Sapphire: Experiences in Scientific Data Mining

Chandrika Kamath

Lawrence Livermore National Laboratory, Livermore, CA 94551

E-mail: kamath2@llnl.gov

Abstract.

The size and the complexity of the data from scientific simulations, observations, and experiments are becoming a major impediment to their analysis. To enable scientists to address this problem of data overload and benefit from their improved data collecting abilities, the Sapphire project team has been involved in the research, development, and application of scientific data mining techniques for nearly a decade. In this paper, I first describe the Sapphire system architecture which was motivated by the needs of a diverse set of applications. Then, using examples from different domains, I discuss our experiences in mining science data and some of the challenges we faced in analyzing data ranging from a megabyte to several terabytes.

1. Introduction

Over the last ten years, the Sapphire project team (https://computation.llnl.gov/casc/sapphire) at Lawrence Livermore National Laboratory, has been involved in the research, development, and application of scientific data mining techniques to various problems involving data from observations, experiments, and simulations. This paper summarizes our approach to addressing the diverse needs of applications and describes some of the lessons learned in the analysis of scientific data sets.

At the start of the project, discussions with domain scientists indicated that we needed to support a multitude of requirements. The raw data available for analysis was in the form of images; structured or unstructured mesh data with physical variables at each mesh point; or time-series data collected by different sensors. The type of analysis required was problem dependent, and ranged from identifying and characterizing structures of interest to classifying structures belonging to a certain category or class. Image data was often noisy, with the noise statistics varying with the sensors used to collect the data. There was also interest in building predictive models, where each experiment or simulation was characterized by several input and output variables, and the goal was to predict the output variables for a given set of input variables, or to identify key input variables to predict the values of the output variables.

To support these diverse requirements, we designed and built a software toolkit (Figure 1) [1] with separate modules for different tasks such as de-noising, background subtraction to identify moving objects in video, dimension reduction to identify key characteristics of objects, pattern recognition for clustering, classification, and so on. Recognizing that for each task, different methods were likely to be appropriate based on the data, we used object-oriented techniques to provide a uniform interface to these algorithms. The focus on modularity also carried over to the packaging of the software, with different libraries supporting different tasks, allowing a user to link to only what is necessary for the solution of their problem.

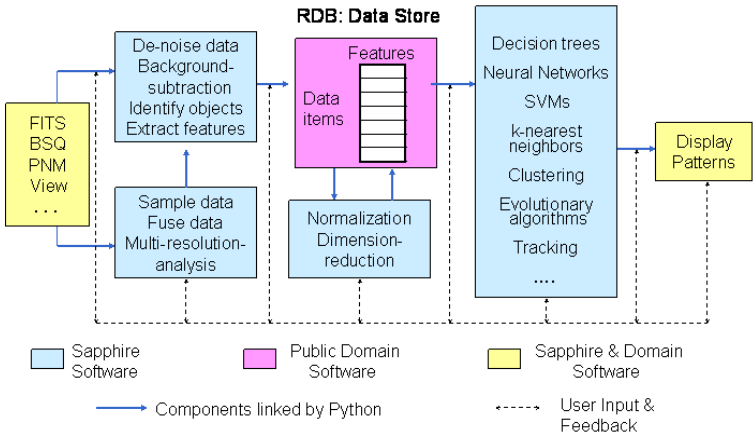


Figure 1. Sapphire system architecture. The compute-intensive parts are highlighted in blue, the domain specific parts in yellow, and the public-domain data store in pink.

Sapphire software incorporates several algorithm advances resulting from our research. It is written in C++ and its modularity, as well as the use of object-oriented design concepts, allow the easy addition of new algorithms as needed. A data set is usually analyzed by linking several modules together, either through a driver code in C++ or via Python. We have applied Sapphire software to a broad range of problems in different domains, including

- Classification of bent-double galaxies in observational astronomy data [2].
- Separating signals, such as El Niño and volcano signals, from surface temperature data obtained from climate simulations [3].
- Similarity-based object retrieval in two- and three-dimensional simulation data [4].
- Detection of human settlements in satellite images [5].
- Identification of key features associated with edge-harmonic oscillations (EHOs) in sensor data from a tokamak [6].
- Detection and tracking of moving objects in video sequences [7, 8].
- Estimating missing features in multi-media information retrieval [9].
- Comparing simulation and experimental data for code validation [10].
- Characterizing and tracking bubbles and spikes in three-dimensional simulations of the Rayleigh Taylor instability [11, 12]
- Classification of orbits in Poincaré plots [13]
- Detection of blobs in experimental images from fusion plasma [14].

Using examples from three problems, one each in observational, experimental, and simulation data, I next describe in more detail some of the issues and challenges in mining scientific data.

2. Classification of bent-double galaxies

Our goal in the analysis of the data from the FIRST survey (http://sundog.stsci.edu) was to develop a classifier to discriminate between bent-double and non-bent double galaxies [2]. Bent-double galaxies are of interest to astronomers as they indicate the presence of clusters of galaxies. The astronomers first identified bent-doubles visually in the images and then cross-validated potential bent-doubles with other astronomical surveys. This was not only subjective, but also became infeasible as the survey grew to nearly a million galaxies. Figure 2 shows

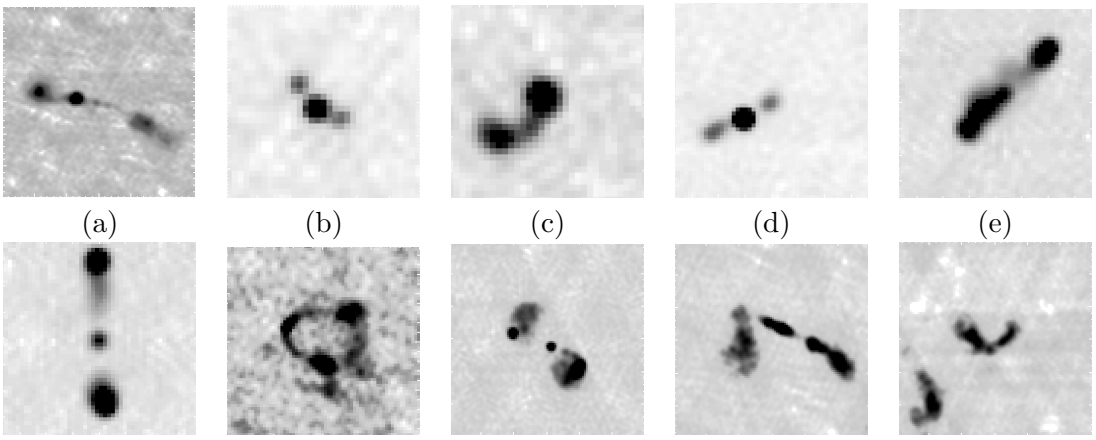


Figure 2. Example radio sources from FIRST (a)-(c) bent-doubles. (d)-(f) non-bent doubles (g)-(j) complex galaxies. Note the similarity between (b) and (d).

some examples of bent-double and non-bent-double galaxies, as well as some galaxies whose morphology is complex enough to be of interest to the astronomers.

Data from the FIRST survey are available in two forms — image maps and a catalog. The catalog is obtained by processing an image map and fitting two-dimensional elliptic Gaussians to each galaxy. Each entry in the catalog describes a single Gaussian and includes features such as the RA and Dec (i.e. coordinates) for the center of the Gaussian, the lengths of the major and minor axes, the peak flux, and the position angle of the major axis (degrees counterclockwise from North). Due to an upper limit on the number of Gaussians used, galaxies with a complex morphology are not approximated well by the catalog. However, for simpler galaxies, the astronomers believed that it made sense to focus on the catalog in our analysis.

Our first task was to identify the catalog entries which formed a single galaxy. Any Gaussians within 0.96 arc minutes of each other were considered to belong to one galaxy. If there were 4 or more such Gaussians, the galaxy was considered complex enough to be of interest and therefore flagged as such. Galaxies with only a single Gaussian were ignored as they were unlikely to be "doubles", bent or otherwise. We then focused on the two- and three-entry galaxies separately as they are described by different features, resulting in feature vectors of different length.

Once the catalog entries forming a galaxy were identified, they were used to extract features representing the galaxy. Our work on the extraction of features is covered in detail elsewhere [2, 15, 16]. In this paper, I focus on the challenges encountered when we use the features to classify the galaxies as bent-doubles or non-bent-doubles. As an example, I consider three-entry galaxies and decision tree classifiers.

We faced two major issues in classification - one was the size and quality of the training data and the other was the accuracy of the results. The training data generated manually by the astronomers was relatively small and unbalanced, with 167 bent-doubles and 28 non-bent doubles. Our goal was to use these 195 galaxies to build a decision tree model (with at most 10% error rate) which could automatically classify 15,000 three-entry galaxies without a class label.

We first used our original set of 195 training examples to refine the set of features until the error rate on 10 runs of 10-fold cross validation dropped below 10%. In the process, we removed features which were not very discriminating and identified more robust techniques to extract the features so that they were insensitive to small changes in the data.

We next used the tree to classify galaxies which had not been assigned a label. Our

Method	Gini		Gain Ratio		Info Gain	
	No Pruning	Pruning	No Pruning	Pruning	No Pruning	Pruning
Single tree	22.79 (0.31)	19.77 (0.18)	22.62 (0.27)	19.83 (0.15)	22.77 (0.39)	19.71 (0.41)
Histogram-based single tree	21.73 (0.34)	20.46 (0.29)	20.85 (0.39)	18.81 (0.17)	22.56 (0.32)	20.96 (0.39)
Histogram-based 10 trees	18.69 (0.28)	18.27 (0.30)	18.10 (0.16)	18.02 (0.22)	18.22 (0.18)	18.42 (0.34)
Sampling-based 10 trees (50%)	18.21 (0.23)	17.31 (0.17)	17.83 (0.20)	17.48 (0.15)	17.79 (0.19)	17.58 (0.18)
Adaboost 10 trees	21.87 (0.42)	20.40 (0.45)	22.37 (0.53)	22.56 (0.47)	20.50 (0.45)	20.75 (0.43)
Bagging 10 trees	19.40 (0.28)	18.35 (0.34)	18.98 (0.34)	18.12 (0.26)	18.98 (0.36)	18.52 (0.35)
ArcX4 10 trees	20.48 (0.39)	20.12 (0.20)	21.67 (0.35)	22.48 (0.41)	21.06 (0.22)	19.77 (0.37)

Table 1. Cross-validation error (std error) using different classification methods, using 485 instances

original plan was to have the astronomers validate these class labels. Though we built a simple interactive tool to make the process easy, we found that it was still rather tedious, subjective, and inconsistent as the labels assigned by an astronomer were subject to the drift common to human labelers. Therefore, we were able to validate only 290 galaxies, of which 92 were bents and 198 non-bents, to compensate for the original unbalanced data.

We evaluated the performance of various tree-based classification algorithms on this new training set of 485 galaxies. These algorithms include the commonly used techniques of Adaboost, bagging, and ArcX4, as well as two methods we proposed, called ASPEN, for Approximate Splitting for Ensembles. One of these methods uses a histogram instead of sorting all the feature values at each node, and the other uses a sample of the instances at each node to make the decision. We consider three splitting criteria - Gini, Gain ratio and Info Gain [17]. Table 1 presents the cross-validation error (and standard error) of 10 runs of 10-fold cross-validation for this data both with and without pruning. We observe that pruning reduces the error rate by avoiding overfitting, which is to be expected. Also, the use of ensemble methods, where more than one tree is created from the same training set using randomization and the results combined using voting, work better. This is again to be expected as decision trees have high variance, which is reduced through the randomization and voting. The best results are obtained with the two ASPEN approaches, followed closely by bagging.

Interestingly, we also observe that the error rate is now closer to 20% though the tree built with the original training data had an error rate of less than 10%. This too is expected as the refinement of features to reduce the error rate was done for the original training set which has a disproportionate number of bent-doubles. It is unlikely that the same set of features would be the best for the enhanced training set. In addition, the 290 galaxies added after validation constituted mainly galaxies which we (non-astronomers) found it difficult to classify. Unfortunately, the astronomers too had problems with these galaxies, either disagreeing on a label or being inconsistent with their labeling when shown the same galaxy on different occasions.

Ideally, we should have again refined the features for the new training set to see if we could reduce its error. However, as we looked more closely at the galaxies which were being misclassified, we found that these were the difficult to classify galaxies, where even the

astronomers disagreed on the class label. This suggested that there was a limit to the accuracy we could expect given the quality and size of the training data. Fortunately, the astronomers were interested in a ranked list of bent-double galaxies as they wanted to schedule telescope time to further study these galaxies. Therefore, we used the different algorithms to rank order the bent-doubles, with a galaxy identified as bent-double by n algorithms being ranked higher than one classified as bent-double by m algorithms, if $m < n$. This allowed the astronomers to prioritize the galaxies which they wanted to investigate further via a telescope.

This classification step in the identification of bent-double galaxies illustrates several challenges. First, the training data may be small, unbalanced, and of poor quality, with mislabeled instances. We must therefore be careful in interpreting the accuracy results. Further, achieving high accuracy may not necessarily be the goal of the scientists; given the subjectiveness and inconsistency in labeling, it is not clear if high accuracy can always be achieved or is desired.

3. Identification of key features for EHOs

Sometimes the goal in a scientific application is not to build a predictive model, but to discover a set of features that may provide insights into the phenomena of interest. In our work in the analysis of sensor data from fusion experiments at the DIII-D tokamak, we were interested in determining which sensor variables were key to the presence of edge-harmonic oscillations (EHOs) in the tokamak. The preferred mode of operation of a tokamak is the high-confinement mode. This comes at a significant cost due to effects of edge localized modes (ELMs) which can cause rapid erosion or even destruction of some components. Recently, a "quiescent H-mode" of operation, without ELMs, has been observed in the DIII-D tokamak. Associated with this mode is a phenomena known as the edge harmonic oscillation (EHO), which can be identified both visually and automatically using simple rules derived from the visual analysis.

Our goal in this analysis was to identify which of the variables being measured by different sensors were relevant to the presence of EHOs. First, with input from the physicists, we extracted the values of 37 candidate variables that describe approximately 700 experiments, each lasting about 6 seconds. Each 50 ms time window of each experiment received a binary label (high/low EHO-ness) using the program that detects EHOs. Next, we preprocessed the data to discard the time windows that contain at least one missing value caused by either an inactive sensor or data from all sensors not being sampled at the same rate. Next, a visual examination of box-plots and histograms of the data revealed many outliers. Using the median value of each variable in each time window eliminated some outliers, but since many still remained, we decided to eliminate the time windows that contained at least one variable in the top or bottom percentile of its range. After the preprocessing, our training set consisted of 41818 instances. Note that unlike the problem of classification of bent-double galaxies, our training data for this problem is not small as it was generated using automated techniques.

The key challenge in this problem was the validation of the results from feature selection algorithms. The scientists did not have any preconceived notion of what features were likely to be important; therefore, we used several approaches to gain confidence in the results. First, as described earlier, we improved the quality of the data so our results were not skewed by the presence of outliers. Second, we tried several different algorithms for feature selection. While we did not expect all techniques to give identical results, we would trust the results more if several techniques selected the same set of features as important. We did not consider one of the most commonly used techniques, namely, principal component analysis (PCA), as it linearly transforms the data into a lower dimensional space, whereas the scientists were interested in a subset of the original 37 features. Third, we introduced a "sentinel" random feature that is uniformly distributed in the interval [0,1]. The rank of this feature was another measure of how much we could trust the results from an algorithm.

We considered the following feature selection techniques:

- The *PCA filter* [18] is a technique derived from the PCA. It first performs a PCA on the data. Then, starting with the eigenvector corresponding to the smallest eigenvalue, it discards the variable with the largest coefficient (in absolute value) in that vector. It then proceeds to the eigenvector corresponding to the next smallest eigenvalue and discards the variable with the largest coefficient, among the variables not discarded earlier. The process is continued until all variables are ranked.
- The *distance filter* uses the Kullback-Leibler (KL) distance between histograms of feature values to estimate how well a feature can separate the data into the two classes, EHOs and non-EHOs. We first discretize the numeric features using $b = \sqrt{|D|}/2$ equally-spaced bins, where $|D|$ is the size of the training data. The histograms are normalized to estimate the probability, $p_j(d = i|n)$, that the j-th feature takes a value in the i-th bin of the histogram, given a class n. For each feature j, we calculate the class separability as

$$\Delta_j = \sum_{m=1}^{c} \sum_{n=1}^{c} \delta_j(m, n),$$

  where c is the number of classes (2, in our case) and $\delta_j(m, n)$ is the KL distance between the distributions corresponding to classes m and n for feature j:

$$\delta_j(m, n) = \sum_{i=1}^{b} p_j(d = i|m) \log \left(\frac{p_j(d = i|m)}{p_j(d = i|n)} \right).$$

  The features are ranked by sorting them in descending order of the distances Δ_j as a good feature will have a large value of the KL distance.
- The *Chi-square filter* ranks features by sorting them in descending order of Chi-square statistics computed from their contingency tables. The contingency tables have one row for every class and the columns correspond to possible values of the feature. Numeric features are represented by histograms, so the columns of the contingency table are the histogram bins. The Chi-square statistic for feature j is

$$\chi_j^2 = \sum_i \frac{(o_i - e_i)^2}{e_i},$$

  where the sum is over all the cells in the contingency table, o_i stands for the observed value, and e_i is the expected frequency of items.
- The *stump filter* is based on the approach used in decision trees to determine the split at each node. The filter only considers the split at the root node of the tree which is based on all the data. Each feature is examined in turn and a split found which minimizes (maximizes) an impurity (purity) measure. We rank the features using the Gini index [17],

$$\sum_b \frac{n_b}{n} \left(1 - \sum_c \left(\frac{n_{bc}}{n_b} \right)^2 \right)$$

  where n is the total number of instances, n_b is the number of instances in branch b, and n_{bc} is the number of instances of class c in branch b.
- The *boosting approach* is an iterative method where in each iteration, the algorithm ranks the features that have not been selected so far and adds the highest-ranking feature to the feature subset. Then, a classifier (naive Bayes, in our case) is trained, and weights assigned to the examples so that misclassified ones have higher weight. These weights are used in the KL distance method to rank the features unselected thus far.

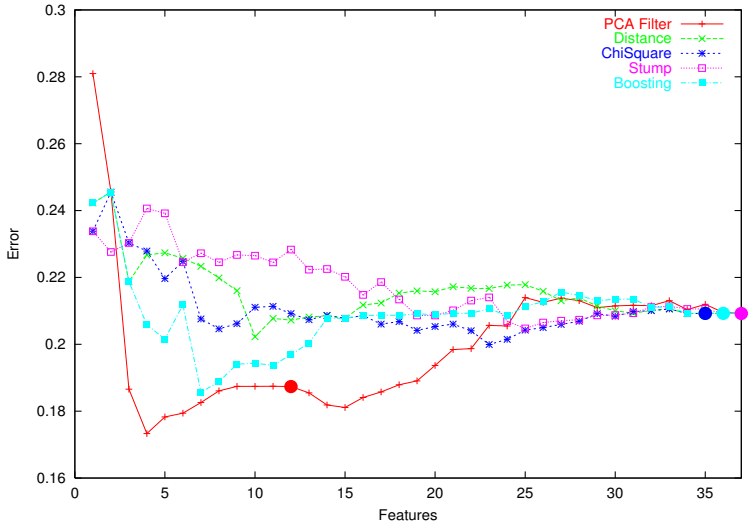


Figure 3. Error rates varying with the number of features for the fusion data. The large dots represent the rankings of the random noise feature.

Figure 3 presents the error rates of a naive Bayes classifier trained on increasingly large feature subsets [6]. The PCA filter produced a compact feature subset that results in the lowest classification error of 17.3% compared to 20.9% obtained with all the features. Note however, that the PCA filter ranks the noise feature (indicated by the red dot) highly. This would lead us to doubt the features selected by this method, especially as the features ranked high by the PCA filter are ranked much lower by other methods. The remaining four methods are remarkably consistent in the features they select as important - six features were ranked in the top ten by four methods and an additional three were ranked in the top ten by three methods.

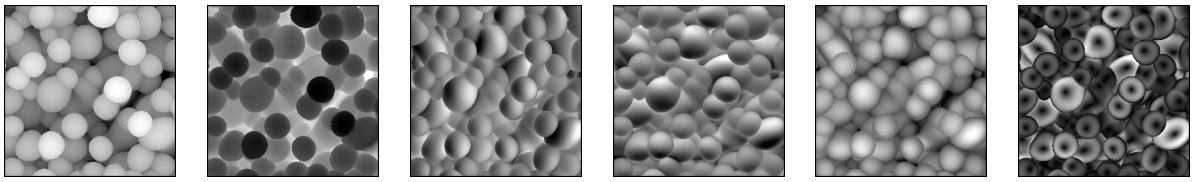
This example illustrates that the quality of the data must be checked before we start the analysis and we must use caution in our interpretation of the results. Further, the use of more than one analysis method is often necessary when the phenomena being analyzed is poorly understood.

4. Counting bubbles and spikes in Rayleigh-Taylor simulations

I next describe a problem where the phenomena being simulated is not only poorly understood, but the data set is so large that one has to be ingenious in solving the problem. Our goal was to identify, count, and track coherent structures known as bubbles and spikes in three-dimensional simulations of the Rayleigh-Taylor instability. This occurs when an initially perturbed interface between a heavier fluid on top of a lighter fluid is allowed to grow under the influence of gravity. The fingers of lighter fluid penetrate the heavier fluid in what are referred to as 'bubbles', while 'spikes' of heavier fluid move into the lighter fluid. With time, these structures, which are initially distinct, continue to evolve; in the process, they may grow, split, merge with surrounding structures, or shrink in size relative to other structures which grow and overtake them.

Our work in using image analysis to count and track the bubbles and spikes is documented elsewhere [11, 12]. Here, I focus on two aspects of the work, namely, the exploitation of domain-specific information and the sensitivity of the results to parameters in the analysis algorithms.

As an example, I will use the larger of two data sets we analyzed, which was obtained using a direct numerical simulation on a three-dimensional computational domain in the form of a cube with uniformly-spaced grid points in the x, y, and z directions, and $\Delta x = \Delta y = \Delta z$. This allowed us to consider the domain as a three-dimensional image, with a grid-point corresponding



(a) (b) (c) (d) (e) (f)

Figure 4. A 300×300 sub-image of the top view of the bubble surface illustrating the (a) height of the surface from original fluid interface; (b) pressure; (c) x-velocity;(d) y-velocity; (e) z-velocity; (f) magnitude of the x-y velocity.

to a pixel. There are 3072 grid points along each dimension; the simulation is run for 248 time steps and five variables are output in single precision at each time step: the pressure, the density, and the x, y, and z velocities at each grid point, resulting in an 80 terabyte data set. Note that unlike an image, the data at each grid point are floating point values, not integers. The simulation was run on a parallel machine using 16384 processors at early time and 65536 processors at late time. It took 17 days to complete, accounting for a total of 2303 single-CPU years. The 3072^3 grid points were partitioned among the processors in vertical columns.

The large size and the distributed nature of the data were not the only challenges faced in the analysis of the data. A key issue was the lack of a precise definition for a bubble, especially one that holds over all time. Our approach was to start with a small subset of the data at every n-th time step, experiment with various algorithms and parameter settings, and then try the best ones on ever larger subsets of the data.

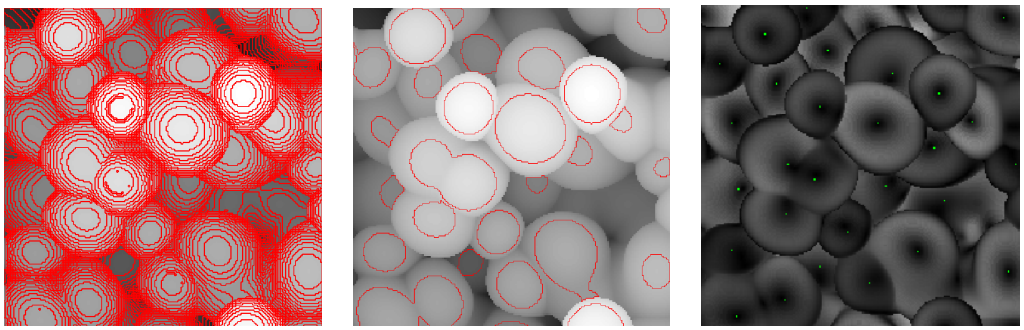
We used a two-phase approach. In the first phase, we first converted the 3-D data into 2-D by using a simple region-growing technique to identify the boundary of the bubbles (spikes) and then considering the image formed by the top (bottom) view of this surface. In the second phase, we counted the bubbles and spikes in these 2-D images. Figure 4 shows a 300×300 subset of the full 3072×3072 images of the top view of various variables at the bubble boundary. These include the height-map, which is the height of a point on the bubble surface relative to the original interface; the pressure; the $x-$, $y-$ and $z-$ velocities; and the magnitude of the x-y velocity. These images are at time step 50 in the simulation, where the initial perturbation has grown to form clearly identifiable bubbles. As the bubbles grow, merge, split, and die off, their shape changes considerably and it is no longer easy to identify the extent of each bubble.

We next applied image segmentation techniques to the data in Figure 4(a) to identify the extent of each bubble, and thus obtain their count. We used a simple region-growing technique which considers neighboring pixels with nearly the same height as forming a single region as shown outlined in red in Figure 5 panel (a). Panel (b) shows the regions after cleanup to remove small regions, merge regions completely within each other, and remove odd shaped regions.

We had also observed that the x-velocity at the bubble surface was negative on the left half and positive on the right; similarly, the y-velocity was negative on the bottom and positive on the top. So, combining the two images could identify the center region of a bubble. Figure 4(e) shows the magnitude of the x-y velocity which is small (indicated by the dark pixels) at both the center of a bubble as well as its perimeter. We could isolate the center pixels as the height near the center did not change as rapidly as near the perimeter (see Figure 5(a)). The centroid of these center pixels formed the tip of a bubble, as highlighted in green in Figure 5(c).

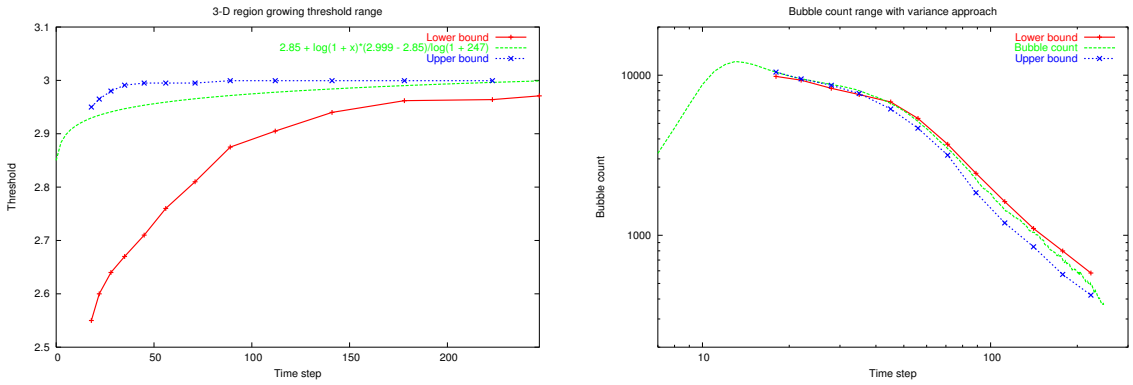
This exploitation of the domain information, led to a much faster algorithm for counting bubble tips, taking only 8 seconds to process a 3072×3072 image in contrast with 2800 seconds for the region-growing approach. As shown in [11], the results from several variants of both approaches are very similar, building our confidence in the results.

For further verification, we considered the sensitivity of the results to the choice of parameters



(a) (b) (c)

Figure 5. The 300×300 height map image from Figure 4 showing the result of segmentation using region growing (a) before and (b) after cleanup. (c) The magnitude of the x-y image with the bubble tip indicated in green.



(a) (b)

Figure 6. (a) The values of the threshold used in phase I of bubble counting - the threshold used in the analysis is shown in green, while the blue and red curves illustrate the range over which we can vary the threshold without a large variation in the bubble counts, as shown in (b).

in our algorithms. In the second phase of the analysis, namely, the identification of bubbles in the two-dimensional images, we could visually check that we were counting almost all the bubbles. However, we also needed to confirm that the results of the first phase were not sensitive to the choice of parameters. Our simple region growing algorithm used a single threshold on the density variable. We varied the value of this threshold and observed how the bubble counts changed when we used a fixed algorithm and set of parameters for phase 2. Since were working with three-dimensional data, we conducted this study at select time steps. Figure 6 shows, in green, the threshold we used in our analysis, and in blue and red, the range over which we could change this threshold without a large change in the bubble counts. These plots show that small changes in the threshold parameter do not result in large changes in the bubble counts.

This problem illustrates the challenges faced in analysis of massive data sets, especially when the structures of interest in the data are poorly defined. It then becomes important to plan the analysis carefully, exploiting domain information where possible, and conducting sensitivity studies to gain confidence in the results.

5. Summary

In this paper, I described the work done by the Sapphire team in the analysis of scientific data. Using three examples, I illustrated some of the challenges in scientific data mining and

our solution approaches. These include improving the quality of data to handle outliers and unbalanced training data; using multiple algorithms and sensitivity analysis to gain confidence in the results; and exploiting domain knowledge where possible to simplify the analysis, improve the quality of the data, and increase the size of the training set.

Acknowledgments

I gratefully acknowledge the contributions of the Sapphire project team to both the implementation of the software and the analysis of data from various applications. The domain scientists graciously shared their data and expertise. The work was partially funded by the DOE Office of Science SciDAC program, the DOE NNSA ASC program, and the LDRD program at Lawrence Livermore National Laboratory.

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

References

[1] Kamath C 2006 *Journal of Physics Conference Series, Volume 46* pp 500–504
[2] Kamath C, Cantu-Paz E, Fodor I and Tang N 2002 *IEEE Computing in Science and Engineering* **4** 52–60
[3] Fodor I and CKamath 2003 *Proceedings, Independent Component Analyses, Wavelets, and Neural Networks, SPIE Proceedings, Volume 5102* (SPIE) pp 25–36
[4] Kamath C, Cantú-Paz E, Cheung S C, Fodor I and Tang N 2005 *Next Generation of Data Mining Applications* ed Kantardzic M and Zurada J (New York, NY: John Wiley) pp 211–232
[5] Kamath C, Sengupta S K, Poland D N and Futterman J A H 2003 *Image Processing: Algorithms and Systems II (Proceedings of SPIE/IS&T Volume 5014)* pp 270–280
[6] Cantú-Paz E, Newsam S and Kamath C 2004 *Proceedings of the SIGKDD International Conference on Knowledge Discovery and Data Mining* pp 788–793
[7] Cheung S C and Kamath C 2005 *Eurasip Journal on applied signal processing* **14** 2330–2340
[8] Kamath C, Gezahegne A, Newsam S and Roberts G M 2005 *Image and Video Communications and Processing, Proceedings of SPIE Volume 5685* pp 442–453
[9] Bagherjeiran A, Love N S and Kamath C 2007 *Proceedings, IEEE International Conference on Image Processing, Volume II* pp 233–236
[10] Kamath C and Miller P L 2007 *IEEE International Conference on Image Processing, Volume III* pp 525–528
[11] Kamath C, Gezahegne A and Miller P L 2006 Analysis of Rayleigh-Taylor instability, Part I: Bubble and spike count Tech. Rep. UCRL-TR-223676 Lawrence Livermore National Laboratory
[12] Gezahegne A and Kamath C 2008 *Proceedings, IEEE International Conference on Image Processing* to appear
[13] Bagherjeiran A and Kamath C 2006 *Proceedings of the SIAM International Conference on Data Mining* pp 574–579
[14] Love N S and Kamath C 2007 *Proceedings, Applications of Digital Image Processing, XXX, SPIE Conference 6696*
[15] Kamath C, Cantú-Paz E, Fodor I and Tang N 2001 *Data Mining for Scientific and Engineering Applications* ed Grossman R, Kamath C, Kegelmeyer W, Kumar V and Namburu R (Boston, MA: Kluwer) pp 95–114
[16] Fodor I K and Kamath C 2001 *Computational Statistics and Data Analysis* **41** 91–122
[17] Breiman L, Friedman J, Olshen R A and Stone C 1984 *Classification and Regression Trees* (Boca Raton, Florida: CRC Press)
[18] Joliffe I T 1973 *Applied Statistics* **22** 21–31