Experimenting with social science models in NetLogo: an introduction

Alex Stivala
Melbourne School of Psychological Sciences,
The University of Melbourne, Australia

SWEN40004 guest lecture
March 17, 2014
`http://munk.csse.unimelb.edu.au/~astivala/swen40004/`

Axelrod's (1997) model

- "Culture" is represented by a vector of F traits each with q possible values
- Cultural similarity between agents a and b can then be measured by number of matching traits
- Agents are placed on a lattice, only neighbours can interact
- At each step, pick a random agent and one if its neighbours
- With probability proportional to their cultural similarity, they interact:
- a random trait on one agent is changed to become equal to that trait on the other
- Keep going until convergence
- At which point all neighbouring agents have either identical or completely distinct culture vectors

What the Axelrod model is modelling about human social processes

- *homophily*: individuals prefer to interact with similar others
- *social influence*: interactions typically cause individuals to become more similar
- At the absorbing state, the lattice is split into cultural domains,
- agents within a domain all have the same culture, and agents on borders of domains have nothing in common and can no longer interact

Axelrod, Robert. The Dissemination of Culture: A Model with Local Convergence and Global Polarization. *Journal of Conflict Resolution* 41:203–226 (1997)

Absorbing states of the model

Two possibilities:

Monocultural All agents have the same culture vector.

Multicultural Any two agents in the same region have the same culture. Any two agents that are neighbors but in different regions have completely different cultures (no traits are the same).

Know your model (1)

- If developing a new model, results may be surprising:

  > *[The dissemination of culture model] is also the most surprising and the most controversial. It is the most surprising in that when I simulated a population with my model, my expectations were wrong as often as they were right, even though I was the one who designed the model. In addition, one of the key results was so counterintuitive that at first I thought it must have been due to a programming error. But it was not.*

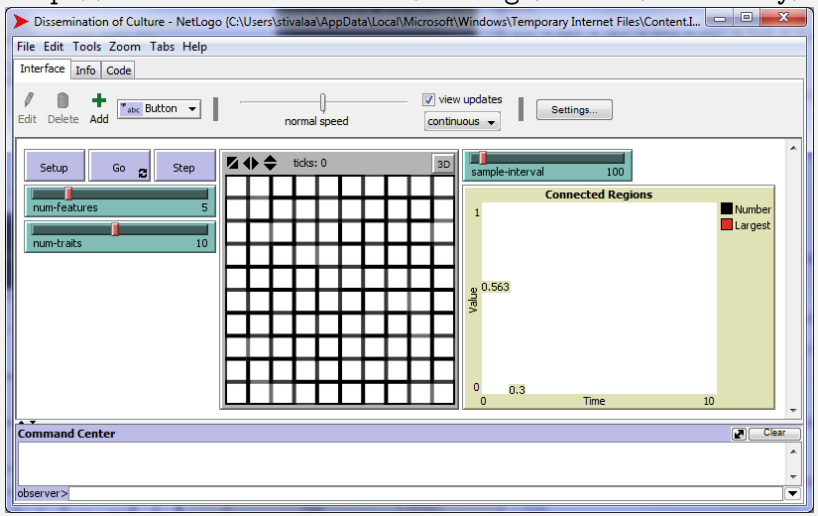
  (Axelrod *The Complexity of Cooperation* (1997) p. 146).
- Multiple independent implementations can help check for implementation errors.

Know your model (2)

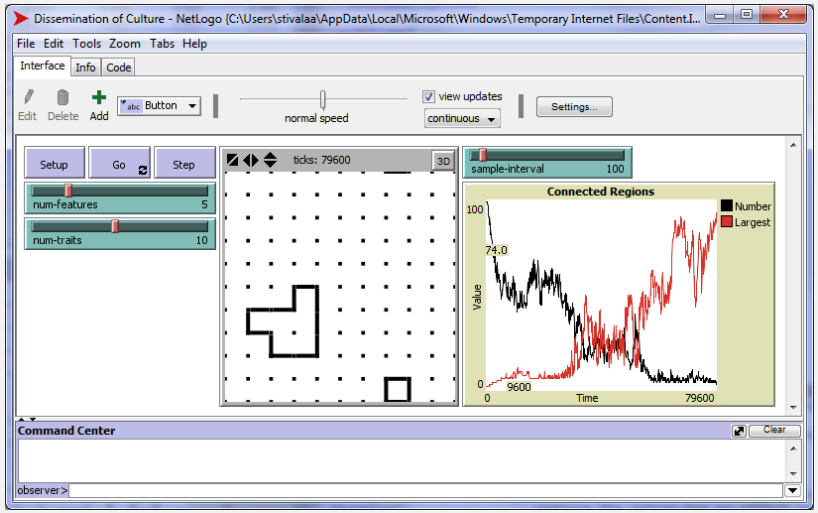
- The description of the model in (for example) an academic paper is where that knowledge is recorded.
- Although NetLogo is high-level enough it may be possible to understand the intended model from it, a NetLogo model is still a single particular implementation.
- If the model is implemented in a lower-level language such as Java or C, especially if not well-documented (and it never is), it will probably be impossible to understand the intended model from it.
- An implementation that doesn't crash is *not* the same thing as a correct implementation.
- Recommended reading:
  Conte, R., *et al.* Manifesto of computational social science. *Eur. Phys. J. Special Topics* 214:325–346 (2012)

The "Dissemination of Culture" Model

By Iain Weaver, available from the NetLogo "User Community Models" page
`http://ccl.northwestern.edu/netlogo/models/community/`



An absorbing state of the model



Adding a count of the number of unique cultures (1)

Add a reporter procedure to compute the `unique-culture-count`

```
to-report count-unique [ #list ]
  report length remove-duplicates #list
end

to-report unique-culture-count
  let all-cultures []
  ask patches [ set all-cultures fput culture
      all-cultures ]
  report count-unique all-cultures
end
```

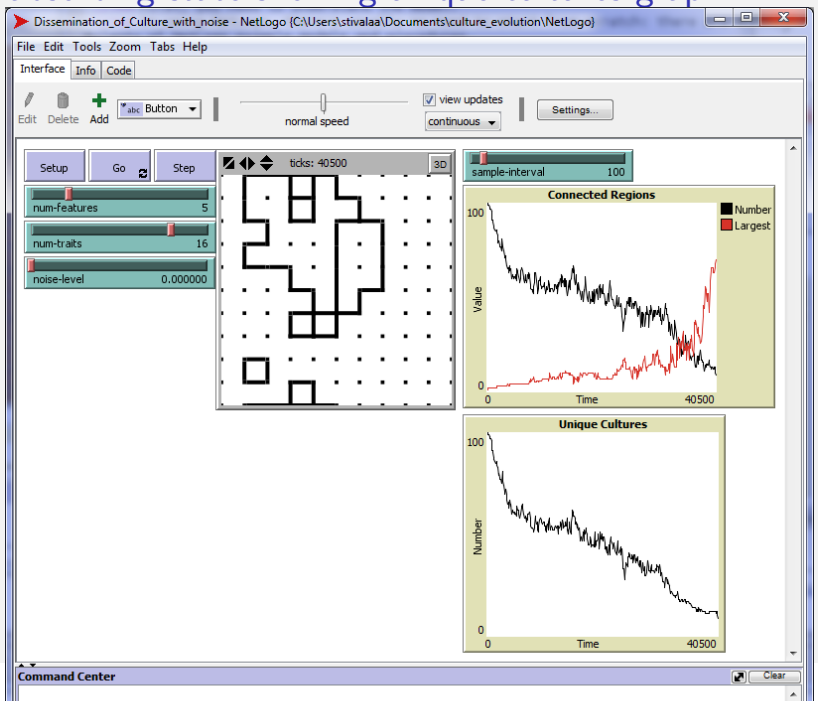
- Giving formal parameters a name starting with # is a convention sometimes used in NetLogo.
- `count-unique` is a generic reusable function.
- Shorter procedures are better: debug by using interactively in the Command Center and agent Monitors.

Adding a count of the number of unique cultures (2)

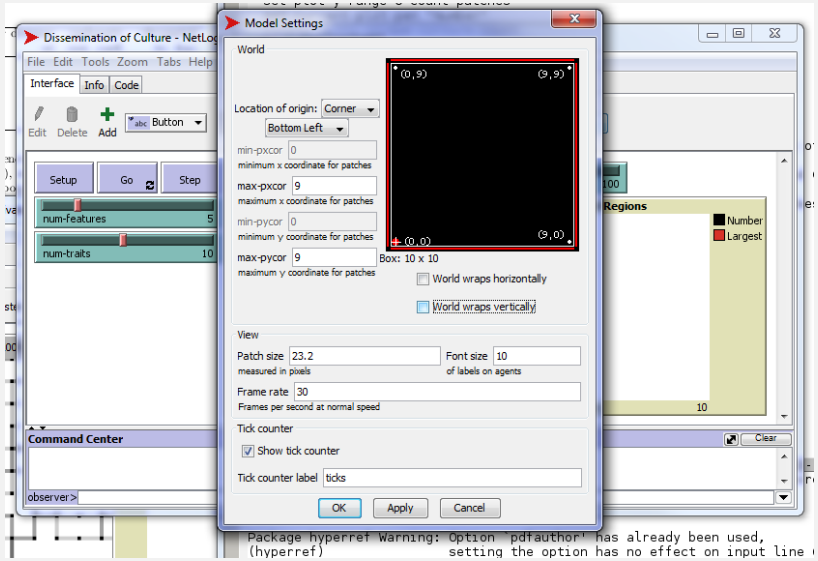
Add a plot element called "Unique Cultures" in the interface, and code to plot the `unique-culture-count` in it.

```
to update-plot

  ...

  set-current-plot "Unique␣Cultures"
  set-plot-x-range 0 ticks
  set-plot-y-range 0 count patches
  set-current-plot-pen "Number"
  plotxy ticks unique-culture-count
end
```

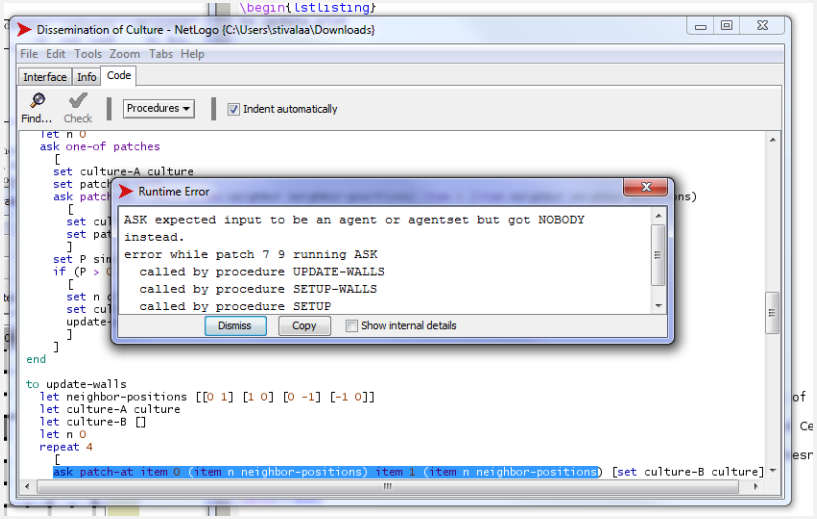

An absorbing state showing unique cultures graph



Change the world... from torus to finite lattice



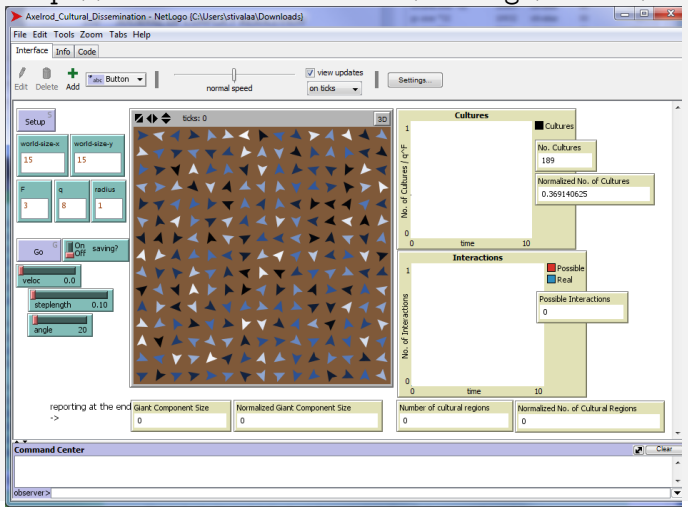
No, you can't change the world (just by clicking on things)



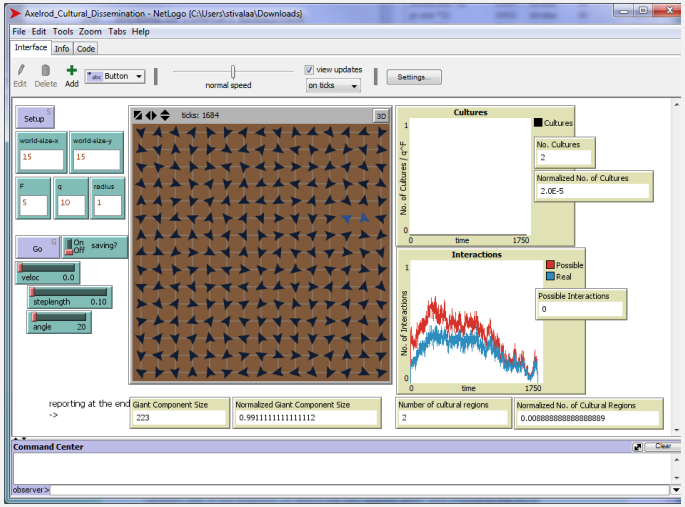
The code assumes that all patches have four neighbors. Changing the code to work when this is not true involves a major re-write.

But there is another version of model that does work on a finite lattice

By Arezky H. Rodriguez, the "Axelrod_Cultural_Dissemination" model, also available from `http://ccl.northwestern.edu/netlogo/models/community/`



An absorbing state of the second model

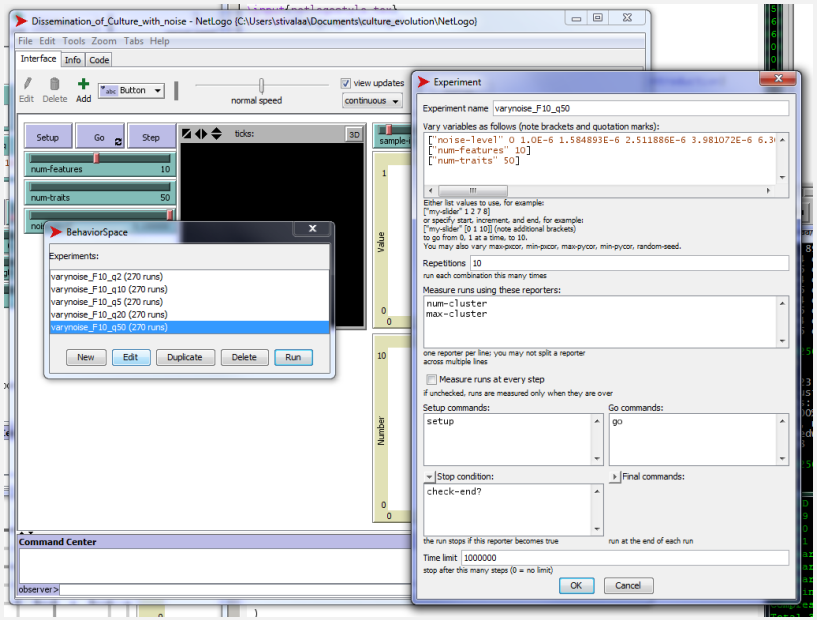


In this version, the agents are turtles, not patches. It also optionally allows them to move (although set to not do so by default).

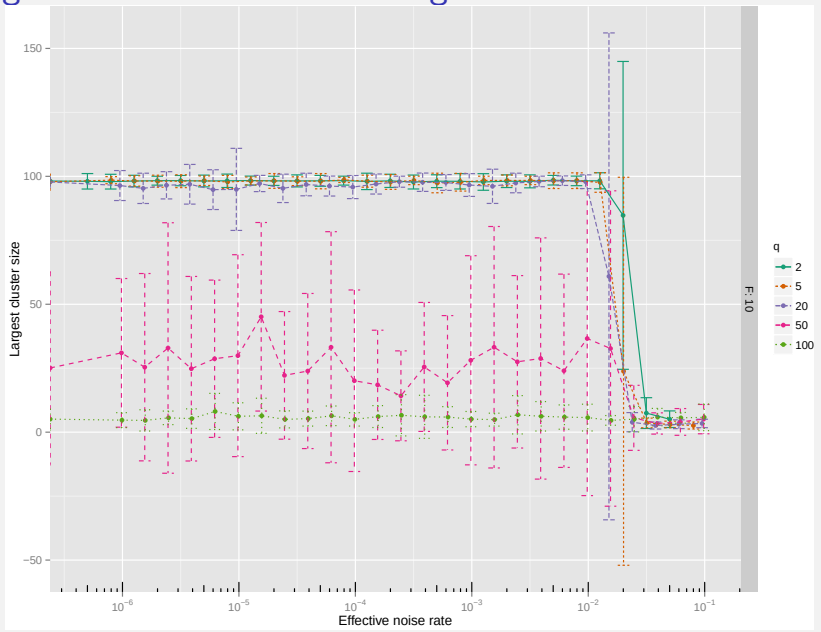
In the long run

- The NetLogo Interface and Command Center and agent Monitors are great for interactively building, debugging, and experimenting with models.
- But we are usually interested in the long-run statistical behavior of models, generally at an absorbing state. (See Axelrod (1997) for example).
- This will usually involve systematically varying parameters, making large numbers of runs, and taking statistics of properties at the end.
- Doing this manually with the interactive interface is not practical.
- The NetLogo "BehaviorSpace" tool can be used to do this, even running experiments in parallel.

BehaviorSpace



Largest cluster size at absorbing state as noise rate varies



Large experiments might need 1000s of hours of CPU time

So you will probably have to write "real code", not NetLogo.

SGI Altix XE cluster, NCI National Facility. `http://nf.nci.org.au/facilities/xe/xe_with_doors.jpg`

Supercomputers looked cooler in the 90s

Thinking Machines CM-5, photographer: Tom Trower, 1993.
`http://people.csail.mit.edu/bradley/cm5/AC93-0146-2_a.jpeg`

Summary

- First, you need to understand the model.
- In using NetLogo, there's usually no need to start from scratch: there are plenty of NetLogo example models and procedures.
- There's more than one way to do things. E.g. the NetLogo Community Models library has two Axelrod models, one implementing the agents as patches, and one as turtles.
- Keep procedures short and as simple as possible, use the Command Center and agent Monitors to test and debug.
- Just because you can interactively change the world settings, doesn't mean the code will work correctly with the new settings.

Thanks and further information

Acknowledgments Prof. Garry Robins, Prof. Yoshihisa Kashima, Dr Michael Kirley, Dr Nicole Ronald

NetLogo `http://ccl.northwestern.edu/netlogo/`

These slides `http://munk.csse.unimelb.edu.au/~astivala/swen40004/`

References:

- Axelrod, Robert. The Dissemination of Culture: A Model with Local Convergence and Global Polarization. *Journal of Conflict Resolution* 41:203–226 (1997) http://jcr.sagepub.com/content/41/2/203.short
- Conte, R., *et al.* Manifesto of computational social science. *Eur. Phys. J. Special Topics* 214:325–346 (2012) `http://link.springer.com/article/10.1140/epjst/e2012-01697-8`