UNIT-3

Unit-3:

Visualization Techniques: Spatial Data: One-Dimensional Data - Two-Dimensional Data – Three-Dimensional Data - Dynamic Data - Combining Techniques.

Geospatial Data: Visualizing Spatial Data- Visualization of Point Data -Visualization of Line Data-Visualization of Area Data - Other Issues in Geospatial Data Visualization

Multivariate Data: Point-Based Techniques - Line- Based Techniques -Region-Based Techniques-Combinations of Techniques – Trees Displaying Hierarchical Structures – Graphics and Networks-Displaying Arbitrary Graphs/Networks.

References:

1. Matthew Ward, Georges Grinstein and Daniel Keim, : Interactive Data Visualization Foundations, Techniques, Applications “,2010

Unit -3 Part-2 notes

2.Geospatial data

Geospatial data is different from other kinds of data in that spatial data describes objects or phenomena with a specific location in the real world. Geospatial data arises in many applications, including credit card payments, telephone calls, environmental records, and census demographics. In this chapter, we provide an overview of the special characteristics and methods that are needed for the visualization of geospatial data, sometimes called *geovisualization*.

2.1 Visualizing Spatial Data

Large spatial data sets can be seen as a result of accumulating samples or readings of phenomena in the real world, while moving along two dimensions in space. Often, spatial data sets are discrete samples of a continuous phenomenon. Nowadays, there exists a large number of applications where it is important to analyze relationships that involve geographic location.
Examples include global climate modeling (e.g., measuring temperature, rainfall, and wind speed), environmental records (e.g., measuring CO2 and other pollution levels), economic and social measures and indicators (e.g., unemployment rate, education level), customer analysis, telephone calls, credit card payments, and crime data. Because of its special characteristics, the basic visualization strategy for spatial data is straightforward; we map the spatial attributes directly to the two physical screen dimensions, resulting in map visualizations.
Maps are the world reduced to points, lines, and areas. The visualization parameters, including size, shape, value, texture, color, orientation, and shape, show additional information about the objects under consideration. According to the U.S. Geological Survey (USGS), map visualizations are defined as a set of points, lines, and areas, all defined both by position reference to a coordinate system (spatial attributes) and by their nonspatial attributes. MacEachren defines geographic visualization as the use of visual

representations to make spatial contexts and problems visible, so as to
engage the most powerful human processing abilities, those associated with
vision .
From the definitions, it becomes clear that we may distinguish spatial
phenomena according to their spatial *dimension* or *extent*:
point phenomena—have no spatial extent; they can be termed zerodimensional
and can be specified by a longitude and latitude coordinate
pair, along with a set of descriptors or attributes. Examples are
buildings, oil wells, aggregated measures, and cities.

- *line phenomena*—have length, but essentially no width; they can be
termed one-dimensional and can be specified by an unclosed series of
longitude and latitude coordinate pairs for each phenomenon. Examples
are large telecommunication networks, roads, and boundaries
between countries. Attributes associated with line phenomena might
include capacities, traffic levels, and names.
- *area phenomena*—have both length and width; they can be termed
two-dimensional and can be specified by a series of longitude and latitude
coordinate pairs that completely enclose a region, along with a
set of attributes for each phenomenon. Examples are lakes, parks, and
political units such as states or counties.
- *surface phenomena*—have length, width, and height; they are termed
two-and-half-dimensional and can be specified by a series of longitude, latitude, and height
coordinate vectors with a set of attributes for each
(longitude, latitude) pair.

Maps can be subdivided into map types based on properties of the data
(qualitative versus quantitative; discrete versus continuous) and the properties
of the graphical variables (points, lines, surface, volumes). Examples of
the resulting maps are

- symbol maps (nominal point data);
- dot maps (ordinal point data);
- land use maps (nominal area data);
- choropleth maps (ordinal area data);
- line diagrams (nominal or ordinal line data);
- isoline maps (ordinal surface data);
- surface maps (ordinal volume data).

Note that the same data may be visualized by different map types. By
aggregating point data within areas, a choropleth map may be generated out
of a dot map, or a land use map out of a symbol map. We may also generate
a density surface from a dot map and display it as an isoline map or a surface
map. If we aggregate the point data within areas and map the number of
points within the areas to their size, we obtain cartogram visualizations.

In exploratory geovisualization, interaction with maps is crucial. In contrast
to traditional cartography, the classification and mapping of the data
can be interactively adapted by the user, and interactive querying as well as
manipulation of the display are possible .. A number of techniques and
systems have been developed that make extensive use of such interaction
capabilities. They allow, for example, a linking of multiple maps or a combination
of maps with standard statistical visualizations, such as bar charts
and line charts, or even with complex multidimensional visualization techniques
such as parallel coordinates or pixel techniques. In

addition, they usually provide an advanced browsing or querying interface.

2.2 Visualization of Point Data

The first important class of spatial data is *point data*. Point data are discrete in nature, but they may describe a continuous phenomenon, for example, temperature measurements at specific locations. Depending on the nature of the data and the task, the designer must decide whether to display the data continuously, versus discrete and smooth, versus abrupt. Figure show the different options. Discrete data are presumed to occur at distinct locations, while continuous data are defined at all locations. *Smooth data* refers to data that change in a gradual fashion, while *abrupt data* change suddenly.

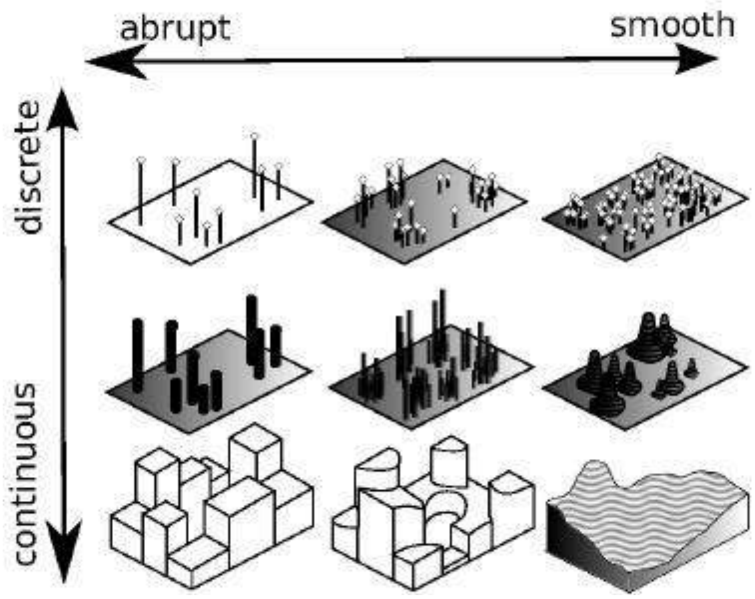


Fig : Discrete versus continuous and smooth versus abrupt

A) DOT MAPS :

Point phenomena can be visualized by placing a symbol or pixel at the location where that phenomenon occurs. This simple visualization is called a *dot map*. A quantitative parameter may be mapped to the size or the color of the symbol or pixel. Circles are the most widely used symbol in dot maps, but squares, bars, or any other symbol can be used as well. If the size of the symbol is used to represent a quantitative parameter, a specific question is how to scale the symbols. Calculating the correct size of the symbols does not necessarily mean that the symbols will be perceived correctly [129]. The perceived size of the symbols does not necessarily correspond to the actual size, due to problems in size perception). The perceived size of the symbols depends on their local neighborhood (e.g., the Ebbinghaus illusion , therefore no global formula for perceptual scaling is possible.

maps are an elegant medium for communicating a wealth of information about the relationships of spatial point phenomena in a compact, convenient, and familiar format. However, when large data sets are drawn on a map, the problem of overlap or overplotting of data points arises in highly populated areas, while low-population areas are virtually empty see Figure), since spatial data are highly nonuniformly distributed in realworld data sets. Examples of such spatial data sets are credit card payments, telephone calls, health statistics, environmental records, crime data, and census demographics. Note that the analysis may involve multiple parameters that may be shown on multiple maps. If all maps show the data in the same way, it may be possible to relate the parameters and detect local correlations, dependencies, and other interesting patterns.

B) PixelMaps:

One approach that does not aggregate the data, but avoids overlap in the two-dimensional display, is the *PixelMap* approach [230]. The idea is to reposition pixels that would otherwise overlap. The basic idea of the repositioning algorithm is to recursively partition the data set into four subsets containing the data points in four equal-sized subregions. Since the data points may not fit into the four subregions, we must determine new extents of the subregions (without changing the four subsets of data points), such that the data points in each subset can be visualized in their corresponding subregion. For an efficient implementation, a quadtree-like data structure manages the required information and supports the recursive partitioning process. The partitioning process works as follows. Starting with the root of the quadtree, in each step, the data space is partitioned into four subregions. The partitioning is made such that the area occupied by each of the subregions (in pixels) is larger than the number of pixels belonging to the corresponding subregion. If—after a few recursions—only a limited number of data points are left in a subregion, the points are positioned by a pixel placement algorithm that positions the first data item at its correct position, and subsequent overlapping data points at nearby unoccupied positions, resulting in a placement that appears locally quasi-random. The details of the algorithm can be found in [230]. A problem of PixelMaps is that in areas with high overlap, the repositioning depends on the ordering of the points in the database. Figure 6.13 presents four time steps of such visualizations, showing the U.S. Telephone Call Volume within a 10-minute interval of the time denoted. The time sequence shows the development of the call volume over time. The visualizations allow an intuitive understanding of the development of the call volume, showing the wake-up from east to west and the drop down in call volume at commuting and lunch time.

2.3 Visualization of Line Data

The basic idea for visualizing spatial data describing linear phenomena is to represent them as line segments between pairs of endpoints specified by longitude and latitude. A standard mapping of line data allows data parameters to be mapped to line width, line pattern, line color, and line labeling. In addition, data properties of the starting and ending points, as well as intersection points, may also be mapped to the visual parameters of the nodes, such as size, shape, color, and labeling. The lines do not need to be straight, but may be polylines or splines, in order to avoid clutter in the display. Which mapping is best depends on the application and the task.

A) Network Maps

Network maps are widely used in a variety of applications. Some approaches only display the connectivity of networks for understanding their general behavior and structure. Eick, and Wills [107] used functions such as aggregation, hierarchical information, node position, and linked displays for investigating large networks with hierarchies and without a natural layout. They used color and shape for coding node information and color and line width for coding link information. Researchers at NCSA [305] added 3D graphics to their network maps to display animations of Internet traffic packets within the network backbone. Becker, Eick and Wilks [28] describe a system called *SeeNet* , which is motivated by research in dynamic statistical graphics. The basic idea is to involve the human and let him/her interactively control the display to focus on interesting patterns. They use two static network displays to visualize the geographic relationships, and a link matrix, which gives equal emphasis to all network links. Another interesting system for visualizing large network data is AT&T's SWIFT-3D System [224, 253, 254]. This system integrates a collection of relevant visualization techniques, ranging from familiar statistical displays to pixel-oriented overviews, with interactive 3D-maps and drag+drop query tools (see Figure)

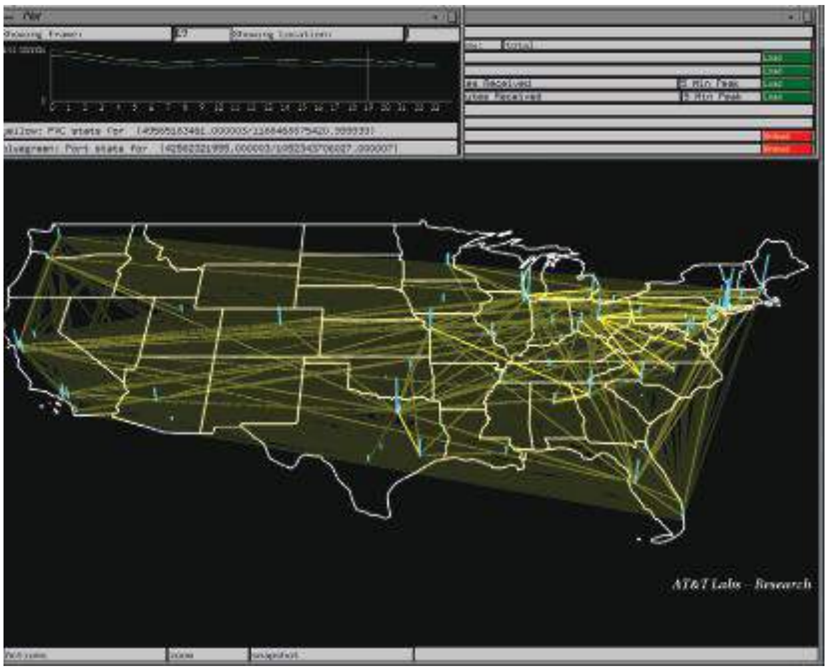


Fig: Swift -3D.

(b) Flow Maps and Edge Bundling

There are a number of approaches that try to avoid the overlap problem of traditional network maps by using curved lines instead of straight lines (see Figure 6.15). While this has been done mostly manually by cartographers in the past, a number of approaches for an algorithmically generated visualization of network maps with curved lines have been proposed. Two prominent examples are Stanford flow maps [318] and Danny Holten's edge bundling .

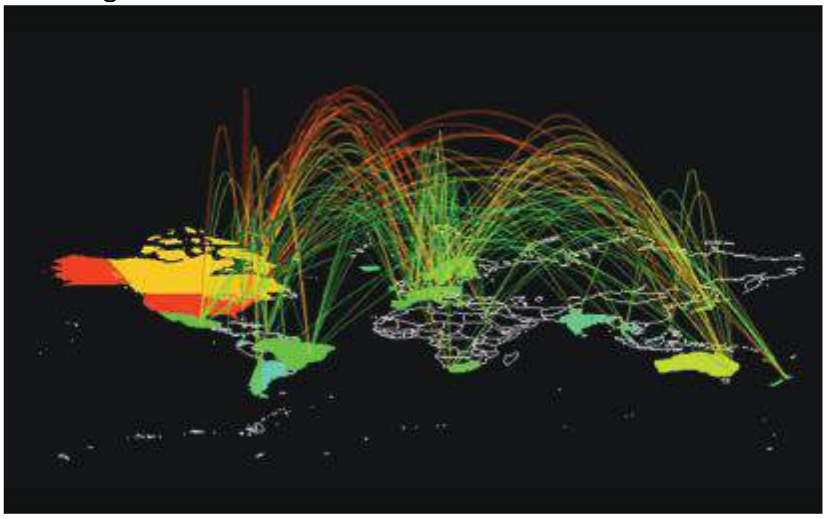


Fig: Arc Map

The *flow map* technique is inspired by graph layout algorithms that minimize edge crossings and node position distortions, while retaining their relative positions. Algorithmically, a hierarchical clustering based on node positions and flows between the nodes is performed to compute a useful merging and rerouting of flows. Details can be found in [318]. In comparison to other computer-generated flow maps (Figure 6.16(a)), the results of the Stanford system show a much clearer picture, with the clutter being significantly reduced (Figure)

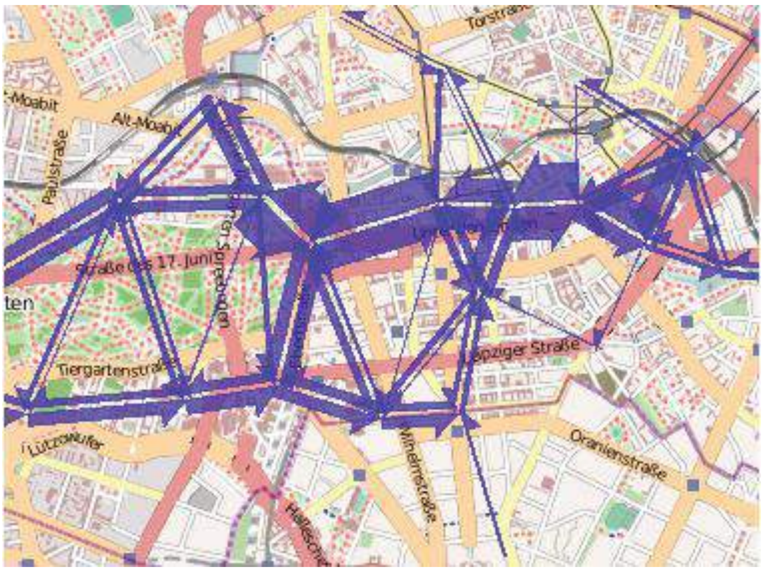


Fig: Flow maps : Flow of tourists in Berlin

Edge bundling also aims at reducing the clutter in line drawings. If a hierarchy is defined on the nodes, the edges can be bundled according to the hierarchy by using the hierarchy as defining points for B-splines connecting two nodes. Nodes connected through the root of the hierarchy are maximally bent, while nodes within the same subhierarchy are only minimally bent. Hierarchical bundling significantly reduces visual clutter. It is a generic method that can be used in conjunction with a number of visualization techniques, including traditional maps, but also standard tree visualizations, circular trees, treemaps, and many others. The example in Figure 6.17 displays edge bundling being applied to a visualization of IP traffic data.

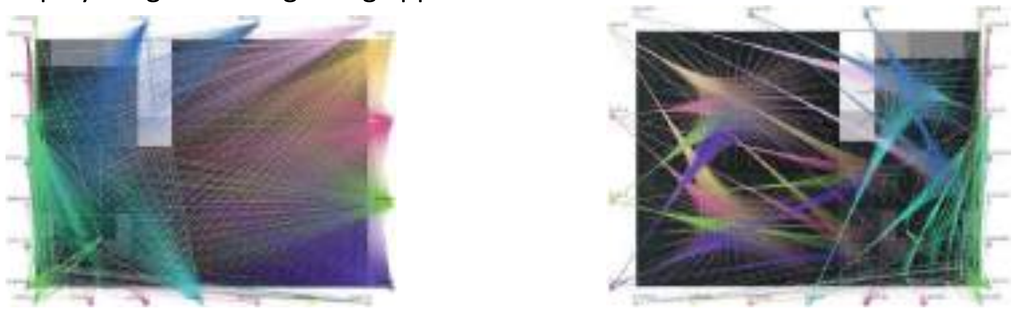


Fig:The visualizations show IP flow traffic from external nodes on the outside to internal nodes, visualized as treemaps on the inside. The edge bundling visualization(right side) significantly reduces the visual clutter compared to the straight line visualization (left side).

2.4 Visualization of Area Data:

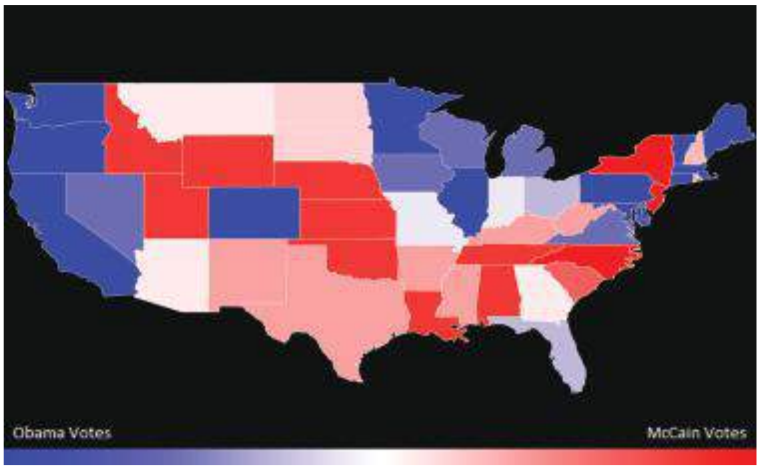
Thematic maps are the main approach to visualizing area phenomena. There are different variants of thematic maps. The most popular type of thematic maps are *choropleth maps* (Greek: choro = area, pleth = value), in which the values of an attribute or statistical variable are encoded as colored or shaded regions on the map. Choropleth maps (Figure 6.18(a)) assume that the mapped attribute is uniformly distributed in the regions. If the attribute has a different distribution than the partitioning into regions, other techniques, such as dasymetric maps, are used. In *dasymetric maps* (Figure 6.19), the variable to be shown forms areas independent of the original regions, e.g., in contrast to choropleth maps, the boundaries of the areas derived from the attribute do not need to match the given map's regions. A third important type of map is an *isarithmic map* (Figure 6.18(b)), which shows the contours of some continuous phenomena. Widely used examples of isarithmic maps are contour maps or topographic maps. If the contours are determined from real data points (such as temperatures measured at a specific location) the maps are called *isometric maps*; if the data are measured for a certain region (such as a county) and, for example, the centroid is considered as the data point, the maps are called *isopleth maps*. One of the main tasks in generating isarithmic maps is the interpolation of the data points to obtain smooth contours, which is done, for example, by triangulation, or inverse distance mapping. A complex, but less frequently used mapping technique, is *cartograms*, in which the size of regions is scaled to reflect a statistical variable, leading to unique distortions of the map geometry. There are different variants of cartograms, ranging from continuous cartograms that retain the topology of the polygon mesh, to noncontinuous cartograms that simply scale each polygon independently to rectangular or circular approximations of the areas..

A) Choropleth Maps

Choropleth maps usually present the area phenomena as shaded polygons that are closed contours of sets of points, where the first and the last points are the same. Examples of closed contours are states, counties, and parks. Choropleth maps are used to emphasize the spatial distribution of one or

more geographic attributes. In generating choropleth maps, the normalization of the data, as well as color or grayscale mapping
, are important design decisions. In Figure), a choropleth map showing the 2008 presidential election results is shown.

A problem of choropleth maps is that the most interesting values are often concentrated in densely populated areas with small and barely visible polygons, and less interesting values are spread out over sparsely populated areas with large and visually dominating polygons. Choropleth maps, therefore tend to highlight patterns in large areas, which may, however, be of lower importance. In the U.S. Census Demographics, for example, such maps tend to highlight patterns in areas where few people live, e.g., the large states in the USA.

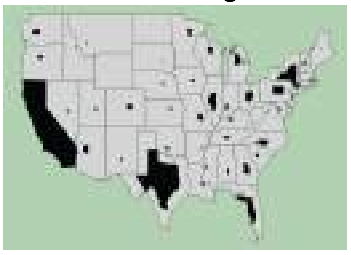


A choropleth map showing U.S. election results of the 2008 Obama versus McCain presidential election.

B) **Cartograms:**

Cartograms are generalizations of ordinary thematic maps that avoid the problems of choropleth maps by distorting the geography according to the displayed statistical value. Cartograms are a specific type of map transformation, where the regions are resized according to a geographically related input variable. Example applications include population demographics election results , and epidemiology .
Several categories of cartogram problems exist. As shown in Figure
. *noncontinuous cartograms* can exactly satisfy area and shape constraints, but don't preserve the input map's topology. Because the scaled
polygons are drawn inside the original regions, the loss of topology doesn't cause perceptual problems. More critical is that the polygon's original size restricts its final size. Consequently, you can't make small polygons arbitrarily large without scaling the entire map, so important areas can be difficult to see, and screen usage can be poor. *Noncontiguous cartograms*, shown in Figure 6.20(b), scale all polygons to their target sizes, perfectly satisfying the area objectives. Shapes can be slightly relaxed, so polygons touch without overlapping, and the map's topology is also highly relaxed, because polygons don't retain their adjacency relationships. Noncontiguous cartograms provide perfect area adjustment, with good shape preservation. However, they lose the map's global shape and topology, which can make perceiving

the generated visualization as a map difficult. *Circular cartograms*, shown in Figure 6.20(c), completely ignore the input polygon's shape, representing each as a circle in the output. In many cases, area and topology constraints are also relaxed, so circular cartograms have some of the same problems as noncontiguous cartograms. The final category is *continuous cartograms*, shown in Figure 6.20(d). Unlike the other categories, continuous cartograms retain a map's topology perfectly, but they relax the given area and shape constraints. In general, cartograms can't fully satisfy shape or area objectives, so cartogram generation involves a complex optimization problem in searching for a good compromise between shape and area preservation. Although continuous cartograms are difficult to generate, the resulting polygonal meshes resemble the original map more than other computergenerated cartogram variants. The rest of this section therefore focuses on continuous cartograms.



2.5 Other Issues in Geospatial Data Visualization

Prerequisite to generating useful visualizations of spatial phenomena are a number of well-known techniques from cartography, including map general ization and map labeling. *Map generalization* is the process of selecting and abstracting information on a map. Generalization is used when a specialpurpose small-scalemap is derived from a large-scalemap containing detailed information. The goal is to adapt the objects on the map in such a way that the objects of interest may be easily perceived in the resulting visualization. Note that map generalizations are application- and task-dependent, e.g., good map generalizations emphasize the map elements that are most important for the task at hand, while still representing the geography in the most accurate and recognizable way. It is important that the visibility and recognizability of the objects displayed on the map outweigh the lost details of items that are generalized. The goal is to preserve the distinguishing characteristics of what makes the map useful in the particular application Examples for typical map generalizations are:

- *Simplify points*—remove or combine points that are not relevant or not separately visible on the small-scale map.
- *Simplify lines*—remove small fluctuations and bends or collapse dual lines to centerlines. Lines that are in danger of touching each other in the scaled version of the map may be removed.
- *Simplify polygons*—remove small fluctuations and bends while preserving the essential shape feature. This includes the removal or simplification of building footprint boundaries while preserving the essential

shape and size, as well as the combination of disjoint but adjacent polygons into a new area based on a specified tolerance distance.
Map labeling deals with placing textual or figurative labels close to points, lines, and polygons. This seems to be an easy and straightforward task, but it has been shown to be a difficult problem [95, 131]. There are a large number of label placement algorithms that differ in their effectiveness, e.g., quality of the results, and their efficiency, e.g., speed of calculating the label placement. Map labeling algorithms are based on heuristics, and in most cases use a rule-based label placement followed by an optimization algorithm, such as local search, greedy algorithms, simulated annealing, random placement, and genetic algorithms [75]. Two examples of labeling software are Label-EZ [133] by MapText, Inc. and Maplex by ESRI [117].
Many other issues are related to the design of effective *geographic information systems* (GIS). A GIS essentially allows users to create interactive queries for dynamic search and exploration, to compare and edit spatial data on different geographic map layers, and finally to present the results of all these operations. Geospatial data visualization is just a part of GIS. The advance of visualization and other relevant GIS functions could benefit each other to make the whole system more powerful and useful.
In recent years, with the advance of fast-growing web technology and APIs (e.g., Flex, AJAX, and the Google MAP API), as well as public availability of digitized geographic data and various economic, social, environmental measures and indicators data via the Internet, the GIS community has developed a large number of interactive high-performance tools for spatial data visualization. These tools significantly increase the awareness and better understanding of public issues among large populations of people. It turns out that the visualization of geospatial data has become exciting to many people.

3.MULTIVARIATE DATA

In this chapter we discuss techniques for the visualization of data that does not generally have an explicit spatial attribute. We organize the presentations based on the graphical primitive used in the rendering, namely, points, lines, or regions, followed by techniques that combine two or more of these types of primitives.

1.Point-Based Techniques

Point plots are introduced as visualizations that project records from an n-dimensional data space to an arbitrary k-dimensional display space, such that data records map to k-dimensional points. For each record, a graphical representation, mark, or other aesthetic entity is drawn at its associated k-dimensional point. Individual visualization techniques identified as point plots define appropriate data projections and specific visual representations. Point plots can be defined to display individual records or summary records, and can be structured by various projection techniques.

A) Scatterplots and Scatterplot Matrices

scatterplots are one of the earliest and most widely used visualization techniques in data analysis. Both 2D and 3D scatterplots can be found in most packages designed to support data and information analysis. Their success stems from our innate abilities to judge relative position within a bounded space.

As the dimensionality of the data increases, the choices for visual analysis consist of:

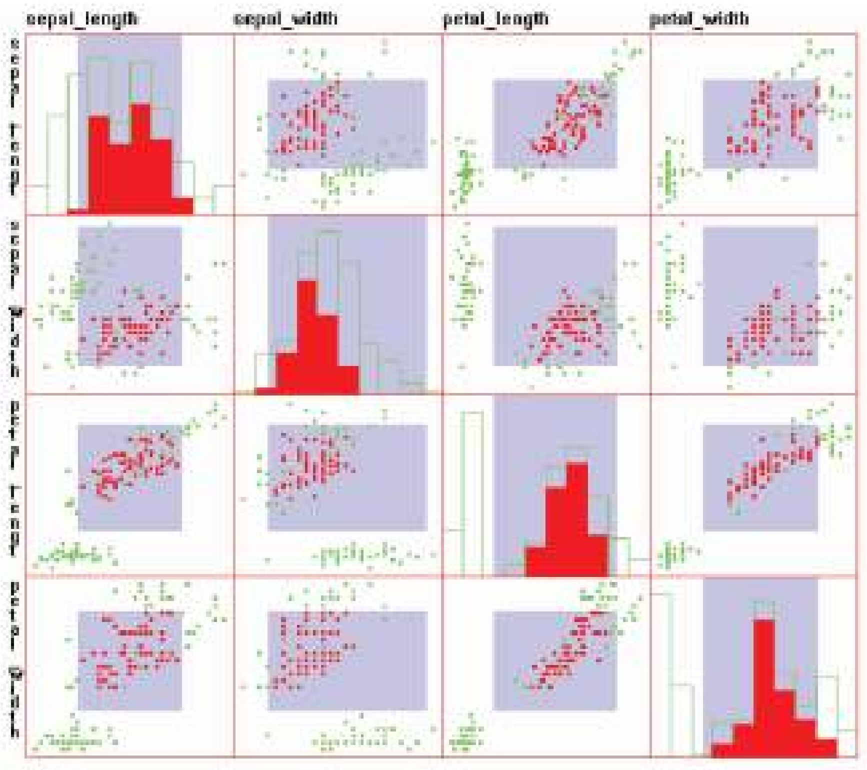
- **dimension sub setting**—allowing the user to select a subset of the dimensions to display, or to develop algorithms to find the dimensions containing the most useful information for the task at hand.

- **dimension reduction**—using techniques such as principal component analysis or multidimensional scaling to transform the high-dimensional data to data of lower dimension, while attempting to preserve as best as possible the relationships among the data points.

- **dimension embedding**—mapping dimensions to other graphical attributes besides position, such as color, size, and shape (though there are limits to how many dimensions can be included this way.

- **multiple displays**—showing, either superimposed or juxtaposed, several plots, each of which contains some of the dimensions.

For the case of multiple displays, the most common approach is to use a scatterplot matrix. This consists of a grid of scatterplots, with the grid having N2 cells, where N is the number of dimensions. Thus, every pairwise plot will be shown twice, differing by a 90 degree rotation. The ordering of the dimensions is usually the same for the horizontal and vertical orientations, resulting in symmetry along the diagonal. The diagonal plots, which would normally plot a variable against itself, are often used to convey the names of the dimensions in the corresponding rows/columns, or sometimes to show a histogram of a given dimension. Figure 8.1 shows an example of a scatterplot matrix.



8.1. A scatterplot matrix with the diagonal plot showing a histogram of each dimension. Note that the points and histogram regions in red indicate selected data.

B.Force-Based Methods

Many techniques for projecting high-dimensional points into 2D or 3D display space have been developed. The key goal is to attempt to maintain the N-dimensional features and characteristics of the data through the projection process, e.g., relationships that exist in the original data must also exist after projection. This, however, is not always possible, especially as the dimensionality of the data increases. The projection may also unintentionally introduce artifacts that may appear in the visualization and are not present in the data. In this section, we describe several such projection methods.

Multidimensional scaling (MDS) is an important class of dimension reduction algorithms commonly used in statistical analysis and information visualization. The basic structure of a typical MDS algorithm is as follows:

1. Given a data set with M records and N dimensions, create an M by M matrix, Ds, that contains similarity measures between each pair of data items. For example, one might use Euclidean distance as a measure of similarity.

2. Assuming that you are projecting the data into K dimensions (e.g., for display purposes, K is usually between 1 and 3), create an M by K matrix, L, to contain the locations for the projected points. These M locations can initially be randomly chosen, or techniques such as principal component analysis (PCA) can be used to create reasonable initial positions.

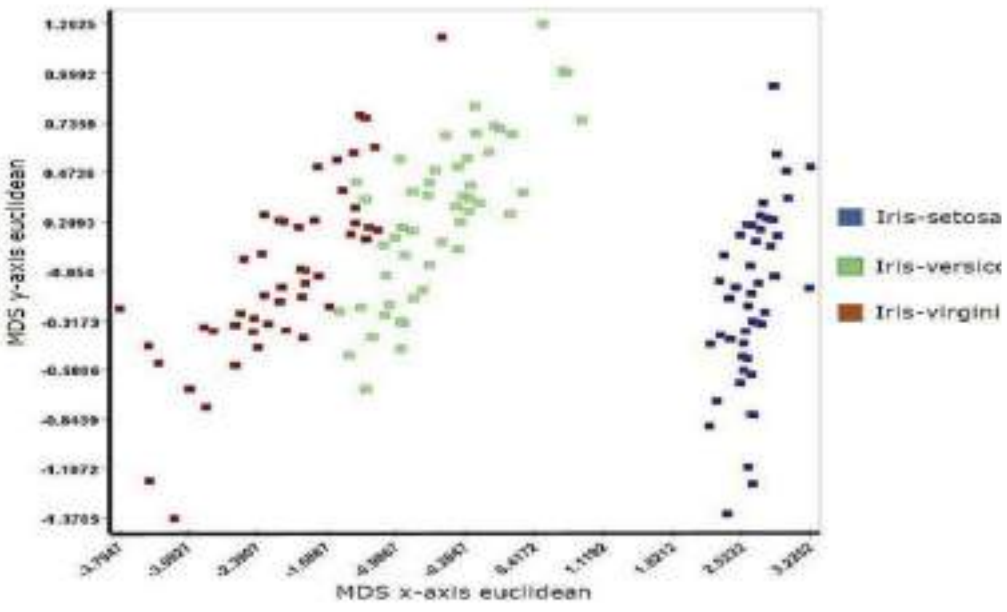
3.Compute an M by M matrix, Ls, that contains the similarities between all pairs of points in L.

4. Compute the value of stress, S, which is a measure of the differences between Ds and Ls. Many such stress measures exist; most assume that the coordinate systems have been normalized so that the maximum distance between points is 1.0.

5. If S is sufficiently small, or hasn’t changed significantly in recent iterations, the algorithm terminates.

6. Otherwise, attempt to shift the positions of points in L in a direction that will reduce their individual stress levels. For example, this might be a weighted sum of displacements based on comparing the point with all other points, or perhaps only with its nearest neighbors. The displacement should be scaled such that points don’t oscillate between positions.

  1. Return to step 3



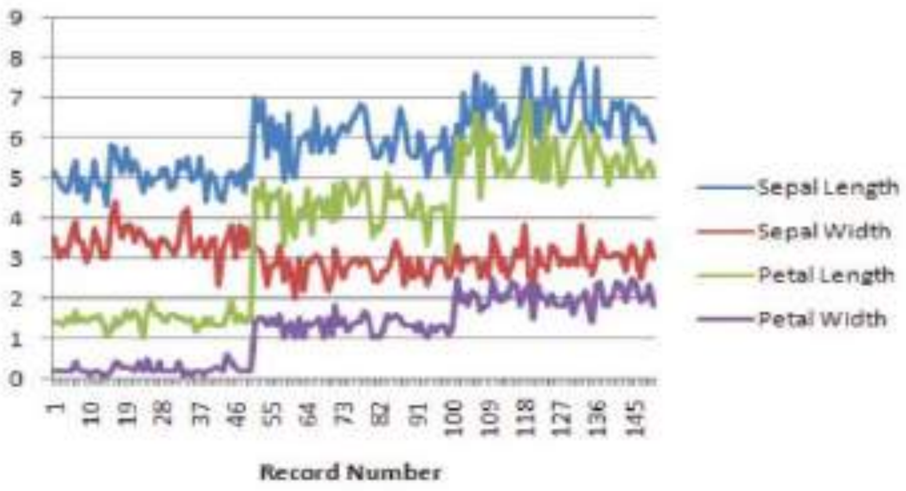
Iris data set projected using MDS.

2.Line-Based Techniques

Point-based methods represent each data value or record with a small mark. In line-based methods, points corresponding to a particular record or dimension are linked together with straight or curved lines. These lines not only reinforce the relationships among the data values, but also convey perceivable features of the data via slopes, curvature, crossings, and other line patterns. We describe a couple of techniques within this class of methods.

A. Line Graphs

A line graph is a univariate visualization technique where the vertical axis represents the range of values for the variable and the horizontal axis represents some ordering of the records in the data set. Most univariate visualization techniques can be extended to multivariate data by either superimposing or juxtaposing the visual representations of individual variables. A commonly used method of this sort is via line graphs. For a modest number of data dimensions, the line plots can be drawn on a common set of axes, differentiating the dimensions using color, line style, width, or other graphical attributes (see Figure).

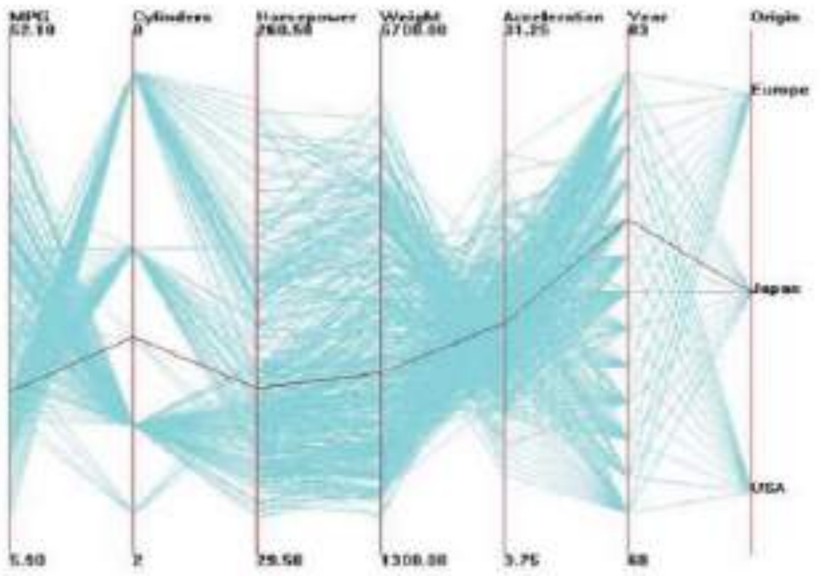


An example of a multivariate line chart, in this case, the four-dimensional Iris data set. Note that the modest separation between dimensions makes it easy to identify trends and outliers.

When the number of dimensions increases, or the dimensions have significant overlap in their data ranges, superimposing becomes more problematic.

B. Parallel Coordinates

Parallel coordinates, also called ||-coords and PCP (for parallel coordinates plot), were first introduced by Inselberg in 1985 as a mechanism for studying high-dimensional geometry. Since then, numerous researchers have studied and enhanced PCPs for use in multivariate data analysis. The basic idea is that axes, rather than being orthogonal, are parallel, with evenly spaced vertical or horizontal lines representing a particular ordering of the dimensions. A data point is plotted as a polyline that crosses each axis at a position proportional to its value for that dimension. Figure shows an example. One could consider a PCP as a line graph after rotating the data, since the values of a record are linked together, as opposed to the values of a dimension.



An example of a 7-dimensional data set visualized with parallel coordinates. single data point is represented as the darkened polyline.

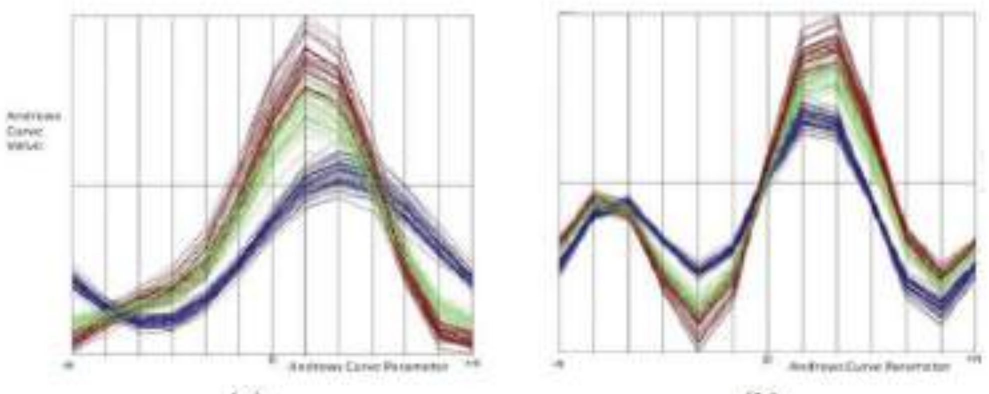
To interpret the plot, one looks for clusters of similar lines (indicating partial correlation between pairs of dimensions), similar crossing points (indicating partial negative correlations), and lines that are either isolated or have a slope that is significantly different from their neighbors (indicating outliers). One problem is that, like scatterplots, parallel coordinates have their strength in showing relationships between pairs of dimensions. To extend this capability, interactive selection and highlighting of records allows users to see relationships that span all dimensions.

C. Andrews Curves

Another line-based visualization for multivariate data is the Andrews curve, developed by David F. Andrews in 1972 . Each multivariate data point D = (d1, d2,...,dN) is used to create a curve of the form

$$f(t) = \frac{d_1}{\sqrt{2}} + d_2 \sin(t) + d_3 \cos(t) + d_4 \sin(2t) + d_5 \cos(2t) + \ldots .$$

For an odd number of dimensions, the final term is dN cos(N−1 2 t), while for an even number of dimensions the final term is dN sin(N 2 t). As in many multivariate visualization techniques, the order of the dimensions can have a significant effect on the resulting Andrews curve. Figures 8.10(a) and 8.10(b) show the same data with different dimension orders. In particular, outliers may become more or less perceivable, depending on the ordering



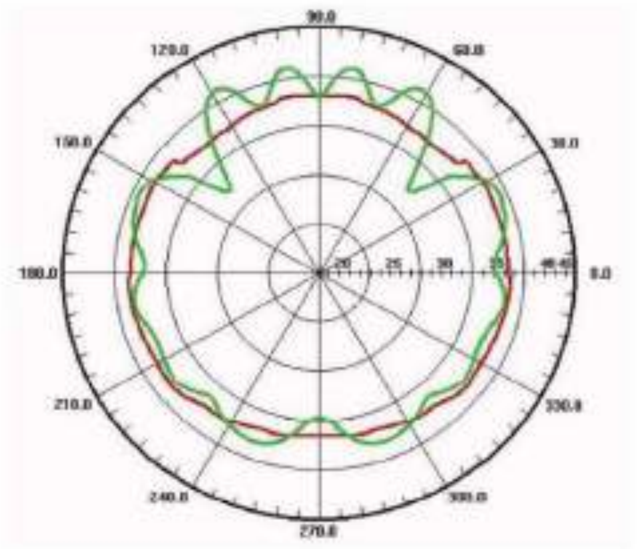
An example of Andrews curves using two different dimension orders: (a) based on the original order of the dimensions (sepal length, sepal width, petal length, petal width); (b) based on the original order of the dimensions in reverse order.

D).Radial Axis Techniques

For each of the techniques that orient the coordinate systems horizontally and/or vertically, there is an equivalent technique that uses a radial orientation. For example, a circular line graph is one in which the plotted lines are offset from a circular base (see Figure). A long graph can be nested by dividing it up into equal size segments and mapping each to a base of different radius. This is a potentially useful way to study cyclic events. Variants on circular line graphs include radar and star graphs.

Many other circular charts have been developed over the years, including:

• polar graphs—point plots using polar coordinates;
• circular bar charts—like circular line graphs, but plotting bars on the base line
• circular area graphs—like a line graph, but with the area under line filled in with a color or texture;
• circular bar graphs—with bars that are circular arcs with a common center point and base line (note the difference between these and circular bar charts: in one, the bar is straight and the base is curved, but vice versa for the other).



An example of a circular line graph. (Image courtesy http://www.cemframework .com/img/PolarPlot1.png.)

3.Region-Based Techniques

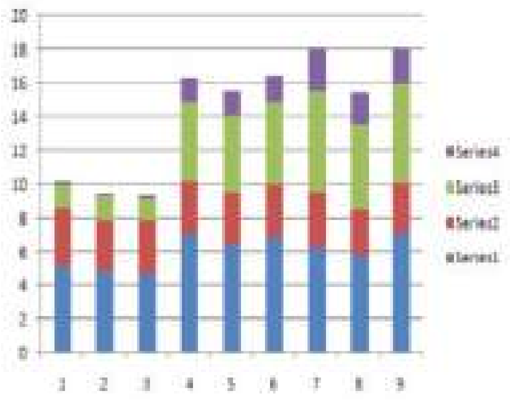
In region-based techniques, filled polygons are used to convey values, based on their size, shape, color, or other attributes. Even though it was mentioned in Chapter 3 that our ability to accurately measure area is noticeably worse than our ability to measure attributes such as length, many effective techniques in this category have been developed. For some, the goal is not to show the raw data itself, but rather summaries or distributions of the values. Many of the region-based techniques were initially designed for univariate data, such as pie charts and bar charts. Some, however, have been extended to multiple dimensions

A.Bar Charts/Histograms

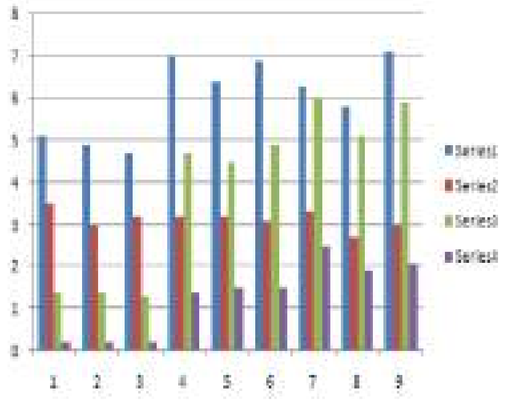
One of the most common visualizations, in addition to line plots, scatterplots, and maps, is the bar chart, where rectangular bars are used to convey numeric values. Humans have high visual acuity when it comes to comparing the length of linear features. Thus, bar charts are a natural choice for visualizing many kinds of data. Both horizontal and vertical bars are routinely used and are readily interpretable, which means that the visualization designer has some flexibility in the way they are integrated into a given application. If each bar is to receive a text label, it is easier to use horizontally oriented bars, but angling the text for vertical bars can also help to solve the problem of lengthy strings

One of the critical decisions that needs to be made when using bar charts is deciding how many bars are needed to best represent the data. If the bars represent the state of N variables, then as long as N is not too large, there can be a one-to-one correspondence between variables and bars. If the goal is to represent a summarization or distribution of a data set, we can use a histogram to convey the number of occurrences of data values. If the data takes on nominal values or a modest number of distinct integers, the decision is simple—just have the same number of bars as there are different values. For continuous data or integer variables with a large range, we need to divide the data into subranges and assign each subrange to a bar.

If the data is multivariate, there are several options for using bar charts. A common alternative is a stacked bar graph, where each bar consists of several shorter bars to represent the values for each dimension (Figure). It is common to vary the color, texture, or other attributes of the bars to make them readily distinguishable within a given bar and comparable between bars. Similarly, bars for different variables can be placed next to each other, giving them a common baseline and thus simplifying their interpretation (Figure). The choice between these approaches is often based on the number of variables, as well as the number of bars. Stacked bars distribute the burden in two directions, while adjacent bars can demand significant space, unless the bar and dimension counts are modes

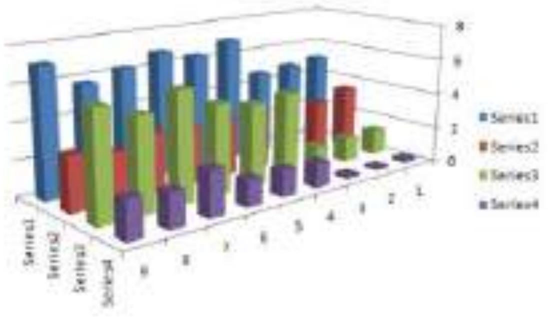


(a) Stacked bar chart.

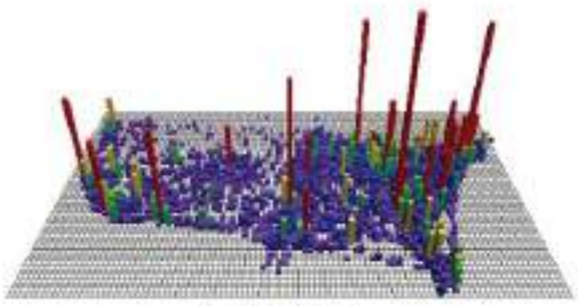


(b) Clustered bar chart.

Figure 8.13. Examples of 2D bar graphs for showing multivariate data.



(a) Bar graphs.



(b) Cityscape.

Figure 8.14. Examples of 3D visualizations for showing multivariate data.

B. Tabular Displays:

Multivariate data is often stored in tables, and a number of visualization techniques have been modeled on this structure. These techniques mostly vary in the types of interactions they support.

Heatmaps are created by displaying the table of record values using color rather than text. For this visualization technique, all data values are mapped to the same normalized color space, and each is rendered as a colored square or rectangle. Using different color maps, as well as allowing users to stretch or compress colors to emphasize or deemphasize some value ranges (as we saw in volume rendering using transfer functions), enhances the usefulness of this technique (Figure)

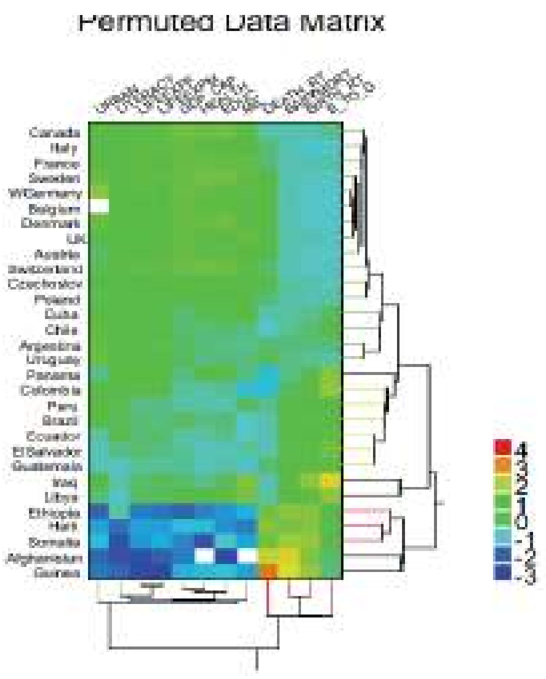
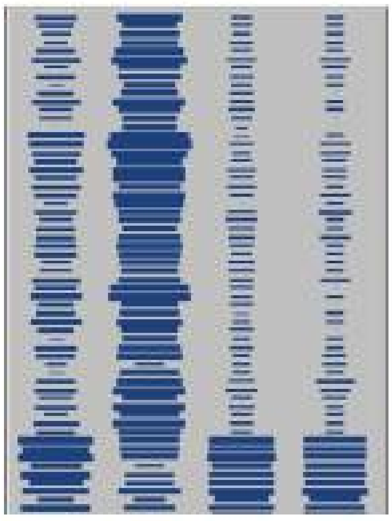


Figure 8.15. A heatmap showing social statistics for several countries from a U.N. survey. Rows and columns have been reordered via clustering. (Image courtesy Leland Wilkinson

Survey plots create a variant on permutation matrices by varying the size of cells, rather than coloring them and aligning cell centers within individual attributes [276]. This alleviates biases in color perception caused by the effects of adjacent colors. However, because measurement of area is more error-prone than measuring length, this method also has its deficiencies (Figure 8.16).



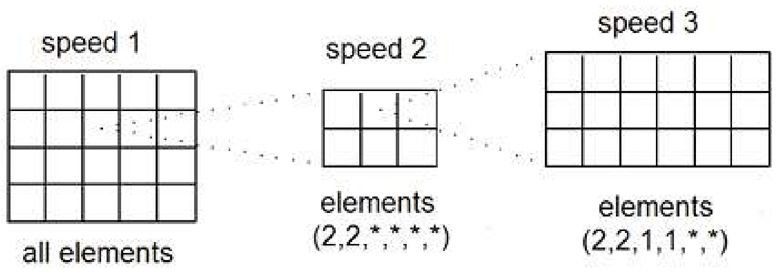
A section of a survey plot as computed by the DataLab tool. Each column is a visual representation of each of the four dimensions of the Iris data set.

C) Dimensional Stacking

Dimensional stacking is a method developed by LeBlanc et al. for map ping data from a discrete N-dimensional space to a two-dimensional image in a manner that minimizes the occlusion of data, while preserving much of the spatial information. Briefly, the mapping is performed as follows: begin with data of dimension 2N + 1 (for an even number of dimensions there would be an additional implicit dimension of

cardinality one). Select a finite cardinality/discretization for each dimension. Choose one of the dimensions to be the dependent variable.
Create ordered pairs of the independent dimensions (N pairs) and assign to each pair a unique value (speed) from 1 to N. The pair corresponding to speed 1 will create a virtual image whose size coincides with the cardinality of the dimensions (the first dimension in the pair is oriented horizontally, the second vertically). At each position of this virtual image, create another virtual image to correspond to dimensions of speed 2, again whose size is dependent on the cardinality of the dimensions involved. Repeat this process until all dimensions have been embedded. In this manner, every location in the discrete high-dimensional space has a unique location in the two dimensional image resulting from the mapping. The concept of the speed of a dimension can best be likened to the digits on an odometer, where digits cycle through their values at different rates.

Thevalue ofthe dependent variableat the location in the high-dimensional space is then mapped to a color/intensity value at that location in the two dimensional image. This embedding process is illustrated in Figure 8.18 with asix-dimensionaldataset, wheredimensions d1,...,d6 havecardinalities 4, 5, 2, 3, 3, and 6, respectively. For clarity, we have not displayed the values associated with a dependent variable, which would be the seventh dimension and would dictate the colors in the smallest grid locations. Figure 8.19 is an example of a dimensional stacking visualization



Conceptualization of dimensional stacking; collapsing six dimensions into two dimensions.

4.Combinations of Techniques:

In addition to the techniques based on points, lines, or regions, there are a number of hybrid techniques that combine features of two or more of these classes. We describe two of the more popular techniques of this type: glyphs and dense pixel displays

A) Glyphs and Icons

In the context of data and information visualization, a glyph1 is a visual representation of a piece of data or information where a graphical entity and its attributes are controlled by one or more data attributes. As an example, the width and height of a box could be controlled by a student’s score on the midterm and final exam for a course, while the color could be associated with the gender of the student. This is a rather broad definition for the term, as it can cover such visual elements as the markers in a scatterplot, the bars of a histogram, or even an entire line plot. However, a narrower definition would not be sufficient to capture the wide range of data visualization techniques that have been developed over the centuries and are termed glyphs.

Many authors have developed lists of graphical attributes to which data values can be mapped. These include: position (1, 2, or 3D), size (length, area, or volume), shape, orientation, material (hue, saturation, intensity, texture, or opacity), line style (width, dashes, or tapers), and dynamics (speed of motion, direction of motion, rate of flashing).
In this section, a wide range of possible mappings for data glyphs are discussed, including:

• one-to-one mappings, where each data attribute maps to a distinct and different graphical attribute;

• one-to-many mappings, where redundant mappings are used to im prove the accuracy and ease with which a user can interpret data val ues; and

• many-to-one mappings, where several or all data attributes map to a common type of graphical attribute, separated in space, orientation, or other transformation. One-to-one mappings are often designed to take advantage of the user's domain knowledge, using intuitive pairings of data to graphical attributes to ease the learning process. Examples include mapping color to temperature, and flow direction to line orientation. Redundant mappings can be useful in situations where the number of data dimensions is low and the desire is to reduce the possibility of misinterpretation. For example, one might map population to both size and color to ease analysis for color-impaired users, and to facilitate comparison of two populations with similar values. Many to-one mappings are best used in situations where it is important to not only compare values of the same dimension for separate records, but also to compare different dimensions for the same record. For example, mapping each dimension to the height of a vertical bar facilitates both intra-record and inter-record comparison.

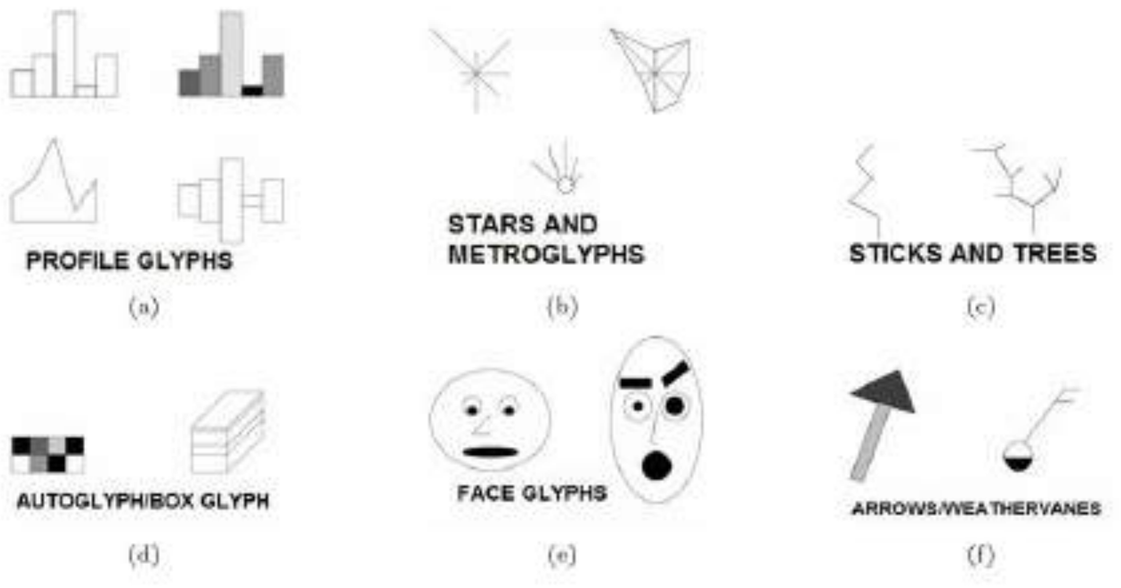


Figure 8.20. Examples of multivariate glyphs (from [445]).

In using glyphs for information visualization, we need to be aware of the many biases and limitations of the technique. First and foremost are the perceptual biases, depending on what graphical attributes are being used. As discussed in Chapter 3, there are some attributes, such as the length of a line, that we can judge more accurately than others, such as orientation or color. Other issues of bias include the fact that relationships between adjacent graphical attributes are much easier to perceive than those that are more distant, with a few notable exceptions (e.g., comparing two ears on a Chernoff face might be easier than comparing two different, but adjacent, facial features). Similarly, comparing two glyphs that are near each other on the screen is easier than if the glyphs are more separated. Finally, the number of data dimensions and records that can be effectively handled with glyphs is limited.

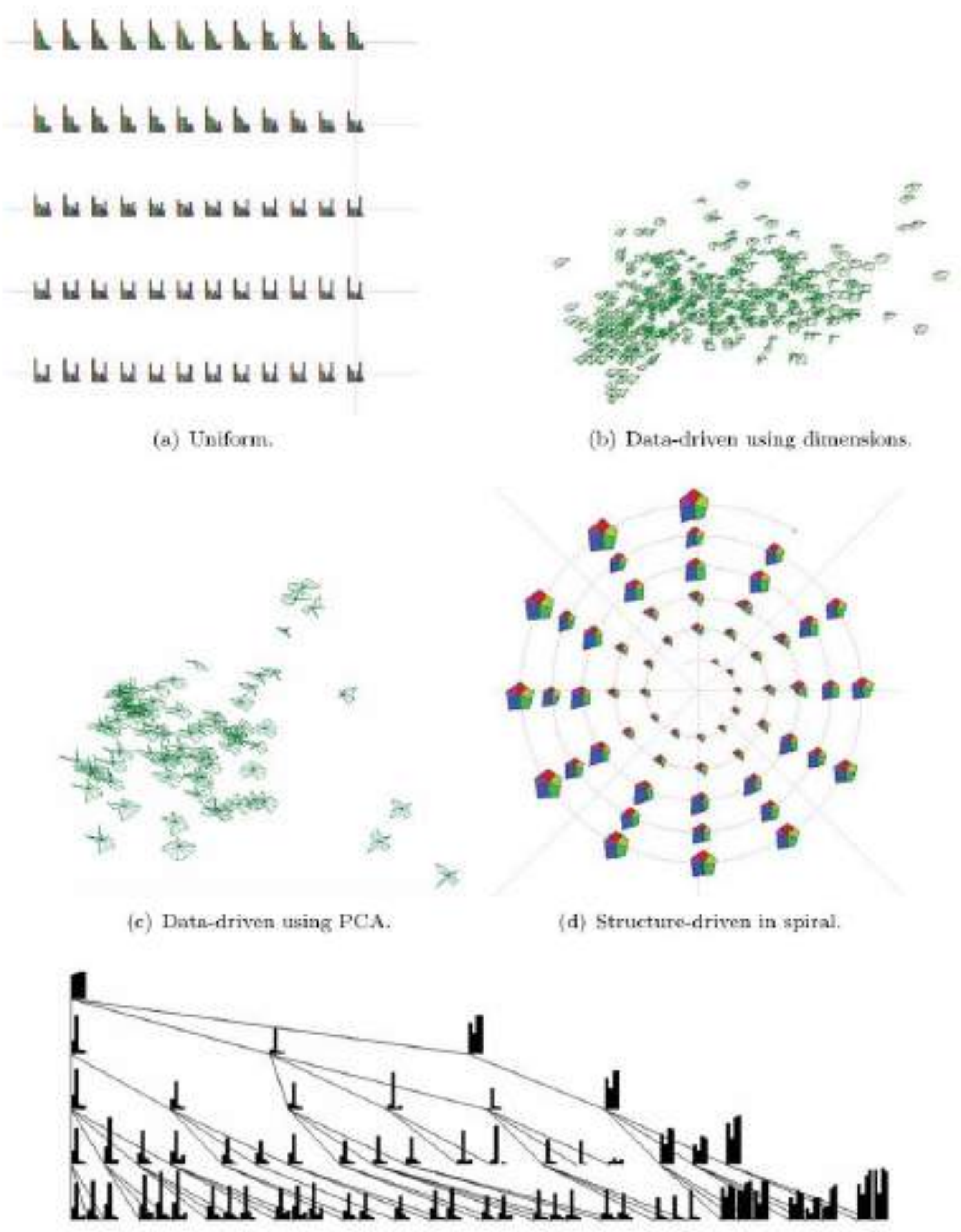
Once a glyph design is chosen, there are N! different dimension orderings that can be used in the mapping. Which ones are likely to reveal the most interesting features? Several ordering strategies can be imagined:

Dimensions could be ordered according to their correlation, so that similar dimensions are mapped adjacent to each other. This can help reveal general trends, as well as expose some outliers. • Dimensions can be mapped in such a way as to promote symmetrically shaped glyphs, which can be easier to perceive and remember. Shapes that are less symmetric than their neighbors will also stand out. • Dimensions can be sorted according to their values in one record. For example, if the data represents a multivariate time series, sort ing the dimensions based on the first record can make the trends over time morepronounced, conveyingwhich dimensional relationships were maintained versus those that changed significantly. • Dimensions can be manually sorted, based on the user's knowledge of the domain. Thus, semantically similar dimensions can be grouped or used for symmetric glyph features, which can simplify the interpreta tion

A final important consideration in designing a glyph-based visualization is the layout of the glyphs on the screen. As described in [445], there are three general classes of layout strategies:

1. uniform—glyphs are scaled and positioned with equal space between them to occupy the entire screen. This strategy eliminates overlaps, while making efficient use of the screen space. Different orderings of records can expose different data features in the same way that different dimension orderings can (Figure)

2. data-driven—data values are used to control the positioning of the glyphs. Two approaches are possible. In the first, two data dimen sions (or three for 3D display) are chosen to set the locations (Fig ure 8.21(b)), while in the second, positions are derived from the data values using algorithms such as PCA (Figure 8.21(c)) or MDS. In ei ther case, we can apply one or more additional passes over the data to reduce overlaps by separating glyphs that are too close.
3. structure-driven—if the data has an implicit or explicit structure to it, such as cyclic or hierarchical, this can be used to control the position ing. For example, glyphs may be laid out in a spiral or a grid to empha size cyclic patterns (Figure 8.21(d)); likewise, any of the multitudes of tree-drawing algorithms (see Chapter 9) can be used to position glyphs to help convey hierarchical relations (Figure 8.21(e)).

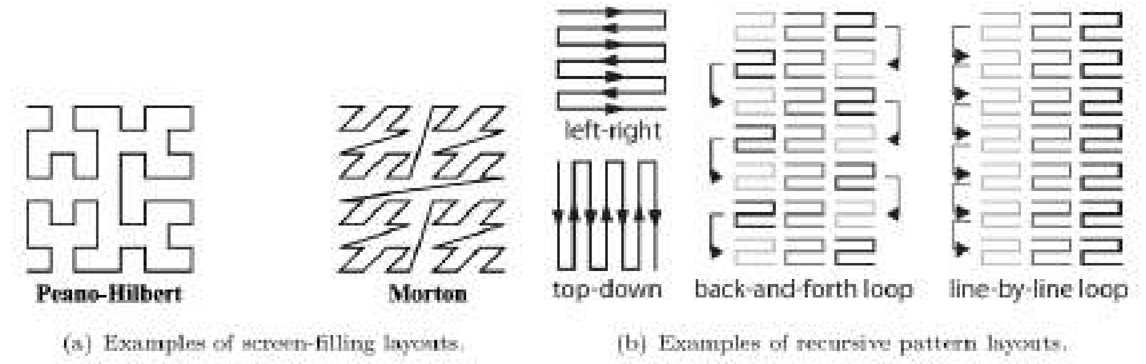

(a) Uniform. (b) Data-driven using dimensions. (c) Data-driven using PCA. (d) Structure-driven in spiral. (e) Structure-driven using algorithms.

3.21. Examples of glyph positioning strategies (from [445]).

B) Dense Pixel Displays

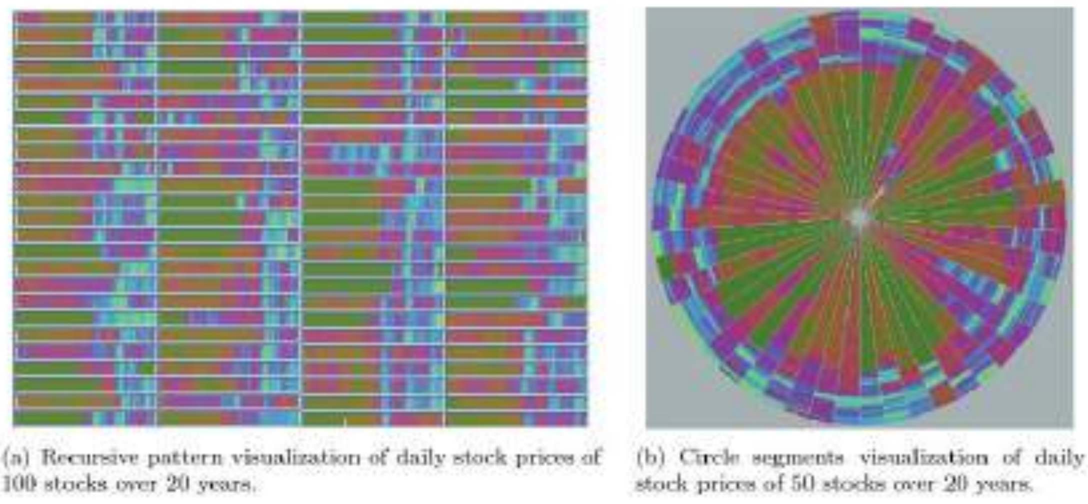
Dense pixel displays (also known as pixel-oriented techniques) are a hybrid between point-based and region-based methods. Pioneered by Keim and his colleagues , these techniques map each value to individual pixels and create a filled polygon to represent each data dimension. The displays make maximal use of the screen space, allowing data sets with millions of values to be shown on a single screen. Each data value will control the color for a single pixel; changing the color map used can potentially reveal new features of the data. Given a data set and a color map, the issues that remain to be resolved are the layout of the data records and their ordering.

In its simplest form, each dimension of a data set will generate a separate subimage within the display. Thus, we can treat each dimension as an inde pendent list of numbers, each of which drives the color of the corresponding pixels. We then need to lay out the elements of this list in a manner that accentuates relationships between points that are close to each other in the list. For example, we might create a subimage where we alternate a left-to right and right-to-left traversal, shifting down one row when we reach the edge of the subimage. Different shapes of subimages can potentially convey different features of the data. Another approach is to use a spiral layout, where the first data point is centered in the subimage and successive points are laid out in concentric squares. Yet another method is to use one of the many space-filling recursive curves as a layout strategy , such as a Peano-Hilbert curve (see Figure (a)). These curves have the feature that points that are close to each other in the ordered list are near to each other on the screen. Other pixel arrangements are the recursive z-pattern (see Figure (a)) or the recursive generalization of a line-by-line arrange ment, called the recursive pattern techniques (see Figure 8.22(b)). A result of the recursive pattern technique arranging the subimages for the different features in a rectangular grid is shown in Fig8,23 (a).



(a) Examples of screen-filling layouts. (b) Examples of recursive pattern layouts.

Figure 8.22. Examples of pixel layout patterns in dense pixel displays. (Right images from [223]. © 1995 IEEE.)



(a) Recursive pattern visualization of daily stock prices of 100 stocks over 20 years. (b) Circle segments visualization of daily stock prices of 50 stocks over 20 years.

Fig 8.23

Amajor issue with pixel-oriented displays is the data ordering. For some data, such as time series, the order is predetermined and fixed. In other cases, however, reordering the records can expose many interesting patterns. For example, if the records are ordered based on one of the dimensions, clusters of values within that dimension will be revealed, as will other dimensions having similar clusters.

Another approach is to order the records based on their N-dimensional distance from a selected point. This can expose clusters involving many dimensions at once, rather than one at a time, and by coloring the pixels based on their distance from the selected point, the user gets some insights into the number of clusters in the data, as well as the gaps between them.