

From Animations To Science

EES 4760/5760

Agent-Based and Individual-Based Computational Modeling

Jonathan Gilligan

Class #6: Tuesday, September 14 2021

Download files

Download files for Butterfly model

Download the following files from the Download page on the class web site:

- A single ZIP file with all the files in it:
https://ees4760.jgilligan.org/models/class_06/class_06.zip
 - Unzip after downloading.
- Or individual files listed on Download page
https://ees4760.jgilligan.org/downloads/butterfly_science/
- Start NetLogo and load [butterfly_class_06a.nlogo](#)

Projects

Individual Semester Project

- Semester Project:
 - Pick a model from an open-source repository ([CoMSES OpenABM](#)), or NetLogo “model library” that you want to work with.
 - Next Tuesday (Sept. 21) we’ll discuss choosing a project model in class.
 - Tue. Sept. 28:
 - One-page description of model and thoughts for extending it
 - Tue. Oct. 12: Examine ODD and code.
 - Short write-up of how model works and output from running it
 - Fri. Oct. 29: ODD for extending model
 - Tue–Thu. Dec. 7–9: Presentations on experiments with extended models
 - Fri. Dec. 10: Write-up of research project (around 10 pages)

Team Project

- Team Project:
 - Each team (2–3 students) will code a model from an ODD in the textbook:
 - The *Business Investor Model* (Ch. 10) studies the *adaptive behavior* design concept
 - The *Telemarketer Model* (Ch. 13) studies the *interaction* design concept.
 - By Friday Sept. 17, complete the survey on Brightspace to indicate your first choice model.
 - **Optionally:** If you and 1–2 classmates want to work together as a team say who is on your team.
 - Use model to do exercises from book
 - Make presentation about what you learned (Tues. Oct. 19).
 - Short written report Fri. Oct. 22.
- Detailed Assignments on [course web site](#).

Behaviorspace

Running Experiments: BehaviorSpace

- Vary any parameter that has a control on the model's interface
- Writes output to `.csv` spreadsheet file (table output is the most useful).
- Note: Data written in spreadsheet might be out of order.

```
"BehaviorSpace results (NetLogo 6.2.0)"  
"enhanced_butterfly_model_class_5.nlogo"  
"vary-q-final-only"  
"09/08/2021 23:22:51:278 -0500"  
"min-pxcor", "max-pxcor", "min-pycor", "max-pycor"  
"0", "149", "0", "149"  
"[run number]", "q", "[step]", "corridor-width"  
"7", "0", "999", "430.47369975615385"  
"18", "0", "999", "435.05661274391844"  
"20", "0", "999", "414.9412750152147"  
"6", "0", "999", "438.3850168325291"  
"3", "0", "999", "445.7111467510242"  
"1", "0", "999", "429.070105050936"  
"19", "0", "999", "420.9500983108795"  
"5", "0", "999", "433.5280318654752"
```


Analyzing Behaviorspace Output

- Behaviorspace output format is annoying
 - Each line is some tick of some run
 - How to organize, and average over runs?
- analyzeBehaviorspace app
 - https://ees4760.jgilligan.org/analyze_behaviorspace
 - Or install on your own computer using R
 - Instructions at <https://github.com/jonathan-g/analyzeBehaviorspace>
 - After installing:

```
library(analyzeBehaviorspace)  
launch_abs()
```

Emergence

Emergence

- A tricky concept.
- Joshua Epstein in *Growing Artificial Societies*: “stable macroscopic patterns arising from the local interaction of agents.”
- Epstein ten years later: “I have always been uncomfortable with the vagueness and occasional mysticism surrounding this word.”
- Epstein now prefers to talk about “*Generative Social Science*”

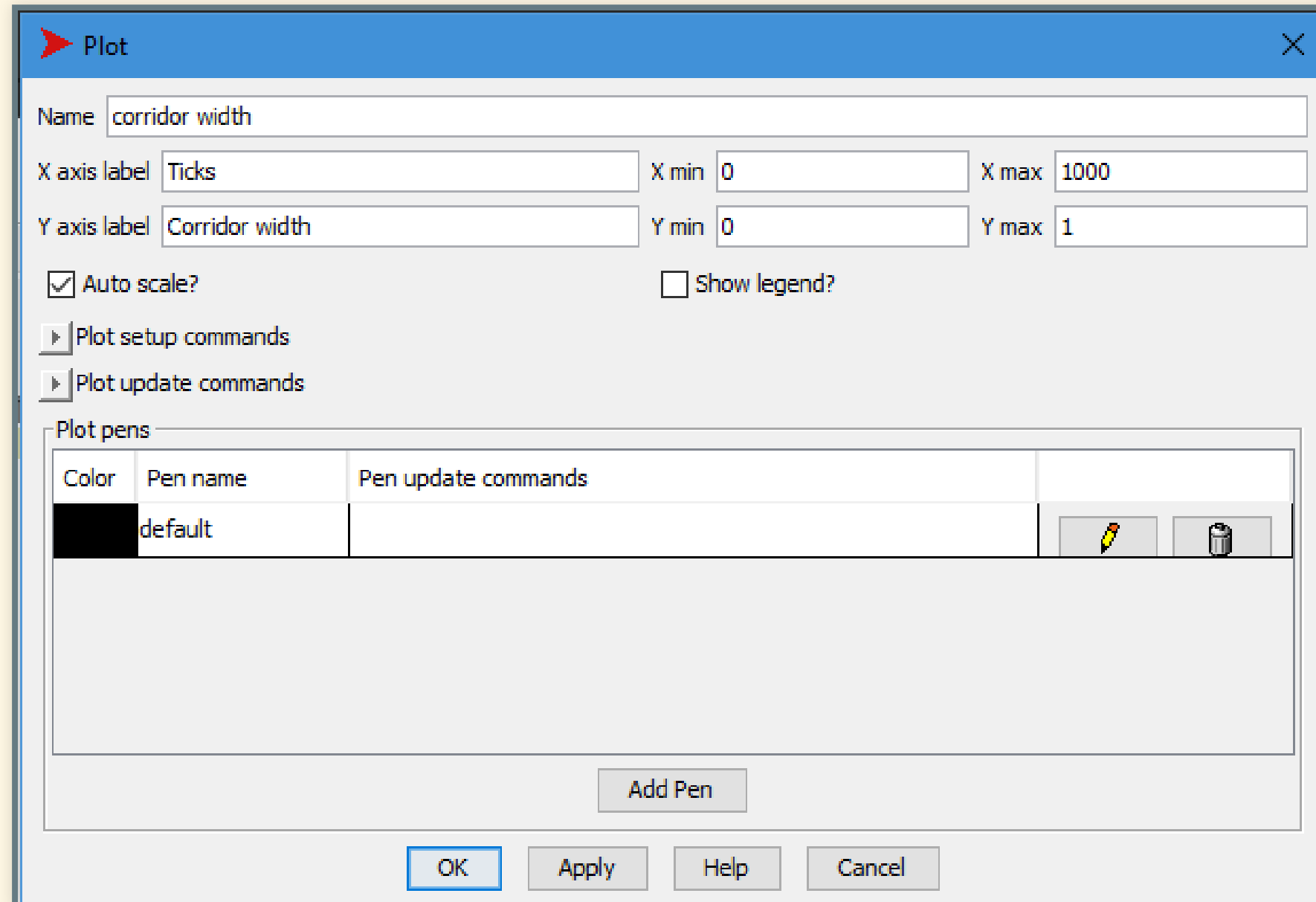
More Experiments with the Butterfly Model

More Experiments with the Butterfly Model

- You should have downloaded:
 - Various versions of NetLogo butterfly models.
 - A digital elevation map of real hills
https://ees4760.jgilligan.org/models/class_06/ElevationData.txt
- Start NetLogo and load `butterfly_class_06a.nlogo`



Plot Corridor Width

- On the interface tab, add a plot



The 'Plot' dialog box is shown with the following settings:

- Name: corridor width
- X axis label: Ticks, X min: 0, X max: 1000
- Y axis label: Corridor width, Y min: 0, Y max: 1
- ☒ Auto scale? ☐ Show legend?
-
-
- Plot pens table:

Color	Pen name	Pen update commands	
Black	default		 
-
- Buttons: OK, Apply, Help, Cancel

- On the code tab, add a line to [go](#) to plot the corridor width

```
plot corridor-width
```

Enhance Interface

- Add a button to export the plot to a file:

```
export-plot "Corridor-width" (word "corridor-output-for-q-" (precision q 2) ".csv")
```

- `precision q 2` rounds `q` off to two decimal places.
- `word` combines several different things into a single text string:
“corridor-output-for-q-0.40.csv”.
- Parentheses tell NetLogo which things `word` should apply to:

```
(word a b c d ... q)
```

will combine the values of variables `a`, `b`, `c`, `d`, ..., `q` into a single text string.

- Add a button to increment `q` by 0.1
- Add a switch `concentrate-turtles` and edit `to setup` to change the code for `crt 500` to include this:

```
ifelse concentrate-turtles
[
  setxy x0 + random 10 - 5 y0 + random 10 - 5
]
[
  setxy random-pxcor random-pycor
]
```

BehaviorSpace

- If your model is having problems, compare it to [butterfly_class_06b.nlogo](#)
- Open BehaviorSpace and create an experiment
 - Call it [experiment](#)
 - Vary [real-terrain](#) between [false](#) and [true](#)
 - Vary [q](#) from 0 to 1 in steps of 0.2
 - set [concentrate-turtles](#) to [true](#)
 - Run 20 repetitions for each value of [q](#).
 - Measure [corridor-width](#) and [mean-elevation](#) at the last tick only
 - Set time limit to 0 to let model run until it stops
- Run BehaviorSpace experiment
 - Save “table” output
 - Speed things up by unchecking “Update view” and “Update plots and monitors”
- Open the analyzeBehaviorspace app at https://ees4760.jgilligan.org/analyze_behaviorspace and use it to compare the relationship between corridor width and [q](#) for each terrain

BehaviorSpace

Experiment

Experiment name

Vary variables as follows (note brackets and quotation marks):

["q" [0 0.2 1]]
["real-terrain" false true]
["concentrate-turtles" true]

Either list values to use, for example:
["my-slider" 1 2 7 8]
or specify start, increment, and end, for example:
["my-slider" [0 1 10]] (note additional brackets)
to go from 0, 1 at a time, to 10.
You may also vary max-pxcor, min-pxcor, max-pycor, min-pycor, random-seed.

Repetitions

run each combination this many times

☒ Run combinations in sequential order

For example, having ["var" 1 2 3] with 2 repetitions, the experiments' "var" values will be:
sequential order: 1, 1, 2, 2, 3, 3
alternating order: 1, 2, 3, 1, 2, 3

Measure runs using these reporters:

corridor-width
mean [elevation] of turtles

one reporter per line; you may not split a reporter
across multiple lines

☐ Measure runs at every step

if unchecked, runs are measured only when they are over

Setup commands:

setup

Go commands:

go

Stop condition:
the run stops if this reporter becomes true

Final commands:
run at the end of each run

Time limit

stop after this many steps (0 = no limit)

OK

Cancel

Practice

Practice

- Work together with a partner
- Add a button to erase the tracks of the turtles (Exercise 5.2)
- Using the realistic terrain, play with q and see what values do best at helping butterflies find mates near hilltops.

Testing

Leaving trails

Turtles leave trails with the pen (because you told them `pen-down` when you created them in `to-setup`). The pens are the same color as the turtles, so it's hard to tell the difference between turtles and trails.

It would be nice to color patches they visited yellow so you could erase the pen trails and see the red turtles contrasting with the yellow trails of visited patches.

- Open the original `"butterfly_class_06a.nlogo"`
 - Add code to color patches yellow when a turtle visits them:
 - At the end of `to move`, add
- ```
set pcolor yellow
```
- Add a button to the interface to erase the pen trails:
    - Give the button the command `clear-drawing` and the display name "erase trails"

# Testing Models

- Using monitors
- Testing for consistency
  - Open the original `butterfly_class_06c.nlogo`
  - In `to go`, after the turtles move add:

```
if (count patches with [visited?]) != (count patches with [pcolor =
yellow])
[print "# visited patches does not match # yellow patches."]
```

# NetLogo output

- Four commands to output to *Command Center*:
  - `show` Indicates which turtle or patch used it. Ends the line after the command.
  - `print` Ends the line after the command.
  - `type` Does not end the line.
  - `write` Does not end the line. Text strings are quoted.
- Similar commands output to the *Output* area
  - `output-show`
  - `output-print`
  - `output-type`
  - `output-write`
- See the "**Output**" section of the **Programming Guide** in the *NetLogo User Manual*.

# More testing

- In `to move` replace the `ifelse` block with this:

```
ifelse random-float 1 < q
[
 ; move uphill
 let current-elevation elevation
 move-to max-one-of neighbors [elevation]
 if elevation < current-elevation
 [show "Turtle is moving downhill."]
]
[move-to one-of neighbors] ; move randomly
```

- Why does the turtle sometimes move downhill when it should be moving uphill?