# Lauren's Examples / Explanations

Let's say Agent A is interacting with Agent B. You could make a function, which calculates the results of the interaction.

The function if you're calculating both sides of the interaction at the same time:

function greenInteraction(opinionA, uncertaintyA, opinionB, uncertaintyB):

some logic / calculations here determines what the new values are

return new\_opinionA, new\_uncertaintyA, new\_opinionB, new\_uncertaintyB

The function if you're only calculating how one agent changes:

function greenInteraction(opinionA, uncertaintyA, opinionB, uncertaintyB):

some logic / calculations here determines what the new values are

return new\_opinionA, new\_uncertaintyA

You could do things differently, for instance you might have a function that takes one Green agent and the opinions and uncertainties of all adjacent Green agents.

function greenInteraction(opinion, uncertainty, adj\_opinions, adj\_uncertainties):

some logic / calculations here determines what the new values are

return new\_opinion, new\_uncertainty

You might use the same function for seeing how Red or Blue agents interact with Green agents. (Remember that Red and Blue agents don't actually change their opinions and uncertainties. But also, you could handle Red and Blue agent interactions the same as Green interactions, where the Red and Blue agents *choose* an opinion and uncertainty value that best suits their cause. Also remember that Red's actions have a cost of potentially losing followers, and Blue's actions have a cost of losing energy.)

Or, you might have a completely different function for handling what the effect of the Red or Blue agent's message is:

function redInteraction(opinion, uncertainty, red\_message):

some logic / calculations here determines what the new values are

(ignore\_red is a boolean)

return new\_opinion, new\_uncertainty, ignore\_red

function blueInteraction(opinion, uncertainty, blue\_message):

some logic / calculations here determines what the new values are

(you could calculate the blue\_energy\_cost somewhere else if you prefer)

return new\_opinion, new\_uncertainty, blue\_energy\_cost

**You can use the same function for handling all kinds of interactions if you want. Or you can use separate functions. It is up to you.**

Also, remember that the more powerful the message that Red/Blue sends, the higher the cost should be. Like, there's a tradeoff between message power and cost. So, for Blue agents, more powerful messages have a higher energy cost. For Red agents, more powerful messages have a higher risk of having Green agents permanently ignore them. You can decide the rules for how this works, and you can add as much or as little randomness to this as you want. But you should make sure the different message power levels are useful, where the Red/Blue agents will choose a different power level depending on the situation.

## Other related information

* The Blue agents have the option of introducing a Grey agent **instead** of their normal move. The Grey agent has a random chance of being on the Blue side (Grey-Blue) or on the Red side (Grey-Red). The probability of one or the other isn't necessarily 50%, it's a parameter you provide at the start of the simulation. Once introduced, a Grey agent immediately makes a move and then leaves.
* Make sure the Red and Blue agents are using clever strategies, because they are trying to win the simulation. Ideally they should do this by learning from previous simulations.
* You should explain your project in your report.
* Normally the opinion variable is a boolean true/false value (for "Vote" and "Not Vote"), and the uncertainty is a floating point number that represents how uncertain a Green agent is.
  + If you are instead using just one floating point number to specify both the vote and uncertainty, that is ok, as long as you can get a boolean "Vote" or "Not Vote" value from it.
  + For example, if you're saying that values from -1 to 0 correspond to "Not Vote", and 0 to 1 correspond to "Vote", or similar, that is ok. You should do what makes sense to you.
  + Just make sure your Green agents do have opinions and uncertainties, because that is an important part of the project specification.

# Examples

(You can choose different rules to the ones I am using, this is just some ideas I had.)

**Example 1**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Green, Vote, 0.4

Agent A and Agent B interact.

Updated values:

Agent A: Green, Vote, 0.4

Agent B: Green, Vote, 0.3

The amount that uncertainty goes up/down, or stays the same, depends on what rules you come up with.

**Example 2**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Green, No Vote, 0.4

Agent A and Agent B interact.

Updated values:

Agent A: Green, Vote, 0.6

Agent B: Green, No Vote, 0.5

If the agents have different opinions, one or both of them could change their opinions, or they could keep the same opinions. You decide how this works.

**Example 3**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Green, No Vote, 0.4

Agent A and Agent B interact.

Updated values:

Agent A: Green, No Vote, 0.7

Agent B: Green, No Vote, 0.5

Remember, these are just random examples, you could do things a different way to me. Agent A's uncertainty could go up, or go down, or stay the same. Agent B's uncertainty could go up, or go down, or stay the same. You decide.

Also, you could make it so that there is a **probability** that an agent's opinion will change, given the circumstances. Or, you could make it so that the same circumstances **always** result in a change of opinion (and different circumstances always result in no change in opinion). The same goes with uncertainties. You decide.

**Example 4**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Green, No Vote, 0.4

Agent C: Green, No Vote, 0.1

Agent A, Agent B, and Agent C all interact.

Updated values:

Agent A: Green, No Vote, 0.5

Agent B: Green, Vote, 0.6

Agent C: Green, No Vote, -0.1

You can handle all interactions in pairs, or handle all interactions at the same time. You decide.

**Example 5**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Green, No Vote, 0.4

Agent A interacts with Agent B

Updated values:

Agent A: Green, No Vote, 0.7

Agent B interacts with Agent A

Updated values:

Agent B: Green, No Vote, 0.3

When two agents interact, you could handle the change in Agent A before handling the change in Agent B. Or you could handle both changes at the same time like in the earlier examples. This makes a difference, because the updated Agent A would have a different effect on Agent B, compared to the original Agent A. In the above example, Agent A switches their opinion before affecting Agent B.

**Example 6**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Red, No Vote, 0.4

Agent A and Agent B interact.

Updated values:

Agent A: Green, Vote, 0.6

Agent B: Red, (no actual change to red)

**Example 6**

Starting values:

Agent A: Green, No Vote, 0.5

Agent B: Red, No Vote, 0.4

Agent A and Agent B interact.

Updated values:

Agent A: Green, No Vote, 0.4

Agent B: Red, (no actual change to red)

**Example 8**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Red, No Vote, 0.4

Agent A and Agent B interact.

Updated values:

Agent A: Green, No Vote, 0.4

Agent B: Red, (no actual change to red)

Again, Agent A's uncertainty could go up, or go down, or stay the same when they change opinion. It is up to you.

**Example 9**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Red, No Vote, 0.4

Agent A and Agent B interact.

Updated values:

Agent A: Green, Vote, 0.4, ignore\_red=True

Agent B: Red, (no actual change to red)

In this example, Agent A has decided to permanently ignore Red.

**Example 10**

Starting values:

Agent A: Green, Vote, 0.5

Agent B: Red, message\_power=5

Agent A and Agent B interact.

Updated values:

Agent A: Green, Vote, 0.3, ignore\_red=True

Agent B: Red, (no actual change to red)

In this example, Red has a message\_power instead of an opinion and uncertainty. This depends on how you want to do things.

A Blue agent interacting with a Green agent is similar to a Red agent interacting with a Green agent. The difference is that Blue doesn't have a risk of being ignored, and instead Blue's actions have an energy cost. Also, Blue can choose to bring in a Grey agent instead of interacting with Greens.

A Grey agent interacting with a Green agent is similar to Red or Blue agents interacting with Green agents. If the Grey agent is a Grey-Red agent, it has the same moves as a Red agent but with no chance of making Green agents permanently ignore Red. If the Grey agent is a Grey-Blue agent, it has the same moves as a Blue agent but with no energy cost. (To clarify, Grey-Blue agents can do the same interaction that Blue agents can do. Grey-Blue agents can't introduce a Grey agent as their move.)