Code:

mdlGlobalVariables.m- This file contains all of the global variables that are used within the rest of the code. My suggestion is to run it first with a blank workspace.

mdlApplyRules.m- Takes in a start state, and applies the ternary logic rules defined by our group, and returns an image vector.

mdlFindTargetsProbabilistic.m- Takes in a start state and image vector, and finds the next step in discrete evolution towards image vector. Randomly chooses one of the options. Returns the vector at that time point.

mdlSiulatemProbabilistic.m-Takes in a path length and a start state, and simulates natural evolution on the start state for the length given. Returns a path, which is a series of time steps.

mdlTreatmentSimulation.m- This is used to simulate treatment. Takes in a start state, treatment vector, number of treatments, delay steps, and natural evolution steps. Returns a path.

mdlFigureGenerator.m-Looks for a path matrix called treatsim, and generates a split figure from it.

Things to note:

To start using, run mdlGlobalVariables, then generate your startstate which should be a 15x1 matrix if using HPA\_HPG model. You can then run mdlTreatmentSimulation, or just mdlSimulationProbabilstic.

To use the figure generator, set mdlTreatSimulation results to treatsim.

Treat Sim also takes in a matrix that defines the treatments; an (n X 2) matrix defines this. The first column represents a node index in the model, so 1 represents stress in our current model, the second column defines this a value of 1 denotes lower, and a value of two means increase. For example, to treat just stress, and make it high, you would make treatmentvector = [1,2]

\*\*Another note, the code written by Dr. Broderick has been modified to handle globals.