Published in final edited form as:

Simulation. 2021 April ; 97(4): 287–296. doi:10.1177/0037549720975075.

Nested active learning for efficient model contextualization and parameterization: pathway to generating simulated populations using multi-scale computational models

Chase Cockrell¹, Jonathan Ozik², Nick Collier², Gary An¹

¹Department of Surgery, University of Vermont, USA

²Argonne National Laboratory, USA

Abstract

There is increasing interest in the use of mechanism-based multi-scale computational models (such as agent-based models (ABMs)) to generate simulated clinical populations in order to discover and evaluate potential diagnostic and therapeutic modalities. The description of the environment in which a biomedical simulation operates (model context) and parameterization of internal model rules (model content) requires the optimization of a large number of free parameters. In this work, we utilize a nested active learning (AL) workflow to efficiently parameterize and contextualize an ABM of systemic inflammation used to examine sepsis.

Contextual parameter space was examined using four parameters external to the model's rule set. The model's internal parameterization, which represents gene expression and associated cellular behaviors, was explored through the augmentation or inhibition of signaling pathways for 12 signaling mediators associated with inflammation and wound healing. We have implemented a nested AL approach in which the clinically relevant (CR) model environment space for a given internal model parameterization is mapped using a small Artificial Neural Network (ANN). The outer AL level workflow is a larger ANN that uses AL to efficiently regress the volume and centroid location of the CR space given by a single internal parameterization.

We have reduced the number of simulations required to efficiently map the CR parameter space of this model by approximately 99%. In addition, we have shown that more complex models with a larger number of variables may expect further improvements in efficiency.

Keywords

Agent-based modeling; machine learning; parameter space exploration

1. Introduction

1.1. Motivation

There is increasing interest in the use of “mechanism”-based multi-scale computational models as an aid to more traditional biomedical research methods, noting that the “mechanisms” depicted in these types of models are the types of molecular and cellular behaviors reported in the basic science literature, as opposed to first principle mechanisms as used in the physical sciences. This approach integrates existing cellular and molecular mechanistic knowledge into tissue-, organ-, and patient-level representations that are used to generate “virtual populations” to investigate potential diagnostic and therapeutic modalities. This is the underlying concept of creating “digital twins” for the study of precision medicine¹ and the regulatory interest in advancing the ability to perform in silico clinical trials.² However, achieving these goals involves several significant hurdles related to the complex nature of the biology being studied, the models used to represent that biology, and the contextualization of those models in an approximation of a clinical environment. In this work, we address some of the challenges associated with the use of agent-based models (ABMs) to generate clinically relevant (CR) simulation experiments, specifically those related to the effective and efficient parameterization of complex ABMs and the evaluation of those parameterizations within a clinical context that itself represents a multi-parametric space. We perform this work using a previously validated ABM of acute systemic inflammation,^{3,4} the Innate Immune Response Agent-Based Model (IIRABM), while studying the clinical context of sepsis, an inflammatory condition with a mortality rate of between 28% and 50%.⁵ Numerous mechanistic computational simulations of acute inflammation and sepsis have been utilized over the past two decades.^{3,6–12} These models have demonstrated that the sepsis population is much more heterogeneous than previously thought and this can be reflected by utilizing a range of multidimensional parameters that correlate to biologically plausible behaviors and phenotypes. Despite insights generated from these methods, there remain considerable challenges in the calibration and parameterization of the models.

For the purposes of this work, we separate the parameters into two classes: context and content. The description (context) of the environment in which a biomedical simulation operates and parameterization of internal model rules (content) requires the optimization of a large number of free parameters; given the wide range of variable combinations, along with the intractability of ab initio modeling techniques that could be used to constrain these combinations, an astronomical number of simulations would be required to achieve this goal. The model’s internal parameterization can be thought of as an abstraction of an in vivo genetic signaling network, while the model context defines the simulated injury to which the model responds. In previous work, this model has successfully replicated the range of cytokine time-course dynamics of sepsis and healing across a wide range of model contexts^{3,4,6}; however, all models have been “genetically identical,” analogous to a typical mouse experiment. In practice, we recognize that there are two primary sources of variation in biological data. In any biological system, the response to a given stimulus contains some intrinsic degree of stochasticity – this can be seen in the range of responses from genetically identical mice to identical stimuli. Alternatively, variance in biological data output can

arise from the genetic variability among individuals represented in the experiment. In order to determine the range of individuals/conditions our model can represent, internal parameterization boundaries (as well as their associated contextual boundaries) must be determined.

The determination of these boundaries can be extremely costly (in terms of the number of simulations required, which is directly related to time, energy, and money) due to both the stochastic replicates necessary to account for biological heterogeneity as well as the combinatorial complexity of the rule/parameter structure. Thus, any gains in efficiency are measured by the number of simulations whose output can be accurately approximated by the machine learning (ML) techniques.

1.2. The importance of mapping parameter space

The problem of combinatorial complexity in the selection of model parameters is well-established in the computational/biological modeling communities, particularly when dealing with highly complex models.^{13–17} In previous work,³ we utilized high-performance computing (HPC) to demonstrate the need for comprehensive “data coverage” among possible model states as well as the importance of internal parameter variation (as compared to model structure) to capture the full range of biological heterogeneity seen clinically. Here, we have extended that work to consider both model internal parameterizations as well as the parameterization which describes the model perturbation that generates disease. We posit that parameter space mapping, which has recently been rendered tractable through improved ML techniques, is a distinct task from traditional sensitivity analysis.¹⁸

In a standard parameter sensitivity analysis,^{19–21} the dependence of model output on variance in a single parameter (or potentially a combination of parameters) is quantified. In a complex ABM, this process requires the consideration of a large number of potentially free parameters, making a comprehensive calibration difficult^{22–26} and limiting the utility of traditional parameter sensitivity analysis techniques^{27,28} through rendering them computationally intractable.

The aforementioned difficulties are compounded when considering the range of biological heterogeneity seen experimentally and clinically.^{3,18} For biomedical systems, there exists no supposition of a unique solution. A sensitivity analysis only operates on the existing parameters in a model: it cannot provide an exploration of alternate, but equally valid, rule configurations. To address this, we suggest “mapping” the model space. As an illustrative, albeit simple, example, consider the following rule, present in the original version of the IIRABM, describing a term that contributes to the differentiation switch from a Th0 cell into a Th1 or Th2 cell:

$$proTh1 = c_1 * IL12 + c_2 * IFN\gamma,$$

where the c_i represents some constant values used to weight the individual contributions of the array of cytokines. One could perform a sensitivity analysis on a model consisting solely of this rule; however, that analysis would miss important bio-plausible model calibrations/

parameterizations. While the rule is coded this way in the model, this representation does not actually represent the assertion made by the rule. A more complete way to write this rule would be as follows:

$$proTh\ 1 = c_1 * IL12 + c_2 * IFN\gamma + \sum_j c_j \Gamma_j,$$

where j sums over the complete set of cytokines on the model, Γ_j represents the concentration of a specific cytokine, and the constant weighting term $c_j = 0$. Thus, a comprehensive sensitivity analysis must consider implicit zeros in model rule construction, which can vastly increase the size of the task.

In addition, the model may only be sensitive to certain parameters in a specific context. Consider a more generic version of the above rule (model parameterization 1), which hypothetically leads to biologically plausible model output:

$$P = c_1 \Gamma_1 + c_2 \Gamma_2,$$

which more completely, would be written as follows:

$$P = c_1 \Gamma_1 + c_2 \Gamma_2 + \sum_{j \neq 1,2} c_j \Gamma_j,$$

where $c_j = 0$. There is no supposition that this hypothetically calibrated rule is uniquely correct. Assume the existence of an additional/alternative calibrated rule (model parameterization 2), which leads to equally biologically plausible model output:

$$P = c_5 \Gamma_5 + c_6 \Gamma_6 + \sum_{j \neq 5,6} c_j \Gamma_j.$$

In model parameterization 1, in which all cytokine multipliers are set to 0 except for cytokines 1 and 2, the model will appear more sensitive to cytokines 1 and 2 than the others. In model parameterization 2, the model will appear more sensitive to cytokines 5 and 6. A sensitivity analysis on one model parameterization is not valid for all other model parameterizations. A comprehensive sensitivity analysis would then have to incorporate the model configuration under which it is being performed, which is computationally intractable for all but the smallest of models. In order to render this task computationally tractable, we have employed a nested active learning (AL) approach in order to efficiently and comprehensively explore the model parameter space.

2. Methods

The primary model analyzed in this work is the IIRABM.^{6,29} The IIRABM is an abstract representation/simulation of the human inflammatory signaling network response to injury; the model has been calibrated such that it reproduces the general clinical trajectories seen in sepsis. The IIRABM operates by simulating multiple cell types and their interactions,

including endothelial cells, macrophages, neutrophils, TH0, TH1, and TH2 cells as well as their associated precursor cells. The simulated system dies when total damage (defined as aggregate endothelial cell damage) exceeds 80%; this threshold represents the ability of current medical technologies to keep patients alive (i.e., through organ support machines) in conditions that previously would have been lethal. The IIRABM is initiated using five external variables – initial injury size, microbial invasiveness, microbial toxigenesis, environmental toxicity, and host resilience.

2.1. Extreme-scale Model Exploration with Swift

During the initial code development, the AL models were trained and integrated using the Extreme-scale Model Exploration with Swift (EMEWS) framework.^{30–33} EMEWS enables the creation of HPC workflows for implementing large-scale model exploration studies. Built on the general-purpose parallel scripting language Swift/T,³⁴ multi-language tasks can be combined and run on the largest open science HPC resources³⁵ via both data-flow semantics and stateful resident tasks. The ability that EMEWS provides for incorporating model exploration algorithms such as AL, implemented in R or Python, allows for the direct use of the many libraries relevant to ML that are being actively developed and implemented as free and open source software.

2.2. Active learning

AL is a sub-field of ML that focuses on finding the optimal selection of training data to be used to train a ML or statistical model.³⁶ AL can be used for classification^{37,38} or regression.^{39,40} AL is an ideal technique for modeling problems in which there is a large amount of unlabeled data and manually labeling that data is expensive. In these circumstances (specifically the costly data labeling) AL provides the most generalizable and accurate model for the cheapest cost, which for the purposes of this work, is computation time.

We have adopted a nested AL structure based on the reasoning that the boundaries of the “clinically relevant”³ space is a function of the model’s internal parameterization. To place this in a clinical context, we would typically expect a patient with a high level of physical fitness, aged 20 years, to be able to withstand a more serious insult than an obese 90-year-old as well as to present different cytokine trajectories and responses to infection. The nested AL structure attempts to approximate this relationship. As model content parameterizations become increasingly “fragile,” the volume of the CR space decreases and its centroid location trends toward less serious (smaller, less virulent bacteria) injuries.

The lower-level AL procedure seeks to find the boundary of the parameter space, deemed as a function of four parameters that describe the context in which the IIRABM operates: two measures of microbial virulence (invasiveness and toxigenesis), host resilience, and environmental toxicity. In this scheme, there are two classes: CR and not CR. We assume that there exists some function:

$$y = f(\vec{x}), x \in \mathcal{X} \subset \mathbb{R}^n, y \in \mathbb{R},$$

which accurately classifies model context parameters that can be approximated given input data from the following training set:

$$D_{train} = \{x_j^t, f(x_j^t)\},$$

for $j = 1, \dots, n$. The Neural Network (NN) model uses a binary cross-entropy⁴¹ loss function, in which the loss is given by the following:

$$L = - \sum_{i=1}^2 y_i \log(\hat{y}_i),$$

where y_i is the ground truth value and $\hat{y}_i$ is the NN-approximated score. The AL algorithm begins with a randomly selected set of 20 points. The IIRABM simulation then runs a fixed number of stochastic replicates of the input points to determine class membership. This information is then used to train the ML model. The algorithm then ranks the remaining unlabeled parameterizations by class-membership uncertainty:

$$\{x_{i+1}\} = \min_x (0.5 - P_i(y | x)).$$

Those parameterizations whose class is most uncertain in the current ML model are then selected for labeling and the process repeats until a stopping criterion is reached; for the purposes of this work, once the cross-validation accuracy crossed 0.95, the algorithm was stopped.

The upper-level AL workflow uses a modified version of dropout-based AL for regression presented by Tsymbalov et al.⁴⁰ The goal of this AL workflow is two-fold: to predict the volume of CR space and to predict the centroid location of CR space, given a model internal parameterization. For each regression task, we assume that there exists a function:

$$y = f(\vec{x}), x \in \mathcal{X} \subset \mathbb{R}^n, y \in \mathbb{R},$$

which approximates a map of CR space as a function of internal model parameterization, which comprises the following training set:

$$D_{train} = \{x_j^t, f(x_j^t)\}$$

for $j = 1, \dots, n$. The NN model uses a mean-squared error (MSE) loss function, given by the following:

$$L = \sum_{i=1}^n (f(x_i^t) - \hat{f}(x_i^t))^2,$$

where f_i is the ground truth value for either the CR volume or centroid and $\hat{f}_i$ is the value regressed by the NN. In this scheme, we utilize a four-layer fully connected NN with a 256-Dropout-128-Dropout architecture. The dropout layer⁴² serves to provide a stochastic variability to the output of the NN.

We discretize the model's internal parameter space into nine bins, representing augmentations or inhibitions to specific cytokine pathways, giving 40,353,607 potential internal model parameterizations. We begin by pre-selecting 10,000 of these internal parameterizations randomly; this random selection then makes up the available pool, $\mathcal{P}$, of unlabeled data. From this pool, we begin the AL procedure by selecting 100 internal parameterizations randomly from $\mathcal{P}$. These internal parameterizations are then fed into the lower-level AL workflow, which is used to map the CR space and return an approximate volume and center-point. This data is then used to train the upper-level neural net (see Figure 1). The trained NN is then used to predict the volume or centroid location for the remaining unlabeled data for 10 stochastic replicates (the dropout layer provides stochasticity). The parameterizations from $\mathcal{P}$ that have the highest variance are selected for labeling, and this process repeats. The pseudocode for this procedure is given below.

1. Initialize training pool $\mathcal{P}_U$; upper-level dataset of internal parameterization vector/CR volume, centroid location pairs, D_{IP} , z_u , the maximum size of the dataset; and m_u samples to be added on each iteration,
2. While $|D_{IP}| < z_u$.
 - a. For each internal parameterization vector i in D_{IP} .
 - i. Initialize training pool $\mathcal{P}_L$; lower-level dataset of coordinate/class pairs, D_{EP} , z_l , the maximum size of the dataset; and m_l samples to be added on each iteration.
 - ii. While $|D_{EP}| < z_l$:
 1. train lower-level network on D_{EP}
 2. obtain rank r_j for each x_j in $\mathcal{P}_L$ according to maximum class-uncertainty;
 3. label the set of m_l coordinate sets from $\mathcal{P}_L$;
 4. add the annotated data to D_{EP}
 5. calculate stopping metrics, stop if appropriate.
 - b. Train the upper-level network on D_{IP}
 - c. Obtain rank r_i for each x_i in $\mathcal{P}_U$ according to maximum regression variance.
 - d. Label the set of m_u parameterizations from $\mathcal{P}_u$.
 - e. Add the annotated data to D_{IP}

- f. Calculate stopping metrics, stop if appropriate.

We note that the upper-level dataset D_{IP} consists of a set of vectors containing inhibition/augmentation factors for the cytokine pathways in the model (11d floating point vectors with values between 0.01 and 10) with their associated CR volumes and centroid locations, while the lower-level dataset D_{EP} consists of a set of 4d coordinates describing the injury/perturbation severity as well as its associated class (CR or not CR).

Source code and input data can be found at https://github.com/An-Cockrell/IIRABM_NestedAL.

3. Results

In the lower-level AL workflow, we map CR space as a function of four parameters, external to the IIRABM's internal rule set. An example of this space can be seen in Figure 2. In this figure, outcome spaces for patients with low environmental toxicity (toxicity = 1) to high environmental toxicity (toxicity = 10) are shown from left to right. Each point represents 4000 in silico patients (40 injury sizes, 100 stochastic replicates). Points are color-coded based on the outcomes generated. The CR space is shown in green.

We utilized seven different ML models to map the CR space: the Artificial Neural Network (ANN),⁴³ Adaptive Boosting,⁴⁴ Naïve Bayesian,⁴⁵ Random Forest,⁴⁶ TreeBag,⁴⁷ AdaBoost M1,⁴⁸ and Bag – Flexible Discriminant Analysis with Generalized Cross Validation.⁴⁹ We compare an ensemble of models to ensure we have selected an appropriate ML strategy to explore the model in the most efficient manner. Results from this are shown in Figure 3, which displays the F -core as a function of AL iteration number (and by proxy, dataset size).

It is readily apparent that a NN is the best type of ML model for mapping this space. By iteration 10, which uses 1000 parameterizations (out of 8800 possible), we can achieve an average class-prediction accuracy of > 98%. The resulting ML model is then utilized to efficiently calculate the location and centroid of the CR space and train the upper-level neural net. This is due to the ease with which NNs can approximate nonlinear functions.⁵⁰ We present an illustration of this in Figure 4, which displays three unique CR spaces for three unique internal model parameterizations, in a three-dimensional slice of a four-dimensional perturbation-parameter space. The legend refers to an exponent determining the strength of augmentation or inhibition for all pathways; the meaning of and rationale for this exponent are described in detail by Cockrell and An.⁶ Red points labeled as “Unique –1” are those points that are unique to the parameterization in which all protein synthesis pathways are inhibited by 90% (10^{-1} of the uninhibited pathway secretion). Green points are unique to the parameterization in which all protein synthesis pathways are augmented by a factor of 10^1 . Teal points are unique to the parameterization in which all protein synthesis pathways are unchanged. Black points are those shared by the maximally inhibited and unchanged protein synthesis pathways.

Results from the upper-level AL procedure are shown in Figure 5. In Figure 5(a), we display the percent volume error as a function of the number of training samples for AL, Random Sampling (RS), and “Actively Not Learning” (ANL). ANL refers to utilizing the opposite of

the AL sampling criterion. In this case, for AL we chose samples that maximized prediction variance; for ANL, we chose samples that minimized prediction variance. As expected, AL outperforms RS (by approximately 10%) and requires fewer samples to converge to the error minimum. In addition, both methods significantly outperform ANL, as expected. In Figure 5(b), we show the standard deviation of the error for the previous three methods. Here, AL significantly outperforms RS in that the intelligent sampling criterion leads to a suite of models with a larger degree of precision in the volume prediction, whereas the changes in standard deviation of the error are minimal for ANL and minimal for RS after the first few samples. Figure 5(c) displays the error (as a Euclidean distance from the predicted centroid point to the true value) as a function of the number of samples. Once again, AL outperforms RS, although by a relatively modest amount.

4. Discussion

One of the primary goals and benefits of agent-based modeling is to use the computational model as an experimental proxy system⁵¹ that is detailed enough to capture the vital aspects of patients and their associated clinical setting. The quest for increasingly detailed representation when using ABMs is a manifestation of the concurrent increase in mechanistic knowledge acquired from ongoing research; the consequence of these highly detailed models is that they contain a multitude of parameters within a highly connected set of rules and/or equations. The perception of intractability of effectively parameterizing such models is considered a major limitation in their use. Our prior work has demonstrated the need to operate with these models across a wide range of their parameter space, which we consider a representation of genetic variability with regards to pathway responsiveness among a clinical population.³ The work presented herein provides a demonstration of methods with which these parameter spaces can be identified.

We have described a nested AL workflow that efficiently and accurately can characterize a high-dimensional and complex mechanism-based multi-scale (in this case, linking molecular events to cell behaviors that constitute a tissue) model. We remove inefficiencies due to oversampling small regions of the parameter space using the Monte Carlo Dropout Uncertainty Estimation approach.⁴⁰ We note that AL outperforms RS in both the volume and centroid location predictions, but the greatest strength comes from the significant increase in precision generated by a suite of AL-trained models. This work demonstrates that comprehensive (and accurate) exploration of computational models with many parameters is both possible and computationally tractable, given current techniques in ML and HPC. In addition, while in some circumstances the gains offered by AL are modest, it does help to minimize the cost of the computation.

In order to increase the utility of this work, we will develop higher resolution maps by labeling external parameter points individually rather than regressing the volume and centroid of the CR space. This will likely include collapsing the nested AL structure into a single network that takes internal and external parameterizations simultaneously, sacrificing some efficiency gained by the nested structure for increased precision in class identification. Pre-trained models can be incorporated into the development and training

of control strategies by enabling the selection of disease conditions with specific cytokine dynamics and mortality rates without a computationally expensive search process.

We note that this work is most relevant for ABMs, as opposed to equation-based models, as there exists no mathematical formalism that can be used to explore model behaviors and outcomes from the rule set/parameterization alone. Exploration requires simulation, which bears a cost. Application of this methodology reduces the cost significantly, with the caveat that the results are accurate approximations.

The ultimate goal of this method when applied to any computational model would be the generation of a relatively compact data structure, in this case, one or more pre-trained NNs, which accurately predicts some feature of interest in the computational model; in this case, that feature is whether or not a specific individual (internal model parameterization/model content) will certainly live or die when it experiences a specific perturbation (perturbation parameterization/model context). The appropriate threshold for accuracy can be determined based on the specific requirements of the application to which it is applied as well as considering the diminishing returns to ML model prediction accuracy as the number of training samples increases.

Funding

This work is based upon work supported by the U.S. Department of Energy, Office of Science (contract DE-AC02-06CH11357) and the National Institutes of Health (grant U01EB025825). This project used HPC resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy (contract DE-AC02-05CH11231), as well as resources provided by the Vermont Advanced Computing Core (VACC). In addition, this research was supported in part by the NIH through resources provided by the University of Chicago Computation Institute (Beagle2) and the Biological Sciences Division of the University of Chicago and Argonne National Laboratory (grant 1S10OD018495-01 and contract DE-AC02-06CH11357).

Author biographies

Chase Cockrell is an Assistant Professor in the Department of Surgery at the University of Vermont Larner College of Medicine. Dr. Cockrell's research focuses on the synthesis of agent-based modeling, high-performance computing, and machine learning to investigate complex biomedical questions.

Jonathan Ozik is a Computational Scientist at Argonne National Laboratory and Senior Scientist with Public Health Sciences Affiliation in the Consortium for Advanced Science and Engineering at the University of Chicago. Dr. Ozik develops applications of large-scale agent-based models, including models of infectious diseases, healthcare interventions, biological systems, water use and management, critical materials supply chains, and critical infrastructure. He also applies large-scale model exploration across modeling methods, including agent-based modeling, microsimulation and machine/deep learning. Dr. Ozik leads the Repast project (repast.github.io) for agent-based modeling toolkits and the Extreme-scale Model Exploration with Swift (EMEWS) framework for large-scale model exploration capabilities on high performance computing resources (emews.github.io). Dr. Ozik has been awarded an R&D 100 award in 2018 for contributions to the Swift/T workflow software

and recognized as a Finalist for the 2020 Gordon Bell Special Prize for High Performance Computing-Based COVID-19 Research.

Nick Collier is a Software Engineer at Argonne National Laboratory and Staff Software Engineer in the Consortium for Advanced Science and Engineering at the University of Chicago. Dr. Collier develops, architects, and implements large-scale agent-based models and frameworks in a variety of application areas, including the transmission of infectious diseases, biological systems, and critical materials supply chains.

Gary An is a Professor of Surgery and Vice-Chairman for Surgical Research in the Department of Surgery at the University of Vermont Larner College of Medicine. He is a founding member of the Society of Complexity in Acute Illness and past president of the Swarm Development Group, one of the original organizations promoting the use of agent-based modeling for scientific investigation. In addition to being an active trauma/critical care surgeon he has worked on the application of complex systems analysis to sepsis and inflammation since 1999. His work consists of development of mechanism-based computer simulations and integration of machine learning and artificial intelligence with multi-scale simulation models for discovery of therapeutic control modalities.

List of abbreviations

AL	active learning
CR	clinically relevant
ANN	Artificial Neural Network
ABM	agent-based model
EMEWS	Extreme-Scale Model Exploration with Swift
HPC	high-performance computing
ML	machine learning
IIRABM	Innate Immune Response Agent Based Model
MSE	mean-squared error

References

1. Gambhir SS, Ge TJ, Vermesh O, et al. Toward achieving precision health. *Sci Translat Med* 2018; 10: eaao3612.
2. Morrison TM, Pathmanathan P, Adwan M, et al. Advancing regulatory science with computational modeling for medical devices at the FDA's Office of Science and Engineering Laboratories. *Frontiers Med* 2018; 5: 241.
3. Cockrell C and An G. Sepsis reconsidered: identifying novel metrics for behavioral landscape characterization with a high-performance computing implementation of an agent-based model. *J Theor Biol* 2017; 430: 157–168. [PubMed: 28728997]
4. An G In silico experiments of existing and hypothetical cytokine-directed clinical trials using agent-based modeling. *Crit Care Med* 2004; 32: 2050–2060. [PubMed: 15483414]

5. Wood KA and Angus DC. Pharmacoeconomic implications of new therapies in sepsis. *Pharmacoeconomics* 2004; 22: 895–906. [PubMed: 15362927]
6. Cockrell RC and An G. Examining the controllability of sepsis using genetic algorithms on an agent-based model of systemic inflammation. *PLoS Computat Biol* 2018; 14: e1005876.
7. An G, Nieman G and Vodovotz Y. Computational and systems biology in trauma and sepsis: current state and future perspectives. *Int J Burn Trauma* 2012; 2: 1–10.
8. Goldman D, Bateman RM and Ellis CG. Effect of sepsis on skeletal muscle oxygen consumption and tissue oxygenation: interpreting capillary oxygen transport data using a mathematical model. *Am J Physiol Heart Circulat Physiol* 2004; 287: H2535–H2544.
9. Vodovotz Y and Billiar TR. In silico modeling: methods and applications tot trauma and sepsis. *Crit Care Med* 2013; 41: 2008. [PubMed: 23863232]
10. Vodovotz Y Computational modelling of the inflammatory response in trauma, sepsis and wound healing: implications for modelling resilience. *Interface Focus* 2014; 4: 20140004. [PubMed: 25285195]
11. Siqueira-Batista R, Gomes AP, Possi MA, et al. Computational modeling of sepsis: perspectives for in silico investigation of antimicrobial therapy. In: II international conference on antimicrobial research (ICAR2012), vol. 1 (eds Siqueira-Batista R, Gomes A, Possi M, et al.), Lisbon, Portugal, 2012, p. 368. Lisbon: ICAR - Book of Abstracts.
12. An G, Nieman G and Vodovotz Y. Toward computational identification of multiscale “tipping points” in acute inflammation and multiple organ failure. *Ann Biomed Eng* 2012; 40: 2414–2424. [PubMed: 22527009]
13. Karp RM. On the computational complexity of combinatorial problems. *Networks* 1975; 5: 45–68.
14. Sneddon MW, Faeder JR and Emonet T. Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. *Nat Meth* 2011; 8: 177–183.
15. Hopfield JJ and Tank DW. “Neural” computation of decisions in optimization problems. *Biol Cybern* 1985; 52: 141–152. [PubMed: 4027280]
16. Edwards R and Glass L. Combinatorial explosion in model gene networks. *Chaos* 2000; 10: 691–704. [PubMed: 12779419]
17. Neumann F and Witt C. Bioinspired computation in combinatorial optimization: Algorithms and their computational complexity. In: proceedings of the 15th annual conference companion on genetic and evolutionary computation, Amsterdam, The Netherlands, 6 July 2013, pp.567–590. New York: Association for Computing Machinery. Springer.
18. Cockrell C and An G. Genetic Algorithms for model refinement and rule discovery in a high-dimensional agent-based model of inflammation. *bioRxiv*. 2019:790394.
19. Saltelli A, Ratto M, Andres T, et al. Global sensitivity analysis: the primer. Chichester, West Sussex, UK: John Wiley & Sons, 2008.
20. Cukier R, Levine H and Shuler K. Nonlinear sensitivity analysis of multiparameter model systems. *J Computat Phys* 1978; 26: 1–42.
21. Saltelli A, Tarantola S, Campolongo F, et al. Sensitivity analysis in practice: a guide to assessing scientific models. New York: Wiley, 2004.
22. Ling Y and Mahadevan S. Quantitative model validation techniques: new insights. *Reliab Eng Syst Saf* 2013; 111: 217–231.
23. Macal CM. Model verification and validation. In: Workshop on threat anticipation: social science methods and models, Chicago, IL, 7–9 April 2005.
24. Calvez B and Hutzler G. Parameter space exploration of agent-based models. In: international conference on knowledge-based and intelligent information and engineering systems, vol. 3684 (eds Khosla R, Howlett RJ and Jain LC. Berlin, Heidelberg: Springer, 2005, 10.1007/11554028_88.
25. Abramson D, Bethwaite B, Enticott C, et al. Parameter space exploration using scientific workflows. In: Allen G, Nabrzyski J, Seidel E, et al. (eds) Computational Science – ICCS 2009. Lecture Notes in Computer Science, vol. 5544. Berlin, Heidelberg: Springer, 2009, 10.1007/978-3-642-01970-8_11
26. Carley KM. Validating computational models, <http://www.casos.cs.cmu.edu/publications/papers.php> (1996, accessed 17 September 2019).

27. Saltelli A and Annoni P. How to avoid a perfunctory sensitivity analysis. *Environ Model Softw* 2010; 25: 1508–1517.
28. Ratto M, Pagano A and Young P. State dependent parameter metamodeling and sensitivity analysis. *Comput Phys Commun* 2007; 177: 863–876.
29. Cockrell C, Christley S and An G. Investigation of inflammation and tissue patterning in the gut using a Spatially Explicit General-purpose Model of Enteric Tissue (SEGMENT). *PLoS Computat Biol* 2014; 10: e1003507.
30. Ozik J, Collier N, Heiland R, et al. Learning-accelerated discovery of immune-tumour interactions. *Mol Syst Des Eng* 2019; 4: 747–760. [PubMed: 31497314]
31. Ozik J, Collier NT, Wozniak JM, et al. Extreme-scale dynamic exploration of a distributed agent-based model with the EMEWS Framework. *IEEE Trans Computat Soc Syst* 2018; 5: 884–895.
32. Ozik J, Collier N, Wozniak JM, et al. High-throughput cancer hypothesis testing with an integrated PhysiCell-EMEWS workflow. *BMC Bioinform* 2018; 19: 483.
33. Ozik J, Collier NT, Wozniak JM, et al. From desktop to large-scale model exploration with Swift/T. In: 2016 winter simulation conference (WSC), Washington, DC, 11–14 December 2016, pp.206–220. IEEE, 10.1109/WSC.2016.7822090.
34. Wozniak JM, Armstrong TG, Wilde M, et al. Swift/T: Large-scale application composition via distributed-memory dataflow processing. In: 2013 13th IEEE/ACM international symposium on cluster, cloud, and grid computing, Delft, Netherlands, 13–16 May 2013, pp.95–102. IEEE, 10.1109/CCGrid.2013.99.
35. Wozniak JM, Jain R, Balaprakash P, et al. CANDLE/Supervisor: a workflow framework for machine learning applied to cancer research. *BMC Bioinform* 2018; 19: 491.
36. Cohn DA, Ghahramani Z and Jordan MI. Active learning with statistical models. *J Artif Intell Res* 1996; 4: 129–145.
37. Brinker K On active learning in multi-label classification. In: Spiliopoulou M, Kruse R, Borgelt C, et al. (eds) *From data and information analysis to knowledge engineering. Studies in classification, data analysis, and knowledge organization*. Berlin, Heidelberg: Springer. 2006, 10.1007/3-540-31314-1_24
38. Huang SJ, Jin R and Zhou ZH. Active learning by querying informative and representative examples. *Adv Neural Inf Proc Syst* 2010: 892–900.
39. Schein AI and Ungar LH. Active learning for logistic regression: an evaluation. *Mach Learn* 2007; 68: 235–265.
40. Tsymbalov E, Panov M and Shapeev A. Dropout-based active learning for regression. In: van der Aalst W, et al. (eds) *Analysis of images, social networks and texts. AIST 2018. Lecture Notes in Computer Science*, 2018, vol. 11179. Springer, Cham, 10.1007/978-3-030-11027-7_24
41. De Boer P-T, Kroese DP, Mannor S, et al. A tutorial on the cross-entropy method. *Ann Oper Res* 2005; 134: 19–67.
42. Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res* 2014; 15: 1929–1958.
43. White H *Artificial neural networks: approximation and learning theory*. Cambridge, MA: Blackwell Publishers, Inc., 1992.
44. Rojas R *AdaBoost and the super bowl of classifiers a tutorial introduction to adaptive boosting*. Technical Reports, 2009.
45. Kononenko I Semi-naive Bayesian classifier. In: *European working session on learning*, 6 3 1991, pp.206–219. Berlin, Heidelberg: Springer, 10.1007/BFb0017015
46. Ho TK. Random decision forests. In: *proceedings of 3rd international conference on document analysis and recognition*, Montreal, Quebec, Canada, 14–16 August 1995, vol. 1, pp. 278–282. IEEE. 10.1109/ICDAR.1995.598994
47. Breiman L Bagging predictors. *Mach Learn* 1996; 24: 123–140.
48. Freund Y and Schapire RE. Experiments with a new boosting algorithm. In: *international conference on machine learning*, Bari, Murray Hill, NJ, 3–6 July 1996, vol. 96, pp.148–156.
49. Hastie T, Tibshirani R and Buja A. Flexible discriminant analysis by optimal scoring. *J Am Stat Assoc* 1994; 89: 1255–1270.

50. Barron AR. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans Inform Theor* 1993; 39: 930–945.
51. An G, Fitzpatrick BG, Christley S, et al. Optimization and control of agent-based models in biology: a perspective. *Bull Math Biol* 2017; 79: 63–87. [PubMed: 27826879]

Figure 1.

Diagram illustrating the nested active learning (AL) workflow. The upper level depicts the AL used to evaluate the internal parameterization of the model (e.g., those that directly influence the behavior of the signaling pathways and cellular behaviors of the Innate Immune Response Agent-Based Model). This process has an internal AL loop that evaluates behavior of a particular combination of internal parameters in terms of its clinically relevant (CR) behavior across a set of parameters defining properties of infecting microbes, environmental contamination, and level of systemic resilience of the host. The lower-level AL loop has a stopping criterion that determines whether a particular parameter combination re-enters the higher level AL loop, at which point the result is fed to the overall parameter space map. ML: machine learning.

Figure 2.

Outcome spaces for patients with low environmental toxicity (toxicity = 1) to high environmental toxicity (toxicity = 10) are shown from left to right. Each point represents 4000 in silico patients (40 injury sizes, each with 100 stochastic replicates). Points are color-coded based on the outcomes generated. Blue points represent simulations that healed under all circumstances. Red points represent simulations that always died from overwhelming infection. Black points represent simulations that either died from overwhelming infection or healed completely. Red points represent simulations that either died from overwhelming infection or hyperinflammatory system failure. Green points represent the clinically relevant simulations, as these parameter sets lead to all possible outcomes. (Color online only.)

Figure 3.

Results from lower-level active learning (AL) active learning – clinically relevant (CR) space (see Figure 2) is mapped as a function of four parameters, external to the Innate Immune Response Agent-Based Model’s internal rule set. We utilized seven different machine learning models to map the CR space: the Artificial Neural Net, Adaptive Boosting, Naïve Bayesian, Random Forest, TreeBag, AdaBoost MI, Bag – Flexible Discriminant Analysis with Generalized Cross Validation. The F -score is shown on the y -axis as a function of the number of AL iterations performed.

Figure 4.

Clinically relevant space for varying internal parameterizations: clinically relevant space for three unique internal parameterizations is mapped as a function of three parameters external to the Innate Immune Response Agent-Based Model's rule set, with the environmental toxicity parameter held fixed at 2 (a low level). The legend refers to an exponent determining the strength of augmentation or inhibition for all pathways. Red points labeled as "Unique -1" are those points that are unique to the parameterization in which all protein synthesis pathways are inhibited by 90% (10^{-1} of the uninhibited pathway secretion). Green points are unique to the parameterization in which all protein synthesis pathways are augmented by a factor of 10^1 . Teal points are unique to the parameterization in which all protein synthesis pathways are unchanged. Black points are those shared by the maximally inhibited and unchanged protein synthesis pathways. (Color online only.)

Figure 5.

Results from upper-level active learning (AL). (a) The percent volume error as a function of the number of input training samples using AL, Random Sampling (RS), and Actively Not Learning (ANL), in which the learning criterion is the opposite of the AL criterion. We see that AL arrives at a more accurate prediction with fewer samples than RS or ANL. (b) The standard deviation of the error of the volume prediction for the three above methods; note that AL not only generates a suite of more accurate models, but also has a much higher degree of precision. (c) The error (in this case a Euclidean distance) in the centroid location prediction. AL once again outperforms RL.