Introduction

In this lab you will explore the problem of how agents can reach the maximum of some quantity. In the tutorial you will create a series of procedures so that agents reach a maximum by following the direction of the local gradient (ie the steepest slope). This is a common method that organisms use to move toward the source of some food, heat or light. After you complete the tutorial you will modify it to illustrate another method for reaching optimum values – namely the idea of evolution in a fitness landscape.

Complete the lab 3 tutorial in the handouts folder. Work through this tutorial carefully. As you learn a new command or reporter take the time to play around with it so that you fully understand how to use it and what it can do. Resist the urge to copy and paste code from the Tutorial to your program. As you type it yourself you will reinforce the logic of each command.

Extension to the Model

As the program is written the turtles are very successful at finding local maxima, but may not find the global maximum, unless the smoothness is set so high that there is only one maximum! At the end of the tutorial you are asked to think about ways to get turtles to find the global maximum. There are a couple strategies you can take here. One is simply to have the turtles jump to a random spot on the screen if they find themselves at a local maximum that is not near enough to the highest value. Modify your program to implement this strategy. A philosophical problem with this strategy from the point of view of the emergence of order based on local interactions is that the turtles may not have a reason to know they are not at the global maximum. The following guiding questions will allow you to investigate a more natural strategy, while learning some new NetLogo features. Save your work in a NetLogo file with naming convention LastName_Firsname_Lab3_nlogo.

Finding Maximum Fitness

Nowak talks about the evolutionary concept of a fitness landscape. The idea is that phenotypic variables such as pigment, or size form variables in a morphological landscape, whose elevation is "fitness". The fitter the organism the more likely it is to produce offspring. Offspring of organisms may have some mutation in one or more of the phenotypic variables, and thus find themselves in a different part of the landscape from their parent. Those that emerge with higher fitness are more likely to reproduce. Over time organisms evolve until most are at a peak of the landscape. This method of using evolution, rather than gradients, to find local maxima can be extremely efficient. It is also easy to implement an algorithm that can find global maxima.

1. Make the following changes to your interface:
   (a) Change the graphics window so that the max-pxcor and max-pycor are both 50 and the patch size is 4.
   (b) Change the smoothness slider so it has a maximum of 50.
   (c) Change the number slider so that it ranges between 0 and 2000 turtles
2. One problem with the way the previous model smoothes the landscape is that smoothing not only makes the landscape less rough, but it also diminishes the difference in elevation between the highest peak and deepest valley. The following changes to the setup-patches procedure solve this problem:
   (a) Replace all occurrences of `elevation` to `fitness`.

(b) After the command that finds the highest and lowest elevation, rescale the `fitness` variable so that the lowest value is 0 and highest value is 100. (Note: You cannot simply redefine `highest` and `lowest` – you have to rescale the `fitness` variable for all the patches by using the `ask patches` command and setting the `fitness` equal to a new value using a suitable rescaling.)
(c) After this command recolor the patches so that the color scale ranges between `0` and `100` rather than `lowest` to `highest`.

3. When creating turtles set their shape to be a circle instead of the default shape.
4. Make the following changes to the `go` procedure:
   (a) Delete references to movement – the turtles do not move in this program.
   (b) Call two new procedures: `birth` and `death`. You will write these procedures.
5. Write the `birth` procedure so that it has the following properties:
   (a) At each time step a turtle hatches one offspring with a probability proportional to its `fitness` (Any turtles that happen to be at zero `fitness` should not reproduce and any turtles that are at `fitness` 100 should always reproduce.)
   (b) Hatch new offspring at a distance `mutation` from their parent and have them face in a random direction. Define `mutation` with a slider ranging from 0 to 10.
6. Write the `death` procedure in a way that keeps the population size *roughly* constant. You can do this by asking each turtle to count how many turtles there are, and if there are more than the starting number ask that turtle to die.
7. Add a new plot and a new monitor to your interface, showing the average `fitness` of the turtles as it changes over time.
8. Now use your new model to investigate how the turtles find maxima. You should see that turtles find their way to the global maximum fairly often, but not always. Occasionally a few turtles in the cluster of turtles may hatch offspring that lower down a local maximum, but closer to a global maximum. Occasionally this will result in the cluster "hopping over" to the global max, but often the attempt to transition peaks fails.
   (a) Experiment with different landscapes. Does the smoothness of the landscape affect the average fitness, the number of clusters, and the chance of reaching a global maximum?
   (b) Experiment with the `mutation` slider. How does increasing `mutation` affect average fitness?
   (c) When turtles get trapped on a local maximum, you can often nudge them over to the global maximum by increasing `mutation` for a while and then decreasing it again. What evolutionary pressures that might increase mutation rates?
9. You may notice that the single cluster of turtles occasionally splits into two groups. This is the type of chance event that is behind the creation of new species). However, in the simulation, one of the clusters eventually dies out. This is an example of what is called genetic drift. Genetic drift is also evident at the beginning of the simulation when you see a mixture of colored turtles at a fitness peak, but ultimately only one color survives. A slight change to the model makes speciation more enduring. In the birth procedure have each turtle count any other turtles within a certain radius (perhaps about one mutation radius). Then, when checking if the turtle is fit enough to reproduce, reduce the fitness indicated by the patch it is on by that amount. We can think of this change implementing the idea that a quasi species has reduced fitness if the population has genomes that are too similar.
10. Occasionally disasters strike and change the fitness landscape (quasi species that were well adapted become un-adapted as the landscape changes. Add a button that implements natural disasters and changes the landscape. See how the mutation rate affects the time it takes for a quasi species to adapt.