:10],balance)

[1] BALANCED
[1] BALANCED
[1] BALANCED
```

```
[1] BALANCED
[1] BALANCED
[1] BALANCED
[1] BALANCED
[1] BALANCED
[1] BALANCED
[1] BALANCED

> # check that models are balanced
> unlist(lapply(m.list,ssCheck))

Marine Coprophagy (oyster)              Lake Findley
                      TRUE                       TRUE
               Mirror Lake                Lake Wingra
                      TRUE                       TRUE
               Marion Lake               Cone Springs
                      TRUE                       TRUE
            Silver Springs            English Channel
                      TRUE                       TRUE
              Oyster Reef               Baie de Somme
                      TRUE                       TRUE

> # if balancing fails, you can use force.balance
> # to repeatedly apply the balancing procedure
> # although this is not the case with our model set
>
> m.list <- lapply(m.list,force.balance)
> ##Check that all the models are balanced
> all(unlist(lapply(m.list,ssCheck)))

[1] TRUE

> # Example Flow Analysis
> F.list <- lapply(m.list, enaFlow)
> # the full results of the flow analysis is now stored in the elements
> # of the F.list.  To get the results for just the first model...
> F.list[[1]]

$T
                  SHRIMP        BENTHIC ORGANISMS
                   124.1                    323.7
 SHRIMP FECES & BACTERIA BENTHIC FECES & BACTERIA
                    21.9                     79.6

$G
                        SHRIMP BENTHIC ORGANISMS
SHRIMP                       0         0.0000000
BENTHIC ORGANISMS            0         0.0000000
SHRIMP FECES & BACTERIA      0         0.6986301
```

```
BENTHIC FECES & BACTERIA      0         0.6645729
                         SHRIMP FECES & BACTERIA
SHRIMP                                 0.1764706
BENTHIC ORGANISMS                      0.0000000
SHRIMP FECES & BACTERIA                0.0000000
BENTHIC FECES & BACTERIA               0.0000000
                         BENTHIC FECES & BACTERIA
SHRIMP                                  0.0000000
BENTHIC ORGANISMS                       0.2459067
SHRIMP FECES & BACTERIA                 0.0000000
BENTHIC FECES & BACTERIA                0.0000000

$GP
                         SHRIMP BENTHIC ORGANISMS
SHRIMP                        0        0.00000000
BENTHIC ORGANISMS             0        0.00000000
SHRIMP FECES & BACTERIA       0        0.04726599
BENTHIC FECES & BACTERIA      0        0.16342292
                         SHRIMP FECES & BACTERIA
SHRIMP                                         1
BENTHIC ORGANISMS                              0
SHRIMP FECES & BACTERIA                        0
BENTHIC FECES & BACTERIA                       0
                         BENTHIC FECES & BACTERIA
SHRIMP                                          0
BENTHIC ORGANISMS                               1
SHRIMP FECES & BACTERIA                         0
BENTHIC FECES & BACTERIA                        0

$N
                         SHRIMP BENTHIC ORGANISMS
SHRIMP                        1         0.1473716
BENTHIC ORGANISMS             0         1.1953471
SHRIMP FECES & BACTERIA       0         0.8351055
BENTHIC FECES & BACTERIA      0         0.7943953
                         SHRIMP FECES & BACTERIA
SHRIMP                                 0.1764706
BENTHIC ORGANISMS                      0.0000000
SHRIMP FECES & BACTERIA                1.0000000
BENTHIC FECES & BACTERIA               0.0000000
                         BENTHIC FECES & BACTERIA
SHRIMP                                 0.03623966
BENTHIC ORGANISMS                      0.29394387
SHRIMP FECES & BACTERIA                0.20535805
BENTHIC FECES & BACTERIA               1.19534712

$NP
                         SHRIMP BENTHIC ORGANISMS
```

```
SHRIMP                          1        0.05649926
BENTHIC ORGANISMS               0        1.19534712
SHRIMP FECES & BACTERIA         0        0.05649926
BENTHIC FECES & BACTERIA        0        0.19534712
                         SHRIMP FECES & BACTERIA
SHRIMP                                         1
BENTHIC ORGANISMS                              0
SHRIMP FECES & BACTERIA                        1
BENTHIC FECES & BACTERIA                       0
                         BENTHIC FECES & BACTERIA
SHRIMP                                 0.05649926
BENTHIC ORGANISMS                      1.19534712
SHRIMP FECES & BACTERIA                0.05649926
BENTHIC FECES & BACTERIA               1.19534712

$ns
     Boundary   TST  TSTp     APL       FCI       BFI       DFI
[1,]    379.6 549.3 928.9 1.44705 0.1199863 0.6910614 0.1542493
           IFI     ID.F    ID.F.I    ID.F.O    HMG.I    HMG.O AMP.I
[1,] 0.1546893 1.002852 0.3603839 0.6126851 2.014161 1.891504     1
     AMP.O mode0.F  mode1.F  mode2.F  mode3.F mode4.F
[1,]     0   379.6 103.7915 65.90846 103.7915   379.6

>
```

We can use the same technique to extract specific information, like just the ratio of Indirect-to-Direct flow for each model.

```
> # Example of extracting just specific information - Indirect Effects Ratio
> IDs <- unlist(lapply(m.list, function(x) enaFlow(x)$ns[8]))
> #Look at the first few ID's
> head(IDs)

Marine Coprophagy (oyster)             Lake Findley
                 0.1546893                0.3669420
               Mirror Lake              Lake Wingra
                 0.4334588                0.4452123
               Marion Lake             Cone Springs
                 0.4391692                0.3105362
```

We can also collect the set of output-oriented integral flow matrices.

```
> # Here is a list containing only the output-oriented integral flow matrices
> N.list <- lapply(m.list,function(x) enaFlow(x)$N)
```

We can also apply the `get.ns` function to extract all of the network statistics for each model. We then use the `do.call` function to reshape the network statistics into a single data frame.

```
> # Collecting and combining all network statistics
> ns.list <- lapply(m.list,get.ns) # returns as list
```

```
> ns <- do.call(rbind,ns.list)  # ns as a data.frame
> # Let's take a quick look at some of the output
> colnames(ns)    # return network statistic names.

 [1] "n"            "L"            "C"            "LD"
 [5] "ppr"          "lam1A"        "mlam1A"       "rho"
 [9] "R"            "d"            "no.scc"       "no.scc.big"
[13] "pscc"         "Boundary"     "TST"          "TSTp"
[17] "APL"          "FCI"          "BFI"          "DFI"
[21] "IFI"          "ID.F"         "ID.F.I"       "ID.F.O"
[25] "HMG.I"        "HMG.O"        "AMP.I"        "AMP.O"
[29] "mode0.F"      "mode1.F"      "mode2.F"      "mode3.F"
[33] "mode4.F"      "AMI"          "ASC"          "OH"
[37] "CAP"          "ASC.CAP"      "OH.CAP"       "TSS"
[41] "CIS"          "BSI"          "DSI"          "ISI"
[45] "ID.S"         "ID.S.I"       "ID.S.O"       "HMG.S.O"
[49] "HMG.S.I"      "NAS"          "NASP"         "mode0.S"
[53] "mode1.S"      "mode2.S"      "mode3.S"      "mode4.S"
[57] "lam1D"        "synergism.F" "mutualism.F" "lam1DS"
[61] "synergism.S" "mutualism.S"

> dim(ns)         # show dimensions of ns matrix

[1] 10 62

> ns[1:5,1:5]     # show selected results

                            n  L     C  LD      ppr
Marine Coprophagy (oyster) 4  4 0.250 1.0 1.000000
Lake Findley               4  6 0.375 1.5 1.004975
Mirror Lake                5  9 0.360 1.8 1.324718
Lake Wingra                5 10 0.400 2.0 2.000000
Marion Lake                5  9 0.360 1.8 1.324718
```

Given this data frame of network statistics, we can construct interesting plots for further analysis. Here we focus on results of the St. Marks Seagrass ecosystem (Baird et al., 1998).

```
> opar <- par(las=1,mar=c(9,7,2,1),xpd=TRUE,mfrow=c(1,2),oma=c(1,1,0,0))
> x=dim(ns)[1] # number of models
> m.select <- 40:45
> bp=barplot(ns$ID.F[m.select],ylab="Indirect-to-Direct Flow Ratio (I/D, Realized)",
+          col="darkgreen",border=NA,ylim=c(0,2))
> text(bp,-0.05,                 # add labels
+       labels=rownames(ns)[m.select],
+         srt=45,adj=1,cex=0.85)
> opar <- par(xpd=FALSE)
> abline(h=1,col="orange",lwd=2)
> #
> plot(ns$FCI,ns$ID.F,pch=20,col="blue",cex=2,
```

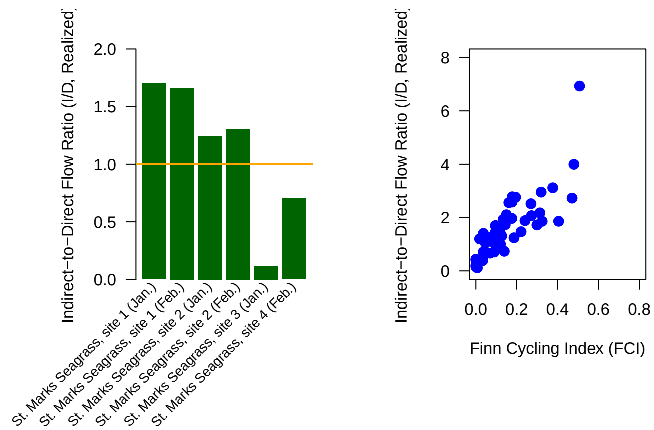

Figure 4: Ratio of Indirect-to-Direct Flow for six ecosystem models (left) and relationship between the Finn Cycling Index and the ratio of Indirect-to-Direct flow in the 56 trophic models.

```
+        ylab="Indirect-to-Direct Flow Ratio (I/D, Realized)",
+        xlab="Finn Cycling Index (FCI)",
+        xlim=c(0,0.8),ylim=c(0,8))
> #
> rm(opar)  # remove the plotting parameters
```

9 Connecting to Other Useful Packages

Another advantage of building the *enaR* package in R is that it lets ecologists take advantage of other types of network analysis and statistical tools that already exist in R. We highlight two examples here.

9.1 sna: Social Network Analysis

The *sna* package for Social Network Analysis is bundled in the *statnet* package and uses the same network data object defined in *network* that we selected to use for *enaR*. Thus, the design decision to use the network data object gives users direct access to *sna* tools.

Multiple measures of network centrality have been proposed, and the *sna* package provides a way of calculating several. Thus, ecologists can now use the sna algorithms to determine different types of centrality for their models.

```
> betweenness(oyster)

[1] 0.0 0.0 0.5 3.5 0.0 9.0

> closeness(oyster)

[1] 0.625 0.000 0.000 0.000 0.000 0.000
```

The *sna* package introduced new graphical capabilities as well. For example, it will create a target diagram of centralities.

```
> m <- troModels[[38]]
> b <- betweenness(m)          # calculate betweenness centrality
> nms <- m%v%'vertex.names'    # get vertex names
> show(nms)
> nms[b<=(0.1*max(b))] <- NA   # exclude less central nodes
> set.seed(3)
> opar <- par(xpd=TRUE,mfrow=c(1,1))
> # create target plot
> gplot.target(m,b,#circ.lab=FALSE,
+               edge.col="grey",
+               label=nms) # show only labels of most central nodes
>              #xlim=c(-1,4))
> rm(opar)
```

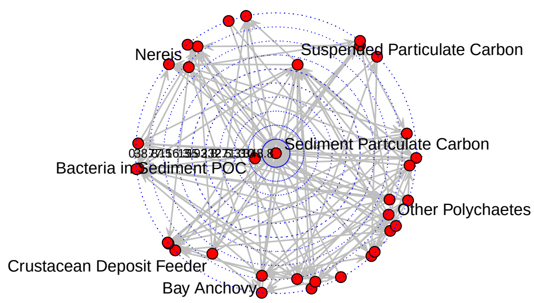

Figure 5: Target plot of node betweenness centrality for the Chesapeake Bay model (mesohaline, carbon, annual).

In addition to the node-level measures, *sna* includes graph-level indices.

```
> centralization(oyster, degree)

[1] 0.45

> centralization(oyster,closeness)

[1] 0.75

> centralization(oyster,betweenness)

[1] 0.41
```

9.2 iGraph

The *iGraph* package can also be useful for analyzing network data. Here are a few examples of using the package. Note that some functions in *iGraph* conflict with other functions already defined, so care is required when using *iGraph*.

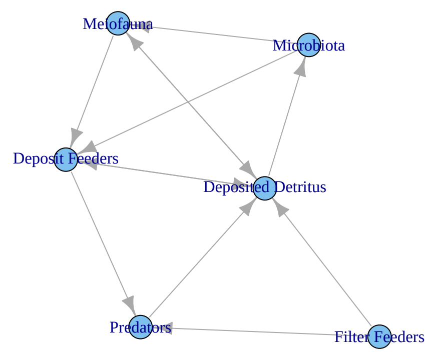

Figure 6: Plot of Oyster reef model using *iGraph*

```
> library(igraph)
> ### The adjacency matrix
> A <- St$A
> ### creating an iGraph graph
> g <- graph.adjacency(A)
> plot(g)  # uses iGraph plot tools
```

iGraph has a different set of visualization tools and generates a different looking graph (Fig. 6).

```
> # betweenness centrality (calculated by iGraph and sna)
> betweenness(g)

    Filter Feeders         Microbiota          Meiofauna
               0.0                0.0                0.5
   Deposit Feeders          Predators Deposited Detritus
               3.5                0.0                9.0

> # shortest path between any two nodes
> shortest.paths(g)

                   Filter Feeders Microbiota Meiofauna Deposit Feeders
Filter Feeders                  0          2         2               2
Microbiota                      2          0         1               1
Meiofauna                       2          1         0               1
Deposit Feeders                 2          1         1               0
Predators                       1          2         2               1
Deposited Detritus              1          1         1               1
                   Predators Deposited Detritus
Filter Feeders             1                  1
Microbiota                 2                  1
Meiofauna                  2                  1
Deposit Feeders            1                  1
Predators                  0                  1
Deposited Detritus         1                  0
```

```
> # average path length in the network (graph theory sense)
> average.path.length(g,directed=TRUE)

[1] 1.52

> diameter(g)  # diameter of the graph

[1] 2

> vertex.connectivity(g)  # connectivity of a graph (group cohesion)

[1] 0

> subcomponent(g,1,'in')  # subcomponent reachable from 1 along inputs

[1] 1

> subcomponent(g,2,'in')  # subcomponent reachable from 2 along inputs

[1] 2 6 1 3 4 5

> subcomponent(g,1,'out') # subcomponent reachable from 1 along outputs

[1] 1 5 6 2 3 4

> subcomponent(g,2,'out') # subcomponent reachable from 2 along output

[1] 2 3 4 6 5

> edge.connectivity(g)

[1] 0

> detach(package:igraph)  # detach igraph package
```

There are other R packages that have graph and network analysis tools, like Bioconductor, that might also be useful for ecologists

10 Summary and Future

This vignette shows how to use several of the key features of the *enaR* package that enables scientists to perform Ecological Network Analysis in R. The vision for this package is that it will provide access to ENA algorithms from both the Ulanowicz and Patten Schools. In its current form it replicates, updates, and extends the functionality of the NEA.m function (Fath and Borrett, 2006). It also includes both ascendency calculations and mixed trophic impacts from the Ulanowicz school of ENA, but there remains many possibilities for future development. We hope to do this in collaboration with users. This vignette also illustrates how users can further analyze their data with other R packages for graph and network analysis like *sna* and *iGraph*. In summary, we hope you find this package useful for your ENA needs.

References

Allesina, S., and C. Bondavalli. 2003. Steady state of ecosystem flow networks: A comparison between balancing procedures. Ecol. Model. **165**:221–229.

Allesina, S., and C. Bondavalli. 2004. WAND: An ecological network analysis user-friendly tool. Environ. Model. Softw. **19**:337–340.

Almunia, J., G. Basterretxea, J. Aistegui, and R. E. Ulanowicz. 1999. Benthic–pelagic switching in a coastal subtropical lagoon. Estuar. Coast. Shelf Sci. **49**:221–232.

Baird, D., H. Asmus, and R. Asmus. 2004*a*. Energy flow of a boreal intertidal ecosystem, the Sylt-Rømø Bight. Mar. Ecol. Prog. Ser. **279**:45–61.

Baird, D., H. Asmus, and R. Asmus. 2008. Nutrient dynamics in the Sylt-Rømø Bight ecosystem, German Wadden Sea: An ecological network analysis approach. Estuar. Coast. Shelf Sci. **80**:339–356.

Baird, D., R. R. Christian, C. H. Peterson, and G. A. Johnson. 2004*b*. Consequences of hypoxia on estuarine ecosystem function: Energy diversion from consumers to microbes. Ecol. Appl. **14**:805–822.

Baird, D., J. Luczkovich, and R. R. Christian. 1998. Assessment of spatial and temporal variability in ecosystem attributes of the St Marks National Wildlife Refuge, Apalachee Bay, Florida. Estuar. Coast. Shelf Sci. **47**:329–349.

Baird, D., J. M. McGlade, and R. E. Ulanowicz. 1991. The comparative ecology of six marine ecosystems. Philos. Trans. R. Soc. Lond. B **333**:15–29.

Baird, D., and H. Milne. 1981. Energy flow in the Ythan Estuary, Aberdeenshire, Scotland. Estuar. Coast. Shelf Sci. **13**:455–472.

Baird, D., and R. E. Ulanowicz. 1989. The seasonal dynamics of the Chesapeake Bay ecosystem. Ecol. Monogr. **59**:329–364.

Baird, D., R. E. Ulanowicz, and W. R. Boynton. 1995. Seasonal nitrogen dynamics in Chesapeake Bay—a network approach. Estuar. Coast. Shelf Sci. **41**:137–162.

Borrett, S. R. 2013. Throughflow centrality is a global indicator of the functional importance of species in ecosystems. Ecol. Indic. **32**:182–196.

Borrett, S. R., R. R. Christian, and R. E. Ulanowicz, 2012. Network Ecology. Pages 1767–1772 *in* A. H. El-Shaarawi and W. W. Piegorsch, editors. Encyclopedia of Environmetrics. John Wiley & Sons, 2nd edition.

Borrett, S. R., B. D. Fath, and B. C. Patten. 2007. Functional integration of ecological networks through pathway proliferation. J. Theor. Biol. **245**:98–111.

Borrett, S. R., and M. A. Freeze. 2011. Reconnecting Environs to their Environment. Ecol. Model. **222**:2393–2403.

Borrett, S. R., M. A. Freeze, and A. Salas. 2011. Equivalence of the realized input and output oriented indirect effects metrics in ecological network analysis. Ecol. Model. **222**:2142–2148.

Borrett, S. R., and O. O. Osidele. 2007. Environ indicator sensitivity to flux uncertainty in a phosphorus model of Lake Sidney Lanier, USA. Ecol. Model. **200**:371–383.

Borrett, S. R., and B. C. Patten. 2003. Structure of pathways in ecological networks: Relationships between length and number. Ecol. Model. **170**:173–184.

Borrett, S. R., and A. K. Salas. 2010. Evidence for resource homogenization in 50 trophic ecosystem networks. Ecol. Model. **221**:1710–1716.

Borrett, S. R., S. J. Whipple, and B. C. Patten. 2010. Rapid Development of Indirect Effects in Ecological Networks. Oikos **119**:1136–1148.

Borrett, S. R., S. J. Whipple, B. C. Patten, and R. R. Christian. 2006. Indirect effects and distributed control in ecosystems 3. Temporal variability of indirect effects in a seven-compartment model of nitrogen flow in the Neuse River Estuary (USA)—Time series analysis. Ecol. Model. **194**:178–188.

Brandes, U., and T. Erlebach, editors. 2005. Network Analysis: Methodological Foundations. Springer-Verlag, Berlin, Heidelberg.

Brylinsky, M., 1972. Steady-state sensitivity analysis of energy flow in a marine ecosystem. Pages 81–101 *in* B. C. Patten, editor. Systems analysis and simulation in ecology, volume 2. Academic Press.

Butts, C. 2008*a*. network: A Package for Managing Relational Data in R. J. Stat. Softw. **24**.

Butts, C. 2008*b*. Social network analysis with sna. J. Stat. Softw. **24**:1–51.

Caswell, H. 2001. Matrix Population Models: Construction, Analysis, and Interpretation. 2nd edition. Sinauer Associates, Sunderland, Mass.

Christensen, V. 1995. Ecosystem maturity—Towards quantification. Ecol. Model. **77**:3–32.

Christensen, V., and C. J. Walters. 2004. Ecopath with Ecosim: Methods, capabilities and limitations. Ecol. Model. **172**:109–139.

Christian, R. R., E. Fores, F. Comin, P. Viaroli, M. Naldi, and I. Ferrari. 1996. Nitrogen cycling networks of coastal ecosystems: influence of trophic status and primary producer form. Ecol. Model. **87**:111–129.

Christian, R. R., and C. R. Thomas. 2003. Network analysis of nitrogen inputs and cycling in the Neuse River Estuary, North Carolina, USA. Estuaries **26**:815–828.

Dame, R. F., and B. C. Patten. 1981. Analysis of energy flows in an intertidal oyster reef. Mar. Ecol. Prog. Ser. **5**:115–124.

Edmisten, J., 1970. Preliminary studies of the nitrogen budget of a tropical rain forest. Pages 211–215 *in* H. T. Odum and R. F. Pigeon, editors. A Tropical Rain Forest. TID-24270, USAEC Technical Information Center.

Fann, S. L., and S. R. Borrett. 2012. Environ centrality reveals the tendency of indirect effects to homogenize the functional importance of species in ecosystems. J. Theor. Biol. **294**:74–86.

Farkas, I., I. Derenyi, A. Barabasi, and T. Vicsek. 2001. Spectra of "real-world" graphs: Beyond the semicircle law. Physical Review E **64**:026704.

Fath, B. D., and S. R. Borrett. 2006. A Matlab© function for Network Environ Analysis. Environ. Model. Softw. **21**:375–405.

Fath, B. D., and B. C. Patten. 1999. Review of the foundations of network environ analysis. Ecosystems **2**:167–179.

Finn, J. T. 1976. Measures of ecosystem structure and function derived from analysis of flows. J. Theor. Biol. **56**:363–380.

Finn, J. T. 1980. Flow analysis of models of the Hubbard Brook ecosystem. Ecology **61**:562–571.

Handcock, M., D. Hunter, C. Butts, S. Goodreau, and M. Morris. 2008. statnet: Software tools for the representation, visualization, analysis and simulation of network data. J. Stat. Softw. **24**:1548.

Hannon, B. 1973. The structure of ecosystems. J. Theor. Biol. **41**:535–546.

Haven, D. S., and R. Morales-Alamo. 1966. Aspects of biodeposition by oysters and other invertebrate filter feeders. Limnol. Oceanogr. **11**:487–498.

Heymans, J. J., and D. Baird. 2000. A carbon flow model and network analysis of the northern Benguela upwelling system, Namibia. Ecol. Model. **126**:9–32.

Hines, D. E., J. A. Lisa, B. Song, C. R. Tobias, and S. R. Borrett. 2012. A network model shows the importance of coupled processes in the microbial N cycle in the Cape Fear River estuary. Estuar. Coast. Shelf Sci. **106**:45–57.

Hinrichsen, U., and F. Wulff. 1998. Biogeochemical and physical controls of nitrogen fluxes in a highly dynamic marine ecosystem—model and network flow analysis of the Baltic Sea. Ecol. Model. **109**:165–191.

Jordan, C. F., J. R. Kline, and D. S. Sasscer. 1972. Relative stability of mineral cycles in forest ecosystems. Am. Nat. **106**:237–253.

Leontief, W. 1936. Quantitative input and output relations in the economic systems of the United States. The Review of Economics and Statistics **18**:105–125.

Leontief, W. W. 1966. Input–Output Economics. Oxford University Press, New York.

Link, J., W. Overholtz, J. O'Reilly, J. Green, D. Dow, D. Palka, C. Legault, J. Vitaliano, V. Guida, M. Fogarty, J. Brodziak, L. Methratta, W. Stockhausen, L. Col, and C. Griswold. 2008. The Northeast US continental shelf Energy Modeling and Analysis exercise (EMAX): Ecological network model development and basic ecosystem metrics. J. Mar. Syst. **74**:453–474.

Miehls, A. L. J., D. M. Mason, K. A. Frank, A. E. Krause, S. D. Peacor, and W. W. Taylor. 2009*a*. Invasive species impacts on ecosystem structure and function: A comparison of Oneida Lake, New York, USA, before and after zebra mussel invasion. Ecol. Model. **220**:3194–3209.

Miehls, A. L. J., D. M. Mason, K. A. Frank, A. E. Krause, S. D. Peacor, and W. W. Taylor. 2009*b*. Invasive species impacts on ecosystem structure and function: A comparison of the Bay of Quinte, Canada, and Oneida Lake, USA, before and after zebra mussel invasion. Ecol. Model. **220**:3182–3193.

Monaco, M. E., and R. E. Ulanowicz. 1997. Comparative ecosystem trophic structure of three US mid-Atlantic estuaries. Mar. Ecol. Prog. Ser. **161**:239–254.

Odum, H. T. 1957. Trophic structure and productivity of Silver Springs, Florida. Ecol. Monogr. **27**:55–112.

Patrício, J., and J. C. Marques. 2006. Mass balanced models of the food web in three areas along a gradient of eutrophication symptoms in the south arm of the Mondego estuary (Portugal). Ecol. Model. **197**:21–34.

Patten, B. C. 1978. Systems approach to the concept of environment. Ohio J. Sci. **78**:206–222.

Patten, B. C., 1991. Network ecology: Indirect determination of the life–environment relationship in ecosystems. Pages 288–351 *in* M. Higashi and T. Burns, editors. Theoretical Studies of Ecosystems: The Network Perspective. Cambridge University Press, New York.

Patten, B. C., R. W. Bosserman, J. T. Finn, and W. G. Cale, 1976. Propagation of cause in ecosystems. Pages 457–579 *in* B. C. Patten, editor. Systems Analysis and Simulation in Ecology, Vol. IV. Academic Press, New York.

Ray, S. 2008. Comparative study of virgin and reclaimed islands of Sundarban mangrove ecosystem through network analysis. Ecol. Model. **215**:207–216.

Richey, J. E., R. C. Wissmar, A. H. Devol, G. E. Likens, J. S. Eaton, R. G. Wetzel, W. E. Odum, N. M. Johnson, O. L. Loucks, R. T. Prentki, and P. H. Rich. 1978. Carbon flow in four lake ecosystems: A structural approach. Science **202**:1183–1186.

Rybarczyk, H., B. Elkaim, L. Ochs, and N. Loquet. 2003. Analysis of the trophic network of a macrotidal ecosystem: the Bay of Somme (Eastern Channel). Estuar. Coast. Shelf Sci. **58**:405–421.

Salas, A. K., and S. R. Borrett. 2011. Evidence for dominance of indirect effects in 50 trophic ecosystem networks. Ecol. Model. **222**:1192–1204.

Sandberg, J., R. Elmgren, and F. Wulff. 2000. Carbon flows in Baltic Sea food webs — a re-evaluation using a mass balance approach. J. Mar. Syst. **25**:249–260.

Scharler, U. 2012. Ecosystem development during open and closed phases of temporarily open/closed estuaries on the subtropical east coast of South Africa. Estuar. Coast. Shelf Sci. **108**:119–131.

Scharler, U., and B. Fath. 2009. Comparing network analysis methodologies for consumer–resource relations at species and ecosystems scales. Ecol. Model. **220**:3210–3218.

Scharler, U. M., and D. Baird. 2005. A comparison of selected ecosystem attributes of three South African estuaries with different freshwater inflow regimes, using network analysis. J. Mar. Syst. **56**:283–308.

Schramski, J. R., D. K. Gattie, B. C. Patten, S. R. Borrett, B. D. Fath, C. R. Thomas, and S. J. Whipple. 2006. Indirect effects and distributed control in ecosystems: Distributed control in the environ networks of a seven-compartment model of nitrogen flow in the Neuse River Estuary, USA—Steady-state analysis. Ecol. Model. **194**:189–201.

Schramski, J. R., D. K. Gattie, B. C. Patten, S. R. Borrett, B. D. Fath, and S. J. Whipple. 2007. Indirect effects and distributed control in ecosystems: Distributed control in the environ networks of a seven-compartment model of nitrogen flow in the Neuse River Estuary, USA—Time series analysis. Ecol. Model. **206**:18–30.

Schramski, J. R., C. Kazanci, and E. W. Tollner. 2011. Network environ theory, simulation and EcoNet© 2.0. Environ. Model. Softw. **26**:419–428.

Tilly, L. J. 1968. The structure and dynamics of Cone Spring. Ecol. Monogr. **38**:169–197.

Ulanowicz, R. E. 1986. Growth and Development: Ecosystems Phenomenology. Springer–Verlag, New York.

Ulanowicz, R. E. 1997. Ecology, the Ascendent Perspective. Columbia University Press, New York.

Ulanowicz, R. E. 2004. Quantitative methods for ecological network analysis. Comput. Biol. Chem. **28**:321–339. URL http://dx.doi.org/10.1016/j.compbiolchem.2004.09.001.

Ulanowicz, R. E., and D. Baird. 1999. Nutrient controls on ecosystem dynamics: The Chesapeake mesohaline community. J. Mar. Syst. **19**:159–172.

Ulanowicz, R. E., C. Bondavalli, and M. S. Egnotovich, 1997. Network Analysis of Trophic Dynamics in South Florida Ecosystem, FY 96: The Cypress Wetland Ecosystem. Annual Report to the United States Geological Service Biological Resources Division Ref. No. [UMCES]CBL 97-075, Chesapeake Biological Laboratory, University of Maryland.

Ulanowicz, R. E., C. Bondavalli, and M. S. Egnotovich, 1998. Network Analysis of Trophic Dynamics in South Florida Ecosystem, FY 97: The Florida Bay Ecosystem. Annual Report to the United States Geological Service Biological Resources Division Ref. No. [UMCES]CBL 98-123, Chesapeake Biological Laboratory, University of Maryland.

Ulanowicz, R. E., C. Bondavalli, J. J. Heymans, and M. S. Egnotovich, 1999. Network Analysis of Trophic Dynamics in South Florida Ecosystem, FY 98: The Mangrove Ecosystem. Annual Report to the United States Geological Service Biological Resources Division Ref. No.[UMCES] CBL 99-0073; Technical Report Series No. TS-191-99, Chesapeake Biological Laboratory, University of Maryland.

Ulanowicz, R. E., C. Bondavalli, J. J. Heymans, and M. S. Egnotovich, 2000. Network Analysis of Trophic Dynamics in South Florida Ecosystem, FY 99: The Graminoid Ecosystem. Annual Report to the United States Geological Service Biological Resources Division Ref. No. [UMCES] CBL 00-0176, Chesapeake Biological Laboratory, University of Maryland.

Ulanowicz, R. E., and J. Kay. 1991. A package for the analysis of ecosystem flow networks. Environmental Software **6**:131–142.

Ulanowicz, R. E., and C. J. Puccia. 1990. Mixed trophic impacts in ecosystems. Coenoses **5**:7–16.

Waide, J. B., J. E. Krebs, S. P. Clarkson, and E. M. Setzler. 1974. A linear system analysis of the calcium cycle in a forested watershed ecosystem. Prog. Theor. Biol. **3**:261–345.

Whipple, S. J., S. R. Borrett, B. C. Patten, D. K. Gattie, J. R. Schramski, and S. A. Bata. 2007. Indirect effects and distributed control in ecosystems: Comparative network environ analysis of a seven-compartment model of nitrogen flow in the Neuse River Estuary, USA—Time series analysis. Ecol. Model. **206**:1–17.

Whipple, S. J., and B. C. Patten. 1993. The problem of nontrophic processes in trophic ecology: Toward a network unfolding solution. J. Theor. Biol. **163**:393–411.

Table 8: Trophic ecosystem networks (58) included in the *enaR* model library.

Models	Units	$n^\dagger$	$C^\dagger$	$Input^\dagger$	$TST^\dagger$	$FCI^\dagger$	Reference
Marine Coprophagy (oyster)	kcal m^{-2} yr^{-1}	4	0.25	379	549	0.12	Haven and Morales-Alamo (1966)
Lake Findley	gC m^{-2} yr^{-1}	4	0.38	21	50	0.30	Richey et al. (1978)
Mirror Lake	gC m^{-2} yr^{-1}	5	0.36	72	217	0.32	Richey et al. (1978)
Lake Wingra	gC m^{-2} yr^{-1}	5	0.40	478	1517	0.40	Richey et al. (1978)
Marion Lake	gC m^{-2} yr^{-1}	5	0.36	87	242	0.31	Richey et al. (1978)
Cone Springs	kcal m^{-2} yr^{-1}	5	0.32	11819	30626	0.09	Tilly (1968)
Silver Springs	kcal m^{-2} yr^{-1}	5	0.28	21296	29175	0.00	Odum (1957)
English Channel	kcal m^{-2} yr^{-1}	6	0.25	1096	2280	0.00	Brylinsky (1972)
Oyster Reef	kcal m^{-2} yr^{-1}	6	0.33	41	83	0.11	Dame and Patten (1981)
Baie de Somme	mgC m^{-2} d^{-1}	9	0.30	876	2034	0.14	Rybarczyk et al. (2003)
Bothnian Bay	gC m^{-2} yr^{-1}	12	0.22	44	183	0.23	Sandberg et al. (2000)
Bothnian Sea	gC m^{-2} yr^{-1}	12	0.24	117	562	0.31	Sandberg et al. (2000)
Ythan Estuary	gC m^{-2} yr^{-1}	13	0.23	1258	4181	0.24	Baird and Milne (1981)
Sundarban Mangrove (virgin)	kcal m^{-2} yr^{-1}	14	0.22	111317	440931	0.19	Ray (2008)
Sundarban Mangrove (reclaimed)	kcal m^{-2} yr^{-1}	14	0.22	38484	103056	0.05	Ray (2008)
Baltic Sea	mg C m^{-2} d^{-1}	15	0.17	603	1973	0.13	Baird et al. (1991)
Ems Estuary	mg C m^{-2} d^{-1}	15	0.19	282	1067	0.32	Baird et al. (1991)
Swartkops Estuary 15	mg C m^{-2} d^{-1}	15	0.17	3544	13996	0.47	Baird et al. (1991)
Southern Benguela Upwelling	mg C m^{-2} d^{-1}	16	0.23	714	2545	0.31	Baird et al. (1991)
Peruvian Upwelling	mg C m^{-2} d^{-1}	16	0.22	14927	33491	0.04	Baird et al. (1991)
Crystal River (control)	mg C m^{-2} d^{-1}	21	0.19	7357	15062	0.07	Ulanowicz (1986)
Crystal River (thermal)	mg C m^{-2} d^{-1}	21	0.14	6018	12032	0.09	Ulanowicz (1986)
Charca de Maspalomas Lagoon	mg C m^{-2} d^{-1}	21	0.12	1486230	6010331	0.18	Almunia et al. (1999)
Northern Benguela Upwelling	mg C m^{-2} d^{-1}	24	0.21	2282	6611	0.05	Heymans and Baird (2000)
Swartkops Estuary	mg C m^{-2} d^{-1}	25	0.17	2859	8949	0.27	Scharler and Baird (2005)
Sunday Estuary	mg C m^{-2} d^{-1}	25	0.16	4440	11937	0.22	Scharler and Baird (2005)
Kromme Estuary	mg C m^{-2} d^{-1}	25	0.16	2571	11087	0.38	Scharler and Baird (2005)
Okefenokee Swamp	g dw m^{-2} y^{-1}	26	0.20	2533	12855	0.48	Whipple and Patten (1993)
Neuse Estuary (early summer 1997)	mg C m^{-2} d^{-1}	30	0.09	4385	13827	0.12	Baird et al. (2004*b*)
Neuse Estuary (late summer 1997)	mg C m^{-2} d^{-1}	30	0.11	4639	13035	0.13	Baird et al. (2004*b*)
Neuse Estuary (early summer 1998)	mg C m^{-2} d^{-1}	30	0.09	4568	14025	0.12	Baird et al. (2004*b*)
Neuse Estuary (late summer 1998)	mg C m^{-2} d^{-1}	30	0.10	5641	15031	0.11	Baird et al. (2004*b*)
Gulf of Maine	g ww m^{-2} yr^{-1}	31	0.35	5053	18381	0.15	Link et al. (2008)
Georges Bank	g ww m^{-2} yr^{-1}	31	0.35	4380	16889	0.18	Link et al. (2008)
Middle Atlantic Bight	g ww m^{-2} yr^{-1}	32	0.37	4869	17916	0.18	Link et al. (2008)
Narragansett Bay	mgC m^{-2} yr^{-1}	32	0.15	693845	3917246	0.51	Monaco and Ulanowicz (1997)
Southern New England Bight	g ww m^{-2} yr^{-1}	33	0.35	4717	17597	0.16	Link et al. (2008)
Chesapeake Bay	mg C m^{-2} yr^{-1}	36	0.09	888791	3227453	0.19	Baird and Ulanowicz (1989)
Mondego Estuary (*Zostera* sp. Meadows)	g AFDW m^{-2} yr^{-1}	43	0.19	4030	6822	0.03	Patrício and Marques (2006)
St. Marks Seagrass, site 1 (Jan.)	mg C m^{-2} d^{-1}	51	0.08	514	1315	0.13	Baird et al. (1998)
St. Marks Seagrass, site 1 (Feb.)	mg C m^{-2} d^{-1}	51	0.08	601	1590	0.11	Baird et al. (1998)
St. Marks Seagrass, site 2 (Jan.)	mg C m^{-2} d^{-1}	51	0.07	602	1383	0.09	Baird et al. (1998)
St. Marks Seagrass, site 2 (Feb.)	mg C m^{-2} d^{-1}	51	0.08	800	1921	0.08	Baird et al. (1998)
St. Marks Seagrass, site 3 (Jan.)	mg C m^{-2} d^{-1}	51	0.05	7809	12651	0.01	Baird et al. (1998)
St. Marks Seagrass, site 4 (Feb.)	mg C m^{-2} d^{-1}	51	0.08	1432	2865	0.04	Baird et al. (1998)
Sylt-Rømø Bight	mg C m^{-2} d^{-1}	59	0.08	683448	1781028	0.09	Baird et al. (2004*a*)
Graminoids (wet)	g C m^{-2} yr^{-1}	66	0.18	6272	13676	0.02	Ulanowicz et al. (2000)
Graminoids (dry)	g C m^{-2} yr^{-1}	66	0.18	3472	7519	0.04	Ulanowicz et al. (2000)
Cypress (wet)	g C m^{-2} yr^{-1}	68	0.12	1418	2571	0.04	Ulanowicz et al. (1997)
Cypress (dry)	g C m^{-2} yr^{-1}	68	0.12	1035	1919	0.04	Ulanowicz et al. (1997)
Lake Oneida (pre-ZM)	g C m^{-2} yr^{-1}	74	0.22	1034	1697	0.00	Miehls et al. (2009*a*)
Lake Oneida (post-ZM)	g C m^{-2} yr^{-1}	76	0.22	810	1462	0.00	Miehls et al. (2009*a*)
Bay of Quinte (pre-ZM)	g C m^{-2} yr^{-1}	74	0.21	984	1509	0.00	Miehls et al. (2009*b*)
Bay of Quinte (post-ZM)	g C m^{-2} yr^{-1}	80	0.21	1129	2039	0.01	Miehls et al. (2009*b*)
Mangroves (wet)	g C m^{-2} yr^{-1}	94	0.15	1531	3265	0.10	Ulanowicz et al. (1999)
Mangroves (dry)	g C m^{-2} yr^{-1}	94	0.15	1531	3272	0.10	Ulanowicz et al. (1999)
Florida Bay (wet)	mg C m^{-2} yr^{-1}	125	0.12	738	2720	0.14	Ulanowicz et al. (1998)
Florida Bay (dry)	mg C m^{-2} yr^{-1}	125	0.13	547	1778	0.08	Ulanowicz et al. (1998)

† n is the number of nodes in the network model, $C = L/n^2$ is the model connectance when L is the number of direct links or energy–matter transfers, $Input = sumz_i$ is the total amount of energy–matter flowing into the system, $TST = \sum\sum f_{ij} + \sum z_i$ is the total system throughflow, and FCI is the Finn Cycling Index (Finn, 1980). Flow based network statistics ($Input$, TST, and FCI) were calculated after models were balanced using the AVG2 algorithm.

Table 9: Biogeochemical ecosystem networks (42) included in the *enaR* model library.

Model	Units	$n^\dagger$	$C^\dagger$	$Input^\dagger$	$TST^\dagger$	$FCI^\dagger$	Reference
Hubbard Brook (Waide)	kg Ca Ha^{-1} yr^{-1}	4	0.25	11	168	0.76	Waide et al. (1974)
Hardwood Forest, NH	kg Ca Ha^{-1} yr^{-1}	4	0.31	11	200	0.80	Jordan et al. (1972)
Douglas Fir Forest, WA	kg Ca Ha^{-1} yr^{-1}	4	0.31	4	54	0.74	Jordan et al. (1972)
Douglas Fir Forest, WA	kg K Ha^{-1} yr^{-1}	4	0.31	0	45	0.97	Jordan et al. (1972)
Puerto Rican Rain Forest	kg Ca Ha^{-1} yr^{-1}	4	0.31	43	274	0.57	Jordan et al. (1972)
Puerto Rican Rain Forest	kg K Ha^{-1} yr^{-1}	4	0.31	20	433	0.86	Jordan et al. (1972)
Puerto Rican Rain Forest	kg Mg Ha^{-1} yr^{-1}	4	0.31	10	70	0.58	Jordan et al. (1972)
Puerto Rican Rain Forest	kg Cu Ha^{-1} yr^{-1}	4	0.31	0	2	0.37	Jordan et al. (1972)
Puerto Rican Rain Forest	kg Fe Ha^{-1} yr^{-1}	4	0.31	0	7	0.95	Jordan et al. (1972)
Puerto Rican Rain Forest	kg Mn Ha^{-1} yr^{-1}	4	0.38	0	7	0.98	Jordan et al. (1972)
Puerto Rican Rain Forest	kg Na Ha^{-1} yr^{-1}	4	0.31	64	140	0.24	Jordan et al. (1972)
Puerto Rican Rain Forest	kg Sr Ha^{-1} yr^{-1}	4	0.31	0	1	0.71	Jordan et al. (1972)
Tropical Rain Forest	g N m^{-2} d^{-1}	5	0.24	10	71	0.48	Edmisten (1970)
Neuse River Estuary (AVG)	mmol N m^{-2} season^{-1}	7	0.45	795	41517	0.89	Christian and Thomas (2003)
Neuse River Estuary (Spring 1985)	mmol N m^{-2} season^{-1}	7	0.45	133	9120	0.91	Christian and Thomas (2003)
Neuse River Estuary (Summer 1985)	mmol N m^{-2} season^{-1}	7	0.45	119	20182	0.96	Christian and Thomas (2003)
Neuse River Estuary Fall 1985)	mmol N m^{-2} season^{-1}	7	0.45	181	8780	0.88	Christian and Thomas (2003)
Neuse River Estuary Winter 1986)	mmol N m^{-2} season^{-1}	7	0.43	187	6880	0.85	Christian and Thomas (2003)
Neuse River Estuary (Spring 1986)	mmol N m^{-2} season^{-1}	7	0.45	128	12915	0.94	Christian and Thomas (2003)
Neuse River Estuary (Summer 1986)	mmol N m^{-2} season^{-1}	7	0.45	165	11980	0.91	Christian and Thomas (2003)
Neuse River Estuary (Fall 1986)	mmol N m^{-2} season^{-1}	7	0.45	100	9863	0.94	Christian and Thomas (2003)
Neuse River Estuary (Winter 1987)	mmol N m^{-2} season^{-1}	7	0.45	691	7907	0.62	Christian and Thomas (2003)
Neuse River Estuary (Spring 1987)	mmol N m^{-2} season^{-1}	7	0.45	334	11533	0.84	Christian and Thomas (2003)
Neuse River Estuary (Summer 1987)	mmol N m^{-2} season^{-1}	7	0.45	90	15621	0.96	Christian and Thomas (2003)
Neuse River Estuary (Fall 1987)	mmol N m^{-2} season^{-1}	7	0.45	85	7325	0.93	Christian and Thomas (2003)
Neuse River Estuary (Winter 1988)	mmol N m^{-2} season^{-1}	7	0.45	171	8680	0.89	Christian and Thomas (2003)
Neuse River Estuary (Spring 1988)	mmol N m^{-2} season^{-1}	7	0.45	176	6898	0.85	Christian and Thomas (2003)
Neuse River Estuary (Summer 1988)	mmol N m^{-2} season^{-1}	7	0.45	132	16814	0.95	Christian and Thomas (2003)
Neuse River Estuary (Fall 1988)	mmol N m^{-2} season^{-1}	7	0.45	128	5732	0.87	Christian and Thomas (2003)
Neuse River Estuary (Winter 1989)	mmol N m^{-2} season^{-1}	7	0.45	291	5739	0.75	Christian and Thomas (2003)
Cape Fear River Estuary (Oligohaline)	nmol N cm^{-3} d^{-1}	8	0.36	3802	7088	0.20	Hines et al. (2012)
Cape Fear River Estuary (Polyhaline)	nmol N cm^{-3} d^{-1}	8	0.36	3068	5322	0.17	unpublished
Lake Lanier (AVG)	mg P m^{-2} day^{-1}	11	0.21	95	749	0.40	Borrett and Osidele (2007)
Baltic Sea	mg N m^{-3} day^{-1}	16	0.15	2348	44510	0.67	Hinrichsen and Wulff (1998)
Chesapeake Bay	mg N m^{-2} yr^{-1}	36	0.12	73430	484325	0.33	Baird et al. (1995)
Chesapeake Bay	mg P m^{-2} yr^{-1}	36	0.12	9402	101091	0.51	Ulanowicz and Baird (1999)
Chesapeake Bay (Winter)	mg P m^{-2} season^{-1}	36	0.08	1009	11926	0.53	Ulanowicz and Baird (1999)
Chesapeake Bay (Spring)	mg P m^{-2} season^{-1}	36	0.10	1932	27325	0.57	Ulanowicz and Baird (1999)
Chesapeake Bay (Summer)	mg P m^{-2} season^{-1}	36	0.12	4184	42935	0.46	Ulanowicz and Baird (1999)
Chesapeake Bay (Fall)	mg P m^{-2} season^{-1}	36	0.10	2276	18904	0.40	Ulanowicz and Baird (1999)
Sylt-Rømø Bight	mg N m^{-2} yr^{-1}	59	0.09	99613	363693	0.23	Baird et al. (2008)
Sylt-Rømø Bight	mg P m^{-2} yr^{-1}	59	0.09	2508	57739	0.66	Baird et al. (2008)

† n is the number of nodes in the network model, $C = L/n^2$ is the model connectance when L is the number of direct links or energy–matter transfers, $Input = sumz_i$ is the total amount of energy–matter flowing into the system, $TST = \sum\sum f_{ij} + \sum z_i$ is the total system throughflow, and FCI is the Finn Cycling Index (Finn, 1980). Flow based network statistics ($Input$, TST, and FCI) were calculated after models were balanced using the AVG2 algorithm.