CHAPTER 18

Network calculations II: a user's manual for EcoNet

C. Kazanci
Department of Mathematics & Faculty of Engineering, University of Georgia, USA.

1 Introduction

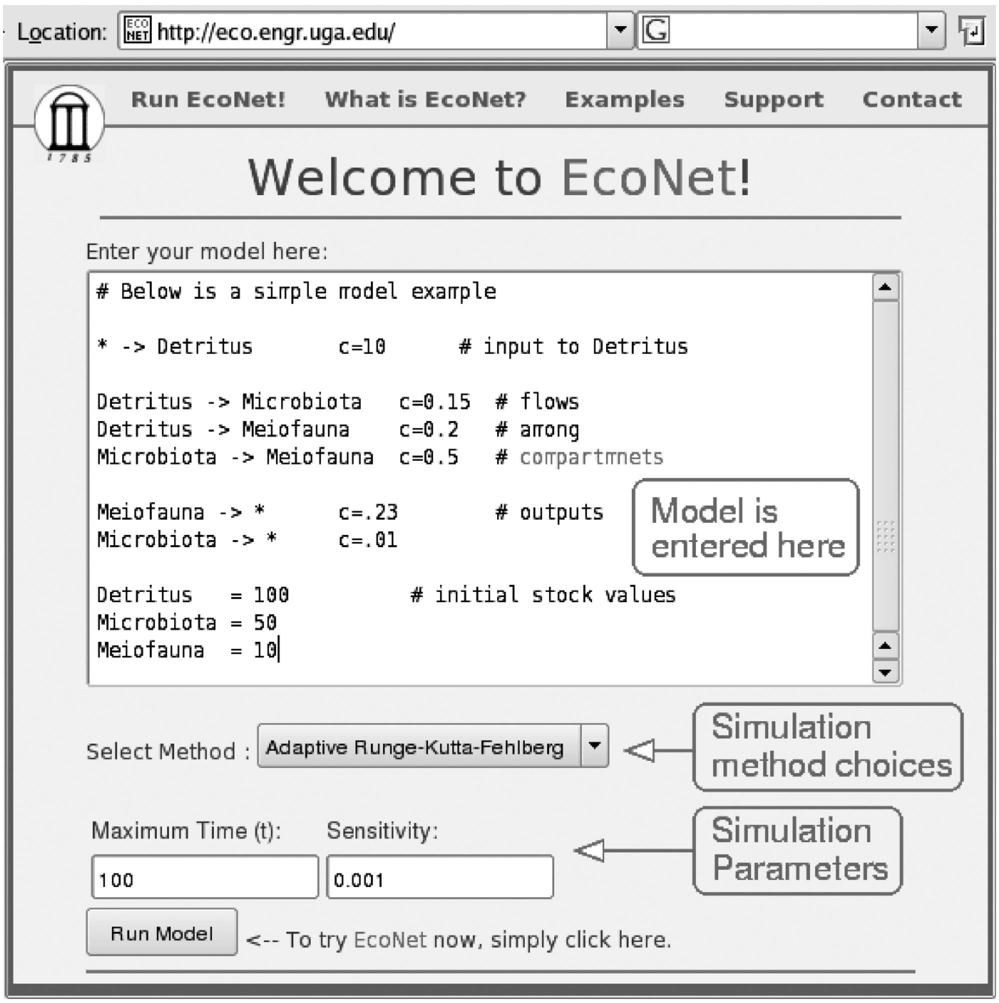
EcoNet is a dynamic simulation and network analysis software for ecological systems that can be expressed as a set of compartments and flows among these compartments. Flow currency can be energy, biomass, or a specific element such as C, N or P. Compartments can represent anything from accumulated organic matter to a group of species.

Actually, any process that can be represented as a stock-flow diagram, related to ecology or not, can be implemented in EcoNet within a few minutes. EcoNet performs deterministic or stochastic simulation from a given initial condition, and then performs ecological network analysis after the system reaches steady-state. Users can utilize EcoNet for simulation only and disregard the network analysis results. Or one can enter a model, which is already at steady-state and use EcoNet only for steady-state network analysis. EcoNet is a versatile software that offers unique features on both aspects.

EcoNet is designed to simplify the model building, simulation and analysis effort. The simplicity of EcoNet interface not only encourages first-time modelers to access a powerful modeling tool, but also minimizes the model-building effort for experienced users, closing the gap between the thought process and the results.

EcoNet runs on a server at http://eco.engr.uga.edu and requires no installation. Instead, users enter their models on a web browser, in a simple and intuitive text-based format (see Fig. 1). For user convenience, the website already contains a simple model, and default options for numerical methods and parameters are pre-selected. Anyone can try and run EcoNet immediately without needing any prior knowledge.

The user simply clicks on "Run Model" to submit a model to the EcoNet server. The server evaluates the model and unless there are errors detected in the model, a sequence of C++ codes, compiled Matlab routines and unix shell scripts work together to generate the simulation and analysis results. EcoNet server then creates a new web-page that contains these results which is then loaded into the user's browser. Typically this process takes less than 4 s.



```
# Below is a simple model example

* -> Detritus       c=10      # input to Detritus

Detritus -> Microbiota   c=0.15  # flows
Detritus -> Meiofauna    c=0.2   # among
Microbiota -> Meiofauna  c=0.5   # compartmnets

Meiofauna -> *      c=.23        # outputs
Microbiota -> *     c=.01

Detritus   = 100          # initial stock values
Microbiota = 50
Meiofauna  = 10
```

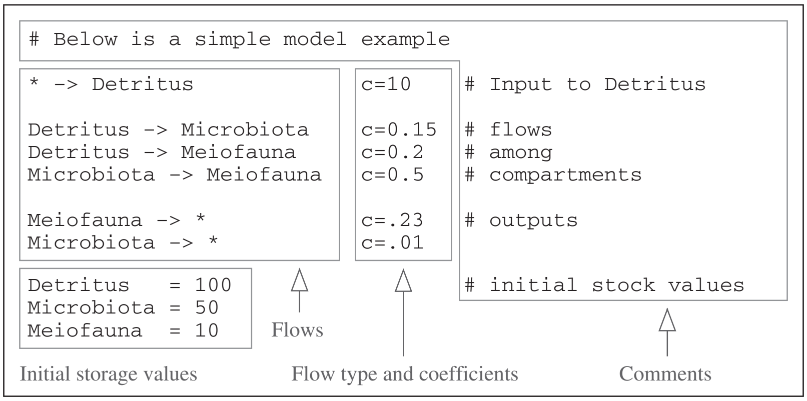
Select Method : Adaptive Runge-Kutta-Fehlberg

Maximum Time (t): 100 Sensitivity: 0.001

Figure 1: EcoNet web interface.

Designing ecological modeling software with a gentle learning curve for the novice user, and powerful features for the demanding expert user is a challenging task. To combine these two contradictory design criteria, we hid sophisticated mathematical techniques and efficient numerical methods behind a simple and flexible user interface. The novice user can use EcoNet without seeing a single differential equation, while an expert user can exploit EcoNet's numerical capabilities for demanding applications.

EcoNet was first online in June 2006. We initially developed its numerical engine back in 2001 at Carnegie Mellon University to analyze statistical properties of large biochemical networks. Failing to find software able to simulate biochemical networks involving over 10,000 molecules and reactions, we developed ours from scratch in C++. This numerical engine can handle non-linear models that contain up to 10^5 compartments and 10^5 flows. Besides deterministic methods, EcoNet features fast stochastic simulation algorithms based on the Langevin equation [1] and Gillespie's Stochastic Algorithm [2].



```
# Below is a simple model example
* -> Detritus                c=10     # Input to Detritus

Detritus -> Microbiota       c=0.15   # flows
Detritus -> Meiofauna        c=0.2    # among
Microbiota -> Meiofauna      c=0.5    # compartments

Meiofauna -> *               c=.23    # outputs
Microbiota -> *              c=.01

Detritus   = 100                      # initial stock values
Microbiota = 50
Meiofauna  = 10
```

Figure 2: A sample EcoNet model.

2 How to create an EcoNet model

During the design process of EcoNet interface, we had several choices for a model input method. Stella [3], a widely used dynamic simulation software, uses a graphical user interface (GUI) where users drag and drop geometrical shapes representing stocks and then define arrows connecting these stocks. EcoNet uses a flexible text-based model input method.

While a GUI-interface is intuitive and user friendly, it is not necessarily easier to use. Typing a text file is considerably quicker and easier than using a mouse to form a diagram. A model in text format is extremely portable and does not need special file formats to be saved or sent. It can be copied and pasted into other applications, and is readable. Furthermore, a text-based interface enables expert users to write codes to automatically generate very large models with thousands of stocks and flows for spatial modeling or statistical analysis.

2.1 EcoNet model structure

EcoNet models consist of four basic information types (Fig. 2), each associated with some special characters:

1. *Flows among compartments:* A directional flow from compartment *A* to compartment *B* is intuitively represented by "`A -> B`" Environment is represented by "`*`," so "`* -> C`" and "`D -> *`" represent the environmental input to compartment *C* and the dissipation from compartment *D*, respectively.
2. *Initial storage value of compartments:* These values are entered by simply typing the compartment name followed by "=" and then the numerical value. If the initial value of compartment *A* is 15 units, this can be represented by "`A = 15`".
3. *Flow type and associated coefficients:* In real life, speed of a flow between two compartments may depend solely on the storage value of the donor compartment, or the recipient compartment, or both (Lotka–Volterra). In some real-life cases, there is an upper-bound on the speed of flow no matter how high the storage values of both compartments are. There are even cases where the flow between two compartments is mediated by the storage value of a third compartment. We call these "different flow types," and naturally the simulation results will change dramatically based on what flow types are used.

EcoNet currently features donor controlled (mass-action type) and donor-recipient controlled (Lotka–Volterra) flow types. However, EcoNet is currently being updated to accommodate a variety of flow types, including Michaelis–Menten [4]. The notations for these flow types are already determined and are presented below, however only the first two are currently functional. We have plans to feature flow types beyond the three listed below. Users can find information on current updates at http://eco.engr.uga.edu/econet/news.html. Different flow types can be used in the same model.

- *Donor controlled flow:* If the speed of flow from compartment A to B is proportional to the storage value of A, the flow type is represented by "`c=2.3`", where 2.3 is a coefficient specific to this flow. In other words:

$$(\text{Flow rate } A \rightarrow B) = 2.3 \times (\text{ Storage value of } A).$$

- *Donor–recipient controlled flow:* If the speed of flow from compartment A to B is proportional to the storage values of both A and B, the flow type is represented by "`r=1.2`", where 1.2 is a coefficient specific to this flow. In other words:

$$(\text{Flowrate } A \rightarrow B) = 1.2 \times (\text{ Storagevalue of } A) \times (\text{ Storagevalueo of } B).$$

- *Michaelis–Menten type flow:* Originally developed for enzymatic reactions, this flow type is useful when flow speed is limited or mediated by other factors. The syntax for defining a Michaelis-Menten type flow from compartment A to B is "`v=3.2,2.1`", where coefficients 3.2 and 2.1 correspond to $V_{\max}$ and K_{m}, respectively. ($V_{\max}$ and K_{m} are two parameters associated with the Michaelis–Menten kinetics [4].) The speed of this flow is interpreted as:

$$(\text{Flow rate } A \rightarrow B) = \frac{3.2 \times (\text{Storage value of } B) \times (\text{Storage value of } A)}{2.1 + (\text{Storage value of } A)}.$$

4. *Comments:* EcoNet ignores anything written in a line after "#".

2.2 EcoNet model flexibility

Text based human–computer interactions are generally very demanding on the user's side. Sometimes, even a blank space must be accounted for when writing a computer code. However, EcoNet users can write their models with a great range of flexibility, as EcoNet does not require strict formatting rules. EcoNet recognizes each stock name, and classifies each user input as an initial condition, a flow, or a flow type. It is EcoNet that does the hard work, not the user. To demonstrate this feature, we re-write the same model in different but valid ways. Consider the following simple model:

```
* -> Phytoplankton          c=3
Phytoplankton ->
Zooplankton c=1
Zooplankton ->
Fish        c=.5
Fish ->
*                        c=.2
Phytoplankton = 10
Zooplankton  = 1
Fish        = 5
```

Although each flow and initial condition is written as separate lines, there is no such restriction. One can write multiple flows and initial conditions in a single line by separating them using commas or semicolons:

```
* -> Phytoplankton   c=3; Phytoplankton - >
Zooplankton c=1
Zooplankton -> Fish   c=.5, Fish   -> * c=.2
Phytoplankton = 10;  Zooplankton=  1, Fish =5
```

EcoNet ignores the order of appearance of flows and initial conditions. For better readability, one may choose to write all initial conditions grouped at the very end, or at the beginning. In case of a large model with multiple parts, users can form groups of flows and initial conditions corresponding to each part. Or one can choose to write initial conditions after each flow:

```
* -> Phytoplankton c=3
Phytoplankton = 10
Fish -> *  c=.2
Fish = 5
Phytoplankton -> Zooplankton c=1;
Zooplankton = 1
Zooplankton -> Fish c=.5
```

In rare cases, one may want to group flow types. This is also possible, and below is an example where there is almost no order; flows, initial conditions and flow types are all mixed together:

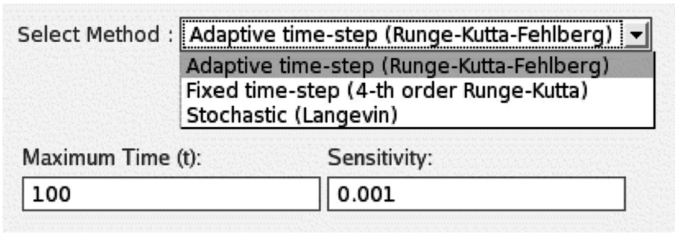
```
* -> Phytoplankton
c=3
Phytoplankton = 10
Phytoplankton ->
Zooplankton
Zooplankton= 1; c=1
Zooplankton -> Fish;
Fish=5
Fish -> *
c=.5, c=.2
```

Note that in this case, EcoNet will associate flow types to flows in the order of appearance. All this flexibility enables users to easily create and modify their models. Despite the fact that these models look very different, EcoNet generates the same equations for each of them.

2.3 A few rules about EcoNet models

Here are a few rules that users should keep in mind when writing their models.

- EcoNet models are case sensitive, so both `Zooplankton` and `zooplankton` cannot be used to refer to the same compartment. On the other hand, users can name two different compartments Zooplankton and zooplankton in the same model without causing any problems.
- Compartment names may contain numerical values in them, but cannot begin with one, so `Zooplankton1`, `Zoolplankton2` are valid compartment names, while `1Zooplankton1` is not.
- "`c`" cannot be used as a compartment name, as it is already used to represent donor-controlled flow type. Users should avoid using single lowercase letters to name compartments, as EcoNet will feature other flow types in the near future and may utilize other lowercase letters for this purpose. Capital single letters can be used without reservation.


Select Method : Adaptive time-step (Runge-Kutta-Fehlberg)

Adaptive time-step (Runge-Kutta-Fehlberg)
Fixed time-step (4-th order Runge-Kutta)
Stochastic (Langevin)

Maximum Time (t): 100

Sensitivity: 0.001

Figure 3: Available simulation methods in EcoNet.

- EcoNet treats the space and tab characters equally, and users can use them freely to create well organized models. However, no space should be used between a flow type and "=" character. So "`c=2`" and "`c= 2`" are valid, while "`c =2`" and "`c = 2`" will give errors. There is no restriction for initial condition definitions, so "`Zooplankton = 3`" is a valid description.
- All numerical values can be represented using scientific notation. EcoNet will interpret "`0.012`", "`1.2e-2`", "`.012`", "`.12e-1`" and "`12e-3`" as the same value 0.012.

While flexibility is a great feature, it also enables users to create disorganized but functional models. So, we equipped EcoNet with a good error tracking feature that generates meaningful error messages when EcoNet notices mistakes in the model. For example, misspelling a compartment name when defining its initial condition, say zoo-plankton, will receive the following complaint from EcoNet: "No initial condition exists for compartment zooplankton."

3 How to run an EcoNet model

After the model is entered, all the user has to do is to choose a simulation method, adjust the parameters if necessary, and hit "Run Model" to retrieve the results. Note that all the information EcoNet needs to run a simulation already exists in the model, so one may wonder why EcoNet even needs a simulation method and parameters?

Similar to other simulation software, EcoNet automatically converts the entered model into a differential equation system, the solution of which is the time-course data of storage values of all compartments. The existence of various methods and parameters is only because there is no magic numerical method capable of solving any given differential equation without human intervention. Depending on the complexity of the equation representing the model, the user may need to choose a different numerical method or adjust parameters. The good news is that EcoNet makes this an easy and simple task.

In most cases, the default method and parameters will work, and no adjustment will be necessary. The default method (Runge–Kutta–Fehlberg) is an adaptive algorithm, meaning that it continuously adjusts itself to the complexity of the solution on the fly. Any problems with the solution can be remedied simply by adjusting the *Sensitivity* parameter. This parameter is correlated to the amount of error allowed between the actual solution and the numerical solution. The smaller the *Sensitivity* parameter, the higher the accuracy. However, accuracy does not come free, as EcoNet will take longer to run when smaller *Sensitivity* values are used. Keep in mind that this relation is not linear, so decreasing *Sensitivity* by half might take EcoNet ten times longer to run.

The other parameter is *Maximum time* (not to be confused with the actual time it takes EcoNet to run the simulation.), which is basically the simulation length of the model in time units. In other words, it is the largest value on the *x*-axis of the time-course plot of all storage values

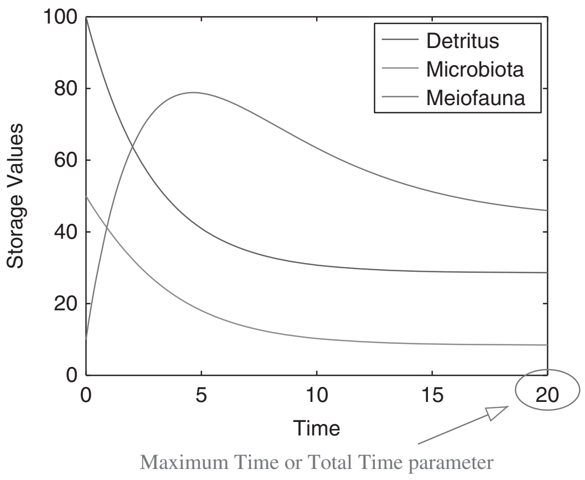


Figure 4: Time-course data of the simple EcoNet model given in Fig. 2.

(see Fig. 4). A particular time unit is not specified, because it has to be consistent with the user-defined flow coefficient units. So, we leave it up to the user to choose the appropriate set of units for their model. The higher the value of this parameter, the longer the computing time.

So, what if something goes wrong and one needs to adjust the simulation parameters to fix the situation, what would be the best strategy? There are only two problems that may occur:

1. *EcoNet does not return the results in a reasonable amount of time.* This means that the simulation takes too long to run. In this case, the user can either decrease the *Maximum time* or increase the *Sensitivity* parameter. As increasing *Sensitivity* will decrease accuracy, we recommend the former. However, if the simulation still takes a long time, the user should try decreasing *Sensitivity*, as very high accuracy is not needed in most cases.
2. *EcoNet runs but the results look unrealistic.* This means that the numerical solution is not accurate. The solution is to decrease the *Sensitivity* parameter. This action will increase accuracy, but EcoNet will take longer to finish the simulation. If it takes too long, then users should decrease the *Maximum time* parameter.

Note that there is a trade-off between accuracy, computing time and *Maximum time.* As we need accuracy, and cannot wait for hours for a simulation, it is wise to start with a smaller *Maximum time* parameter. EcoNet will always return accurate results in a couple of seconds for small enough *Sensitivity* and *Maximum time* parameters, no matter how complicated the model is. The numerical engine of EcoNet is much faster than that of commercially available software such as Matlab and Stella, so finding a good balance between accuracy and speed is not a difficult task.

3.1 Fourth-order Runge–Kutta method

So far, we only talked about the default simulation method, the adaptive Runge–Kutta–Fehlberg algorithm. The second method EcoNet features is the fourth-order Runge-Kutta algorithm (Table 1). This is a fixed-step size method; so unlike the default method, it will not adjust its accuracy based on the complexity of the differential equation. This seemingly inferior method

Table 1: Available numerical methods and associated parameters.

Method type	Method name	Parameters
Adaptive time-step	Runge–Kutta–Fehlberg	*Maximum time, Sensitivity*
Fixed time-step	Fourth-order Runge-Kutta	*Total time, Step size*
Stochastic	Langevin equation	*Total time, Step size*

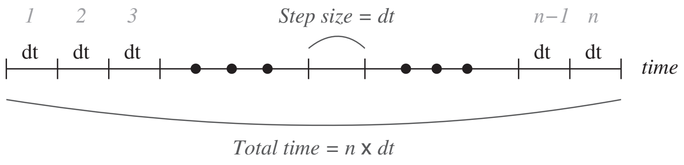
is provided for EcoNet users who want more control over the numerical simulation. When this method is selected, EcoNet will use the step-size defined by the user throughout the simulation, instead of trying to optimize it on the fly.

The strategy to adjust the parameters for this method is exactly the same as the previous one, the only difference being the name of the parameters. The *Sensitivity* parameter is replaced by the *Step size* parameter. Both represent solution accuracy in a similar manner; the smaller the *Step size*, the higher the solution accuracy.

3.2 Numerical solution methods

Understanding the meaning of step size requires some numerical analysis knowledge. Although such knowledge is not necessary to run EcoNet, we would like to provide some additional information for the interested user. We have already mentioned that EcoNet converts the model into a differential equation system.

This differential equation can be fairly complex, which makes it extremely hard to find the exact analytic solution. Therefore, all simulation software uses numerical solution methods, which are iterative computer algorithms that construct approximate solutions. A successful numerical solution will have negligibly small error relative to the analytic solution. Fixed time-step methods construct the solution step-by-step in time, assuming that the solution does not change rapidly during each time step.



This assumption is the main reason for error; rapid changes in the solution may be missed if large step-sizes (*dt*) are used. Therefore, small step-sizes (*dt*) are desirable for accuracy, but using extremely small values will increase the computation time if *Total time* remains the same:

$$\text{Number of iterations } (n) = \frac{\textit{Total time}}{\textit{Step size } (dt)}.$$

Adaptive numerical methods, such as Runge–Kutta–Fehlberg, have an error detection mechanism that continuously monitors the solution accuracy at each iteration. If it detects that the error is larger than the *Sensitivity* parameter defined by the user, it automatically decreases the step-size and repeats the iteration. Similarly, if it senses that the accuracy is much smaller than the *Sensitivity* parameter, it increases the step-size. This way, an adaptive scheme continuously optimizes speed for a given accuracy.

Although this sounds like the perfect numerical method, the weak link in adaptive methods is the error detection mechanism. One can never find the exact error without knowing the exact solution. When the "detected" error and the actual error are not the same, the solution may not be accurate, or the simulation may take longer. However, this rarely happens, and adaptive schemes are preferred over fixed step-sized methods in general. We refer the interested users to numerical analysis texts for more information [5].

3.3 Stochastic method

The third simulation method offered by EcoNet is a stochastic algorithm. Stochastic solvers take the probabilistic system behavior into account and generate different solutions at each run. It seems that stochastic solvers are producing random results. On the contrary, in most cases, stochastic solutions are more accurate because no real biological or ecological system is deterministic in nature. It only makes sense to use a stochastic solver for a system, which is stochastic in nature.

EcoNet features a very fast stochastic method based on the Langevin equation [1]. (Published in 2000, this method is based on a Fokker–Planck type partial differential equation (PDE) derived from the master equation, which is then converted to a stochastic differential equation (SDE).) It is as easy to use as the previous two methods. The parameters, *Total time* and *Step size,* are exactly the same as in the fourth-order Runge–Kutta method, and so is the strategy to adjust the parameters for accurate results.

Users should not confuse a stochastic solution with a deterministic solution driven by a noisy input, or a deterministic solution perturbed randomly at each iteration step. Although these practices will generate random behaving solutions, the noise-term associated with these solutions is wrong, which negates the very advantage of using a stochastic solution method in the first place. This is not a trivial fact, and we refer the interested user to further reading [1, 6]. In simple terms, a stochastic process representing a stock-flow model should be compatible with the so-called *master equation* [7].

This is not an easy condition to satisfy, and true stochastic solvers are complex and hard to implement. This is the main reason why very few software programs feature stochastic methods. As they are not widely available, few users are aware of their power. For example, a single stochastic solution will reveal the inherent variations in the stock values, eliminating the need for many repeated runs for sensitivity analysis.

It seems that stochastic solutions always look like the noisy versions of their deterministic versions (see Fig. 5). This is not true in general; there are cases where only a stochastic method

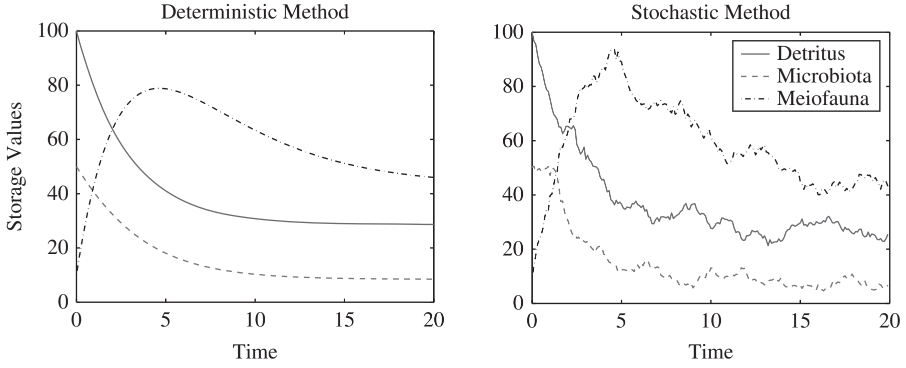


Figure 5: Time-course data of a simple EcoNet model produced using the fourth-order Runge–Kutta method and the stochastic method.

will provide an accurate representation of the modeled system. For example, if the flowing matter in the system exists in scarce discrete quantities, or if the system is capable of multiple steady states, deterministic and stochastic methods may differ significantly.

3.4 From model to differential equation

EcoNet automatically converts the entered model into a differential equation system, and solves it using the chosen method and parameters. This equation system is never displayed explicitly, and there is no need to do so. Still, here we explain how EcoNet does this conversion for the interested user. Let us consider the following simple model:

```
* -> Detritus     c=10        # input to
Detritus
Detritus -> Microbiota  c=0.5 # flows
Detritus -> Meiofauna   c=0.2 # among
Microbiota -> Meiofauna c=0.5 # compartments
Meiofauna -> *    c=.23       # outputs
Microbiota -> *   c=.01
Detritus = 100                    # initial
values                             storage
Microbiota = 50
Meiofauna = 10
```

Let us focus on the compartment `Microbiota`. The storage value of Microbiota is controlled by the three underlined flows. The first flow is an input to Microbiota, while the last two are outputs from Microbiota. Therefore, we can express the storage value change of the Microbiota compartment as follows:

$$\begin{aligned}\frac{d[\texttt{Microbiota}]}{dt} &= (\text{Inputs to } \texttt{Microbiota}) - (\text{Outputs from } \texttt{Microbiota})\\ &= (\text{Speed of flow } \texttt{Detritus} \rightarrow \texttt{Microbiota})\\ &\quad - (\text{Speed of flow } \texttt{Microbiota} \rightarrow \texttt{Meiofauna})\\ &\quad - (\text{Speed of flow } \texttt{Microbiota} \rightarrow *).\end{aligned}$$

The algebraic expression of flow speeds is determined by the flow type, and the repeated use of "`c=`" represents that the only flow type used in this model is donor controlled flow (mass action). Then, based on our previous discussion of flow types, the complete differential equation system is going to be:

$$\begin{aligned}\frac{d[\texttt{Detritus}]}{dt} &= 10 - 0.15[\texttt{Detritus}] - 0.2[\texttt{Detritus}]\\ \frac{d[\texttt{Microbiota}]}{dt} &= 0.15[\texttt{Detritus}] - 0.5[\texttt{Microbiota}] - 0.01[\texttt{Microbiota}]\\ \frac{d[\texttt{Meiofauna}]}{dt} &= 0.2[\texttt{Detritus}] + 0.5[\texttt{Microbiota}] - 0.23[\texttt{Meiofauna}].\end{aligned}$$

The environment is not associated with a limited storage value, so the input from the environment to Detritus is simply represented with the constant 10 in the first line of the equation above. Just as an example, if the third line of the model were replaced with

```
Detritus -> Meiofauna  r=0.2
```

meaning that the flow is not donor-controlled but both donor and recipient-controlled (Lotka–Volterra), EcoNet would construct the following differential equation system:

$$\frac{d[\texttt{Detritus}]}{dt} = 10 - 0.15[\texttt{Detritus}] - 0.2\,[\texttt{Detritus}][\texttt{Meiofauna}]$$

$$\frac{d[\texttt{Microbiota}]}{dt} = 0.15[\texttt{Detritus}] - 0.5[\texttt{Microbiota}] - 0.01[\texttt{Microbiota}]$$

$$\frac{d[\texttt{Meiofauna}]}{dt} = 0.2\,[\texttt{Detritus}][\texttt{Meiofauna}] + 0.5[\texttt{Microbiota}] - 0.23\,[\texttt{Meiofauna}].$$

4 Simulation and analysis results

Once the user clicks on “Run Model”, EcoNet converts the model into a deterministic or a stochastic differential equation system, then finds the numerical solution using the selected method and parameters, and performs various network analyses based on the final state of the system. In this section, we will go over these results for a simple model that we previously presented in Section 3.4 and in Fig. 5. We use the default simulation method and parameters, so the adaptive step-size method (Runge–Kutta–Fehlberg) was selected with parameters *Maximum time* = 100 and *Sensitivity* = 0.001.

4.1 Network diagram

The first result displayed by EcoNet is a network diagram of the model (Fig. 6). A network diagram provides important visual information, which is lacking in the text-based EcoNet model format. Some modeling software use a graphical model building interface where users drag and drop geometrical shapes representing compartments and then define arrows connecting these compartments. These interfaces provide visual information, however, for slightly larger systems, the finished model ends up having too many flow lines crossing each other, creating a messy visual diagram. It becomes harder to build the model, and the final look does not provide much visual information.

A useful diagram should provide insights into system structure and behavior by locating important stocks with many connections in the center, while keeping stocks with fewer connections closer to the edges. Flow lines should be short with little curvature, and should not intersect often. For large systems, tightly coupled compartments should be located close to each other, forming “sub-systems.” And if possible, the locations of compartments should reflect their trophic level, even in the case of cross-level feeding.

These goals are very hard to achieve for the user who is building the model, and actually requires a sophisticated optimization algorithm. Therefore, we designed EcoNet to create a desirable network graph using Graphviz [8], a professional open source graph visualization software, to determine the optimum placement of stocks and flow lines to achieve the listed objectives. EcoNet diagrams are designed to look as clean as possible, and will work even in the case of large models with over 50 compartments and 100 flows. Trophic level of compartments will be well represented regardless of model size.

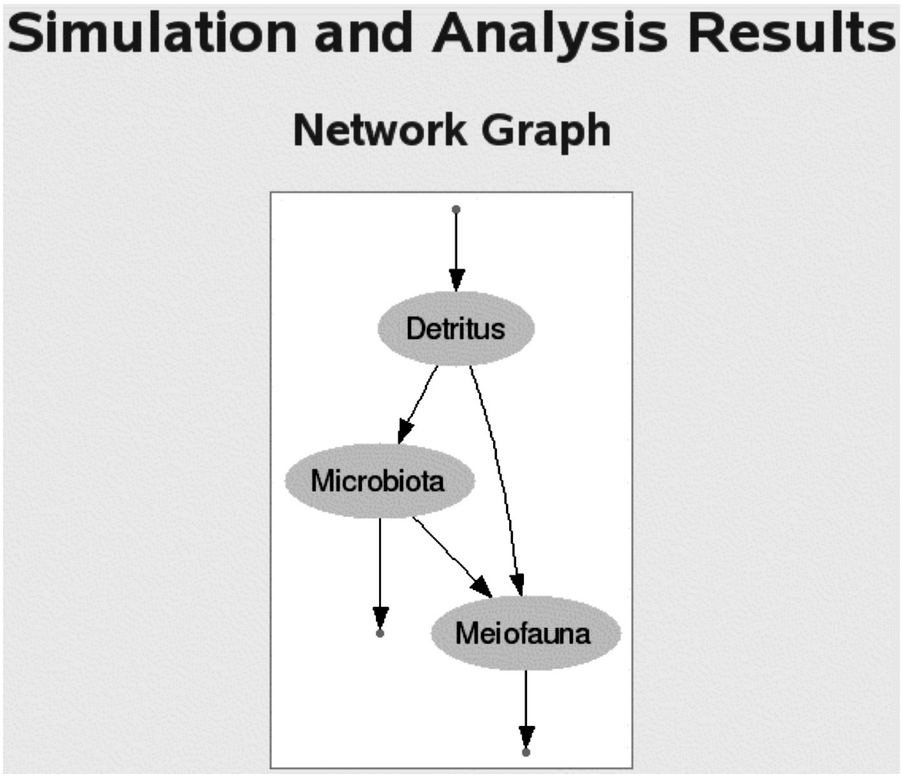
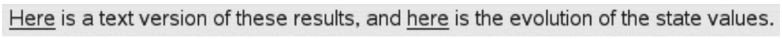


Figure 6: Network diagram.

4.2 Time-course plot and data of compartment storage values

The second figure displayed by EcoNet is the time-course plot of compartment storage values (Fig. 7). EcoNet also provides access to the time-course data via an html-link located toward the end of the page:


Here is a text version of these results, and here is the evolution of the state values.

The second link here points to a text file named “state.dat”. This file contains a matrix of numerical values that EcoNet uses to form the time-course plot.

Users can view the contents of the file by simply clicking on the link, or they can save the file by right-clicking and choosing the “Save Link As...” option.

This feature enables the user to import the time-course data into other software, like Matlab, Octave or Excel (Fig. 7). Therefore, users can easily do further analysis, or recreate the same plot using a graphing software of their choice. This is particularly important because it enables users to zoom on the graph, create publication quality figures, or plot only the compartments of interest. Therefore, we would like to demonstrate how easy this is, using Matlab or Octave (a free scientific computing software similar to Matlab). Matlab and Octave are available for Windows, Mac and Unix; and the following procedure is the same for all platforms:

1. Start Matlab or Octave.
2. Download the EcoNet data as described above.

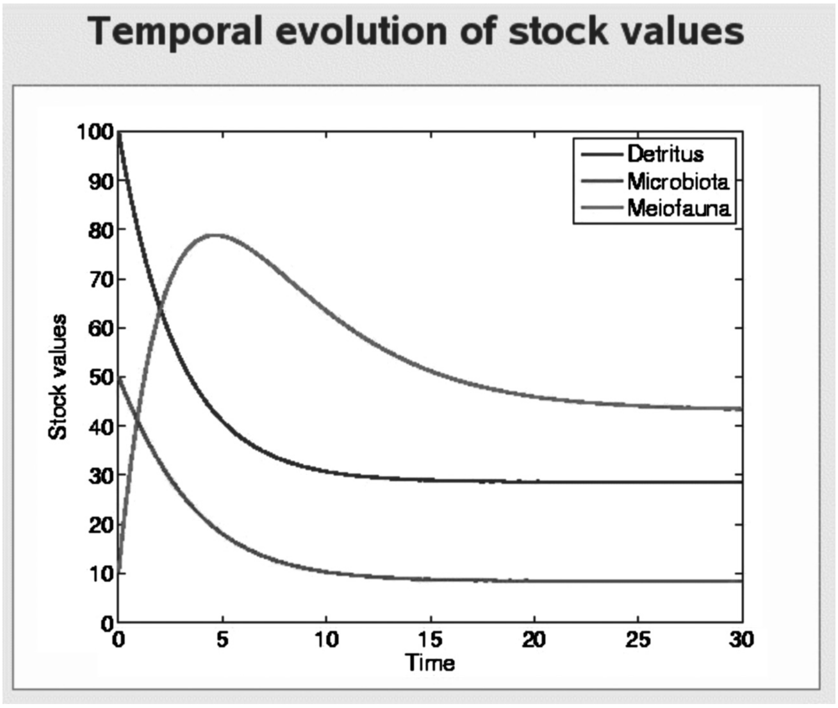


Figure 7: Time-course plot of compartment storage values. Note that although we set *Maximum time* = 100, the simulation ends at *time* = 30. When the *adaptive time-step* method is selected, EcoNet terminates the simulation earlier, if it detects that the system reaches steady state.

3. Copy the download file "state.dat" to Matlab/Octave's current working directory.
4. Type the following on Matlab/Octave prompt:

```
> s = load('state.dat');
```

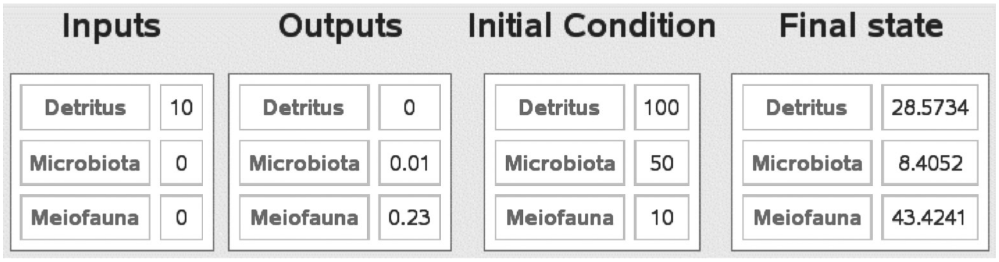
Here "s" is a matrix of time-course data, where the columns represent each compartment and rows represent different time values. Users can easily manipulate this matrix in Matlab/Octave environment for further analysis. To recreate the time-course plot, simply type:

```
> plot(s);
```

Users can easily zoom into this plot, add annotations, and save it in various formats. Plotting only specific compartments comes in handy when the EcoNet plot is dominated by compartments with high storage values. To plot only the first compartment, simply type:

```
> plot(s(:,1));
```

A lot more can be done in the Matlab/Octave environment, like creating multiple plots for each compartment. However, we will not go into further detail, as documentation and many tutorials are available in print and online.



Inputs		Outputs		Initial Condition		Final state	
Detritus	10	Detritus	0	Detritus	100	Detritus	28.5734
Microbiota	0	Microbiota	0.01	Microbiota	50	Microbiota	8.4052
Meiofauna	0	Meiofauna	0.23	Meiofauna	10	Meiofauna	43.4241

Figure 8: Inputs, outputs, initial and final states of each compartment.

4.3 Model information

The rest of the EcoNet results are expressed in vector or matrix format. They provide information on model structure and function, and present network analysis results. The time-course plot is followed by four vectors that contain the following information (Fig. 8):

1. inputs to compartments;
2. outputs from compartments;
3. initial storage values of compartments;
4. storage values of all compartments at the end of the simulation.

4.4 Adjacency matrix

Next, EcoNet displays the adjacency matrix A (Fig. 9), which consists of 0 and 1 entries and indicates if there exists a direct flow between two stocks. In other words,

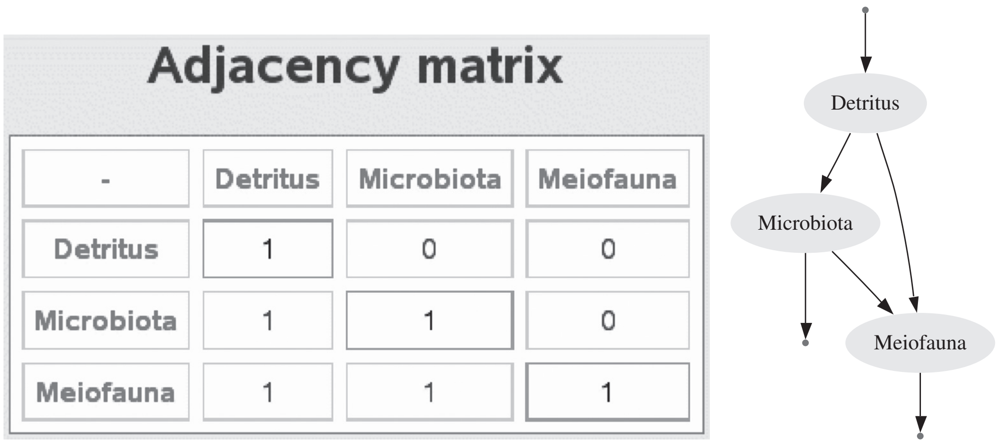
$$a_{ij} = \begin{cases} 1, & \text{if there is a direct flow from compartment } j \text{ to compartment } i \\ 0, & \text{if there is no direct flow from compartment } j \text{ to compartment } i. \end{cases}$$

The definition of A is relatively simple; however, it contains important information regarding the model structure. A sparse adjacency matrix is an indication of a loosely connected system. A row with few non-zero entries indicates the existence of a compartment receiving inputs from many other compartments. Similarly, A will contain a column with few non-zero entries if there exists a compartment with outputs to many other compartments. If A has almost all zero entries above the diagonal, then the model has a tree-structure with very little feedback. If there are well-connected smaller subsystems in a large model, A will contain dense non-zero square blocks close to the diagonal.

4.5 Flow coefficient matrix

The adjacency matrix only contains connectivity information, but does not provide any information on connection strength. Therefore, EcoNet results also include the flow coefficient matrix C (Fig. 10), which can be viewed as a "weighted" version of A. C is also called the *community matrix* and is defined only if the donor controlled flow type is used throughout the system;

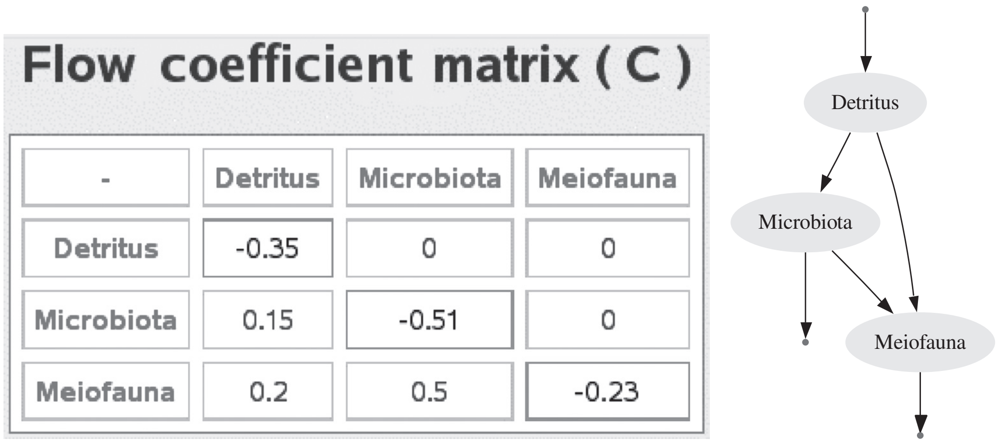
$$c_{ij} = \text{coefficient of the flow from compartment } j \text{ to compartment } i.$$



Adjacency matrix

-	Detritus	Microbiota	Meiofauna
Detritus	1	0	0
Microbiota	1	1	0
Meiofauna	1	1	1

Figure 9: Adjacency matrix.



Flow coefficient matrix (C)

-	Detritus	Microbiota	Meiofauna
Detritus	-0.35	0	0
Microbiota	0.15	-0.51	0
Meiofauna	0.2	0.5	-0.23

Figure 10: Flow coefficient matrix.

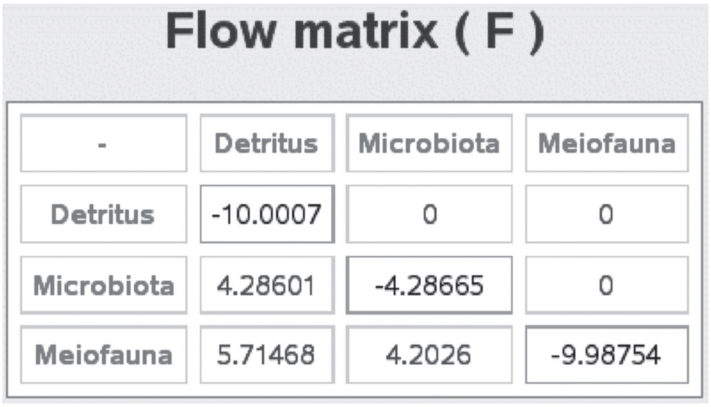
If no such flow exists, then $c_{ij} = 0$. Note that the coordinates of non-zero entries are the same in matrices A and C. The diagonal entries of C are defined as follows:

$$c_{ii} = -(\text{sum of the flow coefficients of all outputs from compartment } i).$$

When the diagonals are defined this way, the differential equation system representing the model can be expressed in a compact form:

$$\frac{dx}{dt} = Cx + z,$$

where x is the storage value vector, and z is the input vector (Fig. 8). The explanation of this equation requires some linear algebra knowledge, and we refer the interested user to further reading [9, 10].



Flow matrix (F)

-	Detritus	Microbiota	Meiofauna
Detritus	-10.0007	0	0
Microbiota	4.28601	-4.28665	0
Meiofauna	5.71468	4.2026	-9.98754

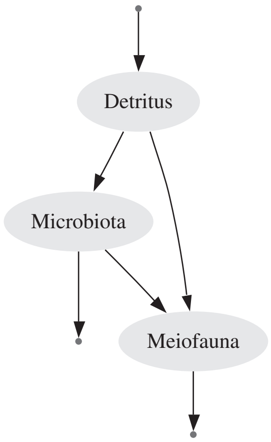


Figure 11: Flow matrix.

4.6 Flow matrix

In addition to connectivity information, *C* provides information on how strong the connections are. When defining *C*, we use the flow coefficients as an indication of the connection strength. Another important measure of connection strength is the flow rate between compartments at steady-state, and this is exactly what the flow matrix *F* provides (Fig. 11);

$$f_{ij} = \text{rate of flow from compartment } j \text{ to compartment } i \text{ at steady-state.}$$

As the compartment storage values change in time, the rate of flow between compartments also change accordingly. Flow rates reach a steady-state with the storage values. EcoNet computes *F* using the steady-state storage values, flow types and the associated flow coefficients. And if the simulation ends before the system reaches steady-state, EcoNet computes the flow matrix based on the final state of the system (Fig. 8). Users can use the time-course plot to see how close the system is to steady-state. (Note that the solution never reaches the exact steady-state, but becomes asymptotically close.)

It seems that *F* and *C* must be somewhat correlated, but in fact, they may differ significantly. Here is an example from Figs. 10 and 11:

$$0.15 = c_{21} < c_{32} = 0.50$$
$$4.286 = f_{21} > f_{32} = 4.202.$$

In general, the following relation holds for a donor-controlled flow from compartment *j* to compartment *i:*

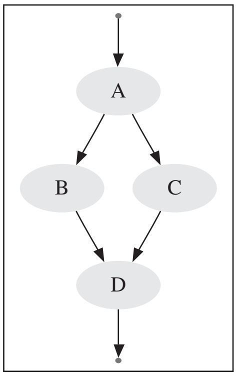
$$f_{ij} = c_{ij} x_j.$$

Here, x_j denotes the storage value of compartment *j* at steady state. Note that *C* is well-defined, only if a donor controlled flow type is used throughout the system. *F* does not have this requirement. If a Lotka–Volterra type flow (donor and recipient controlled) is used from compartment *j* to *i*, then:

$$f_{ij} = r x_i x_j.$$

Here, *r* represents the donor and recipient-controlled flow coefficient from compartment *j* to compartment *i*.

4.7 Network analysis



The flow matrix *F* represents the connection strength between compartments at steady-state. However, even if two compartments are not directly connected, they can affect each other through indirect connections involving other compartments. For example, there are no direct flows between compartments *A* and *D* in the next figure. However, any change in compartment *A* will eventually affect compartment *D*. In this regard, all the previous matrices (*A*, *C* and *F*) fail to represent the true connectivity relation among compartments. In a way, even the model diagram (Fig. 6) can be viewed as misleading because it only represents the direct flows. Particularly in well-connected systems, there is often a greater contribution from indirect flows than from direct. This property is called the *dominance of indirect effects* [9–11].

EcoNet uses Network Environ Analysis (NEA) [9, 10] to quantify the actual relation among compartments, environmental inputs and outputs. Unlike most analysis methods, NEA treats the system as a whole and provides an elegant way to quantify the effects of all indirect flows in the system. NEA is not a one-step analysis, but a series of algebraic operations resulting in scalar and matrix values representing various system-wide properties. EcoNet currently includes *storage* (*S*), *throughflow* (*N*) and *utility* (*U*) analyses.

We should note that NEA is valid only when the system is close to steady-state. EcoNet displays NEA results based on the final state of the system. To get accurate results, users should use the time-course plot to make sure that the system is close to steady-state at the end of the simulation.

4.8 Storage analysis

Storage analysis matrix *S* represents the relation between input flow rates and compartment storage values. In Fig. 12, only Detritus receives a direct environmental input. This input is partially transferred to Microbiota and Meiofauna through indirect flows. S_{ij} represents how much of the storage value of compartment *i* is contributed by a unit of direct environmental input to compartment *j*.

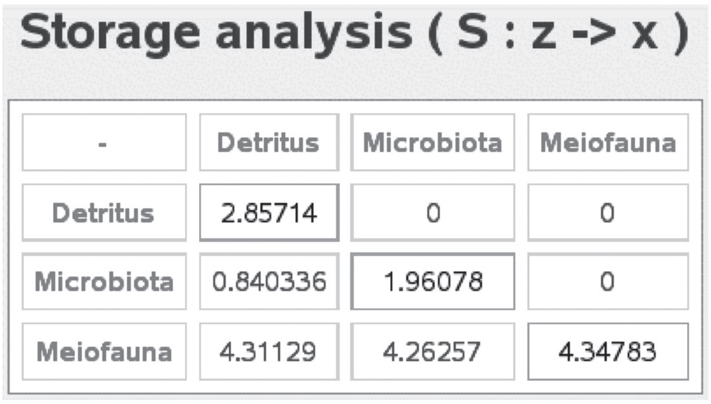
Let us consider the first row of the storage matrix in Fig. 12:

$$s_{11} = 2.857 \quad s_{12} = s_{13} = 0.$$

There are no environmental inputs into Microbiota and Meiofauna; therefore, the storage value of Detritus depends only on its own environmental input ($s_{12} = s_{13} = 0$). The environmental input rate to Detritus is 10 units/time, and the steady-state storage value of Detritus is 28.57 units (see Fig. 8). Therefore, for 1 unit environmental input to Detritus contributes to s_{11} = 2.857 units of storage value in Detritus.

Similarly, the storage value of Microbiota (8.4 units) is all contributed by the environmental input to Detritus; therefore s_{21} = 8.4/10 = 0.84. Although the environmental input to Microbiota is 0 units/time in our model, s_{22} = 1.96 means that each unit of direct environmental input to Microbiota would contribute to s_{22} = 1.96 units of storage value. The fact that s_{23} = 0 informs us that an environmental input to Meiofauna will never affect the storage value of Microbiota.

We should note that if there were an additional flow from Meiofauna to Detritus, there would be no zero entries in *S*, as any environmental input would cycle through all compartments, contributing to all storage values.



Storage analysis (S : z -> x)

-	Detritus	Microbiota	Meiofauna
Detritus	2.85714	0	0
Microbiota	0.840336	1.96078	0
Meiofauna	4.31129	4.26257	4.34783

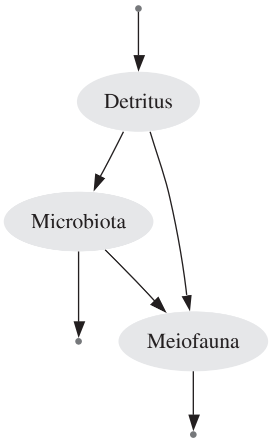


Figure 12: Storage analysis matrix.

We have not discussed how to compute S, and we refer the interested user to references [9, 10]. In mathematical terms, S is a linear mapping from environmental inputs to compartment storage values:

$$S : \overline{z} \rightarrow \overline{x} \quad \overline{x} = S\overline{z}.$$

Here, x represents the vector of all steady-state storage values, and z represents the vector of direct environmental inputs to each compartment.

4.9 Throughflow analysis

The throughflow value of compartment i is defined as the total amount of mass or energy units received by (or lost from) that compartment per unit time at steady-state:

$$T_i = \sum_j f_{ij}.$$

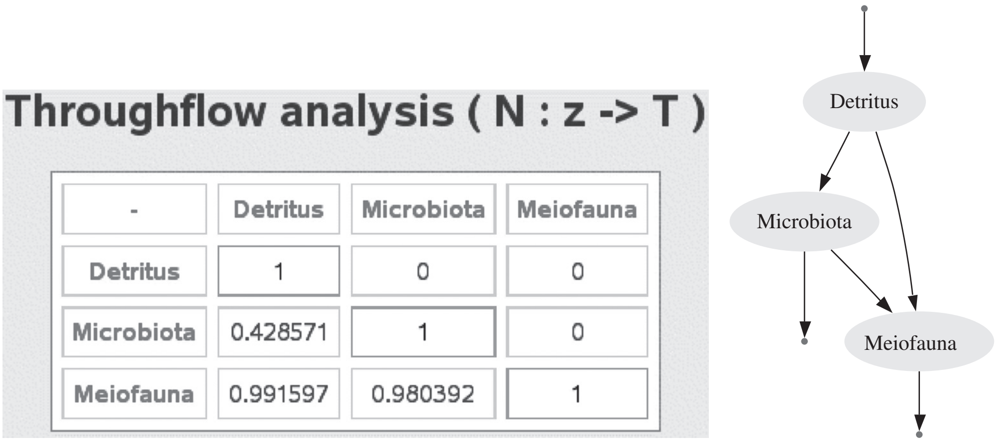
Throughflow analysis is conceptually the same as the storage analysis, the only change being the use of throughflow values instead of storage values. n_{ij} represents how much of the environmental input to compartment j is received by compartment i. Note that this input may reach compartment i through indirect flows that involve many other compartments, and N_{ij} accounts for all such possible paths.

Let us consider the first column of matrix N given in Fig. 13. Obviously, 100% of the environmental input to Detritus is received by Detritus; therefore, $n_{11} = 1$. Coefficient $n_{21} = 0.428$ means that 42.8% of the environmental input to Detritus is received by Microbiota, then 57.2% flows to Meiofauna.

Coefficient $n_{31} = 0.991$ represents the fact that Meiofauna eventually receives 99.1% of the environmental input (to Detritus), through Microbiota and directly from Detritus. Then we can also conclude that 0.9% of the environmental input returns to the environment through dissipation at Microbiota.

To compare the effect of direct and indirect paths from an input to a compartment, EcoNet displays another matrix B, defined as follows:

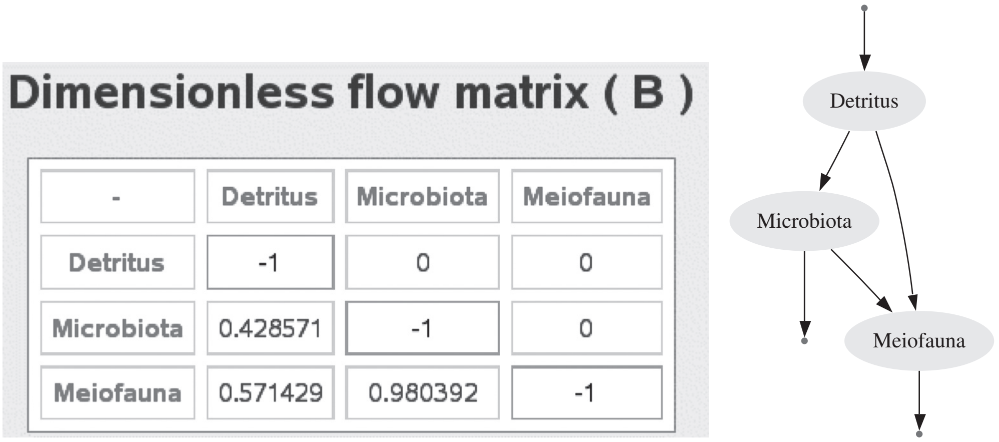
$$b_{ij} = f_{ij}/f_{jj}.$$



Throughflow analysis (N : z -> T)

-	Detritus	Microbiota	Meiofauna
Detritus	1	0	0
Microbiota	0.428571	1	0
Meiofauna	0.991597	0.980392	1

Figure 13: Throughflow analysis matrix.



Dimensionless flow matrix (B)

-	Detritus	Microbiota	Meiofauna
Detritus	-1	0	0
Microbiota	0.428571	-1	0
Meiofauna	0.571429	0.980392	-1

Figure 14: Normalized flow matrix *B*. By definition, entries of *B* do not have units.

B can be viewed as a normalized *F* matrix; where each entry f_{ij} is divided by the diagonal element of the row to which it belongs. While f_{ij} represents the actual flow rate from *j* to *i* at steady-state, b_{ij} represents what ratio of the throughflow of *j* is received by *i*. Note that both *N* and *B* contain normalized values representing how energy or mass units are distributed among compartments. However, *N* accounts for all possible direct or indirect flows between compartments, while *B*, by definition, only represents direct flows among compartments.

Comparing *N* and *B* matrices (Figs. 13 and 14), we see that entries of both matrices match except for:

$$0.991 = n_{31} \neq b_{31} = 0.571.$$

This is expected for a simple three-compartment model where there is one indirect flow. There are two paths from Compartment 1 (Detritus) to Compartment 3 (Meiofauna):

$$\text{Direct flow: } \textit{Detritus} \rightarrow \textit{Meiofauna}$$

$$\text{Indirect flow: } \textit{Detritus} \rightarrow \textit{Microbiota} \rightarrow \textit{Meiofauna}.$$

Both are accounted for in $n_{31} = 0.991$, while $b_{31} = 0.571$ only represents the direct flow. Then we can conclude that $[(0.991 - 0.571)/0.991] \times 100 = 42\%$ of the interaction between Detritus to Meiofauna occurs over the indirect connection.

In mathematical terms, N is a linear mapping from direct environmental inputs to throughflow values:

$$N : \bar{z} \to \bar{T}, \quad \bar{T} = \mathrm{N}\bar{z}.$$

Here, $\bar{T}$ represents a vector of throughflow values. The computation of N is very similar to S, and we refer the interested user to references [9, 10].

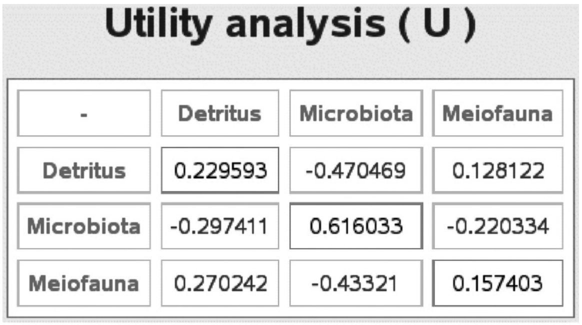
4.10 Utility analysis

Utility analysis is a rather involved part of NEA. It is a powerful method that provides "relationships" among compartments, again including indirect effects. In the following example

$$* \to \textit{tree} \to \textit{deer} \to \textit{wolf} \to *,$$

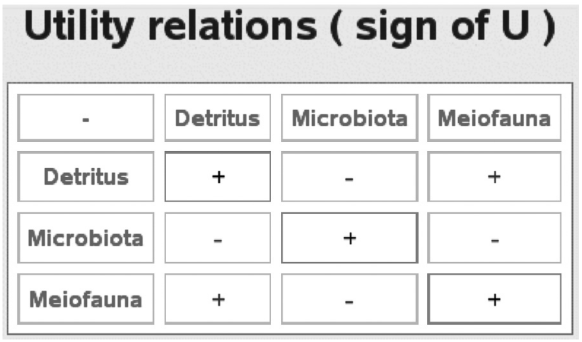
deer and *wolf* have a (–, +) relationship, same as the *tree–deer* relationship. Although not connected with a direct flow, the *tree* and the *wolf* have a mutualistic (+, +) relationship. Figuring out such relationships is straightforward for simple models, but extremely difficult when models involve feedback loops, cycling or cross-level feeding. Utility analysis provides this relationship among all compartments regardless of model complexity.

The second matrix in Fig. 15 represents the relationships among compartments in +/– format, while the first matrix U provides the relationship strength as well. The three-compartment model



Utility analysis (U)

-	Detritus	Microbiota	Meiofauna
Detritus	0.229593	-0.470469	0.128122
Microbiota	-0.297411	0.616033	-0.220334
Meiofauna	0.270242	-0.43321	0.157403



Utility relations (sign of U)

-	Detritus	Microbiota	Meiofauna
Detritus	+	-	+
Microbiota	-	+	-
Meiofauna	+	-	+

Figure 15: Utility analysis matrix, and the sign of *U*.

we provide here is too simple to present the strength of utility analysis. We postpone this discussion to the next section where we analyze John M. Teal's Georgia salt marsh model. For further details on utility analysis, and NEA in general, we refer the reader to references [9–12].

5 Study of an EcoNet model

This section is contributed by Gaston Small, Institute of Ecology, University of Georgia.

We present an eight-compartment model based on John M. Teal's classic study of energy flow in a Georgia salt marsh [13, 14] (Fig. 16). Followed by the model description, we provide the EcoNet model and go over to essential network analysis results.

5.1 Model description

The model represents the energy budget for one square meter of salt marsh. Compartments are measured in kcal/m^2 and flows are measured in kcal/m^2 d.

Energy enters the salt marsh through primary production at compartments salt marsh cordgrass *Spartina* and algae. Much of this energy is lost through respiration by these plants, but the remainder enters the salt marsh food web. Bacteria and insects feed directly on the *Spartina*, and spiders, in turn, feed on the insects.

A substantial portion of the *Spartina* dies and enters the detritus pool. Nematodes feed on bacteria and detritus, and mud crabs feed on the nematodes. Every compartment in the model

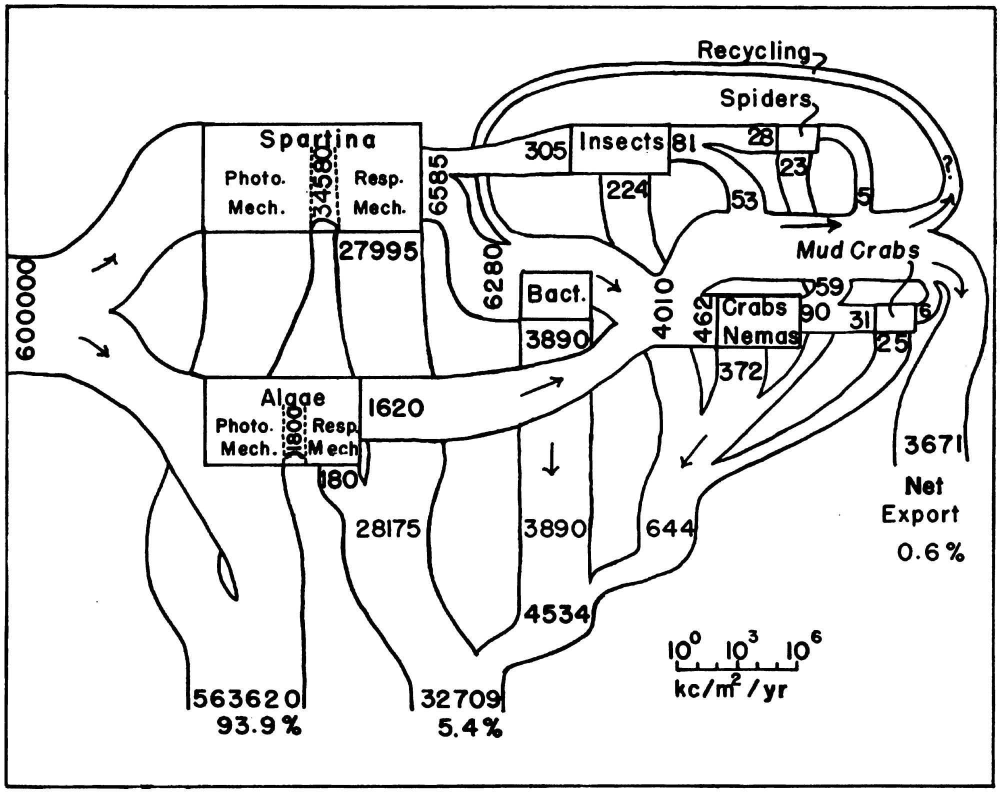


Figure 16: Energy flow diagram of a Georgia salt marsh, by J.M. Teal [13].

```
# Energy flow in the salt marsh
# ecosystem by G. Small,
# based on model by J. Teal

* -> spartina c=94.74
* -> algae c=4.93
spartina -> insects c=0.002
spartina -> detritus c=0.03
spartina -> bacteria c=0.0001
algae -> detritus c=0.02
algae -> nematodes c=0.02
insects -> detritus c=0.2
insects -> spiders c=0.05
detritus -> bacteria c=0.01
detritus -> nematodes c=0.01
bacteria -> nematodes c=0.01
bacteria -> detritus c=0.5
spiders -> detritus c=0.1
nematodes -> detritus c=0.6
nematodes -> mudcrabs c=0.01
mudcrabs -> detritus c=0.1
spartina -> * c=0.15
algae -> * c=0.1
insects -> * c=.1
detritus -> * c=0.02
bacteria -> * c=0.4
spiders -> * c=0.05
nematodes -> * c=0.2
mudcrabs -> * c=0.05

spartina=450, algae=20
insects=1, detritus=250,
bacteria= 800, spiders=0.18
nematodes=0.5, mudcrabs=0.1
```

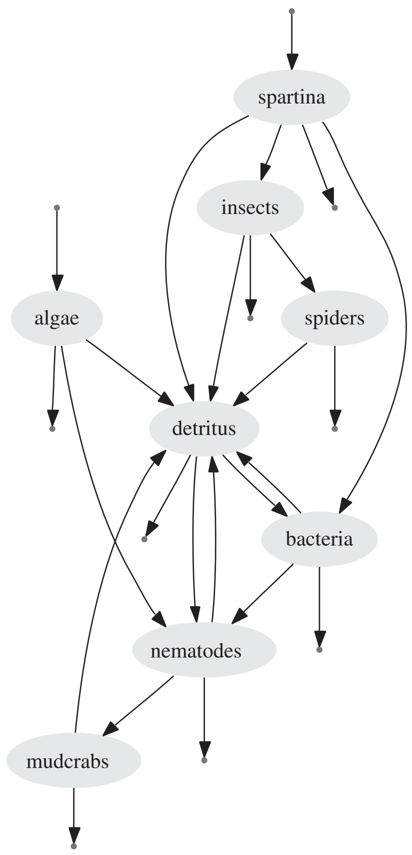


Figure 17: EcoNet model of the energy flow study in a Georgia salt marsh.

contributes to the detritus pool through fecal material (for the animal compartments) and dead tissue. Every compartment except detritus, the only nonliving compartment in this model, dissipates energy through respiration. One of the surprising findings was that this salt marsh exported substantial amounts of energy through detritus (0.6% of incoming radiation). Detritus export is represented by a loss term in the model.

We present the EcoNet model and the diagram in Fig. 17. Because energy, rather than carbon or nutrients is the currency for this model, dissipation rates are high. Energy dissipates rapidly because of respiration losses and detritus export. However, energy does cycle in this ecosystem in the form of detritus. A kilocalorie of energy can make multiple passages through all of the non-plant compartments before exiting the system.

5.2 Flow analysis

Table 2 shows the *B* and *N* matrices displayed by EcoNet. $n_{44} = 1.595 > 1$ represents the fact that a unit of energy in detritus is more likely to cycle through the salt marsh food web and reenter the detritus compartment before being dissipated from the system.

Bacteria, nematodes, and mud crabs are all part of the detritus-based salt marsh food web, and thus have the potential to cycle energy in this ecosystem (n_{55}, n_{77}, $n_{88} > 1$). In contrast, a unit

Table 2: *N* and *B* matrices of Georgia salt marsh model (Fig. 17).

	Spar-tina	Algae	Insects	Detritus	Bacteria	Spiders	Nematodes	Mud crabs
B								
Spartina	–1	0	0	0	0	0	0	0
Algae	0	–1	0	0	0	0	0	0
Insects	0.011	0	–1	0	0	0	0	0
Detritus	0.164	0.142	0.571	–1	0.549	0.667	0.745	0.667
Bacteria	0.005	0	0	0.4	–1	0	0	0
Spiders	0	0	0.143	0	0	–1	0	0
Nematodes	0	0.143	0	0.2	0.011	0	–1	0
Mud crabs	0	0	0	0	0	0	0.0062	–1
N								
Spartina	1	0	0	0	0	0	0	0
Algae	0	1	0	0	0	0	0	0
Insects	0.011	0	1	0	0	0	0	0
Detritus	0.278	0.399	1.063	1.595	0.889	1.063	1.195	1.063
Bacteria	0.117	0.159	0.425	0.638	1.356	0.425	0.478	0.425
Spiders	0.001	0	0.142	0	0	1	0	0
Nematodes	0.056	0.224	0.217	0.326	0.193	0.217	1.244	0.217
Mud crabs	0.0004	0.0014	0.0013	0.0020	0.0012	0.0013	0.0077	1.0013

of energy that enters the *Spartina,* algae, insects, or spiders compartment will never re-enter those compartments (n_{11}, n_{22}, n_{33}, n_{66},= 1), because all their energy is derived from primary production.

The *N* matrix traces the throughflow generated by inputs into various compartments in our model. The non-zero entries in the first row of *N* indicate that a unit of energy input into *Spartina* generates energy flowing through every compartment except algae. Detritus is the largest recipient of energy derived from *Spartina*, both from the direct movement of *Spartina* to detritus ($b_{41} = 0.164$) and from indirect pathways ($n_{41} = 0.278$). Therefore, $n_{41} - b_{41} = 11.4\%$ of this energy travels through the food web before entering the detritus compartment. There is no direct flow from *Spartina* to mud crabs ($b_{81} = 0$), and the shortest path between them is (Fig. 17):

$$\textit{Spartina} \rightarrow \text{Detritus} \rightarrow \text{Nematodes} \rightarrow \text{Mud crabs.}$$

Therefore, mud crabs receive only $n_{81} = 0.04\%$ of energy that is captured by *Spartina* photosynthesis.

Similarly, the second row of *N* indicates that the energy inputs to algae generate throughflows in four of the model compartments. Nearly 40% of this energy will pass through the detritus compartment, and 16% will reach bacteria through consumption of detritus ($n_{42} = 0.399$, $n_{52} = 0.159$). n_{12}, n_{23}, $n_{26} = 0$, therefore, *Spartina,* insects, and spiders cannot capture energy derived from photosynthesis by algae.

Although energy only enters the salt marsh through photosynthesis by *Spartina* and algae, the *B* and *N* matrices trace flows generated by hypothetical inputs into all compartments in the model. For example, of a unit input of energy in nematodes, nearly $b_{47} \approx 75\%$ goes directly to detritus and less than 1% is consumed by mud crabs ($b_{87} = 0.006$), with the remaining 24% lost to respiration.

From the mud crabs, the energy is either lost through respiration or recycled as detritus (see Fig. 17). The energy in detritus can reenter the detrital food web or be washed out to sea. Through the detrital food web, a small fraction of this energy ($n_{87} - b_{87} = 0.15\%$ of the original input) will make at least a second passage through the mud crab compartment.

5.3 Storage and utility analysis

The storage analysis matrix S (Table 3) traces the storage generated by a constant rate of energy inputs into different compartments in the model.

For example, a rate of 1 kcal/m^2 d energy input into *Spartina* will generate $s_{11} = 5.46$ kcal/m^2 stored energy in *Spartina.* Because of recycling in the detritus-based food web, even more energy from this input will be stored in the detritus compartment ($s_{41} = 5.56$ kcal/m^2). Smaller amounts of energy will be stored in all of the other compartments reachable from *Spartina.*

The utility analysis matrix U indicates interaction types and strengths among compartments in this network. Some of these relationships are straightforward. For example, insects feed on *Spartina*, so the addition of a unit of *Spartina* benefits the insects ($s_{31} = 0.025 > 0$), and the addition of a unit of insects is even more detrimental to the *Spartina* ($s_{13} = –0.090 < 0$). Indirect effects can be traced using the utility analysis matrix. For example, the addition of mud crabs has a negative effect on their prey, nematodes ($u_{78} = –0.009 < 0$), and in turn, benefits algae and bacterial compartments, on which nematodes feed ($u_{28} = 0.045 > 0$, $u_{58} = 0.002 > 0$) [15].

Table 3: Storage (S) and utility (U) analysis matrices of the Georgia salt marsh model (Fig. 17).

	Spartina	Algae	Insects	Detritus	Bacteria	Spiders	Nematodes	Mud crabs
S								
Spartina	5.46	0	0	0	0	0	0	0
Algae	0	7.14	0	0	0	0	0	0
Insects	0.031	0	2.86	0	0	0	0	0
Detritus	5.56	7.97	21.26	31.89	17.79	21.26	23.91	21.26
Bacteria	0.128	0.175	0.467	0.701	1.49	0.467	0.525	0.467
Spiders	0.010	0	0.952	0	0	6.67	0	0
Nematodes	0.071	0.279	0.270	0.405	0.239	0.270	1.55	0.270
Mud crabs	0.002	0.009	0.009	0.013	0.008	0.009	0.052	6.68
U								
Spartina	0.016	–0.005	–0.090	–0.003	0.219	0.032	0.038	–0.003
Algae	–0.023	0.187	0.084	–0.001	0.003	–0.037	–0.915	0.045
Insects	0.025	–0.012	0.548	–0.006	0.521	–0.243	0.084	–0.007
Detritus	0.008	0.001	–0.029	0.0004	–0.063	0.013	-0.005	0.0005
Bacteria	0.052	–0.017	–0.202	0.0006	0.785	0.087	-0.031	0.002
Spiders	0.019	–0.013	0.559	–0.006	0.556	0.751	0.086	–0.008
Nematodes	–0.004	0.160	0.012	–0.0002	0.063	–0.006	0.187	–0.009
Mud crabs	–0.002	0.042	0.008	–0.0001	0.028	–0.004	0.050	0.997

Because of multiple branches in the food web, some relationships are more complex, and may be different from what one may derive from the network diagram in Fig. 17. For example, even though insects produce detritus, the addition of a unit of insects is detrimental to the detritus compartment ($u_{43} = -0.029 < 0$), because these insects are consuming (and subsequently respiring) additional energy from *Spartina* that would otherwise go into detritus.

5.4 Further analysis

Network Environ Analysis is a comprehensive theory and includes more analysis results, such as output-based storage and throughflow analysis, amplification, homogenization and synergism. The MATLAB code [16] discussed in the previous chapter provides a thorough NEA analysis for interested users.

EcoNet was first online in June 2006. Although EcoNet had no documentation available online until January 2007, researchers from over 50 countries accessed and used EcoNet during 2006. Being a new software, it still lacks many features, and we have plans to integrate new capabilities.

Aside from adding more NEA results, we are working to integrate Finn's cycling index [17], exergy [18] and ascendancy [19] into EcoNet. We believe that these additions combined with the fast and efficient simulation interface will make EcoNet an invaluable tool for ecological modeling and analysis.

We encourage users to provide feedback, comments and suggestions at http://eco.engr.uga.edu/econet/contact.html.

Acknowledgements

I would like to thank the systems and engineering ecology group at the University of Georgia, in particular, Bernard C. Patten, Ernest W. Tollner, Stuart J. Whipple, David Gattie, Nadia Kellam and John Schramski for their help and guidance in the development of EcoNet. Gaston Small deserves much credit for his valuable contribution. Special thanks go to Shlomo Ta'asan for his guidance in the development of the numerical engine. Creating EcoNet would have been very difficult without the technical support of Florin Manolache. Finally, I would like to thank the editor, Sven Jørgensen, for including EcoNet among many other excellent software in this handbook.

References

[1] Gillespie, D.T., The chemical Langevin equation. *Journal of Chemical Physics*, **113**, pp. 297–306, 2000.

[2] Gillespie, D.T., Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, **81**, pp. 2340–2361, 1977.

[3] Clauset Jr., K.H., Rawley, C.C. & Bodeker, G.C., Stella – software for structural thinking. *Collegiate Microcomput.*, **5(4)**, pp. 311–319, 1987.

[4] Dowd, J.E. & Riggs, D.S., A comparison of estimates of Michaelis-Menten kinetic constants from various linear transformations. *Journal of Biological Chemistry*, **240**, pp. 863–869, 1965.

[5] Kincaid, D.R. & Cheney, E.W., *Numerical Analysis*, 3rd edn, Brooks Cole, 2001.

[6] Gardiner, C.W., *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences*, 2nd edn, Springer-Verlag, 1985.

[7] Gillespie, D.T., A rigorous derivation of the chemical master equation. *Physica A Statistical Mechanics and its Applications*, **188**, pp. 404–425, 1992.

[8] Ellson, J., Gansner, E., Koutsofios, E., North, S.C. & Woodhull, G., Graphviz and dynagraph – static and dynamic graph drawing tools. *Graph Drawing Software*, eds M. Junger & P. Mutzel, Springer-Verlag, pp. 127–148, 2003.

[9] Fath, B.D. & Patten, B.C., Review of the foundations of network environ analysis. *Ecosystems*, **2**, pp. 167–179, 1999.

[10] Patten, B.C., Systems approach to the concept of environment. *Ohio Academic Science*, **78(4)**, pp. 206–222, 1978.

[11] Gattie, D.K., Tollner, E.W. & Foutz, T.L., Network environ analysis: a mathematical basis for describing indirect effects in ecosystems. *Transactions of ASAE*, **48(4)**, pp. 1645–1652, 2005.

[12] Patten, B.C., *Holoecology: The Unification of Nature by Network Indirect Effects*, Kluwer (unpublished).

[13] Teal, J.M., Energy flow in the salt marsh ecosystem of Georgia. *Ecology*, **43**, pp. 614–624, 1962.

[14] Peterson, B.J. & Howarth, R.W., Sulfur, carbon, and nitrogen isotopes used to trace organic matter flow in the salt-marsh estuaries of Sapelo Island, Georgia. *Limnology and Oceanography*, **32**, pp. 1195–1213, 1987.

[15] Pace, M.X., Cole, J.J., Carpenter, S.R. & Kitchell, J.F., Trophic cascades revealed in diverse ecosystems. *Trends in Ecology and Evolution*, **14**, pp. 483–488, 1999.

[16] Fath, B.D. & Borrett, S.R., A Matlab function for Network Environ Analysis. *Environmental Modelling and Software*, **21**, pp. 375–405, 2006.

[17] Finn, J.T., Measures of ecosystem structure and function derived from analysis of flows. *Journal of Theoretical Biology*, **56**, pp. 363–380, 1976.

[18] Jørgensen, S.E. & Svirezhev, Y.M., *Towards a Thermodynamic Theory for Ecological Systems*, 1st edn, Elseiver, 2004.

[19] Ulanowicz, R.E., *Ecology*, *The Ascendent Perspective*, 1st edn, Columbia University Press, 1997.