

APPENDIX I

Lab for Parameter Estimation and Model Comparison

Designed in 2008 by Caroline Bampfylde for BIOL 590 (Models in Ecology).

Modified in 2008 by Gustavo Carrero for the Summer Workshop 2008.

Blau and Neely's Dursban Model

This lab has several 'already created' Matlab files designed as a tool for you to fit the different models studied by Blau and Neely (1975) to the data, and to compute several measures for model validation and selection. The models are summarized in Figure 1 (the same as the handout given in the lecture).

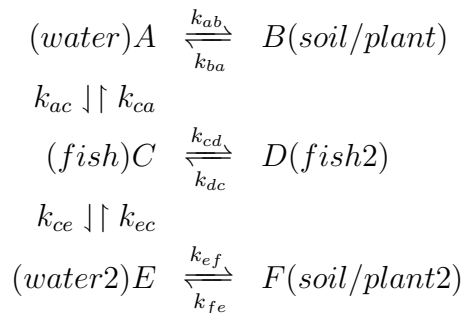
The Matlab files use a fast and efficient algorithm to numerically solve the differential equations. This so-called *Runge-Kutta* algorithm is implemented for a general system of six ordinary differential equations (`ode45`). By setting certain parameters to zero, you can obtain all the different models from Blau and Neely.

From the data and the solution of the equations, the residual sum of squares is computed (RSS). We use nonlinear least squares (NLLS), implemented in Matlab, to fit the data to each model and estimate the parameters.

Download the following three Matlab files from TBA now, and save them to your workspace:

- `blaufun.m` which defines the system of ODEs to solve,
- `blausoln.m` which solves the system of ODEs (to produce predicted values), and
- `blaunonlin.m` which computes the NLLS parameter estimates, along with the RSS for each model.

1. The following is a schematic diagram of all the reactions.



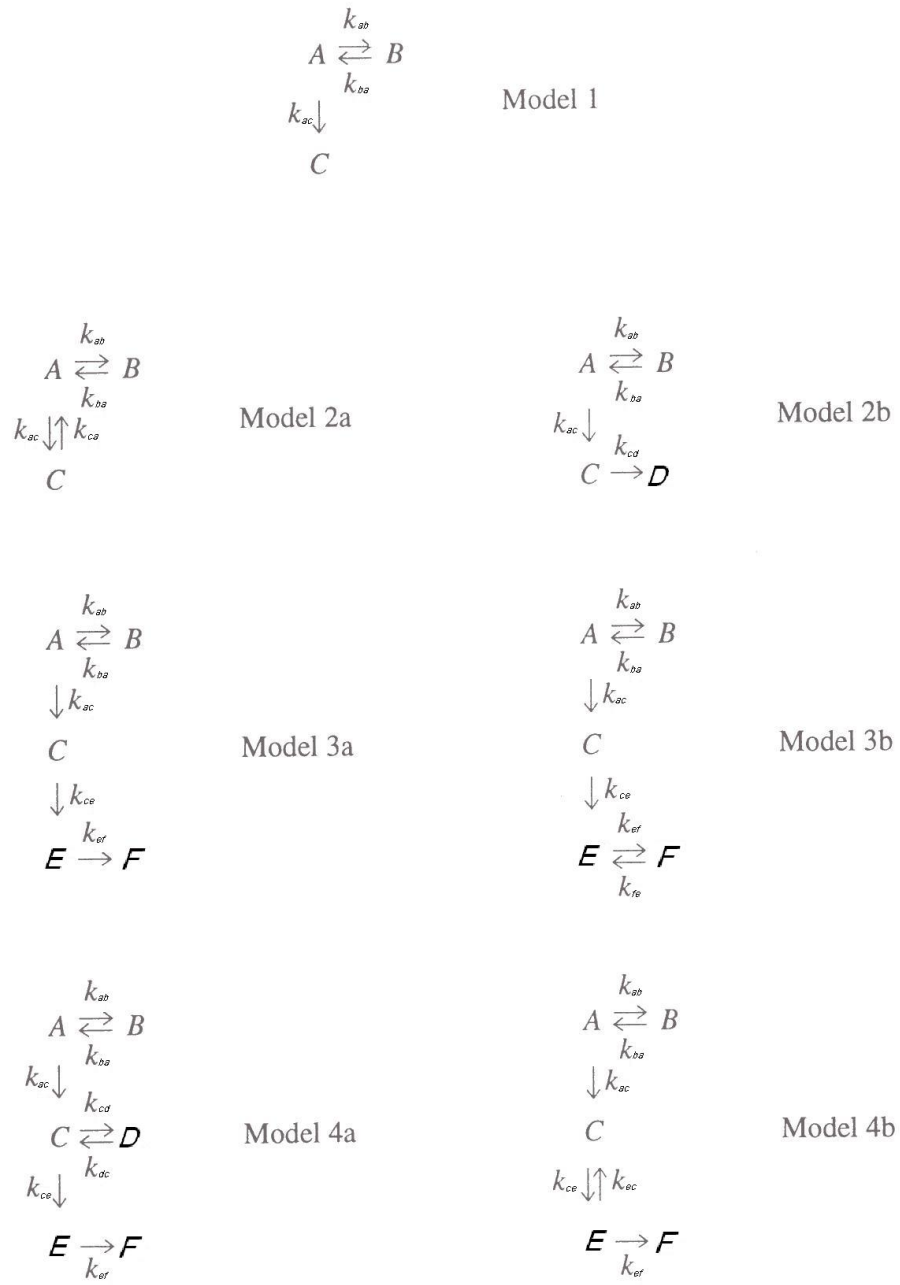


Figure 1: Summary of models used by Blau and Neely (1975) for the data on DURSBN in a simulated pond ecosystem.

Compartments A, E denote water, compartments B, F plants/soil and compartments C, D fish. All reactions are assumed to be first order processes, the rate at which X reacts to Y is denoted by k_{xy} . For each of the seven models in Blau and Neely (Figure 1), write down which of the parameters in the spreadsheet are allowed to vary, what names are given to the parameters in the paper, and which ones are to be set to zero. For example:

Model 1: $k_{ab} = k_1, k_{ba} = k_2, k_{ac} = k_3$ vary, all others are zero.

Note that $k_{i's}$ may not represent the same in different models (see, for example, k_7 in figure 1).

2. The diagram above is translated into differential equations. With the first three equations given as:

$$\frac{dx_a}{dt} = -k_{ab}x_a + k_{ba}x_b - k_{ac}x_a + k_{ca}x_c \quad (1)$$

$$\frac{dx_b}{dt} = k_{ab}x_a - k_{ba}x_b \quad (2)$$

$$\frac{dx_c}{dt} = k_{ac}x_a - k_{ca}x_c - k_{cd}x_c + k_{dc}x_d - k_{ce}x_c + k_{ec}x_e \quad (3)$$

3. Within Matlab, navigate to the folder where you saved the files (this is under current directory, click on the ... next to the drop-down menu). Now in the Matlab command window type `ls` and this will list all the files in the directory. You should see the 3 files you've saved (this is to verify Matlab is working in the correct place). You can open up the files to take a look at the code. Type `edit filename.m` at the Matlab command prompt to open a file for editing/viewing.
4. In the file `blaunonlin.m`, you will first see some lines in green with `%` at the start of each line. These are comments which Matlab does not compute but makes is easier to read the code. After this, comes the raw data from the DURSBN experiment (lines 12-23) which is then manipulated (lines 25-28). Following that is space for the 8 model parameters (lines 30-39) and these are placed in a vector `kk0` (line 41). We are going to start by fitting the data to model 1. Make sure that all the parameters for model 1 are set to 0.5 and all the other parameters are zero.
5. The code then uses a built-in Matlab function `nlinfit` to fit the model to the data, line 44. The output from this fitting gives us the estimate of the parameters (`kkfit`) and the residuals (`R`).
6. The final commands in the file (lines 50-83) plot the data and the model predictions. Most of the terminology should be self-explanatory. If you don't understand a command and want to know what it does, remember that you can type `help command` in the Matlab window. For example, try this for `help plot`.
7. The other files are `blausoln.m` and `blaufun.m`. The first file, `blausoln.m` solves the system of ODEs, given any set of parameters the file produces solutions to the ODEs at time

points t . The second file, `blaufun.m` takes the parameters, time values, and initial conditions and outputs solutions of the ODEs. Lines 6-22 specify the parameters. The parameters are specified by the vector kk . Each model has different combinations of parameters to fit. For Model 1, we are fitting k_{ab}, k_{ba}, k_{ac} so we include the line `kab=kk(1)` etc. and the rest of the parameters are set to zero. For different models you can comment or uncomment lines (using `%`) in order to fit different combination of parameters. Lines 24-29 specify the different components of the model. Lines 31-37 specify the system of differential equations.

8. Now let's fit the first model (Model 1). Start with $k_{ab} = k_{ba} = k_{ac} = 0.5$ and the rest equal to zero. Then in the Matlab command window type `blaunonlin`. This will run the nonlinear least squares routine. Some output on the screen will show you how many iterations the process is taking and how the SSE are decreasing. Once the fitting routine has found a solution it will terminate and give you the estimated values for the parameters ($kkfit$) and RSS . $kkfit$ is a vector and entries are given in the same order you specified in the `blaunonlin` file. RSS is also given and this is the sum of squared deviations at each data point.

The code also produces two figures. Figure 1 plots the observations (symbols) and predictions (lines) for the estimated parameters and provides you the SumSq on the graph itself. You can compare this to Figure 1 in the summary of the paper by Blau and Neely. If you would like to copy this figure (into a word document say), you can use Edit → Copy Figure and then Paste into Word. Figure 2 plots the residuals for each data point for each compartment. You can compare this to Figure 2 in the summary of the paper by Blau and Neely.

Your code should converge to a solution that is close to the one reported by Blau and Neely. (The parameter $k_2 = k_{ba}$ is off, though, probably a typo in the paper.)

9. Now we need to do this for all the seven models. For Model 2a, you will need to change the initial guess for k_{ca} in `blaunonlin` and also change lines 9-10 in `blaufun`. You should uncomment line 9 (remove `%` at start) and comment out line 10 (add `%` at start of line). Once you have made your changes, you need to save the files before running them again in the Matlab window.

Now you can run `blaunonlin` again. This will now fit model 2a to the data. You can keep a record of the residual sum of squares for each model and the estimated parameters.

Continue for all the models. Keep a record of the parameters and the SumSq. Don't forget each time you start fitting a new model, you need to make changes relating to the parameters in **both** the files `blaunonlin` and `blaufun`.

10. Create a table with the following columns and fill in each column: Model Number (j), RSS , $\hat{\sigma}^2$, L_j , $\ln(L_j)$, $\ln(L_j/L_{\max})$ (for models 1, 2b, 3a, and 4a only), χ^2 (for models 1, 2b, 3a, and 4a only), P (for models 1, 2b, 3a, and 4a only), AIC, AICc, and Δ_i (difference between a model's AIC and the smallest AIC in the set of models). Compare with Tables 2 and 3 of the summary of the paper by Blau and Neely. Your values may be slightly different, based on the RSS calculations.

$$\begin{aligned}
\hat{\sigma}^2 &= RSS/n \\
n &= \text{sample size} \\
\ln(L_j) &= -(n/2) \ln(\hat{\sigma}^2) \\
L_j &= \exp(\ln(L_j)) \\
\ln(L_j/L_{\max}) &= \ln(L_j) - \ln(L_{\max}) \\
\chi^2 &= -2(\ln(L_j) - \ln(L_{\max})) \\
AIC &= -2 \ln(L_j) + 2K \\
K &= \text{\#parameters} \\
AIC_c &= AIC + 2K(K+1)/(n-K-1)
\end{aligned}$$

The sample size for the Dursban data is $n = 36$ since there are 3 compartments sampled over 12 sample dates.

The log likelihood calculation is slightly different than the one given in the textbook. However, the difference is simply a constant, which doesn't matter in the subsequent calculations. $\ln(L_j/L_{\max})$ is calculated using the rules for logs. You will need to use `log` in Matlab to calculate the natural logarithm. To calculate the exponential, use `exp`. You can use `max` in Matlab to calculate the maximum entry in a vector.

P is the probability that a χ^2 value as large or larger would occur by random sampling (i.e., the null hypothesis is that the likelihood of model j is equal to the likelihood of the model with the highest likelihood (L_{\max}). P can be found using `chi2cdf` in Matlab. $P=1-\text{chi2cdf}(\chi^2, \text{df})$.

The correction of AIC for small sample sizes is AICc and should be used when the number of data points is small (say $n < 40$). Don't forget when calculating K (number of parameters), that along with estimating all the k_{xy} parameters, you are also estimating $\hat{\sigma}^2$ for each model.

Now you can complete the following questions:

1. For each of the seven models in Blau and Neely (Figure 1), write down which of the parameters in the spreadsheet are allowed to vary (what are they called in the paper?) and which ones are to be set to zero.
2. The differential equations for the first three compartments are given in the lab ($\frac{dx_a}{dt}$, $\frac{dx_b}{dt}$, $\frac{dx_c}{dt}$), write down the other three equations for x_d , x_e , x_f .
3. Create a table with the parameter estimates for each model.
4. Plot the solutions, data and residuals (as a function of time) for model 4a.
5. Write a short paragraph tying in your results to a conclusion regarding which model(s) best describe the processes in the Dursban experiment.