# Guiding Parameter Estimation of Agent-Based Modeling Through Knowledge-Based Function Approximation

William Broniec, Sungeun An, Spencer Rugaber and Ashok K. Goel

*Design Intelligence Laboratory, School of Interactive Computing, Georgia Institute of Technology, 85 Fifth Street NW, Atlanta, GA 30308, USA*

**Abstract**

Parameter estimation is a common challenge in scientific modeling. However, agent-based modeling offers particular challenges: since the system behavior emerges out of local interactions among agents, many solutions are computationally intensive and do not scale with the number of parameters. The challenge is especially acute in interactive agent-based modeling where the goal is to support humans with little domain expertise. We describe a knowledge-based function approximation technique for the problem of parameter estimation in interactive agent-based modeling. Our method uses domain knowledge to decompose a large parameter search space into smaller and simpler spaces, and ranks the spaces by priority of search, thereby making the problem more tractable. We describe three experiments for validating the technique using the VERA system for interactive agent-based modeling.

**Keywords**

Parameter estimation, Agent-based modeling, Genetic algorithms, Optimization, Scientific modeling

## 1. Introduction

A common challenge in science is the generation of a model that can explain a set of observed data. Using a model, scientists can forecast future data and evaluate hypothetical "what-if" scenarios by altering the values of the parameters of the model [6]. AI has developed many methods to (partially) automate the process of scientific modeling [e.g., 7]. Recently, with the rise of ML techniques, symbolic regression has been used to learn both model equations and model parameters from data [25, 26].

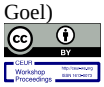
Traditionally many scientific models of complex systems were described with differential equations, for example, the Lotka-Volterra [20] equations for modeling predator-prey relationships in ecology, the Kermack-McKendrick [18] model of epidemiology, and the Bass [3] model for innovation diffusion. Over the last generation, agent-based models have become very popular in some scientific disciplines such as ecology, economics, and epidemiology [5, 23]. While differential equation models are deterministic and describe system-level behavior of homogeneous populations, agent-based models are stochastic and describe individual-level interactions among heterogeneous populations [14, 22]. The parameter estimation problem in agent-based modeling is particularly challenging because the system behavior emerges out of interactions among a large number of individuals and thus it is computationally very intensive and scales with the number of parameters.

Given the large dimensionality of the problem, optimization techniques such as genetic algorithms (GA) can and have been used in conjunction with agent-based modeling to explore the parameter space and find the best parameter set with respect to the optimization function [10, 19, 28]. However, this is an incomplete solution because GAs themselves can require a very large number of iterations to

In A. Martin, K. Hinkelmann, H.-G. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), Proceedings of the AAAI 2021 Spring Symposium on Combining Machine Learning and Knowledge Engineering (AAAI-MAKE 2021) - Stanford University, Palo Alto, California, USA, March 22-24, 2021.

EMAIL: williambroniec@gatech.edu (W. Broniec); sungeun.an@gatech.edu (S. An); spencer@cc.gatech.edu (S. Rugaber); ashok.goel@cc.gatech.edu (A. Goel)

ORCID: 0000-0002-0877-7063 (W. Broniec); 0000-0001-7116-9338 (S. An); 0000-0001-7116-9338 (S. Rugaber); 0000-0001-7116-9338 (A. Goel)



converge and thus are computationally very intensive. Bayesian methods too have been used in conjunction with agent-based modeling so that *a priori* probabilities of the parameters can help bias the estimation process. However, this method requires prior knowledge about the probability distributions of the components being modeled, which is not always available. For example, specification of the parameters of approximately 1.5 million biological species in Smithsonian Institution's Encyclopedia of Life (EOL; eol.org) [21] contain little, if any, information about their prior probabilities.

The problem of parameter optimization is especially acute in interactive agent-based modeling, for example, when an agent-based simulation platform, such as NetLogo (https://ccl.northwestern.edu/netlogo/), is used for supporting human learning. This is in part because of the limited domain expertise of human learners and partly because of a lack of cognitive strategies to search a complex search space. In fact, recent research suggests that human learners typically struggle with estimating the parameter values of agent-based simulations [2].

In this paper, we describe a knowledge-based method to guide a GA to convergence in agent-based simulations for the parameter estimation problem. The research question associated with this effort is: How can AI techniques be applied to agent-based modeling (ABM) in order to (1) given existing process model m, induce a model m* such that m* better explains observed data? (2) analyze, ground, and provide an understanding of the emergent properties of the simulation?

Our method has two phases. First, we develop a categorization of parameters for agent-based simulations, combining and grouping simulation parameters into four types according to their functions in the simulation: start-state, isolated, relationship, or object property. Our method uses this knowledge to decompose the large search space of parameter estimation into smaller and simpler spaces, and ranks the spaces by priority of search, thereby making the problem more tractable. Second, in the resulting smaller and simpler search spaces, our technique uses random variables and polynomial functions that can give a close approximation of the agent-based simulations while being much faster.

We have evaluated our knowledge-based function approximation technique on the Virtual Experimentation Research Assistant (VERA; https://vera.cc.gatech.edu/) [1], a free, public online modeling and simulation tool. In this paper, we illustrate the utility of the proposed method on three models in two separate domains (ecology and epidemiology). We have also evaluated our method with a simulation taken from the NetLogo standard library (https://ccl.northwestern.edu/netlogo/) for external validity.

## 2. Related Work

### 2.1. Agent-Based Modeling

Agent-based modeling (ABM) is a powerful simulation technique that has seen a number of applications in the last few years, including applications to understand complex systems and solve real-world problems [11]. In ABM, a system is modeled as a collection of autonomous individual entities that simulate real systems by interacting with each other within the environment. ABM serves as a "virtual laboratory" where alternative traits for key behaviors can be tested by plugging them into the ABM and testing how well the ABM then reproduces patterns observed in the real system. However, an important drawback of ABM is its time complexity [23]. Interactions between agents will introduce at least polynomial time complexity with regard to the number of agents, and interactions with even higher complexity may also be introduced. Regardless of optimization techniques employed, we necessarily will need to repeatedly make comparisons between the target data and the proposed simulation.

### 2.2. Optimization in ABM

Optimization approaches including genetic algorithms have previously been applied to ABMs to reach global or near-global optima. However, the use of such metaheuristics in the context of ABM brings specific difficulties [9, 10, 19]. First, the computation of the fitness function requires the execution of the interactions among a large number of agents, which implies a high time complexity. Second,

although the property of emergence in ABM is powerful, it does not naturally provide an explanation for how the result ties back to the parameters. Instead, understanding of the parameters comes from statistical "sensitivity analysis" that can be used to determine the most important input variables for an output behavior within the model [15]. It is thus necessary to develop strategies to accelerate the convergence of the algorithm and to understand the parameters. In this paper, we describe a knowledge-based approach based on a categorization of the functional roles of the parameters in the simulation to guide the generic algorithm to address these issues.

## 3. Virtual Laboratory for inquiry-based modeling

VERA supports inquiry-based modeling by providing learners the authentic experience of scientific inquiry (e.g., identifying a problem, proposing multiple hypotheses, testing the hypotheses, and rejecting/accepting the hypotheses) through construction, evaluation, and revision of conceptual models. Hypothesis testing is particularly important because then learners can take a more active role in constructing their own understanding in a feedback loop. However, experimenting by running simulations requires mathematical abilities as well as programming skills because a student should understand complex mathematics to write code in the simulation language. VERA empowers students to test their hypotheses irrespective of their mathematical abilities because it can automatically spawn NetLogo simulations from the conceptual models.

### 3.1 VERA for ecological modeling

VERA for ecological modeling (or VERA-Eco) enables users to build a conceptual model by adding biotic or abiotic components and drawing relationships among them on the model canvas. Conceptual models of ecological phenomena in VERA are expressed in the Component-Mechanism-Phenomenon (CMP) language [17, 27] that derive from the Structure-Behavior-Function theory of modeling complex systems [12]. A CMP model consists of components and relationships between components. A component can be one of three types: biotic, abiotic, and habitat. A relationship relates one component to another in a directed manner (e.g., component X consumes component Y). Figure 1 illustrates a CMP model of phosphorus run-off in the Chesapeake Bay; the large oval boxes in the middle depict habitats, in this case, land and shallow water. (The template on the right depicts simulation parameters and their values.)

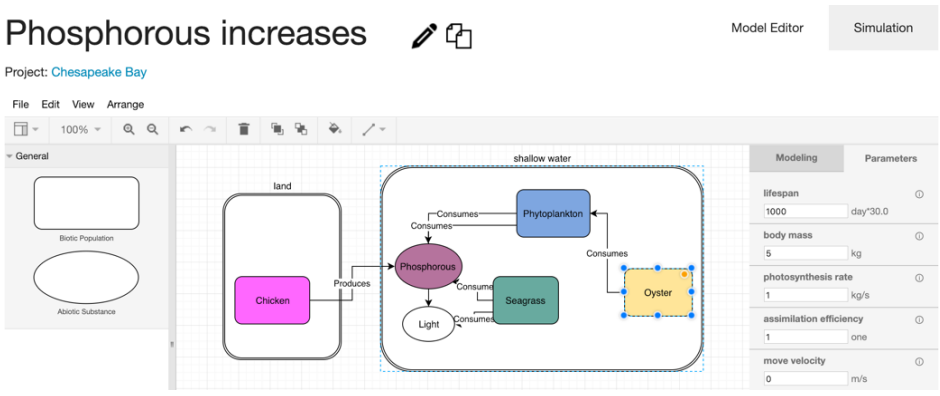



**Figure 1**: A screenshot of the VERA model editor page.

Following our earlier work [16], VERA automatically translate the patterns in the conceptual models into the primitives of agent-based simulation of NetLogo. The running of the simulation enables the user to observe the evolution of the system variables over time and iterate through the generate-evaluate-revise loops. In this way, VERA integrates both qualitative reasoning in the conceptual model and quantitative reasoning in the simulation reasoning on one hand, and explanatory reasoning (conceptual model) and predictive reasoning (simulation) on the other.

VERA thus acts as a virtual laboratory for scientific experimentation. The learner begins with a question. She then generates (potentially) multiple hypotheses for answering the question. In the process, the user may consult EOL for inspiration. Next, she elaborates on the hypotheses by constructing a detailed conceptual model. Then the learner asks VERA to spawn a simulation from the conceptual model. VERA provides the learner with templates of simulation parameters. The user sets initial values for the parameters and may again consult EOL for finding the values. VERA now automatically spawns the simulation and displays the results as graphs, for example, a graph indicating the changes in populations of various species over time. The learner may now experiment with different simulation parameters, or revise the conceptual model, or generate an alternative hypothesis.

## 3.2. VERA for Epidemiological Modeling

At the start of the COVID-19 pandemic, VERA Epidemiology (VERA-Epi) was created to support agent-based versions of compartmental epidemiology models [8]. Just as with VERA-Eco, users develop a graphical representation of a model, provide parameter values, and VERA will generate a subsequent agent-based simulation. The model semantics for VERA-Epi are based on the Harel statechart [13]; nodes now represent the states of individual agents, and edges represent likelihoods for those agents to transition between states.

# 4. Parameter Estimation Method

Figure 2 illustrates our method for automated estimation of the values of the simulation parameters in VERA. Since the search space for optimizing an agent-based simulation is large, the method uses parameter categorization to simplify the structure and reduce the computation while preserving its semantics. Then various functions are applied to approximate the agent-based simulation output. After the ABM approximation process, a genetic algorithm is used to solve the combinatorial problem of finding the optimal set of parameter values for different components.

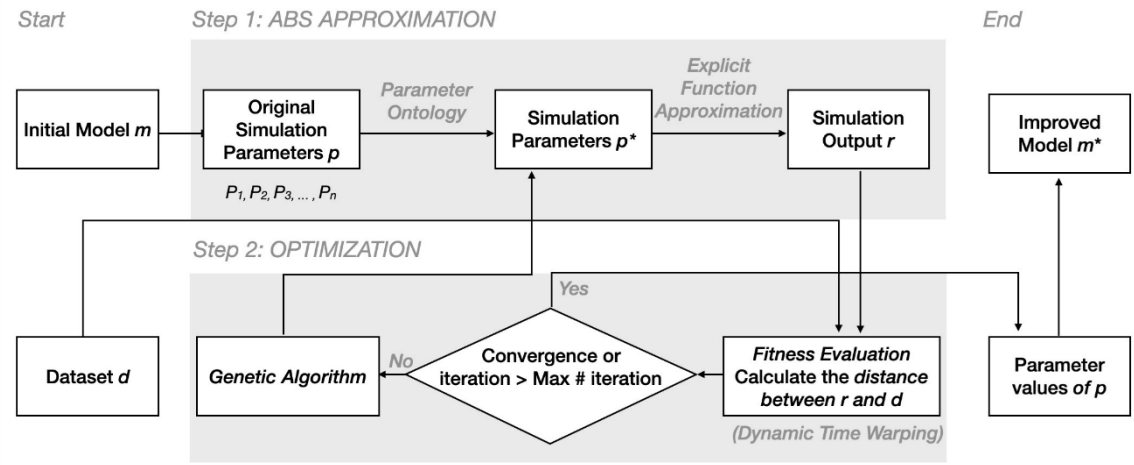



**Figure 2**: Overview of the proposed method for ABM approximation and parameter optimization.

## 4.1. Function Approximation

Given a dataset and an existing model, we want to assist human learners in finding the optimal parameter values that, when used to generate a simulation, yield results closest to that dataset. This can be formalized as an optimization problem where the inputs are the simulation parameters of a model and error is the distance between the simulation output and the initial dataset.

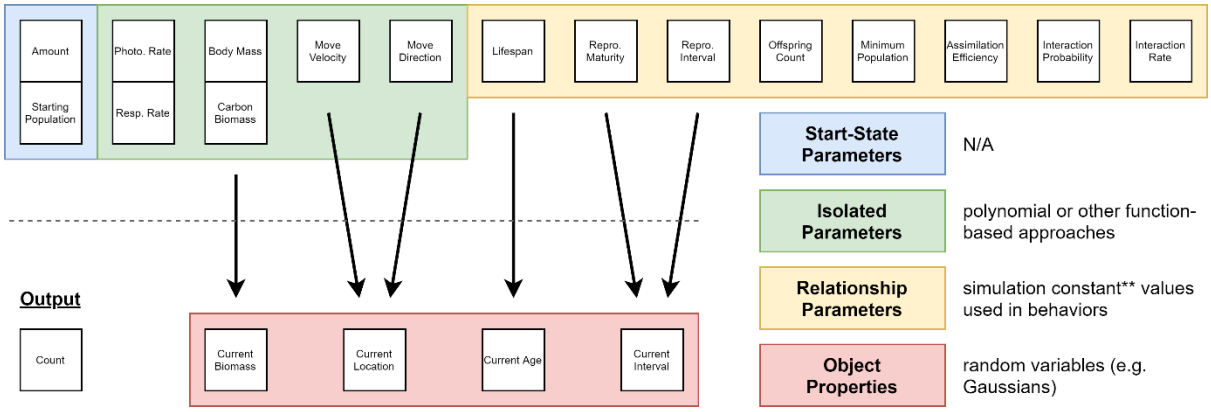



**Figure 3**: Parameter categorization for VERA Ecology

### 4.1.1. Parameter Categorization

First, a distinction needs to be made between object properties and class properties. Object properties are concerned with each agent in the simulation, and their values change each tick of the simulation's clock based on the agents' behaviors (e.g., age, location, etc.). On the other hand, class properties are constant values used to set up the simulation (e.g., starting population, lifespan, body mass, etc.) The top row of Figure 3 below shows the original parameters used in the agent-based simulation (e.g., start-state, isolated, relationship parameters) and the derived properties from the original parameters (e.g., object properties), color-coded by their category. Here are the descriptions of each parameter category:

- *Start-State Parameters*: Simulation values that set up the simulation's starting state, and have no effect after
- *Isolated Parameters*: Parameters describing behaviors that only affect an individual agent and no others
- *Relationship Parameters*: Parameters affecting interactions among different agents
- *Object Property:* Each agent tracks these core values internally

This categorization of simulation parameters can be compared earlier work on the use of ontologies for building agent-based simulations. For example, Benjamin, Patki & Mayer (2006) describe an ontology of components of an agent-based simulation [4]. In contrast, our work focuses on the categorization of the functional role of simulation parameters.

The "stacked" parameters with pairs of blocks connected shown in the start-state and isolated parameters in Figure 3 mean that these pairs of parameters are treated as a single parameter from the eyes of the simulation. This is primarily driven by the semantics of the user interface. Users may benefit in conceptual understanding from different wording as it applies to different classes of agents, while programmatically these two different parameters serve the same purpose or are integrated into a single value used in the simulation. Measured output values in the simulation are also displayed, and in the case of VERA-Eco this is simply the count of each agent class. Instead of optimizing each parameter individually and calculating them repeatedly in the ABM (e.g., "lifespan" does not have to be calculated over and over), behaviors are simulated using polynomial function approximation.

### 4.1.2. Random Variables and Polynomial Functions

Using the parameter categorization, an approximation of the agent-based simulation output can be derived using random variables to model populations of agents and polynomial functions to model agent behaviors. Using random variable distributions as stand-ins for population groups drastically reduces the number of computations performed and the memory used. Different populations may be more accurately modeled by specific distribution functions, but the normal distribution serves as the best

stand-in with an unknown distribution due to the central limit theorem. Therefore, rather than storing biomass for thousands of individual agents, a Gaussian distribution can be represented using two variables, the mean and the variance, to describe the biomass for each age. The same process is applied to represent reproductive interval Gaussians as well.

In the case of Vera-Eco, the simulation initialization (e.g., tick 0) assigns each of the starting populations a random age from 0 to max age (i.e., lifespan - 1) and sets the initial biomass value for each population, and the biomass follows a uniform distribution with the mean of initial biomass value and the variance of 0. Each tick of the simulation, the polynomial functions are applied to these populations to skew the distribution. For example, in every simulation tick, a certain amount of biomass is lost from every agent due to its metabolism as determined by its respiratory rate, which will subtract from the mean while the variance does not change.

However, when there is a relationship between two populations, such as predation, the corresponding consumption events will increase the average biomass for some predator agents and the reduction of some prey agents. In this case, the Gaussians are recomputed off the changing values, and computation of the next behavior proceeds.

## 4.2. Optimization

To obtain the closest values possible to the target dataset, an optimization algorithm is necessary to test and evaluate different parameter sets. Scientific models based on differential equations can rely on regression analysis to achieve this, but agent-based models typically lack such representations. Heuristic search is needed to explore the space, and due to the highly combinatorial nature of estimating parameters, genetic algorithm was selected. Figure 4 shows a standard genetic algorithm representation. The process begins with a set of individual members of a species which is called a Population. A species is characterized by a set of parameters (also known as genes) that together determine the dynamics of the individuals of the species (also known as a chromosome).

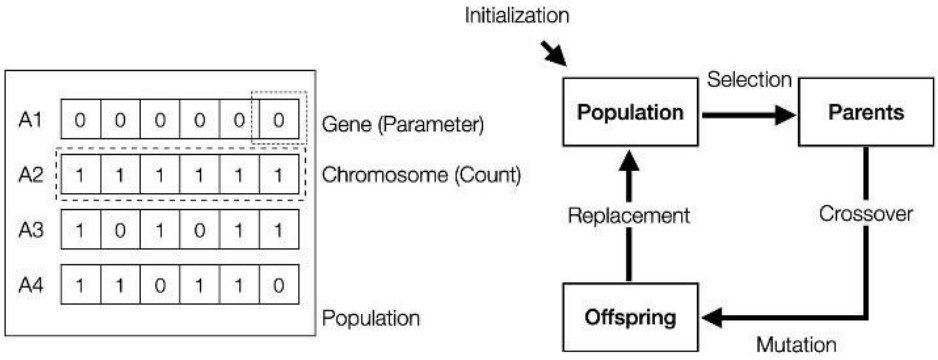



**Figure 4**: A standard genetic algorithm representation and process

The population of chromosomes is initialized randomly. Each chromosome is then evaluated using the difference between the simulation approximation and the target dataset as a fitness evaluation. A selection is made among the population of chromosomes based on these scores, and we obtain a new population named parent population. Recombination (also known as crossover) and mutation operators are then applied to this population which yields new sets of parameter values to continue the process.

### 4.2.1. Fitness Function

To evaluate how "fit" the simulation output r is with respect to dataset d, we compare the similarity between the two sets of output data. Multiple methods including simple Euclidean distance can be used, but we used dynamic time warping (DTW), which is a robust, simple, and efficient measure for computing the dissimilarity between two time-series datasets [24]. DTW belongs to the group of so-called elastic dissimilarity measures and works by optimally aligning (or 'warping') the time series in

the temporal dimension so that the accumulated cost of this alignment is minimal. In its most basic form, this cost can be obtained by dynamic programming, recursively applying:

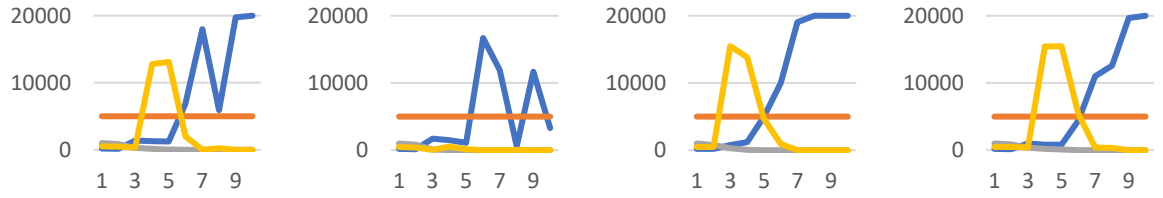
$$Di,j = \delta(xi, yj) + min(Di,j-1, Di-1,j, Di-1,j-1) \quad (1)$$

for i = 1,...,M and j = 1,...,N, being M and N the lengths of our two time series (here the dataset and the new parameter set). As we are using distance as a fitness measure, we used negative distance to represent the fitness of the solution (larger fitness measure means better solutions).

## 5. VERA Ecology Results

Using the genetic algorithm to optimize over the combination of random variables and polynomial functions to approximate our ABM, we get results faster by orders of magnitude at the cost of some accuracy. In Figure 5, Graph (a) below shows a synthetic target dataset and simulation output graph of a simple VERA-Eco model, and Graph (b) shows the same simulation with parameter values randomized. This basic simulation consists of sunlight, two different plants, and a species of bug that consumes both of them. While both graphs show roughly the same pattern for the blue, orange, and grey lines, the population shown in yellow varies drastically. In the left graph, the population rises and then falls after a few cycles, whereas in the simulation dataset it collapses immediately.

| **Target Dataset** | | | **Initial Simulation** | | | **Trial 1** | | | **Trial 2** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bug | Tree | Kudzu | Bug | Tree | Kudzu | Bug | Tree | Kudzu | Bug | Tree | Kudzu |
| 200 | 1,000 | 500 | 200 | 1,000 | 500 | 200 | 1000 | 500 | 200 | 1,000 | 500 |
| 145 | 873 | 481 | 145 | 797 | 406 | 210 | 755 | 483 | 147 | 866 | 480 |
| 1,380 | 302 | 337 | 1,725 | 112 | 48 | 749 | 283 | 15,479 | 992 | 382 | 369 |
| 1,304 | 115 | 12,797 | 1,479 | 12 | 535 | 1,220 | 62 | 13,808 | 817 | 190 | 15,437 |
| 1,246 | 36 | 13,097 | 1,099 | 4 | 161 | 4,939 | 0 | 4,656 | 839 | 87 | 15,465 |
| 6,867 | 0 | 1,991 | 16,694 | 0 | 0 | 9,998 | 0 | 933 | 4,448 | 1 | 5,328 |
| 18,009 | 0 | 13 | 11,900 | 0 | 0 | 19,067 | 0 | 8 | 10,996 | 0 | 408 |
| 5,929 | 0 | 238 | 575 | 0 | 68 | 19,992 | 0 | 0 | 12,563 | 0 | 340 |
| 19,762 | 0 | 1 | 11,655 | 0 | 0 | 20,000 | 0 | 0 | 19,660 | 0 | 3 |
| 19,999 | 0 | 0 | 3,275 | 0 | 1 | 20,000 | 0 | 0 | 19,997 | 0 | 0 |




(a) (b) (c) (d)

**Figure 5**: Results of using our methods on the agent-based simulation of VERA. (a) Left-most graph–Target data d. (b) Middle-left graph–Initial model m. (c) Middle-right graph–Improved model m* on first trial (< 2 minutes). (d) Right-most graph– improved model m* on second trial (= 2 minutes). In each graph: the blue line represents the bug species, the orange line represents sunlight, the grey line represents the tree species, and the yellow line represents the kudzu vine. Sunlight is omitted from the table due to being at a constant value of 5,000 units in all derivations of this simulation.

Graphs (c) and (d) show the results of two independent runs of our function approximation methods. Since the mutation, crossover, and selection are stochastic, each run of the simulation yields different results. In the first graph, the kudzu population (indicated as the yellow line) more closely resembles that from the target dataset while the valley in the bug population (indicated as blue lines) was absent due to compounding error in our approximation. In the second run, the bug population resembles the original dataset even closer.

## 5.1 VERA Epidemiology Results

To test the domain generality of our technique, we used the same optimization framework in VERA-Epi that uses an agent-based version of the SIR model of epidemiology, a basic but significant and well-studied model of disease spread that groups a population into three categories – Susceptible (S), Infected (I), and Recovered (R) – and provides equations that describe the rates at which the sizes of these groups change [29]. Traditionally, the parameters of the SIR model are written as beta (β), the disease transmission rate, and gamma (γ), the recovery rate. The user interface of VERA-Epi presents the user with a larger set of more detailed parameters in the SIR model, but these are reduced to functionally equivalent parameters. The first step in the optimization process is to classify and group the parameters according to the categorization.

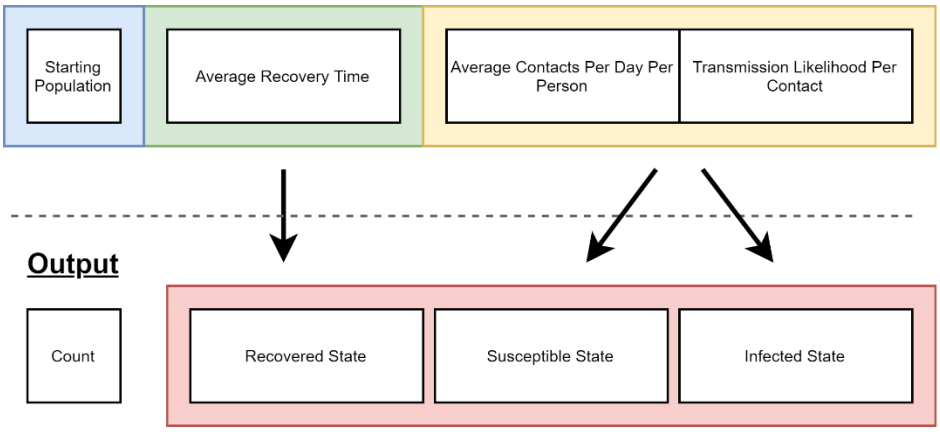



**Figure 6**: Parameter categorization of VERA-Epi

For the VERA-Epi SIR model, we used the same categorization as before to group the parameters (see Figure 6). The starting population value can be used as the initial state, and the only object property that needs to be tracked is the health state of the agent (susceptible, infected, or recovered). The average contacts per day per person is combined with the transmission likelihood per contact to generate a likelihood that an agent will become infected (corresponds to beta of the ordinary SIR model). Average recovery time also impacts state by defining the likelihood an individual agent will recover from infection (corresponds with gamma of the original SIR model).

As we can see, Average Recovery Time is classified as an isolated parameter while Average Contacts Per Day Per Person and Transmission Likelihood are combined into a single relationship parameter. This is because agents recover on their own, irrespective of any other agents. However, agents will only get sick if they come into contact with other agents. Because the agent's state in this model is one of several possibilities rather than a numeric value, it cannot be represented by a typical Gaussian distribution. The distribution selected should be that most appropriate for the target simulation, and because the SIR model makes no representation of "partially sick" or "partially recovered", the simplest solution is to treat each state as simply a separate distribution with zero variance, also known as the Dirac delta distribution. With the parameter space mapped out and the distributions known, this reduction can be plugged into the genetic algorithm method explained above. While the performance gains are not as significant as with VERA-Eco due to this simulation being simpler, it does reduce the time complexity as a function of simulation size. In effect, this reduction closely recreates the original equation form of the SIR model, although still operating on discrete units.

# 6. External Validity

VERA is simply one engine for producing agent-based models. Being able to apply our method to different types of ABMs would increase the external validity of our methods. The “Rabbits, Grass, Weeds” simulation from the NetLogo example library [30] was selected for two main reasons. First, the example was also in the domain of ecology and posited a scenario (rabbits foraging for food) that could be replicated in VERA-Eco but was written with entirely different simulation code. Second, the example possessed only a handful of parameters, providing an example simulation more basic than VERA’s to work with.

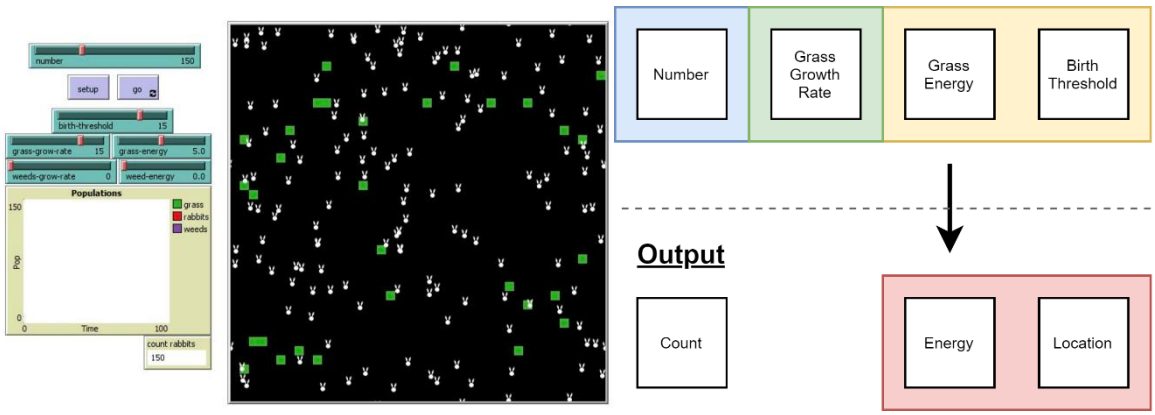



**Figure 7**: (a) Left: The "Rabbits, Grass, Weeds" simulation from the NetLogo example library. This screenshot shows the simulation interface in action with variable sliders on the left controlling the different simulation parameters. (b) Right: Parameter map in the "Rabbits, Grass, Weeds" simulation.

The "Rabbits, Grass, Weeds" simulation is a simplified model of a predator and prey between the rabbits, grass, and weeds. When a rabbit bumps into some grass or weeds, it eats the grass to gain its energy (see Figure 7). If the rabbit gains enough energy, it reproduces. Otherwise, it dies. This simulation consists of six parameters: starting number, birth threshold, grass growth rate, grass energy,

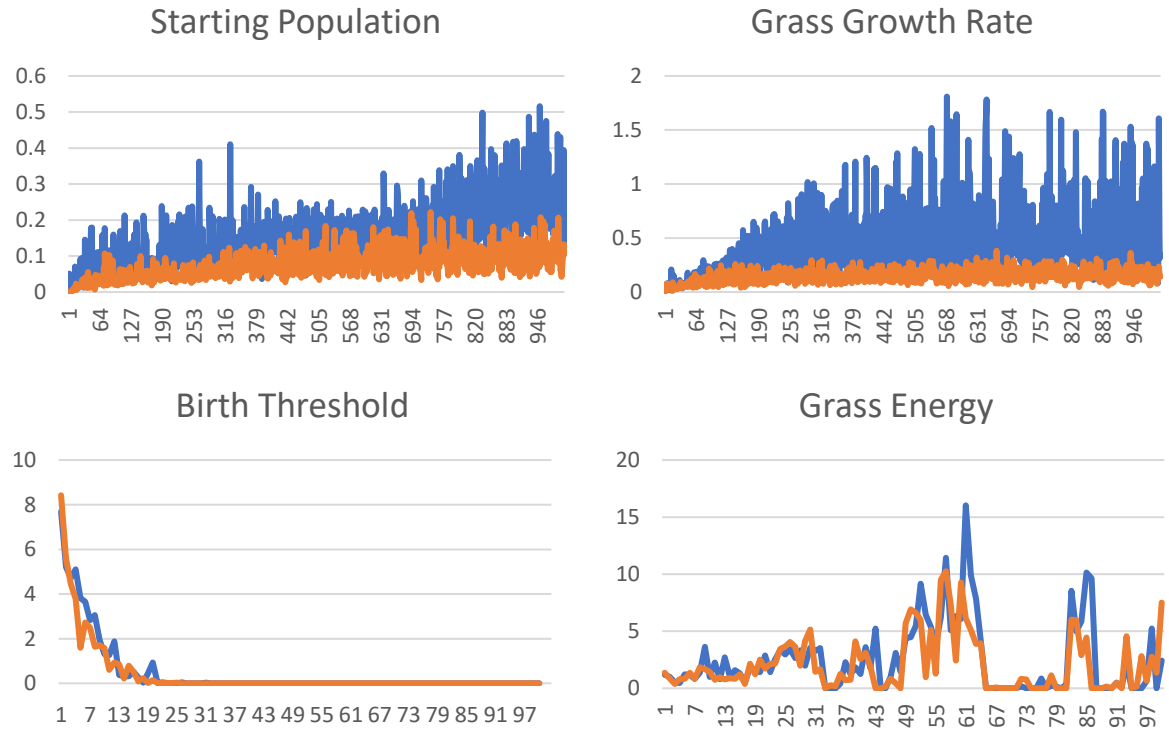



**Figure 8**: Results of Sensitivity Analysis of the different Parameters in the "Rabbits, Grass, Weeds" simulation. Blue line–Original. Orange line–Estimation. (a) Upper-left graph: Size of the rabbit

population. (b) Upper-right graph: Grass growth rate. (c) Lower-left graph: Birth threshold. (d) Lower-right graph: Grass energy.

weeds growth rate, and weeds energy. Each individual rabbit agent has two variable values associated with it–current energy and location. If a rabbit finds some grass, it will consume the grass and gain energy. If the rabbit finds weeds, it will gain no energy. During each tick of the simulation clock, the rabbits expend a fixed amount of energy, and a rabbit that runs out of energy dies, removing it from the simulation.

Using the same categorization described in the previous section (see Section 4.1) to break down the simulation parameters, we get the following map as shown in Figure 7 (b). Grass growth rate, grass energy, and birth threshold are combined to describe energy using polynomial functions, and energy and location of each agent are represented as a set of Gaussian distributions. The grass parameters affect the energy Gaussian of the rabbit population. Location is also a Gaussian distribution in the simulation, but no parameters in this simulation control the location.

Figure 8 shows four graphs with sensitivity analysis of the different parameters–the blue line being the sensitivity analysis of the actual simulation and the orange line being that of the approximation. The x axis for these four graphs are the attempted parameter values, and the y axis is the difference in distance between the outputs. In other words, it shows how much each parameter affects the simulation results. For example, starting population (a) and grass growth rate (b) have minor, roughly linear impacts on the output whereas birth threshold (c) is a sharp cutoff (e.g., if it is too high, rabbits will die before they have a chance to reproduce), and grass energy (d) has a stair step effect.

## 7. Conclusion

We have described a knowledge-based method for speeding up the use of GAs for optimizing agent-based simulations. Specifically, we described a general categorization for classifying simulation parameters that can be used by other agent-based simulations. This categorization of simulation parameters complements and supplements earlier research on ontologies of components of agent-based simulations. The validity of our method was shown by the application examples across domains using the VERA modeling and simulation platform as well as through an external NetLogo predation model. Overall, our system works well for decomposing and understanding the semantic characteristics of the agent-based simulation parameters with exponentially faster results than optimization over the simulation itself. This affords rapid simulations thereby supporting end users.

The primary drawback to our method is error propagation. With one species or a small number of relationships, the simulation is near-exact. With more complex simulations running over longer periods of time, it slowly begins to deviate: some important information may be missing, which can take the simulation into a completely different course. Therefore, our next step is to develop additional strategies to reduce the compounding error in our approximation and to apply the method to more complex examples. Another direction for further work is to conduct a user study to better understand how parameter estimation can facilitate the process of human learning and scientific discovery.

## Acknowledgements

This research is supported in part by an US NSF grant #1636848 (Big Data Spokes: Collaborative: Using Big Data for Environmental Sustainability: Big Data + AI Technology = Accessible, Usable, Useful Knowledge!) and the NSF South Big Data Hub.

## References

[1] S. An, R. Bates, J. Hammock, S. Rugaber, E. Weigel & A. Goel. (2020). Scientific modeling using large scale knowledge. *In Procs. Twenty-first International Conference on AI in Education (AIED'2020),* pp. 20-24.

[2] S. An, S. Rugaber, E. Weigel & A. Goel (2021) Cognitive strategies for navigating high-dimensional parameter spaces in modeling complex systems; submitted for publication.
[3] F. Bass. (1969). A new product growth for model consumer durables. Management science, 15(5), 215-227.
[4] P. Benjamin, M. Patki & R. Mayer. (2006) Using ontologies for simulation modeling. In *Procs. 2006 IEEE Winter Simulation Conference.*
[5] E. Bonabeau & C. Meyer. (2001) Swarm intelligence: A whole new way to think about business. *Harvard Business Review* 79(5), 106-115.
[6] W. Bridewell, J. Sánchez, P. Langley & D. Billman. (2006). An interactive environment for the modeling and discovery of scientific knowledge. *International Journal of Human-Computer Studies*, 64(11), 1099-1114.
[7] W. Bridewell, P. Langley, L. Todorovski & S. Džeroski. (2008). Inductive process modeling. *Machine Learning*, 71(1), 1-32.
[8] W. Broniec, S. An, S. Rugaber, & A. Goel. (2020). Using VERA to explain the impact of social distancing on the spread of COVID-19. *arXiv preprint arXiv:2003.13762.*
[9] E. Cabrera, M. Taboada, M. Iglesias, F. Epelde & E. Luque. (2011). Optimization of healthcare emergency departments by agent-based simulation. *Procedia computer science*, 4, 1880-1889.
[10] B. Calvez & G. Hutzler. (2005). Automatic tuning of agent-based models using genetic algorithms. In *Procs. International Workshop on Multi-Agent Systems and Agent-Based Simulation* (pp. 41-57). Springer, Berlin, Heidelberg.
[11] V. Grimm, U. Berger, F. Bastiansen, et al. (2006). A standard protocol for describing individual-based and agent-based models. *Ecological modelling*, 198(1-2), 115–126.
[12] A. Goel, S. Rugaber & S. Vattam. (2009). Structure, Behavior and Function Models of Complex Systems: The Structure-Behavior-Function Modeling Language. *AIEDAM* 23: 23-35.
[13] D. Harel. (1987). Statecharts: A visual formalism for complex systems, *Science of computer programming.* 231-274.
[14] E. Hunter, B. MacNamee & J. Kelleher. (2018). A comparison of agent-based models and equation based models for infectious disease epidemiology. In Procs. *AICS* (pp. 33-44).
[15] B. Iooss & P. Lemaître. (2015). A review on global sensitivity analysis methods. In *Uncertainty management in simulation-optimization of complex systems* (pp. 101-122). Springer, Boston, MA.
[16] D. Joyner, A. Goel & N. Papin. (2014). MILA--S: generation of agent-based simulations from conceptual models of complex systems. In *Procs. 19th international conference on intelligent user interfaces* (pp. 289-298).
[17] D. Joyner, A. Goel, S. Rugaber, C. Hmelo-Silver & R. Jordan. (2011). Evolution of an Integrated Technology for Supporting Learning about Complex Systems: Looking Back, Looking Ahead. In *Procs. 11th IEEE International Conference on Advanced Learning Technologies*, pp. 257-259.
[18] W. Kermack & A. McKendrick. (1927). A contribution to the mathematical theory of epidemics. In *Procs. Royal Society of London. Series A, Containing papers of a mathematical and physical character*, 115(772), 700-721.
[19] J. Lee, T. Filatova, A. Ligmann-Zielinska, et al. (2015). The complexities of agent-based modeling output analysis. *The journal of artificial societies and social simulation*, 18(4).
[20] A. Lotka. (1910). Contribution to the Theory of Periodic Reaction. *The Journal of Physical Chemistry*, 14, 271-274.
[21] C. Parr, M. Wilson, M. Leary et al et al. (2014). The encyclopedia of life v2: providing global access to knowledge about life on earth. *Biodiversity Data Journal* (2).
[22] H. Parunak, R. Savit & R. Riolo. (1998). Agent-based modeling vs. equation-based modeling: A case study and user's guide. In *Procs. Multi-Agent Systems and Agent-Based Simulation*, 10-25.
[23] S. Railsback & V. Grimm. (2019). *Agent-based and individual-based modeling: a practical introduction*. Princeton University Press.
[24] H. Sakoe & S. Chiba. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1), 43-49.
[25] M. Schmidt & H. Lipson. (2009). Distilling free-form natural laws from experimental data. *Science*, 324(5923), 81-85.
[26] S. Udrescu & M. Tegmark. (2021). AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16), eaay2631.

[27] S. Vattam, A. Goel, S. Rugaber et al. (2011). Understanding complex natural systems by articulating Structure-Behavior-Function models. *Educational Technology & Society*, 14(1): 66-81.
[28] Z. Wang & J. Zhang. (2012). Agent-based modeling and genetic algorithm simulation for the climate game problem. *Mathematical Problems in Engineering*.
[29] H. Weiss. (2013). The SIR model and the foundations of public health. *Materials mathematics*. 0001-17.
[30] U. Wilensky. (2001). NetLogo rabbits grass weeds model. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. *http://ccl.northwestern.edu/netlogo/models/RabbitsGrassWeeds.*