Working with dynamic models for agriculture - A short course
September 2nd to 4th 2010
Supagro, Montpellier, France

Organized by: INRA (France), University of Florida (United States), Montpellier Supagro (France)

Teachers: Daniel Wallach (INRA), David Makowski (INRA), James W. Jones (U.of Florida), Sébastien Roux (INRA), François Brun (ACTA)

Contact for information and registration: bastie@supagro.inra.fr

GENERAL INFORMATION:
Dynamic system models are now widely used in agronomic research and extension. These models consist of differential or difference equations that represent the dynamics of the different components of the system (soil, plant, pathogens, etc). One then solves the equations on a computer to simulate the dynamics of the real system. Such models can thus be used to explore the effects that changes in the environment or in management would have on the system that is modelled. They are used in impact assessment, in evaluating innovative management practices, as decision aids, as diagnostic tools or to aid in planning experiments.

Developing such models and interpreting the results requires detailed domain knowledge concerning the behaviour of the real system. However, this domain knowledge is in general not sufficient. There are also mathematical, statistical and computer considerations which are equally important, but are seldom considered by agricultural modelers. This has led to difficulties in working with such models: modellers are usually researchers in the area being modelled, and they often lack expertise in the methods for working with models. The purpose of this course is to provide agronomic researchers the information and tools that are necessary to work effectively with dynamic models.

Current PhD students and established agronomic researchers who are interested in applications of agricultural models in their research programs should apply. The course will include lectures, demonstrations of models and software for working with the models as well as hands-on practical exercises. Participants will learn how to estimate parameters for dynamic agricultural models, how to evaluate them, and how to conduct sensitivity and uncertainty analyses using modern techniques. Substantial time will be devoted to exercises. At the end of the course, the students should be able to apply the methods on their own. The material will be based on the textbook, “Working with Dynamic Crop Models: Evaluation, Analysis, Parameterization and Applications” edited by D. Wallach, D. Makowski, and J. W. Jones.

Fee: 200€, which includes the book “Working with Dynamic Crop Models: Evaluation, Analysis, Parameterization and Applications”.

- 1 -
PROGRAM (Each topic includes lecture and computer exercises)

Day 1: Thursday, September 2nd 2010
Morning (8:30 to 12:30)
- Course introduction (30 min)
  o Objectives, presentation of lecturers and students
- Introduction to systems and modelling. Examples (1 h 30)
  o What is the systems approach, what is a systems model, how are models used, examples of system models
- The R programming language and basic statistical principles (1 h 30)
  o A first course in this high level, free language. It will be used for the examples and for the exercises
  o Review of basic statistical principles and applications with R
Lunch (12:30 to 14:00)
Afternoon (14:00 to 18:00)
- The R programming language and basic statistical principles - continuation (1 h)
- Computer simulation (2 h 30)
  o Finite difference or differential equations, numerical considerations, programming considerations

Day 2: Friday, September 3rd 2010
Morning (8:30 to 12:30)
- Computer simulation - continuation (30 min)
- Uncertainty and sensitivity analysis (3 h)
  o What it is, why it is useful, sources of uncertainty, Monte Carlo simulations.
  o Local sensitivity analysis, global sensitivity analysis. Application to dynamic system models. Examples and exercises
Lunch (12:30 to 14:00)
Afternoon (14:00 to 18:00)
- Uncertainty and sensitivity analysis - continuation (1 h 30)
- Parameter estimation (calibration) (2 h)
  o The principles underlying parameter estimation, least squares, maximum likelihood, Bayesian methods. The specific problems in applying these methods to dynamic system models.

Day 3: Saturday, September 4th 2010
Morning (8:30 to 12:30)
- Parameter estimation (calibration) – continuation (2 h 30)
- Model evaluation (1 h)
  o Criteria of model quality, with emphasis on predictive accuracy and the quality of decisions.
  o How to estimate the criteria. Examples and exercises
Lunch (12:30 to 14:00)
Afternoon (14:00 to 18:00)
- Model evaluation – continuation (2h30)
  - Final discussion (1h)
List of participants

<table>
<thead>
<tr>
<th>Name</th>
<th>Email Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANDRIANARISOA Sitraka</td>
<td><a href="mailto:andriana@supagro.inra.fr">andriana@supagro.inra.fr</a></td>
</tr>
<tr>
<td>ARCHONTOULIS Sotiris</td>
<td><a href="mailto:sarchont@gmail.com">sarchont@gmail.com</a></td>
</tr>
<tr>
<td>BRUNEL-MUGUET Sophie</td>
<td><a href="mailto:sbrunel@bordeaux.inra.fr">sbrunel@bordeaux.inra.fr</a></td>
</tr>
<tr>
<td>CAVALIERI Andrea</td>
<td><a href="mailto:a.cavalieri@unitus.it">a.cavalieri@unitus.it</a></td>
</tr>
<tr>
<td>FAUX Anne-Michelle</td>
<td><a href="mailto:Anne-Michelle.Faux@uclouvain.be">Anne-Michelle.Faux@uclouvain.be</a></td>
</tr>
<tr>
<td>GABRIEL Jose Luis</td>
<td><a href="mailto:joseluis.gabriel@upm.es">joseluis.gabriel@upm.es</a></td>
</tr>
<tr>
<td>IQBAL Akhtar</td>
<td><a href="mailto:akhtar.iqbal@reims.inra.fr">akhtar.iqbal@reims.inra.fr</a></td>
</tr>
<tr>
<td>KOUAKOU Koffi Patrice</td>
<td><a href="mailto:kouakoukp@gmail.com">kouakoukp@gmail.com</a></td>
</tr>
<tr>
<td>LISAZO Jon</td>
<td><a href="mailto:jon.lizaso@upm.es">jon.lizaso@upm.es</a></td>
</tr>
<tr>
<td>MEYLAN Louise</td>
<td><a href="mailto:louise.meylan@gmail.com">louise.meylan@gmail.com</a></td>
</tr>
<tr>
<td>METAY Aurélie</td>
<td><a href="mailto:metay@supagro.inra.fr">metay@supagro.inra.fr</a></td>
</tr>
<tr>
<td>MOUEN BEDIMO Joseph Aubert</td>
<td><a href="mailto:josephemouen@yahoo.fr">josephemouen@yahoo.fr</a></td>
</tr>
<tr>
<td>MOUSSADEK Rachid</td>
<td><a href="mailto:moussa.inra@gmail.com">moussa.inra@gmail.com</a></td>
</tr>
<tr>
<td>MUNZ Sebastian</td>
<td><a href="mailto:s.munz@uni-hohenheim.de">s.munz@uni-hohenheim.de</a></td>
</tr>
<tr>
<td>PERRIN Aurélie</td>
<td><a href="mailto:perrina@supagro.inra.fr">perrina@supagro.inra.fr</a></td>
</tr>
<tr>
<td>PISTOCCHI Chiara</td>
<td><a href="mailto:c.pistocchi@sssup.it">c.pistocchi@sssup.it</a></td>
</tr>
<tr>
<td>POLGE De COMBRET Lucie</td>
<td><a href="mailto:Lucie.DECOMBRET@advini.com">Lucie.DECOMBRET@advini.com</a></td>
</tr>
<tr>
<td>RUSINAMHODZI Leonard</td>
<td><a href="mailto:l.rusinamhodzi@cgiar.org">l.rusinamhodzi@cgiar.org</a></td>
</tr>
<tr>
<td>SENTHILKUMAR Kalimuthu</td>
<td><a href="mailto:skalimuth@bordeaux.inra.fr">skalimuth@bordeaux.inra.fr</a></td>
</tr>
<tr>
<td>SISSOKO Fagaye</td>
<td><a href="mailto:fagaye_sissoko@yahoo.fr">fagaye_sissoko@yahoo.fr</a></td>
</tr>
<tr>
<td>SWAIN Dillip</td>
<td><a href="mailto:swain@agfe.iitkgp.ernet.in">swain@agfe.iitkgp.ernet.in</a></td>
</tr>
<tr>
<td>TARSITANO Davide</td>
<td><a href="mailto:davide.tarsitano@sac.ac.uk">davide.tarsitano@sac.ac.uk</a></td>
</tr>
<tr>
<td>TOSTI Giacomo</td>
<td><a href="mailto:giacomo.tosti@gmail.com">giacomo.tosti@gmail.com</a></td>
</tr>
<tr>
<td>TRAORE Boubasiditraore</td>
<td><a href="mailto:boubasiditraore@yahoo.fr">boubasiditraore@yahoo.fr</a></td>
</tr>
<tr>
<td>UHRMANN Franz</td>
<td><a href="mailto:franz.uhrmann@iis.fraunhofer.de">franz.uhrmann@iis.fraunhofer.de</a></td>
</tr>
<tr>
<td>Table of Content</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>Exercises on Simulation</td>
<td>P 5</td>
</tr>
<tr>
<td>Exercises on Uncertainty and Sensitivity analysis</td>
<td>P 15</td>
</tr>
<tr>
<td>Exercises on Parameter Estimation</td>
<td>P 21</td>
</tr>
<tr>
<td>Exercises on Model Evaluation</td>
<td>P 37</td>
</tr>
<tr>
<td>Programming in R - Summary of useful instruction</td>
<td>P 43</td>
</tr>
</tbody>
</table>
Purpose. By completing this set of exercises, you will learn how to:

1. Simulate a dynamic crop model using the R programming language
2. Make modifications to the model for improving the ability of the model to predict basic responses to weather
3. Make modifications to the model to include an additional process that affects an existing state variable
4. Make modifications to the model to include a new state variable
5. Create graphs and tables of simulated results with R so that you can present the results to others or analyze the outputs using your own software, such as EXCEL.

It is assumed that you have been exposed to the basic operation of the R programming language and have the Wallach et al. (2006) text book. A lecture will explain the methods used and summarize the exercises.

The Model. The model is given in Chapter 1 of the text book, on page 4. It is a dynamic model of crop growth in which the crop is represented by two state variables, leaf area index (LAI) and total biomass (B). In addition to these two crop variables, a third state variable is used to simulate the cumulative thermal time since plant emergence (TT). This model does not take into account any effects of soil water, nutrients, pests, or diseases, and as such, it is considered a potential production model. In this set of exercises, we represent the state variables as functions of days after emergence, in other words as TT(day), B(day), and LAI(day). In the text book, we used j for time, and in our exercise, day is the same as j on page 4 in the text.

There are five basic operations that computer code must do when simulating dynamic models:

1. Initialize all state variables, parameters, and other variables needed to perform the calculations.
2. Compute rates of change of the state variables, or the change in state variables that occur in any specific time step. On page 4, the changes to the 3 state variables in this model are computed in equations (2), (3), and (4).
3. Update all of the state variables to those of a new time (day in our case) by adding the rate of change multiplied by the time step to the value of the state variable from the previous time step. This is called Euler integration, and the equations are also shown on page 4 (as TT(j+1) = TT(j) + ΔTT(j)), and similar equations for the other 2 state variables. Since we are using “day” instead of “j” in our exercises, the updating of TT is written as TT(day+1) = TT(day) + ΔTT(day).
4. Loop through #2 above through #3 from the starting time of the simulation until the last day to be simulated, incrementing time (day) as this is done. The final results will be values of state variables for each day of the simulation.
5. Save results of the simulations in a file or in graphs for further analysis.

The R program that performs these steps can be written in various ways, but it is helpful to create a function program component for the model itself since one usually wants to use the model to
simulate many situations, such as weather conditions or planting dates, for example. The R code for this set of exercises (MaizeModelESA_1.r) is written in this way; the basic model itself is written as an R function. Programming the model in a function also helps since there will be various ways we will want to work with a model. The overall structure of the R program for running the dynamic maize model is shown in Table 1 below.

| R code for Simulating the Maize Model And For Use in the Simulation Exercises (SimulationEx1.r) |
| Comments |
| Maize model function (maize.model)  
The model obtains parameters, weather data, and first and last simulation day using  
The arguments of the function (param, weather, sdate, and ldate)  
At the end of the function, the model calculations are returned to the main program as vectors: Day, TT, LAI, and B |
| Function to read weather data (read.weather)  
All weather data (day, I, Tmax, and Tmin) are read from the file blagnacSimul.dat, which Has headers that define these variable names  
The function returns weather for the year that one is working with |
| A Main Program that controls the simulations  
This main program does the following:  
• Defines and sets the values of all of the model parameters  
• Sets the simulation starting (sdate) and last (ldate) days  
• Runs the maize model, storing results in the output table  
• Writes the output table in a file in the working directory (output.csv)  
• Displays graphs of output variables (TT, LAI, and B)  
• Displays graphs of input variables (Tmin, Tmax, and I) |

Below is a listing of the function that you will use to begin your work with this set of exercises.
Notice that this maize model function initializes variables, creates the loop to simulate from the start day (sdate) to the last day (ldate), computes rates of changes of state variables, and updates the state variables. But, it does not initialize ALL variables and parameters. The function has as arguments the seven parameters (Tbase, RUE, K, alpha, LAImax, TTM, TTL) of this model as well as all of the weather data in a table (weather) and the first and last date of the simulation.
(sdate, ldate). Those variables are initialized in a main program that controls this function. Also, note that there is no statement for outputs of the state variables. Instead, the state variables are returned to the main program as a table (data.frame) where they can be plotted or saved to a file.

Another point to make is that even though this is a simple model with only 3 state variables (only 2 for the crop), it is organized the same as more complex models that may have 10 or 100 or more state variables. Furthermore, it may be advantageous to create more than one function for models that have many state variables, but even if this is done, the same five steps are used to solve the model. For example, if this model included a soil water model in addition to the crop model, one could create two functions, one for the crop and one for the soil water, for this more complex case. In that case, it would be necessary to pass soil water state variables into the function for the crop and vice versa. We will not ask you to do this in this course, since our main goal is to introduce you to methods for working with dynamic models, and the methods you will use are also useful for more complex models.

There are two other components of the program that are summarized here. First, there is the weather function. This function reads in daily weather data for Blagnac, France. In this file, there are 999 years of weather data that have been generated based on statistical properties of historical records at Blagnac. This simulation program only uses one year of weather data (the first year). The table below shows the first 16 lines of this file, which shows the header labels for day, month, year, and solar radiation (I), maximum temperature (Tmax) and minimum temperature (Tmin) for each day. The file has values separated by tabulation. The first line of data show that this is day 1, month 1, year 1, 3.09 MJ/day solar radiation and 10.6 and 2.2 °C maximum and minimum temperatures, respectively. The function used to read this data file is given in Table 3 below.

<table>
<thead>
<tr>
<th>day</th>
<th>month</th>
<th>year</th>
<th>I</th>
<th>Tmax</th>
<th>Tmin</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3.09</td>
<td>10.6</td>
<td>2.2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1.64</td>
<td>12.2</td>
<td>-3.5</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>6.77</td>
<td>2.6</td>
<td>0.1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>7.74</td>
<td>2.6</td>
<td>2.6</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>3.78</td>
<td>10.6</td>
<td>3.3</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1.63</td>
<td>12.6</td>
<td>2.7</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>2.56</td>
<td>11.0</td>
<td>3.4</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>2.53</td>
<td>10.9</td>
<td>-0.3</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>4.46</td>
<td>11.4</td>
<td>4.6</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>3.40</td>
<td>12.2</td>
<td>5.4</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1.64</td>
<td>12.2</td>
<td>4.4</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>3.01</td>
<td>13.2</td>
<td>6.9</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>1</td>
<td>2.12</td>
<td>8.6</td>
<td>3.2</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
<td>6.53</td>
<td>4.3</td>
<td>4.0</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1</td>
<td>8.47</td>
<td>11.8</td>
<td>6.2</td>
</tr>
</tbody>
</table>

Table 3. Partial listing of the weather data file named blagnacSimul.dat.

```r
# Reading Weather data function
read.weather <- function(working.year=NA)
{
    #day month year R Tmax Tmin rain ETP
    # R : solar radiation (MJ)
    # Tmax : maximum temperature (°C)
    # Tmin : minimum temperature (°C)
    # data are from a weather simulator for Blagnac (Toulouse, France)
    weather<-read.table("blagnacSimul.dat",header=T, sep="\t")
    # if argument working.year is specified, work on one particular year
    if (!is.na(working.year)) {weather<-subset(weather, year==working.year )}
    return (weather)
}
```

Table 4. Function used in R maize model to read in the daily weather data for Blagnac, France.
A main program is required to make the model run. The main program is listed last in the R file as a reminder that all functions need to be defined before the main program can use them.

```R
### MAIN PROGRAM ###
# Session 1: working with the model

# TODO : change path to your working directory (with '/')
path<="C:/Courses/courseESA/Rscript"
setwd(path)

# TODO : change the year of the weather you want (1 to 999)
weather<read.weather(1)

# Define parameter values of the model function
# This is part of the initialization process, but it is done here in the main
# program for reasons that we will see later on
# Tbase : the baseline temperature for growth (°C)
Tbase <- 7.0
# RUE : radiation use efficiency (g.MJ⁻¹)
RUE <- 1.85
# K : extinction coefficient (-)
K <- 0.7
# alpha : the relative rate of leaf area index increase for small values of leaf area
# index ((°C.day⁻¹))
alpha <- 0.00243
# LAImax : maximum leaf area index (m² leaf/m² soil)
LAImax <- 7.0
# TTM : temperature sum for crop maturity (°C.day)
TTM <- 1200
# TTL : temperature sum at the end of leaf area increase (°C.day)
TTL <- 700
# sdate : sowing date
sdate <- 100
# ldate : last date
ldate <- 250

# Run the model by calling the function maize.model with its state variables saved
# in the variable named output
output<-maize.model(Tbase,RUE,K,alpha,LAImax,TTM,TTL,weather,sdate,ldate)

# Write output to a file (to open with notepad or excel)
options(digits=3)
write.table(format(output), file = "output.csv", quote = FALSE, sep = "\t", dec = ".",
row.names = FALSE)

# Produce graphical outputs of the state variables
dev.new()
par(mfrow=c(3,1))
plot(output$day,output$TT, xlab = "day", ylab = "Temperature sum",type="l")
plot(output$day,output$B, xlab = "day", ylab = "Biomass",type="l")
plot(output$day,output$LAI, xlab = "day", ylab = "LAI",type="l")

# Produce graphical outputs of the input weather variables
dev.new()
par(mfrow=c(3,1))
plot(1:365,weather$Tmin, xlab = "day", ylab = "Temperature min")
plot(1:365,weather$Tmax, xlab = "day", ylab = "Temperature max")
```

Table 5. Listing of the main program for running the simple, potential production maize model and for saving its results in the output file that can be read by a text editor or EXCEL and for producing graphs of the simulated variables.
We have presented these sections of the code in separate tables, but all of the code is in one R file (MaizeModelESA_1.r) that is opened in R and then run using the “Run All” command. See the complete listing of the program in the Appendix.

**Example Model Output**

Table 5 shows a partial listing of the output.csv file after formatting the file into columns using EXCEL. The output starts on day 100, which was defined as the starting date, or in this case the day that the crop emerged in a field. This model does not predict germination or emergence. Note that on day 100, TT is equal to 0, LAI and B are 0.01 and 1.0, respectively. These are the values that we entered as initial conditions in the program. Following day 100, thermal time (TT) accumulates according to Tmax and Tmin of the year being simulated, and LAI and B are increasing in response to the temperature, solar radiation, and parameters that define the sensitivity of the crop to those variables.

Note that at the bottom of this table, the LAI and B remain constant for several days, at values of 6.98 and 2913.7, respectively. The crop stopped growing because the thermal time (TT) reached 1200, which was the value of the parameter given to define the thermal time at crop maturity.

Graphs of daily temperatures (Tmax and Tmin) and of solar radiation (I) are also shown in Figure 1 below.

<table>
<thead>
<tr>
<th>day</th>
<th>TT</th>
<th>LAI</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0</td>
<td>0.01</td>
<td>1</td>
</tr>
<tr>
<td>101</td>
<td>4.15</td>
<td>0.011</td>
<td>1.288</td>
</tr>
<tr>
<td>102</td>
<td>10.95</td>
<td>0.012</td>
<td>1.613</td>
</tr>
<tr>
<td>103</td>
<td>14.75</td>
<td>0.013</td>
<td>1.781</td>
</tr>
<tr>
<td>104</td>
<td>21.2</td>
<td>0.014</td>
<td>1.954</td>
</tr>
<tr>
<td>105</td>
<td>25.25</td>
<td>0.015</td>
<td>2.303</td>
</tr>
<tr>
<td>106</td>
<td>31.55</td>
<td>0.017</td>
<td>2.801</td>
</tr>
<tr>
<td>107</td>
<td>37.5</td>
<td>0.018</td>
<td>3.161</td>
</tr>
<tr>
<td>108</td>
<td>43.05</td>
<td>0.020</td>
<td>3.587</td>
</tr>
<tr>
<td>109</td>
<td>48.65</td>
<td>0.022</td>
<td>3.971</td>
</tr>
<tr>
<td>110</td>
<td>51.95</td>
<td>0.023</td>
<td>4.148</td>
</tr>
<tr>
<td>111</td>
<td>57.85</td>
<td>0.026</td>
<td>4.835</td>
</tr>
<tr>
<td>112</td>
<td>63.15</td>
<td>0.028</td>
<td>5.309</td>
</tr>
<tr>
<td>113</td>
<td>68.2</td>
<td>0.030</td>
<td>5.885</td>
</tr>
<tr>
<td>114</td>
<td>70.15</td>
<td>0.031</td>
<td>6.803</td>
</tr>
<tr>
<td>115</td>
<td>70.15</td>
<td>0.031</td>
<td>7.309</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Skip</td>
<td>Lines</td>
</tr>
<tr>
<td>215</td>
<td>1144.3</td>
<td>6.977</td>
<td>2741.318</td>
</tr>
<tr>
<td>216</td>
<td>1157.05</td>
<td>6.977</td>
<td>2779.324</td>
</tr>
<tr>
<td>217</td>
<td>1170.95</td>
<td>6.977</td>
<td>2825.995</td>
</tr>
<tr>
<td>218</td>
<td>1188.3</td>
<td>6.977</td>
<td>2857.519</td>
</tr>
<tr>
<td>219</td>
<td>1203.6</td>
<td>6.977</td>
<td>2913.664</td>
</tr>
<tr>
<td>220</td>
<td>1217.45</td>
<td>6.977</td>
<td>2913.664</td>
</tr>
<tr>
<td>221</td>
<td>1229.8</td>
<td>6.977</td>
<td>2913.664</td>
</tr>
</tbody>
</table>
Exercises

1. Run the model on your own computer to see if you get the same results as shown in the outputs above. Note that you will need to set the path to your working directory where you have your own maize model file and weather file. This is done in the main program where it is identified by the TODO statement. After running the model, confirm that you get the same graphical output as closely as possible. Then, check the output file. For example, you should get values of TT, LAI, and B of 70.15, 0.031, and 7.309, respectively, on day 115. On the last day (day=250), you should have values of TT=1648.3, LAI=6.977, and B=2913.664.

2. Chose year 2 to simulate. This is done in the main program where there is another TODO statement. You should replace the (1) with a (2) in the statement:
   
   weather<read.weather(1).
   
   Run this case and confirm that your answers are different by looking at the weather graphs and by looking at the values from the new output file after

---

**Figure 1.** Graphical presentation of the weather input variables Tmin, Tmax, and I on the left and output state variables TT, B, and LAI on the right side of this figure and from top to bottom, respectively.
you run the model again. To run the model again after replacing the 1 with a 2, you select RUN ALL in the pull down “Edit” menu in R. Examine your results. Does the crop require more days to reach maturity for year 2? Can you explain why? Is the LAI higher or lower, and is biomass higher for year 2 during all of the growing season or part of the season?

3. You are asked to change the graphs of state variables for this run. Plot TT vs. day as before, but now plot LAI vs. TT and B vs. LAI. Run both years and compare the graphs.

4. Start again with your original file, that is the program using year 1 and with the graphs and output file as they were in the file you started with. Now, you are asked to compute a new variable in the model – Cumulative intercepted radiation. Researchers sometimes measure intercepted radiation during an entire season to produce cumulative radiation over time that can be plotted vs. time. This variable is also used to estimate RUE by plotting B(day) vs. CumInt(day). The light intercepted on each day (Int(day)) is computed by the following equation:

\[ \text{Int(day)} = \text{I(day)} \times (1 - \exp(-K \times \text{LAI(day)})) \]

Then,

\[ \text{CumInt(day+1)} = \text{CumInt(day)} + \text{Int(day)} \]

This is a new variable that must have its initial value CumInt(sdate) = 0.0, and this is specified in the initial conditions section of the maize model. You are asked to modify the model function to compute CumInt(day), to pass its results out along with TT, LAI, and B, then, you are asked to add this variable to the output file, and to graph CumInt vs. day and to plot B vs. CumInt. The relationship between B and CumInt should be linear and the slope of this line is an estimate of the radiation use efficiency (RUE). Can you confirm that the slope of the line of B vs. CumInt is indeed equal to RUE?

5. In the formulation of the model on page 4 in Wallach et al. (2006) and in the implementation in the R program, temperature affects development of the plant through the relationship between thermal time when vegetative growth stops and when maturity is reached. Temperature does not affect growth of biomass. Most crop models include relationships between temperature and photosynthesis rates or biomass growth rates. For example, the widely used DSSAT CERES-Maize model has a relationship in the model that reduces RUE when temperatures are not between 16.5 and 33 °C. A piece-wise linear relationship is used to reduce RUE between 6.2 and 16.5 °C, and to also reduce it linearly between 33 and 44 °C. You are asked to modify the maize model to include this temperature effect on RUE. This can be done by specifying that RUEmax is 1.85, which is the value of RUE used in the original model. Then, you should program in the relationship to compute RUE depending on average daily temperature. RUE would be 0.0
if temperature is below 6.2°C, linearly increasing to 1.85 at 16.5 °C, keeping RUE at 1.85 if temperature is between 16.5 and 33 °C, then linearly reducing it on any day when average temperature is between 33 and 44 °C, making RUE equal to 0.0 when temperature is higher than 44 °C. Run the model again with this modification and compare results with what you had earlier. Did values of LAI and B change relative to what they were before? Run the new model version using year 2 of weather data. Compare results with the original year 2 results. Discuss your results. Note that this does not add a new state variable. It only changes the formulation of RUE. First, it was assumed to be a constant and not dependent on temperature. Now, it is assumed to vary with temperature.

6. As a final model modification, you are asked to add a new state variable to the original model – reproductive organ biomass. In the maize crop, vegetative growth essentially stops when the last leaf is grown and when reproductive organs (ears) are formed. For this model version, let’s assume that ear growth starts after the last leaf area increase occurs, which is at thermal time = 700 °C-days in this model. Furthermore, assume that all new growth for the rest of the season goes into reproductive ear growth. You are to add a new state variable (BE), or the biomass of the ear. After increasing the complexity of the model by adding a new state variable, you are asked to again simulate years 1 and 2 to compare results. You are also asked to change TTL to 750 and simulate the crop growth again, then increase TTM to 1300. What can you say about the interactions of total biomass (B), reproductive biomass (BE) relative to the 2 years of weather data and relative to changes in TTL and TTM? Think about what kind of data you would need to develop and evaluate this modification to the model for cultivars you are growing.

7. As a final discussion question on this model, how would you modify the model to incorporate leaf senescence? What would be needed experimentally to model this change to the model?
8. You are asked to modify the R code for this crop model to simulate a predator–prey dynamic model as revised by Holling (1959) and Leslie and Gower (1960). The model is:

\[
\begin{align*}
\frac{dN}{dt} &= rN(1 - \frac{N}{K} - \frac{uP}{k + N}) \\
\frac{dP}{dt} &= sP(1 - \frac{P}{jN})
\end{align*}
\]

Where:

- \( t \) = time, in days
- \( N \) = homogenous population (density) of prey (number per ha)
- \( P \) = homogenous population (density) of predators (number per ha)
- \( r \) = relative rate of prey population growth
- \( K \) = environment carrying capacity for prey (number per ha)
- \( u \) = maximum predation rate (number per predator per day)
- \( k \) = prey number when predation is half its maximum (number per ha)
- \( s \) = predator growth rate constant
- \( j \) = parameter relating maximum predator density to prey density

Using this modified Lotka-Volterra model, implement a simulation of predator and prey dynamics using a time step (\( \Delta t \)) of 0.1 day. Initial population densities are \( N = 25 \) and \( P = \)

The following are values of parameters:

- \( r = 0.4 \)
- \( K = 900 \)
- \( u = 25.0 \)
- \( k = 200 \)
- \( s = 0.05 \)
- \( j = 0.5 \)

a. Plot the densities of \( N \) and \( P \) vs. time, allowing the system to reach steady state
b. Plot \( N \) vs. \( P \)
c. Repeat the exercise using:

- \( r = 0.3 \)
- \( K = 1000 \)
- \( u = 1.8 \)
- \( k = 200 \)
- \( s = 0.05 \)
- \( j = 0.45 \)
Exercises on Uncertainty and Sensitivity analysis

Exercise 1 aims at presenting several basic R functions for computing probability density and for random sampling. In Exercise 2, students are asked to perform an uncertainty analysis with a simple multiplicative model including two uncertain inputs. In Exercises 3 and 4, students are asked to perform uncertainty and sensitivity analysis using various techniques with two real models, one use in plant pathology and one use to simulate wheat biomass in function of climatic variables.

Exercise 1. Probability distributions and random sampling with R

1.1. Assume that a random variable X follows a uniform probability distribution with lower and upper bound equal to 1.09 and 3.8 respectively. Use the R function `dunif` to compute probability density for values of X in the range 0-5.

1.2. Assume that a random variable X follows a Gaussian probability distribution (Normal) with mean equal to 2.5 and standard deviation equal to 0.5. Use the R function `dnorm` to compute probability density for values of X in the range 0-5.

1.3. Plot the probability densities of X in both cases for X values in the range 0-5 (use the R function `plot`).

1.4. Generate a series of 10 values from the two distributions defined in 1.1 and 1.2 using the pseudo-random number generators of R `runif` and `rnorm`.

1.5. Repeat the procedure defined in 1.4 using a different seed specified using `set.seed`.

Exercise 2. Uncertainty analysis with a simple multiplicative model

We consider the following model with one output and two input variables:

\[ y(z_1, z_2) = z_1 \times z_2 \]

The two input variables are assumed uncertain and you are asked to perform an uncertainty analysis using the following steps.

2.1. Define a R function to run the model using the following code (need to be completed)
mod.func<-function(z1,z2) {
    ...
}

2.2. Generate N=100 random values of the two uncertain inputs z1 and z2 assuming that
\( z_1 \sim Uniform(11,31) \) and \( z_2 \sim Normal(40,5^2) \). Use runif and rnorm.

2.3. Compute the N corresponding model outputs using the function mod.func defined in 2.1.

2.4. Plot the N generated values of \( z_1 \) and \( z_2 \) using the hist R function. Overlay the true density values of the two input variables. Is N=100 enough?

2.5. Plot the N computed model outputs and calculate the mean, standard deviation, min, max, and median of these N values. Use instruction summary.

2.6. Increase N and find a value of N leading to stable results.

Exercise 3. Uncertainty and sensitivity analysis using Monte Carlo simulations and regression techniques. Application to an epidemiological model

We consider the simple generic infection model for foliar fungal plant pathogens defined by Magarey et al. (2005):

\[
W = \min \left( W_{\text{max}}, \frac{W_{\text{min}}}{f(T)} \right)
\]

and

\[
f(T) = \left( \frac{T_{\text{max}} - T}{T_{\text{max}} - T_{\text{opt}}} \right) \left( \frac{T_{\text{opt}} - T_{\text{min}}}{T_{\text{opt}} - T_{\text{min}}} \right)^{(T_{\text{opt}} - T_{\text{max}})/(T_{\text{max}} - T_{\text{opt}})} \quad \text{if} \quad T_{\text{min}} \leq T \leq T_{\text{max}} \quad \text{and zero otherwise}
\]

where \( T \) is the mean temperature during wetness period (°C), \( W \) is the wetness duration required to achieve a critical disease intensity (5% disease severity or 20% disease incidence) at temperature \( T \). \( T_{\text{min}}, T_{\text{opt}}, T_{\text{max}} \) are minimum, optimal, and maximum temperature for infection respectively, \( W_{\text{min}} \) and \( W_{\text{max}} \) are minimum and maximum possible wetness duration requirement for critical disease intensity respectively. This model was used to compute the wetness duration requirement in function of the temperature for many species and was included in disease forecast system (Magarey et al. 2005, 2007).

\( T_{\text{min}}, T_{\text{opt}}, T_{\text{max}}, W_{\text{min}}, \) and \( W_{\text{max}} \) are five species-dependent parameters whose values were estimated from experimental data and expert knowledge for different foliar pathogens (e.g., Magarey et al. 2005; EFSA 2008b). However, for some species, these parameters are uncertain due to the limited number of available data (Magarey et al. 2005) and, in such cases, it is important to perform uncertainty and sensitivity analysis.
We consider here the uncertainty ranges of the parameters reported for the pycnidiospores of the fungus Guignardia citricarpa Kiely by EFSA (2008) and displayed in the table below.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Lower bound</th>
<th>Upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tmin (°C)</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>Topt (°C)</td>
<td>25</td>
<td>30</td>
</tr>
<tr>
<td>Tmax (°C)</td>
<td>32</td>
<td>35</td>
</tr>
<tr>
<td>Wmin (h)</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>Wmax (h)</td>
<td>35</td>
<td>48</td>
</tr>
</tbody>
</table>

In this exercise, students are asked to perform an uncertainty and a sensitivity analysis of the model output to the five uncertain model parameters.

3.1. Define a R function to compute the model output \( W \) in function of the temperature \( T \) and of the five model parameters. Complete the following R instructions:

```
Wetness<-function(T, Tmin, Topt, Tmax, Wmin, Wmax) {
    fT<-(Tmax-T)/(Tmax-Topt)*(((T-Tmin)/(Topt-Tmin))^((Topt-Tmin)/(Tmax-Topt)))
    ...
}
```

3.2. Compute and plot the response of \( W \) to \( T \) (for 10<\( T \)<35°C) using a specific set of parameter values, for example \( T_{min}=10 \), \( T_{opt}=25 \), \( T_{max}=35 \), \( W_{min}=12 \), \( W_{max}=35 \).

3.3. Randomly generate \( N=1000 \) values of the five model parameters assuming uniform and independent distributions.

3.4. Compute and plot the \( N \) response curves corresponding to the \( N \) series of parameter values by completing the following instructions

```
T_vec<-seq(from=15, to=32, by=0.1)
W_mat<-matrix(nrow=Num, ncol=length(T_vec))
for (i in 1:Num) {
    ...
}
```
3.5. Look at the code below and try to understand its meaning and its outputs. Interpret the results. Which temperatures T lead to low uncertainty? Why?

```r
mean_vec <- apply(W_mat, 2, mean)
Q0.01_vec <- apply(W_mat, 2, quantile, 0.01)
Q0.1_vec <- apply(W_mat, 2, quantile, 0.1)
Q0.9_vec <- apply(W_mat, 2, quantile, 0.9)
Q0.99_vec <- apply(W_mat, 2, quantile, 0.99)

plot(c(0), c(0), pch=" ", xlab="Temperature (°C)", ylab="Wetness duration requirement (h)", xlim=c(10, 35), ylim=c(10, 60))
lines(T_vec, mean_vec, lwd=3)
lines(T_vec, Q0.9_vec, lty=2)
lines(T_vec, Q0.1_vec, lty=2)
lines(T_vec, Q0.99_vec, lty=9)
lines(T_vec, Q0.01_vec, lty=9)
```

3.6. Study of the effect of the five parameters on the model outputs using correlation coefficients. Complete the following code:

```r
corTmin <- T_vec
corTopt <- T_vec
corTmax <- T_vec
corWmin <- T_vec
corWmax <- T_vec

for (i in 1:length(T_vec)) {
  corTmin[i] <- cor(....)
  corTopt[i] <- ....
  ....
}
```

3.7. Plot the correlation coefficients in function of the temperature T. What is the most important parameter at T=20°C?

3.8. Plot the N values of W at T=20°C in function of the N generated parameter values. Is the use of correlation coefficients recommended in this case?

3.9. We now want to study the effect of the five parameters using another technique based on ANOVA. Create an experimental design combining three values of each of the five parameters. Use the following instructions and interpret the results. How many parameter combinations were generated?
para.mat<-expand.grid(Tmin=c(10, 12.5, 15), Topt=c(25, 27.5, 30), Tmax=c(32, 33.5, 35), Wmin=c(12, 13, 14), Wmax=c(35, 41.5, 48))
print(para.mat)

3.10. Compute the model outputs for all parameter combinations and for three temperatures using the following R code.

T.vec<-c(20, 25, 30)
W.mat<-matrix(nrow=243, ncol=3)

#Computation
for (i in 1:243) {
    W.mat[i,]<-Wetness(T.vec, para.mat$Tmin[i], para.mat$Topt[i], para.mat$Tmax[i], para.mat$Wmin[i], para.mat$Wmax[i])
}

3.11. Perform an ANOVA of W versus Tmin, Topt, Tmax, Wmin, Wmax taking into account all possible interactions and compute sensitivity indices (set T to 30°C). Use and interpret the following code. What is the most important parameter? The second most important? What are the most important interactions?

Tmin<-as.factor(para.mat$Tmin)
Topt<-as.factor(para.mat$Topt)
Tmax<-as.factor(para.mat$Tmax)
Wmin<-as.factor(para.mat$Wmin)
Wmax<-as.factor(para.mat$Wmax)
W<-W.mat[,3]
TAB<-data.frame(W, Tmin, Topt, Tmax, Wmin, Wmax)
Fit<-summary(aov(W~Tmin*Topt*Tmax*Wmin*Wmax, data=TAB))
print(Fit)
SumSq<-Fit[[1]][,2]
Total<-242*var(W)
Indices<-100*SumSq/Total
print(Indices)
print(sum(Indices))
TabIndices<-cbind(Fit[[1]], Indices)
print(TabIndices)
TabIndices<-TabIndices[order(Indices, decreasing=T),]
print(TabIndices)
Exercise 4. Sensitivity analysis using the Morris’s method. Application to WWDM

The purpose of this exercise is to illustrate the use of the Morris’s method to analyze the sensitivity of model output to model inputs. We consider here the WWDM model already presented in a previous lecture. This model simulates three state variables at a daily time step in function of daily temperature and radiation. We consider here one output only: the final biomass at harvest.

The R code is presented in the file UncertaintyAnalysis4.R. Install this file in your folder together with blagnacSimul.dat which includes series of climatic variables. Only the first series is used here.

4.1. Look at the R code. What are the purposes of the functions maize.model, read.weather, define.param, maize.simule?

4.2. The Morris’s method (see course) can be implement using the function morris of the R package sensitivity. Open the help file of morris and try to understand all the arguments. What does the argument r represent? What is the size of the grid considered in the program?

4.3. Run morris using UncertaintyAnalysis4.R. How many model runs were performed? Try to find where this number comes from.

4.4. Interpret the results. What is the most influential parameter?

4.5. Modify the code in order to run morris with 50 trajectories instead of 100. Are the results different?

4.6. Change the seed value and study the stability of the results.

References


Estimating Parameters for Dynamic Models
Exercises

**Purpose.** By completing this set of exercises, you will learn to:

1. Estimate individual parameters for dynamic models using field data collected specifically for this purpose, using the R programming language. In one case, by directly computing the parameter using sample observations.
2. In a second case, by using a least squares parameter estimation method in R.
3. Estimate two parameters of one component of the dynamic crop model, using least squares. In this exercise, you will learn how to “calibrate” a dynamic model that has to be simulated, making use of a single type of measurement (leaf area index in this case) and a least squares method in R. You will also compare those estimates to those obtained when using only one year of data.
4. Estimate four parameters using a Bayesian method (Importance Sampling or the Generalized Likelihood Uncertainty Estimation (GLUE) method. This approach estimates the mean value and distribution of parameters, which are useful for uncertainty and sensitivity analyses.
5. Estimate the same four parameters of the complete dynamic maize model (K, RUE, LAImax, and alpha) using measurements during two years of both leaf area index and biomass. In this exercise, you will also use the least squares method.

**The Model.** The model is the same one that you have been working with already, the dynamic crop model described on page 4 by Wallach (2006) and presented in the Exercises on Simulation of dynamic models in this course. It has 3 state variables, TT(day), LAI(day), and B(day), and a total of seven parameters (Tb, TTL, TTM, K, RUE, Alpha, LAImax). In this set of exercises, we will assume that we already have good estimates of the base temperature (Tb) and the thermal time requirements for duration between plant emergence and last leaf formation (TTL) and duration from emergence to crop maturity (TTM). We will use the different methods to estimate K, RUE, Alpha, and LAImax.

In the first exercises, you will not work with the entire model, instead, you will be able to estimate the parameters in these exercises using only parts of the entire model along with field measurements. These cases will be explained below.

**General Approach.** The use of field data to estimate parameters of dynamic models is sometimes referred to as “calibration of a model” and also “inverse modeling”. When using dynamic models, one usually wants to predict the responses of a system to input variables and parameters are given. In parameter estimation, the response of a system is measured under specific input conditions, and the unknowns become the parameters that are solved for using some statistical method (Figure 1). You will work with R programs that were created to help you learn how to use these different methods for estimating parameters of dynamic model.
Classical use of model:

Parameter estimation (inverse modelling):

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Output (responses)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
<td>model</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outputs:</th>
<th>Calibration data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated Parameters</td>
<td>model</td>
</tr>
</tbody>
</table>

**Figure 1.** Schematic showing relationship between the model, parameters and response predictions in the classical use of models and in parameter estimation methods.

The methods for the first two exercises are straightforward and are used routinely in statistical analysis of data; the second exercise is basically a regression problem and data are used to estimate a parameter for a linear model. However, the approaches used in the other exercises rely on the use of a dynamic model that has to be simulated to obtain predictions. One could write their own program to perform the estimation operations, but there are many algorithms (computer code) that do this, and it is more efficient to make use of them. R has several built-in algorithms, such as “optim” that we will use here. This algorithm can be used for just about any model, dynamic or not, and its use with dynamic models is outlined here.

To use an algorithm like “optim”, one needs to write a main program, a function for the dynamic model (as we have shown already), a function for the criterion used to choose the best parameters (such as least squares or maximum likelihood), a function for reading input data. The main program controls the initialization of the model and initial guesses of parameters, and it then calls the algorithm, passing information into it on the initial guesses of parameters, the function calculating the criterion (i.e., sum of squares of errors between observed and simulated variables). This function calls the model function, passing to the model the latest estimate of parameters, to obtain predictions, which are used to compute the function and pass it to the algorithm. The algorithm makes changes in the parameters and then calls the criterion function again, which calls the model again, etc. The algorithm may repeat this hundreds or even thousands of times to settle in on a set of parameters that result in the best criterion (i.e., least squares). After the algorithm is finished, the optimal results are passed back into the main program where outputs are displayed, saved to a file, and/or graphed. A simple diagram (Figure 2) shows the general structure of programs to estimate parameters in R using optim.

Programs that estimate parameters rely on functions to describe the model and criterion for optimization, in this case, minimizing the error sum of squares. As you create functions for this type of work, it is important to clearly identify the arguments that are passed into the function as well as values that are computed by the model and returned to the calling program. This feature allows the models in our exercises, for example, to predict B and LAI for any set of parameters that are given to it.
Exercises.

1. **Estimation of Light Extinction Coefficient.** Estimate the light extinction coefficient, K, using field measured data. In some models, it is possible to isolate a component of the model, collect field data, and calculate one or more parameters directly. The example used here is the light interception component of the model, Int(day) – the MJ of light captured by the leaves of the crop during a day, is expressed by:

   \[ \text{Int(day)} = \text{I(day)} \times (1 - \exp(-K \times \text{LAI(day)})) \]

   This can be re-written as \( \text{Int(day)} = \text{I(day)} - \text{I}_G\text{(day)} \), where

   \[ \text{I}_G\text{(day)} = \text{I(day)} \times \exp(-K \times \text{LAI(day)}) \]

   \( \text{I(day)} \) is total incoming solar radiation during a day (MJ/m\(^2\)), \( \text{I}_G\text{(day)} \) is the light that passes through the canopy and strikes the ground beneath the leaves of the crop canopy(MJ/m\(^2\)). One can measure \( \text{I(day)} \) by placing a radiometer above the canopy to record incoming light during a day, and one can also measure the light that is transmitted to the soil by placing an instrument at ground level under the canopy. The daily integral of these instantaneous measurements are \( \text{I(day)} \) and \( \text{I}_G\text{(day)} \), respectively.

   The table below shows measurements of \( \text{I(day)} \), \( \text{I}_G\text{(day)} \), and \( \text{LAI(day)} \) on fourteen different days in an experiment. Note that when these three variables are measured, the only unknown in the equation for \( \text{I}_G\text{(day)} \) is the parameter K. Thus, one can rearrange the equation to compute K by:

   \[ K\text{(day)} = - \log\left(\frac{\text{I}_G\text{(day)}}{\text{I}(day)}\right) / \text{LAI(day)} \]
Where LOG is the natural logarithm of the term. In other words, you can compute \( K \) for any day when you have measurements of LAI, \( I \) and \( I_G \). However, one would not want to estimate \( K \) based on a single measurement because of measurement errors.

For this problem, you are asked to use the R program (\texttt{paramEx1.r}) given to you to compute \( K \) for each of the measurement dates and then compute the mean value of \( K \). This table of data are already entered into a file (\texttt{paramEx1\_LightInt.dat}), and this file is read into the R program for you to work with. Also, compute the variance and standard deviation of this parameter.

The R program is available in the C:\ModelMethods\Rscript\param\estim\para mEx1 directory and is named \texttt{paramEx1.r}. Figure 3 below shows the different steps that this R program already does and also shows the calculations that students are asked to add in this exercise.

<table>
<thead>
<tr>
<th>LAI</th>
<th>( I )</th>
<th>( I_G )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.13</td>
<td>12</td>
<td>11.07668</td>
</tr>
<tr>
<td>0.31</td>
<td>17</td>
<td>13.3297</td>
</tr>
<tr>
<td>0.62</td>
<td>23</td>
<td>14.33555</td>
</tr>
<tr>
<td>0.58</td>
<td>11</td>
<td>7.116356</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>3.772761</td>
</tr>
<tr>
<td>1.05</td>
<td>15</td>
<td>7.632222</td>
</tr>
<tr>
<td>1.3</td>
<td>3</td>
<td>1.052276</td>
</tr>
<tr>
<td>1.62</td>
<td>18</td>
<td>5.732225</td>
</tr>
<tr>
<td>2.4</td>
<td>28</td>
<td>4.645473</td>
</tr>
<tr>
<td>3</td>
<td>26</td>
<td>2.888922</td>
</tr>
<tr>
<td>4.1</td>
<td>14</td>
<td>0.760164</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>1.058728</td>
</tr>
<tr>
<td>6.1</td>
<td>23</td>
<td>0.471634</td>
</tr>
<tr>
<td>7</td>
<td>25.5</td>
<td>0.200473</td>
</tr>
</tbody>
</table>

Parameter Estimation Exercise 1
\textit{(paramEx1.r)}

Set path where R script and data are. This is also where files that are created by the R program will be located after successful completion of the exercise (if written out)

Read data from a text file with headers (shown in Table 1)

Display data on R console

**Student Responsibility**
Compute \( K \) (which is named \texttt{data$K} in the R program)
Compute the mean \( K \), using the variable \texttt{meanK}
Compute the variance and standard deviation of \( K \) (\texttt{varK} and \texttt{sdK})
Display \texttt{meanK}, \texttt{varK}, and \texttt{sdK} on the R console

**End of Student Responsibility**

Plot \( K \) values vs. LAI

**Figure 3.** Flow diagram for the exercise to estimate \( K \) from field observations. A skeleton R program is available \textit{(paramEx1.r)}. Students are asked to complete the R program to estimate \( K \) in this exercise. Specific student responsibilities are given above and documented in the R code.
Questions.

a. How does the value of $K$ that you computed compare with the value given in the earlier simulation exercise ($K = 0.70$)?

b. Do you think that $K$ would vary with time of year? Why? Would you want to change your model to include a relationship that changes $K$ with time of year?

c. What do you think about changing $K$ relative to LAI or other factors? How would you do this?

d. You are asked to make a second model run. Use an editor to open the data file with observations (paramEx1_LightInt.dat) and delete half of the observations at random. Remember to keep a backup copy of this file. How much difference was there in your estimate of $K$ using half of the data vs. the full dataset? How much did the standard deviation change?

2. Estimation of RUE using Biomass and Cumulative Intercepted Light Measurements. You are asked to estimate the RUE parameter using two years of data collected (year 1 and year 2) from the Blagnac location. Data are available for this exercise at 10-day intervals during both years on crop biomass ($B(day)$) and the cumulative light intercepted during the season up until the day the $B$ measurements are made. The cumulative light interception is computed by continuously measuring the components of the radiation (incoming light minus light transmitted to the ground). This net light intercepted value is integrated over time during each day and through days such that there is a cumulative light interception value for each day of the growing season. Biomass measurements in this exercise were assumed to be taken at 10-day intervals on 10 different measurement dates in each of years 1 and 2 (see the table below for $B$ and CumInt values for each year). The file (paramEx2_B_CumInt.dat) has year 2 observations appended below those of year 1 to facilitate its input to the R program.

Table 2. Measurements of $B$ and CumInt for two years for estimating RUE.

<table>
<thead>
<tr>
<th>Year 1</th>
<th>Year 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>day</td>
<td>B</td>
</tr>
<tr>
<td>129</td>
<td>22.52</td>
</tr>
<tr>
<td>139</td>
<td>56.75</td>
</tr>
<tr>
<td>149</td>
<td>196.78</td>
</tr>
<tr>
<td>159</td>
<td>439.69</td>
</tr>
<tr>
<td>169</td>
<td>708.88</td>
</tr>
<tr>
<td>179</td>
<td>1108.84</td>
</tr>
<tr>
<td>189</td>
<td>1442.95</td>
</tr>
<tr>
<td>199</td>
<td>1836.5</td>
</tr>
<tr>
<td>209</td>
<td>2170.55</td>
</tr>
<tr>
<td>219</td>
<td>2520.06</td>
</tr>
</tbody>
</table>
The relationship between $B(\text{day})$ and cumulative Light interception (MJ/m$^2$) is assumed to be linear; the model is:

$$B(\text{day}) = \text{RUE} \times \text{CumInt(\text{day})}$$

This model has the form of $y = b*x$, a linear model with no intercept. In this sense, RUE is the $b$ parameter and CumInt(\text{day}) is the independent variable.

In this exercise, an R program has been developed to use these data and a least squares method to estimate RUE. The least squares method is the Nelder-Mead algorithm, also referred to as the Simplex Method. R has a built-in procedure called “optim” to perform this minimization task. The user must define the criterion, in this case, the sum of squares of error between simulated and observed data. The optim procedure modifies the parameters, then calls the criterion to compute the error sum of squares which in turn calls the model to compute predicted values of $B$ in this case. This procedure is repeated, using different parameters each time, until it converges to a minimum value of error sum of squares. The parameters that were used to create this minimum error are the optimal parameters. In this case, we only estimate one parameter (RUE) using measured data of $B$ (Bobs) and cumulative light interception (CumIntobs).

Figure 4. below shows the different steps in this exercise, indicating student responsibilities as well as what the code already contains.

| Parameter Estimation Exercise 2  
| (*paramEx2.r*)
| **Student Responsibility**  
| Define a function for this simple model for $B$ (see text)  
| Define a function to compute model error least square named ls_model_data  
| - In Main Program -  
| Set the path for the R program and data:  
| (C:/ModelMethods/Rscript/param_estim/paramEx2)  
| Read data on $B$, CumInt (given)  
| Display the estimated RUE and the data  
| Plot $B$ vs. CumInt data  
| **End of Student Responsibility**  

- 26 -
Plot predicted B vs. CumInt using initial guess of RUE (1.7 in this case)
Call ls_model_data with initial RUE to compute original error SS

Call optim, storing results in the variable result1
Display the best estimate of RUE (from result1)

Plot data and model predictions using the optimal estimate of RUE

**Figure 4.** Flow diagram of the *paramEx2.r* program showing student responsibility.

**Questions.**

a. You are asked to complete the R code and use it to estimate R. You are only asked to add a statement for the simple model of B vs. CumInt, a function to compute least squares, and to display the answer (RUE) and plot B vs. CumInt. Run the R code and write down the estimate of RUE and the residual error sum of squares.

b. Run the parameter estimation program again. This time, change the initial estimate of RUE from 1.7 to 1.2. Run the program again and compare your results. Change the starting RUE one more time to a value between 1.0 and 2.2. Do you get the same estimate of RUE?

c. Demonstrate the effects of your estimate of RUE by creating error in your observed data file. You should open the data file (*paramEx2_B_CumInt.dat*) and select the last datapoint each year and divide the value of B by 2. Run the program again. How did RUE change?

d. You are asked to create a graph of the sum of squares of error (on the y-axis) vs. RUE (on the x-axis). Verify graphically that the estimate obtained by the R program is the same as when you get the best RUE value from the graph.

To obtain information on the use of the “optim” procedure in R, students should use the help command in R.

3. **Least Squares Estimation of Two LAI parameters.** In this exercise, you are to use the K value estimated above in Exercise 1, and also use the original values of Tb, TTL, and TTM (7.0, 700, and 1200, respectively). In this case, a part of the maize model will be used, the LAI model. However, you do not know the RUE for this set of experiments because in this model, LAI is not coupled directly to B
(although B is dependent on LAI). The model for LAI increase per day (dLAI) is again given below, taken from the R program, for your reference:

```r
if (TT[day] <= TTL) { dLAI <- alpha*dTT*LAI[day]*max(LAImax-LAI[day],0) }
else { dLAI <- 0 }
```

LAI(day+1) is computed by adding dLAI to the previous day’s LAI in the R code as,

```
LAI[day+1] <- LAI[day] + dLAI
```

![Figure 5](image-url)

**Figure 5.** Graph of LAI vs. day of year measured in years 1 and 2. These data are to be used with the LAI dynamic model to estimate alpha and LAImax.

Table 3 below has observations of LAI(day) for two years. These data were assumed to be collected at one sample point on each of the 20 sampling dates in the two seasons of corn growth (Year 1 and Year 2).

| Table 5. Measurements of LAI every 10 days, for Years 1 and 2. |
|------------------|------------------|------------------|------------------|
| **Day** | **LAIobs** | **Day** | **LAIobs** |
| 119 | 0.042 | 119 | 0.023 |
| 129 | 0.15 | 129 | 0.063 |
| 139 | 0.439 | 139 | 0.367 |
| 149 | 1.9 | 149 | 1.178 |
| 159 | 2.1 | 159 | 5.423 |
| 169 | 3.817 | 169 | 4.929 |
| 179 | 7.69 | 179 | 5.139 |
For this exercise, you will need to run the LAI part of the model, using the weather data and parameters that affect LAI. Furthermore, we will assume that we know some of the parameters accurately (Tb, TTL, and TTM) so that we will fix those to their constant values and then estimate the remaining two parameters (alpha and LAImax). The idea is that there are other field techniques that can be used to estimate RUE, TTM, TTL, and K, but LAI_{MAX} and ALPHA are difficult to measure.

We will use the minimum sum of squares as the criterion for estimating the parameters alpha and LAImax using a frequentist approach. In other words, we want the model to predict LAI over time in the two years, and we will select the parameters alpha and LAImax such that the squared error between observed and predicted LAI is at its minimum value.

In this exercise, most of the R code will be provided. A flow chart of the program (paramEx3.r) is given in Figure 4 below. One difference between this exercise and the previous one is that a dynamic model has to be solved to simulate the response that depends on a particular set of parameters.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>189</td>
<td>4.016</td>
<td>189</td>
<td>8.763</td>
</tr>
<tr>
<td>199</td>
<td>4.018</td>
<td>199</td>
<td>4.59</td>
</tr>
<tr>
<td>209</td>
<td>7.178</td>
<td>209</td>
<td>2.717</td>
</tr>
</tbody>
</table>

Parameter Estimation Exercise 3
Estimating alpha and LAImax for a dynamic model
(paramEx3.r)

**Student Responsibility**
Define the LAI model function, the same as in the simulation Exercise, although we will not need to simulate changes in B

**Provided Material**
Read weather data in a function

Create a Least Squares Function
Set the current estimate of alpha and LAImax that are obtained from the algorithm that searches to find the best set of values
Run the model for two years (in separate calls)
Combine results from each year’s predictions
Compute and return the error sum of squares

--- In Main Program ---

- **Student Responsibility**
  
  Set the path for the R program and data:
  
  (C:/ModelMethods/Rscript/param_estim/paramEx3)

---

Read LAIobs data for two years
  
  Show the data on the screen
  
  Plot LAIobs vs. day for both years of observations

---

Define the parameter values; set start and last date
  
  Set the initial values of alpha and LAImax

---

Set initial values of parameters in a vector
  
  Call optim, storing output results in the variable result1
  
  Display result1 on the screen, which displays the optimal parameters, the residual error sum of squares, and other information

---

Retrieve optimal values of alpha and LAImax
  
  Call the least squares error function to compute the final error in prediction
  
  Simulate each year separately, using optimal values of alpha and LAImax
  
  Plot the simulated and observed results for both years

**Figure 6.** Flow diagram of the `paramEx3.r` program showing student responsibility.

---

**Questions.**

a. You are first asked to complete the R program as noted above. Write down the answers that are created in the R console for alpha, LAImax, residual model error, and Root Mean Square Error (RMSE)? You will need to compute RMSE.

b. Now re-run the model after making changes in the initial guesses of alpha and LAImax. Original values were guessed as 0.00243 and 7.0, respectively. For this second run, change these initial estimates to 0.0040 and 4.5, respectively. Run the estimation program again. Are your answers the same as the first time? Do you think that our initial guesses should affect results?

c. Now, change the data file by deleting Year 2 observations (remember to keep a back-up copy of your original data). Run the model a second time, using only
half of the data (from Year 1 observations). How does this reduced amount of data affect the parameter estimates? What about residual error sum of squares and RMSE?

4. **Bayesian Method to Estimate Four Parameters using Two Types of Measurements.** The Bayesian Method called “Importance Sampling” or “Generalized Likelihood Uncertainty Estimation (GLUE)”. In this exercise, you will work with an R program that uses this method, which differs from the frequentist approach used in the last exercise. The model is again the simple maize model used in previous exercises. When using Bayesian methods, you specify a prior distribution for the parameters based on your best knowledge of the numerical values of parameters, such as you might obtain by using a meta analysis of values published in the literature.

Using this prior distribution for each parameter, a number of random sample sets of parameters are created, each of which could be a reasonable set based on prior knowledge. Usually, thousands of sets are randomly created to ensure sufficient samples to create a smooth posterior distribution of parameters. The posterior distribution will depend on the prior distribution, observed data, and a likelihood function. At the end of the process, you will have an empirical distribution of each parameter, from which you can compute the mean parameters as well as the variances, standard deviations, and other moments if desirable.

The R program for this exercise is the file “paramEx4_v2.r”. It is programmed to estimate four of the seven parameters (RUE, K, LAImax, and alpha) using two years of measurements of both LAI and B and eleven dates during each of 2 years (Years 1 and 2). These data are in the file “paramEx4_LAI_B_Obs.dat”, in the directory C:/ModelMethods/Rscript/param_estim/paramEx4. A listing of the data from that file is shown in Table 4 and a graph of both LAI and B vs. time in days is in Figure 7.

Table 4. Observed leaf area index (LAI(day)) and biomass (B(day)) on 11 dates in each of two years. These data are to be used to simultaneously estimate four parameters for the integrated crop model (K, RUE, alpha, and LAImax), using a Bayesian method.

<table>
<thead>
<tr>
<th></th>
<th>Year 1</th>
<th></th>
<th>Year 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Obs</td>
<td>Obs</td>
<td>Obs</td>
<td>Obs</td>
</tr>
<tr>
<td>day</td>
<td>LAI</td>
<td>B</td>
<td>LAI</td>
<td>B</td>
</tr>
<tr>
<td>119</td>
<td>0.054526</td>
<td>5.107045</td>
<td>119</td>
<td>0.049418</td>
</tr>
<tr>
<td>129</td>
<td>0.07603</td>
<td>19.3626</td>
<td>129</td>
<td>0.088963</td>
</tr>
<tr>
<td>139</td>
<td>0.214702</td>
<td>33.05123</td>
<td>139</td>
<td>0.327711</td>
</tr>
<tr>
<td>149</td>
<td>1.117059</td>
<td>114.9022</td>
<td>149</td>
<td>0.82554</td>
</tr>
<tr>
<td>159</td>
<td>0.958079</td>
<td>143.3618</td>
<td>159</td>
<td>2.085218</td>
</tr>
<tr>
<td>169</td>
<td>2.430753</td>
<td>296.553</td>
<td>169</td>
<td>3.813349</td>
</tr>
<tr>
<td>179</td>
<td>3.217657</td>
<td>1009.353</td>
<td>179</td>
<td>5.115119</td>
</tr>
</tbody>
</table>
To use this R program, one has to specify distributions of the parameters to be estimated, constants for the parameters that are not to be estimated, the number of random parameter sets to create, and estimates of the model error variance in simulating each output variable in comparison with observed. The R program already has this information specified. For example, each parameter (K, RUE, LAImax, and alpha) are assumed to be normally distributed with mean and standard deviation values given in the program. The other parameters (Tbase, TTL, and TTM) are given values of 7, 700, and 1200, respectively. The number of parameter sets to create is set to N = 500, and the standard deviations of model errors are given as sigma.LAI = 2.0 and sigma.B = 200.

The results from the program are in the R console in addition to graphs. In particular, posterior distribution parameter means and variances are listed in the console and graphs show the distributions of each parameter (prior and posterior).
Questions.

a. Using the values that are given in the program that you received, run the program and tabulate important results (mean and variance values for K, RUE, LAImax, and alpha). Also, graphical outputs show the prior distributions of the parameters that were sampled (500 in the first case) and the posterior distributions of each parameter (in red). Save these results in files for later comparisons. What can you say about the estimation of each of the parameters? How different were the parameter posterior means and standard deviations relative to the prior estimates? Make a note of the errors in predicting the final biomass and LAI measurements that were made on day 219 in each year (variables LAI_y1_day219, LAI_y2_day219, B_y1_day219, B_y1_day219). Figure 9 shows the prior and posterior distributions for the 4 estimated parameters when N=5,000 runs were made, for you to compare with your results.

b. You are asked to modify the program for a second run. BUT, keep the original program; it is better to create a copy of the R program, then work with the copy for this exercise. In this case, you will examine how a change in prior distribution of two parameters (RUE and K) affect your results. Assume that the prior distributions are uniform for each of these parameters (i.e., you only know the minimum and maximum values). In the R program, you will need to change the way that the samples of K and RUE are computed, replacing the statements where these parameter sets are sampled from normal distributions. The format of the statements to specify
the lower and upper limits of a uniform distribution and to sample from a uniform
distribution for RUE are:

```r
min.RUE<-1.15
max.RUE<-2.10
v.RUE<-runif(N, min=min.RUE, max=max.RUE)
```

You will need similar statements to sample v.K; use a minimum K of 0.50 and a
maximum K of 0.85. N is the number of samples to create; an N value of 500 is
already set. Compare the results of mean parameters and variances in parameters for
this case. Also, discuss the graphed distributions that are presented. Would you have
gained much information from knowing that RUE and K are normally distributed vs.
knowing only the minimum and maximum limits of these parameters? Why?

c. In this case, you will see the effects of changing N, the number of randomly created
parameter sets. Start with the original R file, again copying it so that you are working
with a copy of the original. Now, change N from 500 to only 100. Keep the results
(numerical values as well as the graphs) for comparison with the original case with
N=500. Now, run the program again with N=1,500 (this may require up to 5 minutes,
depending on your computer). Discuss the differences among the runs. Do you think
that 1,500 runs are enough? How would you determine whether the number of runs is
adequate?

d. Consider now that we have measured K in other experiments and we only want to
estimate 3 parameters (RUE, LAImax, and alpha). Thus, you will need to again
modify the original R program (using the copy and keeping the original intact). This
time, you need to remove the statements where N random samples of K are created
and replace it with a constant K of 0.55 (that you measured). We know that these
measurements were not reliable, so this part of the exercise demonstrates the effects
of setting a parameter to an incorrect value, then estimating other important
parameters. You will also need to create N values of K to be the same value of 0.55,
similar to the way that Tbase, TTL, and TTM are set (see the R program). And, you
will need to remove the statements at the very end of the program where mean.K and
var.K are computed. Run the program with N=500. What changes do you see?
Compare all of the results again, numerical and graphical. How does this affect the
other parameters? How does this affect the computed standard deviations of model
predictions, for both LAI and B? You can also try setting other parameters (such as
RUE) to constants and only estimating the remaining set to compare with previous
results. Can you think of a case when setting a parameter to an incorrect value may
not affect your model predictions and perhaps not affect other estimates of model
parameters? Discuss this.

e. One of the requirements of using multiple types of measurements, each with different
units and ultimate errors in prediction, is that estimates of model prediction errors are
needed ahead of time. Some researchers have used variances in measurements for
each observation as estimates. There are more advanced methods for estimating these variances along with parameters, or creating a model of model errors (such as using a coefficient of variation to vary estimates of prediction error for each variable; this has advantages when observed values change considerably). But, for this question, we will continue to assume that we know ahead of time what these errors are. In the R program, we assumed that the standard deviation of LAI measurements (\(\text{sigma.LAI}\)) was 2.0 and of biomass (\(\text{sigma.B}\)) was 200 kg/ha. Note that these variances normalize calculations of likelihood so that the likelihood is dimensionless. For this question, you are asked to change these values to explore how their values will affect results using two additional cases (reduced values and increased values). For the reduced values, use \(\text{sigma.LAI}\) of 1.0 and \(\text{sigma.B}\) of 100. For the increased values, use \(\text{sigma.LAI}\) of 3.0 and \(\text{sigma.B}\) of 400. Make each run, then compare results from these runs with your other results. Discuss your results.

![Graphs showing density distributions](image)

**Figure 9.** Results of the Bayesian method (Importance Sampling) after 5,000 parameter sets were created and analyzed.

5. For this exercise question, use the optim algorithm and a least squares criterion to estimate K, RUE, alpha, and LAImax, holding Tbase, TTL, and TTM at their original values. For
this exercise, you will use the same data as in the Bayesian method (LAI and B data for Bayesian exercise.dat) in the C:/ModelMethods/Rscript/paramEx4 directory. In this case, you do not have an R program, and instead, you are asked to modify the program that was used in Exercise 3, when you estimated alpha and LAImax using only LAIobs data. In this case, you will need to revise the model function from Exercise 3 to include the equations for B, thus creating a system model in which the state variables interact with each other. Much of the program will remain the same as in Exercise 3, although you will need to add the fact that B is observed and used in the different functions.

**Questions:**

a. When you successfully run this program, do you get the same estimates as you do when you use the Bayesian estimation process?

b. Make another run, this time eliminating Year 2 of data. How does this affect your results? Discuss the implications of using this optim (frequentist) method vs. using the Bayesian method, and of using small data sets (e.g., one year vs. two years in this example exercise).

c. Using this method, work through the same questions that were asked in Exercise 4 (Bayesian method). In this case, you will also need to either normalize the error sums of squares using the model error variances.
Evaluating Dynamic Models
Exercises

Purpose. By completing this set of exercises, you will learn to:

1. calculate standard quantitative criteria for measuring how well a model reproduces observed data
2. better understand the distinction between quality of adjustment and quality of prediction
3. estimate the quality of prediction
4. better understand the origins of model error

The Model. The model is the same one that you have been working with already, the dynamic crop model described on page 4 by Wallach (2006) and presented in the Exercises on Simulation of dynamic models in this course. It has 3 state variables, TT(day), LAI(day), and B(day), and a total of seven parameters (Tb, TTL, TTM, K, RUE, Alpha, LAImax). In this set of exercises, we will assume that we already have good estimates of the base temperature (Tb), thermal time requirements for duration between plant emergence and last leaf formation (TTL), duration from emergence to crop maturity (TTM), the light interception coefficient (K) and maximum leaf area index (LAImax). In those cases where parameters are estimated, only the two parameters RUE and Alpha are estimated from the data.

Exercise 1. Adjustment and prediction quality

General instructions

The R script for this exercise is at the address “C:/ModelMethods/Rscript/evaluation/ evaluation_Ex1_2.r” (which may be different if you have installed the exercises elsewhere).

Start R, choose open a script and open this script.

Change the path name to your path.
Choose two years of data.
This is to be done in the following part of the program.
# MAIN PROGRAM #
# Evaluation Ex1 - adjustment
# TODO : change path to your working directory
path<"C:/ModelMethods/Rscript/evaluation/
# Exercice : Choose two years of data (from 1 to 10). year1<-?? year2<-???

Highlight the program from the beginning up to the line #### EXERCISE ####
Run that highlighted part

The program shows the parameter values. Note them.
The program shows the data for the two years you have chosen under the heading “measurements”. The program also calculates predicted values, for specified parameter values and prints out a table with observed and predicted LAI and biomass values. In each case the first column gives the observed values, the next column the predicted values.

The exercise involves using the table of observed and predicted values to calculate various measures of agreement.

**Part 1a. Adjustment, default parameter values**

This part uses default parameter values. (You don’t have to do anything for the parameters. The program automatically uses the default values)

Below is the listing of the exercise part of the program. Add instructions as indicated, one at a time, and execute them (by highlighting them and then choosing “execute selection”). Also, answer the questions.

You will be using your code again, so don’t erase instructions after you’ve written them.

```r
# Exercice
#Bobs is the vector of observed biomass values (all values first year then
# all values second year)
#B is the vector of predicted biomass values
#Nobs=N is the number of values

# plot the data
# Execute the following instructions, one at a time.
plot(B,Bobs)
minmaxB<-min(max(B),max(Bobs))
lines(c(0,minmaxB),c(0,minmaxB))
# what have you just plotted?
plot(B,Bobs-B)
lines(c(0,max(B)),c(0,0))
# what have you just plotted?

# Questions
#Which data point has the largest error?
#Which presentation of error do you prefer? Why?

# Calculate and print the average of the observed values, the average of
# the predicted values and model bias

# Calculate and print the standard deviation of the observed values and of # the predicted values. The standard deviation you need is
```
# (var(x)*(N-1)/N)^0.5 where var(x) is the R instruction that calculates
# sample variance.

# Calculate and print MSE = the mean squared error and RMSE = root
# mean squared error. What are the units of each?

# Calculate the decomposition of MSE into 3 terms.
# The first term is squared bias. Calculate and print the bias squared

# The second term is SDSD. Calculate and print the squared difference
# between the standard deviations of observed and predicted values

# Run the instructions below to get the remainder term
covBBobs<-cov(B,Bobs)*(Nobs-1)/Nobs
r<-covBBobs/(var.Bobs^0.5*var.B^0.5)
LCS<-2*var.Bobs^0.5*var.B^0.5*(1-r)

# Verify that MSE=bias.Squared+SDSD+LCS
# Which term makes the major contribution to MSE?

# Plot the MSE decomposition using the instruction below
barplot(c("MSE"=MSE,"Bias^2"=Bias^2,"SDSD"=SDSD,"LCS"=LCS),main="decomp
sition of MSE", ylab=(unit biomass)^2")

# Calculate and print modeling efficiency

Now run the full program. You should not get any errors.
Note the value of MSE, its components, and the value of modeling efficiency.

**Part 1b. Adjustment, estimated parameter values**

We have calibrated the model for your data. The two parameters that were estimated are RUE and Alpha. Change the program so that it uses those parameters rather than the default values.

To do so, find the following part of the program and remove the # sign from the two lines indicated by an arrow. That changes those two lines from comments to executed instructions. The instructions say to replace the parameter values RUE and alpha by the values calculated for your data.

# parameters have already estimated for all combination of year(as in exercise 5 of
parameters estimations)
# Read the parameter value from a file
data.param<-read.table("data.param.dat",header=T, sep="\t")
select <- 
((data.param$year1==year1)&(data.param$year2==year2))|((data.param$year1==year2)&(data.param$year2==year1))

param <- define.param()["nominal",]
\#param["RUE"] <- data.param[select,"RUE"]
\#param["alpha"] <- data.param[select,"alpha"]

param
# store into a file
write.table(param, file="param_Ex1.dat",quote=F,row.name=F, sep="\t")

Now rerun the full program, including all your instructions.
Note the new parameter values.
Answer the same questions as before.

Note the value of MSE, its components, and the value of modeling efficiency.

Which set of parameter values (default or from calibration) gives a better adjustment to the data?
What would you expect if you had chosen different years of data?
Which term or terms in MSE are particularly affected?

**Exercise 2, Parts 2a, 2b. Prediction using default or estimated parameter values**

We use here the same program as in exercise 1, evaluation_Ex1_2.r, including your own additions

Here we test the model against 20 years of new data.
To do so, you must activate the instructions shown below. They appear in the program shortly before the line #### EXERCISES ####
You don’t have to do anything else, this new data automatically replaces the data from the two years you used previously

    # for part 2, remove comment symbol # from next lines
    #prediction.data<-new.years()
    #print(prediction.data)
    #Bobs<-prediction.data[,1]
    #B<-prediction.data[,2]

Once you have done this, rerun your full program two times, once with the default parameter values (that is part 2a) and once with the parameter values estimated from the two years of initial data (part 2b). (You saw in part 1 how to change from one set of parameters to the other).

This may take up to a minute to run (we are running the model for 20 years now).

Note the value of MSE, its components, and the value of modeling efficiency.
Is your MSE value now an estimate of MSEP? Why? What is the target population?
You now have 4 results (adjustment or prediction, default parameters or estimated parameters). How do they compare? Of course every choice of 2 specific years will give a different result. What do you expect on the average over many choices of two years?

**Exercise 3. Cross validation**

**General instructions**

The R script for this exercise is at the address “C:/ModelMethods/Rscript/evaluation/evaluation_Ex3.r” (which may be different if you have installed the exercises elsewhere).

Start R, choose open a script and open this script.

Change the path name to your path. This is to be done in the following part of the program.

```
# MAIN PROGRAM #
# Evaluation Ex1 - adjustment
# TODO : change path to your working directory
#path<"C:/ModelMethods/Rscript/evaluation/"
```

Choose your experimental design for cross validation (a list of 2 to 10 years between 1 and 20)
This is toward the start of the MAIN PROGRAM.
You make the choice by changing the list of years in the following instruction:
`list.year<-c(1,3,5,7,9,10)`

Run the program. You do not have to do any programming here, just interpret the results. Note that only 2 parameters are estimated, RUE and alpha.

Questions:
What is the overall MSE value? It measures average error for which years?
What are the values of RUE and alpha for the model that is evaluated?

How many times is calibration done for cross validation?
If you had 25 years of data, how many times would you have to calibrate the model for cross validation?
Does the adjustment algorithm converge correctly in all cases?
How do you know?
How many times altogether was the model run for cross validation?
What years are used for calibration in each stage of the cross validation?
What are the individual MSE values for cross validation?
What is the average MSE value for cross validation?
How does this compare with MSE based on all the data? Is it larger or smaller? Is that what you would expect in general?
What is the relation of average MSE from cross validation and MSEP?
What are the parameter values of the model evaluated by cross validation?
Is this the same as the model based on all the data?
Final conclusion: What model parameters would you use for prediction? What is your estimate of prediction error? Also give the square root.

4. Comparing models of different complexity

Objectives
To show how complexity impact adjustment and prediction quality.

Model
Two models:
- Simple: the same model you have been working with
- Complex: Like the simple model except that there are two RUE value, the first “RUE” up to maximum LAI (TTL), the second “RUE_AF” after maximum LAI.

Data
Two sets of data:
- For parameter estimation (list of years)
- For prediction (list of years)

Exercise
1. Choose the years for parameter estimation
   
   "# for parameter estimation"
   list.year_estim<-c(2,5)

2. Choose the years for prediction
   
   "# for prediction"
   list.year_pred<-c(7,10,11)

3. Run the whole program.

4. Questions
   You should see a table and a barplot.

Which model gives the better adjustment to the data?
Will the results be the same for a different choice of adjustment data?
If we had a model four different RUE values, what would be the results for adjustment quality?

Is MSE for prediction better than for adjustment? Will this always be the case?
Which model gives the better prediction?

What is the best model?
What would change if you had more adjustment data?
What would change if you had more prediction data?
Programming in R
Summary of useful instruction

#Numeric vector
\texttt{c(10,6,5.7,1)}
\texttt{x<-c(10,6,5.7,1)}
\texttt{x}
\texttt{x=c(10,6,5.7,1)}
\texttt{x}
\texttt{y<-1:10}
\texttt{y<-seq(1,10, by=1)}
\texttt{y}

#Logical vector
\texttt{x<-x>5}
\texttt{x}
\texttt{z1<-y==3}
\texttt{z1}
\texttt{z2<-y>=1}
\texttt{z2}

#Vector of characters
\texttt{x<-c("I", "like", "Montpellier", "very", "much")}
\texttt{x}

#Matrix
\texttt{M1<-matrix(0, nrow=2, ncol=3)}
\texttt{M1}
\texttt{M2<-rbind(c(1,1,1),c(1,1,1))}
\texttt{M2}
\texttt{M3<-cbind(c(20,20),c(20,20),c(20,20))}
\texttt{M3}
\texttt{M4<-cbind(M1,M2,M3)}
\texttt{M4}

#Vector and matrix manipulation
\texttt{y}
\texttt{y[2]}
\texttt{y[2:6]}
\texttt{M4}
\texttt{M4[2,]}
\texttt{M4[,2]}
\texttt{M4[2,2]}
\texttt{y[2]<-100}
\texttt{y}
\texttt{y[y>=100]<-2}
y
z<-2*y
y2<-y+z
y2
M5<-t(M1)
M2%*%M5

#List
MyList<-list(x,y,M1, M4)
MyList
MyList[[3]]
MyList[[1]][3]

#Data frame
City<-c("Lille", "Montpellier", "Paris")
Rank<-c(13,9,1)
Weather<-c("Rainy", "Sunny", NA)
TAB<-data.frame(City, Rank, Weather)
TAB
TABS$Rank
TAB[2,]

#Simple mathematical operators
1+1
1-1
1*2
1/2
exp(0)
log(1)
sin(pi/2)
x<-c(1,1,1,1,1)
sum(x)

#loops
X<-matrix(seq(1,2000,by=1), nrow=1000, ncol=2)
X

Y<-rep(NA,1000)
Y

for (i in 1:1000) {
  Y[i]<-sum(X[i,])
}
Y

apply(X,1,sum)
apply(X,2,sum)

#Conditional execution
y<-1:100
for (i in 1:100) {
  if (y[i]>50) y[i]<-50
}
y
z<-1:100
z[z>50]<-50
z

#Graphics
par(mfrow=c(2,2))
x<-seq(0,2,by=0.1)*pi
y<-sin(x)
z<-cos(x)
plot(x,y)
plot(x,y,type="l")
plot(x,y,type="l")
points(x,z)
plot(x,y,type="l",xlab="Input", ylab="Output", xlim=c(-1,7))
lines(x,z,lwd=3,lty=4)

#Reading external files
path<" C:/ModelMethods/Rscript"
setwd(path)
weather<-read.table("blagnacSimul.dat",header=T, sep="\t")
weather<-weather[1:100,]

par(mfrow=c(1,2))
hist(weather$Tmin,xlab="Tmin (°C)", main=" ")
plot(weather$Tmin,weather$Tmax, xlab="Tmin (°C)", ylab="Tmax (°C)")

#Function
MeanTemp<-function(Tmin, Tmax) {
  Tmean<-(Tmin+Tmax)/2
  return(Tmean)
}

TmeanBlagnac<-MeanTemp(weather$Tmin, weather$Tmax)
TmeanBlagnac

#Probability distribution
par(mfrow=c(3,2))
set.seed(123)

help(Normal)
help(Uniform)

# Cumulative probability distribution
x <- seq(-10,10,length=100)
plot(x, pnorm(x,mean=0,sd=1), type="l", xlab="x",ylab="F(x)", main="Cumulative distribution")
lines(x, punif(x, -10, 10), type="l", lty=2)

# Probability density
plot(x,dnorm(x, mean=0, sd=1), type="l", xlab="x", ylab="f(x)", main="Densite")
lines(x, dunif(x, -10, 10), type="l", lty=2)

# Random sampling
sampleNorm <- rnorm(50, mean=0, sd=1)
sampleUnif <- runif(50, -10, 10)

plot(sampleNorm, xlab="Index",ylab="x", main="Normal distribution")
abline(h=0)
plot(sampleUnif, xlab="Index",ylab="x", main="Uniform distribution")
abline(h=0)

hist(sampleNorm, xlab="x", main="")
hist(sampleUnif, xlab="x", main="")

#Regression
y<-weather$Tmin[1:100]
x<-weather$Tmax[1:100]

Fit<-lm(y~x)
print(Fit)
summary(Fit)

par(mfrow=c(1,1))
plot(x,y)