# VisuaLyzer: An Approach for Rapid Visualization and Analysis of Epidemiological Data

by

David N. Reshef
S.B., E.E.C.S. M.I.T., 2008

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology
May 2009 [JUNE]

Author________________________________
Department of Electrical Engineering and Computer Science
May 17, 2007

Certified by__________________________
Paris Sabeti
Professor of Evolutionary and Organismic Biology
Thesis Supervisor

Accepted by____
Arthur C. Smith
Professor of Electrical Engineering
Chairman, Department Committee on Graduate Theses

# VisuaLyzer: An Approach for Rapid Visualization and Analysis of Epidemiological Data

by
David Reshef

Submitted to the
Department of Electrical Engineering and Computer Science

May 8, 2009

In Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science

## Abstract

The ability to capture, store, and manage massive amounts of data is changing virtually every aspect of science, technology, and medicine. This new 'data age' calls for innovative methods to mine and interact with information. VisuaLyzer is a platform designed to identify and investigate meaningful relationships between variables within large datasets through rapid, dynamic, and intelligent data exploration. VisuaLyzer uses four key steps in its approach:

1. Data management: Enabling rapid and robust loading, managing, combining, and altering of multiple databases using a customized database management system.
2. Exploratory Data Analysis: Applying existing and novel statistics and machine learning algorithms to identify and quantify all potential associations among variables across datasets, in a model-independent manner.
3. Rapid, Dynamic Visualization: Using novel methods for visualizing and understanding trends through intuitive, dynamic, real-time visualizations that allow for the simultaneous analysis of up to ten variables.
4. Intelligent Hypothesis Generation: Using computer-identified correlations, together with human intuition gathered through human interaction with visualizations, to intelligently and automatically generate hypotheses about data.

VisuaLyzer's power to simultaneously analyze and visualize massive amounts of data has important applications in the realm of epidemiology, where there are many large complex datasets collected from around the world, and an important need to elicit potential disease-defining factors from within these datasets. Researchers can use VisuaLyzer to identify variables that may directly, or indirectly, influence disease emergence, characteristics, and interactions, representing a fundamental first step toward a new approach to data exploration. As a result, the CDC, the Clinton Foundation, and the Harvard School of Public Health have employed VisuaLyzer as a means of investigating the dynamics of disease transmission.

Thesis Supervisor: Pardis Sabeti

Assistant Professor, Harvard University Department of Evolutionary and Organism Biology

Associate Member, Broad Institute of MIT and Harvard

# Acknowledgements

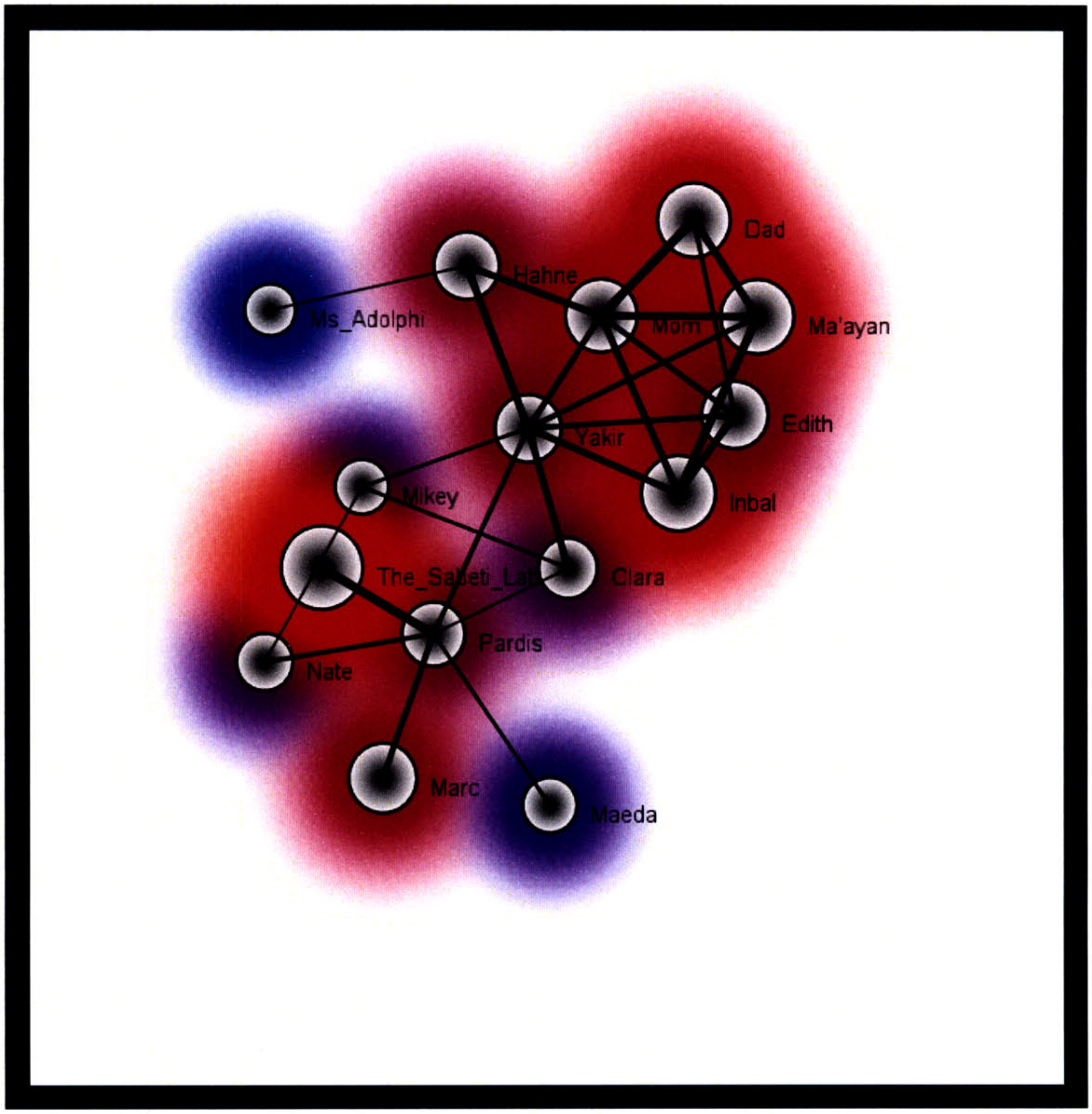


**Figure 0.0 Example of A Gravity Graph: Dave's Life**
It is difficult to explain how much this Gravity Graph means to me.

# Table of Contents

# Chapter 1: Introduction

## *1.1 A New Approach to Data*

We live in the most measured age in history. Our ability to capture and store data is revolutionizing science, technology, and medicine and is changing our ability to answer fundamental questions about our world. In this data deluge, information has changed from a matter of three- or four-dimensional taxonomy to an interconnected web of detailed statistics. The sheer quantity of collected information necessitates a new approach to understanding and utilizing data: viewing it mathematically first, and establishing a context for it later.

This approach has already proven its merits in multiple arenas. For example, the world of internet advertising was revolutionized without the study of the culture or conventions of advertising, but rather with applied mathematics. Internet advertising campaigns simply relied on better data and better analytical tools to match ads to content without any knowledge or assumptions about either. This approach enables the generation and prioritization of interesting hypotheses faster than ever before, based on uncovered trends rather than painstakingly constructed models. This mindset gave rise to the human genome project: information was collected with the idea that the sheer quantity of data would birth interesting hypotheses and findings. The expectation was correct--what followed was an explosion in medicine, biology, and even ethics. Today's data age makes truly agnostic science a realistic possibility, and consequently we must consider a shift in paradigm from hypothesis-driven science to hypothesis-generating science.

## *1.2 Epidemiology in the Data Age*

This wealth of new data will change our ability to combat the public health challenges of today as well as those of the future.

### 1.2.1 A Better Understanding of Disease Emergence and Progression

Despite the scientific information explosion, we still have a great deal to learn about parameters of disease emergence and progression. For example, our understanding of infectious disease outbreaks has evolved from a simple organism-host interaction to a complex microbiological and ecological relationship involving adaptable genomes, microbial development, host variables, and environmental factors. As a result, scientists have sampled hundreds of environmental and clinical factors, from fluctuations in climate to human traffic to syndromic surveillance during periods of epidemics, in an attempt to provide more proactive public health measures to combat disease. In turn, the identification of potentially predictive environmental variables and the calculation of the efficacy of specialized treatments has become computationally difficult and time consuming. This rapidly growing, massive corpus of data calls for an environment that is capable of manipulating and presenting large amounts of data in an easily-accessible, interactive graphic format allows for the rapid identification of the primary variables that define epidemic outbreaks.

### 1.2.2 Rapid Response to Emerging Epidemics

As the experience with SARS, foot and mouth disease (FMD), and other emerging epidemics has made clear, the emergence or reemergence of infectious diseases places significant demands on public health agencies as they try to respond rapidly while maintaining situational awareness and tailoring interventions to the current status of the outbreak. In addition to the difficulties inherent in responding to a crisis in real time, this work has been hampered to some extent by the need to develop new methods while the outbreak evolved (Keeling, Woolhouse et al. 2001; Lipsitch, Cohen et al. 2003; Riley, Fraser et al. 2003; Wallinga and Teunis 2004) or even to make an appropriate choice of existing methods. A famous example of the difficulties caused by this lack of preparedness is the fluctuation in estimates of the case-fatality ratio for SARS, interpreted by many observers as indicating changes in virulence in the virus, whereas in fact it was attributable to inadequate adjustment for dependent censoring (Donnelly, Ghani et al. 2003). These experiences point to the need for a system to allow public health officials and data analysts to visualize and clearly interpret trends in the midst of an epidemic, and to make appropriate estimates of how limited, controlled resources can best be targeted.

## *1.3 VisuaLyzer: A Platform for Exploratory Data Analysis and Visualization*

This thesis introduces VisuaLyzer (Visual + Analyzer), a computational tool for the rapid visual and statistical analysis of large-scale datasets. Furthermore, this thesis examines the specific application of this platform and its unique approach to data analysis toward mining complex epidemiologic and public health data, in order to generate and ultimately test hypotheses about infectious disease transmission.

### 1.3.1 The VisuaLyzer Approach to Data Analysis

The VisuaLyzer tool is intended to provide an intuitive understanding of all meaningful relationships between variables within large datasets through rapid, dynamic, and intelligent data exploration. Fundamentally, VisuaLyzer provides four major utilities:

1. ***Data management:*** The ability to load, manage, combine, and alter multiple databases using a customized database management system
2. ***Exploratory Data Analysis:*** The ability to apply both existing and novel statistical and machine learning algorithms toward identifying all potential dependencies among variables
3. ***Rapid, Dynamic Visualization:*** The ability to simultaneously, visually analyze multiple factors
4. ***Intelligent Hypothesis Generation:*** The usage of automated methods to suggest potentially interesting relationships to users

This platform for data mining and exploration aims to wed the power of computation and the human visual-perceptive system to generate hypotheses in real-time, allowing scientists to ask questions of the data rather than simply confirming what they think are the answers.

The VisuaLyzer approach to data analysis involves four primary steps that stem directly from the above functionality: 1) importing, combing, and manipulating datasets in preparation for analysis, 2) identifying clusters of correlated variables for analysis; 3) visualizing the most significant relationships between variables in identified clusters; and 4) using intelligent algorithms to suggest relationships for exploration to guide hypothesis generation.

### 1.3.2 Step 1: Data Management

In order to provide the user with the ability to manipulate, analyze, and interact with data, VisuaLyzer includes a specifically designed database management system (DBMS), *Viz,* which is based on a column-store infrastructure (Stonebraker et al., 2005). Due to the demands of the visualization environment, which often makes complex queries on the data and requires fast query response times, this DBMS is optimized for speed. However, the inherent Viz infrastructure also allows for a significant level of data compression. Finally, VisuaLyzer includes a robust aggregator, which allows the user to modify, combine, remove, or introduce new variables into databases stored in Viz. The ability to manipulate and intuitively interact with data provided by Viz and the VisuaLyzer aggregator are essential to its exploratory data analysis capabilities.

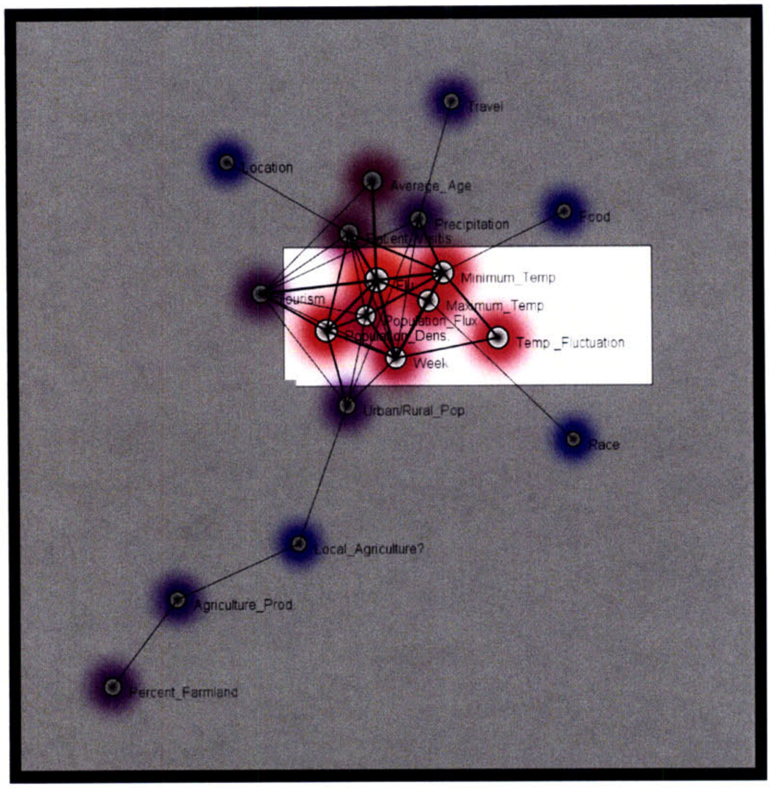
### 1.3.3 Step 2: Identifying Clusters of Dependent Variables for Analysis

To determine which groups of variables from potentially multiple datasets to visualize, it is necessary to systematically identify all the relationships between all variables within these datasets. Furthermore, it is necessary to be able to compare the relative strengths of each of these relationships, independently of whether the two relationships being compared are of the same *type* (i.e. linear, exponential, etc).

To identify *all* associations between *all* pairs of variables within a dataset, independently of the *type* of relationship between the pairs of variables, two novel exploratory data analysis algorithms are introduced. The first is based on the information theory metric, mutual information, and second on the correlation ratio statistic. These algorithms are designed to return a 'correlation coefficient' for each relationship between pairs of variables that is agnostic to the *type* of relationship between the variables. In other words, this correlation coefficient only ranks the *strength* of the relationship, making it possible to compare the relative strengths of, for example, different linear, exponential, and sinusoidal relationships.

To enable an intuitive and efficient interpretation of the output of these algorithms, a novel, specialized visualization called a "Gravity Graph" is introduced, in which the variables in the data set are represented by nodes, and the relationships between them are represented by edges. As indicated by its name, the graph is based on a gravitational model, in which every aspect of a physical, gravitational system between the nodes is dependent on a parameter from the algorithms (for example, the correlation coefficients are used to set gravitational field strengths). Users can interact with this graph by adding or removing variables or relationships between variables and seeing how the gravitational model reacts to these changes. Finally, interesting clusters of related variables can be selected and the most relevant dimensions within these clusters can be identified for analysis. This first step of the VisuaLyzer approach aims to intelligently reduce the amount of dimensions in the dataset, making essential relationships abundantly clear.

Additionally, VisuaLyzer's analytic environment contains a suite of standard algorithms for clustering variables and finding correlations among them, including several clustering algorithms, principle components analysis (PCA), regression analysis, and the ability to calculate multiple correlation coefficients. VisuaLyzer also includes a powerful aggregator and database manipulation machinery to allow users to interact easily with their data.



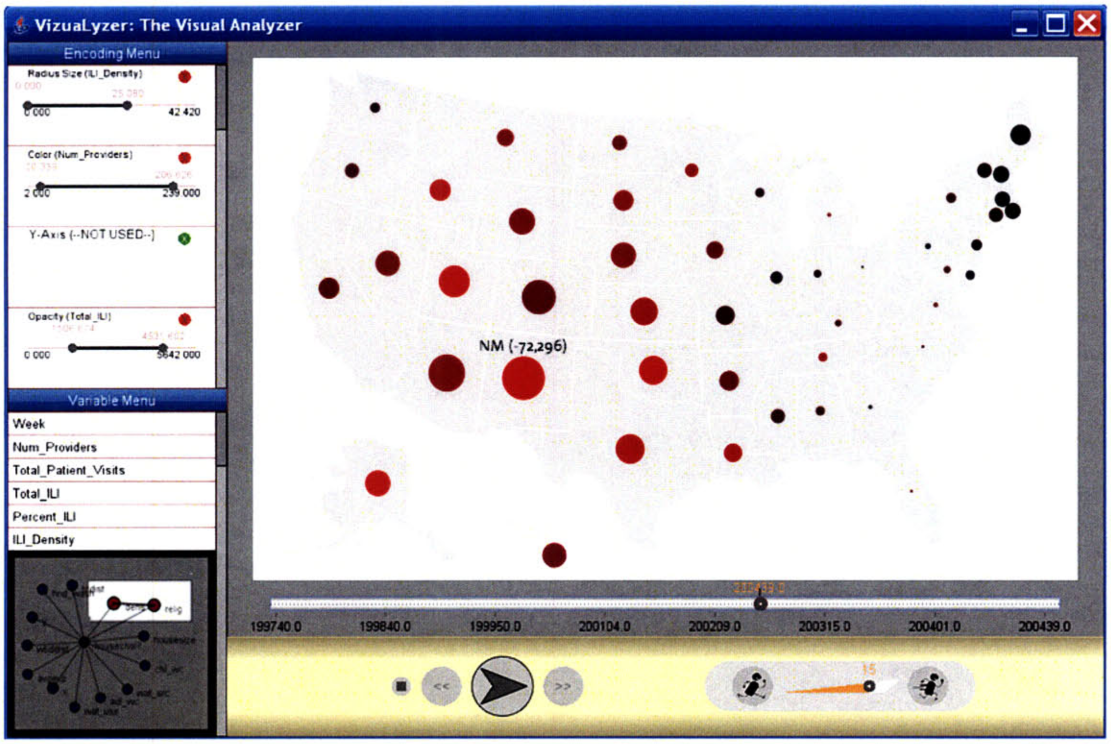
**Gravity Graph**
This is a screenshot of a sample 'gravity graph' that is produced by running mutual information-based correlation algorithm. Each variable in the dataset is a node, and each correlation found is represented as an edge. Nodes' gravitational attraction, color, size, etc. are all set by parameters from the algorithm. This particular graph shows all the relationships between the prevalence of flu (central node), and other variables. Note the highlighted cluster of red (highly correlated) variables that is selected for further analysis. Further details on the mutual information-based algorithm and gravity graphs are presented in Chapter 3.

### 1.3.4 Step 3: A Rapid, Dynamic Visualization Environment

The second component of the VisuaLyzer approach is a clear visualization environment for the rapid analysis of multi-dimensional data. This component aims to do just the opposite of the first: to increase the number of dimensions that can easily be analyzed by a human. Well-designed innovative, visualizations will allow the researcher to see up to as many as ten dimensions at a time in a dynamic, interactive manner, as compared to the traditional two or three in a static manner. This will allow for the analysis of as many of the essential dimensions selected in the first step of the VisuaLyzer approach as possible at a time. In creating intelligent and intuitive visualizations of multi-dimensional data, patterns become obvious within the data and previously unnoticed relationships emerge.

VisuaLyzer's use of intuitive graphics to visualize the complex relationships found between data provides the ability to explore relationships that define the current and future states of diseases. For example, what relationships exist between the introduction of a new drug into a region and disease prevalence in that region, among treated and untreated individuals? What other factors have an impact on this relationship?

A sample screenshot of a "Map View" is shown in the figure below. Despite the fact that the screenshot is static, the scrollbar at the bottom of the view is actively moving through the selected independent variable (in this case, time). In the figure below, as the scrollbar moves through time, the circles grow and shrink according to, for example, the prevalence of a disease at a given location at a particular time point, and change color according to temperature at that location at that particular time point. Several more variables could be loaded onto such a visualization for simultaneous viewing.



**Sample Map View**
This is a screenshot of a sample Map View. In this particular view, number of cases of influenza (size of bubbles) and temperature (color of bubbles) are being displayed for every state in the United States. Furthermore, the scrollbar and controls at the bottom of the view allow the user to 'play' backwards and forwards through an independent variable (in this case, time) and to watch as the bubbles are 'animated' to create an animation of how the disease evolves over time. Further details on the visualization environment and Map Views are presented in Chapter 3.

The ability to view these types of trends rapidly and dynamically provides a powerful tool for identifying confounding factors and generating new ideas about the mechanisms that define public health dynamics.

### 1.3.5 Step 4: Intelligent Hypothesis Generation

Thus far, the VisuaLyzer's new approach to exploratory data analysis and visualization involves identifying all possible correlations between variables within a dataset in a computationally rigorous fashion, and using dynamic visualizations to explore the identified relationships of interest. The final component of this approach is to use randomized and machine learning algorithms to identify further variables of interest based on user-behavior. In other words, when a user loads a set of variables of interest into a visualization, VisuaLyzer's Relationship Suggestor algorithm runs in the background, searching for new variables that would be of interest to the user based on which variables he is currently viewing. This crucial final step of the VisuaLyzer approach is intended to search the space of variables not being examined by the user to find potentially critical relationships that may be affecting the relationships being viewed. Another way of looking at this is that by loading specific variables of interest into a visualization, the user is helping to direct the computer's search for relationships that may be of interest.

## *1.4 Problem Domain*

While every dataset is unique, the VisuaLyzer platform provides a framework for a general approach to exploratory data analysis that remains the same across datasets and across domains. This thesis focuses on the application of this approach toward analyzing epidemiologic and public health data to answer questions like "How does a new influenza strain migrate from Southeast Asia to California?" and furthermore, "What specific conditions contributed to its emergence?" Furthermore, this thesis will explore the application of VisuaLyzer to examine an emerging epidemic, H1N1 (swine flu), and to help design a rapid response strategy to the epidemic in conjunction with the Centers for Disease Control (CDC).

The chapters that follow describe the design and implementation of the VisuaLyzer tool and demonstrate its use on several large-scale epidemiological datasets. Chapter 2 describes the inner workings of VisuaLyzer stores, compresses, manipulates, and interacts with databases. Chapter 3 presents the novel mutual information- and correlation ratio-based algorithms for identifying groups of dependent variables for analysis. Chapter 4 introduces the visualization environment. Chapter 5 presents the algorithms behind VisuaLyzer's intelligent hypothesis generating capabilities. Chapter 6 presents a case study of the VisuaLyzer tool using a combination of approximately 200 independent databases of world health indicators. Finally, Chapter 7 presents conclusions and reflections on the VisuaLyzer approach.

# Chapter 2: Data Management

## *2.1 Viz: A Column-Oriented Database Management System*

Developing a successful data analysis and visualization platform requires particularly fast data processing and querying capabilities. Thus to facilitate and optimize VisuaLyzer's performance, Viz is introduced, which is a database management system (DBMS) specifically optimized for epidemiologic data compression and efficient access by the VisuaLyzer system. Despite being optimized to compress epidemiological data, Viz produces significant compression rates on a wide range of data from varying fields. The following sections will discuss the design of the Viz DBMS and analyze its performance relative to a standard mySQL database.

## *2.2 Factors That Shaped the Design of Viz*

There are a few specific characteristics of epidemiologic data and the VisuaLyzer work load that heavily influenced Viz's design:

1. Viz is a fusion between a data warehouse and a search engine, and has little in common with typical online transaction processing (OLTP) databases. Epidemiology data is collected and uploaded to the system in the bulk rather than through frequent updates. Hence, Viz effectively treats the epidemiology datasets used as static.
2. Large numbers of columns in epidemiology data have relatively few distinct values (for example, binary indicator variables).

3. The VisuaLyzer system's visualizations have a limited set of query types that are run often and could be optimized.
4. The VisuaLyzer system's exploratory data analysis algorithms required sorting and calculating correlations between various combinations of variables. These statistical algorithms access and sort every variable in the epidemiology data, frequently.

These observations drove many of the design decisions made and will be revisited throughout this chapter.

## *2.3 Adapting C-Store: A First Step Toward Compression and Speed*

### 2.3.1 Column Encodings

Given the needs of the VisuaLyzer system and the profile of epidemiologic data, Viz is designed as a customized column-oriented, read-optimized database similar to that described in C-Store (Stonebraker et al., 2005). Since Viz's target data is read-only, it does not contain a Writeable Store, but rather only a Read-Optimized Store. Viz also utilizes the compression schemes described in C-Store to reach a high level of compression on many of the columns in epidemiological data given the minimal distinct values. Viz implements three of the four encoding types described in C-Store.

The first type of encoding, Type 1, is applied to columns that contain relatively few distinct values, in sorted order. Type 1 encoding compresses long sequences of the same value into a single triple: <value, starting index, number of occurrences>. Thus, for every distinct value in the column, only one triple is stored. This encoding is indexed on value using a B+ tree. The distinct values are used as indices for the B+ tree to optimize returning results for queries base on a particular distinct value. Additionally, Viz's implementation of a B+ tree links the leaves of the tree, allowing it to return all the values in the column within a desired range. Figure 2.1 below shows the structure of a Type 1 column, which is frequently used for sorted columns of data, and features optimizations for its common usages.

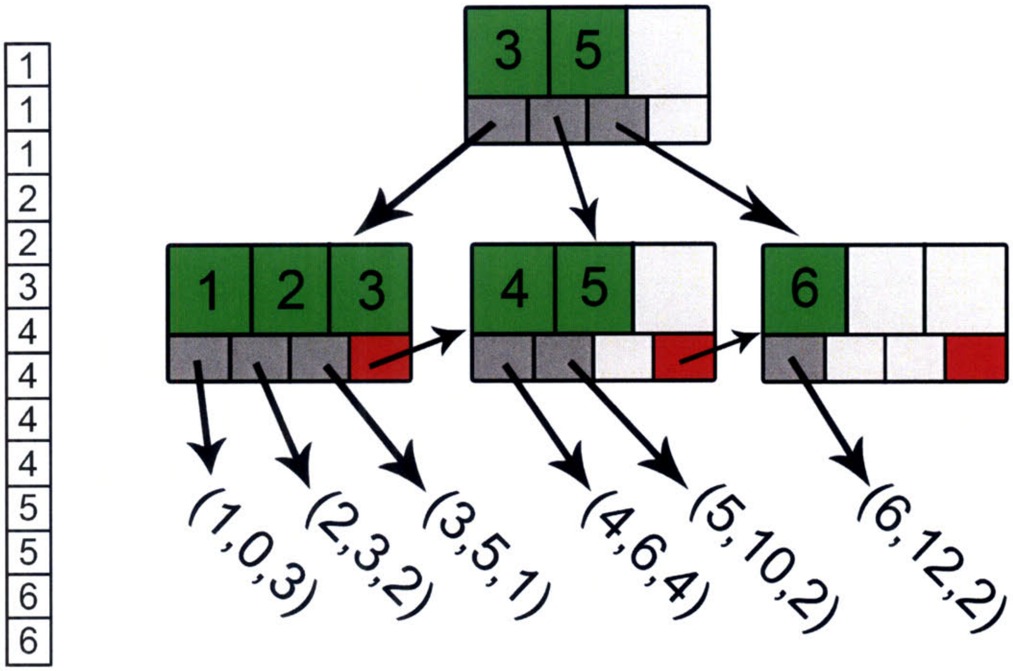


**Figure 2.1. Data Storage of a Type 1 Column**
On the left is a sorted Type 1 column. Note the repetitions of many consecutive values. The B+ tree representation of that column is shown on the right. The B+ tree stores the triplets of <value, starting index, number of occurrences> on the leaf nodes, and entire tree is indexed on the value field of the triplet.

The second type of encoding, Type 2, is applied to columns that are not sorted. Type 2 creates a bitmap for each distinct value that indicates the value's location in the index (see Figure 2.2 below). Since the bitmap has the potential to become very long (a bit for every row in the column), Type 2 is only used when the number of distinct values is less than an eighth of the number of total rows. Since the smallest used types are integers and floats, which are four bytes, the encoded column requires at most four times the space of the original data. The last encoding type, which corresponds to Type 4 from C-Store, simply stores the column of values in an array.

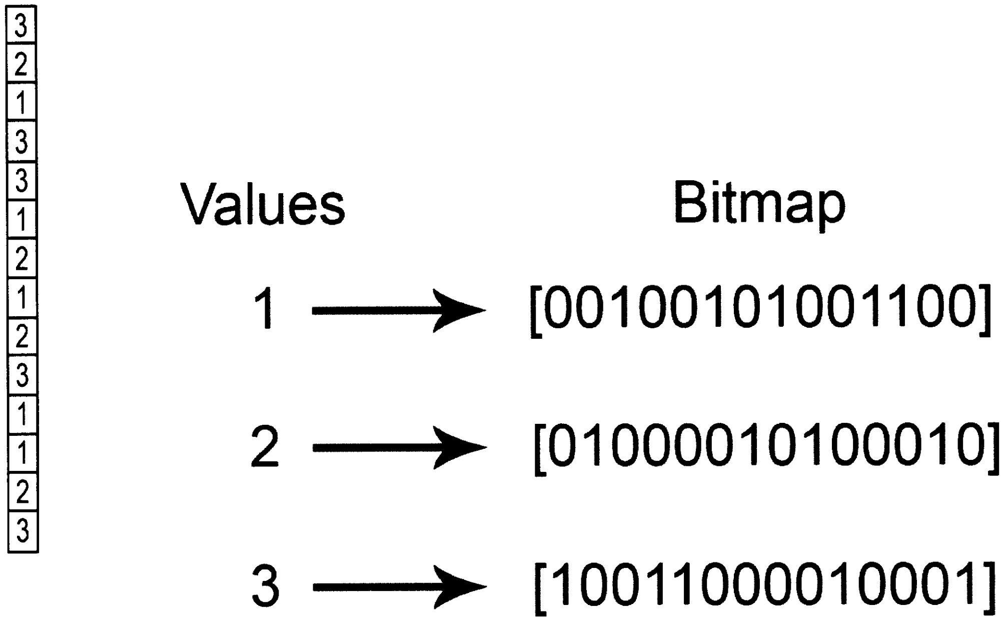


**Figure 2.2. Data Storage of a Type 2 Column**
On the left is an unsorted Type 2 column. Note the repetitions of values dispersed throughout the column. The column is stored as a mapping from each distinct value to bitmaps of their locations on the column. The bitmap has a length equal to the number of values in the column. Thus, the Type 2 Column is only useful in cases where there are a low number of distinct values relative to the length of the column.

### 2.3.2 Using Encoded Columns to Answer Common Queries

In addition to the compression offered by the above column encodings, a column-oriented design offers performance enhancements for many of the queries made by VisuaLyzer's visualizations. Most of the visualizations allow the user to view how a selected set of variables vary depending on the value of another particular variable, like time. A column-oriented design allows these frequent queries to be answered by retrieving only the necessary columns, instead of having to read in the entire dataset row by row. Figure 2.3 below shows a screenshot of a VisuaLyzer Scatter View and the various dynamic filters it uses to create its queries. Figure 2.4 shows a sample query in SQL that represents the type of queries that the visualizations make.

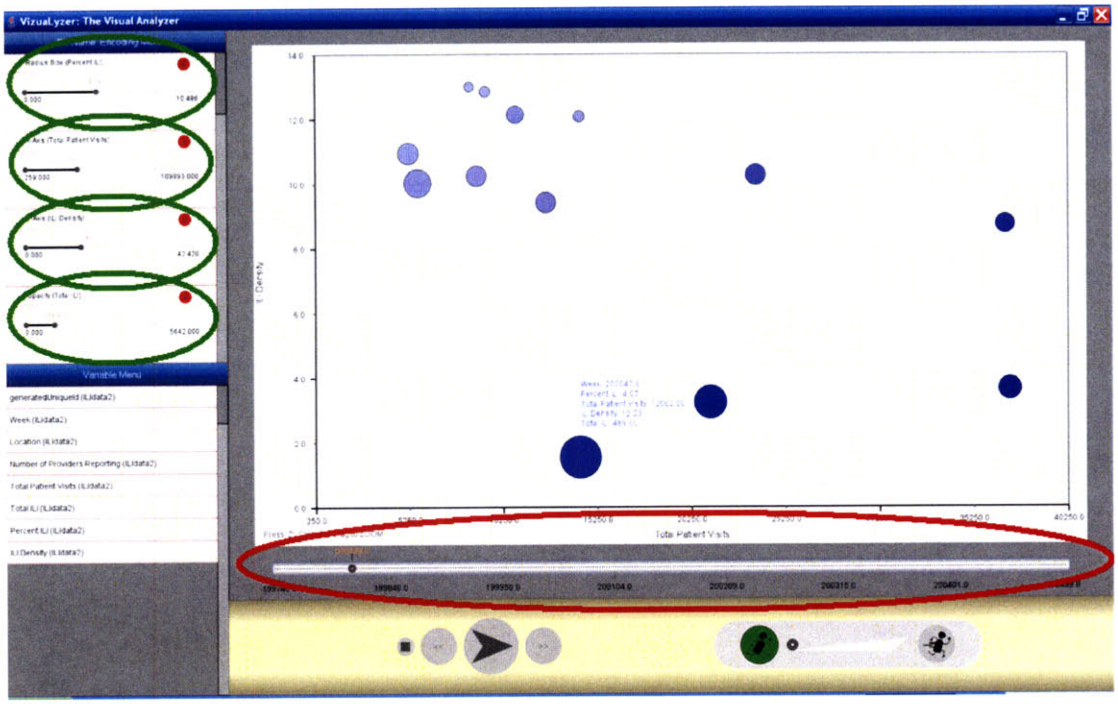


**Figure 2.3. A Screenshot of a VisuaLyzer Visualization**
This screenshot of a 2-D Scatter Visualization shows the selected set of variables that the user is interested in visualizing, highlighted by green ellipses, depending on the value of another particular variable, in this case, time, highlighted by a red ellipse. The variable on the slider encircled by the red ellipse is termed the slider variable, and is especially significant. All these variables represent filters that are used to construct a query.

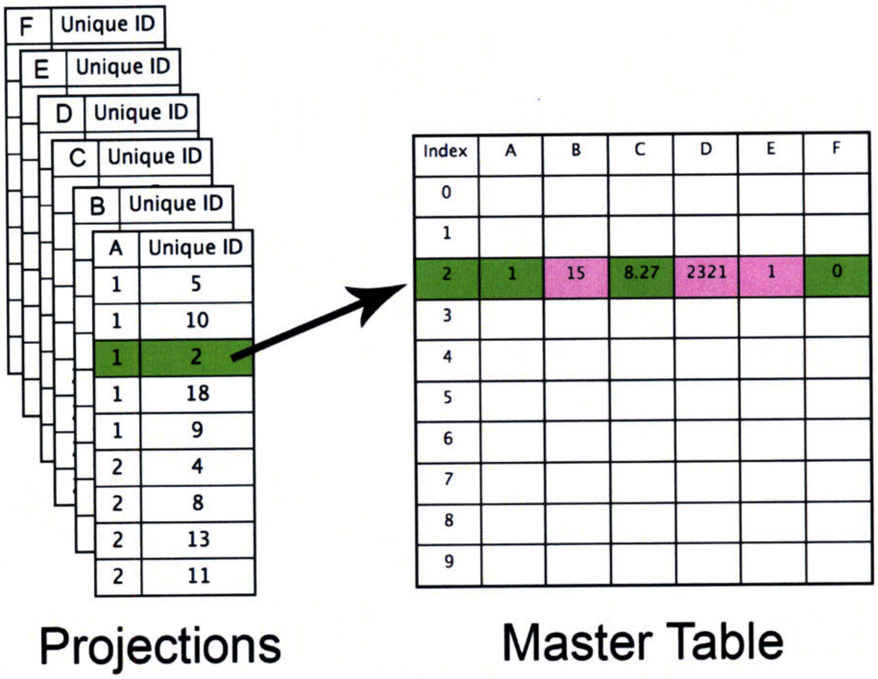
```
SELECT fluData.percentFlu, fluData.numPatientVisits,
     fluData.minTemp, fluData.week,
     fluData.populationDens, fluData.populationFlow
          FROM fluData
               WHERE percentFlu < 90 AND
               percentFlu >10 AND
               numPatientVisits < 20 AND
               numPatientVisits > 0 AND
               minTemp < 75 AND
               minTemp > -10 AND
               week = 199745  AND
               populationDens < 2K AND
               populationDens > 1K AND
               populationFlow < 500 AND
               populationFlow >47 AND
          ORDER BY week
```

**Figure 2.4. A Sample Visualization Query Translated into SQL**
This is the standard structure of a query constructed from a visualization, which seeks values for specific variables for items selected given a set of filters. The VisuaLyzer system requires that such queries be answered on the order of ~.02 seconds, even for very large datasets.

## 2.4 Projections

Instead of implementing join indices as described in C-Store, Viz implements a new idea called projections (Figure 2.5 below). Each data set loaded into Viz is stored in two different ways:

1. ***A master table (or master projection):*** A master projection contains all the columns of a data set and an additional column, corresponding to row number, which serves as the primary key of the master table (called the 'Generated Unique ID'). The columns in the master table are all Type 2 and Type 4 (because Type1 cannot be applied to the unsorted columns), depending on the number of distinct values in a column.
2. ***Helper projections:*** For every single column in the entire master table (i.e. - for every variable in the dataset), a helper projection is created. Helper projections store the columns of the table in sorted order. This is a way of absorbing the cost of sorting the master table by each of its columns ahead of time, and only once. These helper projections consist of two columns: the first is the "variable column," and is a copy of one of the non-primary key columns in the master projection. The second column, the "generated unique ID column," consists of the row at which the variable value appears in the master projection. Both of the columns are sorted on the variable column, meaning that we can often encode the variable column with a Type 1 encoding, while the foreign key column is always Type 4 as it is unsorted and has many unique values.



| A | Unique ID |
|---|---|
| 1 | 5 |
| 1 | 10 |
| 1 | 2 |
| 1 | 18 |
| 1 | 9 |
| 2 | 4 |
| 2 | 8 |
| 2 | 13 |
| 2 | 11 |

| Index | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 0 | | | | | | |
| 1 | | | | | | |
| 2 | 1 | 15 | 8.27 | 2321 | 1 | 0 |
| 3 | | | | | | |
| 4 | | | | | | |
| 5 | | | | | | |
| 6 | | | | | | |
| 7 | | | | | | |
| 8 | | | | | | |
| 9 | | | | | | |

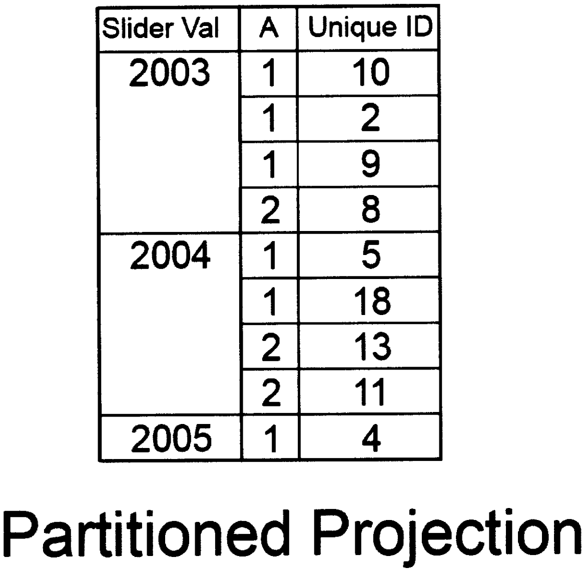
**Figure 2.5. Projections**
Each projection's Unique ID column stores the row on the master table to which the corresponding value in the projection's Variable column belongs. Using the Type 1 Column to filter out rows by value is efficient because of the B+ tree structure. From those row values filtered, the Unique ID for each is used to find other values in that row, in this case B, D, and E. Note that there exists a projection for every variable in the table.

Projections are designed to optimize the combinatorial algorithms and visualization performance. More specifically, VizuaLyzer's exploratory data analysis algorithms require sorting by and calculating the correlation between every possible pair of columns within the dataset, over many different ranges of particular variables. Additionally, the most common operation for visualization is to request the records of a certain column that fall within a desired range, then retrieve corresponding values from these records in the master table for other variables of interest. This becomes a much quicker operation given a sorted copy of every column from which to retrieve the initial range of values and its corresponding primary key in the master projection table.

Given that each variable must be sorted eventually, partitions were chosen as a performance enhancement to avoid repetitive work and as a form of pre-computation to decrease system lag in visualizations. However, creating the necessary projections duplicates each column in the master table. This is a tradeoff between space usage and running time, but given the efficiency of Viz's compression schemes on epidemiology data, it was hypothesized that the size of the dataset would not grow uncontrollably large relative to its original uncompressed form. Furthermore, creating the master table and the collection of helper projections versus the standard projections created by C-Store requires twice the number of lookups in order to save space (from $n^2$ to 3n, where n is the number of columns in the data set).

## 2.5 Partitioned Projections

In addition to the master projection and the collection of helper projections, Viz implements a set of partitioned projections to optimize further common visualization queries. Partitioned projections are helper projections that are further binned by a second variable (see Figure 2.6 below). Range queries performed on the various variables of interest motivated the creation of helper projections sorted on the variables of interest. In a similar vein, to speed up range queries that check how a given variable varies on the value of another variable (the variable on the slide-bar encircled in red in Figure 2.3 above), two rounds of sorting can be performed: 1) on the slider variable, and 2) on the variable of interest. Thus projections whose index columns are first sorted on the slider variable and then sorted on a variable of interest can be produced, and are called partitioned projections.



| Slider Val | A | Unique ID |
|---|---|---|
| 2003 | 1 | 10 |
| | 1 | 2 |
| | 1 | 9 |
| | 2 | 8 |
| 2004 | 1 | 5 |
| | 1 | 18 |
| | 2 | 13 |
| | 2 | 11 |
| 2005 | 1 | 4 |

Partitioned Projection

**Figure 2.6. Partitioned Projections**
This is an example representation of a partitioned projection where the slider value is 'year.' Note that for each year, the matching rows of A and Unique ID form two small columns that create a "mini-projection" for that year. Additionally, note that within each year partition, the two other columns are sorted on the variable column (column 'A' in this case).

To create a partitioned projection on variable A partitioned on slider variable B:

1. Initialize a number of bins corresponding to number of distinct values that B has.
2. Scan through the helper projection of A (HPA), looking up the corresponding value of variable B in the master table for each entry in HPA
3. Place the entry from HPA in the appropriate bin in the new partitioned projection.
4. Concatenate the bins into one projection

The resulting projection is sorted first on B and then on A. Since this is an integer sort, it requires *O(n)* time.

## *2.6 Usage-Driven Creation of Partitioned Projections and Buffer Pool Memory Management*

Given a mechanism for creating partitioned projections, the key to performance enhancement is an intelligent method for deciding which partitioned projections to create. In a table with *n* columns, there would be $O(n^2)$ possible partitioned projections. Furthermore, some datasets are too large to fit all projections (helper and partitioned) in memory. As a solution, Viz bases the creation of partitioned projections on usage statistics.

Each time a variable, denoted here as A, is used in a visualization or statistical algorithm, its ideal projection type is recorded. For example, given a visualization using slider variable B, Viz increments a count for a projection of A partitioned on B. Each projection can be specified by a pair `<slider variable, variable>`. For this purpose, helper projections are considered to be partitioned on `null`.

### 2.6.1 Usage Statistics

Usage patterns show that about four to five variables are typically used as slider variables within a dataset that has on the order of 100 variables. This is due to the fact that only a handful of variables can be intuitively 'scrolled through' on the slider. Thus, one criterion for creating a partitioned projection is that the variable by which the projection is partitioned be used as the slider variable more than 1/5 of the time. Another criterion is that the variable whose values make up the index column of this partitioned projection be frequently used. Assuming a uniform distribution of variable usage, the expected frequency of usage would simply be $1/n$. Viz defines 'frequently used' as more than twice the expected frequency, or $2/n$. Thus, Viz creates partitioned projections if the usage count for a combination of a slider variable with a second variable exceeds $2/(5n)$ of the total variable usage. Varying this criterion threshold would trade off one-time CPU costs of creating the partitioned projection versus the additional cost of using regular projections in place of a partitioned one.

Choosing the number of partitions in a partitioned projection is another interesting tradeoff. If the number of partitions increase, then the computation time for searching for the results decreases proportionally; however, storage space on disk increases due to the fact that one large sorted column is broken into many smaller sorted columns. Viz chooses to proportionally increase the number of partitions based on the number of distinct values of the variable used for partitioning.

### 2.6.2 Buffer Pool Management

Usage statistics are also employed by Viz to maintain a buffer pool for memory management and to define an eviction policy. Given the usage statistics, Viz maintains a priority queue of projections, both helper and partitioned, in sorted order. The creation and eviction of partitions according to usage statistics and memory availability is as follows:

1. If database has never been loaded before
    a. Create Master Projection (add to memory)
    b. Create Helper projections (add as many as can fit to memory)
2. If database has been loaded before, master table, helper and partitioned projections, and previous usage statistics (that were previously created) are loaded from disk into the buffer pool based on previous usage frequency.
3. User begins using VisuaLyzer
    a. As usage statistics accumulate, partitioned projections are created and loaded into the buffer pool
        i. If the buffer pool is full, the least used helper or partitioned projection is evicted

Note that Viz's buffer pool also takes into account which variables are used in the current visualization and does not evict those from memory.

## *2.7 Selectivity Estimates*

To further improve Viz's query response time, very simple selectivity estimates are implemented. These are intended to inform Viz of which filters would be advantageous to apply using projections and when to quit and revert to the master projection to find its results. These basic selectivity estimates simply assume an even probability distribution of data values for each variable. Thus, if variable A's range were from 1 to 100, and a filter on variable A were set from 1 to 50, then the selectivity of that filter would be 0.5. Using these selectivity estimates, Viz decides to use projections to calculate the result-set of filters who have a selectivity estimate of < 1/x, where x is the number of filters in the query. Thus, if using projections for all the filters would cost approximately the same number or less lookups than performing a sequential scan through the master table, Viz uses projections. By only applying filters whose selectivity is < 1/x using partitions, Viz generates a sub-set of records from these filters to look-up in the master table, rather than having to perform an entire sequential scan of the master table. If no filter has a selectivity < 1/x, then Viz simply reverts to performing a sequential scan of the master table. Thus, Viz theoretically never performs more work than one sequential scan through the data per query.

## *2.8 Aggregation*

The Viz DBMS is accompanied by a flexible data aggregator. This aggregator allows a user to manipulate raw data in any way that suits his analysis. A database can be aggregated over any set of variables (for example, by city, each week). Furthermore, the user can specify how individual variables being aggregated should be aggregated. A given variable can be aggregated in one of five ways:

1. ***Max Value:*** Store the maximum value for this variable in each given aggregation
2. ***Min Value:*** Store the minimum value for this variable in each given aggregation
3. ***Average Value:*** Store the average value for this variable in each given aggregation
4. ***Threshold Count:*** Store the count of the number of times this variable was greater than a user-defined threshold, in each given aggregation
5. ***Threshold Percent:*** Store the percentage of times this variable was greater than a user-defined threshold, in each given aggregation

This aggregator also allows the user to create new variables that are aggregates of multiple variables in a given database, and to join databases on common variables using a simple JOIN algorithm.

## 2.9 Analysis of Viz Performance

### 2.9.1 Compression

Viz's C-store-based compression schemes were a great success. As shown below in Figure 2.7, storing data using Viz (master table and all helper projections) requires reliably less disk/memory space than simply storing a copy of the uncompressed data. As shown below, the compression scheme resulted in regular compression of the master table (an entire copy of the dataset) to less than ¼ the original data size, allowing for leeway in creating projections.

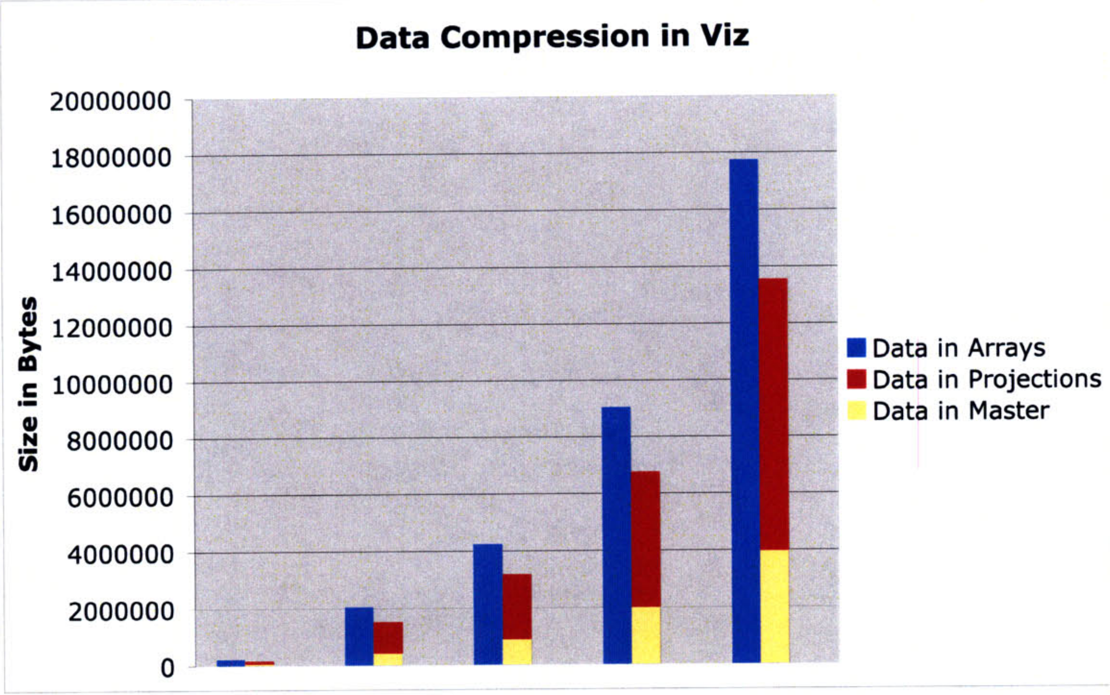


**Figure 2.7 Data Compression Results**
These are measurements of the size of data sets in both standard Java arrays and in Viz after depression. The epidemiology data sets used in this example were ideal for our compression system, as they contained few distinct values in comparison to the number of total values. Creating the master table was reliably below a quarter of the size of the uncompressed equivalent. When the size of additional created projections was included, the total memory used by Viz was consistently still less than storing the data sets uncompressed.

### 2.9.2 VisuaLyzer's Exploratory Data Analysis Algorithms

Viz's specialized column- and projection-based architecture is also successful in generating tangible performance enhancements for VisuaLyzer's mutual information- and correlation ratio-based exploratory data analysis algorithms (see Chapter 3). The usage of projections, which supplies sorted lists of any given variable, proves to be overwhelmingly useful as a means of pre-computation. In a comparison against a VisuaLyzer's implementation that uses the mySQL DBMS, using a dataset containing 21 variables for ~36k items, Viz is clearly superior:

| *Sample Algorithm* | *mySQL Runtime(mins)* | *Viz Runtime(mins)* |
|---|---|---|
| Correlation-Ratio Algorithm | 7.72 | 0.32 |
| Mutual Information Algorithm | 12.82 | 0.37 |

**Table 2.1. Performance Comparison of VisuaLyzer Algorithms Using Viz and mySQL**
Two of the major algorithms used to compute correlations between variables in the VisuaLyzer system significantly perform faster using the Viz DBMS over the mySQL DBMS. The correlation-ratio-based algorithm ran approximately 25x faster, while the mutual information-based algorithm ran approximately 35x faster.

The up-front cost of creating projections upon loading a data set results in a significant improvement in algorithmic runtime. This is valuable given the desired user experience on the VisuaLyzer system.

### 2.9.3 Visualizations: Start-up Delay

A noticeable performance gain to the VisuaLyzer visualization environment is that using the mySQL DBMS, creating each visualization caused a few second delay (proportional to the size of the dataset; = 3.4s for database of 32.1MB) in order to create a table with the relevant portions of the data for that particular visualization. Creating visualization using the Viz DBMS is instantaneous, as all projections and the relevant partitioned projections are created as an un-front cost when the database is loaded.

### 2.9.4 Visualizations: Frame-Rate Comparison

Viz's performance depends on the size of database used. The maximum frame-rate (frames per second = queries per second) attained by a VisuaLyzer visualization is useful as a benchmark for testing the speed of the underlying database management system. As shown in Figure 2.8 below, the Viz DBMS outperforms the mySQL DBMS for databases of all sizes. Furthermore, when tested over a long period of time (loading the same four variables on and off five times with the same slider variable), Viz begins intelligently creating partitioned projections and further outperforms mySQL. As a point of interest, frame-rates of approximately greater than ~20 frames per second appears completely smooth to the user and are therefore desirable.

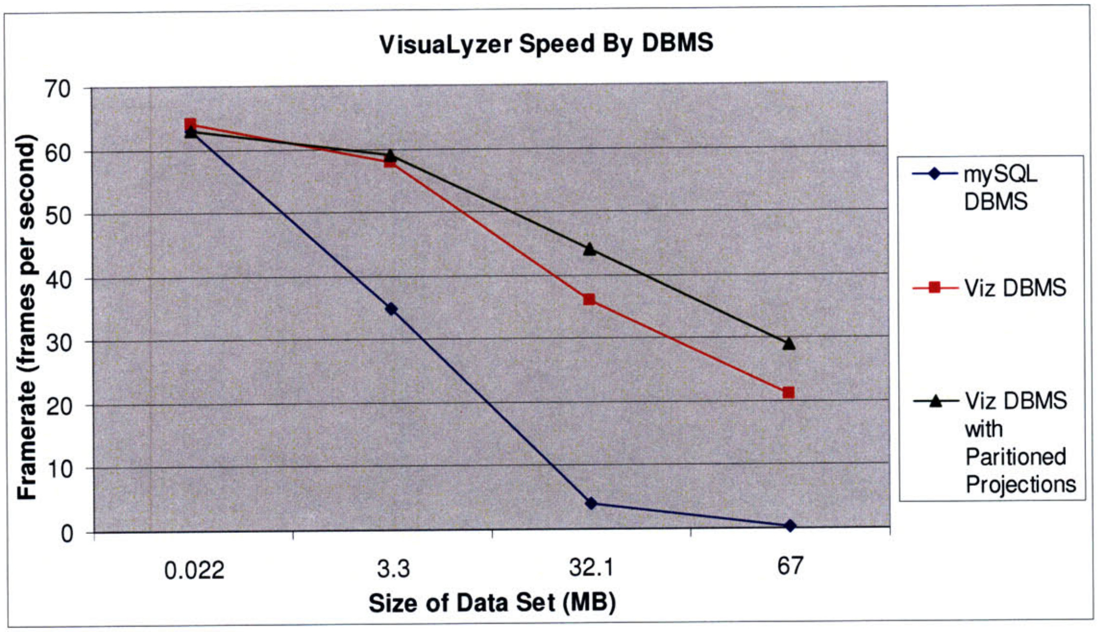


**Figure 2.8 VisuaLyzer Performance Using Various DBMSs**
This graph shows Viz clearly producing better VisuaLyzer performance than the mySQL DBMS for databases of all sizes (in terms of number of frames displayed per second, which is limited by the number of queries the system can make per second). Furthermore, once enough usage statistics accrue for Viz to intelligently create and load partitioned projections, its performance increases further.

### 2.9.5 Visualizations: Sample Query Suite

As a more direct measure of DBMS performance, the Viz DBMS and the mySQL DBMS are compared using a suite of five queries of the format showed in Figure 2.4 above, each with nine `WHERE` clauses. The performance profile closely mimics that shown in Figure 2.8 above, with Viz outperforming mySQL. The average query response times for the test suite on a ~67MB dataset were 3.1 seconds, 0.05 seconds, and 0.03 seconds for mySQL, Viz, and Viz with partitioned projections, respectively.

### 2.9.6 Visualizations: Selectivity Estimates

Even given the simplicity of Viz's selectivity estimates, they appear promising. As shown by Figure 2.9 below, on a ~67MB database, the query strategy that employed selectivity estimates outperforms all three of the above described DBMS implementations, averaging 0.02 seconds on the query test suite (approximately 44 frames per second).

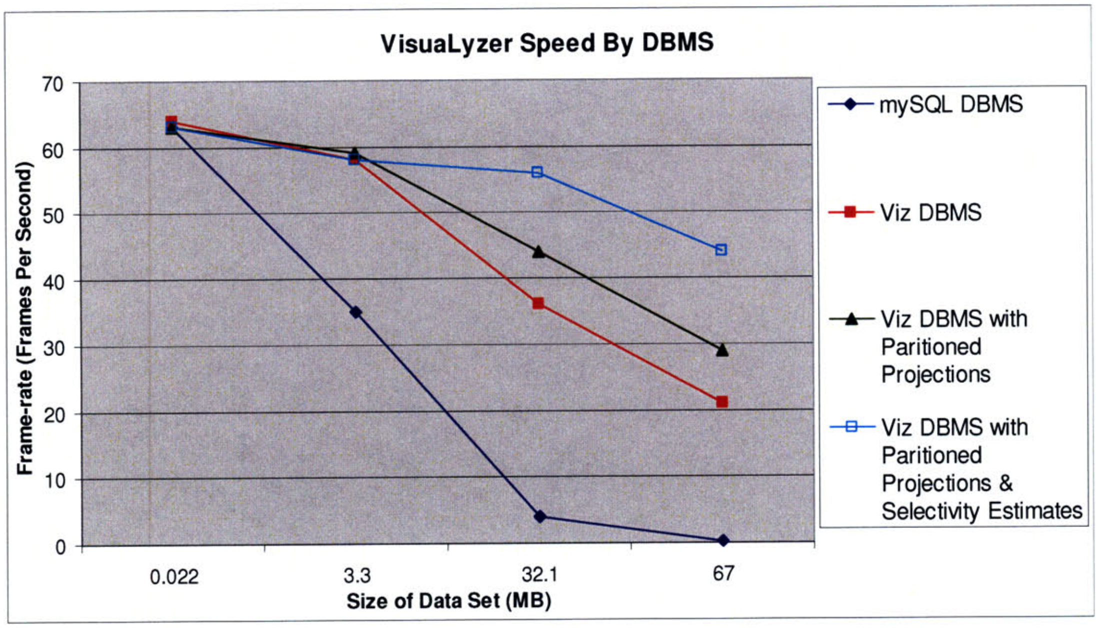


**Figure 2.9 VisuaLyzer Performance Using Various DBMSs**
This graph shows a further performance increase conferred by applying selectivity estimates using the Viz DBMS. With partitioned projections and selectivity estimates, Viz outperforms mySQL for databases of all sizes (that were tested). In the current implementation, selectivity estimates simply assume an even probability distribution of data values for each variable.

## *2.10 Potential Improvements to Viz*

By most metrics, the design of the Viz DBMS is a success. Viz system performance deteriorates slightly for very large datasets, but this deterioration is very slow, and is seen in other comparable DBMSs. Viz's column- and projection- based architecture is extremely successful for VisuaLyzer's exploratory statistical algorithm suite as well as for generating fast results for visualizations of large datasets. While partitioned projections and basic filter selectivity estimates proved to be significant performance enhancers, there is still room for system optimization. For example, creating more accurate selectivity estimates by attempting to learn the true probability distribution for values of every variable might confer further performance increases. This could be implemented using statistical counts similar to those used for projection generation to sample variable distributions. Alternatively, system down-time could be used to "buffer/pre-compute" results for likely future system queries. Designing and implementing Viz was a practical challenge that could serve as a launching point for designing a specialized DBMS for analyzing and visualizing large epidemiologic datasets.

# Chapter 3: Exploratory Data Analysis

To aid in the choice of variables for visualization and analysis, VisuaLyzer uses a suite of analytical and statistical approaches for detecting groups of correlated variables that may be of interest. The VisuaLyzer analytic environment contains implementations of several common algorithms for the purpose of identifying relationships between variables, including k-means clustering, self-organizing maps, regression analysis, principal components analysis, and the ability to calculate standard statistical quantities such as Pearson correlation coefficients.

However, this chapter focuses on a novel family of measures, based on concepts from both statistics and information theory, which efficiently and agnostically quantify the "relatedness" (association) between pairs of variables. Because these methods are grounded in concepts from information theory, they do not need to assume a certain model or regression in order to quantify relatedness between variables. As a result, for instance, a strong sinusoidal relationship, a strong quadratic relationship, and a strong exponential relationship between two variables will all receive high scores when evaluated by these measures. Because of this "agnostic" property, the algorithms involved are quite powerful for automatically pulling out the most striking relationships in a dataset without the need for any information about the mechanisms through which the variables might be related. This property, coupled with the fact that these methods are extremely fast, allows VisuaLyzer to run the algorithms involved on all possible pairs of a set of potentially thousands of variables in order to find, for example, the top ten most related pairs. This chapter describes two novel algorithms that incorporate this family of measures: the 2-dimensional correlation ratio-based algorithm (CR2), and the 3-dimensional mutual information-based algorithm (MI3).

## 3.1 An Introduction to Correlation, Correlation Ratio, and Mutual Information

### 3.1.1 Pearson Correlation Coefficient

Oftentimes, it is useful to be able to identify the relationships (deviations from independence) between multiple variables. A standard method for measuring the strength of such a relationship between two random variables *X* and *Y* with expected values $\mu_x$ and $\mu_y$ and standard deviations $\sigma_x$ and $\sigma_y$ is to calculate the Pearson correlation coefficient ρ:

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X \sigma_Y}, \qquad \textit{Eq. 3.1}$$

where *cov(X, Y)* is the covariance between *X* and *Y*, and *E* is the expected value operator. Pearson correlation coefficients can be used to measure the strength and direction of a *linear* relationship between two variables, but in many cases relationships between variables of interest are non-linear (see Figure 3.1).

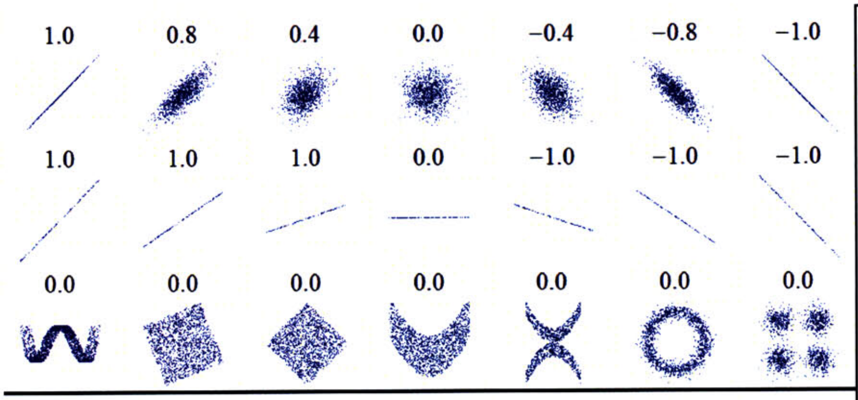


**Figure 3.1.** Several sets of (x, y) points, with the correlation coefficient of x and y for each set. Note that the correlation reflects the noisiness and direction of a linear relationship (top row), but not the slope of that relationship (middle), nor many aspects of nonlinear relationships (bottom). N.B.: the figure in the center has a slope of 0 but in that case the correlation coefficient is undefined because the variance of Y is zero.

*(Imagecreator, 2007)*

Metrics such as correlation ratio statistic and the entropy-based mutual information can be used to detect more general dependencies in data (nonlinear as well as linear relationships).

### 3.1.2 Correlation Ratio

Correlation ratio (CR) is a measure of the relationship between the statistical dispersion within individual categories and the dispersion across an entire sample. This measure can be applied to data that is categorized, such as test scores among students in different math classes (each class would correspond to a separate category). For a given observation $y_{xi}$ where $x$ indicates the category (class) to which this observation belongs and $i$ indicates the label (individual student), let $n_x$ be the number of observations in category $x$, and the mean of category $x$ and the mean of the whole population be

$$\overline{y}_x = \frac{\sum_i y_{xi}}{n_x} \quad \text{and} \quad \overline{y} = \frac{\sum_x n_x \overline{y}_x}{\sum_x n_x}, \qquad \text{Eqns 3.2 and 3.3}$$

respectively. For this data, the correlation ration (η) is defined as

$$\eta^2 = \frac{\sum_x n_x(\overline{y}_x - \overline{y})^2}{\sum_{x,i}(y_{xi} - \overline{y})^2} \qquad \text{Eq. 3.4}$$

and can be thought of as the weighted variance of the category means divided by the variance of all samples. The correlation ratio η takes values between 0 (signifying no dispersion among the means of different categories) and 1 (signifying no dispersion within the different categories).

### 3.1.3 Mutual Information

Mutual information (MI) is an information theory metric that measures the mutual dependence of two random variables. The mutual information of two discrete random variables $X$ and $Y$, $I(X; Y)$, is defined as

$$I(X;Y) = \sum_{y\in Y} \sum_{x\in X} p(x,y) \log\left(\frac{p(x,y)}{p_1(x)\,p_2(y)}\right), \qquad \text{Eq. 3.5}$$

where $p(x,y)$ is the joint probability distribution function of $X$ and $Y$ and $p_1(x)$ and $p_2(y)$ are the marginal probability distribution functions of $X$ and $Y$, respectively. Intuitively, MI measures how much knowing one of these variables reduces the uncertainty about the value of the other. For example, if $X$ and $Y$ are independent, then knowing $X$ does not give any information about $Y$ and vice-versa, thus the MI of these two variables would be 0. On the other hand, if $X$ and $Y$ are identical, then knowing one immediately determines the value of the other, thus their MI would be 1.

MI can also be expressed in terms of the entropies of $X$ and $Y$

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y) \\ &= H(X,Y) - H(X|Y) - H(Y|X), \end{aligned} \qquad \text{Eq. 3.6}$$

where *H(X)* and *H(Y)* are the marginal entropies of *X* and *Y*, *H(X|Y)* and *H(Y|X)* are the conditional entropies of *X* and *Y*, and *H(X,Y)* is the joint entropy of *X* and *Y (J. Liu et al., 2001).* If entropy is regarded as a measure of uncertainty about a random variable, then *H(X|Y)* can be thought of as the amount of uncertainty remaining about *X* after *Y* is known, and thus the above equation shows that the MI between *X* and *Y* translates to the amount of uncertainty in *X* which remains after *Y* is known.

## 3.2 Standard Uses of Correlation Ratio and Mutual Information

CR and MI were traditionally intended for analyzing relationships between an ordered variable (which contains an inherent distance metric that captures the closeness between two of its values) and a second variable which is a member of an unordered set. For example, CR or MI can be used to detect the dependence between different math classes (unordered variable) and student test scores (ordered, real-valued, 'finite' variable which represents some sampling of an unknown underlying distribution). However, in many settings such as epidemiology, finance, and biology, we wish to analyze the relationships between multiple *ordered* (finite) variables. For example, in the epidemiological analysis of disease risk factors, applying MI and CR to identify relationships between resistance levels of different strains of a disease (both ordered 'finite' variables) could be tremendously useful.

Thus, in the past decade, several methods have been developed for estimating CR and MI on finite data, all of which are centered around the concept of partitioning the data with respect to one variable (effectively simulating an unordered variable). The most basic technique used to apply CR/MI to a pair of ordered variables is to use fixed width intervals to 'bin' the data along one axis, and then to use the resulting histogram to estimate mutual information (Silverman, 1986). Moreover, instead of the fixed width intervals, the data can be adaptively partitioned based on the distribution of the data (Rapp 1994, Schreiber 2000).

## 3.3 A Novel Application of Correlation Ratio and Mutual Information to Identifying Dependencies among Finite, Real-Valued Data

This thesis introduces a novel method for applying the correlation ratio (CR) and mutual information (MI) statistics toward identifying dependencies among real-valued variables that represent a finite sampling from an unknown distribution. Instead of searching for an optimal number of partitions in order to apply CR/MI to a given finite variable, this new approach sequentially applies multiple numbers of partitions to the data, analyzing the resulting characteristic curve of the CR/MI score as a function of the number of partitions.

Two novel algorithms that incorporate that incorporate successive binning to attempt to deduce dependencies based on the CR/MI score as a function of number of bins are included in the VisuaLyzer analysis environment. The first is a 2-dimenssional correlation ratio-based algorithm, CR2, and the second is a 3-dimenssional mutual information-based algorithm, MI3, which varies the number of bins for two variables at a time. In designing these algorithms, two key issues are addressed: 1) how to determine partitions sizes and boundaries, and 2) what properties of the graph of CR/MI as a function of the number of bins are most telling of the relationship between two variables.

### 3.3.1 Partitioning (Binning)

Despite the fact that CR2 and MI3 use a spectrum of binnings to infer relationships between variables, they must still use a method to determine how to create each binning. There are three simple methods of splitting up the points $\{(x_i, y_i) : 1 <= i <= n\}$ into $m$ bins:

1. ***Binning by Range:*** Divide the range of the $x_i$ and into $m$ equal pieces of size $(max(\{x_i\}) - min(\{x_i\})/m$ and then let the *k-th* bin be all the data points with $x$ values in the *k-th* piece of the range (see Figure 3.1).
2. ***Binning by Rank-Order:*** Sort the $x_i$ and then let the *k-th* bin be the *k-th* set of $n/m$ points.
3. ***Adaptive Binning by Rank-Order:*** Find values $\{ b_i : 0 <= i <= m\}$ with $b_0 = min(\{x_i\})$ and $b_m = max(\{x_i\})$ such that the *k-th* bin is all the data points with $x$ values falling between $b_{(k-1)}$ and $b_k$, and such that the number of points in each bin is as balanced as possible.

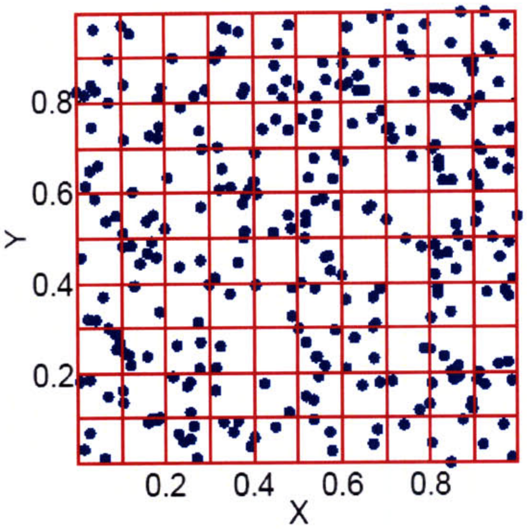


**Figure 3.1. Binning by Range**
This plot of 300 randomly generated points is binning using the Binning by Range approach (in this case, into 10 bins each). This approach simply divides the range of each variable into a given number of equally-sized bins. (R. Steuer et al., 2002)

In weighing the advantages and disadvantages of each of these three binning strategies, the following three datasets will be considered:

1. $D_0 = \{20$ points with x-value 0, and 40 points with x-value $1\}$,
2. $D_1 = \{(0,0), (0.1, 0.1), (1,0.2)\}$; and
3. $D_2 = \{(0,0), (0.1,0.1), (0.2,0.2)\}$

In considering these three examples, it is clear that Binning by Rank-Order produces undesirable results, as it will put data points with the same x-values in different bins. For example, when binning into two bins, it would be desirable to split $D_0$ into its natural two categories, and furthermore, it is not good to have any bins that contain points from the x=0 set and the x=1 in the same bin. Binning by Rank-Order fails on both counts. It also performs poorly when comparing the two data sets $D_1$ and $D_2$, because it treats them identically even though one is clearly more correlated than the other. The only advantage of this method is that the correlation ratio is 1 when the number of bins used is equal to the number of points.

Binning by Range most obviously corresponds to the spatially-intuitive way to bin a dataset. Furthermore, it treats $D_1$ and $D_2$ differently: it suggests that one is more correlated than the other. This means that it is not oblivious to the horizontal distribution of the data in the way that the Binning by Rank-Order is. It also never splits up D_0 in an undesirable way. The primary disadvantage of this method is that it absolutely ignores the distribution of the data, meaning that some bins might have one point while others have fifty, which might not be good because those are not things that should be compared, statistically speaking. This is because if some bins are allowed to contain one point while other are allowed to contain 50 points, single outliers can dominate a given bin if they are allowed to be the only point in that bin. Finally, using this method, some of the bins may be empty. This is fine; they are just ignored, which is natural given the statistics being used.

Finally, Adaptive Binning by Range also corresponds to a spatially-intuitive way to split up a dataset in the sense that no two points with the same x-value will ever lie in different bins. Also, the number of points in each bin will be balanced, which is good in a statistical sense because outliers are not allowed to disproportionately affect the results.

For these reasons, both CR2 and MI3 use Adaptive Binning by Range. MI3 employs a further improvement on binning, which is discussed in Section 3.5.1.

### 3.3.2 Using a Sequence of Binnings (Partitions) to Infer Relationships

Instead of using just one binning, as is the current standard, CR2 and MI3 utilize a spectrum of binnings (i.e. 1 bin, 2 bins, 3 bins, ..., [number of distinct values of a variable] bins).

The CR2 algorithm produces a graph of the number of bins on the x-axis and the corresponding CR score on the y-axis, which presents several interesting characteristics that can be used to judge the strength of a relationship between two variables. This graph is termed the characteristic curve produced from CR2. Examples of the characteristic curves for a randomly generated cloud and a perfect linear relationship are shown in Figure 3.2 B and Figure 3.3 B, respectively. For the randomly generated cloud, the CR increases approximately linearly from 0 to 1 with increasing bin numbers, whereas for a highly [linearly] correlated pair of variables, it appears to approach 1 exponentially.

The MI3 algorithm resembles the CR2 algorithm in many ways; however, it utilizes a third dimension. Using the MI3 algorithm, the graph produced by plotting the number of bins of one variable on the x-axis, the number of bins of a second variable on the y-axis, and the corresponding MI score for that combination of bins on the z-axis presents similarly interesting characteristics, which can be used to judge the strength of a relationship between two variables. This graph is termed the characteristic manifold produced from MI3. Examples of the characteristic manifolds for a randomly generated cloud and a perfect linear relationship are shown in Figure 3.2 C and Figure 3.3 C, respectively. For the randomly generated cloud, this manifold fairly steadily increases toward a maximum value with increasing bin numbers, whereas for a highly [linearly] correlated pair of variables, it appears to approach 1 instantly and to proceed to flatten thereafter.

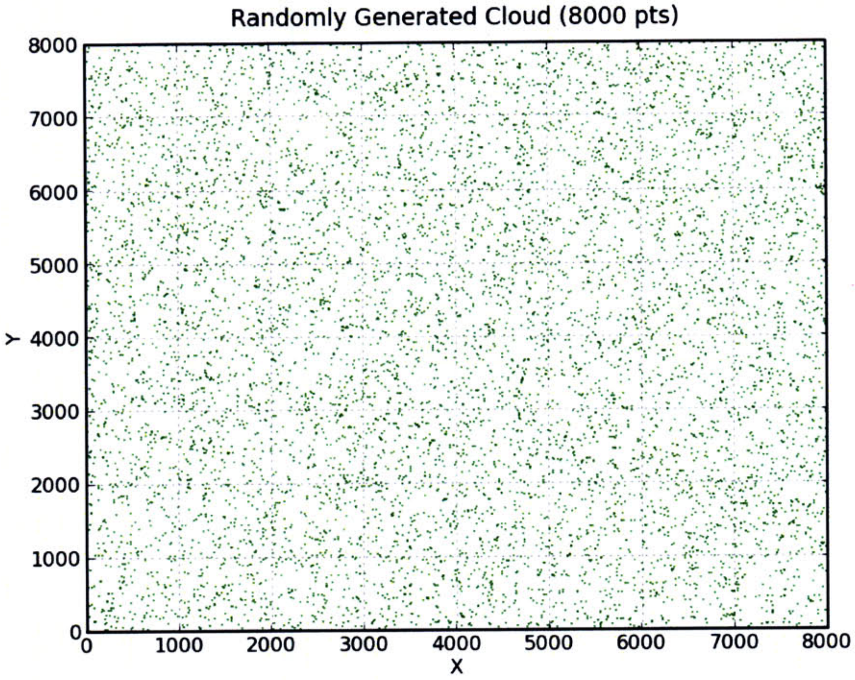


**Figure 3.2 A. A Randomly Generated Cloud of 8000 Points**

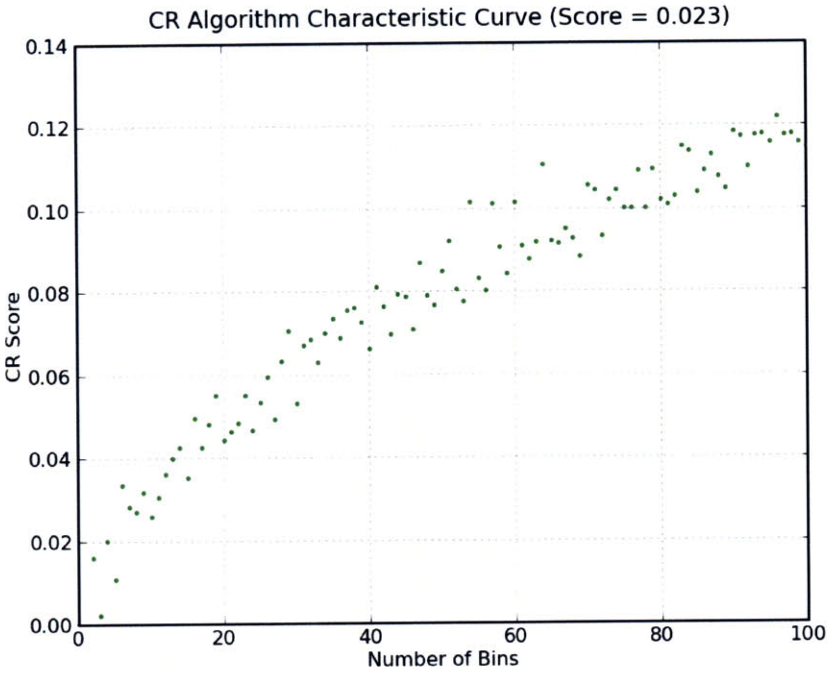


**Figure 3.2 B. Characteristic Curve Generated by CR2 for a Random Cloud**
This is the characteristic curve generated by CR2 for the random cloud shown in Figure 3.2A. This particular run of CR2 was allowed to proceed through 100 binnings (1 bin, 2 bins, …, 100 bins). Note the slow increase in CR score as the number of bins increases.

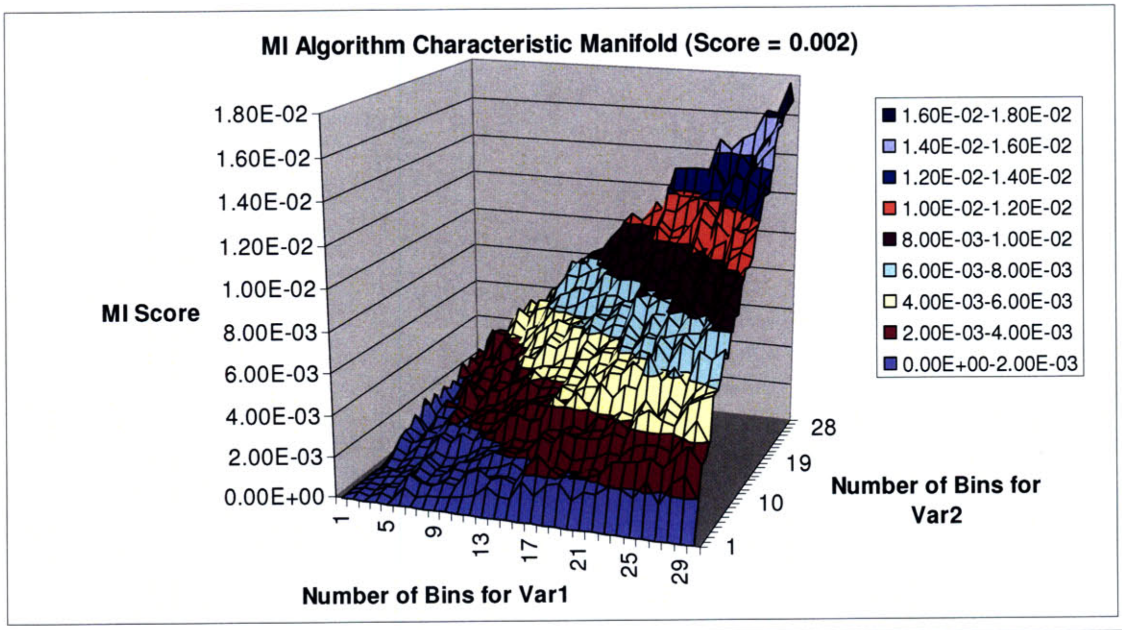


**Figure 3.2 C. Characteristic Manifold Generated by MI3 for a Random Cloud**
This is the characteristic manifold generated by MI3 for the random cloud shown in Figure 3.2A. This particular run of MI3 was allowed to proceed through 30 binnings for each variable {(1 bin, 1bin), (1 bin, 2 bins), (2 bins, 2 bins), …, (30 bins, 30 bins)). Note the fairly slow increase in MI score as the number of bins increases.

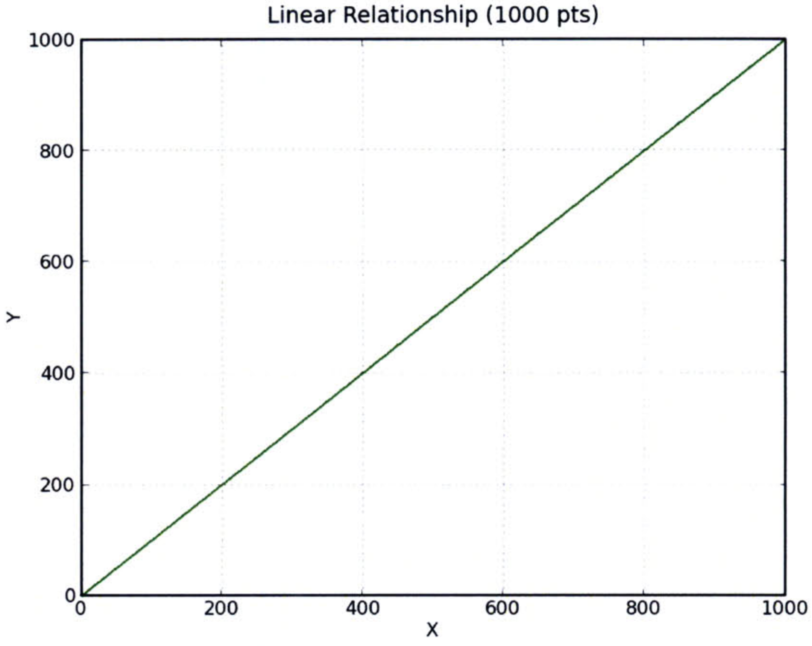


**Figure 3.3 A. A Perfectly Linear Relationship of 1000 Points**

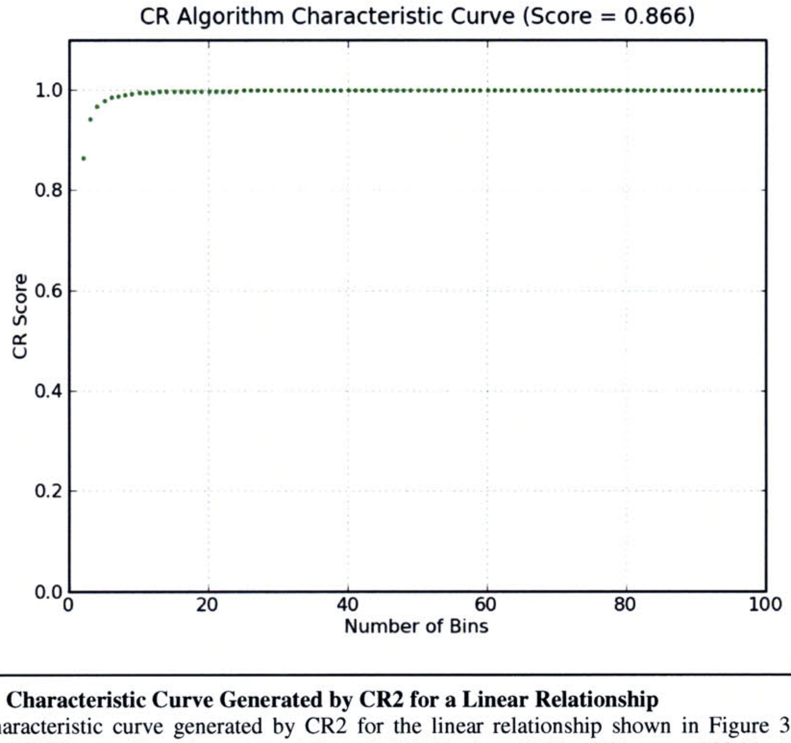


**Figure 3.3 B. Characteristic Curve Generated by CR2 for a Linear Relationship**
This is the characteristic curve generated by CR2 for the linear relationship shown in Figure 3.3A. This particular run of CR2 was allowed to proceed through 100 binnings (1 bin, 2 bins, ..., 100 bins). Note the very rapid initial increase in CR score as the number of bins increases.

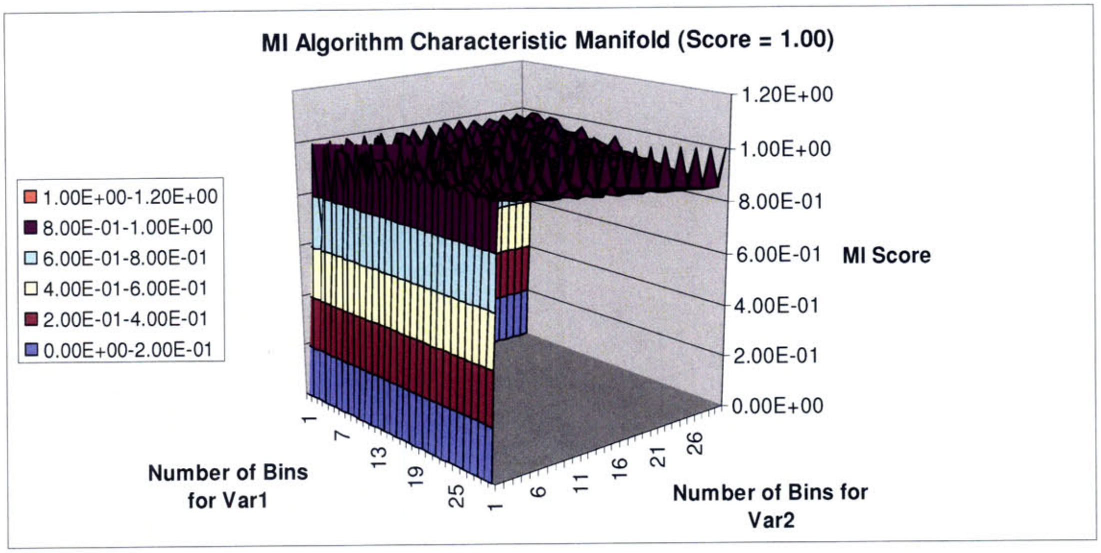


**Figure 3.3 C. Characteristic Manifold Generated by MI3 for a Linear Relationship**
This is the characteristic manifold generated by MI3 for the linear relationship shown in Figure 3.3A. This particular run of MI3 was allowed to proceed through 30 binnings for each variable {(1 bin, 1bin), (1 bin, 2 bins), (2 bins, 2 bins), …, (30 bins, 30 bins)). Note the rapid initial increase in MI score as the number of bins increases.

### 3.3.3 Properties of the CR2 Characteristic Curve and MI3 Characteristic Manifold

There are several interesting properties of the characteristic curves/manifolds produced by CR2 and MI3, such as average value, variance, area/volume under the curve/manifold, and maximum derivative. Furthermore, the latter two identify and assign a relative strength to dependencies between variables, independent of the type of dependency.

The maximum slope of the function of MI/CR score is the most telling characteristic, and works tremendously well across virtually all tested types of relationships. The intuition behind this metric can be explained using the following example. Given a random cloud, the maximum MI/CR score attainable is 1, which is attained when the number of bins approaches the number of points in the cloud. Furthermore, at no point does increasing the number of bins by one gain a significant level of information. In contrast, given a highly correlated pair of variables, a high MI/CR score will be attained with a relatively small number of bins, so removing just one of these bins will result in a large change of score. In a sense, this statistic captures the minimal ratio of the number of bins to the number of data points that is required to describe the relationship between two variables. In other words, if not many bins are required to capture a relationship between two variables, the points that form this relationship must be organized in a very structured manner which can be captured in only a few bins.

A second useful statistic is over-under area/volume. This metric is intended to capture the over-under area/volume between the characteristic curve/manifold of a pair of variables, and the characteristic curve/manifold of a random cloud (that is comprised of the same number of data points). This quantity literally represents the perturbation of the characteristic curve/manifold from that of a random cloud, and intuitively corresponds to the quantity of information contained in the relationship between these two variables. However, while this quantity is finite, its bounds vary between datasets, and there is no convenient normalization that maps it to consistent bounds. Thus, this metric is harder for novice users to interpret, and the maximum slope metric is the official output returned by CR2 and MI3.

## 3.4 The CR2 Algorithm

The CR2 algorithm takes as input a database of $v$ variables, and produces as output a list of $\binom{v}{2}$ triples of the form `<Var1, Var2, CR2 Score>`, where CR2 Score is a new correlation coefficient between 0 and 1. In other words, the algorithm quantifies the absolute strength of every pair of variables in the dataset. An interesting note is that CR is not symmetric (which variable is chosen to be binned affects the resulting CR). Therefore, for each pair of variables the algorithm is run twice, each time binning a different one of the two variables. The algorithm then returns the maximum of the two scores calculated for the pair of variables. The pseudo-code for the CR2 algorithm is presented in Figure 3.4 below. The time complexity of the algorithm is $O(2\cdot \text{maxNumBins}\cdot v^2)$.

```
For all pairs of variables ($v_i$, $v_j$), where $i \neq j$

    For numBins = 1 to maxNumBins
        $C_i$ = CR($v_i$, $v_j$) for numBins bins of $v_i$
        $C_j$ = CR($v_j$, vi) for numBins bins of $v_j$
        CRScores[numBins] = max($C_i$, $C_j$)
```

**Figure 3.4. Pseudo-Code for the CR2 Algorithm**
This is the pseudo-code for CR2. CR(x,y) is a function that calculates the correlation ratio of variables x and y by binning x into numBins bins. For a dataset of $v$ variables CR2 produces as output a list of $\binom{v}{2}$ triples of the form `<Var1, Var2, CR2 Score>`, where CR2 Score is a new correlation coefficient between 0 and 1.

## 3.5 The MI3 Algorithm

The MI3 algorithm takes as input a database of $v$ variables, and produces as output a list of $\binom{v}{2}$ triples of the form `<Var1, Var2, MI3 Score>`, where MI3 Score is a new correlation coefficient between 0 and 1. Similarly to the CR2 algorithm, this algorithm quantifies the absolute strength of every pair of variables in the dataset, but according to the MI statistic. The pseudo-code for the MI3 algorithm is presented in Figure 3.5 below. The time complexity of the algorithm is O(maxNumBins$^2$ $\cdot$v$^2$).

```
For all pairs of variables ($v_i$, $v_j$), where $i \neq j$

    For $numBins_i$ = 1 to maxNumBins
        For $numBins_j$ = 1 to maxNumBins
            $MI_{i,j}$ = MI($v_i$, $v_j$) for
                $numBins_i$ bins of $v_i$ and
                $numBins_j$ bins of $v_j$

            MIScores[$numBins_i$, $numBins_j$] = $MI_{i,j}$
```

**Figure 3.4. Pseudo-Code for the MI3 Algorithm**
This is the pseudo-code for MI3. $MI_{i,j} = MI(v_i, v_j)$is a function that calculates the mutual information of $v_i$ and $v_j$ by binning $v_i$ into $numBins_i$ bins and $v_j$ into $numBins_j$ bins. For a dataset of $v$ variables MI3 produces as output a list of $\binom{v}{2}$ triples of the form <Var1, Var2, MI3 Score>, where MI3 Score is a new correlation coefficient between 0 and 1.

### 3.5.1 An Improvement to MI3: Randomized Binning on One Axis

The above implementation of MI3 produces fantastic results on all classes of functions except aperiodic even functions. This exception is due to the fact that an aperiodic even function has an axis of symmetry. In the case where two bins are used, the bin boundary between these two bins will fall on the axis of symmetry, thus returning a very low mutual information score for this binning configuration. To get around this glitch, MI3 can be improved by binning the first variable using Adaptive Binning by Range, and binning the second variable randomly multiple times. In other words, for a deterministic binning of the first variable, each of the second variable's bin boundaries are decided randomly, $n$ times (where $n$ is O(100)). The MI3 score that is then returned from this configuration of $i$ bins of the first variable and $j$ bins of the second variable is the maximum mutual information of the two variables from all *n* random trials of *i* bins of the first variable (the same every trial) and *j* bins of the second variable (different each trial). This conveniently avoids using an axis of symmetry as a bin boundary, and thus avoids artificially returning a low MI3 score for a given bin configuration. Unfortunately, this increases the runtime of MI3 (by *n*-fold), but the performance increase is worth the cost. The results of this improved MI3 algorithm (MI3+) are demonstrated in the sections below.

## 3.6 A Comparison of CR2 and MI3

In being able to bin both variables, the MI algorithm has access to more information about the relationship between the two variables, as this combinatorial binning creates a potentially asymmetrical manifold, rather than merely a cross-section of that manifold (CR2 output). For this reason, the MI3 algorithm is significantly more accurate than the CR2 algorithm, but also has a longer runtime. Thus, the CR2 algorithm is a more casual, quick-and-dirty glance into the relationships between different variables, while MI3 produces robust results, as demonstrated in Section 3.7 below.

## 3.7 Analysis of Results of CR2 and MI3

CR2 and MI3 are designed to be able to identify and rank the absolute strength of dependencies between variables, regardless of the *type* of relationship between the variables. This ability is afforded by the fundamental mathematical quantities on which they are based. This section is intended to evaluate their ability to rank the absolute strength of a class of different relationships. Section 3.2.2 already demonstrated the algorithms' performance on a random cloud (MI Score = 0.002, CR Score = 0.02) and a linear relationship (MI Score = 1.0, CR Score = 0.87). This section will demonstrate the algorithms' performance, which is summarized in Table 3.1 below, on power, exponential, sinusoidal, and parabolic (aperiodic even) functions. Furthermore, it will explore the power (sensitivity) of the algorithms.

### 3.7.1 CR2, MI3, and MI3+ Performance on a Suite of Relationships

This section will demonstrate the CR2, MI, and MI3+ algorithms' performance on a suite of test relationships, which do not contain noise. Table 3.1 summarizes the results of running the algorithms on several different types of linear and non-linear relationships. CR2 returns a very low score for a random cloud, and fairly high scores for all other relationships. Furthermore, MI3 returns practically perfect scores for a random cloud and all other relationships, with the exception of a half sine wave (due to its axis of symmetry). However, the MI3+ algorithm (MI3 with randomized binning for one variable) outperforms both other algorithms, returning practically perfect values for all functions, including a half sine wave.

***Scores of CR2, MI3, and MI3+ Algorithms for Various Relationships***

| Relationship Type | CR2 Score | MI3 Score | MI3+ Score |
|---|---|---|---|
| *Random Cloud* | 0.02 | 0.002 | 0.004 |
| *Linear (y=x)* | 0.87 | 1.0000 | 1.000 |
| *Power* $(y=x^2)$ | 0.84 | 1.0000 | 1.000 |
| *Exponential* $(y=e^x)$ | 0.85 | 1.0000 | 1.000 |
| *Sinusoidal (y=sin(x))* | 0.90 | 0.999 | 0.999 |
| *Parabolic (y=sin(x)),*<br>*Half Period* | 0.73 | 0.579 | 0.999 |

**Table 3.1. CR2, MI3, and MI3+ Performance on Various Functions**
This table contains the scores returned by the CR2, MI3, and MI3+ (MI with randomized binning for one variable) on various relationships. Note that all of these relationships do not contain noise. For sensitivity analysis, see section 3.7.2. Note that CR2 returns a very low score for a random cloud, and fairly high scores for all other relationships. Furthermore, MI3 returns practically perfect scores for a random cloud and all other relationships, with the exception of a half sine wave (due to its axis of symmetry). However, the MI3+ algorithm outperforms both other algorithms, returning perfect values for all functions, including a half sine wave.

Figures 3.5, 3.6, and 3.7 show the corresponding characteristic curves and manifolds for the power, sinusoidal, and parabolic functions used to generate the data in Table 3.1, respectively.

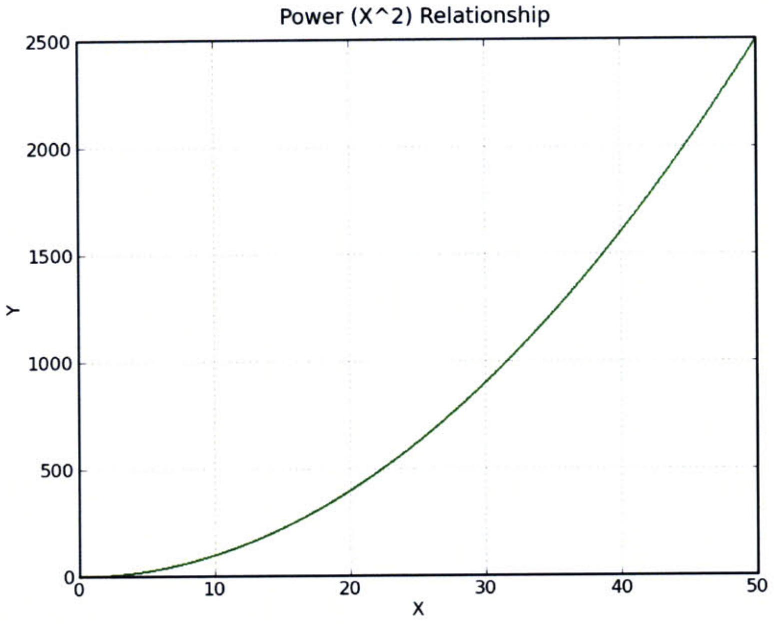


**Figure 3.5 A. A Perfect Power Relationship ($y = x^2$) of 1000 Points**

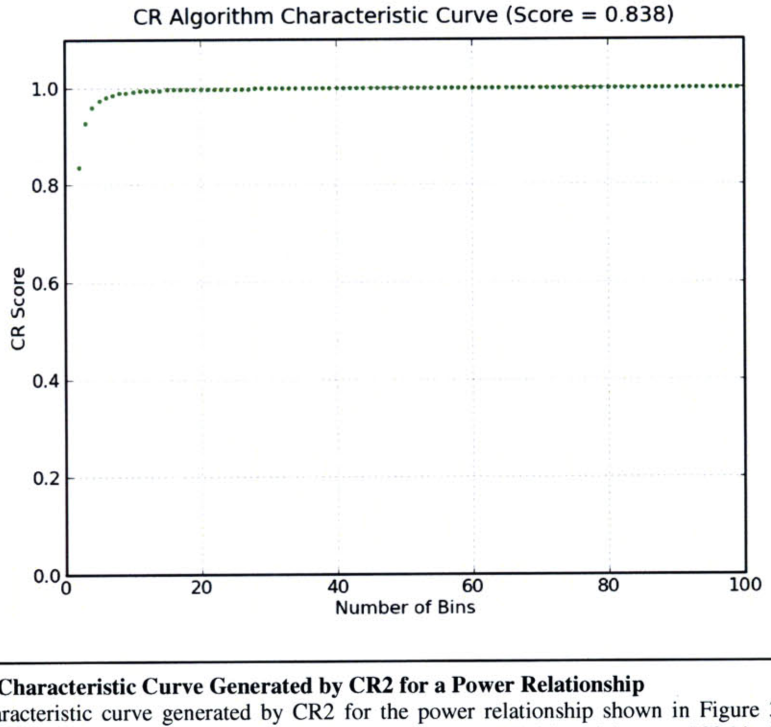


**Figure 3.5 B. Characteristic Curve Generated by CR2 for a Power Relationship**
This is the characteristic curve generated by CR2 for the power relationship shown in Figure 3.5A. This particular run of CR2 was allowed to proceed through 100 binnings (1 bin, 2 bins, ..., 100 bins). Note the fairly rapid initial increase in CR score as the number of bins increases.

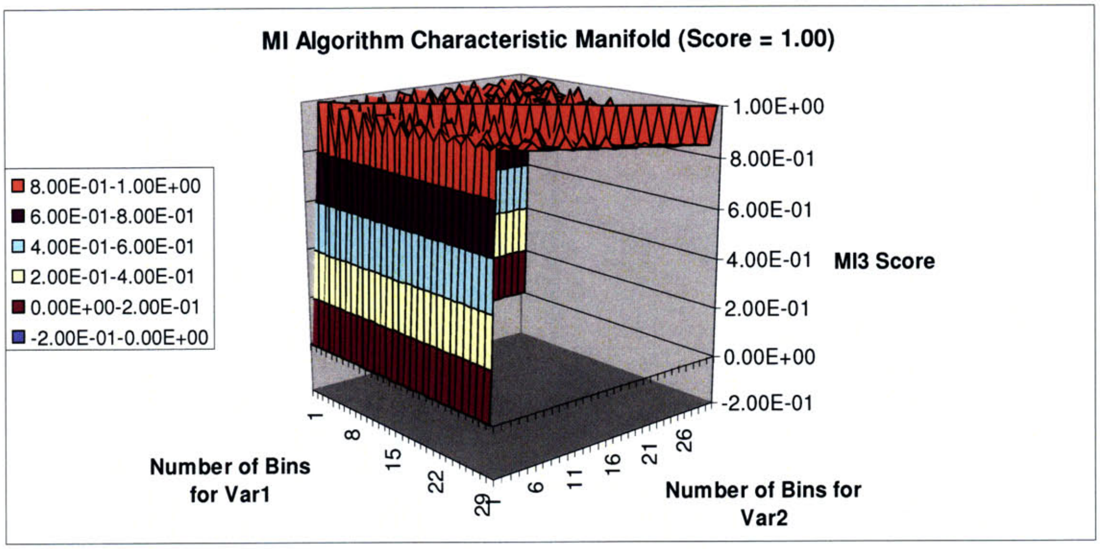


**Figure 3.5 C. Characteristic Manifold Generated by MI3 for a Power Relationship**
This is the characteristic manifold generated by MI3 for the power relationship shown in Figure 3.5A. This particular run of MI3 was allowed to proceed through 30 binnings for each variable {(1 bin, 1bin), (1 bin, 2 bins), (2 bins, 2 bins), …, (30 bins, 30 bins)). Note the rapid initial increase in MI score as the number of bins increases.

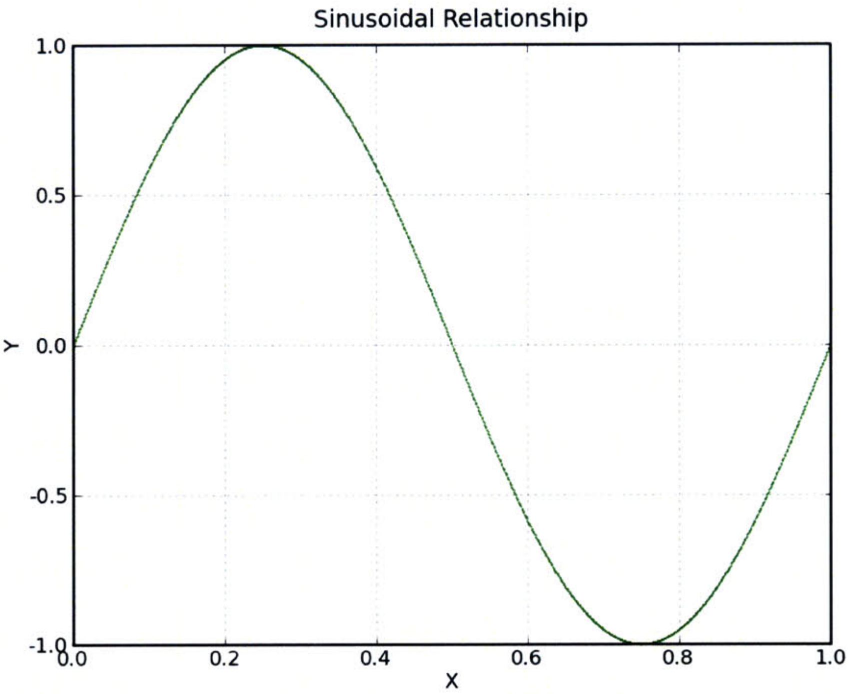


**Figure 3.6 A. A Perfect Sinusoidal Relationship (y = Sin(x)) of 1000 Points**

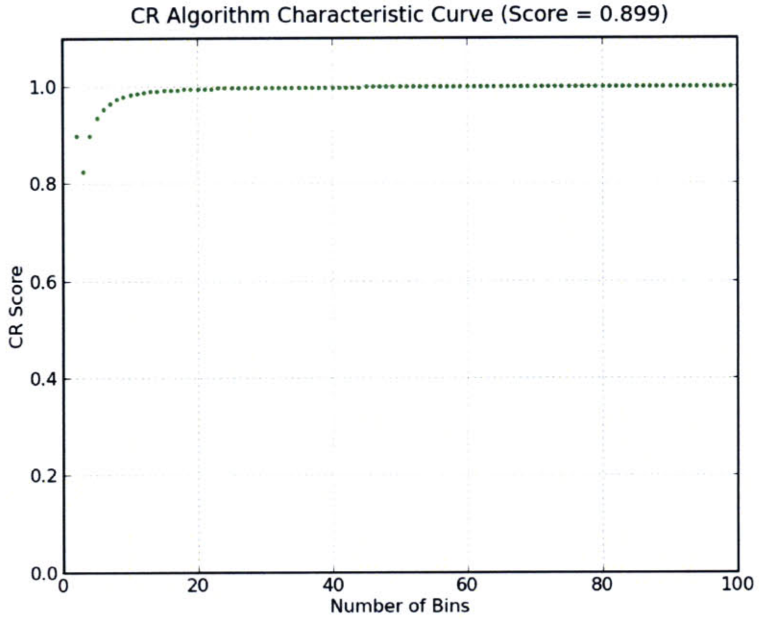


**Figure 3.6 B. Characteristic Curve Generated by CR2 for a Sinuisodial Relationship**
This is the characteristic curve generated by CR2 for the sinusoidal relationship shown in Figure 3.6A. This particular run of CR2 was allowed to proceed through 100 binnings (1 bin, 2 bins, ..., 100 bins). Note the very rapid initial increase in CR score as the number of bins increases.

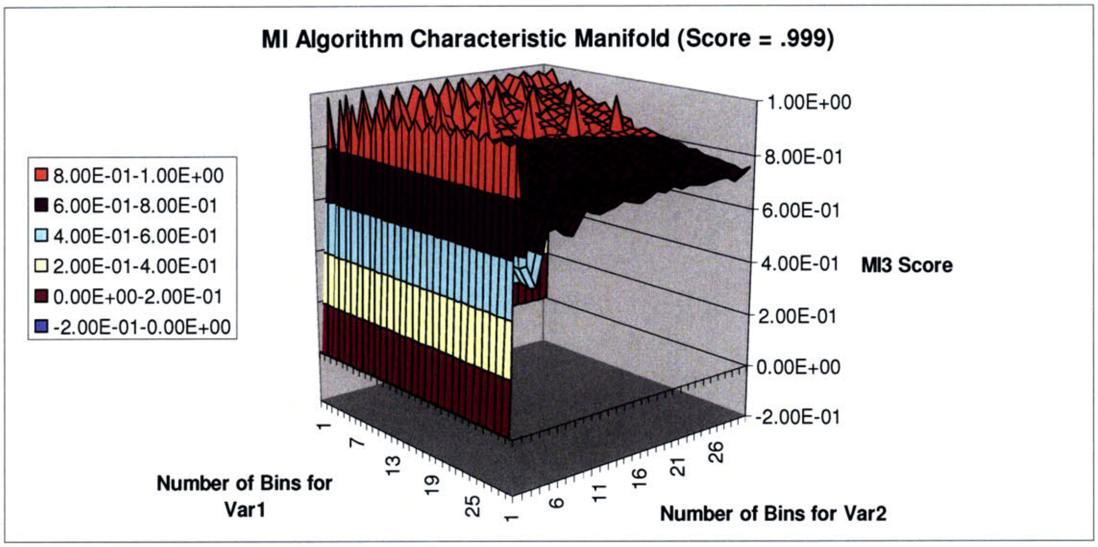


**Figure 3.6 C. Characteristic Manifold Generated by MI3 for a Sinusoidal Relationship**
This is the characteristic manifold generated by MI3 for the sinusoidal relationship shown in Figure 3.6A. This particular run of MI3 was allowed to proceed through 30 binnings for each variable {(1 bin, 1bin), (1 bin, 2 bins), (2 bins, 2 bins), ..., (30 bins, 30 bins)). Note the rapid initial increase in MI score as the number of bins increases.

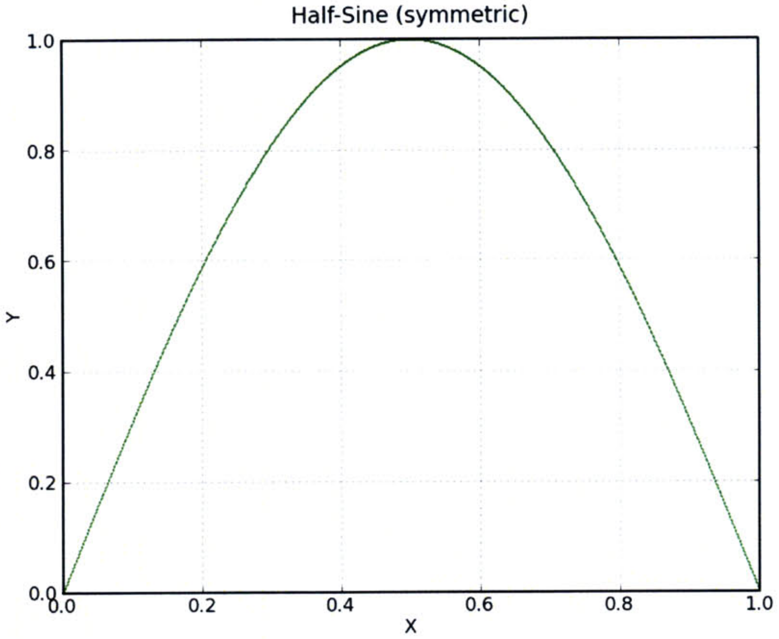


**Figure 3.7 A. A Perfect Half Sine Wave (parabolic) Relationship (y = Sin(x)) of 1000 Points**

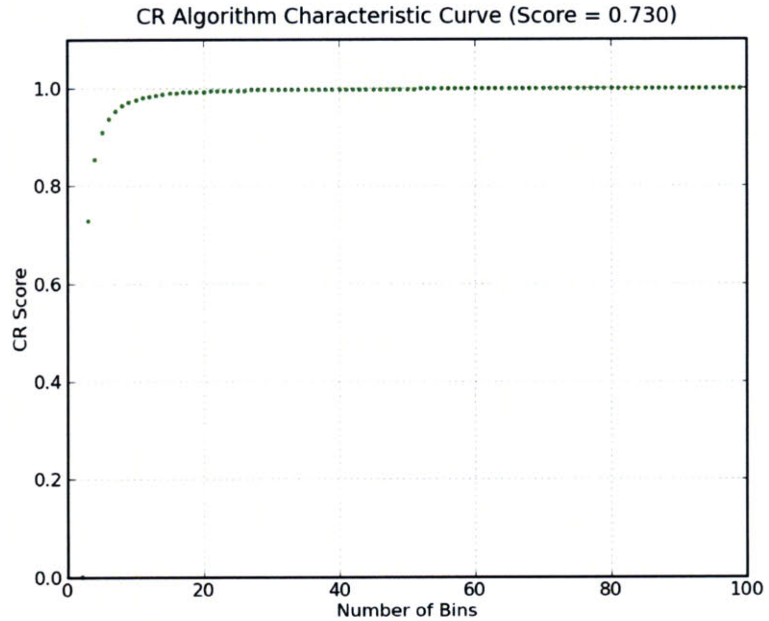


**Figure 3.7 B. Characteristic Curve Generated by CR2 for a Parabolic Relationship**
This is the characteristic curve generated by CR2 for the parabolic relationship shown in Figure 3.7A. This particular run of CR2 was allowed to proceed through 100 binnings (1 bin, 2 bins, ..., 100 bins). Note the rapid initial increase in CR score as the number of bins increases.

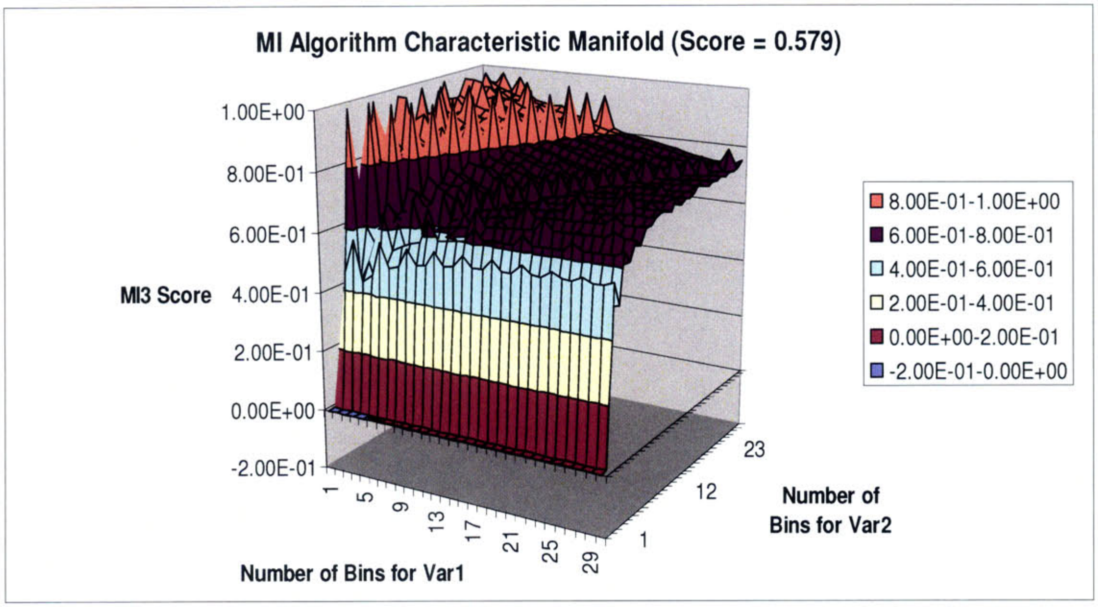


**Figure 3.7 C. Characteristic Manifold Generated by MI3 for a Parabolic Relationship**
This is the characteristic manifold generated by MI3 for the parabolic relationship shown in Figure 3.7A. This particular run of MI3 was allowed to proceed through 30 binnings for each variable {(1 bin, 1bin), (1 bin, 2 bins), (2 bins, 2 bins), …, (30 bins, 30 bins)). Note that the rapid initial increase in MI score takes place over a few different binning configurations, resulting in an unexpectedly low maximum slope, and therefore a low MI3 Score. MI3+ resolves this issue using randomized binning.

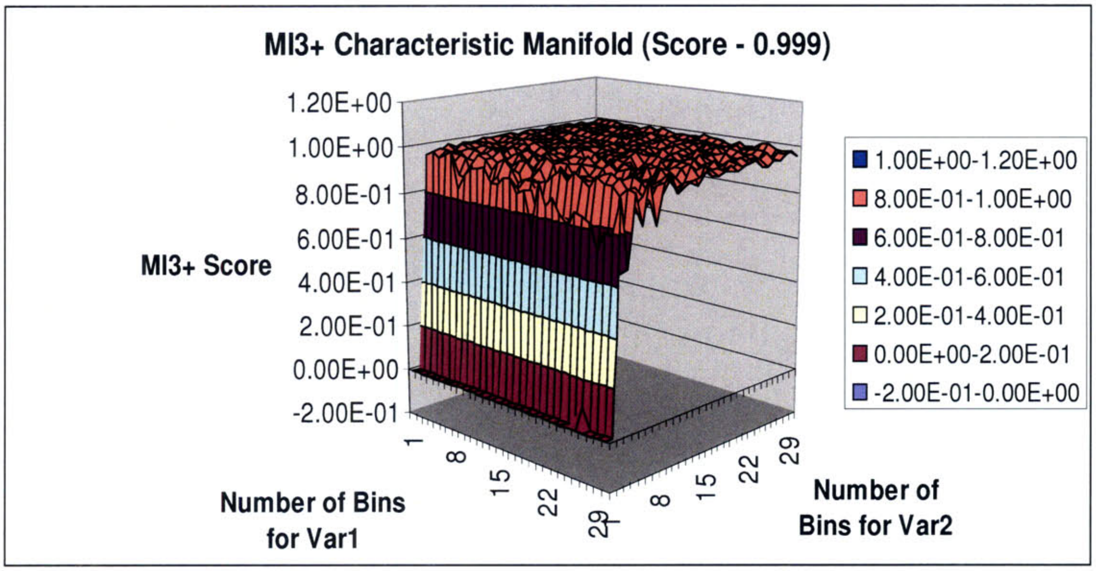
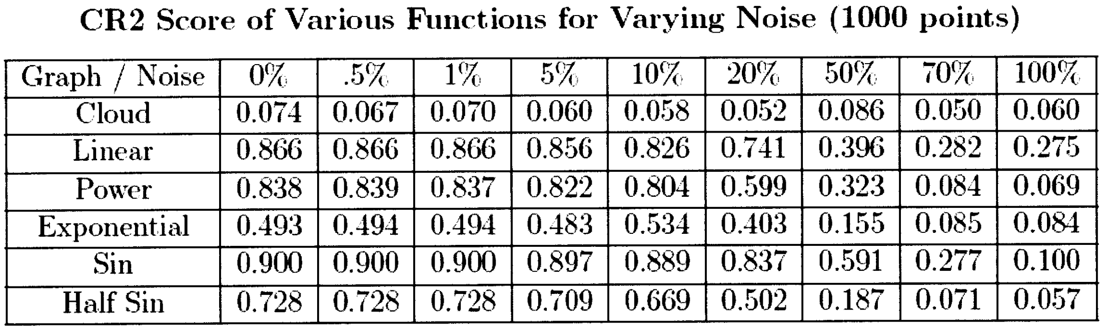


**Figure 3.7 D. Characteristic Manifold Generated by MI3+ for a Parabolic Relationship**
This is the characteristic manifold generated by MI3+ for the parabolic relationship shown in Figure 3.7A. This particular run of MI3+ was allowed to proceed through 30 binnings for each variable {(1 bin, 1bin), (1 bin, 2 bins), (2 bins, 2 bins), …, (30 bins, 30 bins)). Note that given the randomized binning on one variable, the rapid initial increase in MI score takes more rapidly than using the MI3 algorithm, resulting in a more reasonable MI3+ score.

This sample of the diverse class of functions that have been tested using the CR, MI3, and MI3+ algorithms demonstrate the algorithms' ability to detect and measure the absolute strength of virtually all correlations of interest from within a collection of variables.

### 3.7.2 CR2, MI3, and MI3+ Sensitivity to Noise

Most relationships that exist in real-world data are noisy, and therefore an important property of the CR2, MI3, and MI3+ algorithms is how they perform on noisy relationships. Tables 3.2 and 3.3, and Figures 3.8 and 3.9, show CR2 and MI3 scores for various functions that contain varying levels of noise. Naturally, different types of function react differently to noise, so it is expected that the CR2 and MI3 curves for different functions behave differently once noise is introduced.



**CR2 Score of Various Functions for Varying Noise (1000 points)**

| Graph / Noise | 0% | .5% | 1% | 5% | 10% | 20% | 50% | 70% | 100% |
|---|---|---|---|---|---|---|---|---|---|
| Cloud | 0.074 | 0.067 | 0.070 | 0.060 | 0.058 | 0.052 | 0.086 | 0.050 | 0.060 |
| Linear | 0.866 | 0.866 | 0.866 | 0.856 | 0.826 | 0.741 | 0.396 | 0.282 | 0.275 |
| Power | 0.838 | 0.839 | 0.837 | 0.822 | 0.804 | 0.599 | 0.323 | 0.084 | 0.069 |
| Exponential | 0.493 | 0.494 | 0.494 | 0.483 | 0.534 | 0.403 | 0.155 | 0.085 | 0.084 |
| Sin | 0.900 | 0.900 | 0.900 | 0.897 | 0.889 | 0.837 | 0.591 | 0.277 | 0.100 |
| Half Sin | 0.728 | 0.728 | 0.728 | 0.709 | 0.669 | 0.502 | 0.187 | 0.071 | 0.057 |

**Table 3.2. CR2 Scores of Various Functions for Varving Levels of Noise**

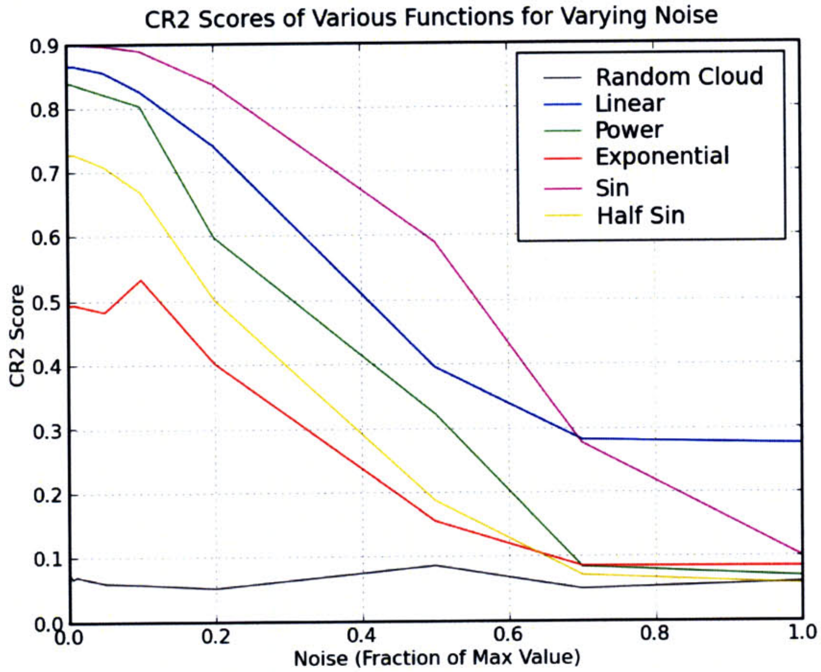


**Figure 3.8. CR2 Scores of Various Functions for Varying Levels of Noise**

**MI3 Score of Various Functions for Varying Noise (1000 points)**

| Graph / Noise | 0% | .5% | 1% | 5% | 10% | 20% | 50% | 70% | 100% |
|---|---|---|---|---|---|---|---|---|---|
| Cloud | 0.020 | 0.017 | 0.016 | 0.019 | 0.023 | 0.020 | 0.020 | 0.017 | 0.022 |
| Linear | 1.000 | 0.962 | 0.919 | 0.722 | 0.557 | 0.421 | 0.064 | 0.037 | 0.027 |
| Power | 1.000 | 0.947 | 0.906 | 0.689 | 0.482 | 0.229 | 0.042 | 0.018 | 0.019 |
| Exponential | 1.000 | 0.649 | 0.482 | 0.159 | 0.100 | 0.038 | 0.020 | 0.017 | 0.018 |
| Sin | 0.979 | 1.000 | 0.979 | 0.894 | 0.776 | 0.591 | 0.222 | 0.035 | 0.018 |
| Half Sin | 0.571 | 0.562 | 0.545 | 0.459 | 0.385 | 0.172 | 0.017 | 0.015 | 0.027 |

**Table 3.3. MI3 Scores of Various Functions for Varving Levels of Noise**

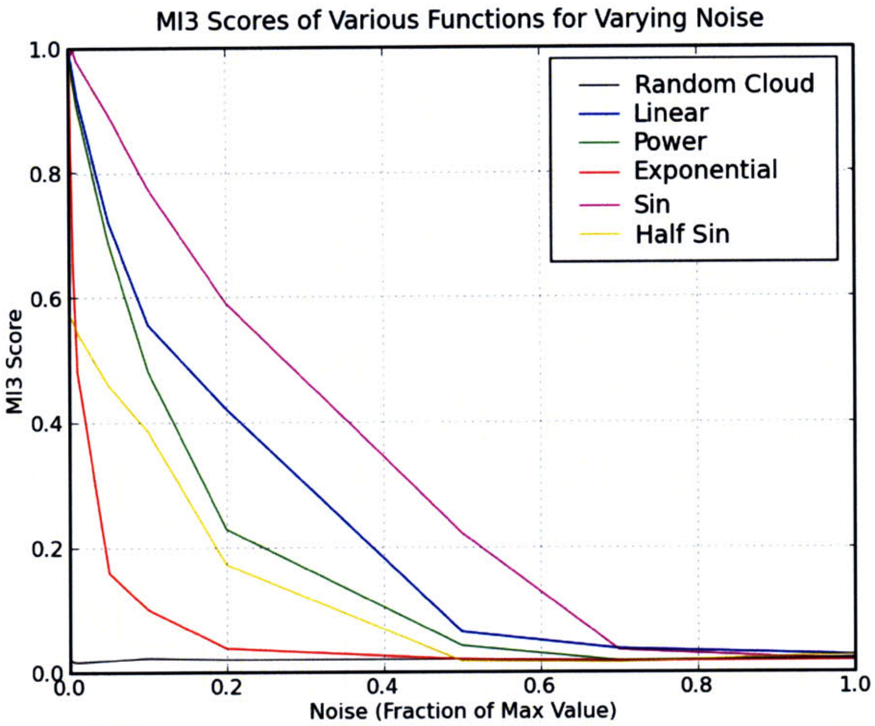


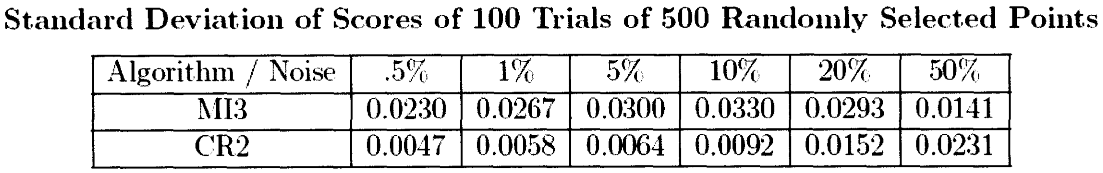
**Figure 3.9. MI3 Scores of Various Functions for Varying Levels of Noise**

A second critical property is how consistent the results of the CR2, MI, and MI3+ algorithms are for similar functions. This can be tested by leave-out cross validation. As an example, a perfect linear relationship containing 1000 points is created, and varying levels of noise are introduced to it. For each level of noise that is introduced, 100 runs of each algorithm are executed, each using a randomly selected sample of 500 of the original 1000 points. The means and standard deviations of each round of 100 trials are presented in Tables 3.4 and 3.5 below. Note that the standard deviations for both algorithms are relatively low, and that they increase slightly with noise as expected (until the functions become so noisy that they start resembling a random cloud, which has a low standard deviation).

## Mean Scores of 100 Trials of 500 Randomly Selected Points

| Algorithm / Noise | .5% | 1% | 5% | 10% | 20% | 50% |
|---|---|---|---|---|---|---|
| MI3 | 0.968 | 0.934 | 0.737 | 0.564 | 0.417 | 0.068 |
| CR2 | 0.866 | 0.866 | 0.856 | 0.826 | 0.738 | 0.396 |

**Table 3.4. Mean CR2 and MI3 Scores of 100 Trials on 500 Randomly Selected Points from 1000 Point Relationships with Varying Levels of Noise**



Standard Deviation of Scores of 100 Trials of 500 Randomly Selected Points

| Algorithm / Noise | .5% | 1% | 5% | 10% | 20% | 50% |
|---|---|---|---|---|---|---|
| MI3 | 0.0230 | 0.0267 | 0.0300 | 0.0330 | 0.0293 | 0.0141 |
| CR2 | 0.0047 | 0.0058 | 0.0064 | 0.0092 | 0.0152 | 0.0231 |

**Table 3.5. Standard Deviations of CR2 and MI3 Scores of 100 Trials on 500 Randomly Selected Points from 1000 Point Relationships with Varying Levels of Noise**

## *3.8 Interactively Selecting Clusters of Associated Variables for Further Analysis (Gravity Graph)*

The CR2, MI3, and MI3+ algorithms produce a clear list of relationships present in a given dataset. However, while each of these relationships is potentially meaningful, it is the interactions amongst groups of variables and relationships that are truly meaningful. To enable an intuitive and efficient interpretation of how these different relationships are related to each other, a novel, specialized, interactive visualization, called a Gravity Graph, is introduced.

### 3.8.1 The Gravity Graph Visualization

A gravity graph is designed to be constructed using the output of the CR2, MI3, or MI3+ algorithms, but can also be constructed from the outputs of regressions or other clustering algorithms. In a Gravity Graph, the variables in the data set are represented by nodes, and the relationships between them are represented by edges. The graph is a dynamic physical equilibrium based on a gravitational model, which is governed by forces that vary in proportion to the strengths of relationships between variables. In other words, every aspect of the physical, gravitational system between nodes is dependent on a parameter from the CR2, MI3, or MI3+ algorithms. Users can intuitively interact with this graph by adding or removing variables or relationships between variables and seeing how the gravitational model reacts to these changes. For example, a user can create a Gravity Graph that contains only a few variables (nodes) and relationships (edges), and can progressively add more variables or relationships into the graph, to see how the physical equilibrium shifts. Relationships can be added in random order, in order by descending relationship strength, or between a specific variable of interest and other variables. An example of the construction of a Gravity Graph for an influenza dataset provided by the CDC is shown in Figure 3.10.

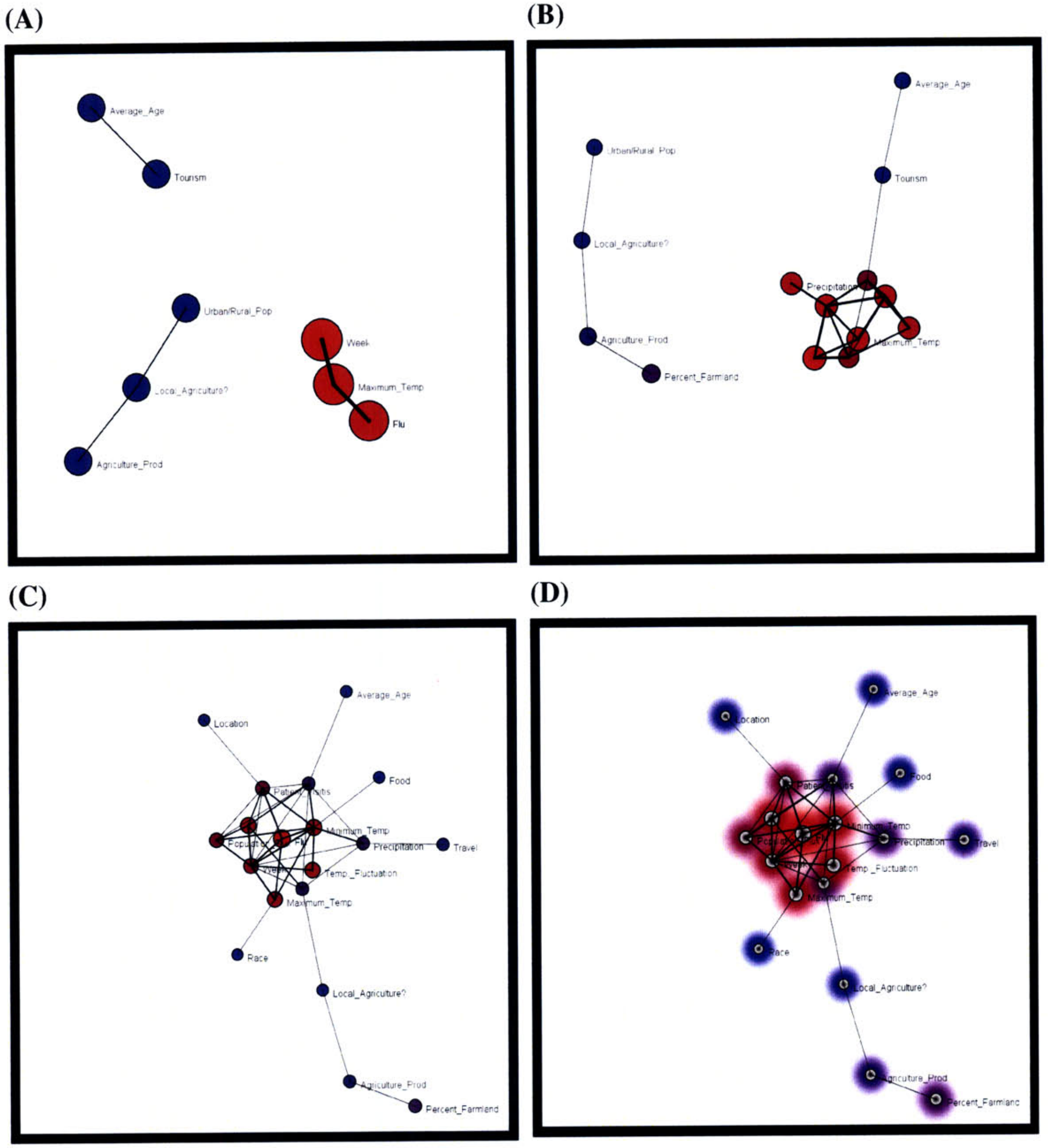


**Figure 3.10. The Construction of a Gravity Graph (US Influenza Data)**
There are several 'screen shots' of Gravity Graph that is constructed from US (state-level) influenza data provided by the CDC. The gravity graph is intended to be an interactive visualization of the relationships between variables. Thus, the user can begin by graphing only a few variables (nodes) and the relationships between them (edges), show in (A). This graph represents a physical equilibrium that is defined by the relationships between the variables being studied. Thus, despite the fact that these screen shots are still, these equilibria shift each time the user adds a new variable or relationship of interest to the graph, and the nodes appear to float and jostle into position until they settle in a 'low energy equilibrium.' (B) and (C) show how the graph develops as the user continues to add variables. Finally, (D) shows the final graph from (C) with a 'gravitational glow,' which is intended to help the user easily identify clusters of highly interrelated variables. In this case, note the red/dark purple (highly correlated) cluster of variables that cluster around the "Influenza Prevalence' node. The variables in this cluster include time, maximum and minimum temperatures, patient visits in hospitals, population flux (travel in and out of a region), all of which help determine the influenza epidemic dynamics.

The following parameters of a Gravity Graph are determined by algorithmic output:

1. ***Gravitational Fields:*** The gravitational field of any node is proportional to the average CR/MI correlation ratio between it and any other nodes (variables) to which it is connected at a given time.
2. ***Edge Strengths:*** The length and thickness of a line (edge) that connects any two nodes is proportional to the relationship between the two corresponding variables. Thus, variables that are very related (have a high CR/MI correlation coefficient) lie very close to each other and are connected by thick edges while weakly related variables end up far away from each other and connected by weaker looking edges.
3. ***Edge Viscosity:*** Edges are modeled by ideal springs. The spring constant of an edge is proportional to the variance in the corresponding correlation. Note, this parameter is only set when the strength of the relationship between two variables can be assessed in multiple different scenarios (for example, for each year in a dataset).
4. ***Coulombic Repulsions:*** Coulombic repulsions exist between every pair of nodes that are not connected by an edge. This is to ensure that variables that are related (have a high CR/MI correlation coefficient) cluster while unrelated variables do not associate with each other.
5. ***Gravitational Glow:*** If the user activates the gravitational glow mode, each node's gravitational field is depicted as a colorful gradient. The radius and color of this circular gradient is proportional to the node's gravitational field and to the number of other nodes to which it is connected. Red gravitational glows represent nodes that are on average highly correlated with their neighboring nodes, while blue gravitational fields represent nodes that are on average not highly correlated with their neighboring nodes. A gravitational field that is a shade of purple indicates a node that is highly correlated to some of its neighbors, and not highly correlated to others. This gravitational glow redundantly highlights groups of highly interrelated variables, allowing the user to select highly correlated 'red pockets'.

The Gravity Graph makes groups of interrelated variables immediately obvious and therefore easy for the investigator to pick out of a huge initial list of variables and relationships. Furthermore, interesting clusters of related variables can be selected and the most relevant dimensions within these clusters can be identified for analysis. Together with the CR2, MI3, and MI3+ algorithms, the Gravity Graph represents a fundamental step of the VisuaLyzer approach, which aims to intelligently reduce the amount of dimensions in the dataset, making essential relationships abundantly clear.

### 3.8.2 Gravity Graph Examples

Figure 3.11 below shows a larger view of a selected subgraph of the Gravity Graph constructed from the output of MI3+ on influenza data. The nodes have jostled into a 'low energy equilibrium,' meaning that highly correlated variables (red) try to move closer to each other, while uncorrelated variables are repelled. Furthermore, this graph is shown in Gravitational Glow mode, in which each node's gravitational field is depicted as a colorful gradient. This is intended to help the user easily identify clusters of highly interrelated variables. In this case, note the red/dark purple (highly correlated) cluster of variables that cluster around the "Influenza Prevalence' node. The variables in this cluster include time, maximum and minimum temperatures, patient visits in hospitals, population flux (travel in and out of a region), all of which help determine the influenza epidemic dynamics.

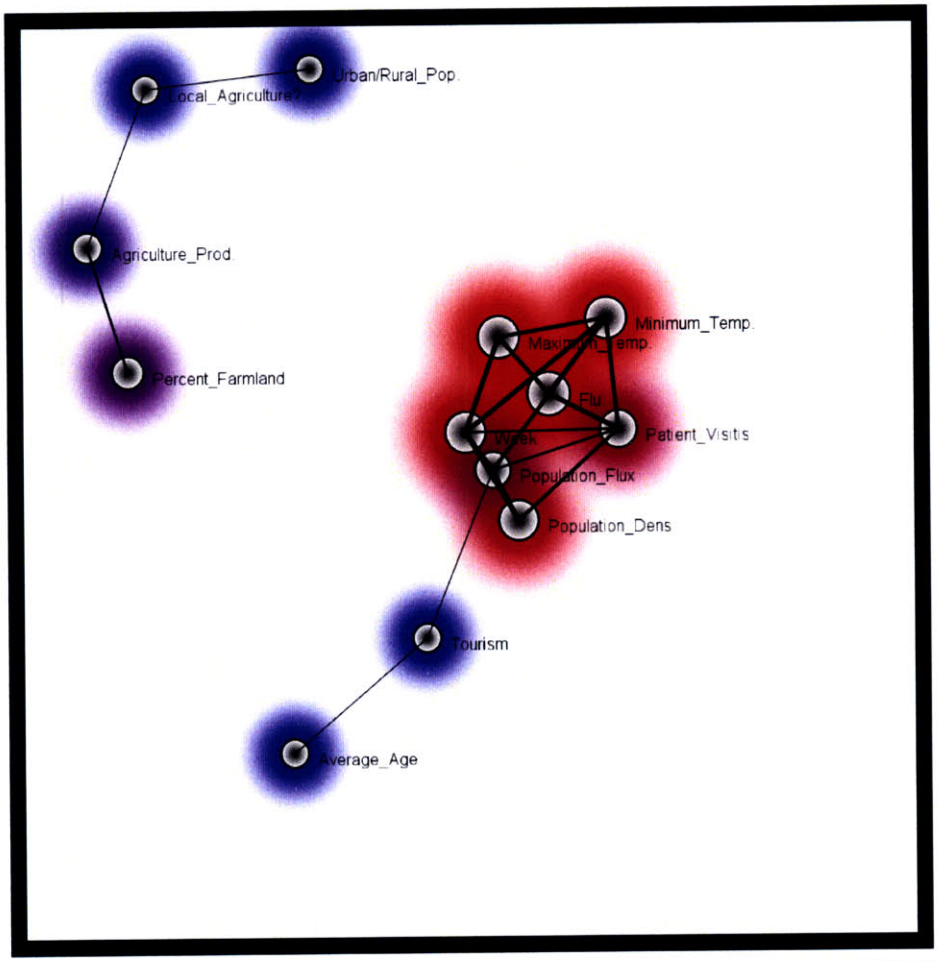


**Figure 3.11. An Influenza Gravity Graph (US Influenza Data)**
There is a screenshot of Gravity Graph that is constructed from US (state-level) influenza data provided by the CDC. This graph represents a physical equilibrium that is defined by the relationships between the variables being studied. Thus, despite the fact that these screen shots are still, these equilibria shift each time the user adds a new variable or relationship of interest to the graph, and the nodes appear to float and jostle into position until they settle in a 'low energy equilibrium.' This graph is shown in Gravitational Glow mode, which is intended to help the user easily identify clusters of highly interrelated variables (red pockets). In this case, note the red/dark purple (highly correlated) cluster of variables that cluster around the "Influenza Prevalence' node. The variables in this cluster include time, population density, maximum and minimum temperatures, patient visits in hospitals, population flux (travel in and out of a region), all of which help determine the influenza epidemic dynamics.

Figure 3.12 is a screenshot of a selected subgraph of the Gravity Graph constructed from the output of MI3+ on cholera data from Bangladesh. Again, note the red cluster of highly related variables which surround the ‘cholera prevalence’ node. In this case, the Gravity Graph captures the factors that help define this particular cholera epidemic. Muslims live in only one quadrant of this region of Bangladesh, at a population density three times that of Hindus. Their increased population density is correlated with more clinics and often access to safer water sources. Therefore, cholera is significantly more rampant in the Hindu regions in this study. Correspondingly, the Gravity Graph shows the cluster of cholera prevalence, religion, population density, distance to water sources, number of clinics, and x and y GPS coordinates.

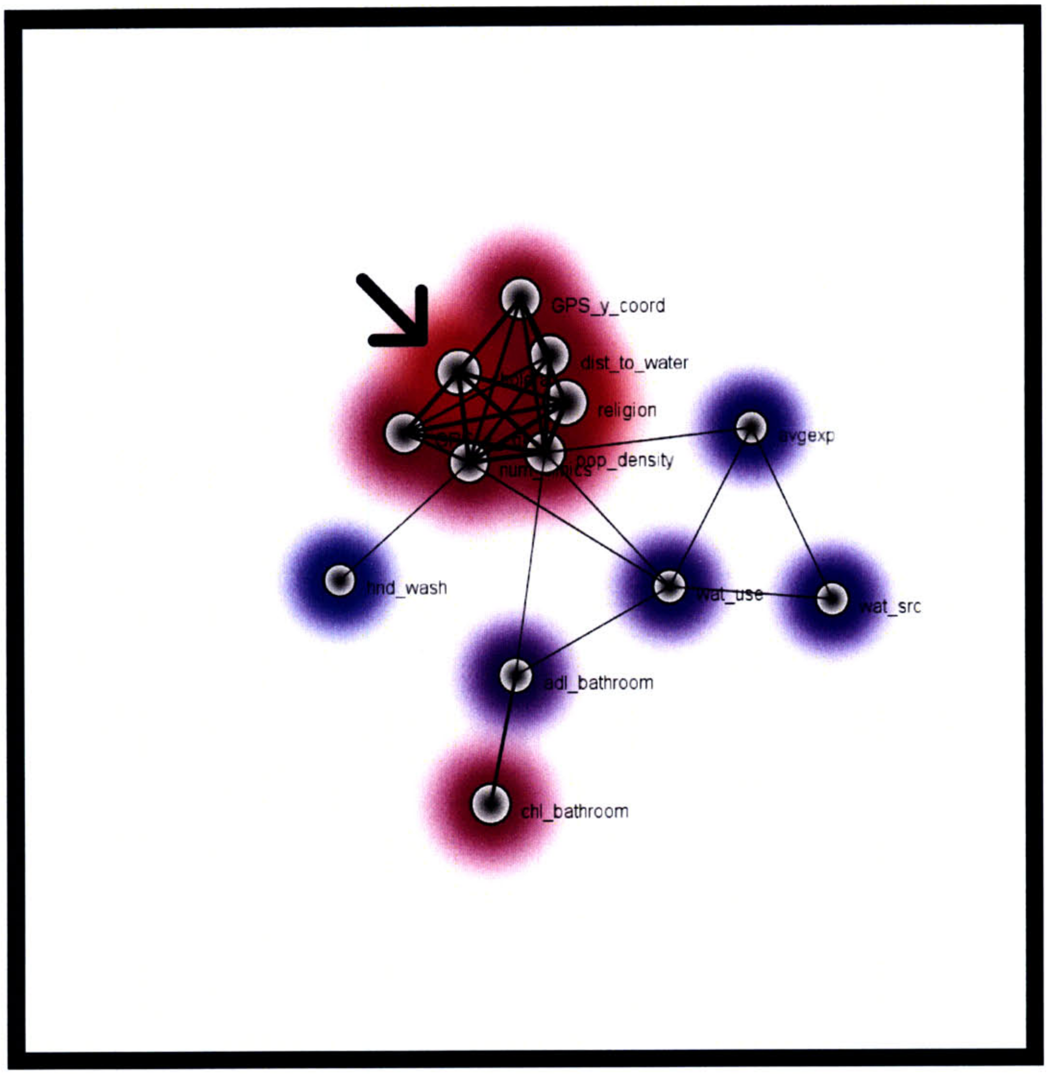


**Figure 3.12. A Cholera Gravity Graph (Bangladesh Cholera Data)**
There is a screenshot of Gravity Graph that is constructed from household-lever cholera data provided by the International Vaccine Initiative. The 'cholera prevalence' node is highlighted by the black arrow. In this case, note the red/dark purple (highly correlated) cluster of variables that cluster around the 'cholera prevalence' node. This Gravity Graph meaningfully captures the factors that help define this particular cholera epidemic. Muslims live in only one quadrant of this region of Bangladesh, at a population density three times that of Hindus. Their increased population density is correlated with more clinics and often access to safer water sources. Therefore, cholera is significantly more rampant in the Hindu regions in this study. Correspondingly, the Gravity Graph shows the cluster of cholera prevalence, religion, population density, distance to water sources, number of clinics, and x and y GPS coordinates.

### 3.8.3 The Usefulness of Gravity Graphs

As demonstrated in Section 3.8.2, Gravity Graphs can be very useful as a quick way to understand the relationships between multiple factors that might be affecting an outcome variable of interest. However, the real power of the VisuaLyzer approach lies in its use of Gravity Graphs to select several interesting clusters of related variables for further analysis. Chapter 4 covers how the relationships between variables in clusters of interest can be examined rapidly and dynamically, using VisuaLyzer's visualization environment. Additionally, using a case study, Chapter 6 discusses a larger, more complex usage of a Gravity Graph together with the visualization environment.

## *3.9 Further Work*

The CR2, MI3, and MI3+ algorithms and gravity graph presented in this chapter are a crucial component of VisuaLyzer's exploratory and rigorous approach to data analysis. Despite their successful performance, there are sever matters which could be further investigated:

1. ***Rigorously defining what these algorithms target:*** It would be tremendously useful to be able to define what these algorithms are looking for in a relationship between two variables. For example, should a sine wave that is repeated for 10 periods return the same score a single sine wave? Are these relationships identical, or is one intrinsically more 'complex' than the other? What is the fundamental definition of 'complexity' that should be used to judge relationships?
2. ***Exploring other properties of the characteristic curve/manifold:*** While maximum slope and over-under area/volume are very telling of relationships between variables, other properties of this new family of characteristic curves/manifolds should be investigated. There are likely properties other than maximum slope and over-under area/volume that will provide useful information about these characteristic curves/manifolds. It will be useful to identify other such key characteristics, and to be able to provide a small set of useful metrics as output.
3. ***Method of Binning:*** This chapter laid the foundation for several binning approaches that can be used in conjunction with CR2, MI3, and MI3+. However, one additional binning approach could use k-means clustering to determine bin boundaries. The performance of this approach should be compared MI3+'s

performance. This method should perform marginally better on functions like sine waves with varying periods; however, it will be significantly more time-intensive.

This new class of algorithms involving a spectrum of correlation ratio and mutual information calculations to detect dependencies among finite real-valued data has shown great promise thus far as tools for data exploration, and should continue to be investigated further.

# Chapter 4: A Dynamic, Rapid Visualization Environment

VisuaLyzer includes a clear visualization environment for the rapid analysis of multi-dimensional data. This component aims to do just the opposite of the first: to increase the number of dimensions that can easily be analyzed by a human. Well-designed, innovative, visualizations allow users to see as many as ten dimensions at a time in a dynamic, interactive manner, as compared to the two or three dimensions that are traditionally viewed in a static manner. This allows for the analysis of as many of the essential relationships identified by the VisuaLyzer analytic suite (Chapter 3) as possible at a time. VisuaLyzer's intuitive multi-dimensional visualizations allow for the rapid identification of trends and potentially for the explanation of relationships between variables of interest. This chapter describes VisuaLyzer's visualization environment and explores its abilities through several examples.

## *4.1 The Visualization Workspace*

VisuaLyzer's visualization environment is modeled as a workspace, which is intended to resemble a 'virtual canvas' on which the user can create and modify various visualizations that pertain to his analysis. This workspace is shown in Figure 4.1 below, and contains several components.

### 4.1.1 The View Menu

The view menu is displayed across the bottom of the visualization environment when no views are being viewed. This menu contains icons for the various views that are included in the visualization environment. Furthermore, users can develop new views and add them to this menu for use. Through this menu, users can launch new views of various types for data exploration.

### 4.1.2 The Workspace Canvas

Once a view is created, it exists in the workspace and is displayed on the workspace canvas. It can be engaged (viewed in full screen), or disengaged (viewed as a component, alongside other components on the workspace canvas). Multiple views can be created and modified at the same time using the workspace, and the workspace canvas is intended to allow the user to explore the data through multiple views concurrently.

### 4.1.3 The Variable Menu

The variable menu is always displayed in the bottom left corner of the visualization environment. It contains a list of all of the variables in a loaded database or set of databases (joined on a primary key). These variables can be selected by the user to be examined in a visualization.

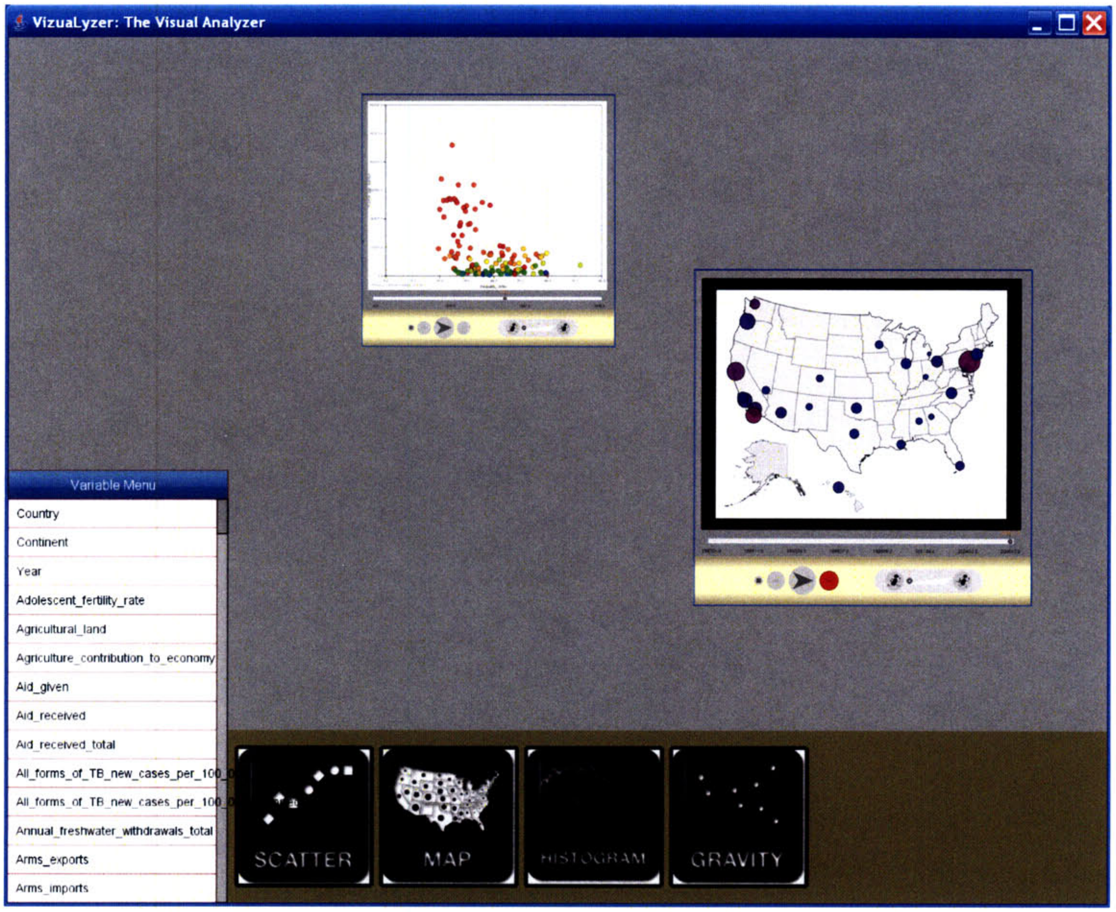

VizuaLyzer: The Visual Analyzer
Variable Menu
Country
Continent
Year
Adolescent_fertility_rate
Agricultural_land
Agriculture_contribution_to_economy
Aid_given
Aid_received
Aid_received_total
All_forms_of_TB_new_cases_per_100_0
All_forms_of_TB_new_cases_per_100_0
Annual_freshwater_withdrawals_total
Arms_exports
Arms_imports
SCATTER
MAP
HISTOGRAM
GRAVITY

**Figure 4.1. The Visualization Workspace**
This is VisuaLyzer's visualization workspace. The menu along the left side is the variable menu, the menu along the bottom is the view menu, and the gray area is the workspace canvas. In this image, two views have been created,

## *4.2 Views*

The visualization environment contains several views, which can be created from the view menu. Views can be thought of as the equivalent of "documents," and can be created and modified at any time. Different views are useful for exploring different data, and the types of views available are discussed in the sections that follow. However, each view contains three standard components that are used to interact with it.

### 4.2.1 The Encoding Menu

Views allow users to explore several dimensions of data simultaneously. This is achieved by mapping different variables to various properties of objects being visualized in a view. These mappings are called encodings, and are specific to each view. The encoding menu contains all of the encodings available for a given view. To visualize a particular variable, a user must drag a variable from the variable menu onto an encoding on the encoding menu. Once a variable is loaded onto an encoding, the encoding's range is set to the range of that particular variable. Furthermore, the encoding contains a slide bar with two filters, which can be used to select a sub-range of the encoded variable to visualize. Figure 4.2 shows an encoding with a variable loaded and filters set.

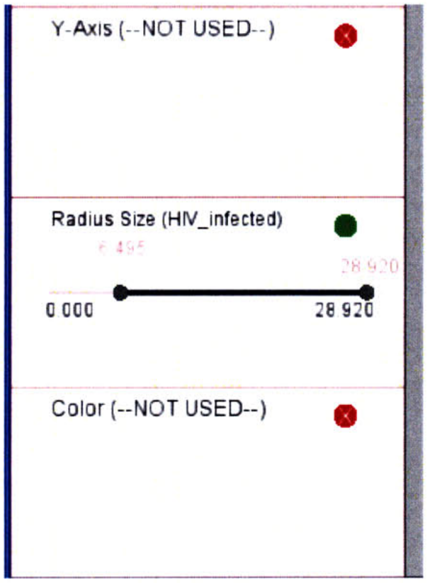


**Figure 4.2. A Sample Encoding**
This is a part of an encoding menu from a Scatter View, which contains a Y-Axis encoding, Radius Size encoding, and a Color encoding. In this case, a variable that represents HIV prevalence has been loaded onto the Radius Size encoding, meaning that in this particular scatter plot, in which each country in the world is represented by a bubble, the size of that bubble would correspond to that country's HIV prevalence rate. Furthermore, a filter has been used to select only countries whose HIV prevalence rate is larger than 6.495. This means that any countries that

A user may visualize as many variables at a time as there are encodings for a given view. Views have anywhere from six to ten encodings. The encoding menu is always displayed in the top left corner of a view when the view is engaged.

### 4.2.2 The Global Scrollbar

Each view contains a "Global Scroll" encoding and a corresponding global scroll bar, which is displayed directly below the plot in a given view. This "Global Scroll" encoding is special in the sense that it can only be loaded with independent variables (for example, with variables such as time or country). This variable must be independent because it is the variable that is used to 'animate' the view using the View Controls (Section 4.2.3). For example, if time is loaded onto the "Global Scroll," then a user can 'play' the visualization forwards and backwards through time using the View Controls, and watch as the data change at each time step.

### 4.2.3 The View Controls

The View Controls are used to animate a view. The View Controls contain Play, Step Forward, Step Backwards, and Stop buttons. It also contains a frame-rate control slide bar, which is used to set the speed at which the view animates (anywhere from 1 to 20 frames per second). The View Control effectively 'play' through the variable loaded onto the Global Scroll.

For example, suppose that a circle is placed on every country on a world map, that the circle's size is proportional to the prevalence of a disease in that country, and that the variable "Year" is loaded onto the Global Scroll. In this case, the buttons in the View Control could be used to 'play through time,' animating the circles on each of the countries, which would grow and shrink according to the prevalence of the disease at each country each year.

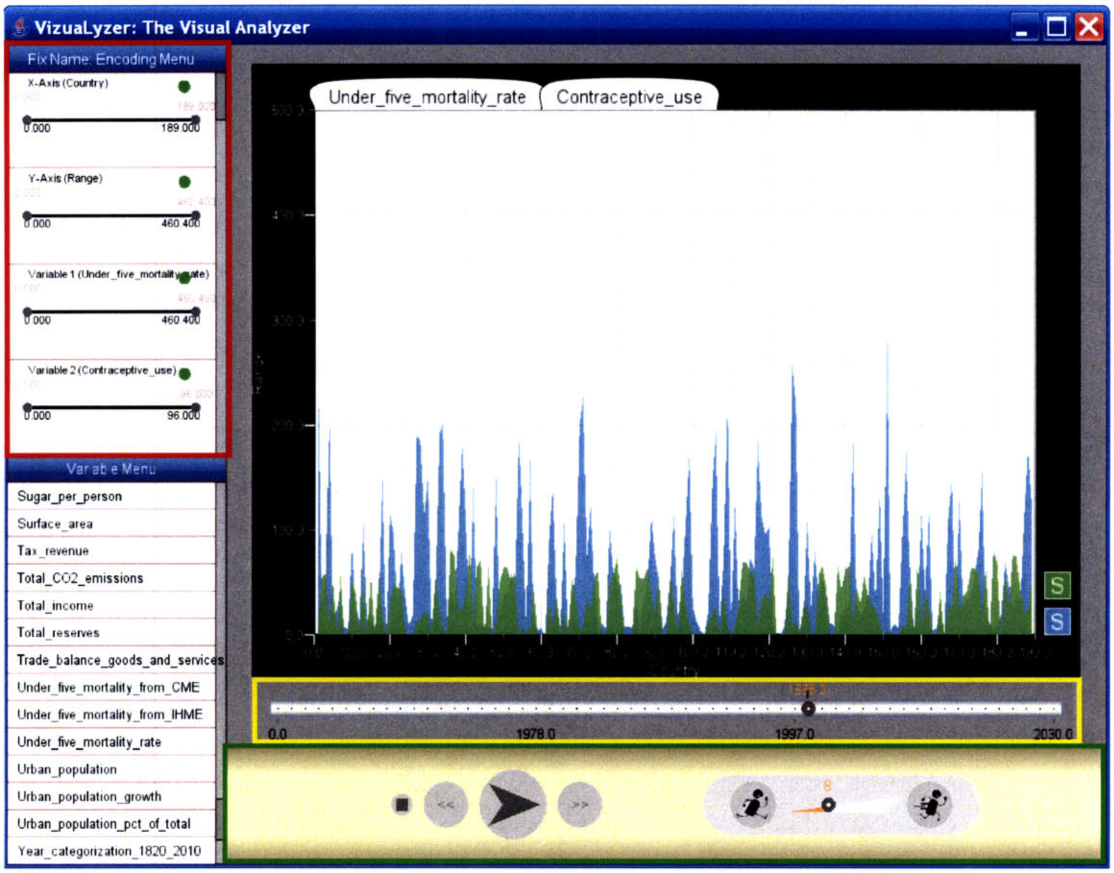


**Figure 4.3. A Sample View (Histogram View)**

This is a sample (engaged) view. The encoding menu is boxed in red, the Global Scroll is boxed in yellow, and the View Controls is boxed in green.

## 4.3 The Map View

A Map View is a view which plots data onto a given map, and is designed for visualizing spatial relationships (Fry, 2007). The Map View is particularly useful in exploring epidemiological datasets as it can be used to elucidate the temporospatial dynamics of diseases or large scale public health problems. For example, Map Views can be used to answer questions like "What relationships exist between the introduction of a new drug into a region and disease prevalence in that region, among treated and untreated individuals?", and "What other factors have an impact on this relationship?"

### 4.3.1 Map View Encodings

Map Views contain the following eight encodings:

1. ***Space (by default):*** The user must specify what the 'items' are in a dataset being viewed on the map (e.g. country, state, etc.).
2. ***Global Scroll:*** An independent variable must be loaded onto this encoding in order to animate the map. In the case of the map view, a temporal variable is usually used.
3. ***Radius Size A and Radius Size B:*** When a variable is loaded onto a Radius Size encoding, a circular bubble will be displayed on each location on the map, whose radius is proportional to the value of this variable. If variables are loaded onto both Radius Size encodings, then two circular bubbles are shown at each location.
4. ***Color A and Color B:*** When a variable is loaded onto a Color encoding, the value of that variable will be represented by the color of the circular bubble that is displayed at each location on the map. This is done by mapping the minimum value of the loaded variable onto one color (say, blue) and the maximum value of the loaded variable onto another color (say, red), and interpolating between the two colors based on the value of the loaded variable at each location. The Color A encoding sets the color of the bubbles associated with Radius Size A encoding and the Color B encoding sets the color of the bubbles associated with Radius Size B encoding.
5. ***Opacity A and Opacity B:*** When a variable is loaded onto an Opacity encoding, the value of that variable will be represented by the opacity of the circular bubble that is displayed at each location on the map. The Opacity A encoding sets the opacity of the bubbles associated with Radius Size A encoding and the Opacity B encoding sets the color of the bubbles associated with Radius Size B encoding.

### 4.3.2 A Simple Map View Example: Influenza in the United States

A Map View can easily be used to examine the temporospatial dynamics of influenza in the United States. A simple example of the Map View can be used to plot the prevalence of influenza through time and space. Figure 4.4 contains three snapshots of an animated Map View which shows an influenza wave passing through the United Sates from west to east. In this view, influenza prevalence is loaded onto the Radius Size A encoding. Despite the fact that these screenshots are static, the Global Scroll at the bottom is actively moving through the selected independent variable (in this case, time), creating an animation of this influenza epidemic over time. Finally, in this particular visualization, note that only three variables are encoded (space, time, and influenza prevalence). More complex Map Views can be constructed by loading more variables onto encodings. For example, the dependence of influenza on temperature can be examined by loading a temperature variable onto a Color encoding.

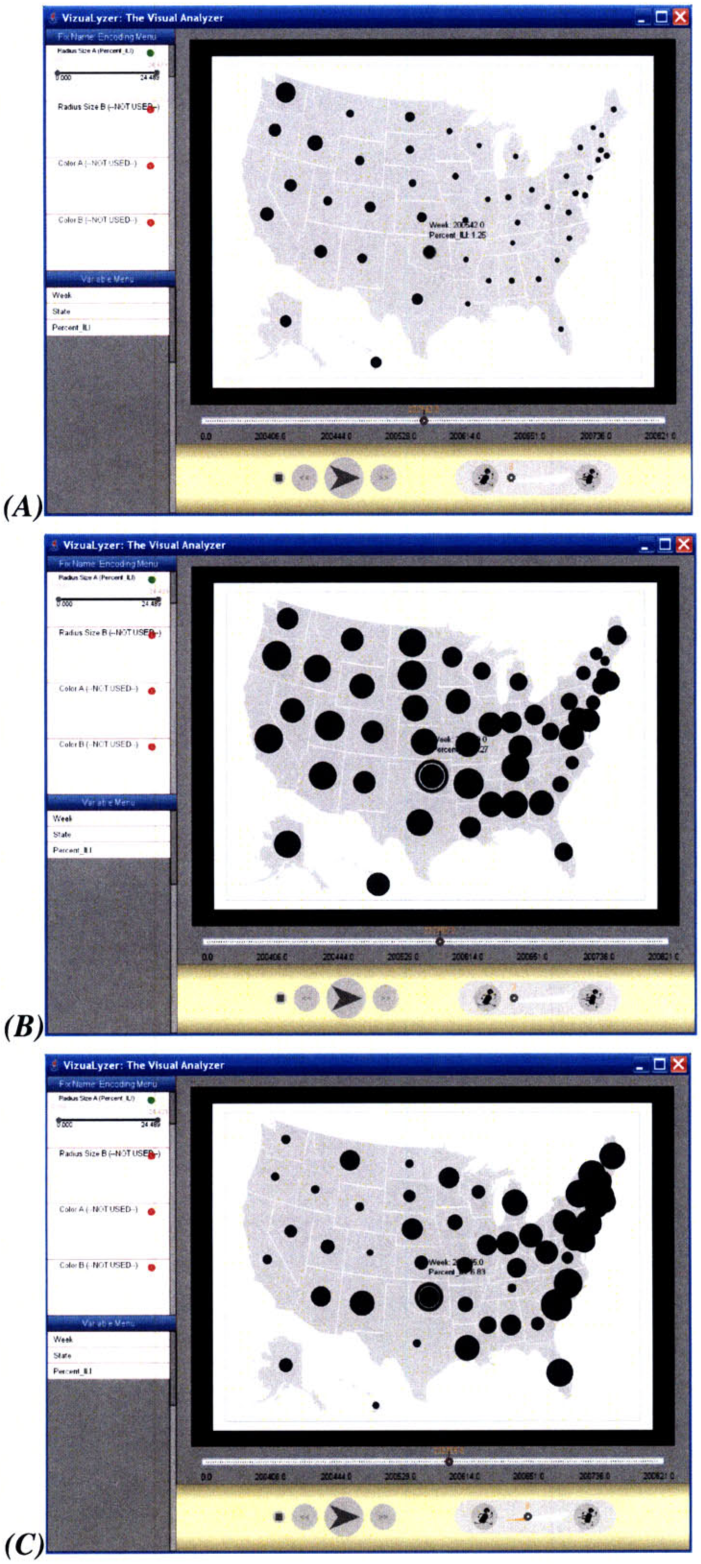
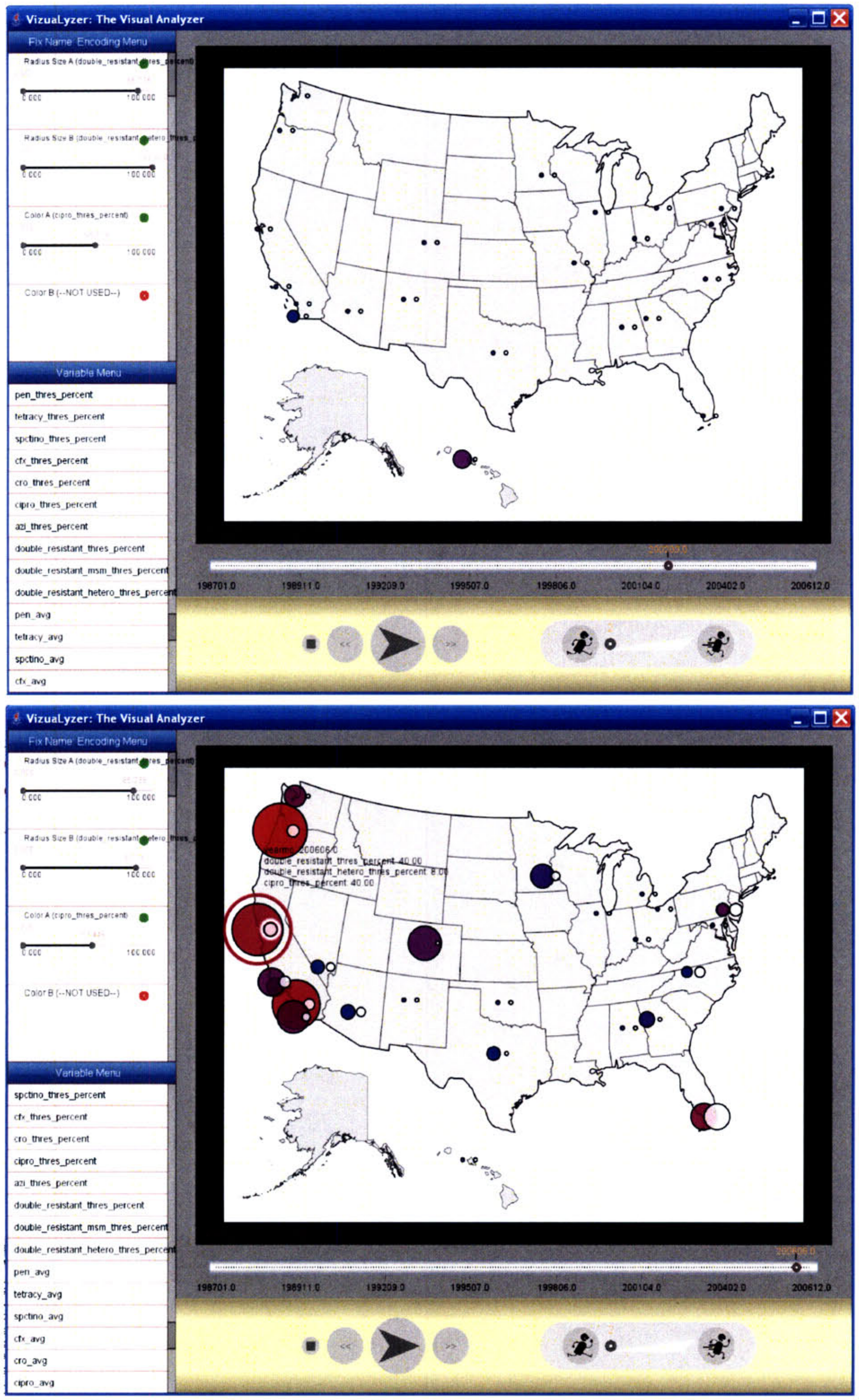


**Figure 4.4. A Map View: Influenza in the United States**

These are three screenshots of a Map View that shows an influenza epidemic spreading across the United States from west to east in 2005. The size of the bubble on each state is proportional to the prevalence of influenza in that state in a given week. Each snapshot was taken two weeks after the previous one. Despite the fact that these screenshots are static, the Global Scroll at the bottom is actively moving through the selected independent variable (in this case, time), creating an animation of this influenza epidemic over time.

### 4.3.3 A More Complex Map View Example: Gonorrhea in the United States

Significantly more complex Map Views than the one shown above can be constructed. For example, a Map View can be used to study the temporospatial dynamics of drug-resistant gonorrhea in the United States. Figure 4.5 below is a Map View plotting the prevalence of a particular drug-resistant strain of gonorrhea, according to geographic region, versus drug prevalence, stratified by gender and sexual orientation, and over time. Again, despite the fact that these screenshots are static, the Global Scroll is actively moving through the selected independent variable (in this case, time), creating an animation of the spread of drug-resistant gonorrhea over time. The data in this figure is reported by 40 clinics around the US and was provided by the CDC. The radius of the circles centered on each major city represent the amount of drug-resistant gonorrhea in that city for a given week, and their color represents the drug prevalence in that city for that week. The red/blue bubbles represent the drug-resistant cases among homosexual men and the white/black bubbles represent the drug-resistant cases among heterosexual men. As the scrollbar moves through time, the circles grow and shrink according to number of cases of gonorrhea, and change color according to drug usage. Notice how over time (from 2001 to 2006), drug usage has increased (circles become more red), while drug resistance emerges particularly on the west coast (circles get larger), specifically within the homosexual population (white circles still relatively small).


VizuaLyzer: The Visual Analyzer

Fix Name Encoding Menu

Radius Size A (double_resistant_thres_percent)

0.000 100.000

Radius Size B (double_resistant_hetero_thres_percent)

0.000 100.000

Color A (cipro_thres_percent)

0.000 100.000

Color B (--NOT USED--)

yearmo: 200606.0
double_resistant_thres_percent: 40.00
double_resistant_hetero_thres_percent: 8.00
cipro_thres_percent: 40.00

**Figure 4.5. A Map View: Drug-Resistant Gonorrhea in the United States**

These are two snapshots of a Map View showing gonorrhea data from 40 clinics in major cities across the US. The radius of the red/blue circles centered on each major city represents the prevalence of a drug-resistant strain of gonorrhea in the homosexual population for a given week. The color of these circles represents amount of drug usage (interpolated between blue = 0% and red = 100%). The radius of the white circles represents the prevalence of that same strain in the heterosexual population. The top snapshot is from early 2002, while the bottom snapshot is from 2006. Notice how over time, drug usage has increased (circles becoming more red), while drug resistance has emerged (circles getting larger) particularly on the west coast, specifically within the homosexual population (white circles still relatively small).

## *4.4 The Scatter View*

A Scatter View is a more general view than a Map View and is applicable to more types of data. In this view, data is plotted on a 2-dimmensional plot as points (circular bubbles or differing shapes). Several properties of each point can be encoded with a variable, and the points can be animated by loading an independent variable onto the Global Scroll. Scatter Views can be useful for plotting just about any kind of data (Rosling, 2007).

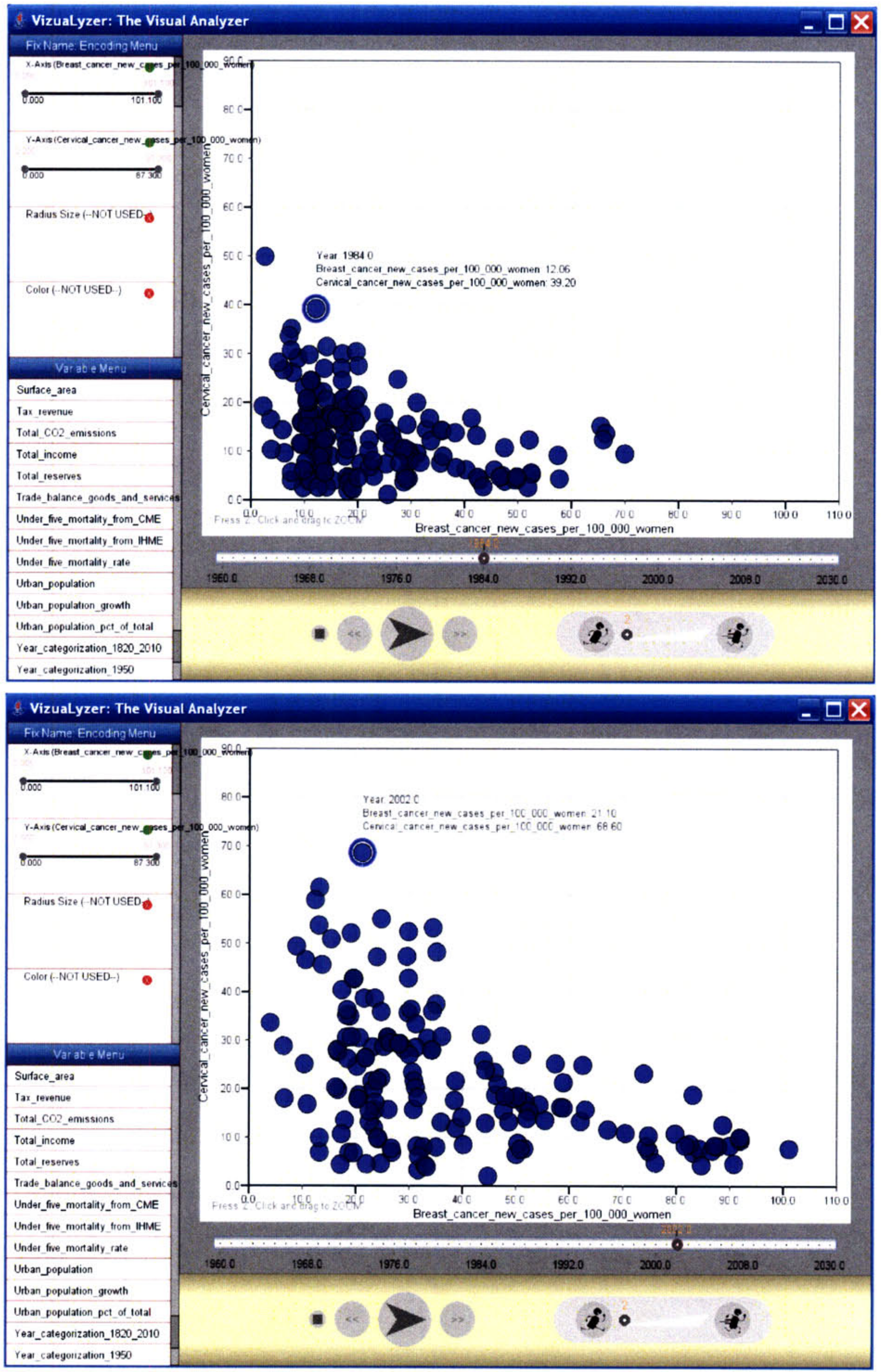
### 4.4.1 Scatter View Encodings

Scatter Views contain the following seven encodings:

1. ***X-Axis:*** When a variable is loaded onto the X-Axis encoding, the x-coordinate of each item being plotted (circular bubble) is defined by the value of this particular variable.
2. ***Y-Axis:*** When a variable is loaded onto the Y-Axis encoding, the y-coordinate of each item being plotted is defined by the value of this particular variable.
3. ***Global Scroll:*** An independent variable must be loaded onto this encoding in order to animate the scatter plot. Unlike in a Map View, any independent variable can be 'played through'. For example, "Country" could be loaded onto the Global Scroll, and when the plot is animated, it would scroll through various country profiles.
4. ***Radius Size:*** When a variable is loaded onto the Radius Size encoding, the size of each point on the scatter plot becomes proportional to the value of this variable.
5. ***Color:*** When a variable is loaded onto the Color encoding, the value of that variable will be represented by the color of the points on the scatter plot. This is done by mapping the minimum value of the loaded variable onto one color (say, blue) and the maximum value of the loaded variable onto another color (say red), and interpolating between the two colors based on the value of the loaded variable at each location.
6. ***Opacity:*** When a variable is loaded onto the Opacity encoding, the value of that variable will be represented by the opacity of the points on the scatter plot.
7. ***Shape:*** When a variable is loaded onto the Shape encoding, each point on the scatter plot is turned into a regular n-polygon where *n* is proportional to the value of this particular variable.

### 4.4.2 A Scatter View Example

A Scatter View can be used to examine global public health trends. For example, Figure 4.6 below uses a Scatter View to show how the rates of breast cancer and cervical cancer compare to each other in countries around the world, and uses the Global Scroll to show how these rates have changed over the last twenty years. To attempt to explain the patterns in cancer rates over these twenty years a more complex Scatter View can be constructed. Figure 4.7 demonstrates the ability to visualize a higher number of dimensions using the Scatter View by encoding two more variables. This figure shows how geography and average income relate to breast cancer and cervical cancer rates around the world.

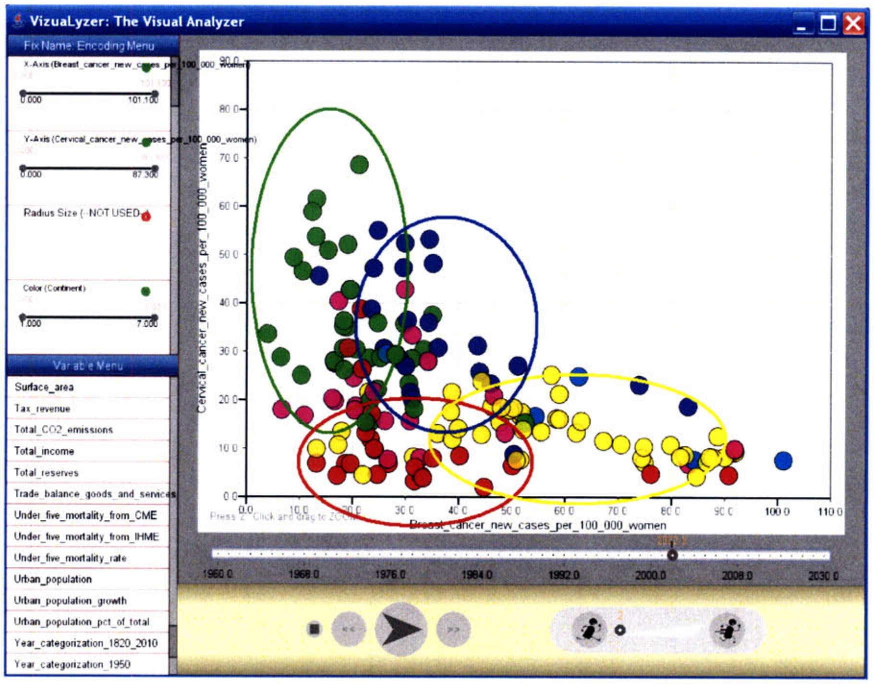

VizuaLyzer: The Visual Analyzer
Fix Name: Encoding Menu
X-Axis (Breast_cancer_new_cases_per_100_000_women)
0.000
101.100
Y-Axis (Cervical_cancer_new_cases_per_100_000_women)
0.000
87.300
Radius Size (--NOT USED--)
Color (--NOT USED--)
Variable Menu
Surface_area
Tax_revenue
Total_CO2_emissions
Total_income
Total_reserves
Trade_balance_goods_and_services
Under_five_mortality_from_CME
Under_five_mortality_from_IHME
Under_five_mortality_rate
Urban_population
Urban_population_growth
Urban_population_pct_of_total
Year_categorization_1820_2010
Year_categorization_1950
Cervical_cancer_new_cases_per_100_000_women
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
Breast_cancer_new_cases_per_100_000_women
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
100.0
110.0
Press 'z'. Click and drag to ZOOM
1960.0
1968.0
1976.0
1984.0
1992.0
2000.0
2008.0
2030.0

Year 1984.0
Breast_cancer_new_cases_per_100_000_women: 12.06
Cervical_cancer_new_cases_per_100_000_women: 39.20

Year 2002.0
Breast_cancer_new_cases_per_100_000_women: 21.10
Cervical_cancer_new_cases_per_100_000_women: 68.60

**Figure 4.6. A Scatter View: Breast Cancer and Cervical Cancer Around the World**

There are two snapshots of a Scatter View showing breast cancer and cervical cancer rates for countries around the world. The number of new cases in a given year of breast cancer is loaded onto the x-axis and the number of new cases in a given year of cervical cancer is loaded onto the y-axis. These snapshots shows a circle for each country in the world (for which there exists data), placed on the plot according to its cancer rates. The time variable "Year" is loaded onto the Global Scroll, and the plot can be animated to watch how these cancer rates change over time. The top snapshot was taken when the Global Scroll was on the year 1980 and the bottom snapshot was taken when the Global Scroll was on the year 2002. Note how the rates of both cancers have risen in these 22 years, but only amongst specific countries in each case. See Figure 4.7 for a further investigation as to what other factors help explain this phenomenon.


VizuaLyzer: The Visual Analyzer
Fix Name: Encoding Menu
X-Axis (Breast_cancer_new_cases_per_100_000_women)
0.000
101.100
Y-Axis (Cervical_cancer_new_cases_per_100_000_women)
0.000
87.300
Radius Size (--NOT USED--)
Color (Continent)
1.000
7.000
Variable Menu
Surface_area
Tax_revenue
Total_CO2_emissions
Total_income
Total_reserves
Trade_balance_goods_and_services
Under_five_mortality_from_CME
Under_five_mortality_from_IHME
Under_five_mortality_rate
Urban_population
Urban_population_growth
Urban_population_pct_of_total
Year_categorization_1820_2010
Year_categorization_1950
Cervical_cancer_new_cases_per_100_000_women
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
100.0
110.0
Breast_cancer_new_cases_per_100_000_women
Press 'Z' Click and drag to ZOOM
1960.0
1968.0
1976.0
1984.0
1992.0
2000.0
2008.0
2030.0

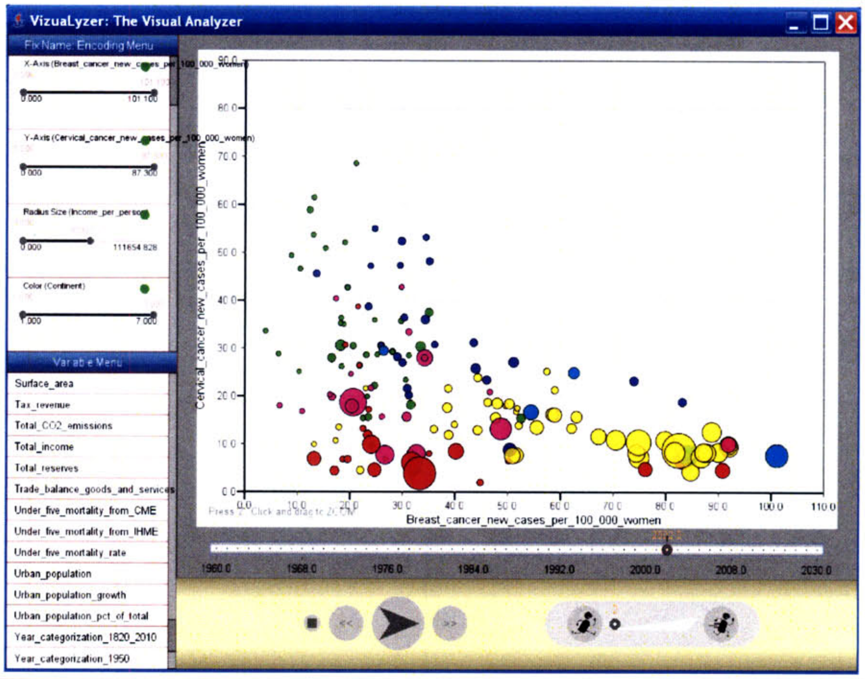

VizuaLyzer: The Visual Analyzer
Fix Name: Encoding Menu
X-Axis (Breast_cancer_new_cases_per_100_000_women)
0.000
101.100
Y-Axis (Cervical_cancer_new_cases_per_100_000_women)
0.000
87.300
Radius Size (Income_per_person)
0.000
111654.828
Color (Continent)
1.000
7.000
Variable Menu
Surface_area
Tax_revenue
Total_CO2_emissions
Total_income
Total_reserves
Trade_balance_goods_and_services
Under_five_mortality_from_CME
Under_five_mortality_from_IHME
Under_five_mortality_rate
Urban_population
Urban_population_growth
Urban_population_pct_of_total
Year_categorization_1820_2010
Year_categorization_1950
Cervical_cancer_new_cases_per_100_000_women
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
100.0
110.0
Breast_cancer_new_cases_per_100_000_women
Press 'Z' Click and drag to ZOOM
1960.0
1968.0
1976.0
1984.0
1992.0
2000.0
2008.0
2030.0

**Figure 4.7. A Scatter View: Breast Cancer and Cervical Cancer Around the World, By Continent and Income Per Person**

There are two snapshots of a Scatter View attempt to examine the pattern by which breast cancer and cervical caner rates have grown in countries around the world. The number of new cases in a given year of breast cancer is loaded onto the x-axis and the number of new cases in a given year of cervical cancer is loaded onto the y-axis. These snapshots shows a circle for each country in the world (for which there exists data), placed on the plot according to its cancer rates. The time variable "Year" is loaded onto the Global Scroll, and these snapshots were taken when the Global Scroll was on the year 2002. Furthermore, the color of each circle represents is geography (yellow = Europe and Central Asia, green = Sub-Saharan Africa, purple = Northern Africa and the Middle East, cyan = North America, blue = South America, red = Oceania). Note how these geographic regions cluster according to their cancer rates, with Africa having the lowest breast cancer rates but the highest cervical cancer rates, and Europe and North America having low cervical cancer rates but high breast cancer rates.

The Bottom plot also encodes the average income per person in each country as that country's bubble radius size. Again, there is a clear trend. Rich countries tend to have higher breast cancer rates and lower cervical cancer rates, and poor countries tend to have lower breast cancer rates and higher cervical cancer rates. These Scatter Views clearly show how different lifestyles affect cancer rates.

## *4.5 The Histogram View*

A Histogram View is useful for exploring the distributions of variables, or for comparing multiple similar variables. Similarly to a Scatter View, a Histogram View plots points on a 2-dimensional scatter plot; however, in a Histogram View, multiple variables can be loaded onto the y-axis to allow for comparisons between different variables' distributions. As with all VisuaLyzer views, a Histogram view can be animated by loading an independent variable onto the Global Scroll.

### 4.5.1 Histogram View Encodings

Histogram Views contain the following seven encodings:

1. ***X-Axis:*** When a variable is loaded onto the X-Axis encoding, the x-coordinate of each item being plotted is defined by the value of this particular variable.
2. ***Variable 1, Variable 2, Variable 3, Variable 4, and Variable 5:*** When a variable is loaded onto a Variable encoding, the y-coordinate of each item being plotted is defined by the value of this particular variable. Up to five different variables can be loaded onto Variable encodings, all of which are mapped to the y-axis. Each different variable is plotted on the view as a different series, using a unique color.
3. ***Global Scroll:*** An independent variable must be loaded onto this encoding in order to animate the scatter plot. Unlike in a Map View, any independent variable can be 'played through'. For example, "Country" could be loaded onto the Global Scroll, and when the plot is animated, it would scroll through the distributions of loaded variables for various countries.

Although a Y-Axis encoding appears on the Histogram View encoding menu, no variables can be loaded onto this encoding. This encoding exists to give the user control over the total range of the y-axis (if the various variables loaded onto the Variable encodings have different ranges).

### 4.5.2 Histogram View Modes

Histogram Views can plot data in one of five ways:

1. bar graphs,
2. line graphs,
3. area plots (area under the curve),
4. scatter plots; and
5. curved line plots (polynomial fits).

In most cases, which mode to use is simply a matter of user preference.

### 4.5.3 Turning Variables On/Off

Given that it is possible to load up to five different variables onto the y-axis simultaneously, a method for toggling variables' visibility on the histogram is necessary. For each variable loaded onto a Variable encoding, a tab is displayed atop the histogram plot. If this tab is clicked, it moves to the background, and its corresponding variable is not displayed on the histogram (but remains encoded and retains all filters placed upon it). To show the variable again, the tab must be clicked again.

### 4.5.4 Stacking Variables

For every variable that is loaded onto a Variable encoding, a small button with an "S" appears to the right of the histogram plot. These buttons can be used to stack the distributions of different variables on top of each other. This may be useful if, for example, variables are in terms of percentages and stacking them would help show the relative proportion of each variable.

### 4.5.5 Histogram View Example: *Streptococcus pneumoniae* in Kenya

A Histogram View can be used to examine trends in the prevalence of various strains of *Streptococcus pneumoniae* extremely rapidly. Figure 4.7 compares the prevalences of four different strains of *Streptococcus pneumoniae* in Kenya during different months of the year. Again, while these snapshots are static, these plots can be animated to reflect the rise and fall of the various strains throughout the different months of the year.

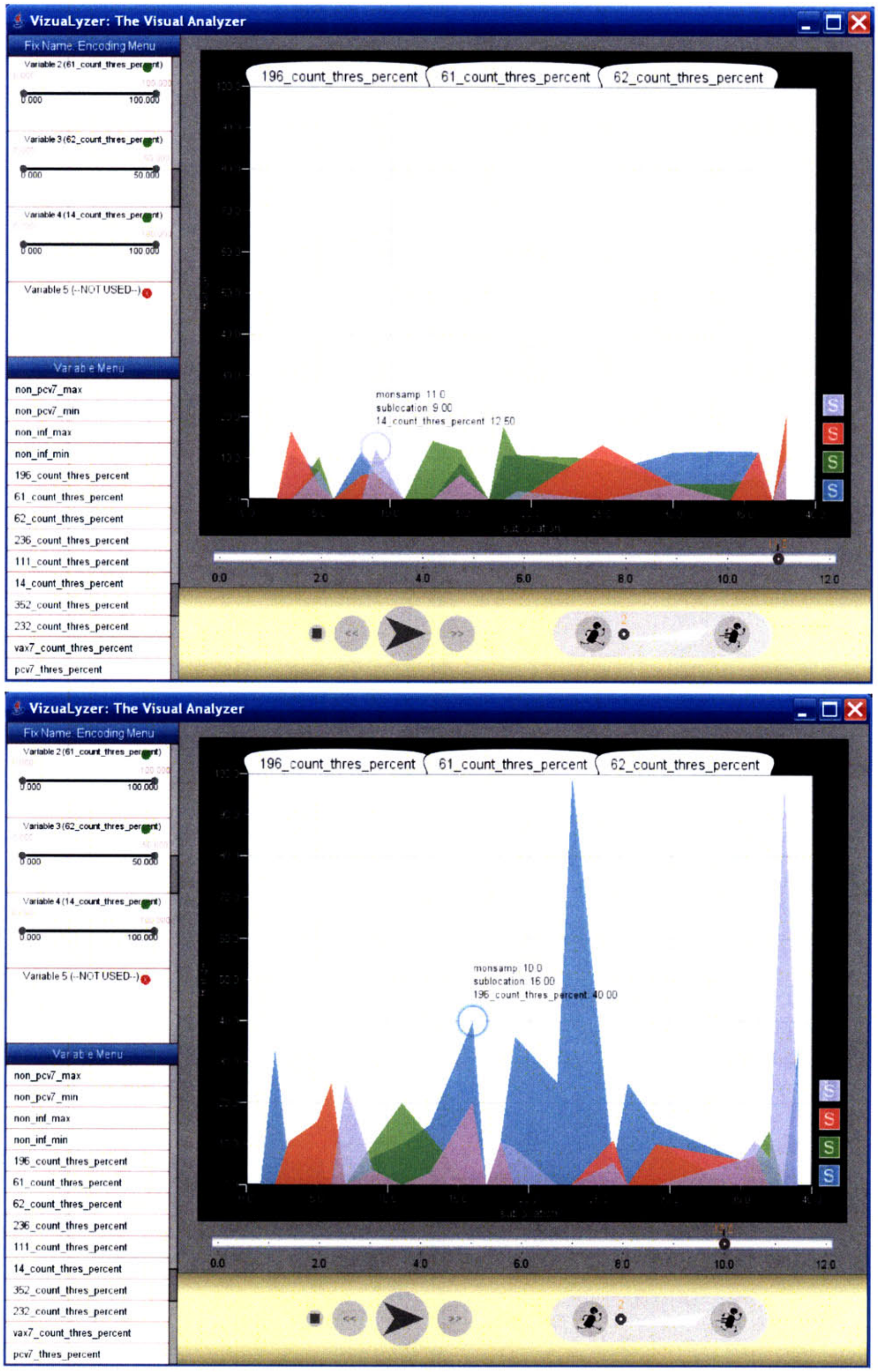


**Figure 4.7. A Histogram View: *Streptococcus pneumoniae* in Kenya**

These are two snapshots of a Histogram View showing the prevalence of four different strains of *Streptococcus pneumoniae*, shown in different colors, from 198 zones in Kenya. The top snapshot shows the prevalence rates for a typical month of the year, while the bottom snapshot shows that in October 2006, several strains (19F and 6A) were particularly prevalent. Despite the fact that these snapshots are static, these plots can be animated to reflect the rise and fall of the various strains

### *4.6 Rapid Data Exploration*

The VisuaLyzer visualization environment is designed to allow for the rapid, dynamic exploration of data. The ability to create views, add and remove variables to views on the fly, and animate views provides users with a powerful toolkit for data mining. Moreover, the ability to interact with visualizations that are capable of intuitively displaying anywhere from six to ten dimensions of data simultaneously provides an incredible advantage over standard graphing techniques. The ability to view trends rapidly and dynamically provides a powerful means for identifying confounding factors and generating new ideas about the mechanisms that define public health dynamics.

# Chapter 5: Intelligent Hypothesis Generation

VisuaLyzer combines its analytical tools and visualizations to form an advanced mechanism for guiding data exploration. Thus far, the VisuaLyzer approach to exploratory data analysis involves identifying all possible correlations between variables within a dataset in a computationally rigorous fashion, and using dynamic visualizations to explore the identified relationships of interest. The final component of this approach uses the algorithmically-identified associations, together with human intuition gathered through human interaction with the visualizations, to intelligently and automatically generate hypotheses that may be of interest to the user.

This final component of the VisuaLyzer approach is its Relationship Suggestor, which employs randomized and machine learning algorithms to identify variable relationships of interest. When a user loads a set of variables of interest into a visualization, VisuaLyzer's Relationship Suggestor runs in the background, searching for new variables that would be of interest to the user based on which variables are currently being viewed. This crucial final step of the VisuaLyzer approach is intended to search the space of variables not being examined by the user to find potentially critical relationships that may be affecting the relationships being viewed. Another way of looking at this is that by loading specific variables of interest into a visualization, the user is training the computer to search for relationships that may be of interest to the user.

## *5.1 The Relationship Suggestor*

VisuaLyzer's Relationship Suggestor is comprised of two algorithms, both of which work together to help intelligently identify relationships that might be of interest to a user. The first algorithm is a randomized algorithm that searches for variables that are dependent on the variables being viewed, and the second is a machine learning algorithm that relies on usage heuristics and collaborative filtering to identify variables that are most commonly visualized with the variables currently being viewed.

### 5.1.1 Randomized Algorithm for Relationship Identification

As a first pass at guiding user exploration, the Relationship Suggestor employs a randomized algorithm to search for variables that are related to the variables that a user has deemed interesting (the variables he has chosen to visualize). This algorithm simply randomly selects a variable from the set of variables that is being visualized and a second variable from the set of variables that is not being visualized, and computes the CR2 and MI3+ score for the relationship between the two. If either the calculated CR2 or MI3+ score is amongst the top 5% of CR2 or MI3+ scores in this dataset, then this relationship is suggested to the user as a potential relationship of interest. The algorithm determines if the CR2 or MI3+ score for a randomly selected relationship is in the top 5% by consulting the lists of CR2 and MI3+ scores generated for the entire dataset from when these algorithms were run on the entire dataset in the analytic environment. If the user did not run the CR2 or MI3+ algorithm on this dataset, then the randomized algorithm simply calculates the distribution of relationships in this dataset by randomly sampling a significant proportion of pairs of variables and calculating their CR2 and MI3+ scores. Note that this only has to be done once for a given dataset.

### 5.1.2 Variable-Based Collaborative Filtering

#### 5.1.2.1 An Introduction to Collaborative Filtering

Collaborative filtering is a machine learning technique that involves the filtering of information or patterns by using information from multiple data sources or users (G. Linden et al, 2003). For example, collaborative filtering can be used to make automatic predictions (filtering) about the interests of a user by collecting information about user-preferences from many users (collaborating). The fundamental assumption made using collaborative filtering is that those who agreed in the past tend to agree again in the future. Collaborative filtering is used by several recommendation systems, including most famously for movie recommendation by the movie rental company Netflix (A. Narayanan et al., 2006).

### 5.1.2.2 An Adaptation of Collaborative Filtering

Traditionally, collaborative filtering is performed on a collection of user ratings or preferences. However, VisuaLyzer uses an adaptation of collaborative filtering that is based on the similarity of the usage of different variables in visualizations.(G. Linden et al, 2003). VisuaLyzer's collaborative filtering algorithm stores a variable-variable matrix, which stores counters that store the number of times each pair of variables has been visualized together, as well as a variable-similarity table, which stores the 'similarity of use' of each pair of variables (as calculated by the algorithm). These two tables are created for each database that is loaded into VisuaLyzer, saved for the lifetime of the database in the VisuaLyzer platform, and updated after the creation of use of every visualization.

Each time a variable is loaded onto an encoding, VisuaLyzer's collaborative filtering algorithm uses the variable-similarity table to find variables similar to the user's currently loaded variables, aggregates those variables, and then recommends the variables that would most likely be interesting to add to the current visualization. Unlike traditional collaborative filtering, this algorithm performs well with limited usage data, producing high-quality recommendations based on as few as two or three variables. Figure 5.1 contains the pseudocode for VisuaLyzer's iterative collaborative filtering algorithm.

```
For each variable $v_i$ that is loaded onto an encoding
    For each variable $v_j$ that is simultaneously loaded
        onto an encoding
        Record that $v_i$ and $v_j$ were visualized together
        (CF[I,j] += 1)

    For each variable $v_j$
```

**Figure 5.1. Pseudo-Code for VisuaLyzer's Relationship Suggestor Collaborative Filtering Algorithm**

### 5.1.2.3 Computing the Similarity between Two Variables

Each variable in the database is represented as a *v*-dimensional vector, where *v* is the number of variables in the loaded database. This vector simply contains the counts of the number of times this variable has been visualized with every other variable in the database. The collaborative filtering algorithm computes the similarity between two variables, A and B, by measuring the cosine of the angle between their respective *v*-dimensional count vectors, as shown in Equation 5.1 below (G. Linden et al, 2003).

$$similarity(\vec{A}, \vec{B}) = \cos(\vec{A}, \vec{B}) = \frac{\vec{A} \bullet \vec{B}}{\|\vec{A}\| * \|\vec{B}\|} \qquad \textbf{Eq. 5.1}$$

### 5.1.2.4 Scalability

VisuaLyzer's collaborative filtering algorithm creates its variable-similarity table offline (whenever no visualizations are engaged). This component of the algorithm is $O(v^2)$, where *v* is the number of variables in the loaded database. Given this variable-similarity table, the algorithm's online component -- finding variables that are similar to the user's currently loaded variables – scales independently of the size of the database. The online component only depends on the number of variables which are currently loaded onto encodings, which is never more than ~10. Thus, the algorithm is fast even for extremely large datasets with a large number of variables.

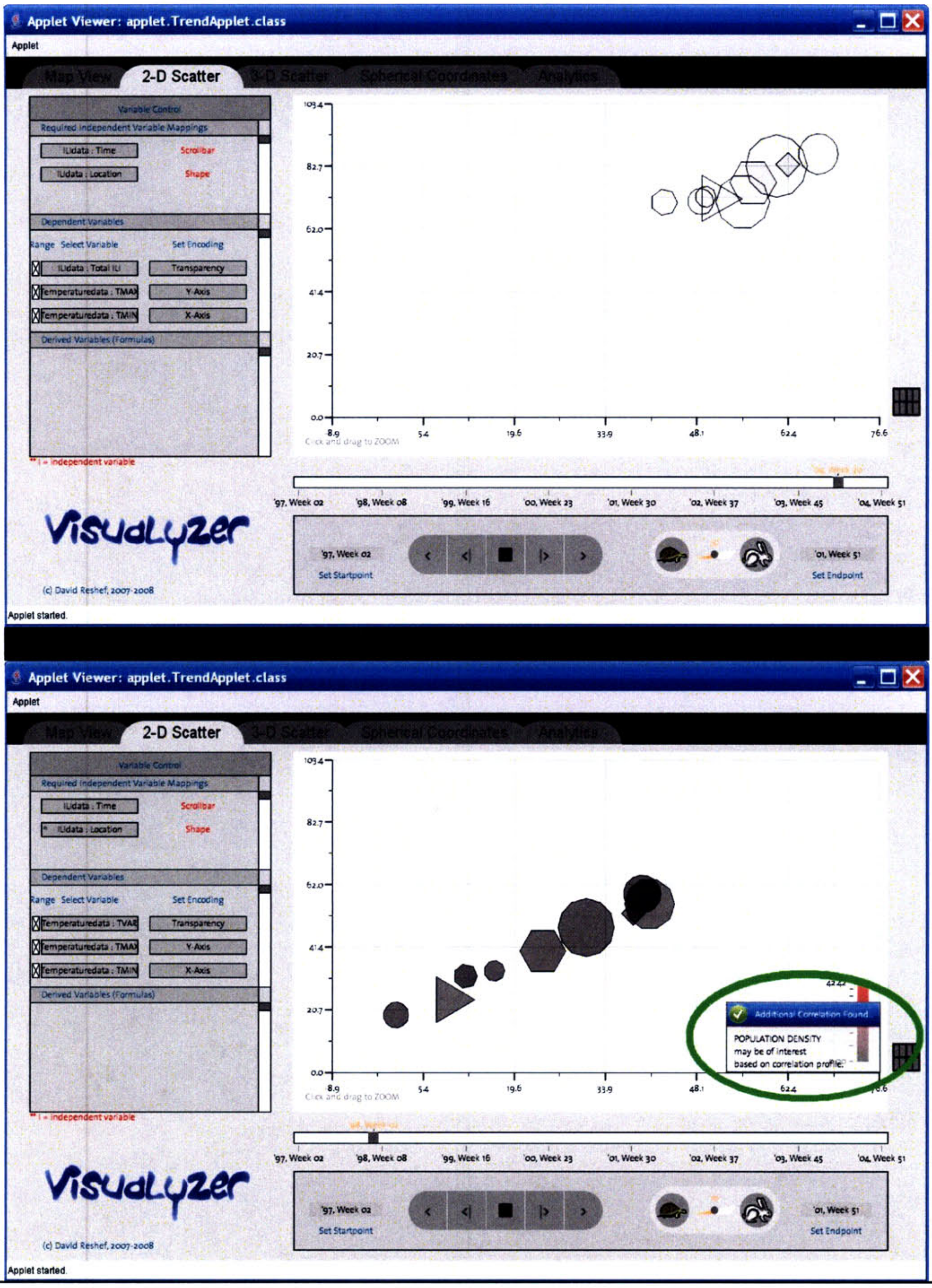
## 5.2 An Example of the Relationship Suggestor at Work

Figure 5.2 shows an example of the randomized algorithm of the Relationship Suggestor applied to an aggregated version of the influenza dataset introduced in Chapter 4. The figure contains snapshots of a Scatter View that is used to demonstrate the seasonality of influenza in regions of the United States. The figure has six variables encoded: time, space (different US regions depicted by different shapes), minimum temperature, maximum temperature, the number of hospital patient visits, and influenza prevalence. Once the view is animated, the Relationship Suggestor's randomized algorithm identifies that population density is also highly correlated with influenza prevalence, and suggests that the user add this variable to the visualization to gain a better understanding of the underlying disease dynamics. Furthermore, if the user chooses to accept the Relationship Suggestor's recommendation, VisuaLyzer helps the user by locating the variable from within the variable menu.

An example of the Relationship Suggestor's collaborative filtering algorithm is provided in the in-depth case study in Chapter 6.

## 5.3 Further Work

While VisuaLyzer's Relationship Suggestor manages to provide reasonable, and frequently useful, recommendations to users, there is a tremendous amount of room for improvement in this realm. Automated recommendation systems is a highly complex field of machine learning that could certainly be used to improve VisuaLyzer's data exploration-guiding abilities.



ILIdata : Time — Scrollbar
ILIdata : Location — Shape

☒ ILIdata : Total ILI — Transparency
☒ Temperaturedata : TMAX — Y-Axis
☒ Temperaturedata : TMIN — X-Axis

'97, Week 02 — '01, Week 51

ILIdata : Time — Scrollbar
ILIdata : Location — Shape

☒ Temperaturedata : TVAR — Transparency
☒ Temperaturedata : TMAX — Y-Axis
☒ Temperaturedata : TMIN — X-Axis

Additional Correlation Found

POPULATION DENSITY
may be of interest
based on correlation profile.

'97, Week 02 — '01, Week 51

**Figure 5.2. VisuaLyzer's Relationship Suggestor (Randomized Algorithm)**

These are two snapshots of the "Scatter View," showing influenza data from 9 regions across the US. The different shapes represent the nine regions in the US, and their opacity (how dark they are) represents the amount of influenza in their corresponding regions (dark = more influenza, clear = no influenza). The regions' position on the plot is determined by the minimum temperature for a particular week (x-axis), and the maximum temperature in that region for that week (y-axis). Finally, the size of the shape is proportional to the number of patient visits to hospitals in each reagion. As the scrollbar moves through time, the shapes move according to their x- and y- axis values, and their colors change to reflect fluctuations in influenza prevalence. Notice that in the top snapshot, temperatures are high (thus the regions are all in the upper right corner), and there is practically no influenza (all the shapes are completely clear). However, in the bottom snapshot, once temperatures drop (the shapes all move toward the lower left corner), all the shaped become dark, showing high levels of influenza.

The message box that is circled in green in the lower right corner is a correlation alert, which pops up in real-time to alert the user that Relationship Suggestor's algorithms running in the background have found an additional variable that behaves similarly to those he is visualizing (Population Density). Adding these automatically identified variables helps the user gain a truer understanding of the

# Chapter 6: A VisuaLyzer Case-Study

The power of the VisuaLyzer platform lies in its ability to integrate various types of analysis through the VisuaLyzer approach. This chapter focuses on thoroughly exploring this approach by walking through each step of the process using a case study. This case study will contain the following steps:

1. A dataset will be introduced and manipulated for ease of analysis.
2. The dataset will be inputted into CR2 and MI3+ and their results will be visualized and explored using a Gravity Graph.
3. A cluster of interrelated variables will be selected from the Gravity Graph for further exploration using the visualization environment.
4. The relationships amongst the variables in the selected cluster will be explored using a Scatter View; and
5. The Relationship Suggestor's collaborative filtering algorithm will suggest other relationships that might be of interest based on the variables being explored.

These steps obviously do not utilize all of VisuaLyzer's functionality, but they certainly embody the process by which a dataset may be explored using VisuaLyzer.

## *6.1 The Data*

The dataset for this case study comprises over 200 separate datasets parsed and downloaded from several sources at the World Health Organization and the United Nations (WHOSIS, 2009, UNdata, 2009, H Rosling, 2007). Each of these 200+ datasets contains yearly country-level data for all countries around the world, starting back as far as the nineteenth century for some countries. The period of time which will be analyzed in this case study is from 1960 to 2005, due to the amount of data available for certain datasets.

The datasets collected include major global indicators, which fall into several broad categories:

1. Demographic and socioeconomic statistics,
2. Health service coverage and resources,
3. Mortality and burden of diseases,
4. Risk factors (water and sanitation, nutrition, etc),
5. Education (literacy rates, math competencies, etc); and
6. Technology (number of cell phones, number of internet users, etc).

## *6.2 Data Management*

The individual 200+ datasets were downloaded and imported into VisuaLyzer, where they were all compiled into a single composite database. While these particular databases did not need further aggregation, many datasets often do. This can be achieved using VisuaLyzer's built-in aggregator at any time. Several of the datasets explored in earlier chapters were aggregated in this manner.

## *6.3 Identifying Clusters of Associated Variables*

The newly created composite database is incredibly rich and complex, containing over 200 dimensions. It is difficult to know how to begin analyzing this database, and it is for this exact purpose CR2 and MI3+ were developed.

### 6.3.1 Applying CR2 and MI3+

The composite database was inputted into both the CR2 and MI3+ algorithms in an attempt to identify and quantify relationships between variables within it. The algorithms evaluate the strength of the relationship between *all* pairs of variables, thus their output was some ~10,000 relationships. Plots of the lists of their respective correlation coefficients are shown below. Note that these lists were sorted before plotting.

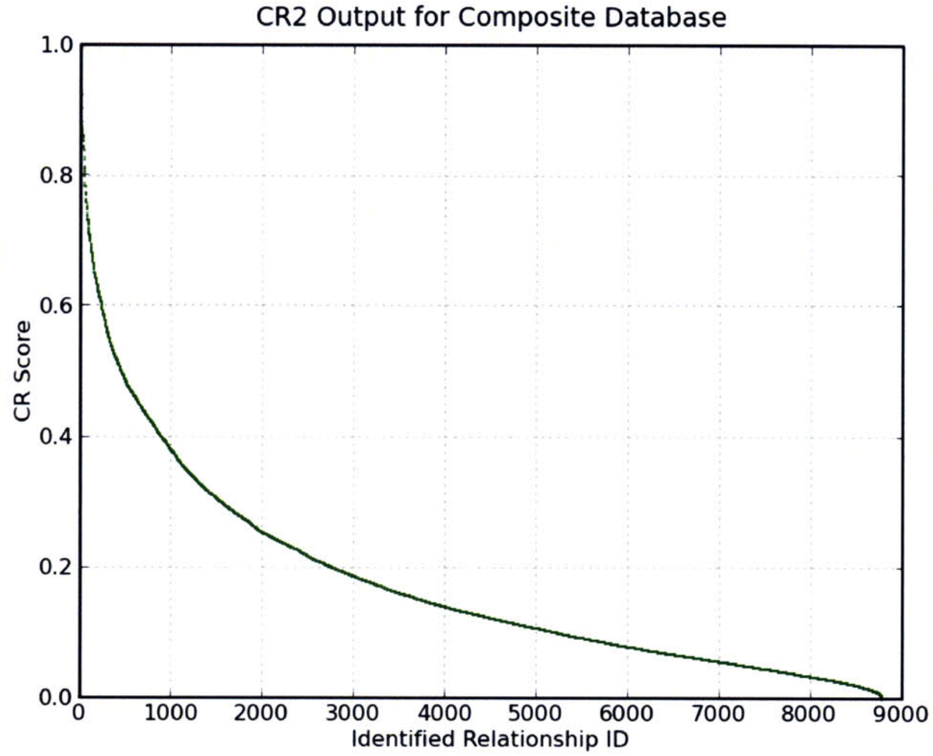


**Figure 6.1. CR2 Output for Composite Database**

This plot contains the output produced when CR2 was run on the composite database. Each point represents a relationship between a pair of variables, and it y-value is its CR2 score. Note that these results are sorted.

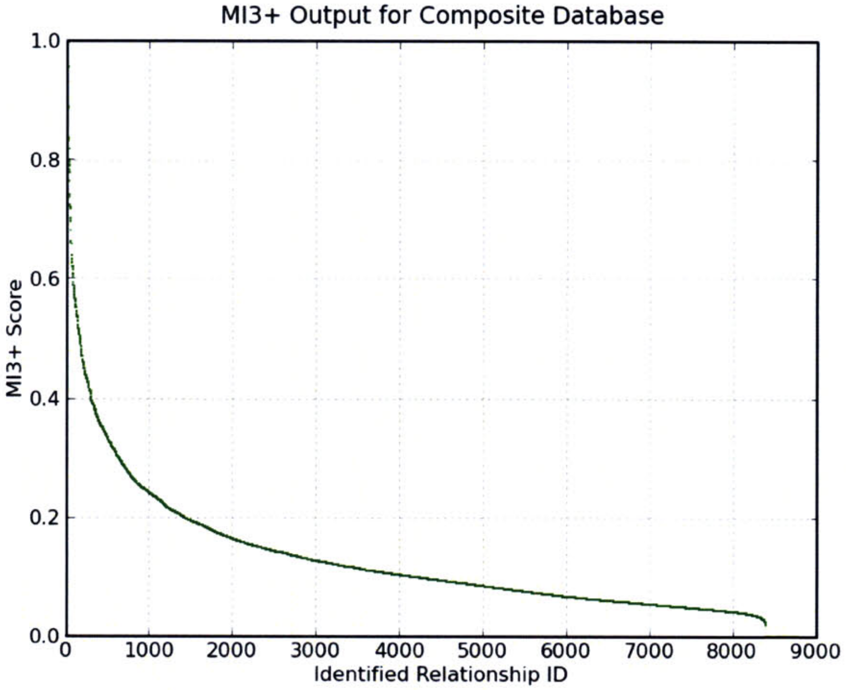


**Figure 6.2. MI3+ Output for Composite Database**

This plot contains the output produced when MI3+ was run on the composite database. Each point represents a relationship between a pair of variables, and it y-value is its MI3+ score. Note that these results are sorted.

### 6.3.2 Constructing a Gravity Graph

Both of these sets of results were then used to construct a Gravity Graph. Figure 6.3 below shows the construction of the Gravity Graph that was built using the CR2 output. This graph was constructed by adding relationships in rank-order by strength. As more relationships are added to the graph, clusters of interrelated variables begin to form. Several noticeable clusters in this graph include:

1. An 'educational cluster' containing various variables like literacy rates, the rate of children that complete primary school, 4th grade math achievement, measles immunization rate(!), etc.
2. An 'energy cluster' containing variables like $CO_2$ emissions, oil production, natural gas reserves, urban population percentage, number of roads paved, population growth, etc.

3. Several different 'health clusters'. One in particular includes variables like infant mortality rate, the number of children per woman, life expectancy, breast cancer rate, the number of births attended by a skilled person, urban population growth, the number of medical doctors, etc.

These are just a few of the various groups of variables that begin to form. In the interest of relating to public health, the next section will explore the 'health cluster' listed above.

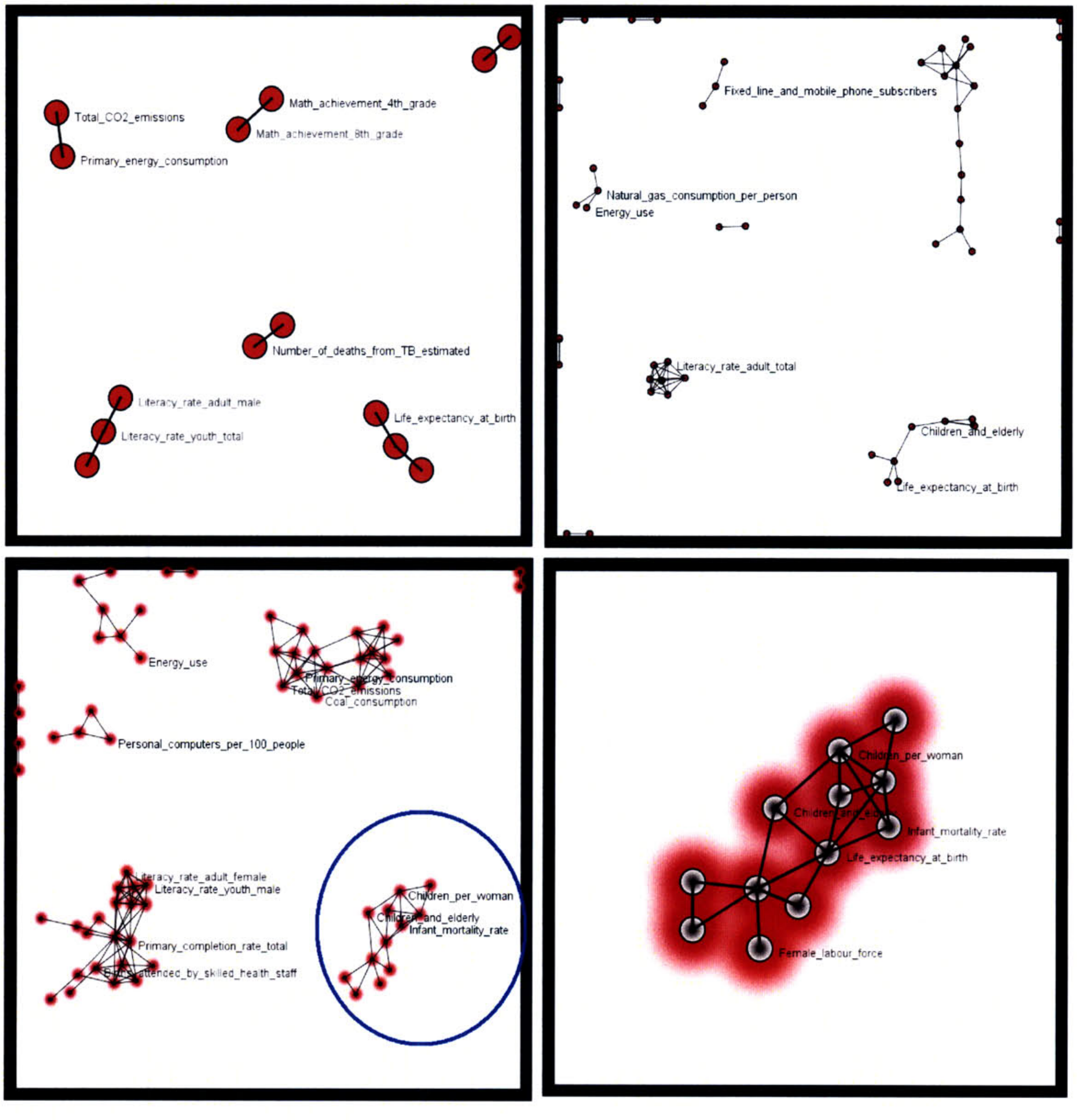

Total_CO2_emissions
Primary_energy_consumption
Math_achievement_4th_grade
Math_achievement_8th_grade
Number_of_deaths_from_TB_estimated
Literacy_rate_adult_male
Literacy_rate_youth_total
Life_expectancy_at_birth
Fixed_line_and_mobile_phone_subscribers
Natural_gas_consumption_per_person
Energy_use
Literacy_rate_adult_total
Children_and_elderly
Life_expectancy_at_birth
Energy_use
Primary_energy_consumption
Total_CO2_emissions
Coal_consumption
Personal_computers_per_100_people
Literacy_rate_adult_female
Literacy_rate_youth_male
Primary_completion_rate_total
Births_attended_by_skilled_health_staff
Children_per_woman
Children_and_elderly
Infant_mortality_rate
Children_per_woman
Children_and_elderly
Infant_mortality_rate
Life_expectancy_at_birth
Female_labour_force

**Figure 6.3. The Construction of the Gravity Graph for the Composite Database**
These four snapshots summarize the construction of the Gravity Graph using CR2 output. In the top left image, only the first several relationships have been inserted. The top right image shows the interactive graph after several more variables and relationships have been introduced into the graph. The lower left image shows the graph when even more variables and relationships have entered the picture. At this point, it is easy to identify the three major clusters that are listed in Section 6.3.2. The cluster of health-related variables that is selected for further examination is circled in blue, and shown in detail in the lower right image. Note that these variables are very correlated (close to pure red gravitational glow).

## *6.4 Exploration using the Visualization Environment*

A Scatter View is a good general-purpose view that can be used to explore the cluster of variables selected using the Gravity Graph. CR2 reports that there is a particularly strong relationship between the mortality rate of children under five years of age in a given country and the average number of children born to each woman in that country. These two variables can be loaded onto a Scatter View. The Scatter View will plot a bubble (circle) to represent each country, and the geographic variable "Continent" can be loaded onto the Color encoding in order to color the countries according to their geographic region. Finally, the temporal variable "Year" can be loaded onto the Global Scroll, in order to provide the ability to 'animate' the visualization by playing forward and backward through time. Thus far, five variables are encoded (country, continent, time, children per woman, and under five mortality rate).

Figure 6.4 below, which re-creates analysis performed by statistician Hans Rosling, contains snapshots of this visualization (H. Rosling, 2007). In comparing the two plots, it is evident that in 1961, the industrialized countries (mostly only Europe and North America) had fewer than four children per woman, and an infant mortality rate of less than about 12%. However, developing countries had on average at least five children per woman, and an infant mortality rate of anywhere between 12 and 40%. The points on the plot are animated when the "play" button on the View Controls is pressed and it is possible to literally watch the data trends between these variables play out over time. In 2005, the world looks completely different. Almost all the countries around the world have migrated into the lower left corner (low infant mortality, few children per woman) previously only occupied by Europe and North America. The extremely noticeable exception is Africa, which has started to migrate toward the lower left, but is lagging behind the others.

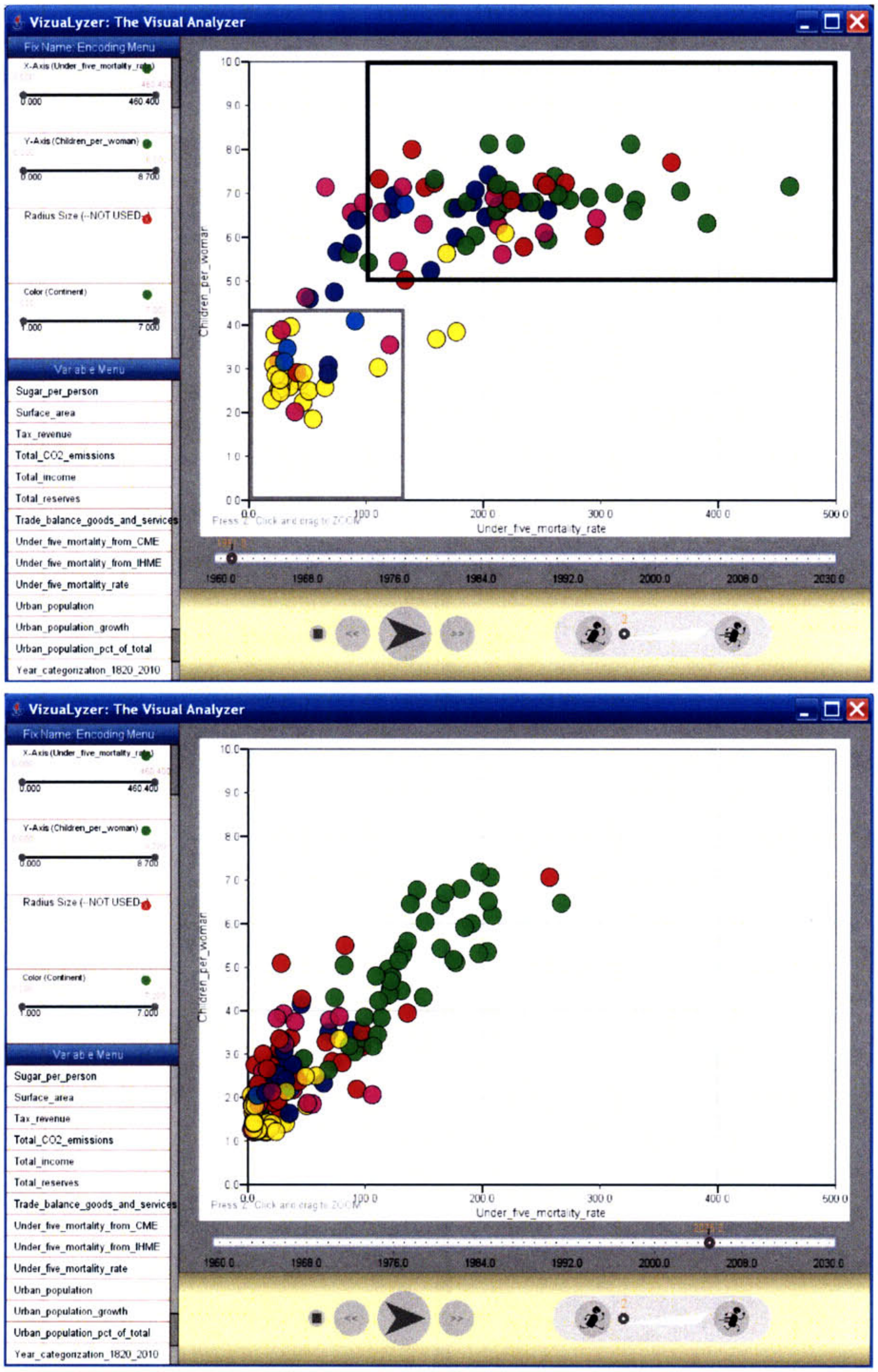
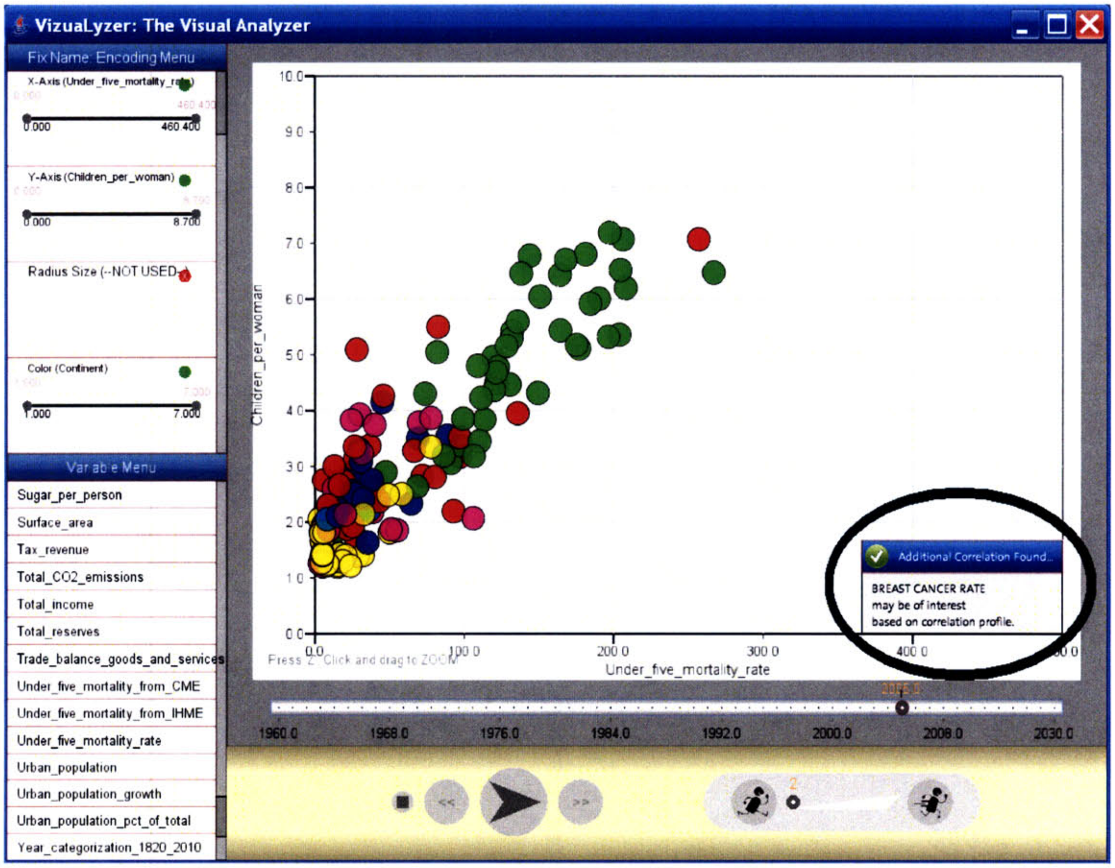

VizuaLyzer: The Visual Analyzer
Fix Name: Encoding Menu
X-Axis (Under_five_mortality_rate)
0.000
460.400
Y-Axis (Children_per_woman)
0.000
8.700
Radius Size (--NOT USED--)
Color (Continent)
1.000
7.000
Variable Menu
Sugar_per_person
Surface_area
Tax_revenue
Total_CO2_emissions
Total_income
Total_reserves
Trade_balance_goods_and_services
Under_five_mortality_from_CME
Under_five_mortality_from_IHME
Under_five_mortality_rate
Urban_population
Urban_population_growth
Urban_population_pct_of_total
Year_categorization_1820_2010
Children_per_woman
Under_five_mortality_rate
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
100.0
200.0
300.0
400.0
500.0
Press 'Z' Click and drag to ZOOM
1960.0
1968.0
1976.0
1984.0
1992.0
2000.0
2008.0
2030.0
<<
>>

**Figure 6.4. Scatter View – Infant Mortality and Children per Woman in 1961 and 2005**
These are two snapshots of the "Scatter View." Each point is a country, whose color indicates its continent (purple = Middle East & North Africa, yellow = Europe and central Asia, green = sub-Saharan Africa, cyan = North America, dark blue = South America, red = Oceania). Each point's position is determined by the infant mortality rate for children under five (x-axis) and the number of children per woman (y-axis) in the given country, for a given year. In the top snapshot (year = 1961) the industrialized countries (mostly only Europe and North America) had fewer than four children per woman, and an infant mortality rate of less than about 12% (gray box in the lower-left corner). However, developing countries had at least five children per woman, and an infant mortality rate of anywhere between 12 and 40% (black box in the upper right). Forty years later, the bottom snapshot shows a completely different picture. Almost all the countries around the world have migrated into the lower left corner previously only occupied mostly by countries in Europe and North America. The extremely noticeable exception is Africa, which has started to migrate toward the lower left, but is lagging behind the others.

## *6.5 Intelligent Hypothesis Generation: The Relationship Suggestor*

Based on the usage of the composite database over multiple visualizations, VisuaLyzer's Relationship Suggestor can intelligently recommend relationships that may be of interest given the variables that are currently being visualized. During the visualization shown in Figure 6.4, the Relationship Suggestor's collaborative filtering algorithm suggested two new relationships for exploration: a country's life expectancy at birth, and the rate of new cases of breast cancer per country (see Figure 6.5). Note that of these two variables, only life expectancy was included in the cluster identified in Section 6.4. This indicates that in recommending a variable that was not clustered with the variables currently being visualized, the collaborative filtering algorithm takes advantage of the human intuition used to create visualizations in making its suggestions, which CR2 and MI3+ could not have done.


VizuaLyzer: The Visual Analyzer
Fix Name: Encoding Menu
X-Axis (Under_five_mortality_rate)
0.000
460.400
Y-Axis (Children_per_woman)
0.000
8.700
Radius Size (--NOT USED--)
Color (Continent)
1.000
7.000
Variable Menu
Sugar_per_person
Surface_area
Tax_revenue
Total_CO2_emissions
Total_income
Total_reserves
Trade_balance_goods_and_services
Under_five_mortality_from_CME
Under_five_mortality_from_IHME
Under_five_mortality_rate
Urban_population
Urban_population_growth
Urban_population_pct_of_total
Year_categorization_1820_2010
Children_per_woman
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
0.0
100.0
200.0
300.0
400.0
Under_five_mortality_rate
Press 'Z' Click and drag to ZOOM
2005.0
1960.0
1968.0
1976.0
1984.0
1992.0
2000.0
2008.0
2030.0
<<
>>
2

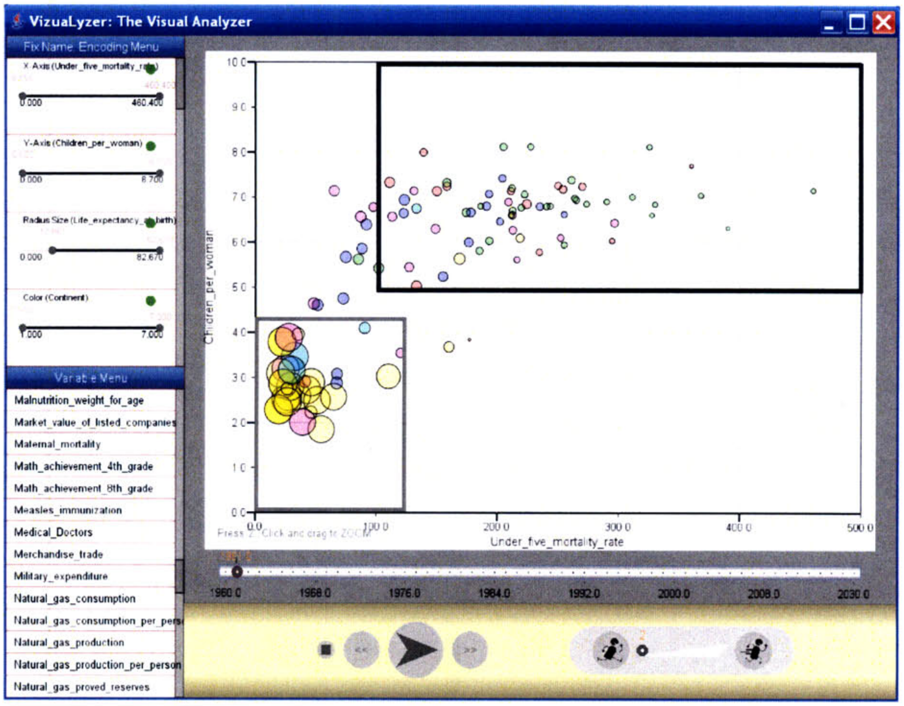
Additional Correlation Found...

BREAST CANCER RATE
may be of interest
based on correlation profile.

**Figure 6.5. Relationship Suggestor Recommending a New Relationship for Exploration (Relationship Alert Circled in Black)**

To explore these relationships in relation to our current visualization, they can simply be loaded onto a free encoding. In this case, Figure 6.6 shows the same Scatter View that was shown in Figure 6.4, but with the life expectancy variable loaded onto the Radius Size encoding (the size of each country's bubble is proportional to its average life expectancy), and the rate of new cases of breast cancer is loaded onto the Opacity encoding (the darker a country's bubble is, the higher the rate of breast cancer in that country). Note that in 1961, the industrialized countries (in the lower-left corner) all have relatively longer life expectancies, while the developing counties, which have on average more children per woman, all have relatively shorter life expectancies. Furthermore, the rate of breast cancer is relatively low for most countries (there are no extremely dark bubbles). In 2002, as the developing countries moved down into the lower-left corner by decreasing their infant mortality rates and average children per woman, they also began having longer life expectancies (their bubbles got larger). Furthermore, as most countries moved into the lower-left corner, their bubbles became darker, indicating growing breast cancer rates. This more complex Scatter View is allowing the exploration of seven dimensions simultaneously, and yet the patterns in the data are still visually striking.

Figure 6.7 shows the same visualization as that shown in Figure 6.6, but with a maximum filter placed on the X-Axis encoding, which zooms in on the lower-left corner of the plot in order to allow better differentiation between countries.


VizuaLyzer: The Visual Analyzer
Fix Name Encoding Menu
X-Axis (Under_five_mortality_rate)
0.000
460.400
Y-Axis (Children_per_woman)
0.000
8.700
Radius Size (Life_expectancy_at_birth)
0.000
82.670
Color (Continent)
1.000
7.000
Variable Menu
Malnutrition_weight_for_age
Market_value_of_listed_companies
Maternal_mortality
Math_achievement_4th_grade
Math_achievement_8th_grade
Measles_immunization
Medical_Doctors
Merchandise_trade
Military_expenditure
Natural_gas_consumption
Natural_gas_consumption_per_person
Natural_gas_production
Natural_gas_production_per_person
Natural_gas_proved_reserves
Children_per_woman
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
100.0
200.0
300.0
400.0
500.0
Under_five_mortality_rate
Press 'Z' Click and drag to ZOOM
1960.0
1968.0
1976.0
1984.0
1992.0
2000.0
2008.0
2030.0

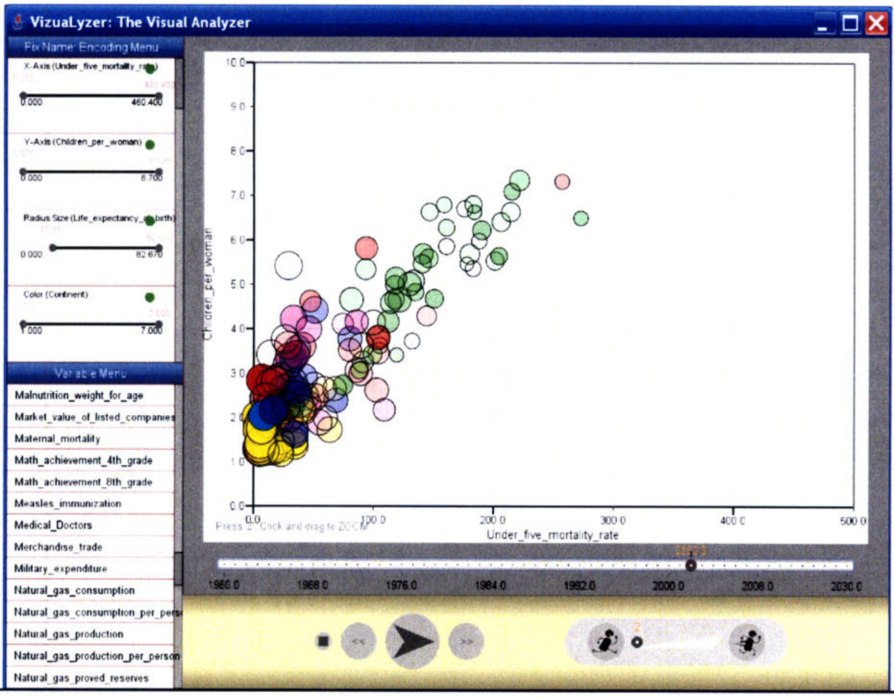

VizuaLyzer: The Visual Analyzer
Fix Name Encoding Menu
X-Axis (Under_five_mortality_rate)
0.000
460.400
Y-Axis (Children_per_woman)
0.000
8.700
Radius Size (Life_expectancy_at_birth)
0.000
82.670
Color (Continent)
1.000
7.000
Variable Menu
Malnutrition_weight_for_age
Market_value_of_listed_companies
Maternal_mortality
Math_achievement_4th_grade
Math_achievement_8th_grade
Measles_immunization
Medical_Doctors
Merchandise_trade
Military_expenditure
Natural_gas_consumption
Natural_gas_consumption_per_person
Natural_gas_production
Natural_gas_production_per_person
Natural_gas_proved_reserves
Children_per_woman
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
100.0
200.0
300.0
400.0
500.0
Under_five_mortality_rate
Press 'Z' Click and drag to ZOOM
1960.0
1968.0
1976.0
1984.0
1992.0
2000.0
2008.0
2030.0

**Figure 6.6. Scatter View – Infant Mortality, Children per Woman, Life Expectancy, and Breast Cancer Rate in 1961 and 2002**

These are two snapshots of the same "Scatter View" shown in Figure 6.4. Each point is a country, whose color indicates its continent (purple = Middle East & North Africa, yellow = Europe and central Asia, green = sub-Saharan Africa, cyan = North America, dark blue = South America, red = Oceania). Each point's position is determined by the infant mortality rate for children under five (x-axis) and the number of children per woman (y-axis) in the given country, for a given year. Furthermore, the point's size is proportional to the life expectancy of the corresponding country and its opacity is proportional to the rate of new breast cancer cases in the corresponding country. Note that in the top snapshot (1961), the industrialized countries (in the lower-left corner) all have relatively longer life expectancies, while the developing counties, which have on average more children per woman, all have relatively shorter life expectancies. Furthermore, the rate of breast cancer is relatively low for most countries (there are no extremely dark bubbles). In the bottom snapshot (2002), as the developing countries moved down into the lower-left corner by decreasing their infant mortality rates and average children per woman, they also began having longer life expectancies (their bubbles got larger). Furthermore, as most countries moved into the lower-left corner, their bubbles became darker, indicating growing breast cancer rates.

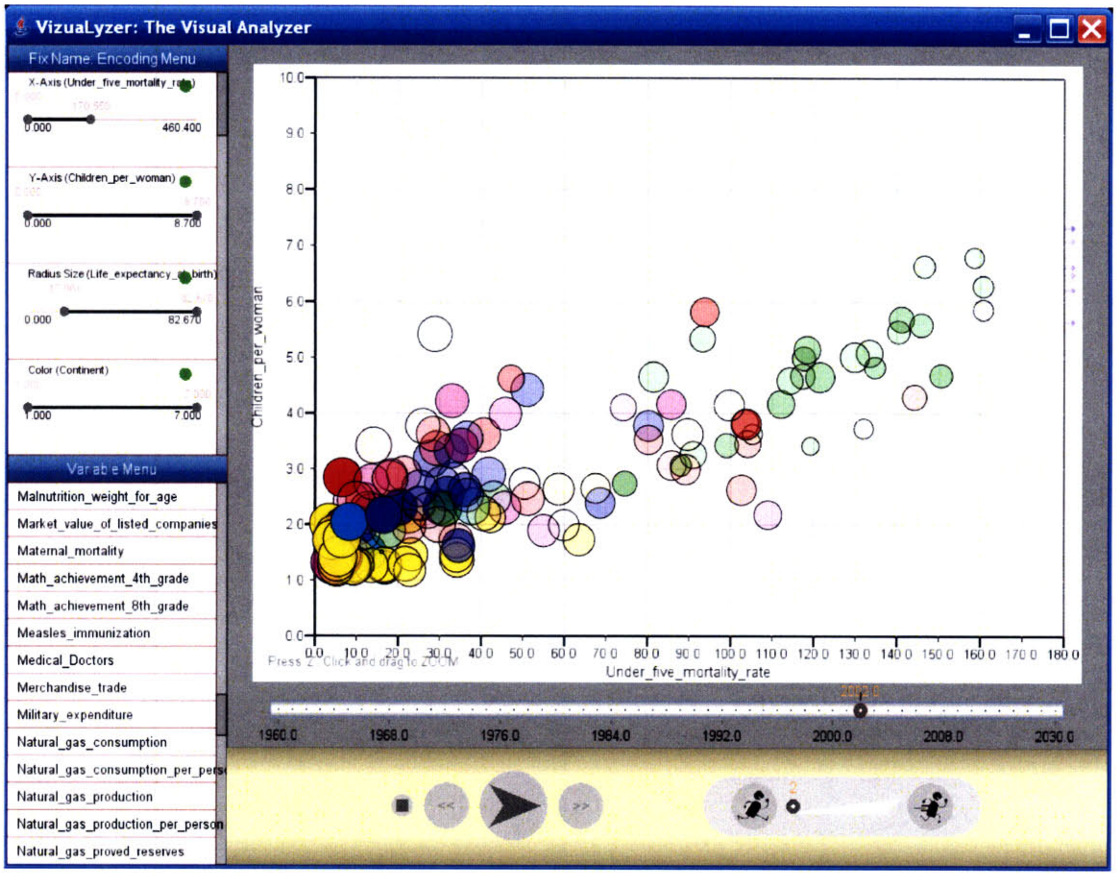


**Figure 6.7. Scatter View Zoom – Infant Mortality, Children per Woman, Life Expectancy, and Breast Cancer Rate in 2002**
This snapshot is of the same Scatter View shown in Figure 6.6, but with a filter placed on the X-Axis encoding, which effectively selects only the lower-left corner of the plot for visualization (zoom). This allows better differentiation between countries in that crowded region. Also note the arrows on the right of the plot which point to the countries that are outside the selected zoom region.

## 6.6 The Result

Exploring just one avenue of this complex database through the VisuaLyzer approach has elucidated so many relationships and inspired even more questions. The analysis presented in this case study was completed in approximately ten minutes of exploration with the system, which gives an idea just how rapidly this tool can mine data. The idea of going through the process of exploring a dataset using the VisuaLyzer system is that even if there is a preconceived result in mind, the tool will force critical thinking about aspects of the dataset that had previously never been considered.

# Chapter 7: Reflections on the VisuaLyzer Approach

The VisuaLyzer platform is designed as an intuitive and intelligent tool for simultaneously managing, analyzing, visualizing, and exploring massive amounts of data. Its ability to manipulate and present large amounts of data in an easily-accessible, interactive, graphic format allows for the rapid identification of significant or interesting relationships among variables. Moreover, it is the VisuaLyzer's seamless blend of analytic and visual environments that makes it such a powerful investigative tool.

The VisuaLyzer approach of identifying *all* relationships within a dataset in a computationally rigorous fashion and thereafter investigating interesting ones in a guided manner represents a fundamentally new approach to data analysis. Traditionally, the idea of blindly searching for correlations within a dataset seems like nonsense. Scientists have been conventionally trained that correlation is not causation, that no conclusion should be drawn simply on the basis of correlation between variables X and Y. After all, it could just be a coincidence. Rather the scientific method dictates that scientists should understand underlying mechanisms, establish a model, then test a hypothesis. However, using cutting-edge tools and principles similar to the VisuaLyzer approach, it is possible to generate and prioritize interesting hypotheses infinitely faster than ever before. By the principles of the data age, correlation instead of causation becomes the *entry* point to science, causing a shift in paradigm from hypothesis-driven science to hypothesis generating science.

## *7.1 This is not a Tool for Answers, but for Exploration*

VisuaLyzer is fundamentally a tool for asking questions of data. It allows users to explore data with extraordinary flexibility and breadth. However, the VisuaLyzer approach is designed for exploring data, rather than for calculating answers and coming to conclusions. The platform is intended to be used for searching the space of *all* relationships that exist within a dataset to identify and explore those which are particularly interesting. The exploration of these relationships is intended to identify patterns and trends that might inspire hypotheses about the underlying scientific mechanisms, which can be tested. The idea behind this approach is that by asking enough questions, eventually the right questions will be asked.

## *7.2 Using VisuaLyzer to Inspire Analysis*

The questions generated about data using the VisuaLyzer approach can help drive the process of analyzing data. As a primary example, the VisuaLyzer platform is in use by the CDC in numerous disease studies including gonorrhea, influenza, and the recent emerging swine flu epidemic. Regarding gonorrhea, the platform's exploratory capability helped generate hypotheses that informed models of the spread of drug-resistant strains of gonorrhea. By exploring the temporospatial dynamics of the interactions of several drug-resistant strains of gonorrhea, as well as the relationships between these strains and other potential intrinsic and extrinsic factors, the CDC is generating ideas regarding intelligent interventions to stop their spread. In a more time critical scenario, the VisuaLyzer's rapid approach toward data exploration is being employed for understanding and approximating the severity of the recent outbreak of swine flu. Again, the expectation is that this type of rapid exploration will generate interesting hypotheses, which can be further investigated. Results from the swine flu analysis are not presented in this thesis as the analysis is still ongoing.

## *7.3. The VisuaLyzer Approach in Epidemiology and Public Health*

With an increasing number of epidemiological studies being performed around the world, and the rapidly accelerating ability to capture and store immense datasets of all kinds, the wealth of collected data in public health has become overwhelming. In turn, the discovery of potential disease-defining factors from this data can be unintuitive and has become computationally difficult.

Using novel tools like VisuaLyzer enables analyzing data without hypotheses about what it might show, without being restricted to preconceived models. Imagine the possibility of applying powerful computing clusters to the rapidly growing data streams in public health and letting statistical algorithms find patterns where science cannot. How about using such patterns to generate multiple parallel hypotheses instead of sequentially exploring individual hypotheses to derive a common pattern? This type of approach will develop highly targeted evidence-based public health interventions faster than ever before. More remarkably, VisuaLyzer and similar tool transform today's massive corpus of data into a laboratory for the human condition, emerging as "threat detective" capable of predicting and preventing outbreaks before they erupt.

# References:

Arvind Narayanan and Vitaly Shmatikov. How to break anonymity of the netflix prize dataset, October 2006. arxiv cs/0610105.

C-Store: A column-oriented DBMS, Stonebraker et al, Proceedings of the 31st VLDB Conference, Trondheim, Norway, 2005

Donnelly, C. A., A. C. Ghani, et al. (2003). "Epidemiological determinants of spread of causal agent of severe acute respiratory syndrome in Hong Kong." Lancet 361(9371): 1761-6.

Fry, B. 2007. Visualizing Data. Sebastopol, CA: O'Reilly Media, Inc.

G. Linden, B. Smith and J. York, Amazon.com Recommendations: Item-to-Item Collaborative Filtering, IEEE Internet Computing, Jan./Feb. 2003.

H. Rosling. Gapminder. Http://www.gapminder.com. January 2008.

Imagecreator. "An example of the correlation of x and y for various distributions of (x,y) pairs." Mathematica 6. 2007-12-08.

Juan Liu; Moulin, P., "Information-theoretic analysis of interscale and intrascale dependencies between image wavelet coefficients," Image Processing, IEEE Transactions on , vol.10, no.11, pp.1647-1658, Nov 2001

Keeling, M. J., M. E. Woolhouse, et al. (2001). "Dynamics of the 2001 UK foot and mouth epidemic: stochastic dispersal in a heterogeneous landscape." Science 294(5543): 813-7.

Lipsitch, M., T. Cohen, et al. (2003). "Transmission dynamics and control of severe acute respiratory syndrome." Science 300(5627): 1966-70.

Rapp,P.E., Zimmerman,I.D., Vining,E.P., Cohen,N., Albano,A.M. and Jim´enez-Monta˜no,M.A. (1994) The algorithmic complexity of neural spike trains increases during focal seizures. J. Neurosci., 14, 4731–4739.

Riley, S., C. Fraser, et al. (2003). "Transmission dynamics of the etiological agent of SARS in Hong Kong: impact of public health interventions." Science 300(5627): 1961-6.

Schreiber,T. and Schmitz,A. (2000) Surrogate time series. Physica D, 142, 346–382.
Silverman,B.W. (1986) Density Estimation for Statistics and Data Analysis. Chapman and Hall, London.

Steuer, R., Kurths, J., Daub, C.O., Weise, J. & Selbig, J. The mutual information: Detecting and evaluating dependencies between variables. Bioinformatics 18 Suppl 2: S231–S240 (2002).

United Nations data. http://data.un.org/ March, 2009.

Wallinga, J. and P. Teunis (2004). "Different epidemic curves for severe acute respiratory syndrome reveal similar impacts of control measures." Am J Epidemiol 160(6): 509-16.

World Health Organization Statisical Information System (WHOSIS). http://www.who.int/whosis/en/index.html. April 2009.