

EPSRC Centre for Doctoral Training in Industrially Focused Mathematical Modelling



Parameter Estimation in Agent-Based Models

Joel Dyer



Contents

1	Introduction	1
2	Representing models as Bayesian networks	2
3	Optimisation-based approaches	4
4	Discussion, conclusions & recommendations	5
5	Potential impact	6

1 Introduction

Over recent decades, and particularly since the financial crisis of 2008, there has been a growing sentiment within the social sciences that a fundamental shift in economic theory is necessary. Classical mathematical models in economics make a number of unrealistic assumptions, including that humans have the foresight, information, and reasoning abilities to understand the full consequences of any decision they take. Such simplifying assumptions were necessary in order that the resulting mathematical models could be solved using simple, explicit methods. However, the rapid growth in the availability of computing power over the past thirty years has gone some way in making such assumptions unnecessary. This has opened up the arena to new models previously dismissed on the basis of their complexity and analytical intractability; models that are more loyal to the true nature of human behaviour and the complex adaptive system that is human society, within which both local and global economies are embedded. The need to develop, understand, and equip policy-makers with such models has never been so apparent to economists as it has been since the financial crisis of 2008 – largely considered to be the worst economic disaster since the Stock Market Crash of 1929. Many economists believe that the effects of the crisis could have been mitigated had policy-makers had access to models that more faithfully represented the behaviours and nature of economic agents. It is for this reason that there is significant interest within the social sciences in the study of **agent-based models** for a broad range of social systems.

Conventional models in economics have necessarily made vastly unrealistic assumptions about human behaviour

What is an agent-based model?

At its most basic level, an ABM consists of the following components:

1. a set of autonomous and possibly heterogeneous decision-making entities – the model's **agents**;
2. the **relationships** and **interactions** between the agents of the model;
3. an **environment**, if such a component is relevant for the purposes of the model.

Agent-based models can be used to simulate directly the agents in an economy (e.g. people, firms, banks, or governments) and their interactions.

An agent in an economic model can be any animate object: an investor, a bank, a government etc. The agents we choose to include in an ABM depend on the aspect of an economy we are interested in. Agents interact with each other and the environment, if one is defined, according to: (i) a set of rules governing the information agents have access to, (ii) the procedure by which the agents process this information, and (iii) and the course of action the agents consequently choose.

Finally, the environment within which the agents interact may be explicitly defined, if it is of relevance to the model – for example, it could be a spatial environment containing obstacles, natural resources, or transportation networks – or an explicit environment may not be defined, in which case only the abstract relationships between the agents are deemed relevant.

Challenges in agent-based modelling


Agent-based models do not generally have explicit solutions: they are so complex that they have to be implemented computationally and defined in terms of an algorithm. As with many computer programs, a problem with ABMs – which becomes more true the more complex the models become – can be the time it takes to run the simulation, known as the computational cost. This can restrict the ability of policy-makers to use them in “what-if” scenario analyses, which will typically involve multiple simulations of the model. A further challenge is that often the answer to the question “What initial conditions should be imposed?” is not easily obtained, and is important to address since different starting conditions can lead to vastly different dynamics. This typically involves the collection and use of real-world data to pin down accurate starting conditions.

Agent-based models are expensive to run and are hard to initialise in such a way that the true state of the world is reflected accurately.

A third challenge with ABMs is that of **parameter estimation**. Parameters in a model are introduced to characterise the relationship between variables. For example, we might have a model of the economy built around the statement that “lifetime earnings increase linearly with the number of years spent in education”. This model includes a parameter that represents the slope of this relationship, and we need to estimate a value for this parameter on the basis of some real-world data.

Agent-based models involve many parameters for which values must be chosen. This is the problem of parameter estimation.

Our aim is to explore a selection of approaches to performing parameter estimation for models that generate time series that are of interest to Improbable, a games company



dedicated to the generation of large-scale complex simulations and virtual worlds for the purpose of assisting real-world decision making.

Glossary of terms

- **ABM:** An agent-based model, or the practice of agent-based modelling.
- **Time series:** A sequence of values in which each successive value is a later record of the same quantity.
- **Random variable:** A mathematical variable whose true value is unknown or changes randomly over time.
- **Probability density function:** A distribution that gives the probability of observing a random variable to assume a value within some range of values.
- **Markov chain:** A model that generates a time series in which the probability of observing a given value in the current time step depends only on the value observed in the previous time step.
- **Bayesian network:** A Bayesian network is a network representation of the random variables in a mathematical model.
- **Autoregressive model:** A sequence of variables in which the current value depends randomly on a linear combination of some number of previous values.
- **Probabilistic programming languages:** A class of languages that enable the construction of probabilistic models (models involving one or more random variables), inference on which may then be performed automatically.
- **Auxiliary variable:** A random variable in a Bayesian network that is neither observed nor of interest to the inference task.

2 Representing models as Bayesian networks

We consider a real-world process that we observe periodically and from which we generate a time series. We suppose that we have some idea about the underlying mechanisms that generate this time series. Suppose further that we construct a mathematical model on this basis, within which we express unknown quantities as random variables, and introduce parameters that control the relationship between these variables. One way to represent this model is with a **Bayesian network**. A network consists of a set of objects (*nodes*) and the relationships between them (*links*): in a Bayesian network, these objects are all of the variables and parameters of our mathematical model, and the links indicate the direct dependence of a random variable on another random variable or parameter.

As a simple first example of this framework, we consider an **autoregressive model** of order $p = 2$, herein called **AR(2)**. In this model, there is a time series X_1, X_2, \dots, X_T that we observe, and we assume that the mechanism underlying the generation of this sequence of observations depends randomly on the linear combination of the two previous values in the sequence, with the outcome drawn from a Gaussian probability density function centered on this linear combination. The model includes 3 parameters: the two coefficients of the linear combination (ϕ_1 and ϕ_2) and the standard deviation σ_ϵ of the Gaussian probability density function. Since the X_t depend directly on these parameters, we can represent this probabilistic mathematical model with a Bayesian network; we show this representation in Panel (a) of Figure 1.

Many members of a class of programming languages – probabilistic programming languages – enable the construction of Bayesian networks. They further have algorithms in place to help answer questions such as “Given some initial belief for the values of the parameters, and also given a set of observations, what probability should *now* be assigned to the parameters taking on certain values?” Mathematically, this involves using **Bayes’ rule**, which relates the updated beliefs – captured by the **posterior** probability distribution – to the old beliefs – captured by the **prior** probability distribution – after including new information from our observations, encoded in the **likelihood** function.

In our first set of simulations, we attempt to recover the true values of parameters ϕ_1 , ϕ_2 , and σ_ϵ^2 given $T = 200$ observations from an AR(2) process by representing the process with a Bayesian network. Using the observations generated by some AR(2) process with

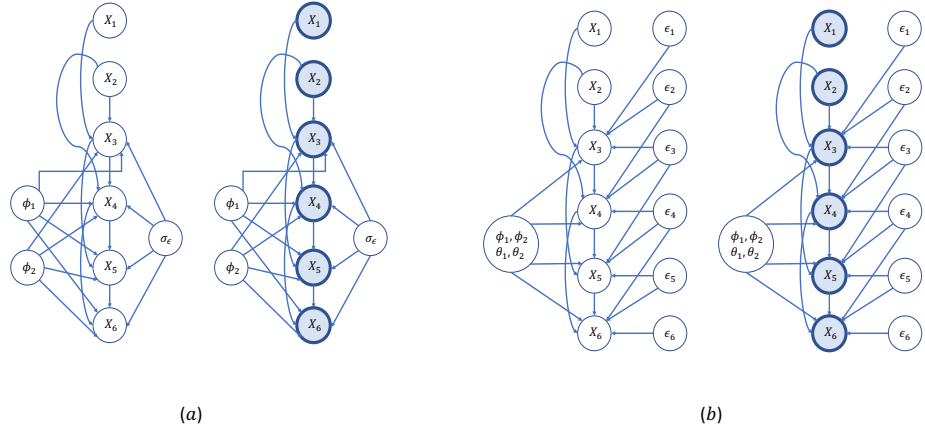


Figure 1 – Panel a): Representation of the AR(2) process as a Bayesian network (left), and the network after observing a time series: shaded nodes indicate observed variables (right). Panel b) Representation of the ARMA(2,2) process as a Bayesian network (left), and the ARMA(2,2) Bayesian network following observation of a time series (right). The four scalar AR and MA coefficients are represented as a single multivariate node to reduce clutter.

known parameter values, we construct the likelihood function for this process and these observations in a probabilistic programming language and sample from the posterior distribution of the parameters, which is proportional to the product of the likelihood and the parameter prior distributions. In Figure 2, we show the posterior distributions for the three parameters having initially stated, via our prior distributions, that we believe ϕ_1 , ϕ_2 , and σ_ϵ lie in the ranges $[0, 1]$, $[0, 1]$, and $[0, 2]$, respectively, with equal probability. We show the mean, median, and maximum value of the distributions with orange, green, and red lines, respectively, and the true parameter values by the black vertical lines. We see that representing the process with a Bayesian network and making use of Bayes rule has enabled us to recover the true values quite accurately.

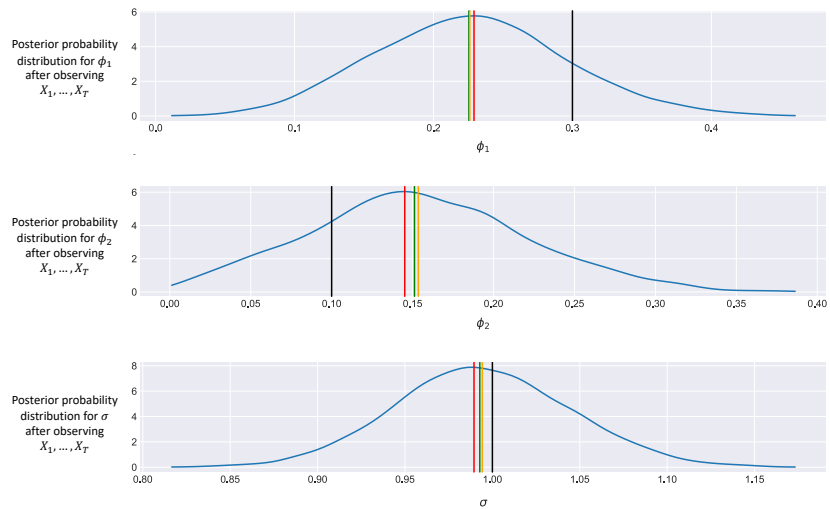


Figure 2 – Posteriors for the three parameters in our AR(2) model (blue lines). The true parameter values, distribution mean, distribution median, and distribution maximum are indicated by black, orange, green, and red lines, respectively.

To more fully understand the merit of this representation, we increase the model complexity; a natural progression to the previous model being a mixed **autoregressive**

We test the ability of a Bayesian network representation to facilitate easier estimation of model parameters.

moving-average model of order (p, q) , herein called $\text{ARMA}(p, q)$. In this model, there are two sequences: the time series of interest (X_1, \dots, X_T) as before; and another $(\epsilon_1, \dots, \epsilon_T)$ for which each value in the sequence is drawn randomly from a Gaussian distribution with mean zero and some user-defined variance. We refer to this sequence as the *noise sequence*. We choose $p = 2, q = 2$, so that each X_t depends randomly on a linear combination of the previous two values in that sequence (the autoregressive component), plus a linear combination of the previous two values of the noise sequence (the moving average component), plus the current value in the noise sequence. In this experiment, we wish to recover the true values of the coefficients ϕ_1 and ϕ_2 of the linear combinations for the autoregressive component, as well as the coefficients θ_1 and θ_2 for the linear combination in the moving average component, given a time series generated by an $\text{ARMA}(2,2)$ process as the observations we use to construct the likelihood function.

To do so, we once again construct the corresponding Bayesian network, illustrated in Panel (b) of Figure 1, with a probabilistic programming language. We assert via our prior distributions that we initially believe they each lie in the range $[0, 1]$, and construct the parameter posterior distributions as the product of the likelihood and the prior distributions. We show these posterior distributions in Figure 3. In this simulation, we see greater disagreement between the three estimates of the true value, with the mean and median generally recovering the parameters more accurately than the maximum. We also report far greater difficulty in obtaining accurate estimates of the true values in this only-slightly-more complex model. This is in part due to the presence of the ϵ_t , neither observed nor of interest to us: so-called **auxiliary variables**. Such variables will be plentiful in models of even greater complexity, and the difficulty that we experienced of estimating in their presence does not bode well for the generalisability of this approach.

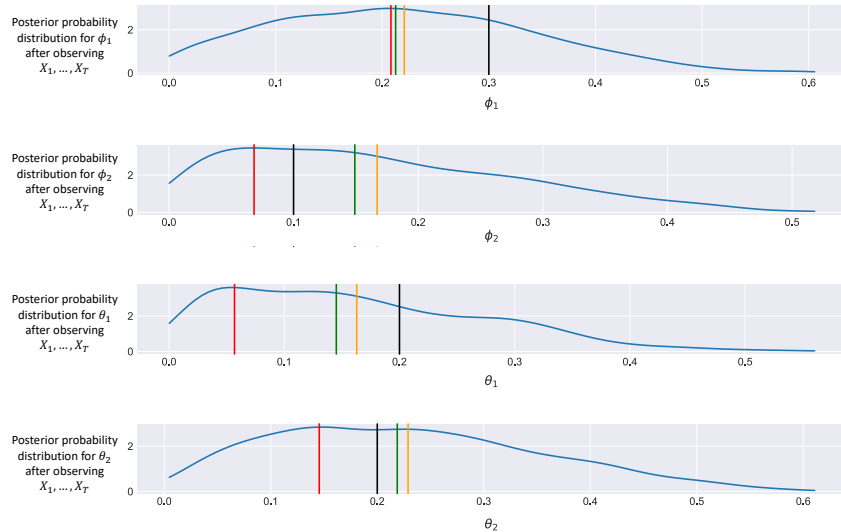


Figure 3 – Posterior distributions for the four parameters in our $\text{ARMA}(2,2)$ model (blue lines). The true parameter values, distribution mean, distribution median, and distribution maximum are indicated by black, orange, green, and red lines, respectively.

3 Optimisation-based approaches

An **optimisation problem** is the mathematical problem of finding the location of the minimum value of some function $\mathcal{L}(\psi)$, where the values of ψ are restricted to the set of allowed values Ψ . We cast the problem of estimating parameters for an ABM as an optimisation problem: we seek input parameters to our ABM such that some properties of our simulated time series, written $X_{1:T_S}$, matches corresponding properties of some real-world data $y_{1:T}$, whose data-generating mechanism we model with our ABM. While many choices of loss function exist, we compare three that capture the following ideas:

KL Ideally, we would like to find input parameters ψ for our ABM such that the probability of observing $y_{1:T}$ in the real world *exactly* matches the probability of

We also consider a handful of optimisation-based methods for recovering the parameters of time series-generating models.

observing it as output from our ABM; that is, **their likelihoods match**. However, there are two problems: ABMs are typically so complex that it is in practice impossible to obtain their likelihoods, making it impossible to compare to the real world; and, more importantly, we do not know the real-world likelihood to begin with. We address these problems by assuming that some simpler, known likelihood function, whose shape is controlled by parameters ω , adequately approximates the *true* likelihood for our ABM, and then determine the ψ that maximises the likelihood of observing $y_{1:T}$ under the approximating likelihood.

AC One limitation of the previous method is that we restrict ourselves to a class of distributions that may be summarised with parameters ω , which is a small subset of the full range of distributions available. A more flexible approach is one in which no parametric form is assumed for the distribution of data: a **non-parametric** approach. One way to do this is by comparing the real-world data $y_{1:T}$ to the ABM output $X_{1:T_S}$ according to the similarity of the time series to its past self. If we are able to find ABM input parameters such that the similarity of the real and simulated data sets to their past selves match well, this may be sufficient criteria to label the ABM as “calibrated”.

L2 Finally, we consider the average of the **squared-differences** between the real and simulated data at each time step. An advantage is that this captures trends in the data; however, here it serves primarily to set an absolute scale for the performance of the previous loss functions.

The first simulation we perform for the purposes of comparing these three methods proceeds as follows. First, we generate “real-world” data $y_{1:T}$ from an ARMA(2,2) model. Then, we choose the ARMA(2,2) model as our “complex ABM” that generates time series $X_{1:T_S}$, for which we wish to estimate values for the autoregressive coefficients. We then use the AR(2) likelihood as the “simpler model with known likelihood” required for the purposes of the KL loss. Finally, we search for the ABM input parameters that minimise each of the losses. In Panel (a) of Figure 4 we show boxplots summarising the distribution of two quantities (the L_1 error and the L_2 error) that measure the inaccuracy of the best estimates for each loss function over many trials. We see that the estimates obtained by minimising the KL and AC losses are comparably accurate, and both significantly outperform the L2 loss.

In a second experiment, we consider a more complex model. We propose a model of **opinion dynamics** (OD) in a social network, in which the agents’ opinions evolve according to their friendship circle and its evolution. We follow a similar procedure. First, we generate real-world data $y_{1:T}$ from the OD ABM. Then, we use the OD ABM as our “complex ABM” that generates time series $X_{1:T_S}$ and for which we seek estimates for two parameters governing the evolution of the agents’ opinions. Next, we propose a Markov chain as the “simpler model with a tractable likelihood”, required for the KL method. Finally, we search over some Ψ for the ABM input parameters that minimise each of the losses. In Panel (b) of Figure 4, we show the distribution of the two inaccuracy measures for each loss function over over many estimation attempts. We see that all three methods perform comparably, with AC performing best on average.

4 Discussion, conclusions & recommendations

We have considered the problem of estimating the parameters governing a set of time-series generating models as a first step in the problem of parameter estimation for agent-based models. The methods we considered consisted of a Bayesian approach – representing a probabilistic model as a Bayesian network – and optimisation-based approaches – seeking parameter values that minimise some loss function. We performed several sets of numerical experiments to explore which methods might work well.

We observed that, while it was possible to recover accurate estimates for the autoregressive coefficients of AR(2) and ARMA(2,2) models, the presence of auxiliary variables in the latter case made obtaining estimates computationally difficult. This is a computational challenge broadly true for Bayesian approaches to learning, and a limitation that exists alongside further general issues with Bayesian network representations of ABMs, including the requirement that the size of the network be known at compile time, which is unsuitable for many ABMs involving processes such as

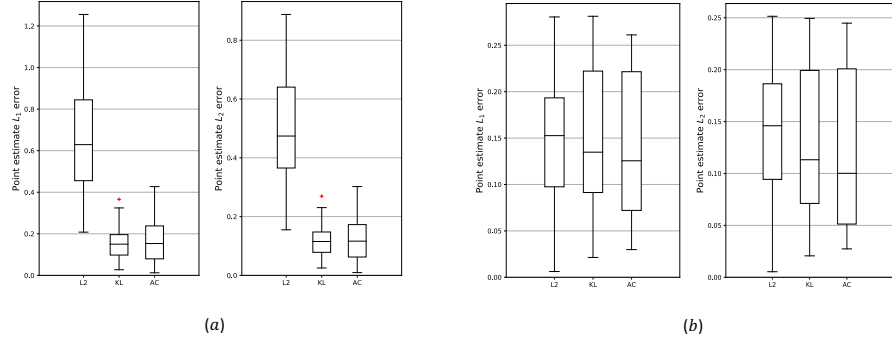


Figure 4 – Boxplots for the distribution of the inaccuracy of estimates obtained with loss functions KL, AC, and L2, described in Section 3, for the ARMA(2,2)-AR(2) model combination (Panel (a)) and OD-Markov chain combination (Panel (b)). The white boxes, divided into two parts by a horizontal line, indicate the values of three common quantities describing distributions of data. The bottom edges of the boxes indicate the first quartile, Q_1 , of the data set: the value in the data set below which 25% of the data lies. The top edges of the boxes indicate the value of the third quartile, Q_3 , of the data set, below which 75% of the data lies. The horizontal bars dividing the boxes indicate the median value of the data set: the data point below which 50% of the data lies in value. The horizontal lines bounding the vertical lines below and above the boxes indicate the lowest and highest values of the data set that lie within $1.5(Q_3 - Q_1)$ of Q_1 and Q_3 , respectively. Red stars indicate any points in the data set outside of this range.

birth-death dynamics that make the size of the graph unknown *a priori*. We also compare the ability of three loss functions to recover the true parameter values for an ARMA(2,2) process and a simple ABM of opinion dynamics. We observed that two of these losses – KL and AC – were able to recover parameter values with comparable accuracy in the first case, though they performed only marginally and inconsistently better than a naive mean-of-squared-differences loss in the case of a true ABM.

However, the work presented here is not conclusive, and both sets of approaches to parameter estimation warrant further investigation. In particular, it would be informative to explore the robustness of the KL method to the choice of “simpler model with known likelihood” used for a given “complex ABM”. Further, exploring alternative methods for mapping from the ABM input parameters to the simple model’s likelihood parameters may enhance the performance of this method. Finally, and more broadly within the parameter estimation problem, the development of a set of standardised tests to determine whether the parameters for an ABM can be estimated in principle will be invaluable to modellers seeking to understand the parameter space of their ABMs.

5 Potential impact

Our work will form the basis for further research on ABM calibration methods. Such work will better enable researchers, both within and outside Improbable, to develop empirically-validated models of social systems, increasing their usefulness to policy-makers.

Dr. Christoforos Anagnostopoulos, Head of Research at Improbable, commented:

“Joel quickly picked up a number of novel concepts and enthusiastically contributed to the research pursued, in particular by pursuing a last-minute pivot from an ecological ABM to an opinion dynamics one which greatly improved the suitability of the methods under investigation to the use case in question. He ran a very large number of numerical experiments that offered us confidence that there is scope for further work with possible industrial and scientific impact in the study of calibration techniques for disinformation modelling, a topic which is very timely and important.”