Dimensional Stacking Visualization of Conductance Space Models of a Neuron

Category: Research

ABSTRACT

In this paper we describe a novel approach to visualizing the behavior of neurons with respect to their maximal conductances. This approach builds on the earlier work of Prinz, Billimoria, and Marder in which they studied the behavior of an 8 dimensional model of a neuron by using a brute force computational technique. Their approach was to select 6 discrete values in each of the 8 dimensions and collect a large amount of simulation data on each of the $6^8 \approx 1.7$ million possible model neurons corresponding to all possible combinations of the 6 discrete values per parameter. We show how dimensional stacking can be used to make biologically interesting inferences from this data as well as to provide a powerful and intuitive interface to the neuron simulation database. We also provide algorithms for finding the most informative ordering of parameters used to create the dimensional stacking visualization. This approach could prove useful in the study of other high dimensional complex dynamical systems.

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: Bioinformatics Visualization, Applications of Visualization, Databases and Data Mining Visualization, Multidimensional Visualization, Large Data Set Visualization, Dimensional Stacking

1 INTRODUCTION

In a research effort by Prinz, Marder, and Billimoria, a large database of neuron data [4] was generated in order to draw correlations between conductance levels of various ion channels and steady-state neuron activity. Statistical analysis of this data enabled them to discover new classes of neurons by looking at populations of neuron models. For example, they found that the periods of tonically spiking neurons formed a strongly bimodal distribution, indicating that there were two distinct classes of spiking neurons, slow spikers and fast spikers. Some natural questions arise about the relationship between the maximal conductance of these neurons and their classification as fast/slow spikers, e.g. what are the locations of these two classes of spikers in conductance space? How are they distributed throughout conductance space? What can be said about the border region between these distributions, if there is one? It was clear that some sort of visualization was called for to answer these types of questions.

In this paper we describe visualization methods we developed to answer these questions. The most significant initial requirement was visualizing the massive amount of data holistically in order to draw further correlations and identify global patterns. Further requirements were to provide transparent access to the data underlying this global view and to give insight into the causes for those patterns. This global view could also act as the launching point for analysis tools, relevant to the inquiry at hand, to apply to a particular data element or region of interest.

The generation of the model database is described in section 2. The dataset has two characteristics which determine our choice of visualization methods:

1. it has vast with 1.7 million rows in only one database table.
2. it is 8-dimensional.

Each row in the database represents one model neuron characterized by a sequence of 8 parameters each of which could take on 6 different values. The 3 GB database contains 2 KB of data (described in detail in Section 2), for each of the ≈ 1.7 million models neurons. Our visualization needed to represent a mapping of the 8 conductance parameters for each neuron model to these other attributes.

Many tools exist for the analysis and visualization of scientific data. However we were not able to find any tools that could handle datasets larger than about 100,000 points and also provide the kind of global view of the database that we required.

It is a useful coincidence that the size of our dataset (1.7 million points) is similar to the number of pixels on a modern computer screen. This suggests using one pixel to display information about each model neuron. We were able to address both the size and multidimensionality issues by employing dimensional stacking, a technique introduced by LeBlanc, Ward, and Wittles [3]. Dimensional stacking is a technique in which multiple dimensions are mapped to one dimension. It requires that the possible values for each dimension are all small integers. In our case, each model neuron is specified by an 8-tuple of integers in the range [0,5], i.e. an 8 digit base 6 number. We project this to 2 dimensions by viewing the model number as a pair of 4 digit base 6 numbers. More precisely, for a set of 4 conductances (x_1, x_2, x_3, x_4), one of the projection coordinates X was computed as

$$X = x_1 * 6^3 + x_2 * 6^2 + x_3 * 6^1 + x_4 * 6^0 \quad (1)$$

Y is defined similarly.

Dimensional Stacking in our case defines a class of projections from an 8 dimensional set of neurons to a 2 dimensional set of pixels. For each permutation of the 8 parameters, one gets a different projection. We use these projections to visualize the way neuron properties depend on their parameters.

In general our approach is to specify a pixel color for each model neuron that corresponds to some combination of properties of the neuron (e.g. a classification of the electrical behavior, or a continuous quantity like the period, or some property of the behavior of a neuron upon a 3nA current injection). Our visualization tools allow one to either change the permutation, to get a more revealing projection, or to change the properties being displayed in the projection.

Finally, once the visualization for a particular property has been constructed, we have found it useful to allow the analyst to interact with the database and the simulator, through this visualization. For example, we have a mode in which a database query can be displayed to provide information about the neuron at the current mouse position. Dragging the mouse across a section of the visualization provides information about how a property changes within a

particular region. Similarly, double clicking on a neuron pixel runs a neuron simulator on that pixel in real time, to generate detailed information not stored in the database.

The end result is a collection of tools we call NeuronVis. We are developing versions of this tool in Matlab, Java, and Scheme. Using dimensional stacking, NeuronVis provides a 2 dimensional view of the neuron data with transparent access to the neuron simulator and underlying database.

This tool has demonstrated that the relationship between the position of a neuron in conductance space and many of its biologically interesting properties has a great deal of structure that was not apparent using only the standard statistical analysis tools (see Section 4).

2 DATABASE CONSTRUCTION

The database used here to illustrate the visualization of high-dimensional datasets by Dimensional Stacking was constructed by simulating and classifying the electrical activity of a model neuron for 1.7 million different combinations of maximal membrane conductances. The model neuron consists 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;
- and a leak current, I_{leak}.

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

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

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 [4].

The non-linear differential equations governing the dynamic variables V, m_i, and h_i form a system of coupled differential equations that is not analytically tractable. Due to the complex interactions between the voltage-dependent membrane currents and the membrane potential, the role of an individual membrane current in shaping the dynamics of the membrane potential - which is the signal neurons use to encode information and communicate - is difficult to understand and often counter-intuitive. The model neuron database used here was constructed to overcome these problems and to better understand how individual membrane currents contribute to the electrical activity of a neuron.

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 conductance increment for each current is given in the legend to Fig. 1).

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 - the most salient features of a neuron's electrical activity pattern - were detected and saved in the database for each model neuron. Based on its pattern of voltage maxima and minima, each model neuron was classified in one of four major categories. Neurons were classified as silent if they showed no voltage maxima or minima (top trace in Fig. 1). Neurons with periodic voltage maxima at equidistant time points were classified as spikers if their action potentials showed no broad shoulder on the down-stroke (second and third traces in Fig. 1) or as one-spike bursters if they showed a broad shoulder (fourth trace) – see [4] for more details on this distinction.

Models with a periodic, but not equidistant sequence of voltage maxima were classified as bursters (fifth and sixth traces), and models for which no repetitive sequence of voltage maxima could be detected were classified as non-periodic (bottom trace in Fig. 1). In addition to the maxima and minima sequences and classification results, the database also contains information about the resting potential for silent neurons, the spike frequency for spikers, the burst frequency, burst duration, duty cycle (burst duration divided by burst period), and number of maxima per burst for bursters, and the average spike frequency for non-periodic neurons.

The behavior of a neuron is not only characterized by its spontaneous activity, but also by its response to synaptic inputs from other neurons. The model neuron database therefore also contains information about the response properties of all 1.7 million neurons. To obtain this information, the injection of two different levels of constant current into each model neuron was simulated and the electrical activity pattern in response to that stimulus was saved and classified as described above for the spontaneous activity. Furthermore, the response of all bursting neurons to brief inputs at different times during their ongoing oscillation was also simulated and saved in the form of phase response curves. More details about these additional simulations and database components are available in [4]

Because only six different values were explored for each maximal conductance Gi, 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 [4] Visualization of the database with Dimensional Stacking or other techniques should therefore lead to inter-

pretable results.

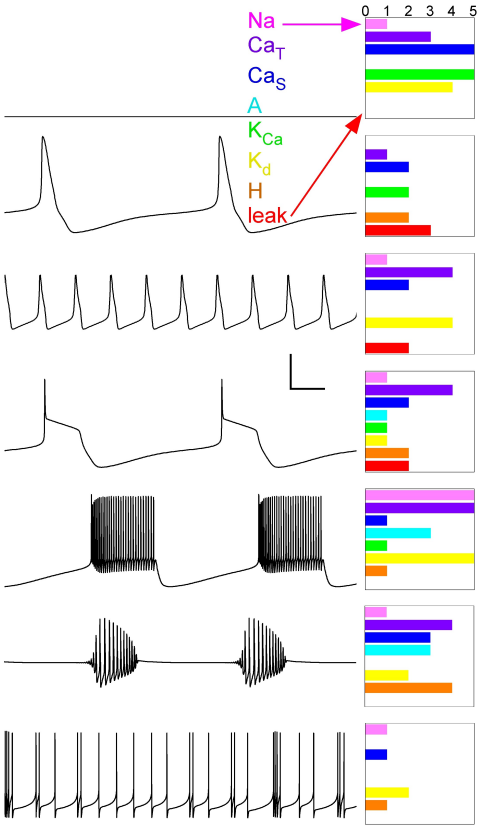

Figure 1: Voltage traces (left) and membrane current composition (right) of selected database neurons. Electrical activity ranges from silent (top) to different types of spiking and bursting to non-periodic activity patterns (bottom). Vertical scale bar is 50mV for all voltage traces, horizontal scale bar is (from top to bottom, in ms): 100, 100, 10, 100, 300, 100, 1000. Conductance levels are given in multiples of (in mS/cm2): 100 (Na), 2.5 (CaT), 2 (CaS), 10 (A), 5 (KCa), 25 (Kd), 0.01 (H and leak). From top to bottom the activity types are: Silent, Slow Spiker, Fast Spiker, One Spike Burster, Burster, Elliptic Burster, Non-periodic

3 Related Work on Visualization

Wong and Bergeron [9] provide the standard introduction to multidimensional multivariate visualization. They distinguish *dimensions* or independent variables (which we call parameters in this paper) from *variate* or dependent variables (which we refer to as values). van Wijk and van Liere [8] introduce HyperSlice which allows the user to navigate the space and change the projection to show the dimensions of most interest. Buja and Asimov [1] discuss Grand Tour methods.

Most of our work has dealt with high dimensional data. While we do have multi-variate data, we tend to be view one variable at a time. The problem we are working on is the sheer volume of data. The data set we use has over 10^6 data points. Earlier work that we are aware of (for example XGobi [6]) cannot handle more than about 10^4 data points.

D.J. Duke further asserts the importance of linking meaning with representation in [2]. His work so far is limited to 10,000 to 100,000 points, and is best suited to displaying relationships between those points. It seems less applicable to the databases we study where the relations between points must be discovered.

4 Dimensional Stacking

Dimensional stacking is a technique introduced by LeBlanc, Ward, and Wittels [3]. It provides a method for taking multi-dimensional data and displaying it in a reduced number of dimensions (typically two or three) in order to provide a simple visualization. In its simplest form, one assumes that the data represent a function on tuples $a = (a_1, ..., a_n)$ where all of the a_i are small integers satisfying $0 \leq a_i < D$ for some small number D. Thus, a can be viewed as a base D number with n digits and this would give a projection down to 1 dimension, or it could be viewed as 2 base D numbers with $n/2$ digits each, and this would give a projection down to 2 dimensions. Dimensional Stacking was ideal for our purposes of making the entire neuron database visible at one time in such a way to make patterns appear.

In our case, we have a database filled with values about model neurons, each of which is uniquely defined by its model number, which is an 8-tuple of integers in the range [0,5], i.e. an 8 digit base 6 number. We reduce this to 2 dimensions by viewing the model number as a pair of 4 digit base 6 numbers. More precisely, we choose a set of 4 conductances (independent variables) (x_1, x_2, x_3, x_4), and compute the X coordinate as:

$$X = x_1 * 6^3 + x_2 * 6^2 + x_3 * 6^1 + x_4 * 6^0 \quad (2)$$

The Y coordinate was calculated in the same fashion using the remaining conductances. We interpret these 2 numbers as the x and y coordinate for a pixel on a 2 dimensional image. Because there were about 1.7 million model neurons this meant there were the same number of pixels, now displayed in a 1296 by 1296 square. Note that by permuting the coordinates in the 8 tuple representing a neuron model number, we obtain a new projection. This means there are 8! or about 40,000 different projections. An immediate issue was which order to put the conductance values in during the dimensional stacking in order to find an instructive data projection. This is a non-trivial issue. We were able to use some simple algorithms to find good projections. We aim to improve upon these algorithms and combine them with human user interaction to find optimal projections in the future.

Besides selecting a projection, the visualization must also specify the color of each neuron. This is done by calculating some discrete and/or continuous properties of each neuron and then associating a color to each such property. For example, if one has a classification of the neurons into K different classes, then one can use a color map that associates each integer from 0 to $K-1$ to one of K distinct colors. Similarly, if one has a continuous parameter for each neuron (say the period of the neuron's electrical activity function, normalized to lie in the range [0,1]), then one can map that real interval into a spectrum of colors. There are well studied [7] strategies for selecting such color maps to make the data easy to read. Finally, one can combine both discrete and continuous values into a color map, e.g. plotting both a classification and the period length, by associating each classification category to a hue and then using the period to determine the intensity of the hue.

4.1 Reading a dimensional stacking visualization

One of the main advantages of the dimensional stacking visualizations is that it introduces a hierarchical structure into the image, which makes it relatively easy to read. For example in Figure 2, one clearly sees the image divide into a 6x6 grid of blocks. This 6x6 grid corresponds to the first conductance in the x and y projections. Similarly, each one of those 36 blocks has a similar 6x6

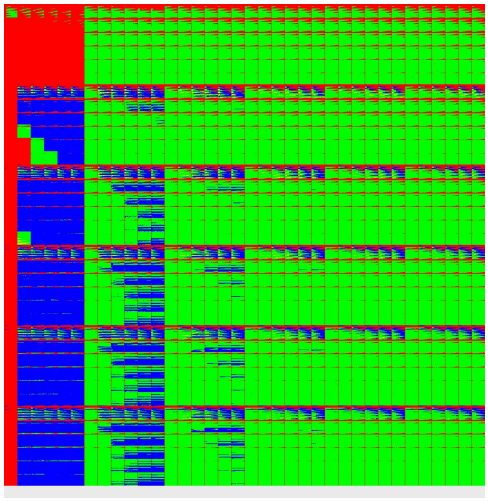

Figure 2: Level 1: Basic Visualization. Silent neurons are red, spikers are blue, bursters are green, irregular neurons are yellow.

decomposition into smaller cells. For example, Figure 3 shows the 2nd block in the 2nd row of Figure 2 and again it is has the clearly discernible structure of a 6x6 grid of blocks. Each of these block is as well a 6x6 grid of blocks, as we can see in Figure 4, which is a close-up of the 6th block in the second row for Figure 3. Finally, each of the 36 blocks in this Figure 4 is a 6x6 block of pixels, each of which corresponds to a model neuron with 8 parameters. Figure 5 shows the 6x6 block of pixels in the 1st row of the fifth column of Figure 4.

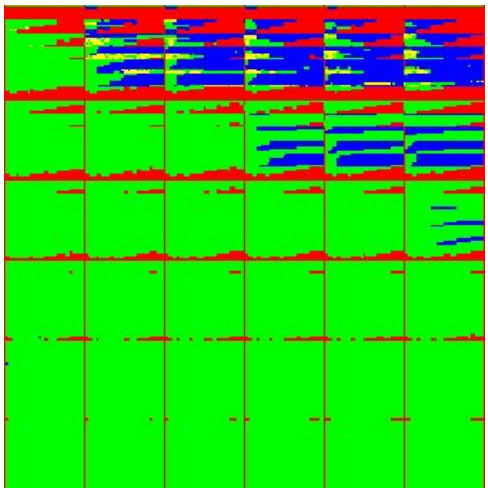

Figure 3: Level 2: Zoom in on 2nd block of 2nd row (of Fig. 2)

4.2 Optimal Projections

The ordering of conductance values has a dramatic effect on the image resulting from our dimensional stacking method. Because there are 8 dimensions (one for each conductance value) in a 1296 X 1296 image there are $8! \approx 40,000$ projections (permutations) of data to pixel mappings. It was quite fortuitous that we happened upon a meaningful projection that revealed nice patterns in our first attempts.

This does not always happen. For example, Fig. 6 and Fig. 7 show two different projections of the database using the same color

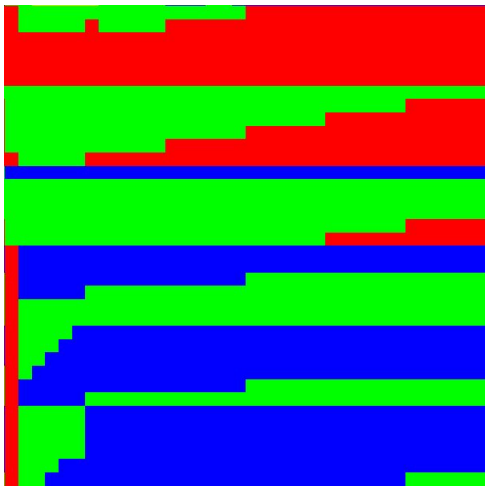

Figure 4: Level 3: Zoom in on 6th block of 2nd row of Level 2 visualization (Fig. 3).

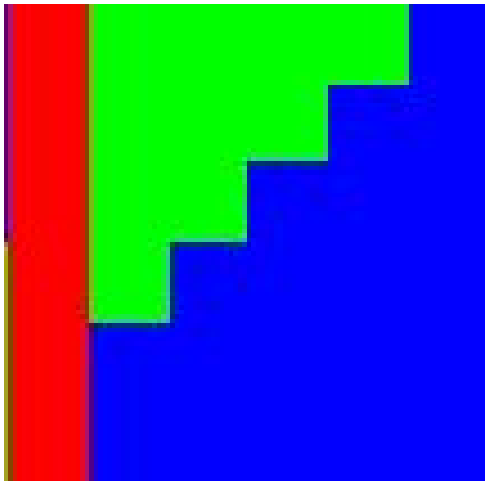

Figure 5: Level 4: Zoom in on 1st block of 5th row of Level 3 visualization (Fig. 4).

assignment of red for slow spikers, blue for fast spikers, green for intermediate spikers, and white for all other neurons. In the second figure, one can much more clearly see the relationships between the four categories of neuron (red, blue, green, white). In Fig. 6, the visualization looks like almost random noise, but in Fig. 7 it's clear that there is a great deal of structure in the data, and by looking at which current corresponds to the first (most significant) digit, one gains an understanding of the space. These images show the fast and slow spikers and clearly indicate that they occupy largely different regions of conductance space with one exception in which there is some overlap. We have determined that there is a hyperplane that separates these two regions with 99% accuracy, and visual examination of the separation indicates that the error appears random and not systematic.

Figures 8, 9 and 10 show another example involving the visualization of the number of maxima in a burst for bursting neurons. For this visualization we started with a random projection (Fig. 8) which showed some interesting structure, after 4 steps of the algorithm to find the optimal projection we obtained Fig. 9, which showed more structure, and when we applied the algorithm until it reached a local optimum (2 more steps), the result (Fig. 10) was

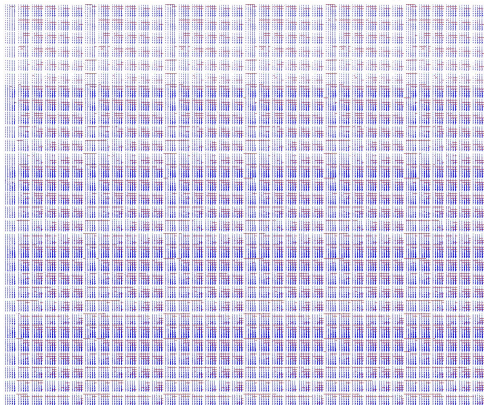

Figure 6: Fast/Slow, random permutation

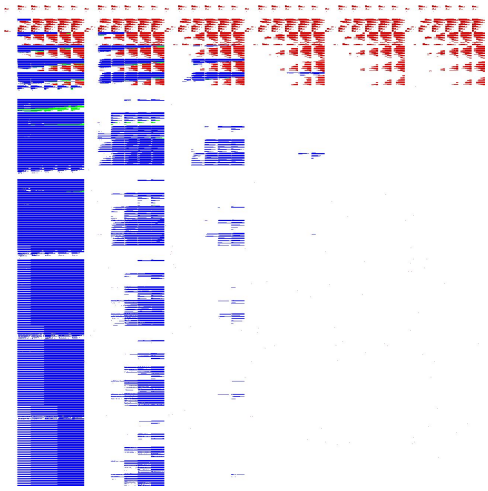

Figure 7: Fast/Slow spiker, better permutation. This is the same data as Fig. 6 with a different permutation of the currents.

a much more revealing image. Without a tool like NeuroVis, it would have been very difficult to discover and analyze the structure revealed by Fig. 10.

It is not surprising that the structure of the subset of neurons which are bursters depends heavily on different currents than the structure of the more general classification, but without tools like NeuronVis, one would not be able to easily learn what the structure is.

We have found that a relatively simple algorithm can be used to rapidly find visually informative projections for our domain. We begin with a simple measure of "visual complexity" of a row or column in which we count the number of times the color changes as we proceed across the row (or down the column). Thus an image that is black on the left half and white on the right half would have a visual complexity measure of 1 for each row and 0 for each column. The algorithm proceeds by computing the visual complexity of the current projection and then iteratively computing new projections that differ from the current one by a single transposition (swap of two coordinates). Whenever a new projection is found that has a lower "visual complexity" measure, then that projection is selected and the algorithm proceeds until it reaches a local minimum. This is simply a depth first search of the tree of possible transpositions. In the examples we have considered a minimum is usually found in under 10 steps.

Most of the regions of solid color in the last figure represent neurons in which the neigbors in any of the four lowest dimensions have the same classification. The similarity between the structure of the cells in the top level 6x6 grid indicates that changes in the top two dimensions have the effect of moving the boundaries between neuron classes in the lower dimensions.

Over a period of minutes or hours the maximal conductances of a real neuron may change incrementally. This corresponds to marking out a path in conductance space. The dimension stacking visualizations allow one to make inferences about when a neuron is likely to switch from one type of behavior to another and, on the contrary, those places where neuron behavior is relatively stable.

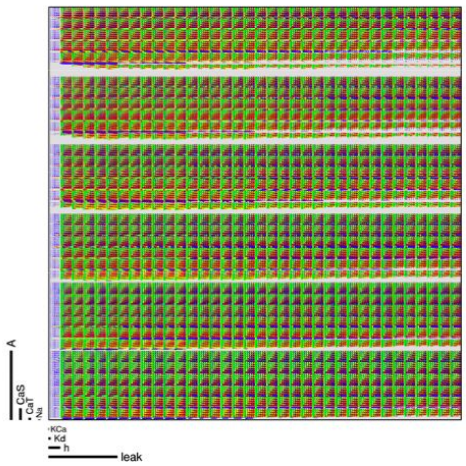

Figure 8: A random permutation showing number of maxima per burst

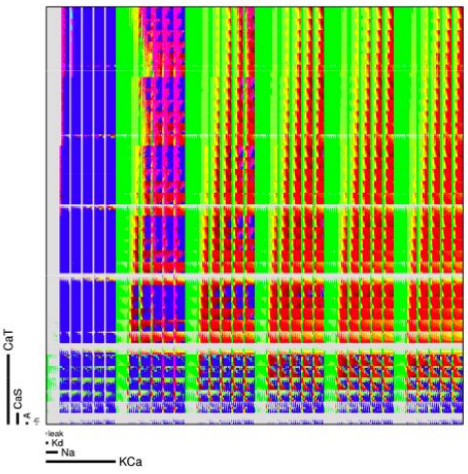

Figure 9: The data from fig 8 after four projection optimization steps

4.3 Data Access

Because the assignment of a color map to a 2D visualization allows us to only display one or two properties (i.e. dependent variables)

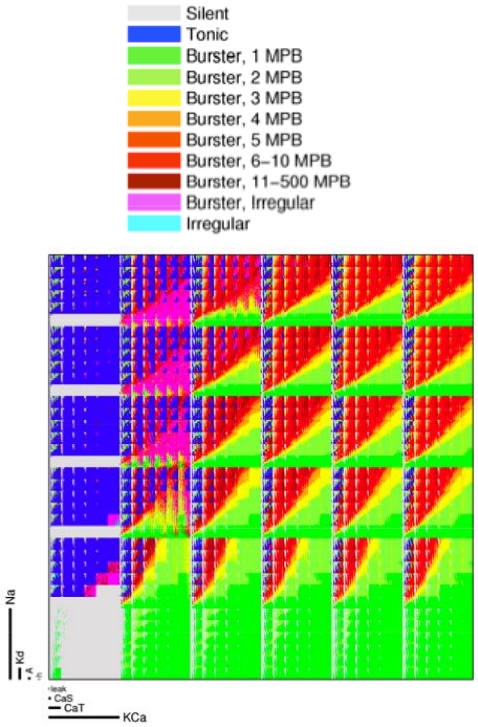

Figure 10: The data from fig 8 displayed using a better permutation. The color code at the top is also used in fig 8 MPB is number of maxima in each burst.

.

out of the hundred or so that the database holds, it is extremely useful to allow the user to look up more of the information from the database quickly. To do this we built a system which allows transparent access to the data underlying the visualization. The idea was to allow the user to see more of the mapping from pixel to neuron model. Our first simple efforts included showing conductance levels for the neuron model corresponding to each pixel as the user passed the mouse pointer over the image. We then decided to allow right clicking on a single pixel to launch various analysis tools.

The first analysis tool was to provide the scientist with a method of directly querying the data and applying mathematical functions to it by allowing click through access to the underlying database. In our first attempts we provided a static query ourselves, though we would like to extend this to be customizable. The current version of this tool returns all the calculated values for the neuron model associated with the pixel in our query (that is, the pixel the pointer specifies).

In this way we make it easier to express queries and mathematical operations directly on the data, and see the results immediately in our visualization, allowing the user to quickly answer questions about details of the behavior in an area of interest.

A second tool, shown in Figure 11, provides access to the simulator that was used in the construction of the database. The simulator takes a few seconds to simulate a single model neuron, so it is acceptable to simulate a few models while the user sits at the console. (To calculate all 1.7 million models takes a couple of months, so it is clearly not practical.) In one particular case, two classes of neurons (slow spikers and 2 max bursters) were found to be near each other in conductance space. Because we could click on a pixel and simulate the neuron model associated with it to see the voltage plot, we were able to better understand the relationship between these two classes of neurons.

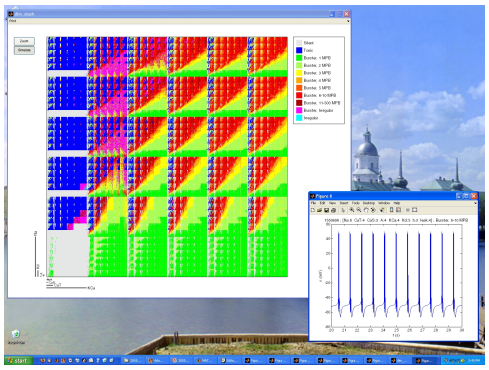

Figure 11: Double clicking on a single pixel allows one to run the simulator to produce the full graph of the electrical activity for that neuron.

5 Future Work

There are a great many possibilities for software tools to support and enhance the visualization methods we have described in this paper.

One obvious possibility is to allow the users to customize the query when they right click the pixel by customizing the coloring function applied to each element to show any value of interest. While we provide some support for such functionality, it could be greatly expanded. We describe in the Data Access section how we allow a scientist to specify a query in SQL, and a color to be applied to all of the data elements (in our case neuron models) returned by that query. We could extend this to take multivariate functions or continuous functions, apply them to the data, then color each pixel in gradients or shades depending on the answers, rather than solid, differentiated colors applied to each resulting query data set. We could also change the query language we are using (SQL) to something that would be more natural to a non-computer scientist.

There could of course be any number of programs to analyze a single cell's behavior. This is leaving the domain of visualization, as knowing what analysis of a cell's behaviour is interesting is a question for the neuroscientist. Our dimensional stacked data image could provide access to any one of these programs through clicking on a pixel.

The dimensional stacking definitely incurs a learning curve in understanding the images we produce. We have found that labeling the axis with lines and ion conductances where they occur is very helpful. A simple set of labelled lines of different lengths with the length of each line equal to the distance of a one unit change in the value of that parameter is a very effective label (There is an example in the lower left corner of Fig. 8). In our prototype we also show the changing conductance values (parameters of the neuron represented by that pixel) as the mouse travels over the image. This helps quite a bit but it would be wise to experiment with more labeling methods, mouse over tool tips, or other methods to give the user a better idea of the mapping between image and data.

We have done some preliminary work into probability distributions of one model going from one steady state to another when undergoing a change in one conductance value. These probability distributions sometimes provide an insight into the boundaries between regions of neuron model activity types in the data. We believe that allowing the user to use this information in choosing optimal projections could provide quick access and insight into data patterns.

There of course are also many other AI algorithms that could be utilized in finding optimal projections, such as simulated annealing. One possibility is letting the user specify the heuristics or param-

eters for these functions in some mixed initiative effort, to quickly find the boundaries between neuron model activity types. One example is that a user could select a group of pixels they believe to be in one class. That selection could then be used in the heuristics of some clustering algorithm to isolate that class of neuron model in the data and image. It would also be interesting to allow users to select groups of neurons, then query them directly. In this way the user could identify a global pattern in the data by looking at our image, select the pixels in that pattern, then query those data elements to see how they correlate to one another, i.e. it might be that they all share one conductance value and that observation makes the pattern emerge.

Another possibility is to be able to change the underlying data by reducing the values which the independant variables can take. In our case, it might be unlikely in nature for there to be neurons with 0 conductance levels. We could therefore eliminate those models from our data set, improving performance and the relevance of our queries. It might also be that we are only interested in certain dimensions and could improve performance of software tools and analysis by removing some parameters for some number of queries.

Because our values for conductance levels were discretized, it would also be interesting to be able to add intermediate values of parameters if something interesting seems to be happening between two discrete values, but that is far too much computation to do in real time.

6 Conclusion

We have found that using dimensional stacking to map data points to pixels gives a comprehensive yet immediate and revealing insight into a large data set. Some tool for finding good permutations of the parameters is clearly necessary, and the simple one we describe above has worked extremely well.

Providing further access to the underlying database by mousing over the pixel of interest provides an intuitive way of probing the data. The combination of both allows an analyst to switch between identifying patterns over the entire data set to examining the implications of these patterns on a per-data-element basis. The per-element analysis can also provide insight into the causes of the patterns observed, or even show that they are merely artifacts of the model used.

The development of the visualization tool NeuroVis described in here was guided by the need for visualization of a database of model neurons, but can readily be generalized to datasets resulting from explorations of other high-dimensional and non-linear dynamical systems with computational brute force, whether within neuroscience [5] or beyond.

References

[1] Andreas Buja and Daniel Asimov. Grand tour methods: An outline. In D. M. Allen, editor, *Computer Science and Statistics: Proceedings of the Seventeenth Symposium on the Interface*, pages 63–67. Elsevier, 1986.

[2] D. J. Duke. Linking representation with meaning (poster). In *VIS '04: Proceedings of the IEEE Visualization 2004 (VIS'04)*, page 5p, Washington, DC, USA, 2004. IEEE Computer Society.

[3] Jeffrey LeBlanc, Matthew O. Ward, and Norman Wittels. Exploring N-dimensional databases. In Aire Kaufman, editor, *IEEE Visualization: Proceedings of the 1st conference on Visualization '90*, pages 230–237. IEEE Computer Society Press, 1990.

[4] 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.

[5] Astrid A. Prinz, Dirk Bucher, and Eve Marder. Similar network activity from disparate circuit parameters. *Nature Neuroscience*, 7(12):1345–1352, Dec 2004.

[6] Deborah F. Swayne, Dianne Cook, and Andreas Buja. XGobi: Interactive dynamic data visualization in the X window system. *Journal of Computational and Graphical Statistics*, 7:113–130, 1998.

[7] Edward R. Tufte. *Visual Explanations: Images and Quantities, Evidence and Narative*. Graphics Press, 1997.

[8] Jarke J. van Wijk and Robert van Liere. Hyperslice - visualization of scalar functions of many variables. In Gregory M. Nielson and R. Daniel Bergeron, editors, *IEEE Visualization*, pages 119–125. IEEE Computer Society, 1993.

[9] Pak Chung Wong and R. Daniel Bergeron. 30 years of multidimensional multivariate visualization. In Gregory M. Nielson, Hans Hagan, and Heinrich Muller, editors, *Scientific Visualization – Overviews, Methodologies, and Techniques*, pages 3–33. IEEE Computer Society Press, 1997.