HeatSpace 1.01 Beta

James Caton

2016-11-14

1 What is HeatSpace?

NetLogo is an agent-based modeling platform whose ease of use and functionality are hard to beat. The goal of this manual is to succinctly show how to generate visualizations that present siginficant portions of a model's parameter space. Those who are not accustomed to agent-based modeling may expect visualizations to come in the form of line plots. Line plots are an easy way to convey a limited amount of information to a user, but it is difficult to convey more than a few lines without confusing the observer. Instead of representing the value of an output spatially, heatmaps represent value using color.[1]

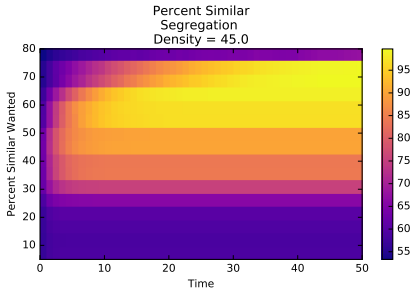

Figure 1: Percent Similar, x-axis: Time, y-axis: Percent Similar Wanted

When we visualize the parameter space, it is not enough to represent this space only using a single run for each combination of parameters. Each combination must be mapped using a substantial number of runs. Usually 20 runs may produce a large enough sample to faithfully represent the parameter space. More is better. If generating data from model of interest is not a high cost endeavor, use more runs.

[1] Juan Castilla Rho for first introduced me to the idea of using heatmaps to represent the BehaviorSpace

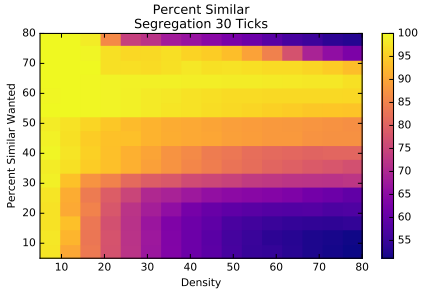

Figure 2: Percent Similar, x-axis: Density, y-axis: Percent Similar Wanted

HeatSpace averages the values generated at each time period during a run. These values can then mapped onto two axes whose values are fixed throughout a run (Figure 2). A single map can be generated for each time period of a run, creating series of frames analogous to a movie. Alterately, the x-axis of a heatmap can represent time (Figure 1) so that the change in the value of an output in light of a change in either the x or y parameter value at period t can be represented.

2 Setting Up Netlogo BehaviorSpace

Netlogo's BehaviorSpace is a powerful feature. It allow the user to test a set of exogenously set parameter values over a given space of periods and for some specified number of repetitions. The program offers the option to generate output in csv and table format automatically, however these are not easy to manage. Instead, we will create one csv of output for every run. These csvs will be comprised of columns. The columns do not have titles in the top row, but are in the order specified under the *prepare-behavior-space-output-method*.

```
globals [
  percent-similar  ;; on the average, what percent of a turtle's neighbors
                   ;; are the same color as that turtle?
                   ;; what percent of the turtles are unhappy?
  percent-unhappy
  similar-neighbors
  total-neighbors
  final-output
  csv-name
]
```

Figure 3: Idenitfy Globals

For this tutorial, we will be using the segregation model from NetLogo. You may name your version of the model however you would like, just remember

to use that name when we input the model name into the HeatSpace program.

First we want to make sure that we have correctly identified our global variables (Figure 3). Take note that I have changed the name of *%-similar-wanted* to *percent-similar-wanted* (Figures 4 and 6) and that *density* and *percent-similar-wanted* will be the variables that we will be altering for the run of BehaviorSpace. The *update-globals* method shown in Figure 5 should not require any changes. Be sure to add methods shown in Figures 6, 7, 8, and 9. I will leave it up to you to figure out how these are related if that interest you. If you copy the methods verbatim, the NetLogo will generate the files that HeatSpace will use to generate heatmaps. (Note: If you would like you can change the name from *0segregation*, just be sure to leave the *0* in front of the string you use and use your unique string when you input the root csv-name into HeatSpace).

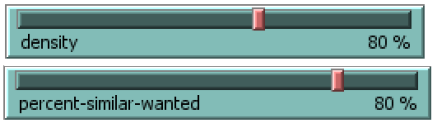

Figure 4: Sliders

```
to update-globals
  set similar-neighbors sum [ similar-nearby ] of turtles
  set total-neighbors sum [ total-nearby ] of turtles
  set percent-similar (similar-neighbors / total-neighbors) * 100
  set percent-unhappy (count turtles with [ not happy? ]) / (count turtles) * 100

end
```

Figure 5: Update Globals

Once you have copied these methods into the text of the nlogo file, you will be able to reference them in the *setup* and *go* methods. In the *setup* method, add *prep-csv-name* above *update-turtles* (Figure 10). In the *go* method, add *prepare-behavior-space-output* and *write-csv csv-name final-output* after *update-globals* (Figure 11). Also be sure to comment out the line that read *if all? turtles [happy?] [stop]*. This will allow the behavior space to run even after all turtles have stopped moving, and thereby ensure that the model generates output for the desired number of periods for every run.

You are now ready to run BehaviorSpace. In this tutorial, we will be changing the values of *density* and *percent-similar-desired*. We use arrays with the following values for each:

[5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80]

Set the number of repetitions to 20 and the time limit to 200. When you run the BehaviorSpace, you can uncheck the *spreadsheet ouput* and *table output* checkbuttons.

```
to prepare-behavior-space-output


  set final-output (list
  density
  percent-similar-wanted
  percent-similar
  percent-unhappy
  similar-neighbors
  total-neighbors

    )
end
```

Figure 6: Prepare Behavior Space Output

```
to prep-csv-name
  set csv-name "Osegregation.csv"
  set csv-name replace-item 0 csv-name (word behaviorspace-run-number)

end
```

Figure 7: Prep CSV Name

```
to write-csv [ #filename #items ]
  ;; #items is a list of the data (or headers!) to write.
  if is-list? #items and not empty? #items
  [ file-open #filename
    ;; quote non-numeric items
    set #items map quote #items
    ;; print the items
    ;; if only one item, print it.
    ifelse length #items = 1 [ file-print first #items ]
    [file-print reduce [ (word ?1 "," ?2) ] #items]
    ;; close-up
    file-close
  ]
end
```

Figure 8: Write CSV

```
to-report quote [ #thing ]
  ifelse is-number? #thing
  [ report #thing ]
  [ report (word "\"" #thing "\"") ]
end
```

Figure 9: Quote

```
to setup
  clear-all
  ;; create turtles on random patches.
  ask patches [
    if random 100 < density [   ;; set the occupancy density
      sprout 1 [
        set color one-of [red green]
      ]
    ]
  ]
  prep-csv-name ;; prepare name of csv as  csv-name
  update-turtles
  update-globals
  reset-ticks
end
```

Figure 10: Setup Model

```
;; run the model for one tick
to go
;  if all? turtles [ happy? ] [ stop ]
  if ticks = max-ticks [stop]
  move-unhappy-turtles
  update-turtles
  update-globals
  prepare-behavior-space-output
  write-csv csv-name final-output  ;; write csv
                                   ;; with csv-name as filename and
                                   ;; final-output list as row for
                                   ;; each period, each column of output
                                   ;; at end of run holds dater for the respective
                                   ;; variables in the final-output list

  tick
end
```

Figure 11: Go

3 Running HeatSpace

Once the csvs are generated, you are ready to create your heatmaps. The program is coded in Python 3, so make sure that you have Python 3 installed

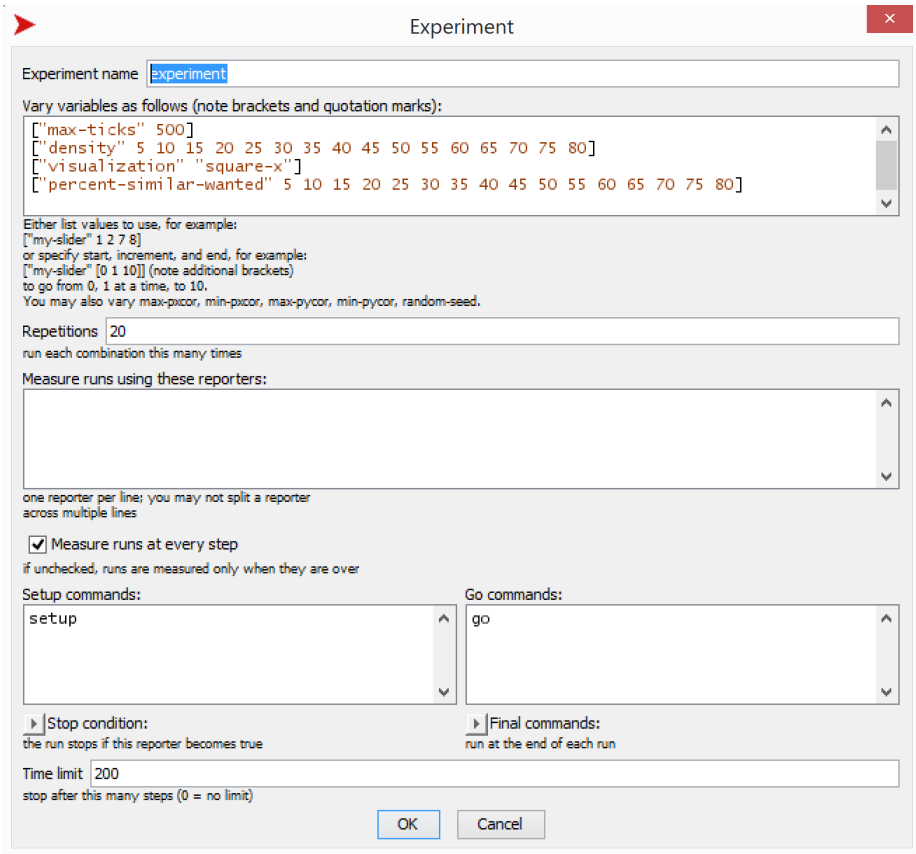

Figure 12: Experiment in BehaviorSpace

on your system.[2] Make sure that you have installed the numpy, pandas, matplotlib, and tkinter libraries. [3]

After you have made sure to install the appropriate libraries, you are ready to run the program. Run the *heatspace_gui.py* file either from the command line as or from the Spyder IDE. You will see the initial window (Figure 13). Click the "Choose NetLogo File" button and find the NetLogo file that you used to generate the data. The name of the file will appear in the button (Figure reffig:initial-gui). the number of variables that you wish to use for output in the body of each heatmap. When you have done this, click the *Gen-*

[2]Download Python 3 at the official python site. Alternately, you can use the Spyder IDE from Ancaconda (which I used for developing this program). Python: https://www.python.org/downloads/, Anaconda: https://www.continuum.io/downloads

[3]If you installed Python 3 without Anaconda, from the command line enter "pip install *library*", where library is the name of the library you would like to install. If you use Anaconda enter "conda install *library*".

erate Variable Names button. This button will expand the window and generate lists from which you will choose the axis and output variables that will be used in the heatmaps (Figure 15).

When we ran the BehaviorSpace, we generate 5120 files. Use this value for the number of files. You can choose different values than I have chosen for the next two boxes. In most cases the model reaches equilibrium early in the run, so I have chosen the value 30 for "Total Periods". (I recommend that you adjust these values and observe the different results. The root name of the csvs generated was "segregation". Under "Model and Run Description" I recommend that you enter the title of the model: "Segregation". We only used two variables in generating the BehaviorSpace, so only choose *Density* and *Percent Similar Wanted* as axis variables. Finally, check the "Include visualizations with time axis?" box so that you will have heatmaps whose x-axis represents time. Choose whichever output variables you would like. You may also generate new variables for output from the existing data using the right most section of the GUI.

You are ready to generate heatmaps. The time taken to do this will depend in part on the number of variables you have chosen to represent, the total number of periods represented, and the interval between each visualization. The visualizations will appear in the console as they are inserted in PDF files placed in the appropriately titled folders in the same directory as the csvs and py-files.

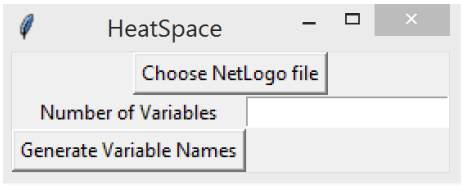

Figure 13: Initial Gui 1

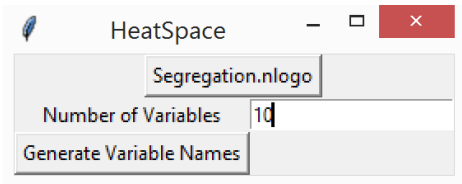

Figure 14: Initial Gui 2

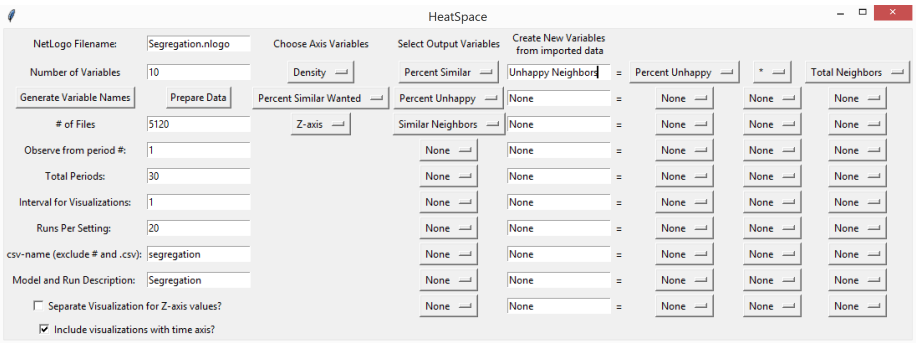

Figure 15: Expanded Gui