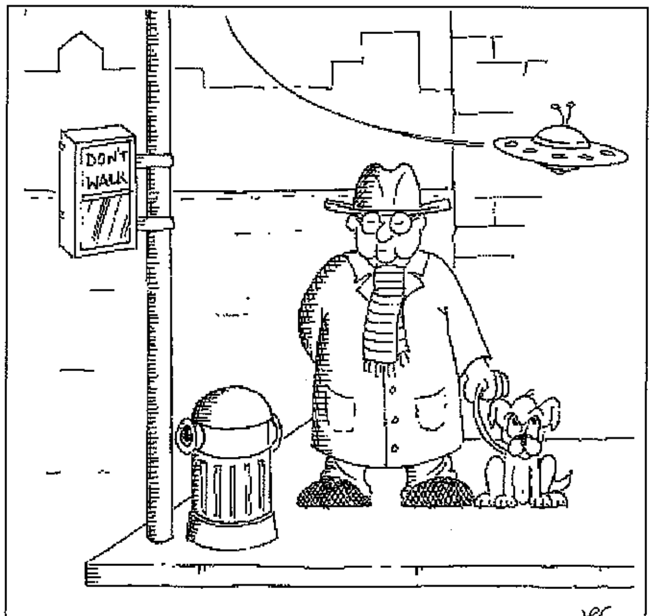
CHAPTER 3

Maps as Numbers

Yes raster is faster, but raster is vaster, and vector just seems more correctior.
—*C. Dana Tomlin,* Geographic Information Systems and Cartographic Modeling *(1990), p. 44*

Your lucky number is 3552664958674928. Watch for it everywhere.
—*Anonymous.* Berkeley Unix *"Fortunes" file*

...the Map must go ever forward, no matter what obstacles and hazards its servants may encounter.
—*Showell Styles.* The Forbidden Frontiers: The Survey of India from 1765 to 1949 *(1970), p. 78*



"Prepare to perish, carbon-based Earth-scum! (GAK!!) OK guys, who fouled up on the scale conversion for the invasion map?"

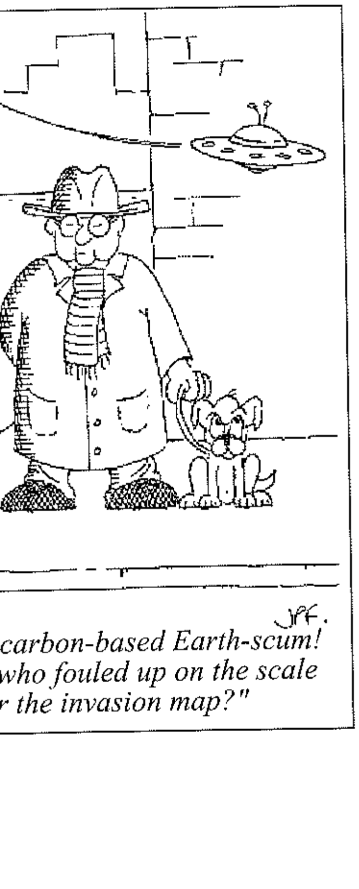
3.1 REPRESENTING MAPS AS NUMBERS

In this chapter we look at the various ways that maps can be represented using numbers. All GISs have to store digital maps somehow. As we will see, there are some critical differences in how the various types of GIS navigate on this ocean of geographic numbers. The organization of the map into digits has a major impact on how we capture, store, and use the map data in a GIS. In Chapter 4 we will see that an important first stage in working with GIS is just getting the map in the right form of numbers into the computer.

Obviously, there are many ways that the conversion of a visual or printed map to a set of digits can be done. Over the years, the designers of GIS and computer mapping packages have devised an amazing number of ways that maps can be converted into numbers. The difference between the ways is not trivial, not only because different types of files and codes are needed, but because the entire way that we think about the data in a GIS is affected. The link between how we imagine the features that we are working with in the GIS and the actual files of bytes and bits inside the computer is a critical one. To the computer, the data are stored in a physical structure that is quite tangible, at least to the GIS software. The physical structure is not only how computer memory such as disk and RAM is used, but also how the files and directories store and access the map and attribute information.

On the physical level, the map, just like the attributes, is eventually broken down into a sequence of numbers, and these numbers are stored in the computer's files. In general, two alternative ways exist of storing the numbers. In the first, each number is saved in the file encoded into binary digits or bits. A number in base 10, decimal, can be converted into base 2, binary, and the binary representation stored as on or off, 1 or 0, in the files on the computer. Eight bits in a row are termed a *byte*, and one byte can hold all numerical values from 0000 0000 to 1111 1111 binary or 0 to 255 decimal. Many computer programmers use shorthand to represent a byte, since in base 16 (hexadecimal) one byte can hold two hexadecimal digits. Hexadecimal runs out of counting digits at 9, so it fills the gaps with letters. Counting in hexadecimal then goes in the sequence `0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F`. To a computer programmer, the range of numbers that can be stored in one byte goes from hexadecimal `00` to `FF`. Typically, as a GIS user you will never see hexadecimal values unless you crash the system or try to look at files stored in this binary format rather than the alternative.

The second way of encoding numbers into files is to treat each number the way that humans do—one decimal digit at a time. Coincidentally, this is the same way that we deal with text, punctuation, and so on, so this format is often called *text* or *ASCII files*. ASCII stands for the American Standard Code for Information Interchange, and the codes are 256 standard meanings for the values that fall into one byte. Some of the ASCII codes are numbers, some lowercase and some uppercase letters, some special characters such as "$" or ">" and some are actually keys to operations (such as the escape codes or tabs). Each fits into one byte, and we store in the file the numbers just as they come, even using indentation and spacing. These files are usually suitable for use with an editor, and they can be printed and read without a program.


carbon-based Earth-scum!
who fouled up on the scale
r the invasion map?"

It is the logical structure of the data that requires us to have a mental "model" of how the physical data represent a geographic feature, just as a sheet map is a flat paper "symbol" model of the landscape it covers. Traditionally in GIS and computer cartography, there were two basic types of data model for map data, and only one for attribute data. Map data could be structured in *raster* or *vector* format, and attributes as flat files.

A raster data model uses a grid, such as the grid formed on a map by the coordinate system, as its model or structure to hold the map data. Each grid cell in the grid is one map unit, often chosen so that each grid cell shows on the GIS map as one screen display point or *pixel*, or on the ground as a whole-number increment in the coordinate system. A pixel is the smallest unit displayable on a computer monitor. If you get a magnifying glass and look at a monitor or a television set, you will see that the picture is made from thousands of these tiny pixels, each made up of a triangle of three phosphor dots, one dot for red, one for green, and one for blue (if your screen displays color). When we capture a map into the raster data model, we have to assign a value to every cell in the grid. The value we assign can be the actual number from the map, such as the terrain elevation in a digital elevation model (DEM), or more usually, it is an index value standing for an attribute that is stored separately in the attribute database.

Key elements in raster data are shown in Figure 3.1. First, the cell size determines the resolution of the data, and the cell size has both a map and a ground expression. We often talk about 30-meter Landsat data, for example, meaning that each cell in the data is 30 meters by 30 meters on the ground. On the map, we may use several pixels to display the grid cell, or on paper we may use a dot of a certain size in a given color. Second, the grid has an *extent*, often rectangular since a grid has columns and rows, and even if we do not wish to store data in the GIS for grid cells outside our region (such as a state), we still have to place something (usually a code for "outside") in the grid cells. Third, when we map features onto the grid there is sometimes an imperfect fit. Lines have uneven widths, points must be moved to the center or the intersection points of the grid, and areas may need to have their edges coded separately. We sometimes have to determine in advance what connections within the grid are legal. For example, taking a single cell, we can allow connections only north, south, east, and west, like the way a rook moves in chess, or we can allow diagonal connections as well. Which we choose can mean a great deal as to how the GIS works at storing and using the features. Fourth, when we deal with a grid, each grid cell can usually only be "owned" by one feature, that is, the one whose attribute it holds. In many cases, map data are not so simple. Soils, for example, are often listed by their percentage of sand, silt, and clay at every point. Finally, when we have a grid, every cell in the grid has to be made big enough to hold the largest value of the attribute or index to be stored in the grid. You may have had the experience of using a spreadsheet or table to store people's names. Even when we store "JaneDoe" with only eight characters, we still have to allow for the occasional very long name. Every grid cell pays the storage penalty of the extra space, and with the total number of cells being the product of the numbers of rows and columns, the amount of space needed

to have a mental "model"
st as a sheet map is a flat
ally in GIS and computer
nap data, and only one for
or format, and attributes as

on a map by the coordinate
grid cell in the grid is one
e GIS map as one screen
ncrement in the coordinate
puter monitor. If you get
set, you will see that the
made up of a triangle of
ne for blue (if your screen
model, we have to assign
be the actual number from
n model (DEM), or more
is stored separately in the

rst, the cell size determines
and a ground expression.
meaning that each cell in
map, we may use several
dot of a certain size in a
r since a grid has columns
ne GIS for grid cells out-
nething (usually a code for
nto the grid there is some-
st be moved to the center
to have their edges coded
hat connections within the
ow connections only north,
, or we can allow diago-
at deal as to how the GIS
leal with a grid, each grid
the one whose attribute it
r example, are often listed
Finally, when we have a
old the largest value of the
ad the experience of using
we store "JaneDoe" with
nal very long name. Every
h the total number of cells
e amount of space needed

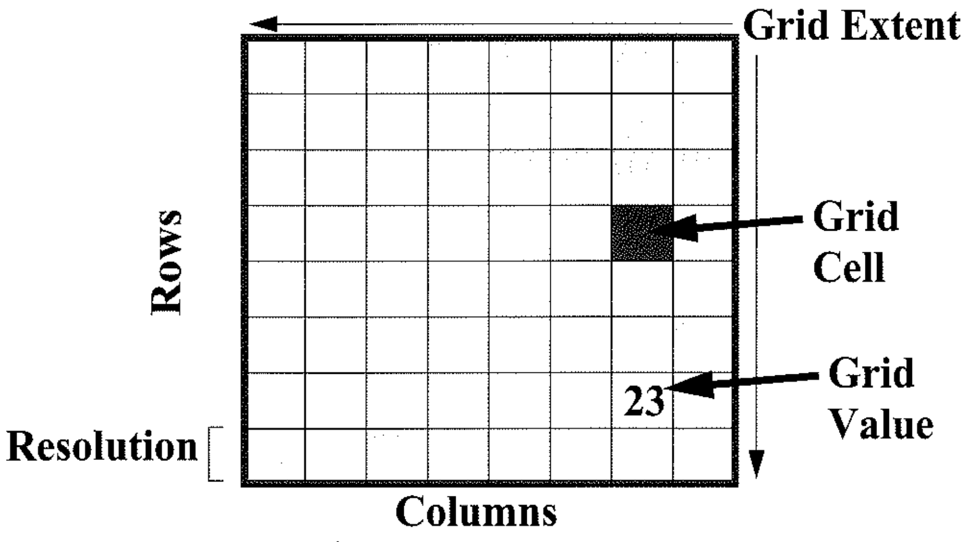


FIGURE 3.1: Generic structure for a grid.

can add up (or rather multiply up) quickly. Storage sizes for grids often increase by powers of 2 as more and more "bytes" of 8 bits are needed to store larger and larger values.

Nevertheless, raster grids have many advantages. They are easy to understand, capable of rapid retrieval and analysis, and are easy to draw on the screen and on computer devices that display pixels. Mark Bosworth imagines raster grids as being like the music of Mozart (Figure 3.2): detailed, repetitive, highly structured, and elegant. Slowly, dainty step by tiny step, they build the theme of the music into a glorious single structure, even though it may have "too many notes."

The other major type of data model for map data is the vector. The vector is composed of points, each one represented by an exact spatial coordinate. For a point or a set of points, vectors just use a list of coordinates. For a line, we use a sequence of coordinates; that is, the sequence of points in the list is the order in which they must be drawn on the map or used in calculations. Note that this gives lines a "direction" in which their points should be read. Areas in the vector model are the space enclosed by a surrounding ring of lines, either one or several of them.

Vectors are obviously very good at representing features that are shown on maps as lines, such as rivers, highways, and boundaries. Unlike the raster grid, where we have to store a grid cell's attribute whether we need it or not, we need only place points precisely where we need them. A square can be four lines connecting four points, for example. Even wiggly lines can be captured quite well in this way, by using more points for the bends and fewer when the line is straight. Using vectors, we can draw an outline map with only a few thousand points, far fewer than the number of grid cells that would be required. To complete the musical analogy, vectors are more like the music of Beethoven.

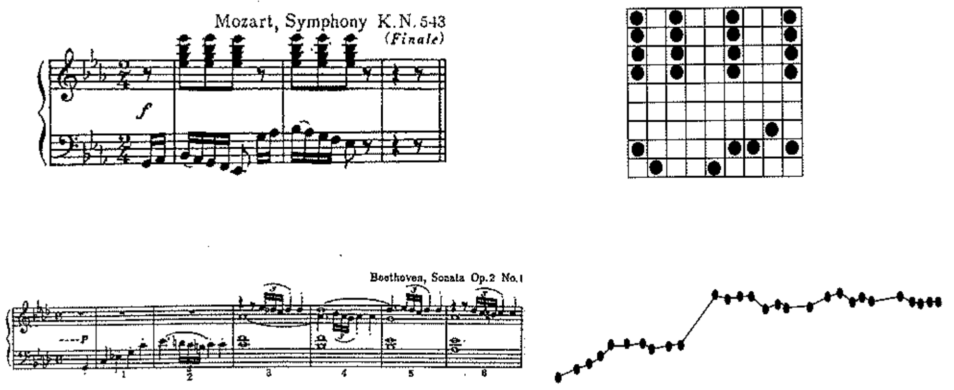


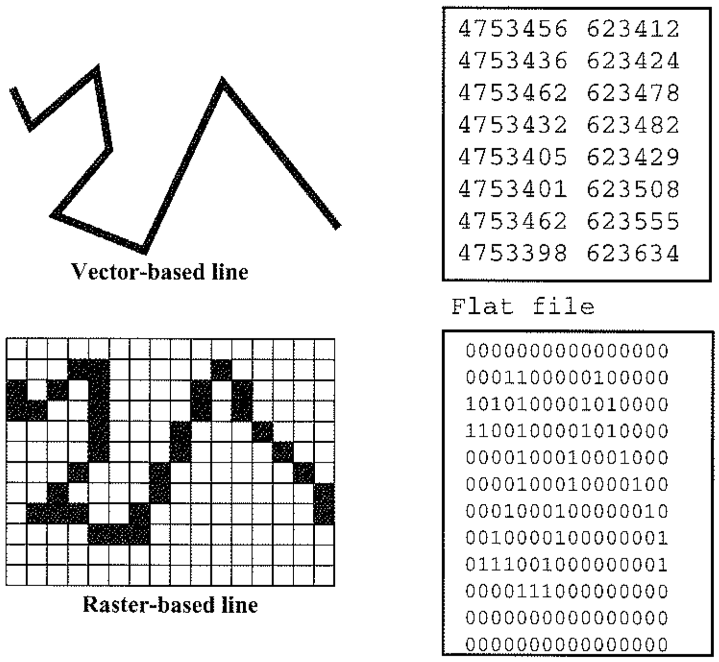
FIGURE 3.2: Data structures as music. Rasters are detailed, repetitive, highly structured, and elegant; slowly, dainty step by tiny step, they build the theme of the music into a glorious single structure, even though it may have "too many notes," like the music of Mozart. Vectors are bold, leaping streaks that go from place to place with rapidity and efficiency, like the music of Beethoven.

Vectors use bold, leaping streaks that go from place to place with rapidity and efficiency. There is little repetition, and the vectors get straight to the guts of the geographic features they represent.

Vectors have the advantage of accuracy, since they can follow features very closely. They are efficient at storing features. They are very suitable for plotting devices that draw with pens or points of light, as the features can be drawn one at a time in their completeness. They can also be adjusted to store information about connectivity to other features, as we discuss under topology in Section 3.4. On the down side, the vector is not very good at representing continuous field variables such as topography, except in a way we consider later called the *triangulated irregular network*. Vectors are also not a good structure to use if the maps to be generated involve filling areas with shades or color.

We have said little yet about the attribute data other than that its model is as a *flat file*. A flat file is how numbers are stored in tables or in a spreadsheet. The model is also a sort of grid, with rows for records and columns for attributes (Figure 3.3). Just like for the raster grids, we have to store values in the cells of the table. As we have already noted, these values must somehow link the data in the flat file to the data in the map. For a raster grid, we could store index numbers in the grid and any number of attributes for the index numbers in the flat file. For example, on a land use map, 1 could stand for forest, 2 for farmland, and 3 for urban. For vector data, we need a little more complexity. Point data are simple; we can even put the coordinates in the flat file itself. Lines and areas, however, have variable numbers of points. We again need to number off or "identify" the lines and store an attribute for the whole line in the flat file. We can do the same for an area, except that we need the line flat file as well to refer to in the polygon or area file. If we called the lines *arcs*, for example, we might need both a polygon attribute table file, and a file of arcs by polygon.

So far we have touched only briefly on the data models that GISs use. In the sections that follow, we first delve more closely into the way that attribute data are stored in files and how the tricks of data storage have improved over the last 30 years.



```
4753456 623412
4753436 623424
4753462 623478
4753432 623482
4753405 623429
4753401 623508
4753462 623555
4753398 623634
```

```
0000000000000000
0001100000100000
1010100001010000
1100100001010000
0000100010001000
0000100010000100
0001000100000010
0010000100000001
0111001000000001
0000111000000000
0000000000000000
0000000000000000
```

FIGURE 3.3: Both vectors and rasters can be stored in flat files, if they are simple.

Then, in the following sections we go into more detail and cover different logical and physical data models that GIS systems use to store the map data. These too have evolved over time and have improved significantly over the years. We will look at some of the formats that are used by GIS data providers and finish the chapter by raising some of the technical problems that have faced GIS users who need to move data between formats and between systems. The latter aspect has immense importance for the future of GIS and will be revisited in Chapter 10.

3.2 STRUCTURING ATTRIBUTES

In Chapter 5, we consider in more detail the logical way that attribute data are stored in files. For now, a simple way to imagine this is to recall the flat file we covered in this chapter. A flat file is a table. Columns store attributes, and rows store records. We know in advance what sort of information is stored in each attribute, whether it is text or numbers, how large the numbers are and so on. We can then write a sequence into the file. For each record we can write the ASCII codes for the values in each attribute (in database terms often called a *field*) in a consistent way. At the end of each record we could start a new line. The file then would be a sort of table or matrix with rows and columns.

It is now easy to see what some of the database operations actually do. For example, if we wished to sort the data we could renumber the lines in the file. If we wanted a particular record, we could search line by line until we found the correct one and then print it. These operations would be much faster if we could encode the numbers in binary or sort them in the file so that the most commonly referenced records were first in the file.

Most database management systems (DBMSs) do exactly this, and some use very clever ways of placing records into files. If you use a database manager, or even a spreadsheet program on a personal computer (PC), you have probably noticed these files, which hold your records in one place.

Another important part of structuring attributes is that the *database dictionary*, the list of all the attributes along with all their characteristics, must also be written into the file. This is sometimes done by using a separate file, but usually the dictionary is written into the top or *header* of the file, before the data begin. So the attribute database part of a GIS is fairly simple. At the most basic it consists of a file. At the most complex, it might be several files in a directory. From a data management point of view, manipulating the attributes is a piece of cake. Unfortunately, the maps are a little harder to deal with.

3.3 STRUCTURING MAPS

Maps have at least two dimensions; in the earth's space they have latitude and longitude, and in the map's space they have the left-right (x) and the up-down (y) directions. They are also scaled-down representations of features, features that can be points, lines, areas, or even volumes. Point features are very simple to deal with, and you could easily argue that you don't really even need a GIS for point features other than to draw them. This is because x and y can be stored just as regular attributes in a standard database. Line and area features are more complicated because they can be different shapes and sizes. A stream and a road would be captured with different numbers of points, and these would not fit easily into the attribute database.

Nothing says that we have to capture points, however. In Section 3.1, we met the two basic models for map data, the vector and the raster. While a GIS can usually deal with data from either format, only one structure can be used for retrieval and analysis of the map. We have already seen how the structure we choose can influence many GIS operations. The data structure also affects the type and amount of error involved in the process and the type of map we can use for display. It is worthwhile, then, to consider in turn each of the ways in which data can be structured.

3.3.1 Vector Data Structures

Vector data structures were the first to be used for computer cartography and GIS because they were simply derived from digitizing tablets, because they are more exact in representing complex features such as land parcels, and because they are easily drawn on pen-type output devices such as plotters. Surprisingly, few people in the early days thought of standardizing how digitizing was to take place, and since there were different technologies, many different formats evolved. The earliest included ASCII files of (x, y) coordinates, but these soon became very unwieldy in size, so binary files rapidly took over.

The first generation of vector files were simply lines, with arbitrary starting and ending points, which duplicated the way a cartographer would draw a map. Obviously the pen would be lifted from the paper to start a new line, but it could be lifted anywhere else. The file could consist of a few long lines, many short lines, or even a mix of the two. Typically, the files were written in binary or ASCII and used a flag or code coordinate to signify the end of a line.

, and some use very clever
iger, or even a spreadsheet
ced these files, which hold

ne *database dictionary*, the
must also be written into
t usually the dictionary is
n. So the attribute database
sts of a file. At the most
nanagement point of view,
the maps are a little harder

ave latitude and longitude,
-down (y) directions. They
can be points, lines, areas,
and you could easily argue
than to draw them. This is
tandard database. Line and
ferent shapes and sizes. A
of points, and these would

In Section 3.1, we met the
ile a GIS can usually deal
l for retrieval and analysis
se can influence many GIS
nt of error involved in the
rthwhile, then, to consider

rtography and GIS because
ey are more exact in rep-
e they are easily drawn on
people in the early days
nd since there were differ-
est included ASCII files of
size, so binary files rapidly

with arbitrary starting and
ld draw a map. Obviously
it could be lifted anywhere
s, or even a mix of the two.
l a flag or code coordinate

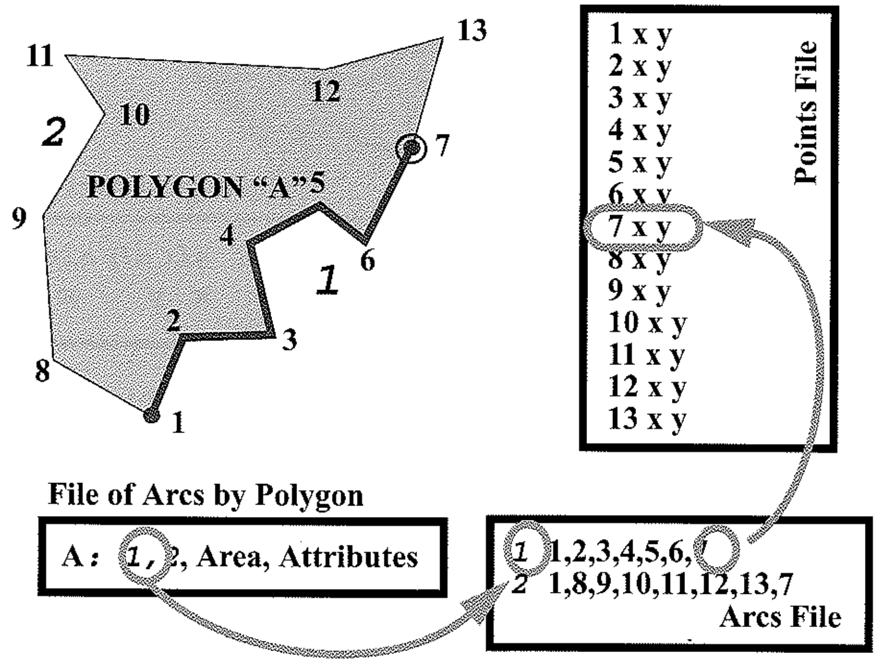
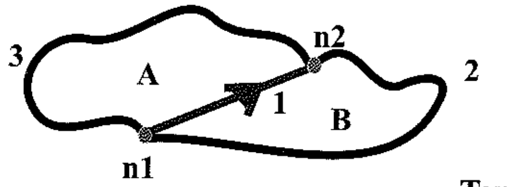


FIGURE 3.4: Arc/node map data structure with files.

As a computer programmer, having to follow the line from one place to another in the file was compared by early programmer Nick Chrisman to following the path of a single strand of spaghetti through a pile of spaghetti on a plate. The name stuck, and to this day unstructured vector data are called *cartographic spaghetti*. A surprising number of systems still use this basic structure, however, and the structure survives in many data formats, such as the Defense Mapping Agency's standard linear format. Most systems allow users to import data in this structure, but almost all now convert it to topological data after entry.

Just as the hierarchical system had caught on as a way of organizing attribute databases, starting in the 1960s a hierarchy for spatial data was worked out that became the arc/node model. Many first-generation systems, including POLYVRT, GIRAS, and ODYSSEY, used this system. This data structure uses the fact that each type of feature—point, line, and area—consists of features with the next fewer dimensions. So area features consist of connected lines, and lines consist of connected points. The most important advantage that this buys us is that we can have separate files for areas, lines, and points. The price is that we need to keep track of links between the files in a fairly arbitrary way. For example, in Figure 3.4, a single polygon consists of two lines or arcs, each with a set of points. The points are ordered from node to node in a sequence.

At the least, we need a file containing the attributes for the polygon, a file listing the arcs within the polygon, and, finally, a file of coordinates that are referenced by the arc file. Figure 3.4 shows that we need to store in each of these files a set of references between the files. For example, an entry in the arcs files states that to get the points for



Topological Arcs File

Arc	*From*	*To*	*PL*	*PR*	*n1x*	*n1y*	*n2x*	*n2y*
1	n1	n2	A	B	x	y	x	y

FIGURE 3.5: A topological structure for the arcs.

arc 2 from the points file they begin at the first coordinate point in the file, followed by the eighth, the ninth, and so on.

During the early days of GIS, several systems evolved different versions of this structure. Obviously, to save space we could write the files in binary. There are few ways, however, to store point, line, and area data that are as efficient. As long as the data are valid, this is a very powerful way to store data for map features. When the system breaks down, however, is when data contain errors, which is virtually always.

A new generation of arc/node data structures arrived after the First International Advanced Study Symposium on Topological Data Structures for Geographic Information Systems, held in 1979. This elegant new structure used the arc as the basis for data storage and relied on reconstructing a polygon when it was needed. The way that this was accomplished turned out to have other practical benefits, as we discuss in Section 3.4. The system stored point data as before, but included in the file of arcs linked to the points file was an abbreviated "skeleton" of the arc (Figure 3.5). This consisted of just the first and last points in the arc, called *end nodes*, and information that related not to this particular arc but to its neighbors in geographic space. This included the arc number of the next connecting arc, and the polygon number of which polygon lay to the left and right of the arc. If the line was just a river or a road, this information was not essential. If, however, the arc was part of a network that formed enclosed areas or polygons, the polygon identifier number became the key to polygon construction.

The way that a polygon could be built was by extracting all of the arcs that a specific polygon had as a neighbor. If the polygon is the right-hand neighbor of each of the arcs, the end nodes can be tested against each other to see which sequence they should be drawn in. The use of the arc as the basic unit meant that when a map was digitized, the user only needed to trace each arc once, instead of twice if each area was traced around the edge. As exactly the same arc was used in both cases, the type of error known as a *sliver* was avoided completely.

One problem with the vector data structure was that it did not really deal very well with geographical surfaces such as topography or air temperature. This was corrected by a research team that devised a new data structure called the *triangulated irregular network* or TIN (Figure 3.6). The TIN is really just a list of points with their coordinates; stored with the points is a file containing information about the topology of a network.

cal Arcs File

c n1y n2x n2y
y x y

arcs.

oint in the file, followed by

ed different versions of this
es in binary. There are few
efficient. As long as the data
features. When the system
s virtually always.
after the First International
es for Geographic Informa-
ed the arc as the basis for
t was needed. The way that
benefits, as we discuss in
included in the file of arcs
the arc (Figure 3.5). This
end nodes, and information
in geographic space. This
e polygon number of which
just a river or a road, this
t of a network that formed
became the key to polygon

cting all of the arcs that a
ight-hand neighbor of each
to see which sequence they
neant that when a map was
ad of twice if each area was
both cases, the type of error

did not really deal very well
erature. This was corrected
d the *triangulated irregular*
oints with their coordinates;
the topology of a network.

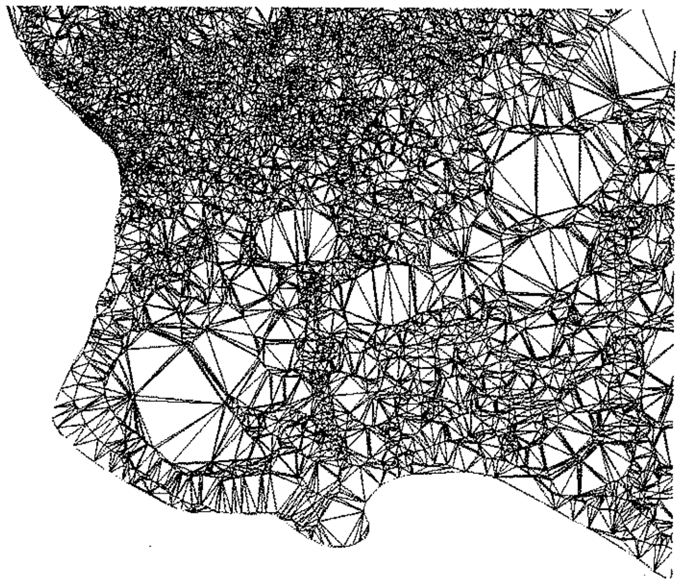


FIGURE 3.6: A triangulated irregular network (TIN) covering metropolitan Lisbon in Portugal. (Courtesy of Elisabete de Silva.)

The network is a set of triangles, constructed by connecting the points in a network of triangles called a *Delaunay triangulation*. This way of drawing triangles is optimal, because changing any one triangle makes the angles within the triangle less similar to each other.

Two sorts of TIN can be built, one with a file containing information about the arcs that connect points, and one containing all the data about one triangle. The TIN became popular as a way of storing topography or land elevation data for visualization and engineering. With a TIN it is easy to draw contours, make a three-dimensional view of an area, estimate how water would run downhill over a digital landscape, or calculate how much material would have to be moved in a construction project. Many GIS programs that work with computer-aided drafting (CAD) systems or with surveying software use TIN as their data structure. The TIN has proven to be both efficient in storing data and versatile in finding new uses within GIS.

3.3.2 Raster Data Structures

Raster or grid data structures have formed the basis for many GIS packages. The grid is a surprisingly versatile way of storing data. The data form an array or matrix of rows and columns. Each pixel or grid cell contains either a data value for an attribute, or an index number that points to a reference in the attribute database. So a pixel containing the number 42, for example, could correspond with the number 42 or "deciduous forest" in the Anderson Level II system, or just the 42nd record in the attribute file.

To write the numbers to a file, we can just start the file with any necessary attribute codes, perhaps the number of rows and columns and the maximum size of one value, and then write the data into the file in binary across all columns for all rows, one long stream of data with a start and an end, like an unraveled sweater. When reading the data back in, we just place the data back into a raster grid of the correct dimensions.

A major advantage of the raster system is that the data form their own map in the computer's memory. An operation such as comparing a grid cell with its neighbors can be performed by looking at the values in the next and preceding row and column of the grid cells in question. However, the raster is not very good at representing lines or points, since each becomes a set of cells in the grid. Lines can become disconnected or "fat" if they cross the grid at too shallow an angle. However, variables that needed the TIN in vector data structures fit easily into the raster. This is particularly suited to data that come from remote sensing or scanning.

One major problem with raster data is the mixed pixel problem. Figure 3.7 provides an example. The photographs show a part of the outline of a lake in an oblique view. In the top picture, there is only one type of land cover, "grass," so all pixels belong in one class. In the bottom picture, there are two types of attributes, water and grass. Even though a wet foot is a sure indicator of a water grid cell, it is hard to assign each pixel to one or the other category. A compromise, and one often used in GIS, is to assign

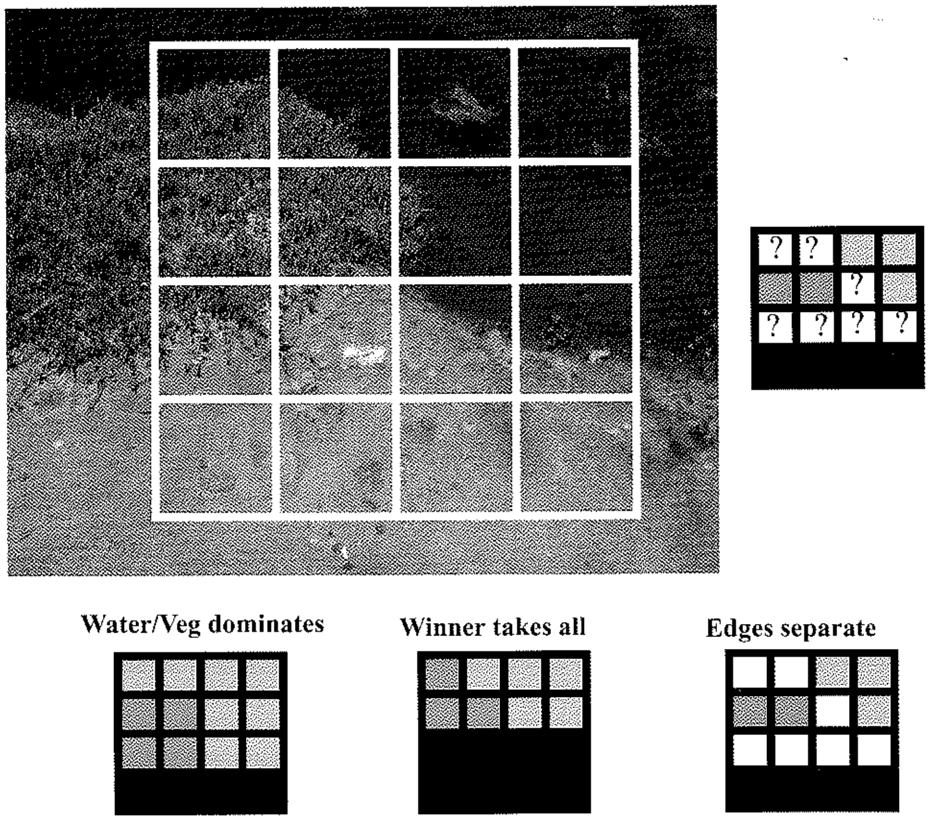
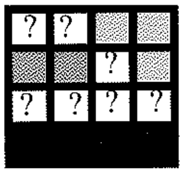


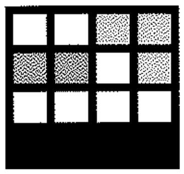
FIGURE 3.7: The mixed pixel problem. Domination can be applied in sequence, the area covering most of the pixel assigned the pixel, or edges can be assigned a separate value. Any of the three options is acceptable, or any other, as long as the rule is applied consistently. (Water = Cyan, Vegetation = Green, Bare Ground = Black).

raster grid of the correct

form their own map in the cell with its neighbors can eding row and column of od at representing lines or n become disconnected or , variables that needed the particularly suited to data

oblem. Figure 3.7 provides a lake in an oblique view. ss," so all pixels belong in ites, water and grass. Even s hard to assign each pixel used in GIS, is to assign





ence, the area covering most of the three options is acceptable, or any = Green, Bare Ground = Black).

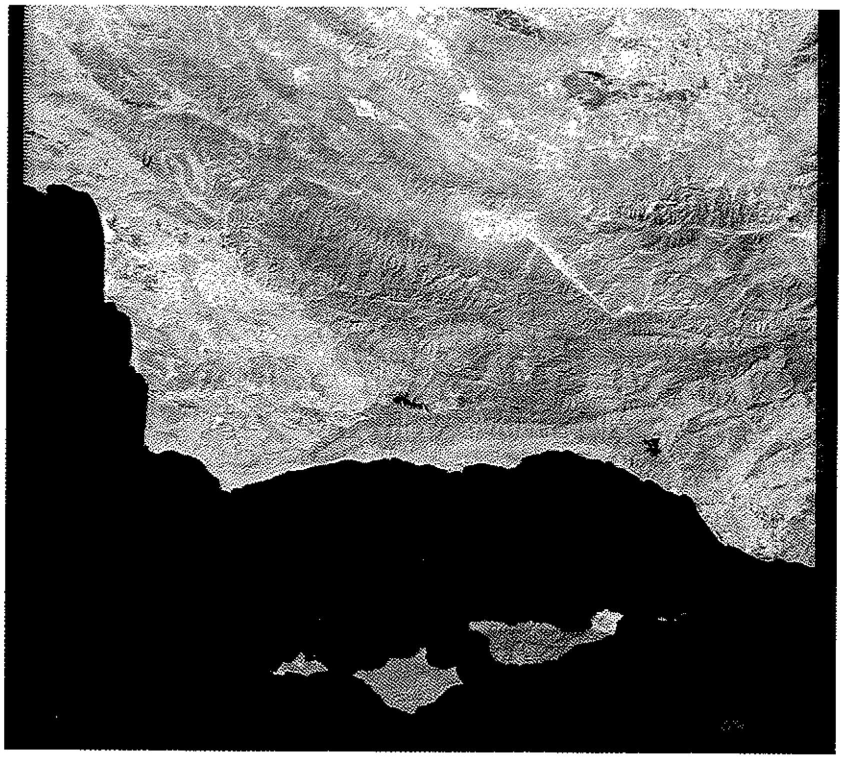


FIGURE 3.8: March 2002 Landsat 7 image of the Santa Barbara channel, CA. This data layer is a grid with a large section of "missing data," in this case, the zeros in the ocean, and the unregistered image edges.

edge pixels, those that are not exclusively in one class or another. Finally, when several classes are involved, we either have to make up rules for assignment, such as assigning a mixed pixel to the class that occupies the most area within it, or we have to put up with edges and mixed pixels. Even when the boundary is absolutely clear, a vector data representation may be better than the raster.

At least two ways have been devised within GISs to deal with the problem that a grid often contains redundant or missing data (Figure 3.8). The first of these is a compression mechanism called *run-length encoding*. Along each row, only changes between attributes and the numbers of pixels of that same attribute are stored. If a whole row is all one class, it is stored as the class and the number of pixels only, quite a saving in space. As the raster becomes more and more varied, however, fewer and fewer savings can be made over the original grid. Many GIS packages and many industry-standard image formats use run-length encoding.

Another way to save space is to use a data structure called a *quad tree*. A quad tree works by dividing a grid into four quadrants, saving a reference to the quadrant of the grid only if it contains data. Then the quadrant is split into four half-size quadrants, and so on until the individual pixel is reached. If we split a quadrant and all the pixels in it have the same attribute, then obviously we would not need to store anything else for that entire quadrant. Each attribute becomes a list of quadrants needed to get to the

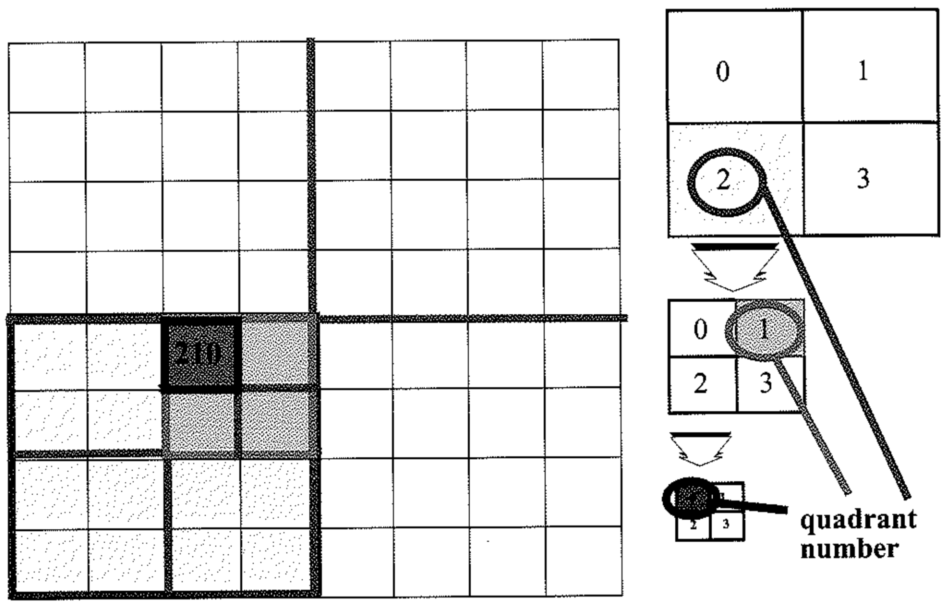


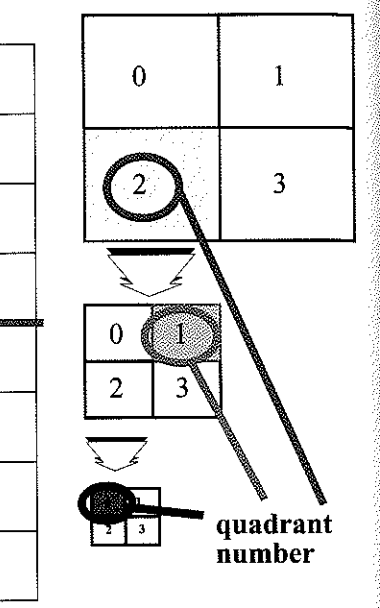
FIGURE 3.9: The quad-tree structure. Reference to code 210.

area, like a coordinate reference (Figure 3.9). Quad trees have been used more in image processing than in GIS, but at least one GIS package, SPANS, uses them as its primary data structure.

3.4 WHY TOPOLOGY MATTERS

When topological data structures became widespread in GIS, some significant benefits resulted, enough that today the vector arc/node data structure with topology probably is the most widespread for GIS data. Typically, a GIS maintains the arc as the basic unit, storing with it the polygon left and right, the forward and reverse arc linkages, and the arc end nodes for testing. This means that each line is stored only once and that the only duplication is the endpoints. The disadvantage is that whenever areas or polygons are to be used, some recomputing is necessary. Most programs save the result, however, such as the computed polygon areas, so that recalculation is unnecessary.

Topology allowed GIS for the first time to do error detection. If a set of polygons is fully connected, and there are no gaps at nodes or breaks in the lines defining the areas, the set of areas is called *topologically clean*. When maps are first digitized, however, this is rarely the case. The topology can be used to check the polygons. Polygon interiors are usually identified by digitizing a point inside a polygon, a label point, and by keeping track of the arcs as they are entered. A polygon gets the label from the label point when the point is found to be inside the polygon. A GIS will have the ability to build the topology from the unconnected arcs. First, each endpoint is examined to see if it is "close" to another. If it is, the points are "snapped" together; that is, their (x, y) coordinates are averaged and each is replaced with exactly the same values (Figure 3.10).



erence to code 210.

ees have been used more in image
SPANS, uses them as its primary

d in GIS, some significant bene-
lata structure with topology prob-
, a GIS maintains the arc as the
the forward and reverse arc link-
that each line is stored only once
sadvantage is that whenever areas
necessary. Most programs save
n areas, so that recalculation is

error detection. If a set of poly-
les or breaks in the lines defining
m. When maps are first digitized,
used to check the polygons. Poly-
nt inside a polygon, a label point,
d. A polygon gets the label from
the polygon. A GIS will have the
rcs. First, each endpoint is exam-
nts are "snapped" together; that is,
aced with exactly the same values

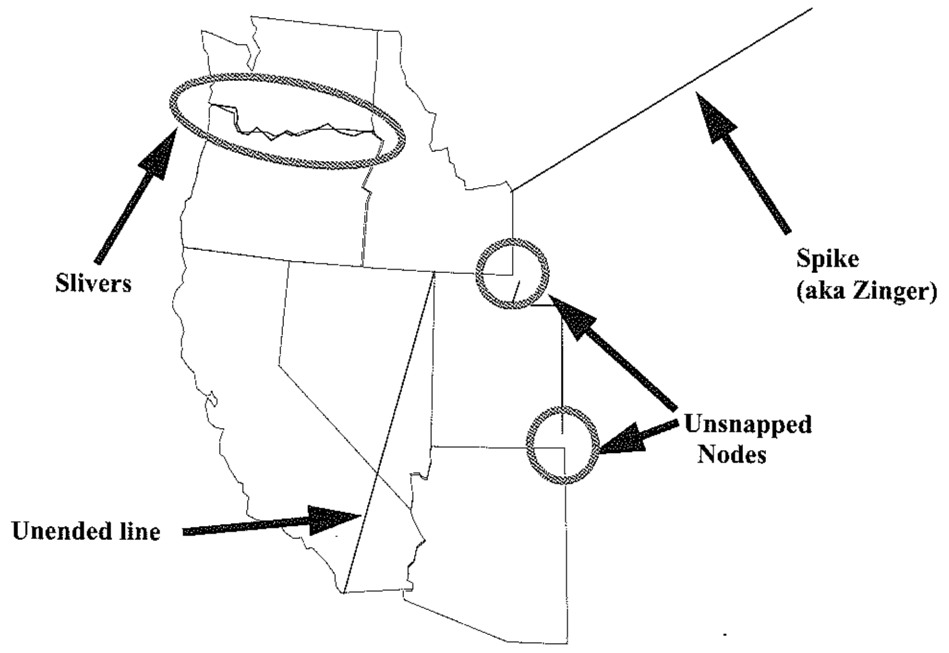


FIGURE 3.10: Example of slivers (unmatched nodes along two lines), unsnapped nodes (endpoints of two lines that should be the same point), spikes (erroneous coordinates), and unended lines.

In addition, arcs that connect the same nodes are tested to see if they are duplicates, and the user is asked which to delete. Any small areas, probably the result of errors called *slivers,* caused by double digitizing, are also eliminated. So, an error automatically detected can become an error automatically eliminated. Obviously, the separation between nodes required before they are treated separately and the size that a polygon must reach before it is retained are critical values for the map. Sometimes called *fuzzy tolerances*, these values should be handled with caution, because they allow the map's features to move around. Short lines, small polygons, or precisely measured point locations have critical significance and should not be deleted by automatic testing for topological completeness. For example, a map of Europe should include Andorra, Monaco, and Liechtenstein.

The primary advantage of having a topologically consistent map is that when two or more maps must be overlain, much of the initial preparation work has been done. What still has to be established are where new points must be added along lines to become nodes, and how to deal with any small or sliver polygons that are created (Figure 3.10). The latter can be a real problem. Many borders between regions, states, counties, and so on match along lines such as rivers, which are generalized differently at different map scales. Although the line should be the same, in fact it is not. Some packages allow the extraction of a line from one map to be "frozen" for use on another. This seemingly small difference can be very significant, especially if areas or densities are being calculated.

The final advantage to topology is that many operations of retrieval and analysis can be conducted without having to continuously deal with the (x, y) data. In some cases,

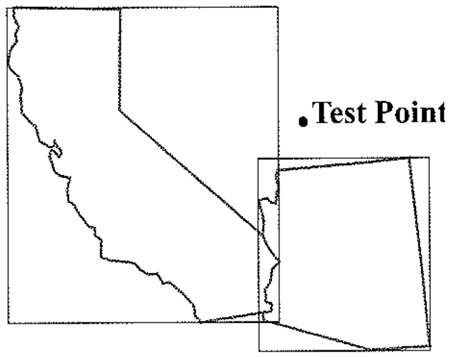


FIGURE 3.11: Bounding rectangles: rectangles that contain a polygon completely. If a test point lies wholly outside the bounding rectangle, it must be outside the enclosed polygon.

pretesting can be done. For example, say that a point is to be tested to see if it falls inside an area. If the point falls outside the bounding rectangle of all the endpoints, the point is most likely outside the region (Figure 3.11). So useful are these bounding rectangles, the highest and lowest *x* and *y* values along an arc, that they are often computed once and saved with the topological information in the arcs file.

3.5 FORMATS FOR GIS DATA

With a 30-year history and with so many alternative ways to structure map and attribute data, it is hardly surprising that most GISs use radically different approaches to handling their content. The data structures used are often invisible as far as the GIS user is concerned. We might not even need to understand exactly what is happening when two maps are overlain. However, if we are to be objective, scientific GIS users, at the very least we must have a full understanding of the errors and transformations involved. Regardless of how a GIS structures its maps as numbers, it must be able to import data from other GIS packages and from the most common data sources, as well as scanned and digitized data, and to convert the result into its own internal format. In some cases this is an open process. Some GIS companies have published and documented their internal or exchange data formats, including Intergraph and Autodesk. Others protect their internal data as a trade secret, in the hope of being able to sell data and data converters as well as their GIS.

The most common data formats for GIS data have been used by so many GIS operations and for so much existing data that a GIS ignores them at its peril. Some are so common that utility programs and even operating systems read, process, and display these formats automatically. These formats include some that have arisen because they are a common data format, such as TIGER and DLG. Others are industry-standard formats, proprietary formats that have been used so much that they are documented and published, although their use may have restrictions.

In the GIS world, a small subset of these formats has become commonplace, and we cover them here for completeness. We then finish the chapter by discussing some of the issues of data exchange between GIS systems and take a look at the accepted national standard, the Spatial Data Transfer Standard (SDTS).

nt

pletely. If a test point lies wholly

ested to see if it falls inside
all the endpoints, the point
these bounding rectangles,
y are often computed once

structure map and attribute
ent approaches to handling
far as the GIS user is con-
happening when two maps
GIS users, at the very least
mations involved. Regard-
e able to import data from
es, as well as scanned and
ormat. In some cases this is
ocumented their internal or
Others protect their internal
and data converters as well

een used by so many GIS
them at its peril. Some are
read, process, and display
have arisen because they are
industry-standard formats,
documented and published,

become commonplace, and
pter by discussing some of
ook at the accepted national

3.5.1 Vector Data Formats

A general distinction between industry and commonly used standards for GIS data is that between formats that preserve and use the actual ground coordinates of the data and those that use an alternative *page coordinate* description of the map. The latter are the coordinates used when a map is being drafted for display in a computer mapping program or in the data display module of a GIS (Figure 3.12).

HPGL

```
IN;IP0 0 8636 11176;
SC-4317 4317 -5586 5586;
SP1;
SC-4249 4249 -5498 5498;

SP1;
PU-2743 847;PD -2743 3132 608 3132
608 847 -2743 847;
```

PostScript

```
%%BeginSetup
11.4737 setmiterlimit
1.00 setflat
/$fst 128 def

%%EndSetup
@sv
/$ctm matrix currentmatrix def
@sv
%%Note: Object
108.58 456.98 349.85 621.50 @E
0 J 0 j [] 0 d 0 R 0 @G
0.00 0.00 0.00 1.00 K
0 0.22 0.22 0.00 @w
 0 0 0 @g
0.00 0.00 0.00 0.00 k
%%RECT 241.272 -164.520 0.000
108.58 621.50 m
349.85 621.50 L
349.85 456.98 L
108.58 456.98 L
108.58 621.50 L
@c
B
@rs
@rs
%%Trailer
end
```

AutoCAD DXF

```
POLYLINE
  8
7
  6
CONTINUOUS
 66
  1
  0
VERTEX
  8
7
 10
-2.742
 20
3.132
  0
VERTEX
  8
7
 10
0.608
 20
3.132
  0
VERTEX
  8
7
 10
0.608
 20
0.848
  0
VERTEX
  8
7
 10
-2.742
 20
0.848
  0
VERTEX
  8
7
 10
-2.742
 20
3.132
  0
SEQEND
  0
ENDSEC
  0
EOF
```

FIGURE 3.12: Some alternative industry-standard vector formats. Headers have been removed. Graphic is the same four-point rectangle in each case.

The Hewlett-Packard Graphics Language (HPGL) is a page description language designed for use with plotters and printers. The format is simple and the files are plain ASCII text. Each line of the file contains one move command, so a line segment connects two successive lines or points. The format works with a minimum of header information, so that files can be written or edited easily. However, the header can be manipulated to change the scaling, size, colors, and so on. The HPGL is an unstructured format and does not store or use topology.

Another industry-standard format is the PostScript page definition language, developed by the Adobe Corporation for use in its desktop and professional publishing products and now in widespread use. So common is this format that most laser-quality printers use it as the printer device control format. PostScript, at least in its vector mode, is a page description language. This means that its coordinates are given with respect to a printed page, say an 8-1/2-by-11-inch sheet.

PostScript uses ASCII files but has particularly complex headers controlling a very large number of functions, such as fonts, patterns, and scaling. PostScript is really a sort of programming language, and to be viewed it must be interpreted. There are many commercial and shareware packages for viewing PostScript files, and many word processors and graphics packages will both read and write the format, although many more write it than read it. In GIS, PostScript is usually used to export or print a finished map rather than data as such.

The popular CAD package AutoCAD by Autodesk has made commonplace the AutoCAD digital exchange format (DXF) for drawing data. The AutoCAD Map GIS software also makes use of these formats. While Autocad uses its own internal format for data storage, it uses an external format called DXF for transfer of the files between computers and between packages. Again, these are simple ASCII files (although there is a binary mode), but in the DXF case there is a very large and mandatory file header containing significant amounts of metadata and file default information.

Although DXF does not support topology, it does allow the user to maintain information in separate layers, a familiar GIS concept. There is considerable support for details of drawings, line widths and styles, colors, and text, for example. DXF is importable by almost all GIS packages other than those that use raster formats. Some GIS packages work directly with the Autocad or other CAD software and can manage these files internally.

Two formats have become widespread largely because so many important data have been made available using them: the DLG and the TIGER formats. The digital line graph (DLG) format of the U.S. Geological Survey's (USGS) National Mapping Division are available at two scales, the scales of the map series from which they were captured (Figure 3.13). These scales are 1 : 100,000, for which almost all of the country has some data available, and 1 : 24,000, with only a small portion of the country but in extreme detail.

The formats of the data are documented formally, and the files are ASCII. They use the ground coordinates in UTM, truncated to the nearest 10 meters to reflect their locational precision and to save space (Figure 3.14). Features are handled in separate files—for example, hydrology, hypsography (contours and topographic features), transportation, and political. Many GIS packages will import these files, but often some extra data manipulation is necessary, such as making the records of some fixed line length in bytes.

ge description language
le and the files are plain
a line segment connects
m of header information,
der can be manipulated
unstructured format and

efinition language, devel-
ional publishing products
ost laser-quality printers
in its vector mode, is a
given with respect to a

x headers controlling a
ling. PostScript is really
e interpreted. There are
ipt files, and many word
format, although many
export or print a finished

made commonplace the
The AutoCAD Map GIS
s its own internal format
nsfer of the files between
CII files (although there
nd mandatory file header
rmation.
he user to maintain infor-
derable support for details
mple. DXF is importable
ormats. Some GIS pack-
nd can manage these files

so many important data
formats. The digital line
ational Mapping Division
which they were captured
l of the country has some
e country but in extreme

he files are ASCII. They
10 meters to reflect their
s are handled in separate
ographic features), trans-
files, but often some extra
of some fixed line length

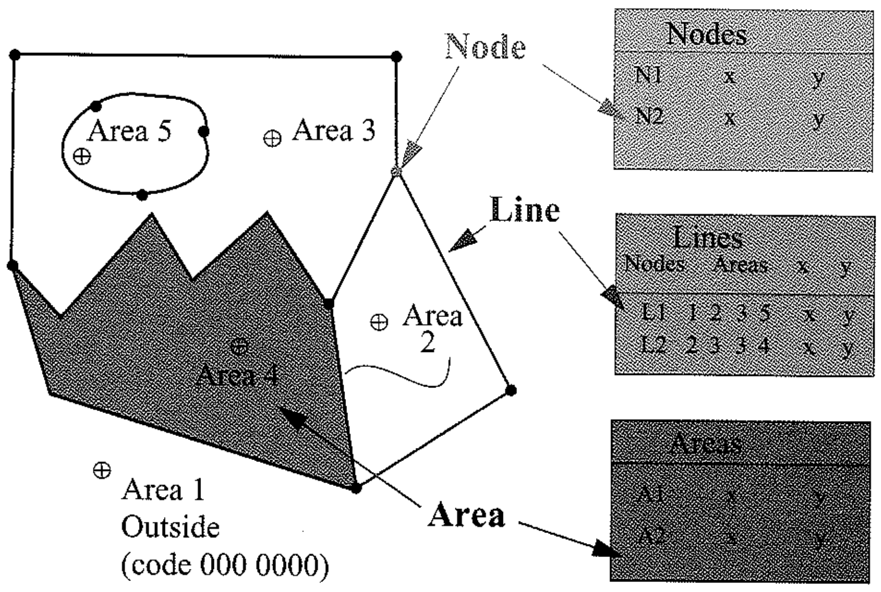


FIGURE 3.13: Sample digital line graph coding format.

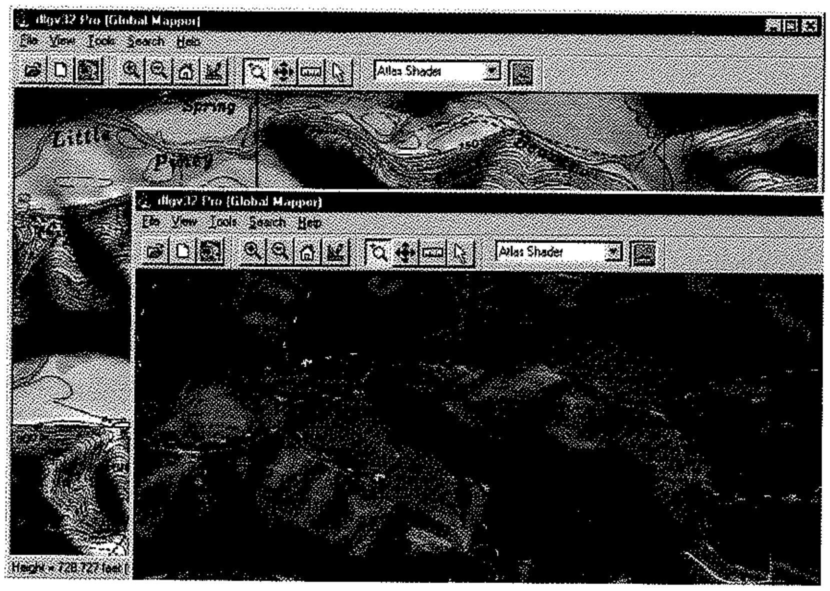


FIGURE 3.14: Sample DLG obtained from the USGS and displayed with the dlgv32 DLG viewer software.

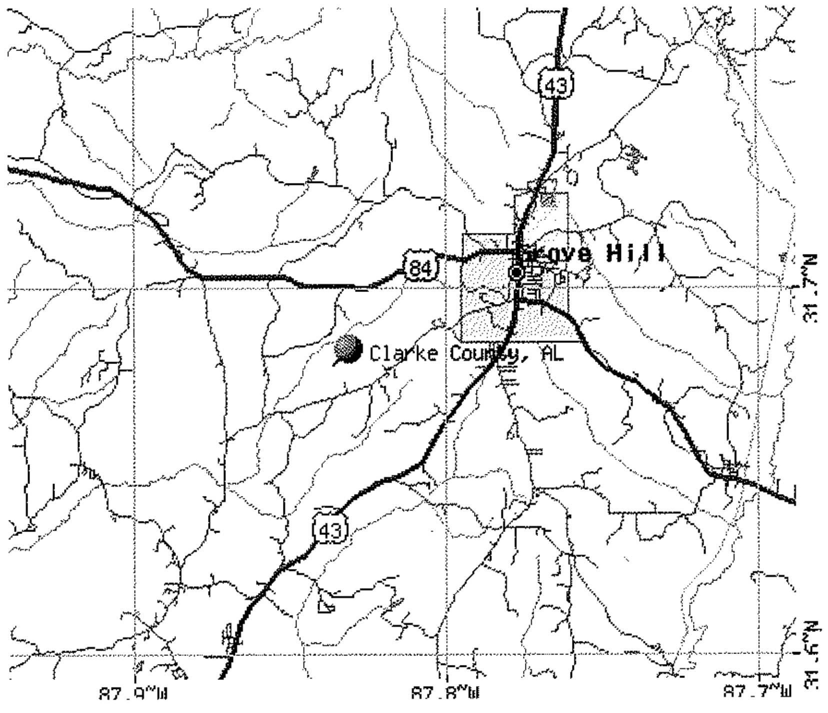
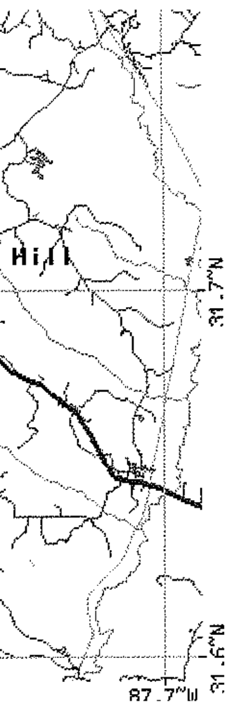


FIGURE 3.15: 2000 Census TIGER files plotted for part of Clarke County, Alabama.

The TIGER formats are from the enumeration maps of the U.S. Census Bureau, as used for the decennial census (Figure 3.15). They are vector files and contain topology; in fact, their predecessor the GBF/DIME files were highly instrumental in popularizing topological data structures. They consist of an arc/node type arrangement, with separate files for points, lines, and areas linked together by cross references. The TIGER terminology calls points *zero cells*, lines *one cells*, and areas *two cells*. The cross indexing means that some features can be encoded as landmarks, and these include rivers, roads, permanent buildings and so forth, which allow GIS layers to be tied together. The TIGER files exist for the entire United States, including Puerto Rico, the Virgin Islands, and Guam. They are block level maps of every village, town, and city, and include geocoded block faces with address ranges of street numbers (Figure 3.16). This means that the address matching function is possible, and a large proportion of GIS use depends highly upon this capability. The data are also obviously referenced to the U.S. Census, so that many population, ethnicity, housing, economic, and other data can also be used with TIGER.

Furthermore, the whole country is available on CD-ROM at minimal cost and over the Internet. Most GIS vendors, and some independents, offer updated and enhanced TIGER files as their own products. Although topologically correct, TIGER has been criticized as not being particularly geographically correct. More recent GIS functions have made the addition of higher levels of geographic accuracy possible, and many data suppliers have enhanced the TIGER files.



arke County, Alabama.

the U.S. Census Bureau, as
files and contain topology;
nstrumental in popularizing
pe arrangement, with sepa-
oss references. The TIGER
two cells. The cross index-
s, and these include rivers,
yers to be tied together. The
rto Rico, the Virgin Islands,
own, and city, and include
s (Figure 3.16). This means
oportion of GIS use depends
erenced to the U.S. Census,
other data can also be used

M at minimal cost and over
ffer updated and enhanced
y correct, TIGER has been
More recent GIS functions
acy possible, and many data

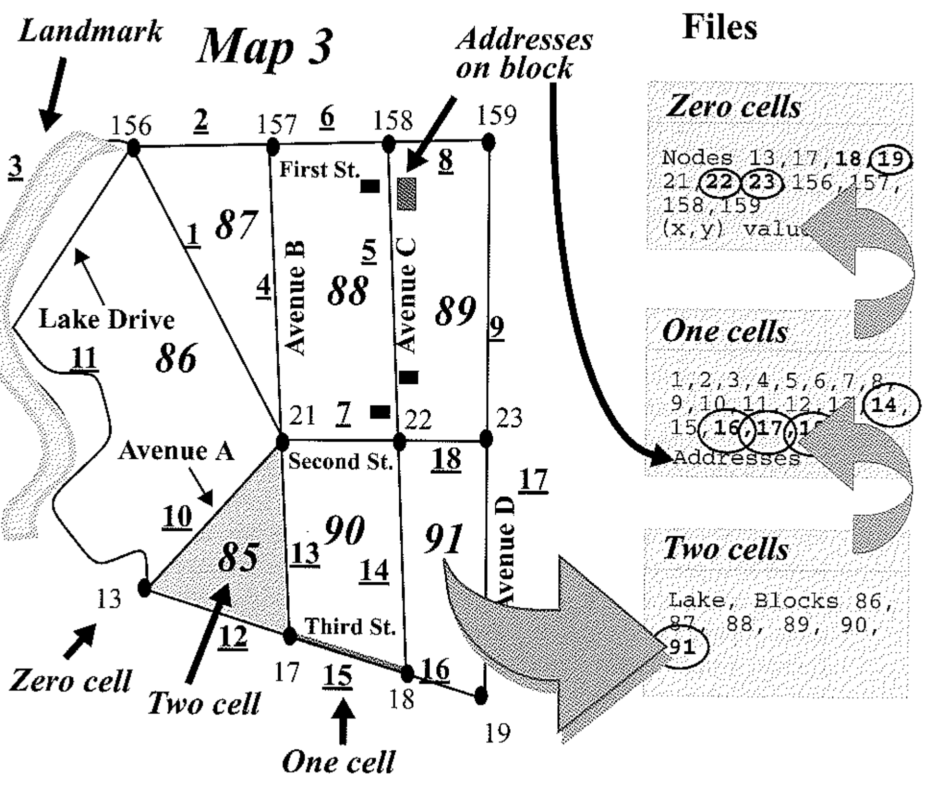


FIGURE 3.16: The U.S. Census Bureau's TIGER data structure.

3.5.2 Raster Data Formats

Raster data formats have been much more widely used, especially since the arrival of networking, because many of them are the same formats that are used to store digital images and pictures. Image formats are particularly simple to create, and as a result there are many. Some of the formats have been optimized for passing the images through networks, and it is these formats that are now most common. Raster formats are mostly similar in how the files are arranged. Few of the formats use ASCII; most use binary. Usually the file structure is a header with a fixed length and a keyword or "magic number" to identify the format. Included in the header is at least the length of one record in bits (called the *image depth*), the number of rows and the number of columns in the file.

Optionally, the file contains a color table. The color table allows the data file to consist of indices, say the numbers 1, 2, 3, 4 and so forth, and each index to correspond to a value, usually three values in each of three bytes as intensities between 0 and 255 in the red, green, and blue, (RGB) respectively. For example, if an image file were to consist of only four colors, red, green, blue, and white, there is no need to store values in each pixel like (255,255,255) for white's RGB value. Instead, the color table could assign white to 0, red to 1, green to 2, and blue to 3. Now no pixel need be more than two bits in size (2 bits can store decimal 0 through 3), as opposed to 24 bits for the full colors. This saves a significant amount of space, since the record size is multiplied up for all rows times all columns.

The final part of the image file is the data, usually all columns for each row. Some formats are padded at the end so that the total number of bytes is a multiple of a key factor, such as 512 bytes. For many GIS files, we do not want color but want to store single-band files, using all of the bits for data. Elevations, for example, which may be in the thousands of feet or meters, need more than one byte per pixel. Alternatively, if the pixel data are simple attributes such as land cover type with only a few categories, then the same value can be assigned to each of the red, green, and blue values if the format requires it.

A surprising number of utility programs exist to convert between raster formats. Among them are Image Alchemy and xv. Many packages also read and write a huge number of formats. Some will convert from raster to vector and vice versa, such as CorelDraw! In some cases, this capability is included within the GIS package.

Some common raster formats are the Tagged Interchange Format (TIF), which can use run length and other image compression schemes and has a number of different forms that are publicly available; the Graphics Interchange Format (GIF), popularized by the online network services, especially CompuServe (the developers), which uses a quite sophisticated compression scheme on the data part of the image; and the JPEG format, which uses a variable resolution compression system offering both partial and full resolution recovery depending on the space available.

Finally, the PostScript industry standard includes a specification for images called encapsulated PostScript. This is a very simple image format indeed, which literally encodes the hexadecimal values to be placed into the image inside an image macro within a regular PostScript file. Many PostScript devices and programs will not handle this format, but those with higher amounts of memory and more advanced capabilities can. This is far better for storing images than maps for GIS use, and it is rarely used to store data; rather, it is used to drive printers and plotters to generate GIS maps.

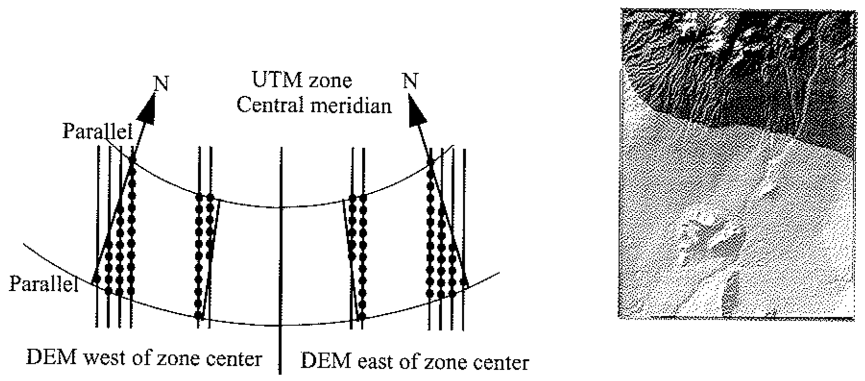


FIGURE 3.17: The 1:24,000, 30-meter DEM format from the USGS. Example on right is the Jackass Flats, Nevada quadrangle, hillshaded. Note the edge effect of the UTM grid angle.

sed to 24 bits for the full
ord size is multiplied up

mns for each row. Some
es is a multiple of a key
t color but want to store
xample, which may be in
ixel. Alternatively, if the
ly a few categories, then
blue values if the format

between raster formats.
o read and write a huge
and vice versa, such as
e GIS package.
Format (TIF), which can
s a number of different
rmat (GIF), popularized
evelopers), which uses a
ne image; and the JPEG
offering both partial and

ication for images called
t indeed, which literally
inside an image macro
programs will not handle
ore advanced capabilities
e, and it is rarely used to
nerate GIS maps.

ble on right is the Jackass Flats,

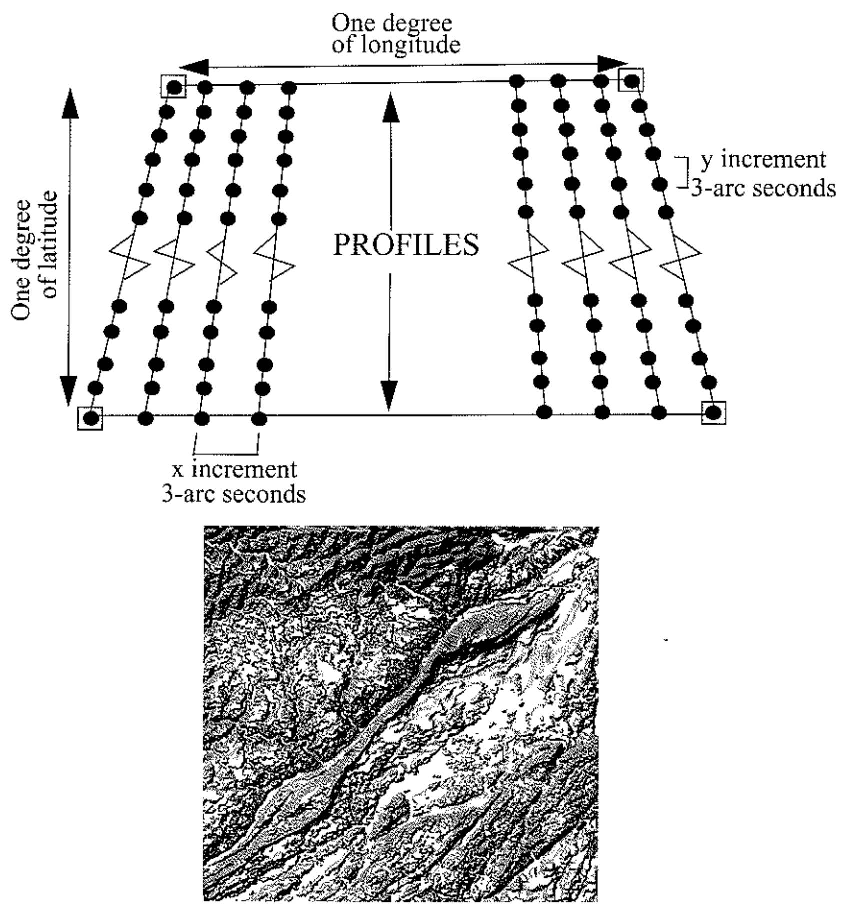


FIGURE 3.18: (Top) The 1 : 250,000, 3-arc second DEM format from the USGS. (Bottom) The Scranton, PA East 1 : 250,000 digital elevation model hill shaded to show topography. Each two-by-one degree quadrangle forms two 1201 by 1201 DEMs.

One raster format has gained widespread acceptance and is often read directly by GIS packages or stand-alone utilities that come with the software. This is the Digital Elevation Model format of the USGS. This format is one in which two types of data are distributed, the 30-meter elevation data from the 1 : 24,000, 7.5-minute quadrangle maps, and the 1 : 250,000, 3-arc second digital terrain data originally supplied by the Defense Mapping Agency but now distributed by the USGS. These are documented formats and are somewhat complex because of the map projections involved. The 30-meter data, for example, are sectioned by quadrangles of latitude and longitude, so in the UTM coordinate base they are stored in have blank sections at the edges where no data are stored. The 3-arc second data are less of a problem to read, but must be projected by the GIS for further use. Some examples of the two data sets and their formats are shown in Figures 3.17 and 3.18.

3.6 EXCHANGING DATA

Exchanging data can be thought of in two ways. First, as we have seen in this chapter, the vector and the raster formats often store similar GIS data in very different ways. The GIS software adopts one of two strategies for dealing with the two types of data. Some systems use only one format exclusively, and provide utilities or import options to bring in and convert the data to the format to be used. Raster-based GIS programs especially use this approach.

Alternatively, the system can support the native format of each type of data, and it can require the GIS operator to explicitly change formats when operations requiring compatability of formats are executed. In both cases, a computer program, part of the GIS, either performs a raster-to-vector or a vector-to-raster conversion. While a full discussion of how these operations work is beyond the scope of this book, it should be clear that going from vector to raster, filling in grid cells as lines cross them or as polygons include them, is relatively simple. The opposite is quite complex (Clarke, 1995).

Raster format data are often output from scanners, when the GIS requires vector data. In many cases, a special suite of software or even a special-purpose computer is used. When done by the GIS, especially for a large data set and for fine resolutions, the process can be very time-consuming. The program must try to follow each line from pixel to pixel, figure out where the end nodes are, and generate a vector equivalent of the line. Often the lines are jagged from the raster effect, and must be smoothed. If the lines are too fat, they sometimes must be thinned first, and this can generate false connections and loops in the lines (Figure 3.19).

The second way to envision data exchange is to consider the issue of transferring data not between formats, but between entirely different computer systems potentially using different GIS packages. This situation is quite ordinary. Different local authorities may have developed their GIS operations around different software. Different projects may have delivered GIS project data in a huge variety of formats. There is also a real need for data exchange. At state or local boundaries, for example, data should be able to be matched for continuity across the borders, just as the data should match from map sheet to map sheet.

The history of GIS has ensured that this commonality and sharing rarely have taken place. Even state and city GIS efforts have often had contradictory or even competing GIS data, and more than a few projects have found it easier to start digitizing and data assembly all over again rather than convert GIS data from an exchange-unfriendly data

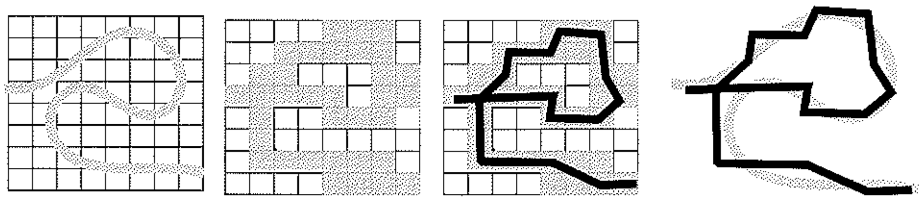


FIGURE 3.19: Errors caused by exchanging data between raster and vector formats. The original (cyan) river after raster-to-vector conversion appears to connect the loop back.

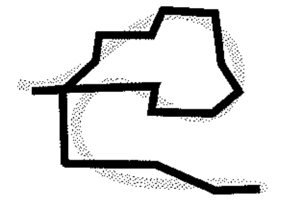
have seen in this chapter,
ı very different ways. The
: two types of data. Some
or import options to bring
l GIS programs especially

of each type of data, and
; when operations requir-
ı computer program, part
-raster conversion. While
he scope of this book, it
grid cells as lines cross
opposite is quite complex

n the GIS requires vector
ecial-purpose computer is
nd for fine resolutions, the
to follow each line from
: a vector equivalent of the
t be smoothed. If the lines
generate false connections

er the issue of transferring
nputer systems potentially
Different local authorities
oftware. Different projects
rmats. There is also a real
mple, data should be able
ta should match from map

d sharing rarely have taken
dictory or even competing
to start digitizing and data
ı exchange-unfriendly data

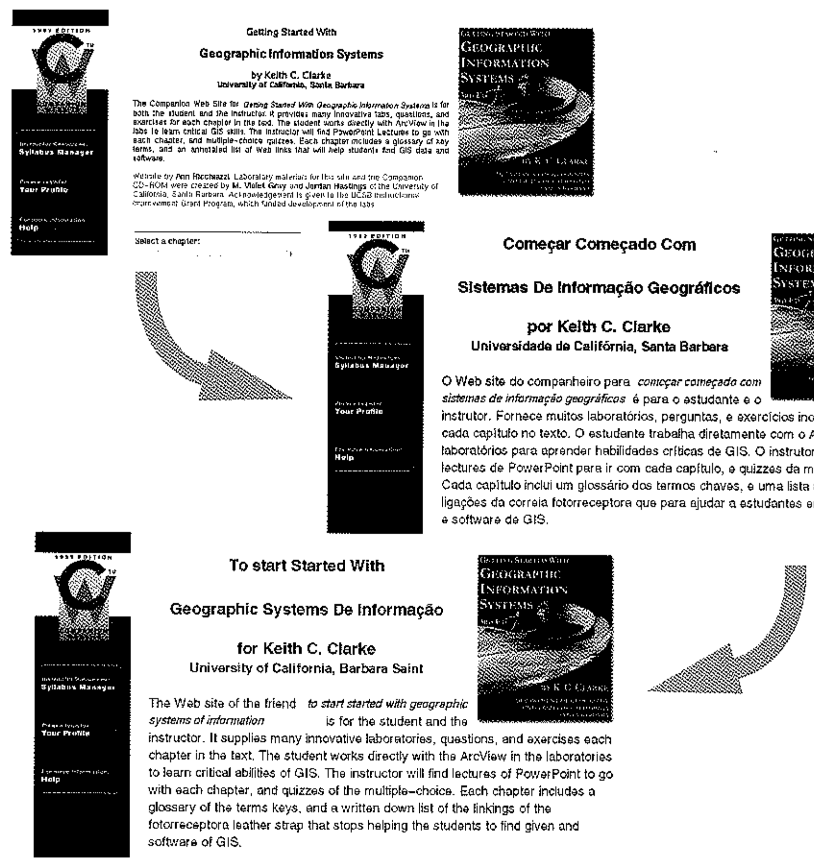


formats. The original (cyan) river

set. Added to this has been the marketing philosophy of many GIS vendors, which has left data formats as proprietary in nature, undocumented and therefore unable to be used as import/export modules for other packages. In the past, data exchange could only be characterized as haphazard and chaotic.

Not exchanging or reusing data between projects, especially within a single organization, is a good example of duplication and waste. There is a real advantage in starting from a single, standard data set for all development or enhancement, especially when GIS operations will eventually bring the data back together for analysis and display. Some industry standards have been quite useful for data exchange, as we saw in Section 3.5. However, two aspects remain a problem. First, none of the industry standards exchange topology with the data, transferring instead only the graphic information. Second, with many different formats, each package has to include a large number of format translators.

A parallel exists between GIS data formats and spoken languages (Figure 3.20). We can get by in isolation by knowing English alone, or perhaps we learn a little French.



Getting Started With

Geographic Information Systems

by Keith C. Clarke
University of California, Santa Barbara

The Companion Web Site for *Getting Started With Geographic Information Systems* is for both the student and the instructor. It provides many innovative labs, questions, and exercises for each chapter in the text. The student works directly with ArcView in the labs to learn critical GIS skills. The instructor will find PowerPoint Lectures to go with each chapter, and multiple-choice quizzes. Each chapter includes a glossary of key terms, and an annotated list of Web links that will help students find GIS data and software.

Website by Ann Fiocchiazzi. Laboratory materials for this site and the Companion CD-ROM were created by M. Violet Gray and Jordan Hastings of the University of California, Santa Barbara. Acknowledgement is given to the UCSB Instructional Improvement Grant Program, which funded development of the labs

Select a chapter:

Começar Começado Com

Sistemas De Informação Geográficos

por Keith C. Clarke
Universidade de Califórnia, Santa Barbara

O Web site do companheiro para *começar começado com sistemas de informação geográficos* é para o estudante e o instrutor. Fornece muitos laboratórios, perguntas, e exercícios inovativos para cada capítulo no texto. O estudante trabalha diretamente com o ArcView nos laboratórios para aprender habilidades críticas de GIS. O instrutor encontrará lectures de PowerPoint para ir com cada capítulo, e quizzes da múltiplo-escolha. Cada capítulo inclui um glossário dos termos chaves, e uma lista anotada das ligações da correia fotorreceptore que para ajudar a estudantes encontrar dados e software de GIS.

To start Started With

Geographic Systems De Informação

for Keith C. Clarke
University of California, Barbara Saint

The Web site of the friend *to start started with geographic systems of information* is for the student and the instructor. It supplies many innovative laboratories, questions, and exercises each chapter in the text. The student works directly with the ArcView in the laboratories to learn critical abilities of GIS. The instructor will find lectures of PowerPoint to go with each chapter, and quizzes of the multiple-choice. Each chapter includes a glossary of the terms keys, and a written down list of the linkings of the fotorreceptora leather strap that stops helping the students to find given and software of GIS.

FIGURE 3.20: The Multiple Translators Problem. The web page for this book translated into Portuguese by Altavista's Babelfish (`http://babelfish.altavista.com`), then back into English. Unless a data translation or spatial operation is error free, the same thing happens with GIS data as they move through operations and across systems.

If we wish to speak to someone who speaks only Russian, however, we need someone who speaks either English and Russian or French and Russian. In the latter case, as I speak my words in French, they arrive at the destination having moved twice between languages. Anyone who has played the children's game "telephone" knows the outcome of this process. From a GIS context, we finish with data of unknown accuracy, source, projection, and with unspecified or imperfectly matched attributes. On one country map, perhaps, all streams and rivers are shown, on the other only the major ones. One might think that the climate was wetter, but in fact the difference is one of interpretation and lack of standardization.

The GIS industry in the United States began a standardization effort in the mid-1980s, which led to a federal standard approved in 1992. This standard, the Federal Information Processing Standard 173, called the Spatial Data Transfer Standard (SDTS), had to be quite broad because of the huge degree of complexity involved. Not only did the standard have to produce a bibliography, a terminology, and a complete list of geographic and map features, it also had to address the problems of data accuracy and the broader metadata issues for data description. The terminology created sets of terms for features and data structures that have become commonplace.

Best of all, the standard included a mechanism for file exchange. As a result, two implementations of the standard, called *profiles*, have been developed, for vector, raster, and point data. Already, data sets are being made available in the vector profile of the standard for DLG and for TIGER data. These have been termed DLG-SDTS and TIGER-SDTS. At the same time, many GIS vendors have incorporated input and output utilities that read and write the data in SDTS format and to SDTS specifications. Back to the spoken language analogy, I have convinced the Russian to learn English, and with luck most of the rest of the world too! The cost was many hours spent debating exactly what was meant by each and every word that was to be used in discussion in everyone's language.

The United States civilian mapping agencies have not been alone in seeking standardization of GIS information for data exchange (Figure 3.21). The U.S. military has drafted standards for use among the Army, Navy, and Air Force called the Tri-Service Spatial Data Standards. Within the members of NATO, an exchange standard called DIGEST was developed, with a vector data profile called the Vector Product Format (VPF). This format is best known as the format in which the Digital Chart of the World was released on CD-ROM. Similar efforts are under way to standardize data exchange in Germany, Australia, South Africa, the European Union, and for worldwide nautical chart data by the International Hydrographic Organization (DX-90). As data become used for global instead of local and national projects, the ability to exchange data internationally will increase in importance. International peacekeeping and disaster relief efforts, for example, involve cooperation and therefore exchange of GIS data among many quite different countries and organizations.

We will return to this issue in Chapter 10, when we discuss the future of GIS. Obviously, the open exchange of data can help those developing and using GIS considerably. The SDTS took many years to develop and met with considerable resistance. Nevertheless, the advantages for a GIS future when data can be imported and exported at will promises a more effective use of GIS, and allows GIS users to concentrate on the science of data analysis and the effectiveness of common-sense information use, rather than the politics of data acquisition.

owever, we need someone
an. In the latter case, as I
ing moved twice between
hone" knows the outcome
nknown accuracy, source,
ites. On one country map,
he major ones. One might
. one of interpretation and

dization effort in the mid-
This standard, the Federal
Transfer Standard (SDTS),
lexity involved. Not only
gy, and a complete list of
lems of data accuracy and
logy created sets of terms
ce.
exchange. As a result, two
veloped, for vector, raster,
n the vector profile of the
d DLG-SDTS and TIGER-
d input and output utilities
ifications. Back to the spo-
English, and with luck most
debating exactly what was
on in everyone's language.
been alone in seeking stan-
21). The U.S. military has
orce called the Tri-Service
exchange standard called
he Vector Product Format
Digital Chart of the World
andardize data exchange in
or worldwide nautical chart
. As data become used for
change data internationally
disaster relief efforts, for
S data among many quite

discuss the future of GIS.
oping and using GIS con-
ith considerable resistance.
be imported and exported
users to concentrate on the
nse information use, rather

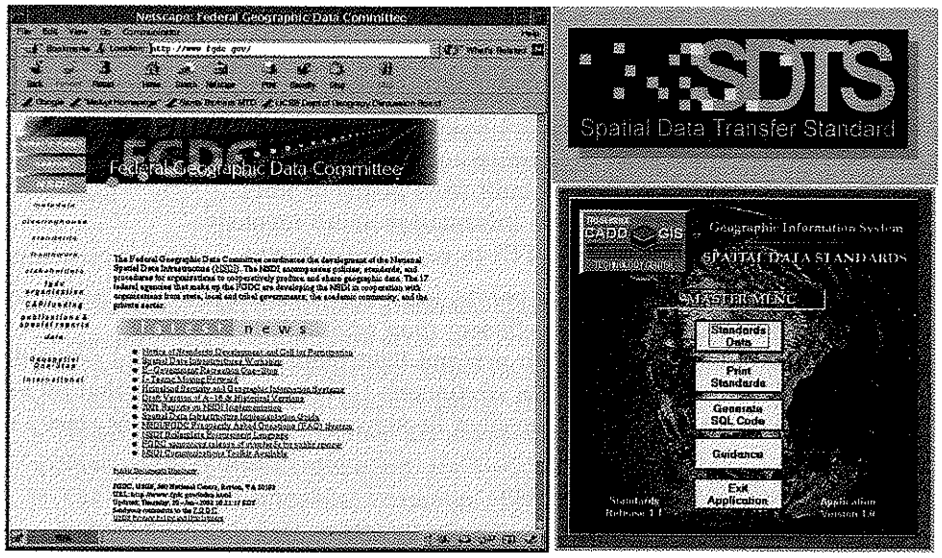

Netscape: Federal Geographic Data Committee
Federal Geographic Data Committee
news
SDTS
Spatial Data Transfer Standard
CADD GIS
Geographic Information System
SPATIAL DATA STANDARDS
MASTER MENU
Standards Data
Print Standards
Generate SQL Code
Guidance
Exit Application

FIGURE 3.21: Some Spatial Data Exchange Standards documents.

3.7 STUDY GUIDE

3.7.1 Summary

CHAPTER 3: Maps as Numbers

Representing Maps as Numbers (3.1)

- **GIS requires that both data and maps be represented as numbers.**
- **The GIS places data into the computer's memory in a physical data structure (i.e., files and directories).**
- **Files can be written in binary or as ASCII text.**
- **Binary is faster to read and smaller; ASCII can be read and edited by humans but uses more space.**
- **Programmers use hexadecimal as shorthand for binary, since two hexadecimal digits correspond to 8 bits (a byte).**
- **A logical data model is how data are organized for use by the GIS.**
- **GISs have traditionally used the logical data models of either raster or vector, and for attributes the flat file.**
- **A raster data model uses a grid.**
  - **One grid cell is one unit or holds one attribute.**
  - **Every cell has a value, even if it is "missing."**
  - **A cell can hold a number or an index value standing for an attribute.**
  - **A cell has a resolution, given as the cell size in ground units.**
  - **Points and lines in raster format have to move to a cell center.**
  - **Lines can become fat. Areas may need separately coded edges.**
  - **Each cell can only be owned by one feature.**
  - **As data, all cells must be able to hold the maximum cell value.**

- **Rasters are easy to understand, easy to read and write, and easy to draw on the screen.**
- **A vector data model uses points stored by their real coordinates.**
  - **Lines and areas are built from sequences of points in order.**
  - **Lines have a direction to the ordering of the points.**
  - **Polygons can be built from points or lines.**
  - **Vectors can store information about topology.**
  - **The model uses TIN to represent volumes.**
- **Vectors can represent points, lines, and area features very accurately.**
- **Vectors are far more efficient than grids.**
- **Vectors work well with pen and light plotting devices, and tablet digitizers.**
- **Vectors are not good at continuous coverages or plotters that fill areas.**
- **Vectors are like the music of Beethoven, rasters are like Mozart's music.**

Structuring Attributes (3.2)

- **Attribute data are stored logically in flat files.**
- **A flat file is a matrix of numbers and values stored in rows and columns, like a spreadsheet.**
- **Both logical and physical data models have evolved over time.**
- **DBMSs use many different methods to store and manage flat files in physical files.**

Structuring Maps (3.3)

- **A GIS map is a scaled-down digital representation of point, line, area, and volume features.**
- **While most GIS systems can handle raster and vector, only one is used for the internal organization of spatial data.**
- **VECTOR**
- **At first, GISs used vector data and cartographic spaghetti structures.**
- **Vector data evolved the arc/node model in the 1960s.**
- **In the arc/node model, an area consists of lines and a line consists of points.**
- **Points, lines, and areas can each be stored in their own files, with links between them.**
- **The topological vector model uses the line (arc) as a basic unit. Areas (polygons) are built up from arcs.**
- **The endpoint of a line (arc) is called a node. Arc junctions are only at nodes.**
- **Stored with the arc is the topology (i.e., the connecting arcs and left and right polygons).**
- **Volumes (surfaces) are structured with the TIN model, including edge or triangle topology.**
- **TINs use an optimal Delaunay triangulation of a set of irregularly distributed points.**
- **TINs are popular in CAD and surveying packages.**
- **RASTER**
- **A grid or raster maps directly onto a programming computer memory structure called an array.**

e, and easy to draw on

oordinates.

ts in order.

nts.

very accurately.

and tablet digitizers.

ers that fill areas.

ke Mozart's music.

rows and columns, like

ver time.

nage flat files in physi-

of point, line, area, and

or, only one is used for

ghetti structures.

line consists of points.

files, with links between

basic unit. Areas (poly-

ctions are only at nodes.

g arcs and left and right

el, including edge or tri-

of irregularly distributed

computer memory struc-

- Grids are poor at representing points, lines, and areas, but good at surfaces.
- Grids are good only at very localized topology, and weak otherwise.
- Grids are a natural for scanned or remotely sensed data.
- Grids suffer from the mixed pixel problem.
- Grids must often include redundant or missing data.
- Grid compression methods used in GIS include run-length encoding and quad trees.

Why Topology Matters (3.4)

- Topological data structures dominate GIS software.
- Topology allows automated error detection and elimination.
- Rarely are maps topologically clean when digitized or imported.
- A GIS has to be able to build topology from unconnected arcs.
- Nodes that are close together are snapped.
- Slivers due to double digitizing and overlay are eliminated.
- The tolerances controlling snapping, elimination, and merging must be carefully considered, because they can move features.
- Complete topology makes map overlay feasible.
- Topology allows many GIS operations to be done without accessing the point files.

Formats for GIS Data (3.5)

- Most GIS systems can import different data formats, or use utility programs to convert them.
- Data formats can be industry standard, commonly accepted, or standard.
- Vector formats are either page definition languages or preserve ground coordinates.
- Page languages are HPGL, PostScript, and AutoCAD DXF.
- True vector GIS data formats are DLG and TIGER, which has topology.
- Most raster formats are digital image formats.
- Most GISs accept TIF, GIF, JPEG, or encapsulated PostScript, which are not georeferenced.
- DEMs are true raster data formats.

Exchanging Data (3.6)

- Most GISs use many formats and one data structure.
- If a GIS supports many data structures, changing structures becomes the user's responsibility.
- Changing vector to raster is easy; raster to vector is hard.
- Data also are often exchanged, or transferred between different GIS packages and computer systems.
- The history of GIS data exchange is chaotic, and has been wasteful.
- Data exchange by translation (export and import) can lead to significant errors in attributes and in geometry.
- In the United States, the SDTS was evolved to facilitate data transfer.
- SDTS became a Federal Standard (FIPS 173) in 1992.

- **SDTS contains a terminology, a set of references, a list of features, a transfer mechanism, and an accuracy standard.**
- **Both DLG and TIGER data are available in SDTS format.**
- **Other standards efforts are DIGEST, DX-90, the Tri-Service Spatial Data Standards, and many other international standards.**
- **Efficient data exchange is important for the future of GIS.**

3.7.2 Study Questions

Representing Maps as Numbers

Define bit, byte, file, attribute, record, hexadecimal, binary. Now define data model, data structure, logical and physical data structure, vector, and raster. Place each word on a "word line" with the computer hardware at one end and the GIS user at the other end.

Make a list of the reasons why different GIS packages might have different data structures.

Structuring Attributes

Make up a small attribute database, say with six records for the locations of six local businesses. Color in or highlight the attributes that contain spatial data, and think about how you would place the six locations on a street map by hand. If your table were to be resorted using one of the attributes, would the spatial indexing still work?

Structuring Maps

Make a table of advantages and disadvantages of vector and raster data for GIS. What sorts of applications would each be most suited to?

Explain Dana Tomlin's statement about raster and vector at the beginning of the chapter.

Vector Data Structures

Define the following: cartographic spaghetti, point file, arc, polygon, topology, forward link, polygon left, TIN.

Raster Data Structures

Define the following: pixel, resolution, grid extent, fat line, mixed pixel, polygon boundary, array.

Why might the attribute in a particular pixel be wrong as soon as it is geocoded?

Why Topology Matters

Write your own definition of topology. Make a simple diagram of one or two polygons, connected by arcs. Label the polygons A, B, C, and so on. Label the arcs 1, 2, 3, and so on. Now create the arcs file as a table. Make the first arc 1, second 2, and so on. Add extra columns to the table for forward and reverse links, and polygon left and right. How do you deal with the "outside"? How might you deal with a "hole" inside a polygon?

ist of features, a transfer

format.
Tri-Service Spatial Data

of GIS.

ow define data model, data
ter. Place each word on a
GIS user at the other end.

ve different data structures.

the locations of six local
patial data, and think about
nd. If your table were to be
g still work?

raster data for GIS. What

e beginning of the chapter.

polygon, topology, forward

ixed pixel, polygon bound-

on as it is geocoded?

m of one or two polygons,
abel the arcs 1, 2, 3, and so
second 2, and so on. Add
polygon left and right. How
"hole" inside a polygon?

Formats for GIS Data

List three characteristics of each of the following GIS data formats: TIGER, DLG, DEM, TIF, GIF, JPEG, HPGL, DXF, PostScript.

Exchanging Data

Make a list of the advantages and disadvantages of sharing GIS data. What obstacles to data sharing exist at the level of one company, a municipality, a state, or between nations?

3.8 EXERCISES

1. *Carefully examine the documentation for your GIS, or find details of a GIS from one of the sources listed in Chapter 1. Make a table with entries for the following: primary data structure, import formats, export formats, attribute data structure, data model, files and directories used, data transfer method. Look up the advantages and disadvantages of those that your GIS supports. Now list three applications your GIS would be best suited to and three it would be worst suited to.*
2. *Using a word processor, editing tool, or your GIS create a file in one of the formats in this chapter, such as HPGL or PostScript. Use an editor or word processor to look at the file if it is in ASCII format. Try to identify the various parts of the file such as the header, the data, and how images are stored. How big is the file? What coordinates are used inside the file?*
3. *Use a library or the Internet to find out as much as possible about data transfer standards. How many different countries have or are working on standards? Use the Internet to download a file in SDTS format, such as a TIGER or DLG file, and try to import it into your GIS. Explain the irony in the statement "the nice thing about standards is that there are so many to choose from."*
4. *By using your GIS, make an assessment of how much about the data structure is "hidden" from the GIS user by performing three different GIS operations and noting how much information the GIS requires you to supply about the data. Make a list of the advantages and disadvantages of hiding the data structure from the GIS user.*

3.9 REFERENCES

3.9.1 Chapter References

Burrough, P. A. and R. A. McDonnell (1998) *Principles of Geographical Information Systems.* Oxford: Oxford University Press.

Clarke, K. C. (1995) *Analytical and Computer Cartography* 2nd ed. Upper Saddle River, NJ: Prentice Hall.

Dutton, G., ed. (1979) *Harvard Papers on Geographic Information Systems.* First International Advanced Study Symposium on Topological Data Structures for Geographic Information Systems. Reading, MA: Addison-Wesley.

Peucker, T. K. and N. Chrisman (1975) "Cartographic data structures." *American Cartographer,* vol. 2, no. 1, pp. 55–69.

Peucker, T. K., R. J. Fowler, J. J. Little, and D. M. Mark (1976) *Digital Representation of Three-dimensional Surfaces by Triangulated Irregular Networks (TIN).* Technical Report No. 10, U.S. Office of Naval Research, Geography Programs.

Samet, H. (1990) *Design and Analysis of Spatial Data Structures.* Reading, MA: Addison-Wesley.

Tomlin, D. (1990) *Geographic Information Systems and Cartographic Modelling.* Upper Saddle River, NJ: Prentice Hall.

3.10 KEY TERMS AND DEFINITIONS

address range: The range from the highest to the lowest street number on one side of a street, on one block.

arc: A line that begins and ends at a topologically significant location, represented as a set of sequential points.

arc-node: Early name for the vector GIS data structure.

area: A two-dimensional (area) feature represented by a line that closes on itself to form a boundary.

array: A physical data structure for grids. Arrays are part of most computer programming languages, and can be used for storing and manipulating raster data.

ASCII: The American Standard Code for Information Interchange. A standard that maps commonly used characters such as the alphabet onto one-byte-long sequences of bits.

attribute: An attribute is a characteristic of a feature that contains a measurement or value for the feature. Attributes can be labels, categories, or numbers. Attributes can be dates, standardized values, or field or other measurements. Item for which data are collected and organized. A column in a table or data file.

Autocad: A leading CAD program by Autodesk, often interfaced with GIS packages and used for digitizing, especially floor plans and engineering graphics.

block face: One side of a street on one block, which is between two street intersections.

bounding rectangle: The rectangular region defined by the maximum extent of a map feature in the *x* and *y* directions. All parts of the feature must lie within or on the edge of the bounding rectangle.

bit: The smallest storable unit within a computer's memory with only an on and an off state, codable with one binary digit.

byte: Eight consecutive bits.

CAD: Computer aided design. Computer software used in producing technical and design-type drawings.

cartographic spaghetti: A loose data structure for vector data, with only order as an identifying property to the features.

color table: Part of the header record in a digital image file that stores specifications of colors based on simple index values, which are then stored in the data part of the image file.

computer memory: A sequence of nonrandom bytes that are recoverable after a computer has been turned off and on again.

data analysis: The process of using organized data to test scientific hypotheses.

database: Any collection of data accessible by computer.

data dictionary: The part of a database containing information about the files, records, and attributes rather than just the data.

data exchange: The exchange of data between similar GIS packages but groups with a common interest.

data format: A specification of a physical data structure for a feature or record.

data model: The logical means of organization of data for use in an information system.

data retrieval: The ability of a database management system to get back from computer memory records that were previously stored there.

data structure: The logical and physical means by which a map feature or an attribute is digitally encoded.

data transfer: The exchange of data between noncommunicating computer systems and different GIS software packages.

DBMS: Database management system. Part of a GIS, the set of tools that allow the manipulation and use of files containing attribute data.

decennial census: The effort required by the U.S. constitution that every 10 years all people be counted and their residences located.

decimal: The counting system when people have 10 fingers.

Delaunay triangulation: An optimal partitioning of the space around a set of irregular points into nonoverlapping triangles and their edges.

DEM: Digital elevation model. A raster format gridded array of elevations.

DIGEST: The NATO transfer standard for spatial data.

DIME: Dual Independent Map Encoding. The data model used for the Census Bureau's Geographic Base Files, the predecessor of TIGER.

digital elevation model: A data format for digital topography, containing an array of terrain elevation measurements.

DLG: A vector format used by the USGS for encoding lines on large-scale digital maps.

double digitized: The same feature captured by digitizing twice.

DXF: Autocad's digital file exchange format, a vector mode industry-standard format for graphic file exchange.

editor: A computer program for the viewing and modification of files.

elevation: The vertical height above a datum, in units such as meters or feet.

encapsulated PostScript: A version of the PostScript language that allows digital images to be included and stored for later display.

end node: The last point in an arc that connects to another arc.

enumeration map: Map designed to show one census enumerator the geographic extent and address ranges within one's district.

export: The capability of a GIS to write data out into an external file and into a nonnative format for use outside the GIS, or in another GIS.

fat line: Raster representation of a line that is more than one pixel wide.

feature: A single entity that composes part of a landscape.

field: The contents of one attribute for one record, as written in a file.

file: A collection of bytes stored on a computer's storage device.

file header: The first part of a file that contains metadata rather than data.

FIPS 173: The Federal Information Processing Standard maintained by the USGS and the National Institute of Standards and Technology that specified a standard organization and mechanism for the transfer of GIS data between dissimilar computer systems. FIPS 173 specifies terminology, features types, and accuracy specifications, as well as a formal file transfer method.

forward/reverse left: Moving along an arc, the identifier for the arc connected in the direction/opposite direction of the arc to the immediate left.

forward/reverse right: Moving along an arc, the identifier for the arc connected in the direction/opposite direction of the arc to the immediate right.

fully connected: A set of arcs in which forward and reverse linkages have identically matching begin and end nodes.

GBF: Geographic Base File. A database of DIME records.

geographical surface: The spatial distribution traced out by a continuously measurable geographical phenomenon, as depicted on a map.

GIF: An industry standard raster graphic or image format.

grid cell: A single cell in a rectangular grid.

grid extent: The ground or map extent of the area corresponding to a grid.

hexadecimal: The counting system that would be used if people had 16 fingers.

hierarchical: System based on sets of fully enclosed subsets and many layers.

HPGL: Hewlett Packard Graphics Language. A device-specific but industry-standard language for defining vector graphics in page coordinates.

image depth: The numbers of bits stored for each pixel in a digital image.

import: The capability of a GIS to bring data in an external file and in a nonnative format for use within the GIS.

industry standard format: A commonly accepted way of organizing data, usually advanced by a private organization.

internal format: A GIS data format used by the software to store the data within the program, and in a manner unsuitable for use by other means.

label point: A point digitized within a polygon and assigned its label or identifier for use in topological reconstruction of the polygon.

landmark: TIGER term for a geographic feature not a part of the census features.

layer: A set of digital map features collectively (points, lines, and areas) with a common theme in coregistration with other layers. A feature of GIS and most CAD packages.

line: A one-dimensional (length) map feature represented by a string of connected coordinates.

logical structure: The conceptual design used to encrypt data into a physical structure.

magic number: Any number that has a specific value for a specialized need.

matrix: A table of numbers with a given number of rows and columns.

metadata: Data about data, usually for search and reference purposes.

missing data: Elements where no data is available for a feature or a record.

mixed pixel: A pixel containing multiple attributes for a single ground extent of a grid cell. Common along the edges of features or where features are ill defined.

node: The end of an arc. At first, any significant point in a map data structure. Later, only those points with topological significance, such as the ends of lines.

page coordinates: The set of coordinate reference values used to place the map elements on the map, and within the map's own geometry rather than the geometry of the ground that the map represents. Often page coordinates are in inches or millimeters from the lower left corner of a standard size sheet of paper, such as A4 or 8 1/2 by 11 inches.

physical structure: The mechanical mapping of a section of computer memory onto a set of files or storage devices.

pixel: The smallest unit of resolution on a display, often used to display one grid cell at the highest display resolution.

point: A zero-dimensional map feature, such as a single elevation mark as specified by at least two coordinates.

polygon: A many-sided area feature consisting of a ring and an interior. An example is a lake on a map.

polygon interior: The space contained by a ring, considered part of a polygon.

polygon left: Moving along an arc, the identifier for the polygon adjacent to the left.

polygon right: Moving along an arc, the identifier for the polygon adjacent to the right.

PostScript: Adobe Corp.'s page definition language. An interpreted language for page layout designed for printers but also an industry standard for vector graphics.

g to a grid.
had 16 fingers.
l many layers.
c but industry-standard

ital image.
file and in a nonnative

rganizing data, usually

ore the data within the

s label or identifier for

ne census features.
l areas) with a common
d most CAD packages.
string of connected co-

to a physical structure.
cialized need.
olumns.
poses.
or a record.
ground extent of a grid
re ill defined.
p data structure. Later,
ds of lines.
place the map elements
geometry of the ground
or millimeters from the
4 or 8 1/2 by 11 inches.
mputer memory onto a

display one grid cell at

on mark as specified by

interior. An example is

t of a polygon.
n adjacent to the left.
on adjacent to the right.
eted language for page
vector graphics.

quad tree: A way of compressing raster data based on eliminating redundancy for attributes within quadrants of a grid.

RAM: The part of a computer's memory designed for rapid access and computation.

raster: A data structure for maps based on grid cells.

ring: A line that closes upon itself to define an area.

run-length encoding: A way of compressing raster data based on eliminating redundancy for attributes along rows of a grid.

SLF: An early Defense Mapping Agency data format.

sliver: Very small and narrow polygon caused by data capture or overlay error that does not exist on the map.

snap: Forcing two or more points within a given radius of each other to be the same point, often by averaging their coordinate.

Spatial Data Transfer Standard: The formal standard specifying the organization and mechanism for the transfer of GIS data between dissimilar computer systems. Adopted as FIPS 173 in 1992, SDTS specifies terminology, features types, and accuracy specifications, as well as a formal file transfer method for any generic geographic data. Subsets for the standard for specific types of data, vector, and raster, for example, are called profiles.

spreadsheet: A computer program that allows the user to enter numbers and text into a table with rows and columns, and then maintain and manipulate those numbers using the table structure.

table: Any kind of organization by placement of records into rows and columns.

TIF: An industry-standard raster graphic or image format.

TIGER: A map data format based on zero-, one-, and two-cells, used by the U.S. Census Bureau in the street-level mapping of the United States.

TIN: A vector topological data structure designed to store the attributes of volumes, usually geographic surfaces.

tolerance: The distance within which features are assumed to be erroneously located different versions of the same thing.

topologically clean: The status of a digital vector map when all arcs that should be connected are connected at nodes with identical coordinates, and the polygons formed by connected arcs have no duplicate, disconnected, or missing arcs.

topology: The property that describes adjacency and connectivity of features. A topological data structure encodes topology with the geocoded features.

USGS: The United States Geological Survey, part of the Department of the Interior and a major provider of digital map data for the United States.

vector: A map data structure using the point or node and the connecting segment as the basic building block for representing geographic features.

volume: A three-dimensional (volume) feature represented by a set of areas enclosing part of a surface, in GIS usually the top only.

VPF: Vector product format, a data transfer standard within DIGEST for vector data.

zero/one/two cell: TIGER terminology for point, line, and area, respectively.