DataStorm: A Data-Enabled System for End-to-End Disaster Planning and Response*

Hans Walter Behrens
Arizona State University
Tempe, AZ, USA
hwb@asu.edu

Yash Garg
Arizona State University
Tempe, AZ, USA
ygarg@asu.edu

ABSTRACT

Data- and model-driven computer simulations are increasingly critical in many application domains. Yet, several critical data challenges remain in obtaining and leveraging simulations in decision making. Simulations may track 100s of parameters, spanning multiple layers and spatial-temporal frames, affected by complex inter-dependent dynamic processes. Moreover, due to the large numbers of unknowns, decision makers usually need to generate ensembles of stochastic realizations, requiring 10s-1000s of individual simulation instances. The situation on the ground evolves unpredictably, requiring continuously adaptive simulation ensembles. We introduce the DataStorm framework for simulation ensemble management, a system to enable end-end-end ensemble planning and optimization, including parameter-space sampling, output aggregation and alignment, as well as state and provenance data management. This system aims to work efficiently, producing results while working within a limited simulation budget, and incorporating a multivariate. It also incorporates a multivariate, spatiotemporal data browser to provide comprehensive data visualization and exploration tools, empowering decision-making based on these improved results.

CCS CONCEPTS

• **Human-centered computing → Scientific visualization**; **Geographic visualization**; *Visual analytics*; • **Computing methodologies → Simulation support systems**;

KEYWORDS

continuous simulations, simulation ensembles, decision support systems, vizualization techniques, multivariate time series

ACM Reference Format:
Hans Walter Behrens and Yash Garg. 2018. DataStorm: A Data-Enabled System for End-to-End Disaster Planning and Response. In *Proceedings of Data Visualization (CSE578 S18).* ACM, New York, NY, USA, Article 1, 9 pages. https://doi.org/****

*Submitted in partial fulfillment of the requirements for our CSE 578 final project. Work supported in part by NSF#1610282 "DataStorm: A Data Enabled System for End-to-End Disaster Planning and Response".

CSE578 S18, Spring 2018, Tempe, AZ USA

ACM ISBN ****…$0.00
https://doi.org/****

1 INTRODUCTION

Data- and model-driven computer simulations are increasingly critical in many application domains [6, 8, 13, 14, 16, 18]. For example, when predicting the evolution of epidemics and assessing the impact of interventions, experts often rely on epidemic models and simulation software, such as GLEaM [4] and STEM [9], and simulation ensemble tools, such as EpiDMS [14]. Similarly, data-driven computer simulations for disaster preparedness and response can play a key role in predicting the evolution of disasters and effectively managing emergencies through intervention measures [1].

Simulators are used in many different fields, from economics to engineering to medicine. However, such simulators require detailed domain-specific knowledge to accurately simulate the topic which they represent. The inputs, outputs, and computational or storage requirements of such simulators may differ dramatically between fields. However, despite these disparities, many novel results can be discovered by examining the interactions between such interdisciplinary simulations.

By synthesizing these individual, incompatible models into a single unified ensemble, such results become possible. However, it is not feasible to predict every possible combination of models; it is therefore desirable to create a unified framework for generating heterogeneous simulation ensembles from any domain, rather than choosing a subset of domains to model. First, we examine the possible impacts of creating such a system.

1.1 Case Study: Hurricane Harvey

One area in which these kinds of heterogeneous simulation ensembles could be widely used is in disaster planning and response. In August 2018, Hurricane Harvey made landfall in Houston, Texas. Although weather forecasters had used detailed models to estimate the time and location of landfall days in advance, residents still had less than 90 minutes of warning to evacuate their homes after the unplanned release of water from a flooded emergency spillway. Although this action was necessary to alleviate even worse flooding in Downtown Houston, no officials had warned the residents that this might take place. Instead of having several days, and potentially up to a week to move possessions to safer, or at least higher, ground, residents instead were forced to leave with almost no warning, greatly worsening their personal situations during the recovery period.

So, why did this situation occur? Currently, almost all disaster planning operates siloed within domain-specific simulation models. For example, although forecasters had executed many hurricane simulations to predict the track and winds of the hurricane, and even the rainfall levels, these results had not been cross-validated against a flooding model. Predicted flood levels were based on the

concept of 100-year floods, purely from historical and statistical data, rather than any existing simulations based on rainfall, elevations, or current building locations. Therefore after landfall, when observed rainfall exceeded historical predictions, flood plans needed to be updated in real-time, without having any existing disaster response plans in place. Although engineering plans were in place to mitigate flooding, it was not predicted that these plans would need to be used, so the affected residents did not receive any warning until shortly before the emergency spillways were opened.

If these models had been integrated together, the flooding simulations could have worked with more-accurate data based on current conditions, rather than statistical averages. The results would likely have been more accurate, and officials could have warned residents of potential flooding with many days of warning, rather than minutes. Damage assessments could have been reduced, evacuation could have been simplified, and a more effective response overall could have been realized.

1.2 Case Study: Tōhoku Earthquake

Another example can be found in Japan, in early 2011. A magnitude 9.0 earthquake struck off the coast, near to the Fukushima Daiichi nuclear power plant. The plant had been built to survive such an event, with simulations during construction planning used to confirm that the building would not be damaged. However, the simulations for resilience against a tsunami had been conducted using smaller-magnitude earthquakes, and failed to predict the large waves caused by this one. As large waves overtopped the tsunami wall, the power plant began to flood. However, backup generators responsible for cooling the now-shutdown plant had been placed in the basement of the plant. The flooding therefore caused these generators to fail almost immediately, leading to a cascading effect that ultimately resulted in the plant's primary generators melting down.

In this case, later engineers had actually run the linked simulations showing that a large earthquake could result in waves larger than the seawall was designed to handle. However, these linked results failed to convince the officials responsible for preparing the plant for a disaster. It is possible that if the results had been conveyed in an easier-to-understand way, or if the simulation ensemble's data provenance had been shown, the entire situation could have been avoided.

1.3 Motivation

These case studies are hardly an exception; this is the rule in many domains, and especially in disaster response. Significant opportunities exist for novel research to disrupt current approaches, increasing ease, accuracy, and effectiveness of heterogeneous simulation ensembles predictive ability.

In this work, we use a hurricane-based disaster as am example scenario. Disasters pose significant challenges for emergency planning and management as effective disaster response requires matching available resources to shifting demands on a number of fronts. The recent hurricanes in the US highlight the importance of predictive and real-time response and decision making. Effectively managing current and future emergencies through real-time and continuous decision making requires data- and model-driven computer simulations for predicting the evolution of disasters and related hazards. However, data uncertainty, interaction complexity, and resource constraints have thus far proved to be significant roadblocks to widespread adoption of these techniques. Simulation models frequently predict only a few results, without regard for the 1000s of interdependent variables in an emergent disaster area, while specialized domain knowledge requirements complicate the development of integrated simulators. The sheer quantity of simulation results, coupled with real-world time and computational constraints, pose further challenges.

1.4 Research Questions

This work attempts to answer several fundamental research questions which have not been previously addressed in the literature:

(1) Can domain-specific simulation models be linked or combined into an extensible, heterogeneous simulation ensemble?
(2) Can heterogeneous simulation ensembles be deployed onto an adaptively-managed cluster to mitigate the impact of computational and memory complexity of the constituent models?
(3) Can the results of such a heterogeneous simulation ensemble be combined, analyzed, and aggregated into a comprehensive and holistic view of the entire system?
(4) Can this comprehensive, integrated view be visualized in an extensible way, without overwhelming the user with information?
(5) Can the use of this visualization system improve outcomes for planning and response activities, relative to siloed domain-specific results?
(6) Can the above goals be accomplished by a single, cohesive, extensible system that can be easily adapted to different domains, models, and visualizations?

1.5 Research Contributions

This paper describes an initial prototype of the previously-described system. Specifically, this paper introduces a novel extensibility framework for linking domain-specific models in an agnostic way. It then describes an extension of an existing open-source system to implement this framework. This is then expanded to include continuous, adaptive execution on top of a distributed cluster to increase execution flexibility and compatibility. Finally, a novel visualization framework for extensible spatiotemporal data exploration is described and implemented.

We include a sample implementation of the DataStorm framework integrating a series of domain-specific models which could be used for disaster planning and response in Florida. Specifically, we include a hurricane simulator (WRF), a flooding simulation (Itzi), and a human mobility simulator (ONE), which are linked together to provide a holistic perspective on the evolution of a hurricane striking the Florida coastline.

2 BACKGROUND & RELATED WORK

Previously, scientific workflow systems such as Kepler [2] and Taverna [10] have supported control- and data-oriented workflows that process and integrate large amounts of scientific data to support

the scientific enterprise. The focus of these tools is the describe and implement data transformations and other processing needed to support scientific analysis. In general, however, these systems do not consider scenarios requiring continuous execution, nor do they provide multi-instance execution or simulation sampling. The CONFLuEnce system [15] builds upon Kepler to add continuous execution support, but does not consider an extension to generate and execute simulation ensembles.

The use of simulation ensembles to improve predictive accuracy has been examined in the literature [16], but is usually restricted to ensembles within a homogeneous domain. The WIFIRE project [7] couples several heterogeneous models, but does not provide ensemble support or prioritize extensibility to other domains. The use of heterogeneous simulation ensembles for epidemics has previously been explored by [14] and [18]. However, these systems are designed around specific domain simulations; DataStorm attempts to increase the generalizability of this approach to any potential simulation. The analysis of high-dimensional time series data in the context of heterogeneous ensembles has also been explored by [12], with emphasis on tensor-based approaches, but generation of these ensembles is not addressed.

The results of these systems is likely to be classified as “big data"; that is, information which cannot be easily stored on a single system, or easily analyzed without additional processing. Therefore, it is governed by the three V’s of big data: volume, velocity, and variety [20]. Here, volume refers to the actual quantity of the data produced, velocity refers to the speed with which the data is produced, and variety refers to the heterogeneity of the data thus produced. All three of these apply to the generated simulation data, and will be addressed in the contexts where is applies.

Once the data has been generated by the model, it must still be visualized by the user in order to translate predictive improvements into real-world effects. Visualization of high-dimensional time series data is not a novel problem [17], but this work does not leverage contextual information to link multiple data sources into a single corpus.

Other tools, designed to work with spatiotemporal visualizations, have also been created [3, 5]. However, these systems are designed to explore specific data, rather than to adapt to a heterogeneous data corpus that is not predefined. However, more general approaches to visualization are explored, and quite relevant to visualization options for different datasets, including those beyond the scope of this work, such as three-dimensional information.

3 ARCHITECTURE

The core of the DataStorm system is built upon the Kepler scientific workflow system, which is used to manage data provenance and execution control flow. However, fundamental restrictions in Kepler necessitate the creation of an additional, novel control plane built on top of this foundation. We refer to this control plane as the DataStorm Flow Engine, or DataStorm-FE (shown in Fig. 1).

While Kepler provides a flexible framework to create executable scientific workflows, including an actor-oriented modeling paradigm, tools for data transformation and access, and a GUI for the design of scientific workflows, it has significant limitations: (a) Kepler is not designed for ensemble executions; it can only take a fully-instantiated model and execute a specified workflow; (b) Kepler does not provide stateful actors, and is not designed for continuous workflow executions; and (c) while Kepler’s Web and Grid service actors allow scientists to utilize computational resources on the net in a distributed scientific workflow, it does not provide native support for parameter space sampling, distributed instantiation, and parallel execution of simulation instances in an ensemble.

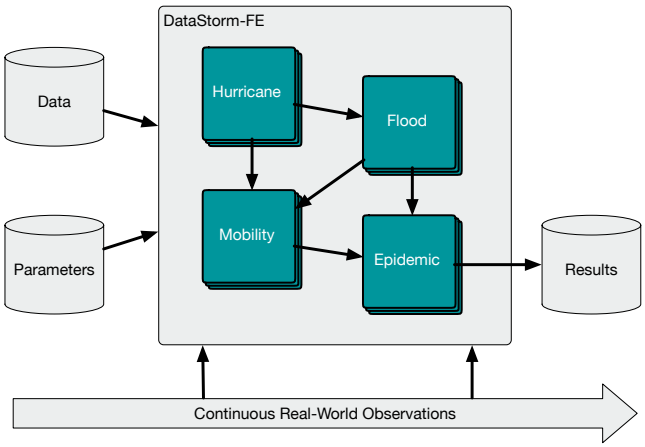


Figure 1: A description of the DataStorm-FE, showing an example of coupled simulations.

3.1 Framework

The data- and decision-flows (i.e., DS-flows) supported by DataStorm-FE are inherently temporal. Therefore, rather than requiring a single-shot execution, where input data for the workflow is consumed and each actor is invoked only once, DS-flows benefit from continuous execution, where input data is consumed continuously and each actor is invoked repeatedly, as up-stream actors produce new results to be consumed and processed.

The core component of the DataStorm-FE system is the DataStorm-Actor or DS-Actor, consisting of several more specialized sub-modules (as seen in Fig. 2) which adapt the domain simulator to interface with the wider system. Each DS-actor is *stateful*, in the sense that, the actor records its own state variables and outputs, and is able to recall its previous state before each execution cycle. As data (represented as multi-variate time series) flows through the system, it is encoded as structured XML, and the provenance of such data is automatically recorded by the system, for later use and analysis. Note that, if real-world conditions change and new observations or inputs are produced, a scenario may need to be re-simulated. In an ensemble, however, any changes will cascade through the system, potentially requiring a re-computation of every downstream simulation. Data provenance also helps identify which subsets of data are affected by a given change, thereby helping reduce redundant work: simulation plans can be created to re-run only the affected data, reducing execution time. Stateful modules may leverage these advantages further, by adjusting parameters mid-execution or aborting redundant executions.

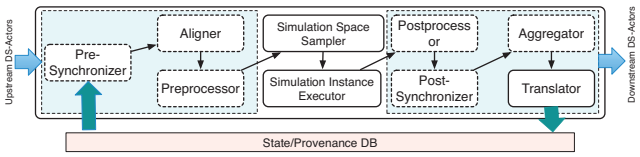


Figure 2: The DS-Actor framework, with internal components.

3.2 Models

The three disaster-related simulation models selected cover hurricanes, flooding, and human mobility. These models were selected for both their relative frequency and their semi-dependent nature. With the Weather Research and Forecasting (WRF) hurricane model [8], we are able to predict the track of a hurricane, as well as associated wind speeds and rainfall. Since this model does not attempt to predict flooding related to these events, we couple the output with Itzï [6], a hydrological simulator which models the flow of floodwaters over the landscape. Additionally, the hurricane and floods have behavioral impacts on the affected population; therefore, the corresponding DS-actors are coupled with the Opportunistic Network Environment simulator [11], which is used to track population movement through transport networks within the affected area.

4 DATA MANAGEMENT

Although the DataStorm-FE controls how and where data and control flows travel, and permits the linkages that empower DataStorm's integrated system, it does not address several fundamental challenges when dealing with these data streams. Specifically, the generation, storage, and analysis of this data is handled by novel systems created to address these challenges, which are deployed in conjunction with DataStorm-FE to enable the desired functionality.

4.1 Generation

For example, one unusual challenge faced by the DataStorm system is that the input data is intrinsically dynamic; although individual models may rely on static inputs in order to execute their domain-specific simulations, the data being provided to the visualization engine is generated by models, rather than observed or gathered from sensors. This contrasts with traditional approaches, in which static data can be generated or gathered, then processed with a one-time cost for cleaning and analysis before use in the system. Since DataStorm data is generated on-the-fly from an exponentially high-dimensional sample space, as individual models may each have tens or hundreds of individual parameters, producing results that flow through the dependency network in complex ways, these approaches are fundamentally incompatible with project goals. This precludes any manual data cleaning or preprocessing prior to visualization or analysis; the velocity and volume of the data is too high to permit such activities.

Instead, we deploy novel simulation space sampling techniques to determine a subset of scenarios to sample, pruning this exponential space to fit within a realistic simulation budget. These techniques sample from the possible simulation space, taking into account previous simulation results, historical observations, and user-selected heuristics to choose input variables that provide the most benefit from the user. This may include the most-likely scenarios, but could also include the most-destructive scenarios, those which can be most easily mitigated, or other possible heuristics. DataStorm provides several sample heuristics, and includes functionality for users to add their own schemes as well.

4.2 Storage

In general, data is passed from module to module using the provenance database when possible, or when the data is too large to store in the database, it is stored on-disk on the systems where it is generated, and pointers to that information are passed in the provenance database instead. These are fetched in a lazy fashion by components which need to access that data, whether that includes the visualization tool or downstream models that are linked to results from upstream.

In cases where data needs to be passed, regardless of location, between two models which do not share a common data format, we use a free-form JSON model based on an extensible template that contains spatial and temporal context, as well as an undefined area for model-specific information that does not conform with predefined keys. In some cases, where the format is shared but the files must still be passed on disk, then the data is passed as raw CSV or other model-specific formats when the consuming model is able to understand those formats.

Since each model in the system may be individually too large for a single desktop or workstation machine to execute, the cluster adapts to the simulation ensemble load by proactively spinning up or shutting down virtual machines within an OpenStack cluster, based on predefined upper bounds configured by the user for each of the systems. This also permits asynchronous and continuous processing, by offloading otherwise-blocking processing to remote systems, which then make use of synchronization flags to notify other participants when results are ready for downstream consumption. These features are achieved through a combination of Kepler triggers, bash scripts, OpenStack API integration, Ansible remote orchestration, and Vagrant virtual machine management.

4.3 Analysis

Despite the high quantity and complexity of the generated data, post-generation analysis must take place in order to convert the data info a usable format. Specifically, this may include tasks such as aggregation, dimensional embedding, or other analysis required to bring the data into a form which is compatible with the visualization subsystem.

However, each model's output may be in a different format as defined by the domain-specific model that generated it. Therefore, each post-processing may need to be model-specific, although generic templates for aggregation can also be provided for adaptation. To resolve this problem, we have implemented a stub location

where the integration team can define a specific module to be executed as a post-processing step for each simulation run. These modules can be written in a variety of languages, such as Java, Python, or C++ as a consequence of the cross-platform extensibility offered by the Kepler architecture, which is used to manage data flow through the system. Additionally, these modules are option, so if the data generated by the model is ready for consumption by the visualization subsystem, then no additional processing is required.

In our sample implementation, this is processed in several ways. For the hurricane data, results are post-processed to extract the eye of the hurricane based on the wind speeds and location, as well as aggregating data at different altitudes into a 2-dimensional spatial map for ease of visualization on the map layer. Additionally, flooding is based on an aggregation of rainfall over time, with a user-defined flooding threshold used to determine the impassibility of a given map cell, which in turn is used to determine whether the road network is partitioned along that edge or not. For the human mobility information, individual human actors are aggregated into groups, to better represent the behavior of larger number of humans traveling in a disaster area, rather than the individually-unpredictable behavior of single human agents.

Note that our system does not attempt to mine, learn, or extract the meaning of the data on behalf of the user directly. Since the framework may be applied to various domains, and therefore requires domain-specific knowledge in order to glean meaning from the interrelated data generated by the simulation ensemble, asking machine models to discern these meanings will produce unacceptably-high error rates due to lack of training information. Therefore, our system focuses primarily on data generation, management, and presentation, rather than analytical processing beyond that required for system function.

5 VISUALIZATION

Without being able to explore the highly-detailed data generated by DataStorm's linked simulation ensembles, users cannot transform these theoretical improvements into practical, real-world actions. Therefore, effectively visualizing this data remains a fundamental goal. Additionally, since the data is intrinsically high-dimensional, containing at minimum a spatial and temporal context, in addition to domain-specific information which may or may not be shared between modules, extra care must be taken to present this data in a way which is both easily accessible to the user yet sufficiently detailed to permit careful exploration of individual subsets.

To this end, DataStorm aims to implement Shneiderman's information-seeking mantra: overview, zoom and filter, details on demand [19]. However, maintaining this goal while also supporting extensibility provides significant additional challenges, both during the initial implementation and domain-specific integration.

5.1 Design Philosophy

Since visualizations may need to display different kinds of data which cannot be predicted prior to deployment and integration, the overall design must remain as modular and extensible as possible. Therefore, we must address several layers of user interaction beyond the visual layer available to the end user. We must also consider how individual models are integrated, how custom data flows can be incorporated into the system, how APIs are exposed to the user, how unique, domain-specific visualizations can be added to the visualization framework, or how existing visualization modules can be customized to user requirements.

Fortunately, we can leverage the presence of a common context to provide a strong foundation for the additional of such models. Specifically, by implementing a shared spatial visualization component (i.e. a map), and pairing that with a temporal slider, any data can be integrated with the system without extensive modification of the underlying foundational layer.

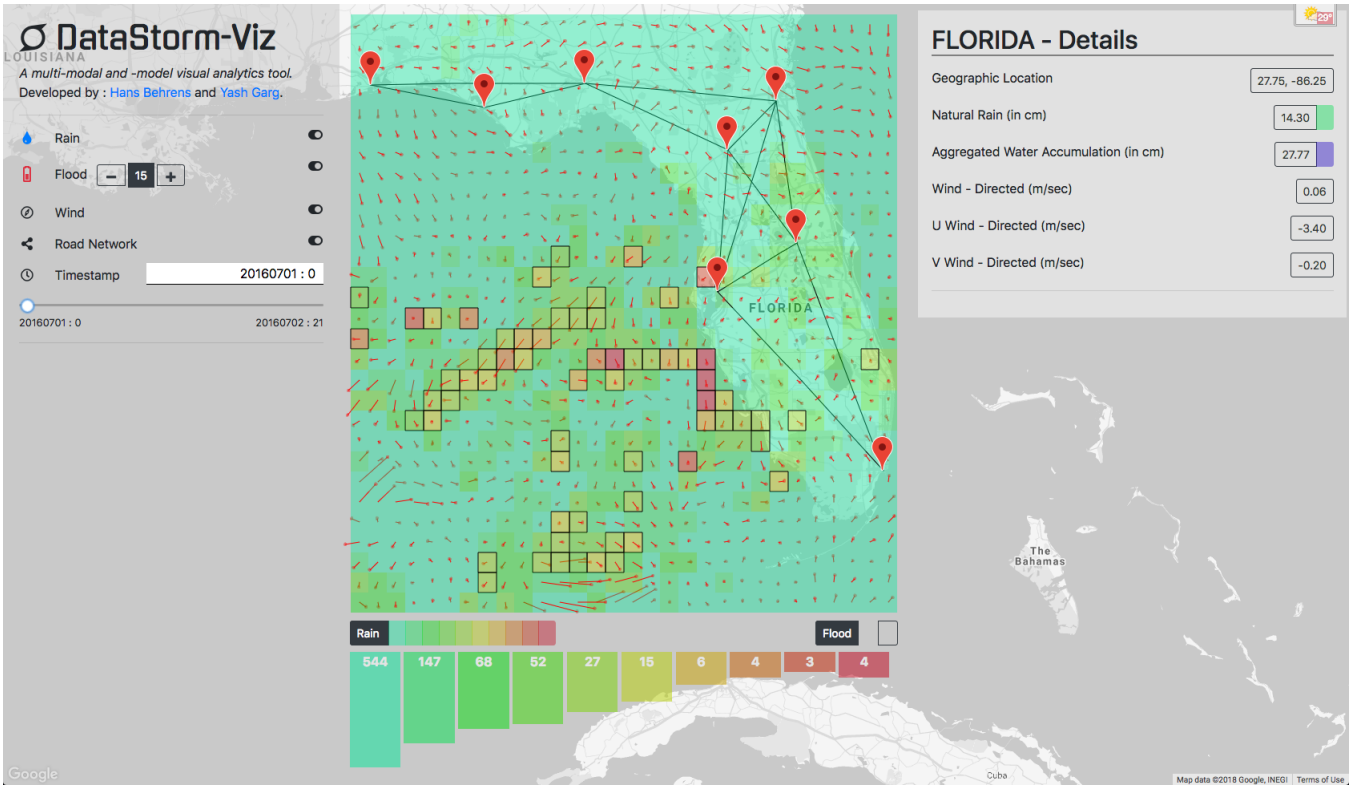
However, since the spatial context is fixed for the simulators, restricting users from leaving the relevant extent is necessary to prevent presentation of confusing areas of no supporting data. Therefore, we intentionally do not permit the user to scroll the map away from the relevant area, which can be more or less restrictive depending on how large an area is being simulated.

Within this context, there may be (potentially very) many different types, or layers, of data to display. Collectively, they may provide an effective overview of the simulation ensemble (seen in Fig. 3), but it is also possible that if too many layers are displayed simultaneously, then the metaphorical "signal to noise ratio" of the screen may become very poor. To address this issue, a maximum cap should be applied to the overview visualization, customizable in both count and specific layers by the system integrator. Since different models will have various data types, and different layers will vary in information density, this permits the end user to customize their "overview" without overly restricting other users with different needs.

Furthermore, these layers should be adjustable by the end user directly, as different end users may have different skill sets and visualization requirements. For example, a weather forecaster may only wish to see a hurricane track to report to the public, while a disaster management official responsible for allocating potable water distribution stations would be interested in flooding, transportation, and human mobility aspects of the disaster area. Therefore, our design incorporates the ability for end-users of the visualization system to enable or disable constituent layers of the system easily, using an on-screen toggle switch. In conjunction with clear, concise layer labels, this empowers users to "zoom and filter" only on the data layers most relevant to them, if they wish to explore the data at a deeper level beyond that offered by the overview. These component views can be seen in Fig. 5d.

However, not every visualization provides sufficient information for a user to make accurate decisions. For example, a heatmap provides a good immediate intuition about the meaning of various levels of rainfall, but it would be unclear at a glance what these colors corresponded to beyond a qualitative estimation ("light", "medium", and "heavy", for example). Instead, a weather reporter may wish to report the quantitative data that underlies the heatmap. How can this data be exposed to the user without affecting the overall visualization? We accomplish this goal by permitting the user to manually select areas of the map to view the quantitative details that contribute to that cell; this corresponds to our goal to provide "details on demand".

Figure 3: A comprehensive overview of the simulated area, with all layers enabled.



5.2 Design Details

5.2.1 Layout. Fundamentally, the goals of DataStorm require domain experts to evaluate and explore data, better informing their decision-making process. Therefore, we chose to put the primary visualization area in the center of the screen, with the largest possible area, in order to

We have designed our visualization with larger screens in mind, as it is more likely to be used in a professional environment, using workstations with paired monitors. It will scale down to smaller sizes, such as laptops, in order to preserve functionality in more environments. However, in such cases, portions of the details pane may be obscured in order to prevent the primary visualization area from being impinged.

Although the map area spans the entire screen, we also included the side panels to provide additional features. Initially, these were fully-opaque, as this provided the best contrast for the text. However, it also contributed to a somewhat cramped feeling in the visualization tool. To ameliorate this issue, a slight transparency was applied to these panels, to allow the underlying map to show through. Since data is not shown on these areas, colorful or "busy" visualizations will not interfere with legibility; however, it also contributes a feeling of open space and spatial context to the user, both desirable properties.

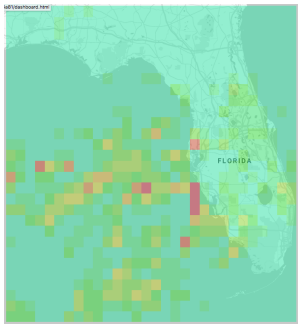
5.2.2 Usability. In our system, the visualization system is designed to be modular, allowing new layers to be added as additional models are integrated, or features are needed for the decision-making process. However, this could potentially lead to a highly-complex, almost cluttered overview when too many layers are visible at a time. To address this issue, we intend to set an upper-limit on the number of visualizations that can be active by default, without constraining the number that can be manually-activated by the user. This balances the initial usability of the system against the overall flexibility of the visualization engine.

Google Maps generally do not constrain the user with regards to spatial navigation; that is, users can zoom in or out, pan, or jump to other areas of the map at any time. However, our simulations model a specific target area, the spatial context that all of our linked models share. Therefore, rather than permit users to navigate to other areas, we constrained their movement to the simulated area, to reduce the likelihood that the user becomes lost or disconnected from the data they're meant to explore.

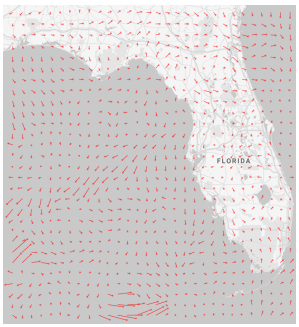
Additionally, different layers in our system may use similar visualization approaches; for example, both water depths and rainfall are represented using heatmaps. To address this issue, we use *context-adaptive rendering*, where layers can have a secondary, fallback representation when their primary visualization is superseded by another layer. The results can be seen in Fig. 7c.

5.2.3 Color. Our system is built upon a monochromatic foundation, with the map, control pane, and details panel grey by default. This was an intentional design choice, meant to allow the users to

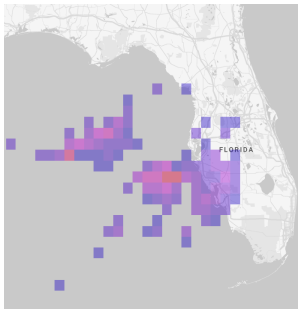
Figure 4: Visualization types, by data layer



(a) Rainfall Heatmap



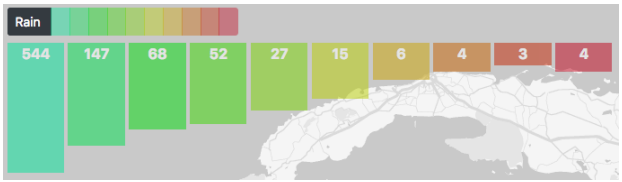
(b) Wind Speed Vectors



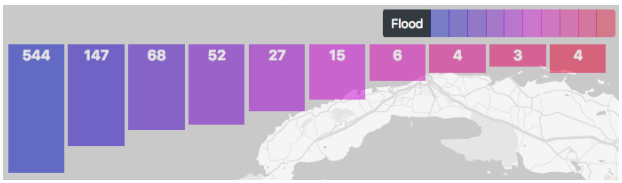
(c) Flood Heatmap

(d) Mobility Graph

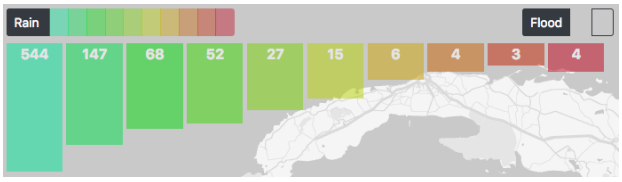
Figure 6: Context-adaptive rendering



(a) Rainfall legend and histogram



(b) Flooding legend and histogram



(c) Adaptive legend, showing threshold cells.

focus on the data itself, and to maximize the potential color palette availabe for representing the data itself.

The heat maps for both rainfall and flooding needed to be considered as well. A green-to-red gradient for rainfall made sense, as this is commonly associated by uses with intensity, which is an appropriate semantic mapping for this layer; low rainfall is light-green or blue, while red rainfall is quite heavy. With respect to flooding, such an association is not as direct. We initially used a solid blue color, representing water, as our depth marker, varying the transparency to represent the relative depths. However, relying on contrast to differentiate quantitatively is not an effective approach, so the alternate color scheme was chosen. We also considered using the same scheme as rainfall, since two heatmaps cannot be displayed simultaneously. However, this led to user confusion as to the data being shown, and so was not used.

5.2.4 Typography. In our motivating example, most results are structured or numeric; that is, they do not need to be visualized in an unstructured or semi-structured form in the same way that text data frequently would be. However, typography still plays a critical role in communicating with the user. To provide a more modern, clean feel, we've chosen to use a sans-serif font family, which provides a cleaner and less-cluttered impression to the user.

5.3 Implementation

In order to provide the widest compatibility and accessibility to our users, and to simplify integration with a simulation cluster that will far exceed the computational power of a single workstation, DataStorm's visualization system is built on a web-based framework designed to be viewed from a user's browser.

This system is primarily implemented in JavaScript, with an Apache Tomcat backend to support libraries and services designed to simplify integration with the rest of the system. This also leverages existing work with visualization modules, such as spatial tools (i.e. maps) without requiring time-consuming redundant coding.

The visualization system runs on a separate virtual machine from the rest of the cluster, and its data has been frozen at a single simulation snapshot to reduce computational overhead. The interface can be accessed directly by navigating to:
`http://129.114.111.124:8080/DVProj/`
The visualization is designed to be viewed at a resolution of $1650x1050$.
All visualization subsystem source code is available at:
`https://github.com/yashgarg1232/DataStormViz`

6 EVALUATION PLAN

Due to the scope of work, any planned evaluation process must span several different sub-components of the system. Specifically, the following aspects should be evaluated:

- The degree of work required for integration of different models into the system.
- The level of performance offered by the system, including (A) computational load and (B) predictive accuracy.
- The robustness of the system to adaptively response to changing user inputs or external data sources.
- The ability of the visualization system to represent the results generated by the integrated models.

- The effectiveness of user planning and response activities when using the system, relative to a baseline.

Although there are several evaluation metrics which should be considered, which effectively increasing the amount of evaluation required, these may also be evaluated in parallel. This decreases the amount of time required for an effective evaluation, although not the costs associated with the same. For example, a software integration team plugging existing models into the extensible model framework may provide feedback related to the APIs, configuration files, and other low-level activities first. Then, the integrated system can begin measuring performance and accuracy characteristics while simultaneously responding to user interactions and simulated or real data continuous data streams. Finally, users can evaluate the visualization systems and effectiveness of planning in parallel, as they work with the system and in the context of their existing process improvement activities.

Additionally, domain-specific baseline models could also be created to illustrate specific instances of that domain's simulation, with the predicted values being measured against the real-world observations. Users could then be presented with the visualization of this scenario, and asked to describe their impression of the event, or to draw similarities between the presented event and real-world events. The frequency with which users correctly identify the underlying real-world event could be used as a proxy to the effectiveness of the system in providing accurate overviews of the data. A similar process could be used to evluate the zoom and filter and details on demand components of the visualization.

However, these rely on the existing of these baseline scenarios. Additionally, a more traditional user study could be undertaken, with user feedback gathered and evaluated using sentiment analysis to determine to what extent the interface was understandable to the users. As a consequence of the extensibility of the visualization components, a large portion of the visual choices are shifted to the integration team; this restricts the ability of the overall project to guide users, although a detailed set of visualization guidelines could be provided if needed.

7 DISCUSSION

7.1 Methodology

In total, our system is able to partially or fully answer our research questions. First, we show that domain-specific models can be linked into a heterogeneous simulation ensemble, through our implementation of a case study based on disaster planning and response. This is achieved using a custom-built integration framework on top of the Kepler scientific workflow system.

Then, this linked ensemble is deployed onto an adaptively-managed cluster, through the use of Ansible, Vagrant, bash, and Python on top of the NSF Chameleon OpenStack system. This provides adaptive scalability which can be ramped up or down to accommodate simulation ensembles of varying size and complexity.

The results of this linked model are then combined as high-dimensional spatio-temporal time series data streams, providing a comprehensive view of the simulated ensemble space based on the complex interrelationships of the individual domain-specific models.

This data is then fed into our custom visualization subsystem, which takes in each individual output stream as a data mode, and represents these outputs in such a way that domain experts are able to explore and filter these results.

User studies to evaluate the efficacy of this visualization system for improving planning and response outcomes is beyond the scope of this work, but is identified as a critical next-step for potential future work.

7.2 Future Work

Although we have implemented a sample scenario on top of this framework, including a comprehensive multi-layered spatiotemporal visualization component, we have not assessed the practical real-world usability of such a system. Future would should include comprehensive user studies for both integration and linkage and visualization template extension, as well as overall system usability.

In particular, the existing visualization templates should be evaluated for usability by target users to confirm that these approaches are understanding in a wide variety of domains. This could be accomplished through the use of user studies, with the identification of target domains and domain experts treated as a priority.

8 CONCLUSION

In this work, we have presented DataStorm, an integrated data generation, simulation, analysis, and visualization engine for heterogeneous simulation ensembles. Although these tools can be used in disaster planning and response to better predict scenario evolution over time, they have been designed to be fundamentally extensible to adapt to various different applications in other domains and disciplines as well.

We have explained our overall system architecture, the data management techniques used to enable our system, and the visual design process used to define our base visualization system. We have detailed our implementation and methodology for achieving these goals, and have also demonstrated a sample scenario built upon this extensible framework, to illustrate a possible use case. This example spanned from the linkage of previously-siloed models into a cohesive and comprehensive modeling system, coupled with our spatiotemporal visualization system to explore the results produced in this way.

Finally, the construction of such a scenario on top of our extensible framework serves to illustrate the possible extension to other domains, although the creation of such is outside the scope of this work. Indeed, we encourage the creation of such simulation ensembles using this framework as a possible exercise to the reader.

REFERENCES

[1] 2008. Committee on Environment and Natural Resources, Grand Challenges for Disaster Reduction, Natl Sci. and Tech. Council, Executive Office of the President. (2008).

[2] Ilkay Altintas, Chad Berkley, Efrat Jaeger, Matthew Jones, Bertram Ludascher, and Steve Mock. 2004. Kepler: an extensible system for design and execution of scientific workflows. In *Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on*. IEEE, 423–424.

[3] Natalia Andrienko, Gennady Andrienko, and Peter Gatalsky. 2003. Exploratory spatio-temporal visualization: an analytical review. *Journal of Visual Languages & Computing* 14, 6 (2003), 503–541.

[4] Duygu Balcan, Bruno Gonçalves, Hao Hu, José J Ramasco, Vittoria Colizza, and Alessandro Vespignani. 2010. Modeling the spatial spread of infectious

diseases: The GLobal Epidemic and Mobility computational model. *Journal of computational science* 1, 3 (2010), 132–145.

[5] Paolo Compieta, Sergio Di Martino, Michela Bertolotto, Filomena Ferrucci, and Tahar Kechadi. 2007. Exploratory spatio-temporal data mining and visualization. *Journal of Visual Languages & Computing* 18, 3 (2007), 255–279.

[6] Laurent Guillaume Courty, Adrián Pedrozo-Acuña, and Paul David Bates. 2017. Itzï (version 17.1): an open-source, distributed GIS model for dynamic flood simulation. *Geoscientific Model Development* 10, 4 (2017), 1835.

[7] C Cowart, J Block, D Crawl, J Graham, A Gupta, M Nguyen, R de Callafon, L Smarr, and I Altintas. 2015. geoKepler Workflow Module for Computationally Scalable and Reproducible Geoprocessing and Modeling. In *AGU Fall Meeting Abstracts*.

[8] Christopher Davis, Wei Wang, Shuyi S Chen, Yongsheng Chen, Kristen Corbosiero, Mark DeMaria, Jimy Dudhia, Greg Holland, Joe Klemp, John Michalakes, et al. 2008. Prediction of landfalling hurricanes with the advanced hurricane WRF model. *Monthly weather review* 136, 6 (2008), 1990–2005.

[9] Daniel Alexander Ford, James H Kaufman, and Iris Eiron. 2006. An extensible spatial and temporal epidemiological modelling system. *International Journal of Health Geographics* 5, 1 (2006), 4.

[10] Duncan Hull, Katy Wolstencroft, Robert Stevens, Carole Goble, Mathew R Pocock, Peter Li, and Tom Oinn. 2006. Taverna: a tool for building and running workflows of services. *Nucleic acids research* 34, suppl_2 (2006), W729–W732.

[11] A Keranen. 2008. Opportunistic network environment simulator. *Special Assignment Report, Helsinki Univ. of Tech.* (2008).

[12] X Li, KS Candan, and ML Sapino. 2018. M2TD: Multi-Task Tensor Decomposition for Sparse Ensemble Simulations. In *ICDE*.

[13] S Liu, Y Garg, KS Candan, ML Sapino, and G Chowell-Puente. 2015. NOTES2: Networks-of-Traces for Epidemic Spread Simulations. In *AAAI*.

[14] Sicong Liu, Silvestro Poccia, K Selçuk Candan, Gerardo Chowell, and Maria Luisa Sapino. 2016. epiDMS: Data management and analytics for decision-making from epidemic spread simulation ensembles. *The Journal of infectious diseases* 214, suppl_4 (2016), S427–S432.

[15] P Neophytou, PK Chrysanthis, and A Labrinidis. 2011. Confluence: Continuous workflow execution engine. In *SIGMOD*.

[16] KB Olsen, SM Day, LA Dalguer, J Mayhew, Y Cui, J Zhu, VM Cruz-Atienza, D Roten, P Maechling, TH Jordan, et al. 2009. ShakeOut-D: Ground motion estimates using an ensemble of large earthquakes on the southern San Andreas fault with spontaneous rupture propagation. *Geophysical Research Letters* 36, 4 (2009).

[17] Roger D Peng. 2008. A method for visualizing multivariate time series data. (2008).

[18] Silvestro Roberto Poccia, Maria Luisa Sapino, Sicong Liu, Xilun Chen, Yash Garg, Shengyu Huang, Jung Hyun Kim, Xinsheng Li, Parth Nagarkar, and K Selcuk Candan. 2017. SIMDMS: Data Management and Analysis to Support Decision Making through Large Simulation Ensembles. In *20th International Conference on Extending Database Technology (EDBT'17)*. OpenProceedings. org, 582–585.

[19] Ben Shneiderman. 2003. The eyes have it: A task by data type taxonomy for information visualizations. In *The Craft of Information Visualization*. Elsevier, 364–371.

[20] Paul Zikopoulos, Chris Eaton, et al. 2011. *Understanding big data: Analytics for enterprise class hadoop and streaming data.* McGraw-Hill Osborne Media.