Integrating Predictive Visualization with the Epidemic Disease Simulation System (EpiSimS)

Chris Bryan, Susan Mniszewski, and Kwan-Liu Ma, *Fellow, IEEE*

Abstract— The Epidemic Simulation System (EpiSimS) is a scalable, complex model for analyzing disease spread within the United States. Due to the high-dimensional parameter space, the long completion time, and the large amount of output data in a single EpiSimS run, simulating the entire input parameter space is unfeasible. By taking a granular sampling of the parameter and aggregate outcome space, regression algorithms can predict outcomes that a particular parameter combination lead to, without having to actually run the simulation. These predictions are viewable using traditional epidemic visualization components: aggregate line charts and spatio-temporal mapping. Our ongoing effort is to integrate predictive algorithms into our base EpiSimS viewer, where a user can load completed runs to build a classifier, choose an arbitrary input parameter set, and see the predicted outcomes using visual means. Our future work involves developing faster and more accurate prediction algorithms into our system and leveraging prediction to enable specific location classification and response countermeasures.

1 Introduction

Disease spread is a complex problem in today's globalized world. Introduction of new pathogens to a susceptible population can potentially lead to a pandemic, as susceptible hosts have little defense against exotic infectors. A number of factors influence a disease's impact, including transmission rate, incubation period, antiviral supplies, and population dynamics. Mosquito-borne illnesses such as chikungunya [5] have had recent outbreaks in North America, highlighting the need to understand the critical parameters in the spread and diffusion of these diseases. Simulation is one way to do this.

Agent-based models (ABMs) are used to simulate the spread of infectious diseases through a population. The Epidemic Simulation System (EpiSimS) is an discrete-event ABM for this task [16]. A single EpiSimS run generates a large set of temporal, geospatial, and multivariate data. It includes population, infectious mosquito vectors, location attributes, and sets of logged events noting times and changes in an agent's disease progression state. Even for a smaller-tier metropolitan area such as Washington DC (approximately 500,000 people), a single run can contain many gigabytes of raw output data. More problematic than the data size, ABM simulations in general can have complex runtimes. The compute and time-resources required for EpiSimS runs can scale to hundreds of compute nodes and many hours for larger population sizes and more complex epidemic models.

For these reasons, it is impractical to simulate for every possible combination of input parameter space, especially when some parameters have a granularity level down to distinct locations and human demographic groups. Working with EpiSimS scientists and researchers, we are integrating predictive analytics into our EpiSimS analysis tool. This allows a user to visually view and analyze the likely outcome of a simulation without having to actually run it; instead the user can simply set the pertinent input parameters and perform the predictions using a regression-based classifier.

This short technical paper describes the process of integrating predictive analytics into the current EpiSimS viewer and how it visualizes the predicted outcomes. Our approach is to take a previously completed set of EpiSimS runs, which represent a coarse-grained sampling, and use these to build a classifier. The classifier predicts output scenarios, both temporally and spatially, when a user selects a combination of input parameters. We utilize the Weka API and software library of algorithms, integrating it into the EpiSimS viewer, to build and run the classifier, perform regressions, and predict outcomes. These predictions are displayed in the viewer's output components, namely a line chart showing statistical temporal data, and a map view showing spatio-temporal disease spread. The predicted outcomes can be analyzed with the other loaded EpiSimS runs.

2 System Components and Background

Our system is composed of three pars: EpiSImS data, an interactive viewer, and a prediction module.

2.1 EpiSimS

EpiSimS (or Epidemic Simulation System) is a large-scale, discrete-event, ABM for infectious diseases in the United States. It is highly customizable in disease states and progression, human demographics and behavior patterns, potential countermeasure actions, and geographic location attributes. For prior research using the EpiSimS system, see [10, 7, 9, 16]. The EpiSimS team recently integrated a patch-based (also called a vector-based) model of infectious disease spread into their simulation [8], that allows for simulation of mosquito-borne illnesses. In these scenarios, mosquitoes infect humans, which then mosquitoes, and so on. Mosquito disease dynamics are represented by an ODE model. If infected, humans progressed through a series of stages, before eventually recovering.

2.2 Disease Visualization and the EpiSimS Viewer

Working with EpiSimS scientists, we built a viewer to analyze EpiSimS results. Visualization has long played an important tool in assisting both the study and analysis of disease spread. Traditional disease visualizations make heavy use of statistical charts or map views to show different aspects of the data, such as temporal or spatial progression[4, 1, 11, 2], and form the base of our system.

Before being able to view a completed EpiSimS run, the raw output data must be pre-processed in a data-management application and uploaded. The processed data is stored in a PostgreSQL database, and can then be queried and interactively analyzed in the viewer, which is built using Processing and Java. Multiple runs can be loaded into the viewer for analysis, and both the input parameters and output data can be shown.

Selected input parameters are shown in a parallel coordinates view. For this current work, only a subset of the parameters were analyzed: the disease transmission rate (β), the mosquito densities (K_v) at locations, and the relative safety (of being bit by a mosquito) when a person is indoors vs when the person is outdoors (*homeIo, workIo, and schoolIo*). For viewing output components, a user can load different visualizations, such as line charts, maps, or clusterings. For

- *Chris Bryan is with the University of California, Davis. E-mail: cjbryan@ucdavis.edu.*
- *Susan Mniszewski is with Los Alamos National Laboratory. E-mail: smm@lanl.gov.*
- *Kwan-Liu Ma is with the University of California, Davis. E-mail: ma@cs.ucdavis.edu.*

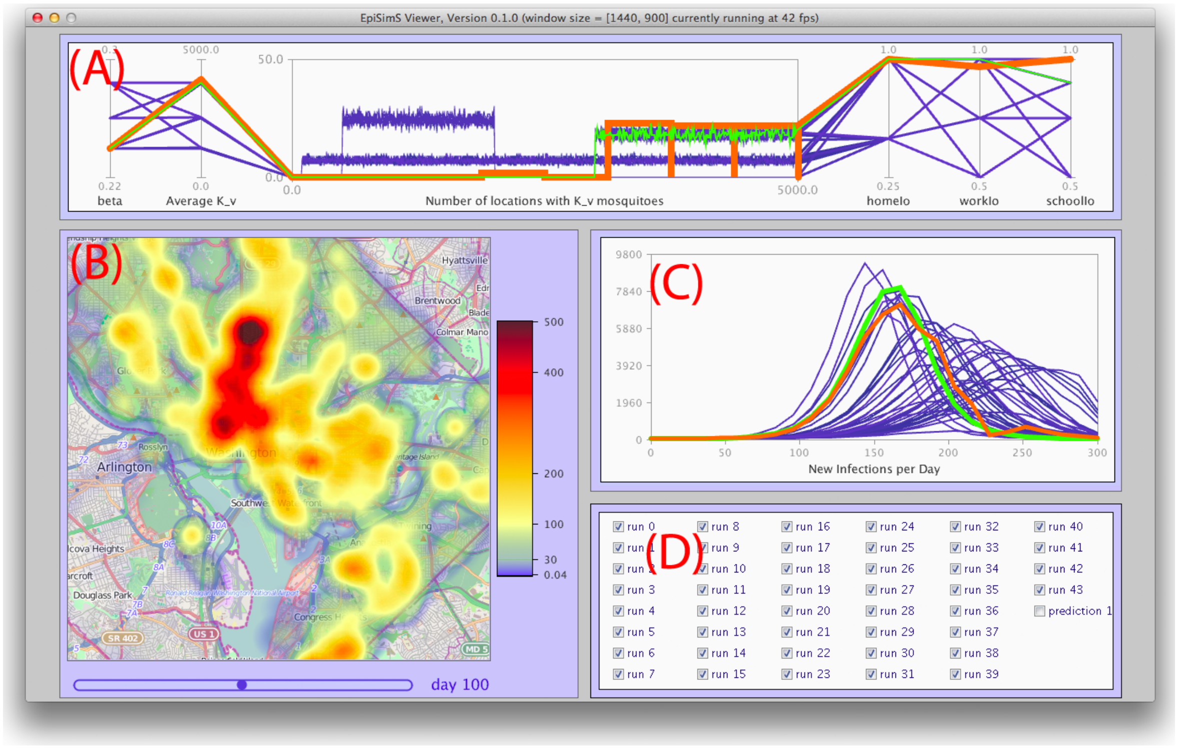


Fig. 1: The EpiSimS viewer, with four components noted: (A) input parameters, (B) map view, (C) output lines chart, and (D) currently loaded runs. The highlighted orange polyline shows in (A) shows the user selected input parameters, which closely mimic an already-run EpiSimS simulation (highlighted in green). The orange line in (C) shows the predicted output line chart and the map view (B) shows the predicted density for day 100, based on the user-defined combination of input parameters in component (A). The green line in (C) shows the highlighted EpiSimS run.

the current work, we limit output views only to what we predict on: the output lines chart and the map viewer. The lines chart shows aggregated statistical information over time, such as "new infections per day," or "attack rate per day." The map shows the spatial diffusion of the epidemic for a single run on a selected day, based on a kernel density estimation [14] using a normal distribution function. For interactive levels of response, in the data pre-processing steps we store spatial density data hierarchically in a pyramid grid tiling scheme. This is similar to how tiles in slippy maps are stored (see [13] for an example). Data is first stored at the lowest tile or region grains, and then at successively higher, aggregated, levels in a tiled grid. For rendering a map view, the viewer needs only to return grid points based on the current view and zoom and then build the density layer based upon the grid's infection values.

2.3 Prediction

Machine learning and data mining can be utilized for predicting outcome targets, based on training from prior, known data. Prediction itself has been extensively used in real-world disease forecasting; some examples include [12, 17], and it has also been used in modeling epidemic simulations [15]. In our case, we are making predictions using the EpiSimS runs themsevles, to predict likely output based on a user-defined set of input parameters.

To accomplish this, we use the the Weka API and libraries. Weka is a suite of software for data mining tasks [3]. It incorporates several standard machine learning algorithms and techniques (and allows for development of new ones) for pre-processing, classification, training, regression, clustering, association rules, and even visualization. It can be used as a standalone tool, using input files to test, train, and predict on, or it can be integrated into an existing project. In the Weka context, the machine learning algorithm is called a classifier. The classifier is built using a training set of data, and then can be tested for accuracy or used to make predictions.

3 Integrating Prediction into the EpiSimS Viewer

This section describes our current work in integrating Weka into the EpiSimS viewer. There are four main components in the viewer that we interact with for prediction, as shown in Figure 1. These are: (A) the currently loaded input parameters, (B) the map view, (C) the output line chart view, and (D) the currently loaded runs list. Before being able to make a prediction, the user must select a set of runs to load into the viewer and selects the input attributes they want to use. The loaded simulations are shown in the runs list (D), and the input parameters are shown in the parallel coordinates view (A).

Each polyline in the parallel coordinates component (Figure 1A) represents a single run. There are two types of input parameters that can be represented. Some parameters, such as β (which is the epidemic transmission rate) are a single constant value for a run. Others, such as the "location count per K_v," have specific values set for each location within the simulation. For this particular case, there are 5000 potential K_v values on the x-axis that a location can fall into, where K_v denotes number of mosquitoes at that location. The y-axis here denotes the number of locations with that K_v value. To avoid having to account for all 5000 potential spaces invidividually when making our prediction, we instead aggregate into a smaller, user-defined number of buckets and take the average value from those buckets (see Figure 2). We then treat each bucket as an input attribute for use in prediction.

The goal of prediction is to view the likely output based on the input parameters chosen by a user. For example, if there are five single parameters and one location-specific parameter that's been broken into eight buckets (as in Figure 1), the input parameter space contains thirteen input attributes to build our classifier. To this, we add line chart and map density point data. The classifier is trained on this data, and then can predict output data for the line chart and the map. Since both the line chart and map have a large domain (the line chart shows 300 simulated days and the map covers 35,000 raw locations for the loaded simulations in Figure 1), we use sampling and aggregation to

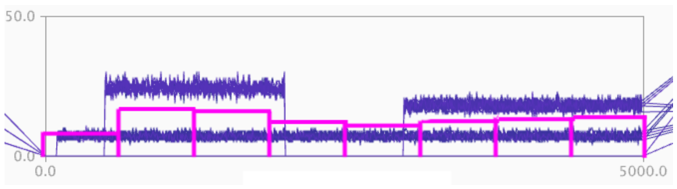


Fig. 2: Breaking up an input parameter with 5000 potential slots into eight buckets (shown in pink). The number of buckets can be set by the user, and the value for each bucket is the average of all the values inside.

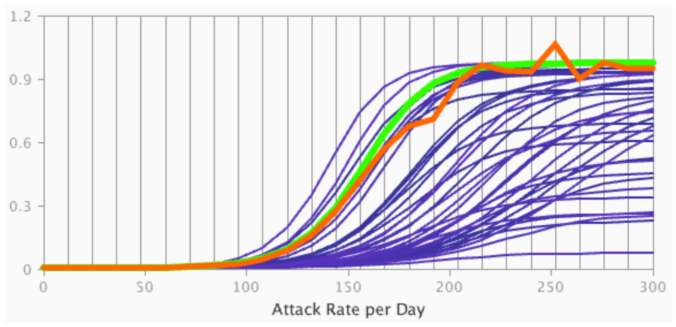


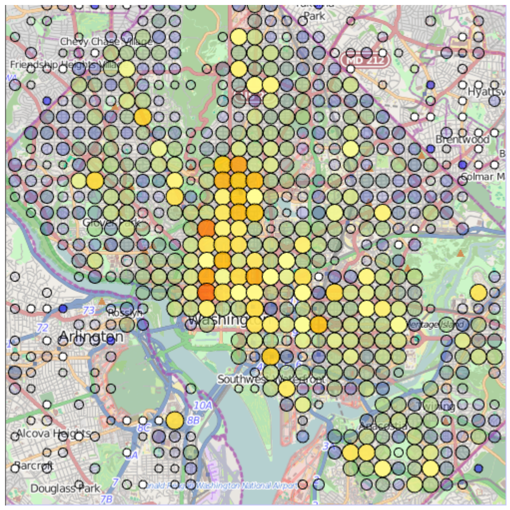
Fig. 3: An example of predicting the attack rate output over time. For the time-dependent predictions in the line chart, we first discretize the prior runs (the blue lines) into discrete, sampled steps, and use a normal distribution function to smooth the data for the input runs. In this case, 25 days have been sampled and smoothed, as denoted by the vertical grey lines. We then regress on each sampled day to generate an overall prediction.

lessen the number of values needed to classify and predict on.

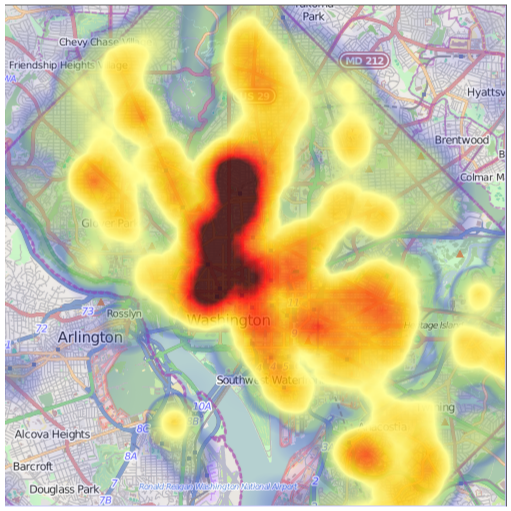
For the line chart, the user sets a number of days to sample on. An example of this is shown in Figure 3, where fifty days are sampled out of a total of 300. To avoid missing spikes and outliers, a normal distribution is applied on each sampled day to smooth it and and include the effect of the surrounding days. The set of sampled points for each completed run is included in building the classifier. When a parameter combination is made and a prediction done, there is an estimate for each sampled day.

For the map view, instead of predicting on each location individually, we utilize the tiled aggregation data points. Based on the map's latitude and longitude boundaries, and the current zoom level, a set of aggregated points that fill certain geographic, temporal, and zoom constraints are retrieved from the database. As these points already represent a more granular level of data, they require less overall work when used generate a classifier and make predictions. (Usually 200-300 per run, depending on the zoom level, map size, and day). Each density point is independently predicted (Figure 4a), and the KDE layer is then applied using the predicted values for that day (Figure 4b).

When using the WEKA API, you first choose an algorithm, and then feed it training data to build a classifier. After this, testing data can be used to check its accuracy, and predictions can be made. Since EpiSimS runs are time-consuming to compute and post-process, there are only a limited number available for training and testing purposes. Since we are using regression to predict numerical outcome values (for example, the total number of infected persons at a particular day in the line chart), we must choose a regression algorithm to classify and predict. Currently, we are using Weka's M5P regression tree algorithm. From the set of loaded runs in the viewer, we utilize a percentage of these (80%) for training to build the classifier, and the remainder (34%) for testing, if desired. Building the classifier is a one time process, based on the loaded data in the sim, although if more runs are added, then it can be rebuilt. When a user draws or selects a new input parameter configuration and chooses to predict its outputs, the classi-



(a)



(b)

Fig. 4: (a) A regular grid of points is used to build the KDE layer in the map, showing the spatial spread of the epidemic. A prediction is made for each point in the grid. (b) The result of applying the KDE based on the predicted values in the grid.

fier iterates through and predicts the output values for the lines chart and for the aggregate density points on the map.

Figure 1 shows an example of a completed prediction. The orange polyline in Figure 1A shows a user's input parameter combination. One particular EpiSimS run is very similar to this parameter combination, and it has been highlighted in green. Figure 1C shows the predicted line chart outcome in orange, and the similar run's output in green. The predicted density mapping at day 100 is shown in Figure 1B. In these two particular runs, there is a low transmission rate (β), but a high number of mosquitoes and high indoor vs outdoor infectivity at home, work, and school. These characteristics lead to a moderately high outbreak, although it would probably be even more extensive if the β value had been higher.

4 Conclusions and Future Work

This short paper goes over an ongoing project of integrating predictive analytics into the EpiSimS analytics viewer. EpiSimS is a large, complex, time-consuming simulation with a large input parameter space. By utilizing prediction, we can take a coarse-grained sampling of various parameter combinations and estimate spread, diffusion, and the overall lifecycle of EpiSimS runs without having to actually perform the simulations. This saves both time and compute resources. Our current efforts are focused on customizing and improving the classifiers and visualizations used in this data, making them more robust, accurate, and flexible within the current viewer. For example, our current spatial prediction predicts each point independently, without regard to the surrounding neighborhood. [6] is an example of a spatio-temporal prediction mapping that is much more refined than ours.

As a future direction, we are investigating using contextually novel or unorthodox visual plots and techniques for fine-grained location and region prediction. EpiSimS scientists are interested in predicting dangerous places, demographics, and disease warning signs, and seeing how localized countermeasures can affect the overall disease spread. If such places or signs are known or predicted prior to the disease becoming a full-blown pandemic, then countermeasures can be taken. Prior EpiSimS efforts have demonstrated that high-traffic locations such as schools or businesses can be a hotspot for spread. By generating fine-grained predictions about which individual locations or regions are likely to be catalysts for spread, a scientists could employ specific countermeasures and location closures early in disease, limiting its impact. [6] does a spatial prediction for resource allocation, but utilizes standard spatial mappings to do. We are working to leverage both innovative visualization techniques and predictive algorithms to assist these types of interactions and analysis.

REFERENCES

[1] S. Afzal, R. Maciejewski, and D. Ebert. Visual analytics decision support environment for epidemic modeling and response evaluation. In *IEEE VAST*, pages 191–200, Oct 2011.

[2] L. N. Carroll, A. P. Au, L. T. Detwiler, T. chieh Fu, I. S. Painter, and N. F. Abernethy. Visualization and analytics tools for infectious disease epidemiology: A systematic review. *Journal of Biomedical Informatics*, (0):–, 2014.

[3] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009.

[4] R. Maciejewski, S. Rudolph, R. Hafen, A. Abusalah, M. Yakout, M. Ouzzani, W. S. Cleveland, S. J. Grannis, and D. S. Ebert. A visual analytics approach to understanding spatiotemporal hotspots. *IEEE TVCG*, 16(2):205–220, Mar. 2010.

[5] D. Mackenzie. Threatwatch: chikungunya virus spreads in the americas. *New Scientist*, 220(2948), 2013.

[6] A. Malik, R. Maciejewski, S. Towers, S. McCullough, and D. Ebert. Proactive spatiotemporal resource allocation and predictive visual analytics for community policing and law enforcement. *IEEE Transactions on Visualization and Computer Graphics*, 99(PrePrints):1, 2014.

[7] S. Mniszewski, S. Y. DelValle, P. Stroud, J. Riese, and S. Sydoriak. Pandemic simulation of antivirals + school closures: buying time until strain-specific vaccine is available. *Computational and Mathematical Organization Theory*, 14(3):209–221, 2008.

[8] S. Mniszewski, C. Manore, C. Bryan, and D. Robert. Towards a hybrid agent-based model for mosquito borne disease. In *SummerSim 2014*, forthcoming.

[9] S. Mniszewski, S. Y. D. Valle, R. Priedhorsky, J. Hyman, and K. Hickman. Understanding the impact of face mask usage through epidemic simulation of large social networks. In *Theories and Simulations of Complex Social Systems*, pages 97–115. 2014.

[10] S. Mniszewski, S. Y. Del Valle, P. Stroud, J. Riese, and S. Sydoriak. Episims simulation of a multi-component strategy for pandemic influenza. In *SpringSim '08*, pages 556–563, 2008.

[11] J. Paparian, S. Brown, D. Burke, and J. Grefenstette. Fred navigator: An interactive system for visualizing results from large-scale epidemic simulations. *2012 IEEE 8th International Conference on E-Science*, 0:1–5, 2012.

[12] L.-Z. Peng, L.-X. Yi, and S.-Y. Hua. A new epidemic disease predicting method. *Intelligent Computation Technology and Automation, International Conference on*, 1:550–553, 2008.

[13] S. Quinn and M. Gahegan. A predictive model for frequently viewed tiles in a web map. *Transactions in GIS*, 14(2):193–216, 2010.

[14] B. Silverman. Density estimation for statistics and data analysis. In *Mono. on State and Appl. Probability*. Chapman & Hall, 1992.

[15] A. Skvortsov, B. Ristic, and C. Woodruff. Predicting an epidemic based on syndromic surveillance. In *Information Fusion (FUSION), 2010 13th Conference on*, pages 1–8, July 2010.

[16] P. Stroud, S. Del Valle, S. Sydoriak, J. Riese, and S. Mniszewski. Spatial dynamics of pandemic influenza in a massive artificial society. *Journal of Artificial Societies and Social Simulation*, 10(4):9, 2007.

[17] X. Zhou, Q. Li, Z. Zhu, H. Zhao, H. Tang, and Y. Feng. Monitoring epidemic alert levels by analyzing internet search volume. *IEEE Trans. Biomed. Engineering*, 60(2):446–452, 2013.