Coupling an Agent-Based Model and Ensemble Kalman Filter for Real-Time Crowd Modelling

Keiran Suchak   Jonathan A. Ward   Le-Minh Kieu

Nick Malleson

January 25, 2023

Abstract

Write abstract here.

KEYWORDS: Agent-Based Model; Crowd Simulation; Data Assimilation; Ensemble Kalman Filter; Data-Driven Agent-Based Modelling.

1   Introduction

Agent-based modelling (ABM) (1; 2) is an effective, albeit not particularly widely used, method for creating simulations of systems that are driven by the behaviours and actions of individuals. Rather than attempting to describe systems in terms of their aggregate properties, agent-based models (ABMs) simulate individual 'agents' (e.g. people, animals, vehicles, particles, etc.) directly. This allows modellers to capture many important characteristics that are inherent to the underlying system and cannot be represented with aggregate models (3; 4).

Despite progress in recent decades to improve the robustness of the methodology (e.g. 5)), ABM still faces a number of challenges (for example see 3; 6; 7) including the understanding and managing of uncertainty (8). In particular, only a handful of studies have attempted to reduce the uncertainty that will naturally arise as a model and a system evolve. Model calibration is not sufficient for this as even a perfectly calibrated model will diverge from a real system over time. Models are typically calibrated once, using historical data, and then projected forward in time to make a prediction independently of any new data that might arise. Hence "live" simulations (9)—i.e. those that are attempting to simulate a system in *real-time*—need a way to update a model state in response to emerging data. Such systems could be transformational in managing a number of different problems. For example, a live simulation of a busy public place such as a public transport hub could allow a manager to better understand how people were moving around and prevent dangerous crowd bottlenecks from forming. Similar 'real time' systems could revolutionise approaches to disaster response or the spread of disease (9).

This paper makes an important contribution that is necessary if agent-based models are to become more widely used in policy, particularly for *real-time* applications. Specifically, its main aim is to demonstrate that an Ensemble Kalman Filter (EnKF)—a method that has shown great value in updating models of physical systems such as the climate—can improve the accuracy with which an agent-based model simulates a system of pedestrians. We present an existing model that can be used to simulate pedestrians in a public transport terminal and show that the model state, i.e. the positions of the pedestrian agents, can be updated with new observations as they emerge to keep the model in line with the evolution of the real system.

Need any more about the results?

The paper is structured as follows: Section 2 reviews the relevant literature; Section 3 outlines the agent-based model, the Ensemble Kalman Filter and the data used; Section 4 outlines the experiments that will be conducted; Section 5 outlines the results; and Section 6 draws conclusions.

2 Background

2.1 Real-Time Agent-Based Modelling

Agent-based modelling (ABM) is technique that has developed as a means of modelling systems that are ultimately driven by discrete, heterogeneous and autonomous *agents* (1; 2). Many human systems are well suited to simulation with ABM because it avoids the loss of accuracy that can occur when individual-level dynamics are 'smoothed out' by models that work at an aggregate level (10). Simulations of short-term human movements, such as pedestrian dynamics—the subject of this paper—are no exception. ABM has become a particularly popular methods for modelling crowds (11) because it captures the individual heterogeneity in a population as well as the interactions between them, both of which are essential for creating reliable micro-level crowd models (12).

Most agent-based crowd models (and agent-based models more broadly) are essentially models of the past. They are designed to represent a particular scenario and then calibrated and validated using historical data. If the model can successfully reproduce these data then it is subsequently used to predict the impacts of internal or external changes to the system, akin to a thought experiment. Such an approach has theoretical and empirical value—for example by revealing the underlying mechanisms behind observed crowd dynamics or for general policy or infrastructure planning, respectively—but breaks down if crowds need to be simulated in *real time*. In this case, various inevitable uncertainties would cause even a perfect model to quickly diverge from reality (13). Despite considerable progress in the area of ABM calibration (5; 14), little progress has been made towards methods that can control the uncertainty in models *during run time*. This makes it hard to predict systems that rapidly evolve, such as pedestrian systems, in real-time using agent-based models.

2.2 Calibration and Data Assimilation

An approach that could potentially resolve the issue of real-time model divergence is that of *data assimilation* (DA). DA has had a variety of uses in the physical sciences, the most notable of which is in weather forecasting, where DA has been on of the key innovations that has caused weather predictions to improve so substantially in recent decades (15). DA refers to a variety of mathematical approaches that allow new observational data to be incorporated into models at run time (16), reducing the uncertainty in the model's predictions.

A fundamental difference between typical optimisation ('calibration') and DA is that rather than trying to find optimal model parameter values, DA attempts to estimate the 'true' system state, taking account of the model's current estimate of the state as well as recent observations. It then updates the model *state* (and, optionally, its parameters) so that the model is closer to the real state of the underlying system. In other words, DA allows model's evolution to be constrained against the observations as they arise (17). Whilst real-time calibration efforts exist (e.g. 18), these are not able to reduce the natural uncertainty in the system state that arises as stochastic models evolve. For example, considering a model of a human crowd, a calibration process may improve the accuracy with which the agents move through an environment. However, a real-time the model will still perform poorly even after calibration as it may have already diverged from reality,

so although the agents behave more realistically they may be located in fundamentally the wrong place!

2.3   Real-Time Agent-Based Models

Although 'data-driven agent-based modelling' is not uncommon, relevant approaches usually refer to *dynamic calibration*—i.e. re-calibrating a model in response to new data (e.g. 18)—or approaches that adapt agent behaviour in some way—for example deriving agent behaviour directly from data (19). Approaches that attempt to update the model state directly, as is the aim with DA, are much rarer.

One of the earliest studies (17) demonstrated that an Ensemble Kalman Filter could be used to optimise an agent-based model, although the model used was much less complex than a more traditional agent-based model with very limited inter-agent interaction. Similarly, attempts have been made to optimise the 'macro' model state space, i.e. aggregate model characteristics (20), but this ignores local spatial interactions between agents, allowing the model to perform well on an aggregate level but not necessarily at a micro-level. These micro-level interactions are crucial for reliably modelling crowds. Recent attempts have been made to model simple pedestrian behaviour using Particle Filters (21; 22; 23) and Unscented Kalman Filters (24; 25). Although these studies use models that include larger numbers of agents as well as agent-agent interactions, making them more representative of typical agent-based models, the environments employed are highly stylised in some cases (23; 24; 25) and in others the number and complexity of agent-agent interactions is limited (21). Entirely novel approaches that adapt theoretical mechanisms from quantum physics (26) or leverage Markov Chain Monte-Carlo methods (27) are interesting, but still in their infancy and not yet appropriate for use in a 'large' model (i.e. one with more than a handful of agents).

In summary, although progress has been made towards robust data assimilation algorithms, this paper presents the most advanced approach. It leverages an Ensemble Kalman Filter—an approach that has been rarely used despite the potential efficiency advantages over the more common Particle Filter—and also uses a model that has been built to represent a real environment rather than an abstract space.

3   Methods

Write methods

3.1   Data and Case Study Area

About GCT and data

3.2   Model

Explain about `StationSim_GCS`.
   Things to note:

- What an 'active' agent is

- Which model variables are known to the ensemble models (start gate and exit gates?)

- 'side stepping' behaviour the agents use to avoid each other and obstacles.

3.3 Ensemble Kalman Filter

- Explain about the Ensemble Kalman Filter (28), which is based on the Kalman Filter (29).

- Point to previous experiments in the thesis. E.g. "Results: The data assimilation scheme was tested for a range of different filter parameter values, and it was found that improvements in filter performance resulted from increases in the ensemble size, reductions in the standard deviation of the observation error and reductions in the number of time-steps between successive attempts to assimilate observational data into the system." (p113)

4 Experiments

The experiments, outlined visually in Figure 1, aim to demonstrate that the EnFK can improve the accuracy of a pedestrian system in comparison to a baseline scenario with no data assimilation. In order to better understand the impact of data assimilation on an agent-based model, rather assess the realism of the model itself, we use the "identical twin" approach (30). In this approach, a 'Base Model' is used. A Base Model is an instance of StationSim_GCS that is used to generate 'pseudo-true' data that are taken as the real-world observations in the experiments (in lieu of data from a real crowd).

(a) Experiment 1: Benchmarking

(b) Experiment 2: Exploring Ensemble-Member Models

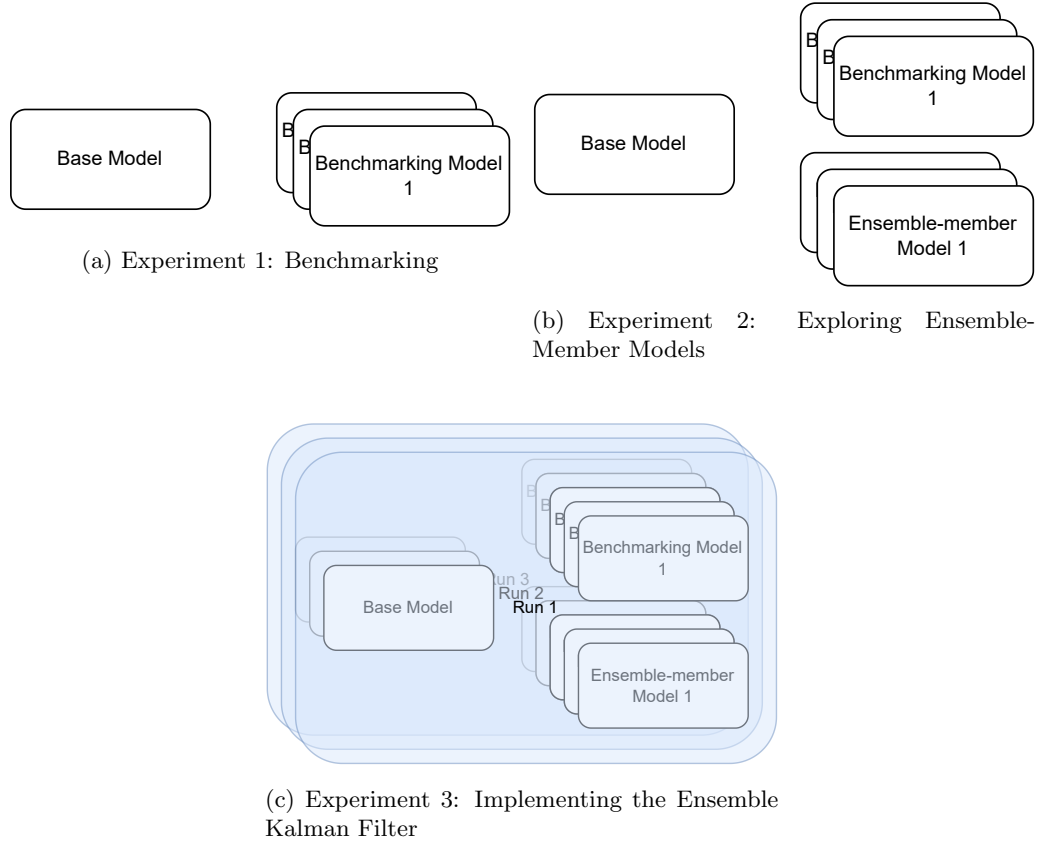(c) Experiment 3: Implementing the Ensemble Kalman Filter

Figure 1: Graphical outline of the three experiments. Note that the 'base model' is use to generate pseudo-true observation data

The initial experiment, ('benchmarking', Figure 1a) seeks to establish a benchmark against which to compare subsequent implementations of the EnKF. This is achieved by running an ensemble of models, each initialised as duplicates of a base model which is used to generate pseudo-truth values for the system state.

The second experiment (Figure 1b) seeks to XXXX. It does this by exploring the variation in the accuracy of individual ensemble members. This is achieved by running a single EnKF which maintains a benchmarking ensemble of models, providing a baseline against which to compare results, along with and ensemble of models that are periodically updated by the EnKF assimilation process. In such a situation, we are able to compare the average error per agent in each of the ensemble member models at each assimilation time-step.

The final experiment (Figure 1c) takes this exploration a step further by seeking to capture the variation in error at an ensemble level. This involves running a collection of EnKFs for the same set of model and filter parameters, and in each case gathering data regarding the variation in the error in the ensemble mean state over time, comparing this with the variation in the corresponding collection of benchmark errors.

4.1 Measuring Error

- Talk about measures used when running experiments with multiple EnKFs to ensure that outliers don't skew results:

    - Median instead of mean error.
    - Only considering time-steps when a sufficient number of models are active.

4.2 Active and inactive agents

As the following sections will discuss, error is calculated by comparing the positions of agents in the simulation with the positions of corresponding agents in the base model (i.e. the pseudo-truth data, discussed in Section 4). To do this, it is necessary to consider whether an agent is 'active' or 'inactive' as once an agent has left the simulation they should not be included in an error calculation. However, an agent might be active in the base (pseudo-truth) model and inactive in some or all of the EnKF ensemble members, or vice versa. Here we assume that an agent is active only while it is active in the EnKF ensemble because in a real situation we would not necessarily have access to the true positions of the individuals in the crowd, so could only assess an agent's status from the information available in the ensemble of models. Hence an agent is considered active if its most common (i.e. modal) status across the ensemble is active.

4.2.1 Agent-level error

Error at the level of the individual agents is quantified by calculating the distance between the position of an agent estimated by the ensemble of models in the EnKF (the 'ensemble mean state') and the position of the corresponding agent in the base model, d_i:

$$d_i = \begin{cases} |\hat{\mathbf{x}}_i - \mathbf{x}_i| & \text{if } i\text{th agent is active;} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $\hat{\mathbf{x}}_i$ is the x-y position of the ith agent estimated by the ensemble of models and $\mathbf{x}_i$ is the x-y position of the ith agent in the base model. The distance between the $\hat{\mathbf{x}}_i$ and $\mathbf{x}_i$ agent is calculated using the Euclidean distance.

4.2.2 Model-level error

To calculate the error across the whole system, we calculate the average distance over all active agents in the system, $\bar{d}$:

$$\bar{d} = \frac{1}{N} \sum_{i=1}^{N} d_i, \tag{2}$$

where N is the number of *active* agents. This average distance, $\bar{d}$, can then be used to measure the error in an ensemble of models given the ensemble mean state for a given time-step and the base model state at the same time-step. In this way, we create a base model which is used to generate a ground truth and an ensemble of models from which we can obtain the average behaviour behaviour by averaging across the ensemble. The accuracy with which this average behaviour simulates the ground truth generated by the base model is assessed by considering the error between the base model and the average of the ensemble.

4.3 Experiment 1: Benchmarking

The initial experiment to be performed is to develop a model baseline, establishing the effectiveness of `StationSim_GCS` in modelling a system in the absence of any information whilst running.

> Is this what the experiment actually does, or is it more about quantifying the uncertainty in the simulation? Aka "ensemble variance"?

This is applied to ensembles with increasing population sizes (as per Table 1) to explore how this behaviour varies with population size.

> Should we remove the last four rows in the table because these are fixed?

Parameter	Value
Population size	$[10, 20, 50, 100]$
Ensemble size	100
Number of entrances	11
Number of exits	11
Environment height	700
Environment width	740

Table 1: Table of model parameters used for estimating the baseline level of error.

In the benchmarking experiments, the following approach is taken:

1. Create an instance of the model to be considered the base model which provides pseudo-truth states of the pedestrian system.

2. Create an ensemble of 100 models, each of which is a copy of the base model; this means that each of the duplicates in the ensemble contain the same information regarding which exits each of the agents will enter and exit through, as well as at what time they will be activated within the model. These models, however, are liable to diverge from the base model due to the collisions that occur between pedestrian agents.

3. Iterate each of the base model and the ensemble of models forward for each time-step. At each time-step, calculate average model state for each agent in the system population, and calculate the average error per agent between this average model state and the pseudo-truth state generated from the base model for this time-step.

6

4.4 Experiment 2: Exploring Ensemble Members

Better word than 'exploring' above? Something more specific?

After having established a benchmark for the accuracy with which the `StationSim_GCS` model can simulate the trajectories of pedestrians moving around the concourse of Grand Central Station in New York), the next experiment aims to explore the variation amongst the models *within* the ensemble of an EnKF whilst observations are being assimilated into the ensemble. In order to achieve this, rather than calculating the distance between the ensemble mean state and the base model state (as per (1)), the error is calculated by the distance between each of the ensemble member model states and the base model state. If we consider d_{ij} to represent the distance error of the jth model's state for the ith agent compared to the ith agent in the base model, this can be calculated as follows:

(earlier point about "ensemble variance"

Explain why this is important

$$d_{ij} = \begin{cases} |\hat{\mathbf{x}}_{ij} - \mathbf{x}_i| & \text{if } i\text{th agent is active}; \\ 0 & \text{otherwise}, \end{cases} \tag{3}$$

where $\hat{\mathbf{x}}_{ij}$ is the x-y position of the ith agent in the jth model and $\mathbf{x}_i$ is the x-y position of the ith agent in the ground state system, i.e. the base model. Given this expression, we adapt (2) slightly to calculate the mean error per agent for each ensemble member model, $\bar{d}_j$, at a given time-step as:

$$\bar{d}_j = \frac{1}{N} \sum_{i=1}^{N} d_{ij}. \tag{4}$$

Based on this, we can construct a vector containing all of the mean errors per agent for each of the ensemble member models:

$$\begin{align} \bar{\mathbf{d}} &= \left[\bar{d}_1, \ldots, \bar{d}_M \right] \tag{5} \\ &= \left[\bar{d}_j \right], \forall j \in (1, M), \tag{6} \end{align}$$

where M is the ensemble size. A vector of average errors per agent for each ensemble member models can then be calculated for each time-step.

Based on this error calculation process, we can explore the variation of error across the ensemble using the following steps:

1. Create an EnKF with a population size of 20 pedestrians, containing a base model and an ensemble of 20 duplicates (prior experimentation showed only marginal performance improvements above an ensemble size of 20 members.

ref Keiran thesis

2. Iterate each of the base model and ensemble of models forward for each time-step. If the time-step is an assimilation time-step, i.e. a time-step in which synthetic data are produced from the base model, the ensemble of models are updated using the update procedure outlined in Section 3.3. Assimilation time-steps occur with a fixed period of 20 time-steps. At each assimilation time-step, errors are calculated between the following pairs:

   - The base model state and each of the ensemble member models, as per (**??**), after updating with observations.

   Keiran could you refer to the specific equations here?

   - The base model state and the mean state of the ensemble of models, as per (**??**), after updating with observations.

   - The base model state and the mean state of the benchmarking ensemble (i.e. the baseline error), as per (**??**).

Need a table of parameters like Table 2?

4.5   Experiment 3: Assessing the EnKF

The previous experiments will show that, as expected, the error in the ensemble mean state accurately reflects the variation in error over time, and that it does not differ greatly from the error in each of the ensemble member models. On this basis, this final experiment focusses on comparing the variation in the ensemble mean state error over time with the error in the benchmarking ensemble as well as with the error in the observations of the pedestrians' locations on the station concourse. This considers both the ensemble mean state before and after assimilating observations, i.e. the forecast error and the analysis error. The experiment will make use of the parameters outlined in Table 2.

Parameter	Value
Population size	20
Ensemble size	100
Assimilation period	20
Observation noise standard deviation	1.0

Table 2: Table of filter parameters used for the EnKF.

As with the benchmarking experiment, errors are calculated as the distance between the 'estimated' agent location and the agent's location in the pseudo-truth base model; see (1). Note that in this case, there are different ways to 'estimate' an agent's location and hence calculate error. We calculate the following errors:

- the base model state compared to the forecast ensemble mean state (i.e. the prior error);

- the base model state compared the analysis ensemble mean state (i.e. the posterior error);

- the base model state compared to the observations provided to the ensemble for updating (i.e. the observation error);

- the base model state compared to the mean state of the benchmarking ensemble (i.e. the baseline error)

We use all of these methods and in each case the distance is calculated in the same manner. Consequently, an average error in each of the four state estimates across all pedestrians can also be calculated in the same manner as per (2).

We explore the variation of error across the ensemble using the following steps:

1. Create an EnKF containing a base model and an ensemble of 20 duplicates of the base model.

2. Iterate each of the base model and ensemble of models forward for each time-step, updating the ensemble using the data assimilation outlined in Section 3.3 every 20 time-steps, calculating the errors using the four methods above.

This process is repeated 20 times for the same model and filter parameter values.

5   Results

5.1   Experiment 1: Benchmarking

The results of Experiment 1 are shown in Figure 2, illustrating how the error varies between a benchmark model (i.e. pseudo-truth data) and an ensemble of 100 models

(without data assimilation) as the population size increases. Each of the time-series follows a similar trajectory. The initial error per agent is very low as the starting locations of the agents are known to the ensemble, but it rises sharply as agents begin to move across the environment towards their respective exit gates. The common peak in error is a consequence of the *activation rate* parameter that controls the rate at which agents enter the environment and effectively places an upper limit on the number of agents that can be present in the model at any one time. The peak in error simply at the point in time when the environment contains the largest number of agents.

The error itself is largely attributed to two factors: variations in the precise location along a gate at which an agent enters (although the entrance gate for each agent is known, the exact position within a gate is random) and the number of agent-agent interactions. At lower agent population sizes, the latter is likely to contribute less towards the error with fewer interactions occurring.

Note: have some commented text about how to contextualise the error if we think we need it (I don't think so).
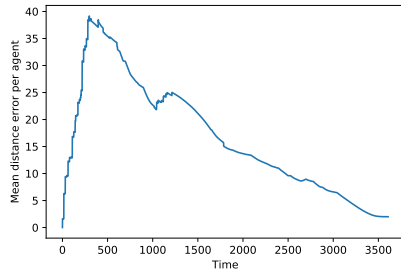
Kieran can you reproduce Figure 2 so that the y axes are the same in each sub-figure? (If it's a pain don't worry, we'll see if the reviewers pick up on it.)
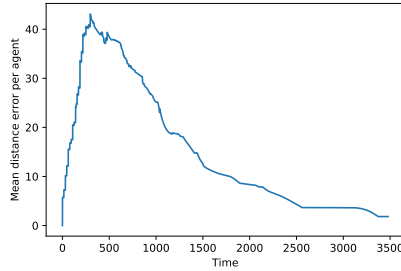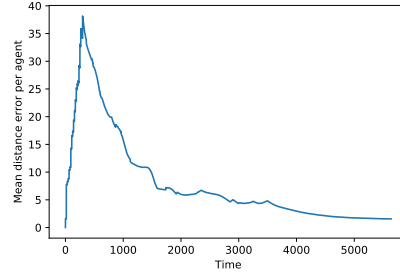

(a) 10 agents
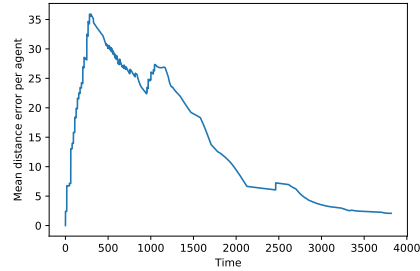
(b) 20 agents

(c) 50 agents

(d) 100 agents

Figure 2: Variation in average error per agent with model time for different population sizes.

In summary, the aim of Experiment 1 has been to create a benchmark estimate for the level of error that might be exhibited *without* data assimilation. Having established this benchmark, the following experiment explores how error varies when data assimilation is implemented.

5.2 Results 2: Exploring Ensemble Members

Experiment 2 consists of running a filter that contains an ensemble of models (the 'filter ensemble') that undergo data assimilation alongside a benchmarking ensemble of models (the 'benchmark ensemble') that have no data assimilation. Both of these ensembles are compared to observations generated by a single base model (the 'pseudo-truth' data). This allows us to compare the performance of the filter ensemble against a similar ensemble that has no data assimilation and, importantly, allows us to compare the distribution of errors *within* the filter ensemble by examining the errors in the individual ensemble models.

Overall Filter Performance

Figure 3 demonstrates that, as expected, the benchmarking error is much larger than the average error per agent calculated from the filter ensemble mean state. As with previous experiments, the benchmarking error is high at beginning of the experiment and declines over the course of the filter run.

> Keiran why in Figure 3 does the error start high (60+) whereas in the previous experiment, Figure 2 it starts at 0?

This shows us that the Ensemble Kalman Filter is extremely effective at using observations to reduce the error in the ensemble.
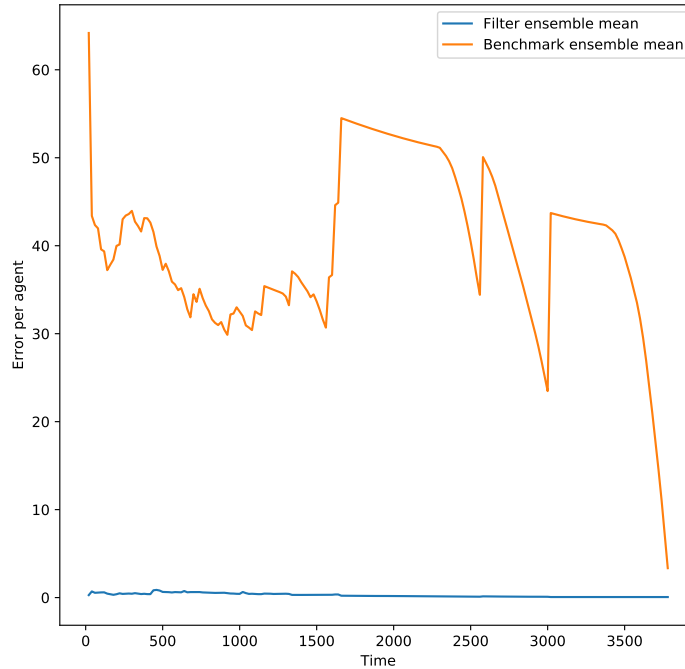
> Keiran please check this last sentence is right.

Figure 3: Line plot of the average error per agent based on the mean state of the benchmarking ensemble and the mean state of the filter ensemble.

Within-Filter Performance

We can now explore how the average error per agent varies across the filter ensemble member models and how this compares to the ensemble mean state. This is shown in Figures 4a and 4b respectively. In Figure 4a, the average error per agent is plotted for each ensemble member model as well for the ensemble mean state (plotted in bold black). We can see that the variations in the error in the individual models largely mirror those seen in the ensemble mean state error. Importantly, we see that the error per agent is considerably lower when the filter is used in comparison to the benchmark.
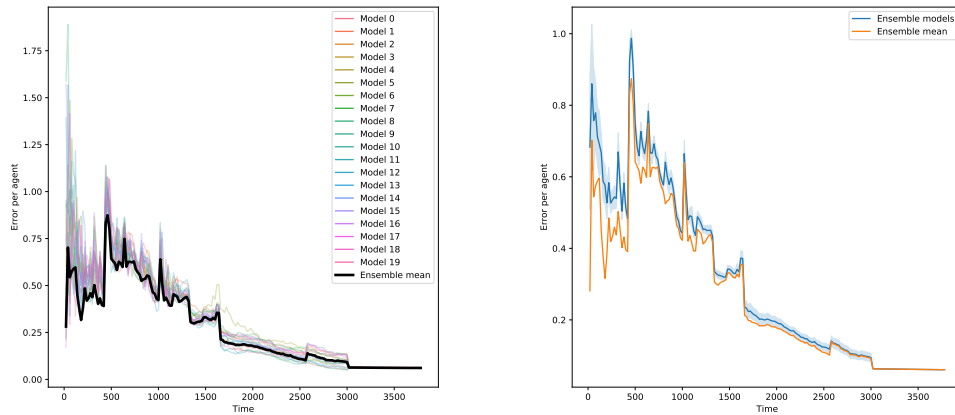
(a) Error per agent based of each ensemble member.

(b) Average error per agent from all ensemble member models, with confidence intervals.

Figure 4: Comparing the ensemble mean error with the errors of the filter ensemble members.

It is worth nothing that the error in the ensemble mean state appears to typically be lower than the errors in the majority of the individual models. This is further supported by Figure 4b which, rather than showing the error of the individual models, plots the mean of the model errors and the 95% confidence interval around it. Whilst we may have expected that these two sets of errors would be identical, this is not the case. However, this is not surprising with hindsight if we consider the hypothetical example illustrated in Figure 5. It immediately becomes clear that the mean agent location (the 'ensemble mean') is likely to have a lower error than the mean of the errors of the individual ensemble members.

In summary, this experiment has demonstrated ...

Keiran what do you think are the main takeaway messages from this section? Other than the Filter is good!

5.3   Results 3: Assessing the EnKF

Having established a model baseline level of error and undertaken some exploration of the variation in error across the ensemble-member models within an EnKF, this section completes the experiments by assessing the success of the EnKF as a means of updating an ABM in real time and explores some of the emerging challenges.
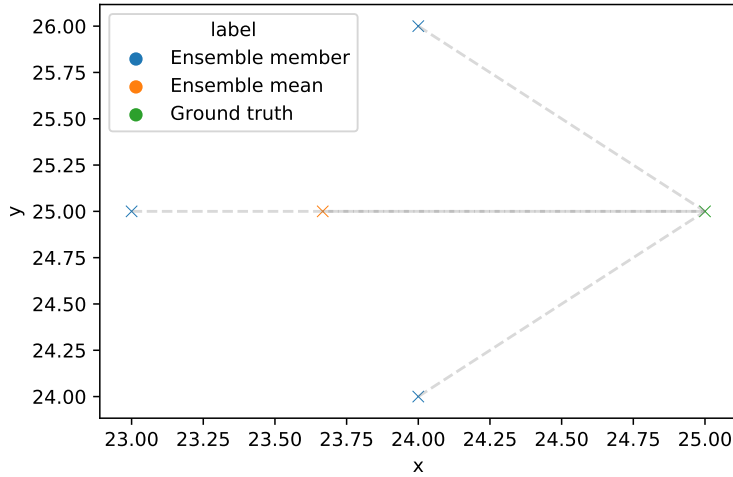
Figure 5: Working example — calculating error based on the ensemble mean compared to the mean error of each individual member models.

NM: is it clear that this section is all about error per agent, not aggregate performance?

Agent Behaviour under EnKF

It is illuminating to explore the behaviour of the individual agents as their state variables are manipulated by the EnKF. To this end, Figure 6 illustrates the prior and posterior positions of two individual agents ('agent A' and 'agent B'). We see that the introduction of observations through data assimilation has resulted in the reduction in the spread of the ensemble-member model representations of the agents, i.e. the uncertainty in the model estimate of the positions has been reduced. This pattern is observed across the other agents in the system.

Figure 6 is fantastic, but could be improved by: (i) aking the observations more obvious (I think they are obscured) (ii) reducing the size of the print area (there is a lot of unnecessary empty space) (iii) removing the points that you're not displaying from the legend. (ii) and (iii) can be done by editting the pdf if it's annoying to re-produce the figure (I can do this) but I'm not sure about (i).

Having established the ability of the EnKF to reduced the uncertainty in the estimates of pedestrians' positions in the environment, we see to tackle a number of additional challenges.

It's great to see that actual impact of the filter on the estimated positions. Any room to elaborate further on this result? Maybe a comparison with the baseline? It just feels as though the section ends a bit abruptly.

Managing Outliers

When running a large number of filters with the same filter and model parameters, there is inevitably some variation in the results. This pertains to both the way in which the average error per agent varies over time, and the length of time a filter takes to reach
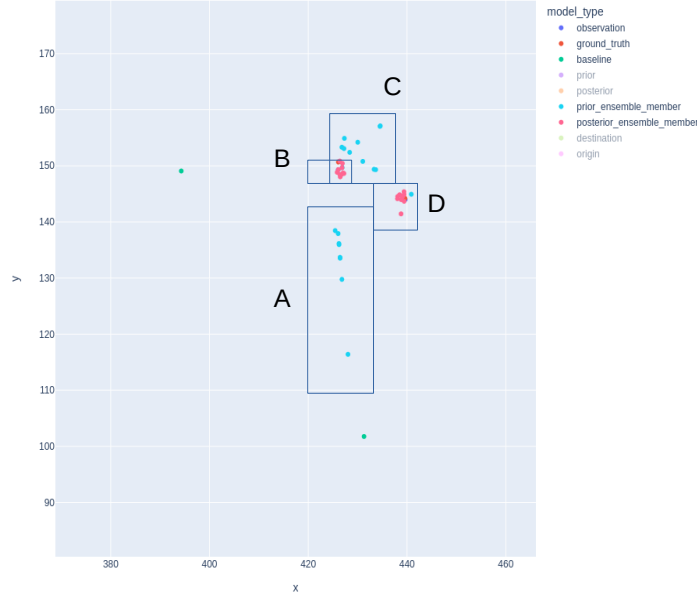
Figure 6: Comparison of prior and posterior positions (in blue and pink, respectively) of two agents ('A' and 'B') in all ensemble member models. Boxes A and B: the prior and posterior positions of agent A. Boxes C and D: the prior and posterior positions of agent B.

completion. Whilst the majority of the filters reach completion within the first 4500 time-steps, which is consistent with the baseline, some do not until approximately 10000 time-steps. This is evidenced in Figure 7. These outliers must be removed as they artificially increase the apparent error of the filter. Hence when calculating error, time-steps after which 90% of the models have completed (typically around 5000 time-steps) are disregarded. In addition, the median, rather than the mean, is used to summarise the overall error per agent.

Filter Performance

When assessing the average error per agent, three comparisons need to be drawn: (i) benchmarking ensembles against the analysis of the filter ensembles; (ii) the forecast of the filter ensembles against the analysis of the filter ensembles; (iii) and the observations against the analysis of the filter ensembles. This section goes on to explore each of these comparison. In each case, the comparison is aided by the use of two figures:

- a line plot of the average error per agent over time where the line represents the median of the collection of filters at each time-step and the shaded area represents the 95% confidence interval around the line;

- a boxplot showing the distribution of these errors when aggregated over time (logged to reduce the visual impact of the skewed distributions);

Figure 8 plots the median error per agent by comparing the benchmarking ensembles to the analysis of the filter ensembles. In Figure 8a, we can see that the average error per agent in the filter analysis states is much lower throughout. This is echoed by the logged boxplot in Figure 8b. The majority of the logged data pertaining to the analysis error
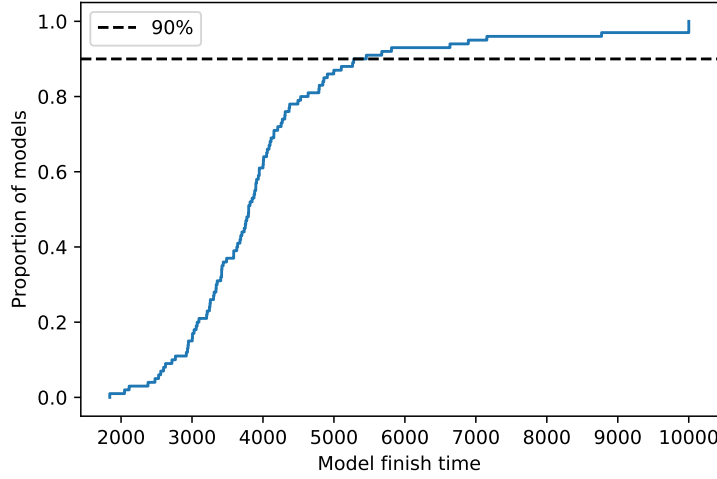
13

Figure 7: Empirical cumulative distribution function (eCDF) plot of filter finish times; dotted line represents a cumulative level of 90%.

lie below 0, indicating that the average error per agent for the analysis is often below 1. In the case of the benchmarking data, however, the majority of the data lie above 0, indicating that in most cases, the average error per agent for the benchmarking ensembles is much higher.
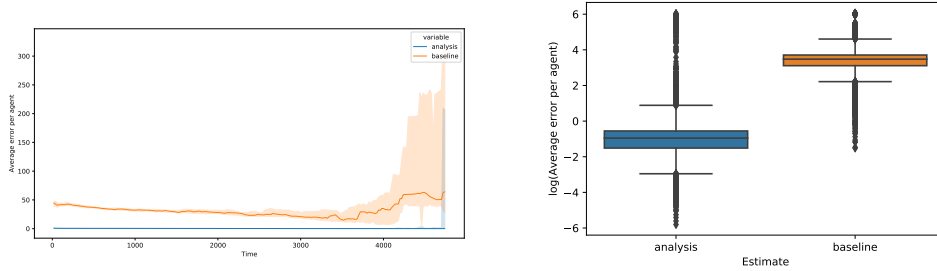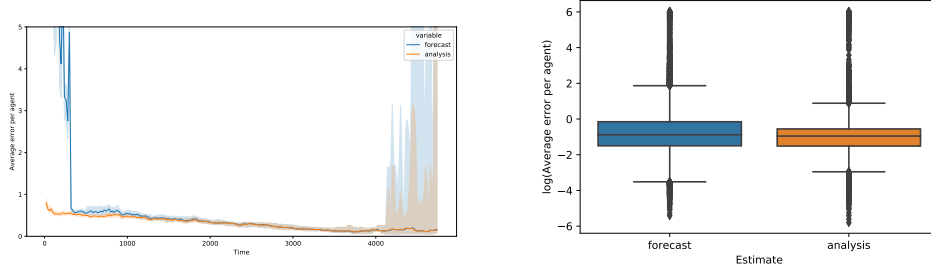

(a) Line plot of average error per agent over time.

(b) Box plot of log of average error per agent.

Figure 8: Comparison of average error per agent between analysis and benchmarking filters

Figure 9 compares the average error per agent in the forecast and analysis states of the filter model ensembles, i.e. the error before and after assimilating data at each time-step. We see that the error in the analysis state is typically an improvement on the error in the forecast state, with the improvements being most noticeable at the beginning of the set of time-steps and at the end of the time-steps (Figure 9a). The difference at the beginning is due to the reasoning outlined in Section **??** — the entrance of multiple agents at the [fix ref] beginning of the filter run time and the entry of agents at points on the gates that do not match the exact entry point of corresponding agents in the base model lead to an initial growth in error. The error in the analysis state does not suffer from this growth as it is updated by the provided observations. The increase in the variance of the errors towards

14

the end of the experiment is a consequence of many filters reaching completion and hence the summary statistics being drawn over a decreasing number of filters. Furthermore, Figure 9b illustrates that, again, the majority of the data lie below 0, indicating that the average error per agent in each case often lies below 1.

(a) Line plot of average error per agent over time. (b) Box plot of log of average error per agent.

Figure 9: Comparison of average error per agent between analysis and forecast.

Finally, Figure 10 compares the variation of the average error per agent in the (pseudo-true) observations and the analysis states of the filter model ensembles. The average observation error is largely constant throughout the experiment (Figure 10a). The increase in error variance is again caused by averaging over a decreasing number of filters. In comparison to the observation error, the analysis error is typically lower for the majority of the time-steps for which the filters are running. This is not always the case, however, as highlighted in Figure 10b. In each case, the errors appear to be low; however, there are substantial outliers pertaining to the analysis error. The observation error, on the other hand, contains relatively few outliers. This is a consequence of how the observations are produced; by adding normally distributed random noise to the base model state.
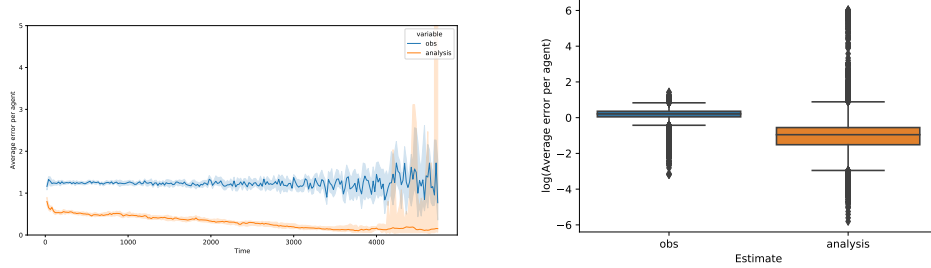
(a) Line plot of average error per agent over time. (b) Box plot of log of average error per agent.

Figure 10: Comparison of average error per agent between analysis and observations.

6   Discussion and Conclusion

This paper has, for the first time, developed an Ensemble Kalman Filter (EnKF) that is able to update the individual positions of agents in an agent-based pedestrian model (ABM) in pseudo-real time. It does this by comparing the pseudo-real positions of the

agents (the 'observations') to the equivalent positions that are being predicted by an ensemble of agent-based models, updating the positions accordingly. The paper demonstrates that the ABM-EnKF system performs considerably better at predicting the positions of the pedestrians than the ABM is capable of doing in isolation.

Similar attempts to use data assimilation to optimise an ABM in real time have often used variations on the particle filter (8; 21; 22; 23). However, a drawback with the particle filter is that it typically requires a very large number of particles to ensure that the space of all possible model states is adequately covered, making it extremely computationally expensive. Although a direct comparison with similar work is difficult as the systems and models are different, here an ensemble of only 100 models was adequate to reliably capture the behaviour of the pedestrians in the pseudo-real system. In other work tens of thousands of ensemble members ('particles') were required (22)

Keiran can you say anything about computation time? Or something mathsy to back up what I have said? Otherwise this paragraph is a little vague

A drawback with the current approach is that it uses the same ABM to generate 'real world' observations (termed *pseudo-true* data) and to predict the behaviour of the system. This means that there is no model discrepancy; i.e. no uncertainty between the data generating mechanism and the simulation. In reality there will always be a degree of model discrepancy and this will undoubtedly influence accuracy of the predictions of the EnKF. The use of pseudo-truth data is common for exploratory studies such as this (see for example 21?) and has the advantage that we can precisely attribute any uncertainty to its underlying causes, but leaves open the question of how well might this approach work in the real world. Immediate future work will begin to test this by incorporating data that are derived from the movements of real people such as **(author?)** (31)

Any other advantages / disadvantages to mention?

Funding

References

[1] Epstein J, Axtell R. Growing Artificial Societies: Social Science from the Bottom Up. Brookings Institution Press; 1996.

[2] Bonabeau E. Agent Based Modeling: Methods and Techniques for Simulating Human Systems. Proceedings of the National Academy of Sciences. 2002;99(90003):7280–7287.

[3] An L, Grimm V, Sullivan A, Turner II BL, Malleson N, Heppenstall A, et al. Challenges, Tasks, and Opportunities in Modeling Agent-Based Complex Systems. Ecological Modelling. 2021;457:109685.

[4] Heppenstall AJ, Crooks AT, See LM, Batty M, editors. Agent-Based Models of Geographical Systems. Springer; 2012.

[5] Thiele JC, Kurth W, Grimm V. Facilitating Parameter Estimation and Sensitivity Analysis of Agent-Based Models: A Cookbook Using NetLogo and 'R'. Journal of Artificial Societies and Social Simulation. 2014;17(3).

[6] Filatova T, Verburg PH, Parker DC, Stannard CA. Spatial Agent-Based Models for Socio-Ecological Systems: Challenges and Prospects. Environmental Modelling & Software. 2013 Jul;45:1–7.

[7] Lippe M, Bithell M, Gotts N, Natalini D, Barbrook-Johnson P, Giupponi C, et al. Using Agent-Based Modelling to Simulate Social-Ecological Systems across Scales. GeoInformatica. 2019;23(2):269–298.

[8] Kieu LM, Malleson N, Heppenstall A. Dealing with Uncertainty in Agent-Based Models for Short-Term Predictions. Royal Society Open Science. 2020;7(1):191074.

[9] Swarup S, Mortveit HS. Live Simulations. In: Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems. AAMAS '20. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems; 2020. p. 1721–1725.

[10] Batty M. Agents, Cells, and Cities: New Representational Models for Simulating Multiscale Urban Dynamics. Environment and Planning A. 2005;37:1373–1394.

[11] Helbing D. Agent-Based Modeling. In: Helbing D, editor. Social Self-Organization. Berlin, Heidelberg: Springer Berlin Heidelberg; 2012. p. 25–70.

[12] Xu ML, Jiang H, Jin XG, Deng Z. Crowd Simulation and Its Applications: Recent Advances. Journal of Computer Science and Technology. 2014 Sep;29(5):799–811.

[13] Heesterbeek H, Anderson RM, Andreasen V, Bansal S, De Angelis D, Dye C, et al. Modeling Infectious Disease Dynamics in the Complex Landscape of Global Health. Science. 2015;347(6227):aaa4339–aaa4339.

[14] van der Vaart E, Beaumont MA, Johnston ASA, Sibly RM. Calibration and Evaluation of Individual-Based Models Using Approximate Bayesian Computation. Ecological Modelling. 2015;312:182–190.

[15] Kalnay E. Atmospheric Modeling, Data Assimilation and Predictability. Cambridge University Press; 2003.

[16] Lewis JM, Lakshmivarahan S, Dhall S. Dynamic Data Assimilation: A Least Squares Approach. Cambridge: Cambridge University Press; 2006.

[17] Ward JA, Evans AJ, Malleson NS. Dynamic Calibration of Agent-Based Models Using Data Assimilation. Royal Society Open Science. 2016;3(4).

[18] Oloo F, Safi K, Aryal J. Predicting Migratory Corridors of White Storks, Ciconia Ciconia, to Enhance Sustainable Wind Energy Planning: A Data-Driven Agent-Based Model. Sustainability. 2018;10(5):1470.

[19] Zhang H, Vorobeychik Y, Letchford J, Lakkaraju K. Data-Driven Agent-Based Modeling, with Application to Rooftop Solar Adoption. In: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems. AAMAS '15. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems; 2015. p. 513–521.

[20] Cocucci TJ, Pulido M, Aparicio JP, Ruíz J, Simoy MI, Rosa S. Inference in Epidemiological Agent-Based Models Using Ensemble-Based Data Assimilation. PLOS ONE. 2022;17(3):e0264892.

[21] Wang M, Hu X. Data Assimilation in Agent Based Simulation of Smart Environments Using Particle Filters. Simulation Modelling Practice and Theory. 2015;56:36–54.

[22] Malleson N, Minors K, Kieu LM, Ward JA, West A, Heppenstall A. Simulating Crowds in Real Time with Agent-Based Modelling and a Particle Filter. Journal of Artificial Societies and Social Simulation. 2020;23(3):3.

[23] Ternes P, Ward JA, Heppenstall A, Kumar V, Kieu LM, Malleson N. Data Assimilation and Agent-Based Modelling: Towards the Incorporation of Categorical Agent Parameters. Open Research Europe. 2021;1:131.

[24] Clay R, Kieu LM, Ward JA, Heppenstall A, Malleson N. Towards Real-Time Crowd Simulation Under Uncertainty Using an Agent-Based Model and an Unscented Kalman Filter. In: Demazeau Y, Holvoet T, Corchado JM, Costantini S, editors. Advances in Practical Applications of Agents, Multi-Agent Systems, and Trustworthiness. The PAAMS Collection. vol. 12092. Cham: Springer International Publishing; 2020. p. 68–79.

[25] Clay R, Ward JA, Ternes P, Kieu LM, Malleson N. Real-Time Agent-Based Crowd Simulation with the Reversible Jump Unscented Kalman Filter. Simulation Modelling Practice and Theory. 2021;113:102386.

[26] Tang D. Data Assimilation in Agent-based Models Using Creation and Annihilation Operators. arXiv:191009442 [cs]. 2019;.

[27] Tang D, Malleson N. Data Assimilation with Agent-Based Models Using Markov Chain Sampling. Open Research Europe. 2022;2:70.

[28] Evensen G. The Ensemble Kalman Filter: Theoretical Formulation and Practical Implementation. Ocean dynamics. 2003;53(4):343–367.

[29] Kalman RE. A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering. 1960;82(1):35–45.

[30] Lueck J, Rife JH, Swarup S, Uddin N. Who Goes There? Using an Agent-based Simulation for Tracking Population Movement. In: Winter Simulation Conference, Dec 8 - 11, 2019. National Harbor, MD, USA; 2019. .

[31] Zhou B, Wang X, Tang X. Understanding Collective Crowd Behaviors: Learning a Mixture Model of Dynamic Pedestrian-Agents. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. Providence, RI: IEEE; 2012. p. 2871–2878.