Interpreting and Reconstructing Data from Sensor Networks

Jonathan H. Guinther
Dartmouth College, Hanover, NH USA
Dartmouth Computer Science Technical Report TR2013-735

Abstract

Geophysical phenomena are often three-dimensional, time-variant, and physically large, making them difficult to measure. As wireless sensor nodes become cheaper, smaller, and more powerful, using a sensor swarm as a sampling framework seems to be a viable approach to this problem. However, samples from the network can be sparse and unstructured, creating an incomplete picture. Thus, the question addressed by this project is the following: How can we easily construct and interpret the best model of the underlying reality over some domain, given a possibly sparse and irregular set of samples? We present the design, implementation, and evaluation of the Signal Reconstruction Panel, a graphical program for tackling this problem. Using a method of Support Vector Regression, we demonstrate the program's performance on a variety of data sets including the Collisionless Terrella Experiment (CTX), the Active Magnetosphere and Planetary Electrodynamics Response Experiment (AMPERE), the Poker Flats Incoherent Scatter Radar (PFISR), and GreenCube5, a detailed study of the Ompompanoosuc River flow.

1 Introduction

As the technology comprising wireless sensor nodes becomes cheaper and more powerful, using a fleet of sensor nodes to observe geophysical phenomena is increasingly viable. Often the phenomena of interest are three-dimensional, time-variant, and physically large. Additionally, samples from the network can be sparse and unstructured, but researchers would like to have accurate information about the locations and times between measurements. Thus, the problem is: given the times and places for which there are data, how can the best model of the underlying reality be constructed over some domain?

In this paper, we present the design, implementation, and evaluation of the Signal Reconstruction Panel (SRP), a graphical program for assimilating and visualizing physical data sets. Using a method of Support Vector Regression (SVR), we demonstrate the programs performance on a variety of different data sets including the Collisionless Terrella Experiment (CTX-http://sites.apam.columbia.edu/apam/plasma/ctx.html), Active Magnetosphere and

Planetary Electrodynamics Response Experiment (AMPERE-http://ampere.jhuapl.edu/), the Poker Flats Incoherent Scatter Radar (PFISR-http://www.amisr.com/), and GreenCube5, an Dartmouth undergraduate physics study of river flow [22]. In CTX, 2-D image maps of particle density are produced from density probes in a vacuum chamber in order to study fundamental plasma physics. A number of spatially fixed detectors make observations over a period of time. AMPERE consists of 77 orbiting satellites that measure features of the earth's ionosphere, such as the magnetic field. PFISR measures ionospheric plasma density, temperature, and drift using the scatter of radiation from a ground-based radar array. Finally, GreenCube5 uses a low cost sensor swarm to map river surface velocities. All of these studies are examples of multipoint sensor networks made to observe geophysical phenomena.

These projects share various properties which motivate the utility of a program like the SRP. First, the systems which they study lack rigorous models of their behavior, often because there are too many variables that are outside scientific control or cannot be measured. Secondly, the data generated by these studies can be reduced to a simple, common form: a list of sample points and their associated values. The points provide information to identify and locate the sample–for instance, latitude and longitude in the case of GreenCube5 and a local, polar coordinate system for CTX (r, θ, and time). The values are the observations from the experiment–a vector describing river flow for GreenCube5 and a particle density measurement for CTX. Since many experiments share this form, we have designed a program which attempts to reconstruct the original signal from any phenomenon that has been sampled in this way.

The SRP originated in the GreenCube lab at Dartmouth College, a primarily undergraduate student research group overseen by physics Professor Kristina Lynch. One of the long-term goals of the lab is to send an orbital sensor swarm through the upper atmosphere in order to measure properties of the aurora–the primary subject of Professor Lynch's research. However, the lab was concerned that there may be some difficulty in both assimilating and interpreting the data collected by the swarm, which would not only move through 3-D space and time, but observe a phenomena that also changes in space and time. Thus there seemed to be a need for a program like the SRP in the GreenCube lab.

Finally, the studies listed above often require the data to be assimilated to a regular grid for computational reasons such as calculating the curl or divergence, two ubiquitous differential vector field operators. Certain data visualization techniques also have the same requirement. The SRP can achieve this condition with no extra cost. In the rest of the paper we discuss the development and evaluation of this tool.

2 Related Work

The problem of reconstructing a signal from sparse, irregular samples is a well-researched problem. A few of these methods include sinc interpolation, multivariate interpolation, data assimilation (e.g. weather forecasting models), machine learning methods (artificial neural nets, support vector machines, etc.), and compressed sensing. The basic ideas, successes,

and drawbacks of some these models are briefly summarized below.

Consider a time-continuous, band-limited signal and a set of uniformly spaced samples taken at the Nyquist Rate of twice the highest frequency present in the signal. Under these conditions, the signal can be exactly recovered using the Whittaker-Shannon interpolation formula, also known as *sinc interpolation* [1]. This algorithm has optimal performance, but it comes at a great cost, namely the restriction on uniformly spaced samples which is all but impossible in many physical applications. Also, if the sampling rate is too low, the reconstructed signal will suffer from aliasing problems. Finally, this algorithm only works for a signal which is a function of a single variable [2].

Multivariate interpolation is a family of interpolation techniques which operate on functions of two or more variables. These can then be divided into methods that operate on regularly and irregularly gridded data. In general, methods that operate on irregular grids also operate on regular grids. Also, multivariate interpolation techniques make no guarantees about quality of the reconstructed signal. Describing the details of these methods is outside the scope of this document; however, a few methods are listed here for reference: triangular irregular networks [3], inverse distance weighting [4], kriging [5], and polyharmonic splines [6].

Data assimilation is the process by which observations are incorporated into a computer model of a real system. This process is most commonly used in weather forecasting. Each step in the procedure consists of combining current observations with the numerical weather prediction model to produce the best guess for the current state of the system. This guess is used as the model for the next iteration. The procedure is designed to balance uncertainty in both the data and the model [7]. These types of analyses are custom-designed for the particular signal under study.

There is a substantial amount of literature on using machine learning and artificial neural networks for field approximation [8, 9, 10]. More recently, Macedo and Castro proposed a framework for vector field reconstruction in arbitrary dimension from an unstructured, sparse, and noisy sampling [11]. Their framework works by learning vector fields with support vector machines and matrix-valued radial basis functions (RBF). Moreover, their approach can guarantee that the field is either free of divergence or curl. Support vector regression is the main technique used in the SRP and discussed at length in the Design section.

Finally, the field of compressive sensing shows remarkable promise for solving some signal reconstruction problems [12]. This technique leverages the fact that most signals can be completely represented by a handful of numbers in some basis. For instance, a signal that is composed of a finite set of sinusoids can be completely and exactly constructed knowing only the non-zero amplitudes of the frequency components. Given some constraints on the sampling procedure, compressive sensing has been shown to produce the original signal *exactly* with far less information than is required by sinc interpolation [13]. Although compressive sensing was not used for this project, the procedure is complementary to this work and warrants a longer discussion.

From *Introduction to Compressed Sensing*:

> Compressed sensing differs from classical sampling (Nyquist-rate uniform sam-

ples) in three important respects. First, sampling theory typically considers infinite-length, continuous-time signals. In contrast, compressed sensing is a mathematical theory focused on measuring finite-dimensional vectors in R^n. Second, rather than sampling the signal at specific points in time, compressed sensing systems acquire measurements in the form of inner products between the signal and more general test functions. Thirdly, the two frameworks differ in the manner in which they deal with signal recovery. In the classical framework, signal recovery is achieved through sinc interpolation—a linear process that requires little computation. In compressed sensing, however, signal recovery is typically achieved using highly non-linear methods.[14]

The following example and explanation is taken from the paper *Sparco: A Testing Framework for Sparse Reconstruction* [15]. Consider the linear system:

$$b = Ax + r$$

where the rows of the m-by-n matrix A are test functions used to sample the signal x, r is an unknown m-vector of additive noise, and b is the set of observed measurements. The goal is to find an appropriate x that is a solution to the system. It is well known how to find the solution that has the smallest l^2 norm provided A is full rank. However, in compressed sensing m is typically less than n and the aim is to find a solution that is sparse so that x has few nonzero elements. In the case where a signal f admits a sparse representation with respect to a basis B ($f = Bx$) and f is sampled with the m-by-n measurement matrix M ($b = Mf$), then A becomes MB. This is rewritten as:

$$b = MBx + r$$

A concrete example might be where x is a series of Fourier coefficients which are mostly zero. Then f is a sinusoidal function that is sparse in the Fourier basis. The classical sampling scheme can be represented in this notation when b is a set of time-domain measurements and the rows of M consist of zeros and a single one. We say the *measurement basis* is the canonical spike basis.

The central problem of compressed sensing is written as the following discrete optimization problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & ||x||_0 \\ \text{subject to} \quad & ||Ax - b||_2 \leq \sigma \end{aligned}$$

where the function $||x||_0$ counts the number of nonzero components of the vector x and the parameter $\sigma \geq 0$ prescribes the desired fit in the set of equations $Ax \cong b$. Unfortunately this problem is combinatorial and NP hard. However it can be replaced with the convex optimization problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & ||x||_1 \\ \text{subject to} \quad & ||Ax - b||_2 \leq \sigma \end{aligned}$$

Methods for solving this problem include interior-point algorithms, gradient projection, and iterative soft thresholding (end of example from *Sparco* [15]).

Compressed sensing is most successful when the matrices M and B are incoherent—meaning that the largest absolute inner product between the columns of the two matrices is small. For this project, M will always be constructed with the spike basis since sensor nodes can only make measurements at singular points in space and time. Thus compressed sensing techniques will only achieve success when the signal is sparse in a basis which is incoherent to the spike basis, such as the Fourier basis. Because many physical signals will not be sparse in the Fourier basis (*i.e.* sinusoidal), the compressed sensing framework proved to be an infeasible choice for this project. In addition, extending compressed sensing for continuous time/space signals is significantly more complicated than the discrete version discussed here. Nevertheless, in certain applications—such as Magnetic Resonance Imaging (MRI)—where samples can be taken as linear combinations of the entire signal, compressed sensing demonstrates incredible results [18].

After reviewing the various techniques discussed here, we settled on Support Vector Regression for use in the tool. This method will be discussed in detail in the Design section.

3 Problem Specification

There are two core inputs to the Signal Reconstruction Panel: (1) the samples and (2) the points which may be unknown and need to be reconstructed. For lack of a better term, the second input will be called "new points" for the duration of the paper. As stated in the Introduction, the samples can be decomposed into points and values. To motivate this separation, consider a sensor swarm equipped with multiple kinds of sensors. The points are the same for each sensor, only the values are different. Similarly, for a set of values from some sensor, one might choose different coordinate systems for a certain analysis (ignore the z coordinate, for instance). Because we have restricted ourselves to signals that occur in the physical world, a number of limitations are imposed on the points and values.

The points can have a dimensionality of at most four–corresponding to the three spatial dimensions and time. The dimension of the values is restricted to one through three and less than or equal to the point dimension. A value dimension greater than one corresponds to a vector quantity, such as wind direction or magnetic field. Otherwise, the values are scalar measurements, such as particle density. The "new points" input follows similar conventions to the points input.

The outputs of the SRP are the predicted values for the new points and a visualization of these values which is interpreted by the user. Finally, there are additional inputs which control how the values are predicted and visualized.

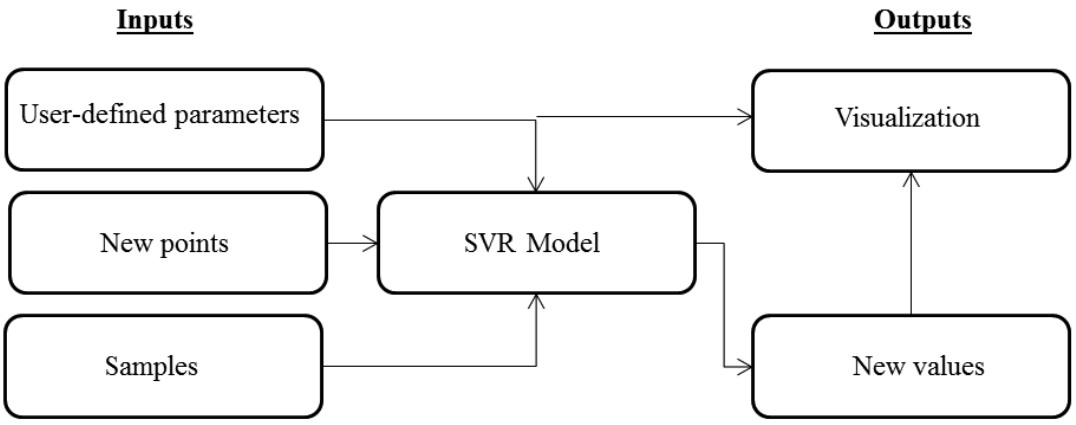


Figure 1: Flow diagram of the Signal Reconstruction Panel

4 Design

The central design choice of this problem is the algorithm by which the samples are used to estimate the values of the new points. The goal of this project was not to develop a new method, but to select the most appropriate one and incorporate it into a useful tool for data interpretation from sensor networks. In the context of this project, the appeal of a certain algorithm can be judged by the following properties: flexibility, customizability, and speed.

Flexibility refers to the algorithm's ability to handle a variety of dimensions in both points and values. The algorithm should be able to effectively process one to four dimensions, including time. By customizability, we mean that the algorithm must produce satisfactory results in the wide variety of data sets it might encounter, from a local experiment like CTX to a global study such as AMPERE. How the results are judged is discussed in the Evaluation section. Finally, the intended user of this software is a researcher who may not have access to significant computing power. Thus the analysis should complete in a timely manner when running on the average laptop computer.

The machine learning method of Support Vector Regression (SVR) was an excellent choice for its performance in these categories. First we will give a brief overview of the algorithm so the reader is aware of key concepts and parameters, then comment on its effectiveness in this project. The following description is paraphrased from *A Tutorial on Support Vector Regression* [19].

Suppose we are given a set of samples $\{(x_1, y_1), \cdots, (x_l, y_l)\} x \in \mathbf{R}^n, y \in \mathbf{R}$. For instance x might be latitude and longitude ($n = 2$) and y might be sea level. In ε-SVR, the basic idea is to find a function $f(x)$ that has at most ε deviation from the observed values y_i for

all the sample points, and at the same time is as flat as possible.

We might consider modeling this data with a linear function f , taking the form

$$f(x) = \langle w, x \rangle + b \text{ with } w \in \mathbf{R}^n, b \in \mathbf{R} \tag{1}$$

In equation (1) *flatness* means a small w which can be ensured by minimizing the norm, $||w||^2$. This can be written as a convex optimization problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \frac{1}{2}||w||^2 \\ \text{subject to} \quad & y_i - \langle w, x_i \rangle - b \leq \varepsilon, \\ & \langle w, x_i \rangle + b - y_i \leq \varepsilon. \end{aligned} \tag{2}$$

It is possible that there is no function f that approximates all pairs (x_i, y_i) with ε precision. In other words, the optimization problem may be infeasible. Slack variables ξ_i, ξ_i^* can be introduced to deal with potentially infeasible constraints. The parameter C in equation (3) controls the how closely the model fits the data.

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \\ \text{subject to} \quad & y_i - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i, \\ & \langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi_i^*, \\ & \xi_i, \xi_i^* \geq 0 \end{aligned} \tag{3}$$

Next, using the duality principle of mathematical optimization theory, a Lagrangian dual problem can be formulated from (3).

$$\begin{aligned} \text{maximize} \quad & -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle x_i, x_j \rangle - \varepsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) \\ \text{subject to} \quad & \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0, \\ & \alpha_i, \alpha_i^* \in [0, C]. \end{aligned} \tag{4}$$

$$w = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)x_i \tag{5}$$

$$f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\langle x_i, x \rangle + b \tag{6}$$

The key point here is that w can be completely described as a linear combination of the sample points x_i. From this it can be shown that for all samples inside the ε-tube (points

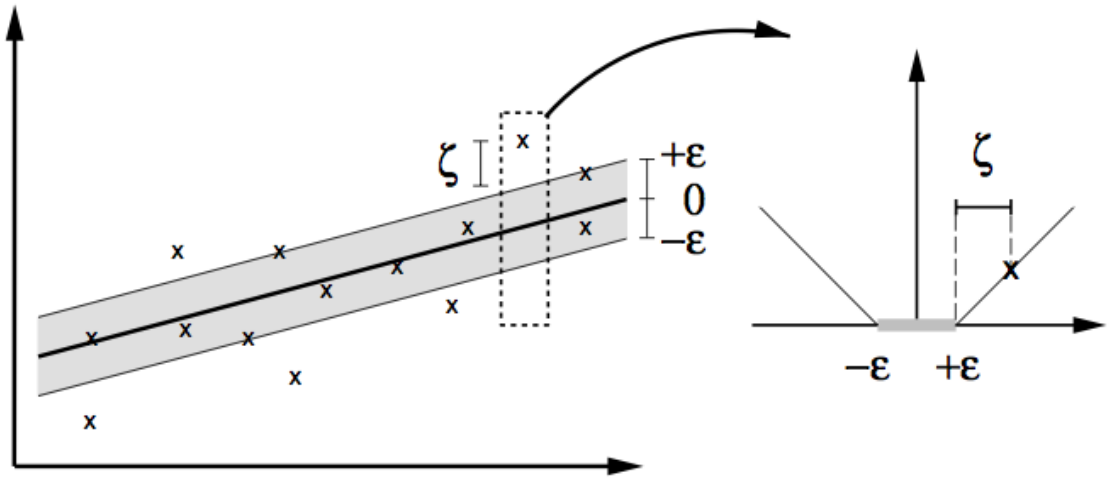


Figure 2: Linear SVR with slack variables. The 'X's are the samples and the bold line is the regression line. The side-diagram shows that a sample can lie outside the ε-tube and still be feasible. (Figure reproduced from: Smola 2004) [19]

within the gray area of Figure 2), the α_i, α_i^* vanish. The samples which have non vanishing coefficients are called *Support Vectors*.

The final step in this brief development of SVR is to employ the *kernel trick*. The essence of this trick is to map the observations into a higher dimensional space so that a linear analysis in this *feature space* produces a non-linear analysis in the *input space*.

Consider the map $\Phi : \mathbf{R}^2 \to \mathbf{R}^3$ with $\Phi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$. In equations (4) and (6), we replace all dot products $\langle x, x' \rangle$ with $k(x, x')$, where $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$. By performing a linear SVR on these preprocessed features, a quadratic function is produced (end of paraphrased example from *A Tutorial on Support Vector Regression* [19]).

Certain aspects of the algorithm make it appealing for our problem. One is the kernel function k, which can significantly alter the algorithm's output and be chosen based on the likely underlying truth in the observed phenomenon. Common kernels are the radial basis function (RBF) kernel and the polynomial kernel. In this project, the RBF kernel was used for several reasons. This kernel can be made to approximate other kernels with appropriate parameter choices. Also there are fewer hyperparameters (*i.e.* parameters of the kernel, not the SVR algorithm) compared to the polynomial kernel [20].

The SVR algorithm is designed to model a data set of any dimensionality, so modeling the four possible dimensions of the physical world poses no problem. By controlling hyper-parameters, one can alter properties that affect the size and shape of features in the model, such as the full-width, half-maximum of the RBF. These aspects of SVR make it superior to most other routines in the context of this problem. The details of SVR can be found here [19].

Using SVR, the control flow of the Signal Reconstruction Panel is as follows. First the user supplies the samples, new points, and parameters to the program. Next SVR is performed

on the samples using the specified parameters to create a model of the original signal. Then the model is queried with the new points, returning the values to the user. In addition, the program creates a visualization of the results that the user can alter to display more useful information. At this point, the user may decide that this SVR model is insufficient, revise their inputs, and begin the process again. Due to the highly visual nature of this project and the revisionary pattern of use, a graphical user interface was implemented. A diagram of this flow is presented in Figure 1.

5 Implementation

In this section we discuss key implementation details of the Signal Reconstruction Panel. Please refer to Figure 3 when GUI components are explained. The project was developed in the MATLAB programming language which was chosen for its powerful, built-in visualization routines. Also, a MATLAB tool for GUIs made designing the SRP layout relatively simple. Three buttons allow the user to specify files for the sample points, sample values, and new points. Each file is in MATLAB's ".mat" format and must follow certain conventions laid out in the README. When the user presses the "Update Display" button, the SVR algorithm runs and a visualization is shown upon completion. The type of visualization is determined automatically by sample data. For instance, if the data are two-dimensional, the visualization will be a surface plot, etc. If the data is four-dimensional, the resulting visualization will be a three-dimensional animation. However, if the user selects the "Animate" option for data of two or three dimensions, the last dimension is assumed to be time and an animation is produced. Again, this convention is specified in the project's README along with more detailed descriptions of the GUI elements.

The SVR algorithm is implemented using the LIBSVM library and represents the bulk of the third-party code used in this project. This library has options for choosing different kernels and parameter values. In the SRP, the radial basis function kernel is used and most of the parameters are held constant. The main control a user has over the model is by altering the four "Feature Size" values in the GUI, one for each dimension. Feature size is not an official parameter of SVR or radial basis functions, however it roughly corresponds to the full-width, half-maximum of the RBF. In LIBSVM the RBF kernel takes the following form:

$$e^{-\gamma|u-v|^2} \tag{7}$$

γ can be written in terms of the full-width, half-max (FWHM) as follows:

$$\gamma = \frac{4 \log 2}{FWHM^2} \tag{8}$$

Thus γ is calculated by replacing FWHM with the largest feature size. The data for the remaining dimensions are scaled according this value and their relative feature sizes. In this way, the FWHMs in each dimension of the gaussian can be individually controlled.

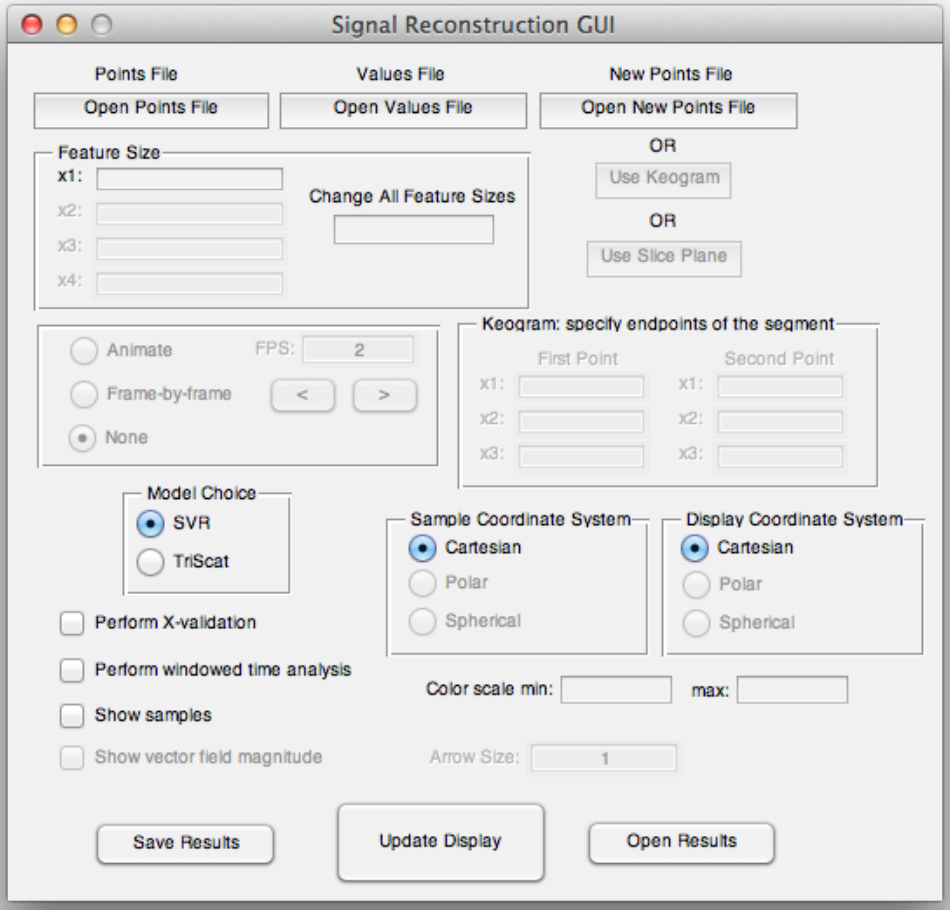
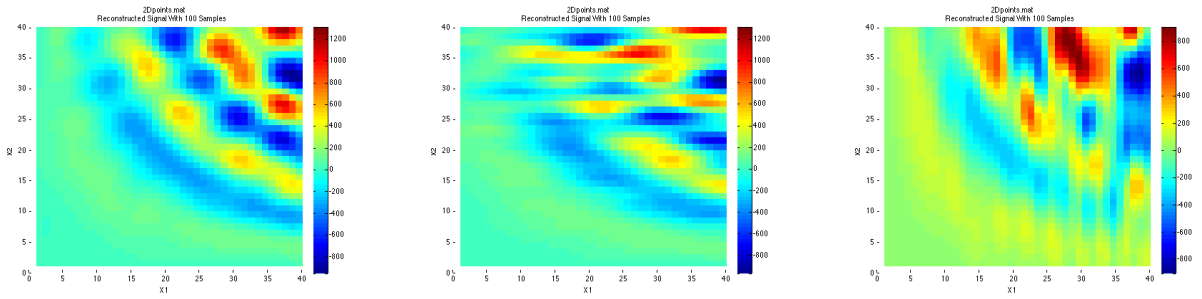


Figure 3: Screenshot of the Signal Reconstruction Panel GUI. Please refer to this figure when GUI elements are discussed.



(a) Equal feature sizes. (b) Large x, small y. (c) Small x, large y.

Figure 4: This figure shows the effect of various feature sizes on the model output.

There are two other features which produce large changes in the results of the Signal Reconstruction Panel. By changing the "Model Choice" from *SVR* to *TriScat*, an entirely new algorithm is used. The algorithm, called TriScatteredInterp (TriScat), is a built-in MATLAB function which can perform interpolation on irregularly scattered data in two or three dimensions. The basic idea is that it will "connect-the-dots"; more precisely, for a set of (x, y) points and values v, it returns a surface of the form $v = F(x, y)$. This surface can then be queried for points to produced interpolated values. In TriScat, the observations are guaranteed to lie on the surface, leaving almost no room for control over the model. This algorithm is a common choice for solving the style of problems discussed in this paper and is a good standard by which to test the SVR algorithm used in the SRP. Its performance is discussed further in the evaluation section.

The second feature which affects the SRP's model is the "Windowed Analysis" option. This option is designed for experiments which collect a large amount data over a long period of time, but whose features have a short time-scale. In this case, performing a single regression over the entire length of the data set is time-consuming and unnecessary, especially if times outside the feature size window should not affect the interpolated value at a given point in time. Thus, when this option is selected, a regression is performed for every time using only samples within a window of one time feature size. This style of analysis performs best when the time sampling is regular and dense.

Finally there are two visualization options whose functions are not obvious: "Use Keogram" and "Use Slice Plane", both of which are alternatives to a "new points" file and ways to reduce the dimensionality of a time-variant visualization. In other words, instead of constructing a list of points whose values need to be interpolated, the user specifies a keogram or slice plane and the program uses the points from these structures as the new points.

A keogram is a 2-D plot where the x-axis is time and the y-axis is distance along the keogram line. The user specifies some line in the space of the data and these points are given as queries to the model for all times. A keogram is appropriate for two or three dimensional, time-variant signals. An example keogram is shown in Figure 5.

A slice plane is a more intuitive visualization for 3-D and 4-D signals. The user specifies a single plane in the space of the data whose points are used as queries in the model for all

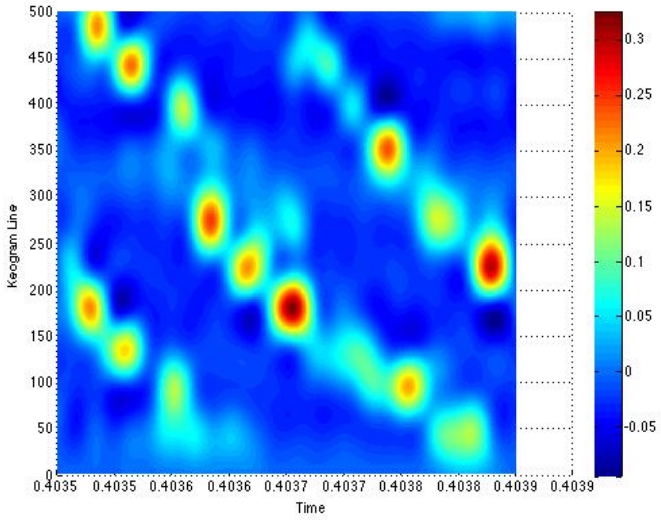


Figure 5: Keogram example from the CTX experiment. This diagram shows that collections of high-valued (red) data move along the specified line. CTX data used with permission.

times. This plane is displayed in the resulting visualization as seen in Figure 6.

6 Example Workflow

In this section we will work through a case study with a set of CTX data and use a variety of features of the SRP. We begin by listing a few key properties of the CTX data. In CTX, spatially fixed detectors measure particle density over a period of time. The detectors all lie in a single plane and are organized in a "donut" fashion. A given trial in CTX may last for half a second, yet there is interesting, discernible structure at a time scale of 0.00001 seconds. Thus the Windowed Analysis is a good option for this experiment. The data are given in polar coordinates as a result of the donut structure and r ranges from 25cm to 72cm.

First, the input files are selected and the feature sizes are determined. We might choose feature sizes of 0.5 in the θ dimension, 5 in the r dimension, and 0.000025 in the time dimension. For the new points, we can create a regular grid in r, θ, and time. Next we must specify the coordinate system of the samples, accomplished by simply selecting the appropriate choice in the "Sample Coordinate System" panel ("Polar" in this case). The "Display Coordinate System" decides the axis labels in the visualization; choosing "Cartesian" will use x and y and illustrate the donut sampling pattern. We should choose the "Animate" option since time is a variable in this experiment. Since there will be many frames, we should increase the frames per second (FPS). A frame from the resulting animation is shown in Figure 7.

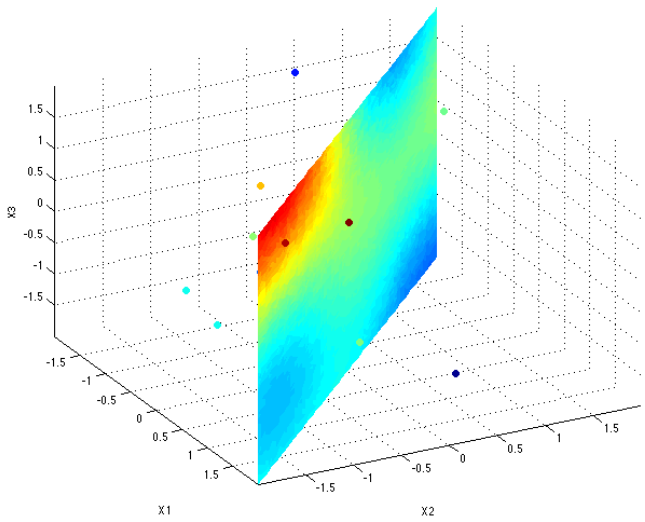
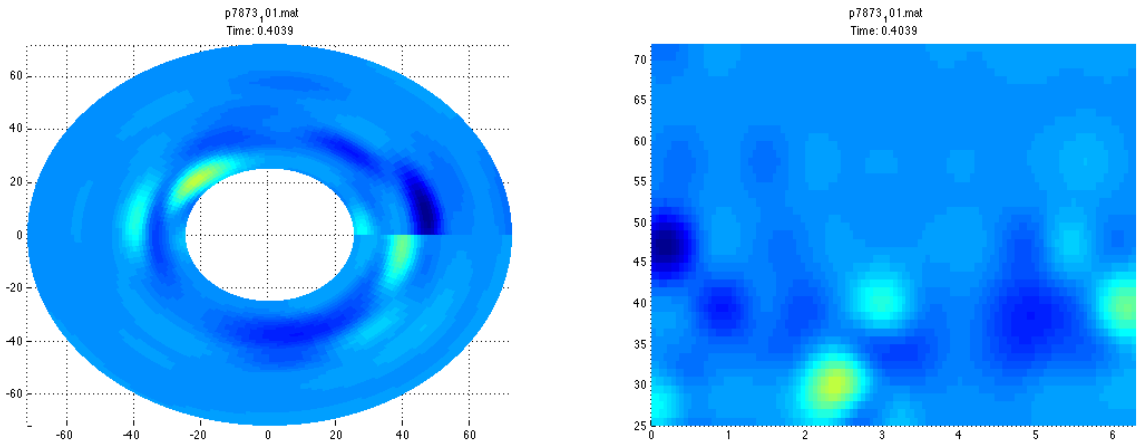


Figure 6: Slice plane example. The dots are the samples.



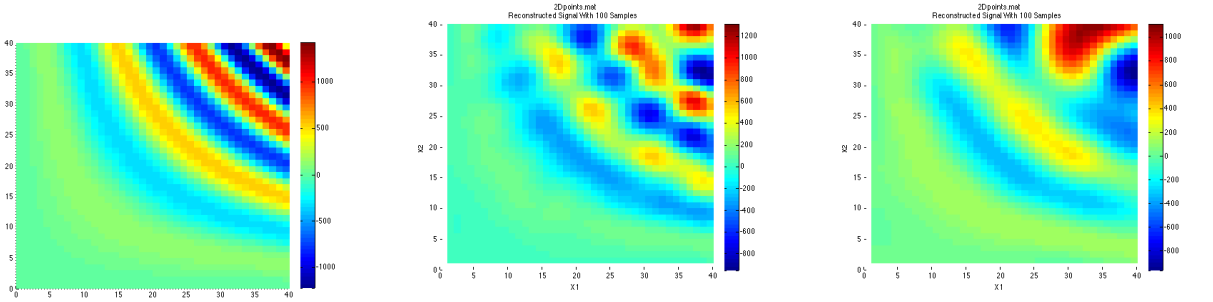
(a) The frame in cartesian coordinates.

(b) The frame in polar coordinates.

Figure 7: A frame from a CTX animation. CTX data used with permission.

7 Evaluation

There are a number of standards by which one can evaluate the performance of the SRP. First is the overall usefulness of the application to its audience, primarily the GreenCube lab and Professor Lynch. Although this standard cannot be quantified, Professor Lynch has been an active user of the SRP throughout its development. She has successfully produced quality analyses of CTX and PFISR data and has commented on the program's user-friendly environment. [21] Now we will describe other standards for gauging performance including cross-validation and comparison with MATLAB's TriScatteredInterp. In addition, we show results from the SRP which either match or improve on those obtained by the original researchers of GreenCube5, CTX, and PFISR.



(a) Original Signal.

(b) SVR with good feature sizes. MSE = 0.048.

(c) SVR with bad feature sizes. MSE = 0.096.

Figure 8: This figure shows how MSE values from cross-validation can be used to assess model performance.

A standard to evaluate the internal performance of the SRP is the cross-validation option in the GUI: "Perform X-validation". Cross-validation trains the model on a subset of the samples and then tests its accuracy on the remaining samples to yield a mean squared error (MSE) value. This option is excellent for testing the accuracy of different feature size choices. An example is shown in Figure 8.

One can also compare MATLAB's TriScatteredInterp algorithm with Support Vector Regression. The mean squared error between 1600 points of the original function in Figure 8(a) and a reconstructed function from TriScatteredInterp shown in Figure 9 was 132,590. The mean squared error for the analysis shown in Figure 8(b) was 63,138. In this example, SVR was more than twice as accurate as TriScatteredInterp. Similar results were observed for different combinations of signals and sampling patterns.

In GreenCube5, GPS payloads floated down the Ompompanoosuc River, measuring the surface velocity. Figures 10, 11, and 12 show the reconstructed velocity of a small portion of the river using both SVR and TriScatteredInterp. It is difficult to judge which is more accurate since the truth is unknown. With TriScat, the river appears more continuous with relatively high velocities shown up to the river boundaries. This seems unrealistic as the velocities should fall to zero as they approach the river bank. Using SVR, the velocities near

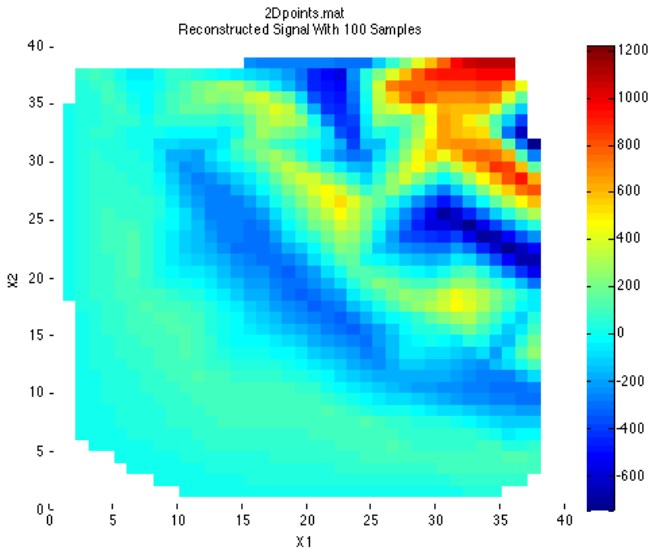


Figure 9: Results from the TriScatteredInterp algorithm on the same data set shown in Figure 8.

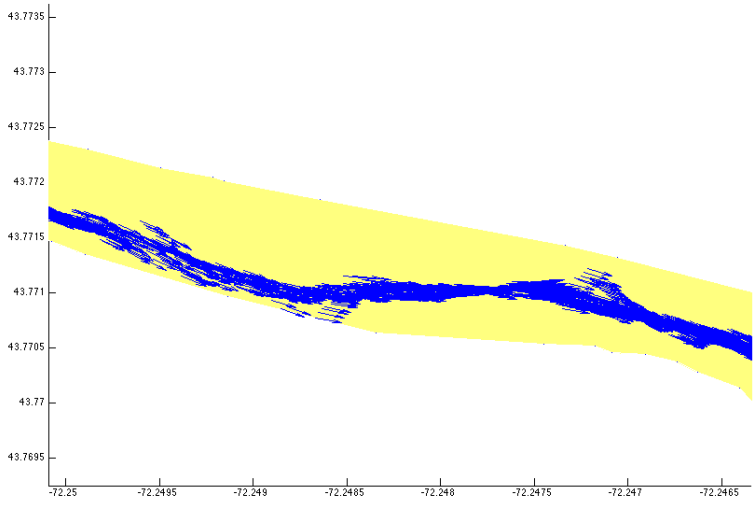


Figure 10: These are the samples taken in a portion of the Ompompanoosuc.

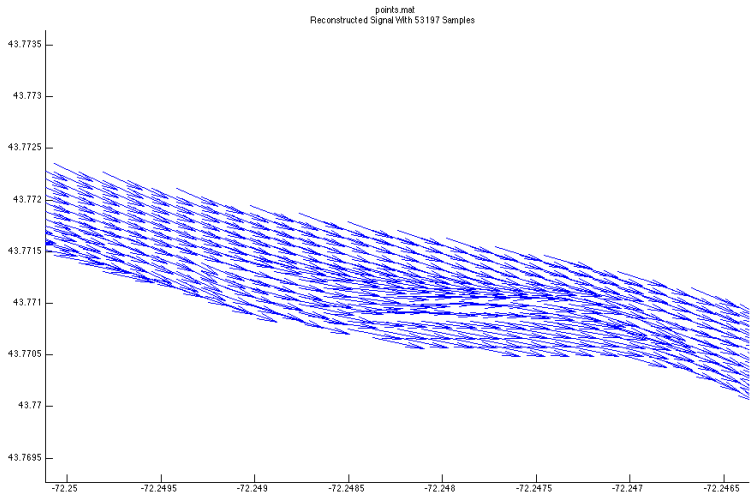


Figure 11: Reconstructed river flow using TriScat.

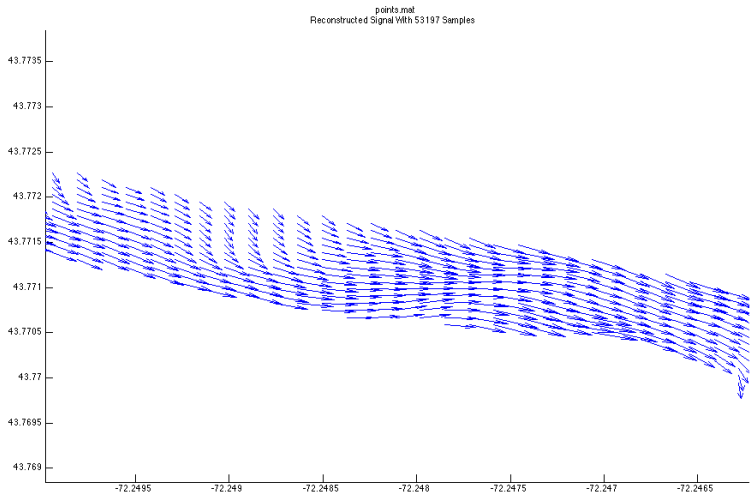
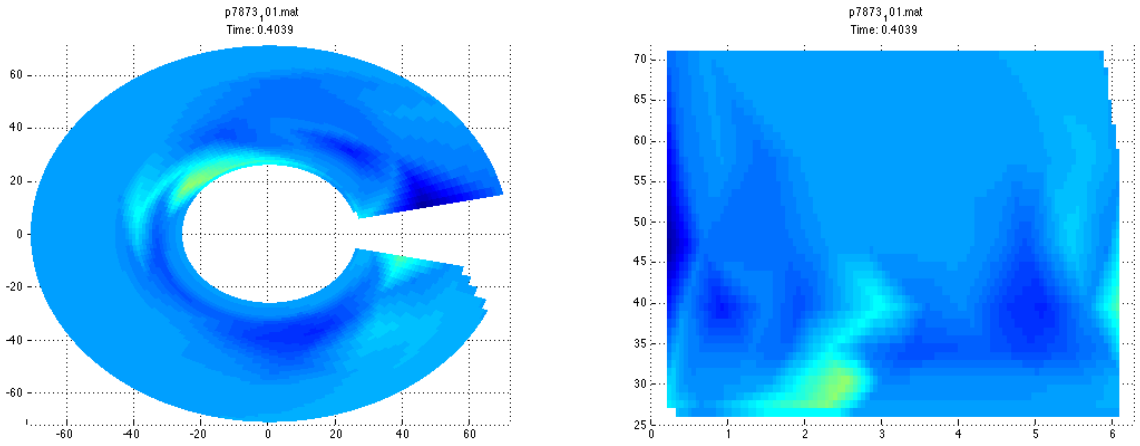


Figure 12: Reconstructed river using SVR.



(a) The frame in cartesian coordinates.

(b) The frame in polar coordinates.

Figure 13: The same frame show in Figure 7 using the researcher's original technique (a simple interpolation equivalent to TriScat). Note how the features seem less defined and more angular. These are artifacts of the interpolation technique, not found in the data. CTX data used with permission.

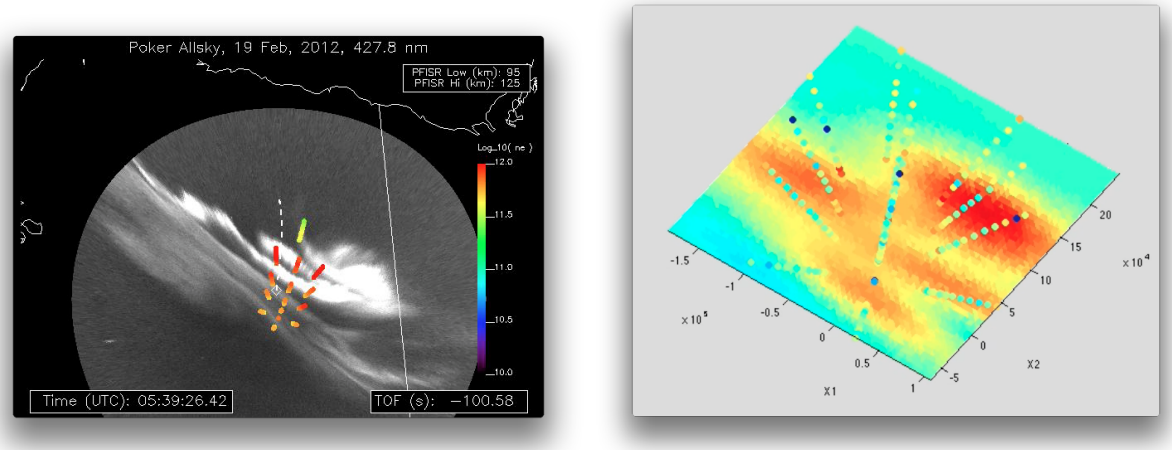
the river bank are closer to zero. Also the reconstruction appears more complex, displaying a wider range of velocities.

8 Conclusion

In this paper, we presented the design, implementation, and evaluation of the Signal Reconstruction Panel (SRP): a graphical program to assimilate and visualize data for researchers in the physical sciences. After surveying a wide variety of algorithms including compressed sensing—a method employed with great success in other fields—the machine learning technique of support vector regression was used as the core reconstruction technique. Additionally, the SRP demonstrated excellent performance on both constructed and actual data sets.

This project will be inherited by the students of the GreenCube lab at Dartmouth College. Although the SRP is a working finished product, there are a number of useful extensions that could be implemented in the future. First, the visualizations of vector data are difficult to intuitively grasp, especially in three dimensions. Work could be done to make these diagrams cleaner and more customizable. Also the number of parameters in the SVR analysis that a user can directly control is relatively small. Although Professor Lynch–the primary tester during the SRP's development–never expressed the need for more direct control, it might arise during some future project.

Perhaps the biggest extension to this project would be the inclusion of custom Lagrangian constraints and boundary conditions, tasks which were researched but never implemented



(a) This image is a view of the aurora from the ground overlaid with the PFISR data.

(b) This figure is a reconstruction of the PFISR data at roughly the same time. The overall structure is similar to the aurora.

Figure 14: In PFISR, ground-based RADAR arrays measure ionospheric plasma density, which is a good proxy for auroral activity. PFISR data used with permission. UAF/GI image (left), D. L. Hampton, used by permission.

due to time constraints. Because SVR is formulated with Lagrangian multipliers, one can imagine the ability to specify custom constraints on the model. A particular case of this was shown in [11] where the authors reconstructed vector fields which were guaranteed to be either curl-free or divergence-free. This extension would require both a framework for specifying constraints and extensive modification to the LIBSVM library. Similarly, one might wish to specify boundary conditions on a certain model. A prime example of this is that river flow should be zero outside the river boundaries.

Lastly, leaving a maintainable library of code is an important goal of this project. A compendium of documentation on the Signal Reconstruction Panel will be compiled and placed on the GreenCube lab's wiki page. This will include examples, the project README, and implementation details. In addition, the source code will be placed in a repository so GreenCube students will have easy access in the future.

9 Acknowledgement

I would like to thank my family and friends for all that they have done to get me to this point. Next I would like to thank Professor Kristina Lynch for mentoring and guiding me, not only on this project but throughout my time at Dartmouth. I would also like to thank my advisor Professor Balkcom letting me tackle this project and giving me advice along the way.

Thanks to the following people for providing me with various data sets: Brian Anderson

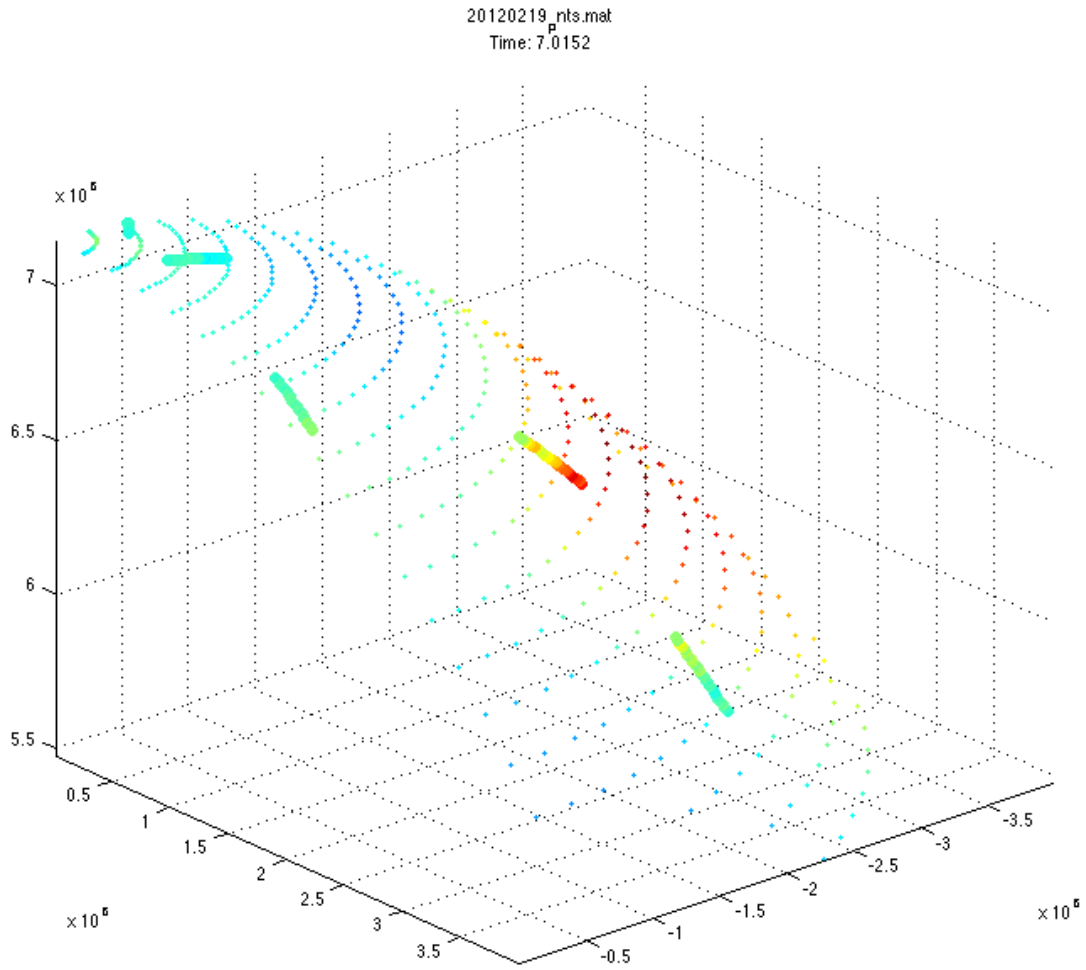


Figure 15: This figure shows an analysis of AMPERE data. A quarter of the upper hemisphere of the earth is shown here. The large strings of dots are five minute windows of data from the orbiting satellites. A direct comparison to researcher data was not completed due to time constraints. AMPERE data used with permission.

Haje Korth at JHU APL for the AMPERE data; Matt Worstell and Michael Mauel at Columbia for the CTX data; and Mike Nicholls and Steven Chen for the PFISR data.

References

[1] H. Nyquist, "Certain topics in telegraph transmission theory", Trans. AIEE, vol. 47, pp. 617644, Apr. 1928 Reprint as classic paper in: Proc. IEEE, Vol. 90, No. 2, Feb 2002.

[2] E. T. Whittaker, "On the Functions Which are Represented by the Expansions of the Interpolation Theory", Proc. Royal Soc. Edinburgh, Sec. A, vol.35, pp. 181194, 1915

[3] "Research Triangular Irregular Network. http://www.ecse.rpi.edu/Homepages/wrf/pmwiki/Research/

[4] D. Shepard (1968). "A two-dimensional interpolation function for irregularly-spaced data". Proceedings of the 1968 ACM National Conference. pp. 517524.

[5] D.G. Krige, A statistical approach to some mine valuations and allied problems at the Witwatersrand, Master's thesis of the University of Witwatersrand, 1951

[6] R.L. Harder and R.N. Desmarais: Interpolation using surface splines. Journal of Aircraft, 1972, Issue 2.

[7] R. Daley, Atmospheric data analysis, Cambridge University Press, 1991.

[8] C. A. Micchelli and M. Pontil. On learning vector-valued functions. Neural Computation, 17(1):177204, 2005.

[9] F. A. Mussa-Ivaldi. From basis functions to basis fields: vector field approximation from sparse data. Biological Cybernetics, 67(6):479 489, 1992.

[10] M. Brudnak. Vector-valued support vector regression. International Joint Conference on Neural Networks, 2006. IJCNN06., pages 15621569, 2006.

[11] I. Macedo and R. Castro. "Learning divergence-free and curl-free vector fields with matrix-valued kernels. 2010.

[12] E. J. Candes. "Compressive sampling". In Proceedings of the International Congress of Mathematicians, 2006.

[13] E. J. Candes, J., T. Tao, and J. Romberg. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inform. Theory, 52(2):489509, 2006.

[14] M.A. Davenport, M.F. Duarte, Y.C. Eldar, and G. Kutyniok, "Introduction to Compressed Sensing," in Compressed Sensing: Theory and Applications, Cambridge University Press, 2012.

[15] E. van den Berg, et al. "Sparco: A Testing Framework for Sparse Reconstruction. University of British Columbia. October 2007.

[16] E. Candes and M. Wakin. "An Introduction to Compressive Sampling. IEEE Signal Processing Magazine. March 2008.

[17] "APL-Led Team Wins NSF Grant to Develop New Observatory for Earth's Space Environment. http://www.jhuapl.edu/newscenter/pressreleases/2008/080721.asp

[18] Lustig, Michael, et al. "Compressed sensing MRI." Signal Processing Magazine, IEEE 25.2 (2008): 72-82.

[19] Smola, Alex J., and Bernhard Schlkopf. "A tutorial on support vector regression." *Statistics and computing* 14.3 (2004): 199-222.

[20] Hsu, Chih-Wei, Chih-Chung Chang, and Chih-Jen Lin. "A practical guide to support vector classification." (2003).

[21] Personal interview with Professor Lynch. April-May 2013.

[22] A. Slagle. "Vector Field Mapping and Analysis Using Finite Sensor Swarms." Dartmouth College. May 2012.

[23] J. Guinther. "GreenCube 5: Mapping River Flow with Sensor Swarms." Dartmouth College. May 2013.