The curse of possibilities

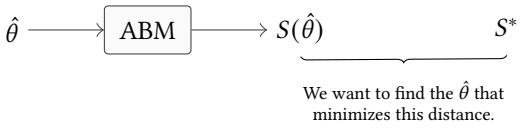
May 12, 2023

Agent-based models have inputs and outputs:

Some models are meant to represent the world. In those cases, we would like the inputs of our model to correspond to some sort of initial condition in the real world and the outputs to correspond to an outcome that can be measured in both world and model. In practice, inputs and outputs are often (but not always) sets of numeric values: a set of parameters θ for the inputs and a set of summary statistics S for the outputs.

The values of some parameters might be known to us, whether it's the hold size of a particular fishing vessel, the maximum running speed of a grey wolf or the base interest rate of the Bank of England in May 2023. We can change those values if we want to explore counterfactual scenarios, but they are still bound by empirical data or by our assumptions. Other parameters, however, are "free": we do not have any direct way of establishing their values. Sometimes that's because the parameter's empirical value is difficult to measure (e.g., the "attraction rate" of a fish-aggregating device) or because the parameter concerns a more theoretical part of our model, for which it is difficult to pluck a value directly from the real world (e.g., the learning rate of a reinforcement-learning agent).

Even if we can't estimate free parameters directly, all is not lost: we can try using what we know about outcomes in the real world to work our way back to a set of parameter values. This task, usually called *calibration* in the ABM community, can be tackled in different ways [10], but the general idea is to find a vector of estimated parameters $\hat{\theta}$ that minimizes the distance between model output for those parameters $S(\hat{\theta})$ and observed summary statistics S^* (i.e., the error of the model):



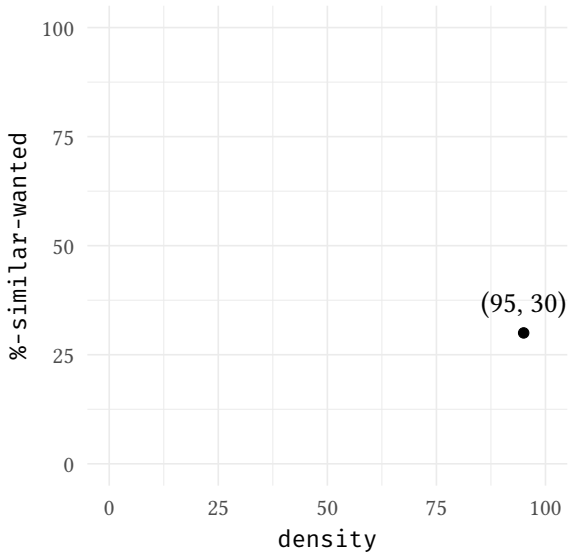
Calibration is a tricky topic, and the best way to do it varies from model to model [1], but this simplified picture gives us a way to think about the various mathematical spaces that we are interacting with during the modelling process. We will develop

these ideas by looking at the NetLogo [12] version [13] of the Schelling model of segregation [9, 8]. *Caveat emptor*: the original Schelling model is a conceptual model, not meant to be empirically calibrated.[1] It is nevertheless useful as an example because it is both simple and well known.

NetLogo's model library contains a Schelling-like model using a movement rule that's different from the original.[2] It is called *Segregation* and has two parameters, listed here with their default values:

Parameter	Value	Description
`density`	95	The percentage of grid cells occupied by agents.
`%-similar-wanted`	30	The agents' desired percentage of neighbours of the same colour as them.

That gives us our initial θ, which in this case is simply a tuple of values: $(95, 30)$. It's easy to see how each possible θ gives us a point in the *parameter space* of the model:



Parameter space is not the only space we're dealing with: there is also the *state space* of the model. In the case of the *Segregation* model, each agent has three relevant[3] variables: `xcor`, `ycor` and `color`. Each of these variables, for each agent, adds

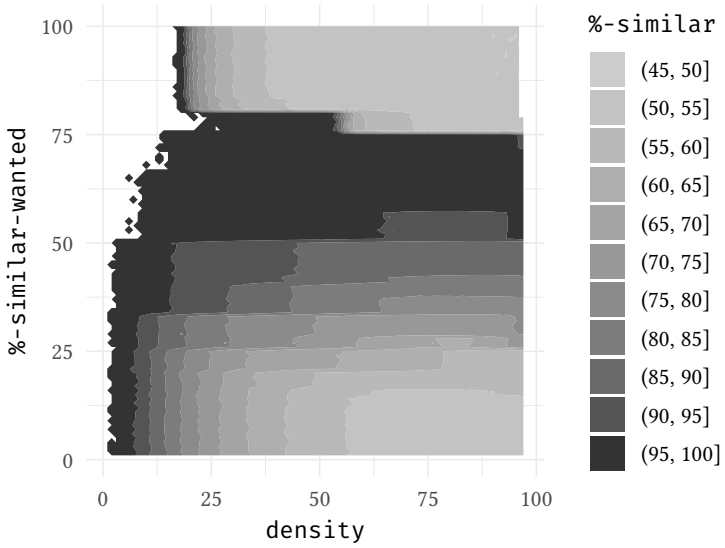
[1] There are plenty of Schelling-inspired empirical models out there. See [15] for a recent example.

[2] A detail which is not completely unrelated to the point that we will be trying to make in this paper.

[3] NetLogo defines various other agent variables that complicate things a bit, like `heading`, `label`, or even `pen-size`, but those can be safely glossed over for now.

a dimension to the model's state space.[4] Running a simulation can be thought of as following a trajectory through the model's state space.

Summary statistics (S) are just a pragmatic way of looking at the model's state space, making it easier to interpret and relate to the real world (since we probably don't have access to the whole state space of our empirical target anyway). In the *Segregation* model, the one relevant summary statistic is `%-similar`. It's defined as the sum of the count of similar neighbours for each agent, divided by the sum of the count of all neighbours for each agent. Imagine that we have an empirical S^* (even if we don't). Since our parameter space is so small, we can brute force our way through it and find the $\hat{\theta}$ that will produce S^*. Easy, right? Look at the results:



Two things should be noticed from the above plot. First, different regions of the parameter space can produce very similar results. If you're looking for a `%-similar` between 70 and 75, many combinations of `%-similar-wanted` and `density` will give you that, so can't know what particular combination occurs in the real world. That's the problem of parameter *identification*. If this was a proper empirical model, we could avoid it by obtaining `density` from data. But then again, if this was a proper empirical model, we would have more than two parameters. Parameter identification is a long-acknowledged challenge [4, 7] that we are not going to confront head-on but that should be kept in the back of the reader's mind while we move along.

The other—perhaps even more obvious—thing is that we are missing results for very low values of `density`, especially when `%-similar-wanted` is high. That is a consequence of the way `%-similar` is defined in the model: it divides the number of similar neighbours by the total number of neighbours. At low densities, it's possible for every agent to be completely isolated, in which case there are no neighbours at all

[4] Things get hairier when a model dynamically creates or remove agents, but let us not think about that too hard.

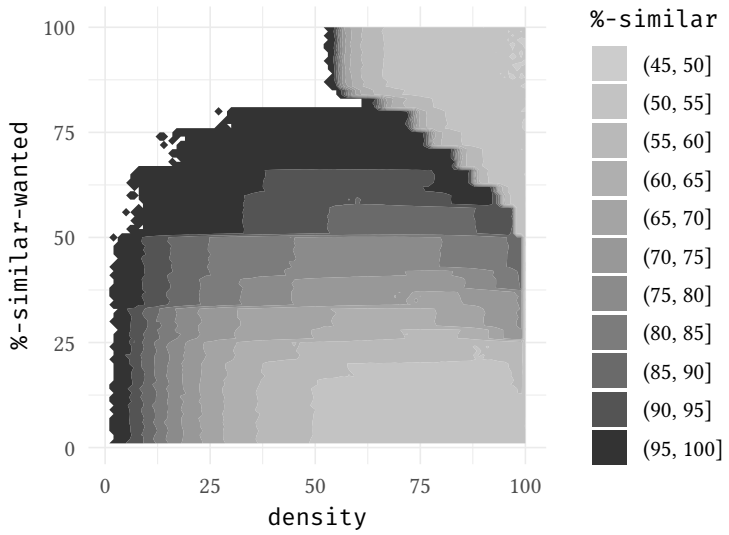
and we end up with a division-by-zero error.[5] This explains why the `density` slider has a lower bound of 50% in the model's interface. Not every region of the parameter space is relevant to a model's analysis, but we tend to shy away from pushing against those boundaries because of the complications they introduce.

Speaking of complications: we have pretended so far, for the sake of simplicity, that *Segregation* is a two-parameter model, but is it really? One nice thing about NetLogo is that it comes with a predefined world ready to be inhabited by your agents (or "turtles", in its parlance): a continuous two-dimensional space divided in "patches". You can jump right into modelling without getting bogged down in the details because some choices have already been made for you. The world, by default, measures 33×33 patches and the space is toroidal (i.e., doughnut-shaped; you can wrap around vertical and horizontal edges). In the *Segregation* model, world size has been set instead to 51×51 and world wrapping kept on. Does it matter? The results might not be too different on a 33×33 grid, or even 100×100, but what about 3×3, or even 50×1? Topology might matter too: in a non-wrapping world, corner agents would only have three potential neighbours instead of eight. Trivial details, maybe—or fun things to try, depending on how you see it—but both world-size and topology are very much a part of the model's parameter space even if you choose not to explore it. We tend to think of model parameters as either numeric (integer or floating-point values), logical (boolean values) or sometimes enumerated (i.e., one of a set of possible choices, often implemented as strings), but this is just the tip of the iceberg.

We mentioned above that NetLogo's *Segregation* model uses its own movement rule: an unhappy turtle picks a random heading and moves forward until it finds a free spot (whether or not it's happy there). Schelling's original rule is that: "an individual discontent with his own neighborhood moves to the nearest vacant spot that surrounds him with a neighborhood that meets his demands." [8] Running the model with Schelling's rule gives us results that are similar but not quite the same:[6]

[5] I had to modify the *Segregation* model to output `"NA"` instead of crashing when running it through *BehaviorSpace*. I ran 50 iterations per condition and report the mean, excluding runs that resulted in NA values.

[6] This is not the point but I can't help to notice how moving to the closest spot that satisfies an agent's demands (which can very well be a spot with no one else around) is much more likely to generate NA values than moving the the first free spot in a random direction (which is much more likely to be right next to someone else).



This is not surprising. Both movement rules support Schelling original insight about how relatively low homophily can still lead to high segregation, so maybe it doesn't matter all that much. But, again, put yourself in the shoes of an empirical modeller trying to calibrate their model: which rule should you use? Perhaps you should try both: make the rule an extra parameter of your model. Congratulations: it now takes twice as long to explore your model's parameter space and your identification problem just got worse: if both rules can generate S^*, which one should you pick? What about the very large number of other rules that could have been used instead? And that's just movement. We haven't said anything on model initialization,[7] updating order, or even how "distance" is defined.[8] All those things are also parameters.

Note how the distinction between "fixed" and "free" parameters applies here too. Suppose you tell me that you don't need a parameter for a certain agent behaviour because we know with great confidence how real-world agents behave in such situations. That's great, but that's no different from knowing the population density because you are modelling a real-world city: it's still a parameter, it's just that you happen to know the value for that particular one.

The point we have been building towards is this: the distinction between parameter space and program space (or "model space", "design space", "specification space") is arbitrary. It is a scary thing to contemplate: it means the parameter space of any model can never be fully explored. We are cursed with effectively infinite possibilities and in order to achieve anything we are forced to make choices, often based on the flimsiest of reasons, which carve down model space to a tiny shell of its expansive self and banishes the rest of it to the realm of the unthinkable.

[7] To quote Schelling himself: "It might make sense to distribute the blank spaces evenly, but I let them be determined at random. (It makes some difference.)" [8, p. 155]

[8] I realised after the fact that Schelling uses Manhattan distance whereas I implemented his rule using Euclidean distance. I do not know if it makes any difference.

Let's take a breath and think about the consequences. Nothing has been said in this paper that is not already known and agent-based modelling is alive and well, so maybe we just keep on keeping on? That's not an unreasonable suggestion. The curse of possibilities is a hump we can get over, and I believe that is something we see in our personal trajectories as modellers. As a beginner, you are unaware of the problem. You just want your model to work, and once it does, you're quite happy to just explore the parameter space as it is. As you gain experience, you start seeing all the possibilities, agonizing over every choice. It's not unlike the old fable of the fox and the cat: the fox is bragging to the cat about how many cunning escape tricks he knows, whereas the cat has only one trick which is to climb up a tree. When the hounds come, the cat immediately gets up to safety but the fox fails to decide which of his many tricks to use and gets caught. With experience, we learn to go back to slightly more cat-like behaviour. We develop intuitions about what works and what doesn't and we become acutely aware of how limited resources (both human and computational) are. Deadlines are looming and papers need to get out so we just call the shot: this is my model, those are its parameters; I'm going to calibrate it and do my sensitivity analysis and everything will be fine. What else could I possibly do?

If there is one call to action in this essay, it is this: at least be explicit. Do not turn a blind eye and hope others will too. It's OK to make arbitrary choices; there is no way around it, but those choices still matter and need to be documented. Chattoe-Brown [2] conducted a review of papers published in JASSS between 1998–2022 using the terms "random movement" or "move randomly". He found that the majority of those papers did not fully explain what that meant and that those that did all meant different things. Yet, as he shows using NetLogo's *Wolf-Sheep Predation* model [14], and as we have seen with Schelling, movement rules affect results.

So should we stop here, at the acceptance stage of grief, and be content in acknowledging the arbitrariness of our modelling choices? I believe we can do better and that the way forward is not just to be explicit, but to be *formally* explicit.

Let's go back to Schelling and consider movement rules. Once we have decided that agents should move to a free spot if they are unhappy, we can think of any possible movement rule as a *function*: it takes as an input the current state of the model (composed of the position and colour of every agent) and outputs the coordinates at which the agent wishes to move.[9] Since possible model states are finite and possible choices also finite, there is a finite (though massive) number of possible mappings between them. And that goes for most behaviours in an agent-based model: if we formally define a behaviour as a function with explicitly typed inputs and outputs, we can get the computer to generate possible implementations of those functions. This becomes part of the calibration process: we can automate the exploration of functional space, just like we automate the exploration of numeric parameter dimensions.

This possibility has recently received attention in the agent-based modelling community under the "Inverse Generative Social Science" (iGSS) moniker [3]. That methodology has even been applied to the Schelling model itself, in a much more sophisticated way than what I am discussing here [5]. The idea that we can use a computer

[9] As an inside: NetLogo subtly encourages modellers to change the state of their agents directly—the *Segregation* movement rule is a good example of that. It's arguably a very intuitive way of thinking, but it also deprives us of the abstraction opportunities that come from modelling behaviours as pure functions.

program to generate computer programs is not new. It was first really developed by John Holland in the 1970s [6] but, like many things in computer science, it dates back to Turing [11].

So if it's an old idea and ABM practitioners are already using it, what am I advocating for? I think that we are on the right track, but there are two things we still need in order to make more progress. The first and most pressing thing is a proper conceptual framework. We can talk about collapsing spaces and how it's all really just the same, but that only muddies the waters. A proper mathematical treatment of these questions would set the stage for the second thing which I believe is needed: better technical tools. At this point you might be recoiling in horror and asking: you are not really suggesting yet another ABM platform, are you? A fair question. I am agnostic as to whether a new platform is required, but there are a few features that I think we are missing.

We need to be able to formally identify the parts of our model that we consider to be parameters and the parts that we don't. This might seem counter-intuitive, as I have just advocated that this delineation is arbitrary, but *au contraire*: it just makes it all the more important to be clear about it. These parameters could be defined as scalar types (i.e., numeric, logical or enumerated) or as functions with their input and output types. Furthermore, for each of those parameters, it should be explicitly stated whether they are fixed or free, and in the latter case what their bounds are. The same care should be taken in defining summary statistics for the model, with the possibility of stating which ones have a corresponding empirical target and which ones are just informational. Again, nothing new here: this is all just good practice. It's also something that our current tools are not very good at helping us with: it requires discipline and experience when, ideally, it should happen almost effortlessly.

One way to think about all this is that we should provide metadata about our model. It would help with documentation, but it would also help with *automation*. Once we know all the parameters with their bounds and all the summary statistics with their targets, we can automate calibration and sensitivity analysis. Once we know the functional definition of our model's behaviours, we can automate the generative search for possible implementations. Extensibility and interoperability should both play an important role in this brave new world. We cannot presume that current methods for calibration, sensitivity analysis and, especially, function generation are the be all and end all and could all be packaged in a static library. New plugins should be easy to write and external libraries (especially those written in Julia, Python and R) easy to leverage.

With such a tool in hand, maybe we could be a bit more like the fox again.[10] We could afford to embrace a much wider space of possibilities without getting paralysed by it. We could focus on the things we really know how to model (there will always be at least a few) and let the machine deal with the things we don't, in a more efficient way that we ever could. It would not solve all our problems, but it would partly lift the curse of possibilities.

[10] I seriously considered *From cat to fox to cat, then back to fox again* as an alternate title for this essay.

References

[1] Ernesto Carrella. "No Free Lunch When Estimating Simulation Parameters". In: *Journal of Artificial Societies and Social Simulation* 24.2 (2021), p. 7. ISSN: 1460-7425.

[2] Edmund Chattoe-Brown. "All The Right Moves? Systematically Exploring the Effects of Random Travel in Agent-Based Models". In: *Advances in Social Simulation. Proceedings of the 17th Social Simulation Conference, 12–16 September 2022*. Springer Proceedings in Complexity. 2023.

[3] Joshua M. Epstein. "Inverse Generative Social Science: Backward to the Future". In: *Journal of Artificial Societies and Social Simulation* 26.2 (2023), p. 9. ISSN: 1460-7425.

[4] Franklin M. Fisher. *The Identification Problem in Econometrics*. McGraw-Hill, 1966.

[5] Chathika Gunaratne et al. "Generating Mixed Patterns of Residential Segregation: An Evolutionary Approach". In: *Journal of Artificial Societies and Social Simulation* 26.2 (2023), p. 7. ISSN: 1460-7425.

[6] John Henry Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press, 1975. ISBN: 978-0-472-08460-9.

[7] Charles F. Manski. *Identification Problems in the Social Sciences*. Harvard University Press, 1995. ISBN: 978-0-674-44284-9.

[8] Thomas C. Schelling. "Dynamic Models of Segregation". In: *The Journal of Mathematical Sociology* 1.2 (1971), pp. 143–186.

[9] Thomas C. Schelling. "Models of Segregation". In: *The American Economic Review* (1969), pp. 488–493.

[10] Jan C. Thiele, Winfried Kurth, and Volker Grimm. "Facilitating Parameter Estimation and Sensitivity Analysis of Agent-Based Models: A Cookbook Using NetLogo and R". In: *Journal of Artificial Societies and Social Simulation* 17.3 (2014), p. 11. ISSN: 1460-7425.

[11] A. M. Turing. "Computing Machinery and Intelligence". In: *Mind* 59.236 (1950), pp. 433–460. ISSN: 0026-4423. JSTOR: 2251299.

[12] Uri Wilensky. *NetLogo*. Center for Connected Learning. 1999.

[13] Uri Wilensky. *NetLogo Segregation Model*. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL, 1997.

[14] Uri Wilensky. *Wolf Sheep Predation Model*. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL., 1997.

[15] Carolina V. Zuccotti et al. "Exploring the Dynamics of Neighbourhood Ethnic Segregation with Agent-Based Modelling: An Empirical Application to Bradford, UK". In: *Journal of Ethnic and Migration Studies* 49.2 (Jan. 2023), pp. 554–575. ISSN: 1369-183X, 1469-9451. DOI: 10.1080/1369183X.2022.2100554.