Geo-Processing, **1(1980) 231—242**
Elsevier Scientific Publishing Company, Amsterdam — Printed in The Netherlands

A FAMILY OF SUBROUTINES FOR PLOTTING GRADUATED SYMBOL MAPS

WOLF D. RASE

Bundesforschungsanstalt für Landeskunde und Raumordnung
Pf 200130, Bonn 2, Federal Republic of Germany

ABSTRACT

Rase, W. D., 1980. A family of subroutines for plotting graduated symbol maps. Geo-Processing, 1:231-242.

To facilitate computer-assistance in the production of thematic maps a family of subroutines for drafting graduated symbols has been designed and implemented. Alternatively length, area or volume may be made proportional to the value to be represented. Symbols can be hatched or filled with area symbols. Because the programs are designed for a minicomputer system priority has been assigned to low memory requirements. To minimize execution time it was decided not to solve the general case alone but also to provide separate subroutines for each symbol type. The visual appearance is improved through elimation of hidden areas and a halo option for the case of overlapping symbols. Restriction to a vector device for graphic output reduces the hidden-surface problem to a hidden-line problem which is geometrically more tractible. Modular design facilitates modification, maintenance and extension of the programs.

1. Graduated symbol maps in cartography

Graduated symbol maps are commonly used in thematic cartography to display absolute values at discrete points. For example, population figures are represented by circles, whose areas are proportional to the values. Variations in the regional distribution are displayed visually by variations in circle sizes. The circle is the most frequently used graduated symbol: circles are easy to draw and give an aesthetically pleasing appearance. Other geometric forms such as regular polygons (triangles, squares, hexagons etc.) or irregular polygons are found less frequently, but are nevertheless used in cartographic practice (fig. 1). In some cases an absolute value is associated with a relative value, for example the number of inhabitants with the population density or with the increase or decrease during some time interval. These relative values are usually represented by filling the symbol area, e.g. hachuring, area symbols, color or a combination of these. To facilitate visual perception area filling is used even without the need to represent an additional relative variable. In this case the filling is the same for all symbols.

The human mind is more accustomed to relating a graphically displayed value to the area of the symbol than to length or (pseudo-) volume. Thus in most cases a symbol

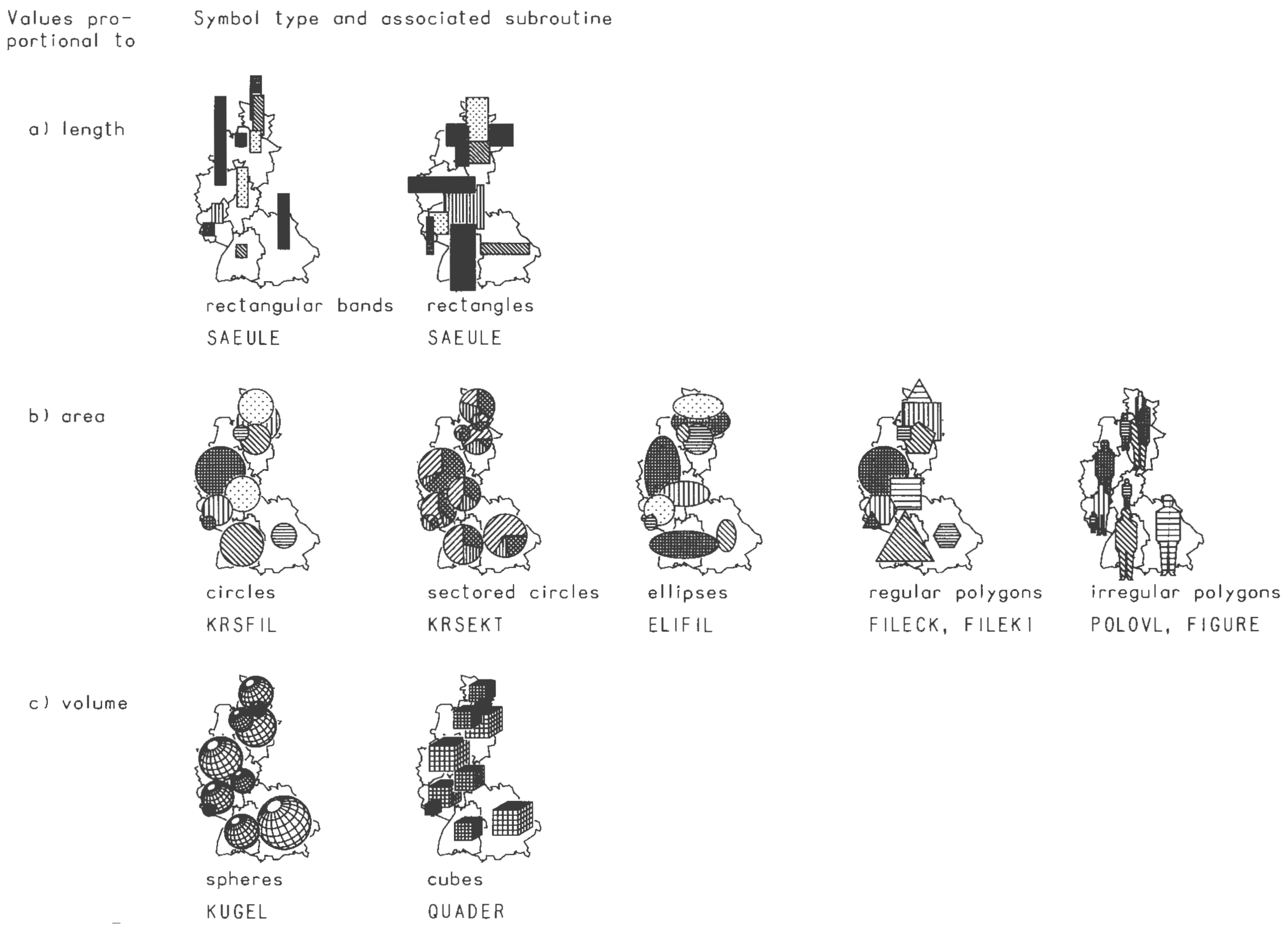


Fig. 1. Graduated symbol maps.

representing area proportionality will be selected (square root method, fig. 1b). The usual underestimation of larger symbols may be corrected by a simple transformation, e.g. by applying an exponent to the area values (Flannery, 1971). Special cases, however, require proportionality in length or volume (fig. 1a, 1c), for example, when the data interval is very narrow or very broad (Robinson et al., 1978).

The simultaneous representation of two or more variables by size, form and filling was mentioned above. The relation of two variables may be expressed in symbol form alone as in fig. 5. The ratio of major to minor axis in the ellipses is equivalent to the ratio of low-income and high-income farms. Other multivariate symbols suggested in the cartographic literature (Arnberger, 1966) are not implemented in the current program versions, but may be constructed with the existing building blocks.

To improve the visual separation of overlapping symbols two techniques are available which have found widespread use in computer graphics. The first is the elimination of hidden (overlapped) areas. The second is the halo effect: a small gap between the symbols enhances the impression of depth (Appel et al., 1979).

2. Objectives in program design and implementation

The manual drafting of graduated symbols is a costly and error-prone task. To economize the production of thematic maps for regional analysis and regional planning an assortment of FORTRAN subroutines has been developed complementing other software for computer-assisted mapping (Rase, 1978). In light of the hardware configuration at the BfLR (PDP-11/45, Calcomp flatbed plotter, Tektronix CRT) and presumably that of other potential users the following restrictions were imposed:

- graphic output on vector device (plotter),
- low memory requirements for execution on a minicomputer system,
- reasonable execution time for interactive use,
- portability to reduce the relative development costs.

The use of a plotter or some other vector device restricts the area filling to lines or area symbols composed of lines. The current program versions support the plotting of hachures (lines and dashed lines) and characters from the Calcomp software. The user has complete freedom in choosing hachure angle, density, dash length, pen number, size and orientation of the characters. Up to three hachures may be combined in one area filling. The software interface to the graphic devices consists of three subroutines provided by the standard Calcomp software (PLOT, NEWPEN, SYMBOL). These routines are generally available for non-Calcomp devices, too, and are easy to emulate.

Low memory requirements and fast execution are mutually exclusive in most cases. With respect to the small address space of the minicomputer, priority has been assigned to low memory requirements. For one symbol type, the regular polygon, two versions have been provided, one version optimized for memory, the other for speed. Test and production runs yielded the surprising result that the Tektronix display is often slower than the CPU for the applications and data sets typical at our institution. Hidden-line elimination in boundary networks (see figures) requires considerable execution time, however. It is not necessary to perform this step in interactive mode as it can be deferred until final drafting.

The decision not only to solve the general case but also to provide individual subroutines for each symbol type led to a considerable reduction in execution time. As mentioned above the circle is the most frequently used graduated symbol. It is possible to define a circle as a polygon (with a large number of vertices) and apply the general case for irregular polygons. This would be very inefficient because the calculation of circle intersections is much simpler than the calculation of polygon intersections. The general case requires much more calculations reducing the speed. Individual routines, on the other hand, require some additional memory. To minimize memory common functions, e.g. calculation of intersections, sorting, line and symbol plotting etc., are provided as commonly used subroutines. An additional benefit of this structure is that improvement and maintenance are less troublesome, and building overlay structures is simplified.

At the moment the subroutines are assembled into dedicated main programs to fulfil actual production requirements at our institution. A number of service routines for plotting boundary networks, legends and other map features have been provided to minimize manual drafting. Some of these functions can be found in the map examples. The subroutines will be incorporated into a future interactive system for the design and display of graduated symbol maps.

3. The algorithm

The basic algorithm is the same for all symbols, with one exception to be discussed below. For each single symbol the following steps are performed by separate subroutines:

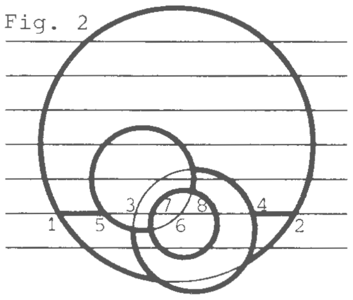
a. the overlapping symbols are identified;
b. hachure or surface texture is plotted, omitting the the hidden or haloed line segments;
c. the symbol outline is plotted, again eliminating hidden or haloed segments.

This functional separation facilitates the tuning of the programs for specific applications. In the current version the identification of overlapping symbols is time-consuming for very large numbers of symbols. A user may replace this step by his own, more efficient version, for example by segmenting the data into patches, by defining neighbourhood relations prior to plotting etc. The rest of the code (hidden-line elimination, plotting of the outline) is unaffected.

The algorithm for hidden-line elimination is quite simple compared with the algorithms used for display of three-dimensional objects, e.g. surfaces, molecule models or buildings (Sutherland et al., 1974). According to cartographic practice smaller symbols cover the larger ones. The symbols have zero thickness, thus there are no projective transformations or depth-priority problems. Due to the restriction to a vector device the problem of hidden areas is reduced to a hidden-line problem. This lower level of geometric complexity simplifies the algorithm considerably and leads to faster execution times.

The easiest way to explain the algorithm for elimination of hidden lines is through the example in fig. 2.

Fig. 2



The largest circle is intersected by a number of hachure lines parallel to the horizontal axis. For the second line from the bottom the intersection values (x-coordinates) are S1=0.5 and S2=7.5. The coordinates are stored in a table, together with the attribute "neutral" or "0" in numerical form. The overlapping circle C2 has the intersection values S3=2.8 and S4=6.2. The attributes for overlapping circles are defined as "entry" and "exit", or "1" and "-1" in numerical form. Coordinates and attributes are stored, as well as the coordinates and attributes of the remaining circles. The complete list is then as follows:

No.	S	A
1	0.5	0
2	7.5	0
3	2.8	1
4	6.2	-1

5	1.9	1
6	4.1	-1
7	3.2	1
8	5.2	-1

When the list of overlapping circles is exhausted the table is sorted into ascending order on S:

No.	S	A	Sum	Action
1	0.5	0	0	start plot
5	1.9	1	1	end plot
3	2.8	1	2	skip
7	3.2	1	3	skip
6	4.1	-1	2	skip
8	5.2	-1	1	skip
4	6.2	-1	0	start plot
2	7.5	0	0	end plot

The line segments are plotted only if the sum of attributes equals zero. In our example only the segments from S1 to S5 and from S4 to S2 are drawn, the other segments are omitted. (The similarity with a LIFO stack is quite obvious: the attributes "entry" and "exit" are equivalent to "push" and "pop" operations; the line is plotted only in the neutral position of the stack.)

For hachure angles not equal to zero it is necessary to calculate the euclidian distances of the intersections from an endpoint. An alternative is rotation of the coordinate system and retransformation into the original coordinate system for plotting. This latter alternative turned out to be the more efficient procedure. The algorithm is thus a relative of the "segment-polygon algorithm" for the hachuring of polygons (Brassel et al., 1979).

The calculation of the intersections of a straight line with a circle (or a circle with a circle) is elementary. The arithmetic formulas can be found in most textbooks of analytical geometry. The intersection calculation for polygons, however, is less straightforward. Because direct analytical solutions do not exist, each edge of the polygon has to be checked for intersections. Thus execution time is proportional to the number of vertices in the polygon. Large numbers of symbols and/or vertices may lead to excessive execution times.

The three-dimensional appearance of spheres (fig. 6) is achieved by plotting globe-like shapes with parallels and meridians. Other techniques for the three-dimensional representation of surfaces, e.g. inclined contours, analytical shading and other line-oriented techniques (Peucker et al., 1974) may be implemented in future program versions. The user will have more freedom to choose the appropriate surface texture to improve visual perception. The surface of the sphere is the case for which the standard algorithm described above is not used. The purely analytical solution -intersection of rotated ellipses and circles- is much more cumbersome and time-consuming than the heuristic-iterative approach applied. The algorithm resembles the "bisection method" for the solution of algebraic equations (Gruenberger et al., 1965).

The halo effect is generated by increasing the symbol size temporarily during the area-filling step. The circle is again the simplest case; it is sufficient to increase the radius by a certain factor. For irregular polygons a line parallel to the outline outside the polygon has to be computed. The special problems of parallel line generation, e.g. small angles resulting in long spikes, loops and points with equivalent coordinates, have to be recognized and solved. In most cases the original number of vertices is doubled resulting in increased processing time.

4. Examples and execution times

Figures 3 to 6 are examples of production runs using data from demographic and agricultural records. Generally the execution time depends on

- number of symbols,
- number of overlaps,
- symbol size,
- hachure density,
- symbol type, number of polygon vertices.

The factors may influence each other. Thus it is not possible to predict execution time precisely. The examples merely provide a rule-of-thumb for the estimation of the execution time of real production runs.

The time figures below are elapsed time, not CPU time, for execution of the specific subroutine including output to the plot file. The jobs were run on a DEC PDP-11/45 minicomputer system under RSX11-M. The programs were compiled with the FORTRAN 4-PLUS compiler.

Fig.	Function	Subroutine	min:sec
3	Regular polygons	FILECK	1:45
4	Sectored circles	KRSEKT	2:05
5	Ellipses	ELIFIL	1:55
6	Spheres	KUGEL	5:26

For circles, ellipses and regular polygons execution time is quite moderate.

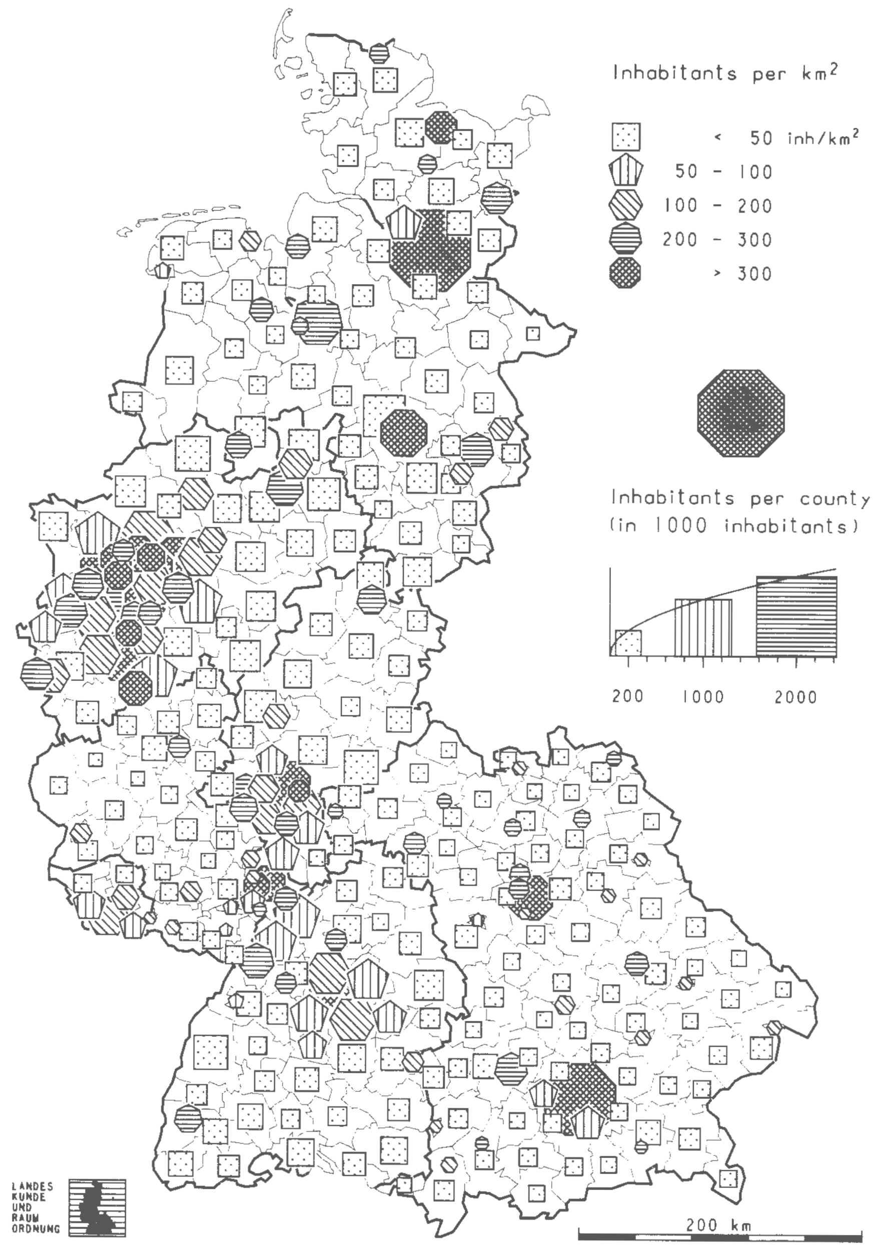


Fig. 3. Population and population density 1977.

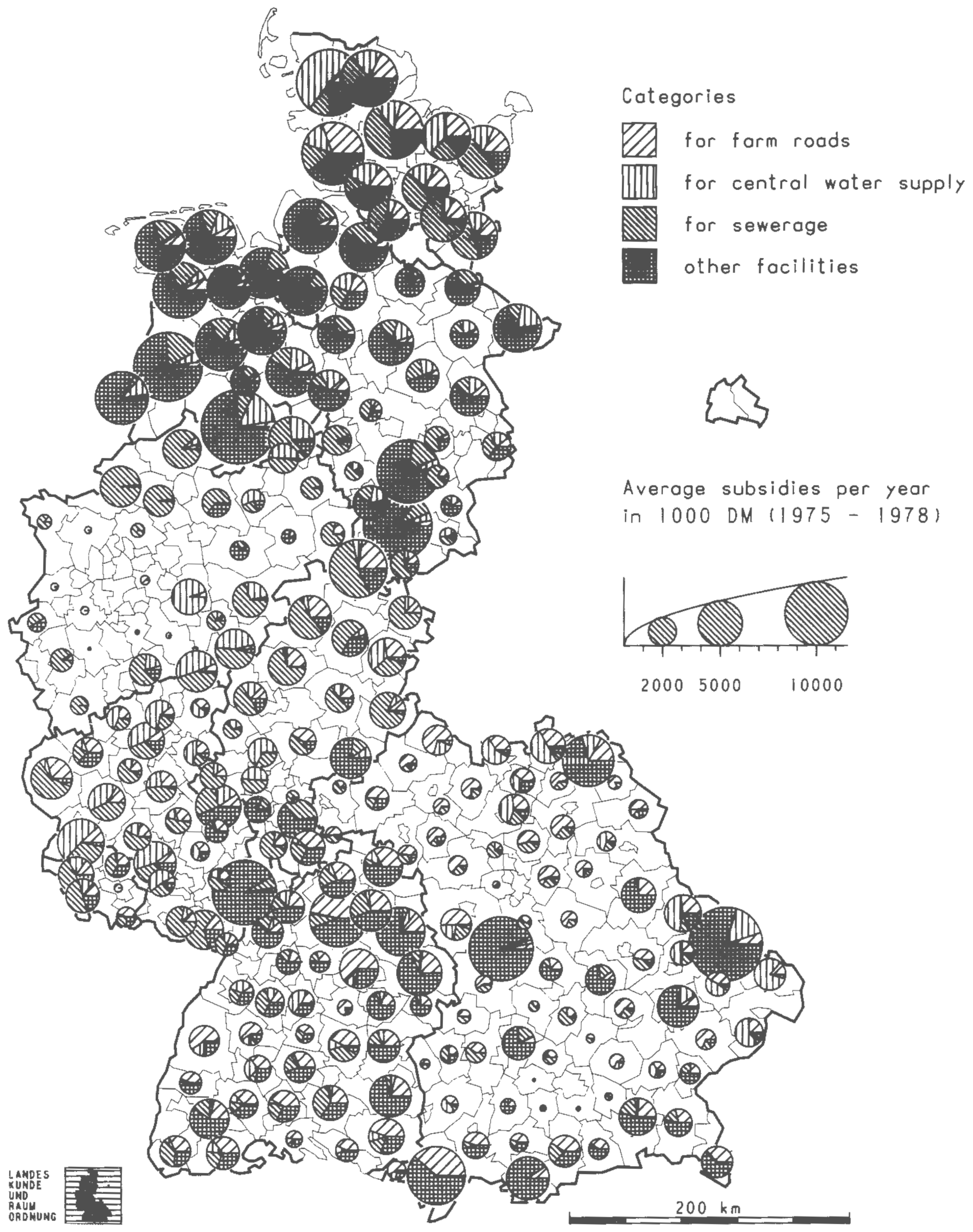


Fig. 4. Subsidies for agricultural infrastructure.

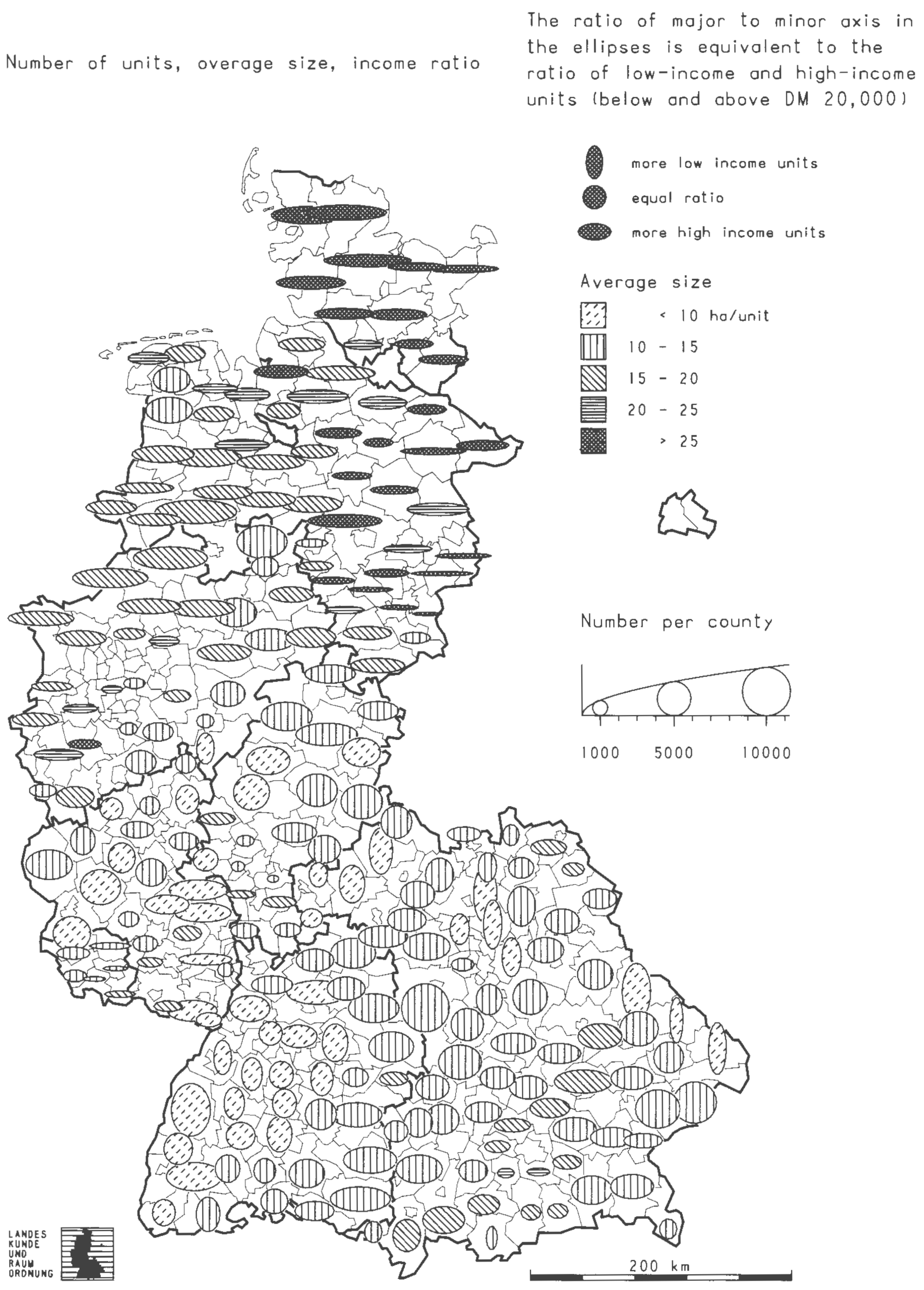

Number of units, average size, income ratio
The ratio of major to minor axis in the ellipses is equivalent to the ratio of low-income and high-income units (below and above DM 20,000)
more low income units
equal ratio
more high income units
Average size
< 10 ha/unit
10 - 15
15 - 20
20 - 25
> 25
Number per county
1000
5000
10000
200 km
LANDES KUNDE UND RAUM ORDNUNG

Fig. 5. Agricultural production units 1977.

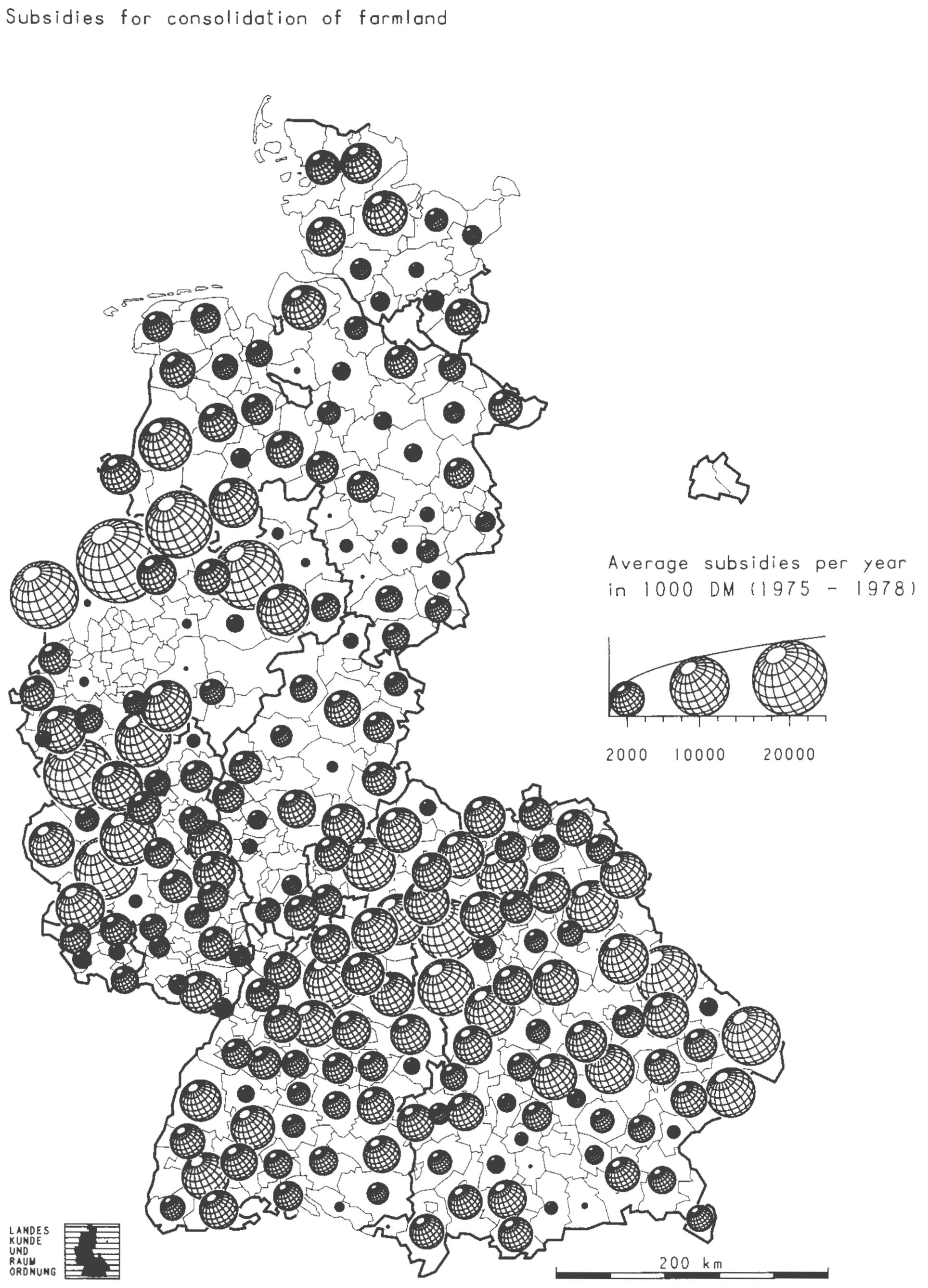

Subsidies for consolidation of farmland
Average subsidies per year in 1000 DM (1975 - 1978)
2000
10000
20000
200 km
LANDES KUNDE UND RAUM ORDNUNG

Fig. 6. Agricultural subsidies 1975-1978.

Spheres need some more time due to the surface texture. Irregular polygons are much more time-consuming. The data of fig. 3 was executed with the man-like shapes in fig. 1b (40 vertices per polygon). The symbol plotting step (including area filling) alone took 40 minutes, the hidden-line elimination in the boundary network appr. 23 hours. The execution times clearly mark the limits of the minicomputer and some subroutines, respectively.

For questions concerning the availability of the programs please contact the author.

References

1. Appel, A., Rohlf, F. J., Stein, J. A., , 1979, The haloed line effect for hidden line elimination. Computer Graphics, Vol. 13, No. 2, August 1979, pp. 151 - 157

2. Arnberger, E., 1966, Handbuch der thematischen Kartographie, Wien 1966

3. Brassel, K. E., Fegeas, R., 1979, An algorithm for shading of regions on vector display devices. Computer Graphics, Vol. 13, No. 2, August 1979, pp. 126 - 133

4. Flannery, J. J., 1971, The relative effectiveness of some common graduated point symbols in the presentation of quantitative data. The Canadian Cartographer, Vol. 8, No. 2, December 1971, pp. 96 - 109

5. Gruenberger, F., Jaffray, G., 1965, Problems for computer solution, New York 1965, pp. 11 - 18

6. Peucker, T. K., Tichenor, M., Rase, W. D., 1974, The computer version of three relief representations. Display and Analysis of Spatial Data, edited by J. C. Davis and M. J. McCullagh, London 1974, pp. 173 - 186

7. Rase, W. D., 1978, Computerunterstützte Zeichnung thematischer Karten für die Bundesraumordnung. Kartographische Nachrichten, 28. Jhrg., H. 8, Dezember 1978, pp. 201 - 209

8. Robinson, A., Sale, R., Morrison, J., 1979, Elements of cartography, 4th edition, New York 1978

9. Sutherland, I. E., Sproull, R. F., Schumacker, R. A., 1974, A characterization of ten hidden-surface algorithms. Computer Surveys, Vol. 6, No. 1, March 1974, pp. 1 - 55