# Scientific Modeling Exercises in NetLogo

**For these exercizes, please download the free agent-based modeling platform Netlogo (http://ccl.northwestern.edu/netlogo/). There are also several more examples available at www.modelingcommons.org.**

---

## Lesson One: Hello World

### Files needed: 01HelloWorld.nlogo

### Concepts Demonstrated:

- Format of procedure
- Linking of button to procedure
- Printing to Command Center using output-print
- comments

**Homework:** Do an Internet search on "NetLogo manual" to find all NetLogo defined procedures and what they do in the NetLogo Dictionary. NetLogo defined procedures are organized by type first and then alphabetically.
**Homework, part II:** Its always a great idea to read and understand every procedure in a model. **For this and every future lesson**, edit the code to watch behavior change when rerunning the model. Don't worry about creating bugs in the existing code. Creating bugs is just as important as making new behavior: fixing these experimentation bugs will make it easier to debug NetLogo code later.

## Lesson Two: Hello Modeling World

### Files needed: 02HelloWorld.nlogo

### New Concepts Demonstrated:

- Global variables
- Setup and Go procedure simulation modeling format
- Mixing text with variable values using output-type and output-print methods
- Variable incrementing over time steps
- Cyclic variable concept (Representing a variable that cycles with a sin function between a minimum value and maximum value over a specific number of time steps)

**Homework:** Read about and understand every procedure in the model. Edit the code to watch behavior change when rerunning the model.

# Lesson Three: Hello World

### Files needed: 03HelloWorld.nlogo

### New Concepts Demonstrated:

- Monitoring variables in a monitor box

**Homework Assignment:** Create additional monitor boxes for the other global variables.
**Hint:** Right click in the interface, choosing "Monitor" where you want place a new monitor box.

# Lesson Four: Hello World

### Files needed: 04HelloWorld.nlogo

### New Concepts Demonstrated:

- Using an output area to display output-type and output-print

**Homework Assignment:** Have the output area display information for the other global variables and averageSolarInsolation.

# Lesson Five: Hello World

### Files needed: 05HelloWorld.nlogo

### New Concepts Demonstrated:

- Creating text in Interface to ease user understanding
- Plotting a variable
- One dimensional random walk variable using random method (random method returns a random value flat distribution)
- Saving results in a output area to a text file. An example output file is located at outputData/05HelloWorld output.txt. (When the .txt is changed to .csv, Excel and other programs will read the results as tables.)

**Homework:** Edit the output commands so that the .csv contains additional information. (Don't forget to modify the header string so that the .csv file will still present the information in a tabular format.)
**Hint:** You might find it easier to open a copy of the .nlogo file is a text editor to view the code while watching the model operate.

# Lesson Six: Random Cycle

### Files needed: 06RandomCycle.nlogo

### New Concepts Demonstrated:

- Random Seed, or not.
- random-normal (random method that returns a value with a normal distribution around a mean value)
- Stochastic Cyclical Variable: Variable is defined by minimum mean value, maximum mean value, day of year where maximum mean day is applied, standard deviation of Gaussian distribution function, and day-to-day temporal autocorrelative proportion.
- Procedure for converting a day number to text (1 is January 1, 365 is December 31). Over time, accounts for leap year.

**Homework Assignment:** Think of an aspatial cyclic variable that you may need in the research you do. Determine the necessary parameters for the variable and implement that variable in NetLogo.

# Lesson Seven: Patches

NetLogo patches are what a raster grids are to GIS: they define a grid of variable values across the study area. NetLogo's "ask patches []" command performs commands on a patch by patch basis.

### Files needed: 07Patches.nlogo

### New Concepts Demonstrated:

- Patches

**Homework:** Adjust the number/density of patches in the model. Then, open a patch monitor to watch a single patch's variables change while the model is running. Finally, modify the AdjustGrass procedure to have different results happening in the patches.

# Lesson Eight: Patches, part II

### Files needed: 08Patches.nlogo

### New Concepts Demonstrated:

- Multiple patch interaction. In this system grass and weeds compete for the available space in each patch cell.

**Homework Assignment:** Identify a problem requiring multiple themes distributed across the study area as patches. Modify the math to account for interaction between the themes. Predator/prey models where both predators and prey exist in large numbers in each patch cell would be ideal solution, but feel free to use your own ideas.

# Lesson Nine: Objects

### Files needed: 09Objects.nlogo

### New Concepts Demonstrated:

- Objects as turtles.
- Turtles store energy.
- Turtles get energy from patches.
- Turtles reproduce with enough energy.
- Turtles die if they run out of energy.
- Turtles move if at location where energy decreases.

**Homework:** Modify the code logic such that turtles' life cycle will reduce or eliminate the possibility of complete die off. (It is "cheating" that the code currently has a conditional that when there are no turtles left, create a new turtle in the middle of the map.)

# Lesson Ten: Objects

### Files needed: 10Objects.nlogo

### New Concepts Demonstrated:

- Variable that is stochastic around an average value.

**Homework Assignment:** Incorporate at least two random cyclic variables into model in place of the existing turtleEnvironmentalEnergyNeeds variable and the turtle speed constant value of 1:
1) Replace the constant '1' in the line "fd 1 ;; turtles move forward one step if hungry" with a variable that cycles annually. (Turtles more more rapidly in the summer than in the winter for example.)
2) Modify the variable turleEnvironmentalEnergyNeeds so that it cycles annually with stochastic behavior. (Turtles need more energy in the summer than in the winter for example.)
NetLogo code for implementing random cyclic variables exists within exercise 06RandomCycle.nlogo. Also, implement a slider bar that adjusts the temporal autocorrelative nature of the turleEnvironmentalEnergyNeeds variable.

# Lesson Eleven: Reading from CSV Files

### Files needed: 11ReadFromCSV.nlogo, and 11turtleCSVfile.csv

### New Concepts Demonstrated:

- File management methods
- Various string methods
- Reading data from comma separated file and placing them into variables.
- GIS extension: coordinate transformation from real world coordinates to NetLogo map coordinates.

**Homework:** Make the turtles do something. They should use the information in their turtle variables to modify their behavior or abilities.

# Lesson Twelve: Calculate Arrays

### Files needed: 12calcArrays.nlogo

### New Concepts Demonstrated:

- Use of array extension to store variables of "general purpose" procedure.

**Homework:** Make the turtles do something else. They should use the information in their turtle variables to modify their behavior or abilities. Demonstrate the behavior in the appropriate monitoring widgets.

# Lesson Thirteen: Multiple Procedure Files

So far, all of the lessons' models were in a single file, the .nlogo file. However, this style would make projects with multiple coders difficult, if not impossible, to coordinate. If you were able to break up your project into multiple files, you would able to have each person writing code on their own files, reducing the chance that people will wreak each others code while working at the same time. Also, should you write a procedure that you expect to use in other models, you could separate out the logic for that procedure into a different file in order to modularize it. The lesson demonstrates that ability. The CalculateAXplusConstant procedure from the previous lesson is a handy tool because it performs the calculation necessary to realize dependent variable values from a multiple linear regression analysis. (So, should you model independent variables based on some research, it will be easy to demonstrate the cause/effect conclusions from research that performed a regression analysis.)

This model pretends you know how fast a person can run based on their age, gender, and income. (Imagine someone performed multiple linear regression analysis on the results of a race.)

### Files needed: 13MultipleProcedureFiles.nlogo, and CalculateAXplusConstant.nls

### Concepts Demonstrated:

- Separating code into different files

- NetLogo, for all intensive purposes, concatenates all .nls files to the .nlogo file. So it reasonably easy to divide up a .nlogo model into multiple files. The next lesson will walk the reader through the procedure necessary to do so safely.

**Homework Assignment:** Add an error realizer variable for each turtle in this model to mimic the error term of a multiple regression analysis. (Knowing an agent's gender, age, and income does NOT mean you know the speed at the turtle will travel: the error term will represent the variability of the dependent variable even we know the independent variables.)

## Lesson Fourteen: Multiple Procedure Files, part II

Click on the link above to see lesson 14.

## Lesson Fifteen: GIS Data Import

This lesson covers the import of the two main forms of GIS data most similar to data structures within NetLogo: patches and objects. Raster maps, in the form of ESRI ascii grids are conceptually related to patches and point shape files are related to turtles.

**Files needed: 15GISdata.nlogo, SetupTurtlesFromShapeFile.nls, 15data.shx, 15data.shp, 15data.dbf, 15data.sbx, 15data.sbn, and TestDEM.asc.**

**New Concepts Demonstrated:**

- Reading a raster dataset.
- Reading a point vector dataset.
- Capturing vector dataset attributes into NetLogo variables.

**Homework Assignment:** Use or create a shapefile that contains information and have the attributes from that shapefile placed in a turtle.

## Lesson Sixteen: GIS Vector Data

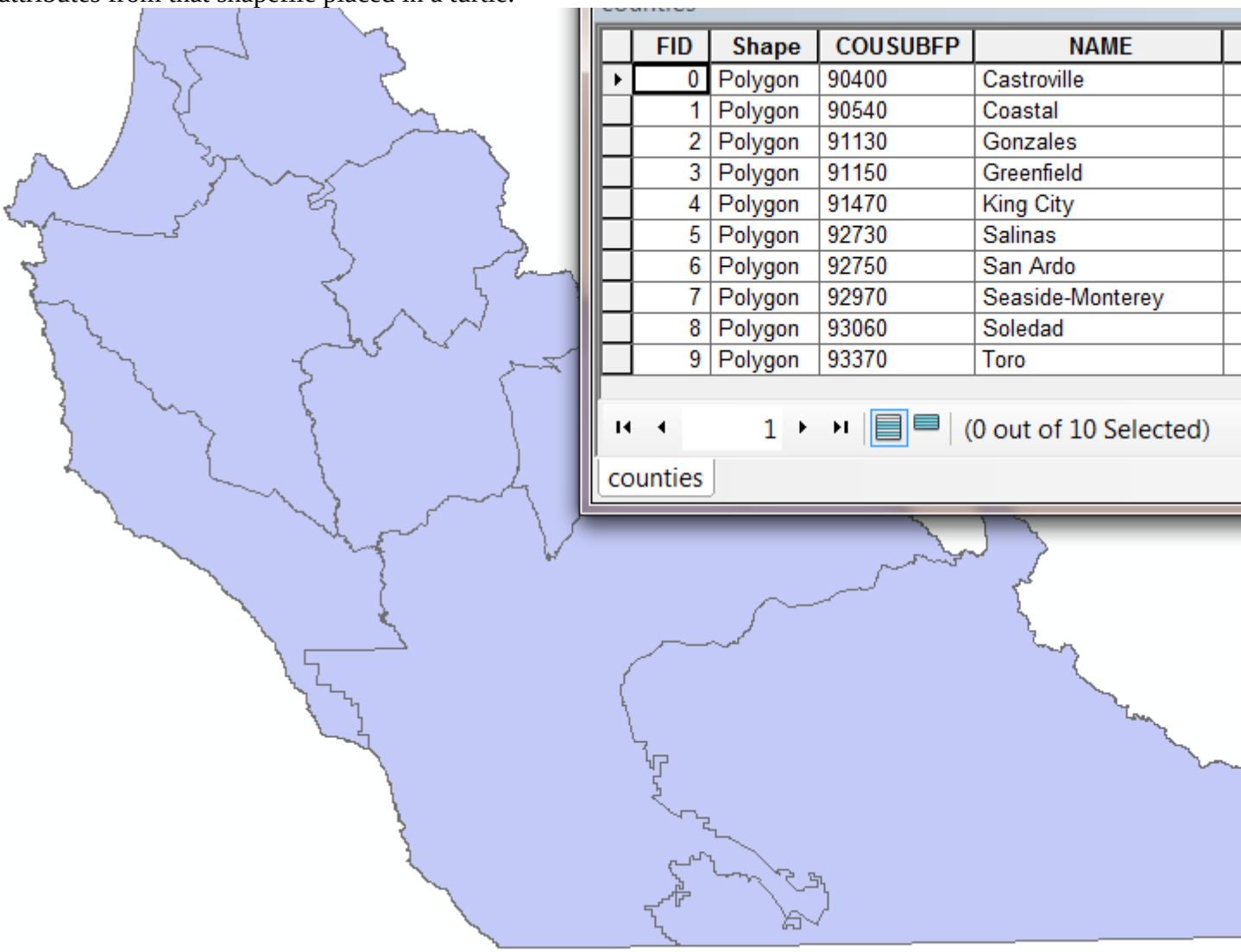
This lesson demonstrates the options you must choose when importing attribute information from within polygon vector shapefiles. In short, if the attribute you want in patches has a interval or ratio level of measurement, http://en.wikipedia.org/wiki/Level_of_measurement, the attribute column in the shapefile should contain integers or floating point values. If you want the resulting patch values to be exactly the values in the attribute column without any averaging, have the values be strings in the columns, converting the strings to numbers within NetLogo. In the lesson's shapefile, the id column contains numbers while the idText column contains strings.

**Files needed: 16GISdataVector.nlogo, counties.shp, counties.dbf, counties.prj, counties.sbn, counties.sbx, counties.shp.xml, and counties.shx.**

## New Concepts Demonstrated:

- Reading a polygon vector dataset.

**Homework Assignment:** Use or create a shapefile that contains information and have the attributes from that shapefile placed in a turtle.



| FID | Shape | COUSUBFP | NAME |
|---|---|---|---|
| 0 | Polygon | 90400 | Castroville |
| 1 | Polygon | 90540 | Coastal |
| 2 | Polygon | 91130 | Gonzales |
| 3 | Polygon | 91150 | Greenfield |
| 4 | Polygon | 91470 | King City |
| 5 | Polygon | 92730 | Salinas |
| 6 | Polygon | 92750 | San Ardo |
| 7 | Polygon | 92970 | Seaside-Monterey |
| 8 | Polygon | 93060 | Soledad |
| 9 | Polygon | 93370 | Toro |