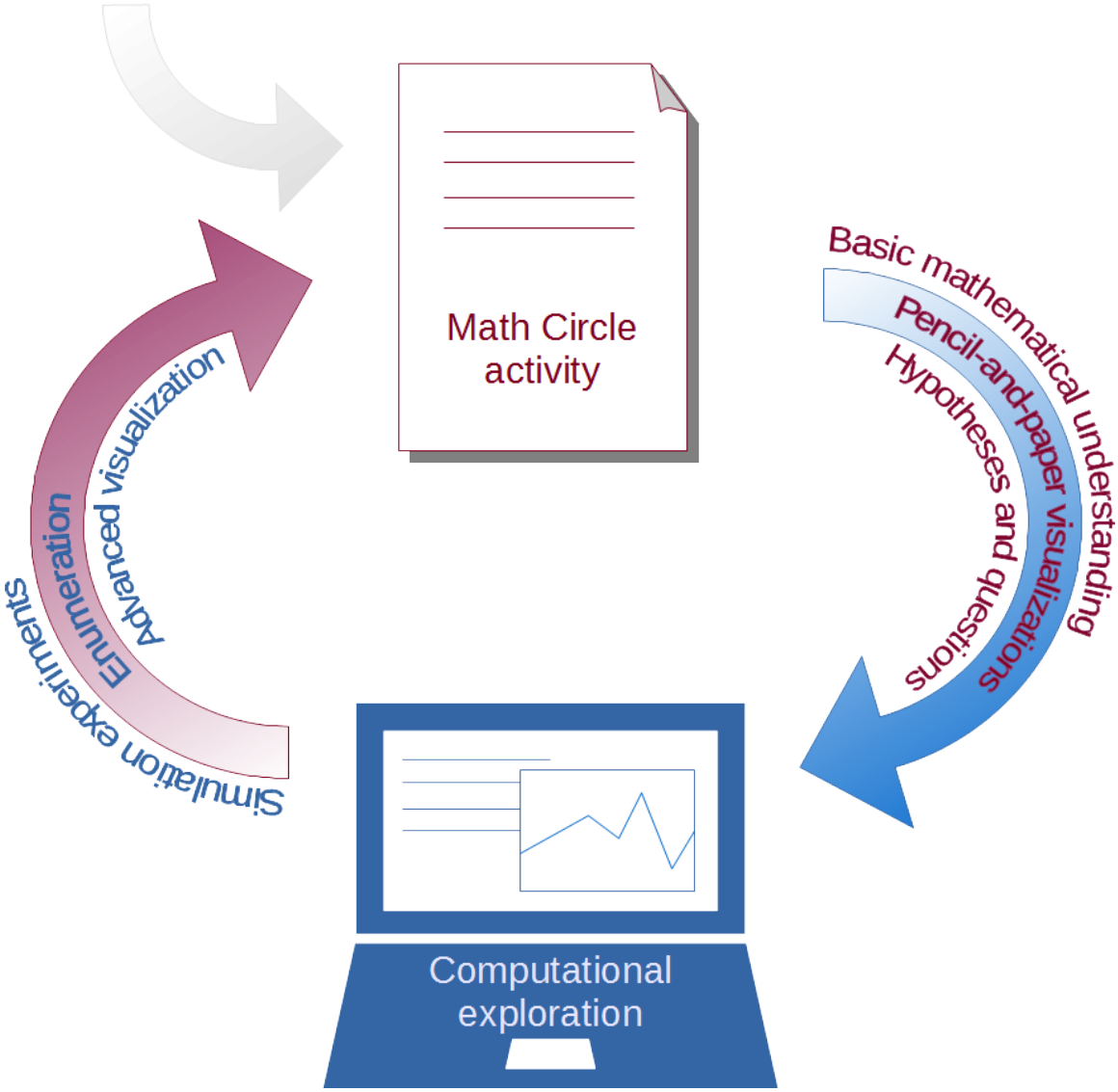
The Missing "M" in STEM:
A Math Circles & Modeling Approach



JMM 2017 Atlanta
James C Taylor & Nicholas Bennett
jtaylor@sfprep.org and nickbenn@g-r-c.com
Math Circles Collaborative of New Mexico
http://www.mathcirclesnm.org/missing-m

Investigating Folding Fractals: Enter the Dragon
Visualization and Algorithm Variations

Introduction

Starting with your Folding Fractal activity Step 6, you tried a new way of visualizing the outcome of our paper folding. By setting the paper strip on edge, then replicating that pattern on graph paper, the folds "jump" out of the one-dimensional world of lists and into a two-dimensional pattern. A problem: you can only fold the *physical* paper with its right angle turns so many times. The computer does not have these limitations! What if we "fold" the paper into a sequence of turns and steps (the straight sections of paper between the folds) performed by a virtual turtle on the computer?

There is more than one way to produce such a list of turns (the bends in your paper strip), generating our fractal "dragon" (you'll see what we mean). How can our insights gained in Steps 1 through 7 help us create such production rules?

These are the sorts of questions that – along with other factors – sometimes lead us to build and use computational models. In this case, we'll use NetLogo, an agent-based modeling and simulation tool, to construct a model of the turtle stepping and turning. (If you haven't used NetLogo before, your circle leader will introduce you to the basic elements necessary for this exploration.)

Getting started

In the turtle's world of NetLogo, the turtle (once created) can walk and turn with some basic commands. Once you have created your starting turtle in the center of the world and given it its initial characteristics such as direction, color, and size in your setup procedure, you must create the most important element of your model. This will be the list of turns to be made. This will be your core algorithm.

Turn list algorithm, #1

The NetLogo turtle (once created) can walk and turn with some basic commands, go *forward*, *right* and *left*. You must construct a list of those right and left turns as a function of the number of times folded.

You'll start with *pseudocode* (your circle leader will get you started). Here are your words or commands:

- **drop** *n* -Positive *n* drops from the left, and negative *n* from the right
- **not** *list* -in the turn *list***,** turn a left turn into a right turn or a right into a left
- **reverse** *list* -reverse or flip the *list* from left to right
- **concatenate** *list1 list2*
  -concatenate turn lists *list1* and *list2*

Step-by-step, take your original list of folds—starting with ∨ ∨ ∧ (or perhaps expressed in 0s and 1s, L or R)—and use these commands to take any list and build into the list the next set of folds. Compare your results with your lists of folds from the math circle activity to check your algorithm.

Your circle leader will now introduce you to NetLogo commands for list manipulation so that you can code your list folding procedure.

Turn list algorithm, #2

In your math circle handout, Step 7 suggests a different way of constructing your list of turns. Invent your own pseudocode for that process, express your algorithm in that pseudocode, and then implement it as a second list constructing procedure in NetLogo code.

Initial implementation

See if you can implement your conceptual model in NetLogo. Keep in mind that your implementation will need the following elements:

- An initialization procedure, which directly or indirectly (by invoking one or more other procedures) creates all needed agents.
- A procedure that tells the turtle to walk the entire list of paper folds, given the number of times you fold the paper.

You might find it helpful to keep the following in mind:

1. By convention, most NetLogo models use **setup** and **go** as the names of the main initialization and iteration procedures.
2. The **slider**, **switch**, **chooser**, and **input** user interface controls allow a user to interact with a model by graphically manipulating the value of global variables. (The configuration settings of each such control also defines a global variable, without having to include it in the **globals** block in the code.)
3. Your **go** procedure should only address the movement of your turtle. You should call a separate procedure which implements your turn list-generating algorithm.

Working with step length

As you folded your paper, what happened to the space (length of the paper) between the folds?

- How much did that space change from your folding of the paper from one time to the next?
- Why would you want your NetLogo model to reflect this?
- Can you replicate this adjustment in your model?