RESEARCH PROGRAM ON Dryland Systems

CGIAR

December 2015

Manual for Computer Lab Practice on Multi-Agent System (MAS) for Simulating Coupled Community-Landscape System Dynamics

Quang Bao Le

Food security and better livelihoods
for rural dryland communities

Manual for Computer Lab Practice on Multi-Agent System (MAS) for Simulating Coupled Community-Landscape System Dynamics

Quang Bao Le

Used for Training of Trainer (ToT) Course on "**Integrated, gendered Systems Modelling Research Approach to Dryland Systems: From Concepts to Practices and Implementation**", 13 - 22 September 2015, Cairo, Egypt, Organized by CRP Dryland Systems (PMU) and ICARDA (CDU).

Computer Lab 1: Getting started with NetLogo

Objectives

The primary objective of this chapter is to familiarize you with the basic elements of NetLogo:

- the Interface tab
- Information tab
- Procedures tab;
- Graphical displays and their controls;
- The four built-in types of entities;
- The basic organization of NetLogo code.

A Quick Tour of NetLogo

First start

Start NetLogo 4.1, click on "Help" and then "NetLogo User Manual". This opens NetLogo's extensive documentation, which appears in your web browser. The "**Interface Guide**" and "**Programming Guide**" will be essential as soon as you start writing your own programs, so you should also become familiar with them.

Before proceding any further, work through "**Tutorial #1: Models**" in the NetLogo User Manual. Make sure you understand the setup and go buttons, sliders and switches, plots and monitors, what the "view" is and how to adjust its settings, how the view's coordinate system and max-pxcor and max-pycor work, and how to open and play with the Models Library (*your play at home*, *please*)

Look at and try some of the models in the **Models Library**. Let pick **Wealth Distribution** model (**File\Models Library\Social Science\Wealth Distribution**). Be sure to look at the **Information tab** to see a written description of the model and what you can use it for and learn from it.

> *One part of the models library is especially important: the "**Code examples**". This section includes many well-documented examples of how to do specific things (e.g., control the order in which agents execute, read input files and write output files, import a picture, histogram your results, control colors) in NetLogo. Whenever you are not sure how to program something, you should look in the code examples for ideas or, often, code you can use directly.*

From the window of Wealth Distribution model, please explore:

Interface tab

There are 3 important elements:

- Elements allow users to **run a programming procedure** or the whole simulation program (the **Button**)
- Elements allow users to **externally adjust model global variables (i.e. model parameters)** in different modes: continuous value (the **Slider**), dummy choice (the **Switch**), multinomial choice (the **Chooser**), enter an event value (the **Input**).
- Elements allow users to observe behavior of different system entities during a simulation run: the **World** showing spatial behavior of modeled entities over time, the **Plot** showing temporal dynamics of an system property at an aggregated level (specified by users). The **Monitor** reports an instant numeric at an certain time point.

Output helps reporting and recods textually (can be convert to numeric files) the outputs as specified.

The Information table

containing information about the model. Users can edit the information as wanted.

Procedure/Code tab

The programming pad for the model codes.

Agent types

There are four types of "agent" in NetLogo:

- Mobile agents, which in NetLogo are referred to as "**turtles**". (Later, we will learn how to create our own "breeds", or kinds, of turtles.)
- **Patches**, which are the square cells that represent space. The patches exist in the "world", the rectangular grid of patches displayed on the Interface tab.
- **Links**, which each connect two turtles and provide a way to represent networks.
- **The observer**, which can be thought of as an overall controller of a model and its displays. The observer does things such as create the other agents and contain global variables.

  Note that this use of the word "agent" is a little different from how it is typically used by agent-based modelers. When talking about ABMs instead of NetLogo, we use "agent" to refer to the individuals in a model that make up the population or system we are modeling, not to things like patches and the observer.

Variables

Each of these types of agent has certain variables and commands that they use. They include **built-in variables** and **user-defined variables**

There are several important *built-in variables* for each agent type, which you can find in the NetLogo dictionary by clicking on the special category "Variables". These built-in variables represent things like location and color, which are used in almost all models.

User-defined variables can be **set at the begin of the Procedure space, or set within a program procedure (having the meaning within the context of the procedure only)**

When you write a program you define the additional variables that your model needs for each agent type. Variables belonging to the observer are automatically **"global" variables**",

Computer Lab 2: Construct an OOP structure of your simulation model

Preparation

- A document file of some “**code resources.doc**” (in your USB stick)

- Printed agent-based simulation chart for "Example System" (**Fig. 1**)

- Printed OOP slide (**Fig. 2**)

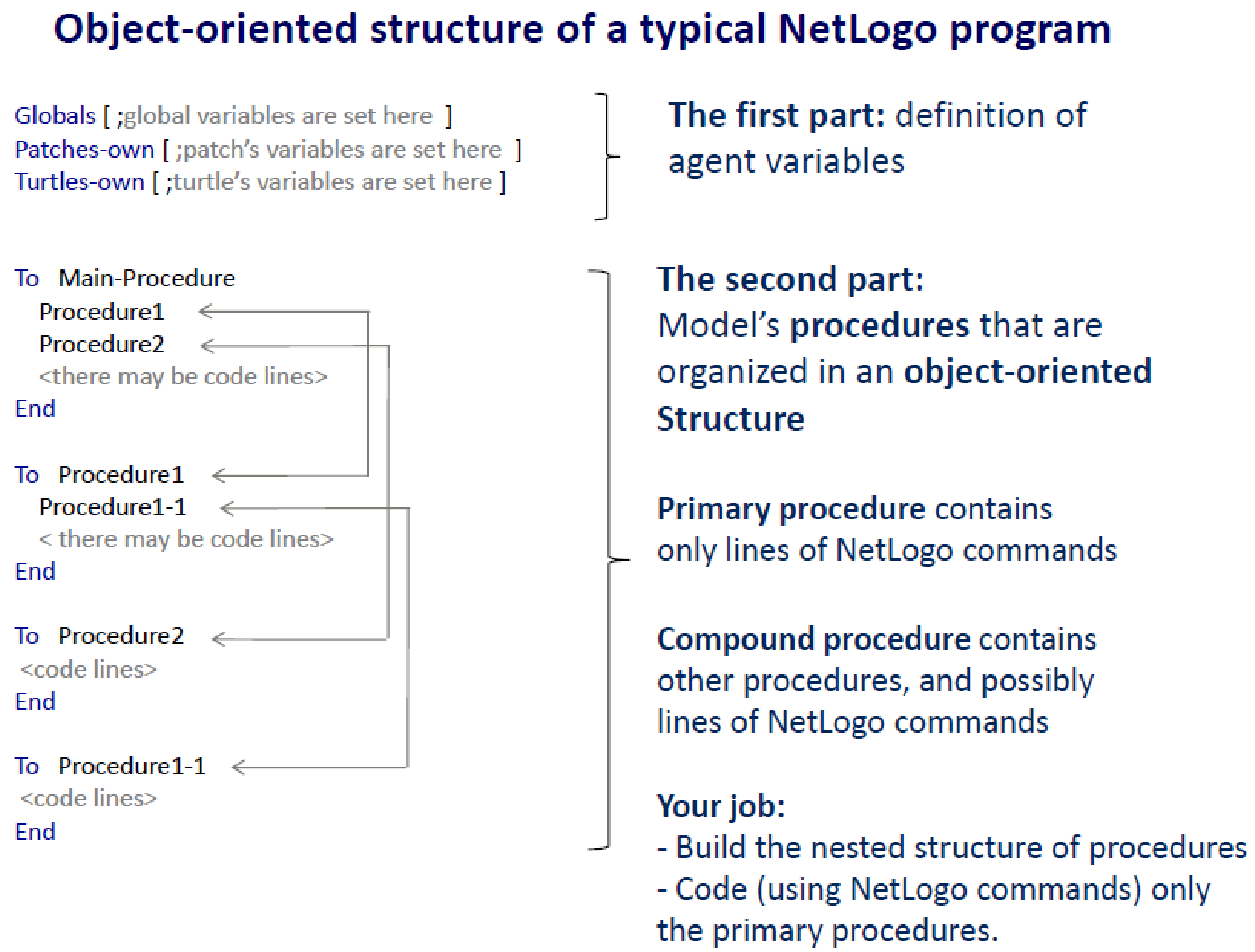


Fig. 1 Object-oriented structure of a typical NetLogo simulation program

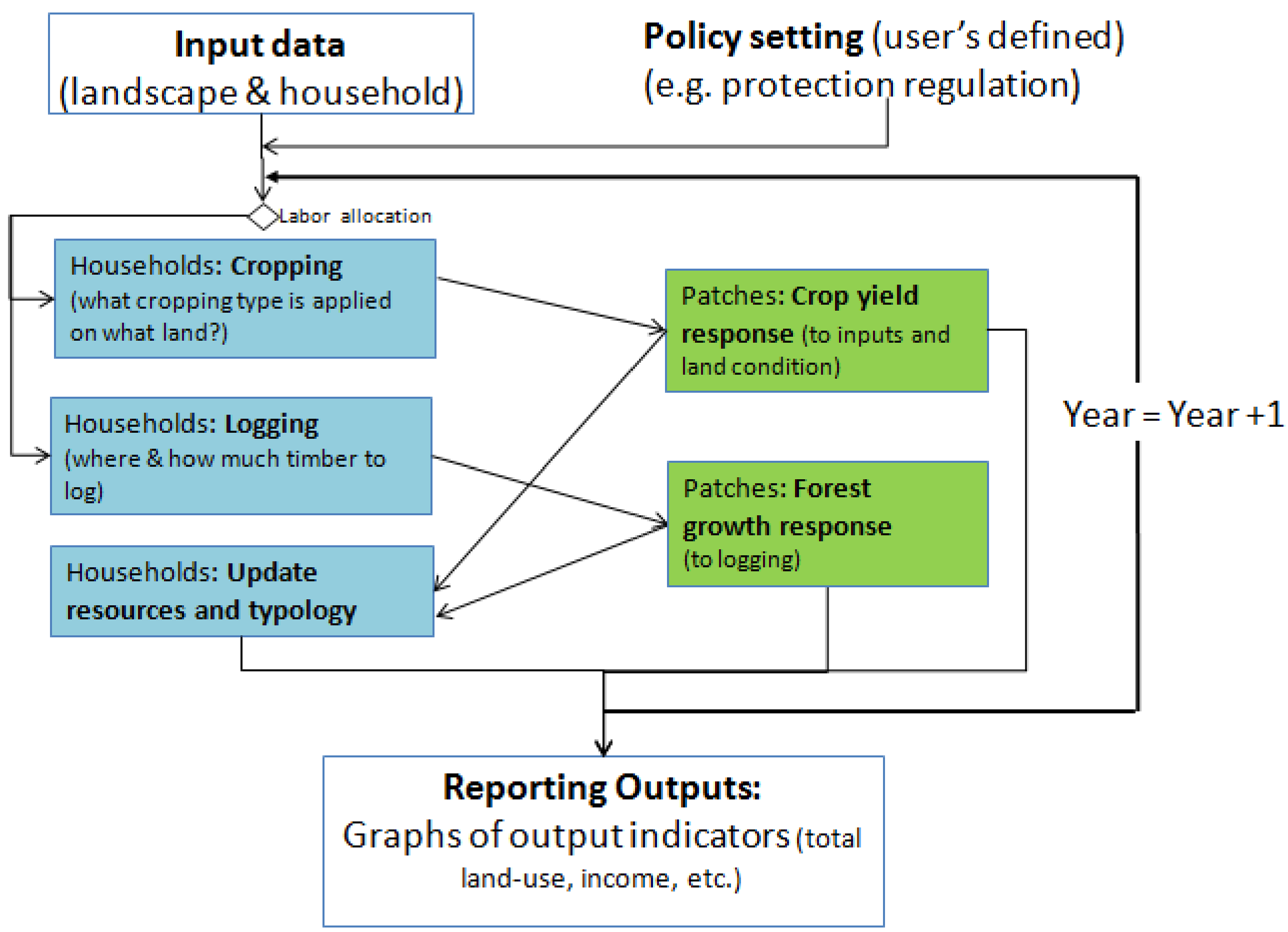


Fig. 2. Agent-based simulation procedure for the "Example System" (a sub-set of Hong Ha commune, Vietnam)

Let's set-up your agent's variables in correct places

```
global
[
; to avoid spend time to type, please copy relevant those from code resources.doc
; and paste here
]
patches-own
[
; to avoid spend time to type, please copy relevant those from code resources.doc
; and paste here
]

turtles-own
[
; to avoid spend time to type, please copy relevant those from code resources.doc
; and paste here
]
```

Let's layout the data import procedure

First, let outline the broad procedure:

```
To Import-Data
Import-Landscape-Data
Import-
Household-Data
End
```

Then, immediately layout the procedures included in the Import-Data

```
To Import-Landscape-Data
; details will be elaborated later
End

To Import-Household-Data
; details will be elaborated later
End
```

Let link `Import-Data`, `Import-LandscapeData`, `Import-HouseholdData` to command buttons in the Interface tab.

Let's layout the main simulation procedure

Let's learn the structure of the "main" procedure from the WD

model: In Procedure tab of the WD model

- Click the drop-down menu "Procedure"
- Click on "Go" (this is the main procedure of the WD model)
- Copy the Go procedure and Paste to the Procedure tab of our working model (Landuse_change1.nlogo)
- Modify the sub-procedures in Go in according to the behavior concept we discussed in the Lecture Note (20April.2011)

Design the main (named Simulate) procedure that follows the flowchart of the simulation process (see the "Flowchart of simulation process" slide):

```
To Simulate
; Household's procedures
Farmland-Choices ; the household decides to practice what farming
type on what
                 ; patch
Logging          ; the household decides to log tree(s) in what
patch

; Patch's procedures
```

```
Crop-Yield-Response     ; a patch's sub-model for crop yield
response
Forest-Growth        ; a patch's sub-model for forest growth

; household's procedures
Income-Update      ; a procedure for updating household income
Age-Update         ; a procedure for updating the age of the
household head
Land-Update        ; a procedure for updating household's land
Household-Type-Recategorize  ; a procedure for reclassifying
household's wealth class
; Patch's procedures
Forest-type-conversion  ; a patch's sub-model for forest type
transition based on
                        ; forest-yield
tick                    ; the modeled system advances one time
unit (i.e. one tick)
Plotting-outputs        ; a set of sub-procedures to plot output
indicators
if ticks >= stop-when [stop] ; stop-when is a user-defined variable
in Interface
                             ; tab, i.e. kind of global variables
End
```

Computer Lab 3: Temporally and spatially modeling an environmental process

Objectives

To implement a dynamic forest growth model (ForestGrowthResponse procedure) in the MAS-HES model, thereby learning how to *formulate* and *integrate* an environmental dynamic model into an MAS model of the coupled human-environmental system

Why should forest growth dynamics be considered?

To justify the relevance of including a specific environmental process *E* (among many other environmental processes) in your modeling framework, you should give some reasons to justify why.

In the regional context of our case study (Vietnam uplands), some reason s for consideration of forest growth dynamics are:

- Forest coverage and quality are crucial for maintaining many key ecosystem services such as watershed protection, biodiversity, long-term livelihoods, traditional culture, etc..
- Forest conservation /management is an important aspect of national policies
- Local communities are largely forest independents.

What do criteria need for a sub-model of forest growth dynamics within our MAS-LUCC model?

You can search in related literature, or develop yourself a model of forest growth dynamics. With the current status of forest growth modeling science, there would be “too” many options for you. Here, you need to think about what criteria need for defining your choice of such a model. The minimal criteria would be:

- Theoretically sound
- Spatially and temporally explicit (as required by our MAS-HES model),
- Responsive to human interventions, such as logging activities,
- Fairly simple and easy to parameterize empirically.

The model of forest growth dynamics described in Le et al. (2008) (PDF file in your References folder) would meet these criteria. Thus, this can be an option.

Sub-model of forest growth

Predictive variable: forest stand basal area at time point *t* (${}^{t}P_G$) as an indicator of forest yield of a forested patch. Unit: m^2 of tree basal at breast height per ha of land (m^2/ha)

Main function/equation: $${}^{t}P_G = ({}^{t-1}P_G + {}^{t-1}Z_G) - G_{removal} \qquad (1)$$

where:

${}^{t-1}P_G$ - stand basal area in the previous year (t-1)

${}^{t-1}Z_G$ - natural increment of stand basal area in the previous year (t-1)

Thus, component (${}^{t-1}P_G + {}^{t-1}Z_G$) in equation (1) is about the natural dynamics of forest.

$G_{removals}$ – the amount of basal area removed from the patch, caused by logging activities of household agents. Thus, the component $G_{removals}$ is event-driven, dependent on the decision-making of household agents. If there is no logging activity on the patch, $G_{removals}$= 0 and the dynamics is totally a natural growth.

Function to calculate Z_G: $\quad Z_G = a(P_G)^{\varepsilon} - b(P_G)$ (2)

where: ε is a very small constant ($\varepsilon \rightarrow 0$),

a and *b* are coeficients that are estimated as follows:

$$a = {}^{max}Z_G \,/\, [({}^{equil}P_G)^{\varepsilon}\,(\varepsilon^{\varepsilon/(1-\varepsilon)} - \varepsilon^{1/(1-\varepsilon)})] \quad (3)$$

$$b = {}^{max}Z_G \,/\, [{}^{equil}P_G\,(\varepsilon^{\varepsilon/(1-\varepsilon)} - \varepsilon^{1/(1-\varepsilon)})] \quad (4)$$

where ${}^{max}Z_G$ is the maximal growth rate of stand basal area, and ${}^{equil}P_G$ is the stand basal area at the equilibrium state of the forest stand (also called natural basal area) (see Fig. 1). The values ${}^{max}Z_G$ and ${}^{equil}P_G$ are often available in forest science literature, or can be estimated by forestry experts. The constant ε can be set at a very small value (e.g. $\varepsilon = 10^{-6}$).

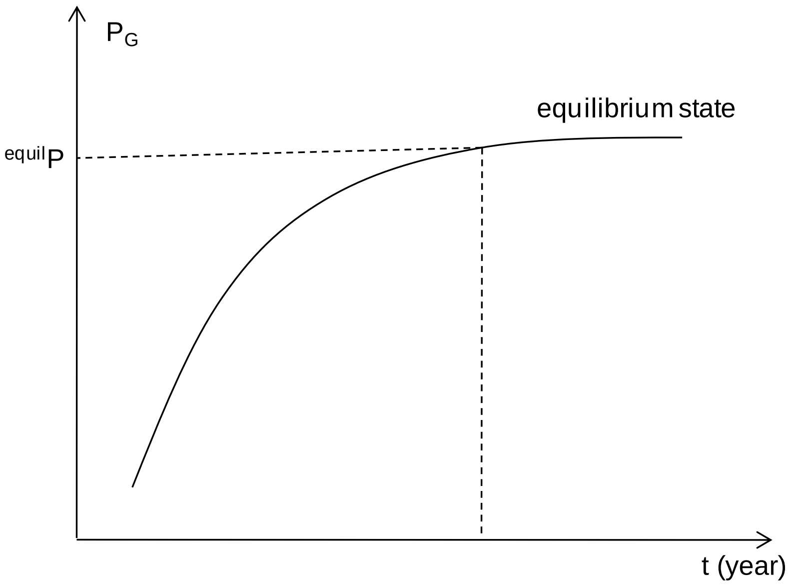


Figure 1. Forest growth curve (without human interventions) and the equilibrium state.

Input parameters – natural growth compoment (${}^{t-1}P_G + {}^{t-1}Z_G$): In short, in order to predict the forest yield (${}^{t}P_G$) in the absence of human intervention, we only need to know the input parameters marked in yellow in equations (1) - (4), i.e. P_G, ${}^{max}Z_G$, and ${}^{equil}P_G$.

Our strategy to provide inputs for the natural compoment of the forest growth sub-model is as follows:

P_G – we will generate a grid of current forest stand basal area based on plot-based measurements and the current land use/cover map (Generate-ForestYield procedure).

${}^{max}Z_G$ and ${}^{equil}P_G$ – will be estimated by forestry experts or found from scientific literature.

Function to calculate $G_{removals}$: $\quad G_{removals} = G_{logged} + G_{damage} + G_{mortality}/T$ (5)

where:

G_{logged} – the harvested amount, i.e. the basal area logged by human agent(s),

G_{damage} - logging damage, taking place imediately at the time of logging event

$G_{mortality}$ - logging-driven mortality, occuring over some years (T) after the logging event (see Fig. 2).

The calculation of G_{damage} and $G_{mortality}$ can be based on the empirical study of logging impacts:

$$G_{damage} = {}^{t-1}P_{Gr}\,(0.0052\; G_{logged}/g_{logged} + 0.0536) \quad (6)$$

$$G_{mortality} = {}^{t-1}P_{Gr}\,(0.0058\; G_{logged}/g_{logged} + 0.0412) \quad (7)$$

where:

g_{logged} – the average basal area of a logged tree.

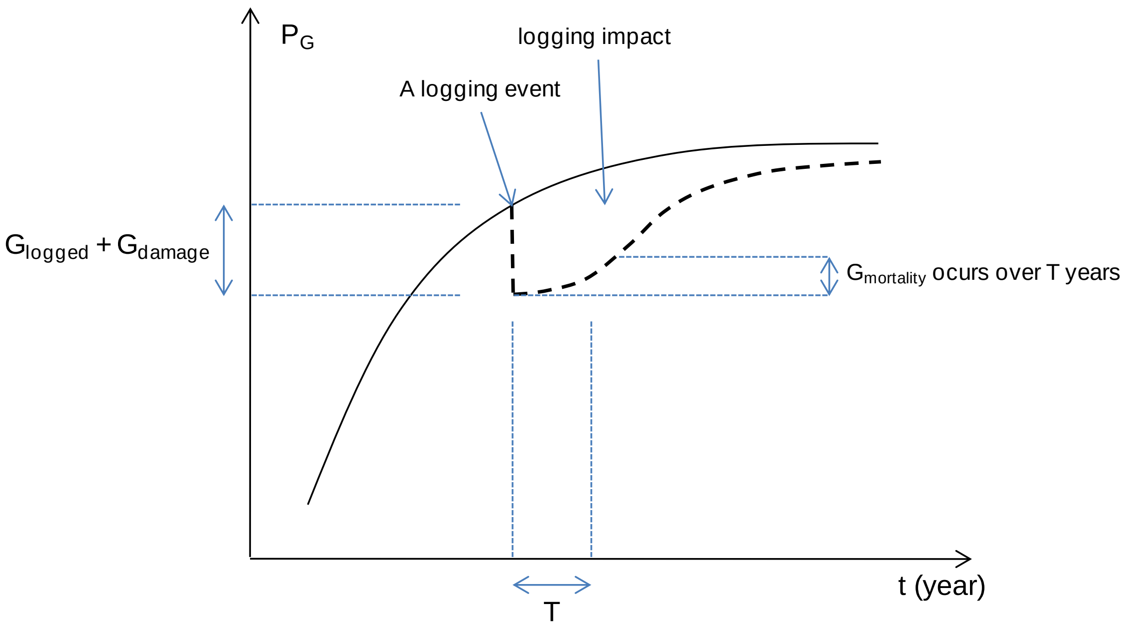


Figure 2. Different quanta of logging impact on natural forest growth dynamics.

***Input parameters – human-induced component $G_{removals}$*:** In short, in order to quantify the loss of forest yield ($G_{removals}$) caused by logging activities of human agents, we only need to know the input parameters marked in yellow in equations (5) - (7), i.e. P_G, G_{logged} and g_{logged}

Our strategy to provide inputs for this component:

P_G – by Generate-ForestYield procedure

G_{logged} – determined by Logging procedure, which is a household/turtles decision-making routine (see the Simulate procedure)

g_{logged} – can be easily estimated by interviewing local key informants.

Lesson can be learnt and possible analogical extensions:

- It is not bad if one can, analogically process-based, follow the same steps to specify another sub-model of environmental phenomenon.
- Other meaningful environmental processes to think about: soil nutrient dynamics, crop yield dynamics, etc.

Let's implement the theoretical model above in the framework of our MAS model

(`ForestGrowthResponse` procedure and some other associated "small" procedure)

- Let's open your previous version: **landuse_change_2013_ver1.nlogo**

- Create a new procedure for generating the input for variable P_G (**`Generate-Forestyield`** procedure)

  - Please see the Generate-Forestyield procedure in the *Code Resources.doc* attached (part 2)
  - Read the explaining text and make sure that you can understand the procedure. *If you have any questions, please write me an e-mail and I will help immediately.*
  - Cut-and-Paste the Generate-Forestyield and Show-StandBasalArea procedures into the Procedure tab of your model.

Then, include the Generate-Forestyield procedure in the Import-Landscape procedure. This means that when you import landscape data, the model will automatically generate the current state of P_G.

To **import-landscape**
import-landuse
import-elevation
import-slope
import-wetness
import-upslope
import-droad
generate-forestyield
End

- Create the `ForestGrowthResponse` procedure
  - Please see the `ForestGrowthResponse` procedure in the *Code Resources.doc* attached (part 3)
  - Read the explaining text and make sure that you can understand the procedure. *If you have any questions, please write me an e-mail and I will help immediately.*
  - Cut-and-Paste this procedure into the Procedure tab of your model.

Switch to the *Interface* tab to design the “stop-when” chooser: *Interface* tab → *Button* dropdown menu → click Chooser → Link to your “Stop-when” variable and define the time choices.

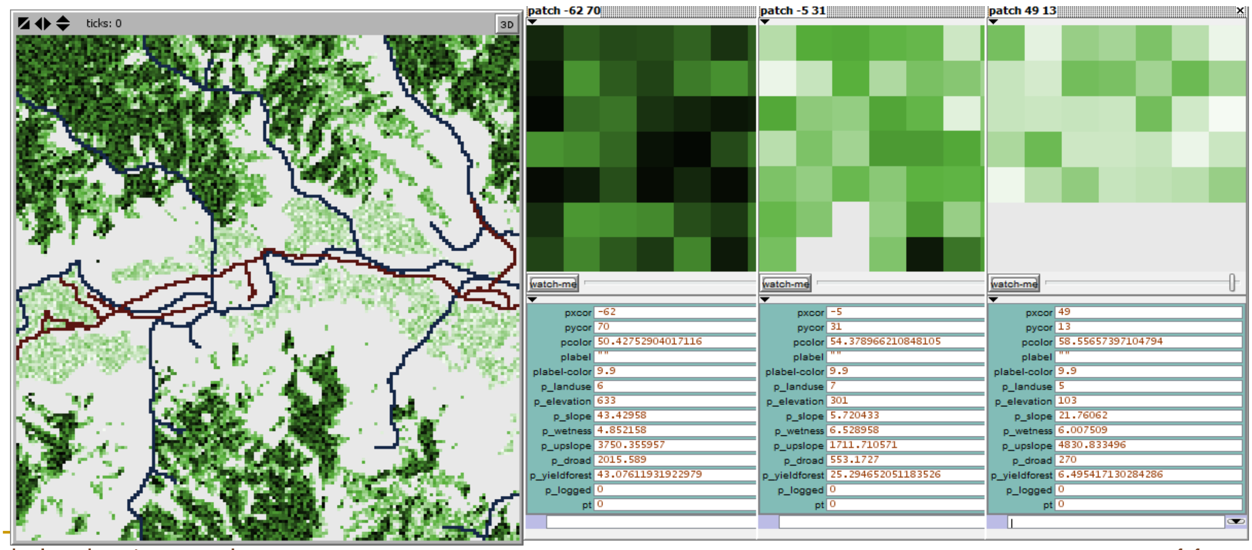
Don’t forget to *Save as*: landuse_change_2011_ver2.nlogo

Please note that the process of your model developing is often iterative like that. It is very usual and always requires your “connective thinking” during the working process.

- Do “`To Simulate`”:
  - Select your simulation period (Stop-when ?) (e.g. 50 years)
  - Right mouse-click on one rich forested patch (i.e. in dark green color) → select “Inspect patch x y”. Do the same thing with green and light green patches. So, now you have three small windows for monitoring how the ForestGrowthResponse procedure works in different forested patches (cover by different forest cover types)
  - Click “To Simulate”, watch the changing in the overall landscape, the local forest landscape in the three small window, the changing in the forest yield values in the three inspected patches.

  (see Fig. 3 to image how the product look like).

a. Initial status (tick = 0)



b. After 20 years (tick = 20)

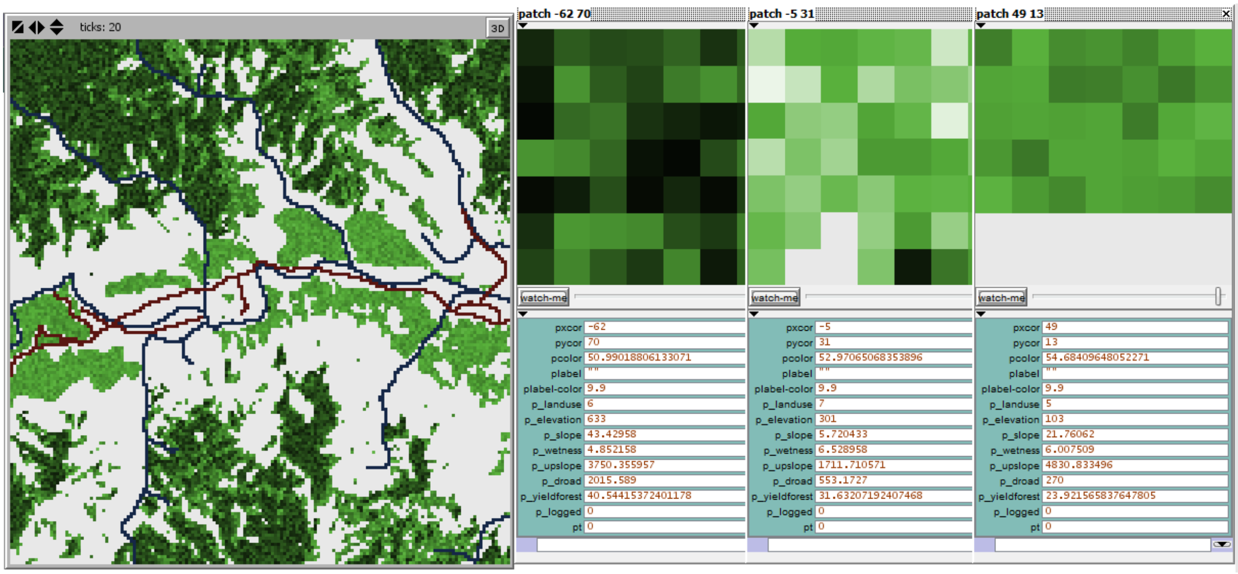


Figure 3. Forest yield in three different patches (of different forest cover types) at the initial state (a) and after 20 years (b).

Your homework

- Concentratively spend 2-3 hours to go through this tutorial. Write down any problem you face. If you have a problem, at first spend sometimes to think how to solve it (discussion within your group would be great!). If you still face the problem, please communicate with me via e-mail (or a short meeting with me if you feel necessary).
- Please give your remarks about the differences in the forest growth in the three different inspected patches (as in Fig. 3). And explain why?
- What are the limitations of the model? What should be the meaningful extensions of the model?
- In the *Interface* tab, please try to design plots to monitor the forest yield dynamics at different spatial extents:
  - 3 monitoring plots for forest yield in 3 forested patches (with low, medium and rich initial forest yield)
  - 3 monitoring plots for spatial average of forest yield in the areas of 3 different forest type: forest plantation (p_landuse = 5), open/poor natural forest (p_landuse = 7), and closed/rich natural forest (p_landuse = 6).

Computer Lab 4: Modeling human decision-making process: Heuristic process/rule-based approach

Objectives

- To import household data and create the *tenure links* between household and land resources
- To understand and implement a heuristic *rule-based decision-making* process on selective logging activities of mountain farmers in central Vietnam (Logging procedure) in the MAS-LUCC model, thereby establishing a *first version of a coupled human-environment system.*

Overall structure of household decision-making process about land and forest uses

A review
There are different approaches to formulise household's decision-making about land uses. Each of them has certain *pros* and *cons* with respect to the real human behavior.

- The *goal-drive/optimization approach* represents optimizing (or rational) behavior of agents, i.e. agents take simultaneous decisions by solving a mathematical programming model. The optimization approach can capture economic trade-offs in land-use decisions. However, the approach is often discarded as being unrealistic in the context of developing countries.
- The *heuristic process/rule-based approach* represents reflex behavior, i.e. agents take sequential decisions following a decision tree or a rule set for selecting options based on current conditions. The heuristic approach is comprehensive in many aspects: no goal assumed, no anticipated values of the options needed, and no requirement of large amounts of economic information. However, rigorous validation of a set of many rules across a diverse human population is a great challenge.
- Since no single technique or theory for modeling human decision-making is desirably comprehensive, researchers within the MAS community search *hybrid approaches* to model land-use decision making of the real-world population.

A hybrid approach
Here, we follow, for instanc, a hybrid approach as described in Le et al. (2008). Household's decision-making process consists of two sequential steps:

- *Strategic decisions:* The agent strategically partitions its resources pool (e.g. labor) into sub-budgets for each production line (e.g. crop production, livestock production, forest harvesting, of-farm activities, etc.). The composition of resource flows to production lines is specific for each livelihood typology that is defined by a specific structure of household assets. The livelihood structure of an households includes five core categories of assets (or capitals):
    - human assets: e.g. labor, education, health, skills, etc.,
    - social assets: e.g. membership of social associations, political powers, etc.
    - financial assets: e.g. cash income,
    - natural assetss: e.g. land endosement, owned annimals, etc.
    - physical assets: e.g. transportation means, farming tools/equipments, etc.

  Livelihood typology of household can be identified based on prior information, literature and/or multivariate statistics (e.g. principle component analysis – PCA, cluster analysis – CA) using multivariate datasets obtained by surveys.
  Because these livelihood variables can vary over time (e.g. income and land endowment), the livelihood typology of household can be changed accordingly, resulting in changes in the land-use behaviour of the households.
- *Annual/seasonal decisions*: In each production line, iterative processes occur and are constrained by the allocated resource budget (e.g. labor budget). To be relevant to

concrete production types, heuristic process/rule-based or the goal-oriented approach will be chosen to model annual/seasonal decisions.
In Hongha commune:
- Forest extraction activities: heuristic process/rule-based approach seems relevant.
- Crop production: goal-oriented/optimization approach may be relevant.

Import household data

A household dataset as shown in **household69.xls** can be obtained from either secondary sources (e.g. population census data, etc.), or household survey (with a land-use focus).

To be readable by NetLogo's commands, the Excel file **household69.xls** should be saved as a text file (either *DOS* or *Table delimited text* format) (e.g. **household69.txt**), in which the first row of variable names was deleted.

The import-households procedure consists should consist of three sequential steps:

- Import the matrix of household variables (file: household69.txt) into the turtle variables.

```
file-open "<path to your data folder>\\household69.txt"
foreach sort turtles
[ask ?
  [set h_var1 file-read
   set h_var2 file-read
   set h_var3 file-read
   ...
   set h_varn file-read
  ]
]
file-close
```

  Note: the top-down order of the household variables listed in the codes above has to exactly match with the left-to-right order of variable columns in the data file (household69.txt).

- Locates households at their house's positions over the landscape

```
ask turtles
[set  xcor  h_x   ;  h_x  is  the  Easting  coordinate  of  the
household's house
set  ycor  h_y     ;  h_x  is  the  Northing  coordinate  of  the
household's house
set shape "person"
set color red
set size 2
]
```

The whole import-households procedure can be seen in the **code recources.doc** file (section 5).

Define strategic labor allocations

A `LivelihoodStrategy` procedure that generates strategic labor allocation for different production lines based on livelihood typologies.

```
;***********************************************
To LivelihoodStrategy
ask turtles with [h_type = 1] ; paddy-based and poor farmers
   [set L-farming     0.6 * H_labor * 360 ; 60% time for farming
activities
```

```
    set L-logging     0.2 * H_labor * 360 ; 20 % time for logging
activities
   ]
ask turtles with [h_type = 2] ; uplandcrop-based and poor farmers
   [set L-farming     0.5 * H_labor * 360 ; 50% time for farming
activities
    set L-logging     0.3 * H_labor * 360 ; 30% time for logging
activities
   ]
ask turtles with [h_type = 3] ; off-farm-oriented and better-off
farmers
   [set L-farming     0.5 * H_labor * 360 ; 40% time for farming
activities
    set L-logging     0.1 * H_labor * 360 ; 10% time for logging
activities
   ]
End
;*********************************************
```

Notes:

- The initial household livelihood typology (h_type variable) can be pre-defined based on multi-variate statistic analysis of household data.
- It is possible to model the change of h_type variable over time (see Le et al., 2008). For a given household, if h_type changes then its strategic labor allocation also change accordingly. However, in the versions of the model within this course, we assume that the h_type variable does not change overtime.

An example of heuristic process/rule-based procedure of household decision-making (Logging procedure)

Naturally, the process that farmers to decide their logging activities can be as follows:

- *Where to log*?
- Criterion 1: timber availability. E.g. considered patches are with `p_yieldforest` > 29 (29 m2/ha is the threshold for rich/closed natural forest).
- Criterion 2: minimal transaction cost. Considered patches should be nearest to household's house.
- Criterion 3: accessibility to the timber resources. This links to forest protection zoning policy.

- Given the location for logging determined, *how much timber to log?*
- For each logged patch: $G_{logged} = n \times g_{logged}$
  where: n- number of tree to log and g_{logged} – average size of the logged tree(s). These two parameters can be estimated through rapid field surveys or local forestry experts.

- Repeat the above steps until the allocated labor budget is finished

The detail of Logging procedure

```
;*********************************************

To Logging
; Now, to log forest tree
ask patches [set P_logged 0] ; set P_logged to the default value (0)
```

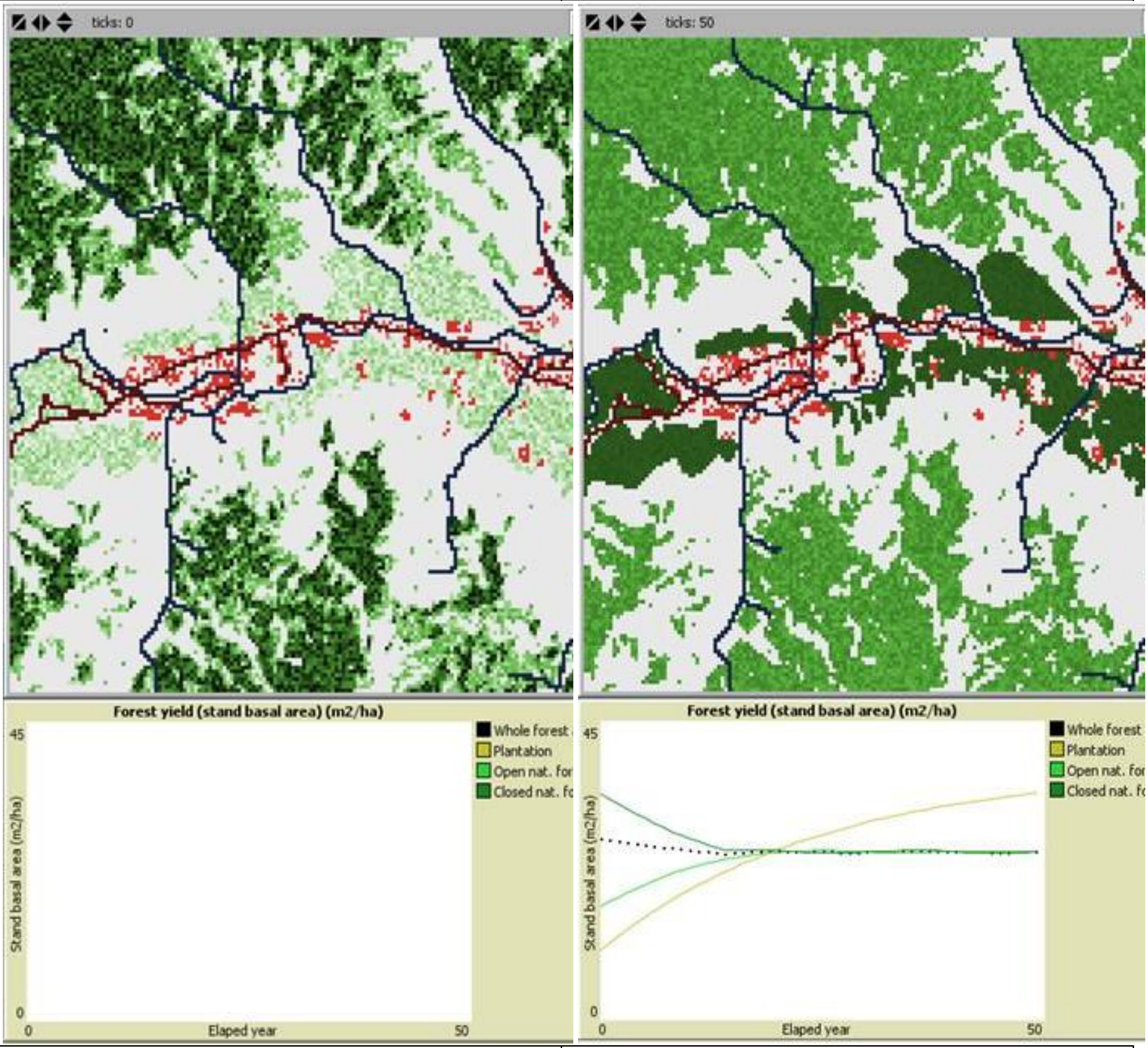
```
ask turtles
   [set h_income 0
    while [L-logging > 0 ]
          [let min-distance 0
           let nearest-forest-patch  min-one-of patches with [p_landuse != 5 and P_yieldforest > 28.56 and p_logged = 0] [distance myself]
           ifelse nearest-forest-patch != nobody [set min-distance distance nearest-forest-patch] [stop]
           let logged-patch one-of patches with [p_landuse != 5 and P_yieldforest > 28.56 and distance myself >= min-distance and distance myself <= min-distance + 50 and p_logged = 0]
           if logged-patch = nobody [set logged-patch one-of patches with [p_landuse != 5 and P_yieldforest > 28.56 and p_logged = 0]]
           ; To log and remove a forest tree (of about 60- 90cm of dbh by axes), the household usually spends about "Labor-need-to-log-a-tree" +- 3 days
           let labor-spent Labor-need-to-log-a-tree + random 3
           if ([p_slope] of logged-patch > slope-threshold and random-float 1 > protection-power / 100) or      ([p_slope] of logged-patch <= slope-threshold)
              [ask logged-patch [set p_logged 1]
               set h_income h_income + labor-spent * (50 + random 10); income from logging activities in Hongha: 50,000 - 60,000 VND /day.
               ]
           set L-logging  L-logging - labor-spent ; substract the labor sub-budget L-logging by the lalor quantum spent
          ]
    ]

End
;*********************************************************
```

Don't forget to *Save* your model as: landuse_change4.nlogo

With the model **landuse_change4.nlogo**, in the *Interface* tab please set:
"`Labor-need-to-log-a-tree`" = 20 (to increase the speed of simulation)
"`Protection-Power`" = 0 (no protection)
"`Stop-when`" = 50 years

Run "`To Simulate`"

The results are shown in Figure 1.



(a) Initial landscape	(b) Simulated landscape after 50 years and forest yield dynamics

Figure 1. Initial landscape (a) and simulated landscape after 50 years and forest yield dynamics (b).

Your homework

- Spend 2-3 hours to go through this tutorial.
- Include a more parameter of forest protection zoning policy:

- Design a global variable "`Slope-threshold`" in the *Interface* tab.
- In the Logging procedure, add/adjust code lines so that the procedure is satisfying the following protection regulation:

  If the slope of the forested patch is higher than "`Slope-threshold`", households have a (`1 – "Protection-power"`) chance to log trees there.

- Write Income-Update sub-procedure of the main `SIMULATION` procedure, which takes the prices of log products and input costs (logging wage, transportation). These price parameters should be set as sliders in the Interface tab so that they are adjustable by users.

- Write a sub-procedure `ForestLand-Update` of the main SIMULATION procedure, which updates changes in forest types and shows that change in the map window in the Interface tab.
- Within Plots-Update, write a sub-procedure Plotting-Income to plot annual household incomes driven from logging activities. The plot includes four income curves: average incomes of household type 1, 2, 3 and the overall average.

RESEARCH PROGRAM ON
Dryland Systems

The CGIAR Research Program on Dryland Systems aims to improve the lives of 1.6 billion people and mitigate land and resource degradation in 3 billion hectares covering the world's dry areas.

Dryland Systems engages in integrated agricultural systems research to address key socioeconomic and biophysical constraints that affect food security, equitable and sustainable land and natural resource management, and the livelihoods of poor and marginalized dryland communities. The program unifies eight CGIAR Centers and uses unique partnership platforms to bind together scientific research results with the skills and capacities of national agricultural research systems (NARS), advanced research institutes (ARIs), non-governmental and civil society organizations, the private sector, and other actors to test and develop practical innovative solutions for rural dryland communities.

The program is led by the International Center for Agricultural Research in the Dry Areas (ICARDA), a member of the CGIAR Consortium. CGIAR is a global agriculture research partnership for a food secure future.

For more information, please visit

drylandsystems.cgiar.org

Led by:

In partnership with: