Viewing of 2-dimensional data

V. Pohánka
Geophysical Institute of the Slovak Academy of Sciences[1]

A b s t r a c t : Advantages and disadvantages of the viewing of 2-dimensional data in the form of a text and an image are discussed and the new way of viewing combining the advantageous properties of these two ways is proposed. This way was realized by introducing a suitable data file format and by writing the programs for viewing and manipulating with the data.

Key words: displaying of data, file format, viewing program

1. Introduction

In various scientific disciplines we meet the data sets containing values of some physical quantity at certain points of some planar surface. More exactly, the members of such data set are the triples (x, y, f), where x, y are the coordinates (in some coordinate system in the plane) and f is the value of the physical quantity at the point with coordinates x, y. This value can be obtained by the measurement or by the calculation from some other physical quantities. We shall treat here the question how to view such data sets in order to obtain from them the maximal amount of information in a human readable form. The problem is that the data set usually contains very large number of members and therefore it is not easy to obtain the whole available information contained in the data set.

2. Viewing of data files

For simplicity we restrict ourselves to the data sets for which the data points represent a regular 2-dimensional net. This need not be the case

[1] Dúbravská cesta 9, 845 28 Bratislava, Slovak Republic
e-mail: Vladimir.Pohanka@savba.sk

if the data are obtained by a measurement, but in this case it is usual to calculate a regular data net using some interpolation method.

The usual form of the data set is a binary file (which may contain the values of the triples (x, y, f) or only the values of f ordered in some way). The other possible forms of the same data set can be a text file (which can again have one of the mentioned two forms) and an image file which is a graphical representation of the data (for example, a file in the `bmp` format). In the latter case the image file can be constructed, for example, by representing the data points as pixels of the image and the data values as colors of these pixels: the RGB value for each pixel of the image is then calculated from the actual value of the quantity f at this point according to some formula. Of course, there are many other possibilities: for example, the image file can contain only the isolines connecting the points with the same data value.

The information contained in the text data file can be equivalent to that contained in the corresponding binary file (if the text representation has enough significant digits); on the other hand, the image data file contains (almost always) less information than the corresponding binary file.

Accordingly, there are in principle two ways to view the data contained in such 2-dimensional data set: we can display either the text file or the image file. By the first way it is possible to get information about the value f for any given data point (x, y); on the other hand, this way is very cumbersome as we can see at once only a very small part of the data; it is almost impossible to comprehend the whole data set. Further, as the data are in the text file linearly ordered, it is not possible to see the 2-dimensional local behaviour of the displayed quantity (it would be the case only if the data would be stored in the file in lines corresponding to the points with the same y-coordinate).

By the displaying of the image file we can easily grasp the course of the displayed quantity in the whole data domain. However, this way has a major disadvantage: we see the global and local behaviour of this quantity, but it is impossible to obtain from such an image the exact values of this quantity at the points we are interested in.

Therefore it would be comfortable to have such a way of viewing of the 2-dimensional data which would allow to combine the advantageous properties of both mentioned methods. This way is described in the next section:

it is essentially a combination of the two mentioned methods of displaying of the data. This (or similar) way is already used in several professional software packages (for example, *TNTmips*, *Leica Geosystems* or *RSI*), but, to the knowledge of the author, in the common practice it is not used as frequently as it would deserve.

3. Storing and displaying of the data

The most comfortable way of extracting the whole information contained in the data file is, according to our opinion, to display simultaneously the whole data as an image (colors representing the values of the displayed quantity) and, in the text form, the data parameters (coordinates x, y and the value of the displayed quantity f) at the pixel under the cursor. In this way it is easy to get information about the exact value of the displayed quantity at any data point. In order to facilitate this way of viewing of the data, it is advantageous to represent the data in a form suitable for both storing and displaying.

Therefore we introduce here the file format for efficient storing of the data and the programs for viewing and manipulating with the data. These programs were written by the author in the C language for the Linux (Red Hat 9) operating system and the KDE desktop environment. Of course, these tasks can be performed in many different ways; the mentioned programs are presented only as an example.

Consider any 2-dimensional data set with a regular net of data points. For the points of this net we introduce the internal coordinates i_x, i_y, which are integers satisfying the inequalities

$$0 \le i_x \le n_x - 1, \qquad 0 \le i_y \le n_y - 1, \tag{1}$$

where n_x and n_y are positive integers. The net of points has therefore $n_x \times n_y$ points.

In accord with the usual way of displaying pixels on the screen, we group the data points in lines containing points with the same y-coordinate; the lines are ordered from the top to the bottom (according to the decreasing y-coordinate), and, within each line, the points are ordered from the left to the right (according to the increasing x-coordinate). As for the internal

coordinates, i_x grows from the left to the right, and i_y grows from the top to the bottom.

Let x_a, x_b, s_x, y_a, y_b, s_y, (where $s_x > 0$, $s_y > 0$), be the minimal and maximal value and the step of the coordinates x and y of the data points, respectively; these parameters satisfy the conditions

$$x_b - x_a = (n_x - 1)s_x, \qquad y_b - y_a = (n_y - 1)s_y. \tag{2}$$

The rectangular coordinates x, y of the data points are therefore given by

$$x = x_a + i_x s_x, \qquad y = y_b - i_y s_y. \tag{3}$$

As the data net of points is regular, the data itself need not be represented as the triples (x, y, f), but the data file may contain only the values f ordered in the above described way. As the data values are real numbers, they are usually stored in the data file as floating point numbers in the `float` or `double` representation; the former has 7 and the latter 15 significant decimal digits. Although the precision of the shorter representation is usually sufficient for the data values obtained by a measurement, if the data values are calculated, it is almost inevitable to use (in the calculation itself) the longer representation (even if the initial values have not such a precision). We shall therefore assume that the data file contains the values f as a `double` floating point numbers. If we choose ⟨`name`⟩ as the name identifying the data set, such data file will be denoted as ⟨`name`⟩`.bdf`, where `bdf` means **b**inary **d**ata file containing **f**loating point numbers.

Of course, it is not sufficient to have only this data file, as it contains no information about the position of the data points or the numbers n_x, n_y. We shall assume that for each file ⟨`name`⟩`.bdf` we have the associated text file ⟨`name`⟩`.tdp` (where `tdp` means **t**ext file containing **d**ata **p**arameters), which contains the values of x_a, x_b, s_x, y_a, y_b, s_y (satisfying conditions (2)). Naturally, the number of values in the file ⟨`name`⟩`.bdf` has to be equal to $n_x n_y$.

As we have seen, by the `double` floating point representation of the data we store each data value with precision far exceeding the actual precision of this value; on the other hand, the `float` floating point representation would need only 4 bytes for each data value, but the precision of this representation is only 7 decimal digits. This is less than the precision of the `int` integer representation (9 decimal digits) stored in binary form also in 4 bytes. It

would be therefore advantageous to represent the data values as (signed) integers: this is easily possible, as usually the values of the whole data set have the same (or comparable) absolute precision. The representation of the value f by the integer i can be performed by a suitable choice of real numbers a and b ($b > 0$) such that we can have with a sufficient accuracy for each data point

$$f \approx a + i/b. \tag{4}$$

The value of the integer i can be then obtained by the formula

$$i = \mathrm{round}((f - a)b), \tag{5}$$

where round(x) is the function rounding the `double` value to the nearest integer (away from zero).

Let f_a and f_b be the smallest and the greatest value of the quantity f in the whole data set; if the integer i can acquire the values from $-I$ to I, according to (4) we can represent the data values by integers if

$$f_b - I/b \leq a \leq f_a + I/b, \tag{6}$$

and thus

$$f_b - f_a \leq 2I/b. \tag{7}$$

The latter inequality allows us to find the suitable value of b; then we can choose a using the former inequality. As the `int` integer can have the values from -2^{31} to $2^{31} - 1$, we choose $I = 2^{31} - 1$.

For simplicity we shall assume that the quantity $|f_b - f_a|$ is in the range between 1 and $2I$ (if it is not, it can be transformed by the suitable change of units); then it is possible to choose the value of the parameter b as an integer. As $b > 0$, we can represent b by an `unsigned int` integer and thus its value can be between 1 and 2^{32}. On the other hand, the values f can be absolutely much greater than $|f_b - f_a|$; the same then holds for the parameter a. In order to represent also this parameter by an integer, we write a as kK/b, where k is an `int` integer and $K = 2^{16}$. In the similar way it is possible to represent the parameters x_a, x_b, y_a, y_b by integers: if we assume that the length unit is one meter, each of these parameters can be written as m/L, where m is an `int` integer and $L = 2^{10}$ (thus these parameters are absolutely in the range between 2^{-10} and 2^{21}).

Now we can introduce the proposed representation of the 2-dimensional data as the ⟨`name`⟩.`fdi` file, where `fdi` means **f**loating point **d**ata represented by **i**ntegers. Similarly as by the `bmp` file format (details see in *BMP*), each `fdi` file contains a header and the data. The header contains 7 `int` integers and one `unsigned int` integer (totally 32 bytes), the data are represented as $n_x n_y$ `int` integers (each occupying 4 bytes).

The header of the `fdi` file contains the following integer parameters:
– 2 integers n_x and n_y,
– 2 integers representing the real numbers $2^{10}x_a$ and $2^{10}x_b$,
– 2 integers representing the real numbers $2^{10}y_a$ and $2^{10}y_b$,
– integer representing the real number $2^{-16}ab$,
– unsigned integer representing the real number $b-1$.

The data part of the `fdi` file contains the data values represented by integers according to the formulae (4) and (5); the data values are ordered as described above.

Note the difference between a `bmp` file and an `fdi` file: the header of the former file has 1078 bytes compared to 32 bytes for the latter file; the data part contains 4 bytes for each data point, but the latter file represents the data values with the precision of 9 decimal digits, while the former file contains only the information about the RGB values at each data point.

The `fdi` files represent a suitable tool for storing of the 2-dimensional data; they can be created using the program `bdftofdi`. This program creates the file ⟨`name`⟩.`fdi` from the given files ⟨`name`⟩.`bdf` and ⟨`name`⟩.`tdp`.

Viewing of the data is performed using the program `kview`. This program displays the `fdi` file of the dimensions up to $2^{16} \times 2^{16}$ data points (one data point corresponds to one pixel) using a linear transformation of the data values into 201 colors (colors are coded from -100 to 100, code 0 corresponds to the white color):

$$c = \mathrm{round}(r), \quad r = \frac{100}{f_+ - f_-}(2f - f_+ - f_-), \tag{8}$$

where f_- and f_+ are the values of f which have to be represented by colors -100 and 100, respectively. The parameters f_- and f_+ can be chosen independently on the parameters of the coding of the `fdi` file and they can be varied by the displaying. This allows to choose the interval of the data values which are displayed by the colors (these are the values f for which

color c defined by (8) satisfies the inequality $-100 \leq c \leq 100$); any value outside this interval is represented by the black color.

The data image is displayed within a window which fills the whole screen with the exception of a narrow border; if the whole data image cannot be displayed in this window, the window can be moved left, right, up or down in steps of 8, 64 or 512 pixels. The most important property of the program is the displaying of the information about the coordinates and data value at the point corresponding to the position of the cursor. In the lower border of the window, the following quantities are displayed:
– the position of the cursor with respect to the upper left corner of the data image, thus the numbers i_x and i_y appearing in the formula (3),
– color at the position of the cursor, thus the number c appearing in the formula (8),
– coordinates x and y of the data point calculated according to the formula (3),
– the data value f at this point.
In the upper border of the window, there are displayed the maximal possible values of the coordinates i_x and i_y, thus the numbers n_x-1 and n_y-1 (see the formula (1)), and the name of the data file. The color scale with the color codes is displayed in the left border of the window.

Finally, the program `bdfdouble` is used for doubling of the data net dimensions: it creates from the given files ⟨`name`⟩`.bdf` and ⟨`name`⟩`.tdp` two new files ⟨`name`⟩`_d.bdf` and ⟨`name`⟩`_d.tdp`, which contain data in the regular net of points with the step equal to the half of the original one. The additional data values are calculated from the original ones using the interpolation method developed by the author (it is an advanced moving least squares method). This allows to create the file ⟨`name`⟩`_d.fdi` besides the original file ⟨`name`⟩`.fdi`. If the original data image has $n_x \times n_y$ pixels, the doubled data image has $(2n_x-1) \times (2n_y-1)$ pixels. Note that, on the contrary to the zooming of an ordinary image, this doubled image contains an exact information about the data values at the additional pixels and these values have almost the same credibility as the original ones.

4. Conclusion and outlook

It is evident that the described displaying method is by far superior to the

displaying of the data image in the form of a `bmp` file (or an image file in any other image format): it allows to obtain easily and immediately the exact information about the data values at any data point. Moreover, the viewing program can be easily generalized to be able to display several quantities defined at the same net of points: the exact values of all these quantities at the given point can be displayed at once, while it could be possible to toggle between the images of these quantities. It has to be hoped that the described method of displaying of 2-dimensional data will be soon widely used and that it will replace the traditional viewing of the data images.

Note: the programs described in the previous section are considered as a free software and they can be obtained on the request from the author.

Acknowledgments. The author is grateful to VEGA, the Slovak Grant agency (projects No. 2/3057/23 and 2/3004/23), and to the Science and Technology Assistance Agency of the Slovak Republic (contract No. APVT-51-002804) for the partial support of this work.

References

BMP: http://www.daubnet.com/formats/BMP.html
Leica Geosystems: http://www.erdas.com
RSI: http://www.rsinc.com
TNTmips: http://www.microimages.com