NeuroVis: exploring interaction techniques by combining dimensional stacking and pixelization to visualize multidimensional multivariate data

Category: Research

ABSTRACT

Applying a dimensional stacking layout to multidimensional pixelization images yields a highly informative visualization with new possibilities for interactive techniques such as dimension reordering and ad hoc database queries. Pixelization is the mapping of each data point in some set to a pixel in an image. Dimensional stacking is a layout method where N dimensions are projected into 2. In this paper we present NeuroVis, an application that combines both methods to support interactive visual data mining of multidimensional multivariate data. An informal evaluation of how it has been used in neuroscience investigations and classrooms is included. We also provide a generalization of the technique for use in other domains and discuss its utility and limitations.

CR Categories: I.3.6 [Computing Methodologies]: Methodology and Techniques—Interaction Techniques; I.6.6 [Computing Methodologies]: Simulation and Modeling—Simulation Output Analysis J.3 [Computer Applications]: Life and Medical Sciences

Keywords: Visual data mining, Interaction, Large Data Set Visualization, Hierarchy visualization, High-dimensional data

1 INTRODUCTION

Pixelization is the mapping of each data point in some set to a pixel in an image. Dimensional stacking is a layout method where N dimensions are projected into two. NeuroVis combines both methods to support visual data mining and new possibilities for interaction techniques. In the resulting tool, pixels provide a type of "click-through" access to underlying databases and applications. This is in contrast to many tools which usually provide indirect access to data sources through the visualization interface. NeuroVis also provides transparent database access by allowing users to select color mappings to data attributes through open ended SQL queries. Result sets can consist of subsets of the data being visualized, or even Real values that result from computations on the dataset. Subsets are mapped to a single color while Reals are mapped to a color gradient. While this requires the user to be familiar with SQL syntax, it provides a much greater flexibility than many visualization tools which limit queries to a type of set membership test.

NeuroVis also utilizes dimension reordering in an unconventional way. Previous work has largely focused on clutter reduction or optimal clustering [**?**][**?**] [**?**]. We have found that different dimension orders in dimensional stacking and pixelization reveal different information, and that there is not necessarily one optimal ordering. Also, clustering based on similarity is not quite applicable to this scheme. Section **??** discusses how NeuroVis supports interactive dimension reordering to provide visual data mining.

In this paper we focus on how NeuroVis has been applied to the neuroscience domain (thus the name). We include an informal evaluation of its use in neuroscience investigations and classrooms. Section **??** provides a generalization of the technique for use in other domains along with a discussion of its limitations.

2 THE DATA

NeuroVis began as a collaboration between researchers in the neuroscience and computer science departments of Brandeis University. A database was constructed to better understand the role of a neuron's individual membrane currents in shaping the dynamics of membrane potential, the signal neurons use to encode information and communicate. Data from several simulations of a model neuron was classified according to electrical activity and recorded. The impetus for NeuroVis was to provide a tool for the visualization and analysis of this database. The following two subsections describe the data from a high level computer science perspective and a more detailed neuroscience perspective respectively. The final subsection provides the motivation for applying visualization in this domain.

2.1 The Computer Science

The model neuron used in constructing the database consisted of 8 conductance values or parameters, each of which could take on 6 possible values. Every possible combination of parameter values was run through a simulator yielding 1,679,616 simulations. Various characteristics were recorded for each simulation and stored in a database. To use the terminologies of Stolte [**?**] the conductances or parameters could be considered independent variables and were viewed as dimensions of the model neuron conductance space. The recorded characteristics of each simulation could be considered dependent variables or "measures". One table in the database held the dimensions as columns. In all tables of the database, the parameter values were concatenated, converted to decimal, and used as a primary key. For instance, the row with primary key 322209 refers to a model neuron simulation with the parameter values 10523413. The resulting database was comprised of many tables, some of which only contained rows for simulations with certain characteristics.

2.2 The Neuroscience

Simulations were conducted with a model neuron consisting of a single isopotential electrical compartment enclosed by a cell membrane with eight voltage-dependent ion currents flowing through channels in the membrane:

- a Na^+ current, I_{Na};
- a fast Ca^{2+} current I_{CaT}
- a slow Ca^{2+} current, I_{CaS};
- a transient K^+ current, I_A;
- a Ca^{2+}-dependent K^+ current, I_{KCa},
- a delayed rectifier K^+ current, I_{Kd};
- a hyperpolarization-activated inward current, I_H;
- a leak current, I_{leak}.

The sum of all membrane currents determines the dynamics of the voltage V across the membrane according to

$$C \cdot dV/dt = -\sum_i \mathrm{I_i} - \mathrm{I_{input}}$$

where C is the electrical capacitance of the cell membrane and I_{input} is the input current. According to Ohm's law, each current is

$$I_i = G_i \cdot m_i^p \cdot h_i^q \cdot (V - E_i)$$

where G_i is the maximal conductance (corresponding to all ion channels that conduct I_i being open), the dynamic variables m_i and h_i (which are between 0 and 1) determine what percentage of channels is open, p and q are integer numbers, and E_i is the reversal potential of the current, which depends on the relative abundances of the ions underlying I_i inside and outside the membrane.

The variables m_i and h_i describe the voltage-dependent opening and closing of ion channels and change according to

$$\tau_x \cdot dx/dt = x_\infty - x$$

where x is the variable m_i or h_i, τ_x is a time constant, and x_∞ is the equilibrium value of x. Both τ_x and x_∞ typically have a sigmoid dependence on voltage. Further details and all model neuron parameters are available in [**?**].

To generate the database, the maximal conductances G_i of the eight membrane currents were varied independently while all other parameters of the model neuron were held constant. In a biological neuron, this would correspond to changing the relative amounts of ion channels of different types in the neuronal membrane without affecting how the opening and closing of the channels depends on voltage. In the database, each of the eight maximal conductances could take six different equidistant values between 0 and a physiologically meaningful maximal value.

The electrical activity of all possible combinations of G_i, corresponding to $6^8 = 1,679,616$ model neurons with different membrane current composition, was numerically simulated. During the simulation, local voltage maxima and minima and other characteristics were detected and saved in the database for each model neuron. Also, the response of all bursting neurons to brief inputs at different times during their ongoing oscillation was simulated and saved in the form of phase response curves. More details about the simulations and database components are available in [**?**]

Because only six different values were explored for each maximal conductance G_i, the database samples the eight-dimensional conductance space of the model neuron with a relatively sparse grid. The success of attempts at visualizing the distribution of different types of electrical activity in this conductance space depends on whether the sparse sampling captures the salient features of the distributions of silent, spiking, bursting, and non-periodic neurons (and of sub-categories of these groups) in conductance space. Previous statistical analysis indicates that the distributions of different types of activity in conductance space are indeed sufficiently well-behaved – i.e., contiguous and compact – to allow analysis with a sparse sampling grid [**?**]. Visualization of the database with Dimensional stacking or other techniques should therefore lead to interpretable results.

2.3 Why Visualize?

The simulator used in constructing the database contained a system of coupled, non-linear, differential equations (governing the dynamic variables V, m_i, and h_i) which are not analytically tractable. This provided the impetus to both construct the database and find a visualization to help understand it. The goal of visualization was to provide an insight into the conductance space and provide a type of visual data mining. Once features were discovered visually, analysts could then use traditional statistical techniques for elucidating the dynamics behind those patterns. Statistics could then motivate later visualizations and so on. The next section summarizes a number of techniques and software we reviewed as we set about constructing a visualization tool for the neuroscience database.

3 Related Research

There are a number of methods for multidimensional and/or multivariate visualization [**?**][**?**][**?**]. The following characteristics of our data figured heavily into our considerations:

- It was relatively large with 1,679,616 elements, each containing 2KB of data
- It was multidimensional (8 dimensions to be exact)
- It was in a database format

Most software and computers could not load our entire data set into working memory. For instance, XGobi was unable to load a table from our database without freezing [**?**]. Because of the data's size and format, it was loaded into a SQL database. This would provide standard querying capabilities and also serve as a back end for visualization tools.

Several traditional techniques were reviewed. The Parallel coordinates [**?**] approach was ruled out due to issues of obfuscation from overlapping lines. The number and size of Chernoff faces, icons, or glyphs required for our data set would require an inordinate amount of panning and scrolling [**?**]. One option was to construct a matrix of two dimensional scatter plots, where every combination of two conductance parameters was displayed. As Ward notes in [**?**], this approach may be relatively insensitive to the size of a dataset but complicates the discovery of data trends involving more than 2 dimensions. Also, the more dimensions there are, the more possible combinations of two, the more scatter plots, and the less screen real estate for each. When cross-referencing plots to determine functional dependencies between more than 2 dimensions, this can lead to confusion and inefficient panning and scrolling.

One method that has been quite successful at visualizing large amounts of data is called pixelization[**?**]. It is the simple yet powerful technique of mapping every data point of some set to a pixel in a two dimensional image. Keim has shown the effective use of pixelization in visualizing large datasets [**?**] and prescribes design imperatives in [**?**].

Keim's application VisDB implements pixelization and maps each dimension to a different window [**?**]. Layout is determined by space filling curves [**?**] and is the same for each window. To determine functional dependencies in the data, users compare collocated regions in the windows for each dimension of interest. This encounters issues similar to those of scatter plots in that it requires users to toggle between multiple displays when comparing multiple dimensions. The semantics of Keim's layout are that data items with similar values or "distances" from user queries are placed next to each other. It was our preference to let dimensions determine the axes of a visualization and thus the layout. Measures or attribute values are then seen in the context of the dimension values that determine them. This may be more relevant for our data set since the model neuron conductances or dimensions can be considered independent variables and all other columns of the database can be considered functionally dependent on their values. A favorable side effect is that only one frame is used for a visualization of the entire database, thus users can determine data trends without having to toggle between windows or scroll between scatter plots.

Our approach was to hierarchically arrange the dimensions on each axes. Stolte et. al. have extended Polaris to support visual data mining of hierarchical data [**?**]. Our data set however was not

intrinsically hierarchical, we were merely using a hierarchical arrangement to determine layout. In [**?**], Leblanc et. al. extend one dimensional hierarchical methods [**?**] to 2D. Called dimensional stacking, this approach is implemented in Ward's XmdvTool [**?**]. We decided to apply a dimensional stacking layout to pixelization. While LeBlanc et. al. make a subtle reference to the possibility of one data element being mapped to one pixel, this combination of methods has not been thoroughly investigated. We have found that combining both approaches uniquely facilitates visual data mining and data trend discovery. This is discussed in detail in the section entitled Interaction.

One of our primary objectives was to have a visualization that leveraged a database backend to support visual data mining and querying. Many of the software tools that use the term database actually operate on flat files. Those that do integrate with databases often provide indirect access through a graphical user interface. While this provides powerful supports for users who do not know query syntax, it limits flexibility for those who do. For instance, Polaris allows users to drag and drop DB columns to construct a visualization which implicitly executes a DB query. VisDB associates sliders with query parameters so that they can be adjusted in real time. This approach ends up imposing a constraint on queries so that they may only test for set membership (e.g. they must return a subset of the overall database and can not return real numbers resulting from a function executed on a certain column of all rows). Our goal was to provide as transparent an access as possible to the underlying database.

In the next section we provide a description of applying a dimensional stacking layout to a pixelization, of the neuroscience database. Later sections refer to related research efforts in the context of similarities and differences with our approach. This is most informative in the Interaction section which details the interaction techniques facilitated by our methods.

4 Visualization

This section presents our approach as it was applied to the model neuron simulation data. The section Generalization demonstrates how our approach can be applied in other domains. It also describes limitations of our methods in the context of what format of data they are most applicable to.

Each model neuron was uniquely defined by its conductance parameter values, an 8-tuple of integers in the range $[0,5]$. The conductances were KCa, Na, CaS, H, CaT, Kd, A, and leak and served as the dimensions of our data set. The entire space of possible parameter values and model neurons was therefore 8 dimensional. We reduced this to 2 dimensions by partitioning the parameters into 2 sets of 4 elements each, and viewing them as a pair of 4 digit base 6 numbers. These numbers served as the decimal coordinates of the pixel associated with the model neuron bearing those parameter values. More precisely, we chose a set of 4 conductances (independent variables) x_1, x_2, x_3, x_4 and computed a pixel's x coordinate as:

$x = x_1 * 6^3 + x_2 * 6^2 + x_3 * 6^1 + x_4 * 6^0$

The y coordinate was calculated in the same fashion using the remaining conductances. The multiplication by powers of 6 is due to the fact that each dimension has 6 possible values and thus a base of 6. The algorithm for deriving decimal values from dimensions with differing bases (or possible value cardinalities) is described in [**?**]. 6^8 model neurons mapped to 6^8 pixels. 8 dimensions partitioned into two groups of four created a 1296 by 1296 square image.

The above calculations result in an image where dimensions are hierarchically embedded within one another as shown in Figure **??**. Pixel coordinates for this image were determined from dimensions ordered on each axis as follows:

$X = CaS * 6^3 + A * 6^2 + Na * 6^1 + H * 6^0$

$Y = CaT * 6^3 + leak * 6^2 + Kd * 6^1 + KCa * 6^0$

In previous literature, dimensions were referred to as "faster" or "slower" [**?**]. We refer to the dimension associated with the most significant digit in one of the pixels' decimal coordinates as the most significant dimension for that axis. In Figures **??** and **??**, the most significant dimension on the X axis is CaS and the most significant dimension on the Y axis is CaT. The large blue square in the bottom left corner of Figure **??** is the area where CaT and CaS are 0. The purple squares are where leak and A are 0, the yellow square are where Na and Kd are 0, and the small grid-like red points are where KCa and H are 0.

The origin of our dimensional stacking images are in the bottom left corner. Values for each dimension increase going upwards and to the right. Figure **??** shows a dimensional stacking image of the resting potential of silent neurons in mV. Figure **??** is a zoomed in image of Figure **??** where $CaS = 0$ and $CaT = 1$. Figure **??** is a zoomed in image of Figure **??** where $A = 2$ and $leak = 0$. Figure **??** is a zoomed in image of Figure **??** where $Na = 0$ and $Kd = 0$.

Over a period of minutes or hours the conductances of a real neuron may change incrementally. With this layout, the neuroscientists were be able to trace the path of a single neuron through the conductance space as its activity changed. We believe similar analysis could be performed with other data sets, especially time series data. The next section describes how NeuroVis implemented this visualization technique to support visual data mining and exploratory analysis.

5 Interaction with NeuroVis

NeuroVis has been implemented in various programming languages including JScheme [**?**] and Java. The primary features are now converging in the Java version which uses JDBC to communicate with a MySQL database, however, can integrate with any relational database. Figure **??** shows NeuroVis displaying a zoomed out image of the resting potential of silent neurons. Going left to right, the buttons circled in red in Figure **??** are:

- pointer: this turns the mouse cursor back to the default pointer and deactivates any tools
- zoom in
- zoom out
- pan
- NeuroInfo
- Simulator

The NeuroInfo and Simulator tools are described below along with the ColorMapper. The View menu provides access to the ColorMapper which is shown in Figure **??**. The remaining buttons and menus are for basic application functions such as saving an image.

5.1 Query color specification

Users directly specify the colors associated with pixels by writing SQL queries. This is done using the ColorMapper shown in Figure **??**. Users can add or remove rows from the table of the ColorMapper. Clicking inside a cell on the left column of the table launches a text editor for entering a query. Clicking inside a cell on the right column launches a color chooser for selecting a color to be associated with the result set of the query in the cell to the left. If result sets intersect, colors specified lower in the table take precedence. The queries shown in Figure **??** created the color map for the images shown in Figures **??**, **??**, **??**, **??** and **??**. These figures represent both the same data and the same colormap; they only appear different because of zooming or differing dimension orders.

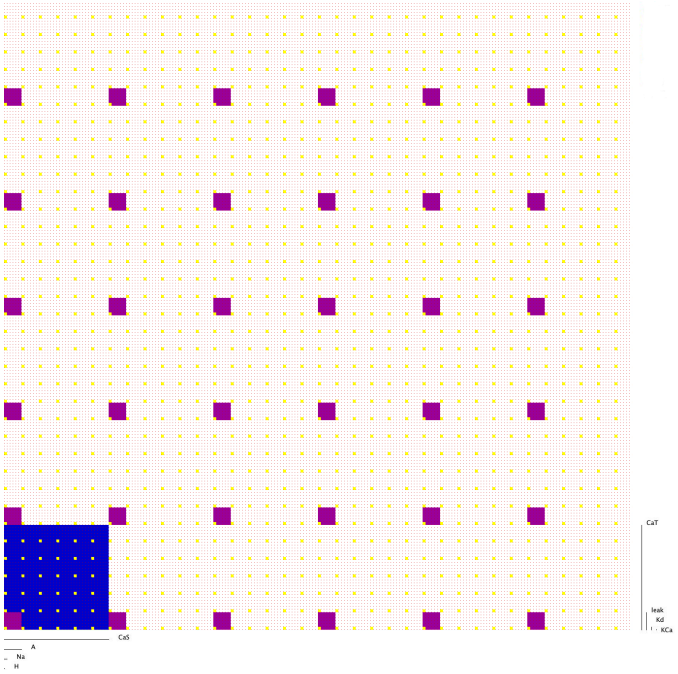

Figure 1: Dimensional stacking image with certain values of 0 in color.

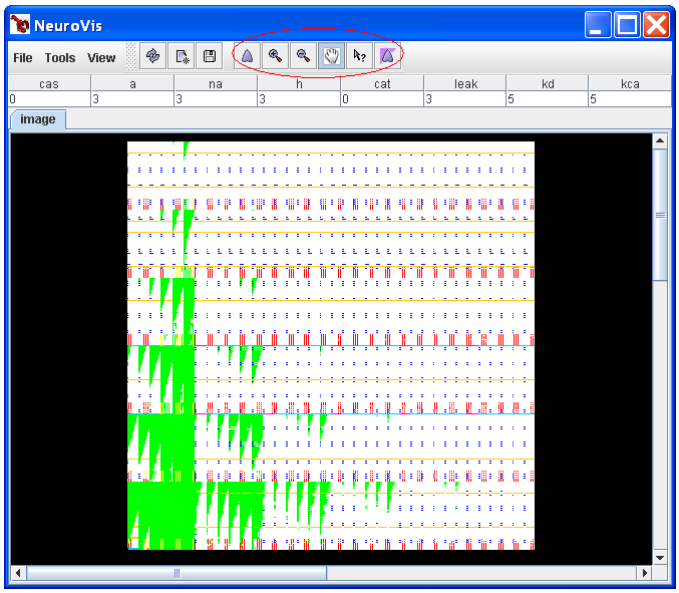

Figure 2: NeuroVis zoomed out, showing silent neuron resting potential in mV.

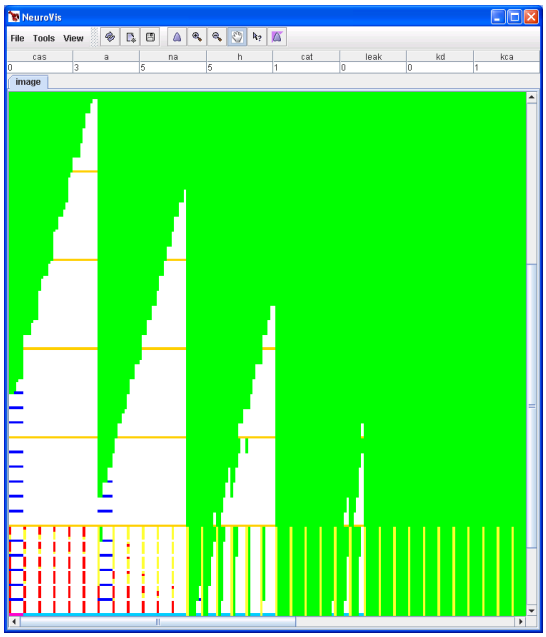

Figure 3: NeuroVis zoomed in to CaS = 0 and CaT = 1.

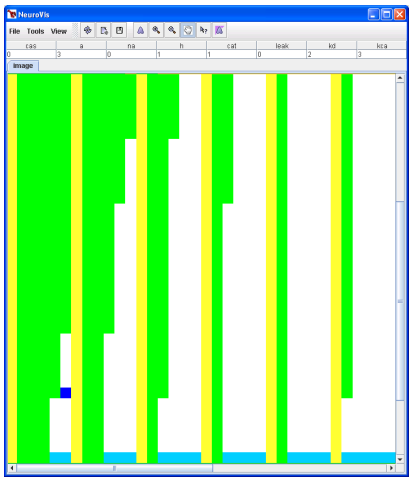

Figure 4: NeuroVis further zoomed in to A = 2 and leak = 0.

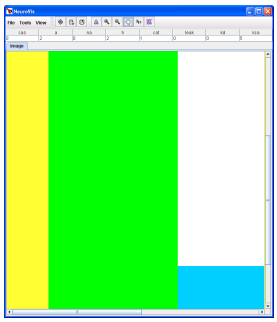

Figure 5: NeuroVis further zoomed in to Na = 0 and Kd = 0.

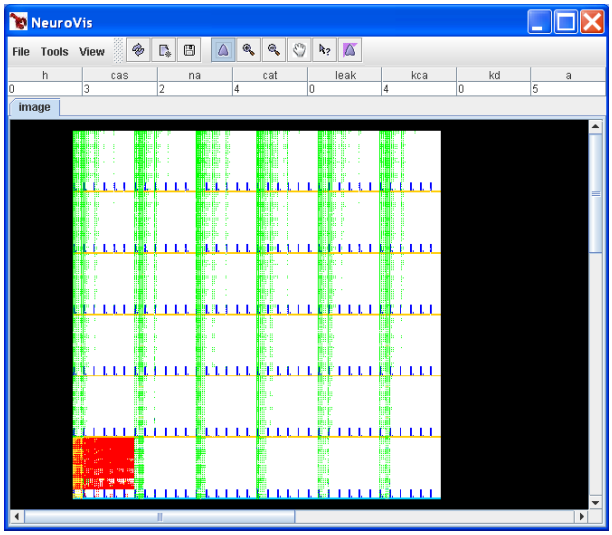

Figure 6: NeuroVis zoomed out, showing silent neuron resting potential in mV

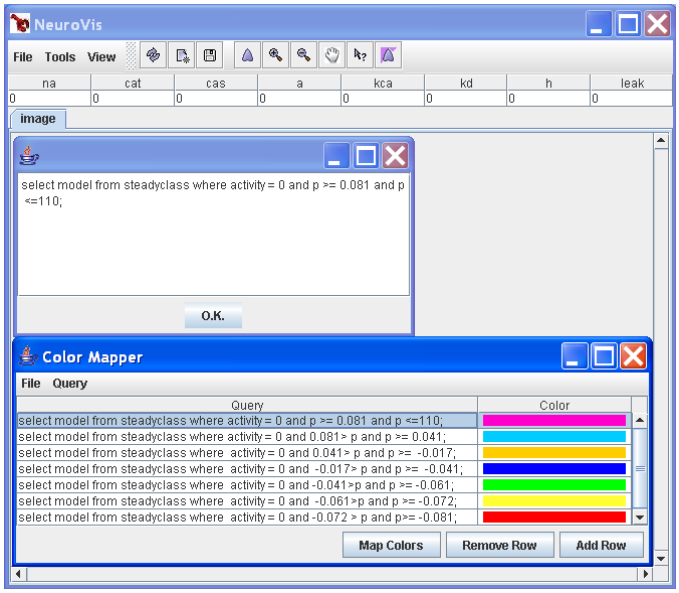

Figure 7: ColorMapper: Clicking a cell in the left column launches a text editor for typing queries. Clicking a cell in the right column launches a color chooser for associating a color with the result set of the query.

The JScheme version of NeuroVis also accepts queries that generate Real values and maps them to a color gradient (currently being ported to Java version). Figure **??** shows a gradient coloring of the silent neuron resting potential shown in Figures **??**, **??**, **??**, **??** and **??**. Its legend is discretized for readability.

5.2 Click-through data access

Each pixel in NeuroVis images is associated with a model neuron simulation. Pixel coordinates are determined by the dimension values fed in to each simulation. NeuroVis captures mouse events as the pointer moves over each pixel and reports the dimension values used in the simulation associated with each. The pixel can be considered a gateway to all information relevant to its associated simulation or data point in the dataset. This information is accessed by picking one of the click-through access tools, NeuroInfo or Simulator, and then clicking on a pixel/simulation/data point of interest.

5.2.1 NeuroInfo

The NeuroInfo tools executes a pre-written query on the data point associated with the pixel that a user clicks on. A user can select a visual cluster or region of interest and click various pixels within it to get a better understanding of the underlying data. Figure **??** shows the execution of a simple query that selects the conductance values for a data point. When the NeuroInfo tool is selected, this frame pops up every time the use clicks a pixel. There is no constraint on the type of query that can be specified. Again, this provides more flexibility than many visualization tools, however requires the user to be at least somewhat familiar with SQL syntax.

5.2.2 Model Neuron Simulator

The simulator that was used for generating the database was coded up in a lightweight JScheme component. Users can select the Simulator tool then click on any regions or pixels of interest to see the actual simulation associated with that data point. This has proven extremely useful for neuroscientists who happen upon interesting visual clusters and want to understand the simulations that are creating them. The Simulator frame is shown in Figure **??**.

5.3 Dimension Reordering

Because we had 8 dimensions, there were 8!/2 or approximately 20,000 possible projections or images. The division by 2 results from extracting identity flips, i.e. swapping each pixel's x and y coordinate effectively reflects the 1296 X 1296 image along a diagonal. This dynamic occurs in any dimension set with an even cardinality.

It is usually considered desirable to graphically cluster displays in order to show a distinct structure in the data being visualized [**?**][**?**][**?**]. However, the usual motives and methods for clustering are not always applicable to our combination of dimensional stacking and pixelization. For instance, Ankerst proposes clustering dimensions with similar attribute values [**?**], however in NeuroVis there is a difference between clustering dimension values and clustering the colored pixels in an image. In general, the most significant dimensions determine the visual clustering of an image.

Much of the previous work on dimension reordering has focused on clutter reduction or making a visualization more discernible [**?**][**?**]. We have found that there is not necessarily a best order for dimensions in a dimensionally stacked pixelization. Using NeuroVis, different dimension orders reveal different types of information. By interactively reordering dimensions, users can visually mine data and get a better understanding of the parameter space. Extensions to Polaris have come close to capturing this dynamic by supporting "pivoting" the axes of a visualization and drilling up and

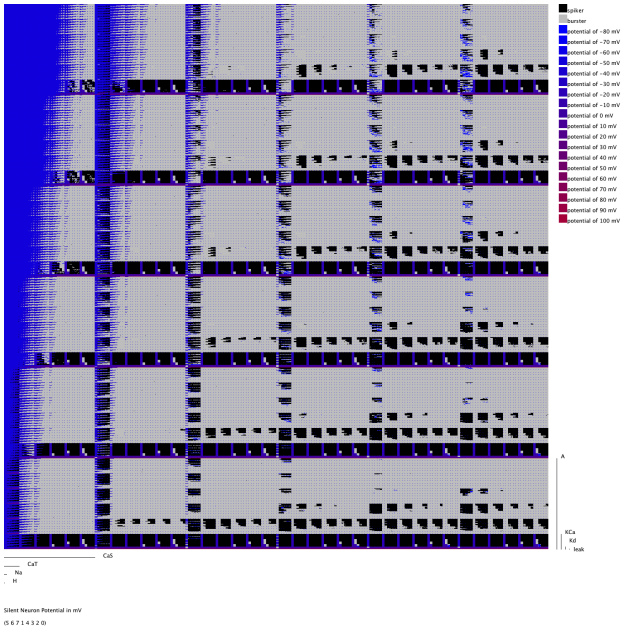

Figure 8: Gradient coloring of dimensional stacking image of resting potential of silent neurons

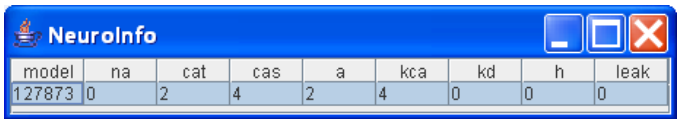

Figure 9: Frame displaying a query result set after NeuroInfo tool was selected and user clicked on a pixel.

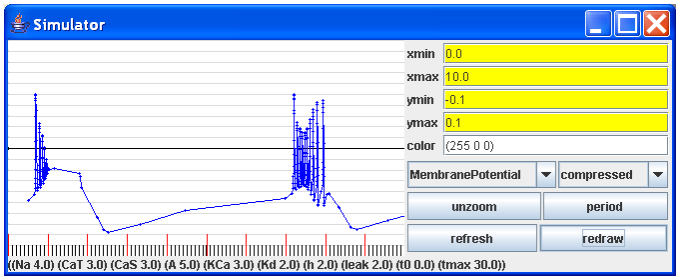

Figure 10: Frame displaying a neuron model simulation associated with one of the pixels/data points.

down the data hierarchy [**?**]. However, it does not specifically support dimension reordering which would be analogous to switching the order of set elements on one hierarchy level.

For instance, Figures **??** and **??** both show the same data and the same color map. The only different is the dimension orders. Using the dimension ordering in Figure **??**, investigators were able to derive a linear constraint for the green pixels (and their associated data point regions):

$6 * Leak + 6 * A + Kd + KCa >= 2 + 24 * CaS + 10 * CaT + 3 * Na + H$

Using the dimension ordering in Figure **??**, investigators were able to derive linear constraints for the red pixels:

$H < 1, Leak < 1$

There is no global optimal dimension order but several local maxima that show a particular clustering, each revealing different information about the underlying data. In general, the most significant dimensions determine the clustering of an image. For instance, when H and leak are ordered as the most significant dimensions, a red square appears where their values are 0, the lower left corner of Figure **??**. The query associated with red pixels, shown in Figure **??**, selects neurons with a resting potential between -80 mV and -73 mV. We can therefore deduce that low values of H and leak lead to a resting potential between -80 mV and -73 mV. This is analogous to a type of visual principle components analysis, where reordering dimensions can reveal not only data clusters, but what dimensions play a central role in creating them.

In various implementations we have combined manual dimension reordering with algorithms for automatically finding hyper planes and local maxima. This can be considered a type of mixed initiative, simulated annealing when following this usage pattern:

- manually specify a dimension order
- use tool to automatically find nearest local maxima
- reorder dimensions manually to get closer to a particular cluster
- rerun automatic algorithms, ...

6 Evaluation

NeuroVis has been used in investigations performed by neuroscientists and as an educational tool in a neuroscience classroom. Some of these experiences are described below.

One of the research questions we have been examining is the 8-dimensional condition on the conductance values that determines when a neuron will be silent (i.e., its resting potential stays at a single value). A second question is to determine the actual value of the resting potential for silent neurons based on the conductance values. We approached this problem by first using standard statistical tools to determine the range and distribution of resting potential values for silent neurons and then used that to construct a query in which all non-silent neurons were white and the silent neurons were given a color based on their resting potential. The images in Figures **??** and **??** show two different projections of this query. By repeatedly swapping dimensions to obtain new views we were able to determine that the silent neurons fall into six groups, where each group is quite well described by a linear inequality. The six domains defined by these inequalities correspond quite closely to six of the seven peaks in the histogram of resting potentials for silent neurons. The seventh peak corresponds to those neurons that lie in the intersection of two of the regions. We have not completed our analysis of the location of silent neurons on conductance space, but the tool has allowed us to discover (and visualize) the basic structure of this set of neurons and to tease out the fairly complex relationships among these six mutually intersecting linear half-spaces.

The next project we have embarked on is to study the relationship between conductance values and spiking/bursting behavior. It is relatively easy to create a query which indicates the number of maxima per period of each non-silent neuron, but the problem of counting the number of spikes per period is more complex, as it involves making somewhat arbitrary decisions about how to define a spike. We have also run into the problem of trying to understand the "irregular" bursters which never settled down into any repeating period during the initial simulation.

The ability to query the database by clicking on a pixel has allowed us to determine that the "irregular bursters" are those that occur near the boundary of the tonic spikers and consists of bursters that start tonically spiking for dozens or hundreds of spikes, but then fall out of the tonic spiking pattern.

6.1 Educational use

The NeuroVis tool has also been used in a graduate level Neuroscience course. Students were introduced to the tool and asked to investigate various relationships between conductance values and neuron behavior. A recent assignment was to

> Describe the relationship between tonic spiking and the Na, Kd, KCa, and CaT conductances.

It turns out that the tonic spiking region is a disjoint union of two regions with linear boundaries separating them from each other, as well as linear boundaries separating them from the silents and from the bursters. These fairly complex regions are difficult to describe analytically, but the students were able to make interesting observations about these complex non-linear relationships by simply exploring several teacher-selected queries and by manually reordering the dimensions and using the mouse to select and examine individual neurons (thereby discovering patterns that are not visible in the color map alone).

We are also using the NeuroVis tool to study other neuron model databases generated in a similar brute force manner, but which correspond to neurons with other properties, as well as to small networks of neurons.

7 Generalization

7.1 Limitations on data formats

NeuroVis employs interaction techniques like click-through data access that are applicable in many domains. We believe the application of dimensional stacking and pixelization is more limited and is best used with sparse grid simulations of dynamical systems. The combination of dimensional stacking and pixelization can be applied to any database where columns can be separated into two types:

- independent variables that serve as dimensions and have < 20 possible discrete values
- dependent variables which can have any number of continuous or discrete values

The dependent variables can have any number of functional dependencies on any subset of dimensions. However, it is desirable for attributes to not have functional dependencies on other dependent attributes as these will be harder to determine from our visualization. It is possible to discretize certain columns of a database to serve as independent variables or dimensions, however one must be careful not to map two data points to the same pixel unless some layering paradigm is being employed. These constraints are similar to those placed on visualizations of hierarchical data. In [**?**], Stolte also refers to independent variables as dimensions but refers to what we call "attributes" as "measures". The methods in NeuroVis are not

7.2 General application of dimensionally stacked pixelization

One can interpret a database or dataset as a function on tuples $T = (t_1, \ldots, t_n)$ where all of the t_i are small integers satisfying $t_i < D$ for some small number D. Each t_i is considered to be one dimension and the cardinality of possible values it can take on is its base. When all dimension bases are equal, T can then be viewed as a base D number with n digits.

For example, lets suppose $n = 4$ and the dimensions a, b, c, and d take on the values 0 and 1. We can interpret these dimensions as a 4 digit binary number. Ordering the dimensions in alphabetical order, the number 0101 would map to $a = 0$, $b = 1$, $c = 0$ and $d = 1$ or the decimal number 5. It is possible to have an odd number of dimensions and/or dimensions with unequal bases i.e. $0 <= t_i < j$, ..., $0 <= t_{i+1} < k$ and $j \neq k$. We account for this below.

To get a 2D representation we map every combination of dimension values to a pixel coordinate. First we partition the dimensions into two groups. Though any number of dimensions can be in either group, we generally follow the rule: $X = n/2$ and $Y = n/2 + n \; mod \; 2$ where n is the number of dimensions. The decimal values of coordinates are calculated the same as in [**?**]. The reverse operation follows this algorithm:

1. Given some decimal number m and an order of dimensions, start with the right most (or least significant) dimension and iterate left
2. $t_n = m$ modulo (the base of t_n)
3. m = m / (the base of t_n)
4. repeat for t_{n-i} until the value for t_0 is calculated

For the decimal value 7 and 3 dimensions with the bases 2, 3, 2:

1. $7 \; mod \; 2 = 1$ (keep for right most parameter = 1)
2. $7/2 = 3$ (ignore remainder pass on as new num)
3. $3 \; mod \; 3 = 0$ (pre-pend to current answer = 01)
4. $3/3 = 1$(ignore remainder pass on as new num)
5. $1 \; mod \; 2 = 1$ (pre-pend to current answer = 101)
6. = 101

The x coordinate of each pixel is derived from X and y from Y. The width of a pixelization is the product of the bases in X, and the height is the product of the bases in Y.

8 Conclusion

In this paper we have presented NeuroVis, a tool for interactive visualization and exploratory analysis of multidimensional neuroscience data. Its combination of dimensional stacking and pixelization has revealed new possibilities for interaction techniques. In this scheme, dimension reordering provides a type of visual data mining and principle components analysis where users can fish for data trends, examine the dimensions that determine them, and then write queries to define them.

While the visualization techniques of NeuroVis are best applied to sparse grid simulations of dynamical systems (such as model neurons), the interaction techniques are more widely applicable. Allowing users to specify color maps by directly querying

a database provides transparent data access and more control than most applications allow. Allowing those queries to return either subsets of the database or Real numbers also provides more flexibility. The obvious drawback is that users must have some familiarity with SQL syntax.

The concept of click-through data access is quite powerful because the visual representation of a data point is also a gateway to underlying data and applications relevant to it. In NeuroVis users clicked on pixels to launch other applications or execute a database query. However, one can easily imagine linking such functionality to glyphs or lines in a parallel coordinates display [**?**][**?**].

NeuroVis has proven useful in neuroscience investigations and as an educational tool in a neuroscience class. It has helped in elucidating biological findings that are now making their way into neuroscience publications. Our goals for the future are to continue development on NeuroVis, extending its current capabilities and perhaps supporting same time / different place collaboration. We also intend to generalize the implementation of NeuroVis and create a tool that allows users to spontaneously define data parameters. NeuroVis is already less dependent on its underlying data than many other applications, since users can specify what databases to log into on startup. We believe a generalization of its interaction techniques can inform the design of future tools for the visualization and analysis of multidimensional scientific information.

9 Acknowledgements

We would like to thank Professor Eve Marder and Dr. Adam L. Taylor from the Brandeis University neuroscience department for many helpful discussions on developing the NeuroVis tool.

References

[1] D. F. Andrews. Plots of high dimensional data. *Biometrics*, 28:125–136, 1972.

[2] Mihael Ankerst, Stefan Berchtold, and Daniel A. Keim. Similarity clustering of dimensions for an enhanced visualization of multidimensional data. In *INFOVIS '98: Proceedings of the 1998 IEEE Symposium on Information Visualization*, page 52, Washington, DC, USA, 1998. IEEE Computer Society.

[3] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68:361–368, 1973.

[4] S. K. Feiner and Clifford Beshers. Worlds within worlds: metaphors for exploring n-dimensional virtual worlds. In *UIST '90: Proceedings of the 3rd annual ACM SIGGRAPH symposium on User interface software and technology*, pages 76–83, New York, NY, USA, 1990. ACM Press.

[5] Timothy J. Hickey. Scheme-based web programming as a basis for a cs0 curriculum. In *SIGCSE '04: Proceedings of the 35th SIGCSE technical symposium on Computer science education*, pages 353–357, New York, NY, USA, 2004. ACM Press.

[6] Alfred Inselberg and Bernard Dimsdale. Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *VIS '90: Proceedings of the 1st conference on Visualization '90*, pages 361–378, Los Alamitos, CA, USA, 1990. IEEE Computer Society Press.

[7] Christopher Joslyn, Clayton Lewis, and Brigitta Domik. Designing glyphs to exploit patterns in multidimensional datasets. In *CHI '95: Conference companion on Human factors in computing systems*, pages 198–199, New York, NY, USA, 1995. ACM Press.

[8] Daniel A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):59–78, 2000.

[9] Daniel A. Keim. Visual exploration of large data sets. *Commun. ACM*, 44(8):38–44, 2001.

[10] Daniel A. Keim, Mihael Ankerst, and Hans-Peter Kriegel. Recursive pattern: A technique for visualizing very large amounts of data. In *VIS '95: Proceedings of the 6th conference on Visualization '95*, page 279, Washington, DC, USA, 1995. IEEE Computer Society.

[11] Daniel A. Keim and Hans-Peter Kriegel. Visualization techniques for mining large databases: A comparison. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):923–938, Dec 1996.

[12] Jeffrey LeBlanc, Matthew O. Ward, and Norman Wittels. Exploring n-dimensional databases. In *VIS '90: Proceedings of the 1st conference on Visualization '90*, pages 230–237, Los Alamitos, CA, USA, 1990. IEEE Computer Society Press.

[13] Ted Mihalisin, John Timlin, and John Schwegler. Visualizing multivariate functions, data, and distributions. *IEEE Comput. Graph. Appl.*, 11(3):28–35, 1991.

[14] Wei Peng, Matthew O. Ward, and Elke A. Rundensteiner. Clutter reduction in multi-dimensional data visualization using dimension reordering. In *INFOVIS '04: Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'04)*, pages 89–96, Washington, DC, USA, 2004. IEEE Computer Society.

[15] Astrid A. Prinz, Cyrus P. Billimoria, and Eve Marder. An alternative to hand-tuning conductance-based models: Construction and analysis of data bases of model neurons. *Journal of Neurophysiology*, 90:3998–4015, Dec 2003.

[16] Chris Stolte, Diane Tang, and Pat Hanrahan. Query, analysis, and visualization of hierarchically structured data using polaris. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 112–122, New York, NY, USA, 2002. ACM Press.

[17] Deborah F. Swayne, Dianne Cook, and Andreas Buja. Xgobi: Interactive dynamic data visualization in the x window system. *Journal of Computational and Graphical Statistics*, 7(1):113–130, 1998.

[18] M. Ward and D. Keim. Screen layout methods for multidimensional visualization, 1997.

[19] Matthew O. Ward. Xmdvtool: integrating multiple methods for visualizing multivariate data. In *VIS '94: Proceedings of the conference on Visualization '94*, pages 326–333, Los Alamitos, CA, USA, 1994. IEEE Computer Society Press.

[20] Jing Yang, Wei Peng, Matthew O. Ward, and Elke A. Rundensteiner. Interactive hierarchical dimension ordering, spacing and filtering for exploration of high dimensional datasets. In *IEEE Symposium on Information Visualization*, page 14, 2003.