Chapter 4
Multivariate Analysis

We have already introduced the concept of multivariate data, which are collections of data in which many attributes (usually more than four) change with respect to one or more independent attributes. We find numerous examples in the activities that we carry out every day. Take, for instance, the act of entering a shop to buy a cellular phone. We are normally greeted by an overwhelming number of models, all offering the most diverse features. Frequency bands, Bluetooth, camera, GPRS, MMS, WAP, speakerphone, and even Wi-Fi are just some of the numerous modern features that often influence our choice of one cell phone over another. To decide which cell phone to buy, we take the features that interest us the most into consideration and, from the various models, we choose the phone that most satisfies our requirements. Since there are many models, with features that evolve every six months in any case, very often we turn to the sales assistants to help us in the difficult task of choosing the cell phone that's right for us.

This is an example in which a decision process (the purchase of goods), based on a collection of multidimensional data (the technical features of the phones), is aided by an external agent (the sales assistant) with experience in and knowledge of the goods that we wish to purchase. This process works excellently when the number of instances (cell phones) and attributes (features) is fairly contained. What would happen if the number of cell phones were extremely large, let's say a hundred or even a thousand? Would the salesperson manage to cope with all of this data and advise us on the right phone without the aid of supplementary tools?

It's not by mere chance that one of the sectors in which information visualization is enjoying great success are the decision support systems, which are a specific class of software systems that help in decision-making activities. These systems are characterized by a great amount of data and numerous attributes and are very important as they provide essential information for managers, analysts, and directors, who are required to make decisions that are crucial for the running of the company.

In this chapter, we will analyze some of the most common techniques of the visual representation of multivariate data. We will also show some successful cases and examples of representations for the explorative analysis of large quantities of multivariate data.

R. Mazza, *Introduction to Information Visualization*,
DOI: 10.1007/978-1-84800-219-7_4, © Springer-Verlag London Limited 2009

4.1 The Problem of Multivariate Visualization

In Chapter 2, we looked at some examples of bivariate and trivariate data, represented simply by a scatterplot on Cartesian axes. Scatterplots are very simple and intuitive visual forms and work well when there are two dependent attributes. Their values are mapped along the values of the Cartesian axes. However, the number of situations in which one or two dependent attributes are involved is very limited, and most real problems have a rather high number of dependent attributes to analyze.

Scatterplots can be extended, allowing the mapping of more than two attributes. We can, for example, extend the scatterplots by adding further visual elements on which data mapping can be carried out (like shape, dimension, color, and texture), or adding a third dimension, represented by the perspective of the picture.

Working with these types of extensions, scatterplots can visually represent data in which up to seven different attributes can vary. Figure 4.1 is an example of a scatterplot that shows data from 174 countries and aims to compare the level of wealth (represented by the gross national income of each inhabitant) and the state of health of the resident population (represented by the number of deaths in children under five years old for every 1,000 births). These values are respectively mapped onto the x-axis and y-axis. Furthermore, they seek to report the number of inhabitants (mapped to the dimension of the graphical elements) and the continent to which each nation belongs (mapped to the color of the graphical element). One immediately notices an almost linear correlation between the wealth and the state of health: The state of health improves with an increase in the wealth in the population. There are also isolated cases that go against the grain with respect to the general trend; the most evident is represented by Cuba, whose population has an excellent level of health (even higher than that of the United States!), despite the wealth as being comparable to that of India. Other interesting information is drawn from the color of the circles. For example, the African nations are almost all grouped in the lower left-hand corner of the graph, indicating the state of extreme poverty and the terrible health levels in these unfortunate nations.

The scatterplot in Fig. 4.1 effectively highlights the relationship among wealth, state of health, number of inhabitants, and the continent to which each nation belongs. It is worth remembering, however, that the scatterplots are not suitable for all types of problems and data. If we wish to add other factors, such as the number of working hours per year, the average rent of an 80-square-meter apartment in the center of the capital, the cost of 1Kg. of bread, the number of paid holidays per year, etc., the scatterplot is no longer appropriate and so it’s necessary to find appropriate forms of visualization. In the following sections, we will illustrate visualization techniques for multivariate data, grouping them as geometric, iconic, and pixel-based.

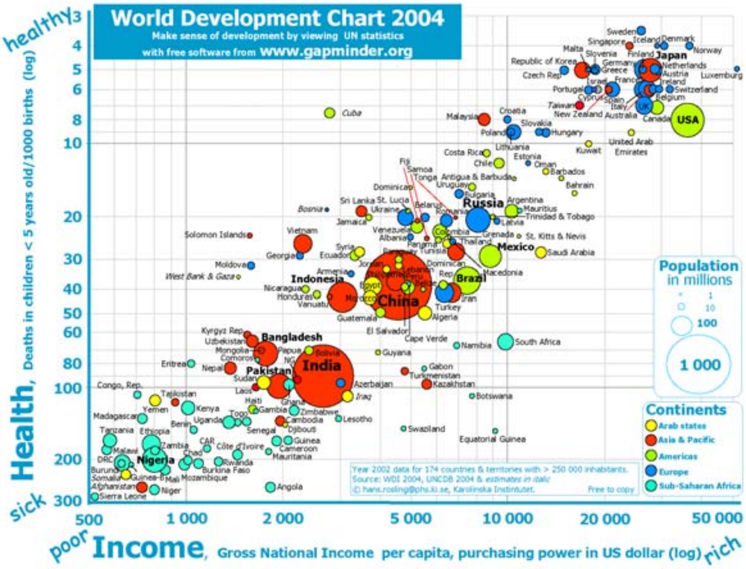


Fig. 4.1 Scatterplot representing four-variate attributes. Image published on the website http://www.gapminder.org on data derived from the United Nations statistical studies, with copy permission.

4.2 Geometric Techniques

Geometric techniques in information visualization consist of mapping the data of the attributes on a geometric space. Scatterplots belong to this category but are, unfortunately, limited by the fact that they have only two Cartesian axes on which to map two dimensions. In 1981, Alfred Inselberg, a researcher at IBM, had the brilliant idea of defining a geometric space through an arbitrary number of axes, arranging them parallely, instead of perpendicularly, as had been done in the Cartesian diagrams [28]. This was the origin of one of today's most common technique of visual representation of multivariate data.

4.2.1 Parallel Coordinates

This technique takes its name from the method with which the values of the attributes are represented: Every attribute corresponds to an axis and the axes are arranged to be parallel and equally spaced. Each record of the dataset is represented by a polygonal chain that connects the values of the attributes on its axes.

We'll look at an example to help us understand exactly how this type of representation works. Let's suppose that we have to represent the following data through parallel coordinates.

	Age	Weight	Sex
Vincenzo	32	75	M
Piero	24	63	M
Luisa	28	60	F
Giulia	18	58	F

Each record of the dataset has a different color to facilitate understanding of the generation process. It deals with trivariate data, two of which are numbers (age and weight) and one is categorical (sex). The name of the person is considered independent data. We represent these data by mapping values onto three parallel exes, using a quantitative spatial substrate for the first two attributes and nominal for the last (see Section 2.1.2). By joining the points corresponding to the coordinates of each record, we achieve the representation shown in Fig. 4.2.

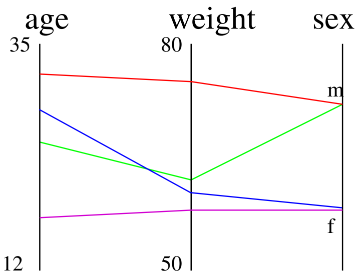


Fig. 4.2 Parallel coordinates representation.

We will now analyze a concrete case, using a dataset provided by *Xmdvtool* software, developed by Matthew Ward at Worcester Polytechnic in 1994, as an example [64]. The software is public domain and can be freely downloaded from the website.[1] The dataset provided contains the technical specifications of 392 car models produced in the 1970s, with seven dependent attributes. By analyzing the dataset using Xmdvtool software, we attain the visual representation illustrated in Fig. 4.3.

Parallel coordinates are a very powerful tool for the explorative analysis of data. For example, an inverse relationship between fuel consumption (MPG) and the number of cylinders in an automobile is easily discernable: The intersection of the lines that join the values between the two axes clearly demonstrates how the cars with a

[1] http://davis.wpi.edu/ xmdv/

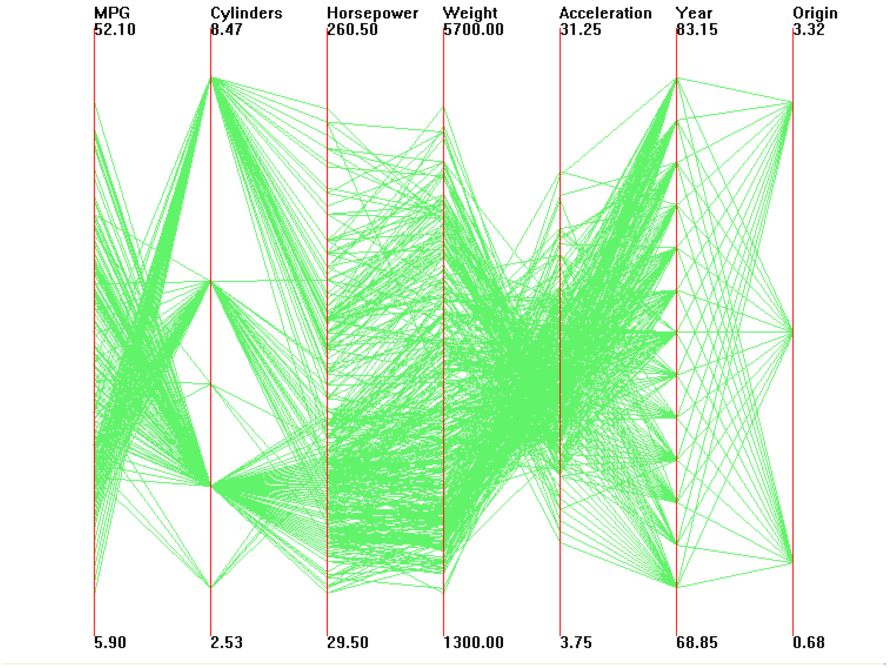


Fig. 4.3 Parallel coordinates for 730 elements with 7 variant attributes. Picture generated by Xmdvtool software.

high number of cylinders (in the upper part of the axes) cover fewer miles per gallon than those of fewer cylinders. Another clear inverse relationship is that between the weight of the automobile and acceleration: The heavier automobiles usually have a shorter acceleration (meaning the time necessary to reach a certain speed starting from standstill).

Although they are a very powerful tool of explorative analysis, parallel coordinates can present some problems with very large datasets (for instance, in datasets with 5,000 elements). In such cases, the visual representation could be too dense to distinguish the lines, reducing the representation to a single polygon of uniform color. This problem is intrinsic to any visual representation: The space available on the screen may be insufficient to contain all the visual elements. Also, the arrangement of the axes in the parallel coordinates is decisive for the analysis of the dependence between the various attributes. In particular, the dependence between the attributes represented by the immediately adjacent axes is obvious, while a direct analysis between attributes represented by distant axes might not be possible (for example, in Fig. 4.3, the correlation between the acceleration and MPG of the automobile is not immediately noticeable).

It is possible to intervene by means of interaction. First, the software enables the reordering of the axes. If we want to study the direct correlation between two attributes, we move the axes so that they are positioned one beside the other.

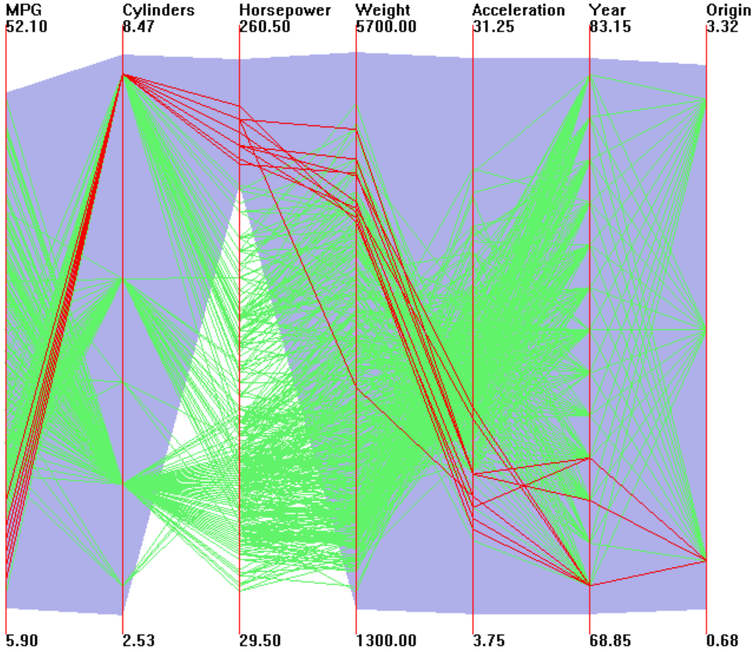


Fig. 4.4 Brushing of the values on the coordinate parallel axes. The red lines represent the elements that satisfy all of the limits on the values denoted by the violet polygon. Image generated by Xmdvtool software.

Another procedure that can be carried out on the parallel coordinates graph is the *brushing* of values of one or more attributes. In Fig. 4.4, the purple polygon selects, for each axis, the values on which the brushing is to be carried out. The red-colored lines mark the elements on which values fall on these intervals. By placing the cursor over an axis, one can read the corresponding value. Brushing can be very useful for carrying out explorative analysis of values. In Fig. 4.4, for example, we have brushed the automobiles that have less than 200 horsepower. Through brushing, we can deduce that all of the automobiles with this characteristic have 8 cylinders, were manufactured in the same country and, curiously, were all produced before 1973.

4.2.2 Scatterplot Matrix

Scatterplot matrices represent an interesting extension to the common 2D scatterplot, to simply and intuitively represent a generic number of multivariate attributes. This very simple technique consists of representing pairs of attributes, through bidimensional scatterplots, and putting the scatterplots side by side to share the same

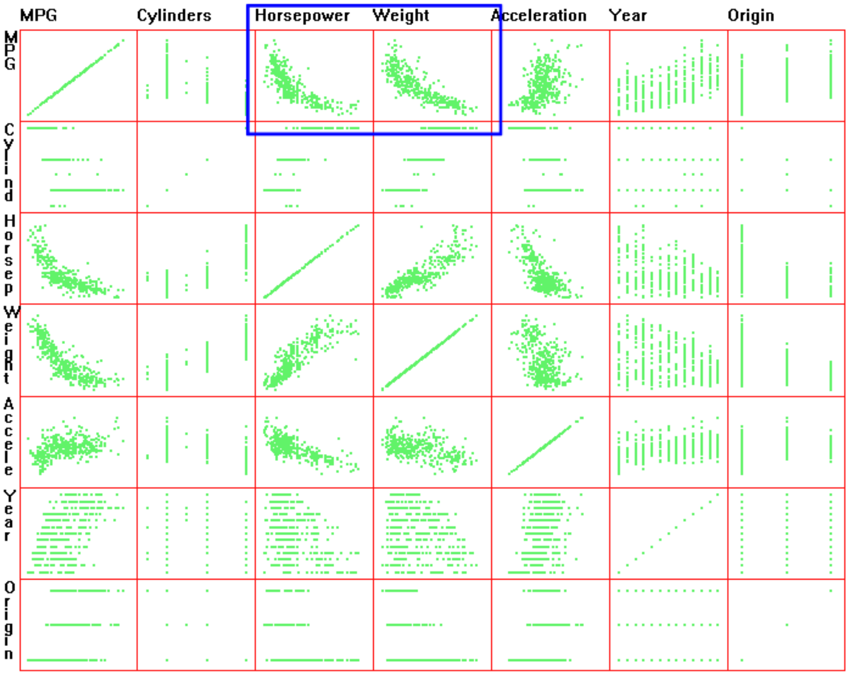


Fig. 4.5 Scatterplot matrices present multivariate data very simply and intuitively. Image generated by Xmdvtool software.

axes. In this way, an $N \times N$ matrix is formed, where N is the number of attributes to represent. Figure 4.5 illustrates an example, generated through Xmdvtool software, using the same dataset that was previously used for the parallel coordinates.

From the figure, the type of correlation that runs between the pairs of attributes can immediately be drawn. For example, it is very clear that, by increasing the horsepower and weight, the number of miles covered (in terms of miles per gallon, MPG) diminishes dramatically (see graphs included in the blue frame). We would also have been able to reach the same result through parallel coordinates, but, to analyze the various dependence between the pairs of attributes, we would have had to move the axes or perform brushing. Using the scatterplot matrices, however, makes the correlation between the pairs immediately visible, without needing to adjust the visual representation. On the other hand, the scatterplot matrices can present some inconveniences. In particular, for N attributes, it is necessary to form an $N \times N$ scatterplot matrix; therefore, the space available to represent the points is very limited. It is impossible to put labels to indicate the individual points, or select a particular point with the mouse to read the value of its coordinate. Moreover, the collective vision of all the attributes that we have observed in the scatterplot in Fig. 4.1 is lost when the scatterplot matrix is used.

4.2.3 TableLens

Spreadsheet applications, like Microsoft Excel, have become one of the most widespread types of software. Thanks to a very intuitive visual interface (but also thanks to some very effective marketing initiatives), this type of tool is part of the software equipment of every computer in use in a professional or domestic environment. The great intuition that the creators of this software had in the early 1980s was to use a structure similar to multiplication tables to perform the calculations, a very simple data organization that we have been used to since our early school years.[2] Following this principle, in 1994 John Lamping and Ramana Rao proposed a visual analysis tool for data called *TableLens* [48]. Its structure was inspired by spreadsheet applications, but its characteristic is to represent data using horizontal bars rather than numeric values. In particular, data are represented on a matrix, where attributes are represented on columns and every instance of data is reported on a row of the matrix. The numerical values of an attribute of the dataset are mapped to the length of horizontal bars. Visually, the horizontal bars can be represented in a very limited space. This way it manages to represent a large quantity of attributes and instances in a single screen and allows the user to immediately identify possible patterns, trends, and relationships among the attributes. See an example in Fig. 4.6.

An interesting property of this type of representation is the possibility for the user to interact with the visualization to

- change the order of the columns,
- hide or show columns,
- sort the data by the values of a column,
- show the values of some instances without losing the context of the entire visualization.

An example is shown in Fig. 4.7, where the data are sorted by the values of the first column on the left, by simply clicking on the heading of the column of the values to be sorted, and the data of some instances are made visible by clicking on the rows of the values to be visualized.

4.2.4 Parallel Sets

The geometric techniques discussed so far can be used with any type of attribute (quantitative, ordinal, or categorical). As we have already seen in Section 2.1.2, the

[2] It is fitting to recall that the creator of this type of application was Dan Bricklin, who invented VisiCalc, the first spreadsheet application for the Apple II personal computer.

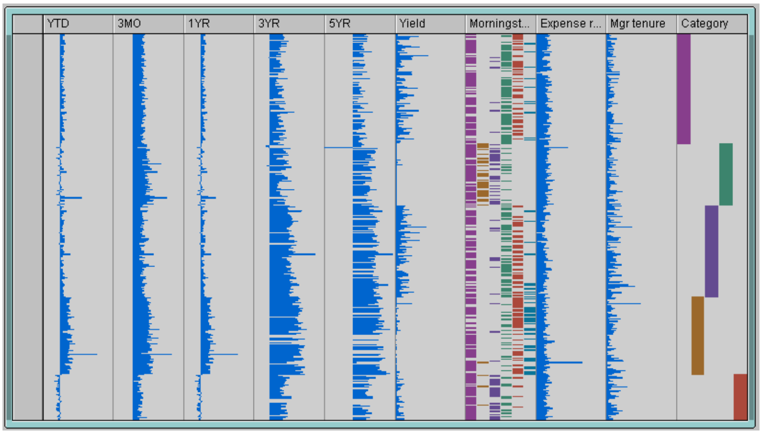
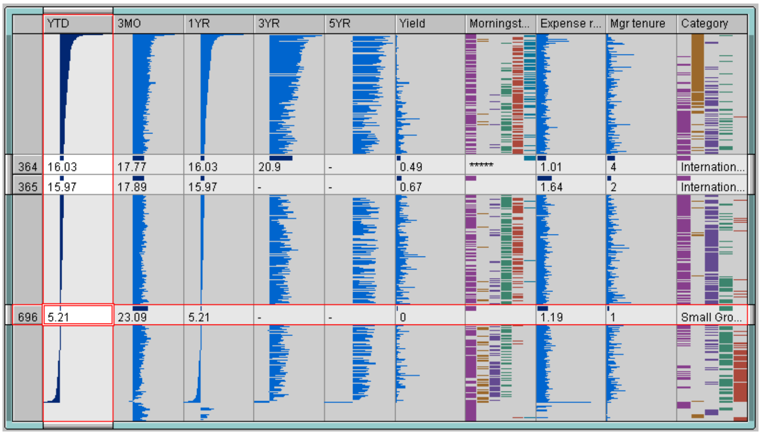


Fig. 4.6 TableLens representation. Image generated by ©Business Objects TableLens software.



	YTD	3MO	1YR	3YR	5YR	Yield	Morningst...	Expense r...	Mgr tenure	Category
364	16.03	17.77	16.03	20.9	-	0.49	*****	1.01	4	Internation...
365	15.97	17.89	15.97	-	-	0.67		1.64	2	Internation...
696	5.21	23.09	5.21	-	-	0		1.19	1	Small Gro...

Fig. 4.7 TableLens sorted by the values of the first column on the left. Image generated by ©Business Objects TableLens software.

quantitative type attributes find a natural mapping onto the axes (in the case of parallel coordinates and scatterplots) or into the length of horizontal bars (in the case of TableLens), in that the values are reported on the axis in an order that corresponds to the order of the data; in the case of ordinal or categorical data, it is necessary to find a mapping that divides the available spaces into a discrete collection of subregions. For example, in Fig. 4.3, the production countries of the automobiles are positioned on the last axis to the right of the parallel coordinates, dividing the space uniformly and mapping the production countries on this axis.

Systems specifically created for the representation of categorical data are rarely proposed. Among these, we can mention *parallel sets*, a technique developed by a group of researchers from the VRVis Institute of Vienna [36]. Parallel sets take inspiration from the parallel coordinates but, in contrast, the *frequency* of the values

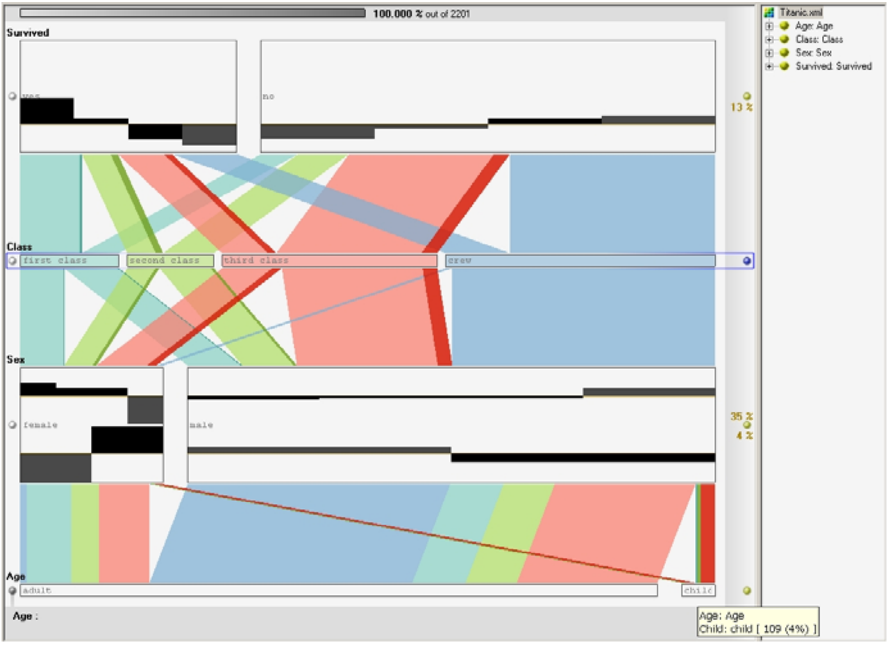


Fig. 4.8 Representation of the datasets of the Titanic disaster using parallel sets. Image reproduced with the permission of Helwig Hauser, VRVis Zentrum für Virtual Reality und Visualisierung Forschungs-GmbH, Vienna.

in the dataset substitutes for the representation of each single instance. This type of representation, unlike the parallel coordinates, turns out to be more appropriate in the case of categorical data and, due to the aggregation of frequency data, it manages to process very large datasets without the problem of space we have seen with parallel coordinates.

Let's see how it works with an example. Fig. 4.8 shows a representation in parallel sets of a dataset derived from the victims of the Titanic disaster, following a collision with an iceberg on the night of April 14, 1912. The dataset has four dependent attributes: the age of the passengers, the sex, the class in which they traveled, and whether they survived the disaster. The layout is reminiscent of the parallel coordinates, but, in this case, the axes have been replaced by a number of rectangular boxes that represent the categories. The width of these rectangles corresponds to the frequency of the corresponding category (for example, in the zone that represents the sex, there were 470 female passengers and 1,731 male passengers; the size of the rectangle reflects this proportion very clearly). Also, quantitative attributes can be mapped onto the axes, as we can see in the case of age. The attributes arranged next to each other are linked by connections that, in this case, represent the values of the frequencies in which the conditions are verified. For example, in the figure, the attribute at the top, survivors, is subdivided into two areas: yes, no. The survivors of the disaster are positioned in relationship to the passenger class to which they belong. For each class item two connections link the areas “yes” and “no” of the survivors, showing the percentage of survivors in each class. The figure very clearly shows that most of the first-class passengers survived, while the majority of the third-class passengers and the crew perished in the disaster. Besides the frequency for every single category, the rectangular block can also contain a histogram that shows, through appropriate statistical calculation, the degree of dependence of each value in the category with the values in the other category with which it is put in relation. The greater the dependence, the larger the histogram.

4.3 Icon Techniques

Another family of techniques of representation of multivariate data that uses the geometric properties of a figure is called *icon techniques*. The name comes from the fact that a geometric figure (an icon, which in this case is also called *glyph*) can have a number of features that may vary: color, shape, size, orientation, etc. The basic idea consists of associating each attribute with a feature in the geometric figure and mapping the data to the extensive properties of each feature. We will examine two well-known techniques: *star plots* and *Chernoff faces*.

4.3.1 Star Plots

A simple and relatively intuitive geometric figure is represented by a star-shaped polygon, whose vertices are defined by a collection of axes that all have the same origin (see Fig. 4.9). Every instance of the dataset can be represented by a “star” in which the values of each attribute of the instance are mapped to the length of each vertex. By joining the points that correspond to each vertex, a geometric figure is obtained, whose shape globally describes the instance of the dataset.

This technique, called a *star plot* (or also a *star glyph*), can be useful for comparing different instances of a dataset, by simply comparing the polygonal shapes derived from each glyph. In Fig. 4.10, we can, for example, compare the statistical data of some climatic values represented by a star plot. The attributes represented are the average annual precipitation, the average annual temperatures, the average maximum annual temperatures, the average minimum annual temperatures, the record for the maximum temperature and the record for the minimum temperature. These data come from the `weatherbase.com` website, which compiles multiyear statistics (reported in tabular version in Table 4.1). The various attributes are mapped to the length of each vertex of the star starting from the right and proceeding counterclockwise.

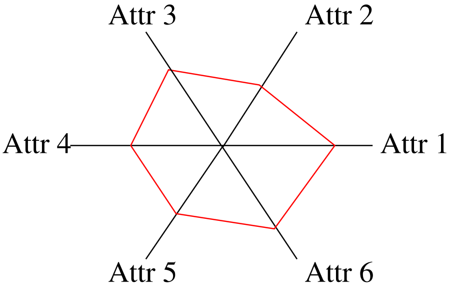


Fig. 4.9 Star plot.

City	Precip. average	Temp. average	Temp. max average	Temp. min average	Record max	Record min
Athens	37	17	21	13	42	−3
Bucharest	58	11	16	5	49	−23
Canberra	62	12	19	6	42	−10
Dublin	74	10	12	6	28	−7
Helsinki	63	5	8	1	31	−36
Hong Kong	218	23	25	21	37	2
London	75	10	13	5	35	−13
Madrid	45	13	20	7	40	−10
Mexico City	63	17	23	11	32	−3
Moscow	59	4	8	1	35	−42
New York	118	12	17	8	40	−18
Porto	126	14	18	10	34	−2
Rio de Janeiro	109	25	30	20	43	7
Rome	80	15	20	11	37	−7
Tunis	44	18	23	13	46	−1
Zurich	107	9	12	6	35	−20

Table 4.1 Annual climatic values in Celsius of some world cities. Values from http://www.weatherbase.com.

From the star plot representation in Fig. 4.10, we notice that Moscow and Helsinki have similar climatic characteristics, as do Athens and Tunis. By simply comparing the shapes generated by the star plot, we are able to visually distinguish grouping of elements of different datasets.

An obvious limit to this type of representation lies in the scalability: For a number of elements that is not exceedingly high, the space occupied on the screen immediately becomes so dense that it is difficult to clearly make out the various iconic forms. Furthermore, the icon techniques are applicable when the qualitative, and not the quantitative, aspect of the various attributes of the dataset is to be explored.

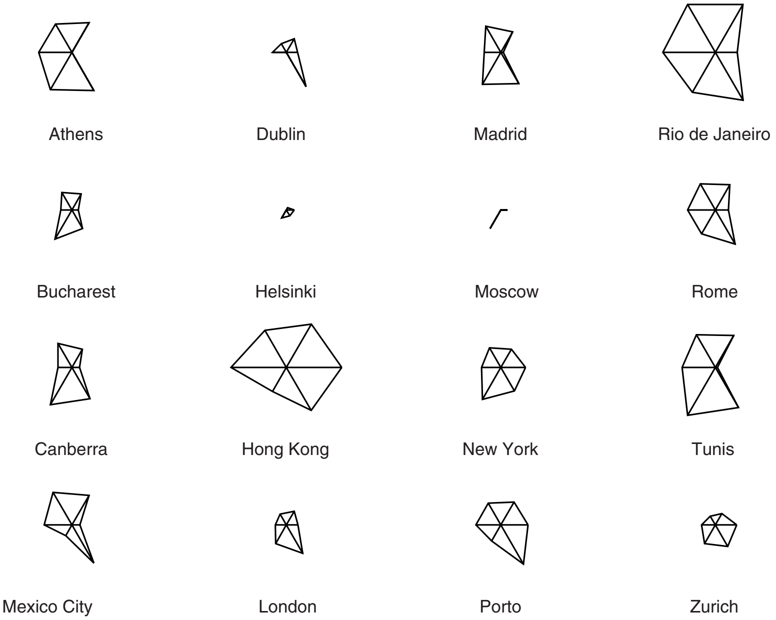


Fig. 4.10 Star plot of the annual climatic data of some cities. Image generated by the S-PLUS tool.

4.3.2 Chernoff Faces

Herman Chernoff put forward an interesting icon technique in which the element of representation is not an inexpressive and dull polygon, but a “face” [10]. Since humans are particularly capable of, and used to, recognizing even the slightest alteration in human facial expressions, Chernoff proposed mapping the attributes of a collection of multivariate data to the form, dimensions, and orientation of human facial features, like the eyes, nose, mouth, ears, etc. Fig. 4.12 gives an example of representation, through Chernoff faces, of the climatic data of the cities we have observed. The mapping is displayed in Fig. 4.11.

Like the star plot, with this representation we are able to deduce similarities between the cities of Helsinki and Moscow and Athens and Tunis. This type of mapping utilizes familiar facial features and is immediately perceptible to us, adding greater expressive power than the star plot. For example, having chosen to map the area of the face to the average precipitation of the city, we immediately see how Hong Kong is a very rainy city. The choice of mapping between attributes and elements of the face is critical and, if badly carried out, can lead to incorrect observations. For example, having chosen to map the width and curvature of the mouth to the record minimum and maximum temperature, respectively, we are incorrectly led

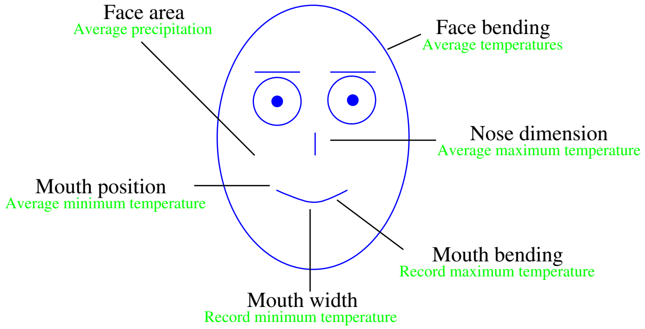


Fig. 4.11 Chernoff face.

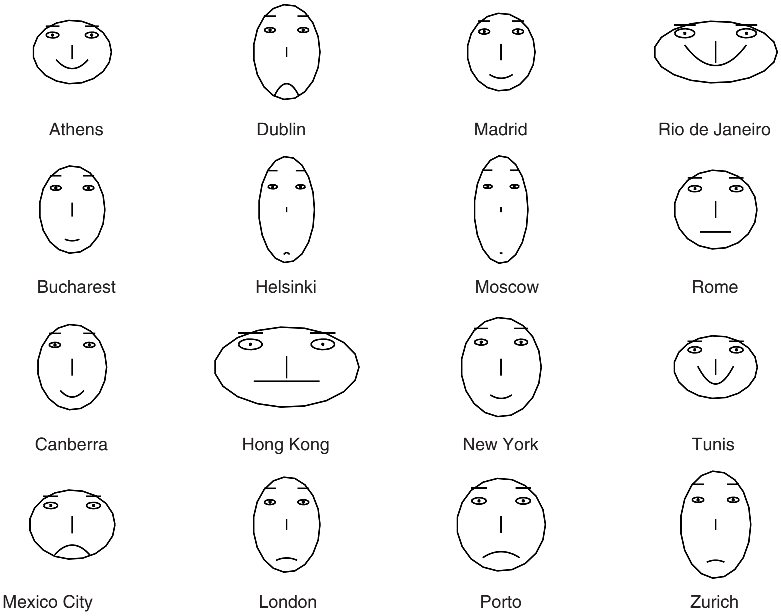


Fig. 4.12 Climatic data of some cities represented by Chernoff faces. Image generated by the S-PLUS statistics tool.

to understand that cities like Rio de Janeiro and Tunis have better climatic conditions than the others, just because the face has a smiling expression.

This type of representation is very interesting because of its way of presenting data, even though it has been criticized over the years by numerous visualization experts, in that the symmetry of parts like the eyes, eyebrows, and ears, present in pairs on the face, build an unnecessary duplication that is not actually present, for instance, in the star plot. Furthermore, studies have shown how the choice of mapping of an attribute can bring about differences of up to 25% in terms of cluster perception in the dataset. This means that to classify two faces as "similar" is widely influenced by the choice of mapping a specific attribute to one facial feature instead of another.

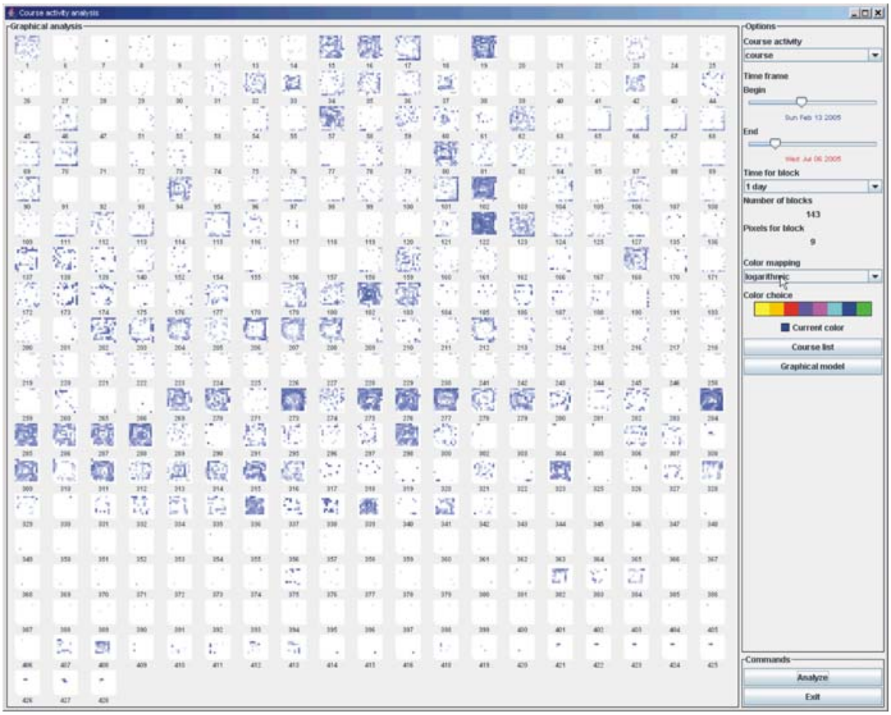
4.4 Pixel-Oriented Techniques

To maximize the number of elements to represent, some techniques use the pixels of the screen as basic units of representation. In effect, the pixel represents the smallest part, the "atomic" unit, beyond which it is impossible to subdivide the representation. A computer screen with a $1,024 \times 768$ pixels resolution could therefore potentially represent $786,432$ separate elements of a univariate dataset, mapping the data of an attribute to the color of the pixel of the corresponding element. This way we obtain a upper limit of visible elements in a single screen, beyond which it is theoretically impossible to go. In practice, this limit is never reached, first because part of the screen is dedicated to containing functional and aesthetic elements of the representation such as buttons, borders, text elements, and others; second, because there is a very limited number of situations in which it is sufficient to map a unit of information to the color of a single pixel of the screen.

Let's look at how to apply these techniques to a collection of multivariate data. The goal consists of representing the greatest amount of data in a single screen, mapping each value into the color of a pixel of the screen and grouping the data that belong to a certain attribute in a specific area, called a *window*. Daniel Keim [31] has studied the approach from a theoretical point of view and has defined a series of factors that need to be considered when applying this technique:

- *Shape of the window*. Usually represented in rectangular shapes and arranged in a matrix on the screen. Other shapes have been proposed, but the rectangle remains the most suitable to best take advantage of the screen's physical space.
- *Visual mapping*. What does each pixel represent?
- *Arrangement of the pixels*. How should the pixels be arranged in each window?
- *Color mapping*. How should the colors of the pixels be mapped?
- *Ordering of the windows*. In what order are the windows arranged in the screen's physical space?

As an example of such applications, Fig. 4.13 shows a tool for monitoring the use of online courses in an e-learning platform [41]. The e-learning platforms are, by now, a very common tool in universities. For management or statistical purposes, it can be extremely useful to comprehend the level of the students' use of each online course. The problem is that a platform can run hundreds (or even thousands) of courses, and, therefore, tracking the level of use of each course can be very taxing and complicated. The application shown in Fig. 4.13 serves precisely this purpose. This is a sort of dashboard of the use of resources in the e-learning platform. The main window is subdivided into a control area (right) and a matrix of little windows



Course activity: course

Time frame

Begin: Sun Feb 13 2005

End: Wed Jul 06 2005

Time for block: 1 day

Number of blocks: 143

Pixels for block: 9

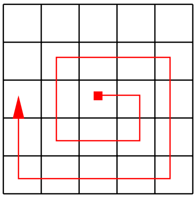
Color mapping: logarithmic

Color choice: 🟨🟧🟥🟪🟪🟦🟦🟩

🟦 Current color

Fig. 4.13 Example of an application that uses the pixel as a basic unit. Image published in [41] and reproduced with the permission of Springer.

(left). Each window represents an online course run by the platform, while the pixels represent a unit of time (1 minute, 1 hour, 1 day, etc.). The pixels are distributed according to a *temporal spiral*, with the first pixel (based on order of time) arranged in the center of the little windows, then extending to the edges in a clockwise direction.



The pixels are colored according to the amount of use of the course during the time corresponding to the pixel. A color scale is used with a single hue (blue), and intensity represents the use of the course in that unit of time: A light color means little course use (completely white corresponds to no activity) and a strong blue color, on the other hand, points to intense use. From Fig. 4.13, which represents as many as 345 courses in a single screen, it is clear which courses have had the most activity during the semester. By analyzing the positioning of the colored pixels in detail, it

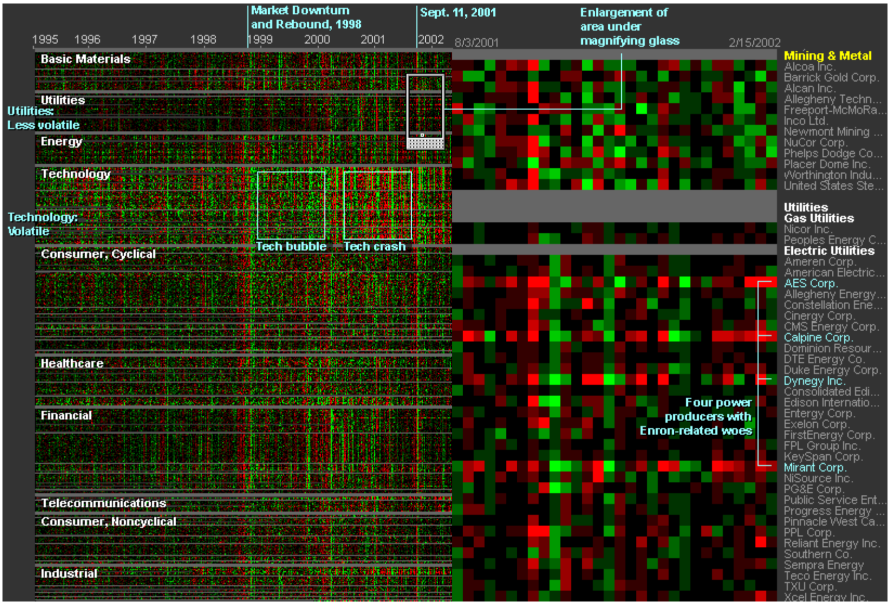


Fig. 4.14 Market Radar is an application managed by SmartMoney for analyzing the stock market share history. Image reproduced with the permission of ©smartmoney.com.

is possible to understand if the activities were concentrated at the start or the end of the semester, or if they are distributed throughout the semester. The advantages of this approach are notable, not only for the quantity of data that we manage to represent in a single screen, but also for the simplicity of the interpretation, which, from a cognitive point of view, does not call for particular effort.

Another interesting example is *Market Radar*, produced by SmartMoney,[3] a company that offers stock market share investors a wide number of analytical tools that use modern techniques for the visual representation of information. Market Radar analyzes the history of shares quoted on the U.S. stock exchange (Fig. 4.14). It visualizes the price variation of 500 shares over the last 8 years, making it possible to examine the entire market dynamic, over a substantially long period of time, at a glance. Market Radar takes the weekly price variations into account: Each week is represented by a column in the left part of the figure, while a different row of the matrix corresponds to each stock price.

Stocks are grouped in sectors, to study the progress of the various types of sectors or industries. Each weekly price change is represented by a dot in the matrix on the left: green if the stock went up in that week, red if it went down, while the intensity of the color reflects the variation with respect to the previous week. In the example reported in this figure, it is clearly noticeable how the shares belonging to the technology sector have the highest instability, characterized by abrupt highs

[3] http://www.smartmoney.com.

and lows in very short periods of time. It is also easy to make out the periods of strong growth in this sector around 1999 (with a prevalence of green pixels), and a successive period of serious crisis for all technology departments after the first half of 2000.

The user can put a sort of magnifying glass on a particular area in the matrix, enabling the observation of the magnified detail on the right, where the performance of every company can be clearly distinguished. The example shown in the figure displays an extremely negative week for most stocks, corresponding to the attack on the Twin Towers in New York on September 11, 2001.

4.5 Conclusion

In this chapter, we have described some of the most common techniques used to represent multivariate data, which characterize the vast majority of real-world problems. The challenge is to visually represent multidimensional data in a two-dimensional screen space. This limitation obliges us to resort to a trick in order to map several attributes into spatial substrates and graphical elements. We divided these techniques into three categories: geometric, iconic, and pixel-based (depending on the main approach adopted), showing some applications that have reached a good level of adoption in the market.