Simulating Crowds in Real-Time with Agent-Based Modelling and a Particle Filter

Kevin Minors*[1], Andrew West†[1] and Nicolas Malleson‡[2]

[1]Leeds Institute for Data Analytics, University of Leeds, Leeds LS2 9JT, UK
[2]School of Geography, University of Leeds, Leeds LS2 9JT, UK

January 18, 2019

KEYWORDS: Agent-Based Model, Data Assimilation, Particle Filter.

1 Abstract

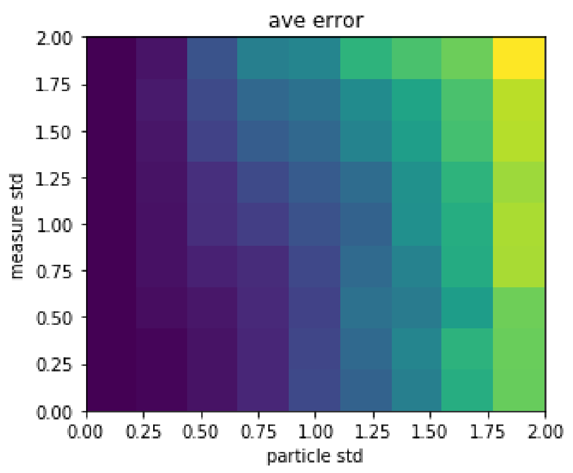
In recent years, cities have used the increase in data to make decisions that result in the city becoming more efficient and sustainable through automation. These cities are known as smart cities. One of the many important questions smart cities would like to answer is how do people behave? Being able to model the behaviour of a population in a city will prove very fruitful, as it will inform a variety of policy decisions. It is often very difficult to gain insight into human behaviour from a macroscopic point of view in real time.

One way to solve this issue is to use agent based models (Macal and North, 2005). These models attempt to simulate the behaviours of a population from a microscopic point of view by using simulated individuals, the 'agents'. The behaviour of the agents can be programmed to reflect the various dynamics of a particular city. One key component of agent based models is that there is some collective behaviour forming that is not programmed into individuals, known as 'emergence'. Agent based models are generally given some initial conditions and then they evolve forward in time independently from the real world environment they are simulating. This often results in the models diverging away from reality due to the limited accuracy of the initial conditions and model behaviour. To solve this problem, these models can use real time data from reality to update the state of the model in real time. This is known as data assimilation. New data is used to ensure the model continues to accurately reflect reality.

There are many ways to carry out data assimilation. One such method is known as a particle filter (Doucet et al., 2000). Particle filters run numerous instances of the agent based model, known as 'particles', and allow them to develop forward in time. As new information is received about the

*k.minors@leeds.ac.uk
†gyawe@leeds.ac.uk
‡n.s.malleson@leeds.ac.uk

true state of reality, the particle filter weights the particles based on how similar they are to the model state determined by the new information. Particles with a large weight are model runs that are very similar to the information received while particles with small weights are model runs that do not agree well with the new data. The particles are then resampled based on their weight, so that particles with a large weight are more likely to be chosen to be duplicated and particles with small weights are more likely to be discarded. In this way, the resampling results in particles with larger weights surviving and particles with small weights dying off. At this stage, the particles run forward in time until the next piece of information is received and the process continues.



This particle filter method allows the models to run while also being able to update itself with new data in real time, allowing the model to better simulate reality over long periods of time. One of the advantages of using particle filters over other types of data assimilation is that it can handle nonlinear dynamics, which applies to a wide range of applications. One of the disadvantages is that it can be quite computationally expensive as complicated models require a significant number of particles in order to accurately assimilate the new data. Problems with particle filters include all particles collapsing to one state due to a lack of diversity, known as the degeneracy problem, and all particles failing to be similar to the new information due to not enough particles being simulated.

The agent based model we are working on represents an environment like a corridor. In the model, agents enter the corridor from one side and walk through the corridor to exits on the other side. Just like people in real life, the agents have different speeds and different destinations. The emergence we find in this agent based model is crowding. As agents cross paths attempting to go to their exit and as slow agents block the path of fast agents, crowds form. We chose this model because it is simple enough for us to carry out data assimilation on it in a meaningful way and because it is complicated enough to have the emergent behaviour of crowding. In this case, as we do not have data from any corridor in particular, we use the model itself as our reality from which we receive new data. More details can be found on our Github page (Malleson, 2018).

References

Doucet, A., Godsill, S., and Andrieu, C. (2000). On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10:197–208.

Macal, C. M. and North, M. J. (2005). Tutorial on Agent-Based Modelling and Simulation. In Kuhl, M. E., Steiger, N. M., Armstrong, F. B., and Joines, J. A., editors, *Proceedings of the 2005 Winter Simulation Conference*, pages 2–15. Institute Of Electrical And Electronics Engineers, Piscataway, NJ.

Malleson, N. (2018). Urban analytics: Dynamic urban simulation technique. `https://github.com/Urban-Analytics/dust`. Accessed: 18/01/2019.