Alternate methods for display

Contours

One way of increasing dynamic range in an image is to represent changes by means of contour lines rather than by absolute grey shading. By choosing the right contours any desired dynamic range can be achieved.

Contours are lines of constant something, whatever quantity we want to display. Most of you are familiar with topographic maps, for instance.

It is also possible to use non-evenly-spaced contours in order to emphasize certain characteristics of data. A common choice is logarithmic contours which stretch smaller values at the expense of compressing larger ones.

Contouring has been particularly used in surveying and topographic applications. Hikers soon get to know how to interpret the bumps and wiggles in a map so as to minimize hiking effor. This method of presentation for these data provides readability, quantitative application, and large dynamic range.

Resolution is not well defined in a contour map. Thus most topographic data are also depicted as shaded-relief images which underlie the contours. Shaded-relief maps indicate the local derivative (slope) of a region as a grey level, giving a hint of the local surface shape.

Often contouring is used for displays of non-topographic two-dimensional data. One example from the text is from radio astronomy. Here the brightness of a double radio source (Cygnus A) is shown:

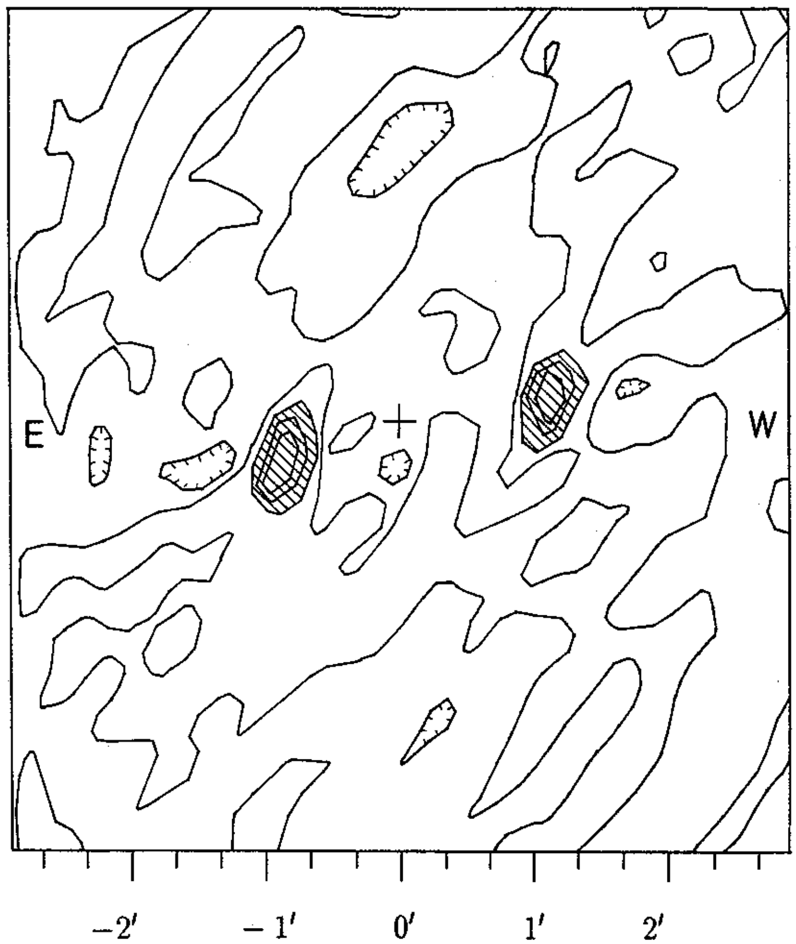


Figure 2-7 The double radio source Cygnus A, the first "radio star" to be discovered (1946), represented by equally spaced contours of brightness at 2.8 cm wavelength (Stull et al., 1975). The angular resolution, indicated by the scale of arcminutes, was achieved by rotation synthesis (a form of aperture synthesis) with the five-element minimum-redundancy array at Stanford.

<u>Producing contour plots by computer</u>

There are many algorithms around for producing computer-generated contour plots. Each image processing package has at least one, and a sample set of code is given at the end of Chapter 2 in the text. Once again, higher resolution data to begin with will generate better looking, and more accurate, contours.

Shaded relief

As mentioned above, details of surface shape are obtained if an image is formed by assigning an intensity to each point proportional to surface slope. Because slopes are both positive and negative, typically the dynamic range is split in half with a slope of zero assigned to the center, positive values ~~about~~ above it, and negative below.

For example, consider a data matrix

$$
\begin{matrix}
d_{11} & d_{12} & d_{13} & \cdots \\
d_{21} & d_{22} & d_{22} & \\
d_{31} & d_{32} & d_{33} & \\
\vdots & & &
\end{matrix}
$$

Assign an image R_{ij} according to

$$
R_{ij} = F \cdot \begin{bmatrix} \text{derivative of} \\ d \text{ in some} \\ \text{direction} \end{bmatrix} + \text{offset}
$$

$$
= F \cdot (d_{ij} - d_{i,j-1}) + 128 \qquad \text{, say,}
$$

for a 256-level (1 byte) system. This columnwise difference leads to the appearance of illumination from the left, of course any illumination direction may be chosen to highlight certain featurs. Such calculations are very fast and quite interpretable by experienced users.

Practical aspects of shaded relief generation

Unless data are very smooth initially, simple application of the above formula will result in a very noisy image. Why? If the data are contaminated by a white Gaussian noise process, differentiation tends to accentuate the appearance of the noise. In terms of linear filters, a differentiator has a transfer function that increases with frequency 6 dB/octave, so that the derivative will have relatively more high-frequency energy than the original process. If the quantities we are interested in are rather low frequency, they can be masked.

Hence when we calculate shaded relief, we usually average several adjacent values before display. We can view the shaded relief algorithm as a simple convolution of the form

$$s = k * f$$

where f is the original data, s is the shaded relief image, and k is a differentiating kernel. In the earlier formula

$$k = \begin{bmatrix} -1 & +1 \end{bmatrix}$$

To accommodate averaging, then, we could have $k = \begin{bmatrix} -1 & +1 \\ -1 & +1 \\ -1 & +1 \end{bmatrix}$ to achieve illumination from the left. Illumination from the top might then be

$$k = \begin{bmatrix} -1 & -1 & -1 \\ +1 & +1 & +1 \end{bmatrix}$$

and so forth.

Discrete numerical displays

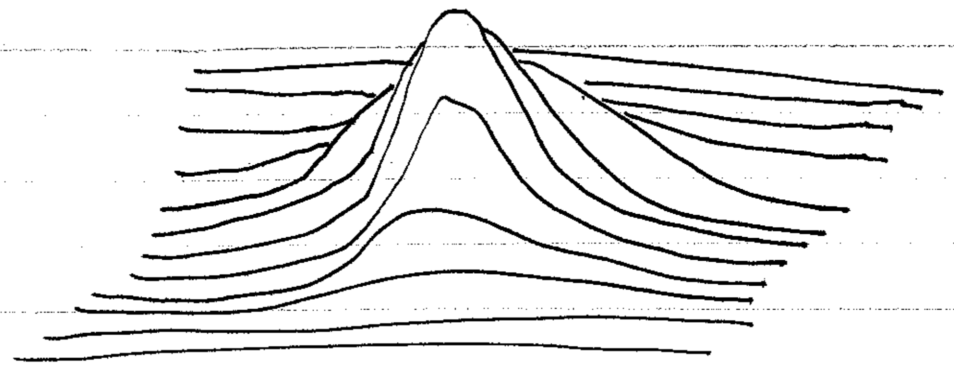
One of the earliest means of displaying data was simply to print out the numerical values, suitably scaled. One example might be:

0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	1	2	2	3	3	2	1	0
0	1	2	3	3	4	4	2	1	0
1	2	2	3	4	6	5	3	1	0
1	2	2	3	4	5	4	3	1	0
0	1	3	3	4	3	3	2	1	0
0	0	1	3	3	3	2	2	1	0

This representation makes it rather easy to estimate contour lines by eye. It is, however, not a compact representation for us. For a computer it's great.

Staggered profiles

Another way to present image data is to use staggered profiles, which are a series of cuts through an object plotted one after another. If the succeeding lines are displaced slightly, or staggered, interpretation is easier. More sophisticated algorithms hide "hidden lines".



↑
Even my crude drawing is immediately recognizable as a hill.

Choice of display

The multiplicity of display modes means that there is a choice to be made whenever an image is to be presented. One should consider carefully the source distribution and the information to be conveyed in selecting the appropriate display. Optimal choices will get your message across in the clearest way, and a little reflection will often result in your work being perceived more easily and by more people.