Phyllotaxis

We discussed the phenomenon of phyllotaxis, which is the arrangement of plant components as they grow. The basic process is quite simple. A primordium is produced at the meristem and then moves a short distance away in the radial direction. Then another primordium is produced at the meristem at some angle relative to the previous one. Each new primordium is formed at this same angle relative to the previous one. In this lab you will use Netlogo to construct two different models of phyllotaxis. The first is one where the divergence angle is prescribed. The second model is an emergent one, where no angle is specified, but new primordia are produced in a location where there is most space.

Procedure

Start a new Netlogo file and save if using the naming convention:

`lastname_firstname_Computer_Lab_06_Phyllotaxis.nlogo`

Create a world with max-pxcor and max-pycor equal to 99 and patch size 3.
In both models we'll use two different types of turtles: meristem and primordia. In Netlogo you can do this by defining breeds. In this way you can give different instruction to different types of turtles. In the procedures tab type the following lines

```
breed [primordia]
breed [meristem]
```

Divergence Model:

In this model, start by creating one meristem at the center of the screen. Make a `setup-divergence` procedure in which you create a meristem (`create-meristem` 1) and set its `size` to 5, its `color` to `green`, and ts `shape` to 'circle'. Also, let's make it start by facing the top of the screen: `set` its `heading` to 0.

Now create a `go-divergence` procedure. In this procedure

- `ask` the `meristem` to `hatch` 1 primordia, (`hatch-primordia 1`)setting its its `size` to 1, its `color` to a color of your choice and its `shape` to a shape of your choice.
- `ask` the `meristem` to increase its `heading` by an amount equal to the `divergence-angle`. Define the `divergence-angle` with a slider as some number between 0 and 180 with increment 0.5.
- `ask` the `primordia` to move forward and amount equal to `step` – a slider you define on the interface ranging in values between 0.1 and 5 in increments of 0.1.
- `ask` the `primorida` to increase their `size` by a suitable factor so that by the time they reach the edge of the screen they have a size of about 15. You could make this with a `factor` variable defined on a slider, or you simply make the `size` proportional to how far the seed is from the center `patch 0 0`.
- An important final step is to remove primordia that are about to move off the edge of the screen. To do this, have primordia `die` if their distance to `patch` 0 0 is bigger than `max-pxcor - size`. Look up the primitive `distancexy` in the dictionary to see how to do this.

Put the `setup-divergence` and `go-divergence` buttons on the interface and try out your model. Test different angles to see if the continued fraction method yields the expected number of parastiches. Do you get a Fibonacci number of parastiches if the golden angle is chosen?

Emergent Model:
In this model we will not make reference to any divergence angle. Instead we will have the meristem decide where to hatch primordia based on where there is most available space

First create a `setup-emergent` procedure. Instead of one meristem at the center lets create 1000, and have them set their `color` to green, their `size` to 1 and have them all move forward by 5. This should create a circular ring – representing the outer cells of the meristem, where primordia are created. Now in the setup ask one of the meristem to create a primordium, setting its shape and color to something you like and have it move forward one step.

Now make a `go-emergent` procedure. In this procedure

- Ask the meristem cells to find the closest primordia to them. A command like this will do the job:

  ```
  set closest-primordia min-one-of primordia [distance myself]
  ```

  You'll need to add the variable `closest-primordia` as a `meristem-own` variable at the top of your procedures tab, just below where you define the breeds. Notice the syntax in the above statement. The `min-one-of` reporter finds primordia with the least value of whatever is in the command block. In this case, the command block is `[distance myself]` which reports the distance from each of the primordia to the meristem that is doing the checking.

- Now `ask` the `meristem` whose `closest-primordia` is furthest away to hatch a new primordia – giving it the same properties as the first primordia you made. Hint: To identify the correct primordia you'll want to use the `max-one-of` reporter, and the command block you'll want is [`distance closest-primordia`].
- Now ask the primordia to move, grow and die in exactly the same way as you did in the `go-divergence` procedure.

Now add the appropriate buttons on the interface and try this with different step sizes. You should see that if you start with step larger than 2 you get five parastiches. With step starting at less than this you may get 7 parastiches. If you start with step at around 1 it is hard to find well defined parastiches. This is perhaps not what you expected, but actually is in keeping with experiments. When a meristem first starts creating primordia they move apart relatively fast, but as more and more primordia are made the rate at which the move apart slows down. Try starting the simulation with step larger than 2, and then slowly reducing the step size from 2 down to about 0.5 (do it very gradually). You should see that now parastiches form at every step size – and that you get a Fibonnaci number of spirals. Ian Stewart discusses this point in chapter 6 of Life's other secret.

Enhancements

- Find a way to modify the procedure so that the `step` size starts at something above 2, when the number of primordia is small and then slowly diminishes to around 0.5 as the number of primordia grows. You'll need to find some way to make this happen slowly, otherwise all your primordia will stop.
- Write a parastiches procedure which creates a visual representation of parastiches by connecting all primordia with their nearest neighbor. First ask each primordia to find the other primordia it is closest to, and then use the `create-link-with` primitive to connect it to its closest primordia. This command joins the turtle with another turtle using a `link`. You can think of a `link` as being a special breed of turtles which connect to other turtles. Look in the netlogo manual for way to change the color and shape of links. You could add a button to create the parastiches, or you could add it to your `go` procedures. If you add it to your go procedures you will need to ask all previous links to die before you run the parastiches procedure, otherwise you'll end up with a maze of links, since the primordia which are closest change as the primordia move.
- Add petals at the outer edge of your phyllotaxis pattern. Make a new breed of turtle called petals. Then write a procedure that makes primordia which are about to reach the end of the screen hatch a petal and then die. In order for this to work right, I suggest making all primordia move forward by a step, check if their distance from patch 0 0 is large enough, then hatch a petal and die. Give the petals a suitable shape color and size. Then put a make-petals button on the interface. Run the go procedure for a while, and when you have a pattern you like, stop it and then press the make-petals button. You can export the view as an image using the file menu at the top.