



Module 6 : Modelling Behaviour
 Session 6C : Agent Based Simulation using MESA Part 2
 Dr Daniel Chalk

"Agents of Change"

Agent Based Simulation and MESA

Part 2

Dr Daniel Chalk

Batch Runs

- Live runs of the model are useful, but if we want to really explore the results of the model, we need to run our model in batch, running the model lots of times and testing different parameter values.
- MESA allows us to do this very easily using a library called batchrunner
- All we need to do is specify which parameter values we want to remain the same, which ones we want to change, how many times we want to run the model (and for how long) and batchrunner does the rest.
- I'll demonstrate using a modified version of the code we worked on in the last session that includes immunisation. You can check this out in your own time.

Changing our disease_run module

- Up to now, we've just used the disease_run module to launch a server so we can watch the model run live.
- Now we're going to use the disease_run module to run the model in batch.
- First, let's comment out the server stuff from the run file as we don't want to launch the server if we're running in batch. But we can leave it here so if we want to view it live again, we can just uncomment it (and comment out the new code)

```
"""Use this section for watching model run on a server with visualisation"""  
  
#from disease_server import server  
#server.port = 8521 # the default  
#server.launch()
```

Changing our disease_run module

- Now let's write the file for a batch run. First, let's import the libraries we need.

```
"""Use this section for running the model in batch runs"""

# Import everything from our disease_model module (all classes and functions)
from disease_model import *
# The arange function of numpy allows us to create evenly spaced numbers
# within a given interval - we'll use that below
from numpy import arange
# Import matplotlib so we can plot our results
import matplotlib.pyplot as plt
# Import the BatchRunner class so we can run our model in batch
from mesa.batchrunner import BatchRunner
```

Changing our disease_run module

- Now, let's specify the fixed parameter values – those inputs to the Model class constructor that we want to stay the same throughout all runs

```
# Set up a dictionary to store the attributes we want to keep fixed
fixed_params = {"width":10,
                "height":10,
                "initial_infection":0.8,
                "transmissibility":0.5,
                "mean_length_of_disease":10,
                "mean_imm_duration":20,
                "prob_being_immunised":0.05}
```


Changing our disease_run module

- Then, we need to specify those that we want to change on different runs. We need to specify the range of values to test, and the increment.
- Here, we'll test different numbers of agents from 2 to 100 (in increments of 1) and different levels of movement from 0 to 1 (in increments of 0.1)

```
# Set up a dictionary to store the attributes we want to change between
# tested scenarios, and the values to test. Here, we'll test different
# numbers of agents in the model between 2 and 100 (in increments of 1), and
# different levels of movement from 0 to 1 in increments of 0.1
variable_params = {"N":range(2,100,1),
                   "level_of_movement":arange(0.0,1.0,0.1)}
```

Changing our disease_run module

- We also need to specify the number of iterations to run for each combination of parameter values tested, and the number of time steps in each run

```
# Run each combination 5 times, for 365 days of simulated time  
num_iterations = 5  
num_steps = 365
```


Changing our disease_run module

- Now we just need to create a batchrunner with all this information and tell it to run through all combinations. We'll also store the results in a Pandas DataFrame, and plot the results as a scatter plot.

```
# Set up a BatchRunner with the above information
batch_run = BatchRunner(Disease_Model,
                        fixed_parameters=fixed_params,
                        variable_parameters=variable_params,
                        iterations=num_iterations,
                        max_steps=num_steps,
                        model_reporters=
                        {"Total_Infected":calculate_number_infected,
                        "Total_Imm":calculate_number_immunised}
                        )

# Tell the BatchRunner to run
batch_run.run_all()

# Store the results in a Pandas DataFrame
run_data = batch_run.get_model_vars_dataframe()

# Plot a scatter plot of level of movement vs total infected
plt.scatter(run_data.level_of_movement, run_data.Total_Infected)
```

CAUTION!!!!

- Be careful how many different combinations of parameter values you test in your batch run. You can very easily get to huge numbers of combinations that can bring down even the fastest computers...

Exercise 1

You are now going to use everything you've learned in this module, including your new skills in MESA, to **design** and **build** an Agent Based Simulation.

You will work in your groups for 2 hours. You should do the following :

- Discuss some ideas about potential problem areas that you could use Agent Based Simulation to tackle. You may choose one of the ideas you came up with in the Cellular Automata session if you like, or think of something new!
- Choose one of these ideas, and draw up a design for how the model would work – what will be your agents? What behaviours and movement patterns will they have? Will they interact, and if so, how? What will your cells represent? What kind of cell neighbourhoods will you have etc
- Build a prototype “dummy” model using MESA. I'm not expecting you to have the data you need to parameterise it (although maybe you will) but just have a go at building your ABS design in MESA. Put on a nice interface, including any user interface elements (e.g. sliders) to allow the user to play around with your model.

After 2 hours, I'll ask each group to present their models! And I'll also ask you all to share them on Slack so other people can play with them too!

Further Reading

- MESA Documentation and Tutorials :
<https://mesa.readthedocs.io/en/master/>
- My thesis (if you don't want to pay £69) :
<https://ore.exeter.ac.uk/repository/bitstream/handle/10036/96455/ChalkD.pdf?sequence=1>
- Individual Based Modelling and Ecology
Book by Volker Grimm and Steven F Railsback
(ignore the title – good for general ABS interest)
- Cellular Automata
<http://mathworld.wolfram.com/CellularAutomaton.html>