2 Data Models

Introduction

Data in a GIS represent a simplified view of physical *entities*, the roads, mountains, accident locations, or other features we wish to identify. Data include information on the spatial location and extent of the entities, and information on their nonspatial properties.

Each entity is represented by a *spatial feature* or *cartographic object* in the GIS, and so there is an entity–object correspondence. Because every computer system has limits, only a subset of essential characteristics is recorded. As illustrated in Figure 2-1, we may represent landcovers in a region by a set of polygons. These polygons are associated with a set of essential characteristics that define each landcover, perhaps vegetation type, ownership, or landuse.

Essential characteristics are subjectively chosen by the spatial data developer. The essential characteristics of a forest would be different in the eyes of a logger than those of a conservation officer, a hunter, or a hiker. Objects are abstract representations of reality that we store in a spatial database, and they are imperfect representations because we can only record a subset of the characteristics of any entity. No one abstraction is universally better than any other, and the goal of the GIS developer is to define objects that support the

Figure 2-1: A physical entity is represented by a spatial object in a GIS. Here, lakes (dark areas) and other land cover types are represented by polygons.

intended use at the desired level of detail and accuracy.

A *spatial data model* (Figure 2-2) may be defined as the objects in a spatial database plus the relationships among them. The term "model" is fraught with ambiguity because it is used in many disciplines to describe many things. Here the purpose of a spatial data model is to provide a formal means of representing and manipulating spatially referenced information. In Figure 2-1, our data model consists of two parts. The first part is a set of polygons (closed areas) recording the edges of distinct land uses, and the second part is a set of numbers or letters associated with each polygon. The data model may be considered the most recognizable level in our computer abstraction of the real world. Data structures and binary machine code are successively less recognizable, but more computer-compatible, forms of the spatial data.

Coordinates are used to define the spatial location and extent of geographic objects. A coordinate most often consists of a pair or triplet of numbers that specify location in relation to a point of origin. The coordinates quantify the distance from the origin when measured along standard directions. Single or groups of coordinates are organized to represent the shapes and boundaries that define the objects. Coordinate information is an important part of the data model, and models differ in how they represent these coordinates. Coordinates are usually expressed in one of many standard coordinate systems. The coordinate systems are usually based upon standardized map projections (discussed in Chapter 3) that unambiguously define the coordinate values for every point in an area.

Typically, attribute data complement coordinate data to define cartographic objects (Figure 2-3). Attribute data are collected and referenced to each object. These

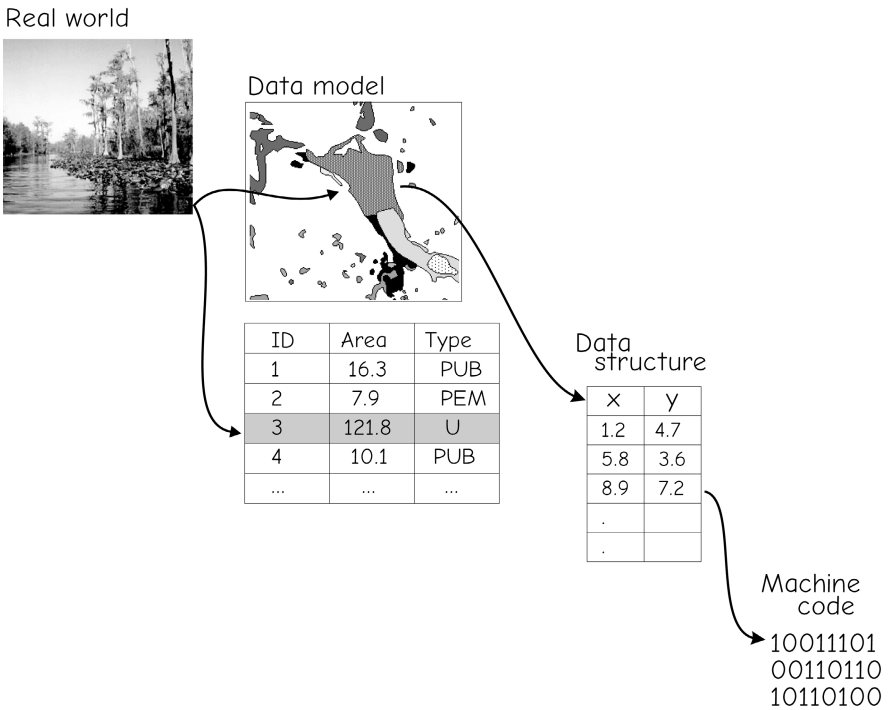


Figure 2-2: Levels of abstraction in the representation of spatial entities. The real world is represented in successively more machine-compatible but humanly obscure forms.

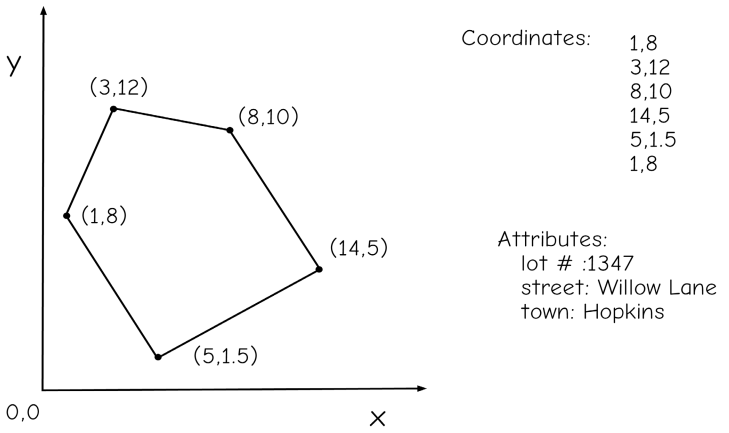


Figure 2-3: Coordinate and attribute data are used to represent entities.

attribute data record the non-spatial components of an object, such as a name, color, pH, or cash value. Keys, labels, or other indices are used so that the coordinate and attribute data may be viewed, related, and manipulated together.

Most conceptualizations view the world as a set of layers (Figure 2-4). Each layer organizes the spatial and attribute data for a given set of cartographic objects in the region of interest. These are often referred to as *thematic layers*. As an example, consider a GIS database that includes a soils data layer, a population data layer, an elevation data layer, and a roads data layer. The roads layer contains only roads data, including the location and properties of roads in the analysis area (Figure 2-4). There are no data regarding the location and properties of any other geographic entities in the roads layer. Information on soils, population, and elevation are contained in their respective data layers. Through analyses we may combine data to create a new data layer; for example, we may identify areas that have high elevation and join this information with the soils data. This combination may create a new data layer with a new, composite soils/elevation variable.

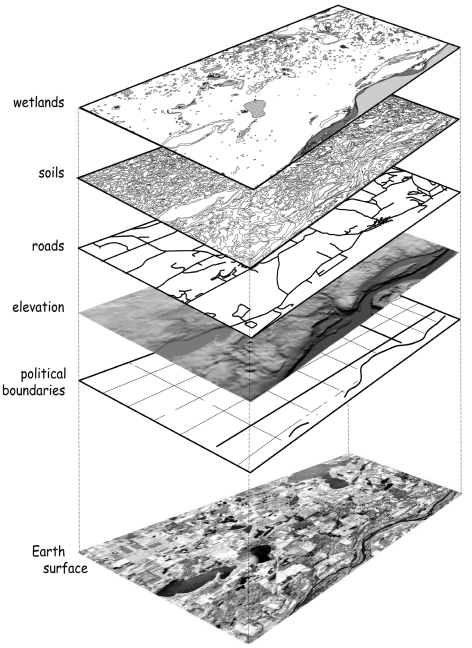


Figure 2-4: Spatial data are often stored as separate thematic layers, with objects grouped based on a set of properties, e.g., water, roads, or land cover, or some other agreed-upon set.

Coordinate Data

Coordinates define location in two- or three-dimensional space. Coordinate pairs, x and y, or coordinate triples, x, y, and z, are used to define the shape and location of each spatial object or phenomenon.

Spatial data in a GIS most often use a *Cartesian* coordinate system, so named after Rene Descartes, the system's originator. Cartesian systems define two or three *orthogonal* (right angle, or 90^o) axes. Two-dimensional Cartesian systems define x and y axes in a plane (Figure 2-5, left). Three-dimensional Cartesian systems in addition define a z axis, orthogonal to both the x and y axes. The origin is defined with zero values at the intersection of the orthogonal axes (Figure 2-5, right). Cartesian coordinates are usually specified as decimal numbers that increase from bottom to top and from left to right.

Two-dimensional Cartesian coordinate systems are the most common choice for mapping small areas. Small is a relative term, but here we mean maps of farm fields, land and property, cities, and counties. We typically introduce acceptably small errors for most applications when we ignore the Earth's curvature over these small areas. When we map over larger areas, or need the highest precision and accuracy, we usually must choose a three-dimensional system.

Coordinate data may also be specified in a *spherical coordinate system*. Hipparchus, a Greek mathematician of the 2nd century B.C., was among the first to specify locations on the Earth using angular measurements on a sphere. The most common spherical system uses two angles of rotation and a radius distance, r, to specify locations on a modeled earth surface (Figure 2-6). The first angle of rotation, the longitude (λ), is measured around the imaginary axis on which the Earth spins, an axis that goes through the North and South Poles. Zero is set for a location in England, and the angle is positive eastward and negative westward (Figure 2-6). The zero longitude, also known as the *Prime Meridian* or the *Greenwich Meridian*, was first specified through the Royal Greenwich Observatory in England, but measurement improvements, crustal movements, and changes in conventions now place zero longitude about 102 meters

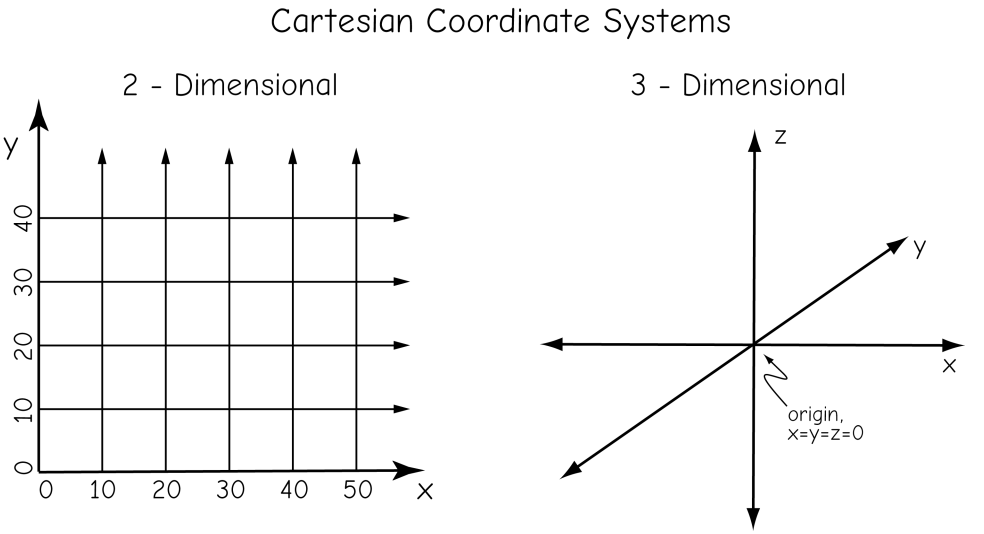


Figure 2-5: Two dimensional (left) and three dimensional (right) Cartesian coordinate systems.

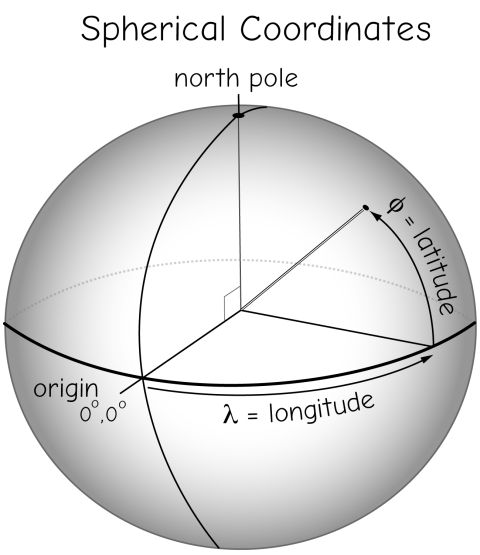


Figure 2-6: Three-dimensional, spherical coordinates may define location by two angles of rotation, λ and ϕ, and a radius vector, r, to a point on a sphere.

(335 feet) east of the Greenwich Observatory. East or west longitudes are specified as angles of rotation away from the zero line.

A second angle of rotation, measured along lines that intersect both the north and south poles, is used to define a latitude (ϕ, Figure 2-6). Latitudes are specified as zero at the equator, a line encircling the Earth that is always halfway between the North and South Poles. By convention latitudes increase to maximum values of 90 degrees in the north and south, or, if a sign convention is used, from -90 at the South pole to 90 at the North pole. Lines of constant longitude are called meridians, and lines of constant latitude are called parallels (Figure 2-7) . Parallels run parallel to each other in an east-west direction around the Earth. The meridians are north/south lines that converge to intersect at the poles.

Because the meridians converge, geographic coordinates do not form a Cartesian system. A Cartesian system defines lines on a right-angle, planar grid. Geographic coordinates occur on a curved surface, and the longitudinal lines cross at the poles. This convergence means the distance spanned by a degree of longitude varies from south to north. A degree of longitude spans approximately 111.3 kilometers at the equator, but 0 kilometers at the poles. In contrast, the ground distance for a degree of latitude varies only slightly, from 110.6 kilometers at the equator to 111.7 kilometers at the poles. The slight difference with latitude is due to a non-spherical Earth, something we'll describe a bit later.

Convergence causes distortion in regular geometric figures specified in geographic coordinates (Figure 2-8, left). For example, "circles" with a fixed radius in geographic units, such as 5^{o}, are not circles on the surface of the globe, although they may appear as circles when the Earth surface is "unrolled" and plotted with distortion on a flat map; note the erroneous size and shape of Antarctica at the bottom of Figure 2-8.

Because the spherical system for geographic coordinates is non-Cartesian, formulas for area, distance, angles, and other geometric properties that work in a Cartesian coordinates give errors when applied to geographic coordinates. Areas are calculated after converting to a projected system, described later in this chapter.

There are two primary conventions used for specifying the magnitudes of latitude and longitude (Figure 2-9). The first uses a leading letter, N, S, E, or W to indi-

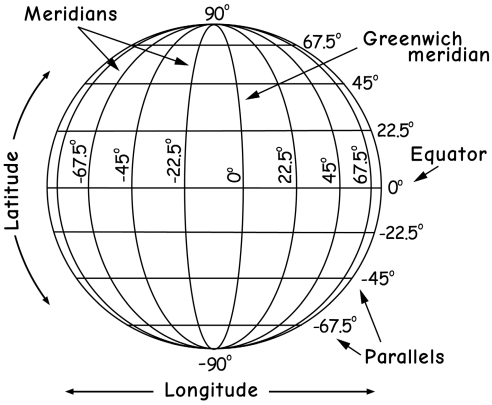


Figure 2-7: Nomenclature of geographic latitudes and longitudes.

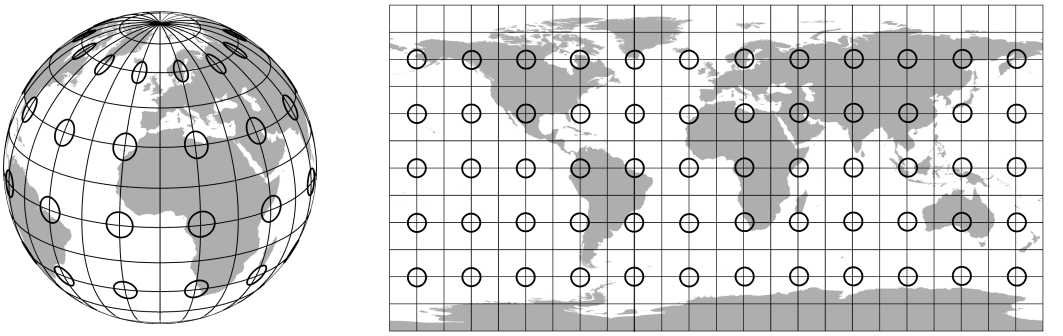


Figure 2-8: Geographic coordinates on a spherical (left) and Cartesian (right) representation. Notice that "circles" defined by a 5 degree radius are distorted near the poles, as shown on the spherical representation, but appear as circles in the highly distorted Cartesian plot of geographic coordinates (right).

cate direction, followed by a number to indicate location. Northern latitudes are preceded by an N and southern latitudes by an S; for example, N90^o, S10^o. Longitude values are preceded by an E or W, respectively; for example W110^o. Longitudes range from 0 to 180 degrees east or west. Note that the east and west longitudes meet at 180 degrees, so that E180^o equals W180^o.

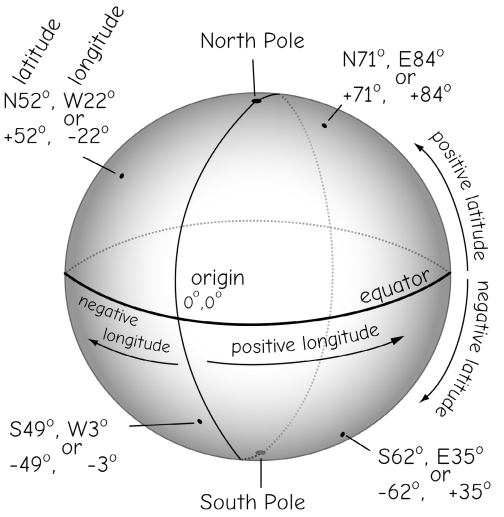


Figure 2-9: Spherical coordinates of latitude and longitude are most often expressed as directional (N/S, E/W), or as signed numbers. Latitudes are positive north, negative south; longitudes are positive east, negative west.

Signed coordinates are the second common convention for specifying latitude and longitude in a spherical system. Northern latitudes are positive and southern latitudes are negative, and eastern longitudes positive and western longitudes negative. Latitudes vary from -90 degrees to 90 degrees, and longitudes vary from -180 degrees to 180 degrees. By this convention the longitudes "meet" at the maximum and minimum values, so -180^o equals 180^o.

Coordinates may easily be converted between these two conventions. North latitudes and east longitudes are converted by removing the leading N or E, respectively. South latitudes and west longitudes are converted by first removing the leading S or W, respectively, and then changing the sign of the remaining number from a positive to a negative value.

Spherical coordinates are most often recorded in a degrees-minutes-seconds (DMS) notation; N43^o 35′ 20″ for 43 degrees, 35 minutes, and 20 seconds of latitude. In DMS, each degree is made up of 60 minutes of arc, and each minute is in turn divided into 60 seconds of arc (Figure 2-10). This yields 60 times 60 or 3600 seconds for each degree of latitude or longitude. Note that the ancient Babylonians established these splits, of 360 degrees for a complete circle, with degrees and minutes subse-

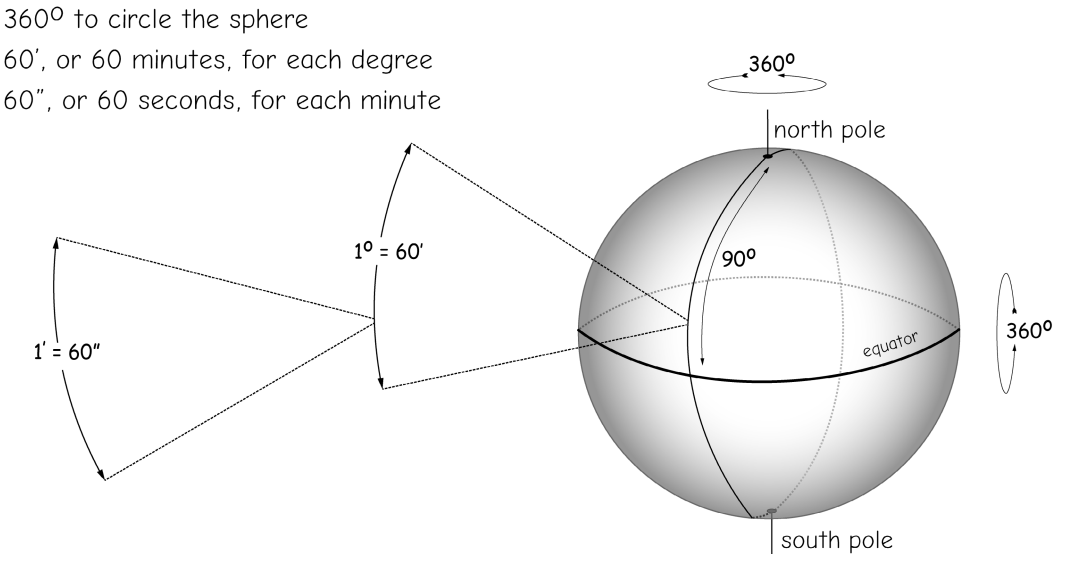


Figure 2-10: There are 360 degrees in a complete circle, with each degree composed of 60 minutes, and each minute composed of 60 seconds.

quently divided into 60 units, and we've carried this convention down to today.

Spherical coordinates may also be expressed as decimal degrees (DD). When using DD the degrees take the usual -180 to 180 (longitude) and -90 to 90 (latitude) ranges, but minutes and seconds are reported as a decimal portion of a degree (from 0 to 0.99999...). In our previous example, N43° 35′ 20″ would be reported as 43.5888. DMS may be converted to DD by:

$$DD = DEG + MIN/60 + SEC/3600 \quad (2.1)$$

Examples of the forward and reverse conversion between decimal degrees and degrees-minutes-seconds units are shown in Figure 2-11.

DD from DMS
DD = D + M/60 + S/3600
e.g.
DMS = 32° 45' 28"

DD = 32 + 45/60 + 28/3600
= 32 + 0.75 + 0.0077778
= 32.7577778

DMS from DD
D = integer part
M = integer of decimal part x 60
S = 2nd decimal x 60
e.g.
DD = 24.93547
D = 24
M = integer of 0.93547 x 60
= integer of 56.1282
= 56
S = 2nd decimal x 60
= 0.1282 * 60 = 7.692
so DMS is
24° 56' 7.692"

Figure 2-11 Examples for converting between DMS and DD expressions of spherical coordinates.

An Ellipsoidal Earth

While we often describe the Earth's shape as a sphere, it is better approximated as an ellipsoid. An ellipsoid is a spherical solid with unequal radii along perpendicular axes (Figure 2-12). It may be envisioned as being "flattened" along its axis of revolution. This flattening is quite small, and is approximately one part in 300. This would be imperceptible to the unaided eye translated to human scales, for example, 30 cm over the length of a soccer pitch, but is significant enough to need modeling for the most precise measurements and navigation on the surface of the Earth. Many navigation and measurement estimates have two sets of formulas, one an approximation based on a purely spherical globe, and a more precise, and much more complicated set based on an ellipsoidal shape.

Note that the words spheroid and ellipsoid are often used interchangeably. GIS software often prompts the user for a spheroid when defining a coordinate projection, and then lists a set of ellipsoids. An ellipsoid is sometimes referred to as an "oblate" spheroid. Thus, it is less precise but still correct to refer to an ellipsoid more generally as a spheroid.

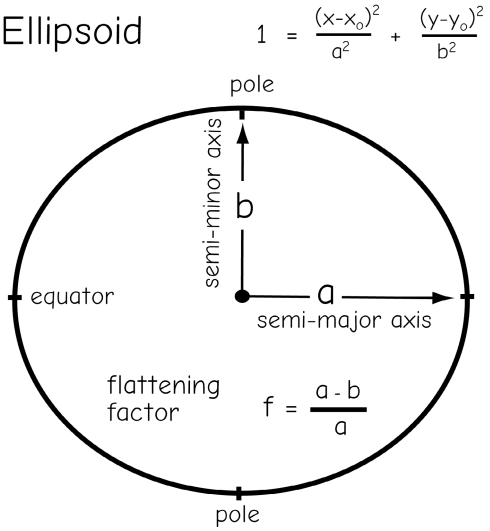


Figure 2-12: Ellipsoidal shape, a spherical figure with unequal semi-major and semi-minor axes.

The best estimates of the Earth's radii, a and b, have evolved as measurements systems have improved. Today, the best estimate for a is 6,378,137.0 meters (m), and for b 6,356,752.3 m. When a simple spheroidal shape is assumed where a equals b, some number between these two is usually applied, often the mean value of 6,367,444.7 m.

Converting Arc to Surface Distances

At times we need to calculate the distance on the surface of the Earth that is spanned by an arc measure. For example, I might have two sets of coordinates that differ by 10 seconds of arc, and wish to estimate the distance between them. We can approximate the surface distance on a circle or sphere by the formula:

$$d = r \cdot \theta \qquad (2.2)$$

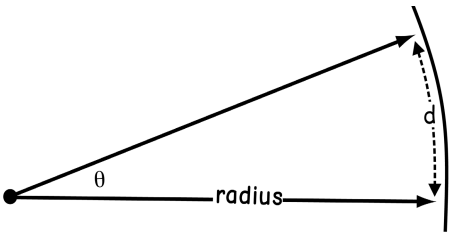
where d is the approximate ground distance, r is the radius of the circle or sphere, and θ is the angle of the arc. Note that the angle is typically specified in radian measure, defined as 2π radians per the 360 degrees in a complete circle, or approximately 57.2957795 degrees per radian. Figure 2-13 shows an example calculation of arc length, using the average earth radius.

This is an approximate formula because it assumes a perfectly spherical earth, with equal semi-major and semi-minor axes. As described earlier, the true shape is better approximated as an ellipsoid. The error in the calculated surface distance is typically less than the flattening factor, or 1 part in 300, so the spherical formula provides an acceptable estimate in many applications.

Also note that this formula applies to a generic arc angle, measured without regard to the latitude/longitude system. Substituting latitude values will result in a reasonably

accurate answer, but substituting longitude values anywhere but along the equator will result in an error, with larger errors near the poles, because longitude values converge at the poles. The formula is best used as a first approximation of distance spanning generic surface angles, not longitude.

The great circle distance should be used to estimate the surface distance between two points with known latitude/longitude (Figure 2-14). A *great circle* is defined as any line resulting from the intersection of a plane passing through the center of a globe. The equator and meridians are great circles, while lines of equal latitude other than the equator are not great circles. A great circle distance is the shortest path on the surface of the Earth between two points, and long-distance airline routes approximate great circles. As with all trigonometric formulas, you should know if your calculations expect degree or radian measures as input, and convert accordingly. If a spreadsheets by default uses radians in trigonometric functions, degrees must be divided by 57.2957 (converted to radians) prior to calculation (again, see Figure 2-13).



$d = radius \bullet \theta$

where θ is measured in radians,
with
1 radian = 57.2957^{o}

Given an Earth radius of 6,378,137m, how much distance is spanned by 10" of arc?

Arc= $10''/3600''/1^{o} = 0.00277778^{o}$

$=0.00277778^{o}/57.2957$ degrees/radian
$= 0.000048481435$ radians

$d = 6378137m \bullet 0.000048481435$
$= 309.2$ meters

Figure 2-13: Example calculation of the approximate surface distance spanned by an arc.

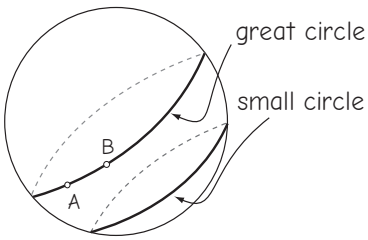
Great Circle Distance

Spherical approximation

Consider two points on the Earth's surface,

A with latitude, longitude of (ϕ_A, λ_A), and

B, with latitude, longitude of (ϕ_B, λ_B)



The great circle distance between points on a sphere is given by the formula:

$d = r \cdot \cos^{-1}[(\sin(\phi_A) \cdot \sin(\phi_B) + \cos(\phi_A)\cos(\phi_B) \cdot \cos(\lambda_A - \lambda_B)]$,

where d is the shortest distance on the surface of the Earth from A to B, and r is the Earth's radius, approximately 6378 km.

As an example, the distance between Paris, France, and Seattle, USA, is:

Latitude, longitude of Paris, France = 48.864716^{o}, 2.349014^{o}
Latitude, longitude of Seattle, USA = 47.655548^{o}, -122.30320^{o}

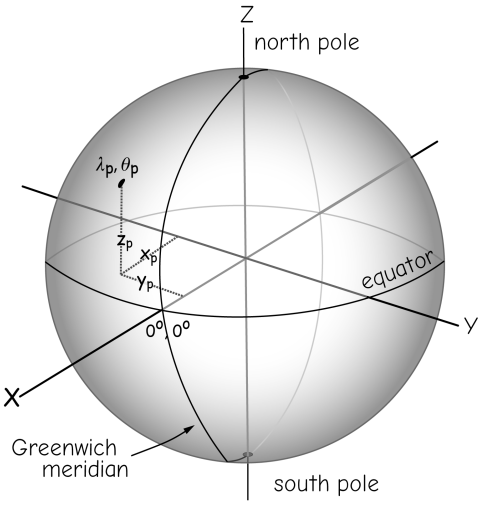
$d = 6378 \cdot \cos^{-1}[(\sin(48.864716) \cdot \sin(47.655548) +$
$\cos(48.864716) \cdot \cos(47.655548) \cdot \cos\{2.349014 - (-122.30320)\}]$
$= 8,043.6558km$

Figure 2-14: Calculation of the great circle distance between points.

Conversion from Geographic to 3-Dimensional Cartesian Coordinates

We sometimes must convert between latitude/longitude coordinates and a 3-D Cartesian coordinate system (Figure 2-15). This Cartesian system is aligned with the Z axis through the geographic North Pole, and the X and Y axes forming a plane on the equator. Mathematical formulas allow us to calculate any X, Y, and Z given any latitude, longitude, and Earth radii. These formulas are commonly used by geodesists in the most precise surveys, but are also embedded in many softwares that convert between different versions of our coordinate data.

There are two different sets of equations, one assuming a spherical Earth, and a more accurate assuming an ellipsoidal Earth. A detailed discussion of these is best left for an advanced course, and so formulas are included in Appendix C for reference.



longitude, latitude from known 3-D Cartesian

$$\lambda_p, \theta_p = F(x_p, y_p, z_p)$$

3-D Cartesian from known latitude, longitude

$$x_p, y_p, z_p = G(\lambda_p, \theta_p)$$

Figure 2-15: Formulas exist to convert between known spherical geographic coordinates (latitude and longitude on a spheroid) and corresponding 3-D Cartesian coordinates (see appendix C).

Geographic and Magnetic North

There is often confusion between magnetic north and geographic north. Magnetic north and the geographic north do not coincide (Figure 2-16). Magnetic north is the location towards which a compass points. The geographic North Pole is the average northern location of the Earth's axis of rotation. If you were standing on the geographic North Pole with a compass, it would point approximately in the direction of northern Canada, towards magnetic north some 600 kilometers away.

Because magnetic north and the geographic North Pole are not in the same place, a compass does not point at geographic north when observed from most places on Earth. The compass will usually point east or west of geographic north, defining an angular difference in direction to the poles. This angular difference is called the magnetic *declination* and varies across the globe, and also has varied through time. The specifica-

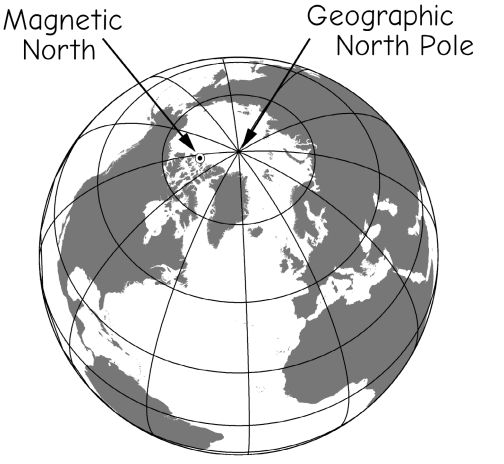


Figure 2-16: Magnetic north and the geographic North Pole.

tion of spherical coordinate systems is always in reference to the geographic North Pole, not magnetic north.

Note that our definition of geographic north is the average location of the Earth's axis of rotation. We say average because the Earth "wobbles," or nutates, on its axis. This means the axis location varies slightly, within a circle about 9 meters (30 feet) across, so the apparent North pole "wanders" within this circle. The nutation has a period of 433 days, with the pole returning back to its original location over that time. The nutation is different than the annual 23.5 degree tilt across the equinoxes.

Attribute Data and Types

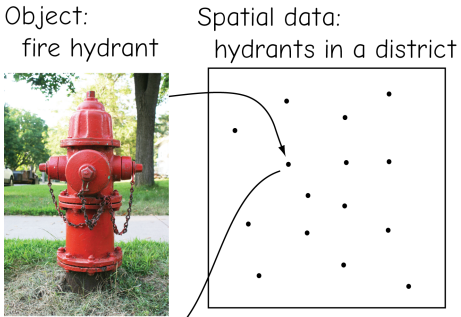
Attribute data are used to record the nonspatial characteristics of an entity. Attributes, also called *items* or *variables*, may be envisioned as a list of characteristics that describe the features we represent in a GIS. Color, depth, weight, owner, vegetation type, or land use are examples of variables that may appear as attributes. Attributes have values; for example, a fire hydrant may be colored red, yellow, or orange, have 1 to 4 flanges, and a rating of high, medium, or low.

As noted in Chapter 1, a finite number of characteristics is selected to represent a set of features. Sometimes only one attribute is stored for each location, for example, a simple layer that contains smog concentration in the atmosphere. Attributes in a city layer may include the population, total municipal area, the name of the current mayor, and the county and state containing the city. These attributes may be sufficient for some of the intended uses of the layer, but there are many other attributes that may be used to characterized a city. The number and type of attributes stored for a feature are by definition a subset of those possible, and there is no set that is universally superior to other potential sets. The set of attributes reflects the needs of the data developer and users, and should be carefully assessed for any data layer prior to development of a layer or its use in analysis.

Attributes are often presented in tables and arranged in rows and columns (Figure 2-17). Each row corresponds to a spatial object, and each column corresponds to an attribute. Tables are often organized and managed using a specialized computer program called a database management system (DBMS, described more fully in Chapter 8).

Attributes may be in many forms, but all attributes can be categorized as nominal, ordinal, or interval/ratio attributes.

Nominal attributes are variables that provide descriptive information about an object. The color is recorded for each hydrant in Figure 2-17. Other examples of nominal data are vegetation type, a city name, the owner of a parcel, or soil series. There is no implied order, size, or quantitative information contained in nominal attributes.



ID	Flanges	Color	Rating
1	3	red	high
2	3	red	high
3	2	green	low
4	3	red	mid
.	.	.	.

Figure 2-17: Attributes are typically envisioned as arranged by columns and rows, with objects arranged in rows, and attributes aligned in columns.

Nominal attributes may also be images, film clips, audio recordings, or other descriptive information, for example, GIS for real estate often have images of the buildings as part of the database. Image, video, or sound recordings stored as attributes are sometimes referred to as "BLOBs" for *binary large objects*.

Ordinal attributes imply a ranking or order by their values. An ordinal attribute may be descriptive, such as high, mid, or low, or it may be numeric; for example, an erosion class may be given a value from 1 to 10. The order reflects only rank, and not the scale. An ordinal value of four has a higher rank than a value of two, but we can't infer that the attribute value is twice as large, because we can't assume the scale is linear.

Interval/ratio attributes are used for numeric items where both rank order and absolute difference in magnitudes are represented, for example, the number of flanges in the second column of Figure 2-17. These data are often recorded as real numbers on a linear scale. Area, length, weight, height, or depth are a few examples of attributes that are represented by interval/ratio variables.

Items have a *domain*, a range of values they may take. Colors might be restricted to red, yellow, and green; cardinal direction to north, south, east, or west; and size to all positive real numbers.

Common Spatial Data Models

Spatial data models begin with a conceptualization. Consider a road map suitable for use at a statewide or provincial level. This map is based on a conceptualization that defines roads as lines. These lines connect cities and towns that are shown as discrete points or polygons on the map. Road properties may include only the road type, for example, an interstate highway, county road, or some other type of road. The roads have a width represented by the drawing symbol on the map, however, this width, when scaled, may not represent the true road width. This conceptualization identifies each road as a linear feature that fits into a small number of categories. All state highways are represented equally although they may vary. Some may have wide shoulders, others not, or dividing barriers of concrete, versus a broad vegetated median. We realize that differences exist within our conceptualization.

There are two main conceptualizations used for digital spatial data. The first conceptualization defines discrete objects using a *vector data model.* Vector data models use discrete elements such as points, lines, and polygons to represent the geometry of real-world entities (Figure 2-18).

Farm fields, roads, wetlands, cities, and census tracts are examples of entities that are often represented by discrete objects. Points are often used to define the locations of "small" objects such as wells, buildings, or ponds. Lines may be used to represent linear objects, for example, rivers or roads, or to identify the boundary between what is a part of the object and what is not a part of the object. We may map landcover for a region of interest, and we categorize discrete areas as a uniform landcover type. A forest may share an edge with a pasture, and this boundary is represented by lines. The boundaries between two polygons may not be discrete on the ground. For example, a forest edge may grade into a mix of trees and grass, then to pasture; however in the vector conceptualization, a line between two landcover types will be drawn to indicate a discrete, abrupt transition. Lines and points have coordinate locations, but points have no dimension, and lines have no dimension perpendicular to their direction. Area features may be defined by a closed, connected set of lines.

The second common conceptualization identifies and represents grid cells for a given region of interest. This conceptualiza-

tion employs a *raster* data model (Figure 2-18). Raster cells are arrayed in a row and column pattern to provide "wall-to-wall" coverage of a study region. Cell values are used to represent the type or quality of mapped variables. The raster model is used most commonly with variables that may change continuously across a region. Elevation, mean temperature, slope, average rainfall, cumulative ozone exposure, or soil moisture are examples of phenomena that are often represented as continuous fields. Raster representations are also sometimes used to represent discrete features, for example, class maps of vegetation or political units.

Data models are at times interchangeable in that many phenomena may be represented with either the vector or raster approach. For example, elevation may be represented as a surface (continuous field) or as a series of lines representing contours of equal elevation (discrete objects). Data may be converted from one conceptual view to another; for example, the location of contour lines (lines of equal elevation) may be determined by evaluating the raster surface, or a raster data layer may be derived from a set of contour lines. These conversions entail some costs both computationally and perhaps in data accuracy.

The decision to use either a raster or vector conceptualization often depends on the most frequent operations performed. Slope is more easily determined when elevation is represented in a raster data set. However, discrete contours are often the preferred format for printed maps, so the discrete conceptualization of a vector data model may be preferred for this application. The best data model for a given application depends on the most common operations, the experiences and views of the GIS users, the form of available data, and the influence of the data model on data quality.

Other, less common data models are sometimes used. A triangulated irregular network (TIN) is an example of such a data model, employed to represent surfaces, such as elevations, through a combination of point, line, and area features. Variants or other representations related to raster data models also exist. We will introduce and discuss less common data models later in this and other chapters.

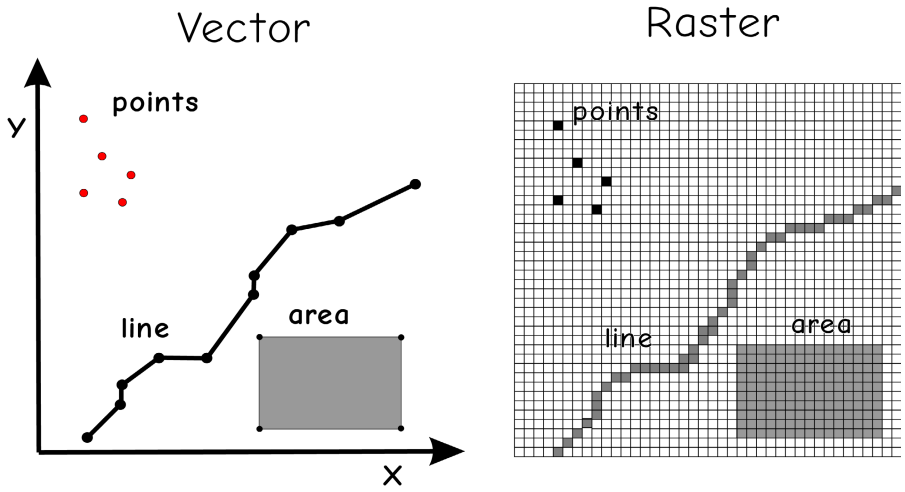


Figure 2-18: Vector and raster data models.

Vector Data Models

A vector data model uses sets of coordinates and associated attribute data to define discrete objects. Groups of coordinates define the location and boundaries of discrete objects, and these coordinate data plus their associated attributes are used to create vector objects representing the real-world entities (Figure 2-19).

There are three basic types of vector objects: points, lines, and polygons (Figure 2-20). A point uses a single coordinate pair to represent the location of an entity that is considered to have no dimension. Gas wells, light poles, accident location, and survey points are examples of entities often represented as point objects. Some of these have real physical dimension, but for the purposes of the GIS users they may be represented as points. In effect, this means the size or dimension of the entity is not important, only its location.

Attribute data are attached to each point, and these attribute data record the important nonspatial characteristics of the point entities. When using a point to represent a light pole, important attribute information might be the height of the pole, the type of light and power source, and the last date the pole was serviced.

Linear features, often referred to as lines or *arcs*, are represented as lines when using vector data models. Lines are most often represented as an ordered set of coordinate pairs. Each line is made up of line segments that run between adjacent coordinates in the ordered set (Figure 2-20). A long, straight line may be represented by two coordinate pairs, one at the start and one at the end of the line. Curved linear entities are most often represented as a collection of short, straight, line segments, although curved lines are at times represented by a mathematical equation describing a geometric shape. Lines typically have a starting point, an ending point, and intermediate points to represent the shape of the linear entity. Starting points and ending points for a line are sometimes referred to as *nodes*, while intermediate points in a line are referred to as *vertices* (Figure 2-20). Attributes may be attached to the whole line, line segments, or to nodes and vertices along the lines.

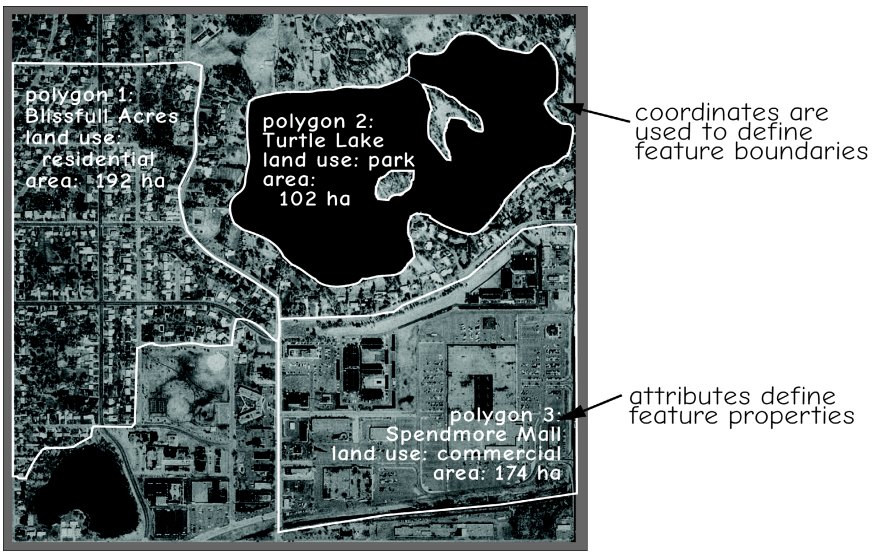


Figure 2-19: Coordinates define spatial location and shape. Attributes record the important non-spatial characteristics of features in a vector data model.

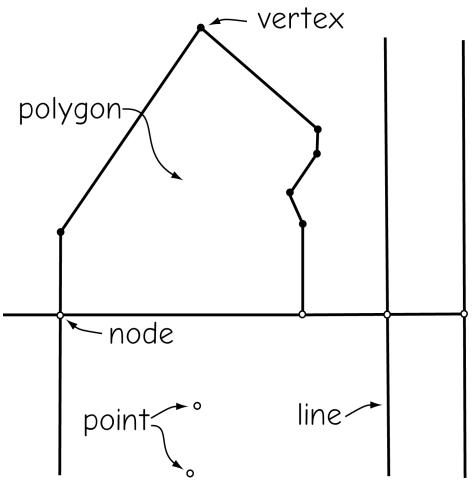


Figure 2-20: Points, nodes, and vertices define points, line, and polygon features in a vector data model.

Area entities are most often represented by closed polygons. These polygons are formed by a set of connected lines, either one line with an ending point that connects back to the starting point, or as a set of lines connected start-to-end. Polygons have an interior region and may entirely enclose other polygons in this region. Polygons may be adjacent to other polygons and thus share "bordering" or "edge" lines with other polygons. Attribute data such as area, perimeter, landcover type, or county name may be linked to each polygon.

Note that there is no uniformly superior way to represent features. Some feature types may appear to be more "naturally" represented one way: manhole covers as points, roads as lines, and parks as polygons. However, in a very detailed data set, the manhole covers may be represented as circles, and both edges of the roads may be drawn and the roads represented as polygons. The representation depends as much on the detail, accuracy, and intended use of the data set as our common conception or general shape of the objects.

A single set of features may be represented differently, depending on the interests and purposes of the GIS users (Figure 2-21). A point layer may be chosen when general feature position is needed, for example, general building location (Figure 2-21, top). Other users may be interested in the outline of the feature and so require representation by lines, while polygon representations may

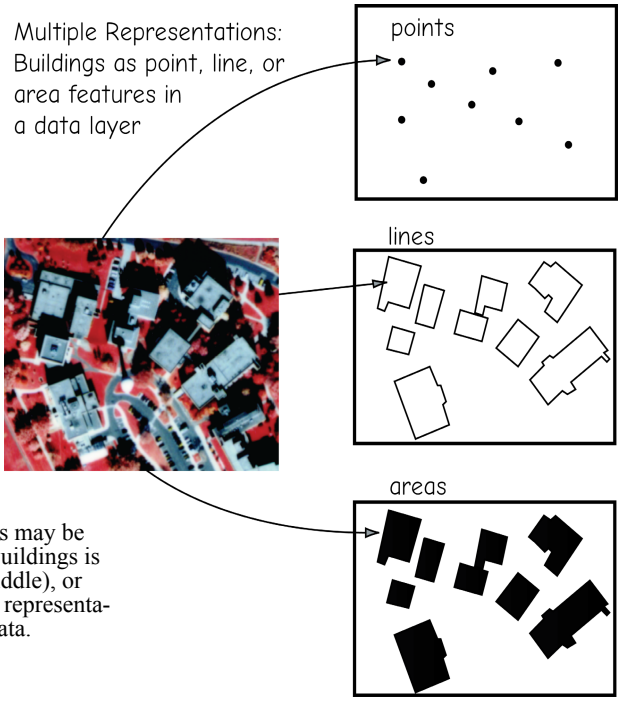


Figure 2-21: Alternate conceptualizations may be used to represent features. Here a set of buildings is represented by either point (top), line (middle), or polygon features (bottom). The preferred representation depends on the intended use of the data.

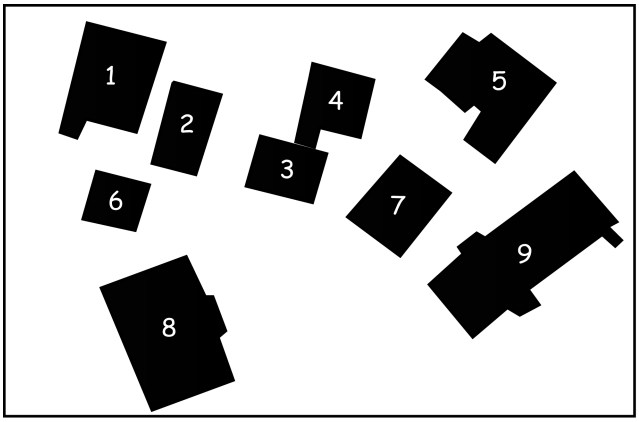
be preferred for other applications (Figure 2-21 mid and lower, respectively). Our intended use often determines our conceptual model and hence vector type used to represent a feature.

In the most common vector models, there is an attribute table associated with each vector layer, and a single row in the table corresponding to each feature in the data layer (Figure 2-22).This table holds the attributes of interest for each feature.

Each column holds an attribute value for a given feature. All values in a column have the same type, so for any given column, all entries might be ordinal, or interval ratio, or a BLOB, or some other defined type. An identifier value, or ID, is typically included, and this value is often unique within the table, with an unrepeated value assigned for each row and corresponding feature (Figure 2-22).

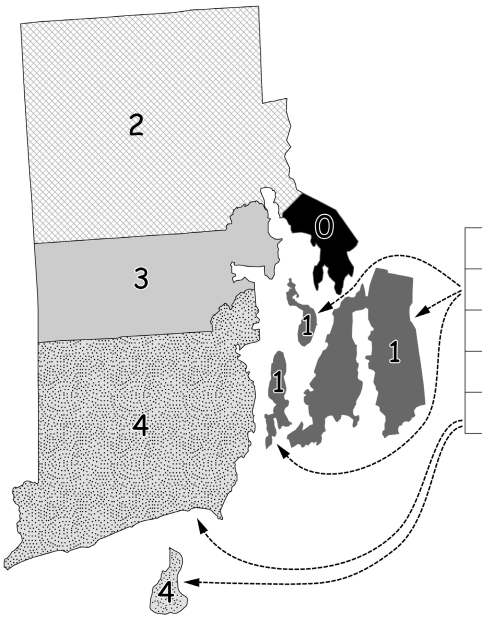
Vector layers sometimes have a "many to one" relationship between geographic features and table rows (Figure 2-23). In these instances, many spatially distinct features are matched with a row, and the row attributes apply to all the distinct features. This is common when representing islands, groups of buildings, or other clusters of features that make up a perceived "whole thing." These are sometimes referred to as *multipart features*, because multiple geographic objects may correspond to one row. .

Multipart features may also be used when there are extremely large volumes of data, for example, when millions of point



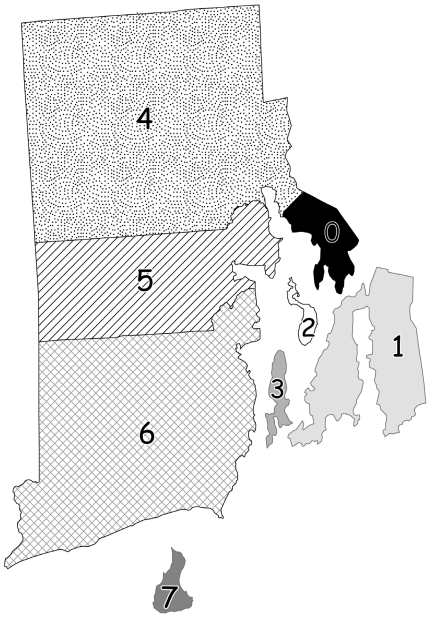
ID	Building Name	Floors	Roof Type
1	Hodson Hall	6.0	flat, sealed tar
2	Borlaug Hall	5.5	pitched 9/12, tile
3	Guilford Technology Bldg.	4.0	flat, gasket
4	Shop Annex	2.5	flat, sealed tar
5	Animal Sciences Bldg.	1.0	pitched 12/12, tile
6	Administration Bldg.	14.0	pitched 6/12, metal
7	Climate Sciences Center	6.0	flat, sealed tar
8	Grantham Tower	1.0	pitched, 9/12, tile
9	Biological Sciences Bldg.	9.0	pitched 12/12, tile

Figure 2-22: An example of the most common vector data model structure. Geographic features correspond to rows in a table with an identifier (ID) and a set of attributes arrayed in columns.



Multipart Features

ID	County	Area
0	Bristol	73.5
1	Newport	286.5
2	Providence	1,121.9
3	Kent	455.6
4	Washington	912.3



Single-part Features

ID	County	Area
0	Bristol	73.5
1	Newport	243.5
2	Newport	14.8
3	Newport	24.8
4	Providence	1,121.9
5	Kent	912.3
6	Washington	884.7
7	Washington	27.6

Figure 2-23: Example of multipart and single-part features. Here, counties for Rhode Island, a state in the Eastern U.S.A., are shown with one table entry for each county (top), with multipart features in a layer, and with one table entry for each distinct polygon (bottom), with only single-part features in the layer. Note that calculations, analysis, and interpretation may differ for multipart v.s. single-part features.

observations are collected automatically with laser scanners or similar devices. Tables are often slower to process than point features, and so reducing the table size by grouping into multi-part features may shorten many operations.

Care is warranted when converting multipart features to single-part features. The most common problems arise for aggregate variables in polygon layers, such as total counts. For example, population data are often delivered by census areas such as states. Many states, such as Hawaii, have several parts and are represented by a multipart shape. The population is associated with the aggregated set of polygons corresponding to a state (Figure 2-24). When converted to single-part shapes, the attributes are often copied for each component polygon. In our example all single-part polygons will be assigned the attribute values for the multipart feature, in effect repeating counts for each part. Each polygon will have the population count associated with it, so any aggregation or calculation based on population will likely be in error, sometimes substantially so.

Attributes for converted shapes may be corrected. If component data are available, they can be assigned to each of the single-part features. If not, then some weighting scheme may be available, for example, if there is a correlation between area and count. Until they are reviewed and appropriately adjusted, single-part attributes derived from multipart features should be used with caution.

Polygon Inclusions and Boundary Generalization

Vector data frequently exhibit two characteristics, polygon inclusions and boundary generalization. These characteristics are oft-ignored, but may affect the use of vector data, occasionally with dire consequences. These concepts must be understood, their presence evaluated, and effects weighed in the use of vector data sets.

Polygon inclusions are areas in a polygon that are different from the rest of the polygon, but still part of the polygon. Inclusions occur because we typically assume an area represented by a polygon is homogeneous, but this assumption may be wrong, as illustrated in Figure 2-25. The figure shows a vector polygon layer representing raised landscaping beds (a). The general attributes for the polygon may be coded; for example, the surface type may be recorded as cedar mulch. The area noted in Figure 2-25b shows a walkway that is an inclusion in a raised bed. This walkway has a concrete sur-

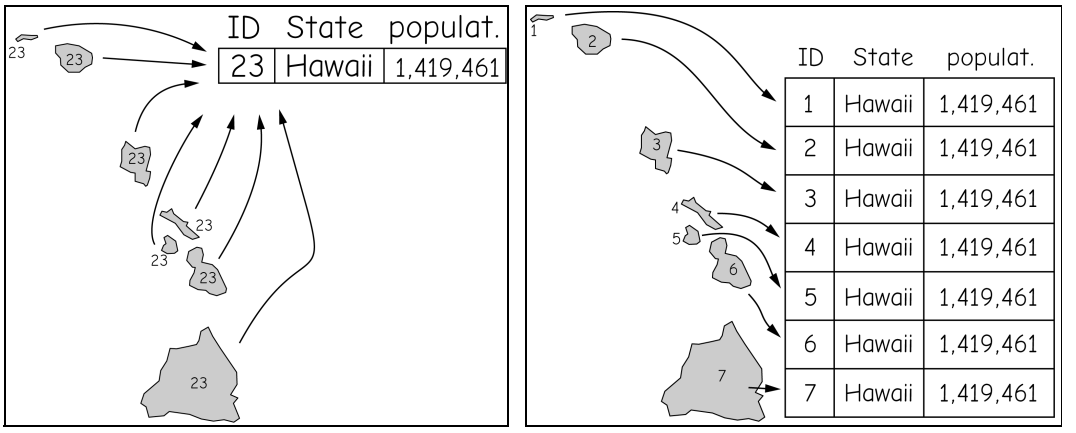


Figure 2-24: Multipart to single-part conversion may lead to errors in subsequent analysis because attributes may be copied from the original, multi-part cluster (left, above), to each single-part component (above right). Density, sums, or other derived variables often should be re-calculated for single-part features, but often are not, resulting in errors.

face. Hence, this walkway is an unresolved inclusion within the polygon.

One solution creates a polygon for each inclusion. This often is not done because it may take too much effort to identify and collect the boundary location of each inclusion, and there typically is some lower limit, or *minimum mapping unit*, on the size of objects we care to record in our data. Inclusions are present in some form in many polygon data layers.

Boundary generalization is the incomplete representation of boundary locations. This problem stems from the typical way we represent linear and area features in vector data sets. As shown in Figure 2-25c, polygon boundaries are represented as a set of connected, short, straight-line segments. The segments are a means to trace the position of line features, or the boundaries separating area features. For curved lines, these straight line segments may be viewed as a sampling of the true curve, and there is typically some deviation of the line segment from the "true" curved boundary. The amount of generalization depends on many factors, and should be so small as to be unimportant for any intended use of the spatial data. However, since many data sets may have unforeseen uses, or may be obtained from a third party, the boundary generalization should be recognized and evaluated relative to the specific requirements of any given spatial analysis. There are additional forms of generalization in spatial data, and these are described more thoroughly in Chapter 4.

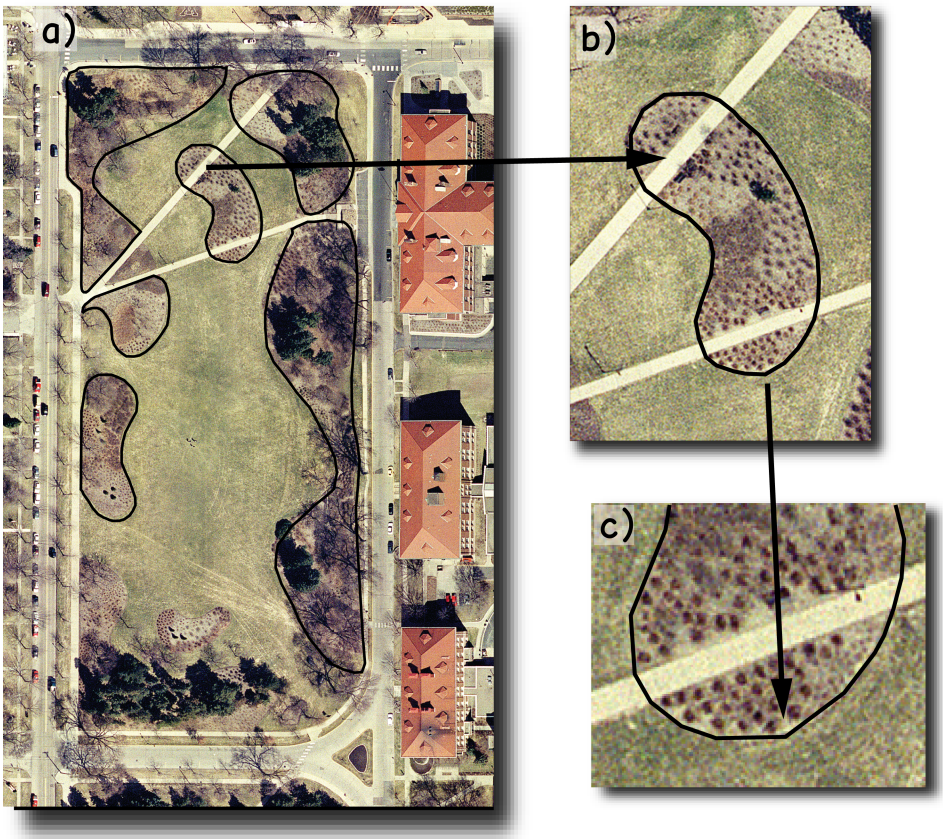


Figure 2-25: Examples of polygon inclusions (sidewalk inclusion in flower bed shown in a and b), and boundary generalization (c) in a vector data model. These approximations typically occur as a consequence of adopting a vector representation, and their impacts must be considered when using vector data.

Vector Topology

Vector data often contain *vector topology,* enforcing strict connectivity and recording adjacency and planarity. Early systems employed a spaghetti data model (Figure 2-26a), in which lines may not intersect when they should, and may overlap without connecting. The spaghetti model severely limits spatial data analysis and is little used except for very basic data entry or translation. Topological models create an intersection and place a node at each line crossing, record connectivity and adjacency, and maintain information on the relationships between and among points, lines, and polygons in spatial data. This greatly improves the speed, accuracy, and utility of many spatial data operations.

Topological properties are conserved when converting vector data among common coordinate systems, a common practice in GIS analysis (described in Chapter 3). Polygon adjacency is an example of a topologically invariant property, because the list of neighbors for any given polygon does not change during geometric stretching or bending (Figure 2-26, b and c). These relationships may be recorded separately from the coordinate data.

Topological vector models may vary, and enforce particular types of topological relationships. *Planar topology* requires that all features occur on a two-dimensional surface. There can be no overlaps among lines or polygons in the same layer (Figure 2-27). When planar topology is enforced, lines may not cross over or under other lines. At each line crossing there must be an intersection.

The left side of Figure 2-27 shows nonplanar graphs. In the top left figure, four line segments coincide. At some locations the lines intersect at a node, shown as white-filled circles, but at some locations a line passes over or under another line segment. These lines are nonplanar. The top right of Figure 2-27 shows planar topology enforced for these same four line segments. Nodes are found at each line crossing.

Polygons can also be nonplanar, as shown at the bottom left of Figure 2-27. Two polygons overlap slightly at an edge. This may be due to an error; for example, the two polygons share a boundary but have been recorded with an overlap, or there may be two areas that overlap in some way. If topological planarity is enforced, these two polygons must be resolved into three separate, nonoverlapping polygons. Nodes are placed

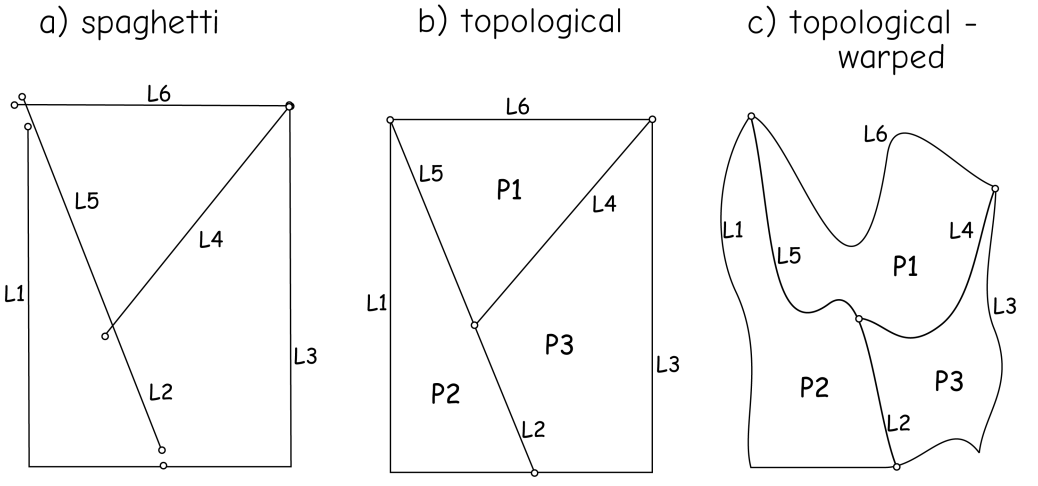


Figure 2-26: Spaghetti (a), topological (b), and topological warped (c) vector data. Figures b and c are topologically identical because they have the same connectivity and adjacency.

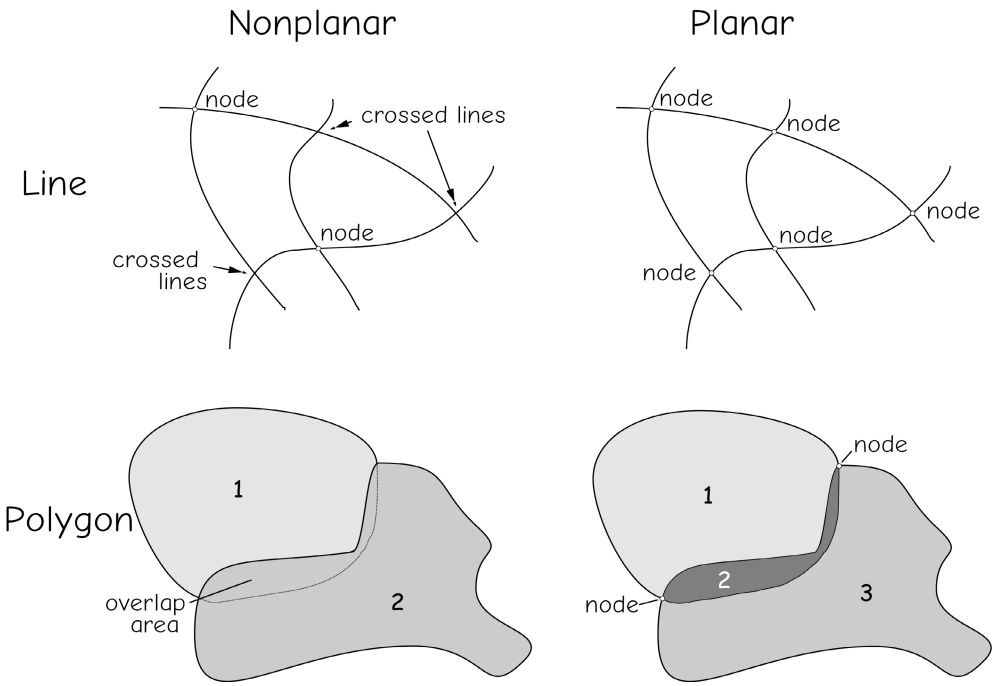


Figure 2-27: Nonplanar and planar topology in lines and polygons.

at the intersections of the polygon boundaries (lower right, Figure 2-27).

There are additional topological constructs besides planarity that may be specified. For example, polygons may be exhaustive, in that there are no gaps, holes, or "islands" allowed. Line direction may be recorded, so that a "from" and "to" node are identified in each line. Directionality aids the representation of river or street networks, where there may be a natural flow direction.

There is no single, uniform set of topological relationships that are included in all topological data models. Different vendors have incorporated different topological information in their data structures. Planar topology is often included, as are representations of *adjacency* (which polygons are next to which) and *connectivity* (which lines connect to which).

Some GIS software create and maintain detailed topological relationships in their data. This results in more complex and perhaps larger data structures, but access is often faster, and topology provides more consistent, "cleaner" data. Other systems maintain little topological information in the data structures, but compute and act upon topology as needed during specific processing.

Topology may also be specified between layers, because we may wish to enforce spatial relationships between entities that are stored separately. As an example, consider a data layer that stores property lines (cadastral data), and a housing data layer that stores building footprints (Figure 2-28). Rules may be specified that prevent polygons in the housing data layer from crossing property lines in the cadastral data layer. This would indicate a building that crosses a property line. Most such instances occur as a result of small errors in data entry or misalignment among data layers. Topological restrictions between two data layers avoid these inconsistencies. Exceptions may be

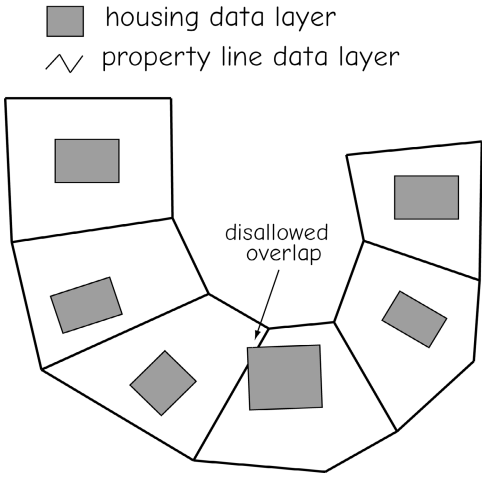


Figure 2-28: Topological rules may be enforced across data layers. Here, rules may be specified to avoid overlap between objects in different layers.

granted in those few cases when a building truly does cross property lines.

There are many other types of topological constraints that may be enforced, both within and between layers. *Dangles*, lines that do not connect to other lines, may be proscribed, or limited to be greater or less than some threshold length. Lines and points may be required to coincide, for example, water pumps as points in one data layer and water pipes as lines in another, or lines in separate layers may be required to intersect or be coincident. While these topological rules add complexity to vector data sets, they may also improve the logical consistency and value of these data.

Topological vector models often use codes and tables to record topology. As described above, nodes are the starting and ending points of lines. Each node and line is given a unique identifier. Sequences of nodes and lines are recorded as a list of identifiers, and point, line, and polygon topology recorded in a set of tables. The vector features and tables in Figure 2-29 illustrate one form of this topological coding.

Many GIS software systems are written such that the topological coding is not visible to users, nor directly accessible by them. Tools are provided to ensure the topology is created and maintained, that is, there may be directives that require that polygons in two layers do not overlap, or to ensure planarity for all line crossings. However, the topological tables these commands build are often quite large, complex, and linked in an obscure way, and therefore hidden from users.

Point topology is often quite simple. Points are typically independent of each other, so they may be recorded as individual identifiers, perhaps with coordinates included, and in no particular order (Figure 2-29, top).

Line topology typically includes substantial structure and identifies at a minimum the beginning and ending points of each line (Figure 2-29, middle). Variables record the topology and may be organized in a table. These variables may include a line identifier, the starting node, and the ending node for each line. In addition, lines may be assigned a direction, and the polygons to the left and right of the lines recorded. In most cases left and right are defined in relation to the direction of travel from the starting node to the ending node.

Polygon topology may also be defined by tables (Figure 2-29, bottom). The tables may record the polygon identifiers and the list of connected lines that define the polygon. Edge lines are often recorded in sequential order. The lines for a polygon form a closed loop and thus, the starting node of the first line in the list also serves as the ending node for the last line in the list. Note that there may be a “background” polygon defined by the outside area. This background polygon is not a closed polygon like all the rest; however it may be defined for consistency and to provide entries in the topology tables.

Topological vector models greatly enhance many vector data operations. Adjacency analyses are reduced to a “table look-up”, a quick and easy operation in most software systems. For example, an analyst may want to identify all polygons adjacent to a city. Assume the city is represented as a sin-

gle polygon. Adjacency analysis reduces to 1) scanning the polygon topology table to find the polygon labeled "city" and reading the list of lines that bound the polygon, and 2) scanning this list of lines for the city polygon, accumulating a list of all left and right polygons. Polygons adjacent to the city may be identified from this list. List searches on topological tables are typically much faster than searches involving coordinate data.

Topological vector models also enhance many other spatial data operations. Network and other connectivity analyses are concerned with the flow of resources through defined pathways. Topological vector models explicitly record the connections of a set of pathways and so facilitate network analyses. Overlay operations are also enhanced when using topological vector models. The mechanics of overlay operations are discussed in greater detail in Chapter 9; however, we will state here that they involve identifying line adjacency, intersection, and resultant polygon formation. The interior and exterior regions of existing and new polygons must be determined, and these regions depend on polygon topology. Hence, topological data are useful in many spatial analyses.

Topological data models often have an advantage of smaller file sizes, largely because coordinate data are recorded once. For example, a nontopological approach often stores polygon boundaries twice. Lines 52 and 53 at the bottom of Figure 2-29 will be recorded for both polygon A and polygon B. Long, complex boundaries in polygon data sets may double their size. This increases both storage requirements and processing.

There are limitations and disadvantages to topological vector models. First, there are

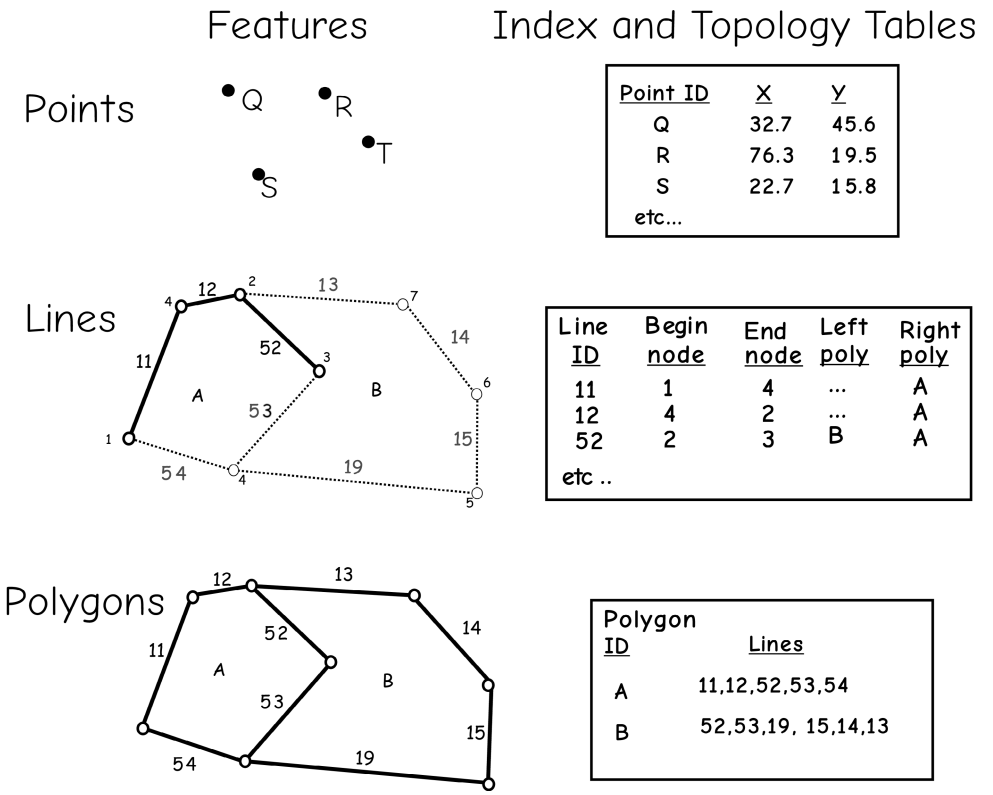


Figure 2-29: An example of vector features and corresponding topology tables. Information on the adjacency, connectivity, and other spatial relationships may be stored in topology tables, and joined to features by indices, here represented by values in the ID columns.

computational costs in defining the topological structure of a vector data layer. Software must determine the connectivity and adjacency information, assign codes, and build the topological tables. Computational costs are typically quite modest with current computer technologies.

Second, the data must be very "clean", in that all lines must begin and end with a node, all lines must connect correctly, and all polygons must be closed. Unconnected lines or unclosed polygons will cause errors during analyses. Significant human effort may be required to ensure clean vector data because each line and polygon must be checked. Software may help by flagging or fixing "dangling" nodes that do not connect to other nodes, and by automatically identifying all polygons. Each dangling node and polygon may then be checked, and edited as needed to correct errors.

Limitations and the extra editing are far outweighed by the gains in efficiency and analytical capabilities provided by topological vector models. Many current vector GIS packages use topological vector models in some form.

Vector Features, Tables, and Structures

Topological vector models are commonly used to define spatial features in a data layer. As we described earlier in this chapter, geographic features are associated with nonspatial attributes. Typically a table is used to organize the attributes, and there is a linkage between rows in the attribute table and the spatial components of a data layer (Figure 2-30, top). In most GIS software, we can most easily view the tables and a graphic representation of the spatial data as a linked table and digital map (Figure 2-30, top).

There is commonly a *one-to-one linkage* between each entry in the attribute table and each feature in the data layer. This means for each feature in the data layer there is one and only one entry in the attribute table. Occasionally there may be layers with a many-to-one relationship between attribute table entries and multiple features in a data layer.

Most GIS employ underlying file structures to organize components of the spatial data. An example organization is shown in the bottom half of Figure 2-30, where the topological elements are recorded in a linked set of tables, here, one each for the polygons, lines, and nodes and vertices. Most GIS maintain the spatial and topological data as a single or cluster of linked files. This internal file structure is often insulated from direct manipulation by the GIS user, but underlies nearly all spatial data manipulations. A user may directly edit or otherwise manipulate table values, usually with the exception of the ID, and the underlying topology and coordinate data are accessed via requests to display, change, or analyze the spatial data components. Data layers may also include additional information (not shown) on the origin, region covered, date of creation, edit history, coordinate system, or other characteristics of a data set.

Note that not all GIS store coordinate and topological data in non-tabular file structures. Coordinates, points, lines, polygons, and other composite features may be stored in tables similar to attribute tables. It is premature to discuss the details of these *spatially enabled* databases, because they are based on something called a *relational data model*, described in detail in Chapter 8. Faster computers support this generally more flexible approach, allowing simpler and more transparent access across different types of GIS software.

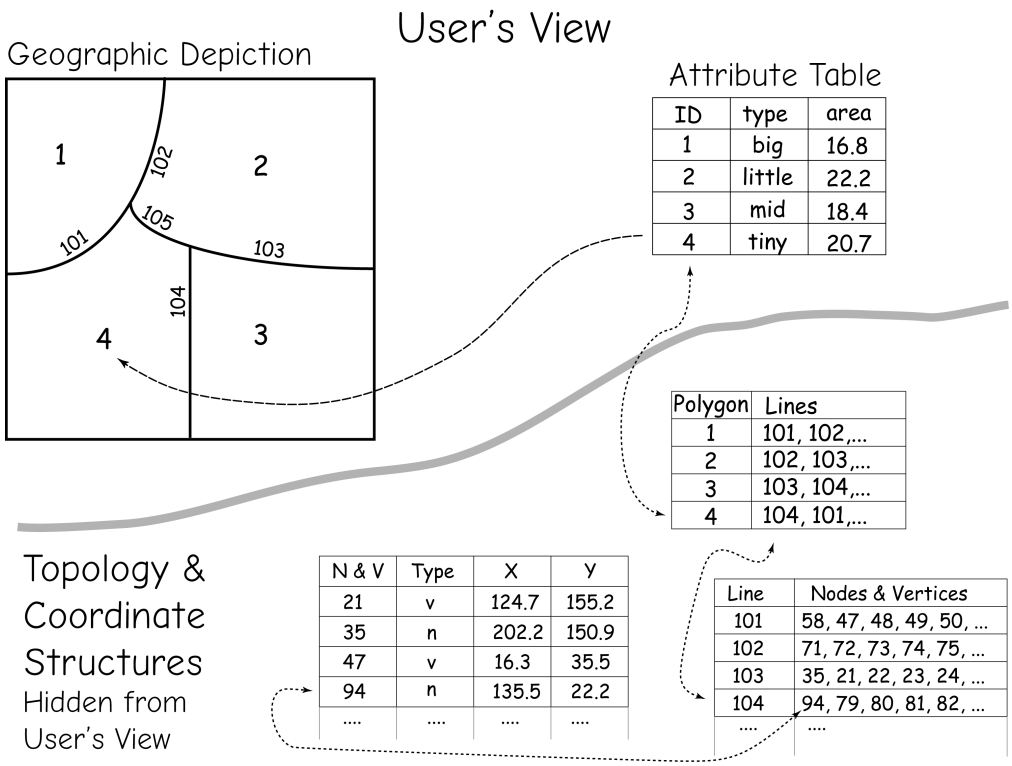


Figure 2-30: Features in a topological data layer typically have a one-to-one relationship with entries in an associated attribute table. The attribute table typically contains a column with a unique identifier, or ID, for each feature. Topology and coordinate data are often hidden from the user, but linked to the attribute and geographic features through pointers and index variables, described in the Data and File Structures section, later in this chapter.

Raster Data Models

Models and Cells

Raster data models define the world as a regular set of cells in a grid pattern (Figure 2-31). Typically these cells are square and evenly spaced in the x and y directions. The phenomena or entities of interest are represented by attribute values associated with each cell location.

Raster data models are the natural means to represent "continuous" spatial features or phenomena. Elevation, precipitation, slope, and pollutant concentration are examples of continuous spatial variables. These variables characteristically show significant changes in value over broad areas. The gradients can be quite steep (e.g., at cliffs), gentle (long, sloping ridges), or quite variable (rolling hills). Raster data models depict these gradients by changes in the values associated with each cell.

Raster data sets have a *cell dimension*, defining the edge length for each square cell (Figure 2-31). For example, the cell dimension may be specified as a square 30 meters on each side. The cells are usually oriented parallel to the x and y directions, and the coordinates of a corner location are specified.

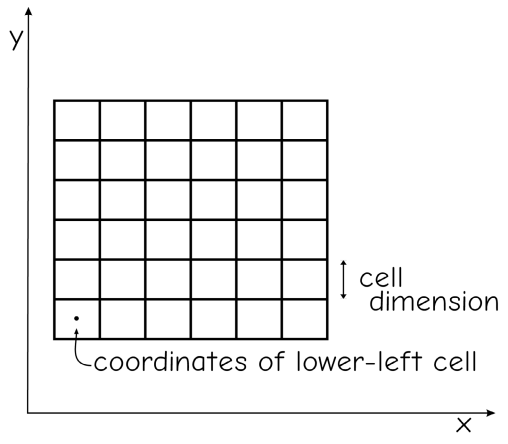


Figure 2-31: Important defining characteristics of a raster data model.

When the cells are square and aligned with the coordinate axes, the calculation of a cell location is a simple process of counting and multiplication. A cell location may be calculated from the cell size, known corner coordinates, and cell row and column number. For example, if we know the lower-left cell coordinate, all other cell coordinates may be determined by the formulas:

$$N_{cell} = N_{lower\text{-}left} + row * cell\ size \qquad (2.3)$$

$$E_{cell} = E_{lower\text{-}left} + column * cell\ size \qquad (2.4)$$

where N is the coordinate in the north direction (y), E is the coordinate in the east direction (x), and the `row` and `column` are counted starting with zero from the lower left cell.

There is often a trade-off between spatial detail and data volume in raster data sets. The number of cells needed to cover a given area increases four times when the cell size is cut in half (Figure 2-32). Smaller cells provide greater spatial detail, but at the cost of larger data sets.

The cell dimension also affects the spatial precision of the data set, and hence positional accuracy. The cell coordinate is usually defined at a point in the center of the cell. The coordinate applies to the entire area covered by the cell. Positional accuracy is typically expected to be no better than approximately one-half the cell size. No matter the true location of a feature, coordinates are truncated or rounded up to the nearest cell center coordinate. Thus, the cell size should be no more than twice the desired accuracy and precision for the data layer represented in the raster, and often it is specified to be smaller.

Each raster cell represents a given area on the ground and is assigned a value that may be considered to apply to the entire cell.

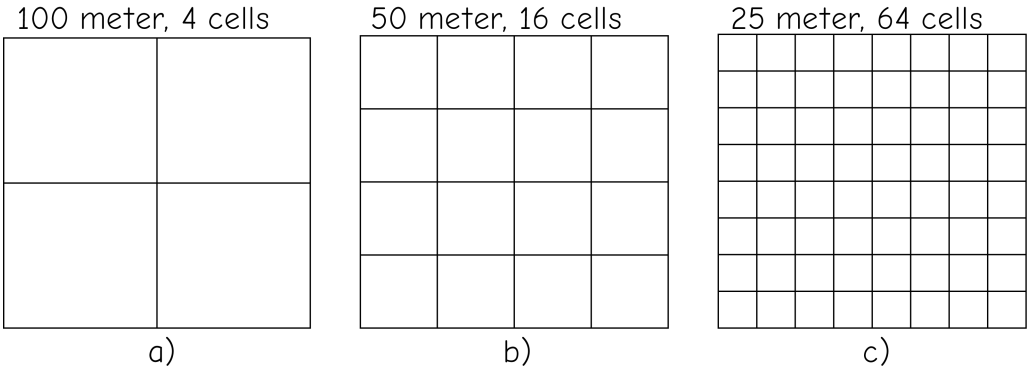


Figure 2-32: The number of cells in a raster data set depends on the cell size. For a given area, a linear decrease in cell size causes an exponential increase in cell number, e.g., halving the cell size causes a four fold increase in cell number.

If the variable is uniform across the raster cell, the value will be correct over the cell. However, under most conditions there is within-cell variation, and the raster cell value represents the average, central, or most common value found in the cell. Consider a raster data set representing annual weekly income with a cell dimension that is 300 meters (980 feet) on a side. Further suppose that there is a raster cell with a value of 710. The entire 300 by 300 meters area is considered to have this value of 710 pesos per week. There may be many households within the raster cell that do not earn exactly 710 pesos per week. However, the 710 pesos may be the average, the highest point, or some other representative value for the area covered by the cell. While raster cells often represent the average or the value measured at the center of the cell, they may also represent the median, maximum, or another statistic for the cell area.

An alternative interpretation of the raster cell applies the value to the central point of the cell. Consider a raster grid containing elevation values. Cells may be specified as 200 meters square, and an elevation value assigned to each square. A cell with a value of 8000 meters (26,200 feet) may be assumed to have that value at the center of the cell, but this value will not be assumed to apply to the entire cell.

A raster data model may also be used to represent discrete data (Figure 2-33), for example, to represent landcover in an area. Raster cells typically hold numeric or single-letter alphabetic characters. A coding scheme defines what land cover type the discrete values signify. Each code may be found at many raster cells.

Raster cell values may be assigned and interpreted in at least seven different ways (Table 2-1). We have described three: a raster cell as a point physical value (elevation), as a statistical value (average income), and as discrete data (landcover). Raster values may also be used to represent points and

a	a	a	a	r	f	f	a	a	a	a	a
a	a	a	a	r	f	f	a	a	a	a	a
a	a	a	f	r	f	f	a	a	a	a	a
a	a	a	r	r	f	f	a	a	a	a	a
a	a	a	r	f	f	f	a	a	a	a	a
a	f	f	r	f	f	f	a	a	a	a	a
a	f	f	r	f	u	f	a	a	a	a	a
h	h	h	h	h	h	h	h	h	h	h	h
f	f	r	u	u	u	u	a	a	a	a	a
f	f	r	f	u	u	a	a	a	a	a	a
f	f	f	r	f	f	a	a	a	a	a	a
f	f	f	f	r	f	a	a	a	a	a	a

a = agriculture u = developed
f = forest r = river
h = highways

Figure 2-33: Discrete or categorical data may be represented by codes in a raster data layer.

Table 2-1: Types of data represented by raster cell values (from L. Usery, pers. comm.).

Data Type	Description	Example
point ID	alpha-numeric ID of closest point	hospital
line ID	alpha-numeric ID of closest line	nearest road
contiguous region ID	alpha-numeric ID for dominant region	state
class code	alpha-numeric code for general class	vegetation type
table ID	numeric position in a table	row
physical analog	numeric value representing surface value	elevation
statistical value	numeric value from a statistical function	population density

lines, as the IDs of lines or points that occur closest to the cell center.

Point and line assignment to raster cells may be complicated when there are multiple features within a single cell. For example, when light poles are represented in a raster data layer, cell value assignment is straightforward when there is only one light in a cell (Figure 2-34, near A). When there are multiple poles in a single cell there is some ambiguity, or generalization in the assignment (Figure 2-34, near B). One common solution represents one feature from the group, and retains information on the attributes and characteristics of that feature. This entails some data loss. Another solution is to reduce the raster cell size so that there are no multiple features in a cell. This may result in impractically large data sets. More complex schemes may record multiple instances of features in a cell, but these then may slow access or otherwise decrease the utility that comes from the simple raster structure.

Similar problems may occur when there are multiple line segments within a raster cell, for example, when linear features such as roads are represented in a raster data set. When two or more roads meet, they will do so within a raster cell, and some set of attributes must be assigned (Figure 2-34*C*). Since attributes are assigned by cells, some precedence must be established, with one line given priority over others.

Raster cell assignment also may be complicated when representing what we typically think of as discrete, uniform areas. Consider the area in Figure 2-35. We wish to represent this area with a raster data layer,

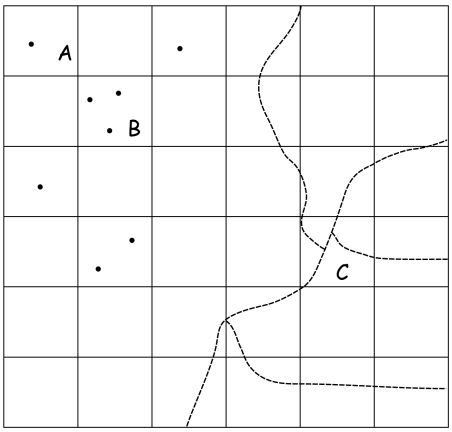


Figure 2-34: Raster cell assignment requires decisions when multiple objects occur in the same cell.

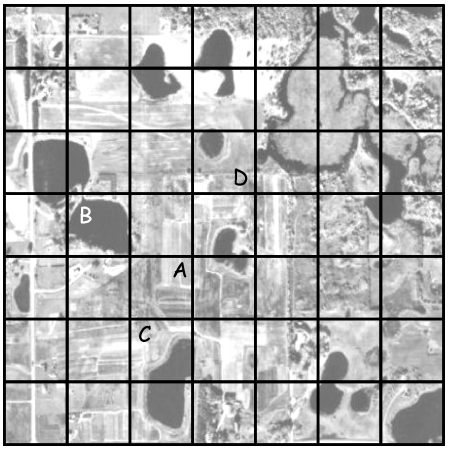


Figure 2-35: Raster cell assignment with mixed landscapes. Upland areas are lighter greys, water the darkest greys.

with cells assigned to one of two class codes, one each for land or water. Water bodies appear as darker areas in the image, and the raster grid is shown overlain. Cells may contain substantial areas of both land and water, and the proportion of each class may span from zero to 100 percent. Some cells are purely one class and the assignment is unambiguous; for example, the cell labelled A in the Figure 2-35 contains only land. Others are unambiguous, such as cell B (water) or D (land). Some are nearly equal in their proportion of land and water, as in cell C.

One common method to assign classes for mixed cells is called "winner-take-all". The cell is assigned the class of the largest area type. Cells A, C, and D would be assigned the land class, cell B the water class. Another option applies preference in cell assignment. If any of an "important" type is found, then the cell is assigned that value, regardless of the proportion. If we specify a preference for water, then cells B, C, and D in Figure 2-35 would be assigned the water type, and cell A the land type.

Regardless of the assignment method used, Figure 2-35 illustrates two phenomena when discrete objects are represented using a raster data model. First, some areas that are not the assigned class are included in some rasters cells. These "inclusions" are inevitable because cells must be assigned to a discrete class, the cell boundaries are rigidly assigned, and the class boundaries on the ground rarely line up with the cell boundaries. Some mixed cells occur in nearly all raster layers. The GIS user must acknowledge these inclusions, and consider their impact on the intended spatial analyses.

Second, differences in class assignment rules may substantially alter the data layer, as shown in our simple example. In more complex landscapes, there will be more potential cell types, which may increase the assignment sensitivity. Decreasing the raster cell size reduces the significance of classes in the assignment rule, but at the cost of increased data volumes.

The occurrence of more than one line or point within a raster cell can result in similar assignment problems. If two points occur, then which point ID is assigned? If two lines occur, then which line ID should be assigned? Some rule must be developed; for example, the point that falls nearest the center may be assigned, or the line with the longest segment within the raster cell. Similar to when area features are assigned to rasters, "inclusions" and dependence on the class assignment rules affect the output.

Raster Features and Attribute Tables

Raster layers may also have associated attribute tables. This is most common when nominal data are represented, but may also be used with ordinal or interval/ratio data. Just as with topological vector data, features in the raster layer may be linked to rows in an attribute table, and these rows may describe the essential nonspatial characteristics of the features.

Figure 2-36a and b show data represented in a raster model. Figure 2-36a shows a raster data set that maintains a one-to-one relationship between raster cells and in the data table. An additional column, cell-ID,

must be added to uniquely identify each raster location. The corresponding attributes IDorg, class, and area are repeated for each cell. Note that the area values are the same for all cells and thus all rows in the table.

A one-to-one correspondence is rarely used with raster data sets because it often would require an unmanageably large size of attribute table. This small example results in 100 rows for the attribute table, but we often use raster data sets with billions of cells. If we insist on a one-to-one cell/attribute relationship, the table may become too large. Even simple processes such as sorting, searching, or subsetting records become prohibitively time consuming. Display and redraw rates become low, reducing the utility of these data, and decreasing the likelihood that GIS will be effectively applied.

To avoid these problems, a many-to-one relationship is usually allowed between the raster cells and the attribute table (Figure 2-36b). Many raster cells may refer to a single row in the attribute column. This substantially reduces the size of the attribute table for most data sets although it does so at the cost of some spatial ambiguity. There may be multiple, noncontiguous patches for a specific type. For example, the upper left and lower right portion of the raster data set in Figure 2-36b are both of class 10. Both are recognized as distinct features in the vec-

a) Raster, one-to-one

A	A	A	A	B	B	B	B	B	B
A	A	A	A	B	B	B	B	B	B
A	A	A	A	B	B	B	B	B	B
A	A	A	B	B	B	B	B	B	B
A	A	A	C	C	B	B	B	B	B
C	C	C	C	C	D	D	D	D	D
C	C	C	C	C	D	D	D	D	D
C	C	C	C	C	D	D	D	D	D
C	C	C	C	C	D	D	D	E	E
C	C	C	C	C	D	D	E	E	E

attribute table
(cell 1 is upper-left corner)

cell-ID	IDorg	class	area
1	A	10	0.8
2	A	10	0.8
3	A	10	0.8
4	A	10	0.8
5	B	11	0.8
6	B	11	0.8
7	B	11	0.8
. . .	. . .	. . .	. . .
100	E	10	0.8

b) Raster, many-to-one

10	10	10	10	11	11	11	11	11	11
10	10	10	10	11	11	11	11	11	11
10	10	10	10	11	11	11	11	11	11
10	10	10	11	11	11	11	11	11	11
10	10	10	15	15	11	11	11	11	11
15	15	15	15	15	21	21	21	21	21
15	15	15	15	15	21	21	21	21	21
15	15	15	15	15	21	21	21	21	21
15	15	15	15	15	21	21	21	10	10
15	15	15	15	15	21	21	10	10	10

attribute table

class	area
10	18.4
11	24.0
15	21.6
21	13.6

Figure 2-36: Raster data models rarely maintain this one-to-one relationship between cells and attributes (a), because table access and performance usually suffer. A many-to-one relationship between cells and table rows is adopted more often (b).

tor and one-to-one raster representation, but are represented by the same attribute entry in the many-to-one raster representation. This reduces the size of the attribute table, but at the cost of reducing the flexibility of the attribute table. Many-to-one relationships effectively create multipart areas. The data for the represented variable may be summarized by class; however, these classes may or may not be spatially contiguous.

An alternative is to maintain the one-to-one relationship, but to index all the raster cells in a contiguous group, thereby reducing the number of rows in the attribute table. This requires software to develop and maintain the indices, and to create them and reconstitute the indexing after spatial operations. These indexing schemes add overhead and increase data model complexity, thereby removing one of the advantages of raster data sets over vector data sets.

A Comparison of Raster and Vector Data Models

The question often arises, "Which are better, raster or vector data models?" The answer is neither and both. Neither of the two classes of data models is better in all conditions or for all data. Both have advantages and disadvantages relative to each other and to additional, more complex data models. In some instances it is preferable to maintain data in a raster model, and in others in a vector model. Most data may be represented in both, and may be converted among data models. As an example, land cover may be represented as a set of polygons in a vector data model or as a set of identifiers in each cell in a raster grid. The choice often depends on a number of factors, including the predominant type of data (discrete or continuous), the expected types of analyses, available storage, the main sources of input data, and the expertise of the human operators.

Raster data models exhibit several advantages relative to vector data models. First, raster data models are particularly suit-

Table 2-2: A comparison of raster and vector data models**.**

Characteristic	Raster	Vector
data structure	usually simple	usually complex
storage requirements	larger for most data sets without compression	smaller for most data sets
coordinate conversion	may be slow due to data volumes, and require resampling	simple
analysis	easy for continuous data, simple for many layer combinations	preferred for network analyses, many other spatial operations more complex
spatial precision	floor set by cell size	limited only by positional measurements
accessibility	easy to modify or program, due to simple data structure	often complex
display and output	good for images, but discrete features may show "stairstep" edges	maplike, with continuous curves, poor for images

able for representing themes or phenomena that change frequently in space. Each raster cell may contain a value different than its neighbors. Thus trends as well as more rapid variability may be represented.

Raster data structures are generally simpler than vector data models, particularly when a fixed cell size is used. Most raster models store cells as sets of rows, with cells organized from left to right, and rows stored from top to bottom. This organization is quite easy to code in an array structure in most computer languages.

Raster data models also facilitate easy overlays, at least relative to vector models. Each raster cell in a layer occupies a given position corresponding to a given location on the Earth's surface. Data in different layers align cell-to-cell over this position. Thus, overlay involves locating the desired grid cell in each data layer and comparing the values found for the given cell location. This cell look-up is quite rapid in most raster data structures, and thus layer overlay is quite simple and rapid when using a raster data model.

Finally, raster data structures are the most practical method for storing, displaying, and manipulating digital image data, such as aerial photographs and satellite imagery. Digital image data are an important source of information when building, viewing, and analyzing spatial databases. Image display and analysis are based on raster operations to sharpen details on the image, specify the brightness, contrast, and colors for display, and to aid in the extraction of information.

Vector data models provide some advantages relative to raster data models. First, vector models often lead to more compact data storage, particularly for discrete objects. Large homogenous regions are recorded by the coordinate boundaries in a vector data model. These regions are recorded as a set of cells in a raster data model. The perimeter grows more slowly than the area for most feature shapes, so the amount of data required to represent an area increases much more rapidly with a raster data model. Vector data are much more compact than raster data for most themes and levels of spatial detail.

Vector data are a more natural means for representing networks and other connected linear features. Vector data by their nature store information on intersections (nodes) and the linkages between them (lines). Traffic volume, speed, timing, and other factors may be associated with lines and intersections to model many kinds of networks.

Vector data models are easily presented in a preferred map format. Humans are familiar with continuous line and rounded curve representations in hand- or machine-drawn maps, and vector-based maps show these curves. Raster data often show a "stair-step" edge for curved boundaries, particularly when the cell resolution is large relative to the resolution at which the raster is displayed. Vector data may be plotted with more visually appealing continuous lines and rounded edges.

Vector data models facilitate the calculation and storage of topological information. Topological information aids in performing adjacency, connectivity, and other analyses in an efficient manner. Topological information also allows some forms of automated error and ambiguity detection, leading to improved data quality.

Conversion Between Raster and Vector Models

Spatial data may be converted between raster and vector data models. Vector-to-raster conversion involves assigning a cell value for each position occupied by vector features. Vector point features are typically assumed to have no dimension. Points in a raster data set must be represented by a value in a raster cell, so points have at least the dimension of the raster cell after conversion from vector-to-raster models. Points are usually assigned to the cell containing the point coordinate. The cell in which the point resides is given a number or other code iden-

tifying the point feature occurring at the cell location. If the cell size is too large, two or more vector points may fall in the same cell, and either an ambiguous cell identifier assigned, or a more complex numbering and assignment scheme implemented. Typically a cell size is chosen such that the diagonal cell dimension is smaller than the distance between the two closest point features.

Vector line features in a data layer may also be converted to a raster data model. Raster cells may be coded using different criteria. One simple method assigns a value to a cell if a vector line intersects with any part of the cell (Figure 2-37a, left). This ensures the maintenance of connected lines in the raster form of the data. This assignment rule often leads to wider than appropriate lines because several adjacent cells may be assigned as part of the line, particularly when the line meanders near cell edges. Other assignment rules may be applied, for example, assigning a cell as occupied by a line only when the cell center is near a vector line segment (Figure 2-37a, right). "Near" may be defined as some sub-cell distance, for instance, 1/3 the cell width. Lines passing through the corner of a cell will not be recorded as in the cell. This may lead to thinner linear features in the raster data set, but often at the cost of line discontinuities.

The output from vector-to-raster conversion depends on the algorithm used, even though you use the same input. This brings up an important point to remember when applying any spatial operation. The output often depends in subtle ways on the spatial operation. What appear to be quite small dif-

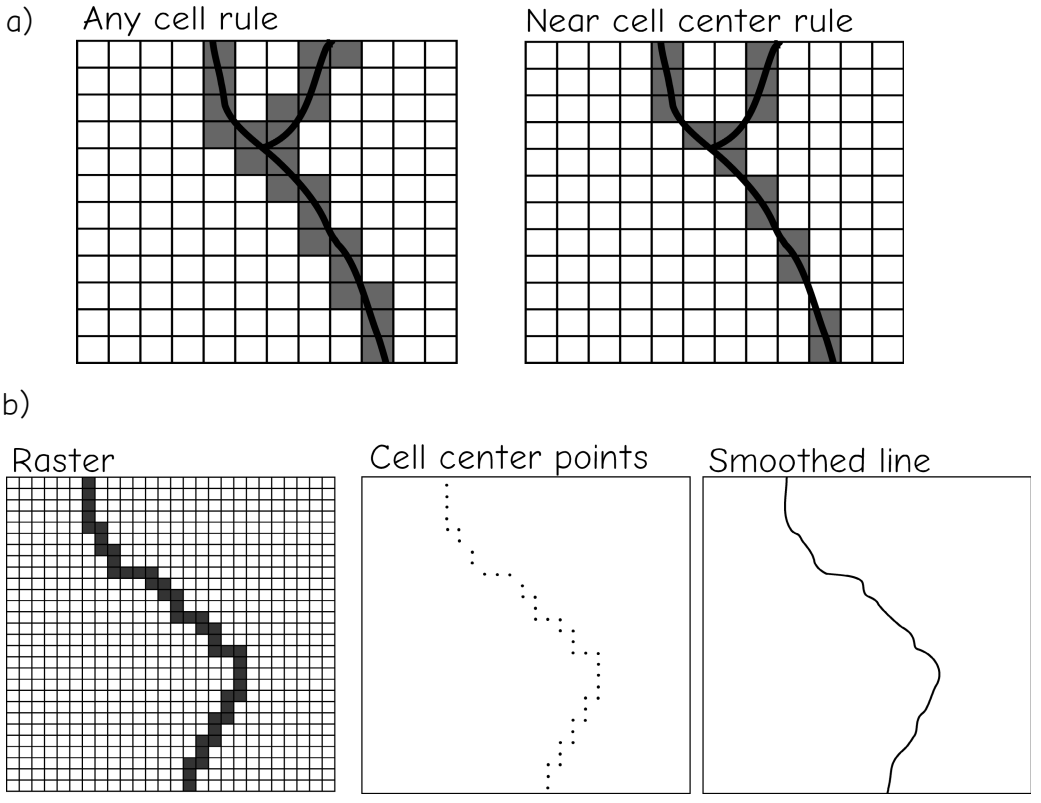


Figure 2-37: Vector-to-raster conversion (a) and raster-to-vector conversion (b). In **a**, cells are assigned in a raster if they intersect with a converted vector. The left and right panels show how two assignment rules result in different raster coding near lines. Panels in **b** show how raster data may be converted to vector formats, and may involve line smoothing or other operations to remove the "stair-step" effect.

ferences in the algorithm or key defining parameters may lead to quite different results. The ease of spatial manipulation in a GIS provides a powerful and often easy-to-use set of tools. The GIS user should bear in mind that these tools can be more efficient at producing errors as well as more efficient at providing correct results. Until sufficient experience is obtained with a suite of algorithms, in this case vector-to-raster conversion, small, controlled tests should be performed to verify the accuracy of a given method or set of constraining parameters.

Up to this point we have covered vector-to-raster data conversion. Data may also be converted in the opposite direction, from raster to vector data. Point, line, or area features represented by raster cells are converted to corresponding vector data coordinates and structures. Point features are represented as single raster cells. Each vector point feature is usually assigned the coordinate of the corresponding cell center.

Linear features represented in a raster environment may be converted to vector lines. Conversion to vector lines typically involves identifying the continuous connected set of grid cells that form the line. Cell centers are typically taken as the locations of vertices along the line (Figure 2-37b). Lines may then be "smoothed" using a mathematical algorithm to remove the "stair-step" effect.

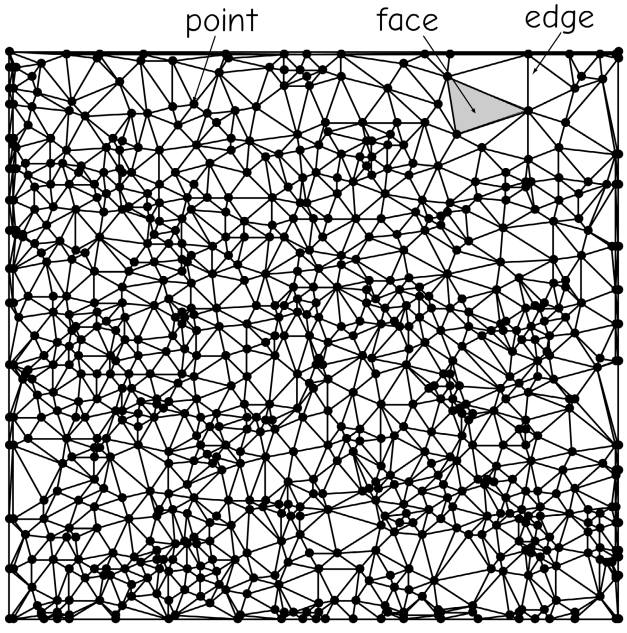
Other Data Models

Triangulated Irregular Networks

A *triangulated irregular network* (TIN) is a data model commonly used to represent terrain heights. Typically the x, y, and z locations for measured points are entered into the TIN data model. These points are distributed in space, and the points may be connected such that the smallest triangle possible spans any three adjacent points. The TIN forms a connected network of triangles (Figure 2-38). *Delaunay triangles* are created such that the lines from one triangle do not cross the lines of another. Line crossings are avoided by identifying the *convergent circle* for a set of three points (Figure 2-38).The convergent circle is defined as the circle passing through all three points. A triangle is drawn only if the corresponding convergent circle contains no other sampling points. Each triangle defines a terrain surface, or facet, assumed to be of uniform slope and aspect over the triangle.

The TIN model typically uses some form of indexing to connect neighboring points. Each edge of a triangle connects to two points, which in turn each connect to other edges. These connections continue recursively until the entire network is spanned. Thus, the TIN is a rather more complicated data model than the simple raster grid.

While the TIN model may be more complex than simple raster models, it may also be much more appropriate and efficient when storing terrain data in areas with variable relief. Relatively few points are required to represent large, flat, or smoothly continuous areas. Many more points are desirable when representing variable, discontinuous terrain. Surveyors often collect more samples per unit area where the terrain is highly variable. A TIN easily accommodates these differences in sampling density, resulting in more, smaller triangles in the densely sampled area. Rather than imposing a uniform cell size and having multiple measurements for some cells, one measurement for others, and no measurements for most cells, the TIN preserves each measurement point at each location.



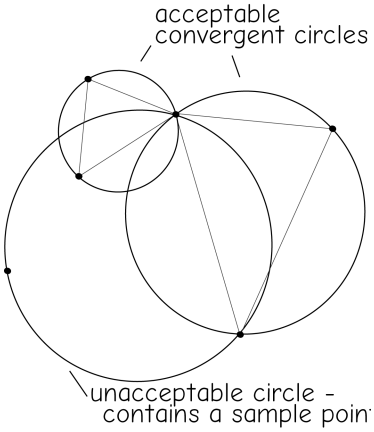


Figure 2-38: A TIN data model defines a set of adjacent triangles over a sample space (left). Sample points, facets, and edges are components of TIN data models. Triangles are placed by convergent circles. These intersect the vertices of a triangle and contain no other possible vertices (below).

Object Data Models

The *object data model* is an alternative for structuring spatial data. The object data model incorporates much of the philosophy of object-oriented programming into a spatial data model. A main goal is to raise the level of abstraction so that the data objects may be conceptualized and addressed in a more natural way. Object models attempt to encapsulate the information and operations (often called "methods") into discrete objects. These objects could be geographic features, such as a city. Spatial and attribute data associated with a given city would be incorporated in a single city object. This object may include not only information on the city boundary, but also streets, building locations, waterways, or other features that might be in separate data structures in a layered topological vector model. The topology could be included, but would likely be incorporated within the single object. Topological relationships to exterior objects such as adjacent cities or counties may also be represented.

Object model approaches have been adopted by at least one major vendor of GIS software and are applied in a number of fields. We typically conceptualize a small set of features, or types of real-world objects, that we wish to represent in our GIS database. These features themselves might be composed of other features, and these features by yet other features. Buildings might contain floors, and floors contain rooms. We easily conceptualize a building as a nesting of these sometimes complex objects or features.

As another example, we might want to represent an electrical power distribution system using a GIS. This would include a number of different types of features, including power plants, transportation networks to bring fuel to the plants, plant buildings, power lines, and customer buildings. We might represent these as vector points, lines, and polygons. However, these might be difficult to conceptualize and cumbersome to manage if we separate the features into a set of vector layers. Complex objects, for example, a substation, might be composed of transformers, regulators, interconnected networks, and powerlines. We might have difficulty representing this with simple vector layers. We might need multiple point layers or types to represent transformers vs. regulators, and one or more vector layers to represent the interconnections and the power lines. We might also need to enforce rules about specific kinds of connections or relationships; for example, there is always a regulator between a power source and a transformer.

Object models for spatial data often follow a *logical model*, a user's view of the real objects we portray with a GIS (Figure 2-39). This model includes all the "things" of interest, and the relationships among them. Things, or objects, might include power poles, transformers, powerlines, meters, and customer buildings, and relationships among them would include a transformer on a pole, lines between poles, and meters at points

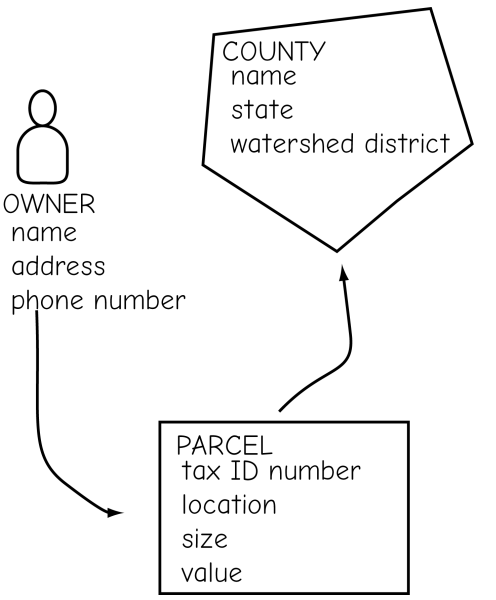


Figure 2-39: Objects in a GIS database may be conceptualized in a diagram, or logical model of how they are related. Here, three types of objects are represented, with owners associated with parcels, and parcels associated with counties.

along the lines. The logical model is often represented as a box-and-line diagram.

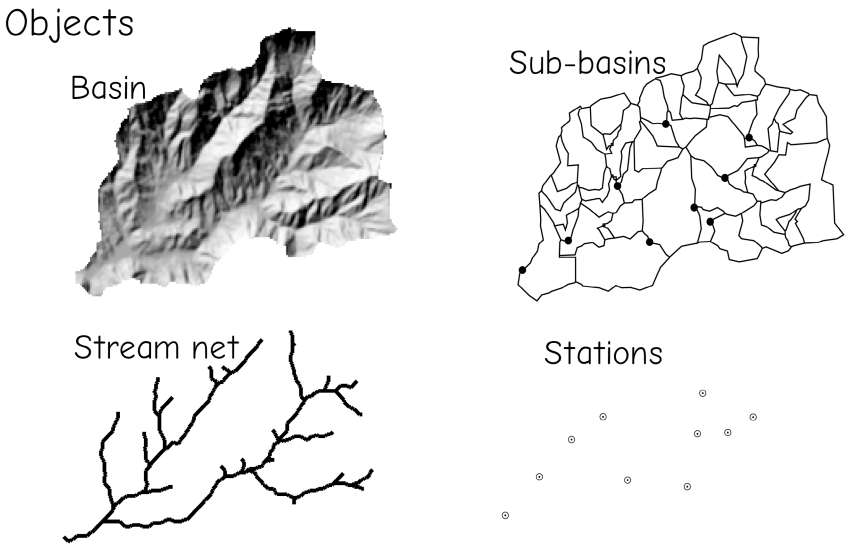
Most object models define the properties of each object, and the relationships among the same and different types of objects. Pipes may have a diameter, material type, they may be connected to valves and tanks, and run under buildings. The pipes may be represented by lines and the valves by points, but these vector elements are enhanced in the object model because the specific pipe and pipe properties may be linked to the specific valve attached to a given location. The properties of the pipe are encapsulated within the pipe object. Valves may be required to be coincident with pipes, to match pipe properties, and pipelines required to begin and end with a valve.

Because relationships are stored in the database, we can transparently represent the topology and change the topological rules as needed. Relationships are an explicit part of the object-oriented design, so rules on object properties, such as "pipe materials may be only clay, steel, PVC, or copper," or "service lines must connect to main lines," may be explicitly embedded in the database. In other data models these relationships are often embedded in specific, hard-to-modify computer code, a disadvantage that adds cost and time to modifying a database.

Object models can be specified to automatically transfer properties within classes of objects, a capability called *inheritance*. We may create a generic valve object, with a maximum pressure rating, cost, and material type. Within this general class we may create a number of valve subclasses, for example, emergency cut-off valves, primary control valves, or shunt valves. These subclasses will inherit all the property variables from a generic valve in that each has a cost, maximum pressure, and material, and is required to exist on a pipe, but each subclass may also have additional, unique properties.

Figure 2-40 shows an example of an object data model for hydrologic basins and related stream networks and features. The top frame shows graphics of the features, in this example basins, sub-basins, a stream network, and features on the stream network such as measurement or sampling stations. The bottom panel shows the feature types, attributes, and properties in the object model. Note that there are both object properties and topological relationships represented, and that multiple feature types may be represented in the object model. Main basins, subbasins, and stations are all types of water features. Basins may be made up of multiple kinds of area features. Subbasin objects may be composed of point, line, and area features, all constrained topologically to be within basins or coincident with stream segments, as appropriate. Stream networks are defined as segments and junctions, and stations of several types may be present, all constrained to be located on stream segments.

The object data model has both advantages and disadvantages when compared to traditional topological vector and raster data models. Some geographic entities may be naturally and easily identified as discrete units for particular problems, and so may be naturally amenable to an object oriented approach. Some proponents claim object models are more easily implemented across a wider range of database software, particularly for complex models. However, object data models are less useful for representing continuously varying features, such as elevation. In addition, for many problems object definition and indexing may be quite complex. Software developers have had difficulty developing generic tools that may quickly implement object models, so there is an added level of specialized training required. Finally, we note that there is no widely accepted, formal definition of what constitutes an object data model.



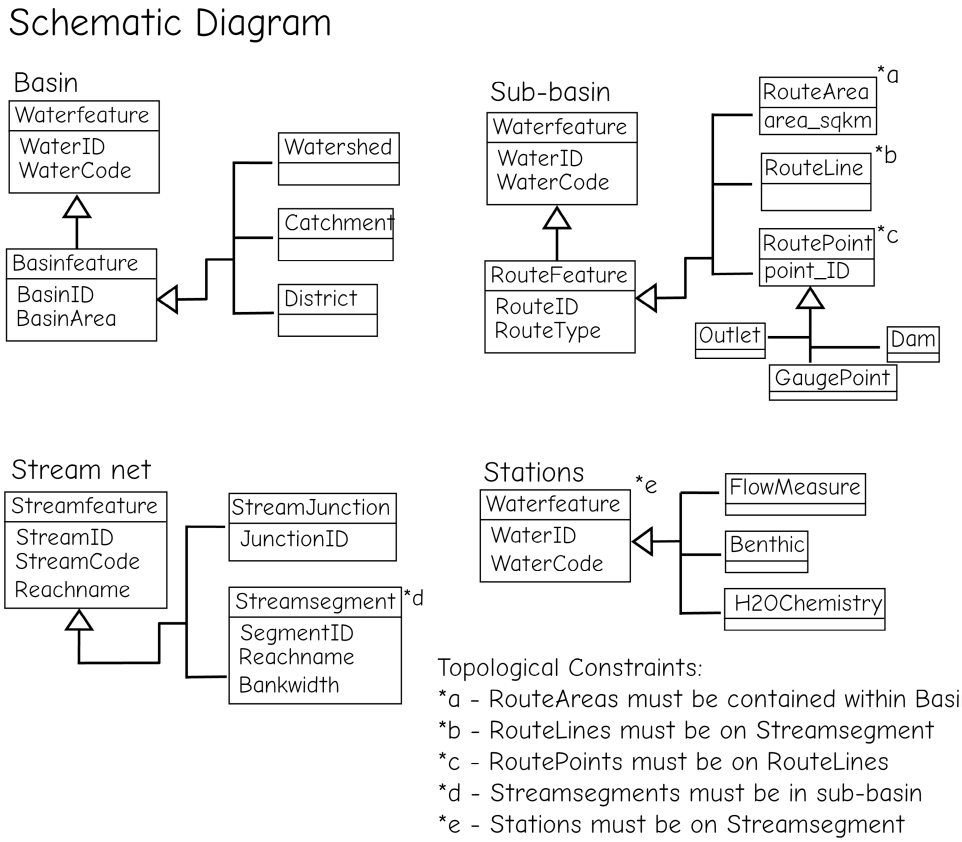


Figure 2-40: Object-oriented data models allow us to encapsulate complex objects that may be a combination of many different features and feature types, while explicitly identifying the embedded complexity in a standard way. Constraints such as topological relationships across objects may also be represented.

Three-Dimensional Data Models

GIS in built environments are increasingly integrating three-dimensional (3D) information such as building heights, roof shapes, and other height-related characteristics (Figure 2-41). This is in part to support analysis, and in part to generate visualizations from at or near ground-level, for example, building appearance from a nearby road. Improved data capture allows for the rapid development of 3D data that must be integrated into an appropriate data model.

Several 3D data models have been proposed, with "vector-like" models more commonly applied than "raster-like" models. The latter employ the concept of voxels, or volume elements, essentially cubes of a fixed dimension. Flat or baseline areas have zero voxels stacked over the surface, with a requisite number of voxels "staked" at each raster cell to represent height. While easy to access and simple for all the reasons a raster set is, they also suffer from the same drawbacks, particularly the trade-off between precision, or voxel size, and data volumes.

Many alternative "vector-like" 3D models have been proposed, many defining a body element, in addition to the standard point, line, and polygon elements. Points are used to create lines, lines for polygons, and polygons to create bodies. Much like 2D vectors, 3D vectors add indexing schemes, with the added complication of a z coordinate to any element above the base plane of the data layer.

While vector 3D models are more common, no one model form or standard has been widely adopted, with representations generally software-specific. Three dimensional GIS products have become quite mature, for example, the 3D spatial analyst and 3D CityEngine products from ESRI are full-featured, stable, productive tools, supporting start to finish workflows, from 2D to 3D data conversion, 3D spatial data ingestion, processing and organization, and output to project, video, and interactive Web services. Three-dimensional GIS processing tools are available from other vendors, and as a plug-in for QGIS.

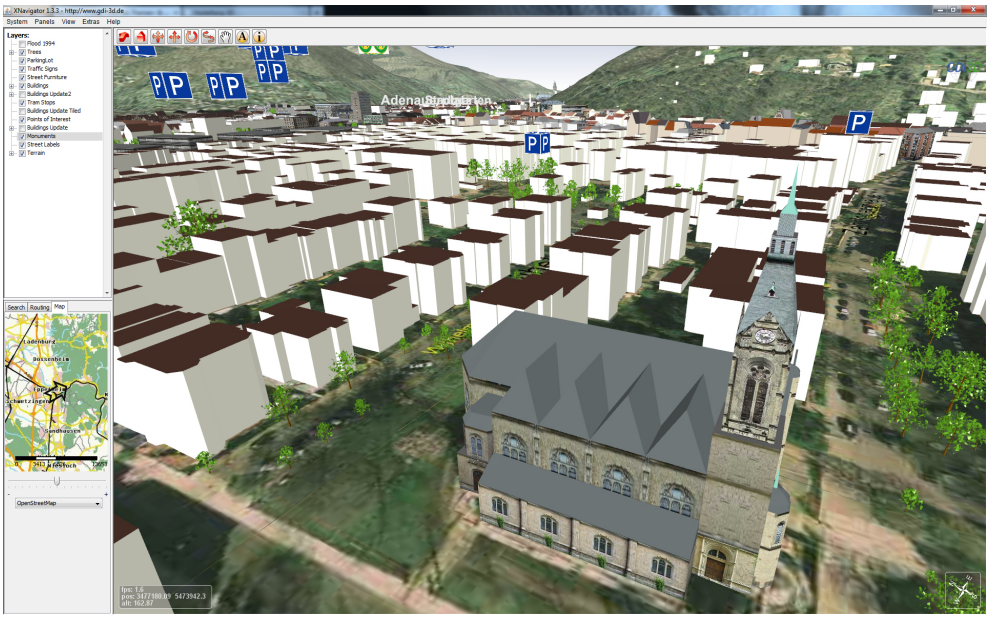


Figure 2-41: An example of 3D spatial data displayed for a region in Germany. The third dimension aids in many planning and assessment functions, and is becoming common in managing the built environment. (courtesy City of Heidelberg and the University of Heidelberg).

Multiple Models

Digital data may often be represented using any one of several data models. The analyst must choose which representation to use. Digital elevation data are perhaps the best example of the use of multiple data models to represent the same theme (Figure 2-42). Digital representations of terrain height have a long history and widespread use in GIS. Elevation data and derived surfaces such as slope and aspect are important in hydrology, transportation, ecology, urban and regional planning, utility routing, and a number of other activities that are analyzed using GIS. Because of this widespread importance, digital elevation data are commonly represented in a number of data models.

Raster grids, TINs, and vector contours are the most common data structures used to organize and store digital elevation data. Raster and TIN data are often called *digital elevation models* (DEMs) or *digital terrain models* (DTMs) and are commonly used in terrain analysis. Contour lines are most often used as a form of input, or as a familiar form of output. Historically, hypsography (terrain heights) was depicted on maps as contour lines (Figure 2-42). Contours represent lines of equal elevation, typically spaced at fixed elevation intervals across the mapped areas. Because many important analyses are more difficult using contour lines, most digital elevation data are stored using raster or TIN models.

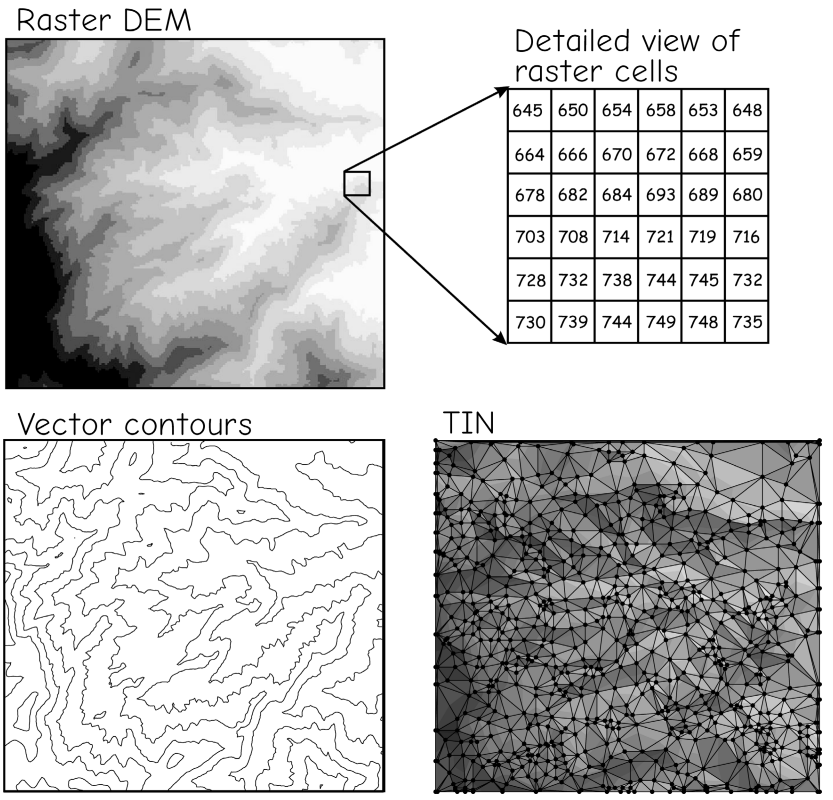


Figure 2-42: Data may often be represented in several data models. Digital elevation data are commonly represented in raster (DEM), vector (contours), and TIN data models.

Data and File Structures

Binary and ASCII Numbers

No matter which spatial data model is used, the concepts must be translated into a set of numbers stored on a computer. All information stored on a computer in a digital format may be represented as a series of 0's and 1's. These data are said to be stored in a *binary* format, because each digit may contain one of two values, 0 or 1. Binary numbers are in a base of 2, so each successive column of a number represents a power of two.

We use a similar column convention in our familiar ten-based (decimal) numbering system. As an example, consider the number 47, which we represent using two columns. The seven in the first column indicates there are seven units of one. The four in the tens column indicates there are four units of ten. Each higher column represents a higher power of ten. The first column represents one (10^0=1), the next column represents tens (10^1=10), the next column hundreds (10^2=100), and upward for successive powers of ten. We add up the values represented in the columns to decipher the number.

Binary numbers are also formed by representing values in columns. In a binary system each column represents a successively higher power of two (Figure 2-43). The first (rightmost) column represents 1 (2^0 = 1), the second column (from right) represents twos (2^1 = 2), the third (from right) represents fours (2^2 = 4), then eight (2^3 = 8), sixteen (2^4 = 16), and upward for successive powers of two. Thus, the binary number 1001 represents the decimal number 9: a one from the rightmost column, and eight from the fourth column (Figure 2-43).

Equivalent numbers

binary	decimal
00000001	1
00000010	2
00000011	3
00000100	4
00000101	5
00000110	6
00000111	7
00001000	8
00001001	9
00001010	10
....	....

Figure 2-43: Binary representation of decimal numbers.

Each digit or column in a binary number is called a *bit*, and eight columns, or bits, are called a *byte*. A byte is a common unit for defining data types and numbers, for example, a data file may be referred to as containing 4-byte integer numbers. This means each number is represented by 4 bytes of binary data (or 8 x 4 = 32 bits).

Several bytes are required when representing larger numbers. For example, one byte may be used to represent 256 different values. When a byte is used for nonnegative integer numbers, then only values from 0 to 255 may be recorded. This will work when all values are below 255, but consider an elevation data layer with values greater than 255. If the data are not rescaled, then more than one byte of storage is required for each value. Two bytes will store up to 65,536 different numbers. Terrestrial elevations measured in feet or meters are all below this value, so two bytes of data are often used to store elevation data. Real numbers such as 12.19 or 865.3 typically require more bytes, and are effectively split, that is, two bytes for the whole part of the real number, and four bytes for the fractional portion.

Binary numbers are often used to represent codes. Spatial and attribute data may then be represented as text or as standard codes. This is particularly common when raster or vector data are converted for export or import among different GIS software systems. For example, ArcGIS, a widely used GIS, produces several export formats that are in text or binary formats. Idrisi, another popular GIS, supports binary and alphanumeric raster formats.

One of the most common number coding schemes uses ASCII designators. ASCII stands for the American Standard Code for Information Interchange. ASCII is a standardized, widespread data format that uses seven bits, or the numbers 0 through 126, to represent text and other characters. An extended ASCII, or ANSI (American National Standards Institute) scheme, uses these same codes, plus an extra binary bit to represent numbers between 127 and 255. These codes are then used in many programs, including GIS, particularly for data export or exchange.

ASCII codes allow us to easily and uniformly represent alphanumeric characters such as letters, punctuation, other characters, and numbers. ASCII converts binary numbers to alphanumeric characters through an index. Each alphanumeric character corresponds to a specific number between 0 and 255, which allows any sequence of characters to be represented by a number. One byte is required to represent each character in extended ASCII coding, so ASCII data sets are typically much larger than binary data sets. Geographic data in a GIS may use a combination of binary and ASCII data stored in files. Binary data are typically used for coordinate information, and ASCII or other codes may be used for attribute data.

Pointers and Indexes

Data files may be linked by file *pointers*, *indexes*, or other structures. A pointer is an address or index that connects one file location to another. Pointers are a common way to organize information within and across multiple files. Figure 2-44 depicts an example of the use of pointers to organize spatial data. In Figure 2-44, the polygon is composed of a set of lines. Pointers are used to link the set of lines that form each polygon. There is a pointer from each line to the next line, forming a chain that defines the polygon boundary.

Pointers help by organizing data in such a way as to improve access speed. Unorganized data would require time-consuming searches each time a polygon boundary was to be identified. Pointers also allow efficient use of storage space. In our example, each line segment is stored only once. Several polygons may point to the line segment as it is typically much more space efficient to add pointers than to duplicate the line segment.

Shapefiles are a common vector spatial data format that uses an index to link files. Shapefiles were originally developed by ESRI, inc., as a way to store point, line, and polygon features, although they have since

been adopted as a common format for data interchange and analysis. Shapefiles are supported by Autocad, QGIS, MapWindow, Manifold, and most other GIS softwares that process vector data.

Shapefiles represent layers with a cluster of files. Each file has the same base name but a different filename extension, indicated by a suffix, for example, the ".shp" in the filename "boundary.shp." A transportation data layer stored in shapefile format might have the base name of roads, with different suffixes for different files:

roads.shp

roads.shx

roads.dbf

roads.prj

etc....

The first three files above are all required to represent a vector data layer using shapefiles. These files are connected using indices, numbers that identify connections and groupings for various components. The .shp files contain the coordinates that represent each road, organized by line segments. There is general information for each segment, and then a list of coordinates and other data for the segment. This is followed by general information for the next segment, and another list. Since road lengths vary, so will each record (string of numbers) for each road. Note that adjacent road segments are often near each other in the file, but don't have to be. When multiple segments connect at a junction, for example, at a crossroad, not all connections can be sequentially ordered in the list.

Segments in the roads.shp file are indexed by pointers in the roads.shx file. Part of the information stored for a segment is the identifiers of connecting segments. The roads.shx file contains indices that point to the segment records in the .shp files, based on these identifiers. This speeds access, because without indexing, the software would have to search the .shp file each time it needed to find adjacent segments in a road.

The roads.dbf file also uses an index to point to the combined roads in the .shp and .shx files. A group of segments may be used to form a line, and associated with a set of attributes stored in a dbf file, for example, attributes on road name, surface type, or speed limit. By appropriate use of pointers and indices, largely hidden to the user, this group of three shapefiles implements our vector data model.

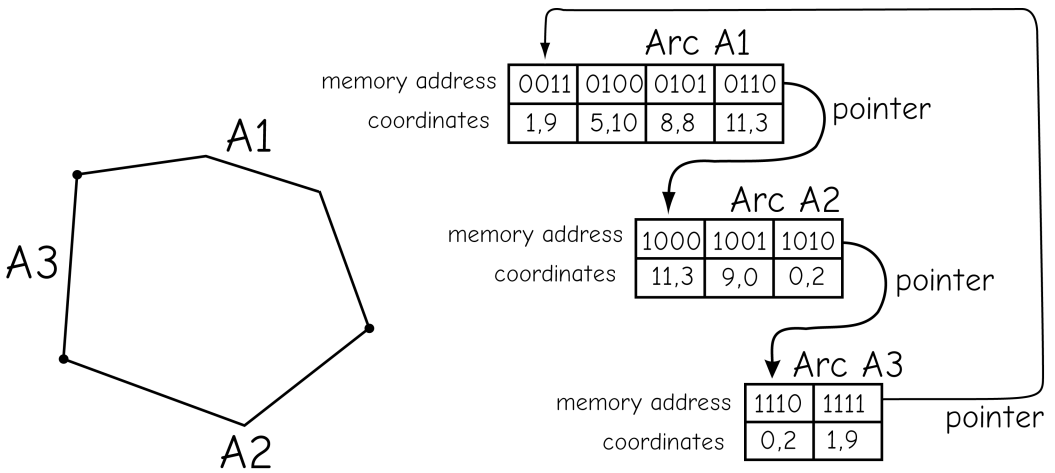


Figure 2-44: Pointers are used to organize vector data. Pointers reduce redundant storage and increase speed of access.

Because pointers and indices are key elements in organizing the spatial data, altering them directly will usually cause problems. Typically these indices are created by the software during processing, and updated as needed when data are added, modified, or analyzed. Pointers may be visible, for example, the OID columns in the .dbf tables used with shapefiles, but manually changing the values will often ruin the data layer. You should know the identity and use of pointers in your data sets, so that you don't change them inadvertently.

Pointers, indexing, and multifile layers are not limited to vector data. Many raster formats store a majority of the cell data in one file, and additional, linked information in an associated file. You must be careful when transferring a data layer to include all the associated files. For example, copying the roads.shp and roads.dbf files to a new location does not copy a usable data layer. The software expects a .shx file; an incomplete file set is often useless.

Data Compression

We often compress spatial data files because they are large. Data compression reduces file size while maintaining the information contained in the file. Compression algorithms may be "lossless," in that all information is maintained during compression, or "lossy," in that some information is lost. A lossless compression algorithm will produce an exact copy of the original when it is applied and then the appropriate decompression algorithm applied. A lossy algorithm will alter the data when it is applied and the appropriate decompression algorithm applied. Lossy algorithms are most often used with image data, where substantial degradation still leaves a useful image, and are uncommonly applied to thematic spatial data, where any data degradation is typically not tolerated.

Data compression is most often applied to discrete raster data, for example, when representing polygon or area information in a raster GIS. There are redundant data elements in raster representations of large homogenous areas. Each raster cell within a homogenous area will have the same code as most or all of the adjacent cells. Data compression algorithms remove much of this redundancy.

Run length coding is a common data compression method. This compression technique is based on recording sequential runs of raster cell values. Each run is recorded as the value found in the set of adjacent cells and the run length, or number of cells with the same value. Seven sequential cells of type A might be listed as A7 instead of AAAAAAA. Thus, seven cells would be represented by two characters. Consider the data recorded in Figure 2-45, where each line of raster cells is represented by a set of run-length codes. In general, run length coding reduces data volume, as shown for the top three rows in Figure 2-45.

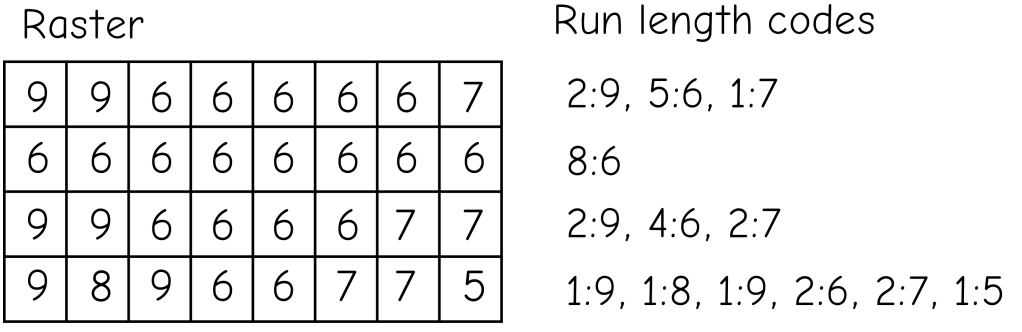


Figure 2-45: Run-length coding is a common and relatively simple method for compressing raster data. The left number in the run-length pair is the number of cells in the run, and the right is the cell value. Thus, the 2:9 listed at the start of the first line indicates a run of length two for the cell value 9.

Note that in some instances run-length coding increases the data volume, most often when there are no long runs. This occurs in the last line of Figure 2-45, where frequent changes in adjacent cell values result in many short runs. However, for most thematic data sets containing area information, run-length coding substantially reduces the size of raster data sets.

There is also some data access cost in run-length coding. Standard raster data access involves simply counting the number of cells across a row to locate a given cell. To locate a cell in run-length coding we must sum along the run-length codes to identify a cell position. This is typically a minor additional cost, but in some applications the trade-off between speed and data volume may be objectionable.

Quad tree representations are another raster compression method. Quad trees are similar to run-length codings in that they are most often used to compress raster data sets when representing area features. Quad trees may be thought of as a raster data structure with a variable spatial resolution. Raster cell sizes are combined and adjusted within the data layer to fit into each specific area feature (Figure 2-46). Large raster cells that fit entirely into one uniform area are assigned the value corresponding to that area; for example, the three largest cells in Figure 2-46 are all assigned the value a. Successively smaller cells are then fit, halving the cell dimension at each iteration, again fitting the largest cell that will fit in each uniform area. This is illustrated in the top-left corner of Figure 2-46. Successively smaller cells are defined by splitting "mixed cells" into four quadrants, and assigning the values a or b to uniform areas. This is repeated down to the smallest cell size that is needed to represent uniform areas at the required detail.

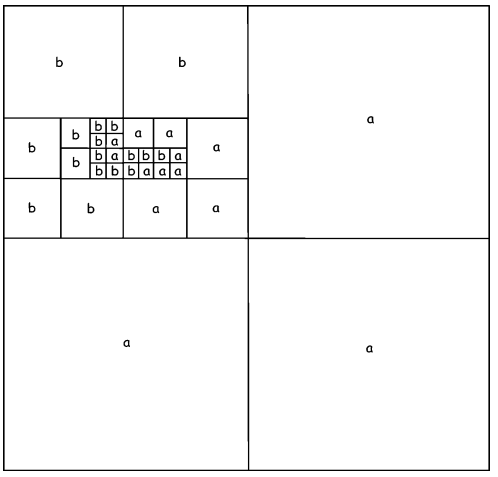


Figure 2-46: Quad tree compression.

The varying cell size in a quad tree representation requires more sophisticated indexing than simple raster data sets. Pointers are used to link data elements in a tree-like structure, hence the name quad trees. There are many ways to structure the data pointers, from large to small, or by dividing quandrants, and these methods are beyond the scope of an introductory text. Further information on the structure of quad trees may be found in the references at the end of this chapter.

There are many other data compression methods that are commonly applied. LZW is a lossless compression method commonly applied to image and raster data sets, particularly GIF images and TIFF formats. JPEG and wavelet compression algorithms are often applied to reduce the size of spatial data, particularly image or other data, although as implemented these are lossy algorithms. Generic bit- and byte-level compression methods may be applied to any files for compression or communications. There is usually some cost in time to the compression and decompression.

Raster Pyramids

We sometimes intentionally increase the size of our raster data sets without increasing the resolution in a process known as *pyramiding*. We create pyramids to increase display speeds when viewed at small scales ("zoomed out"). Long redraw times often hinder use of large data sets, particularly when panning frequently. When displayed at

very small scales, the cell size of a data set may be smaller than the resolution of the computer screen. A raster data set 1,000,000 pixels across has 1000 times the data that can be displayed on a monitor with 1000 pixel horizontal resolution. However, display software must wade through all 1,000,000 data elements in a row to pick the 1 cell in 1000 to display. While clever software can help, there are limits to how much we can speed up the redraws.

Pyramiding in effect saves subsampled copies of the cells at various resolutions. In our example above, pyramids may do the equivalent of saving every two, every four, every 10, every 30, and every 100 cells, all within the same raster data set. The software then compares the display scale to the dimensions of the data set, and chooses the most appropriate cell resolution to display. Redraws are much faster, and transparent to the user.

Note that we say pyramids "in effect" save copies of cells at various resolutions. This is the simplest method, but often not the most efficient for space or speed of access. Sophisticated indexing may be used to point to the cells at the appropriate resolutions.

Note that pyramiding comes at a cost, both in the size and complexity of the raster data set. Indexing schemes complicate the simple raster data structure, and the software must be able to navigate the indexing scheme. Already large raster data sets may be inflated from a few percent to several times, although in practice it is typically less than a doubling of size.

Common File Formats

A small number of file formats are commonly used to store and transfer spatial data. Some of these file structures arose from distribution formats adopted by governmental departments or agencies. Others are based on formats specified by software vendors, and some have been devised by standards-making bodies. Some knowledge of the types, properties, and common naming conventions of these file formats is helpful to the GIS practitioner.

Common geographic data formats may be placed into three large classes: raster, vector, and attribute. Raster formats may be further split into single-band and multi-band file types. Multiband raster data sets are most often used to store and distribute image data, while single-band raster data sets are used to store both single-band images and nonimage spatial data. Table 2-3 summarizes some of the most common spatial data formats.

Most GIS softwares provide some utility for data import and export from standard formats, as well as formats specific to the data types in the purview of the software; for example, a terrain analysis package would likely support a broad range of raster data formats.

There is an open source initiative and tool by the name of GDAL, for Geospatial Data Abstraction Library, with provides a cross-platform utility to translate among many common vector and raster file formats. Command-line tools are described and downloadable through links at www.gdal.org. This free utility is flexible, and often can be used to extend the reach of commercial packages, by first using GDAL to convert files from unsupported to supported types, and then importing these into the target software. GDAL source code is available under an Open Geospatial Foundation license, and so have been incorporated into many other open source tools.

Table 2-3: Common formats for spatial data.

Type and source	Extension	Characteristics (R=Raster, V=Vector, A=Attribute, I=Image)
Comma Separated Value	.csv	Common ASCII text format used to distribute attribute and often vector information (A, V).
DXF, AutoDesk	.dxf	Drawing exchange file, an ASCII or binary file for exchanging spatial data (V)
DWG, Autodesk	.dwg	Native binary file used by AutoDesk to store geographic data and drawings in AutoCAD (V)
Geodatabase, ESRI	.gdb, .mdb	ESRI container for many data types (R, V, A, I)
GeoJSON, open standard	.json, .geojson	Open standard for representing and displaying simple geographic features (V, A).
GeoPackage, open standard	.gpkg	Open standard for representing vector and raster data, compatible with SQlite
GeoTIFF, open standard	.TIF, .TFF	An extension for georeferencing Aldus-Adobe public domain TIFF format (R)
GPX, open standard	.gpx	A specification based on XML for basic GNSS data
Imagine, ERDAS	.img	Multiband capable image format (R)
Interchange, ESRI	.e00	ASCII text file for vector and identifying attribute data (V).
Keyhole Markup Language, Google	.KML	XML extension for displaying and annotating features and images (V, I, A)
LAS, ASPRS	.LAS	Laser point cloud data storage (V)
shapefile, ESRI	.shp, .shx, .dbf, .prj, and others	Three or more binary files that include the vector coordinate, attribute, and other information (V)
TIGER, U.S. Census	tgrxxyyy, stfzz	Set of files by U.S. census areas, xx is a state code, yyy an area code, zz numbers for various file types
MIF/MID, MapInfo	.mif, .mid	Map Interchange File, vector and raster data transport from MapInfo (V,R)
NetCDF, OGC	.cdf	Machine-independent data formats for scientific data arrays (R, A, I)
NLAPS, NASA	various in a directory	Image data from various Landsat satellites, in a specified directory structure (I, R)
SDTS, U.S. Government	none	Spatial Data Transfer Standard, specifies the spatial objects, attributes, reference system (R,V, A)

Summary

In this chapter we have described the main ways of conceptualizing spatial entities, and of representing these entities as spatial features in a computer. We commonly employ two conceptualizations, also called spatial data models: a raster data model and a vector data model. Both models use a combination of coordinates, defined in a Cartesian or spherical system, and attributes, to represent our spatial features. Features are usually segregated by thematic type in layers.

Vector data models describe the world as a set of point, line, and area features. Attributes may be associated with each feature. A vector data model splits that world into discrete features, and often supports topological relationships. Vector models are most often used to represent features that are considered discrete, and are compatible with vector maps, a common output form.

Raster data models are based on grid cells and represent the world as a "checkerboard," with uniform values within each cell. A raster data model is a natural choice for representing features that vary continuously across space, such as temperature or precipitation. Data may be converted between raster and vector data models.

We use data structures and computer codes to represent our conceptualizations in more abstract, but computer-compatible forms. These structures may be optimized to reduce storage space and increase access speed, or to enhance processing based on the nature of our spatial data.

Suggested Reading

Arctur, D., Zeiler, M. (2004). *Designing Geodatabases: Case Studies in GIS Data Modeling*. Redlands: ESRI Press.

Batcheller, J.K., Gittings, B.M., Dowers, S. (2007). The performance of vector oriented data storage in ESRI's ArcGIS. *Transactions in GIS*, 11:47–65.

Batty, M., Xie, Y. (1991). Model structures, exploratory spatial data analysis, and aggregation. *International Journal of Geographical Information Systems*, 8:291–307.

Bhalla, N. (1991). Object-oriented data models: a perspective and comparative review. *Journal of Information Science*, 17:145–160.

Boguslawski, P., Gold, C.M., Ledoux, H. (2011). Modeling and analysing 3D buildings with a primal/dual data structure. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66:188–197.

Bregt, A.K., Denneboom, J., Gesink, H.J., van Randen, Y. (1991). Determination of rasterizing error: a case study with the soil map of The Netherlands. *International Journal of Geographical Information Systems*, 5:361-367.

Carrara, A., Bitelli, G., Carla, R. (1997). Comparison of techniques for generating digital terrain models from contour lines. *International Journal of Geographical Information Systems*, 11:451–473.

Congalton, RG. (1997). Exploring and evaluating the consequences of vector-to-raster and raster-to-vector conversion. *Photogrammetric Engineering and Remote Sensing*, 63:425–434.

Downs, R.M. (1998). The geographic eye: seeing through GIS. *Transactions in GIS,* 2:111–121.

Holroyd, F., Bell, S. B. M. (1992). Raster GIS: Models of raster encoding. *Computers and Geosciences*, 18:419–426.

Joao, E.M. (1998). *Causes and Consequences of Map Generalization*. London: TaylorFrancis.

Kumler, M.P. (1994). An intensive comparison of triangulated irregular networks (TINs) and digital elevation models. *Cartographica*, 31:1–99.

Langram, G. (1992). *Time in Geographical Information Systems*. London: Taylor and Francis.

Laurini, R., Thompson, D. (1992). *Fundamentals of Spatial Information Systems*. London: Academic Press.

Lee, J. (1991). Comparison of existing methods for building triangular irregular network models of terrain from grid digital elevation models. *International Journal of Geographical Information Systems*, 5:267–285.

Masser, I. (2005). *GIS Worlds: Creating Spatial Data Infrastructures*. Redlands: ESRI Press.

Nagy, G., Wagle, S.G. (1979). Approximation of polygonal maps by cellular maps. *Communications of the Association of Computational Machinery*, 22:518–525.

Peuquet, D.J. (1984). A conceptual framework and comparison of spatial data models. *Cartographica*, 21:66–113.

Peuquet, D.J. (1981). An examination of techniques for reformatting digital cartographic data. Part II: the raster to vector process. *Cartographica*, 18:375–394.

Piwowar, J.M., LeDrew, E.F., Dudycha, D.J. (1990). Integration of spatial data in vector and raster formats in geographical information systems. *International Journal of Geographical Information Systems*, 4:429–444.

Peuker, T. K., Chrisman, N. (1975). Cartographic Data Structures. *The American Cartographer*, 2:55–69.

Rana, S. (2004). *Topological Data Structures for Surfaces: An Introduction to Geographical Information Science*. New York: Wiley.

Rigaux, P., Scholl, M., Voisard, A. (Eds.) (2002). *Spatial Databases: with Application to GIS*. New York: Elsevier

Rossiter, D.G. (1996). A theoretical framework for land evaluation. *Geoderma*, 72:165–190.

Shaffer, C.A., Samet, H., Nelson R.C. (1990). QUILT: a geographic information system based on quadtrees. *International Journal of Geographical Information Systems*, 4:103–132.

Slocum, T.A., McMaster, R.B, Kessler, F.C., Howard, H.H. (2005). *Thematic Cartography and Geographic Visualization.* 2nd ed. New York: Prentice-Hall.

Tomlinson, R.F. (1988). The impact of the transition from analogue to digital cartographic representation. *The American Cartographer,* 15:249–262.

Tuan, A.N.G. (2013). Overview of three-dimensional GIS data models. *International Journal of Future Computer and Communication*, 2:270–274.

Wedhe, M. (1992). Grid cell size in relation to errors in maps and inventories produced by computerized map processes. *Photogrammetric Engineering and Remote Sensing*, 48:1289–1298.

Wilkie, D., Sewall, J., Lin, M.C. (2011). Transforming GIS data into functional road models for large-scale traffic simulation. *IEEE Transactions on Visualization and Computer Graphics*, 18:890–901.

Wise, S. (2002). *GIS Basics*. New York: Taylor & Francis.

Worboys, M.F., Duckham, M. (2004). *GIS: A Computing Perspective*. 2nd ed. Boca Raton: CRC Press.

Zeiler, M. (1999). *Modeling Our World: The ESRI Guide to Geodatabase Design*. Redlands: ESRI Press.

Zhi-Jun L., D.E. Weller (2007). A stream network model for integrated watershed modeling. *Environmental Modeling and Assessment*, DOI:10.1007/s10666–007–9083–9.

Study Questions

2.1 - How is an entity different from a cartographic object?

2.2 - Describe the successive levels of abstraction when representing real-world spatial phenomena on a computer. Why are there multiple levels, instead of just one level in a spatial data representation?

2.3 - Define a data model and describe three primary differences between the two most commonly used data models.

2.4 - Characterize the following lists as nominal, ordinal, or interval/ratio:

a) 1.1, 5.7, -23.2, 0.4, 6.67

b) green, red, blue, yellow, sepia

c) white, light grey, dark grey, black

d) extra small, small, medium, large, extra large

e) forest, woodland, grassland, bare soil

f) 1, 2, 3, 4, 5, 6, 7.

2.5 - Characterize the following lists as nominal, ordinal, or interval/ratio:

a) Spurs, Citizens, Reds, Hornets, Baggies, Toffees, Potters

b) pinch, handful, bucket, bushel, truckload

c) 6.2, 7.8, 1.1, 0.5, 19.3

d) gram, kilogram, metric ton

e) Mexico, Canada, Argentina, Guyana, Martinique.

f) small, smaller, smallest

2.6 - Complete the following coordinate conversion table, converting the listed points from degrees-minutes-seconds (DMS) to decimal degrees (DD), or from DD to DMS. See Figure 2-11 for the conversion formula.

Point	DMS	Decimal Degrees
1	36°45'12"	36.75333
2	114°58'2"	
3	85°19'7"	
4		14.00917
5		275.00001
6		0.99528
7	183°19'22"	

2.7 - Complete the following coordinate conversion table, converting the listed points from degrees-minutes-seconds (DMS) to decimal degrees (DD), or from DD to DMS. See Figure 2-11 for the conversion formula.

Point	DMS	Decimal Degrees
1	97°45'10"	97.75278
2	122°10'2"	
3	15°0'12"	
4		322.19861
5		152.65583
6		5.75
7	23°12'50"	

2.8 - What is topology, and why is it important? What is planar topology, and when might nonplanar be more useful than planar topology?

2.9 - What are the respective advantages and disadvantages of vector data models vs. raster data models?

2.10 - Under what conditions are mixed cells a problem in raster data models? In what ways may the problem of mixed cells be addressed?

2.11 - Indicate which of the following are allowable geographic coordinates:

a) N45 45’ 45” b) longitude -127.34795 c) S96 12’ 33”

d) E 66 15’ 60” e) W -12 23’ 55” f) N 56.9999

2.12 - Indicate which of the following are allowable geographic coordinates:

a) N145 45’12” b) latitude -62.34795 c) S110 52’ 43”

d) S 49 15’ 60” e) N 89 59’ 59” f) S -46.6000

2.13 - Express the following base 10 numbers in binary notation:

a) 2 b) 8 c) 9 d) 17

e) 0 f) 128 g) 22 h) 19

2.14 - Express the following base 10 numbers in binary notation:

a) 1 b) 23 c) 256 d) 4

e) 11 f) 10 g) 3 h) 20

2.15 - Express the following binary numbers in base 10 notation:

a) 0101 b) 0001 c) 1111 d) 00101101

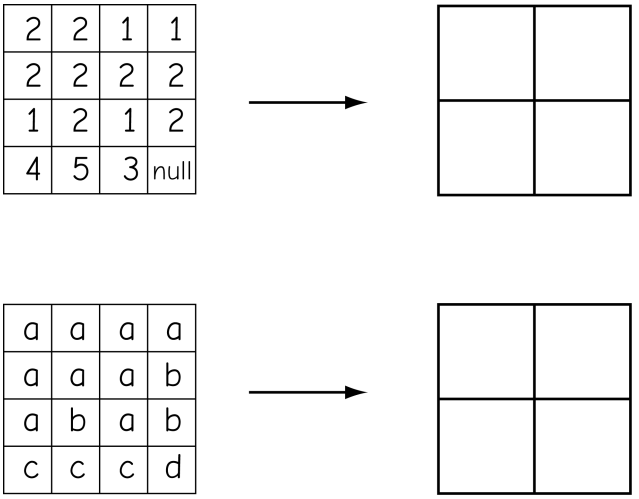
e) 1101 f) 1011 g) 10000001 h) 11111111

2.16 - Express the following binary numbers in base 10 notation:

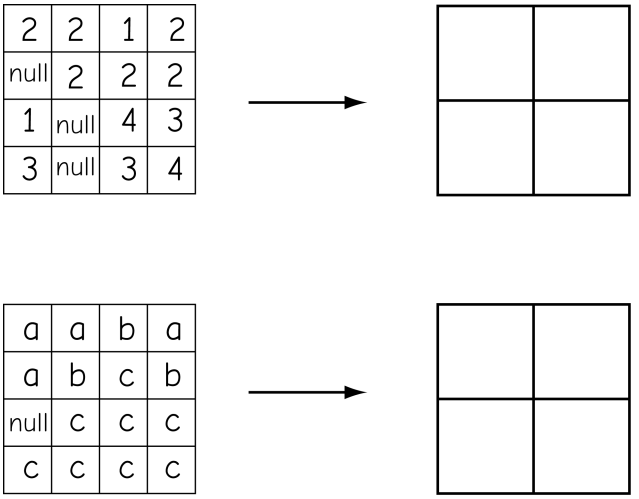
a) 1110 b) 1001 c) 0011 d) 10000101

e) 1000 f) 1010 g) 10010001 h) 11110000

2.17 - The following figure shows change in raster resolution, combining four small cells on the left to create and output for each corresponding larger cell on the right. Fill in the two rasters on the right, for the interval/ratio data (top), and the nominal data (bottom). Assume null values are not ignored, and a majority rule for nominal data.



2.18 - The following figure shows change in raster resolution, combining four small cells on the left to create and output for each corresponding larger cell on the right. Fill in the two rasters on the right, for the interval/ratio data (top), and the nominal data (bottom). Assume null values are not ignored, and a majority rule for nominal data.



2.19 - What is a triangulated irregular network?

2.20 - Why do we use binary numbers in computers?

2.21 - Why do we need to compress data? Which are most commonly compressed, raster data or vector data? Why?

2.22 - What are pointers when used in the context of spatial data, and how are they helpful in organizing spatial data?

2.23 - What are the main concepts behind object data models, and how do they differ from other data models?

2.24 - Write the run length coding for each of the rows in this raster:

b	b	a	a	a	c	a	a	a
c	c	b	b	d	d	d	a	a
b	b	b	b	b	b	b	b	b
e	c	f	b	a	d	f	b	a
a	s	a	f	f	f	b	b	a

2.25 - Write the run length coding for each of the rows in this raster:

c	c	c	c	a	a	a	a	a
a	a	b	b	d	d	d	a	a
e	e	e	f	f	f	f	f	e
a	a	a	a	a	a	a	a	a
c	c	a	a	a	b	f	d	e