Computer Practical: Agent-Based Modelling

Paul Connolly and David Topping, November 2021

1 Overview

Agent-based models (ABMs) are a class of computational models for simulating the actions and interactions of entities known as *agents*. The computer simulation is used to model the effect that these agents have on the system as a whole. These models are particularly helpful for environmental research in a number of areas. The most obvious use may be in simulating how organisms interact with each other and the environment, or even how the outbreak of COVID19 is transmitted to and from people around the world.

Often modelling uses computational methods to solve *differential equations*, which govern physical laws, to make predictions about the behaviour of a system under different initial or boundary conditions. This has been the approach so far in Environmental Modelling. However, in ABM we try to simulate the behaviour of a system by considering the behaviour of constituent entities, known as agents, and how the agents are linked to each other. As an example we could use agent-based modelling to consider the effect of environmental change on migration [e.g. Thober et al., 2018, who found 'Agent-based models (ABMs) are valuable tools to study migration because they can represent individual migration decisions of human'].

2 Model formulation

We will model two groups with a specific number of members and define rules how these members interact. We call these groups A and B. In this practical we will initialise a 2-D space of 100×100 cells, and randomly position 1000 members from both groups across this space. Once we have assigned those positions we assume the positions remain fixed. We then loop forward-in-time and assume the groups interact according to the distance between them. This 'attack-distance' is defined as a square 10 cells horizontally and vertically away from the each member in question. Figure 1 shows how two groups may interact with their changing surroundings in an agent-based model. Now we turn to the specifics of implementation for the practical.

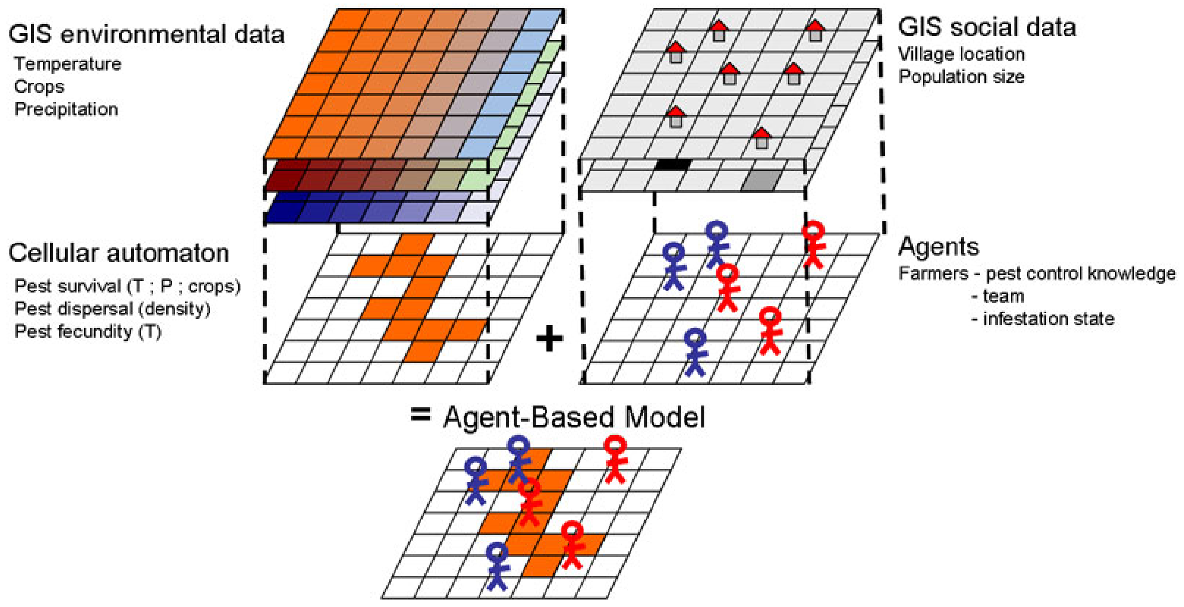

Figure 1: Top left shows a simulation of the environment; top right shows the initialisation of agents (in this case people) that interact with the environment and each other. Bottom shows the agents on the same grid as the environmental data—University of Surrey `https://mosimtec.com/agent-based-modeling-examples/`

How do we implement individual members? In Python we can use the *class object* (`https://docs.python.org/3/tutorial/classes.html`) which allows us to create many individual instances with specific properties or methods. We have not covered the class object in our courses yet, so do not worry about specifics. If you are interested you can look at the code at Listing 1 to see how a class is defined.

```python
# class, defining agents with their position and group membership
class agent:
    def __init__(self,x,y,group):
        self.score = 100 # agent's life score
        self.x = x
        self.y = y
        self.group = group
```

Listing 1: How to define an *agent* class for tracking data in the simulation with properties *score*, *x*, *y* and *group* in Python

We create an agent class by calling it like we would do a function. This is known as creating an instance of an agent class or *instantiating* the agent class. We could pass some starting *x* and *y* coordinates, and also the name of the group we wish this agent to belong to. For example, in Listing 2 we create an instance of this class:

```python
first_agent = agent(x=10,y=12,group="A")
```

Listing 2: How we instantiate an agent

Our simulation will now store and keep track of this agent as an individual 'entity'. We can access the current coordinates of this agent by using the '.' operator, so 'first_agent.x' will reveal the x coordinate. Likewise, we have prescribed an initial 'score' which we can use to monitor whether this drops to 0 and thus the agent is removed / dies. In our example, we can store each class instance in a Python list.

Thus, in the algorithm, we implement the following:

- Each agent searches their local space, defined as "'attackRange"' units horizontally and vertically from its current location.
- Each agent has a randomly distributed attack 'damage' to a member of the other group, ranging from 10 to 60 units extracted from the agent's "'score"'.
- Once a particular agent has a lifescore at or below 0 they are removed from the 2-D space.

After we loop through these rules we can asses the impact of a number of initial conditions or strategies imposed on each group. To visualise the location of each agent, we create a 2D numeric space with 0 for no entries, 1 for group *A* and 2 for group *B*. See Algorithm 1 for a summary.

Algorithm 1: Algorithm for simulating the agent based model

```
Result: Remaining numbers of species A and B
Define the properties each agent has through definition of a class instance;
Initialise population over a 2D space, creating a pre-defined number of instances of A and B;
Code the strategy each group has wrt searching a local space and acting on the other specie;
while a specific number of iterations is met do
    Move through each cell in the 2D space for each iteration, implementing the aforementioned
      strategies ;
end
```

3 Downloading the code

Log into the server computer with an SSH window. Also, log into the server computer with an SFTP window. Space the windows out on your screen so that you can easily move between them.
If you have not done this already, download the code you will be using today from GitHub by typing the following:

```
git clone https://github.com/EnvModelling/Env_modelling
```

this should download the code to your working directory.
In order to access the files you will need to change the working directory to where the files are. Type the following:
`cd Env_modelling/Spatio-temporal-modelling`
followed by the `tab` key and the command should auto-complete. Then press the `enter` key.

Type `ls` followed by the `enter` key. The screen should list the files in this directory.

4 Viewing and Editing the code

We will use the `nano` text editor to view some files. In this practical there is one file of interest called `Agent_based_2D.py`. To look at the file type:

```
nano -l Agent_based_2D.py
```

the `-l` means to show line numbers. The code can be edited in the text editor and saved by typing `Ctrl-X` at the same time and pressing `Y` to save the file. The important lines are 14-17 of this file.

5 Running the model

The procedure for running the model is to:

1. Edit the file `Agent_based_2D.py` to configure the model.
2. Type `python3 Agent_based_2D.py` at the command line, followed by `enter` to run the model.

After running the model plots of the model simulation will be shown in `/tmp/<username>/time_series.png` and `/tmp/<username>/animation.gif`, which you can download using SFTP by typing:

```
get /tmp/<username>/<filename>.png
get /tmp/<username>/animation.gif
```

6 Experiments

FOR ALL EXPERIMENTS: I STRONGLY SUGGEST YOU OPEN UP A WORD OR POWERPOINT DOCUMENT AND INSERT THE FIGURES AND MAKE SOME NOTES AS YOU GO. YOU COULD BE ASKED QUESTIONS ABOUT THEM IN THE ASSESSMENT.

6.1 Standard Simulation

The standard model has 1000 of each type of agent and group B's search strategy is random over the attack-range.

```
    python3 Agent_based_2D.py
```

Download the resulting output image files from the SFTP window

```
    get /tmp/<username>/time_series.png
    get /tmp/<username>/animation.gif
```

The species concentration is shown as a fraction of the initial number.

Question: Is one group more successful than the other? Why might this be? Do you notice any interesting behaviour in the animation plot related to the positioning of the different groups?

6.2 Reducing the number in group B

Set `NUMB=600`.

This changes the number of agents in group B to 600. Following this run the model by typing:

```
python3 Agent_based_2D.py
```

Download the resulting output image files from the SFTP window

```
get /tmp/<username>/time_series.png
get /tmp/<username>/animation.gif
```

Question: Which is the most successful group now? Look at both plot and animation and say something in your own words about what is happening?

6.3 Change the strategy of group A to match group B

Set `NUMB=1000` and `RAN_A=True`

This increases the number in group B back to the default and changes the strategy of group A to match that of group B, i.e. random search of the area rather than sequential. What do you predict the result to be? Run the model by typing:

```
python3 Agent_based_2D.py
```

Download the resulting output image files from the SFTP window

```
get /tmp/<username>/time_series.png
get /tmp/<username>/animation.gif
```

Description: Did you get what you expected? Run the model a few times and see if the result changes. Do you get changes in the results, and if so why or why not?

6.4 Increase the attack range of A to 20

Reset `RAN_A=False`

Go to line 199 and add 10 to the attack range, i.e.:
`oneRoundAgentA(i = x, j = y,attackRange=(ATTR+10))`
This will add 10 to the attack range for group A. What do you predict the result to be? Run the model by typing:

```
python3 Agent_based_2D.py
```

Download the resulting output image files from the SFTP window

```
get /tmp/<username>/time_series.png
get /tmp/<username>/animation.gif
```

Description: Was this the expected result? Why do you think you get this result?

References

Jule Thober, Nina Schwarz, and Kathleen Hermans. Agent-based modeling of environment-migration linkages: a review. *Ecology and Society*, 23(2):art41, 2018. ISSN 1708-3087. doi: 10.5751/ES-10200-230241. URL `https://www.ecologyandsociety.org/vol23/iss2/art41/`.