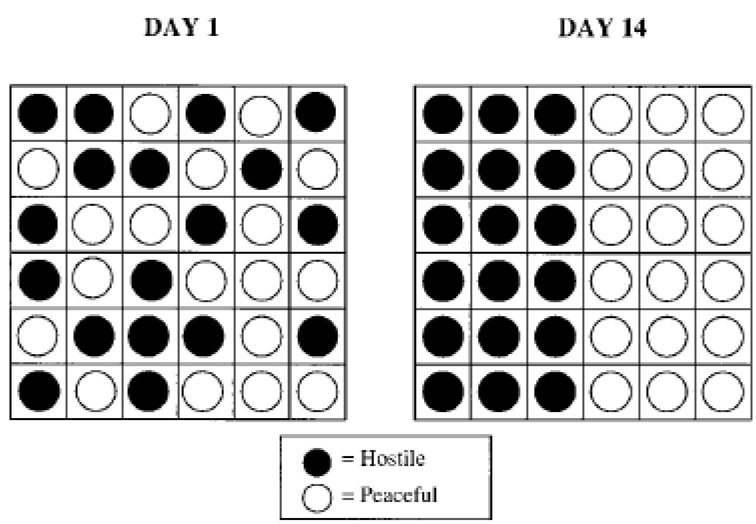
Final Project for PSYCH 20A Fall 2018
DUE December 11 at 5:00 PM

Submit four files:

makeNeighborhood.m
displayNeighborhood.m
evolveNeighborhood.m
KLB_fig5.m

In class, we went over a simulation from Kenrick, Li, and Butner (2003) in which we simulated neighborhoods full of hostile and peaceful individuals and observed how their behavioral changed over time following a simple update rule. The rule was called "majority rule" and it said: for each person in the neighborhood, look at their nearest neighbors (including diagonals) and count up how many hostile and peaceful neighbors they have. If they have more hostile than peaceful neighbors, then that person also become hostile. If they have a majority of peaceful neighbors, then they become peaceful. If they have an equal number of hostile and peaceful neighbors, then they don't change. Here is an example of what a neighborhood might look like on day 1 (random initialization of whether a person is hostile or peaceful) and on day 14 (i.e. after applying the rule multiple times):



After presenting this initial simulation, the authors discuss the effect of having some highly influential or low-threshold or "short-fuse" individuals in the neighborhood. These individuals only need to have *one* hostile neighbor to turn hostile (instead of a majority of neighbors). See page 13 of the paper. Importantly, note that it's possible for someone to be *both* peaceful and short-fuse or hostile and not short-fuse. Whether someone is short-fuse or not is *independent* of whether or not they are hostile or peaceful on any given day. Hostile/peaceful status can change from day-to-day, but whether someone is short-fuse or not does not change (think of it like a personality trait). Also, just because there is such an individual in the neighborhood (or more than one!) doesn't necessarily guarantee that the entire neighborhood is going to become hostile. In fact, it might have no effect and the entire neighborhood might become peaceful after some time. As in the simulation we did in class, the final result depends a lot on the starting conditions. However, the larger the initial proportion of short-fuse individuals, then the starting conditions start to matter less and less (see Figure 5 of their paper).

For this project, modify the code from class to include short-fuse individuals. I encourage you to first play around with the existing code. socialSimulation is the main bit of code that calls of the other functions. makeNeighborhood makes the initial neighborhood. evolveNeighborhood takes a neighborhood as input and updates it based once based on our evolution rules and outputs a new neighborhood. displayNeighborhood makes either a plot or an image of the neighborhood.

1) Modify makeNeighborhood to take another argument which corresponds to the number of short-fuse individuals and mark them in the neighborhood somehow.
2) Modify displayNeighborhood so that the short-fuse individuals are distinguished by having a red outline instead of a black one.
3) Modify evolveNeighborhood so that a different update rule (the short-fuse rule) is used for the short-fuse individuals.
   Hint: depending on how you solved part 1, computing the number of hostile individuals by adding up all of the numbers in a 3x3 region may no longer work and you may have to modify that bit of the code as well. Think carefully about what happens if you need change the short-fuse individual from hostile to peaceful or vice versa and how that might be represented.

Run your code a few times using different starting positions every time (change the seed in the socialSimulation script for example) and see how this affects the final distribution of people. You can play around with other aspects of the code and see what happens, like changing the size of the neighborhood.

4) Write a new script called KLB_fig5.m that will replicate the three plots from Figure 5. In order to do this, you will have to modify makeNeighborhood again to manipulate the initial proportion of hostile individuals, perhaps with a new argument. Display the plots as three subplots in a single figure window. Your script should call the above functions. The resulting figure won't be an exact replica of the one in the paper because there is some randomness involved. Run the simulation 10 times for each starting proportion of hostile individuals (you don't have to look at 1%, 2%, 3% hostile, etc.; can go every few % as in their figure). Plot the individual points for each run of the simulation for each percentage as they do in Figure 5.

See also Conway's Game of Life.