Eurographics/ IEEE-VGTC Symposium on Visualization 2009
H.-C. Hege, I. Hotz, and T. Muntzner
(Guest Editors)

Volume 28 (2009), Number 3

Pairwise Axis Ranking for Parallel Coordinates of Large Multivariate Data

1060

Abstract

Parallel coordinates has proven to be a scalable navigation framework for multivariate data. In parallel coordinate visualizations, human perception leverages spatial locality to determine the existence of multivariate patterns. However, when data with thousands of variables are at hand, we do not have a comprehensive solution to algorithmically select the right set of variables and order them to best uncover important or potentially insightful patterns. To further complicate the matter, important patterns may be dependent upon domain-specific properties. In this paper, we present a set of algorithms to rank axes based upon the importance of bivariate relationships among the variables. We showcase the efficacy of the proposed system by demonstrating autonomous detection of patterns. We demonstrate our approach using a modern large-scale dataset of time-varying climate simulation.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Methodology and Techniques. G.1.6 [Optimization]: Global Optimization. I.2.8 [Artificial Intelligence]: Problem Solving, Control Methods, and Search. J.2 [Computer Applications]: Physical Sciences and Engineering.

1. Introduction

Visualization and computational tools are necessary for the analysis of large multivariate data. Parallel coordinates rendering has proven very useful in this area as it allows intuitive visualization of a multidimensional attribute space. It is very natural to visually "chain together" a sequence of variables in a layout that scales linearly with data dimension. Ideally, a parallel coordinate rendering should provide a view containing sufficient information to guide users to the most interesting parts of the underlying data. There are numerous challenges addressed to varying degrees by current literature relating to proper axis ordering, axis scaling, axis shape, number of axes, rendering technique, clutter reduction, interactivity, etc. In this paper, we aim to address the problem of automatically selecting the proper order of axes by ranking them based upon an underlying system of metrics which specifies relationships between each of the variables.

The traditional approach for axis ordering is to rely on the user to drag axes into positions to discover and elucidate a desired pattern. This goal is quite achievable when the data at hand is manageably small in terms of the total number of data points and variables that must be considered at any particular time. However, without sufficient computational tools, this task is quite daunting as we increasingly need to handle datasets with hundreds or thousands of variables. Our goal is to systematically address the axis ordering problem with scalable algorithmic methods.

A parallel coordinates plot (PCP) can only show a handful of axes on most screens without cognitively overloading the user or obscuring patterns due to visual clutter. Therefore, the task of selecting a relatively small subset of increasingly large multivariate datasets becomes ever more complex. Indeed, the ability to choose the "right" handful of multivariate relationships can be highly context-specific. To approach this, we note that users innately leverage spatial locality and relate a given attribute to at most two (the axes before and after) other attributes in a PCP. By doing so, we reduce the problem space of multivariate relationships down to the sequence of bivariate relationships which PCPs innately represent. We provide a pair of algorithms to optimally or near-optimally rank the variables of an N-variable dataset based upon its $O(N^2)$ bivariate relationship while abstracting the concept of "right" to a user-specifiable metric (correlation, positive skew, etc.).

The novelty of our work stems from our taking an optimization driven perspective to explore the full potential of using parallel coordinates to visualize datasets with a large number of concurrent variables. We provide a pair of algorithms to autonomously find the most interesting patterns. We also devise a PCP rendering method to better reveal patterns based upon underlying bivariate relationships visually in parallel coordinates. We demonstrate our system on IPCC climate simulation data. In total, we show parallel coordinate renderings with axes selected from thousands of variables.

In the remainder of this paper, we first give a summary of related works in Section 2. In Section 3 we introduce our metrics for characterizing patterns. Section 4 introduces the general optimization framework and how the metrics can be used to detect specific patterns. Our depth-based PCP renderer is introduced in Section 5. Finally, our results and discussion are provided in Section 6, and then concluded in Section 7.

2. Background

Parallel coordinates, popularized in large part by [Ins85], have become increasingly popular as a scalable technique for visualization and interaction with large multivariate data. A parallel coordinate plot (PCP) is a generalization of a Cartesian scatterplot in which axes are drawn parallel to one another. This type of diagram highlights the more common case of parallelism, rather than orthogonality, present in higher-dimensional geometry. PCPs also allow an arbitrarily large number of dimensions to scale intuitively within the plane, whereas human perception degrades quickly as dimensions higher than three are projected to a 2D display.

PCPs developed as a way to accurately visualize and thereby gain insights from multidimensional geometry. From their onset, several mathematical properties were proven which enhanced their interpretation by defining analogues between parallel coordinate space and two-dimensional Cartesian scatterplots [MW02]. These included the point-line, translation-rotation, and cusp-inflection point dualities [Ins85,ID94]. This technique quickly found its way into Vis [ID90].

There has been much research to alleviate some of the inherent weaknesses of PCPs such as visual clutter when dealing with large data. Techniques for clutter reduction include clustering, subsampling, and axis redirection. In [FWR99], the authors use a clustering algorithm to create a hierarchical representation for PCPs and render a graduated band to visually encode variance within a cluster. In [JLJC05], the authors use K-means clustering, a high precision texture to reveal specific types of clusters, multiple transfer functions, and an animation of cluster variance to accurately convey the clustered data while minimizing clutter. In [ED06], the authors provide a sampling lens and demonstrate that random sampling of lines within the lens is the optimum choice in the tradeoff between their accuracy metric and performance. In [WL96], the authors use the grand tour algorithm to generate a d-space general rigid rotation of coordinate axes which can be used to confirm separability of clusters.

Perceptual properties such as the importance of axis orderings were considered as early as [Weg90]. While [Weg90] gives equations for selecting an order from a set of axes, he in no way addresses optimality criteria. The work most similar to this paper is the work of [PWR04] in which a dataspace clutter metric was introduced in combination with heuristic algorithms for determining axis ordering. However, we more systematically address the axis ordering problem with our introduction of customizable metrics, globally optimal ranking, and near-optimal ranking algorithms with lower theoretical complexity.

PCP implementations often operate on binned data and even uncluttered PCPs often demonstrate data ambiguities. The traditional straight-line rendering can be augmented by using energy-minimization to render curved lines as in [ZYCQ07]. Likewise, the authors in [GK03] used Gestalt principles of good continuation in order to address ambiguities from line crossovers and shared values. Binning can cause outliers to dominate data expression or be filtered out altogether which has lead [Nov06] to more thoroughly analyze the preservation of outliers while capturing the major patterns in PCPs.

In additional to use solely as a visualization technique, PCPs can be used as an intuitive navigation and query methodology. Recent research has demonstrated the ability to interactively alter axis ordering, interactively brush range queries, and use axis scaling to refine those queries [SFJKY07]. The introduction of a navigation element to the visualization leads naturally to more complex data mining techniques. In [FL03], the authors use parallel coordinates, and its circular variant, as multidimensional visualization elements in a taxonomy of other techniques. They also highlight user interface and analysis concerns, highlighting the strengths and weaknesses of PCPs in the context of other visualization methods.

There are also several variants of the traditional parallel coordinates rendering. These include the circular variant in which axes are drawn as spokes of a wheel [FL03]. This approach was extended by [JCJ05] who created three-dimensional parallel coordinate renderings by placing axes like tent pegs in a circle around a central axis. PCPs were also extruded into three-dimensional renderings by treating each line as a plane that could arbitrarily be transformed [WLG97].

Closely related to our rendering technique is the line density plot of Miller and Wegman [MW91]. The authors suggest that in crowded plots, visualizing the density of lines—rather than the lines themselves—may be more informative. They investigate the line density of normally and uniformly distributed random variable pairs. We apply a similar concept to empirical datasets and use line density plots to perform bump mapping.

3. Metrics

When attempting to determine an optimized axis layout, the criteria that makes such a layout desirable may vary depending upon the task at hand. For example, it is often the case that users want layouts which minimize the amount of visual clutter so that patterns can be easily detected; on the other hand, researchers testing their own clutter reduction methods may be interested in testing on only the most cluttered layouts.

We propose that the proper level of abstraction necessary for this type of problem is to optimize based on the level of some user-defined metric which should be designed to capture the property of interest. In this way, we propose a general set of algorithms which can be applied for truly optimal axis layout in whichever manner is deemed relevant by the user, codified by a matrix of normalized metric values. To relate parallel coordinates using the common spreadsheet metaphor, each data item corresponds to a row and each column to an axis in the PCP. The final metric value thus encodes the information of interest in how one such dimensional axis relates to all other dimensions for the given data.

The problem then becomes, which metric captures the properties of potential interest for my type of data and the question I want to investigate? While listing all possibilities is innately intractable, we find that a surprising few satisfy most needs along with mechanisms for creating variants suited for a particular purpose. In this section, we focus on a single pair of variables to develop quantitatively defined metrics which capture distinctive patterns in data space. In this paper, we provide the general mechanism to use any given metric to autonomously reveal meaningful patterns in large data. This capability facilitates the user's first need of quickly previewing the most unique pattern in a real-world dataset containing thousands of variables.

Data space is the traditional environment for metric calculation since a metric is mathematically defined as a measure of distance. Metrics used by our system are square matrices with the number of rows and columns equal to the number of data dimensions. As such, there is a plethora of applicable mathematical metric definitions, frequently involving calculation of trans-dimensional variance over all data points. The art of the system is to sufficiently constrain the possible metric space to a measure appropriate for the current investigation. However, beginning users should not be expected to know or to care about individually defining their own golden standard metric. In that regard we provide several useful ones with widespread applicability while leading into possibilities for the more sophisticated user.

One of the most commonly-used data space metrics is correlation. While this is typically measured using Pearson's correlation, we have found that other measures of correlation such as entropy-based mutual information or even literature keyword correlation are better than Pearson's in certain domain-specific contexts. While correlation does not necessarily translate into causation, it is often used as an indicator that can direct an analyst's attention to potentially novel knowledge discovery or warrant further investigation to determine the active mechanism of causation.

More complex non-linear relationships are useful in contexts that warrant increased specificity. For example, general modeling techniques such as Bayes' rule for estimating risk or physics-based models for estimating a property of interest. Once the metric has been calculated and used to extract a dimensional ranking, other dimensions which are highly related to the computed metric are immediately visible and can in turn be used to increase or decrease the complexity of the current model.

4. Ranking Algorithms

Our goal is to develop a general system which autonomously generates a near-optimal axis ordering by ranking variables to obviate key bivariate relationships for a given dataset. The only required data is a matrix containing values for each axis pair which roughly denote the importance or strength of a relationship between two attributes based upon a user-selected or computationally defined metric. Here we present a pair of algorithms which provide an optimality/time tradeoff based upon a user's given data size, computational resources, and time constraints.

4.1. Optimal Ranking

Once given an $N \times N$ matrix corresponding to all pairwise attribute metrics, the problem can be solved in the domain of graph algorithms by treating it as an adjacency matrix. In this scheme, there is a graph of N vertices and N^2 edges from which we want to extract what we shall refer to as an "optimized k-walk" where k is the number of axes desired in the parallel coordinate plot. This can be seen as a generalization of the Traveling Salesman Problem in which the salesperson must make an optimized visit to $k \leq N$ cities and reduces to the classical problem for $k = N$.

The brute force method for solving an "optimized k-walk" is to simply take every possible N *choose* k subset and permute every subset's k variables to find the maximum sum of edge weights between consecutive pairs. This method is $O((N\ choose\ k) * k!)$ or $O(\frac{n!}{(n-k)!})$ and can be used to find a subset of k values which maximizes the fitness of the resulting ranking derived from this simple optimization equation:

$$\sum_{i=1,k-1} Weight(i, i-1) \tag{1}$$

While this method is guaranteed to find a globally optimum axis ranking, it is NP-complete and therefore computationally intractable for all but the smallest datasets. Even for a dataset with only $N = 63$ variables and $k = 7$ axes, if calculating the optimum layout from 7! axis arrangements took only 1 millisecond, the entire calculation would still take approximately 6.5 days (our result using a 2.1Ghz Intel Core 2). We stopped the global search algorithm for $N = 126$ variables using $k = 7$ axes after running 3 months. This algorithm is embarrassingly parallel in that each *N choose k* set of axes could be passed to a core, permutation tested, and respond back with a fitness which hashes into a sorted data struct. However, we felt that this was unnecessary as approximate algorithms would suffice. For this reason, we implemented some simple alternatives to this brute force approach that typically calculate a layout so near to optimal that the difference is negligible.

4.2. Greedy Pairs Algorithm

Due to the NP-complete nature of the true optimization problem, we developed several approximation algorithms which make various degrees of fitness/time tradeoffs. We include only one such algorithm here as a simple example upon which other algorithms could be based.

When we calculate an axis layout, it would make sense to keep the pairs with the strongest relationship next to one another rather than adding single axes greedily. In this greedy-based algorithm, shown in figure 1, we begin by finding the k largest weights in the graph using the naive selection algorithm in $O(k|V|^2)$ time, where $|V|$ is the number of vertices in the graph (dimensions in the dataset) and k is the number of axes to be shown in the final PCP. While this results in the highest pairwise values, order matters since we would like to string these pairs together in a way that maximizes the total fitness. We choose k pairs, resulting in $2k$ axes, despite only needing k axes because we may have perfect pair overlap. Since each weight has two associated axes, each pair is permuted to find the pairwise sequence which maximizes the sum of weights from the first k consecutive axes (thus selecting those axes discarding any additional axes). This algorithm runs on the order of $k|V|^2 + k!$ and typically performs negligibly close to the pure optimal algorithm. This algorithm stably sorted 734 axes in 3.6 milliseconds on a 2.1Ghz Intel Core 2.

5. Depth-Enhanced PCP Rendering

Traditional rendering of parallel coordinates involves rendering the multidimensional data as a series of polylines. The intersections between the polylines and the parallel axes spatially indicate the values for each observation. Ideally, the viewer's visual cognition system will identify patterns in the lines indicating relationships between variables. However, such a simple display easily becomes cluttered for even small datasets and trends are difficult to discern.

The goal of our research is to automatically make trends highly visible to the user. Accordingly, we have developed a novel method of parallel coordinate rendering that emphasizes variable relationships with easily perceived 3-dimensional cues. Instead of treating each line separately, we render the series of lines as a planar surface and shade each point on the surface according to the number of lines that intersect at the point. Our 3-D approach enables the human perceptual system to quickly parse the display to find interesting trends, which may emerge as ridges or valleys.

Our renderer first rasterizes all polylines and calculates the depth complexity, or the number of times each pixel is drawn into. Pixels of high complexity represent locations where many lines intersect. We then use this depth complexity image to calculate a normal map. The lines are cleared and a single bump-mapped quadrilateral is drawn in their place. Each fragment's normal is retrieved from the normal map texture, and its depth complexity is used to index into an RGBA transfer function. The normal and color are used to perform traditional Phong lighting.

Example renderings of our method for two artificial datasets are shown in figure 2. Occlusion in traditional line renderings often masks or subdues trends. By enhancing the parallel coordinate display with a normal map derived from the depth complexity image, these trends are strongly emphasized through color and specular and diffuse lighting. For the uncorrelated dataset in (a), the depth complexity image has nearly constant slope, yielding slow color changes and flat surfaces in the enhanced rendering. The correlated dataset in (b), however, contains a strong ridge where many observations overlap.

The resulting heightfield can be scanned quickly for prominent features or manipulated by the user. The light source can be translated interactively in three dimensions to investigate the surface cues through shading changes. The opacity of regions of low or high complexity can be modulated with a transfer function widget. Since our approach uses color to denote depth complexity, individual lines are not colored separately. To support this, we allow the user to further modulate opacity with a second transfer function indexed by each line's value on the selected axis.

6. Results and Discussions

We have tried many metrics but will use Pearon's correlation in results for this section as it is the easiest to visually verify. Since our optimization framework maximizes the metric under various constraints, a target of 1 or -1 will yield highly correlated or inversely correlated results. If you set the target to $[-0.2, 0.2]$, then the system will instead show things that are mostly uncorrelated, very dissimilar from [-1,1]. We will showcase some of our results in sections 6.2- 6.3 after first describing a dataset currently undergoing active exploration in section 6.1.

6.1. Climate Simulation Data

We use the greedy pair algorithm to determine an optimal ordering of axes based upon correlation in the following examples. The proposed system is compatible with any multivariate data and the examples presented in this section will utilize climate data. The climate data used here contains 63 physical variables recorded on a monthly basis for 10 years (2000-2009) of IPCC climate simulation. In total, we consider $63 \times 12 \times 10 = 7560$ attribute dimensions by treating time steps independently. Moreover, the simulation grid corresponding to land points constitutes 7,311 polylines for each axis layout. Most systems will not contain so many dimensions, but we wanted to demonstrate the speed and flexibility of our system on such a large, real-world dataset. Due to this large datasize, we use a greedy pairs algorithm for the most timely performance unless otherwise noted. Throughout this paper, we typically use 7 axes based upon the limits of human cognition to 7 units of information for short-term memory [Mil56].

6.2. Ostentatious Patterns

When we use the Pearson's correlation metric on the climate data, the system returns the most highly correlated variables. In this example we begin with only 63 climate variables for January of 2000 and compute the truly globally optimum layout using the correlation metric. As shown in figure 3, the system has detected several variables which are highly correlated. This type of plot verifies the system is working correctly since all these variables are differing measures of temperature that should be roughly correlated; left to right these are: TREFMXAV-maximum average temperature at two meters, ZBOT-temperature and humidity at two meters, TBOT-atmospheric air temperature, THBOT-atmospheric air potential temperature, TV-vegetation temperature, TSA-air temperature at two meters, and TG-ground temperature.

In this example, we add the temporal dimension for an order of magnitude more variables. Since we treat variables from separate timesteps independently, the system has selected a layout which shows the common-sense relation of an attribute's self-correlation across disparate timesteps. As shown in figure 4, snowfall during the summer months of 2000-2009 is somewhat consistent. This example was chosen for a few reasons. First, it highlights the fact that the system can detect patterns which may be surprising and unexpected, prompting the user to create metric variants through constraints. Second, while repeating axes are not allowed in this example, additional removal of correlation with other variables throughout time can be a desirable property and is supported by our system. Third, the years for the layout axes are unordered and a constraint which has time increasing to the right may be more intuitive.

6.3. Constraints for Innate Patterns

By taking the above constraints into account, a user may now be interested in seeing inversely correlated variables (by subtracting the constrained matrix from a matrix of ones). The result can be seen in figure 5. RSSUN/RSSHA are measures of leaf stomatal resistance which is dependent upon the incident photosynthetically active radiation. In this PCP, the system has selected axes which are inversely proportional in equidistant months (alternating April/October) in which the direct rays of the sun are at their maximal relative difference. This example shows that the system can display patterns which are novel and interesting to naive users but which make complete sense to domain-specific experts.

Global warming is a common concern that scientists attempt to verify and understand when looking at climate data. One of many ways to gauge global warming is in the variance of snow depth throughout many years. By using correlation, our system produced the PCP shown in figure 6 which shows a strong correlation in snow depth throughout the years. As may be expected, there are many locations which have no snow (red line at bottom), a few that have a little snow (blue area), and more that are typically covered in snow (green area). However, there are some highlighted ridges in our rendering corresponding to grid points whose snow depth have varied significantly and should be checked for location of important polar ice caps.

7. Conclusion and Future Work

In conclusion, we have provided a general mechanism for the optimized ranking for axis ordering in parallel coordinates visualizations along with algorithms to manage sub-optimal ranking tradeoff for time depending upon data size. We have developed a depth-enhanced parallel coordinate renderer that uses surface cues to more effectively display trends over traditional line drawings. The results demonstrated clearly show the automatic ranking and selection of meaningful patterns in PCP axes for large, time-dependent, multivariate, climate data.

There are several partially completed items that we consider as potential future work. First, formalize work on defining constraints on the bivariate matrix to provide intuitive control for users while pruning the search space. Second, we have already coded up a learning system to help users determine which metrics are important for a specific pattern of interest and now analyzing the results. Third, there are several other graph-based algorithms which could be used for determining an axis ordering such as pairwise shortest path between two specific variables of interest. Fourth, the optimization framework presented here optimizes on the basis of bivariate relationships but a more general multivariate trend detection mechanism would allow the detection of nonlinear (sinusoidal) patterns.

References

[ED06] ELLIS G., DIX A.: Enabling automatic clutter reduction in parallel coordinate plots. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 717–724.

[FL03] FERREIRA M. C., LEVKOWITZ H.: From visual data exploration to visual data mining: A survey. *IEEE Transactions on Visualization and Computer Graphics 09*, 3 (2003), 378–394.

[FWR99] FUA Y. H., WARD M. O., RUNDENSTEINER E. A.: Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings of IEEE Visualization '99* (1999), pp. 43–50.

[GK03] GRAHAM M., KENNEDY J.: Using curves to enhance parallel coordinate visualisations. In *IV '03: Proceedings of the Seventh International Conference on Information Visualization* (2003), pp. 10–16.

[ID90] INSELBERG A., DIMSDALE B.: Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *Proceedings of IEEE Visualization'90* (1990), pp. 361–378.

[ID94] INSELBERG A., DIMSDALE B.: Multidimensional lines i: Representation. *SIAM J. Appl. Math. 54*, 2 (1994), 559–577.

[Ins85] INSELBERG A.: The plane with parallel coordinates. *The Visual Computer 1*, 2 (1985), 69–91.

[JCJ05] JOHANSSON J., COOPER M., JERN M.: 3-dimensional display for clustered multi-relational parallel coordinates. In *IV '05: Proceedings of the Ninth International Conference on Information Visualization* (2005), pp. 188–193.

[JLJC05] JOHANSSON J., LJUNG P., JERN M., COOPER M.: Revealing structure within clustered parallel coordinates displays. In *INFOVIS '05: Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization* (2005), p. 17.

[Mil56] MILLER G. A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review 63*, 2 (1956), 81–97.

[MW91] MILLER J. J., WEGMAN E. J.: Construction of line densities for parallel coordinate plots. In *Computing and Graphics in Statistics* (1991), Springer, pp. 107–123.

[MW02] MOUSTAFA R. E. A., WEGMAN E. J.: On some generalizations of parallel coordinate plots. In *Seeing a Million: A Data Visualization Workshop* (2002).

[Nov06] NOVOTNY M.: Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 893–900.

[PWR04] PENG W., WARD M. O., RUNDENSTEINER E. A.: Clutter reduction in multi-dimensional data visualization using dimension reordering. In *Proceedings of the IEEE Symposium on Information Visualization* (2004), IEEE Computer Society, pp. 89–96.

[SFJKY07] STEED C. A., FITZPATRICK P. J., JANKUN-KELLY T. J., YANCEY A. N.: Practical application of parallel coordinates to hurricane trend analysisy. In *Proceedings of IEEE Visualization'07* (2007).

[Weg90] WEGMAN J. E.: Hyperdimensional data analysis using parallel coordinates. *J. Am. Stat. Assn 85*, 411 (1990), 664–675.

[WL96] WEGMAN E. J., LUO Q.: *High Dimensional Clustering Using Parallel Coordinates and the Grand Tour*. Tech. Rep. 124, 1996.

[WLG97] WEGENKITTL R., LOFFELMANN H., GROLLER E.: Visualizing the behavior of higher dimensional dynamical systems. In *Proceedings of IEEE Visualization '97* (1997), pp. 119–126.

[ZYCQ07] ZHOU H., YUAN X., CHEN B., QU H.: Visual clustering in parallel coordinates. In *Proceedings of IEEE Visualization'07* (2007).

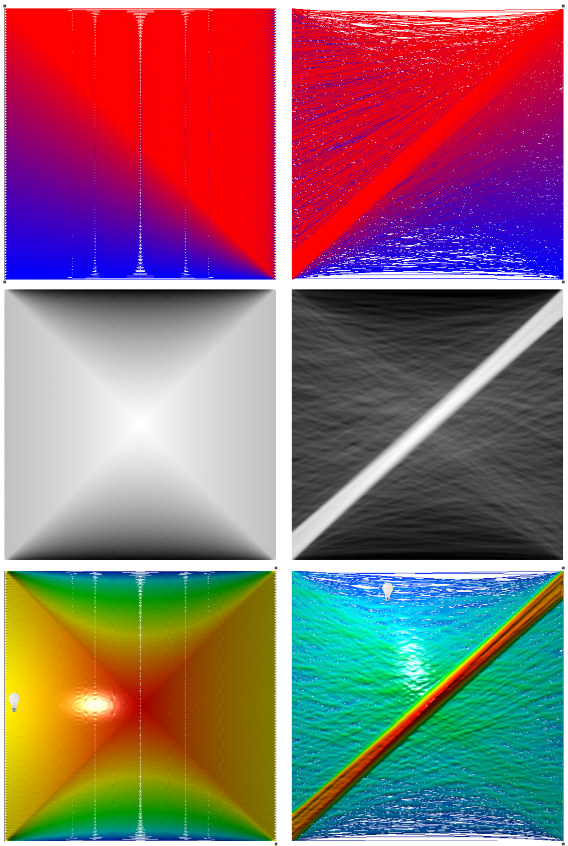
```
int getFitness(int numAxes,
               int* layout, graph* g) {
  int i, v1,v2, fitness=0;
  for(i=1; i<numAxes; i++) {
     v1=layout[i-1]; v2=layout[i];
     fitness += g->adjmat[v1][v2];
  }
  return fitness;
}

int getHighestEdges(int numAxes, graph* g,
     int** tX, int** tY, int allowRepeats) {
  int i,j,k, val,max, axis,got;
  int tarX[numAxes], tarY[numAxes];
  for(axis=0; axis<numAxes; axis++) {
     max=SHRT_MIN; // find k best pairs
     for(i=0;i<g->numVerts;i++)
       for(j=0;j<i;j++) { //entire matrix
         got=check(tarX,tarY,i,j,allowRepeats);
         val = g->adjmat[i][j]; //new max?
         if (val!=INAN && got==0 && max<val) {
             max=val; tarX[axis]=j; tarY[axis]=i;
         }//end max check
     }//end O(V^2)
  }//end naive k best pairs O(kV^2)
  *tX = tarX; *tY = tarY; return 0;
}

int* maxPermPairs(int num, int* tarX, int* tarY,
                  graph* g) {
   int i,k, max, fitness = -1;
   int a[num], layout[num+1],bestLayout[num+1];
   for(i=0; i<num; i++) a[i]=i;
   for_all_permutations_of_a[i] {
      for(k=0;k<(num+1)/2;k++) { //best perm?
        layout[2*k  ] = tarX[a[k]];
        layout[2*k+1] = tarY[a[k]];
      }
      if((num+1)%2==1)
        layout[num]=tarX[a[(num+1)/2]];
      fitness=getFitness(num+1,layout,g);
      if(fitness>max) {
         max = fitness;
         memcpy(bestLayout,layout);
      }
   } // End O(num!)
   return bestLayout;
}

int* greedyPairs(int numAxes=7, graph* g) {
   int *tarX, *tarY;
   getHighestEdges(numAxes, g, &tarX, &tarY, 0);
   return maxPermPairs(numAxes, tarX, tarY, g);
}
```

Figure 1: *Pseudocode for the quick, near-optimal greedy pairs algorithm.*



(a) Uncorrelated (b) Highly Correlated

Figure 2: *Detecting trends in parallel coordinate displays made easier with 3-D surface cues.* ***(top)*** *Traditional line rendering of two generated datasets. Column (a) represents an extremely uncorrelated dataset where every data item on the first axis is connected to every data item on the second. Column (b) is a dataset where half of the observations are randomly generated and half are randomly offset from an inverse relationship.* ***(middle)*** *Depth complexity images of the line renderings in which white indicates a high number of intersecting lines.* ***(bottom)*** *Our method of PCP rendering with surface cues. The line rendering is bump-mapped using the depth complexity image.*

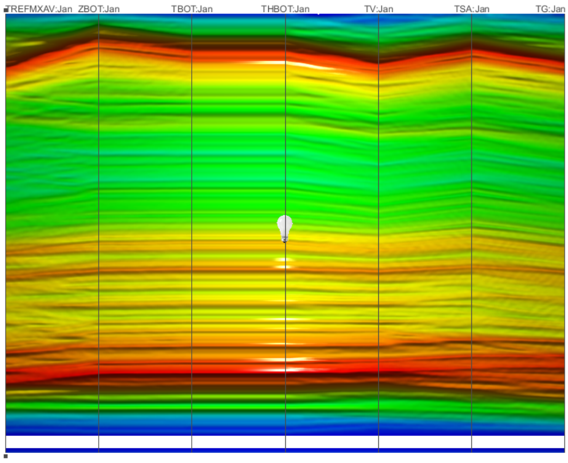


Figure 3: *The system finds a strong correlation between various measures of temperature in Jan'00.*

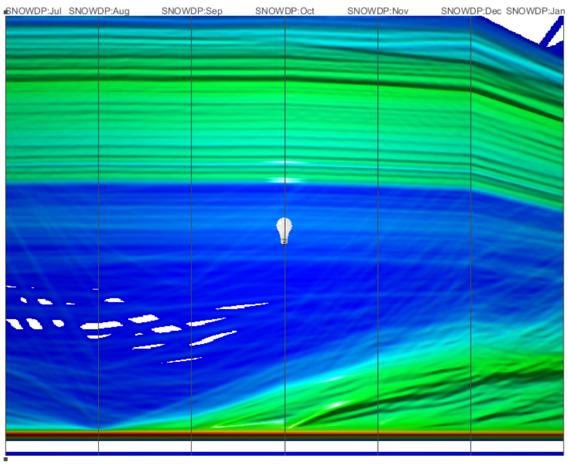


Figure 4: *Constraints are included to keep the system from finding repeated results of self-correlation through time.*

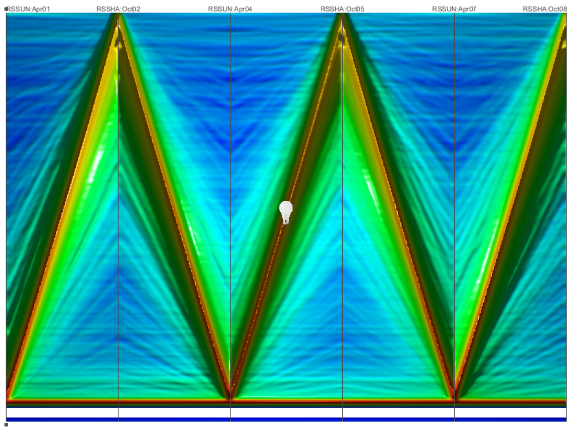


Figure 5: *Inverse correlation with consistent time constraints which relates the variance of radiation intensity on leaves as a function of the earth's tilt throughout the seasons.*

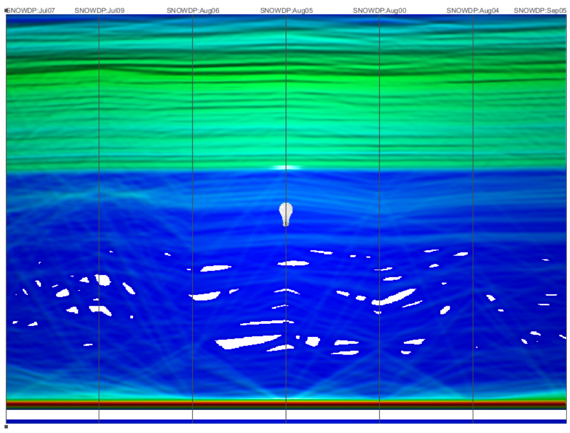


Figure 6: *One way of measuring global warming showing strong correlation of snow depth between years. Our rendering technique also shows V-shaped highlights corresponding to grid locations that my warrant further investigation for snow/ice melting.*