A System for Visual Analysis of Radio Signal Data

Tarik Crnovrsanin, Student Member, IEEE, Chris Muelder, and Kwan-Liu Ma, Fellow, IEEE

Abstract—Analysis of radio transmissions is vital for military defense as it provides valuable information about enemy communication and infrastructure. One challenge to the data analysis task is that there are far too many signals for analysts to go through by hand. Even typical signal meta data (such as frequency band, duration, and geographic location) can be overwhelming. In this paper, we present a system for exploring and analyzing such radio signal meta-data. Our system incorporates several visual representations for signal data, designed for readability and ease of comparison, as well as novel algorithms for extracting and classifying consistent signal patterns. We demonstrate the effectiveness of our system using data collected from real missions with an airborne sensor platform.

Index Terms—Intelligence Analysis, Coordinated and Multiple Views, Time-varying data, Geographic/Geospatial Visualization

1 INTRODUCTION

The radio frequency spectrum is complex and dense, with thousands of events occurring simultaneously every second in a typical suburban environment. These events can include both authorized and unauthorized frequency usage from the Federal Communications Commission (FCC) perspective, potentially criminal activity from a legal perspective, or even naturally occurring noise phenomena. Differentiation between signals of interest (SOI) and non-signals of interest (NSOI) is important not only for domestic radio frequency use management, but also for military intelligence gathering. For military applications in particular, rapid analysis of such data is vital. However, this classification task is resource intensive because of the wide variety of signaling systems that are both in use now and expected to become available in the future. The wide variance in signals has led to the development of Signal Intelligence (SIGINT) sensors which can capture the information from these signals in real-time.

The growing use of SIGINT sensors in today’s military enterprise dramatically increases the amount of data flowing into Intelligence, Surveillance, and Reconnaissance (ISR) data processing centers. Modern SIGINT sensors ingest nearly all signals in the environment simultaneously and thus produce vast amounts of signal data at an incredible rate. Traditional tools found in ISR processing centers can easily overwhelm an operator who is inundated by the volume of incoming data.

There are also constraints on the amount of processing that can be done ahead of time. In many situations, signal information is best collected from the air with specialized antennae as a high vantage point reduces line-of-sight blockage and eases moving the signal collection platform to the required location. However, the airborne collection of Radio Frequency (RF) signals puts severe constraints on the size, weight and power of the equipment, and thus the capabilities that can be installed. Dedicated streaming hardware is used to continuously sample measurements of the external characteristics of transmissions, including frequency, signal-to-noise ratio (SNR), bandwidth, up time of the signal, off-time of the signal, and, in a multi-antenna collection system, the direction from which the transmission originates. However, these per-signal measurements are instantaneous, and signals need to be comprised of many such samples. So the additional hardware is often dedicated to compiling these samples into meaningful transmissions. This still leaves the operators with having to sift through innumerable signals in order to find particular higher order patterns (such as communications) that they might be interested in.

Analytics can be applied to pull out specific features, but there are numerous analytic methods that could be applied, and the potential for future development of any number of situational analytics. Visual analytics approaches would provide a framework to manage such analytics, allowing analysts to focus on the more important high level tasks. We have developed a system that provides a visual workflow to manage a suite of such analytics by providing summarizations of potentially interesting signal traffic patterns, while still exposing the underlying analytics to extract specific patterns that they might be interested, and enabling the development or application of situationally specific analytic processes. In this manner, our system aims to aid in deriving solid intelligence in near real-time, providing the ability for operators to quickly identify combatants and potential opportunists while discounting allies in time-critical situations.

Due to the scale of the data and the complex relationships between transmissions, understanding signal data is a nontrivial task. To our knowledge, the visualization community has not substantially explored this type of data. We have developed an interactive visual analysis system to support this task by working closely with expert analysts to obtain constant feedback and to guide the design and development of our system. When talking to analysts, we found that they were particularly interested in answering the following questions:

- Can we identify communications in the pattern of the signals?
- Is it possible to discover and locate signal repeaters?
- How can we guide analysts to signals of interest?
- What general discovery can we make about the data?

In this paper, we present a system designed to explore and analyze radio signal data. This system consists of a combination of several visualizations and algorithms that help analysts answer important questions. The contributions of our work are:

- A new system for visual representations of radio signal data.
- An interface to manage the workflow of radio signal analytics.
- A novel algorithm for finding repetitive (digital) signal patterns.
- Demonstrations of identifying repeaters, communications, and repetitive patterns.

Most importantly, we have created an effective visual analytics solution to a very important application.
Our work draws from a variety of existing research, including wavelet visualization, communication visualization, geospatial visualization, parallel computation system analysis, and the coefficient of variation.

Wavelet analysis Wavelet analysis has a wide range of uses in computer science [10]. In this section, we focus on its application to signal processing [22], which is directly applicable to our work. Miller et al. [19] applied wavelet transformations to custom digital signals constructed from words within a document. The resulting wavelets are used to analyze the characteristics of the narrative flow in the frequency domain, such as theme changes. Faith et al. [9] applies wavelets to optical wireless signals and then runs PCA on the resulting wavelets to create a scatterplot visualization. Barford et al. [3] used wavelets for anomaly detection. The pseudo-spline filter can expose distinct characteristics of each class of anomalies: outages, flash crowds, attacks and measurement failures. Mueller et al. [20] applies wavelet scalograms to network scan patterns. The resulting wavelets are used to create a graph in which a node is a scan and an edge exists between two nodes if they are highly similar. Our wavelet use is inspired by these works, but differentiated by the discrete nature of the signal data to which our system is tailored to.

Communication detection There is extensive research on the duration of gaps, pauses and overlaps in conversations [8, 13, 12] which focuses on person to person communication. Walkie talkie conversations do not share the same characteristics as person to person conversations, but the research provides some good general rules that are applicable to our work. While some existing works discuss analysis of the contents of communication records [4, 5, 7], the data we were working with only consisted of the signals’ metadata. However, in situations where the signal contents are available, our system could work in concert with such approaches as our metadata analysis by extracting conversations that such content-based techniques could be applied to.

Geospatial visualization Visualization of geospatial data (or ‘Geo-visualization’) has been an ongoing research topic for many years [17], which has produced many geospatial visualizations and analytics [1, 2]. Some approaches focus on the particulars of radio signal data. Han et al. [11] presented a visual analytics system for the development of signal fingerprinting-based localization systems. Though their approach works with radio signals, it depends on the intricacies of precise, indoor signals, and would not scale up to the size or uncertainty of our data. Wood et al. [27] apply graph/matrix based techniques to the analysis of pairs of discrete origin and destination locations, such as a communication network could form. However, our data is too noisy for such a discrete technique. Rather, we borrow from techniques for geospatial that can handle uncertainty such as splatting and heat map techniques, as in the works of MacEachren or Thomson [18, 25].

Parallel computing systems Our signal data visualization problem shares surprising similarities with parallel computing visualization. Each processor, like a frequency band, has a start time and duration for each task, similar to a signal. For instance, Gantt charts [26] look very similar to frequency versus time plots (which can be seen on the bottom of Figure 3). Many scalable performance visualizations [29, 23, 21] use techniques that show system resource utilization, the timings and durations of parallel events, or the application executions. These use the aforementioned Gantt charts and other applicable visualizations such as histograms and Kiviat diagrams. Spear et al. [24] and Landge et al. [16] focused on node-link diagrams or matrices to display the communication topology. Communication between processes is important as it has direct impact on performance in a parallel environment. Many toolkits combine both types of visualizations.

Coefficient of variation The coefficient of variation is used in probability theory and statistics as a way to show the variability of the mean in relation to the standard deviation. Xu et al. [28] proposed a new similarity metric called variation coefficient similarity based on an extension of the Dice and cosine similarity measures. They demonstrated the effectiveness of their metric by comparing it to three other prompt similarity metrics. Though the metric shares the same name as coefficient of variation, it is not the same. Their metric works on a set of vectors and relies on an alpha value.
Once the data is loaded, we also apply another preprocessing step, in order to help reduce noise in the data and pre-compute simple metrics. For instance, signals have a minimum frequency difference so they do not interfere with each other. When a signal is measured, it can have slight variation in frequency but will be within the legal band for that signal. So we use this step to bin frequency bands that are approximately identical. Additionally, the system sorts the data by frequency bands and time. We also compute additional metrics that are useful in later, more complicated analytics, such as the proximity of the plane to the signal. All these calculation and the data are also converted and stored for faster future loads.

**Analytics** Our system currently has five implemented analytic methods; four of them designed around specific tasks, plus one for more general exploration and discovery. The repeater algorithm finds signal repeaters by looking for collections of signals that have the same start and duration times. The communication detection algorithm is a rule based approach that tries to find a set of signals that make up a communication pattern based on time between signals and their locations. The windowed variance analytic algorithm finds series of signals that have high variance in temporal duration by applying a coefficient of variation metric. The digital pattern distinction algorithm also uses the coefficient of variation across a sequence of events, in order to separate series of signals that make up a digital patterns that exhibit consistently low variance from the remainder of analog signals. Lastly, the wavelet transformation is for more general analysis instead of a specific task; it samples the data from time windows defined by user parameters, then projects the higher dimensional results of the wavelets to a two dimensional representation using PCA. These analytics were developed to be modular, so that the user can dynamically link up the analytics as desired, or easily implement new analytics.

**Visualizations** When creating the visual system we had two goals in mind: keep the visualization intuitive for our expert users, and allow the users to gain insights for making crucial decisions. The visual representations must be kept simple because of the sheer volume of data and associated analysis tasks. Many classification tasks are difficult to compute automatically with certainty, such as determining whether signals are part of a communication, or if they are digital or analog transmissions. Rather than making these decisions completely computationally, we use analytics to compute probabilities that a series of events are one or more of these types of signals. We then plot these candidates and allow the user to inspect and group them as appropriate.

As Figure 3 shows, our system has three different views: the main, map and timeline. The main view shows the results of the algorithm methods and can toggle between different outputs and their views. The map view provides geospatial information and a reference point for the results. The timeline allows the user to filter the data based on attributes in the data and also serves as the overview for the data. The signal inset is triggered when an aggregate data point is clicked on and shows the underlying pattern. The workflow panel exposes the data workflow to the user, and enables the user to connect and combine the analytics as desired. If the user is interested in a particular analytic or combination of analytics, he would put the corresponding algorithms into the workflow and link the results to the visualizations.

4 **Analytics methods**

In this section we describe algorithms we use. Most identify specific features of interest, such as repeaters or communications, while the wavelet algorithm is more for general exploration.

4.1 **Windowed Variance**

The Windowed Variance analytic was developed to measure how repetitive or consistent the detected signals are. We compute this works by first creating time windows based on user defined parameters: window size and step size, which allows for overlapping windows. We then create lists of events that fall within each window. As long as events either start or end inside the time window, they are included in the calculation. While this does duplicate overlapping events, trimming an event to fit inside the window would introduce variance into sequences which had little to no variance, such as digital signals. This would unfavorably bias the algorithm, so instead we always use the whole signal lengths.

Once we have the sequences of signals per time window, for each sequence we compute the Coefficient of Variation (CV), defined as the standard deviation divided by the mean, for both the durations of each signal and the gaps between signals. Time windows that have only one or two events are skipped because there is not enough data to calculate the CV for the gap and duration. The run time of this algorithm is $O(2N)$: one pass to create the time windows and a second to calculate the CV. As each band and window is independent, this process is parallelized to make it even more efficient by using threads.

In this manner, signal patterns of high variance (such as communications) can be separated from those of low variance (such as digital signals). This approach is general enough to handle multiple cases in between these extremes, and provides the spectrum of occurrences to the user in case there are interesting patterns in the middle somewhere. However, if the user is only interested in isolating just the digital signals (or filtering out the digital signals), the DPD algorithm is more focused to that specific task.

4.2 **Digital Pattern Distinction (DPD)**

Digital signals often exhibit an extremely regular pattern of consistent signal durations, whereas analog signals are more varying. That is, we define digital patterns as sets of signals where transmission durations and gaps between transmissions are very consistent. Being able to identify a digital or analog signal pattern can help to reduce the problem set; for instance, communication should generally only consist of analog signals when the conversation is among people.

While the goal of the Windowed Variance is to determine how consistent or inconsistent patterns are within constant sized windows, the DPD algorithm was designed to identify and classify sequences of highly consistent sequences of arbitrary length, in order to extract the digital patterns specifically. To compute the DPD, for each frequency band, we spawn a thread that iterates over events, keeping track of a running coefficient of variation (CV) for the durations of both the signals and the gaps between them. To avoid having to completely re-calculate the mean and standard deviation at each iteration, we use an incremental formulation to compute the CV:

$$E(x_{n+1}) = \frac{nE(x_n) + x_{n+1}}{n+1} \quad E(x_n^2) = \frac{nE(x_n^2) + x_{n+1}^2}{n+1}$$

$$\sigma(x_n) = \sqrt{E(x_n^2) - (E(x_n))^2}$$

$$CV = \frac{\sigma}{E(x_n)}$$

(1)

where $E(x_n)$ and $E(x_n^2)$ are zero and $E(x)$ is the running average.

For each event, if both the gap CV and the signal CV are below a threshold then the event is appended to the current sequence, and the algorithm continues on to the next event in the frequency band. If adding the event to the list would exceed the CV’s threshold, the algorithm will terminate the current sequence and save the statistical metrics. The algorithm will then continue with the event that it could not add and repeat the process. In this manner, repetitive digital signals will form long sequences of low variance, while analog signals will not. There is no consensus in the literature of a good CV value. We use 10% because there is a slight variance in duration and gap that can be due to many factors e.g. noise or calibration. The CV can also be changed by the user to fit their needs.

One constraint of using the CV is that it does not work on interval scales, but since the durations of both the signals and gaps are positive ratio scales, we do not run into this problem. Another potential side effect of this approach is that as more events are added, each new event has less impact on the mean and standard deviation. Thus, a sequence of events could start very regular and gradually become more erratic but still be added to the sequence. As there is some data collection error though, this actually helps in creating longer sequences, even if there is some noise or dropped signals. Even in the case where an event is not captured correctly and a long sequence is split in half, both parts should still have the same CV in both gap and duration and so
4.3 Repeater detection

A repeater is a device that receives one set of signals and rebroadcasts them - often to extend the range between low powered devices or to cross terrain such as hills that would block communication. In our signal data, a repeater shows up as a series of events that have the same start time and duration across at least two different frequencies (i.e. pairs of the initial transmissions and the rebroadcast transmissions).

To detect these, the repeater algorithm iterates through a sorted list of all events, and groups events that share the same start time and duration as candidate repeated signals (i.e. synchronous events from two or more frequency bands). Then we group candidates with identical frequency bands, as multiple repeated signals on the same frequencies are likely the same repeater. While it is possible for two different repeaters to share some subset of frequency bands, they would generally interfere with each other if they were operating on the same set of frequencies. This process is straightforward since the initial candidates have their events sorted by time. We do not incorporate the geographic information in this computation because of its low precision. Also, it is possible for a repeater to be mobile. And even pairs of signals in which both signals lack good geographic information can be relevant for creating or modify a color gradient and provide histogram of each data property. The algorithm workflow (F) allows the user to visually select the algorithmic methods by directing the flow of data from source to view. The search window (G) gives the user the ability to draw a specific signal pattern and search for it in the results. Previously drawn patterns can also be loaded.

The run time of this algorithm is $O(N)$, and like the Windowed Variance algorithm, it is heavily parallelizable due to the calculation being independent of the frequency band.

4.4 Communication detection

For communication, we created a rule based algorithm. We looked at several papers on the proper duration of gaps and pauses between communicating individuals. Most of the research, however, is done on conversation that is either over the phone or in person. Since our data is comprised of half-duplex (i.e. only one transmit at a time) handheld transceivers, conventional conversation rules do not apply quite as rigorously. Thus we applied some more relaxed rules. First, the duration of each signal in the communication needs to be at least one second. We found this to be reasonable since any confirmation takes more than one second to transmit when following radio transmission etiquette.
for 0 < k < n (where n is the smallest number large enough for D_0). At each recursion the µ values are the mean of the corresponding data series, which approximate the variance at each resolution, and hence at each frequency scale. More complicated wavelets can be calculated by changing the functions used to calculate D_k and S_k. We found that the basic D_k and S_k functions above provided sufficient results for our signal data. Though we are generating a significant amount of data from sampling and overlapping, the amount of stored data is only log_2(n) in size per wavelet.

Once we have computed the wavelets, we need a visual representation. Our goal is to place wavelets with similar signatures next to each other, so there are many possible techniques, such as clustering or dimensionality reduction. We chose to use the dimensionality reduction technique known as PCA [15], as it is simple, but good at extracting the most prevalent trends in the data. We found that it also arranges the points based on duration and consistency. While more complicated dimensionality reduction techniques exist, PCA produced reasonable results that were sufficient for our analysis.

5 Views

In order to interact with and understand the results of the analytic processes, we use a number of visual representations and interfaces.

4.5 Wavelets and Dimensionality Reduction

One way to look at the data is to analyze patterns of activity according to their similarity. We can define these patterns by treating each frequency band in the data as a time series. Then the sequence of captured signals expresses itself as a square wave in this time series and similar patterns can be detected through frequency analysis. We choose to use wavelet scalograms [20] both because they are naturally tuned to such square wave patterns (unlike Fourier analysis which works with sinusoids), and because wavelets are rather resistant to phase shifts and noise: similar patterns will have similar wavelet signatures even if the patterns are shifted slightly or parts of the pattern are missing. Conversely, different signals should produce different wavelets.

While wavelets are often useful in signal processing applications for general frequency analyses, in which the data is continuous, the captured signal data is stored as discrete data made up of events with start times and durations. To generate time series to use in the wavelet scalograms, we first sample the data according to sliding time windows, which are defined by user controlled parameters such as window size, sampling rate and overlap amount. Since the data is sorted temporally within each frequency band, these windowed time series can be generated in a streaming manner. For each frequency band, we first initialize all sample point values in each window’s time series to 0. Then we iterate over the events that intersect temporally with the time window, setting the values in the intersection to 1. Each window’s time series now comprises the D_k array used in the wavelet calculation. The scalogram (µ_0, µ_1, ...) we calculate recursively as:

\[
D_k = (d_{k,1}, ..., d_{k,2^{n-1}}) = \left(\frac{d_{k-1,1} + d_{k-1,2}}{2}, ..., \frac{d_{k-1,2^{n-2}} + d_{k-1,2^{n-1}}}{2}\right)
\]

\[
S_k = (s_{k,1}, ..., s_{k,2^{n-1}}) = \left(\frac{s_{k-1,1} - d_{k-1,2}}{2}, ..., \frac{s_{k-1,2^{n-2}} - d_{k-1,2^{n-1}}}{2}\right)
\]

\[
µ_k = \sum_{i=1}^{2^n} \frac{s_i}{2^n}
\]

The timeline, which can be seen in Figure 3.C, provides a series of simple plots to filter and examine the data. In all timeline plots, the x-axis corresponds to time, while the y-axis is either one of a number of derived values or an attribute from the dataset. For instance, plotting frequency band versus time (as in Figure 3.C) produces a Gantt chart that provides a simple and intuitive visual summary of the data set. In this example, the signals are colored via a user defined color map (defined via the color map editor in Figure 3.E).

One derived values that was found helpful is the proximity of the airplane to the signal sources, as closer signals are more detected more precisely. We use two proximity metrics: the first is the standard proximity which we calculate in a similar manner as in [6] and the second proximity metric divides the signals based on which side of the airplane a signal originated from. This allows us to see when the airplane makes a turn. It also separates entities that look close in proximity space but are on opposite sides of the airplane. While not that relevant to the results shown in this paper, our collaborators found this very helpful in analyzing the behavior of the data collection platform itself.

The map view provides geospatial reference to the user and is shown in Figure 3.A. We use the Google Maps API to generate the background map and plot the points using OpenGL. It was necessary to include the map not just to provide geospatial information but also to help interpret the results from the analytical methods. When a selection is made in the main view, the map view can provide additional functionality depending on which metric is being viewed. For repeaters, we draw lines between pairs of repeated signals. This helps identify which source is the repeater, as a single repeater would link to multiple transmitters. For communications, we draw lines between the initial transmitters and the first responders.

Sometimes the GPS locations are not accurate and thus showing error ellipses is useful. We map the size of the ellipse to transparency, where the bigger the ellipse the more transparent it becomes, illustrated in Figure 2. This makes accurate GPS information stand out while the less accurate fades away, and having several overlapping big ellipses accumulates to give a better approximate signal location.

The main view, pictured in Figure 3.B, holds the visual results computed by our algorithms. As the points in this plot are aggregations, and not individual signals, we can not employ the same per signal color mappings used in other views. Instead, color is mapped to another selected property, such as the density on screen or the variance of the aggregated data. The axes depend on the analytic. For the wavelet projection, the axes are the first two dimensions of the PCA projection. For Windowed Variance, the axes are the standard deviations of the duration of signals and of the gaps between signals for each time window, on a log scale. For the digital, communication, and repeater algorithms, we map number of associated events to the x-axis and average duration of the events to the y-axis.

We also added a search window, shown in Figure 3.G, which provides the capability to look for a particular signal pattern in conjunction with the wavelet view. The user either draws a pattern in the top segment of the panel or loads a previously saved pattern to commence a search. The system then calculates the wavelet scalogram of the
search pattern, and uses the PCA's component matrix to project the search pattern into the wavelet visualization. We use a cross-hair representation as a glyph to help the user identify where in the PCA space the pattern is located, with a circle of a fixed radius around the target pattern to both make the target more easily visible and to indicate neighbors. Results inside the circle are displayed under the search button in the search window, ranked by proximity to the target signal pattern. Saving patterns from one dataset and loading them into another dataset allows the user to see how it is mapped in the different space, which is important since PCA is not consistent between datasets.

In the signal inset, there are two ways of showing the underlying signal pattern, which is shown in Figure 4. The standard way to visualize signal data is to draw lines that represent the start and end time of each signal on the x-axis. The y-axis splits the window based on how many sets of signals are portrayed. Our method keeps the same x-axis setup but changes the y-axis so that each sequential event is above the previous one. Then we connect all the signals forming something similar to a line plot. Steep slopes represent a quick succession of events, while gradual slopes show long pauses between events. One benefit of our method is it provides a visual metaphor for the signal patterns, for instance, when digital patterns are presented as various types of bars and lines. The standard visualization technique similar to [14]. In our implementation of this technique, we first render to a high precision density buffer \( D \) which keeps track of the total amount of overplot and to a high precision color buffer \( C \) which blends the input color information with opacity inversely proportional to the density information to result in an average color that is a user defined minimum opacity level and of the total amount of overploting that occurred. As before, we apply a logarithmic transfer function, but then we use the resulting number of elements increases. The standard 8-bit alpha buffer only allows for a maximum overplotting of 256. Furthermore, the opacity in this manner we guarantee that any outliers will have at least maximum level of overplotting that occurred. By calculating the final operation panels. Selecting an operation displays its parameters at the bottom. Each operation has its own rules on how many and which operations it can be connected to. Clicking on another operations links the filter and exclude selections have to be handled carefully in algorithms where the data is aggregated. Removal of one signal from within an aggregated group of signals would invalidate or at least change the value of the group's derived metric. To avoid such errors, we simply remove the derived point when this happens. For highlighting color interactions are applied only to their representation within the detailed signal insets instead of the aggregate point.

The user can adjust the colors by selecting a premade colormap or by manually changing the colormap, and can select which property to map to each color. The left color wheel provides color selection for the highlighting, and the color legend shows the current color map. The color legend is fully customizable with abilities to add new color or change existing ones. Any changes made to the color legend are saved, even when switching between colormaps. The calculation tab allows parameters to be changed and the algorithms to be queued up. For wavelet calculation, size of time window, sampling and overlap can be set. The last tab is the interface to the time view and controls what attribute is plotted.

5.1 Opacity Tone Mapping

Our approach renders large amounts of data to the screen, often results in many points or lines overplotting. A common way to resolve this overplot is to make them semitransparent and use alpha blending to combine them. However, this very quickly runs into limitations as the number of elements increases. The standard 8-bit alpha buffer only allows for a maximum overplotting of 256. Furthermore, the opacity has to be set so low that outliers are nearly invisible. In order to keep both the opacity of outliers high and the combined opacity of dense overlap from overflowing the alpha buffer, we utilize opacity scaling techniques similar to [14]. In our implementation of this technique, we first render to a high precision density buffer \( D \) which keeps track of the total amount of overplot and to a high precision color buffer \( C \) which blends the input color information with opacity inversely proportional to the density information to result in an average color that is fully opaque. We then combine these buffers with a transfer function to render the final pixels \( P \) to the screen. In order to be able to handle many orders of magnitude variance in the data density, we then combine these buffers with a logarithmic transfer function to render the final pixels \( P \) to the screen, which is defined as:

\[
P_{x,y} = C_{x,y} \times (o_{\text{min}} + (1 - o_{\text{min}}) \times \frac{\log(D_{x,y})}{\log(D_{\text{max}})})
\]

(3)

Where \( o_{\text{min}} \) is a user defined minimum opacity level and \( D_{\text{max}} \) is the maximum level of overplotting that occurred. By calculating the final opacity in this manner we guarantee that any outliers will have at least opacity \( o_{\text{min}} \), that no overplotting exceeds the maximum opacity and that the system can handle orders of magnitude of overplotting.

In several views, we alternately use just the density buffer to generate a density heat map, discarding the initial color mapping. As before, we apply a logarithmic transfer function, but then we use the resulting value as a lookup into a 1D color map texture. In this heat map, the pixels are computed as:

\[
P_{x,y} = \text{ColorMap}(\frac{\log(D_{x,y})}{\log(D_{\text{max}})})
\]

(4)

6 Exploration and Interaction

We aimed to keep our UI and interaction design as simple as possible. Within the main window, the user can mouse over or click on a point in the main view to create an signal inset that shows the underlying signal pattern(s) that the algorithm used to compute that point. Several of these windows can be opened with a click for comparison. The window can be resized or moved, and each window points back to the aggregated value. For a direct comparison, multiple points can be loaded into one window.

We use the rectangle and lasso selection in our system. The rectangle selection is used in the timeline while lasso is used in the other two views. In the timeline, the user is interested in a particular range for a given attribute, so it makes sense to use rectangle selection. Lasso selection is used when the selection process is difficult, as is the case in the other views where the user has to sort through multiple points to get the selection he wants.

The control menu, pictured in Figure 3.D, has three tabs. The main tab controls which calculated view is shown, allows minor visualization adjustments, and changes global selection properties such as configuring which operations affect which views (e.g. the user might want to exclude all repeaters from the map but still keep them in the time view). There are three types of selections: filter, highlight, and exclude. The filter and exclude selections have to be handled carefully in algorithms where the data is aggregated. Removal of one signal from within an aggregated group of signals would invalidate or at least change the value of the group's derived metric. To avoid such errors, we simply remove the derived point when this happens. For highlighting color interactions are applied only to their representation within the detailed signal insets instead of the aggregate point.

The user can adjust the colors by selecting a premade colormap or by manually changing the colormap, and can select which property to map to each color. The left color wheel provides color selection for the highlighting, and the color legend shows the current color map. The color legend is fully customizable with abilities to add new color or change existing ones. Any changes made to the color legend are saved, even when switching between colormaps. The calculation tab allows parameters to be changed and the algorithms to be queued up. For wavelet calculation, size of time window, sampling and overlap can be set. The last tab is the interface to the time view and controls what attribute is plotted.

6.1 Algorithm Workflow

We had two goals in mind when designing the algorithm workflow: easy to use and modular. We achieve these goals by providing the user with a visual metaphor while at the same time making it easy to add new operations to the system. For instance, for a programmer to add a new layout calculation, they would just need to inherit the operation class and fill in the virtual functions to create this new layout.

Figure 3.F shows the workflow view. The drop down menu adds operation panels. Selecting an operation displays its parameters at the bottom. Each operation has its own rules on how many and which operations it can be connected to. Clicking on another operations links them together. Operations can chain together, feeding the results of one operation as input into others. A link or operation can easily be removed with right click. The view has standard panning, zooming and moving of operations. Any changes are saved and loaded when the system is rebooted. The user can also choose to load a saved workflow or export the current workflow.

There are two unique operations, the data and output operation. The data operation is the starting point and does not take any input. The output operation grabs the preferred x and y axis of its input. When the compute button is pressed, all outputs are added to the drop-down menu for viewing. There, the user can select which results he wants to see and change x and y variables if he wants.

7 Case Studies

To evaluate our system, we provide case studies demonstrating each type of algorithm and their effectiveness. The particular data sets used
in this work were collected during test flights of a spectrum measurement detection platform in two geographical areas in the US. The data includes real world environmental noise and large volume of events from NSOIs that make the separation of real transmissions from noise induced transmissions difficult. Each flight generated a large log of transmission events, composed of geospatial information (including uncertainty ellipse), start/end times, signal quality, bandwidth, and Signal to Noise Ratio (SNR).

7.1 Repeaters

In this case, we use our system to explore one of the datasets collected from flight tests performed near the U.S.-Mexico border. We start by applying the repeater detection algorithm, and plotting it in the main view, as shown in Figure 5. Since we compute potential candidates, not every point is necessarily a repeater. For instance, some of the selected points do not have enough GPS data to determine whether they are repeaters or not, while others contain noise signals. For instance, there is a large concentration of noise points in the bottom left, as there is a high chance of finding incidental pairs with low count and duration. The points in the regions that extend almost asymptotically along both axes are more likely to be of interest. While this plot is rather skewed, it is these tails that are important, so it would be counter-productive to address this by applying a log-log scale.

To inspect these results in the map view, we select points from the upper left of the plot, as candidates that have more signal pairs are more likely to be repeaters. By mousing over each point, we can use the signal inset to inspect each pattern. Once a good candidate is found, it can be selected. Then, we draw lines on the map between each pair of events that have the same start time and duration. Ideally, we expect to see star patterns, where several client transmitters are utilizing one central repeater. In Figure 5, we show two different repeater candidates. The top candidate has most of the signal pairs pointing to a small region of space, while there are a few pairs that point a bit to right. When looking back at the signal inset we see a large amount of signal pairs and then a long break followed by a couple more signal pairs. This indicates that the first set of signals is the repeater on the left and the small set is either the same repeater moved or more likely a different repeater using the same set of frequency bands. In the bottom candidate, we can see constant stream of signal pairs which all points to a signal region, indicating a highly likely repeater location.

7.2 Communications

This case searched for communication patterns using the communication detection algorithm to extract likely candidates, and plotting the number of transmissions versus the total length of the conversation for each candidate. We found that communication patterns occur less frequently than any of the other patterns. While this could be due to our algorithmic settings being too strict or not inclusive enough, it is also quite possible that there were simply very few conversations occurring when the data was collected. Further verification with ground truth knowledge would be needed to confirm either way.

As before, we inspect the candidates using the geospatial view. By plotting a line between communication transmitters in series on the map, one would expect the line to simply go back and forth between the communicators as they take turns talking. In Figure 6, we see two examples of this representation. By selecting group A, a star pattern is created by one transmitter that is mostly stationary and a number of surrounding transmitters. As there were multiple, temporally distinct conversations, and as one party was stationary, it is possible that this is a ‘dispatch’-type communication. Selecting group B reveals a similar nearby pattern that at first would look like the communication comes from many locations, but displaying the error ellipses reveals that it is possible for the communication to simply be between two parties. The combination of the error ellipses better triangulate their locations than any single transmission by itself.

7.3 Wavelet Algorithm

In Figure 7, we show the results from the wavelet calculation mapping color to density heat map. We have opened up several signal insets for comparison. The wavelets separate windowed samples of the signal by pattern behavior, with longer duration signals tended towards the right, analog patterns toward the bottom, and digital patterns towards the top. This comes from the wavelet scalograms, in which digital patterns have a sharp spike in one or two dimensions in a scalogram while analog patterns are more even across dimension.

Wavelets are useful when the user is looking for a general pattern about the dataset. For instance, Figure 7 show wavelet structure of two different dataset. The image on the bottom has far fewer points in the top region, indicating that it has less long digital-like patterns. A benefit of the wavelets is that the user can filter map and timeline views based on durations or digital and analog behavior.
higher frequency bands. We look at two low frequency bands points
have low frequency bands, as normally digital signals are found in the
duration. What is also interesting is that these high average high counts
of low duration. So one unexpected feature of this dataset is that there
shown in Figure 9. Most digital signals would have high signal counts
look at analog signal patterns. Just like in the digital selection, the
recomputed. Another potential use for the Windowed Variance is to
plotting the SD of the gap and duration, the user can quickly decide
variation in signal length, gaps, or both.

![Image 49x646 to 167x738]

The wavelet view plots sequence segments projected down
from a high dimensional space. Inspection reveals that sparser patterns
ended up towards the left, with more digital-like patterns at the upper
right and more analog patterns in the lower right. Different datasets
have different distribution of signal types. From the lack of points in mid
upper region of the bottom image, it is clear that the top dataset has
more digital signals than the bottom dataset.

![Image 58x372 to 283x554]

Fig. 7. The wavelet view plots sequence segments projected down
from a high dimensional space. Inspection reveals that sparser patterns
ended up towards the left, with more digital-like patterns at the upper
right and more analog patterns in the lower right. Different datasets
have different distribution of signal types. From the lack of points in mid
upper region of the bottom image, it is clear that the top dataset has
more digital signals than the bottom dataset.

![Image 174x646 to 289x738]

Fig. 8. The Windowed Variance distributes patterns based on their vari-
ance in gap and duration. The y-axis is the variance in the gap while the
x-axis is the variance in duration. Analog patterns are found in the top
right because they have large variance in both gap and duration.

7.4 Repetitive Patterns

We examine both the DPD algorithm, Windowed Variance and their
views. We first start with the Windowed Variance. We set the window
size to 60 seconds with no overlap between windows. Figure 8 shows
the results of the Windowed Variance computation. The big cluster of
patterns to the top right are analog signal patterns as they have large
variation in both gap and duration. However, the striated groups to
the left and bottom of the plot are generally digital signals, with low
variation in signal length, gaps, or both.

The Windowed Variance view is useful when the user is looking
for a pattern that is not entirely digital. For instance, there might be a
series of events that have the same duration but different gap times. By
plotting the SD of the gap and duration, the user can quickly decide
how much variance to allow in either direction through selection. This
gives Windowed Variance an advantage over DPD where it has to be
recomputed. Another potential use for the Windowed Variance is to
look at analog signal patterns. Just like in the digital selection, the
user can decide how much variation to allow.

We computed the DPD algorithm with a 10% CV. The results are
shown in Figure 9. Most digital signals would have high signal counts
of low duration. So one unexpected feature of this dataset is that there
are several peaks of high signal counts with moderate to high average
duration. What is also interesting is that these high average high counts
have low frequency bands, as normally digital signals are found in the
higher frequency bands. We look at two low frequency bands points
and notice that we can triangulate their position. With this piece of
information, it would be possible to commence another fly-by to gather
more information if warranted.

7.5 Combining Approaches

Our discussion so far has focused on each algorithm separately, and
how each one can help the user find a particular pattern or generally ex-

dlore a dataset. We also have shown how the time view and map view
can be used in conjunction with the results to facilitate in the discovery
of patterns. However, repeaters, communication and DPD/Windowed
Variance can also be combined to find more complex patterns.

For instance, the user could combine the DPD and repeater algo-
rithms. This would find digital patterns mirrored across multiple fre-
cuencies. While this could be a digital signal being retransmitted,
it could also be a broadcast from a single source of a digital pattern
across multiple frequencies, such as an alert or beacon. First, we
would construct our workflow that would go from the data, into the
digital algorithm and pipe those results into the repeater algorithm.
We filter the map based on candidates we found, as shown in Figure
10. Mousing over the points, we notice candidates that have many
error ellipses overlapping, indicating very likely candidates as all the
signals are coming from the same location, as would indicate a beacon
instead of a digital repeater. Zooming in on the candidates reveals that
the signals originate from the McClellan airfield. Thus, it is reasonable
to deduce that this is an air-control navigational beacon.

Each combination of algorithms would detect different patterns. A
repeater communication would use a repeater to extend the range of a
conversation. A common example situation is near mountains where
the handheld devices would be communicating over a repeater at the
top of the mountain. Digital communication could be two or more
machines interfacing with each other. A combination of all three is
possible as well: Two machines could be communicating over a re-
peater to extend the range or broadcast to other frequency bands.

8 Expert User Collaboration

This project was aimed at the development of a visual tool for expert
analysts to use to derive actionable insights from large data sets of sim-
ple raw measurements, with the goals of providing them with highly
efficient, interactive analytic methods and an interface that would be
intuitive enough for them to learn. As such, throughout the devel-

one, many of which led to concrete insights.

Due to the specific expertise of the target user base, it was unfeasi-
ble to find a sufficient number of users for a formal user study. Also,
many potential applications for our system involve sensitive or clas-
sified data, so critical evaluation of the analysts’ work flows or tool
usage is also not viable. However, through regular user tests and feed-
back from the few expert users we were working with, we were able
to informally evaluate the utility and intuitiveness of the tool. Here,
we describe examples of their usage of most major components in the
system, many of which led to concrete insights.

The analysts found the tools easy to use. In general, the frequency
view was a clear base for the analyst to work from, as the users found
it easy to understand. In particular, the time vs frequency view was
often the initial view of choice of the signal environment. Previously,
the analysts interacted with the data one frequency at a time. The
timeline representation made understanding patterns out of a dense
environment much easier while retaining the users’ mental picture of
the data space. The color editor allowed subtle shading within a data
type and helped to highlight or exclude outliers visually. Features of
the color wheel such as saving of the color wheel settings were found
helpful for repeated analysis over several sessions or for collaboration.
The addition of highlighting and filtering greatly improved the expert users’ exploration of the data set over their previous methods. Rapid selection of different data types and zooming into detail features for more in-depth analysis allowed the analysts to rapidly explore the features of the environment from the macro viewpoint down to detailed perspectives of related events.

Some of the data analytics (such as communication) often only identified a few events. The ability to see those events within the context of the entire frequency set allowed the users insight into whether the algorithm had selected a true communication channel or not. Because the number of such events may only be a few dozen out of the 500K to 1M events in the data, finding the related events needed to be efficient. Highlighting was sufficient for analytics with about 100 events. Filtering was important with fewer events. A common process was that the fast filtering would be used to remove all nonselected data and then zooming would be used to get closer to the selected data. Restoring all the events was a single button press which would put all the data back in the display. This sequence was intuitive to the users and helped in the rapid validation of low event analytic output.

The repeater analytic produced interesting results that still needs more analysis. The algorithm required absolute time synchronicity of the events that were put forward as candidates. On zooming into the details of the events, the expert users rapidly hypothesized new analytic algorithms for refined or different results.

Novel axis definitions in the digital, communication and repeater analytic produced the results that still needs more analysis. The algorithm required absolute time synchronicity of the events that were put forward as candidates. On zooming into the details of the events, the expert users rapidly hypothesized new analytic algorithms for refined or different results.

Novel axis definitions in the digital, communication and repeater analytic helped translate the algorithms results into operator-centric understandings. The analytics produced many candidates and the novel axis definitions helped the experts in navigating those candidates, since they related to the experts’ understood signal concepts. For example, one axis of the digital detection metric is the signals’ mean duration, which lets users utilize their domain knowledge to separate digital signals into different duration subclasses. For the communication analytic the number of events in the communication was used as an axis, which allowed the users to easily distinguish by numbers of interchanges. These flexibilities made exploration of highly likely events possible.

While not one of the original goals, the expert users also found the tool helpful for identification of subtle installation and measurement errors at a macro level. Our tool revealed indicators at the overview level which guided the users to drill down into specific times and measurements from a multi-variable perspective using colorization by data type and filtering and selection across the views. Although the data analysis was done on datasets after the flight tests were over, this analysis was able to point to subtle installation issues that were specific to particular frequencies and aircraft flight dynamics. These phenomena were not visible with previous data centric analysis methods as these subtle errors had not been previous hypothesized or analyzed.

9 Conclusion

In this paper we have presented a visual analysis system for exploring and analyzing radio signal meta-data. We showed how the visual representations help analysts identify, understand, and compare different signal patterns. In designing our system, we have developed novel algorithms for quantitative detection of patterns such as digital transmission or communications. We have shown how combinations of such algorithms along with visual interactions can lead to further insights. We have also demonstrated the usefulness of our system, particularly the algorithms and visualizations, with experimental results derived from data collected from actual airborne sensor platform tests. Such a system could only be more powerful when put into the hands of analysts who need to understand intercepted radio transmission patterns in the field, either forensically or in real-time.

Though the work in this paper has been focusing on exploring offline data, in the future, our system will move towards integration with real-time data collection. Our current analytics aim to be efficient enough for real-time usage, but future situations or additional analytics might necessitate further scaling and acceleration techniques, such as GPU techniques or out-of-core processing capabilities. Incorporating relevant real-time properties such as the location/orientation of the plane could also be relevant in real-time situations. Lastly, the contents of recorded signal data is generally sensitive information, so we only had access to the signal metadata. However, when the signal contents are available, their analysis would be a beneficial subsequent step; our approach can be used to identify sequences of signals that could form a communication, at which point the analyst would apply other communication analysis approaches based on content.

10 Acknowledgments

This research has been sponsored in part by the Northrop Grumman Corporation, National Science Foundation through grants IIS-1320229 and CCF-1025269, and Department of Energy through grants DE-FC02-06ER25777 and DE-FC02-12ER26072.
REFERENCES


